68HC908MR24
Advance Information

M68HC08
Microcontrollers

Rev. 4.1
MC68HC908MR24/D
August 1, 2005

freescale.com

freescale™
semiconductor

List of Sections

Table of Contents

Section 1. General Description

Section 2. Memory Map

Section 3. Random-Access Memory (RAM)

Section 4. FLASH Memory

Section 5. Configuration Register (CONFIG)

Section 6. Central Processor Unit (CPU)

Section 7. System Integration Module (SIM)

Section 8. Clock Generator Module (CGM)

Section 9. Pulse-Width Modulator for Motor Control (PWMMC)

Section 10. Monitor ROM (MON)

Section 11. Timer Interface A (TIMA)

Section 12. Timer Interface B (TIMB)

Section 13. Serial Peripheral Interface Module (SPI)

Section 14. Serial Communications Interface Module (SCI)

Section 15. Input/Output (I/O) Ports

Section 16. Computer Operating Properly (COP)

Section 17. External Interrupt (IRQ)

Section 18. Low-Voltage Inhibit (LVI)

Section 19. Analog-to-Digital Converter (ADC)

Section 20. Power-On Reset (POR)

Section 21. Electrical Specifications

Section 22. Mechanical Specifications

Section 23. Ordering Information

Glossary

List of Figures

Figure **Title** **Page**

Figure **Title** **Page**

List of Tables

List of Tables

Section 1. General Description

1.1 Contents

1.2 Introduction

The MC68HC908MR24 is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCUs). The M68HC08 Family is based on the customer-specified integrated circuit (CSIC) design strategy. All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

1.3 Features

Features of the MC68HC908MR24 include:

- High-performance M68HC08 architecture
- Fully upward-compatible object code with M6805, M146805, and M68HC05 Families
- 8-MHz internal bus frequency
- 24 Kbytes of on-chip electrically erasable in-circuit programmable read-only memory (FLASH)
- On-chip programming firmware for use with host personal computer
- FLASH data security[1]
- 768 bytes of on-chip random-access memory (RAM)
- 12-bit, 6-channel center-aligned or edge-aligned pulse-width modulator (PWMMC)
- Serial peripheral interface module (SPI)
- Serial communications interface module (SCI)
- 16-bit, 4-channel timer interface module (TIMA)
- 16-bit, 2-channel timer interface module (TIMB)
- Clock generator module (CGM)

---

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

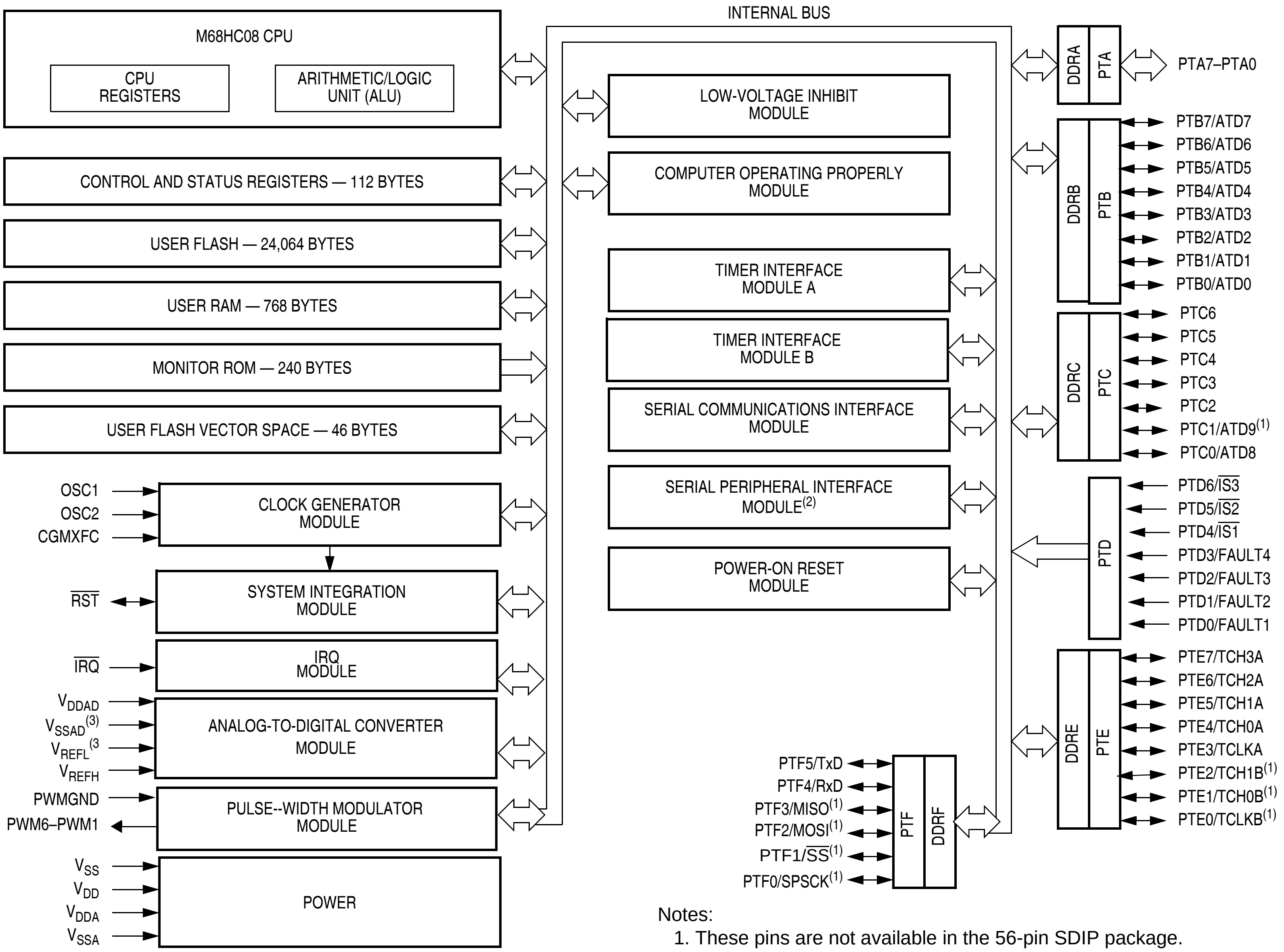
- Low-voltage inhibit (LVI) module with software selectable trip points
- 10-bit, 10-channel analog-to-digital converter (ADC)
- System protection features:
  - Optional computer operating properly (COP) reset
  - Low-voltage detection with optional reset
  - Illegal opcode detection with optional reset
  - Illegal address detection with optional reset
  - Fault detection with optional PWM disabling
- 64-pin plastic quad flat pack (QFP)
- 56-pin shrink dual in-line package (SDIP)
- Low-power design (fully static with wait mode)
- Master reset pin ($\overline{\text{RST}}$) and power-on reset (POR)

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
- C language support

1.4 MCU Block Diagram

Figure 1-1 shows the structure of the MC68HC908MR24.



Notes:

1. These pins are not available in the 56-pin SDIP package.
2. This module is not available in the 56-pin SDIP package.
3. In the 56-pin SDIP package these pins are bonded together.

Figure 1-1. MCU Block Diagram

1.5 Pin Assignments

Figure 1-2 shows the 64-pin QFP pin assignments and **Figure 1-3** shows the 56-pin SDIP pin assignments.

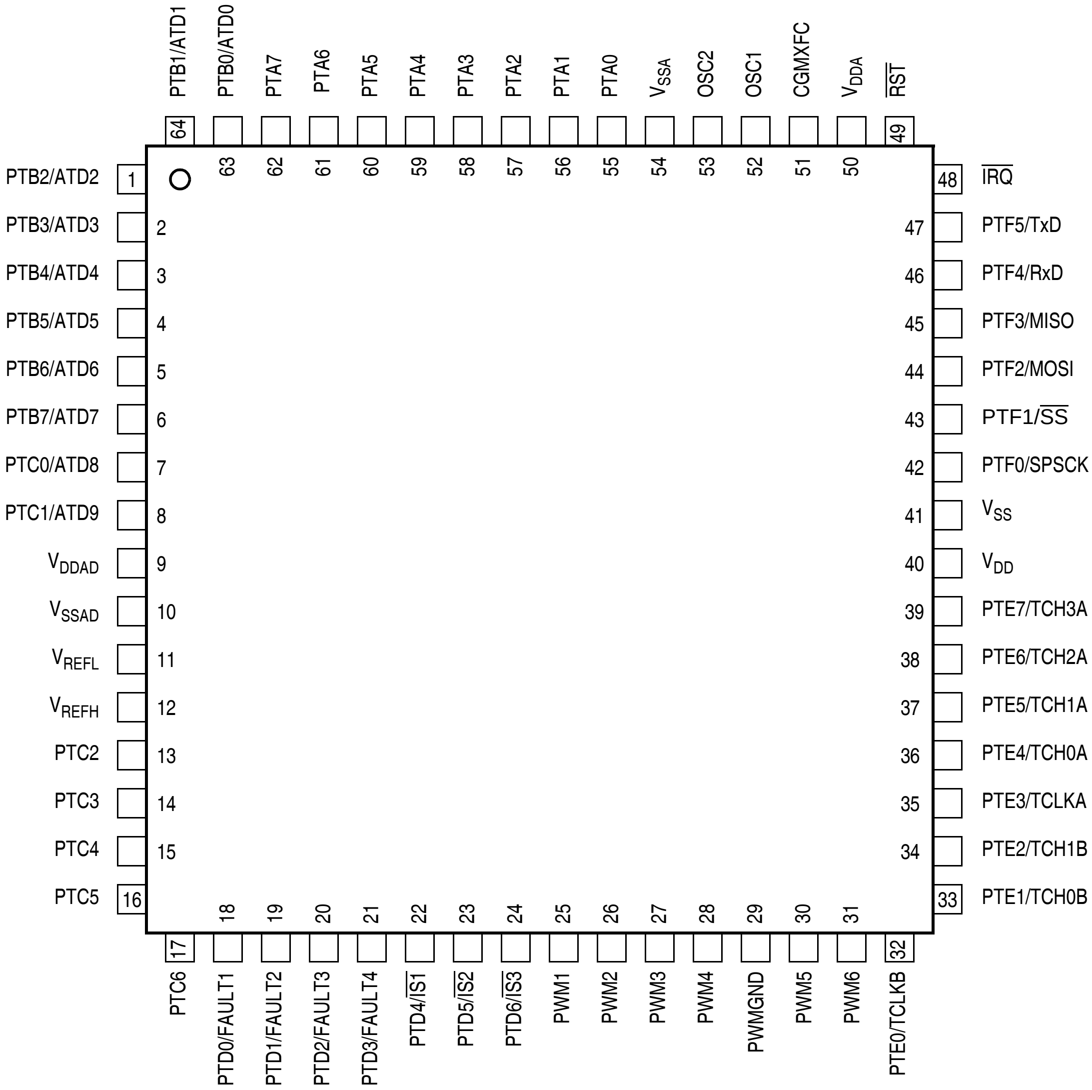
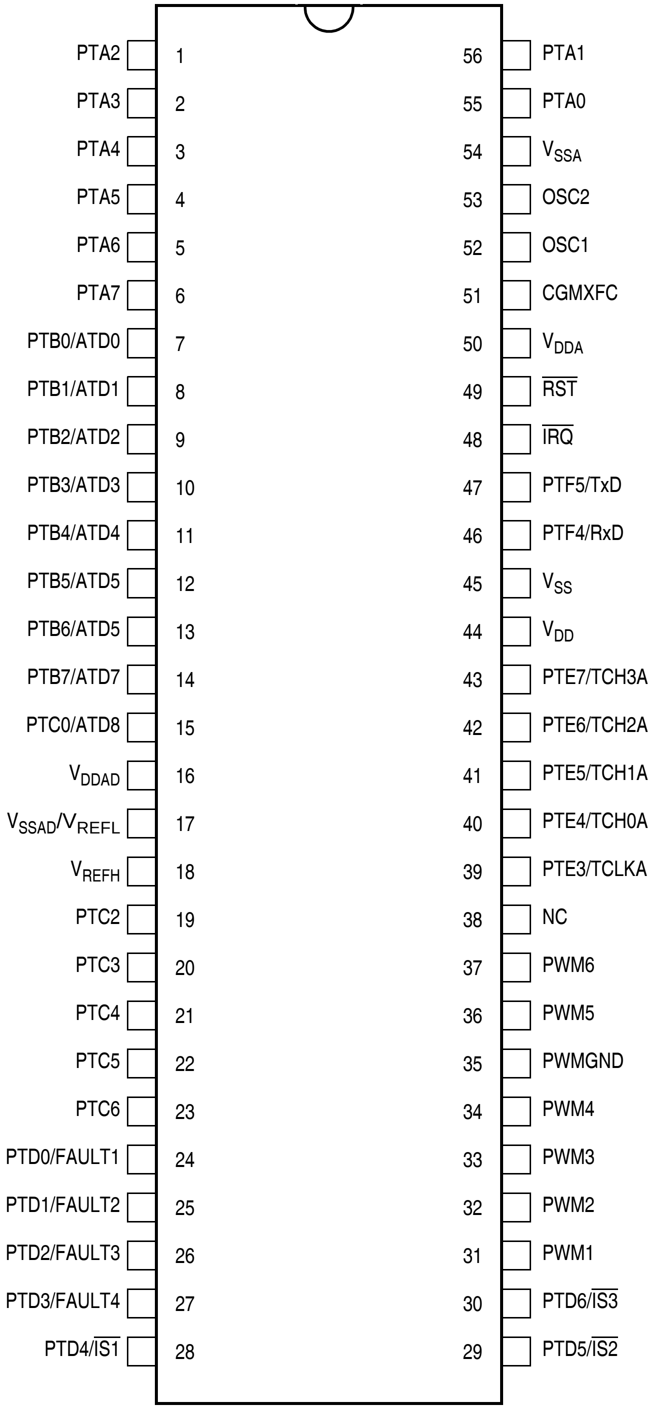


Figure 1-2. 64- Pin QFP Pin Assignments



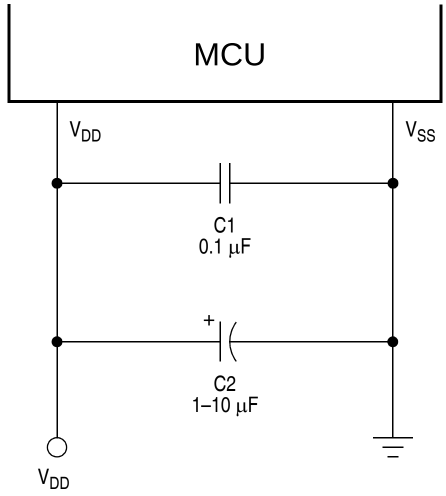
Note: PTC1, PTE0, PTE1, PTE2, PTF0, PTF1, PTF2, and PTF3 are removed from this package.

Figure 1-3. 56-Pin SDIP Pin Assignments

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

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

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



Note: Component values shown represent typical applications.

Figure 1-4. Power Supply Bypassing

1.5.2 Oscillator Pins (OSC1 and OSC2)

The OSC1 and OSC2 pins are the connections for the on-chip oscillator circuit. For more detailed information, see **Section 8. Clock Generator Module (CGM)**.

1.5.3 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. See **Section 7. System Integration Module (SIM)**.

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

$\overline{\text{IRQ}}$ is an asynchronous external interrupt pin (see **Section 17. External Interrupt (IRQ)**).

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

V_{DDA} and V_{SSA} are the power supply pins for the analog portion of the clock generator module (CGM). Decoupling of these pins should be as per the digital supply. See **Section 8. Clock Generator Module (CGM)**.

1.5.6 External Filter Capacitor Pin (CGMXFC)

CGMXFC is an external filter capacitor connection for the CGM. See **Section 8. Clock Generator Module (CGM)**.

1.5.7 Analog Power Supply Pins (V_{DDAD} and V_{SSAD})

V_{DDAD} and V_{SSAD} are the power supply pins for the analog-to-digital converter. Decoupling of these pins should be as per the digital supply. See **Section 19. Analog-to-Digital Converter (ADC)**.

1.5.8 ADC Voltage Decoupling Capacitor Pin (V_{REFH})

V_{REFH} is the power supply for setting the reference voltage V_{REFH}. Connect the V_{REFH} pin to the same voltage potential as V_{DDAD}. See **Section 19. Analog-to-Digital Converter (ADC)**.

1.5.9 ADC Voltage Reference Low Pin (V_{REFL})

V_{REFL} is the lower reference supply for the ADC. Connect the V_{REFL} pin to the same voltage potential as V_{SSAD}. See **Section 19. Analog-to-Digital Converter (ADC)**.

1.5.10 Port A Input/Output (I/O) Pins (PTA7–PTA0)

PTA7–PTA0 are general-purpose bidirectional input/output (I/O) port pins. See **Section 15. Input/Output (I/O) Ports**.

1.5.11 Port B I/O Pins (PTB7/ATD7–PTB0/ATD0)

Port B is an 8-bit special function port that shares all eight pins with the analog-to-digital converter (ADC). See **Section 19. Analog-to-Digital Converter (ADC)** and **Section 15. Input/Output (I/O) Ports**.

1.5.12 Port C I/O Pins (PTC6–PTC2 and PTC1/ATD9–PTC0/ATD8)

PTC6–PTC2 are general-purpose bidirectional I/O port pins. (See **Section 15. Input/Output (I/O) Ports**.) PTC1/ATD9–PTC0/ATD8 are special function port pins that are shared with the analog-to-digital converter (ADC). See **Section 19. Analog-to-Digital Converter (ADC)** and **Section 15. Input/Output (I/O) Ports**.

1.5.13 Port D Input-Only Pins (PTD6/$\overline{\text{IS3}}$–PTD4/$\overline{\text{IS1}}$ and PTD3/FAULT4–PTD0/FAULT1)

PTD6/$\overline{\text{IS3}}$–PTD4/$\overline{\text{IS1}}$ are special function input-only port pins that also serve as current sensing pins for the pulse-width modulator module (PWMMC). PTD3/FAULT4–PTD0/FAULT1 are special function port pins that also serve as fault pins for the PWMMC. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)** and **Section 15. Input/Output (I/O) Ports**.

1.5.14 PWM Pins (PWM6–PWM1)

PWM6–PWM1 are dedicated pins used for the outputs of the pulse-width modulator module (PWMMC). These are high-current sink pins. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)** and **Section 21. Electrical Specifications**.

1.5.15 PWM Ground Pin (PWMGND)

PWMGND is the ground pin for the pulse-width modulator module (PWMMC). This dedicated ground pin is used as the ground for the six high-current PWM pins. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)**.

1.5.16 Port E I/O Pins (PTE7/TCH3A–PTE3/TCLKA and PTE2/TCH1B–PTE0/TCLKB)

Port E is an 8-bit special function port that shares its pins with the two timer interface modules (TIMA and TIMB). See **Section 11. Timer Interface A (TIMA)**, **Section 12. Timer Interface B (TIMB)**, and **Section 15. Input/Output (I/O) Ports**.

1.5.17 Port F I/O Pins (PTF5/TxD–PTF4/RxD and PTF3/MISO–PTF0/SPSCK)

Port F is a 6-bit special function port that shares two of its pins with the serial communications interface module (SCI) and four of its pins with the serial peripheral interface module (SPI). See **Section 13. Serial Peripheral Interface Module (SPI)**, **Section 14. Serial Communications Interface Module (SCI)**, and **Section 15. Input/Output (I/O) Ports**.

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:

- 24 Kbytes of FLASH
- 768 bytes of random-access memory (RAM)
- 46 bytes of user-defined vectors
- 240 bytes of monitor read-only memory (ROM)

2.3 Unimplemented Memory Locations

Some addresses are unimplemented. Accessing an unimplemented address can cause an illegal address reset. In the memory map and in the input/output (I/O) register summary, unimplemented addresses are shaded.

Some I/O bits are read only; the write function is unimplemented. Writing to a read-only I/O bit has no effect on MCU operation. In register figures, the write function of read-only bits is shaded. Similarly, some I/O bits are write only; the read function is unimplemented. Reading of write-only I/O bits has no effect on MCU operation. In register figures, the read function of write-only bits is shaded.

2.4 Reserved Memory Locations

Some addresses are reserved. Writing to a reserved address can have unpredictable effects on MCU operation. In the memory map, **Figure 2-1**, and in the I/O register summary, **Figure 2-2**, reserved addresses are marked with the word reserved.

Some I/O bits are reserved. Writing to a reserved bit can have unpredictable effects on MCU operation. In register figures, reserved bits are marked with the letter R.

2.5 I/O Section

Addresses $0000–$005F, shown in **Figure 2-2**, contain most of the control, status, and data registers. Additional I/O registers have these addresses:

- $FE01, SIM reset status register (SRSR)
- $FE08, FLASH control register (FLCR)
- $FE0F, LVI status and control register (LVISCR)
- $FF80, FLASH block protect register (FLBPR)
- $FFFF, COP control register (COPCTL)

$0000 ↓ $005F	I/O REGISTERS — 96 BYTES
$0060 ↓ $035F	RAM — 768 BYTES
$0360 ↓ $9FFF	UNIMPLEMENTED — 40,096 BYTES
$A000 ↓ $FDFF	FLASH — 24,064 BYTES
$FE00	RESERVED
$FE01	SIM RESET STATUS REGISTER (SRSR)
$FE02	RESERVED
$FE03	RESERVED
$FE04	RESERVED
$FE05	RESERVED
$FE06	RESERVED
$FE07	RESERVED
$FE08	FLASH CONTROL REGISTER (FLCR)
$FE09	UNIMPLEMENTED
$FE0A	UNIMPLEMENTED
$FE0B	UNIMPLEMENTED
$FE0C	UNIMPLEMENTED
$FE0D	UNIMPLEMENTED
$FE0E	UNIMPLEMENTED
$FE0F	LVI STATUS AND CONTROL REGISTER (LVISCR)
$FE10 ↓ $FEFF	MONITOR ROM — 240 BYTES
$FF00 ↓ $FF7F	UNIMPLEMENTED — 112 BYTES
$FF80	FLASH BLOCK PROTECT REGISTER
$FF81 ↓ $FFD1	UNIMPLEMENTED — 80 BYTES
$FFD2 ↓ $FFFF	VECTORS — 46 BYTES

Figure 2-1. Memory Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register	Read:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
	(PTA)	Write:								
	See page 324.	Reset:	Unaffected by reset							
$0001	Port B Data Register	Read:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
	(PTB)	Write:								
	See page 326.	Reset:	Unaffected by reset							
$0002	Port C Data Register	Read:	0	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
	(PTC)	Write:	R							
	See page 328.	Reset:	Unaffected by reset							
$0003	Port D Data Register	Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
	(PTD)	Write:	R	R	R	R	R	R	R	R
	See page 330.	Reset:	Unaffected by reset							
$0004	Data Direction Register A	Read:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
	(DDRA)	Write:								
	See page 324.	Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B	Read:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
	(DDRB)	Write:								
	See page 326.	Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C	Read:	0	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
	(DDRC)	Write:	R							
	See page 328.	Reset:	0	0	0	0	0	0	0	0
$0007			Unimplemented							
$0008	Port E Data Register	Read:	PTE7	PTE6	PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
	(PTE)	Write:								
	See page 331.	Reset:	Unaffected by reset							
$0009	Port F Data Register	Read:	0	0	PTF5	PTF4	PTF3	PTF2	PTF1	PTF0
	(PTF)	Write:	R	R						
	See page 333.	Reset:	Unaffected by reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 1 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000A			Unimplemented							
$000B			Unimplemented							
$000C	Data Direction Register E (DDRE) See page 332.	Read:	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000D	Data Direction Register F (DDRF) See page 334.	Read:	0	0	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
		Write:	R	R						
		Reset:			0	0	0	0	0	0
$000E	TIMA Status/Control Register (TASC) See page 218.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	R			
		Reset:	0	0	1	0	0	0	0	0
$000F	TIMA Counter Register High (TACNTH) See page 221.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0010	TIMA Counter Register Low (TACNTL) See page 221.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0011	TIMA Counter Modulo Register High (TAMODH) See page 222.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0012	TIMA Counter Modulo Register Low (TAMODL) See page 222.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0013	TIMA Channel 0 Status/Control Register (TASC0) See page 228.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0014	TIMA Channel 0 Register High (TACH0H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 2 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0015	TIMA Channel 0 Register Low (TACH0L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0016	TIMA Channel 1 Status/Control Register (TASC1) See page 228.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0
$0017	TIMA Channel 1 Register High (TACH1H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0018	TIMA Channel 1 Register Low (TACH1L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0019	TIMA Channel 2 Status/Control Register (TASC2) See page 223.	Read:	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	TOV2	CH2MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$001A	TIMA Channel 2 Register High (TACH2H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$001B	TIMA Channel 2 Register Low (TACH2L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$001C	TIMA Channel 3 Status/Control Register (TASC3) See page 223.	Read:	CH3F	CH3IE	0	MS3A	ELS3B	ELS3A	TOV3	CH3MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0
$001D	TIMA Channel 3 Register High (TACH3H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$001E	TIMA Channel 3 Register Low (TACH3L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 3 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$001F	Configuration Register (CONFIG) See page 70.	Read:	EDGE	BOTNEG	TOPNEG	INDEP	LVIRST	LVIPWR	Bit 1	COPD
		Write:								
		Reset:	0	0	0	0	1	1	0	0
$0020	PWM Control Register 1 (PCTL1) See page 176.	Read:	DISX	DISY	PWMINT	PWMF	ISENS1	ISENS0	LDOK	PWMEN
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0021	PWM Control Register 2 (PCTL2) See page 179.	Read:	**LDFQ1**	**LDFQ0**	0	**IPOL1**	**IPOL2**	**IPOL3**	**PRSC1**	**PRSC0**
		Write:			▨					
		Reset:	0	0	0	0	0	0	0	0
$0022	Fault Control Register (FCR) See page 182.	Read:	FINT4	FMODE4	FINT3	FMODE3	FINT2	FMODE2	FINT1	FMODE1
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	Fault Status Register (FSR) See page 184.	Read:	FPIN4	FFLAG4	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	U	0	U	0	U	0	U	0
$0024	Fault Acknowledge Register (FTACK) See page 186.	Read:	0	0	DT6	DT5	DT4	DT3	DT2	DT1
		Write:	▨	FTACK4	▨	FTACK3	▨	FTACK2	▨	FTACK1
		Reset:	0	0	0	0	0	0	0	0
$0025	PWM Output Control Register (PWMOUT) See page 187.	Read:	0	OUTCTL	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1
		Write:	▨							
		Reset:	0	0	0	0	0	0	0	0
$0026	PWM Counter Register High (PCNTH) See page 173.	Read:	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0027	PWM Counter Register Low (PCNTL) See page 173.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0028	PWM Counter Modulo Register High (PMODH) See page 174.	Read:	0	0	0	0	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	X	X	X	X

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 4 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0029	PWM Counter Modulo Register Low (PMODL) See page 174.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	X	X	X	X	X	X	X	X
$002A	PWM 1 Value Register High (PVAL1H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002B	PWM 1 Value Register Low (PVAL1L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002C	PWM 2 Value Register High (PVAL2H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002D	PWM 2 Value Register Low (PVAL2L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002E	PWM 3 Value Register High (PVAL3H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002F	PWM 3 Value Register Low (PVAL3L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0030	PWM 4 Value Register High (PVAL4H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0031	PWM 4 Value Register Low (PVAL4L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0032	PWM 5 Value Register High (PMVAL5H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 5 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0033	PWM 5 Value Register Low (PVAL5L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0034	PWM 6 Value Register High (PVAL6H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0035	PWM 6 Value Register Low (PMVAL6L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0036	Dead-Time Write-Once Register (DEADTM) See page 181.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0037	PWM Disable Mapping Write-Once Register (DISMAP) See page 181.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0038	SCI Control Register 1 (SCC1) See page 304.	Read:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0039	SCI Control Register 2 (SCC2) See page 307.	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003A	SCI Control Register 3 (SCC3) See page 310.	Read:	R8	T8	0	0	ORIE	NEIE	FEIE	PEIE
		Write:	R		R	R				
		Reset:	U	U	0	0	0	0	0	0
$003B	SCI Status Register 1 (SCS1) See page 312.	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:	R	R	R	R	R	R	R	R
		Reset:	1	1	0	0	0	0	0	0
$003C	SCI Status Register 2 (SCS2) See page 316.	Read:	0	0	0	0	0	0	BKF	RPF
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 6 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003D	SCI Data Register (SCDR) See page 317.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$003E	SCI Baud Rate Register (SCBR) See page 317.	Read:	0	0	SCP1	SCP0	0	SCR2	SCR1	SCR0
		Write:	R	R			R			
		Reset:	0	0	0	0	0	0	0	0
$003F	IRQ Status/Control Register (ISCR) See page 348.	Read:	0	0	0	0	IRQF	0	IMASK1	MODE1
		Write:	R	R	R	R		ACK1		
		Reset:	0	0	0	0	0	0	0	0
$0040	ADC Status and Control Register (ADSCR) See page 364.	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
$0041	ADC Data Register High (ADRH) See page 367.	Read:	0	0	0	0	0	0	AD9	AD8
		Write:	R	R	R	R	R	R	R	R
		Reset:	Unaffected by reset							
$0042	ADC Data Register Low (ADRL) See page 368.	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:	R	R	R	R	R	R	R	R
		Reset:	Unaffected by reset							
$0043	ADC Clock Register (ADCLK) See page 369.	Read:	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0
		Write:								R
		Reset:	0	1	1	1	0	0	0	0
$0044	SPI Control Register (SPCR) See page 280.	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0045	SPI Status and Control Register (SPSCR) See page 282.	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:	R		R	R	R			
		Reset:	0	0	0	0	1	0	0	0
$0046	SPI Data Register (SPDR) See page 285.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 7 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0047			Unimplemented							
↓										
$0050			Unimplemented							
$0051	TIMB Status/Control Register (TBSC) See page 244.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	R			
		Reset:	0	0	1	0	0	0	0	0
$0052	TIMB Counter Register High (TBCNTH) See page 247.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0053	TIMB Counter Register Low (TBCNTL) See page 247.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0054	TIMB Counter Modulo Register High (TBMODH) See page 248.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0055	TIMB Counter Modulo Register Low (TBMODL) See page 248.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0056	TIMB Channel 0 Status/Control Register (TBSC0) See page 249.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0057	TIMB Channel 0 Register High (TBCH0H) See page 254.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0058	TIMB Channel 0 Register Low (TBCH0L) See page 254.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 8 of 10)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0059	TIMB Channel 1 Status/Control Register (TBSC1) See page 249.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0
$005A	TIMB Channel 1 Register High (TBCH1H) See page 254.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$005B	TIMB Channel 1 Register Low (TBCH1L) See page 254.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$005C	PLL Control Register (PCTL) See page 124.	Read:	PLLIE	PLLF	PLLON	BCS	1	1	1	1
		Write:		R			R	R	R	R
		Reset:	0	0	1	0	1	1	1	1
$005D	PLL Bandwidth Control Register (PBWC) See page 126.	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	XLD	0	0	0	0
		Write:		R			R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$005E	PLL Programming Register (PPG) See page 128.	Read:	MUL7	MUL6	MUL5	MUL4	VRS7	VRS6	VRS5	VRS4
		Write:								
		Reset:	0	1	1	0	0	1	1	0
$005F			Unimplemented							
$FE00			Unimplemented							
$FE01	SIM Reset Status Register (SRSR) See page 107.	Read:	POR	PIN	COP	ILOP	ILAD	0	LVI	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	1	0	0	0	0	0	0	0
$FE03			Unimplemented							
$FE08	FLASH Control Register (FLCR) See page 59.	Read:	FDIV1	FDIV0	BLK1	BLK0	HVEN	MARGIN	ERASE	PGM
		Write:								
		Reset:	0	0	0	0	0	0	0	0

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 9 of 10)

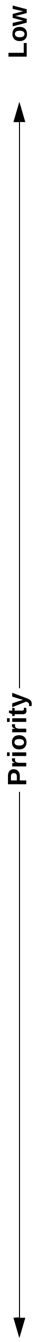
Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0F	LVI Status and Control Register (LVISCR) See page 352.	Read:	LVIOUT	0	TRPSEL	0	0	0	0	0
		Write:	R	R		R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$FF80	FLASH Block Protect Register (FLBPR) See page 66.	Read:	0	0	0	0	BPR3	BPR2	BPR1	BPR0
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	0	0	0	0
$FFFF	COP Control Register (COPCTL) See page 340.	Read:	Low byte of reset vector							
		Write:	Clear COP counter							
		Reset:	Unaffected by reset							

U = Unaffected X = Indeterminate R = Reserved **Bold** = Buffered

Figure 2-2. Control, Status, and Data Registers Summary (Sheet 10 of 10)

Table 2-1 is a list of vector locations.

Table 2-1. Vector Addresses



Address	Vector
$FFD2	SCI transmit vector (high)
$FFD3	SCI transmit vector (low)
$FFD4	SCI receive vector (high)
$FFD5	SCI receive vector (low)
$FFD6	SCI error vector (high)
$FFD7	SCI error vector (low)
$FFD8	SPI transmit vector (high)(1)
$FFD9	SPI transmit vector (low)(1)
$FFDA	SPI receive vector (high)(1)
$FFDB	SPI receive vector (low)(1)
$FFDC	A/D vector (high)
$FFDD	A/D vector (low)
$FFDE	TIM B overflow vector (high)
$FFDF	TIM B overflow vector (low)
$FFE0	TIM B channel 1 vector (high)
$FFE1	TIM B channel 1 vector (low)
$FFE2	TIM B channel 0 vector (high)
$FFE3	TIM B channel 0 vector (low)
$FFE4	TIM A overflow vector (high)
$FFE5	TIM A overflow vector (low)
$FFE6	TIM A channel 3 vector (high)
$FFE7	TIM A channel 3 vector (low)
$FFE8	TIM A channel 2 vector (high)
$FFE9	TIM A channel 2 vector (low)
$FFEA	TIM A channel 1 vector (high)
$FFEB	TIM A channel 1 vector (low)
$FFEC	TIM A channel 0 vector (high)
$FFED	TIM A channel 0 vector (low)

1. The SPI module is not available in the 56-pin SDIP package.

Table 2-1. Vector Addresses (Continued)

Address	Vector
$FFEE	PWMMC vector (high)
$FFEF	PWMMC vector (low)
$FFF0	FAULT 4 (high)
$FFF1	FAULT 4 (low)
$FFF2	FAULT 3 (high)
$FFF3	FAULT 3 (low)
$FFF4	FAULT 2 (high)
$FFF5	FAULT 2 (low)
$FFF6	FAULT 1 (high)
$FFF7	FAULT 1 (low)
$FFF8	PLL vector (high)
$FFF9	PLL vector (low)
$FFFA	IRQ vector (high)
$FFFB	IRQ vector (low)
$FFFC	SWI vector (high)
$FFFD	SWI vector (low)
$FFFE	Reset vector (high)
$FFFF	Reset vector (low)

2.6 Monitor ROM

The 240 bytes at addresses $FE10–$FEFF are reserved ROM addresses that contain the instructions for the monitor functions. See **Section 10. Monitor ROM (MON)**.

Section 3. Random-Access Memory (RAM)

3.1 Contents

3.2 Introduction

This section describes the 768 bytes of random-access memory (RAM).

3.3 Functional Description

Addresses \$0060–\$035F 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 160 bytes of RAM. Because the location of the stack RAM is programmable, all page zero RAM locations can be used for input/output (I/O) control and user data or code. When the stack pointer is moved from its reset location at \$00FF, direct addressing mode instructions can access efficiently all page zero RAM locations. Page zero RAM, therefore, provides ideal locations for frequently accessed global variables.

Before processing an interrupt, the central processor unit (CPU) uses five bytes of the stack to save the contents of the CPU registers.

NOTE: *For M68HC05 and M1468HC05 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 4. FLASH Memory

4.1 Contents

4.2 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 and erase operations are enabled through the use of an internal charge pump.

4.3 Functional Description

The FLASH memory is an array of 24,064 bytes with an additional 46 bytes of user vectors and one byte of block protection. An erased bit reads as a logic 0 and a programmed bit reads as a logic 1. Program and erase operations are facilitated through control bits in a memory mapped register. Details for these operations appear later in this section.

Memory in the FLASH array is organized into pages within rows. There are eight pages of memory per row with eight bytes per page. The minimum erase block size is a single row, 64 bytes. Programming is performed on a per-page basis, eight bytes at a time.

The address ranges for the user memory and vectors are:

- $A000–$FDFF
- $FF80, block protect register (FLBPR)
- $FE08, FLASH control register (FLCR)
- $FFD2–$FFFF (These locations are reserved for user-defined interrupt and reset vectors.)

When programming the FLASH, just enough program time must be used to program a page. Too much program time can result in a program disturb condition, in which case an erased bit on the row being programmed becomes unintentionally programmed. Program disturb is avoided by using an iterative program and margin read technique known as the smart page programming algorithm. The smart programming algorithm is required whenever the user is programming the array (see **4.7 FLASH Program/Margin Read Operation**).

As well, to avoid the program disturb issue, each row should not be programmed more than eight times before it is erased. The eight program cycle maximum per row aligns with the architecture's eight pages of storage per row. The margin read step of the smart programming algorithm is used to ensure programmed bits are programmed to sufficient margin for data retention over the device's lifetime.

The row architecture for this array is:

- $A000–$A03F (row 0)
- $A040–$A07F (row 1)
- $A080–$A0CF (row 2)
- ----------------------------
- $FFBF–$FFFF (row 511)

NOTE: *Programming tools are available from Freescale. Contact a local Freescale representative for more information.*

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

4.4 FLASH Control Register

The FLASH control register controls FLASH program, erase, and margin read operations.

Address: $FE08

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	FDIV1	FDIV0	BLK1	BLK0	HVEN	MARGIN	ERASE	PGM
Reset:	0	0	0	0	0	0	0	0

Figure 4-1. FLASH Control Register (FLCR)

FDIV1 — Frequency Divide Control Bit

This read/write bit together with FDIV0 selects the factor by which the charge pump clock is divided from the system clock. See **4.5 FLASH Charge Pump Frequency Control**.

FDIV0 — Frequency Divide Control Bit

This read/write bit together with FDIV1 selects the factor by which the charge pump clock is divided from the system clock. See **4.5 FLASH Charge Pump Frequency Control**.

BLK1— Block Erase Control Bit

This read/write bit together with BLK0 allows erasing of blocks of varying size. See **4.6 FLASH Erase Operation** for a description of available block sizes.

BLK0 — Block Erase Control Bit

This read/write bit together with BLK1 allows erasing of blocks of varying size. See **4.6 FLASH Erase Operation** for a description of available block sizes.

---

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

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 be set only if either PGM = 1 or ERASE = 1 and the proper sequence for program/margin read or erase is followed.

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

MARGIN — Margin Read Control Bit

This read/write bit configures the memory for margin read operation. MARGIN cannot be set if the HVEN = 1. MARGIN will automatically return to unset if asserted when HVEN is set.

1 = Verify operation selected
0 = Verify 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 set 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 set at the same time.

1 = Program operation selected
0 = Program operation unselected

4.5 FLASH Charge Pump Frequency Control

The internal charge pump required for program, margin read, and erase operations is designed to operate most efficiently with a 2-MHz clock. The charge pump clock is derived from the bus clock. **Table 4-1** shows how the FDIV bits are used to select a charge pump frequency based on the bus clock frequency. Program and erase operations cannot be performed if the bus clock frequency is below 2 MHz.

Table 4-1. Charge Pump Clock Frequency

FDIV1	FDIV0	Pump Clock Frequency	Bus Clock Frequency
0	0	Bus frequency ÷ 1	1.8 MHz–2.5 MHz
0	1	Bus frequency ÷ 2	3.6 MHz–5 MHz
1	0	Bus frequency ÷ 2	3.6 MHz–5 MHz
1	1	Bus frequency ÷ 4	7.2 MHz –8 MHz

4.6 FLASH Erase Operation

Use this step-by-step procedure to erase a block of FLASH memory:

1. Set the ERASE bit, the BLK0, BLK1, FDIV0, and FDIV1 bits in the FLASH control register. See **Table 4-2** for block sizes. See **Table 4-1** for FDIV settings.
2. To ensure this target portion of the array is unprotected, read the block protect register address $FF80. See **4.8 FLASH Block Protection** and **4.9 FLASH Block Protect Register** for more information.
3. Write to any FLASH address with any data within the block address range desired.
4. Set the HVEN bit.
5. Wait for a time, t_{Erase}.
6. Clear the HVEN bit.
7. Wait for a time, $t_{Kill,}$ for the high voltages to dissipate.
8. Clear the ERASE bit.
9. After a time, t_{HVD}, the memory can be accessed in read mode again.

NOTE: *While these operations must be performed in the order shown, other unrelated operations may occur between the steps. Once the HVEN bit is set, the array cannot be read.*

Mask interrupts prior to setting the HVEN bit.

Table 4-2 shows the various block sizes which can be erased in one erase operation.

Table 4-2. Erase Block Sizes

BLK1	BLK0	Block Size, Addresses Cared
0	0	Full array: 24 Kbytes
0	1	One-half array: 16 Kbytes (A14)
1	0	Eight rows: 512 bytes (A14–A9)
1	1	Single row: 64 bytes (A14–A6)

In step 2 of the erase operation, the cared addresses are latched and used to determine the location of the block to be erased. For instance, with BLK0 = BLK1 = 0, writing to any FLASH address in the range $A000–$FFFF will enable the full-array erase.

4.7 FLASH Program/Margin Read Operation

NOTE: *After a total of eight program operations have been applied to a row, the row must be erased before further programming to avoid program disturb. An erased byte will read $00.*

Programming of the FLASH memory is done on a page basis. A page consists of eight consecutive bytes starting from address $XXX0 or $XXX8. The purpose of the margin read mode is to ensure that data has been programmed with sufficient margin for long-term data retention. While performing a margin read, the operation is the same as for ordinary read mode except that a built-in counter stretches the data access for an additional eight cycles to allow sensing of the lower cell current. Margin read mode imposes a more stringent read condition on the bit cell to ensure that the bit cell is programmed with enough margin for long-term data retention. During these eight cycles the COP counter continues to run. The user must account for these extra cycles within COP feed loops. A margin read cycle can only follow a program operation.

To program and margin read the FLASH memory, use this step-by-step procedure:

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 block protect register.
3. Write data to the eight bytes of the page being programmed. This requires eight separate write operations.
4. Set the HVEN bit.
5. Wait for a time, t_{PROG}.
6. Clear the HVEN bit.
7. Wait for a time, t_{HVTV}.
8. Set the MARGIN bit.
9. Wait for a time, t_{VTP}.
10. Clear the PGM bit.
11. Wait for a time, t_{HVD}.
12. Read back data in verify mode. This is done in eight separate read operations which are each stretched by eight cycles. Once the HVEN bit is set, the array cannot be read.
13. Clear the MARGIN bit.

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

This program/margin read sequence is repeated throughout the memory until all data is programmed. For minimum overall programming time and least program disturb effect, the smart programming algorithm should be followed. (See **4.6 FLASH Erase Operation**.)

NOTE: *Mask interrupts prior to setting HVEN.*

Smart Programming Algorithm

Page Program/Margin Read Procedure

Note: This algorithm is mandatory for programming the FLASH.

Note: This page program algorithm assumes the page/s to be programmed are initially erased.

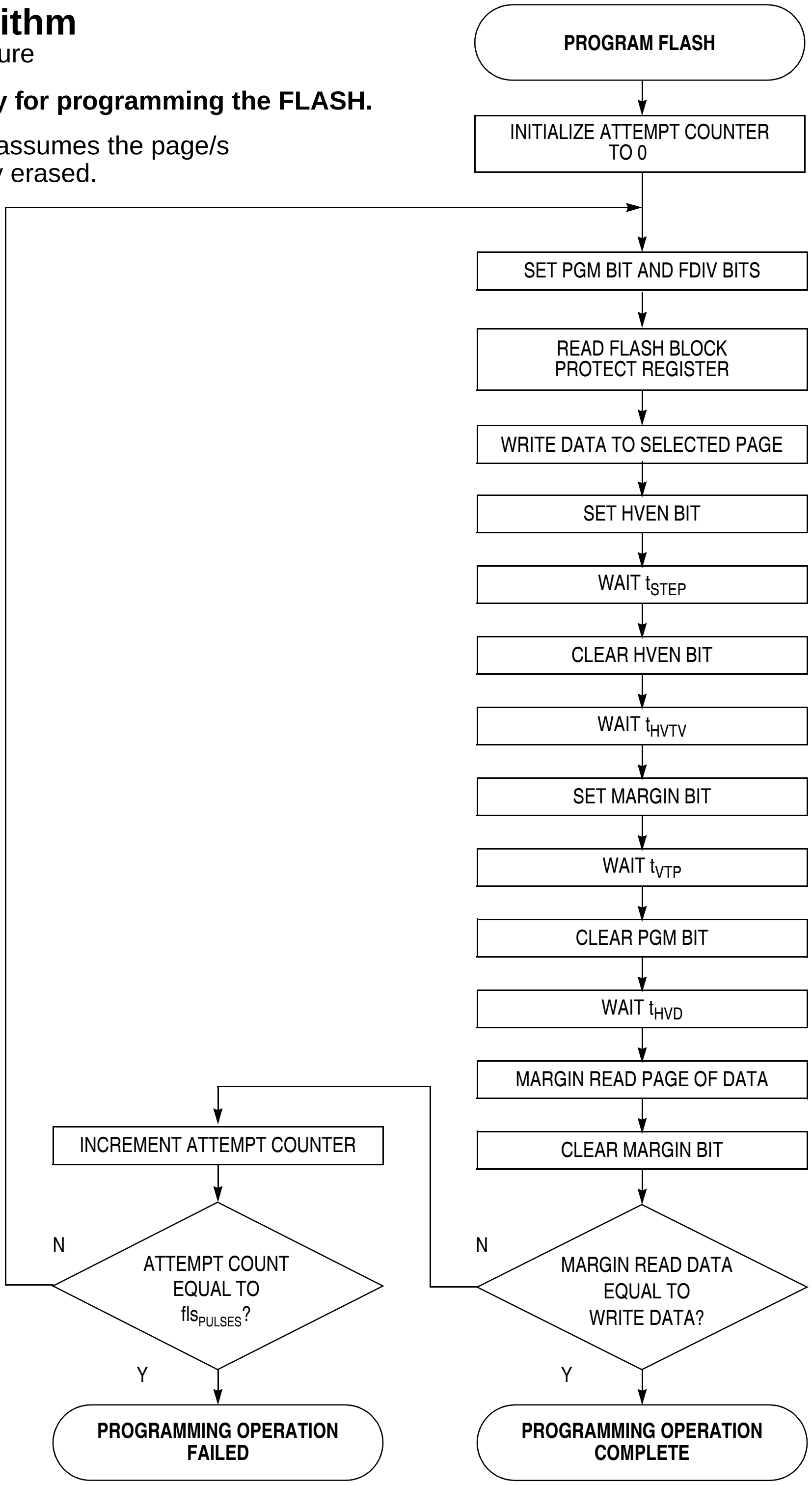


Figure 4-2. Smart Programming Algorithm

4.8 FLASH Block Protection

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.*

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 blocks of memory from unintentional erase or program operations due to system malfunction. This protection is done by reserving a location in the memory for block protect information and requiring that this location be read to enable setting of the HVEN bit. When the block protect register is read, its contents are latched by the FLASH control logic. If the address range for an erase or program operation includes a protected block, the PGM or ERASE bit is cleared which prevents the HVEN bit in the FLASH control register from being set so that no high voltage is allowed in the array.

When the block protect register is erased (all 0s), the entire memory is accessible for program and erase. When bits within the register are programmed, they lock blocks of memory address ranges as shown in **4.9 FLASH Block Protect Register**. The block protect register itself can be erased or programmed only with an external voltage, V_{HI}, present on the $\overline{\text{IRQ}}$ pin. This voltage also allows entry from reset into the monitor mode.

4.9 FLASH Block Protect Register

The block protect register is implemented as a byte within the FLASH memory. Each bit, when programmed, protects a range of addresses in the FLASH.

Address: $FF80

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	BPR3	BPR2	BPR1	BPR0
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 4-3. FLASH Block Protect Register (FLBPR)

BPR3 — Block Protect Register Bit 3

This bit protects the memory contents in the address range $F000–$FFFF.

1 = Address range protected from erase or program
0 = Address range open to erase or program

BPR2 — Block Protect Register Bit 2

This bit protects the memory contents in the address range $E000–$FFFF.

1 = Address range protected from erase or program
0 = Address range open to erase or program

BPR1 — Block Protect Register Bit 1

This bit protects the memory contents in the address range $C000–$FFFF.

1 = Address range protected from erase or program
0 = Address range open to erase or program

BPR0 — Block Protect Register Bit 0

This bit protects the memory contents in the address range $A000–$FFFF.

1 = Address range protected from erase or program
0 = Address range open to erase or program

By programming the block protect bits, a portion of the memory will be locked so that no further erase or program operations may be performed. Programming more than one bit at a time is redundant. If both bit 1 and bit 2 are set, for instance, the address range $C000 through $FFFF is locked. If all bits are erased, then all of the memory is available for erase and program. The presence of a voltage + V_{HI} on the $\overline{\text{IRQ}}$ pin will bypass the block protection so that all of the memory, including the block protect register, is open for program and erase operations.

4.10 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. When the MCU is put into wait mode, the charge pump for the FLASH is disabled so that either a program or erase operation will not continue. If the memory is in either program mode (PGM = 1, HVEN = 1) or erase mode (ERASE = 1, HVEN = 1), then it will remain in that mode during wait. Exit from wait must now be done with a reset rather than an interrupt because if exiting wait with an interrupt, the memory will not be in read mode and the interrupt vector cannot be read from the memory.

Section 5. Configuration Register (CONFIG)

5.1 Contents

5.2 Introduction

This section describes the configuration register (CONFIG). This register contains bits that configure these options:

- Resets caused by the low-voltage inhibit (LVI) module
- Power to the LVI module
- Computer operating properly (COP) module
- Top-side pulse-width modulator (PWM) polarity
- Bottom-side PWM polarity
- Edge-aligned versus center-aligned PWMs
- Six independent PWMs versus three complementary PWM pairs

5.3 Functional Description

The configuration register (CONFIG) is used in the initialization of various options. The configuration register 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 MCU, it is recommended that this register be written immediately after reset. The configuration register is located at $001F and may be read at anytime.

NOTE: *On a FLASH device, the options are one-time writeable by the user after each reset. The registers are not in the FLASH memory but are special registers containing one-time writeable latches after each reset. Upon a reset, the configuration register defaults to predetermined settings as shown in* ***Figure 5-1****.*

If the LVI module and the LVI reset signal are enabled, a reset occurs when V_{DD} *falls to a voltage,* V_{LVRx}*, and remains at or below that level for at least nine consecutive CPU cycles. Once an LVI reset occurs, the MCU remains in reset until* V_{DD} *rises to a voltage,* V_{LVRX}*.*

Address: $001F

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EDGE	BOTNEG	TOPNEG	INDEP	LVIRST	LVIPWR	Bit 1	COPD
Reset:	0	0	0	0	1	1	0	0

Figure 5-1. Configuration Register (CONFIG)

EDGE — Edge-Align Enable Bit

EDGE determines if the motor control PWM will operate in edge-aligned mode or center-aligned mode. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)**.

1 = Edge-aligned mode enabled

0 = Center-aligned mode enabled

BOTNEG — Bottom-Side PWM Polarity Bit

BOTNEG determines if the bottom-side PWMs will have positive or negative polarity. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)**.

1 = Negative polarity
0 = Positive polarity

TOPNEG — Top-Side PWM Polarity Bit

TOPNEG determines if the top-side PWMs will have positive or negative polarity. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)**.

1 = Negative polarity
0 = Positive polarity

INDEP — Independent Mode Enable Bit

INDEP determines if the motor control PWMs will be six independent PWMs or three complementary PWM pairs. See **Section 9. Pulse-Width Modulator for Motor Control (PWMMC)**.

1 = Six independent PWMs
0 = Three complementary PWM pairs

LVIRST — LVI Reset Enable Bit

LVIRST enables the reset signal from the LVI module. See **Section 18. Low-Voltage Inhibit (LVI)**.

1 = LVI module resets enabled
0 = LVI module resets disabled

LVIPWR — LVI Power Enable Bit

LVIPWR enables the LVI module. See **Section 18. Low-Voltage Inhibit (LVI)**.

1 = LVI module power enabled
0 = LVI module power disabled

Bit 1

Writing a 0 or a 1 to bit 1 has no effect on MCU operation. Bit 1 operates the same as the other bits within this write-once register operate.

COPD — COP Disable Bit

COPD disables the COP module. See **Section 16. Computer Operating Properly (COP)**.

1 = COP module disabled
0 = COP module enabled

Section 6. Central Processor Unit (CPU)

6.1 Contents

6.2 Introduction

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

6.3 Features

Features of the CPU include:

- Fully upward, object-code compatibility 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 wait mode

6.4 CPU Registers

Figure 6-1 shows the five CPU registers. CPU registers are not part of the memory map.

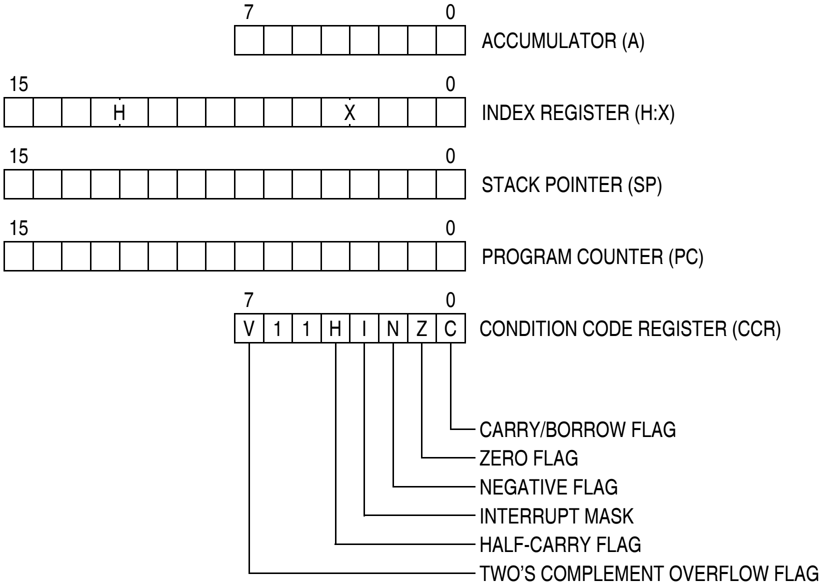


Figure 6-1. CPU Registers

6.4.1 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.

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

Figure 6-2. Accumulator (A)

6.4.2 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.

	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 6-3. Index Register (H:X)

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

6.4.3 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 6-4. Stack Pointer (SP)

NOTE: *The location of the stack is arbitrary and may be relocated anywhere in RAM. Moving the SP out of page zero ($0000–$00FF) frees direct address (page zero) space. For correct operation, the stack pointer must point only to RAM locations.*

6.4.4 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 6-5. Program Counter (PC)

6.4.5 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 functions of the condition code register are described here.

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

X = Indeterminate

Figure 6-6. 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 or 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 Bit

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 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

6.5 Arithmetic/Logic Unit (ALU)

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

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

6.6 Instruction Set Summary

Table 6-1 provides a summary of the M68HC08 instruction set.

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

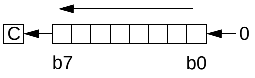
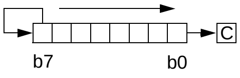
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X ADC *opr*,SP ADC *opr*,SP	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A9 B9 C9 D9 E9 F9 9EE9 9ED9	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 ← (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 ← (SP) + (16 « M)	–	–	–	–	–	–	IMM	A7	ii	2
AIX #*opr*	Add Immediate Value (Signed) to H:X	H:X ← (H:X) + (16 « M)	–	–	–	–	–	–	IMM	AF	ii	2
AND #*opr* AND *opr* AND *opr* AND *opr*,X AND *opr*,X AND ,X AND *opr*,SP AND *opr*,SP	Logical AND	A ← (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 ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3

Table 6-1. Instruction Set Summary (Sheet 2 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BCLR *n*, *opr*	Clear Bit n in M	Mn ← 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
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{\text{IRQ}}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{\text{IRQ}}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 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	–	–	↕	↕	–	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 ← (PC) + 2 + *rel* ? (Z) | (N ⊕ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) =1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3

Table 6-1. Instruction Set Summary (Sheet 3 of 8)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3
BRCLR *n*,*opr*,*rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0) 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

Table 6-1. Instruction Set Summary (Sheet 4 of 8)

Source Form	Operation	Description	Effect on CCR V	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
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
COM *opr* COMA COMX COM *opr*,X COM ,X COM *opr*,SP	Complement (One’s Complement)	$M \leftarrow (\overline{M}) = \$FF - (M)$ $A \leftarrow (\overline{A}) = \$FF - (M)$ $X \leftarrow (\overline{X}) = \$FF - (M)$ $M \leftarrow (\overline{M}) = \$FF - (M)$ $M \leftarrow (\overline{M}) = \$FF - (M)$ $M \leftarrow (\overline{M}) = \$FF - (M)$	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

Table 6-1. Instruction Set Summary (Sheet 5 of 8)

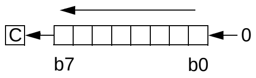
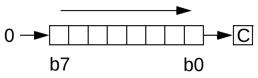
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
INC *opr*	Increment	M ← (M) + 1	↕	–	–	↕	↕	–	DIR	3C	dd	4
INCA		A ← (A) + 1							INH	4C		1
INCX		X ← (X) + 1							INH	5C		1
INC *opr*,X		M ← (M) + 1							IX1	6C	ff	4
INC ,X		M ← (M) + 1							IX	7C		3
INC *opr*,SP		M ← (M) + 1							SP1	9E6C	ff	5
JMP *opr*	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR	BC	dd	2
JMP *opr*									EXT	CC	hh ll	3
JMP *opr*,X									IX2	DC	ee ff	4
JMP *opr*,X									IX1	EC	ff	3
JMP ,X									IX	FC		2
JSR *opr*	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	BD	dd	4
JSR *opr*									EXT	CD	hh ll	5
JSR *opr*,X									IX2	DD	ee ff	6
JSR *opr*,X									IX1	ED	ff	5
JSR ,X									IX	FD		4
LDA #*opr*	Load A from M	A ← (M)	0	–	–	↕	↕	–	IMM	A6	ii	2
LDA *opr*									DIR	B6	dd	3
LDA *opr*									EXT	C6	hh ll	4
LDA *opr*,X									IX2	D6	ee ff	4
LDA *opr*,X									IX1	E6	ff	3
LDA ,X									IX	F6		2
LDA *opr*,SP									SP1	9EE6	ff	4
LDA *opr*,SP									SP2	9ED6	ee ff	5
LDHX #*opr*	Load H:X from M	H:X ← (M:M + 1)	0	–	–	↕	↕	–	IMM	45	ii jj	3
LDHX *opr*									DIR	55	dd	4
LDX #*opr*	Load X from M	X ← (M)	0	–	–	↕	↕	–	IMM	AE	ii	2
LDX *opr*									DIR	BE	dd	3
LDX *opr*									EXT	CE	hh ll	4
LDX *opr*,X									IX2	DE	ee ff	4
LDX *opr*,X									IX1	EE	ff	3
LDX ,X									IX	FE		2
LDX *opr*,SP									SP1	9EEE	ff	4
LDX *opr*,SP									SP2	9EDE	ee ff	5
LSL *opr*	Logical Shift Left (Same as ASL)		↕	–	–	↕	↕	↕	DIR	38	dd	4
LSLA									INH	48		1
LSLX									INH	58		1
LSL *opr*,X									IX1	68	ff	4
LSL ,X									IX	78		3
LSL *opr*,SP									SP1	9E68	ff	5
LSR *opr*	Logical Shift Right		↕	–	–	0	↕	↕	DIR	34	dd	4
LSRA									INH	44		1
LSR*X*									INH	54		1
LSR *opr*,X									IX1	64	ff	4
LSR ,X									IX	74		3
LSR *opr*,SP									SP1	9E64	ff	5
MOV *opr,opr*	Move	$(M)_{Destination} \leftarrow (M)_{Source}$ H:X ← (H:X) + 1 (IX+D, DIX+)	0	–	–	↕	↕	–	DD	4E	dd dd	5
MOV *opr*,X+									DIX+	5E	dd	4
MOV #*opr,opr*									IMD	6E	ii dd	4
MOV X+,*opr*									IX+D	7E	dd	4
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5

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

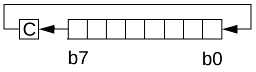
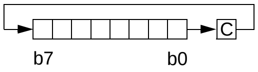
Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
NEG *opr* NEGA NEGX NEG *opr*,X NEG ,X NEG *opr*,SP	Negate (Two's Complement)	M ← –(M) = $00 – (M) A ← –(A) = $00 – (A) X ← –(X) = $00 – (X) M ← –(M) = $00 – (M) M ← –(M) = $00 – (M)	↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	4 1 1 4 3 5
NOP	No Operation	None	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap A	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		3
ORA #*opr* ORA *opr* ORA *opr* ORA *opr*,X ORA *opr*,X ORA ,X ORA *opr*,SP ORA *opr*,SP	Inclusive OR A and M	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	AA BA CA DA EA FA 9EEA 9EDA	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
PSHA	Push A onto Stack	Push (A); SP ← (SP) – 1	–	–	–	–	–	–	INH	87		2
PSHH	Push H onto Stack	Push (H); SP ← (SP) – 1	–	–	–	–	–	–	INH	8B		2
PSHX	Push X onto Stack	Push (X); SP ← (SP) – 1	–	–	–	–	–	–	INH	89		2
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

Table 6-1. 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				
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X SBC *opr*,SP SBC *opr*,SP	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM DIR EXT IX2 IX1 IX SP1 SP2	A2 B2 C2 D2 E2 F2 9EE2 9ED2	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask	I ← 1	–	–	1	–	–	–	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X STA *opr*,SP STA *opr*,SP	Store A in M	M ← (A)	0	–	–	↕	↕	–	DIR EXT IX2 IX1 IX SP1 SP2	B7 C7 D7 E7 F7 9EE7 9ED7	dd hh ll ee ff ff ff ee ff	3 4 4 3 2 4 5
STHX *opr*	Store H:X in M	(M:M + 1) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STOP	Enable $\overline{\text{IRQ}}$ 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

Table 6-1. 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				
TST *opr* TSTA TSTX TST *opr*,X TST ,X TST *opr*,SP	Test for Negative or Zero	(A) – \$00 or (X) – \$00 or (M) – \$00	0	–	–	↕	↕	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	3 1 1 3 2 4
TSX	Transfer SP to H:X	H:X ← (SP) + 1	–	–	–	–	–	–	INH	95		2
TXA	Transfer X to A	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer H:X to SP	(SP) ← (H:X) – 1	–	–	–	–	–	–	INH	94		2
WAIT	Enable Interrupts; Stop Processor	I bit ← 0	–	–	0	–	–	–	INH	8F		1

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

6.7 Opcode Map

See **Table 6-2**.

Table 6-2. Opcode Map

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

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

*Pre-byte for stack pointer indexed instructions

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

Section 7. System Integration Module (SIM)

7.1 Contents

7.2 Introduction

This section describes the system integration module (SIM). Together with the central processor unit (CPU), the SIM controls all MCU activities. A block diagram of the SIM is shown in **Figure 7-1**.

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:
  - 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 7-1 shows the internal signal names used in this section.

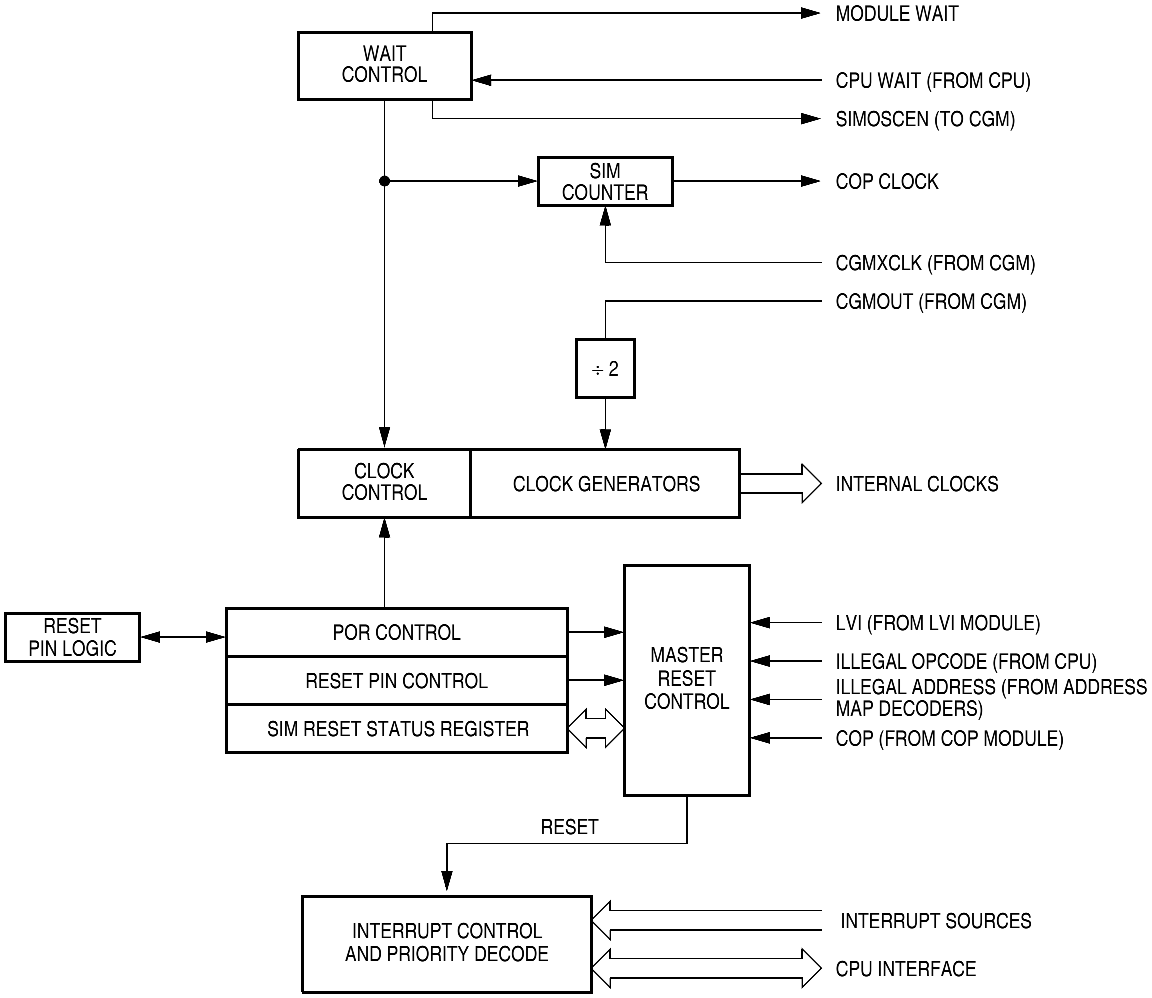


Figure 7-1. SIM Block Diagram

Table 7-1. Signal Name Conventions

Signal Name	Description
CGMXCLK	Buffered version of OSC1 from clock generator module (CGM)
CGMVCLK	Phase-locked loop (PLL) circuit output
CGMOUT	PLL-based or OSC1-based clock output from CGM module (Bus clock = CGMOUT divided by two)
IAB	Internal address bus
IDB	Internal data bus
PORRST	Signal from the power-on reset module to the SIM
IRST	Internal reset signal
R/$\overline{W}$	Read/write signal

7.3 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 7-2**. This clock can come from either an external oscillator or from the on-chip phase-locked loop (PLL) circuit. See **Section 8. Clock Generator Module (CGM)**.

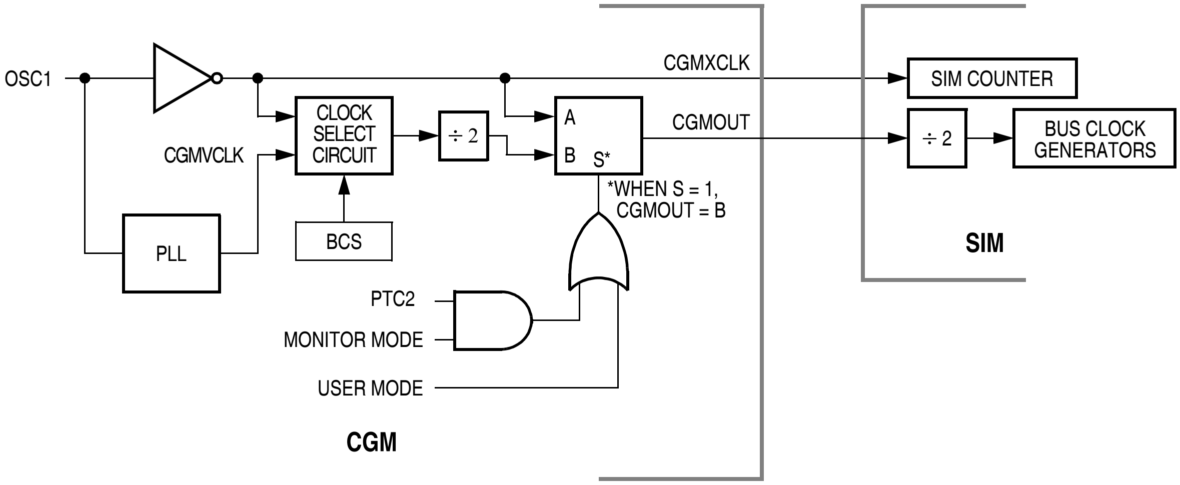


Figure 7-2. CGM Clock Signals

7.3.1 Bus Timing

In user mode**,** the internal bus frequency is either the crystal oscillator output (CGMXCLK) divided by four or the PLL output (CGMVCLK) divided by four. See **Section 8. Clock Generator Module (CGM)**.

7.3.2 Clock Startup from POR or LVI Reset

When the power-on reset (POR) module or the low-voltage inhibit (LVI) module generates a reset, the clocks to the CPU and peripherals are inactive and held in an inactive phase until after the 4096 CGMXCLK cycle POR timeout has completed. The $\overline{\text{RST}}$ pin is driven low by the SIM during this entire period. The internal bus (IBUS) clocks start upon completion of the timeout.

7.3.3 Clocks in Wait 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.

7.4 Reset and System Initialization

The MCU has these reset sources:

- Power-on reset module (POR)
- External reset pin ($\overline{\text{RST}}$)
- Computer operating properly (COP) module
- Low-voltage inhibit (LVI) module
- Illegal opcode
- Illegal address

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 **7.5 SIM Counter**), but an external reset does not. Each of the resets sets a corresponding bit in the SIM reset status register (SRSR). See **7.7.2 SIM Reset Status Register**.

7.4.1 External Pin Reset

Pulling the asynchronous $\overline{\text{RST}}$ pin low halts all processing. The PIN bit of the SIM reset status register (SRSR) is set as long as $\overline{\text{RST}}$ is held low for a minimum of 67 CGMXCLK cycles, assuming that neither the POR nor the LVI was the source of the reset. See **Table 7-2** for details. **Figure 7-3** shows the relative timing.

Table 7-2. PIN Bit Set Timing

Reset Type	Number of Cycles Required to Set PIN
POR/LVI	4163 (4096 + 64 + 3)
All others	67 (64 + 3)

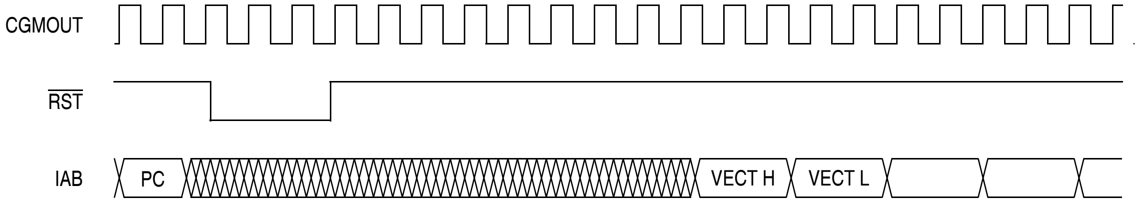


Figure 7-3. External Reset Timing

7.4.2 Active Resets from Internal Sources

All internal reset sources actively pull the $\overline{\text{RST}}$ pin low for 32 CGMXCLK cycles to allow resetting of external peripherals. The internal reset signal (IRST) continues to be asserted for an additional 32 cycles (see **Figure 7-4**). An internal reset can be caused by an illegal address, illegal opcode, COP timeout, LVI, or POR. (See **Figure 7-5**.)

NOTE: *For LVI or POR resets, the SIM cycles through 4096 CGMXCLK cycles during which the SIM forces the $\overline{\text{RST}}$ pin low. The internal reset signal then follows the sequence from the falling edge of $\overline{\text{RST}}$, as shown in **Figure 7-4**.*

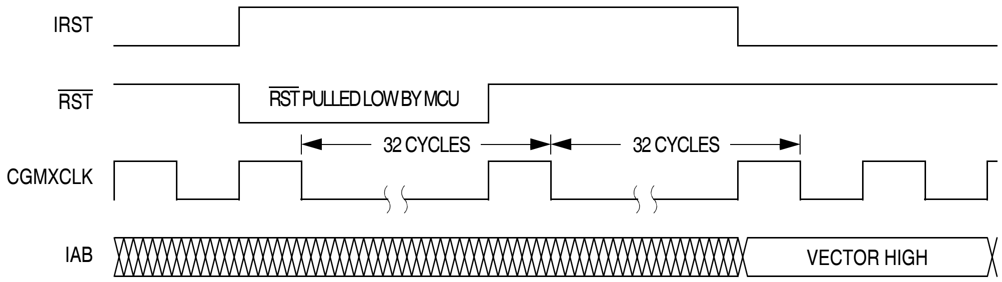


Figure 7-4. Internal Reset Timing

The COP reset is asynchronous to the bus clock.

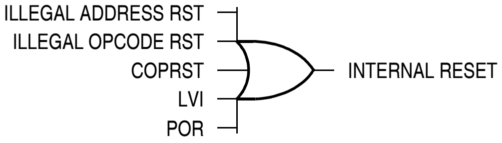


Figure 7-5. Sources of Internal Reset

The active reset feature allows the part to issue a reset to peripherals and other chips within a system built around the MCU.

7.4.2.1 Power-On Reset (POR)

When power is first applied to the MCU, the power-on reset (POR) module 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 CGMXCLK cycles. Sixty-four 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.

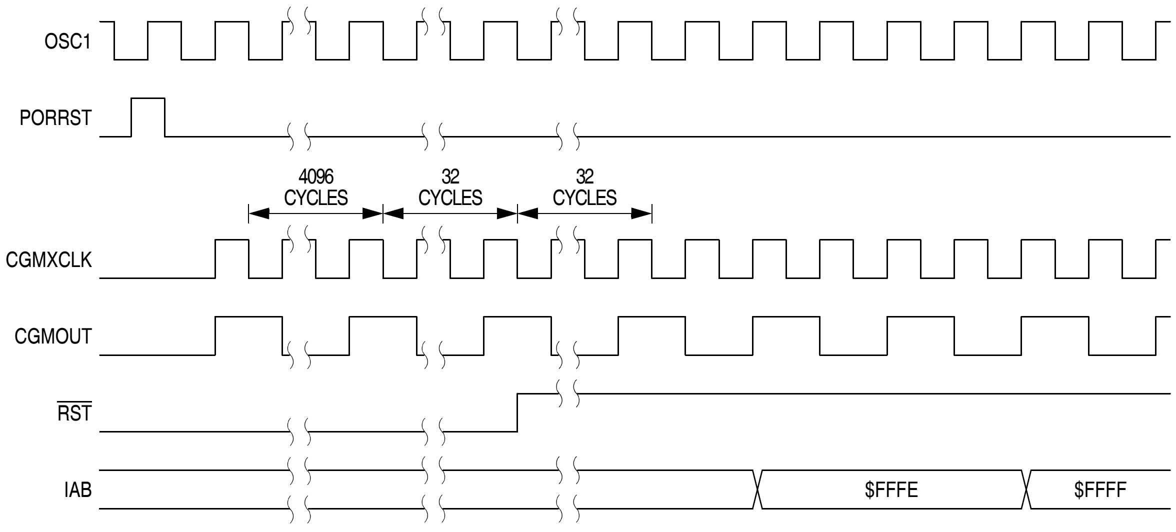


Figure 7-6. POR Recovery

7.4.2.2 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 $\overline{\text{RST}}$ pin for all internal reset sources.

To prevent a COP module timeout, write any value to location $FFFF. Writing to location $FFFF clears the COP counter and bits 12–4 of the SIM counter. The SIM counter output, which occurs at least every $2^{13} – 2^4$ CGMXCLK cycles, drives the COP counter. The COP should be serviced as soon as possible out of reset to guarantee the maximum amount of time before the first timeout.

The COP module is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ}}$ pin is held at 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 $\overline{\text{RST}}$ or the $\overline{\text{IRQ}}$ pin. This prevents the COP from becoming disabled as a result of external noise. During a break state, V_{HI} on the $\overline{\text{RST}}$ pin disables the COP module.

7.4.2.3 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.

Because the MC68HC908MR24 has stop mode disabled, execution of the STOP instruction will cause an illegal opcode reset.

7.4.2.4 Illegal Address Reset

An opcode fetch from addresses other than FLASH or RAM addresses generates an illegal address reset (unimplemented locations within memory map). 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.

7.4.2.5 Low-Voltage Inhibit (LVI) Reset

The low-voltage inhibit (LVI) module asserts its output to the SIM when the V_{DD} voltage falls to the V_{LVRx} voltage and remains at or below that level for at least nine consecutive CPU cycles (see **21.6 DC Electrical Characteristics (V_{DD} = 5.0 Vdc ± 10%)**). The LVI bit in the SIM reset status register (SRSR) is set, and the external reset pin ($\overline{\text{RST}}$) is held low while the SIM counter counts out 4096 CGMXCLK cycles. Sixty-four CGMXCLK cycles later, the CPU is released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

7.5 SIM Counter

The SIM counter is used by the power-on reset (POR) module to allow the oscillator time to stabilize before enabling the internal bus (IBUS) clocks. The SIM counter also serves as a prescaler for the computer operating properly (COP) module. The SIM counter overflow supplies the clock for the COP module. The SIM counter is 13 bits long and is clocked by the falling edge of CGMXCLK.

7.5.1 SIM Counter During Power-On Reset

The power-on reset (POR) module 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 (CGM) module to drive the bus clock state machine.

7.5.2 SIM Counter and Reset States

External reset has no effect on the SIM counter. The SIM counter is free-running after all reset states. For counter control and internal reset recovery sequences, see **7.4.2 Active Resets from Internal Sources**.

7.6 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

7.6.1 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 return-from-interrupt (RTI) instruction recovers the CPU register contents from the stack so that normal processing can resume. **Figure 7-7** shows interrupt entry timing. **Figure 7-9** 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 7-8**.

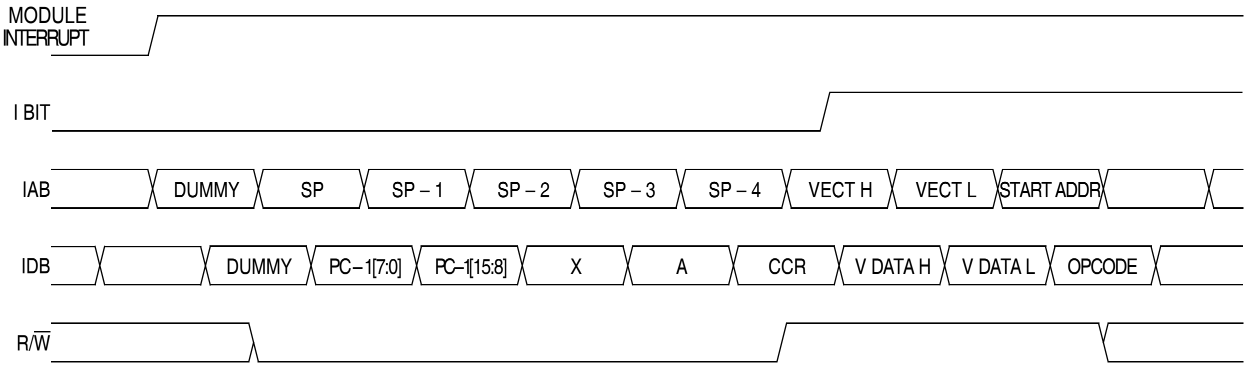


Figure 7-7. Interrupt Entry

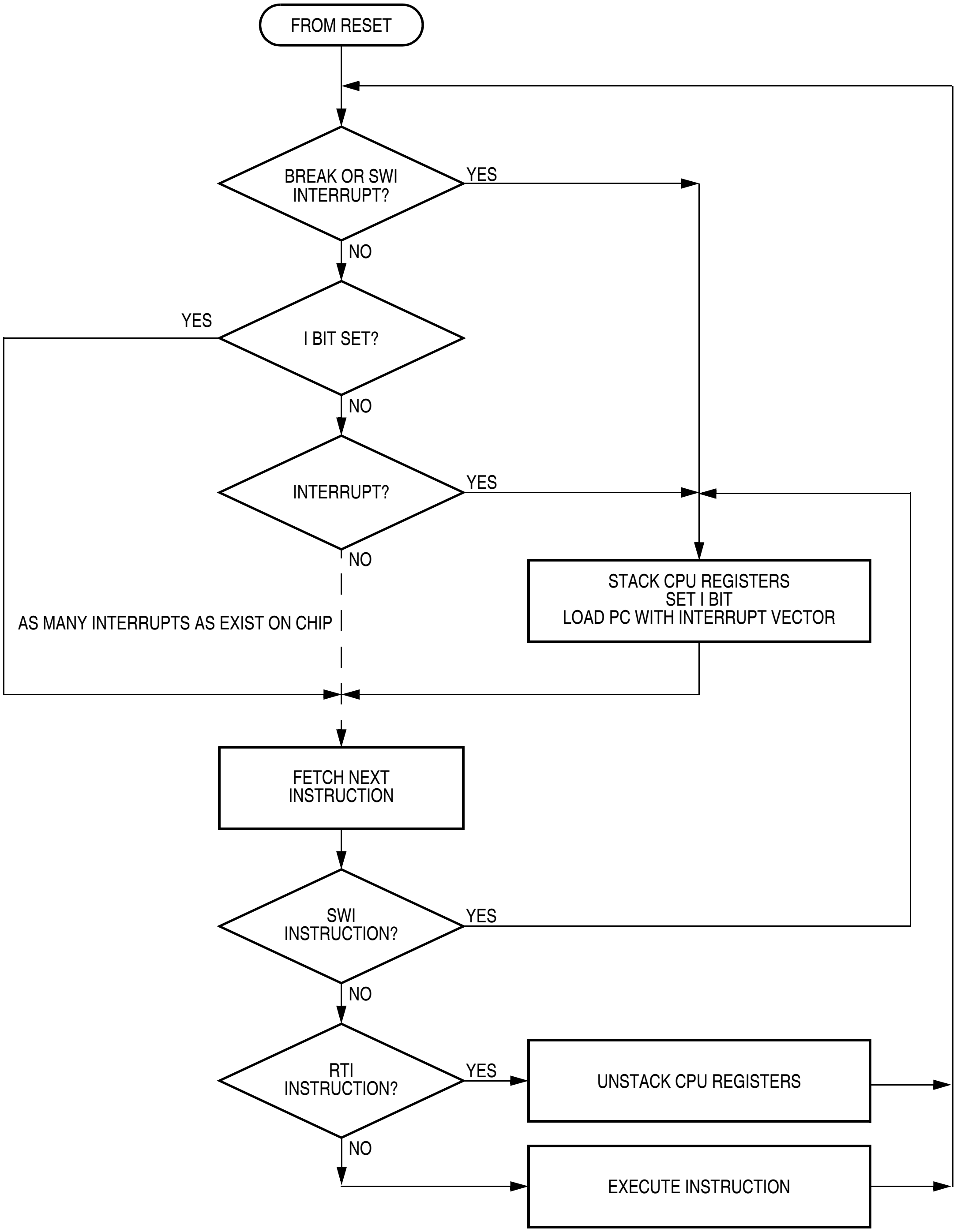


Figure 7-8. Interrupt Processing

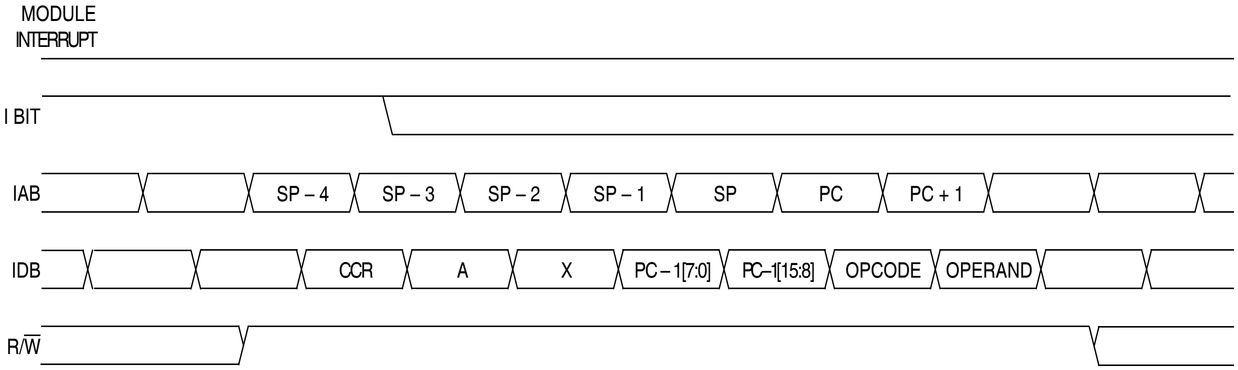


Figure 7-9. Interrupt Recovery

7.6.1.1 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 7-10** 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 load-accumulator-from-memory (LDA) instruction is executed.

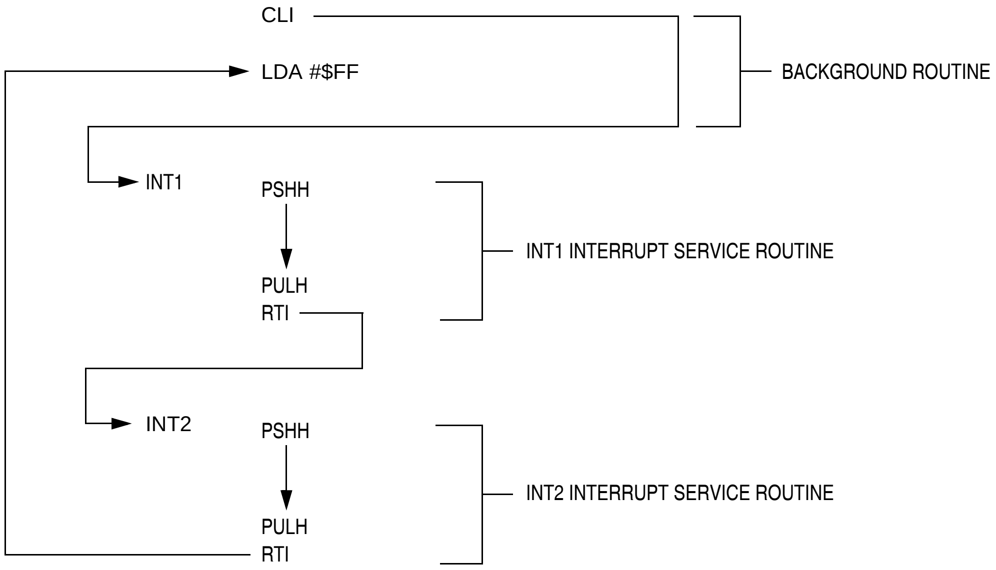


Figure 7-10. Interrupt Recognition Example

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.*

7.6.1.2 Software Interrupt (SWI) Instruction

The software interrupt (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.

7.6.2 Reset

All reset sources always have equal and highest priority and cannot be arbitrated.

7.7 Low-Power Mode

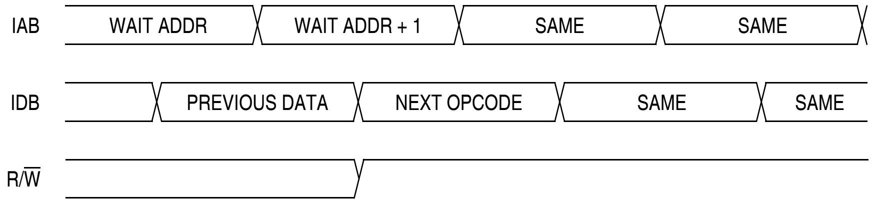
Executing the WAIT instruction puts the MCU in a low power-consumption mode for standby situations. The SIM holds the CPU in a non-clocked state. WAIT clears the interrupt mask (I) in the condition code register, allowing interrupts to occur.

7.7.1 Wait Mode

In wait mode, the CPU clocks are inactive while the peripheral clocks continue to run. **Figure 7-11** 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. 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.

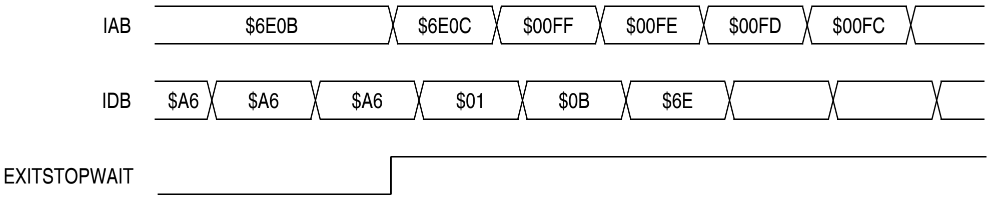
Wait mode can also be exited by a reset. If the COP disable bit, COPD, in the configuration register is logic 0, then the computer operating properly module (COP) is enabled and remains active in wait mode.



Note: Previous data can be operand data or the WAIT opcode, depending on the last instruction.

Figure 7-11. Wait Mode Entry Timing

Figure 7-12 and **Figure 7-13** show the timing for wait recovery.



Note: EXITSTOPWAIT = $\overline{\text{RST}}$ pin or CPU interrupt

Figure 7-12. Wait Recovery from Interrupt

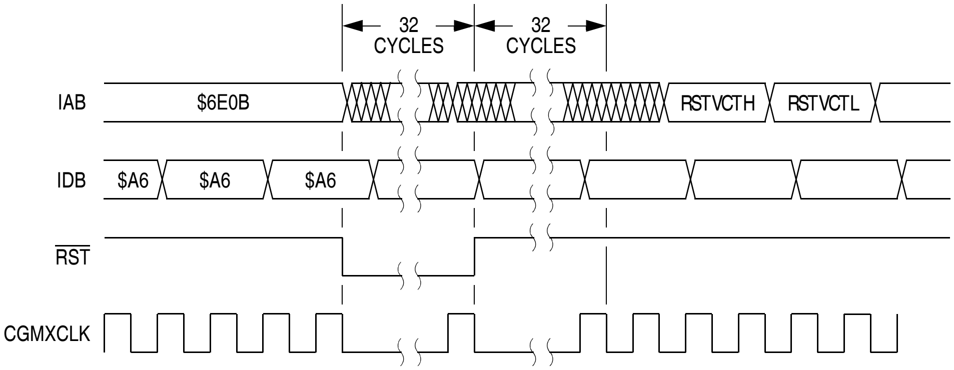


Figure 7-13. Wait Recovery from Internal Reset

7.7.2 SIM Reset Status Register

The SIM reset status register (SRSR) contains six flags that show the source of the last reset. 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	0	LVI	0
Write:	R	R	R	R	R	R	R	R
Reset:	1	0	0	0	0	0	0	0

R = Reserved

Figure 7-14. 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

LVI — Low-Voltage Inhibit Reset Bit
1 = Last reset was caused by the LVI circuit
0 = POR or read of SRSR

Section 8. Clock Generator Module (CGM)

8.1 Contents

8.2 Introduction

This section describes the clock generator module (CGM, version A). The CGM generates the crystal clock signal, CGMXCLK, which operates at the frequency of the crystal. The CGM also generates the base clock signal, CGMOUT, from which the system integration module (SIM) derives the system clocks.

CGMOUT is based on either the crystal clock divided by two or the phase-locked loop (PLL) clock, CGMVCLK, divided by two. The PLL is a frequency generator designed for use with crystals or ceramic resonators. The PLL can generate an 8-MHz bus frequency without using a 32-MHz external clock.

8.3 Features

Features of the CGM include:

- PLL with output frequency in integer multiples of the crystal reference
- Programmable hardware voltage-controlled oscillator (VCO) for low-jitter operation
- Automatic bandwidth control mode for low-jitter operation
- Automatic frequency lock detector
- Central processor unit (CPU) interrupt on entry or exit from locked condition

8.4 Functional Description

The CGM consists of three major submodules:

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

Figure 8-1 shows the structure of the CGM.

8.4.1 Crystal Oscillator Circuit

The crystal oscillator circuit consists of an inverting amplifier and an external crystal. The OSC1 pin is the input to the amplifier and the OSC2 pin is the output. The SIMOSCEN signal from the system integration module (SIM) enables the crystal oscillator circuit.

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

CGMXCLK can be used by other modules which require precise timing for operation. The duty cycle of CGMXCLK is not guaranteed to be 50 percent and depends on external factors, including the crystal and related external components.

An externally generated clock also can feed the OSC1 pin of the crystal oscillator circuit. Connect the external clock to the OSC1 pin and let the OSC2 pin float.

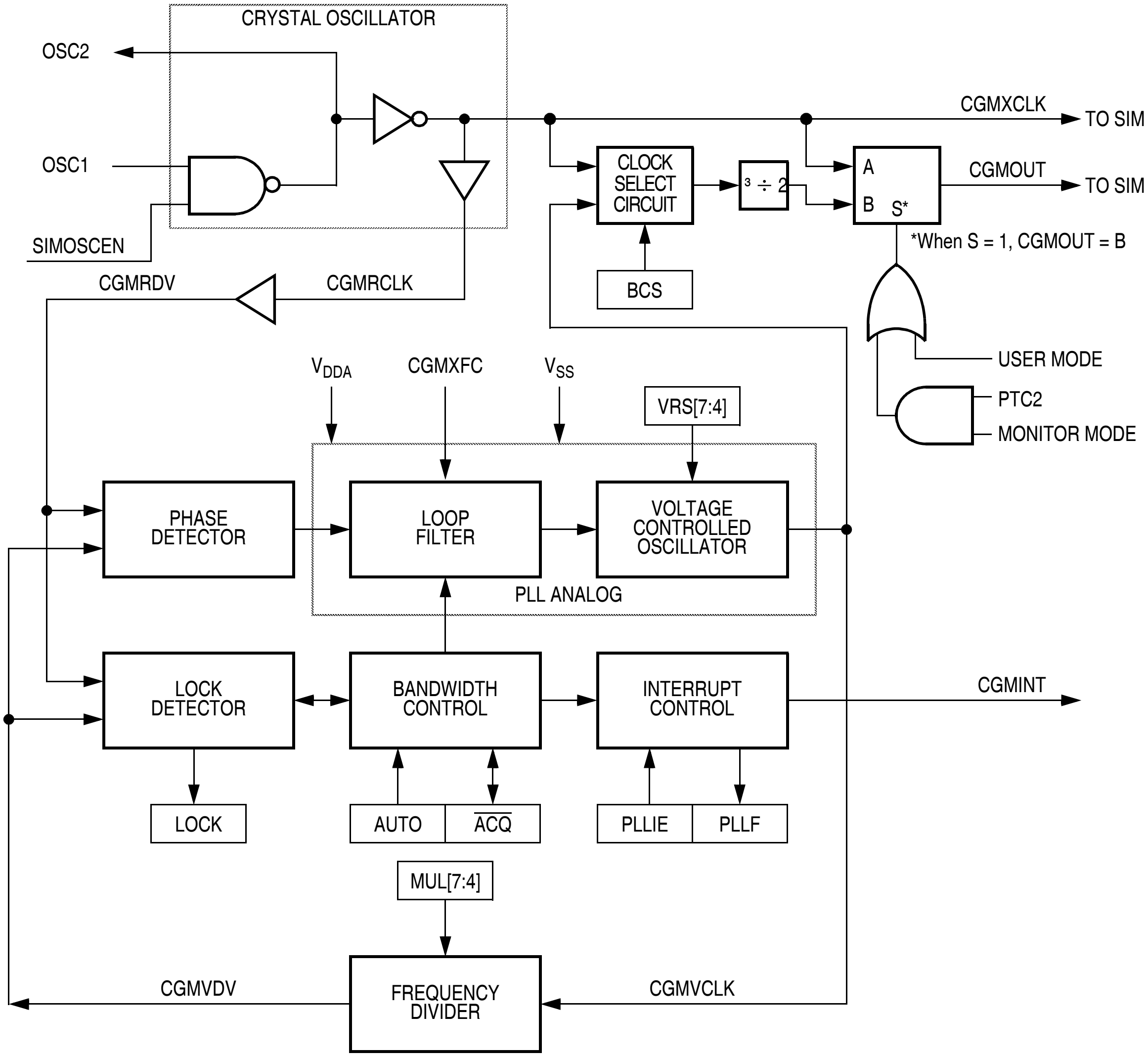


Figure 8-1. CGM Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$005C	PLL Control Register (PCTL) See page 124.	Read:	PLLIE	PLLF	PLLON	BCS	1	1	1	1
		Write:		R			R	R	R	R
		Reset:	0	0	1	0	1	1	1	1
$005D	PLL Bandwidth Control Register (PBWC) See page 126.	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	XLD	0	0	0	0
		Write:		R			R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$005E	PLL Programming Register (PPG) See page 128.	Read:	MUL7	MUL6	MUL5	MUL4	VRS7	VRS6	VRS5	VRS4
		Write:								
		Reset:	0	1	1	0	0	1	1	0

R = Reserved

Figure 8-2. CGM I/O Register Summary

8.4.2 Phase-Locked Loop Circuit (PLL)

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

8.4.2.1 PLL Circuits

The PLL consists of these circuits:

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

The operating range of the VCO is programmable for a wide range of frequencies and for maximum immunity to external noise, including supply and CGMXFC noise. The VCO frequency is bound to a range from roughly one-half to twice the center-of-range frequency, f_{VRS}. Modulating the voltage on the CGMXFC pin changes the frequency within this range. By design, f_{VRS} is equal to the nominal center-of-range frequency, f_{NOM}, (4.9152 MHz) times a linear factor, L or $(L)f_{NOM}$.

CGMRCLK is the PLL reference clock, a buffered version of CGMXCLK. CGMRCLK runs at a frequency, f_{RCLK}, and is fed to the PLL through a buffer. The buffer output is the final reference clock, CGMRDV, running at a frequency $f_{RDV} = f_{RCLK}$.

The VCO's output clock, CGMVCLK, running at a frequency f_{VCLK}, is fed back through a programmable modulo divider. The modulo divider reduces the VCO clock by a factor, N. The divider's output is the VCO feedback clock, CGMVDV, running at a frequency $f_{VDV} = f_{VCLK}/N$. (See **8.4.2.4 Programming the PLL** for more information.)

The phase detector then compares the VCO feedback clock, CGMVDV, with the final reference clock, CGMRDV. A correction pulse is generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the external capacitor connected to CGMXFC based on the width and direction of the correction pulse. The filter can make fast or slow corrections depending on its mode, described in **8.4.2.2 Acquisition and Tracking Modes**. The value of the external capacitor and the reference frequency determines the speed of the corrections and the stability of the PLL.

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

8.4.2.2 Acquisition and Tracking Modes

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

1. Acquisition mode — In acquisition mode, the filter can make large frequency corrections to the VCO. This mode is used at PLL startup or when the PLL has suffered a severe noise hit and the VCO frequency is far off the desired frequency. When in acquisition mode, the $\overline{\text{ACQ}}$ bit is clear in the PLL bandwidth control register. (See **8.6.2 PLL Bandwidth Control Register**.)
2. Tracking mode — In tracking mode, the filter makes only small corrections to the frequency of the VCO. PLL jitter is much lower in tracking mode, but the response to noise is also slower. The PLL enters tracking mode when the VCO frequency is nearly correct, such as when the PLL is selected as the base clock source. (See **8.4.3 Base Clock Selector Circuit**.) The PLL is automatically in tracking mode when not in acquisition mode or when the $\overline{\text{ACQ}}$ bit is set.

8.4.2.3 Manual and Automatic PLL Bandwidth Modes

The PLL can change the bandwidth or operational mode of the loop filter manually or automatically.

In automatic bandwidth control mode (AUTO = 1), the lock detector automatically switches between acquisition and tracking modes. Automatic bandwidth control mode also is used to determine when the VCO clock, CGMVCLK, is safe to use as the source for the base clock, CGMOUT. See **8.6.2 PLL Bandwidth Control Register**. If PLL interrupts are enabled, the software can wait for a PLL interrupt request and then check the LOCK bit. If interrupts are disabled, software can poll the LOCK bit continuously (during PLL startup, usually) or at periodic intervals. In either case, when the LOCK bit is set, the VCO clock is safe to use as the source for the base clock. See **8.4.3 Base Clock Selector Circuit**. If the VCO is selected as the source for the base clock and the LOCK bit is clear, the PLL has suffered a severe noise hit and the software must take appropriate action, depending on the application. See **8.7 Interrupts** for information and precautions on using interrupts.

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

- The $\overline{\text{ACQ}}$ bit (see **8.6.2 PLL Bandwidth Control Register**) is a read-only indicator of the mode of the filter. For more information, see **8.4.2.2 Acquisition and Tracking Modes**.
- The $\overline{\text{ACQ}}$ bit is set when the VCO frequency is within a certain tolerance, Δ_{TRK}, and is cleared when the VCO frequency is out of a certain tolerance, Δ_{UNT}. For more information, see **8.9 Acquisition/Lock Time Specifications**.
- The LOCK bit is a read-only indicator of the locked state of the PLL.
- The LOCK bit is set when the VCO frequency is within a certain tolerance, Δ_{Lock}, and is cleared when the VCO frequency is out of a certain tolerance, Δ_{UNL}. For more information, see **8.9 Acquisition/Lock Time Specifications**.
- CPU interrupts can occur if enabled (PLLIE = 1) when the PLL's lock condition changes, toggling the LOCK bit. For more information, see **8.6.1 PLL Control Register**.

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

- $\overline{\text{ACQ}}$ is a writable control bit that controls the mode of the filter. Before turning on the PLL in manual mode, the $\overline{\text{ACQ}}$ bit must be clear.
- Before entering tracking mode ($\overline{\text{ACQ}}$ = 1), software must wait a given time, t_{ACQ} (see **8.9 Acquisition/Lock Time Specifications**), after turning on the PLL by setting PLLON in the PLL control register (PCTL).
- Software must wait a given time, t_{AL}, after entering tracking mode before selecting the PLL as the clock source to CGMOUT (BCS = 1).
- The LOCK bit is disabled.
- CPU interrupts from the CGM are disabled.

8.4.2.4 Programming the PLL

Use this 9-step procedure to program the PLL. **Table 8-1** lists the variables used and their meaning.

Table 8-1. Variable Definitions

Variable	Definition
f_{BUSDES}	Desired bus clock frequency
$f_{VCLKDES}$	Desired VCO clock frequency
f_{RCLK}	Chosen refereence crystal frequency
f_{VCLK}	Calculated VCO clock frequency
f_{BUS}	Calculated bus clock frequency
f_{NOM}	Nominal VCO center frequency
f_{VRS}	Shifted FCO center frequency

1. Choose the desired bus frequency, f_{BUSDES}.

   Example: f_{BUSDES} = 8 MHz

2. Calculate the desired VCO frequency, $f_{VCLKDES}$.

   $$f_{VCLKDES} = 4 \text{ x } f_{BUSDES}$$

   Example: $f_{VCLKDES}$ = 4 x 8 MHz = 32 MHz

3. Using a referernce frequency, f_{RCLK}, equal to the crystal frequency, calculate the VCO frequency multiplier, N. Round the result to the nearest integer.

   $$N = \frac{f_{VCLKDES}}{f_{RCLK}}$$

   $$\text{Example: } N = \frac{32\text{ MHz}}{4\text{ MHz}} = 8\text{ MHz}$$

4. Calculate the VCO frequency, f_{VCLK}.

   $$f_{VCLK} = N \text{ x } f_{RCLK}$$

   Example: f_{VCLK} = 8 x 4 MHz = 32 MHz

5. Calculate the bus frequency, f_{BUS}, and compare f_{BUS} with f_{BUSDES}.

$$f_{BUS} = \frac{f_{VCLK}}{4}$$

$$\text{Example: } N = \frac{32\text{ MHz}}{4\text{ MHz}} = 8\text{ MHz}$$

6. If the calculated f_{BUS} is not within the tolerance limits of your application, select another f_{BUSDES} or another f_{RCLK}.
7. Using the value 4.9152 MHz for f_{NOM}, calculate the VCO linear range multiplier, L. The linear range multiplier controls the frequency range of the PLL.

$$L = \text{round}\left(\frac{f_{VCLK}}{f_{NOM}}\right)$$

$$\text{Example: } L = \frac{32\text{ MHz}}{4.9152\text{ MHz}} = 7\text{ MHz}$$

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

$$f_{VRS} = L \text{ x } f_{NOM}$$

$$\text{Example: } f_{VRS} = 7 \text{ x } 4.9152\text{ MHz} = 34.4\text{ MHz}$$

NOTE: *For proper operation,*

$$f_{VRS} - f_{VCLK} | \leq \frac{f_{NOM}}{2}$$

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

9. Program the PLL registers accordingly:
    a. In the upper four bits of the PLL programming register (PPG), program the binary equivalent of N.
    b. In the lower four bits of the PLL programming register (PPG), program the binary equivalent of L.

8.4.2.5 Special Programming Exceptions

The programming method described in **8.4.2.4 Programming the PLL** does not account for possible exceptions. A value of zero for N or L is meaningless when used in the equations given. To account for these exceptions:

- A zero value for N is interpreted exactly the same as a value of one.
- A zero value for L disables the PLL and prevents its selection as the source for the base clock. (See **8.4.3 Base Clock Selector Circuit**.)

8.4.3 Base Clock Selector Circuit

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

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

8.4.4 CGM External Connections

In its typical configuration, the CGM requires seven external components. Five of these are for the crystal oscillator and two are for the PLL.

The crystal oscillator is normally connected in a Pierce oscillator configuration, as shown in **Figure 8-3**. **Figure 8-3** shows only the logical representation of the internal components and may not represent actual circuitry.

The oscillator configuration uses five components:

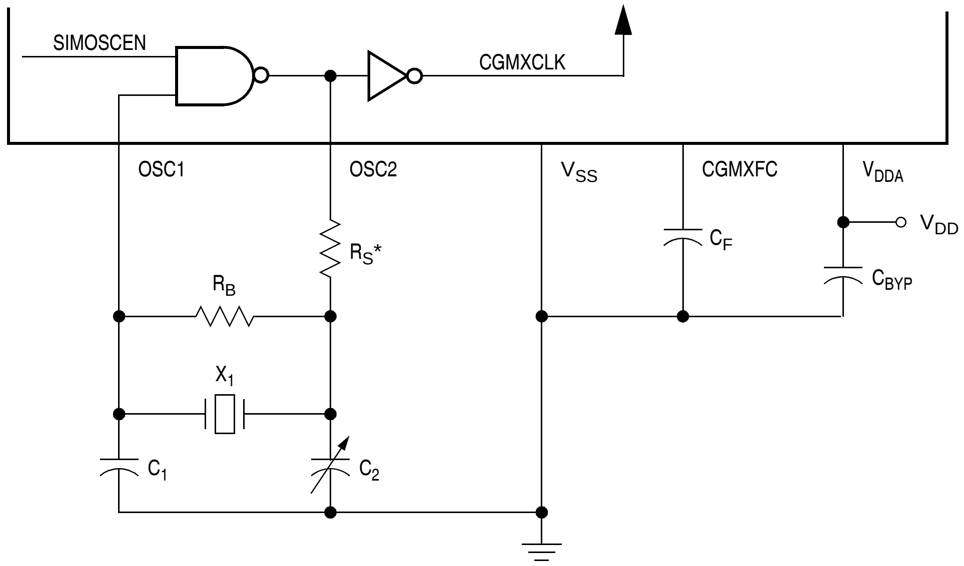
1. Crystal, X_1
2. Fixed capacitor, C_1
3. Tuning capacitor, C_2 (can also be a fixed capacitor)
4. Feedback resistor, R_B
5. Series resistor, R_S (optional)

The series resistor (R_S) is included in the diagram 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.

Figure 8-3 also shows the external components for the PLL:

- Bypass capacitor, C_{BYP}
- Filter capacitor, C_F

NOTE: *Routing should be done with great care to minimize signal cross talk and noise. (See* ***8.9 Acquisition/Lock Time Specifications*** *for routing information and more information on the filter capacitor's value and its effects on PLL performance.)*



*R_S can be 0 (shorted) when used with higher-frequency crystals. Refer to manufacturer's data.

Figure 8-3. CGM External Connections

8.5 I/O Signals

This section describes the CGM input/output (I/O) signals.

8.5.1 Crystal Amplifier Input Pin (OSC1)

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

8.5.2 Crystal Amplifier Output Pin (OSC2)

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

8.5.3 External Filter Capacitor Pin (CGMXFC)

The CGMXFC pin is required by the loop filter to filter out phase corrections. A small external capacitor is connected to this pin.

NOTE: *To prevent noise problems, C_F should be placed as close to the CGMXFC pin as possible, with minimum routing distances and no routing of other signals across the C_F connection.*

8.5.4 PLL Analog Power Pin (V_{DDA})

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

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

8.5.5 Oscillator Enable Signal (SIMOSCEN)

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

8.5.6 Crystal Output Frequency Signal (CGMXCLK)

CGMXCLK is the crystal oscillator output signal. It runs at the full speed of the crystal (f_{XCLK}) and comes directly from the crystal oscillator circuit. **Figure 8-3** shows only the logical relation of CGMXCLK to OSC1 and OSC2 and may not represent the actual circuitry. The duty cycle of CGMXCLK is unknown and may depend on the crystal and other external factors. Also, the frequency and amplitude of CGMXCLK can be unstable at startup.

8.5.7 CGM Base Clock Output (CGMOUT)

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

8.5.8 CGM CPU Interrupt (CGMINT)

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

8.6 CGM Registers

These registers control and monitor operation of the CGM:

- PLL control register (PCTL)
  See **8.6.1 PLL Control Register**.
- PLL bandwidth control register (PBWC)
  See **8.6.2 PLL Bandwidth Control Register**.
- PLL programming register (PPG)
  See **8.6.3 PLL Programming Register**.

Figure 8-4 is a summary of the CGM registers.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$005C	PLL Control Register (PCTL) See page 124.	Read:	PLLIE	PLLF	PLLON	BCS	1	1	1	1
		Write:		R			R	R	R	R
		Reset:	0	0	1	0	1	1	1	1
$005D	PLL Bandwidth Control Register (PBWC) See page 126.	Read:	AUTO	LOCK	$\overline{\text{ACQ}}$	XLD	0	0	0	0
		Write:		R			R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$005E	PLL Programming Register (PPG) See page 128.	Read:	MUL7	MUL6	MUL5	MUL4	VRS7	VRS6	VRS5	VRS4
		Write:								
		Reset:	0	1	1	0	0	1	1	0

R = Reserved

Notes:
1. When AUTO = 0, PLLIE is forced to logic 0 and is read-only.
2. When AUTO = 0, $\overline{\text{PLLF}}$ and LOCK read as logic 0.
3. When AUTO = 1, $\overline{\text{ACQ}}$ is read-only.
4. When PLLON = 0 or VRS[7:4] = $0, BCS is forced to logic 0 and is read-only.
5. When PLLON = 1, the PLL programming register is read-only.
6. When BCS = 1, PLLON is forced set and is read-only.

Figure 8-4. CGM I/O Register Summary

8.6.1 PLL Control Register

The PLL control register (PCTL) contains the interrupt enable and flag bits, the on/off switch, and the base clock selector bit.

Address: $005C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PLLIE	PLLF	PLLON	BCS	1	1	1	1
Write:		R			R	R	R	R
Reset:	0	0	1	0	1	1	1	1

R = Reserved

Figure 8-5. PLL Control Register (PCTL)

PLLIE — PLL Interrupt Enable Bit

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

1 = PLL interrupts enabled
0 = PLL interrupts disabled

PLLF — PLL Interrupt Flag

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

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

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

PLLON — PLL On Bit

This read/write bit activates the PLL and enables the VCO clock, CGMVCLK. PLLON cannot be cleared if the VCO clock is driving the base clock, CGMOUT (BCS = 1). See **8.4.3 Base Clock Selector Circuit**. Reset sets this bit so that the loop can stabilize as the MCU is powering up.

1 = PLL on
0 = PLL off

BCS — Base Clock Select Bit

This read/write bit selects either the crystal oscillator output, CGMXCLK, or the VCO clock, CGMVCLK, as the source of the CGM output, CGMOUT. CGMOUT frequency is one-half the frequency of the selected clock. BCS cannot be set while the PLLON bit is clear. After toggling BCS, it may take up to three CGMXCLK and three CGMVCLK cycles to complete the transition from one source clock to the other. During the transition, CGMOUT is held in stasis. See **8.4.3 Base Clock Selector Circuit**. Reset clears the BCS bit.

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

NOTE: *PLLON and BCS have built-in protection that prevents the base clock selector circuit from selecting the VCO clock as the source of the base clock if the PLL is off. Therefore, PLLON cannot be cleared when BCS is set, and BCS cannot be set when PLLON is clear. If the PLL is off (PLLON = 0), selecting CGMVCLK requires two writes to the PLL control register. See* ***8.4.3 Base Clock Selector Circuit****.*

PCTL[3:0] — Unimplemented Bits

These bits provide no function and always read as logic 1s.

8.6.2 PLL Bandwidth Control Register

The PLL bandwidth control register (PBWC):

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

Address: $005D

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

R = Reserved

Figure 8-6. PLL Bandwidth Control Register (PBWC)

AUTO — Automatic Bandwidth Control Bit

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

1 = Automatic bandwidth control
0 = Manual bandwidth control

LOCK — Lock Indicator Bit

When the AUTO bit is set, LOCK is a read-only bit that becomes set when the VCO clock, CGMVCLK, is locked (running at the programmed frequency). When the AUTO bit is clear, LOCK reads as logic 0 and has no meaning. Reset clears the LOCK bit.

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

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

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

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

1 = Tracking mode
0 = Acquisition mode

XLD — Crystal Loss Detect Bit

When the VCO output, CGMVCLK, is driving CGMOUT, this read/write bit can indicate whether the crystal reference frequency is active or not. To check the status of the crystal reference, follow these steps:

1. Write a logic 1 to XLD.
2. Wait N × 4 cycles. (N is the VCO frequency multiplier.)
3. Read XLD.

1 = Crystal reference is not active.
0 = Crystal reference is active.

The crystal loss detect function works only when the BCS bit is set, selecting CGMVCLK to drive CGMOUT. When BCS is clear, XLD always reads as logic 0.

PBWC[3:0] — Reserved for Test

These bits enable test functions not available in user mode. To ensure software portability from development systems to user applications, software should write 0s to PBWC[3:0] whenever writing to PBWC.

8.6.3 PLL Programming Register

The PLL programming register (PPG) contains the programming information for the modulo feedback divider and the programming information for the hardware configuration of the VCO.

Address: $005E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MUL7	MUL6	MUL5	MUL4	VRS7	VRS6	VRS5	VRS4
Reset:	0	1	1	0	0	1	1	0

Figure 8-7. PLL Programming Register (PPG)

MUL[7:4] — Multiplier Select Bits

These read/write bits control the modulo feedback divider that selects the VCO frequency multiplier, N. See **8.4.2.1 PLL Circuits** and **8.4.2.4 Programming the PLL**. A value of $0 in the multiplier select bits configures the modulo feedback divider the same as a value of $1. Reset initializes these bits to $6 to give a default multiply value of 6.

Table 8-2. VCO Frequency Multiplier (N) Selection

MUL7:MUL6:MUL5:MUL4	VCO Frequency Multiplier (N)
0000	1
0001	1
0010	2
0011	3
↓	↓
1101	13
1110	14
1111	15

NOTE: *The multiplier select bits have built-in protection that prevents them from being written when the PLL is on (PLLON = 1).*

VRS[7:4] — VCO Range Select Bits

These read/write bits control the hardware center-of-range linear multiplier L, which controls the hardware center-of-range frequency f_{VRS}. See **8.4.2.1 PLL Circuits**, **8.4.2.4 Programming the PLL** and **8.6.1 PLL Control Register**. VRS[7:4] cannot be written when the PLLON bit in the PLL control register (PCTL) is set. See **8.4.2.5 Special Programming Exceptions**. A value of $0 in the VCO range select bits disables the PLL and clears the BCS bit in the PCTL. See **8.4.3 Base Clock Selector Circuit** and **8.4.2.5 Special Programming Exceptions** for more information.

Reset initializes the bits to $6 to give a default range multiply value of 6.

NOTE: *The VCO range select bits have built-in protection that prevents them from being written when the PLL is on (PLLON = 1) and prevents selection of the VCO clock as the source of the base clock (BCS = 1) if the VCO range select bits are all clear.*

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

8.7 Interrupts

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

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

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

8.8 Wait Mode

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

The WAIT instruction does not affect the CGM. Before entering wait mode, software can disengage and turn off the PLL by clearing the BCS and PLLON bits in the PLL control register (PCTL). Less power-sensitive applications can disengage the PLL without turning it off. Applications that require the PLL to wake the MCU from wait mode also can deselect the PLL output without turning off the PLL.

8.9 Acquisition/Lock Time Specifications

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

8.9.1 Acquisition/Lock Time Definitions

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

1 MHz ±50 kHz. Fifty kHz = 5% of the 1-MHz step input. If the system is operating at 1 MHz and suffers a –100-kHz noise hit, the acquisition time is the time taken to return from 900 kHz to 1 MHz ±5 kHz. Five kHz = 5% of the 100-kHz step input.

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

The discrepancy in these definitions makes it difficult to specify an acquisition or lock time for a typical PLL. Therefore, the definitions for acquisition and lock times for this module are:

- Acquisition time, t_{ACQ}, is the time the PLL takes to reduce the error between the actual output frequency and the desired output frequency to less than the tracking mode entry tolerance, Δ_{TRK}. Acquisition time is based on an initial frequency error, $(f_{DES} - f_{ORIG})/f_{DES}$, of not more than ±100 percent. In automatic bandwidth control mode (see **8.4.2.3 Manual and Automatic PLL Bandwidth Modes**), acquisition time expires when the $\overline{\text{ACQ}}$ bit becomes set in the PLL bandwidth control register (PBWC).
- Lock time, t_{Lock}, is the time the PLL takes to reduce the error between the actual output frequency and the desired output frequency to less than the lock mode entry tolerance, Δ_{Lock}. Lock time is based on an initial frequency error, $(f_{DES} - f_{ORIG})/f_{DES}$, of not more than ±100 percent. In automatic bandwidth control mode, lock time expires when the LOCK bit becomes set in the PLL bandwidth control register (PBWC). See **8.4.2.3 Manual and Automatic PLL Bandwidth Modes**.

Obviously, the acquisition and lock times can vary according to how large the frequency error is and may be shorter or longer in many cases.

8.9.2 Parametric Influences on Reaction Time

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

The most critical parameter which affects the reaction times of the PLL is the reference frequency, f_{RDV}. This frequency is the input to the phase detector and controls how often the PLL makes corrections. For stability, the corrections must be small compared to the desired frequency, so several corrections are required to reduce the frequency error. Therefore, the slower the reference the longer it takes to make these corrections. This parameter is also under user control via the choice of crystal frequency, f_{XCLK}.

Another critical parameter is the external filter capacitor. The PLL modifies the voltage on the VCO by adding or subtracting charge from this capacitor. Therefore, the rate at which the voltage changes for a given frequency error (thus change in charge) is proportional to the capacitor size. The size of the capacitor also is related to the stability of the PLL. If the capacitor is too small, the PLL cannot make small enough adjustments to the voltage and the system cannot lock. If the capacitor is too large, the PLL may not be able to adjust the voltage in a reasonable time. See **8.9.3 Choosing a Filter Capacitor**.

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

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

8.9.3 Choosing a Filter Capacitor

As described in **8.9.2 Parametric Influences on Reaction Time**, the external filter capacitor, C_F, is critical to the stability and reaction time of the PLL. The PLL is also dependent on reference frequency and supply voltage. The value of the capacitor must, therefore, be chosen with supply potential and reference frequency in mind. For proper operation, the external filter capacitor must be chosen according to this equation:

$$C_F = C_{FACT}\left(\frac{V_{DDA}}{f_{RDV}}\right)$$

For acceptable values of C_{FACT}, see **8.9 Acquisition/Lock Time Specifications**. For the value of V_{DDA}, choose the voltage potential at which the MCU is operating. If the power supply is variable, choose a value near the middle of the range of possible supply values.

This equation does not always yield a commonly available capacitor size, so round to the nearest available size. If the value is between two different sizes, choose the higher value for better stability. Choosing the lower size may seem attractive for acquisition time improvement, but the PLL can become unstable. Also, always choose a capacitor with a tight tolerance (±20 percent or better) and low dissipation.

8.9.4 Reaction Time Calculation

The actual acquisition and lock times can be calculated using the equations here. These equations yield nominal values under these conditions:

- Correct selection of filter capacitor, C_F
  See **8.9.3 Choosing a Filter Capacitor**.
- Room temperature operation
- Negligible external leakage on CGMXFC
- Negligible noise

The K factor in the equations is derived from internal PLL parameters. K_{ACQ} is the K factor when the PLL is configured in acquisition mode, and

K_{TRK} is the K factor when the PLL is configured in tracking mode. See **8.4.2.2 Acquisition and Tracking Modes**.

$$t_{ACQ} = \left(\frac{V_{DDA}}{f_{RDV}}\right)\left(\frac{8}{K_{ACQ}}\right)$$

$$t_{AL} = \left(\frac{V_{DDA}}{f_{RDV}}\right)\left(\frac{4}{K_{TRK}}\right)$$

$$t_{Lock} = t_{ACQ} + t_{AL}$$

Note the inverse proportionality between the lock time and the reference frequency.

In automatic bandwidth control mode, the acquisition and lock times are quantized into units based on the reference frequency. See **8.4.2.3 Manual and Automatic PLL Bandwidth Modes**. A certain number of clock cycles, n_{ACQ}, is required to ascertain that the PLL is within the tracking mode entry tolerance, Δ_{TRK}, before exiting acquisition mode. A certain number of clock cycles, n_{TRK}, is required to ascertain that the PLL is within the lock mode entry tolerance, Δ_{Lock}. Therefore, the acquisition time, t_{ACQ}, is an integer multiple of n_{ACQ}/f_{RDV}, and the acquisition to lock time, t_{AL}, is an integer multiple of n_{TRK}/f_{RDV}. Also, since the average frequency over the entire measurement period must be within the specified tolerance, the total time usually is longer than t_{LOCK} as calculated in the previous example.

In manual mode, it is usually necessary to wait considerably longer than t_{Lock} before selecting the PLL clock (see **8.4.3 Base Clock Selector Circuit**) because the factors described in **8.9.2 Parametric Influences on Reaction Time** may slow the lock time considerably.

Section 9. Pulse-Width Modulator for Motor Control (PWMMC)

9.1 Contents

9.2 Introduction

This section describes the pulse-width modulator for motor control (PWMMC, version A). The MC68HC908MR24 PWM module can generate three complementary PWM pairs or six independent PWM signals. These PWM signals can be center-aligned or edge-aligned. A block diagram of the PWM module is shown in **Figure 9-1**.

A12-bit timer PWM counter is common to all six channels. PWM resolution is one clock period for edge-aligned operation and two clock periods for center-aligned operation. The clock period is dependent on the internal operating frequency (f_{op}) and a programmable prescaler. The highest resolution for edge-aligned operation is 125 ns (f_{op} = 8 MHz). The highest resolution for center-aligned operation is 250 ns (f_{op} = 8 MHz).

When generating complementary PWM signals, the module features automatic dead-time insertion to the PWM output pairs and transparent toggling of PWM data based upon sensed motor phase current polarity.

A summary of the PWM registers is shown in **Figure 9-2**.

9.3 Features

Features of the PWMMC include:

- Three complimentary PWM pairs or six independent PWM signals
- Edge-aligned PWM signals or center-aligned PWM signals
- PWM signal polarity control
- 20-mA current sink capability on PWM pins
- Manual PWM output control through software
- Programmable fault protection
- Complementary mode featuring:
  - Dead-time insertion
  - Separate top/bottom pulse width correction via current sensing or programmable software bits

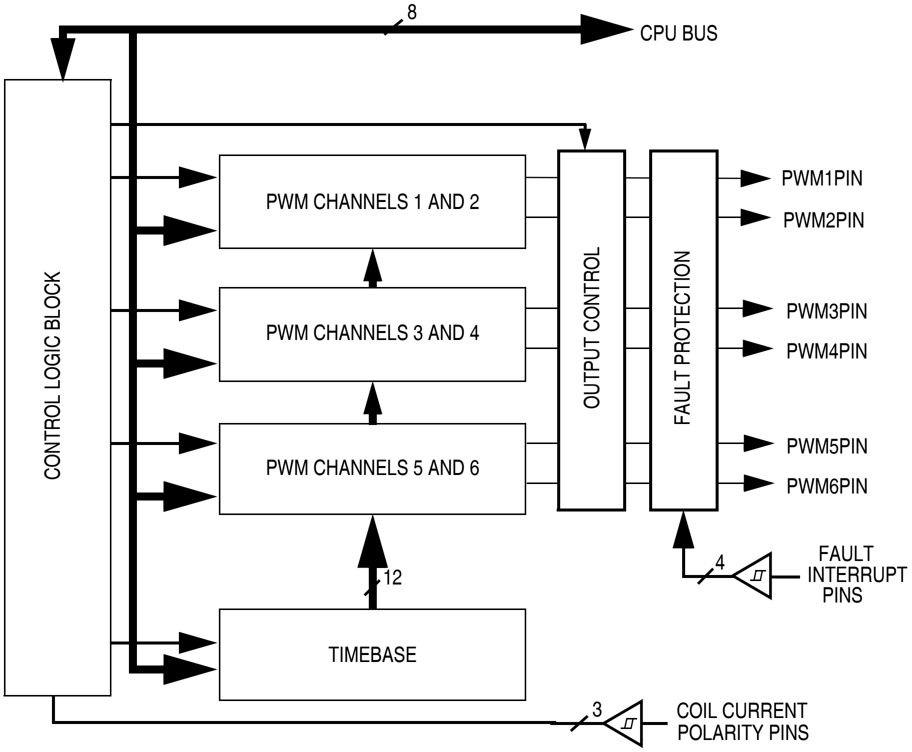


Figure 9-1. PWM Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	PWM Control Register 1 (PCTL1) See page 176.	Read:	DISX	DISY	PWMINT	PWMF	ISENS1	ISENS0	LDOK	PWMEN
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0021	PWM Control Register 2 (PCTL2) See page 179.	Read:	**LDFQ1**	**LDFQ0**	0	**IPOL1**	**IPOL2**	**IPOL3**	**PRSC1**	**PRSC0**
		Write:			▨					
		Reset:	0	0	0	0	0	0	0	0
$0022	Fault Control Register (FCR) See page 182.	Read:	FINT4	FMODE4	FINT3	FMODE3	FINT2	FMODE2	FINT1	FMODE1
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	Fault Status Register (FSR) See page 184.	Read:	FPIN4	FFLAG4	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	U	0	U	0	U	0	U	0
$0024	Fault Acknowledge Register (FTACK) See page 186.	Read:	0	0	DT6	DT5	DT4	DT3	DT2	DT1
		Write:	▨	FTACK4	▨	FTACK3	▨	FTACK2	▨	FTACK1
		Reset:	0	0	0	0	0	0	0	0
$0025	PWM Output Control Register (PWMOUT) See page 187.	Read:	0	OUTCTL	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1
		Write:	▨							
		Reset:	0	0	0	0	0	0	0	0
$0026	PWM Counter Register High (PCNTH) See page 173.	Read:	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0027	PWM Counter Register Low (PCNTL) See page 173.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	0	0	0	0	0	0	0	0
$0028	PWM Counter Modulo Register High (PMODH) See page 174.	Read:	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	▨	▨	▨	▨				
		Reset:	0	0	0	0	X	X	X	X

R = Reserved   **Bold** = Buffered   X = Indeterminate

Figure 9-2. Register Summary (Sheet 1 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0029	PWM Counter Modulo Register Low (PMODL) See page 174.	Read: Write:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Reset:	X	X	X	X	X	X	X	X
$002A	PWM 1 Value Register High (PVAL1H) See page 175.	Read: Write:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Reset:	0	0	0	0	0	0	0	0
$002B	PWM 1 Value Register Low (PVAL1L) See page 175.	Read: Write:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Reset:	0	0	0	0	0	0	0	0
$002C	PWM 2 Value Register High (PVAL2H) See page 175.	Read: Write:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Reset:	0	0	0	0	0	0	0	0
$002D	PWM 2 Value Register Low (PVAL2L) See page 175.	Read: Write:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Reset:	0	0	0	0	0	0	0	0
$002E	PWM 3 Value Register High (PVAL3H) See page 175.	Read: Write:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Reset:	0	0	0	0	0	0	0	0
$002F	PWM 3 Value Register Low (PVAL3L) See page 175.	Read: Write:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Reset:	0	0	0	0	0	0	0	0
$0030	PWM 4 Value Register High (PVAL4H) See page 175.	Read: Write:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Reset:	0	0	0	0	0	0	0	0
$0031	PWM 4 Value Register Low (PVAL4L) See page 175.	Read: Write:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Reset:	0	0	0	0	0	0	0	0

R = Reserved   **Bold** = Buffered   X = Indeterminate

Figure 9-2. Register Summary (Sheet 2 of 3)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0032	PWM 5 Value Register High (PVAL5H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0033	PWM 5 Value Register Low (PVAL5L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0034	PWM 6 Value Register High (PVAL6H) See page 175.	Read:	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0035	PWM 6 Value Register Low (PMVAL6L) See page 175.	Read:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0036	Dead-Time Write-Once Register (DEADTM) See page 181.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0037	PWM Disable Mapping Write-Once Register (DISMAP) See page 181.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1

R = Reserved

Bold = Buffered

X = Indeterminate

Figure 9-2. Register Summary (Sheet 3 of 3)

9.4 Timebase

This section provides a discussion of the timebase.

9.4.1 Resolution

In center-aligned mode, a 12-bit up/down counter is used to create the PWM period. Therefore, the PWM resolution in center-aligned mode is two clocks (highest resolution is 250 ns @ f_{op} = 8 MHz) as shown in **Figure 9-3**. The up/down counter uses the value in the timer modulus register to determine its maximum count. The PWM period will equal: [(timer modulus) x (PWM clock period) x 2].

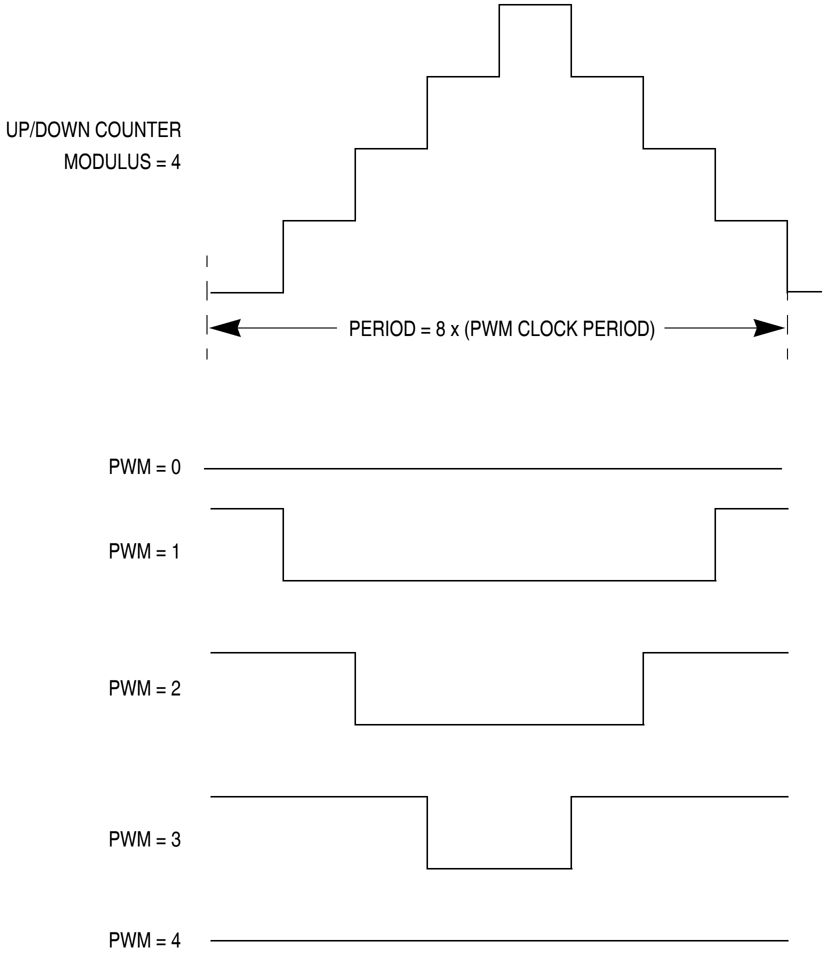


Figure 9-3. Center-Aligned PWM (Positive Polarity)

For edge-aligned mode, a 12-bit up-only counter is used to create the PWM period. Therefore, the PWM resolution in edge-aligned mode is one clock (highest resolution is125 ns @ f_{op} = 8 MHz) as shown in **Figure 9-4**. Again, the timer modulus register is used to determine the maximum count. The PWM period will equal:

[(timer modulus) x (PWM clock period)].

Center-aligned operation versus edge-aligned operation is determined by the option EDGE. See **5.3 Functional Description**.

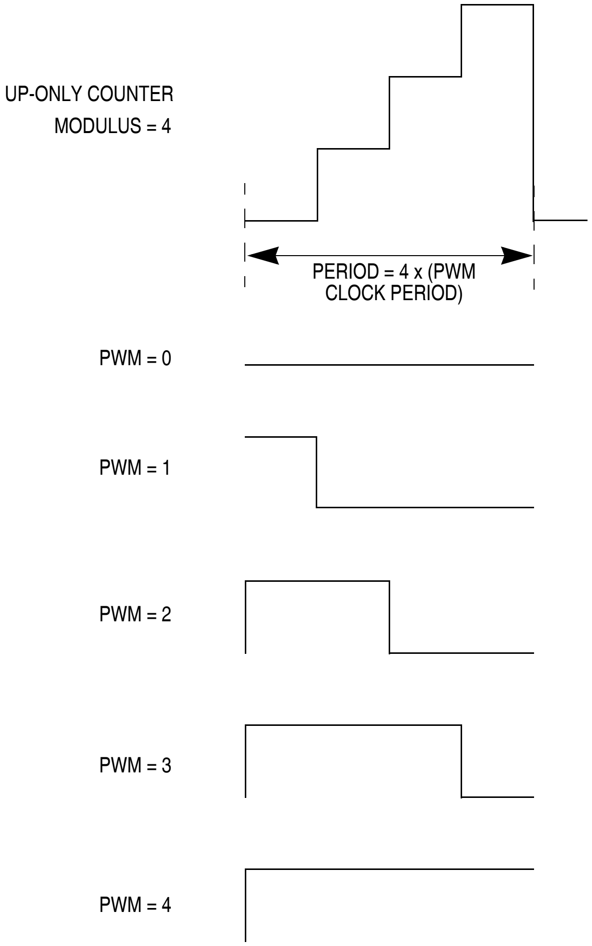


Figure 9-4. Edge-Aligned PWM (Positive Polarity)

9.4.2 Prescaler

To permit lower PWM frequencies, a prescaler is provided which will divide the PWM clock frequency by 1, 2, 4, or 8. **Table 9-1** shows how setting the prescaler bits in PWM control register 2 affects the PWM clock frequency. This prescaler is buffered and will not be used by the PWM generator until the LDOK bit is set and a new PWM reload cycle begins.

Table 9-1. PWM Prescaler

Prescaler Bits PRSC1 and PRSC0	PWM Clock Frequency
00	f_{op}
01	$f_{op}/2$
10	$f_{op}/4$
11	$f_{op}/8$

9.5 PWM Generators

Pulse-width modulator (PWM) generators are discussed in this subsection.

9.5.1 Load Operation

To help avoid erroneous pulse widths and PWM periods, the modulus, prescaler, and PWM value registers are buffered. New PWM values, counter modulus values, and prescalers can be loaded from their buffers into the PWM module every one, two, four, or eight PWM cycles. LDFQ1 and LDFQ0 in PWM control register 2 are used to control this reload frequency, as shown in **Table 9-2**. When a reload cycle arrives, regardless of whether an actual reload occurs (as determined by the LDOK bit), the PWM reload flag bit in PWM control register 1 will be set. If the PWMINT bit in PWM control register 1 is set, a CPU interrupt request will be generated when PWMF is set. Software can use this interrupt to calculate new PWM parameters in real time for the PWM module.

Table 9-2. PWM Reload Frequency

Reload Frequency Bits LDFQ1 and LDFQ0	PWM Reload Frequency
00	Every PWM cycle
01	Every 2 PWM cycles
10	Every 4 PWM cycles
11	Every 8 PWM cycles

For ease of software, the LDFQx bits are buffered. When the LDFQx bits are changed, the reload frequency will not change until the previous reload cycle is completed. See **Figure 9-5**.

NOTE: *When reading the LDFQx bits, the value is the buffered value (for example, not necessarily the value being acted upon).*

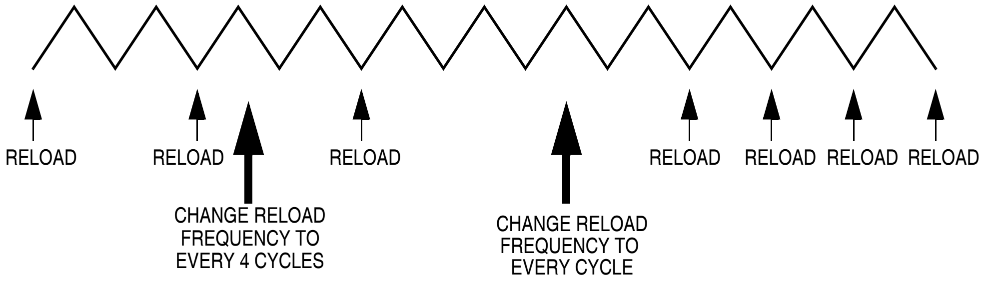


Figure 9-5. Reload Frequency Change

PWMINT enables CPU interrupt requests as shown in **Figure 9-6**. When this bit is set, CPU interrupt requests are generated when the PWMF bit is set. When the PWMINT bit is clear, PWM interrupt requests are inhibited. PWM reloads will still occur at the reload rate, but no interrupt requests will be generated.

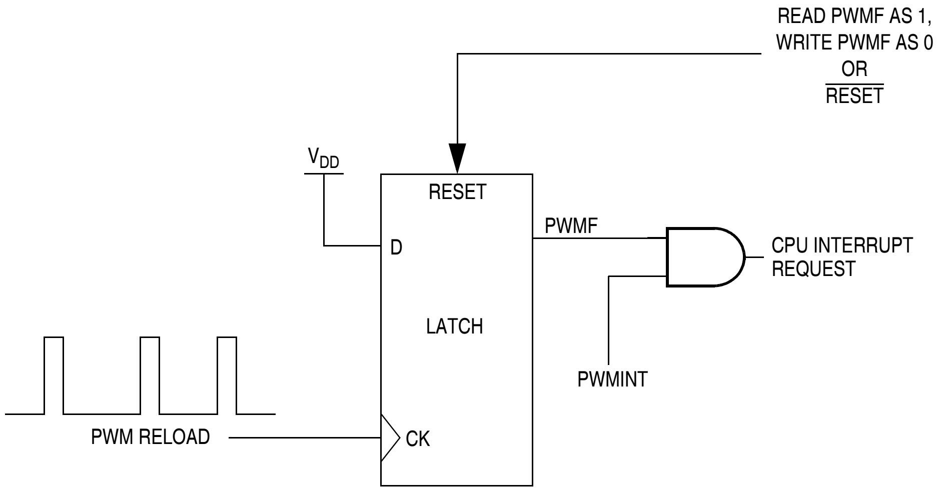


Figure 9-6. PWM Interrupt Requests

To prevent a partial reload of PWM parameters from occurring while the software is still calculating them, an interlock bit controlled from software is provided. This bit informs the PWM module that all the PWM parameters have been calculated, and it is “okay” to use them. A new modulus, prescaler, and/or PWM value cannot be loaded into the PWM module until the LDOK bit in PWM control register 1 is set. When the LDOK bit is set, these new values are loaded into a second set of registers and used by the PWM generator at the beginning of the next PWM reload cycle as shown in **Figure 9-7**, **Figure 9-8**, **Figure 9-9**, and **Figure 9-10**. After these values are loaded, the LDOK bit is cleared.

NOTE: *When the PWM module is enabled (via the PWMEN bit), a load will occur if the LDOK bit is set. Even if it is not set, an interrupt will occur if the PWMINT bit is set. To prevent this, the software should clear the PWMINT bit before enabling the PWM module.*

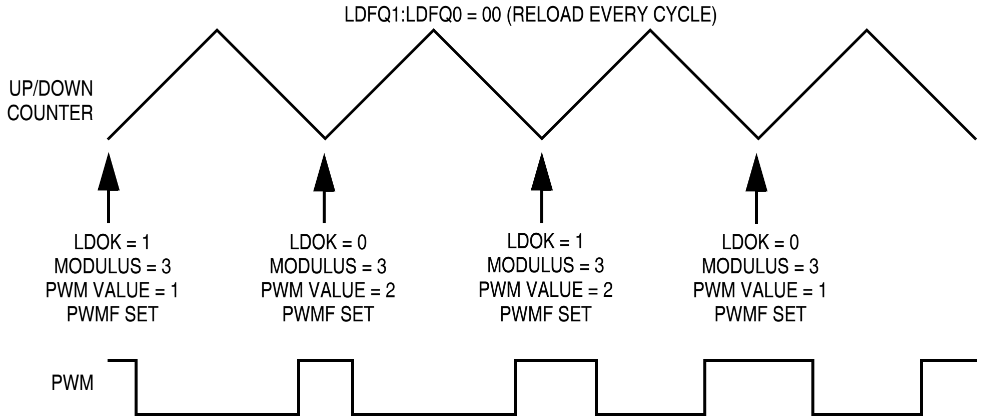


Figure 9-7. Center-Aligned PWM Value Loading

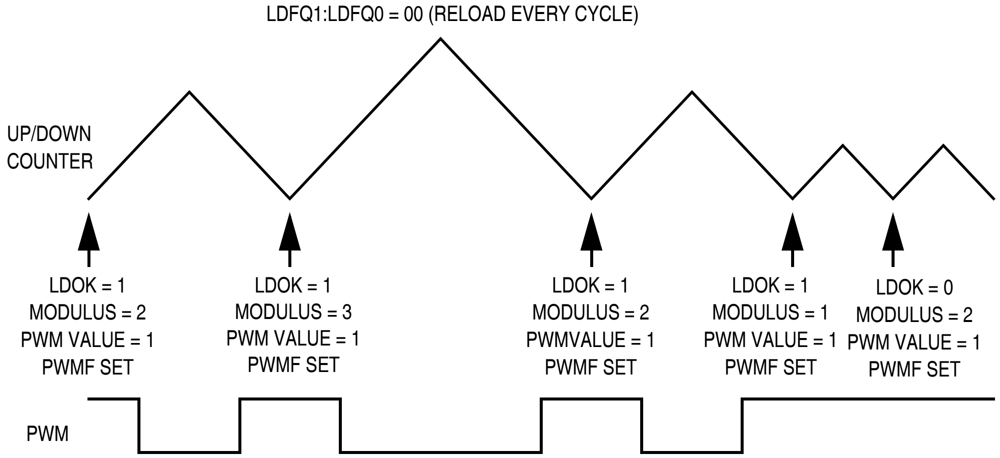


Figure 9-8. Center-Aligned Loading of Modulus

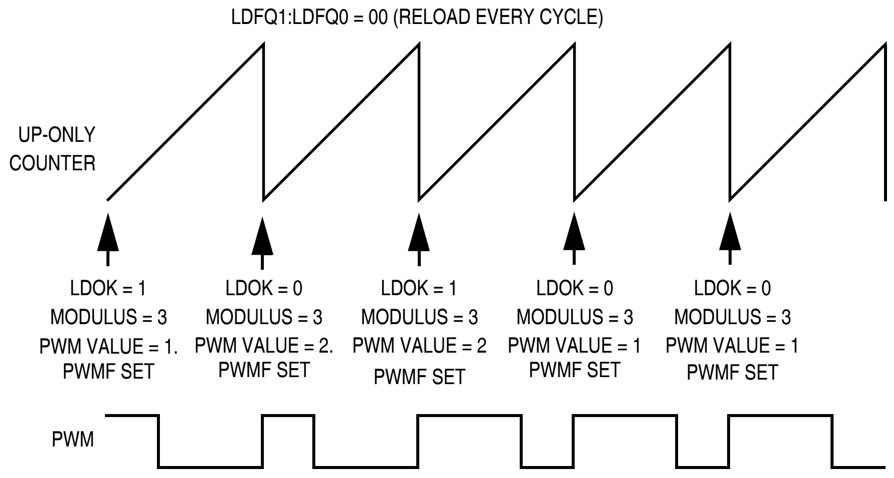


Figure 9-9. Edge-Aligned PWM Value Loading

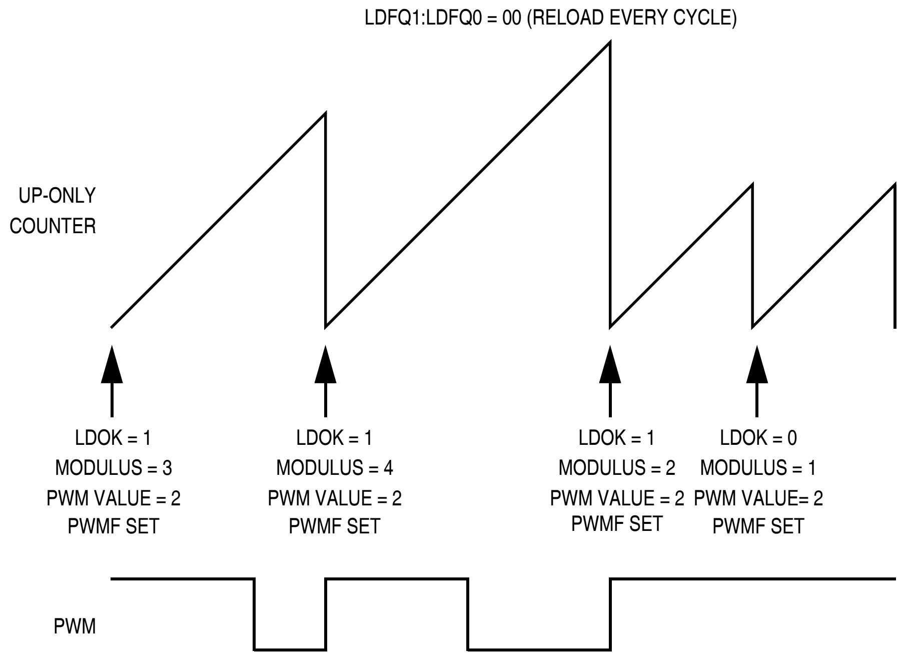


Figure 9-10. Edge-Aligned Modulus Loading

9.5.2 PWM Data Overflow and Underflow Conditions

The PWM value registers are 16-bit registers. Although the counter is only 12 bits, the user may write a 16-bit signed value to a PWM value register. As shown in **Figure 9-3** and **Figure 9-4**, if the PWM value is less than or equal to zero, the PWM will be inactive for the entire period. Conversely, if the PWM value is greater than or equal to the timer modulus, the PWM will be active for the entire period. Refer to **Table 9-3.**

NOTE: *The terms "active" and "inactive" refer to the asserted and negated states of the PWM signals and should not be confused with the high-impedance state of the PWM pins.*

Table 9-3. PWM Data Overflow and Underflow Conditions

PWMVALxH:PWMVALxL	Condition	PWM Value Used
\$0000–\$0FFF	Normal	Per register contents
\$1000–\$7FFF	Overflow	\$FFF
\$8000–\$FFFF	Underflow	\$000

9.6 Output Control

This subsection discusses output control.

9.6.1 Selecting Six Independent PWMs or Three Complementary PWM Pairs

The PWM outputs can be configured as six independent PWM channels or three complementary channel pairs. The option INDEP determines which mode is used (see **5.3 Functional Description**). If complementary operation is chosen, the PWM pins are paired as shown in **Figure 9-11**. Operation of one pair is then determined by one PWM value register. This type of operation is meant for use in motor drive circuits such as the one in **Figure 9-12**.

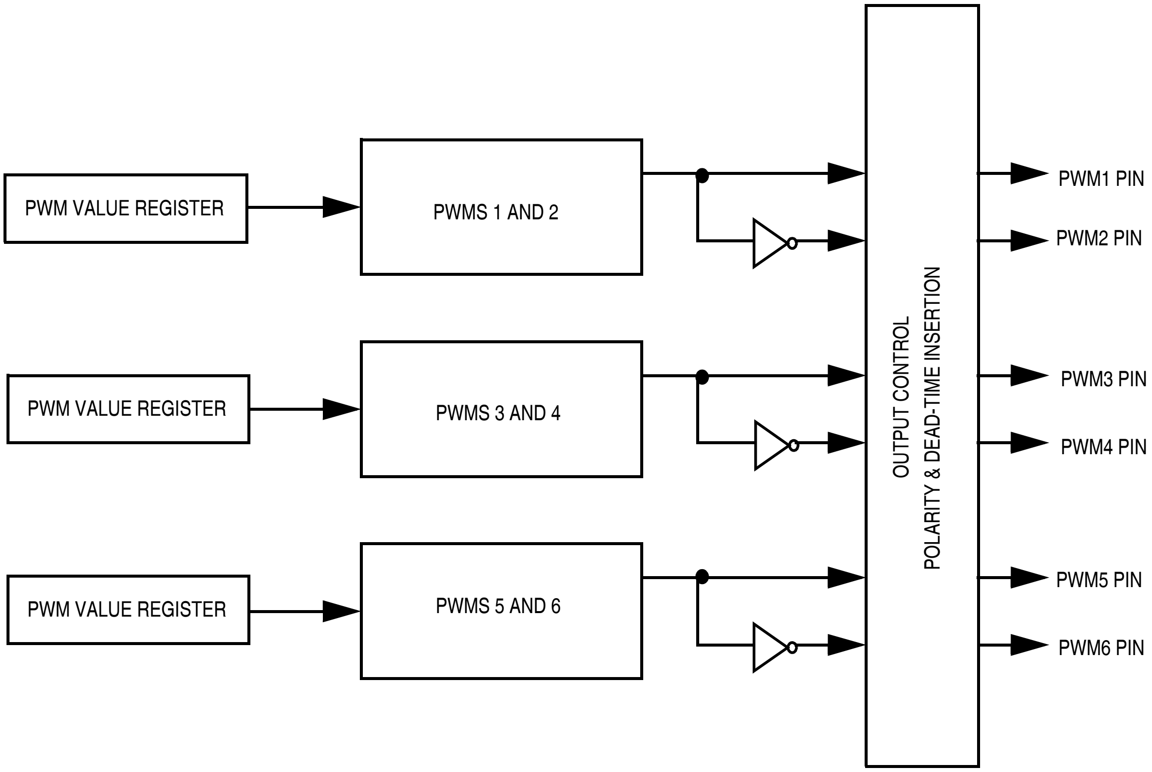


Figure 9-11. Complementary Pairing

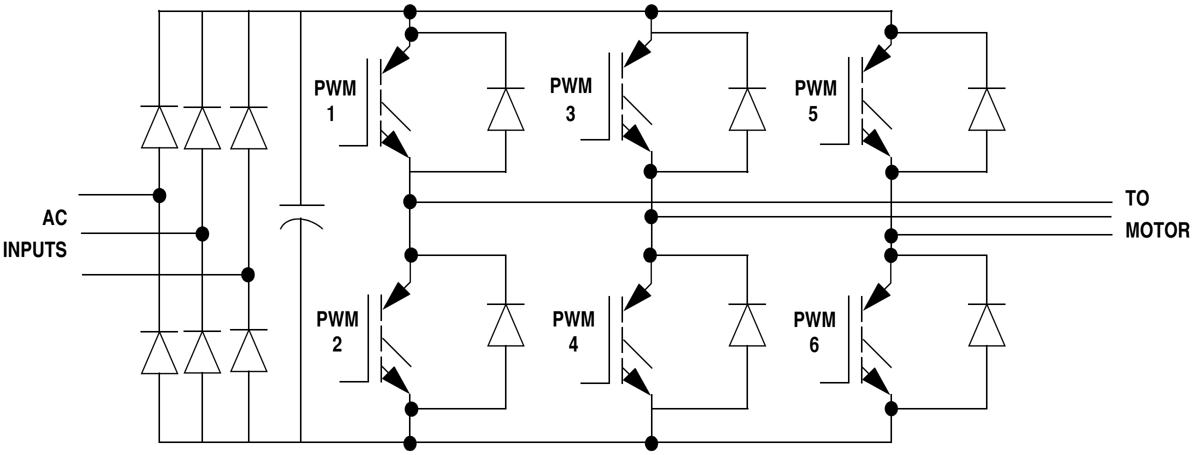


Figure 9-12. Typical AC Motor Drive

When complementary operation is used, two additional features are provided:

- Dead-time insertion
- Separate top/bottom pulse width correction to correct for distortions caused by the motor drive characteristics

If independent operation is chosen, each PWM has its own PWM value register.

9.6.2 Dead-Time Insertion

As shown in **Figure 9-12**, in complementary mode, each PWM pair can be used to drive top-side/bottom-side transistors.

When controlling DC-to-AC inverters such as this, the top and bottom PWMs in one pair should ***never*** be active at the same time. In **Figure 9-12**, if PWM1 and PWM2 were on at the same time, large currents would flow through the two transistors as they discharge the bus capacitor. The IGBTs could be weakened or destroyed.

Simply forcing the two PWMs to be inversions of each other is not always sufficient. Since a time delay is associated with turning off the transistors in the motor drive, there must be a dead-time between the deactivation of one PWM and the activation of the other.

A dead-time can be specified in the dead-time write-once register. This 8-bit value specifies the number of CPU clock cycles to use for the dead-time. The dead-time is not affected by changes in the PWM period caused by the prescaler.

Dead-time insertion is achieved by feeding the top PWM outputs of the PWM generator into dead-time generators, as shown in **Figure 9-13**. Current sensing determines which PWM value of a PWM generator pair to use for the top PWM in the next PWM cycle. See **9.6.3 Top/Bottom Correction with Motor Phase Current Polarity Sensing**. When output control is enabled, the odd OUT bits, rather than the PWM generator outputs, are fed into the dead-time generators. See **9.6.5 PWM Output Port Control.**

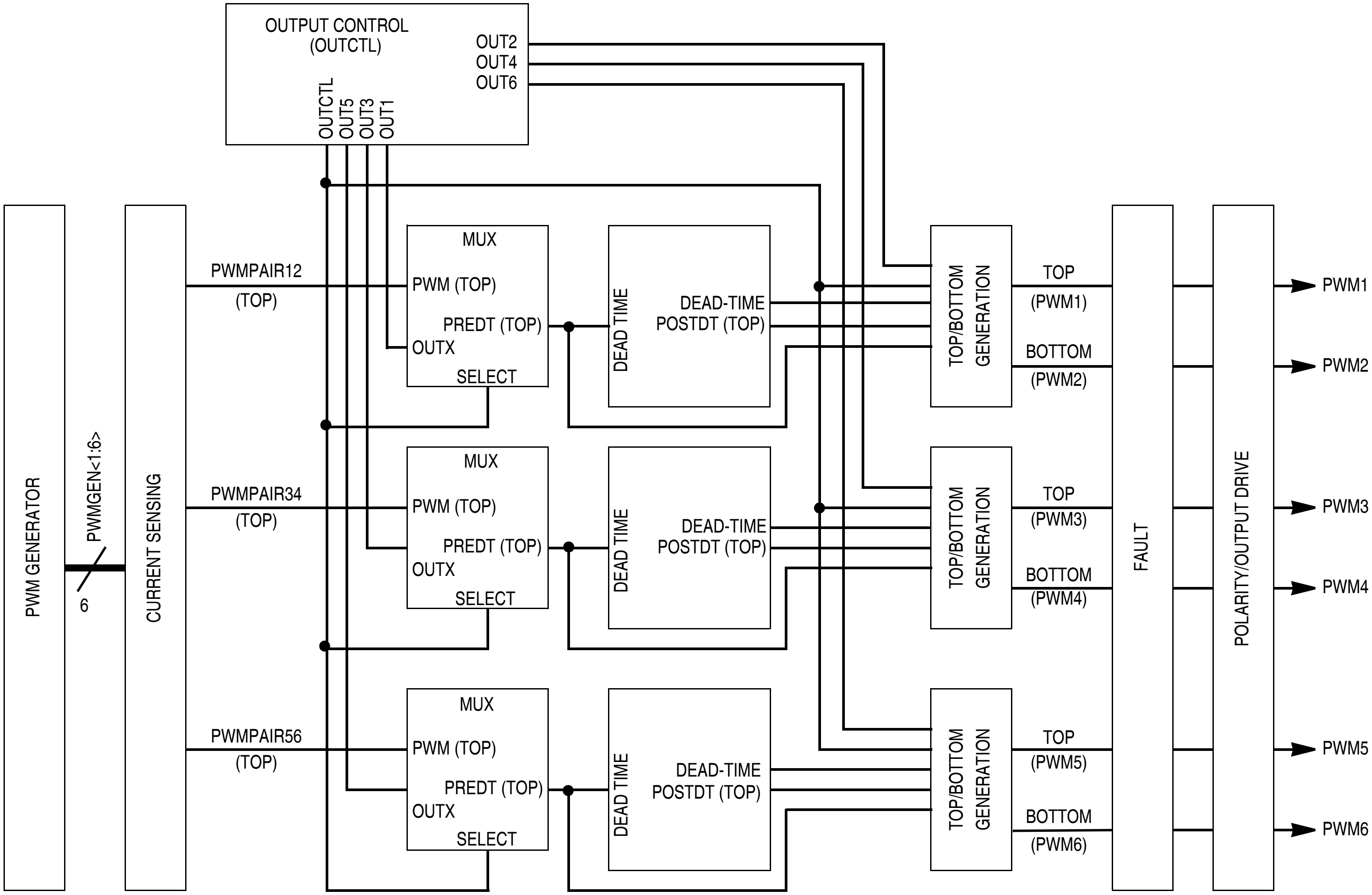


Figure 9-13. Dead-Time Generators

Whenever an input to a dead-time generator transitions, a dead-time is inserted (for example, both PWMs in the pair are forced to their inactive state). The bottom PWM signal is generated from the top PWM and the dead-time. In the case of output control enabled, the odd OUTx bits control the top PWMs, the even OUTx bits control the bottom PWMs *with respect to the odd OUTx bits* (see **Table 9-6**). **Figure 9-14** shows the effects of the dead-time insertion.

As seen in **Figure 9-14**, some pulse width distortion occurs when the dead-time is inserted. The active pulse widths are reduced. For example, in **Figure 9-14**, when the PWM value register is equal to two, the ideal waveform (with no dead-time) has pulse widths equal to four. However, the actual pulse widths shrink to two after a dead-time of two was inserted. In this example, with the prescaler set to divide by one and center-aligned operation selected, this distortion can be compensated for by adding or subtracting half the dead-time value to or from the PWM register value. This correction is further described in **9.6.3 Top/Bottom Correction with Motor Phase Current Polarity Sensing**.

Further examples of dead-time insertion are shown in **Figure 9-15** and **Figure 9-16**. **Figure 9-15** shows the effects of dead-time insertion at the duty cycle boundaries (near 0 percent and 100 percent duty cycles). **Figure 9-16** shows the effects of dead-time insertion on pulse widths smaller than the dead time.

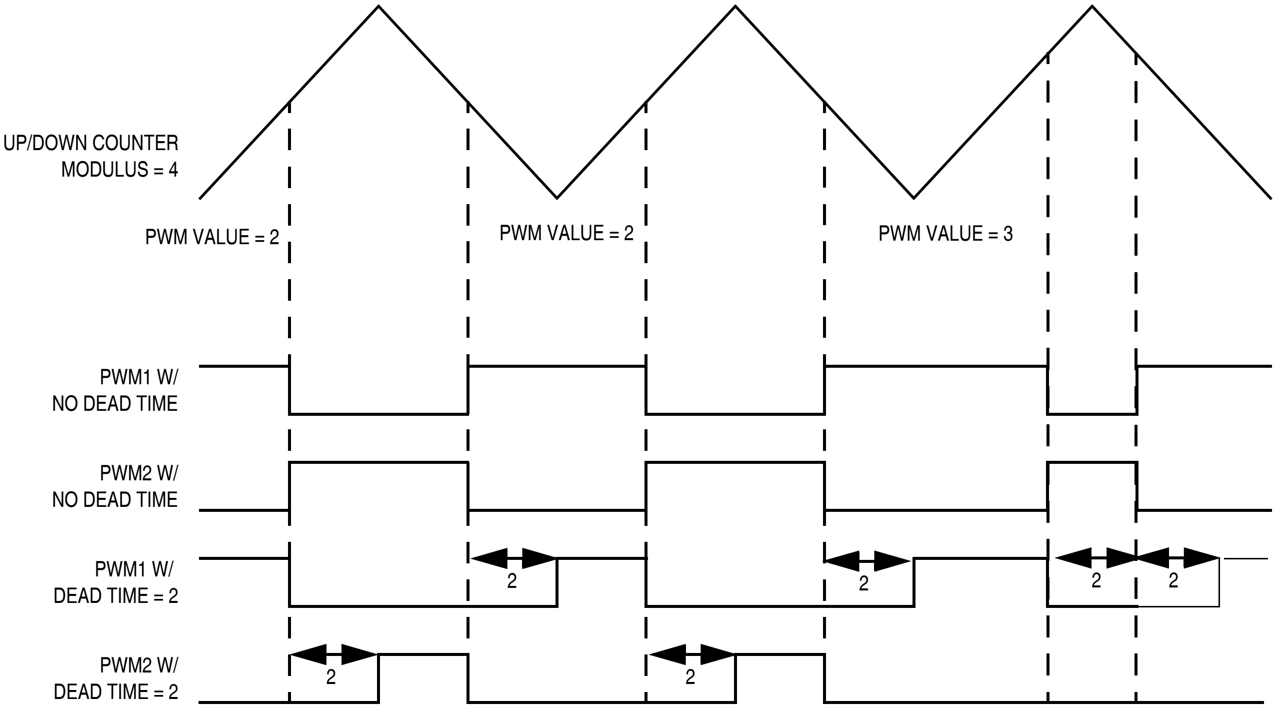


Figure 9-14. Effects of Dead-Time Insertion

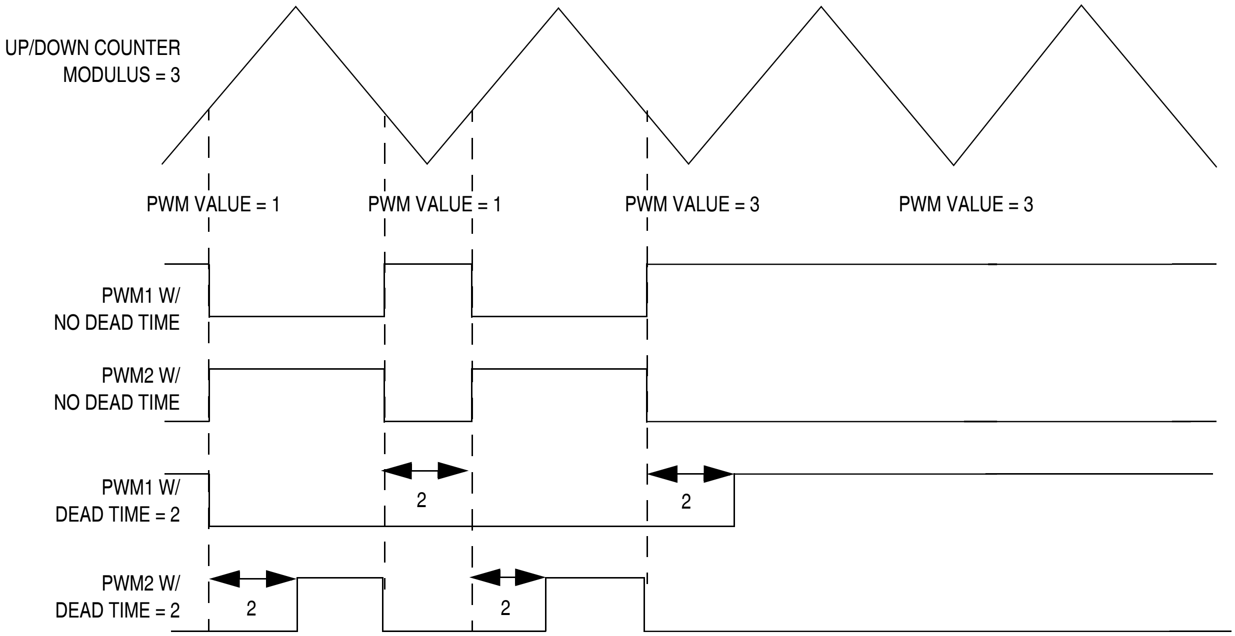


Figure 9-15. Dead-Time at Duty Cycle Boundaries

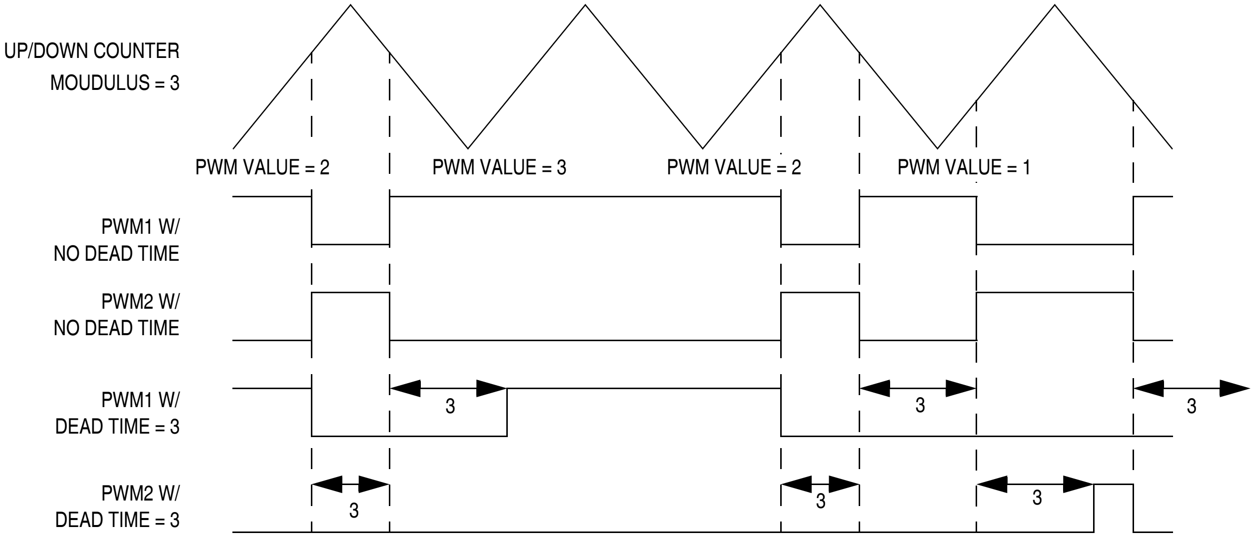


Figure 9-16. Dead Time and Small Pulse Widths

9.6.3 Top/Bottom Correction with Motor Phase Current Polarity Sensing

Ideally, when complementary pairs are used, the PWM pairs are inversions of each other, as shown in **Figure 9-17**. When PWM1 is active, PWM2 is inactive, and vice versa. In this case, the motor terminal voltage is never allowed to float and is strictly controlled by the PWM waveforms.

However, when dead time is inserted, the motor voltage is allowed to float momentarily during the dead-time interval, creating a distortion in the motor current waveform. This distortion is aggravated by dissimilar turn-on and turn-off delays of each of the transistors.

For a typical motor drive inverter as shown in **Figure 9-12**, for a given top/bottom transistor pair, only one of the transistors will be effective in controlling the output voltage at any given time depending on the direction of the motor current for that pair. To achieve distortion correction, one of two different correction factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values and placing them in an odd/even PWM register pair. By supplying the PWM module with information regarding which transistor (top or bottom) is controlling the output voltage at any given time (for instance, the current polarity for that motor phase), the PWM module selects either the odd or even numbered PWM value register to be used by the PWM generator.

Current sensing or programmable software bits are then used to determine which PWM value to use. If the current sensed at the motor for that PWM pair is positive (voltage on current pin ISx is low) or bit IPOLx in PWM control register 2 is low, the top PWM value is used for the PWM pair. Likewise, if the current sensed at the motor for that PWM pair is negative (voltage on current pin ISx is high) or bit IPOLx in PWM control register 2 is high, the bottom PWM value is used. See **Table 9-4**.

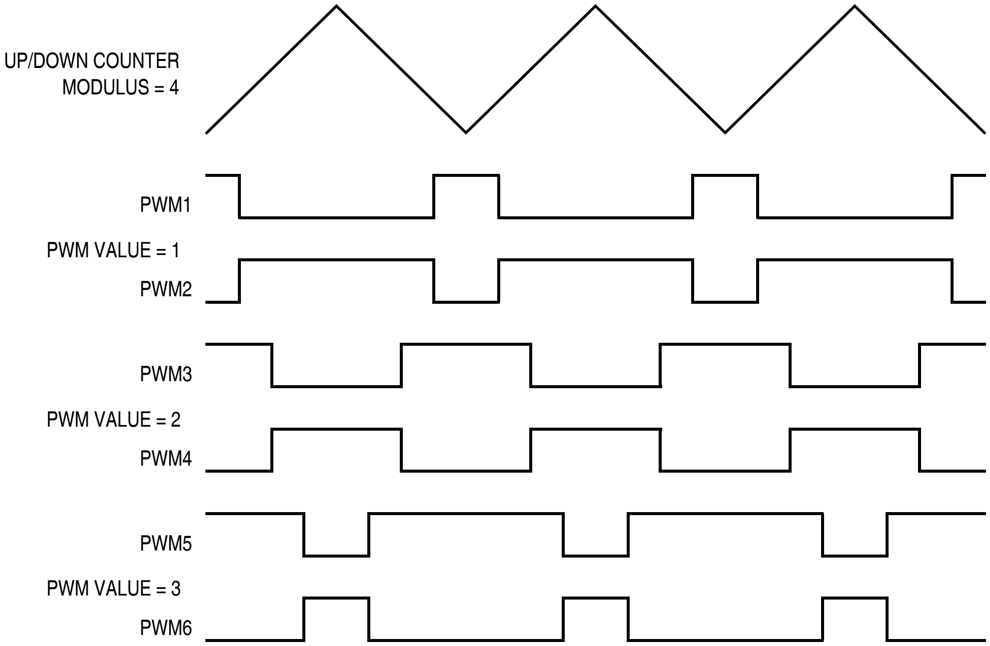


Figure 9-17. Ideal Complementary Operation (Dead Time = 0)

NOTE: *This text assumes the user will provide current sense circuitry which causes the voltage at the corresponding input pin to be low for positive current and high for negative current. See* ***Figure 9-18*** *for current convention. In addition, it assumes the top PWMs are PWMs 1, 3, and 5 while the bottom PWMs are PWMs 2, 4, and 6.*

Table 9-4. Current Sense Pins

Current Sense Pin or Bit	Voltage on Current Sense Pin or IPOLx Bit	PWM Value Register Used	PWMs Affected
$\overline{\text{IS1}}$ or IPOL1	Logic 0	PWM value register 1	PWMs 1 and 2
$\overline{\text{IS1}}$ or IPOL1	Logic 1	PWM value register 2	PWMs 1 and 2
$\overline{\text{IS2}}$ or IPOL2	Logic 0	PWM value register 3	PWMs 3 and 4
$\overline{\text{IS2}}$ or IPOL2	Logic 1	PWM value register 4	PWMs 3 and 4
$\overline{\text{IS3}}$ or IPOL3	Logic 0	PWM value register 5	PWMs 5 and 6
$\overline{\text{IS3}}$ or IPOL3	Logic 1	PWM value register 6	PWMs 5 and 6

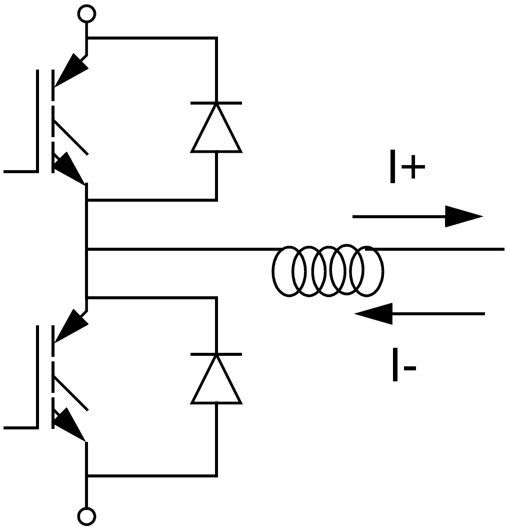


Figure 9-18. Current Convention

To allow for correction based on different current sensing methods or correction controlled by software, the ISENS1 and ISENS0 bits in PWM control register 1 are provided to choose the correction method. These bits provide correction according to **Table 9-5**.

Table 9-5. Correction Methods

Current Correction Bits ISENS1 and ISENS0	Correction Method
00 01	Bits IPOL1, IPOL2, and IPOL3 used for correction
10	Current sensing on pins $\overline{\text{IS1}}$, $\overline{\text{IS2}}$, and $\overline{\text{IS3}}$ occurs during the dead time.
11	Current sensing on pins $\overline{\text{IS1}}$, $\overline{\text{IS2}}$, and $\overline{\text{IS3}}$ occurs at the half cycle in center-aligned mode and at the end of the cycle in edge-aligned mode.

If correction is to be done in software or is not necessary, setting ISENS1:ISENS0 = 00 or = 01 causes the correction to be based on bits IPOL1, IPOL2, and IPOL3 in PWM control register 2. If correction is not required, the user can initialize the IPOLx bits and then only load one PWM value register per PWM pair.

To allow the user to use a current sense scheme based upon sensed phase voltage during dead time, setting ISENS1:ISENS0 = 10 causes the polarity of the Ix pin to be latched when both the top and bottom PWMs are off (for example, during the dead time). At the 0 percent and 100 percent duty cycle boundaries, there is no dead time so no new current value is sensed.

To accommodate other current sensing schemes, setting ISENS1:ISENS0 = 11 causes the polarity of the current sense pin to be latched half-way into the PWM cycle in center-aligned mode and at the end of the cycle in edge-aligned mode. Therefore, even at 0 percent and 100 percent duty cycle, the current is sensed.

Distortion correction is only available in complementary mode. At the beginning of the PWM period, the PWM uses this latched current value or polarity bit to decide whether the top PWM value or bottom PWM value is used. **Figure 9-19** shows an example of top/bottom correction for PWMs 1 and 2.

NOTE: *The IPOLx bits and the values latched on the ISx pins are buffered so that only one PWM register is used per PWM cycle. If the IPOLx bits or the current sense values change during a PWM period, this new value will not be used until the next PWM period. The ISENSx bits are NOT buffered; therefore, changing the current sensing method could affect the present PWM cycle.*

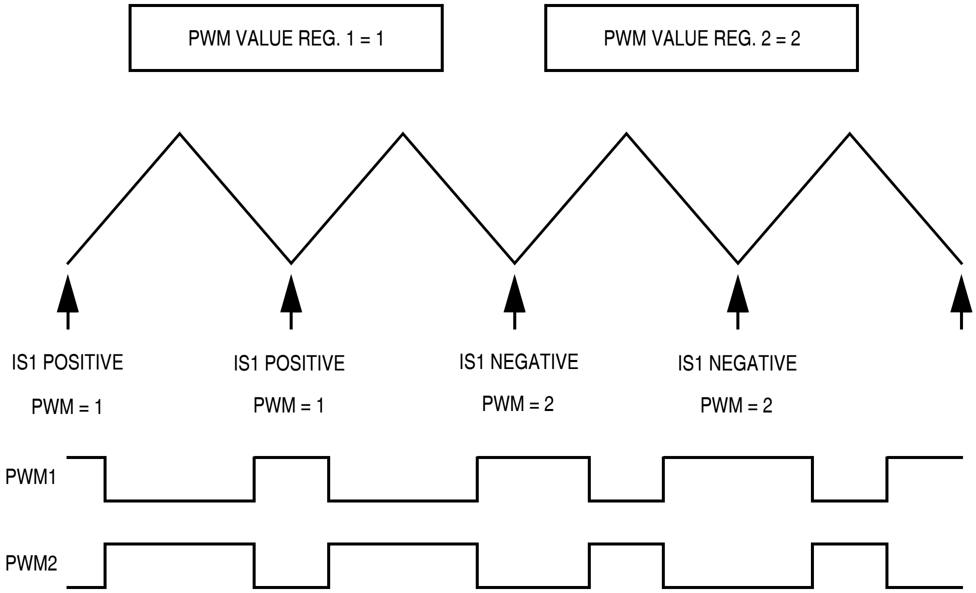


Figure 9-19. Top/Bottom Correction for PWMs 1 and 2

When the PWM is first enabled by setting PWMEN, PWM value registers 1, 3, and 5 will be used if the ISENSx bits are configured for current sensing correction. This is because no current will have previously been sensed.

9.6.4 Output Polarity

The output polarity of the PWMs is determined by two options: TOPNEG and BOTNEG. The top polarity option, TOPNEG, controls the polarity of PWMs 1, 3, and 5. The bottom polarity option, BOTNEG, controls the polarity of PWMs 2, 4, and 6. Positive polarity means that when the PWM is active, the PWM output is high. Conversely, negative polarity means that when the PWM is active, PWM output is low. See **Figure 9-20**.

NOTE: *Both bits are found in the CONFIG register, which is a write-once register. This reduces the chances of the software inadvertently changing the polarity of the PWM signals and possibly damaging the motor drive hardware.*

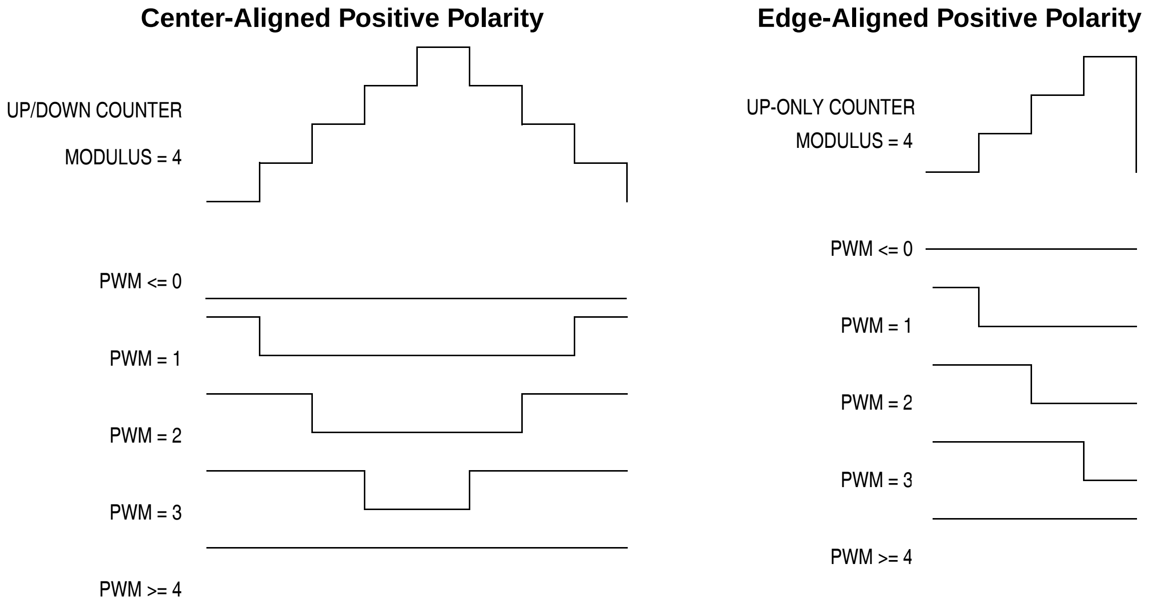


Figure 9-20. PWM Polarity

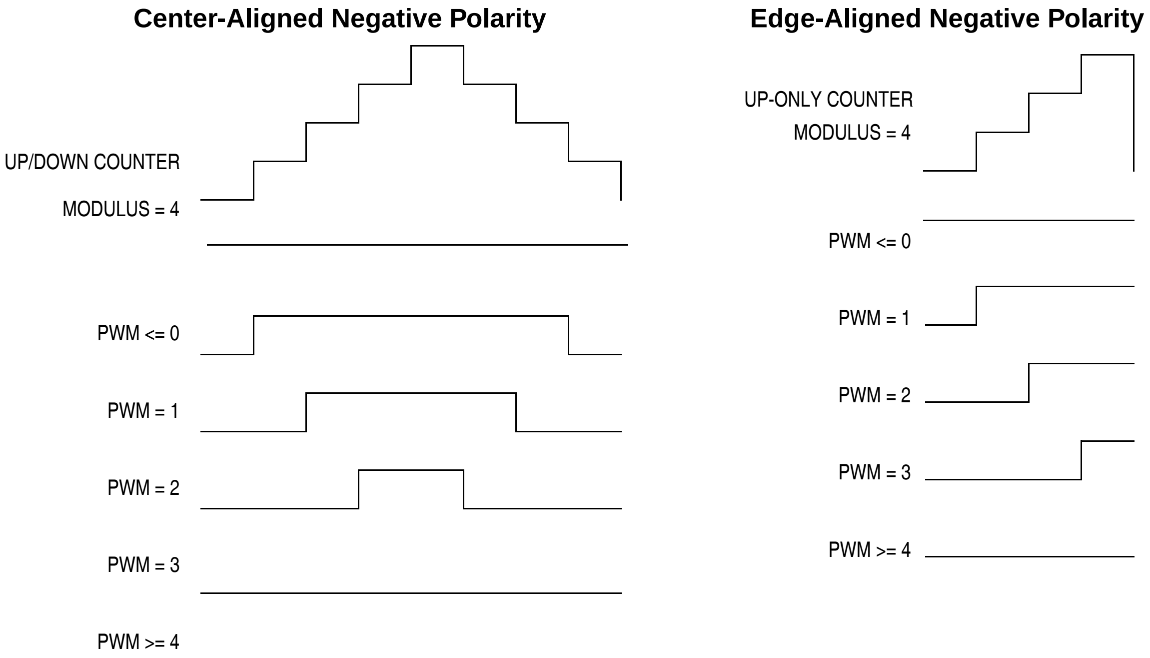


Figure 9-19. PWM Polarity (Continued)

9.6.5 PWM Output Port Control

Conditions may arise in which the PWM pins need to be individually controlled. This is made possible by the PWM output control register (PWMOUT) shown in **Figure 9-21**.

Address: $0025

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	OUTCTL	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1
Write:	▨							
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-21. PWM Output Control Register (PWMOUT)

If the OUTCTL bit is set, the PWM pins can be controlled by the OUTx bits. These bits behave according to **Table 9-6**.

Table 9-6. OUTx Bits

OUTx Bit	Complementary Mode	Independent Mode
OUT1	1 — PWM1 is active. 0 — PWM1 is inactive.	1 — PWM1 is active. 0 — PWM1 is inactive.
OUT2	1 — PWM2 is complement of PWM 1. 0 — PWM2 is inactive.	1 — PWM2 is active. 0 — PWM2 is inactive.
OUT3	1 — PWM3 is active. 0 — PWM3 is inactive.	1 — PWM3 is active. 0 — PWM3 is inactive.
OUT4	1 — PWM4 is complement of PWM 3. 0 — PWM4 is inactive.	1 — PWM4 is active. 0 — PWM4 is inactive.
OUT5	1 — PWM5 is active. 0 — PWM5 is inactive.	1 — PWM5 is active. 0 — PWM5 is inactive.
OUT6	1 — PWM 6 is complement of PWM 5. 0 — PWM6 is inactive.	1 — PWM6 is active. 0 — PWM6 is inactive.

When OUTCTL is set, the polarity options TOPPOL and BOTPOL will still affect the outputs. In addition, if complementary operation is in use, the PWM pairs will not be allowed to be active simultaneously, and dead time will still not be violated. When OUTCTL is set and complimentary operation is in use, the odd OUTx bits are inputs to the dead-time generators as shown in **Figure 9-14**. Dead time is inserted whenever the odd OUTx bit toggles as shown in **Figure 9-22**. Although dead time is not inserted when the even OUTx bits change, there will be no dead-time violation as shown in **Figure 9-23**.

Setting the OUTCTL bit does not disable the PWM generator and current sensing circuitry. They continue to run, but are no longer controlling the output pins. In addition, OUTCTL will control the PWM pins even when PWMEN = 0. When OUTCTL is cleared, the outputs of the PWM generator become the inputs to the dead-time and output circuitry at the beginning of the next PWM cycle.

NOTE: *To avoid an unexpected dead-time occurrence, it is recommended that the OUTx bits be cleared prior to entering and prior to exiting individual PWM output control mode.*

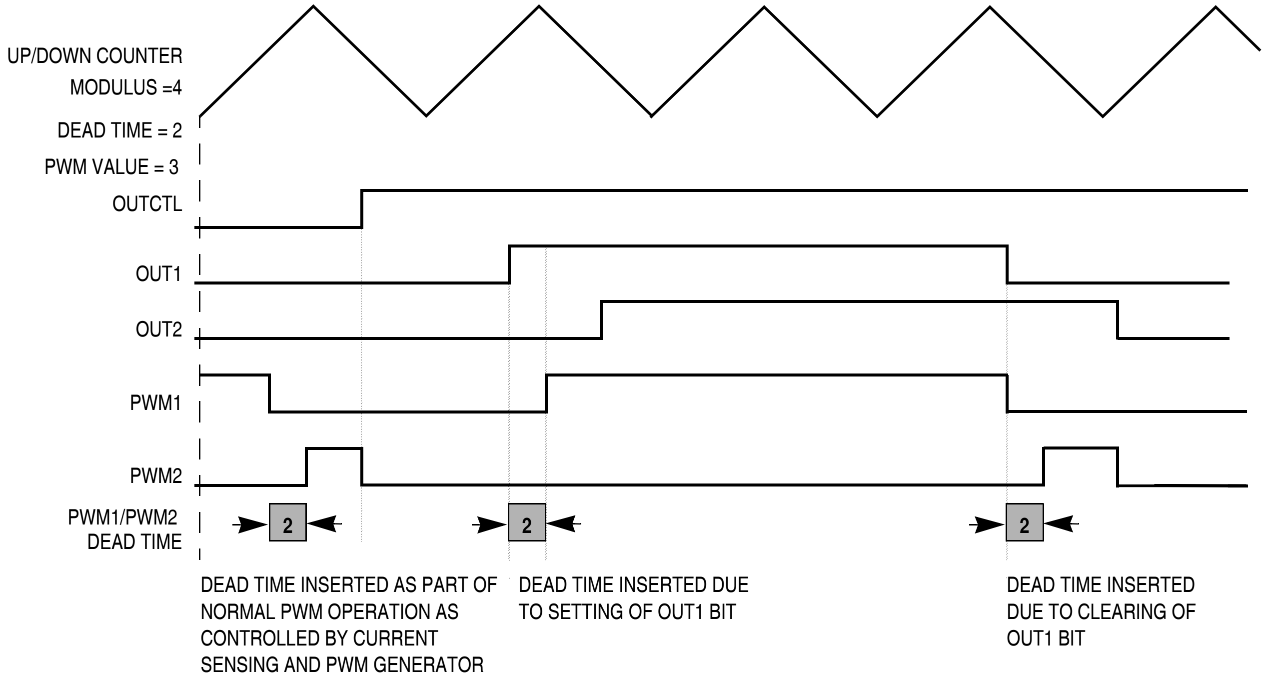


Figure 9-22. Dead-Time Insertion During OUTCTL = 1

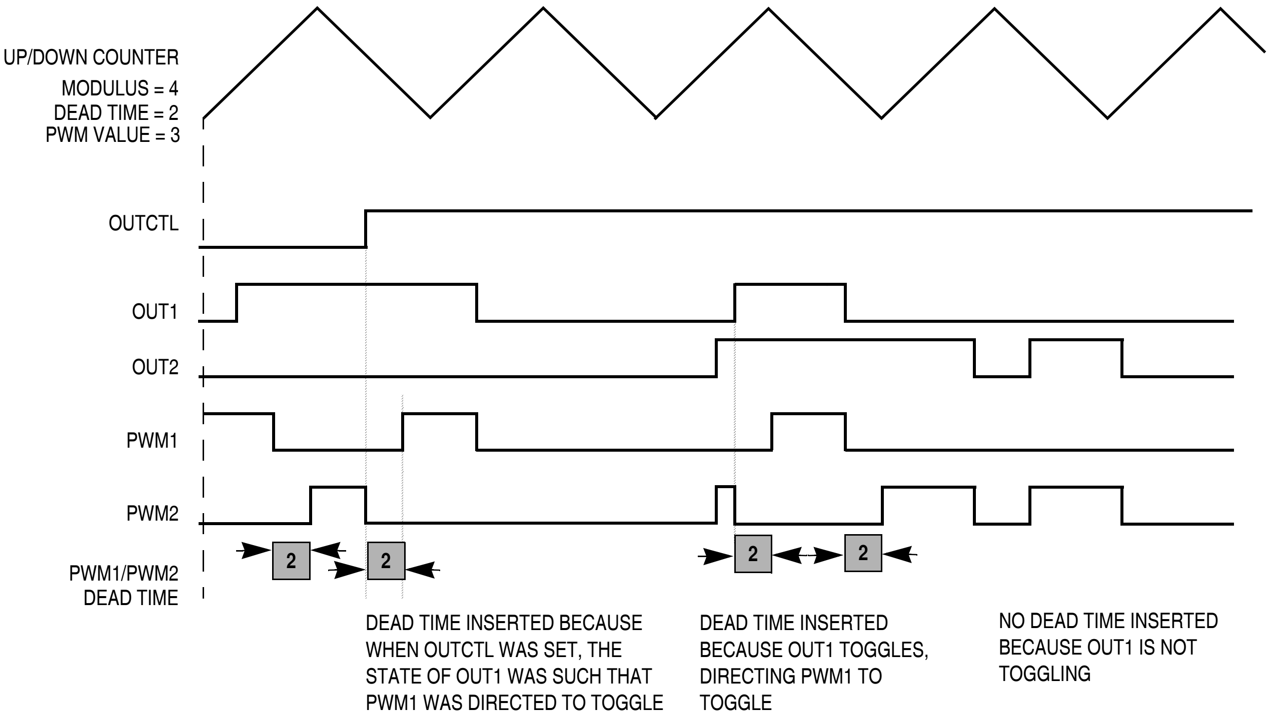


Figure 9-23. Dead-Time Insertion During OUTCTL = 1

9.7 Fault Protection

Conditions may arise in the external drive circuitry which require that the PWM signals become inactive immediately, such as an overcurrent fault condition. Furthermore, it may be desirable to selectively disable PWM(s) solely with software.

One or more PWM pins can be disabled (forced to their inactive state) by applying a logic high to any of the four external fault pins or by writing a logic high to either of the disable bits (DISX and DISY in PWM control register 1). **Figure 9-25** shows the structure of the PWM disabling scheme. While the PWM pins are disabled, they are forced to their inactive state. The PWM generator continues to run — only the output pins are disabled.

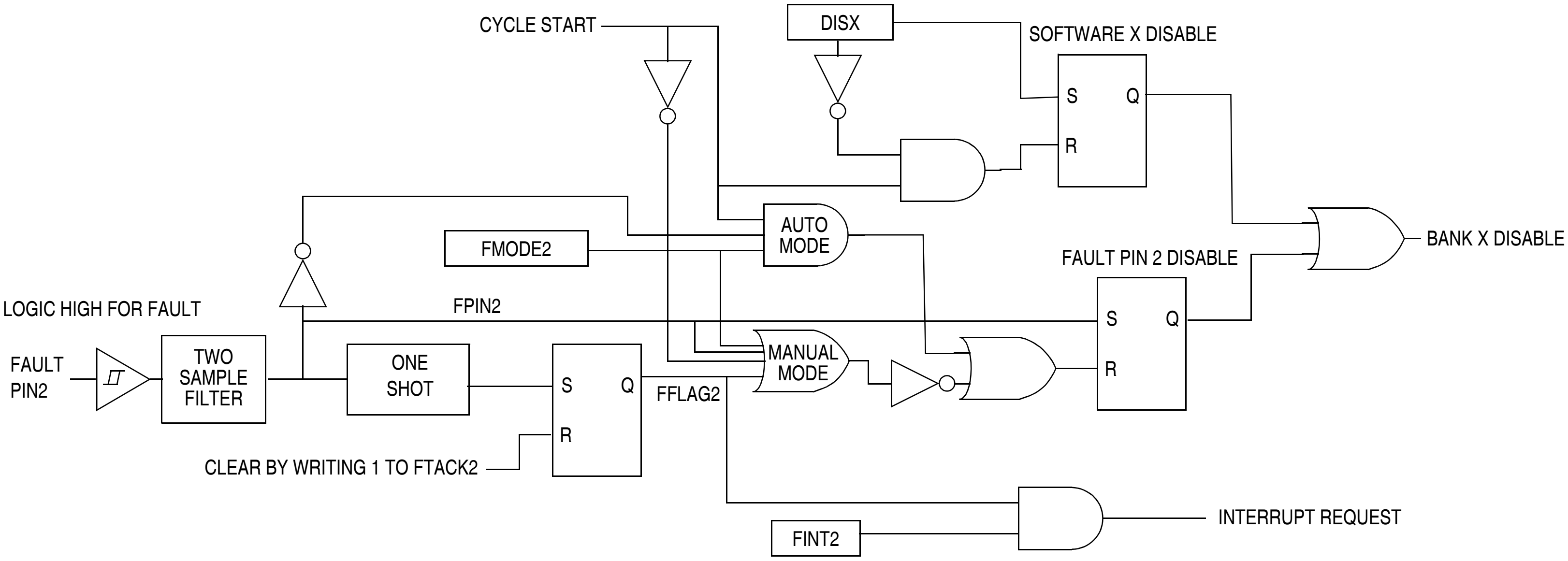
To allow for different motor configurations and the controlling of more than one motor, the PWM disabling function is organized as two banks, bank X and bank Y. Bank information combines with information from the disable mapping register to allow selective PWM disabling. Fault pin 1, fault pin 2, and PWM disable bit X constitute the disabling function of bank X. Fault pin 3, fault pin 4, and PWM disable bit Y constitute the disabling function of bank Y. **Figure 9-24** and **Figure 9-26** show the disable mapping write-once register and the decoding scheme of the bank which selectively disables PWM(s). When all bits of the disable mapping register are set, any disable condition will disable all PWMs.

A fault can also generate a CPU interrupt. Each fault pin has its own interrupt vector.

Address: $0037

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

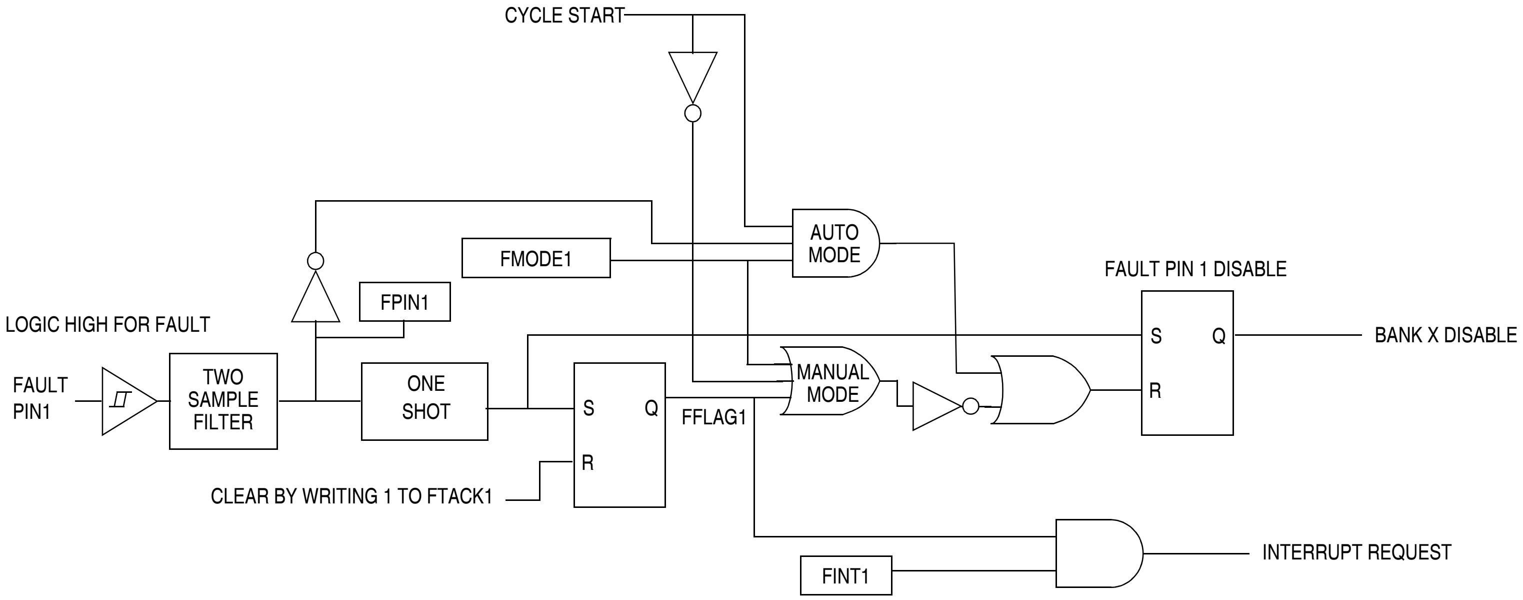
Figure 9-24. PWM Disable Mapping Write-Once Register (DISMAP)



The example is of fault pin 2 with DISX. Fault pin 4 with DISY is logically similar and affects BANK Y disable.

Note: In manual mode (FMODE = 0), faults 2 and 4 may be cleared only if a logic level low at the input of the fault pin is present.

Figure 9-25. PWM Disabling Scheme (Sheet 1 of 2)



The example is of fault pin 1. Fault pin 3 is logically similar and affects BANK Y disable.

Note: In manual mode (FMODE = 0), faults 1 and 3 may be cleared regardless of the logic level at the input of the fault pin.

Figure 9-25. PWM Disabling Scheme (Sheet 2 of 2)

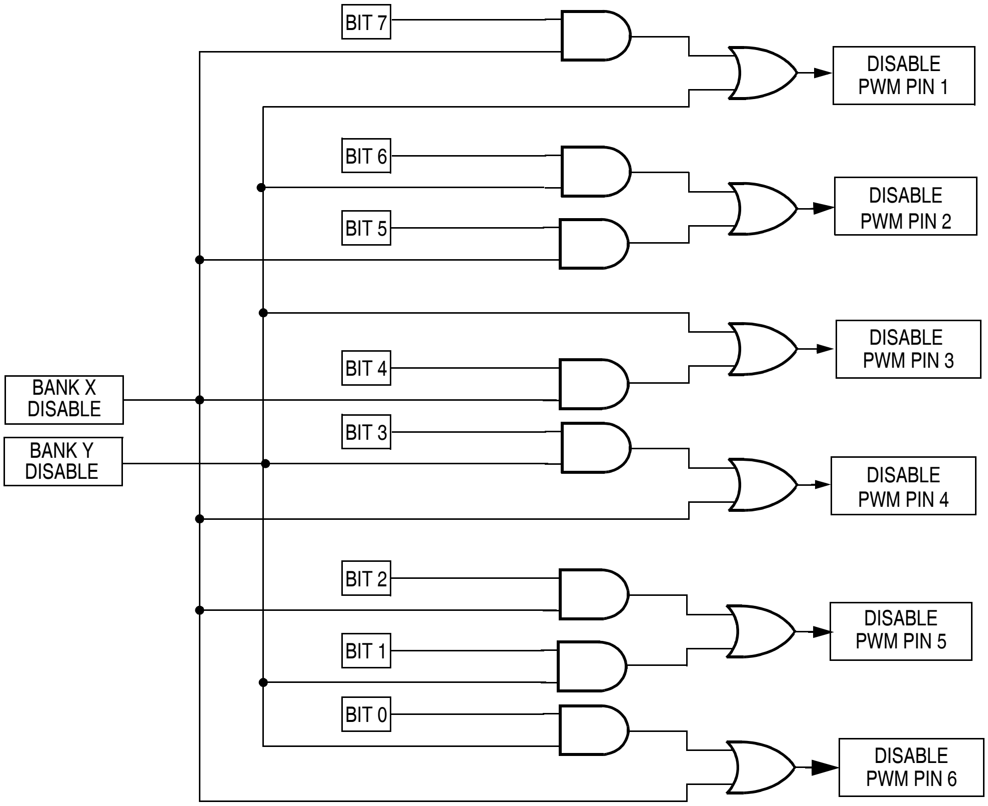


Figure 9-26. PWM Disabling Decode Scheme

9.7.1 Fault Condition Input Pins

A logic high level on a fault pin disables the respective PWM(s) determined by the bank and the disable mapping register. Each fault pin incorporates a filter to assist in rejecting spurious faults. All of the external fault pins are software-configurable to re-enable the PWMs either with the fault pin (automatic mode) or with software (manual mode). Each fault pin has an associated FMODE bit to control the PWM re-enabling method. Automatic mode is selected by setting the FMODEx bit in the fault control register. Manual mode is selected when FMODEx is clear.

9.7.1.1 Fault Pin Filter

Each fault pin incorporates a filter to assist in determining a genuine fault condition. After a fault pin has been logic low for one CPU cycle, a rising edge (logic high) will be synchronously sampled once per CPU cycle for two cycles. If both samples are detected logic high, the corresponding FPIN bit and FFLAG bit will be set. The FPIN bit will remain set until the corresponding fault pin is logic low and synchronously sampled once in the following CPU cycle.

9.7.1.2 Automatic Mode

In automatic mode, the PWM(s) are disabled immediately once a filtered fault condition is detected (logic high). The PWM(s) remain disabled until the filtered fault condition is cleared (logic low) and a new PWM cycle begins as shown in **Figure 9-27**. Clearing the corresponding FFLAGx event bit will not enable the PWMs in automatic mode.

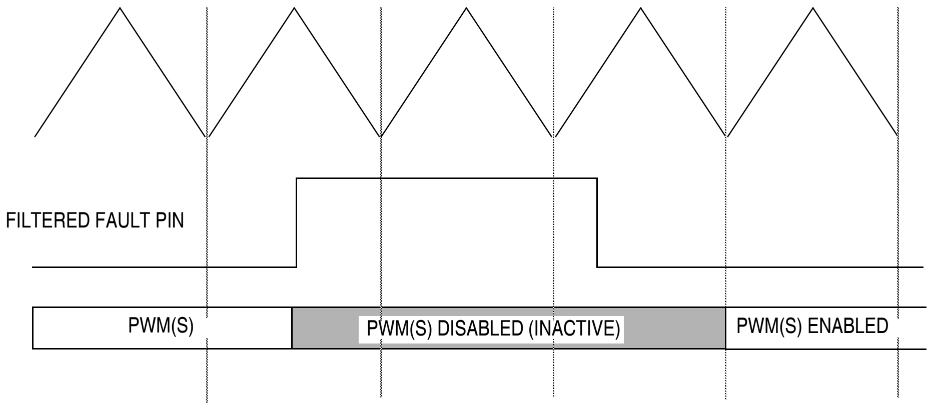


Figure 9-27. PWM Disabling in Automatic Mode

The filtered fault pin's logic state is reflected in the respective FPINx bit. Any write to this bit is overwritten by the pin state. The FFLAGx event bit is set with each rising edge of the respective fault pin after filtering has been applied. To clear the FFLAGx bit, the user must write a 1 to the corresponding FTACKx bit.

If the FINTx bit is set, a fault condition resulting in setting the corresponding FFLAG bit will also latch a CPU interrupt request. The interrupt request latch is not cleared until one of the following actions occurs:

- The FFLAGx bit is cleared by writing a 1 to the corresponding FTACKx bit.
- The FINTx bit is cleared. This will not clear the FFLAGx bit.
- A reset automatically clears all four interrupt latches.

If prior to a vector fetch, the interrupt request latch is cleared by one of the above actions, a CPU interrupt will no longer be requested. A vector fetch does not alter the state of the PWMs, the FFLAGx event flag, or FINTx.

NOTE: *If the FFLAGx or FINTx bits are not cleared during the interrupt service routine, the interrupt request latch will not be cleared.*

9.7.1.3 Manual Mode

In manual mode, the PWM(s) are disabled immediately once a filtered fault condition is detected (logic high). The PWM(s) remain disabled until software clears the corresponding FFLAGx event bit and a new PWM cycle begins. In manual mode, the fault pins are grouped in pairs, each pair sharing common functionality. A fault condition on pins 1 and 3 may be cleared, allowing the PWM(s) to enable at the start of a PWM cycle regardless of the logic level at the fault pin. See **Figure 9-28**. A fault condition on pins 2 and 4 can only be cleared, allowing the PWM(s) to enable, if a logic low level at the fault pin is present at the start of a PWM cycle. See **Figure 9-29**.

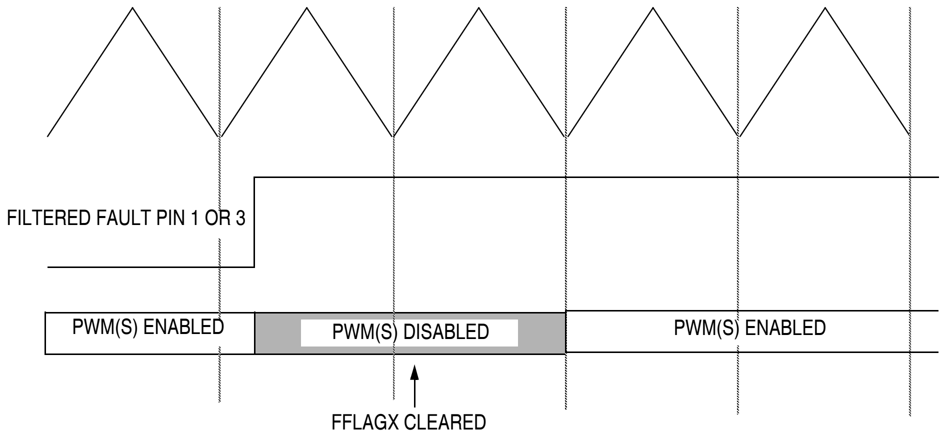


Figure 9-28. PWM Disabling in Manual Mode (Example 1)

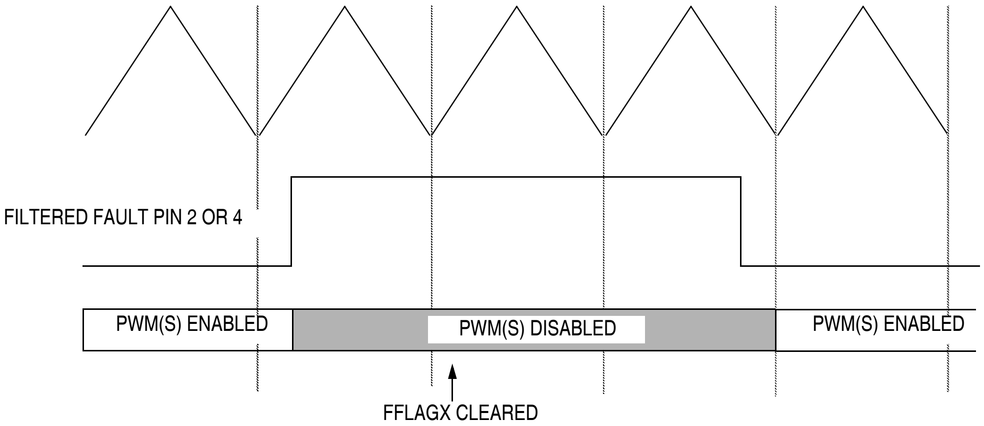


Figure 9-29. PWM Disabling in Manual Mode (Example 2)

The function of the fault control and event bits is the same as in automatic mode except that the PWMs are not re-enabled until the FFLAGx event bit is cleared by writing to the FTACKx bit and the filtered fault condition is cleared (logic low).

9.7.2 Software Output Disable

Setting PWM disable bit DISX or DISY in PWM control register 1 immediately disables the corresponding PWM pins as determined by the bank and disable mapping register. The PWM pin(s) remain disabled until the PWM disable bit is cleared and a new PWM cycle begins as shown in **Figure 9-30**. Setting a PWM disable bit does not latch a CPU interrupt request, and there are no event flags associated with the PWM disable bits.

9.7.3 Output Port Control

When operating the PWMs using the OUTx bits (OUTCTL = 1), fault protection applies as described in this section. Due to the absence of periodic PWM cycles, fault conditions are cleared upon each CPU cycle and the PWM outputs are re-enabled, provided all fault clearing conditions are satisfied.

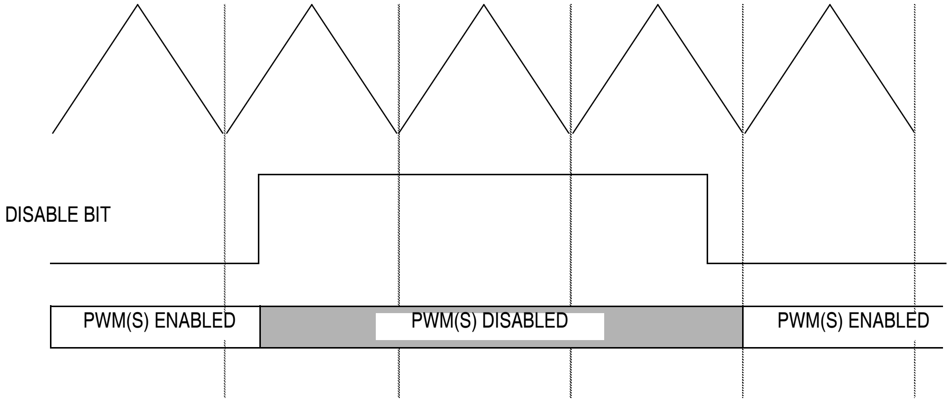


Figure 9-30. PWM Software Disable

9.8 Initialization and the PWMEN Bit

For proper operation, all registers should be initialized and the LDOK bit should be set before enabling the PWM via the PWMEN bit. When the PWMEN bit is first set, a reload will occur immediately, setting the PWMF flag and generating an interrupt if PWMINT is set. In addition, in complementary mode, PWM value registers 1, 3, and 5 will be used for the first PWM cycle if current sensing is selected.

NOTE: *If the LDOK bit is not set when PWMEN is set after a $\overline{\text{RESET}}$, the prescaler and PWM values will be zero, but the modulus will be unknown. If the LDOK bit is not set after the PWMEN bit has been cleared then set (without a RESET), the modulus value that was last loaded will be used.*

If the dead-time register (DEADTM) is changed after PWMEN or OUTCTL is set, an improper dead-time insertion could occur. However, the dead time can never be shorter than the specified value.

*Because of the equals-comparator architecture of this PWM, the modulus = 0 case is considered illegal. Therefore, the modulus register is not reset, and a modulus value of 0 will result in waveforms inconsistent with the other modulus waveforms. See **9.10.2 PWM Counter Modulo Registers**.*

When PWMEN is set, the PWM pins change from high impedance to outputs. At this time, assuming no fault condition is present, the PWM pins will drive according to the PWM values, polarity, and dead time. See the timing diagram in **Figure 9-31**.

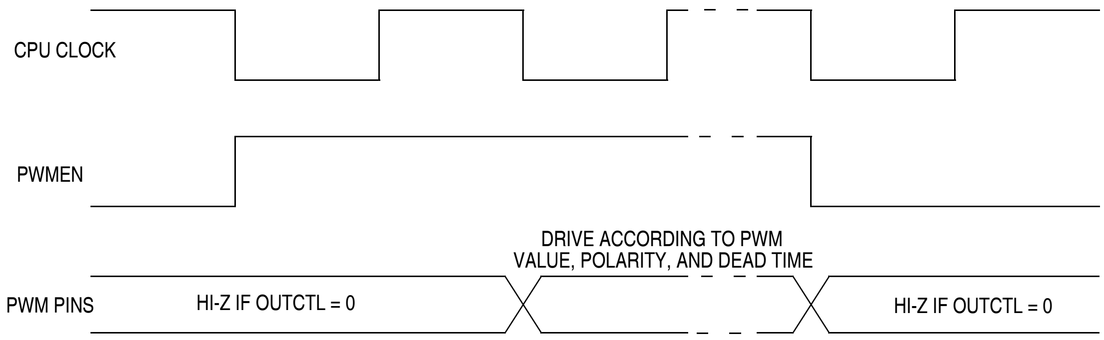


Figure 9-31. PWMEN and PWM Pins

When the PWMEN bit is cleared, this will occur:

- PWM pins will be three-stated unless OUTCTL = 1.
- PWM counter is cleared and will not be clocked.
- Internally, the PWM generator will force its outputs to 0 to avoid glitches when the PWMEN is set again.

When PWMEN is cleared, these features remain active:

- All fault circuitry
- Manual PWM pin control via the PWMOUT register
- Dead-time insertion when PWM pins change via the PWMOUT register

NOTE: *The PWMF flag and pending CPU interrupts are NOT cleared when PWMEN = 0.*

9.9 PWM Operation in Wait Mode

When the microcontroller is put in low-power wait mode via the WAIT instruction, all clocks to the PWM module will continue to run. If an interrupt is issued from the PWM module (via a reload or a fault), the microcontroller will exit wait mode.

Clearing the PWMEN bit before entering wait mode will reduce power consumption in wait mode because the counter, prescaler divider, and LDFQ divider will no longer be clocked. In addition, power will be reduced because the PWMs will no longer toggle.

9.10 Control Logic Block

This subsection provides a description of the control logic block.

9.10.1 PWM Counter Registers

The PWM counter registers (PCNTH and PCNTL) display the 12-bit up/down or up-only counter. When the high byte of the counter is read, the lower byte is latched. PCNTL will hold this latched value until it is read.

Address: $0026

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-32. PWM Counter Register High (PCNTH)

Address: $0027

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

▨ = Unimplemented

Figure 9-33. PWM Counter Register Low (PCNTL)

9.10.2 PWM Counter Modulo Registers

The PWM counter modulus registers (PMODH and PMODL) hold a 12-bit unsigned number that determines the maximum count for the up/down or up-only counter. In center-aligned mode, the PWM period will be twice the modulus (assuming no prescaler). In edge-aligned mode, the PWM period will equal the modulus.

Address: $0028

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8
Write:	▨	▨	▨	▨				
Reset:	0	0	0	0	X	X	X	X

▨ = Unimplemented X = Indeterminate

Figure 9-34. PWM Counter Modulo Register High (PMODH)

Address: $0029

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

X = Indeterminate

Figure 9-35. PWM Counter Modulo Register Low (PMODL)

To avoid erroneous PWM periods, this value is buffered and will not be used by the PWM generator until the LDOK bit has been set and the next PWM load cycle begins.

NOTE: *When reading this register, the value read is the buffer (not necessarily the value the PWM generator is currently using).*

Because of the equals-comparator architecture of this PWM, the modulus = 0 case is considered illegal. Therefore, the modulus register is not reset, and a modulus value of 0 will result in waveforms inconsistent with the other modulus waveforms. If a modulus of 0 is loaded, the counter will continually count down from $FFF. This operation will not be tested or guaranteed (the user should consider it illegal). However, the dead-time constraints and fault conditions will still be guaranteed.

9.10.3 PWM X Value Registers

Each of the six PWMs has a 16-bit PWM value register.

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

Bold = Buffered

Figure 9-36. PWMx Value Registers High (PVALxH)

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

Bold = Buffered

Figure 9-37. PWMx Value Registers Low (PVALxL)

The 16-bit signed value stored in this register determines the duty cycle of the PWM. The duty cycle is defined as: (PWM value/modulus) x 100.

Writing a number less than or equal to 0 causes the PWM to be off for the entire PWM period. Writing a number greater than or equal to the 12-bit modulus causes the PWM to be on for the entire PWM period.

If the complementary mode is selected, the PWM pairs share PWM value registers.

To avoid erroneous PWM pulses, this value is buffered and will not be used by the PWM generator until the LDOK bit has been set and the next PWM load cycle begins.

NOTE: *When reading these registers, the value read is the buffer (not necessarily the value the PWM generator is currently using).*

9.10.4 PWM Control Register 1

PWM control register 1 (PCTL1) controls PWM enabling/disabling, the loading of new modulus, prescaler, PWM values, and the PWM correction method. In addition, this register contains the software disable bits to force the PWM outputs to their inactive states (according to the disable mapping register).

Address: $0020

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DISX	DISY	PWMINT	PWMF	ISENS1	ISENS0	LDOK	PWMEN
Reset:	0	0	0	0	0	0	0	0

Figure 9-38. PWM Control Register 1 (PCTL1)

DISX — Software Disable Bit for Bank X

This read/write bit allows the user to disable one or more PWM pins in bank X. The pins that are disabled are determined by the disable mapping write-once register.

1 = Disable PWM pins in bank X
0 = Re-enable PWM pins at beginning of next PWM cycle

DISY — Software Disable Bit for Bank Y

This read/write bit allows the user to disable one or more PWM pins in bank Y. The pins that are disabled are determined by the disable mapping write-once register.

1 = Disable PWM pins in bank Y
0 = Re-enable PWM pins at beginning of next PWM cycle

PWMINT — PWM Interrupt Enable Bit

This read/write bit allows the user to enable and disable PWM CPU interrupts. If set, a CPU interrupt will be pending when the PWMF flag is set.

1 = Enable PWM CPU interrupts
0 = Disable PWM CPU interrupts

NOTE: *When PWMINT is cleared, pending CPU interrupts are inhibited.*

PWMF — PWM Reload Flag

This read/write bit is set at the beginning of every reload cycle regardless of the state of the LDOK bit. This bit is cleared by reading PWM control register 1 with the PWMF flag set, then writing a logic 0 to PWMF. If another reload occurs before the clearing sequence is complete, then writing logic 0 to PWMF has no effect.

1 = New reload cycle began.
0 = New reload cycle has not begun.

NOTE: *When PWMF is cleared, pending PWM CPU interrupts are cleared (not including fault interrupts).*

ISENS1 and ISENS0 — Current Sense Correction Bits

These read/write bits select the top/bottom correction scheme as shown in **Table 9-7**.

Table 9-7. Correction Methods

Current Correction Bits ISENS1 and ISENS0	Correction Method
00 01	Bits IPOL1, IPOL2, and IPOL3 are used for correction.
10	Current sensing on pins $\overline{\text{IS1}}$, $\overline{\text{IS2}}$, and $\overline{\text{IS3}}$ occurs during the dead time.
11	Current sensing on pins $\overline{\text{IS1}}$, $\overline{\text{IS2}}$, and $\overline{\text{IS3}}$ occurs at the half cycle in center-aligned mode and at the end of the cycle in edge-aligned mode.

1. The polarity of the $\overline{\text{ISx}}$ pin is latched when both the top and bottom PWMs are off. At the 0% and 100% duty cycle boundaries, there is no dead time, so no new current value is sensed.
2. Current is sensed even with 0% and 100% duty cycle.

NOTE: *The ISENSx bits are not buffered. Changing the current sensing method can affect the present PWM cycle.*

LDOK— Load OK Bit

This read/write bit loads the prescaler bits of the PMCTL2 register and the entire PMMODH/L and PWMVALH/L registers into a set of buffers. The buffered prescaler divisor, PWM counter modulus value, and PWM pulse with take effect at the next PWM load. Set LDOK by reading it when it is logic 0 and then writing a logic 1 to it. LDOK is automatically cleared after the new values are loaded or can be manually cleared before a reaload by writing a 0 to it. Reset clears LDOK.

1 = Load prescaler, modulus, and PWM values.
0 = Do not load new modulus, prescaler, and PWM values.

NOTE: *The user should initialize the PWM registers and set the LDOK bit before enabling the PWM.*

A PWM CPU interrupt request can still be generated when LDOK is 0.

PWMEN — PWM Module Enable Bit

This read/write bit enables and disables the PWM generator and the PWM pins. When PWMEN is clear, the PWM generator is disabled and the PWM pins are in the high-impedance state (unless OUTCTL = 1).

When the PWMEN bit is set, the PWM generator and PWM pins are activated.

For more information, see **9.8 Initialization and the PWMEN Bit**.

1 = PWM generator and PWM pins enabled
0 = PWM generator and PWM pins disabled

9.10.5 PWM Control Register 2

PWM control register 2 (PCTL2) controls the PWM load frequency, the PWM correction method, and the PWM counter prescaler. For ease of software and to avoid erroneous PWM periods, some of these register bits are buffered. The PWM generator will not use the prescaler value until the LDOK bit has been set, and a new PWM cycle is starting. The correction bits are used at the beginning of each PWM cycle (if the ISENSx bits are configured for software correction). The load frequency bits are not used until the current load cycle is complete.

NOTE: *The user should initialize this register before enabling the PWM.*

Address: $0021

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	**LDFQ1**	**LDFQ0**	0	**IPOL1**	**IPOL2**	**IPOL3**	**PRSC1**	**PRSC0**
Write:			▨					
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented **Bold** = Buffered

Figure 9-39. PWM Control Register 2 (PCTL2)

LDFQ1 and LDFQ0 — PWM Load Frequency Bits

These buffered read/write bits select the PWM CPU load frequency according to **Table 9-8**.

NOTE: *When reading these bits, the value read is the buffer value (not necessarily the value the PWM generator is currently using).*

The LDFQx bits take effect when the current load cycle is complete regardless of the state of the load okay bit, LDOK.

Table 9-8. PWM Reload Frequency

Reload Frequency Bits LDFQ1 and LDFQ0	PWM Reload Frequency
00	Every PWM cycle
01	Every 2 PWM cycles
10	Every 4 PWM cycles
11	Every 8 PWM cycles

NOTE: *Reading the LPFQx bit reads the buffered values and not necessarily the values currently in effect.*

IPOL1 — Top/Bottom Correction Bit for PWM Pair 1 (PWMs 1 and 2)

This buffered read/write bit selects which PWM value register is used if top/bottom correction is to be achieved without current sensing.
1 = Use PWM value register 2
0 = Use PWM value register 1

NOTE: *When reading this bit, the value read is the buffer value (not necessarily the value the output control block is currently using).*

The IPOLx bits take effect at the beginning of the next load cycle, regardless of the state of the load okay bit, LDOK.

IPOL2 — Top/Bottom Correction Bit for PWM Pair 2 (PWMs 3 and 4)

This buffered read/write bit selects which PWM value register is used if top/bottom correction is to be achieved without current sensing.

1 = Use PWM value register 4
0 = Use PWM value register 3

NOTE: *When reading this bit, the value read is the buffer value (not necessarily the value the output control block is currently using).*

IPOL3 — Top/Bottom Correction Bit for PWM Pair 3 (PWMs 5 and 6)

This buffered read/write bit selects which PWM value register is used if top/bottom correction is to be achieved without current sensing.

1 = Use PWM value register 6
0 = Use PWM value register 5

NOTE: *When reading this bit, the value read is the buffer value (not necessarily the value the output control block is currently using).*

PRSC1 and PRSC0 — PWM Prescaler Bits

These buffered read/write bits allow the PWM clock frequency to be modified as shown in **Table 9-9**.

NOTE: *When reading these bits, the value read is the buffer value (not necessarily the value the PWM generator is currently using).*

Table 9-9. PWM Prescaler

Prescaler Bits PRSC1 and PRSC0	PWM Clock Frequency
00	f_{op}
01	$f_{op}/2$
10	$f_{op}/4$
11	$f_{op}/8$

9.10.6 Dead-Time Write-Once Register

The dead-time write-once register (DEADTM) holds an 8-bit value which specifies the number of CPU clock cycles to use for the dead time when complementary PWM mode is selected. After this register is written for the first time, it cannot be rewritten unless a reset occurs. Dead time is not affected by changes to the prescaler value.

Address: $0036

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

Figure 9-40. Dead-Time Write-Once Register (DEADTM)

9.10.7 PWM Disable Mapping Write-Once Register

The PWM disable mapping write-once register (DISMAP) holds an 8-bit value which determines which PWM pins will be disabled if an external fault or software disable occurs. For a further description of disable mapping, see **9.7 Fault Protection**. After this register is written for the first time, it cannot be rewritten unless a reset occurs.

Address: $0037

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

Figure 9-41. PWM Disable Mapping Write-Once Register (DISMAP)

9.10.8 Fault Control Register

The fault control register (FCR) controls the fault-protection circuitry

Address: $0022

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	FINT4	FMODE4	FINT3	FMODE3	FINT2	FMODE2	FINT1	FMODE1
Reset:	0	0	0	0	0	0	0	0

Figure 9-42. Fault Control Register (FCR)

FINT4 — Fault 4 Interrupt Enable Bit

This read/write bit allows the CPU interrupt caused by faults on fault pin 4 to be enabled. The fault protection circuitry is independent of this bit and will always be active. If a fault is detected, the PWM pins will still be disabled according to the disable mapping register.

1 = Fault pin 4 will cause CPU interrupts.
0 = Fault pin 4 will not cause CPU interrupts.

FMODE4 — Fault Mode Selection for Fault Pin 4 Bit
(automatic versus manual mode)

This read/write bit allows the user to select between automatic and manual mode faults. For further descriptions of each mode, see **9.7 Fault Protection**.

1 = Automatic mode
0 = Manual mode

FINT3 — Fault 3 Interrupt Enable Bit

This read/write bit allows the CPU interrupt caused by faults on fault pin 3 to be enabled. The fault protection circuitry is independent of this bit and will always be active. If a fault is detected, the PWM pins will still be disabled according to the disable mapping register.

1 = Fault pin 3 will cause CPU interrupts.
0 = Fault pin 3 will not cause CPU interrupts.

FMODE3 — Fault Mode Selection for Fault Pin 3 Bit
(automatic versus manual mode)

This read/write bit allows the user to select between automatic and manual mode faults. For further descriptions of each mode, see **9.7 Fault Protection**.

1 = Automatic mode
0 = Manual mode

FINT2 — Fault 2 Interrupt Enable Bit

This read/write bit allows the CPU interrupt caused by faults on fault pin 2 to be enabled. The fault protection circuitry is independent of this bit and will always be active. If a fault is detected, the PWM pins will still be disabled according to the disable mapping register.

1 = Fault pin 2 will cause CPU interrupts.
0 = Fault pin 2 will not cause CPU interrupts.

FMODE2 — Fault Mode Selection for Fault Pin 2 Bit
(automatic versus manual mode)

This read/write bit allows the user to select between automatic and manual mode faults. For further descriptions of each mode, see **9.7 Fault Protection**.

1 = Automatic mode
0 = Manual mode

FINT1 — Fault 1 Interrupt Enable Bit

This read/write bit allows the CPU interrupt caused by faults on fault pin 1 to be enabled. The fault protection circuitry is independent of this bit and will always be active. If a fault is detected, the PWM pins will still be disabled according to the disable mapping register.

1 = Fault pin 1 will cause CPU interrupts.
0 = Fault pin 1 will not cause CPU interrupts.

FMODE1 — Fault Mode Selection for Fault Pin 1 Bit
(automatic versus manual mode)

This read/write bit allows the user to select between automatic and manual mode faults. For further descriptions of each mode, see **9.7 Fault Protection**.

1 = Automatic mode
0 = Manual mode

9.10.9 Fault Status Register

The fault status register (FSR) is a read-only register that indicates the current fault status.

Address: $0023

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	FPIN4	FFLAG4	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	U	0	U	0	U	0	U	0

▨ = Unimplemented U = Unaffected

Figure 9-43. Fault Status Register (FSR)

FPIN4 — State of Fault Pin 4 Bit

This read-only bit allows the user to read the current state of fault pin 4.

1 = Fault pin 4 is at logic 1.
0 = Fault pin 4 is at logic 0.

FFLAG4 — Fault Event Flag 4

The FFLAG4 event bit is set within two CPU cycles after a rising edge on fault pin 4. To clear the FFLAG4 bit, the user must write a 1 to the FTACK4 bit in the fault acknowledge register.

1 = A fault has occurred on fault pin 4.
0 = No new fault on fault pin 4

FPIN3 — State of Fault Pin 3 Bit

This read-only bit allows the user to read the current state of fault pin 3.

1 = Fault pin 3 is at logic 1.
0 = Fault pin 3 is at logic 0.

FFLAG3 — Fault Event Flag 3

The FFLAG3 event bit is set within two CPU cycles after a rising edge on fault pin 3. To clear the FFLAG3 bit, the user must write a 1 to the FTACK3 bit in the fault acknowledge register.

1 = A fault has occurred on fault pin 3.
0 = No new fault on fault pin 3.

FPIN2 — State of Fault Pin 2 Bit

This read-only bit allows the user to read the current state of fault pin 2.

1 = Fault pin 2 is at logic 1.
0 = Fault pin 2 is at logic 0.

FFLAG2 — Fault Event Flag 2

The FFLAG2 event bit is set within two CPU cycles after a rising edge on fault pin 2. To clear the FFLAG2 bit, the user must write a 1 to the FTACK2 bit in the fault acknowledge register.

1 = A fault has occurred on fault pin 2.
0 = No new fault on fault pin 2

FPIN1 — State of Fault Pin 1 Bit

This read-only bit allows the user to read the current state of fault pin 1.

1 = Fault pin 1 is at logic 1.
0 = Fault pin 1 is at logic 0.

FFLAG1 — Fault Event Flag 1

The FFLAG1 event bit is set within two CPU cycles after a rising edge on fault pin 1. To clear the FFLAG1 bit, the user must write a 1 to the FTACK1 bit in the fault acknowledge register.

1 = A fault has occurred on fault pin 1.
0 = No new fault on fault pin 1.

9.10.10 Fault Acknowledge Register

The fault acknowledge register (FTACK) is used to acknowledge and clear the FFLAGs. In addition, it is used to monitor the current sensing bits to test proper operation.

Address: $0024

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	DT6	DT5	DT4	DT3	DT2	DT1
Write:	▨	FTACK4	▨	FTACK3	▨	FTACK2	▨	FTACK1
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-44. Fault Acknowledge Register (FTACK)

FTACK4 — Fault Acknowledge 4 Bit

The FTACK4 bit is used to acknowledge and clear FFLAG4. This bit will always read 0. Writing a 1 to this bit will clear FFLAG4. Writing a 0 will have no effect.

FTACK3 — Fault Acknowledge 3 Bit

The FTACK3 bit is used to acknowledge and clear FFLAG3. This bit will always read 0. Writing a 1 to this bit will clear FFLAG3. Writing a 0 will have no effect.

FTACK2 — Fault Acknowledge 2 Bit

The FTACK2 bit is used to acknowledge and clear FFLAG2. This bit will always read 0. Writing a 1 to this bit will clear FFLAG2. Writing a 0 will have no effect.

FTACK1 — Fault Acknowledge 1 Bit

The FTACK1 bit is used to acknowledge and clear FFLAG1. This bit will always read 0. Writing a 1 to this bit will clear FFLAG1. Writing a 0 will have no effect.

DT6 — Dead Time 6 Bit

Current sensing pin IS3 is monitored immediately before dead time ends due to the assertion of PWM6.

DT5 — Dead Time 5 Bit

Current sensing pin IS3 is monitored immediately before dead time ends due to the assertion of PWM5.

DT4 — Dead Time 4 Bit

Current sensing pin IS2 is monitored immediately before dead time ends due to the assertion of PWM4.

DT3 — Dead Time 3 Bit

Current sensing pin IS2 is monitored immediately before dead time ends due to the assertion of PWM3.

DT2 — Dead Time 2 Bit

Current sensing pin IS1 is monitored immediately before dead time ends due to the assertion of PWM2.

DT1 — Dead Time 1 Bit

Current sensing pin IS1 is monitored immediately before dead time ends due to the assertion of PWM1.

9.10.11 PWM Output Control Register

The PWM output control register (PWMOUT) is used to manually control the PWM pins.

Address: $0025

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	OUTCTL	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1
Write:	▨							
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 9-45. PWM Output Control Register (PWMOUT)

OUTCTL— Output Control Enable Bit

This read/write bit allows the user to manually control the PWM pins. When set, the PWM generator is no longer the input to the dead-time and output circuitry. The OUTx bits determine the state of the PWM pins. Setting the OUTCTL bit does not disable the PWM generator. The generator continues to run, but is no longer the input to the PWM dead-time and output circuitry. When OUTCTL is cleared, the outputs of the PWM generator immediately become the inputs to the dead-time and output circuitry.

1 = PWM outputs controlled manually
0 = PWM outputs determined by PWM generator

OUT6–OUT1— PWM Pin Output Control Bits

These read/write bits control the PWM pins according to **Table 9-10**.

Table 9-10. OUTx Bits

OUTx Bit	Complementary Mode	Independent Mode
OUT1	1 — PWM1 is active. 0 — PWM1 is inactive.	1 — PWM1 is active. 0 — PWM1 is inactive.
OUT2	1 — PWM2 is complement of PWM 1. 0 — PWM2 is inactive.	1 — PWM2 is active. 0 — PWM2 is inactive.
OUT3	1 — PWM3 is active. 0 — PWM3 is inactive.	1 — PWM3 is active. 0 — PWM3 is inactive.
OUT4	1 — PWM4 is complement of PWM 3. 0 — PWM4 is inactive.	1 — PWM4 is active. 0 — PWM4 is inactive.
OUT5	1 — PWM5 is active. 0 — PWM5 is inactive.	1 — PWM5 is active. 0 — PWM5 is inactive.
OUT6	1 — PWM 6 is complement of PWM 5. 0 — PWM6 is inactive.	1 — PWM6 is active. 0 — PWM6 is inactive.

9.11 PWM Glossary

CPU cycle — One internal bus cycle ($1/f_{op}$)

PWM clock cycle (or period) — One tick of the PWM counter ($1/f_{op}$ with no prescaler). See **Figure 9-46**.

PWM cycle (or period)

- Center-aligned mode: The time it takes the PWM counter to count up and count down (modulus * $2/f_{op}$ assuming no prescaler). See **Figure 9-46**.
- Edge-aligned mode: The time it takes the PWM counter to count up ($modulus/f_{op}$). See **Figure 9-46**.

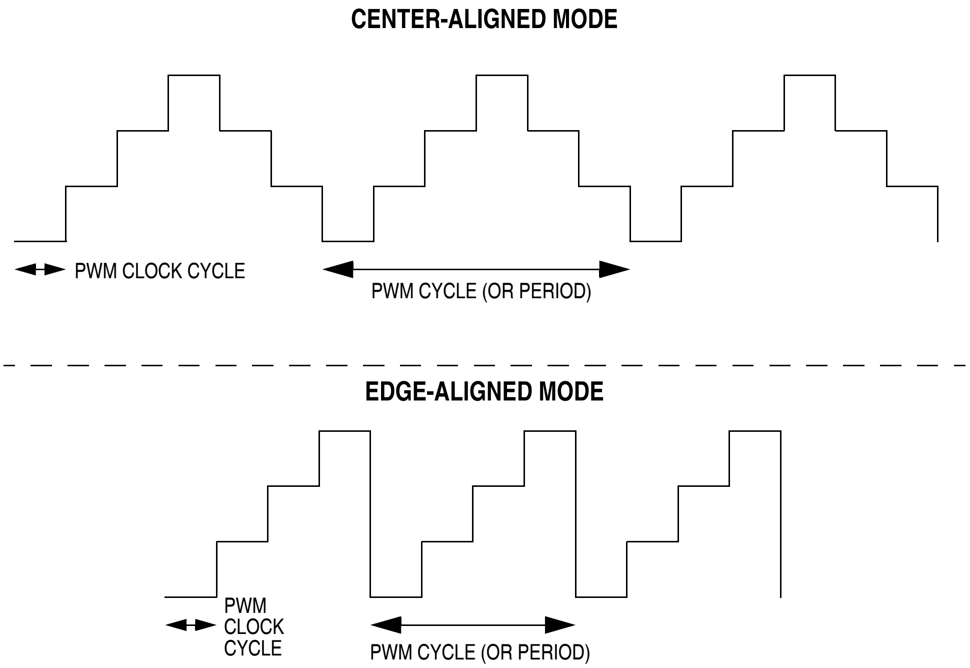


Figure 9-46. PWM Clock Cycle and PWM Cycle Definitions

PWM Load Frequency — Frequency at which new PWM parameters get loaded into the PWM. See **Figure 9-47**.

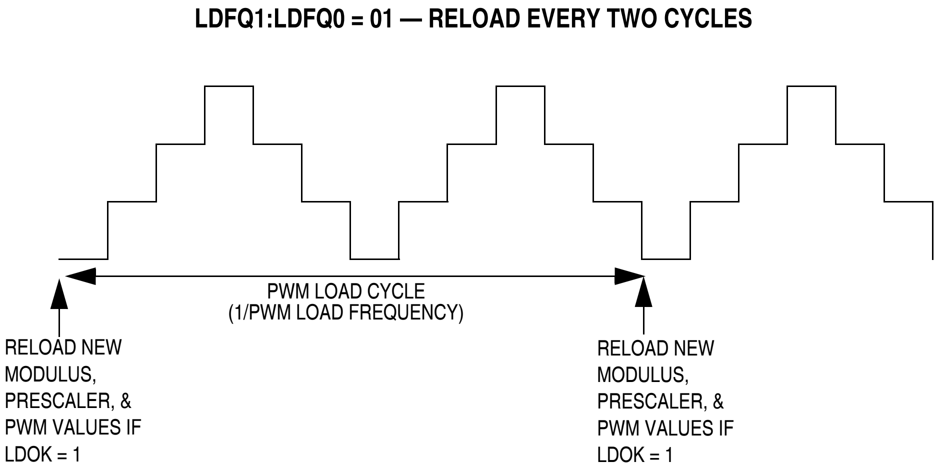


Figure 9-47. PWM Load Cycle/Frequency Definition

Section 10. Monitor ROM (MON)

10.1 Contents

10.2 Introduction

This section describes the monitor read-only memory (ROM). The monitor ROM (MON) allows complete testing of the MCU through a single-wire interface with a host computer.

10.3 Features

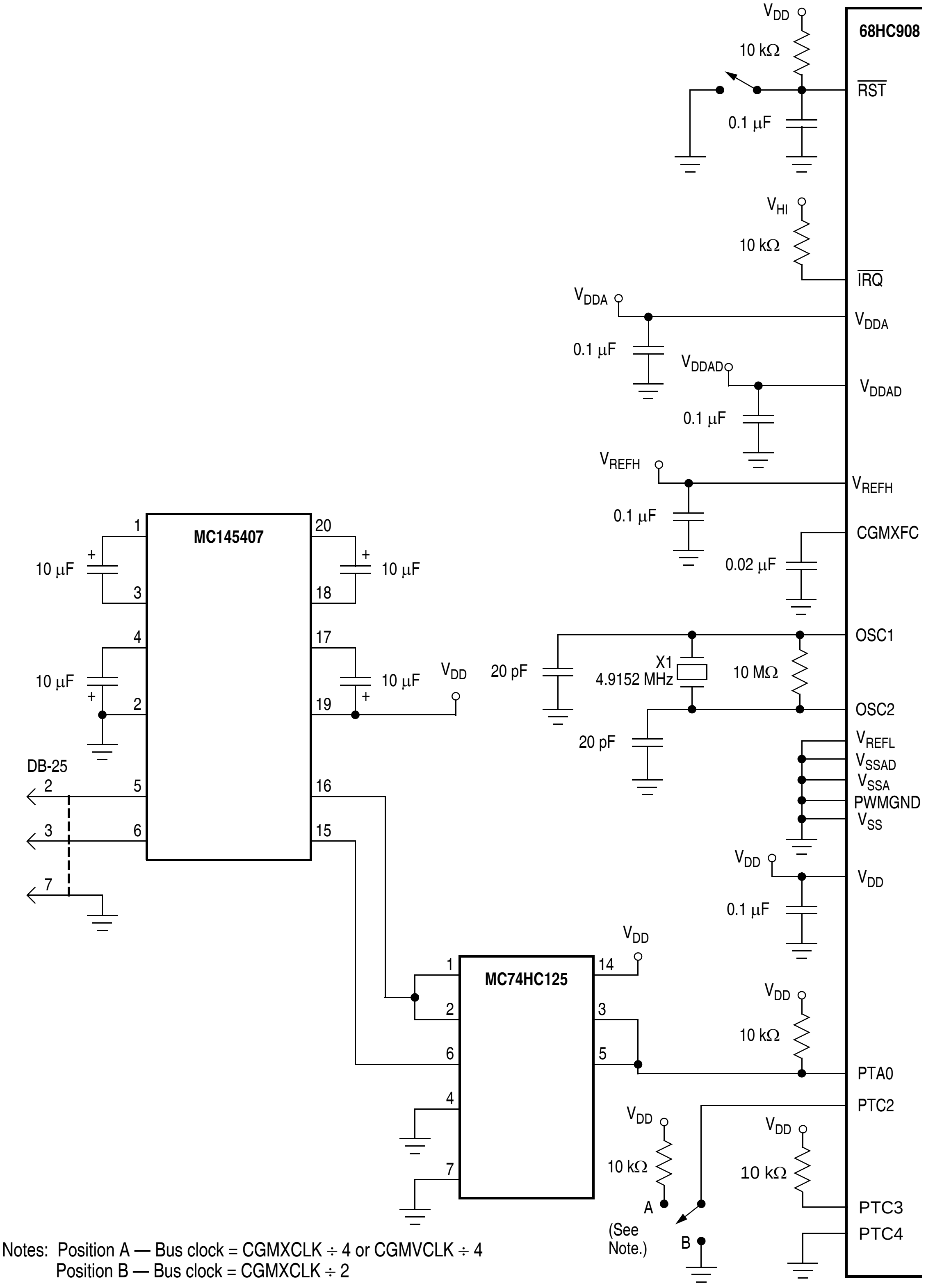
Features of the monitor ROM include:

- Normal user-mode pin functionality
- One pin dedicated to serial communication between monitor ROM and host computer
- Standard mark/space non-return-to-zero (NRZ) communication with host computer
- 4800 baud–28.8 Kbaud communication with host computer
- Execution of code in random-access memory (RAM) or ROM
- FLASH programming

10.4 Functional Description

The monitor ROM receives and executes commands from a host computer. **Figure 10-1** shows a sample circuit 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 host-computer code in RAM while all 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.



Notes: Position A — Bus clock = CGMXCLK ÷ 4 or CGMVCLK ÷ 4
Position B — Bus clock = CGMXCLK ÷ 2

Figure 10-1. Monitor Mode Circuit

10.4.1 Entering Monitor Mode

Table 10-1 shows the pin conditions for entering monitor mode.

Table 10-1. Mode Selection

$\overline{\text{IRQ}}$ Pin	PTC3 Pin	PTC4 Pin	PTA0 Pin	PTC2 Pin	Mode	CGMOUT	Bus Frequency
V_{HI}	1	0	1	1	Monitor	$\frac{\text{CGMXCLK}}{2}$ or $\frac{\text{CGMVCLK}}{2}$	$\frac{\text{CGMOUT}}{2}$
V_{HI}	1	0	1	0	Monitor	CGMXCLK	$\frac{\text{CGMOUT}}{2}$

Enter monitor mode by either:

- Executing a software interrupt instruction (SWI) or
- Applying a logic 0 and then a logic 1 to the $\overline{\text{RST}}$ pin

Once out of reset, the MCU waits for the host to send eight security bytes. After receiving the security bytes, the MCU sends a break signal (10 consecutive logic 0s) to the host computer, indicating that it is ready to receive a command. The break signal also provides a timing reference to allow the host to determine the necessary baud rate.

Monitor mode uses alternate vectors for reset and SWI. 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. The computer operating properly (COP) module is disabled in monitor mode as long as V_{HI} is applied to either the $\overline{\text{IRQ}}$ pin or the $\overline{\text{RST}}$ pin. (See **Section 7. System Integration Module (SIM)** for more information on modes of operation.)

NOTE: *Holding the PTC2 pin low when entering monitor mode causes a bypass of a divide-by-two stage at the oscillator. 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 percent duty cycle at maximum bus frequency.*

Table 10-2 is a summary of the differences between user mode and monitor mode.

Table 10-2. Mode Differences

Modes	Functions				
	COP	**Reset Vector High**	**Reset Vector Low**	**SWI Vector High**	**SWI Vector Low**
User	Enabled	$FFFE	$FFFF	$FFFC	$FFFD
Monitor	Disabled(1)	$FEFE	$FEFF	$FEFC	$FEFD

1. If the high voltage (V_{HI}) is removed from the $\overline{\text{IRQ}}$ pin or the $\overline{\text{RST}}$ pin, the SIM asserts its COP enable output. The COP is the COPD bit in the configuration register.

10.4.2 Data Format

Communication with the monitor ROM is in standard non-return-to-zero (NRZ) mark/space data format. (See **Figure 10-2** and **Figure 10-3**.)

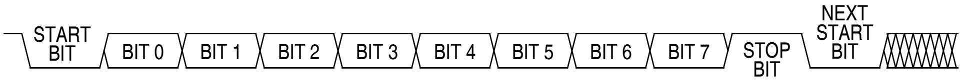


Figure 10-2. Monitor Data Format

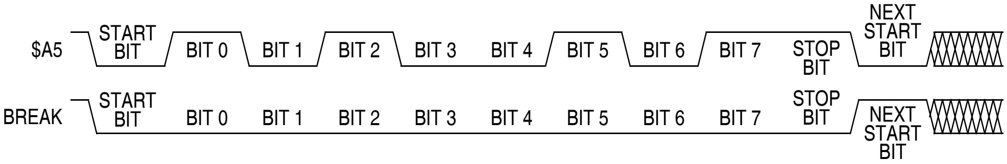


Figure 10-3. Sample Monitor Waveforms

The data transmit and receive rate can be anywhere from 4800 baud to 28.8 Kbaud. Transmit and receive baud rates must be identical.

10.4.3 Echoing

As shown in **Figure 10-4**, the monitor ROM immediately echoes each received byte back to the PTA0 pin for error checking.

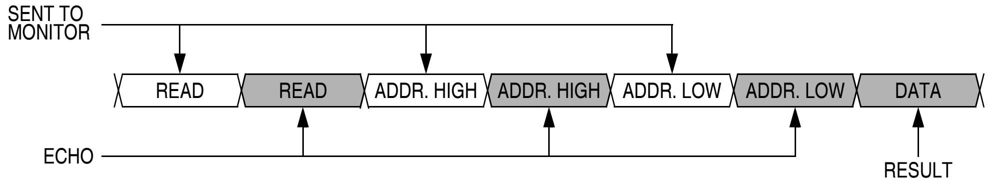


Figure 10-4. Read Transaction

Any result of a command appears after the echo of the last byte of the command.

10.4.4 Break Signal

A start bit followed by nine low bits is a break signal. See **Figure 10-5**. When the monitor receives a break signal, it drives the PTA0 pin high for the duration of two bits before echoing the break signal.

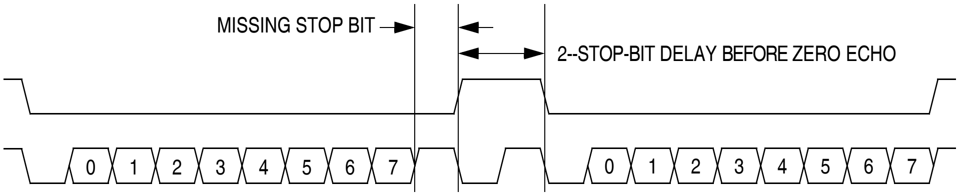


Figure 10-5. Break Transaction

10.4.5 Commands

The monitor ROM uses these commands:

- READ, read memory
- WRITE, write memory
- IREAD, indexed read
- IWRITE, indexed write
- READSP, read stack pointer
- RUN, run user program

Table 10-3. READ (Read Memory) Command

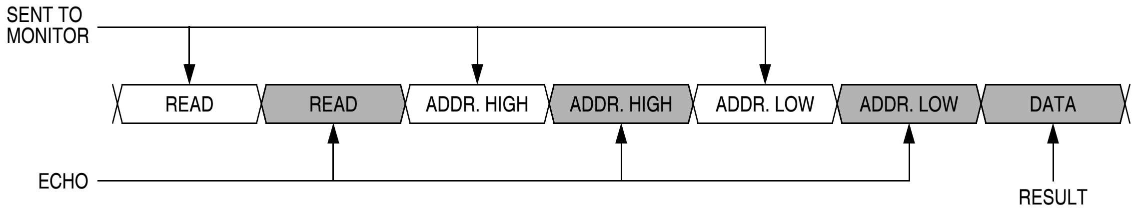
Description	Read byte from memory
Operand	Specifies 2-byte address in high byte:low byte order
Data returned	Returns contents of specified address
Opcode	$4A
Command sequence	

Table 10-4. WRITE (Write Memory) Command

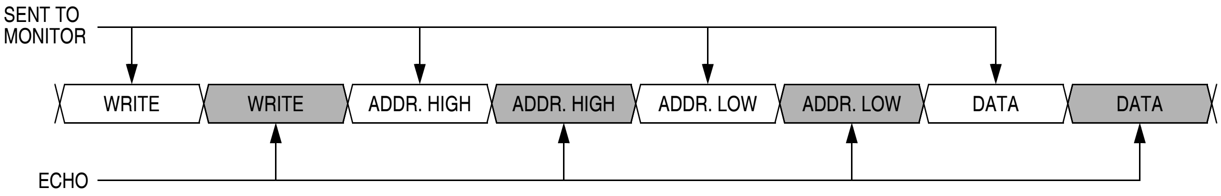
Description	Write byte to memory
Operand	Specifies 2-byte address in high byte:low byte order; low byte followed by data byte
Data returned	None
Opcode	$49
Command sequence	

Table 10-5. IREAD (Indexed Read) Command

<table>
<tr><td>Description</td><td>Read next 2 bytes in memory from last address accessed</td></tr>
<tr><td>Operand</td><td>Specifies 2-byte address in high byte:low byte order</td></tr>
<tr><td>Data returned</td><td>Returns contents of next two addresses</td></tr>
<tr><td>Opcode</td><td>$1A</td></tr>
<tr><td colspan="2">Command sequence

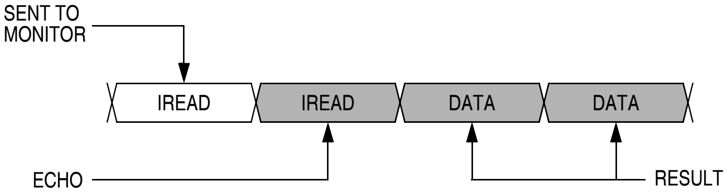
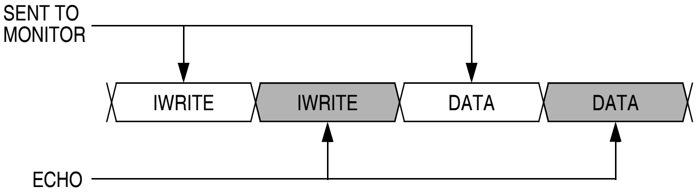

</td></tr>
</table>

Table 10-6. IWRITE (Indexed Write) Command

<table>
<tr><td>Description</td><td>Write to last address accessed + 1</td></tr>
<tr><td>Operand</td><td>Specifies single data byte</td></tr>
<tr><td>Data returned</td><td>None</td></tr>
<tr><td>Opcode</td><td>$19</td></tr>
<tr><td colspan="2">Command sequence


</td></tr>
</table>

NOTE: *A sequence of IREAD or IWRITE commands can sequentially access a block of memory over the full 64-Kbyte memory map.*

Table 10-7. READSP (Read Stack Pointer) Command

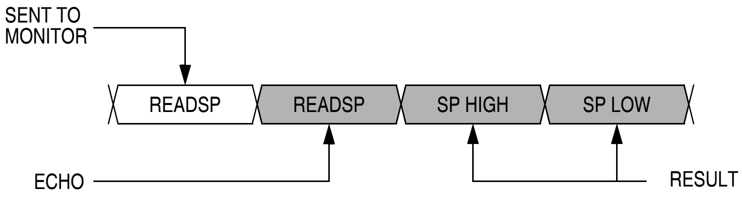
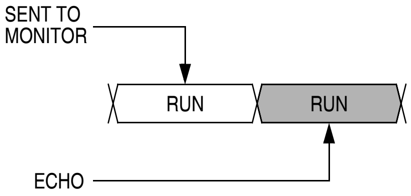
<table>
<tr><td>Description</td><td>Reads stack pointer</td></tr>
<tr><td>Operand</td><td>None</td></tr>
<tr><td>Data returned</td><td>Returns stack pointer in high byte:low byte order</td></tr>
<tr><td>Opcode</td><td>$0C</td></tr>
<tr><td colspan="2">Command sequence

</td></tr>
</table>

Table 10-8. RUN (Run User Program) Command

<table>
<tr><td>Description</td><td>Executes RTI instruction</td></tr>
<tr><td>Operand</td><td>None</td></tr>
<tr><td>Data returned</td><td>None</td></tr>
<tr><td>Opcode</td><td>$28</td></tr>
<tr><td colspan="2">Command sequence

</td></tr>
</table>

10.4.6 Baud Rate

With a 4.9152-MHz crystal and the PTC2 pin at logic 1 during reset, data is transferred between the monitor and host at 4800 baud. If the PTC2 pin is at logic 0 during reset, the monitor baud rate is 9600.

10.5 Security

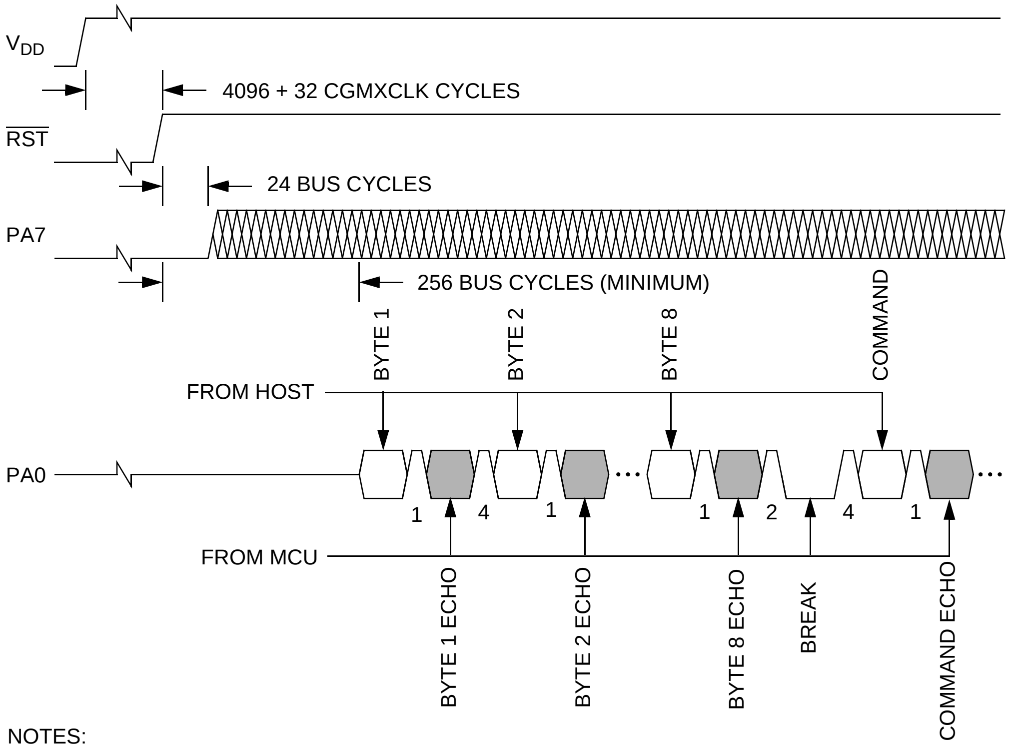
A security feature discourages unauthorized reading of FLASH locations while in monitor mode. The host can bypass the security feature at monitor mode entry by sending eight security bytes that match the bytes at locations $FFF6–$FFFD. Locations $FFF6–$FFFD contain user-defined data.

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 10-6**.)

Upon power-on reset, if the received bytes of the security code do not match the data at locations $FFF6–$FFFD, the host fails to bypass the security feature. The MCU remains in monitor mode, but reading a FLASH location returns an invalid value and trying to execute code from FLASH causes an illegal address reset. After receiving the eight security bytes from the host, the MCU transmits a break character, signifying that it is ready to receive a command.

NOTE: *The MCU does not transmit a break character until after the host sends the eight security bytes.*



NOTES:
1 = Echo delay, 2 bit times
2 = Data return delay, 2 bit times
4 = Wait 1 bit time before sending next byte.

Figure 10-6. Monitor Mode Entry Timing

To determine whether the security code entered is correct, check to see if bit 6 of RAM address $40 is set. If it is, then the correct security code has been entered and FLASH can be accessed.

If the security sequence fails, the device can be reset (via power-pin reset only) and brought up in monitor mode to attempt another entry. After failing the security sequence, the FLASH mode can also be bulk erased by executing an erase routine that was downloaded into internal RAM. The bulk erase operation clears the security code locations so that all eight security bytes become $00.

Section 11. Timer Interface A (TIMA)

11.1 Contents

11.2 Introduction

This section describes the timer interface module A (TIMA). The TIMA is a 4-channel timer that provides a timing reference with input capture, output compare, and pulse-width modulator functions. **Figure 11-1** is a block diagram of the TIMA.

11.3 Features

Features of the TIMA include:

- Four 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 modulator (PWM) signal generation
- Programmable TIMA clock input:
  - 7-frequency internal bus clock prescaler selection
  - External TIMA clock input (4-MHz maximum frequency)
- Free-running or modulo up-count operation
- Toggle any channel pin on overflow
- TIMA counter stop and reset bits

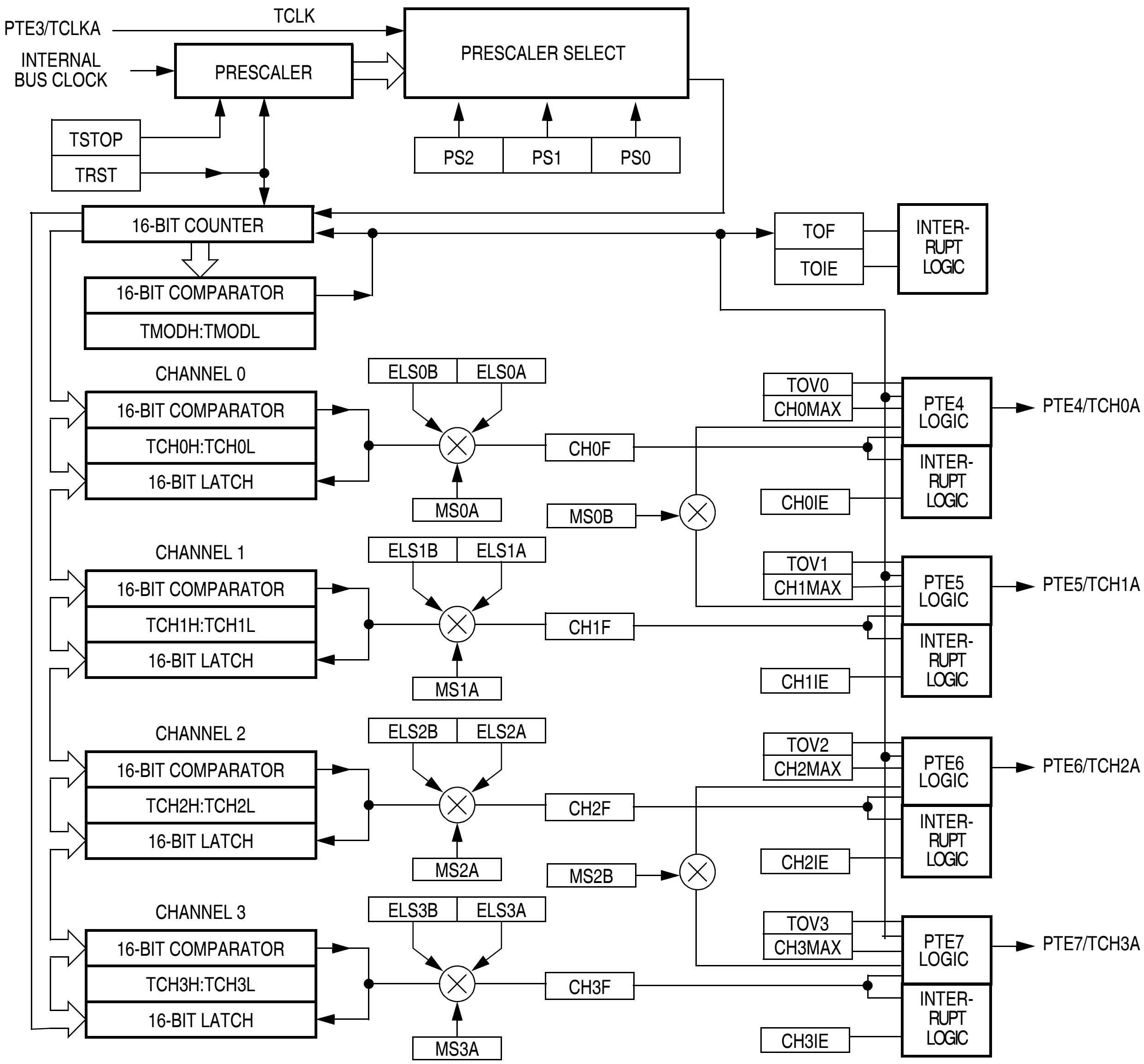


Figure 11-1. TIMA Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000E	TIMA Status/Control Register (TASC) See page 218.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	R			
		Reset:	0	0	1	0	0	0	0	0
$000F	TIMA Counter Register High (TACNTH) See page 221.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0010	TIMA Counter Register Low TACNTL) See page 221.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0011	TIMA Counter Modulo Register High (TAMODH) See page 222.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0012	TIMA Counter Modulo Register Low (TAMODL) See page 222.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0013	TIMA Channel 0 Status/Control Register (TASC0) See page 223.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0014	TIMA Channel 0 Register High (TACH0H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0015	TIMA Channel 0 Register Low (TACH0L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0016	TIMA Channel 1 Status/Control Register (TASC1) See page 223.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 11-2. TIM I/O Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0017	TIMA Channel 1 Register High (TACH1H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0018	TIMA Channel 1 Register Low (TACH1L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0019	TIMA Channel 2 Status/Control Register (TASC2) See page 223.	Read:	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	TOV2	CH2MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$001A	TIMA Channel 2 Register High (TACH2H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$001B	TIMA Channel 2 Register Low (TACH2L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$001C	TIMA Channel 3 Status/Control Register (TASC3) See page 228.	Read:	CH3F	CH3IE	0	MS3A	ELS3B	ELS3A	TOV3	CH3MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0
$001D	TIMA Channel 3 Register High (TACH3H) See page 228.	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$001E	TIMA Channel 3 Register Low (TACH3L) See page 228.	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

R = Reserved

Figure 11-2. TIM I/O Register Summary (Continued)

11.4 Functional Description

Figure 11-1 shows the TIMA structure. The central component of the TIMA is the 16-bit TIMA counter that can operate as a free-running counter or a modulo up-counter. The TIMA counter provides the timing reference for the input capture and output compare functions. The TIMA counter modulo registers, TAMODH–TAMODL, control the modulo value of the TIMA counter. Software can read the TIMA counter value at any time without affecting the counting sequence.

The four TIMA channels are programmable independently as input capture or output compare channels.

11.4.1 TIMA Counter Prescaler

The TIMA clock source can be one of the seven prescaler outputs or the TIMA clock pin, PTE3/TCLKA. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIMA status and control register select the TIMA clock source.

11.4.2 Input Capture

An input capture function has three basic parts:

- Edge select logic
- Input capture latch
- 16-bit counter

Two 8-bit registers, which make up the 16-bit input capture register, are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The polarity of the active edge is programmable. The level transition which triggers the counter transfer is defined by the corresponding input edge bits (ELSxB and ELSxA in TASC0–TASC3 control registers with x referring to the active channel number). When an active edge occurs on the pin of an input capture channel, the TIMA latches the contents of the TIMA counter into the TIMA channel registers, TACHxH–TACHxL. Input captures can generate TIMA CPU interrupt requests. Software can determine that an input capture event has occurred by enabling input capture interrupts or by polling the status flag bit.

The result obtained by an input capture will be two more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization.

The free-running counter contents are transferred to the TIMA channel status and control register (TACHxH–TACHxL, see **11.8.5 TIMA Channel Registers**) on each proper signal transition regardless of whether the TIMA channel flag (CH0F–CH3F in TASC0–TASC3 registers) is set or clear. When the status flag is set, a CPU interrupt is generated if enabled. The value of the count latched or “captured” is the time of the event. Because this value is stored in the input capture register 2 bus cycles after the actual event occurs, user software can respond to this event at a later time and determine the actual time of the event. However, this must be done prior to another input capture on the same pin; otherwise, the previous time value will be lost.

By recording the times for successive edges on an incoming signal, software can determine the period and/or pulse width of the signal. To measure a period, two successive edges of the same polarity are captured. To measure a pulse width, two alternate polarity edges are captured. Software should track the overflows at the 16-bit module counter to extend its range.

Another use for the input capture function is to establish a time reference. In this case, an input capture function is used in conjunction with an output compare function. For example, to activate an output signal a specified number of clock cycles after detecting an input event (edge), use the input capture function to record the time at which the edge occurred. A number corresponding to the desired delay is added to this captured value and stored to an output compare register (see **11.8.5 TIMA Channel Registers**). Because both input captures and output compares are referenced to the same 16-bit modulo counter, the delay can be controlled to the resolution of the counter independent of software latencies.

Reset does not affect the contents of the input capture channel registers.

11.4.3 Output Compare

With the output compare function, the TIMA 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 TIMA can set, clear, or toggle the channel pin. Output compares can generate TIMA CPU interrupt requests.

11.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in **11.4.3 Output Compare**. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIMA channel registers.

An unsynchronized write to the TIMA 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 TIMA overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIMA may pass the new value before it is written.

Use this method 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 channel x TIMA overflow interrupts and write the new value in the TIMA overflow interrupt routine. The TIMA 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.

11.4.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the PTE4/TCH0A pin. The TIMA channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIMA channel 0 status and control register (TASC0) links channel 0 and channel 1. The output compare value in the TIMA channel 0 registers initially controls the output on the PTE4/TCH0A pin. Writing to the TIMA channel 1 registers enables the TIMA channel 1 registers to synchronously control the output after the TIMA overflows. At each subsequent overflow, the TIMA channel registers (0 or 1) that control the output are the ones written to last. TASC0 controls and monitors the buffered output compare function, and TIMA channel 1 status and control register (TASC1) is unused. While the MS0B bit is set, the channel 1 pin, PTE5/TCH1A, is available as a general-purpose I/O pin.

Channels 2 and 3 can be linked to form a buffered output compare channel whose output appears on the PTE6/TCH2A pin. The TIMA channel registers of the linked pair alternately control the output.

Setting the MS2B bit in TIMA channel 2 status and control register (TASC2) links channel 2 and channel 3. The output compare value in the TIMA channel 2 registers initially controls the output on the PTE6/TCH2A pin. Writing to the TIMA channel 3 registers enables the TIMA channel 3 registers to synchronously control the output after the TIMA overflows. At each subsequent overflow, the TIMA channel registers (2 or 3) that control the output are the ones written to last. TASC2 controls and monitors the buffered output compare function, and TIMA channel 3 status and control register (TASC3) is unused. While the MS2B bit is set, the channel 3 pin, PTE7/TCH3A, 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. Writing to the active channel registers is the same as generating unbuffered output compares.*

11.4.4 Pulse-Width Modulation (PWM)

By using the toggle-on-overflow feature with an output compare channel, the TIMA can generate a PWM signal. The value in the TIMA counter modulo registers determines the period of the PWM signal. The channel pin toggles when the counter reaches the value in the TIMA counter modulo registers. The time between overflows is the period of the PWM signal.

As **Figure 11-3** shows, the output compare value in the TIMA channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIMA to clear the channel pin on output compare if the state of the PWM pulse is logic 1. Program the TIMA to set the pin if the state of the PWM pulse is logic 0.

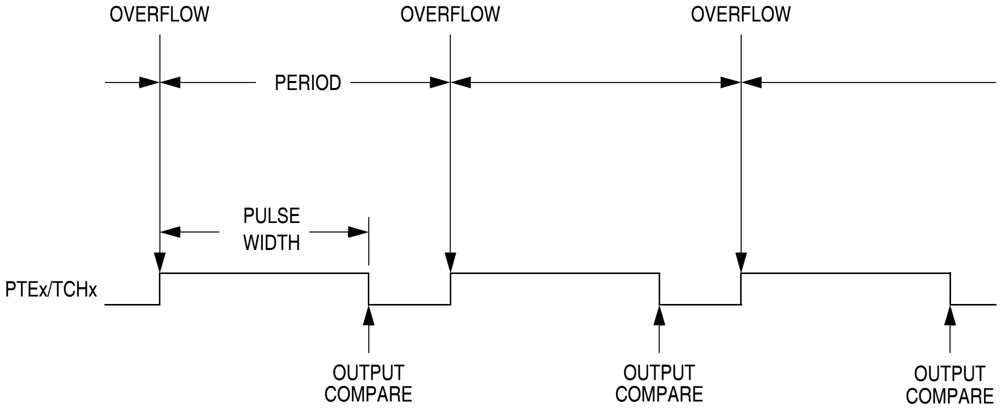


Figure 11-3. PWM Period and Pulse Width

The value in the TIMA 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 TIMA counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000 (see **11.8.1 TIMA Status and Control Register**).

The value in the TIMA 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 TIMA channel registers produces a duty cycle of 128/256 or 50 percent.

11.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in **11.4.4 Pulse-Width Modulation (PWM)**. The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the value currently in the TIMA channel registers.

An unsynchronized write to the TIMA 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 TIMA overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIMA may pass the new value before it is written to the TIMA channel registers.

Use this method 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 channel x TIMA overflow interrupts and write the new value in the TIMA overflow interrupt routine. The TIMA 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 percent 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.*

11.4.4.2 Buffered PWM Signal Generation

Channels 0 and 1 can be linked to form a buffered PWM channel whose output appears on the PTE4/TCH0A pin. The TIMA channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS0B bit in TIMA channel 0 status and control register (TASC0) links channel 0 and channel 1. The TIMA channel 0 registers initially control the pulse width on the PTE4/TCH0A pin. Writing to the TIMA channel 1 registers enables the TIMA channel 1 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIMA channel registers (0 or 1) that control the pulse width are the ones written to last. TASC0 controls and monitors the buffered PWM function, and TIMA channel 1 status and control register (TASC1) is unused. While the MS0B bit is set, the channel 1 pin, PTE5/TCH1A, is available as a general-purpose I/O pin.

Channels 2 and 3 can be linked to form a buffered PWM channel whose output appears on the PTE6/TCH2A pin. The TIMA channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS2B bit in TIMA channel 2 status and control register (TASC2) links channel 2 and channel 3. The TIMA channel 2 registers initially control the pulse width on the PTE6/TCH2A pin. Writing to the TIMA channel 3 registers enables the TIMA channel 3 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIMA channel registers (2 or 3) that control the pulse width are written to last. TASC2 controls and monitors the buffered PWM function, and TIMA channel 3 status and control register (TASC3) is unused. While the MS2B bit is set, the channel 3 pin, PTE7/TCH3A, 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. Writing to the active channel registers is the same as generating unbuffered PWM signals.*

11.4.4.3 PWM Initialization

To ensure correct operation when generating unbuffered or buffered PWM signals, use this initialization procedure:

1. In the TIMA status and control register (TASC):
   a. Stop the TIMA counter by setting the TIMA stop bit, TSTOP.
   b. Reset the TIMA counter by setting the TIMA reset bit, TRST.
2. In the TIMA counter modulo registers (TAMODH–TAMODL), write the value for the required PWM period.
3. In the TIMA channel x registers (TACHxH–TACHxL), write the value for the required pulse width.
4. In TIMA 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 11-2**.)
   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 11-2**.)

NOTE: *In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0 percent 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 TIMA status control register (TASC), clear the TIMA stop bit, TSTOP.

Setting MS0B links channels 0 and 1 and configures them for buffered PWM operation. The TIMA channel 0 registers (TACH0H–TACH0L) initially control the buffered PWM output. TIMA status control register 0 (TASC0) controls and monitors the PWM signal from the linked channels. MS0B takes priority over MS0A.

Setting MS2B links channels 2 and 3 and configures them for buffered PWM operation. The TIMA channel 2 registers (TACH2H–TACH2L) initially control the PWM output. TIMA status control register 2 (TASC2) controls and monitors the PWM signal from the linked channels. MS2B takes priority over MS2A.

Clearing the toggle-on-overflow bit, TOVx, inhibits output toggles on TIMA 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 percent duty cycle output.

Setting the channel x maximum duty cycle bit (CHxMAX) and clearing the TOVx bit generates a 100 percent duty cycle output. (See **11.8.4 TIMA Channel Status and Control Registers**.)

11.5 Interrupts

These TIMA sources can generate interrupt requests:

- TIMA overflow flag (TOF) — The TOF bit is set when the TIMA counter value rolls over to $0000 after matching the value in the TIMA counter modulo registers. The TIMA overflow interrupt enable bit, TOIE, enables TIMA overflow CPU interrupt requests. TOF and TOIE are in the TIMA status and control register.
- TIMA channel flags (CH3F–CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. Channel x TIMA CPU interrupt requests are controlled by the channel x interrupt enable bit, CHxIE.

11.6 Wait Mode

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

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

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

11.7 I/O Signals

Port E shares five of its pins with the TIMA:

- PTE3/TCLKA is an external clock input to the TIMA prescaler.
- The four TIMA channel I/O pins are PTE4/TCH0A, PTE5/TCH1A, PTE6/TCH2A, and PTE7/TCH3A.

11.7.1 TIMA Clock Pin (PTE3/TCLKA)

PTE3/TCLKA is an external clock input that can be the clock source for the TIMA counter instead of the prescaled internal bus clock. Select the PTE3/TCLKA input by writing logic 1s to the three prescaler select bits, PS[2:0]. See **11.8.1 TIMA Status and Control Register**. The minimum TCLK pulse width, $TCLK_{LMIN}$ or $TCLK_{HMIN}$, is:

$$\frac{1}{\text{bus frequency}} + t_{SU}$$

The maximum TCLK frequency is the least: 4 MHz or bus frequency ÷ 2.

PTE3/TCLKA is available as a general-purpose I/O pin or ADC channel when not used as the TIMA clock input. When the PTE3/TCLKA pin is the TIMA clock input, it is an input regardless of the state of the DDRE3 bit in data direction register E.

11.7.2 TIMA Channel I/O Pins (PTE4/TCH0A–PTE7/TCH3A)

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. PTE2/TCH0 and PTE4/TCH2 can be configured as buffered output compare or buffered PWM pins.

11.8 I/O Registers

These input/output (I/O) registers control and monitor TIMA operation:

- TIMA status and control register (TASC)
- TIMA control registers (TACNTH–TACNTL)
- TIMA counter modulo registers (TAMODH–TAMODL)
- TIMA channel status and control registers (TASC0, TASC1, TASC2, and TASC3)
- TIMA channel registers (TACH0H–TACH0L, TACH1H–TACH1L, TACH2H–TACH2L, and TACH3H–TACH3L)

11.8.1 TIMA Status and Control Register

The TIMA status and control register:

- Enables TIMA overflow interrupts
- Flags TIMA overflows
- Stops the TIMA counter
- Resets the TIMA counter
- Prescales the TIMA counter clock

Address: $000E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
Write:	0			TRST	R			
Reset:	0	0	1	0	0	0	0	0

R = Reserved

Figure 11-4. TIMA Status and Control Register (TASC)

TOF — TIMA Overflow Flag

This read/write flag is set when the TIMA counter resets to $0000 after reaching the modulo value programmed in the TIMA counter modulo registers. Clear TOF by reading the TIMA status and control register when TOF is set and then writing a logic 0 to TOF. If another TIMA 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 = TIMA counter has reached modulo value.
0 = TIMA counter has not reached modulo value.

TOIE — TIMA Overflow Interrupt Enable Bit

This read/write bit enables TIMA overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIMA overflow interrupts enabled
0 = TIMA overflow interrupts disabled

TSTOP — TIMA Stop Bit

This read/write bit stops the TIMA counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIMA counter until software clears the TSTOP bit.

1 = TIMA counter stopped
0 = TIMA counter active

NOTE: *Do not set the TSTOP bit before entering wait mode if the TIMA is required to exit wait mode. Also when the TSTOP bit is set and the timer is configured for input capture operation, input captures are inhibited until the TSTOP bit is cleared.*

TRST — TIMA Reset Bit

Setting this write-only bit resets the TIMA counter and the TIMA prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIMA counter is reset and always reads as logic 0. Reset clears the TRST bit.

1 = Prescaler and TIMA counter cleared
0 = No effect

NOTE: *Setting the TSTOP and TRST bits simultaneously stops the TIMA counter at a value of $0000.*

PS[2:0] — Prescaler Select Bits

These read/write bits select either the PTD6/ATD14/TCLK pin or one of the seven prescaler outputs as the input to the TIMA counter as **Table 11-1** shows. Reset clears the PS[2:0] bits.

Table 11-1. Prescaler Selection

PS[2:0]	TIMA Clock Source
000	Internal bus clock ÷1
001	Internal bus clock ÷ 2
010	Internal bus clock ÷ 4
011	Internal bus clock ÷ 8
100	Internal bus clock ÷ 16
101	Internal bus clock ÷ 32
110	Internal bus clock ÷ 64
111	PTE3/TCLKA

11.8.2 TIMA Counter Registers

The two read-only TIMA counter registers contain the high and low bytes of the value in the TIMA counter. Reading the high byte (TACNTH) latches the contents of the low byte (TACNTL) into a buffer. Subsequent reads of TACNTH do not affect the latched TACNTL value until TACNTL is read. Reset clears the TIMA counter registers. Setting the TIMA reset bit (TRST) also clears the TIMA counter registers.

NOTE: *If TACNTH is read during a break interrupt, be sure to unlatch TACNTL by reading TACNTL before exiting the break interrupt. Otherwise, TACNTL retains the value latched during the break.*

Register Name and Address: TACNTH — $000F

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

Register Name and Address: TACNTL — $0010

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

R = Reserved

Figure 11-5. TIMA Counter Registers (TACNTH and TACNTL)

11.8.3 TIMA Counter Modulo Registers

The read/write TIMA modulo registers contain the modulo value for the TIMA counter. When the TIMA counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIMA counter resumes counting from $0000 at the next clock. Writing to the high byte (TAMODH) inhibits the TOF bit and overflow interrupts until the low byte (TAMODL) is written. Reset sets the TIMA counter modulo registers.

Register Name and Address: TAMODH — $0011

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

Register Name and Address: TAMODL — $0012

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

Figure 11-6. TIMA Counter Modulo Registers (TAMODH and TAMODL)

NOTE: *Reset the TIMA counter before writing to the TIMA counter modulo registers.*

11.8.4 TIMA Channel Status and Control Registers

Each of the TIMA 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 TIMA overflow
- Selects 100 percent PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

Register Name and Address: TASC0 — $0013

	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

Register Name and Address: TASC1 — $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
Write:	0		R					
Reset:	0	0	0	0	0	0	0	0

Register Name and Address: TASC2 — $0019

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	TOV2	CH2MAX
Write:	0							
Reset:	0	0	0	0	0	0	0	0

Register Name and Address: TASC3 — $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH3F	CH3IE	0	MS3A	ELS3B	ELS3A	TOV3	CH3MAX
Write:	0		R					
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 11-7. TIMA Channel Status and Control Registers (TASC0–TASC3)

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 TIMA counter registers matches the value in the TIMA channel x registers.

When CHxIE = 0, clear CHxF by reading TIMA 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 TIMA CPU interrupts 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 TIMA channel 0 and TIMA channel 2 status and control registers.

Setting MS0B disables the channel 1 status and control register and reverts TCH1A pin to general-purpose I/O.

Setting MS2B disables the channel 3 status and control register and reverts TCH3A pin 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 11-2**.

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 TCHxA pin once PWM, input capture, or output compare operation is enabled. See **Table 11-2**. 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 TIMA status and control register (TASC).*

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 E, and pin PTEx/TCHxA is available as a general-purpose I/O pin. However, channel x is at a state determined by these bits and becomes transparent to the respective pin when PWM, input capture, or output compare mode is enabled. **Table 11-2** shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

Table 11-2. Mode, Edge, and Level Selection

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output preset	Pin under port control; initialize timer output level high
X1	00		Pin under port control; initialize timer 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 TIMA channel register for input capture operation, make sure that the PTEx/TACHx 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 TIMA 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 TIMA counter overflow.
0 = Channel x pin does not toggle on TIMA counter overflow.

NOTE: *When TOVx is set, a TIMA 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 0, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100 percent. As **Figure 11-8** shows, the CHxMAX bit takes effect in the cycle after it is set or cleared. Also, TOVx bit takes effect in the cycle in which it is set or cleared. The output stays at the 100 percent duty cycle level until the cycle after CHxMAX is cleared.

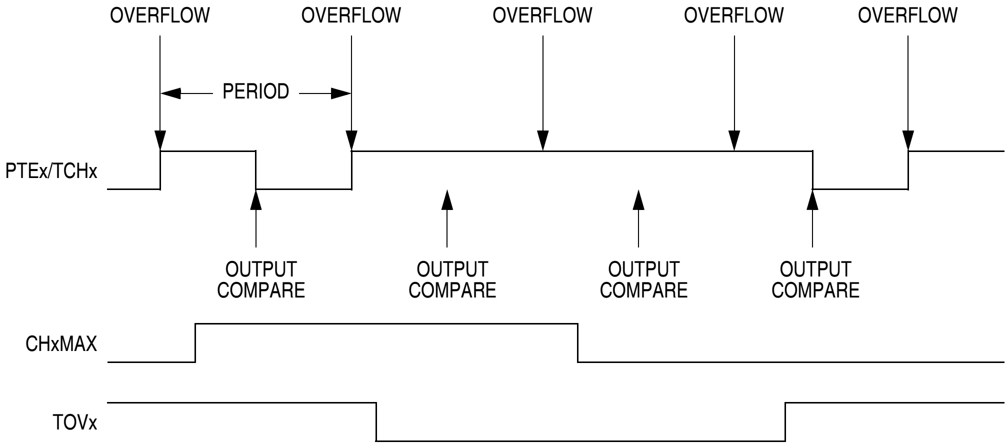


Figure 11-8. CHxMAX Latency

11.8.5 TIMA Channel Registers

These read/write registers contain the captured TIMA counter value of the input capture function or the output compare value of the output compare function. The state of the TIMA channel registers after reset is unknown.

In input capture mode (MSxB:MSxA = 0:0), reading the high byte of the TIMA channel x registers (TACHxH) inhibits input captures until the low byte (TACHxL) is read.

In output compare mode (MSxB:MSxA ≠ 0:0), writing to the high byte of the TIMA channel x registers (TACHxH) inhibits output compares until the low byte (TACHxL) is written.

Register Name and Address: TACH0H — $0014

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

Register Name and Address: TACH0L — $0015

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Register Name and Address: TACH1H — $0017

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

Register Name and Address: TACH1L — $0018

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Register Name and Address: TACH2H — $001A

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

Figure 11-9. TIMA Channel Registers (TACH0H/L–TACH3H/L)

Register Name and Address: TACH2L — $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Register Name and Address: TACH3H — $001D

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

Register Name and Address: TACH3L — $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Figure 11-9. TIMA Channel Registers (TACH0H/L–TACH3H/L) (Continued)

Section 12. Timer Interface B (TIMB)

12.1 Contents

12.2 Introduction

This section describes the timer interface module B (TIMB). The TIMB is a 2-channel timer that provides a timing reference with input capture, output compare, and pulse-width modulation functions. **Figure 12-1** is a block diagram of the TIMB.

NOTE: *The TIMB module is not available in the 56-pin shrink dual in-line package (SDIP).*

12.3 Features

Features of the TIMB 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 TIMB clock input:
  - 7-frequency internal bus clock prescaler selection
  - External TIMB clock input (4-MHz maximum frequency)
- Free-running or modulo up-count operation
- Toggle any channel pin on overflow
- TIMB counter stop and reset bits

12.4 Functional Description

Figure 12-1 shows the TIMB structure. The central component of the TIMB is the 16-bit TIMB counter that can operate as a free-running counter or a modulo up-counter. The TIMB counter provides the timing reference for the input capture and output compare functions. The TIMB counter modulo registers, TBMODH–TBMODL, control the modulo value of the TIMB counter. Software can read the TIMB counter value at any time without affecting the counting sequence.

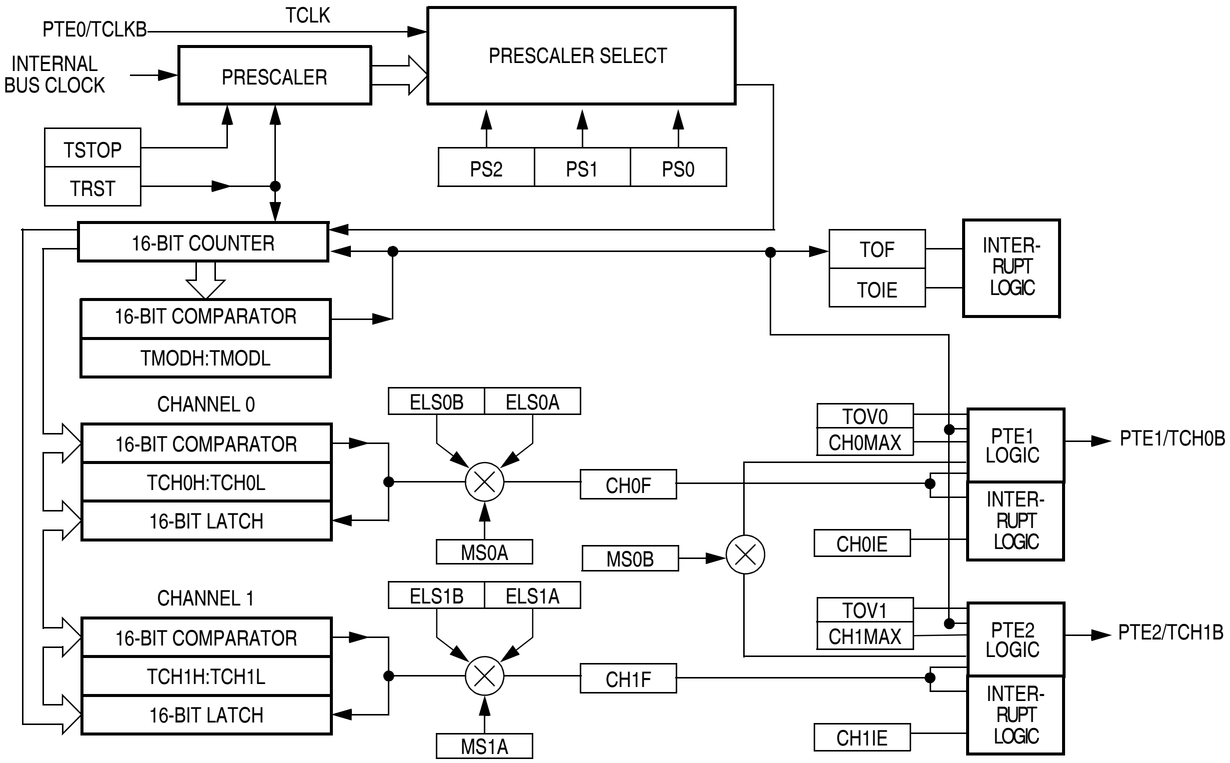


Figure 12-1. TIMB Block Diagram

The two TIMB channels are programmable independently as input capture or output compare channels.

NOTE: *The TIMB module is not available in the 56-pin SDIP.*

12.4.1 TIMB Counter Prescaler

The TIMB clock source can be one of the seven prescaler outputs or the TIMB clock pin, PTD4/ATD12. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIMB status and control register select the TIMB clock source.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0051	TIMB Status/Control Register (TBSC) See page 244.	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST	R			
		Reset:	0	0	1	0	0	0	0	0
$0052	TIMB Counter Register High (TBCNTH) See page 247.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0053	TIMB Counter Register Low (TBCNTL) See page 247.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$0054	TIMB Counter Modulo Register High (TBMODH) See page 248.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0055	TIMB Counter Modulo Register Low (TBMODL) See page 248.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0056	TIMB Channel 0 Status/Control Register (TBSC0) See page 249.	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0057	TIMB Channel 0 Register High (TBCH0H) See page 254.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0058	TIMB Channel 0 Register Low (TBCH0L) See page 254.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0059	TIMB Channel 1 Status/Control Register (TBSC1) See page 249.	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0		R					
		Reset:	0	0	0	0	0	0	0	0
$005A	TIMB Channel 1 Register High (TBCH1H) See page 254.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$005B	TIMB Channel 1 Register Low (TBCH1L) See page 254.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

R = Reserved

Figure 12-2. TIMB I/O Register Summary

12.4.2 Input Capture

An input capture function has three basic parts:

- Edge select logic
- Input capture latch
- 16-bit counter

Two 8-bit registers, which make up the 16-bit input capture register, are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The polarity of the active edge is programmable. The level transition which triggers the counter transfer is defined by the corresponding input edge bits (ELSxB and ELSxA in TBSC0–TBSC1 control registers with x referring to the active channel number). When an active edge occurs on the pin of an input capture channel, the TIMB latches the contents of the TIMB counter into the TIMB channel registers, TCHxH–TCHxL. Input captures can generate TIMB CPU interrupt requests. Software can determine that an input capture event has occurred by enabling input capture interrupts or by polling the status flag bit.

The result obtained by an input capture will be two more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization.

The free-running counter contents are transferred to the TIMB channel status and control register (TBCHxH–TBCHxL, see **12.8.5 TIMB Channel Registers**) on each proper signal transition regardless of whether the TIMB channel flag (CH0F–CH1F in TBSC0–TBSC1 registers) is set or clear. When the status flag is set, a CPU interrupt is generated if enabled. The value of the count latched or “captured” is the time of the event. Because this value is stored in the input capture register 2 bus cycles after the actual event occurs, user software can respond to this event at a later time and determine the actual time of the event. However, this must be done prior to another input capture on the same pin; otherwise, the previous time value will be lost.

By recording the times for successive edges on an incoming signal, software can determine the period and/or pulse width of the signal. To measure a period, two successive edges of the same polarity are captured. To measure a pulse width, two alternate polarity edges are captured. Software should track the overflows at the 16-bit module counter to extend its range.

Another use for the input capture function is to establish a time reference. In this case, an input capture function is used in conjunction with an output compare function. For example, to activate an output signal a specified number of clock cycles after detecting an input event (edge), use the input capture function to record the time at which the edge occurred. A number corresponding to the desired delay is added to this captured value and stored to an output compare register (see **12.8.5 TIMB Channel Registers**). Because both input captures and output compares are referenced to the same 16-bit modulo counter, the delay can be controlled to the resolution of the counter independent of software latencies.

Reset does not affect the contents of the input capture channel register (TBCHxH–TBCHxL).

12.4.3 Output Compare

With the output compare function, the TIMB 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 TIMB can set, clear, or toggle the channel pin. Output compares can generate TIMB CPU interrupt requests.

12.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in **12.4.3 Output Compare**. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIMB channel registers.

An unsynchronized write to the TIMB 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 TIMB overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIMB may pass the new value before it is written.

Use this method 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 channel x TIMB overflow interrupts and write the new value in the TIMB overflow interrupt routine. The TIMB 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.

12.4.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the PTE1/TCH0B pin. The TIMB channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIMB channel 0 status and control register (TBSC0) links channel 0 and channel 1. The output compare value in the TIMB channel 0 registers initially controls the output on the PTE1/TCH0B pin. Writing to the TIMB channel 1 registers enables the TIMB channel 1 registers to synchronously control the output after the TIMB overflows. At each subsequent overflow, the TIMB 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 TIMB channel 1 status and control register (TBSC1) is unused. While the MS0B bit is set, the channel 1 pin, PTE2/TCH1B, 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. Writing to the active channel registers is the same as generating unbuffered output compares.*

12.4.4 Pulse-Width Modulation (PWM)

By using the toggle-on-overflow feature with an output compare channel, the TIMB can generate a PWM signal. The value in the TIMB counter modulo registers determines the period of the PWM signal. The channel pin toggles when the counter reaches the value in the TIMB counter modulo registers. The time between overflows is the period of the PWM signal.

As **Figure 12-3** shows, the output compare value in the TIMB channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIMB to clear the channel pin on output compare if the state of the PWM pulse is logic 1. Program the TIMB to set the pin if the state of the PWM pulse is logic 0.

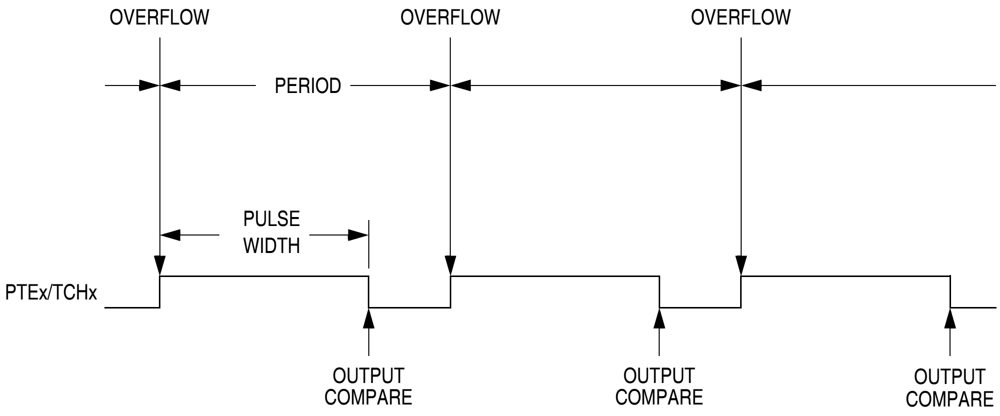


Figure 12-3. PWM Period and Pulse Width

The value in the TIMB 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 TIMB counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000 (see **12.8.1 TIMB Status and Control Register**).

The value in the TIMB 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 TIMB channel registers produces a duty cycle of 128/256 or 50 percent.

12.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in **12.4.4 Pulse-Width Modulation (PWM)**. The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the value currently in the TIMB channel registers.

An unsynchronized write to the TIMB 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 TIMB overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIMB may pass the new value before it is written to the TIMB channel registers.

Use this method 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 channel x TIMB overflow interrupts and write the new value in the TIMB overflow interrupt routine. The TIMB 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 percent 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.*

12.4.4.2 Buffered PWM Signal Generation

Channels 0 and 1 can be linked to form a buffered PWM channel whose output appears on the PTE1/TCH0B pin. The TIMB channel registers of the linked pair alternately control the pulse width of the output.

Setting the MS0B bit in TIMB channel 0 status and control register (TBSC0) links channel 0 and channel 1. The TIMB channel 0 registers initially control the pulse width on the PTE1/TCH0B pin. Writing to the TIMB channel 1 registers enables the TIMB channel 1 registers to synchronously control the pulse width at the beginning of the next PWM period. At each subsequent overflow, the TIMB channel registers (0 or 1) that control the pulse width are the ones written to last. TBSC0 controls and monitors the buffered PWM function, and TIMB channel 1 status and control register (TBSC1) is unused. While the MS0B bit is set, the channel 1 pin, PTE2/TCH1B, 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. Writing to the active channel registers is the same as generating unbuffered PWM signals.*

12.4.4.3 PWM Initialization

To ensure correct operation when generating unbuffered or buffered PWM signals, use this initialization procedure:

1. In the TIMB status and control register (TBSC):
    a. Stop the TIMB counter by setting the TIMB stop bit, TSTOP.
    b. Reset the TIMB counter by setting the TIMB reset bit, TRST.
2. In the TIMB counter modulo registers (TBMODH–TBMODL), write the value for the required PWM period.
3. In the TIMB channel x registers (TBCHxH–TBCHxL), write the value for the required pulse width.
4. In TIMB channel x status and control register (TBSCx):
    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 12-2**.)
    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 12-2**.)

NOTE: *In PWM signal generation, do not program the PWM channel to toggle on output compare. Toggling on output compare prevents reliable 0 percent 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 TIMB status control register (TBSC), clear the TIMB stop bit, TSTOP.

Setting MS0B links channels 0 and 1 and configures them for buffered PWM operation. The TIMB channel 0 registers (TBCH0H–TBCH0L) initially control the buffered PWM output. TIMB status control register 0 (TBSC0) controls and monitors the PWM signal from the linked channels. MS0B takes priority over MS0A.

Clearing the toggle-on-overflow bit, TOVx, inhibits output toggles on TIMB 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 percent duty cycle output.

Setting the channel x maximum duty cycle bit (CHxMAX) and clearing the TOVx bit generates a 100 percent duty cycle output. (See **12.8.4 TIMB Channel Status and Control Registers**.)

12.5 Interrupts

These TIMB sources can generate interrupt requests:

- TIMB overflow flag (TOF) — The TOF bit is set when the TIMB counter value rolls over to $0000 after matching the value in the TIMB counter modulo registers. The TIMB overflow interrupt enable bit, TOIE, enables TIMB overflow CPU interrupt requests. TOF and TOIE are in the TIMB status and control register.
- TIMB channel flags (CH1F–CH0F) — The CHxF bit is set when an input capture or output compare occurs on channel x. Channel x TIMB CPU interrupt requests are controlled by the channel x interrupt enable bit, CHxIE.

12.6 Wait Mode

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

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

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

12.7 I/O Signals

Port E shares three of its pins with the TIMB:

- PTD4/ATD12 is an external clock input to the TIMB prescaler.
- The two TIMB channel I/O pins are PTE1/TCH0B and PTE2/TCH1B.

12.7.1 TIMB Clock Pin (PTD4/ATD12)

PTD4/ATD12 is an external clock input that can be the clock source for the TIMB counter instead of the prescaled internal bus clock. Select the PTD4/ATD12 input by writing logic 1s to the three prescaler select bits, PS[2:0]. See **12.8.1 TIMB Status and Control Register**. The minimum TCLK pulse width, $TCLK_{LMIN}$ or $TCLK_{HMIN}$, is:

$$\frac{1}{\text{bus frequency}} + t_{SU}$$

The maximum TCLK frequency is the least: 4 MHz or bus frequency ÷ 2.

PTD4/ATD12 is available as a general-purpose I/O pin or ADC channel when not used as the TIMB clock input. When the PTD4/ATD12 pin is the TIMB clock input, it is an input regardless of the state of the DDRE0 bit in data direction register E.

12.7.2 TIMB Channel I/O Pins (PTE1/TCH0B–PTE2/TCH1B)

Each channel I/O pin is programmable independently as an input capture pin or an output compare pin. PTE1/TCH0B and PTE2/TCH1B can be configured as buffered output compare or buffered PWM pins.

12.8 I/O Registers

These input/output (I/O) registers control and monitor TIMB operation:

- TIMB status and control register (TBSC)
- TIMB control registers (TBCNTH–TBCNTL)
- TIMB counter modulo registers (TBMODH–TBMODL)
- TIMB channel status and control registers (TBSC0 and TBSC1)
- TIMB channel registers (TBCH0H–TBCH0L and TBCH1H–TBCH1L)

12.8.1 TIMB Status and Control Register

The TIMB status and control register:

- Enables TIMB overflow interrupts
- Flags TIMB overflows
- Stops the TIMB counter
- Resets the TIMB counter
- Prescales the TIMB counter clock

Address: $0051

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
Write:	0			TRST	R			
Reset:	0	0	1	0	0	0	0	0

R = Reserved

Figure 12-4. TIMB Status and Control Register (TBSC)

TOF — TIMB Overflow Flag

This read/write flag is set when the TIMB counter resets to $0000 after reaching the modulo value programmed in the TIMB counter modulo registers. Clear TOF by reading the TIMB status and control register when TOF is set and then writing a logic 0 to TOF. If another TIMB 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 = TIMB counter has reached modulo value.
0 = TIMB counter has not reached modulo value.

TOIE — TIMB Overflow Interrupt Enable Bit

This read/write bit enables TIMB overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIMB overflow interrupts enabled
0 = TIMB overflow interrupts disabled

TSTOP — TIMB Stop Bit

This read/write bit stops the TIMB counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIMB counter until software clears the TSTOP bit.

1 = TIMB counter stopped
0 = TIMB counter active

NOTE: *Do not set the TSTOP bit before entering wait mode if the TIMB is required to exit wait mode. Also, when the TSTOP bit is set and the timer is configured for input capture operation, input captures are inhibited until TSTOP is cleared.*

TRST — TIMB Reset Bit

Setting this write-only bit resets the TIMB counter and the TIMB prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIMB counter is reset and always reads as logic 0. Reset clears the TRST bit.

1 = Prescaler and TIMB counter cleared
0 = No effect

NOTE: *Setting the TSTOP and TRST bits simultaneously stops the TIMB counter at a value of $0000.*

PS[2:0] — Prescaler Select Bits

These read/write bits select either the PTD4/ATD12 pin or one of the seven prescaler outputs as the input to the TIMB counter as **Table 12-1** shows. Reset clears the PS[2:0] bits.

Table 12-1. Prescaler Selection

PS[2:0]	TIMB Clock Source
000	Internal bus clock ÷1
001	Internal bus clock ÷ 2
010	Internal bus clock ÷ 4
011	Internal bus clock ÷ 8
100	Internal bus clock ÷ 16
101	Internal bus clock ÷ 32
110	Internal bus clock ÷ 64
111	PTE0/TCLKB

12.8.2 TIMB Counter Registers

The two read-only TIMB counter registers contain the high and low bytes of the value in the TIMB counter. Reading the high byte (TBCNTH) latches the contents of the low byte (TBCNTL) into a buffer. Subsequent reads of TBCNTH do not affect the latched TBCNTL value until TBCNTL is read. Reset clears the TIMB counter registers. Setting the TIMB reset bit (TRST) also clears the TIMB counter registers.

NOTE: *If TBCNTH is read during a break interrupt, be sure to unlatch TBCNTL by reading TBCNTL before exiting the break interrupt. Otherwise, TBCNTL retains the value latched during the break.*

Register Name and Address: TBCNTH — $0052

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

Register Name and Address: TBCNTL — $0053

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

R = Reserved

Figure 12-5. TIMB Counter Registers (TBCNTH and TBCNTL)

12.8.3 TIMB Counter Modulo Registers

The read/write TIMB modulo registers contain the modulo value for the TIMB counter. When the TIMB counter reaches the modulo value, the overflow flag (TOF) becomes set, and the TIMB counter resumes counting from $0000 at the next clock. Writing to the high byte (TBMODH) inhibits the TOF bit and overflow interrupts until the low byte (TBMODL) is written. Reset sets the TIMB counter modulo registers.

Register Name and Address: TBMODH — $0054

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

Register Name and Address: TBMODL — $0055

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

Figure 12-6. TIMB Counter Modulo Registers (TBMODH and TBMODL)

NOTE: *Reset the TIMB counter before writing to the TIMB counter modulo registers.*

12.8.4 TIMB Channel Status and Control Registers

Each of the TIMB 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 TIMB overflow
- Selects 100 percent PWM duty cycle
- Selects buffered or unbuffered output compare/PWM operation

Register Name and Address: TBSC0 — $0056

	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

Register Name and Address: TBSC1 — $0059

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
Write:	0		R					
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 12-7. TIMB Channel Status and Control Registers (TBSC0–TBSC1)

CHxF — Channel x Flag

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 TIMB counter registers matches the value in the TIMB channel x registers.

When CHxIE = 0, clear CHxF by reading TIMB 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 TIMB CPU interrupts 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 TIMB channel 0.

Setting MS0B disables the channel 1 status and control register and reverts TCH1B 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 12-2**.

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 once PWM, input capture, or output compare operation is enabled. See **Table 12-2**. 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 TIMB status and control register (TBSC).*

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 E, and pin PTEx/TCHxB is available as a general-purpose I/O pin. However, channel x is at a state determined by these bits and becomes transparent to the respective pin when PWM, input capture, or output compare mode is enabled. **Table 12-2** shows how ELSxB and ELSxA work. Reset clears the ELSxB and ELSxA bits.

NOTE: *Before enabling a TIMB channel register for input capture operation, make sure that the PTEx/TBCHx 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 TIMB 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 TIMB counter overflow.
0 = Channel x pin does not toggle on TIMB counter overflow.

Table 12-2. Mode, Edge, and Level Selection

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output preset	Pin under port control; initialize timer output level high
X1	00		Pin under port control; initialize timer 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: *When TOVx is set, a TIMB 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 0, setting the CHxMAX bit forces the duty cycle of buffered and unbuffered PWM signals to 100 percent. As **Figure 12-8** shows, the CHxMAX bit takes effect in the cycle after it is set or cleared. The output stays at the 100 percent duty cycle level until the cycle after CHxMAX is cleared.

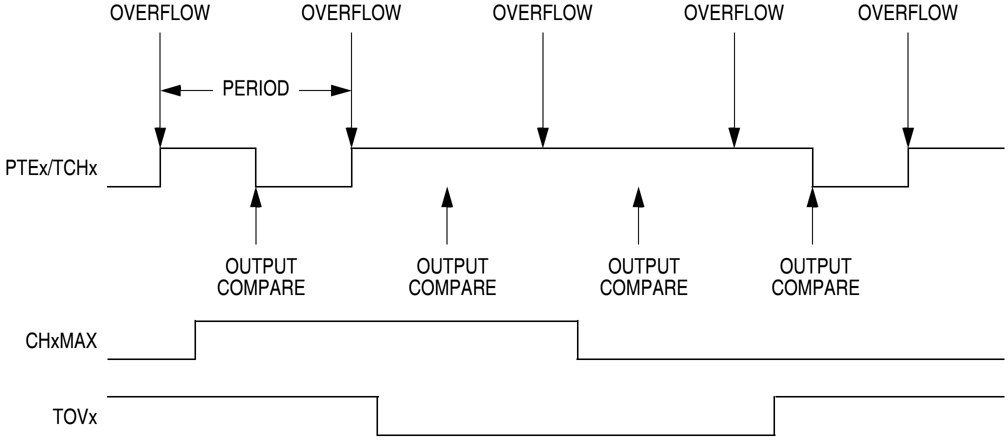


Figure 12-8. CHxMAX Latency

12.8.5 TIMB Channel Registers

These read/write registers contain the captured TIMB counter value of the input capture function or the output compare value of the output compare function. The state of the TIMB channel registers after reset is unknown.

In input capture mode (MSxB–MSxA = 0:0), reading the high byte of the TIMB channel x registers (TBCHxH) inhibits input captures until the low byte (TBCHxL) is read.

In output compare mode (MSxB–MSxA ≠ 0:0), writing to the high byte of the TIMB channel x registers (TBCHxH) inhibits output compares until the low byte (TBCHxL) is written.

Register Name and Address: TBCH0H — $0057

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

Register Name and Address: TBCH0L — $0058

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Register Name and Address: TBCH1H — $005A

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

Register Name and Address: TBCH1L — $005B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Figure 12-9. TIMB Channel Registers (TBCH0H/L–TBCH1H/L)

Section 13. Serial Peripheral Interface Module (SPI)

13.1 Contents

13.2 Introduction

The serial peripheral interface (SPI) module allows full-duplex, synchronous, serial communications with peripheral devices.

13.3 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 with central processor unit (CPU) service:
  - 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

13.4 Pin Name Conventions

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

- $\overline{\text{SS}}$, slave select
- SPSCK, SPI serial clock
- MOSI, master out/slave in
- MISO, master in/slave out

SPI pins are shared by parallel I/O ports or have alternate functions. The full name of an SPI pin reflects the name of the shared port pin or the name of an alternate pin function. The generic pin names appear in the text that follows. **Table 13-1** shows the full names of the SPI I/O pins.

Table 13-1. Pin Name Conventions

Generic Pin Names:	**MISO**	**MOSI**	**SPSCK**	**$\overline{\text{SS}}$**
Full Pin Names:	PTF3/MISO	PTF2/MOSI	PTF0/SPSCK	PTF1/$\overline{\text{SS}}$

13.5 Functional Description

Figure 13-1 shows the structure of the SPI module and **Figure 13-2** shows the locations and contents of the SPI I/O registers.

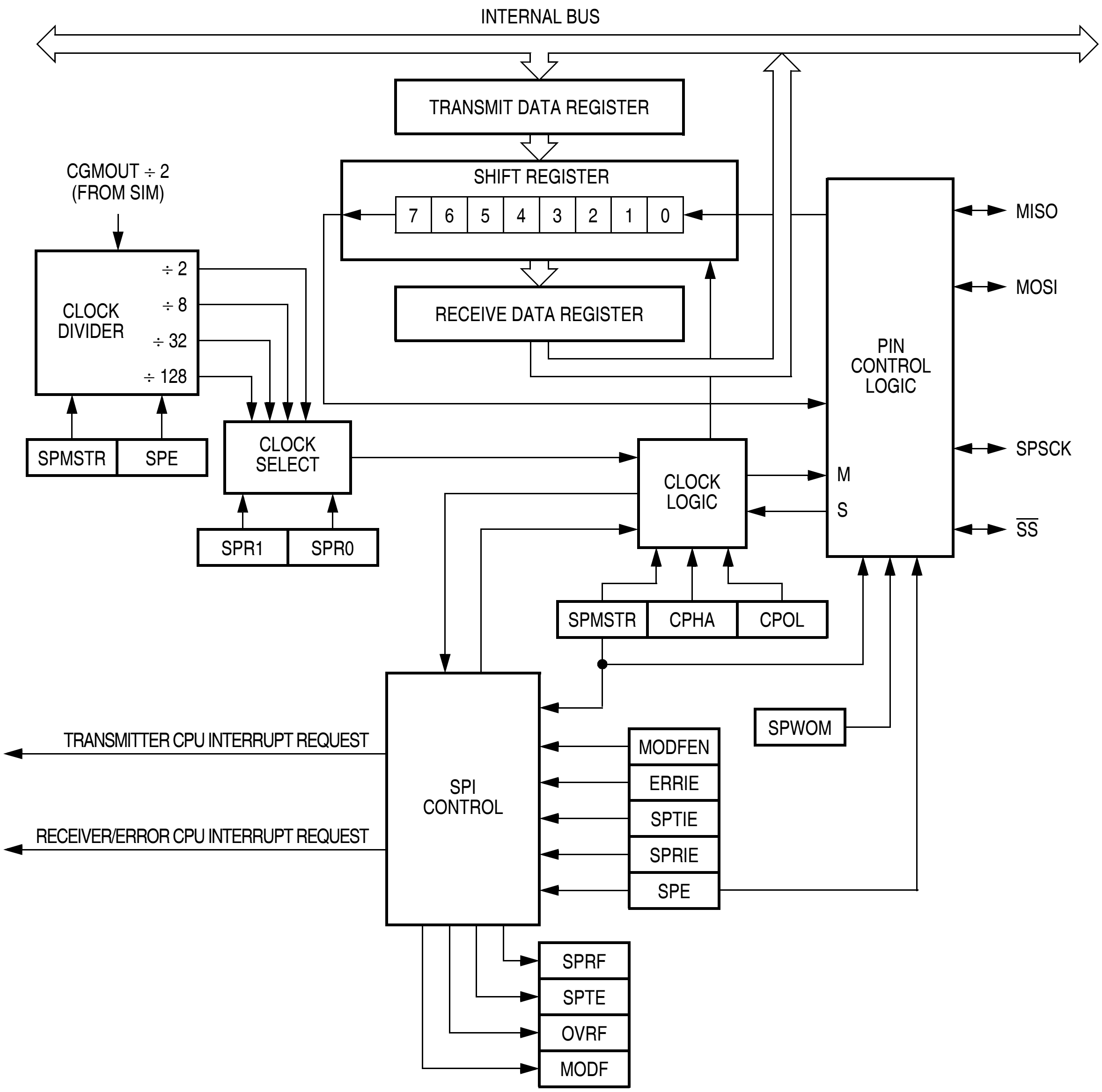


Figure 13-1. SPI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0044	SPI Control Register (SPCR) See page 280.	Read:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
		Write:								
		Reset:	0	0	1	0	1	0	0	0
$0045	SPI Status and Control Register (SPSCR) See page 282.	Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
		Write:	R		R	R	R			
		Reset:	0	0	0	0	1	0	0	0
$0046	SPI Data Register (SPDR) See page 285.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							

R = Reserved

Figure 13-2. SPI I/O Register Summary

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. All SPI interrupts can be serviced by the CPU.

13.5.1 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* ***13.13.1 SPI Control Register***.

Only a master SPI module can initiate transmissions. Software begins the transmission from a master SPI module by writing to the SPI 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 13-3**.

The SPR1 and SPR0 bits control the baud rate generator and determine the speed of the shift register. See **13.13.2 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.

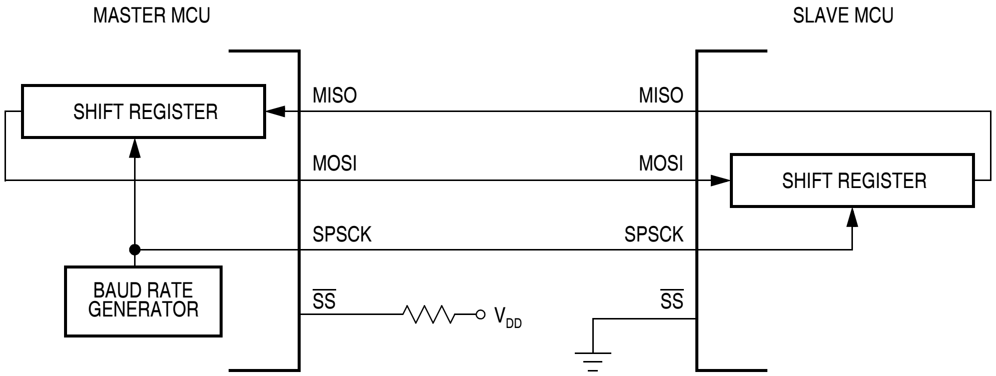


Figure 13-3. Full-Duplex Master-Slave Connections

13.5.2 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 **13.7.2 Mode Fault Error**.

In a slave SPI module, data enters the shift register under the control of the serial clock from the master SPI module. After a byte enters the shift register of a slave SPI, it transfers to the receive data register, and the SPRF bit is set. To prevent an overflow condition, slave software then must read the receive data register before another full byte enters the shift register.

The maximum frequency of the SPSCK for an SPI configured as a slave is the bus clock speed (which is twice as fast as the fastest master SPSCK clock that can be generated). The frequency of the SPSCK for an SPI configured as a slave does not have to correspond to any SPI baud rate. The baud rate only controls the speed of the SPSCK generated by an SPI configured as a master. Therefore, the frequency of the SPSCK for an SPI configured as a slave can be any frequency less than or equal to the bus speed.

When the master SPI starts a transmission, the data in the slave shift register begins shifting out on the MISO pin. The slave can load its shift register with a new byte for the next transmission by writing to its transmit data register. The slave must write to its transmit data register at least one bus cycle before the master starts the next transmission. Otherwise, the byte already in the slave shift register shifts out on the MISO pin. Data written to the slave shift register during a transmission remains in a buffer until the end of the transmission.

When the clock phase bit (CPHA) is set, the first edge of SPSCK starts a transmission. When CPHA is clear, the falling edge of $\overline{\text{SS}}$ starts a transmission. See **13.6 Transmission Formats**.

NOTE: *If the write to the data register is late, the SPI transmits the data already in the shift register from the previous transmission.*

SPSCK must be in the proper idle state before the slave is enabled to prevent SPSCK from appearing as a clock edge.

13.6 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.

13.6.1 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).*

13.6.2 Transmission Format When CPHA = 0

Figure 13-4 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 ($\overline{\text{SS}}$) 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 **13.7.2 Mode Fault Error**.) When CPHA = 0, the first SPSCK edge is the MSB capture strobe. Therefore, the slave must begin driving its data before the first SPSCK edge, and a falling edge on the $\overline{\text{SS}}$ pin is used to start the slave data transmission. The slave's $\overline{\text{SS}}$ pin must be toggled back to high and then low again between each byte transmitted as shown in **Figure 13-5**.

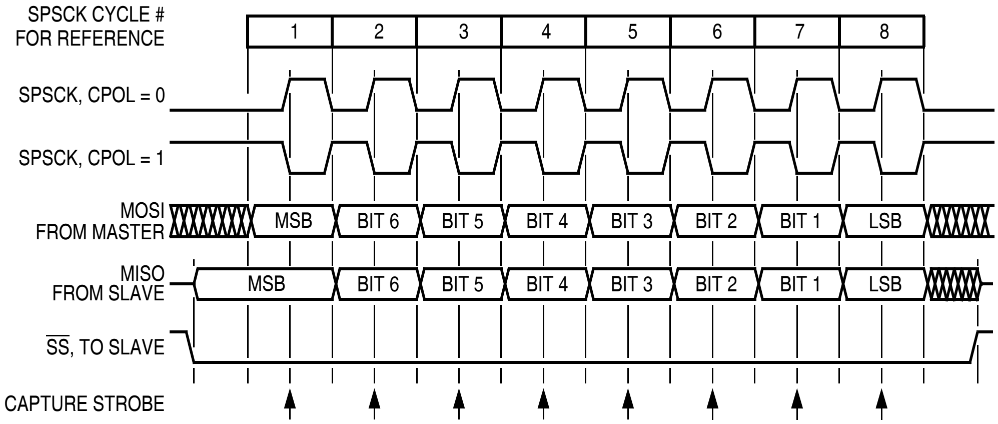


Figure 13-4. Transmission Format (CPHA = 0)

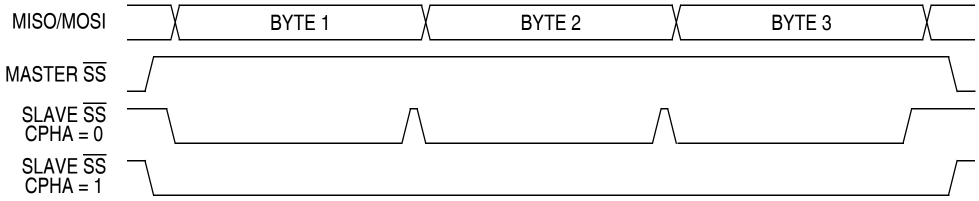


Figure 13-5. CPHA/$\overline{\text{SS}}$ Timing

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.

13.6.3 Transmission Format When CPHA = 1

Figure 13-6 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 **13.7.2 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.

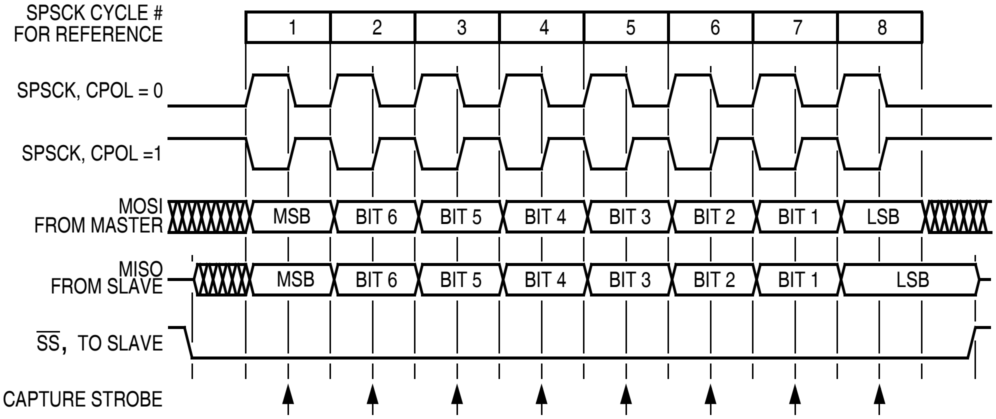


Figure 13-6. Transmission Format (CPHA = 1)

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.

13.6.4 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 13-7**. 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 13-7**. 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.

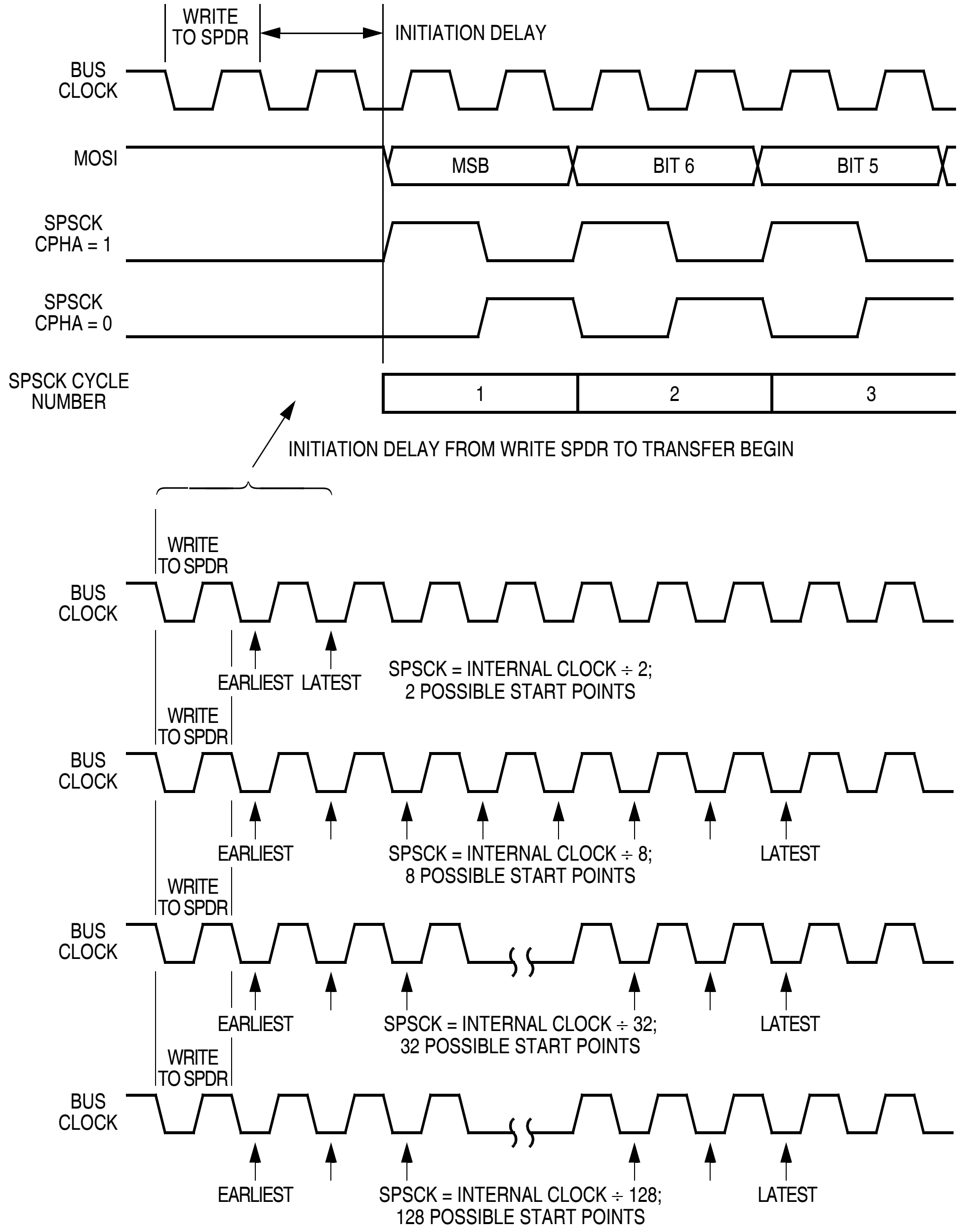


Figure 13-7. Transmission Start Delay (Master)

13.7 Error Conditions

These 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.

13.7.1 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. 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. MODF and OVRF can generate a receiver/error CPU interrupt request. See **Figure 13-10**. 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 13-8** shows how it is possible to miss an overflow. The first part of **Figure 13-8** shows how it is possible to read the SPSCR and SPDR to clear the SPRF without problems. However, as illustrated by the second transmission example, the OVRF bit can be set in between the time that SPSCR and SPDR are read.

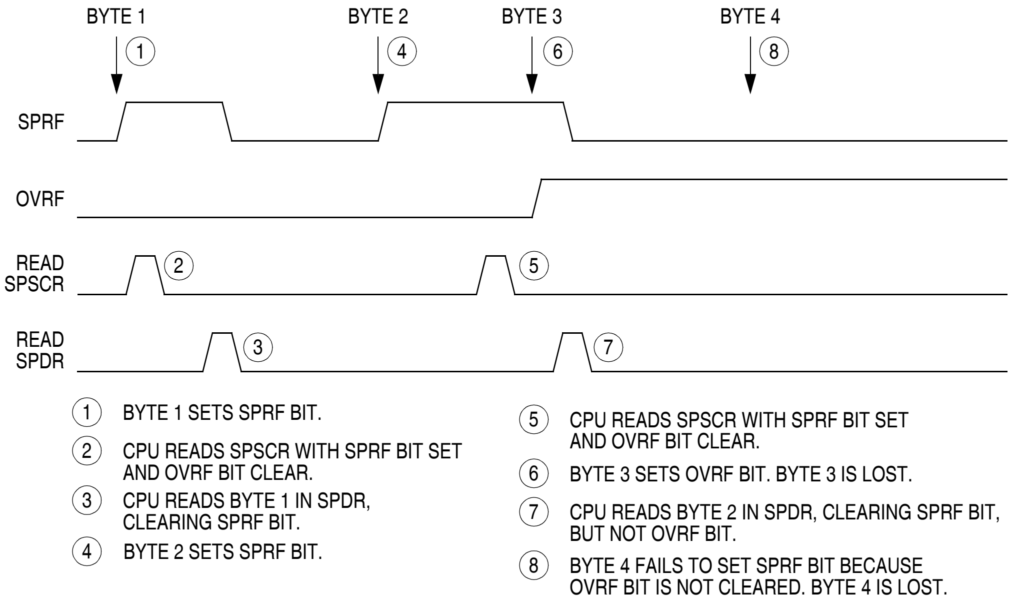
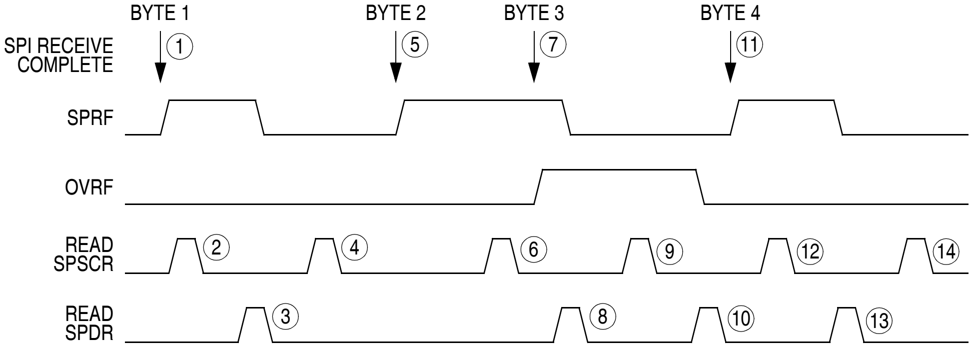


Figure 13-8. Missed Read of Overflow Condition

In this case, an overflow can easily be missed. 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 13-9** illustrates this process. Generally, to avoid this second SPSCR read, enable the OVRF interrupt to the CPU by setting the ERRIE bit.



① BYTE 1 SETS SPRF BIT.

② CPU READS SPSCR WITH SPRF BIT SET AND OVRF BIT CLEAR.

③ CPU READS BYTE 1 IN SPDR, CLEARING SPRF BIT.

④ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

⑤ BYTE 2 SETS SPRF BIT.

⑥ CPU READS SPSCR WITH SPRF BIT SET AND OVRF BIT CLEAR.

⑦ BYTE 3 SETS OVRF BIT. BYTE 3 IS LOST.

⑧ CPU READS BYTE 2 IN SPDR, CLEARING SPRF BIT.

⑨ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

⑩ CPU READS BYTE 2 SPDR, CLEARING OVRF BIT.

⑪ BYTE 4 SETS SPRF BIT.

⑫ CPU READS SPSCR.

⑬ CPU READS BYTE 4 IN SPDR, CLEARING SPRF BIT.

⑭ CPU READS SPSCR AGAIN TO CHECK OVRF BIT.

Figure 13-9. Clearing SPRF When OVRF Interrupt Is Not Enabled

13.7.2 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, $\overline{\text{SS}}$, 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 $\overline{\text{SS}}$ pin of a slave SPI goes high during a transmission.
- The $\overline{\text{SS}}$ 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. MODF and OVRF can generate a receiver/error CPU interrupt request. See **Figure 13-10**. 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 these 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 **13.6 Transmission Formats**.

NOTE: *Setting the MODF flag does not clear the SPMSTR bit. Reading SPMSTR when MODF = 1 will indicate a mode fault error occurred in either master mode or slave mode.*

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 procedure must occur with no MODF condition existing or else the flag is not cleared.

13.8 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests as shown in **Table 13-2**.

Table 13-2. SPI Interrupts

Flag	Request
SPTE transmitter empty	SPI transmitter CPU interrupt request (SPTIE = 1, SPE = 1)
SPRF receiver full	SPI receiver CPU interrupt request (SPRIE = 1)
OVRF overflow	SPI receiver/error interrupt request (ERRIE = 1)
MODF mode fault	SPI receiver/error interrupt request (ERRIE = 1)

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, provided that the SPI is enabled (SPE = 1). (See **Figure 13-10**.)

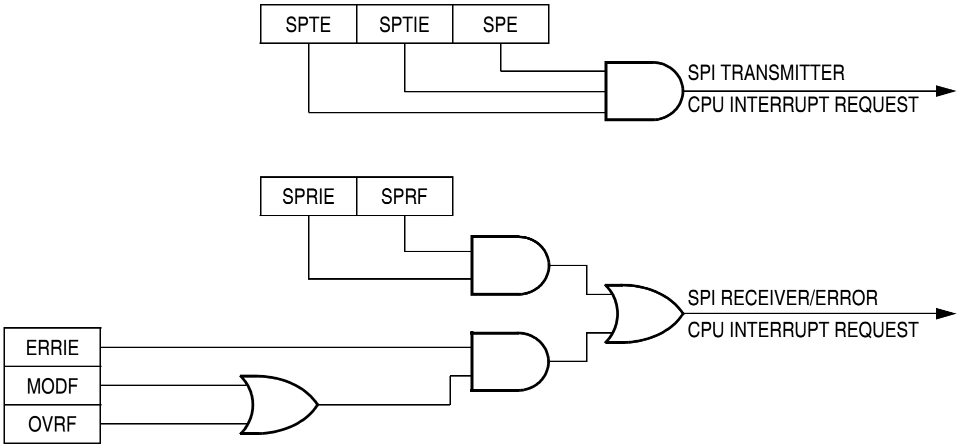


Figure 13-10. SPI Interrupt Request Generation

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.

These 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 can generate either an SPI receiver/error CPU interrupt.
- 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 can generate either an SPTE CPU interrupt request.

13.9 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 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:

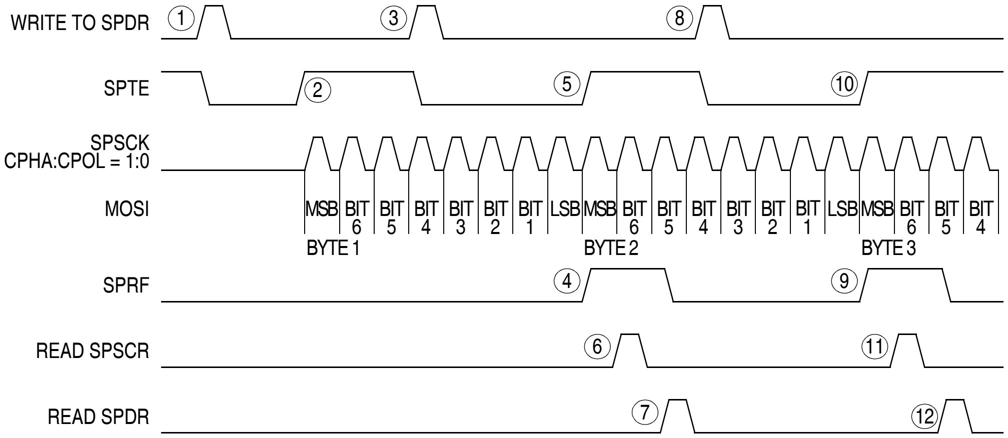
- All control bits in the SPCR register
- All control bits in the SPSCR register (MODFEN, ERRIE, SPR1, and SPR0)
- The status flags SPRF, OVRF, and MODF

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 occuring in an SPI that was configured as a master with the MODFEN bit set.

13.10 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 13-11** 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 13-11. SPRF/SPTE CPU Interrupt Timing

For a slave, 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.

13.11 Low-Power Mode

The WAIT instruction puts the MCU in a low power-consumption standby 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 **13.8 Interrupts**.

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.

13.12 I/O Signals

The SPI module has five I/O pins and shares four of them with a parallel I/O port. The pins are:

- MISO — Data received
- MOSI — Data transmitted
- SPSCK — Serial clock
- $\overline{\text{SS}}$ — Slave select

The SPI has limited inter-integrated circuit (I^2C) capability (requiring software support) as a master in a single-master environment. To communicate with I^2C peripherals, MOSI becomes an open-drain output when the SPWOM bit in the SPI control register is set. In I^2C communication, the MOSI and MISO pins are connected to a bidirectional pin from the I^2C peripheral and through a pullup resistor to V_{DD}.

13.12.1 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 $\overline{\text{SS}}$ pin is at logic 0. To support a multiple-slave system, a logic 1 on the $\overline{\text{SS}}$ pin puts the MISO pin in a high-impedance state.

When enabled, the SPI controls data direction of the MISO pin regardless of the state of the data direction register of the shared I/O port.

13.12.2 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.

13.12.3 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.

13.12.4 $\overline{\text{SS}}$ (Slave Select)

The $\overline{\text{SS}}$ pin has various functions depending on the current state of the SPI. For an SPI configured as a slave, the $\overline{\text{SS}}$ is used to select a slave. For CPHA = 0, the $\overline{\text{SS}}$ is used to define the start of a transmission. See **13.6 Transmission Formats**. Since it is used to indicate the start of a transmission, the $\overline{\text{SS}}$ must be toggled high and low between each byte transmitted for the CPHA = 0 format. However, it can remain low between transmissions for the CPHA = 1 format. See **Figure 13-12**.

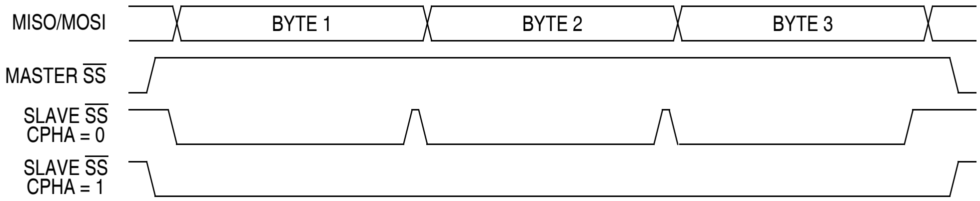


Figure 13-12. CPHA/$\overline{SS}$ Timing

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 **13.13.2 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 **13.7.2 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 13-3**.

Table 13-3. SPI Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	State of $\overline{\text{SS}}$ Logic
0	X(1)	X	Not Enabled	General-purpose I/O; $\overline{\text{SS}}$ ignored by SPI
1	0	X	Slave	Input-only to SPI
1	1	0	Master without MODF	General-purpose I/O; $\overline{\text{SS}}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

1. X = don't care

13.12.5 V_{SS} (Clock Ground)

V_{SS} is the ground return for the serial clock pin, SPSCK, and the ground for the port output buffers. To reduce the ground return path loop and minimize radio frequency (RF) emissions, connect the ground pin of the slave to the V_{SS} pin of the master.

13.13 I/O Registers

Three registers control and monitor SPI operation:

- SPI control register, SPCR
- SPI status and control register, SPSCR
- SPI data register, SPDR

13.13.1 SPI Control Register

The SPI control register (SPCR):

- Enables SPI module interrupt requests
- Selects CPU interrupt requests or DMA service 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: $0044

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SPRIE	R	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
Reset:	0	0	1	0	1	0	0	0

R = Reserved

Figure 13-13. 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

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 13-4** and **Figure 13-6**. 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 13-4** and **Figure 13-6**. To transmit data between SPI modules, the SPI modules must have identical CPHA bits. When CPHA = 0, the $\overline{SS}$ pin of the slave SPI module must be set to logic 1 between bytes. See **Figure 13-12**. Reset sets the CPHA bit.

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 data register. Therefore, the slave data register must be loaded with the desired transmit data before the falling edge of $\overline{\text{SS}}$. Any data written after the falling edge is stored in the data register and transferred to the shift register at the current transmission.

When CPHA = 1 for a slave, the first edge of the SPSCK indicates the beginning of the transmission. The same applies when $\overline{\text{SS}}$ is high for a slave. The MISO pin is held in a high-impedance state, and the incoming SPSCK is ignored. In certain cases, it may also cause the MODF flag to be set. See **13.7.2 Mode Fault Error**. A logic 1 on the $\overline{\text{SS}}$ pin does not in any way affect the state of the SPI state machine.

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 Bit

This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. See **13.9 Resetting the SPI**. Reset clears the SPE bit.

1 = SPI module enabled
0 = SPI module disabled

SPTIE— SPI Transmit Interrupt Enable Bit

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

13.13.2 SPI Status and Control Register

The SPI status and control register (SPSCR) 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: $0045

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SPRF	ERRIE	OVRF	MODF	SPTE	MODFEN	SPR1	SPR0
Write:	R		R	R	R			
Reset:	0	0	0	0	1	0	0	0

R = Reserved

Figure 13-14. 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 (DMAS = 0), 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 $\overline{\text{SS}}$ pin goes high during a transmission with the MODFEN bit set. In a master SPI, the MODF flag is set if the $\overline{\text{SS}}$ 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 = $\overline{\text{SS}}$ pin at inappropriate logic level
0 = $\overline{\text{SS}}$ 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.*

For an idle master of idle slave that has no data loaded into its transmit buffer, the SPTE will be set again within two bus cycles since 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 transmision 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.

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 $\overline{\mathrm{SS}}$ 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 $\overline{\mathrm{SS}}$ pin is not available as a general-purpose I/O regardless of the value of MODFEN. See **13.12.4 SS (Slave Select)**.

If the MODFEN bit is low, the level of the $\overline{\mathrm{SS}}$ 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 **13.7.2 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 13-4**. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Table 13-4. SPI Master Baud Rate Selection

SPR1: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

13.13.3 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 13-1**.

Address: $0046

	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:	Indeterminate after reset							

Figure 13-15. 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 14. Serial Communications Interface Module (SCI)

14.1 Contents

14.2 Introduction

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

14.3 Features

Features of the SCI module include:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 32 programmable baud rates
- Programmable 8-bit or 9-bit character length
- Separately enabled transmitter and receiver
- Separate receiver and transmitter CPU interrupt requests
- Separate receiver and transmitter
- 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

14.4 Functional Description

Figure 14-1 shows the structure of the SCI module. The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

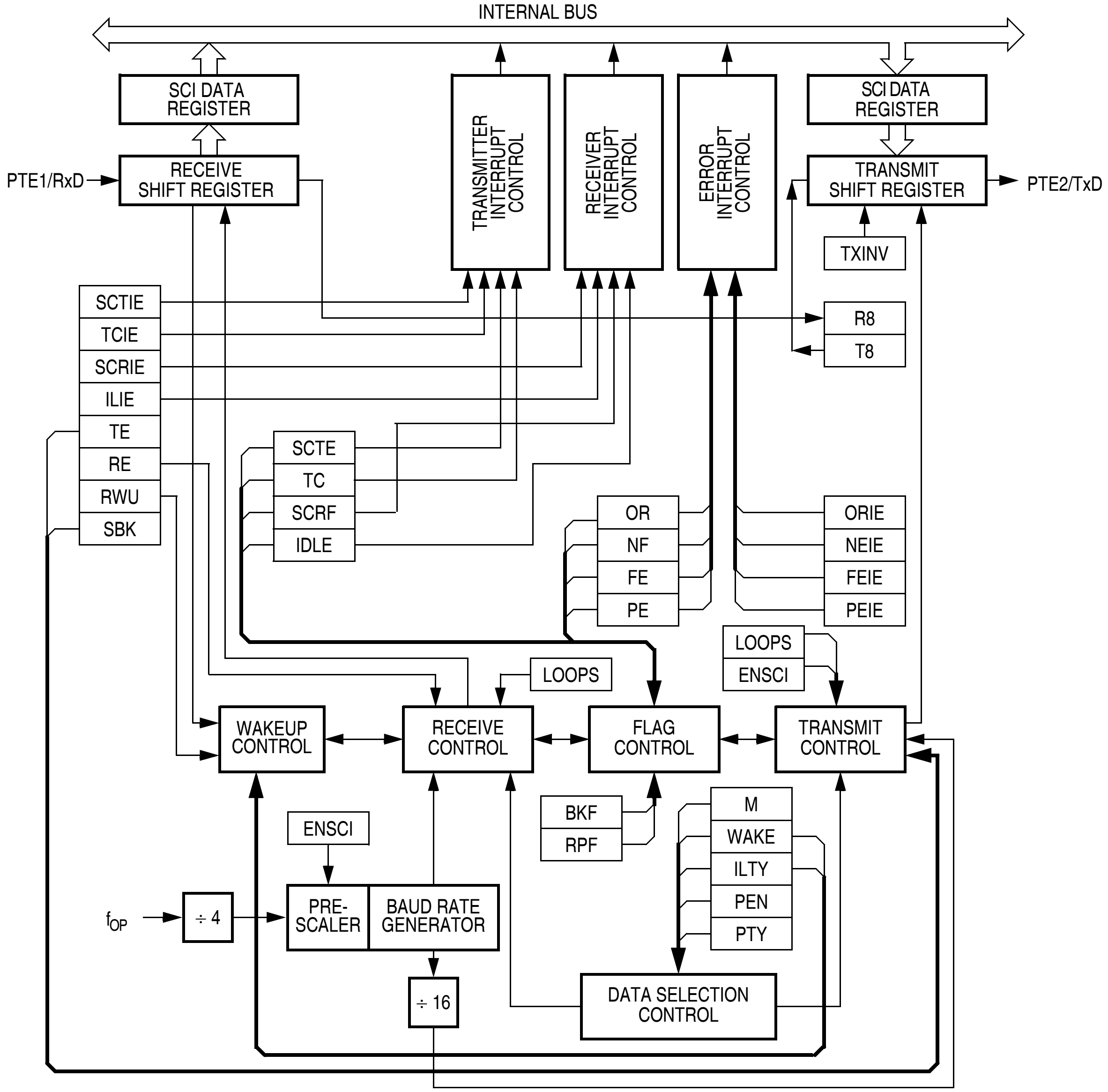


Figure 14-1. SCI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0038	SCI Control Register 1 (SCC1) See page 304.	Read: Write:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
		Reset:	0	0	0	0	0	0	0	0

Figure 14-2. SCI I/O Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0039	SCI Control Register 2 (SCC2) See page 307.	Read:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$003A	SCI Control Register 3 (SCC3) See page 310.	Read:	R8	T8	0	0	ORIE	NEIE	FEIE	PEIE
		Write:	R		R	R				
		Reset:	U	U	0	0	0	0	0	0
$003B	SCI Status Register 1 (SCS1) See page 312.	Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
		Write:	R	R	R	R	R	R	R	R
		Reset:	1	1	0	0	0	0	0	0
$003C	SCI Status Register 2 (SCS2) See page 316.	Read:	0	0	0	0	0	0	BKF	RPF
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
$003D	SCI Data Register (SCDR) See page 317.	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0
		Reset:	Unaffected by reset							
$003E	SCI Baud Rate Register (SCBR) See page 317.	Read:	0	0	SCP1	SCP0	0	SCR2	SCR1	SCR0
		Write:	R	R			R			
		Reset:	0	0	0	0	0	0	0	0

R = Reserved U = Unaffected

Figure 14-2. SCI I/O Register Summary (Continued)

14.4.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in **Figure 14-3**.

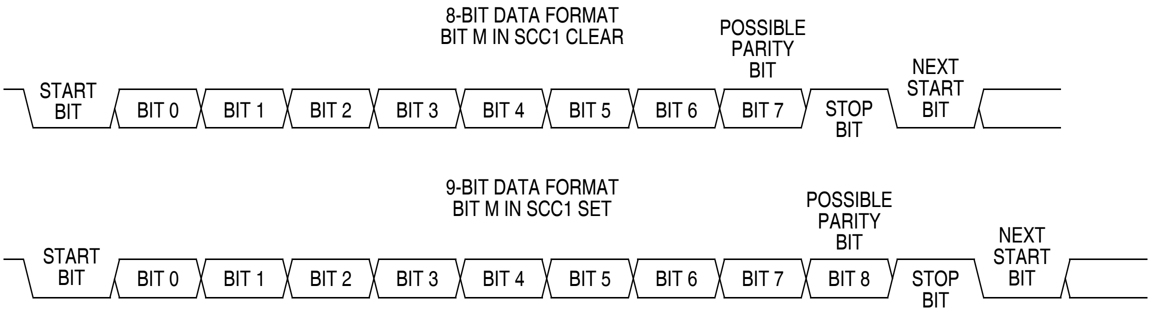


Figure 14-3. SCI Data Formats

14.4.2 Transmitter

Figure 14-4 shows the structure of the SCI transmitter.

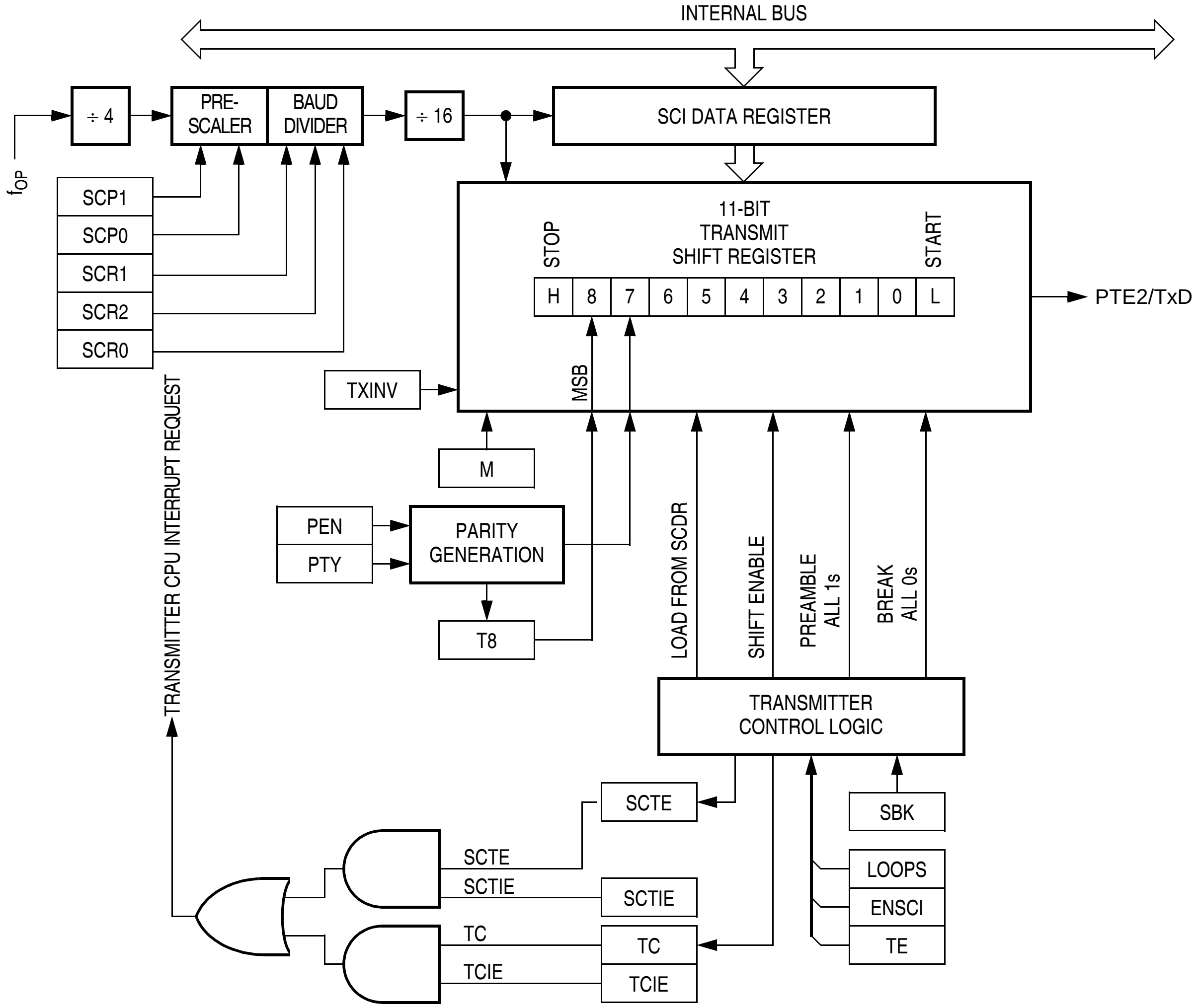


Figure 14-4. SCI Transmitter

14.4.2.1 Character Length

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

14.4.2.2 Character Transmission

During an SCI transmission, the transmit shift register shifts a character out to the PTE2/TxD pin. The SCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register. To initiate an SCI transmission:

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

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

When the transmit shift register is not transmitting a character, the PTE2/TxD pin goes to the idle condition, logic 1. If at any time software clears the ENSCI bit in SCI control register 1 (SCC1), the transmitter and receiver relinquish control of the port F pins.

14.4.2.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCC2 loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCC1. 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.

The SCI 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.

Receiving a break character has these effects on SCI registers:

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

14.4.2.4 Idle Characters

An 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 PTE2/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 PTE2/TxD pin. Setting TE after the stop bit appears on PTE2/TxD causes data previously written to the SCDR to be lost.*

A good time to toggle the TE bit is when the SCTE bit becomes set and just before writing the next byte to the SCDR.

14.4.2.5 Inversion of Transmitted Output

The transmit inversion bit (TXINV) in SCI control register 1 (SCC1) reverses the polarity of transmitted data. All transmitted values, including idle, break, start, and stop bits, are inverted when TXINV is at logic 1. See **14.8.1 SCI Control Register 1**.

14.4.2.6 Transmitter Interrupts

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

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

14.4.3 Receiver

Figure 14-5 shows the structure of the SCI receiver.

14.4.3.1 Character Length

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

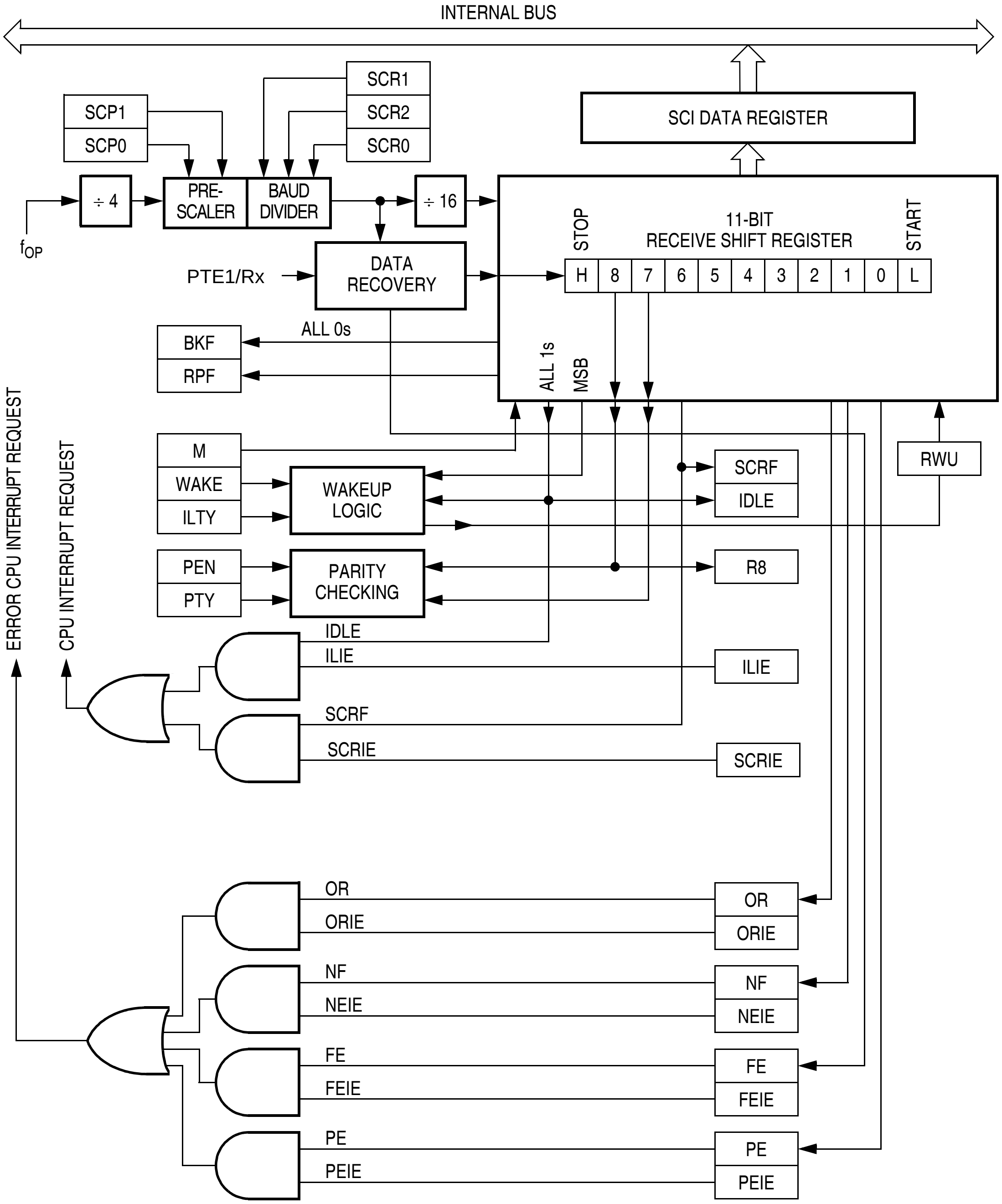


Figure 14-5. SCI Receiver Block Diagram

14.4.3.2 Character Reception

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

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

14.4.3.3 Data Sampling

The receiver samples the PTE1/RxD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock is resynchronized at these times (see **Figure 14-6**):

- 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 return a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples return 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.

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. **Table 14-1** summarizes the results of the start bit verification samples.

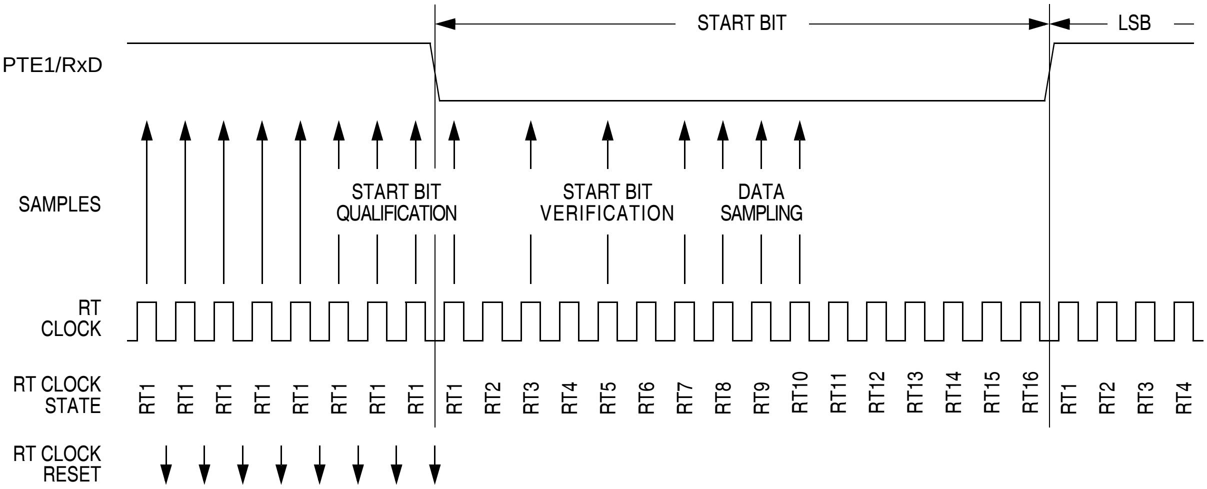


Figure 14-6. Receiver Data Sampling

Table 14-1. Start Bit Verification

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

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. **Table 14-2** summarizes the results of the data bit samples.

Table 14-2. 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 14-3** summarizes the results of the stop bit samples.

Table 14-3. 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

14.4.3.4 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. The FE flag is set at the same time that the SCRF bit is set. A break character that has no stop bit also sets the FE bit.

14.4.3.5 Receiver Wakeup

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

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

- Address mark — An address mark is a logic 1 in the most significant bit position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the SCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.
- Idle input line condition — When the WAKE bit is clear, an idle character on the PTE1/RxD pin wakes the receiver from the standby state by clearing the RWU bit. The idle character that wakes the receiver does not set the receiver idle bit, IDLE, or the SCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit.

NOTE: *Clearing the WAKE bit after the PTE1/RxD pin has been idle can cause the receiver to wake up immediately.*

14.4.3.6 Receiver Interrupts

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

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

14.4.3.7 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 SCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the SCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in SCC3 enables NF to generate SCI error CPU interrupt requests.
- Framing error (FE) — The FE bit in SCS1 is set when a logic 0 occurs where the receiver expects a stop bit. The framing error interrupt enable bit, FEIE, in SCC3 enables FE to generate SCI error CPU interrupt requests.
- Parity error (PE) — The PE bit in SCS1 is set when the SCI detects a parity error in incoming data. The parity error interrupt enable bit, PEIE, in SCC3 enables PE to generate SCI error CPU interrupt requests.

14.5 Wait Mode

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

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

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

14.6 SCI During Break Module Interrupts

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

To allow software to clear status bits during a break interrupt, write a 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.

14.7 I/O Signals

Port F shares two of its pins with the SCI module. The two SCI input/output (I/O) pins are:

- PTE2/TxD — Transmit data
- PTE1/RxD — Receive data

14.7.1 PTE2/TxD (Transmit Data)

The PTE2/TxD pin is the serial data output from the SCI transmitter. The SCI shares the PTE2/TxD pin with port F. When the SCI is enabled, the PTE2/TxD pin is an output regardless of the state of the DDRF5 bit in data direction register F (DDRF).

14.7.2 PTE1/RxD (Receive Data)

The PTE1/RxD pin is the serial data input to the SCI receiver. The SCI shares the PTE1/RxD pin with port F. When the SCI is enabled, the PTE1/RxD pin is an input regardless of the state of the DDRF4 bit in data direction register F (DDRF).

14.8 I/O Registers

These I/O registers control and monitor SCI operation:

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

14.8.1 SCI Control Register 1

SCI control register 1 (SCC1):

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

Address: $0038

	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 14-7. SCI Control Register 1 (SCC1)

LOOPS — Loop Mode Select Bit

This read/write bit enables loop mode operation. In loop mode the PTE6/RxD pin is disconnected from the SCI, and the transmitter output goes into the receiver input. Both the transmitter and the receiver must be enabled to use loop mode. Reset clears the LOOPS bit.

1 = Loop mode enabled
0 = Normal operation enabled

ENSCI — Enable SCI Bit

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

1 = SCI enabled
0 = SCI disabled

TXINV — Transmit Inversion Bit

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

1 = Transmitter output inverted
0 = Transmitter output not inverted

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

M — Mode (Character Length) Bit

This read/write bit determines whether SCI characters are eight or nine bits long. See **Table 14-4**. The ninth bit can serve as an extra stop bit, as a receiver wakeup signal, or as a parity bit. Reset clears the M bit.

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

WAKE — Wakeup Condition Bit

This read/write bit determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit (MSB) position of a received character or an idle condition on the PTE6/RxD pin. Reset clears the WAKE bit.

1 = Address mark wakeup
0 = Idle line wakeup

ILTY — Idle Line Type Bit

This read/write bit determines when the SCI starts counting 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 SCI parity function. See **Table 14-4**. When enabled, the parity function inserts a parity bit in the most significant bit position. See **Figure 14-3**. Reset clears the PEN bit.

1 = Parity function enabled
0 = Parity function disabled

PTY — Parity Bit

This read/write bit determines whether the SCI generates and checks for odd parity or even parity. See **Table 14-4**. Reset clears the PTY bit.

1 = Odd parity
0 = Even parity

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

Table 14-4. Character Format Selection

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

14.8.2 SCI Control Register 2

SCI control register 2 (SCC2):

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

Address: $0039

	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 14-8. SCI Control Register 2 (SCC2)

SCTIE — SCI Transmit Interrupt Enable Bit

This read/write bit enables the SCTE bit to generate SCI transmitter CPU interrupt requests. Setting the SCTIE bit in SCC3 enables SCTE CPU interrupt requests. Reset clears the SCTIE bit.

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

TCIE — Transmission Complete Interrupt Enable Bit

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

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

SCRIE — SCI Receive Interrupt Enable Bit

This read/write bit enables the SCRF bit to generate SCI receiver CPU interrupt requests. Setting the SCRIE bit in SCC3 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 SCI receiver CPU interrupt requests. Reset clears the ILIE bit.

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

TE — Transmitter Enable Bit

Setting this read/write bit begins the transmission by sending a preamble of 10 or 11 logic 1s from the transmit shift register to the PTE2/TxD pin. If software clears the TE bit, the transmitter completes any transmission in progress before the PTE2/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 SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.*

RE — Receiver Enable Bit

Setting this read/write bit enables the receiver. Clearing the RE bit disables the receiver but does not affect receiver interrupt flag bits. Reset clears the RE bit.

1 = Receiver enabled
0 = Receiver disabled

NOTE: *Writing to the RE bit is not allowed when the enable SCI bit (ENSCI) is clear. ENSCI is in SCI control register 1.*

RWU — Receiver Wakeup Bit

This read/write bit puts the receiver in a standby state during which receiver interrupts are disabled. The WAKE bit in SCC1 determines whether an idle input or an address mark brings the receiver out of the standby state and clears the RWU bit. Reset clears the RWU bit.

1 = Standby state
0 = Normal operation

SBK — Send Break Bit

Setting and then clearing this read/write bit transmits a break character followed by a 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 too early causes the SCI to send a break character instead of a preamble.*

14.8.3 SCI Control Register 3

SCI control register 3 (SCC3):

- Stores the ninth SCI data bit received and the ninth SCI data bit to be transmitted
- Enables SCI receiver full (SCRF)
- Enables SCI transmitter empty (SCTE)
- Enables the following interrupts:
  - Receiver overrun interrupts
  - Noise error interrupts
  - Framing error interrupts
  - Parity error interrupts

Address: $003A

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

R = Reserved

U = Unaffected

Figure 14-9. SCI Control Register 3 (SCC3)

R8 — Received Bit 8

When the SCI is receiving 9-bit characters, R8 is the read-only ninth bit (bit 8) of the received character. R8 is received at the same time that the SCDR receives the other eight bits.

When the SCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7). Reset has no effect on the R8 bit.

T8 — Transmitted Bit 8

When the SCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register. Reset has no effect on the T8 bit.

ORIE — Receiver Overrun Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the receiver overrun bit, OR.

1 = SCI error CPU interrupt requests from OR bit enabled
0 = SCI error CPU interrupt requests from OR bit disabled

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.

1 = SCI error CPU interrupt requests from NE bit enabled
0 = SCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.

1 = SCI error CPU interrupt requests from FE bit enabled
0 = SCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables SCI receiver CPU interrupt requests generated by the parity error bit, PE. See **14.8.4 SCI Status Register 1**. Reset clears PEIE.

1 = SCI error CPU interrupt requests from PE bit enabled
0 = SCI error CPU interrupt requests from PE bit disabled

14.8.4 SCI Status Register 1

SCI status register 1 (SCS1) contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $003B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:	R	R	R	R	R	R	R	R
Reset:	1	1	0	0	0	0	0	0

R = Reserved

Figure 14-10. SCI Status Register 1 (SCS1)

SCTE — SCI Transmitter Empty Bit

This clearable, read-only bit is set when the SCDR transfers a character to the transmit shift register. SCTE can generate an SCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an SCI transmitter CPU interrupt request. In normal operation, clear the SCTE bit by reading SCS1 with SCTE set and then writing to SCDR. Reset sets the SCTE bit.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

TC — Transmission Complete Bit

This read-only bit is set when the SCTE bit is set, and no data, preamble, or break character is being transmitted. TC generates an SCI transmitter CPU interrupt request if the TCIE bit in SCC2 is also set. TC is 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 — SCI Receiver Full Bit

This clearable, read-only bit is set when the data in the receive shift register transfers to the SCI data register. SCRF can generate an SCI receiver CPU interrupt request. When the SCRIE bit in SCC2 is set, SCRF generates a CPU interrupt request. In normal operation, clear the SCRF bit by reading SCS1 with SCRF set and then reading the SCDR. Reset clears SCRF.

1 = Received data available in SCDR
0 = Data not available in SCDR

IDLE — Receiver Idle Bit

This clearable, read-only bit is set when 10 or 11 consecutive logic 1s appear on the receiver input. IDLE generates an SCI 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 SCI error CPU interrupt request if the ORIE bit in SCC3 is also set. The data in the shift register is lost, but the data already in the SCDR is not affected. Clear the OR bit by reading SCS1 with OR set and then reading the SCDR. Reset clears the OR bit.

1 = Receive shift register full and SCRF = 1
0 = No receiver overrun

Software latency may allow an overrun to occur between reads of SCS1 and SCDR in the flag-clearing sequence. **Figure 14-11** 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 SCI detects noise on the PTE1/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

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 SCI error CPU interrupt request if the FEIE bit in SCC3 also is set. Clear the FE bit by reading SCS1 with FE set and then reading the SCDR. Reset clears the FE bit.

1 = Framing error detected
0 = No framing error detected

PE — Receiver Parity Error Bit

This clearable, read-only bit is set when the SCI detects a parity error in incoming data. PE generates 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

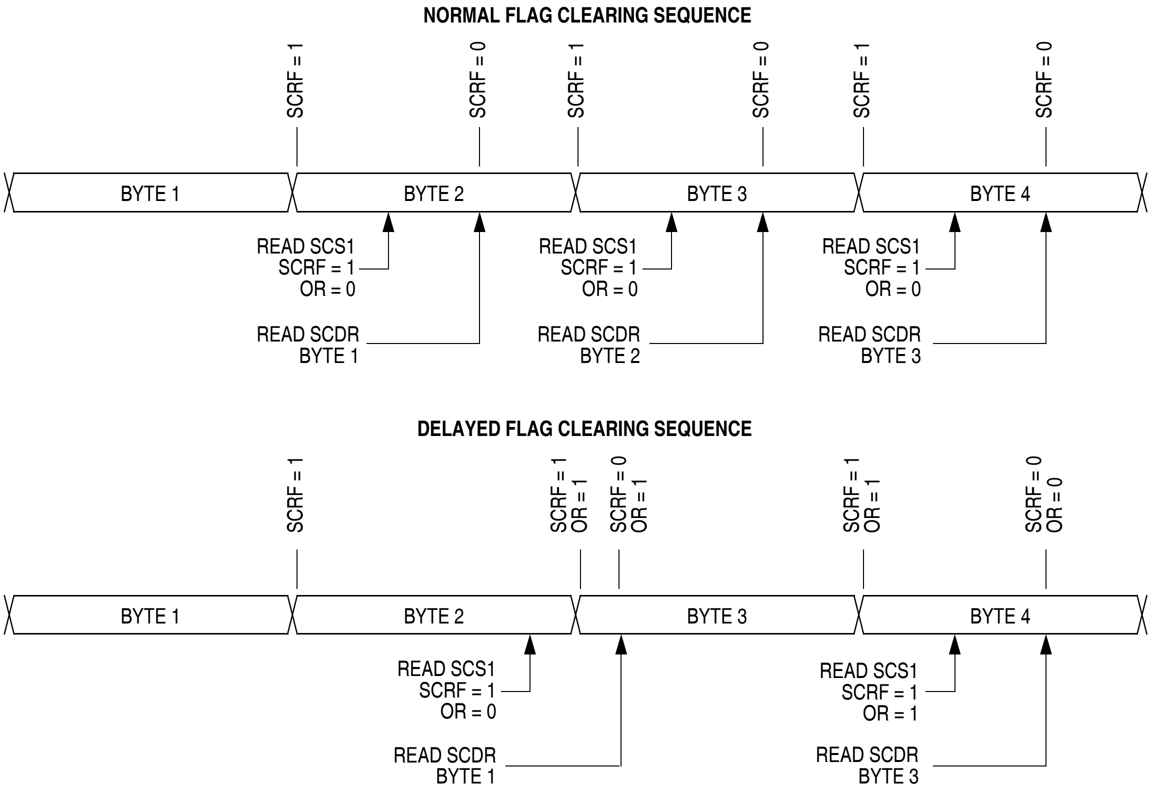


Figure 14-11. Flag Clearing Sequence

14.8.5 SCI Status Register 2

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

- Break character detected
- Incoming data

Address: $003C

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

R = Reserved

Figure 14-12. SCI Status Register 2 (SCS2)

BKF — Break Flag

This clearable, read-only bit is set when the SCI detects a break character on the PTE1/RxD pin. In SCS1, the FE and SCRF bits are also set. In 9-bit character transmissions, the R8 bit in SCC3 is cleared. BKF does not generate a CPU interrupt request. Clear BKF by reading SCS2 with BKF set and then reading the SCDR. Once cleared, BKF can become set again only after logic 1s again appear on the PTE1/RxD pin followed by another break character. Reset clears the BKF bit.

1 = Break character detected
0 = No break character detected

RPF —Reception-in-Progress Flag

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 SCI module or entering stop mode can show whether a reception is in progress.

1 = Reception in progress
0 = No reception in progress

14.8.6 SCI Data Register

The SCI data register (SCDR) is the buffer between the internal data bus and the receive and transmit shift registers. Reset has no effect on data in the SCI data register.

Address: $003D

	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 14-13. SCI Data Register (SCDR)

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

Reading address $003D accesses the read-only received data bits, R7:R0. Writing to address $003D writes the data to be transmitted, T7:T0. Reset has no effect on the SCI data register.

14.8.7 SCI Baud Rate Register

The baud rate register (SCBR) selects the baud rate for both the receiver and the transmitter.

Address: $003E

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

R = Reserved

Figure 14-14. SCI Baud Rate Register (SCBR)

SCP1 and SCP0 — SCI Baud Rate Prescaler Bits

These read/write bits select the baud rate prescaler divisor as shown in **Table 14-5**. Reset clears SCP1 and SCP0.

Table 14-5. SCI Baud Rate Prescaling

SCP1:SCP0	Prescaler Divisor (PD)
00	1
01	3
10	4
11	13

SCR2–SCR0 — SCI Baud Rate Select Bits

These read/write bits select the SCI baud rate divisor as shown in **Table 14-6**. Reset clears SCR2–SCR0.

Table 14-6. SCI Baud Rate Selection

SCR2:SCR1:SCR0	Baud Rate Divisor (BD)
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Use this formula to calculate the SCI baud rate:

$$\text{Baud rate} = \frac{f_{OP}}{64 \times PD \times BD}$$

where:

f_{OP} = internal operating frequency

PD = prescaler divisor

BD = baud rate divisor

Table 14-7 shows the SCI baud rates that can be generated with a 4.9152-MHz crystal with the CGM set for an f_{op} of 7.3728 MHz and the CGM set for an f_{op} of 4.9152 MHz.

Table 14-7. SCI Baud Rate Selection Examples

SCP1:SCP0	Prescaler Divisor (PD)	SCR2:SCR1:SCR0	Baud Rate Divisor (BD)	Baud Rate (f_{OP} = 7.3728 MHz)	Baud Rate (f_{OP} = 4.9152 MHz)
00	1	000	1	115,200	76,800
00	1	001	2	57,600	38,400
00	1	010	4	28,800	19,200
00	1	011	8	14,400	9600
00	1	100	16	7200	4800
00	1	101	32	3600	2400
00	1	110	64	1800	1200
00	1	111	128	900	600
01	3	000	1	38,400	25,600
01	3	001	2	19,200	12,800
01	3	010	4	9600	6400
01	3	011	8	4800	3200
01	3	100	16	2400	1600
01	3	101	32	1200	800
01	3	110	64	600	400
01	3	111	128	300	200
10	4	000	1	28,800	19,200
10	4	001	2	14,400	9600
10	4	010	4	7200	4800
10	4	011	8	3600	2400
10	4	100	16	1800	1200
10	4	101	32	900	600
10	4	110	64	450	300
10	4	111	128	225	150
11	13	000	1	8861.5	5907.7
11	13	001	2	4430.7	2953.8
11	13	010	4	2215.4	1476.9
11	13	011	8	1107.7	738.5
11	13	100	16	553.8	369.2
11	13	101	32	276.9	184.6
11	13	110	64	138.5	92.3
11	13	111	128	69.2	46.2

Section 15. Input/Output (I/O) Ports

15.1 Contents

15.2 Introduction

Thirty-seven bidirectional input-output (I/O) pins and seven input pins form six parallel ports. All I/O pins are programmable as inputs or outputs.

When using the 56-pin package version of the MC68HC908MR24:

- Set the data direction register bits in DDRC such that bit 1 is written to a logic 1 (along with any other output bits on port C).
- Set the data direction register bits in DDRE such that bits 0, 1, and 2 are written to a logic 1 (along with any other output bits on port E).
- Set the data direction register bits in DDRF such that bits 0, 1, 2, and 3 are written to a logic 1 (along with any other output bits on port F).

NOTE: *Connect any unused I/O pins to an appropriate logic level, either* V_{DD} *or* V_{SS}*. Although PWM6–PWM1 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 324.	Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
		Reset:	Unaffected by reset							
$0001	Port B Data Register (PTB) See page 326.	Read: Write:	PTB7	PTB6	PTB5	PTB4	PTB3	PTB2	PTB1	PTB0
		Reset:	Unaffected by reset							
$0002	Port C Data Register (PTC) See page 328.	Read:	0	PTC6	PTC5	PTC4	PTC3	PTC2	PTC1	PTC0
		Write:	R							
		Reset:	Unaffected by reset							

R = Reserved

Figure 15-1. I/O Port Register Summary

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0003	Port D Data Register (PTD) See page 330.	Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
		Write:	R	R	R	R	R	R	R	R
		Reset:	Unaffected by reset							
$0004	Data Direction Register A (DDRA) See page 324.	Read:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0005	Data Direction Register B (DDRB) See page 326.	Read:	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0006	Data Direction Register C (DDRC) See page 328.	Read:	0	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Write:	R							
		Reset:	0	0	0	0	0	0	0	0
$0008	Port E Data Register (PTE) See page 331.	Read:	PTE7	PTE6	PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
		Write:								
		Reset:	Unaffected by reset							
$0009	Port F Data Register (PTF) See page 333.	Read:	0	0	PTF5	PTF4	PTF3	PTF2	PTF1	PTF0
		Write:	R	R						
		Reset:	Unaffected by reset							
$000A			Unimplemented							
$000B			Unimplemented							
$000C	Data Direction Register E (DDRE) See page 332.	Read:	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000D	Data Direction Register F (DDRF) See page 334.	Read:	0	0	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
		Write:	R	R						
		Reset:			0	0	0	0	0	0

R = Reserved

Figure 15-1. I/O Port Register Summary (Continued)

15.3 Port A

Port A is an 8-bit, general-purpose, bidirectional I/O port.

15.3.1 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							

Figure 15-2. Port A Data Register (PTA)

PTA[7:0] — 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.

15.3.2 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 15-3. Data Direction Register A (DDRA)

DDRA[7:0] — Data Direction Register A Bits

These read/write bits control port A data direction. Reset clears DDRA[7:0], 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 15-4 shows the port A I/O logic.

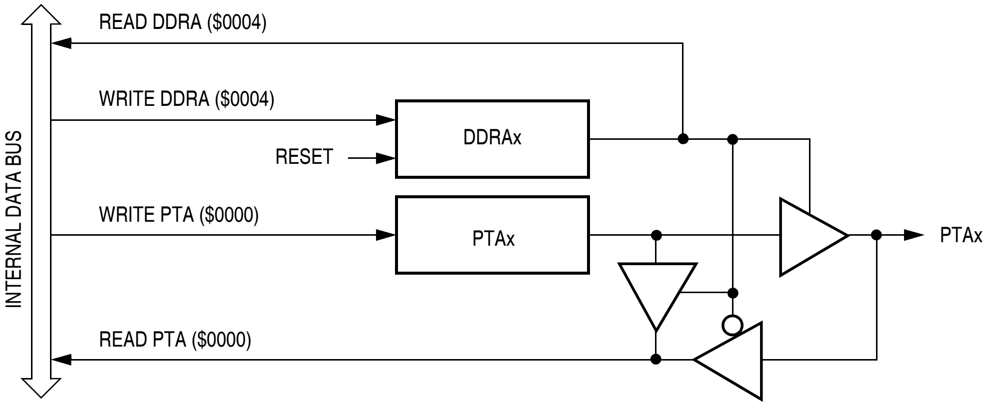


Figure 15-4. Port A I/O Circuit

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 15-1** summarizes the operation of the port A pins.

Table 15-1. Port A Pin Functions

DDRA Bit	PTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PTA	
			Read/Write	Read	Write
0	X[1]	Input, Hi-Z[2]	DDRA[7:0]	Pin	PTA[7:0][3]
1	X	Output	DDRA[7:0]	PTA[7:0]	PTA[7:0]

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

15.4 Port B

Port B is an 8-bit, general-purpose, bidirectional I/O port that shares its pins with the analog-to-digital convertor (ADC) module.

15.4.1 Port B Data Register

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

Figure 15-5. Port B Data Register (PTB)

PTB[7:0] — 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.

15.4.2 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 15-6. Data Direction Register B (DDRB)

DDRB[7:0] — Data Direction Register B Bits

These read/write bits control port B data direction. Reset clears DDRB[7:0], 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 15-7 shows the port B I/O logic.

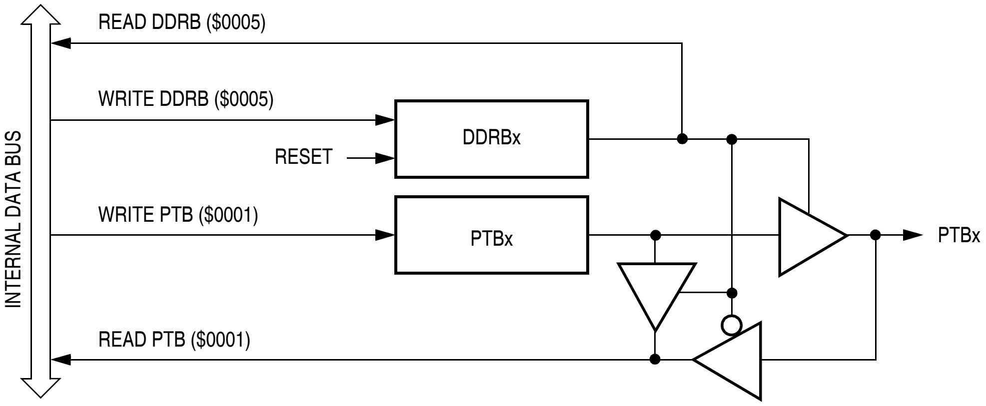


Figure 15-7. Port B I/O Circuit

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 15-2** summarizes the operation of the port B pins.

Table 15-2. 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)	DDRB[7:0]	Pin	PTB[7:0](3)
1	X	Output	DDRB[7:0]	PTB[7:0]	PTB[7:0]

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

15.5 Port C

Port C is a 7-bit, general-purpose, bidirectional I/O port that shares two of its pins with the analog-to-digital convertor module (ADC).

15.5.1 Port C Data Register

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

Address: $0002

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

R = Reserved

Figure 15-8. Port C Data Register (PTC)

PTC[6:0] — 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.

15.5.2 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:	R							
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 15-9. Data Direction Register C (DDRC)

DDRC[6:0] — Data Direction Register C Bits

These read/write bits control port C data direction. Reset clears DDRC[6:0], 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 15-10 shows the port C I/O logic.

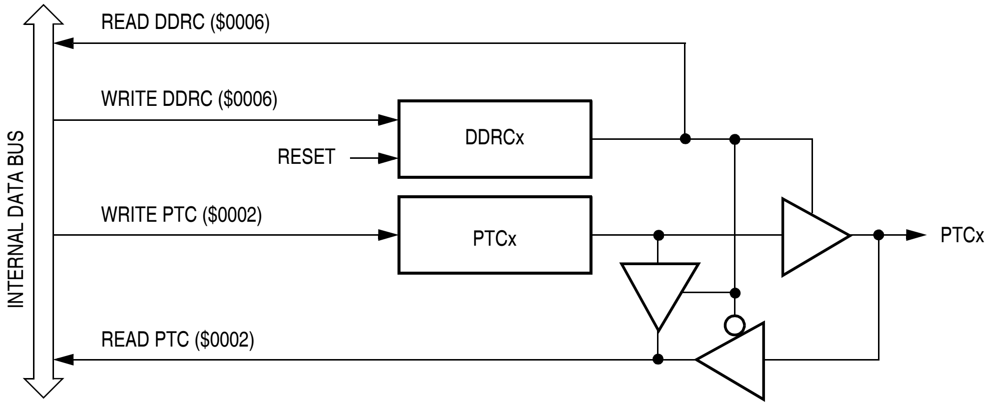


Figure 15-10. Port C I/O Circuit

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 15-3** summarizes the operation of the port C pins.

Table 15-3. Port C Pin Functions

DDRC Bit	PTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PTC	
			Read/Write	Read	Write
0	X[1]	Input, Hi-Z[2]	DDRC[6:0]	Pin	PTC[6:0][3]
1	X	Output	DDRC[6:0]	PTC[6:0]	PTC[6:0]

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

15.6 Port D

Port D is a 7-bit, input-only port that shares its pins with the pulse width modulator for motor control module (PMC).

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

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Write:	R	R	R	R	R	R	R	R
Reset:	Unaffected by reset							

R = Reserved

Figure 15-11. Port D Data Register (PTD)

PTD[6:0] — Port D Data Bits

These read/write bits are software programmable. Reset has no effect on port D data.

Figure 15-12 shows the port D input logic.

Figure 15-12. Port D Input Circuit

Reading address $0003 reads the voltage level on the pin. **Table 15-1** summarizes the operation of the port D pins.

Table 15-4. Port D Pin Functions

PTD Bit	Pin Mode	Accesses to PTD	
		Read	Write
X[1]	Input, Hi-Z[2]	Pin	PTD[6:0][3]

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

15.7 Port E

Port E is an 8-bit, special function port that shares five of its pins with the timer interface module (TIM) and two of its pins with the serial communications interface module (SCI).

15.7.1 Port E Data Register

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

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTE7	PTE6	PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
Reset:	Unaffected by reset							

Figure 15-13. Port E Data Register (PTE)

PTE[7:0] — Port E Data Bits

PTE[7:0] are read/write, software-programmable bits. Data direction of each port E pin is under the control of the corresponding bit in data direction register E.

NOTE: *Data direction register E (DDRE) does not affect the data direction of port E pins that are being used by the TIMA or TIMB. However, the DDRE bits always determine whether reading port E returns the states of the latches or the states of the pins.*

15.7.2 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: Write:	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
Reset:	0	0	0	0	0	0	0	0

Figure 15-14. Data Direction Register E (DDRE)

DDRE[7:0] — Data Direction Register E Bits

These read/write bits control port E data direction. Reset clears DDRE[7:0], 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 15-15 shows the port E I/O logic.

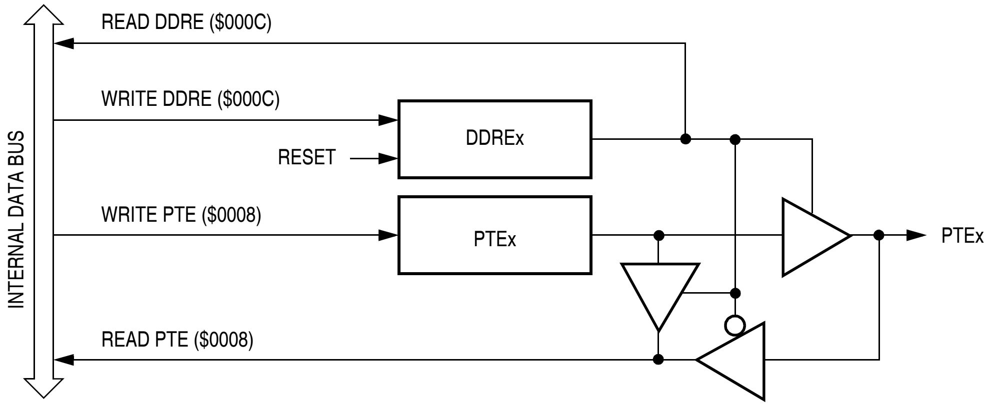


Figure 15-15. Port E I/O Circuit

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 15-5** summarizes the operation of the port E pins.

Table 15-5. 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]	DDRE[7:0]	Pin	PTE[7:0][3]
1	X	Output	DDRE[7:0]	PTE[7:0]	PTE[7:0]

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

15.8 Port F

Port F is a 6-bit, special function port that shares four of its pins with the serial peripheral interface module (SPI) and two pins with the serial communications interface (SCI).

15.8.1 Port F Data Register

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

Address: $0009

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	PTF5	PTF4	PTF3	PTF2	PTF1	PTF0
Write:	R	R						
Reset:	Unaffected by reset							

R = Reserved

Figure 15-16. Port F Data Register (PTF)

PTF[5:0] — Port F Data Bits

These read/write bits are software programmable. Data direction of each port F pin is under the control of the corresponding bit in data direction register F. Reset has no effect on PTF[5:0].

NOTE: *Data direction register F (DDRF) does not affect the data direction of port F pins that are being used by the SPI or SCI module. However, the DDRF bits always determine whether reading port F returns the states of the latches or the states of the pins.*

15.8.2 Data Direction Register F

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

Address: $000D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
Write:	R	R						
Read:			0	0	0	0	0	0

R = Reserved

Figure 15-17. Data Direction Register F (DDRF)

DDRF[5:0] — Data Direction Register F Bits

These read/write bits control port F data direction. Reset clears DDRF[5:0], configuring all port F pins as inputs.

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

NOTE: *Avoid glitches on port F pins by writing to the port F data register before changing data direction register F bits from 0 to 1.*

Figure 15-18 shows the port F I/O logic.

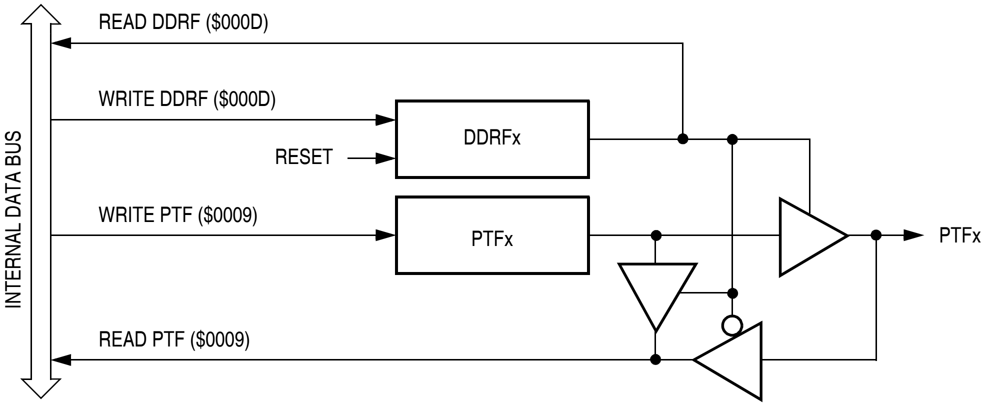


Figure 15-18. Port F I/O Circuit

When bit DDRFx is a logic 1, reading address $0009 reads the PTFx data latch. When bit DDRFx is a logic 0, reading address $0009 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit. **Table 15-6** summarizes the operation of the port F pins.

Table 15-6. Port F Pin Functions

DDRF Bit	PTF Bit	I/O Pin Mode	Accesses to DDRF	Accesses to PTF	
			Read/Write	Read	Write
0	X[1]	Input, Hi-Z[2]	DDRF[6:0]	Pin	PTF[6:0][3]
1	X	Output	DDRF[6:0]	PTF[6:0]	PTF[6:0]

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

Section 16. Computer Operating Properly (COP)

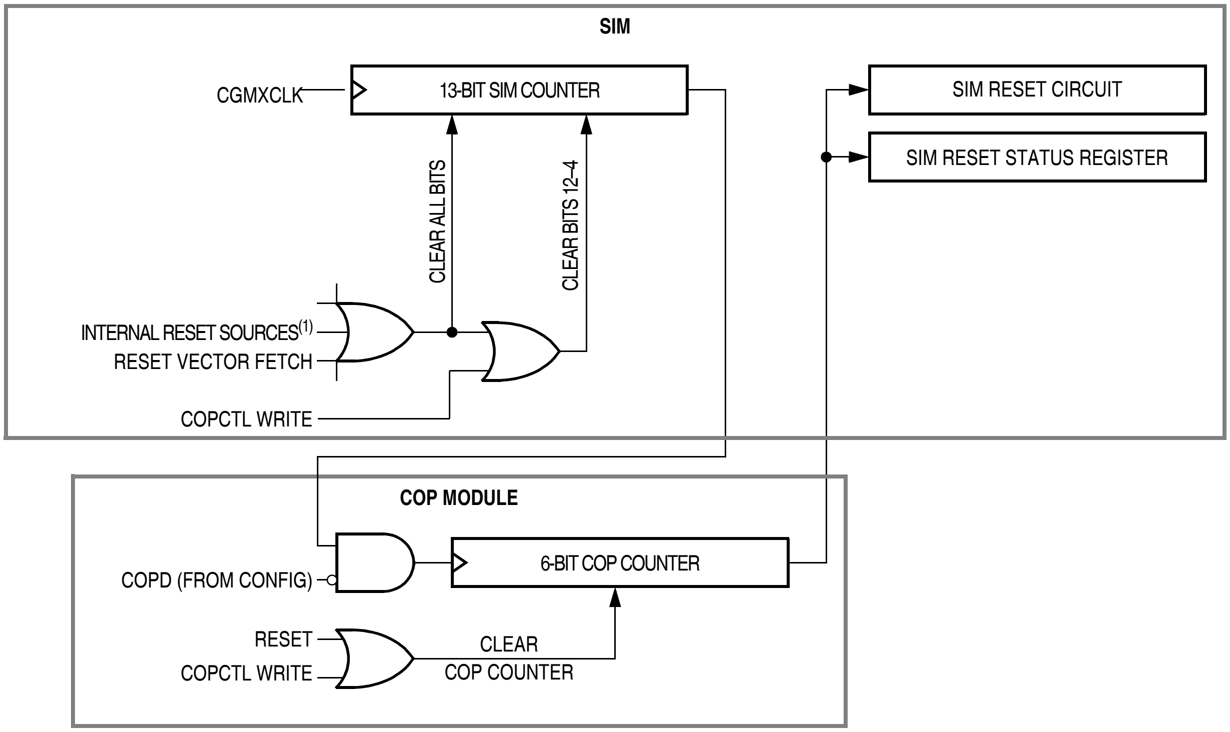
16.1 Contents

16.2 Introduction

This section describes the computer operating properly module, 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 periodically clearing the COP counter.

16.3 Functional Description

Figure 16-1 shows the structure of the COP module.



Note
1. See 7.4.2 Active Resets from Internal Sources.

Figure 16-1. COP Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FFFF	COP Control Register (COPCTL) See page 340.	Read:	Low byte of reset vector							
		Write:	Clear COP counter							
		Reset:	Unaffected by reset							

Figure 16-2. COP I/O Register Summary

The COP counter is a free-running, 6-bit counter preceded by the 13-bit system integration module (SIM) counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after $2^{18} - 2^4$ CGMXCLK cycles. With a 4.9152-MHz crystal, the COP timeout period is 53.3 ms. Writing any value to location $FFFF before overflow occurs clears the COP counter and prevents reset.

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 CGMXCLK cycles and sets the COP bit in the SIM reset status register (SRSR). See **7.7.2 SIM Reset Status Register**.

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.*

16.4 I/O Signals

This section describes the signals shown in **Figure 16-1**.

16.4.1 CGMXCLK

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

16.4.2 COPCTL Write

Writing any value to the COP control register (COPCTL) (see **16.5 COP Control Register**) clears the COP counter and clears bits 12 through 4 of the SIM counter. Reading the COP control register returns the reset vector.

16.4.3 Power-On Reset

The power-on reset (POR) circuit in the SIM clears the SIM counter 4096 CGMXCLK cycles after power-up.

16.4.4 Internal Reset

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

16.4.5 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the SIM counter.

16.4.6 COPD (COP Disable)

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

16.5 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 16-3. COP Control Register (COPCTL)

16.6 Interrupts

The COP does not generate CPU interrupt requests.

16.7 Monitor Mode

The COP is disabled in monitor mode when V_{HI} is present on the $\overline{\text{IRQ}}$ pin or on the $\overline{\text{RST}}$ pin.

16.8 Wait Mode

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

The COP continues to operate during wait mode.

Section 17. External Interrupt (IRQ)

17.1 Contents

17.2 Introduction

This section describes the external interrupt (IRQ) module, which supports external interrupt functions.

17.3 Features

Features of the IRQ module include:

- A dedicated external interrupt pin, $\overline{\text{IRQ}}$
- Hysteresis buffers

17.4 Functional Description

A logic 0 applied to any of the external interrupt pins can latch a CPU interrupt request. **Figure 17-1** shows the structure of the IRQ module.

Interrupt signals on the $\overline{\text{IRQ}}$ pin are latched into the IRQ1 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 (ISCR). Writing a logic 1 to the ACK1 bit clears the IRQ1 latch.
- Reset — A reset automatically clears both interrupt latches.

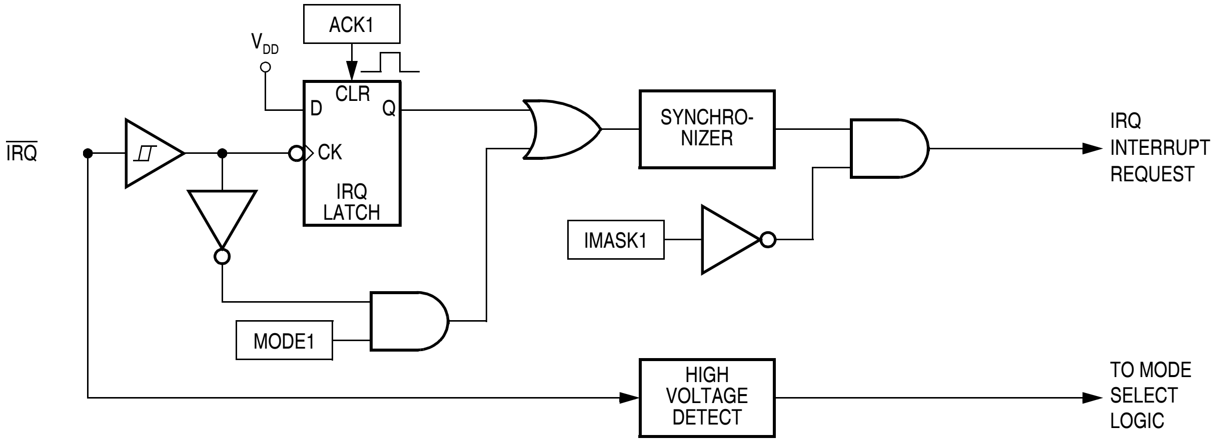


Figure 17-1. IRQ Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$003F	IRQ Status/Control Register (ISCR) See page 348.	Read:	0	0	0	0	IRQF	0	IMASK1	MODE1
		Write:	R	R	R	R		ACK1		
		Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 17-2. IRQ I/O Register Summary

The external interrupt pins are falling-edge-triggered and are software-configurable to be both falling-edge and low-level-triggered. The MODE1 bit in the ISCR controls the triggering sensitivity of the $\overline{\mathrm{IRQ}}$ pin.

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

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

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

The vector fetch or software clear can occur before or after the interrupt pin returns to logic 1. As long as the pin is low, the interrupt request remains pending.

When set, the IMASK1 bit in the ISCR masks all external interrupt requests. A latched interrupt request is not presented to the interrupt priority logic unless the IMASK bit is clear.

NOTE: *The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including external interrupt requests.*
(See ***Figure 17-3****.)*

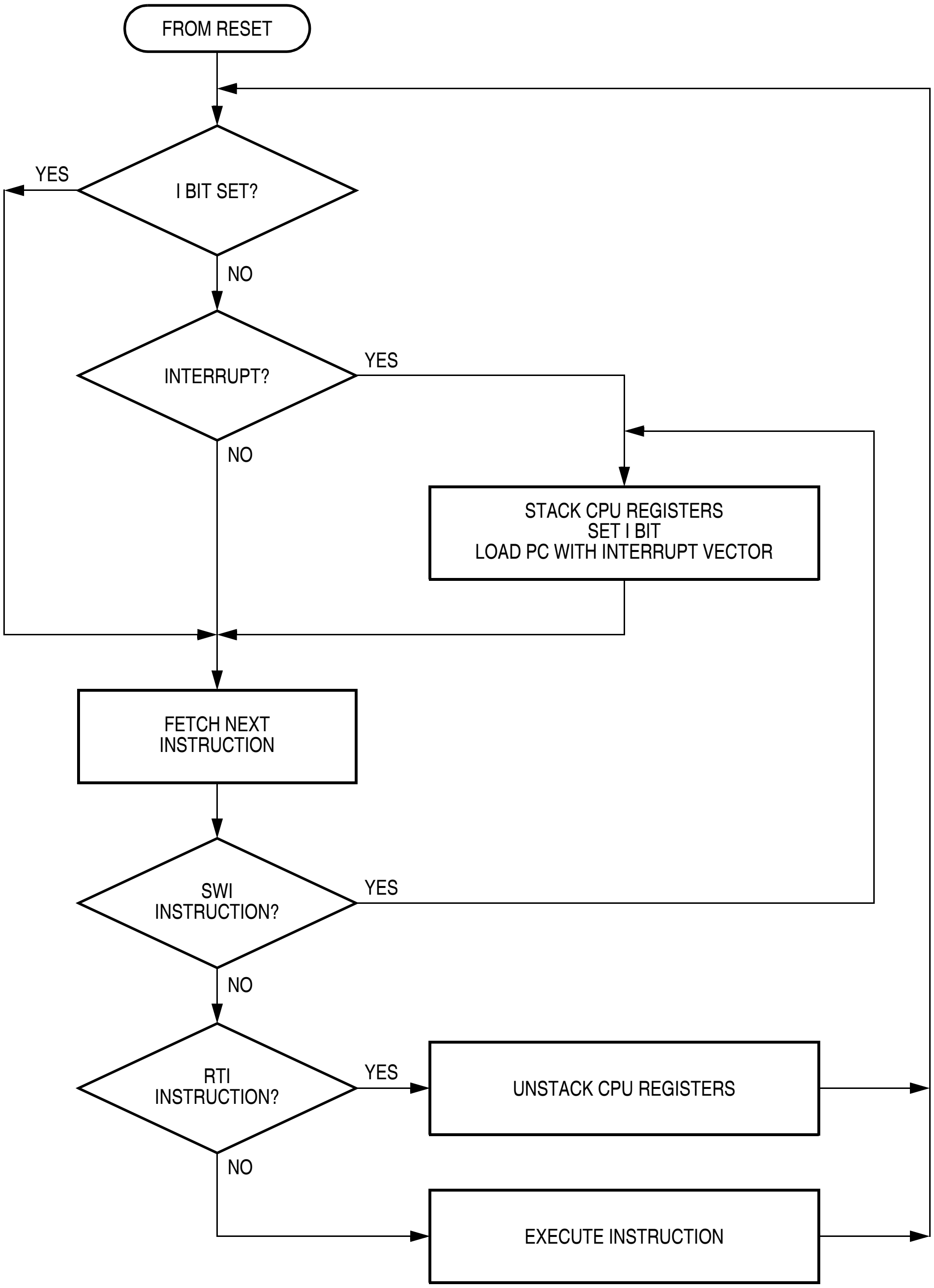


Figure 17-3. IRQ Interrupt Flowchart

17.5 $\overline{\text{IRQ}}$ Pin

A logic 0 on the $\overline{\text{IRQ}}$ pin can latch an interrupt request into the IRQ latch. A vector fetch, software clear, or reset clears the IRQ latch.

If the MODE1 bit is set, the $\overline{\text{IRQ}}$ pin is both falling-edge-sensitive and low-level-sensitive. With MODE1 set, both of these actions must occur to clear the IRQ1 latch:

- Vector fetch, software clear, or reset — A vector fetch generates an interrupt acknowledge signal to clear the latch. Software can generate the interrupt acknowledge signal by writing a logic 1 to the ACK1 bit in the interrupt status and control register (ISCR). The ACK1 bit is useful in applications that poll the $\overline{\text{IRQ}}$ pin and require software to clear the IRQ1 latch. Writing to the ACK1 bit can also prevent spurious interrupts due to noise. Setting ACK1 does not affect subsequent transitions on the $\overline{\text{IRQ}}$ pin. A falling edge that occurs after writing to the ACK1 bit latches another interrupt request. If the IRQ1 mask bit, IMASK1, is clear, the CPU loads the program counter with the vector address at locations \$FFFA and \$FFFB.
- Return of the $\overline{\text{IRQ}}$ pin to logic 1 — As long as the $\overline{\text{IRQ}}$ pin is at logic 0, the IRQ1 latch remains set.

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

If the MODE1 bit is clear, the $\overline{\text{IRQ}}$ pin is falling-edge-sensitive only. With MODE1 clear, a vector fetch or software clear immediately clears the IRQ1 latch.

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

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

17.6 IRQ Status and Control Register

The IRQ status and control register (ISCR) has these functions:

- Clears the IRQ interrupt latch
- Masks IRQ interrupt requests
- Controls triggering sensitivity of the $\overline{\text{IRQ}}$ interrupt pin

Address: $003F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	IRQF	0	IMASK1	MODE1
Write:	R	R	R	R		ACK1		
Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 17-4. IRQ Status and Control Register (ISCR)

ACK1 — IRQ Interrupt Request Acknowledge Bit

Writing a logic 1 to this write-only bit clears the IRQ latch. ACK1 always reads as logic 0. Reset clears ACK1.

IMASK1 — IRQ Interrupt Mask Bit

Writing a logic 1 to this read/write bit disables IRQ interrupt requests. Reset clears IMASK1.

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

MODE1 — IRQ Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin. Reset clears MODE1.

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

IRQF — IRQ Flag

This read-only bit acts as a status flag, indicating an IRQ event occurred.

1 = External IRQ event occurred
0 = External IRQ event did not occur

Section 18. Low-Voltage Inhibit (LVI)

18.1 Contents

18.2 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 to the LVI trip voltage.

18.3 Features

Features of the LVI module include:

- Programmable LVI reset
- Programmable power consumption
- Digital filtering of V_{DD} pin level
- Selectable LVI trip voltage

18.4 Functional Description

Figure 18-1 shows the structure of the LVI module. The LVI is enabled out of reset. The LVI module contains a bandgap reference circuit and comparator. The LVI power bit, LVIPWR, enables the LVI to monitor V_{DD} voltage. The LVI reset bit, LVIRST, enables the LVI module to generate a reset when V_{DD} falls below a voltage, V_{LVRX}, and remains at or below that level for nine or more consecutive CGMXCLK. V_{LVRX} and V_{LVHX} are determined by the TRPSEL bit in the LVISCR (see **Figure 18-2**). LVIPWR and LVIRST are in the configuration register (CONFIG). See **Section 5. Configuration Register (CONFIG)**.

Once an LVI reset occurs, the MCU remains in reset until V_{DD} rises above a voltage, $V_{LVRX} + V_{LVHX}$. V_{DD} must be above $V_{LVRX} + V_{LVHX}$ for only one CPU cycle to bring the MCU out of reset. See **7.4.2.5 Low-Voltage Inhibit (LVI) Reset**. The output of the comparator controls the state of the LVIOUT flag in the LVI status register (LVISCR).

An LVI reset also drives the $\overline{\text{RST}}$ pin low to provide low-voltage protection to external peripheral devices. See **21.6 DC Electrical Characteristics (V_{DD} = 5.0 Vdc ± 10%)**.

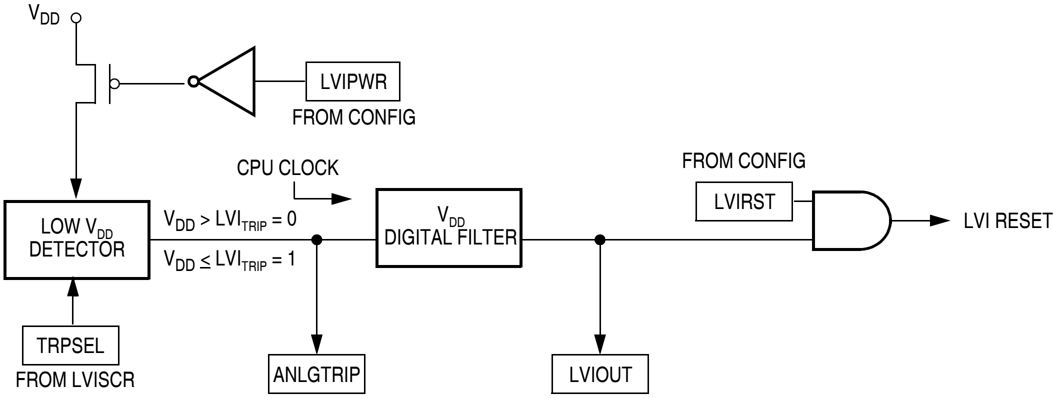


Figure 18-1. LVI Module Block Diagram

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0F	LVI Status and Control Register (LVISCR) See page 352.	Read:	LVIOUT	0	TRPSEL	0	0	0	0	0
		Write:	R	R		R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0

R = Reserved

Figure 18-2. LVI I/O Register Summary

18.4.1 Polled LVI Operation

In applications that can operate at V_{DD} levels below V_{LVRX}, software can monitor V_{DD} by polling the LVIOUT bit. In the configuration register, the LVIPWR bit must be at logic 1 to enable the LVI module, and the LVIRST bit must be at logic 0 to disable LVI resets. See **Section 5. Configuration Register (CONFIG)**. TRPSEL in the LVISCR selects V_{LVRX}.

18.4.2 Forced Reset Operation

In applications that require V_{DD} to remain above V_{LVRX}, enabling LVI resets allows the LVI module to reset the MCU when V_{DD} falls to the V_{LVRX} level and remains at or below that level for nine or more consecutive CPU cycles. In the MOR, the LVIPWR and LVIRST bits must be at logic 0 to enable the LVI module and to enable LVI resets. TRPSEL in the LVISCR selects V_{LVRX}.

18.4.3 False Reset Protection

The V_{DD} pin level is digitally filtered to reduce false resets due to power supply noise. In order for the LVI module to reset the MCU,V_{DD} must remain at or below V_{LVRX} for nine or more consecutive CPU cycles. V_{DD} must be above $V_{LVRX} + V_{LVHX}$ for only one CPU cycle to bring the MCU out of reset. TRPSEL in the LVISCR selects $V_{LVRX} + V_{LVHX}$.

18.4.4 LVI Trip Selection

The TRPSEL bit allows the user to chose between 5 percent and 10 percent tolerance when monitoring the supply voltage. The 10 percent option is enabled out of reset. Writing a logic 1 to TRPSEL will enable 5 percent option.

NOTE: *The microcontroller is guaranteed to operate at a minimum supply voltage. The trip point (V_{LVR1} or V_{LVR2}) may be lower than this. See* ***21.6 DC Electrical Characteristics (V_{DD} = 5.0 Vdc ± 10%)****.*

18.5 LVI Status and Control Register

The LVI status register (LVISCR) flags V_{DD} voltages below the V_{LVRX} level.

Address: $FE0F

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

R = Reserved

Figure 18-3. LVI Status and Control Register (LVISCR)

LVIOUT — LVI Output Bit

This read-only flag becomes set when the V_{DD} voltage falls below the V_{LVRX} voltage for 32 to 40 CGMXCLK cycles. See **Table 18-1**. Reset clears the LVIOUT bit.

Table 18-1. LVIOUT Bit Indication

V_{DD}		LVIOUT
At Level:	**For Number of CGMXCLK Cycles:**	
$V_{DD} > V_{LVRX} + V_{LVHX}$	Any	0
$V_{DD} < V_{LVRX}$	< 32 CGMXCLK cycles	0
$V_{DD} < V_{LVRX}$	Between 32 & 40 CGMXCLK cycles	0 or 1
$V_{DD} < V_{LVRX}$	> 40 CGMXCLK cycles	1
$V_{LVRX} < V_{DD} < V_{LVRX} + V_{LVHX}$	Any	Previous value

TRPSEL — LVI Trip Select Bit

This bit selects the LVI trip point. Reset clears this bit.

1 = 5 percent tolerance. The trip point and recovery point are determined by V_{LVR1} and V_{LVH1}, respectively.

0 = 10 percent tolerance. The trip point and recovery point are determined by V_{LVR2} and V_{LVH2}, respectively.

NOTE: *If LVIRST and LVIPWR are at logic 0, note that when changing the tolerence, LVI reset will be generated if the supply voltage is below the trip point.*

18.6 LVI Interrupts

The LVI module does not generate interrupt requests.

18.7 Wait Mode

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

With the LVIPWR bit in the configuration register programmed to logic 1, the LVI module is active after a WAIT instruction.

With the LVIRST bit in the configuration register programmed to logic 1, the LVI module can generate a reset and bring the MCU out of wait mode.

Section 19. Analog-to-Digital Converter (ADC)

19.1 Contents

19.2 Introduction

This section describes the 10-bit analog-to-digital converter (ADC).

19.3 Features

Features of the ADC module include:

- 10 channels with multiplexed input
- Linear successive approximation
- 10-bit resolution, 8-bit accuracy
- Single or continuous conversion
- Conversion complete flag or conversion complete interrupt
- Selectable ADC clock
- Left or right justified result
- Left justified sign data mode
- High impedance buffered ADC input

19.4 Functional Description

Ten ADC channels are available for sampling external sources at pins PTC1/ATD9:PTC0/ATD8 and PTB7/ATD7:PTB0/ATD0. To achieve the best possible accuracy, these pins are implemented as input-only pins when the analog-to-digital (A/D) feature is enabled. An analog multiplexer allows the single ADC converter to select one of the 10 ADC channels as ADC voltage IN (ADCVIN). ADCVIN is converted by the successive approximation algorithm. When the conversion is completed, the ADC places the result in the ADC data register (ADRH and ADRL) and sets a flag or generates an interrupt. See **Figure 19-1**.

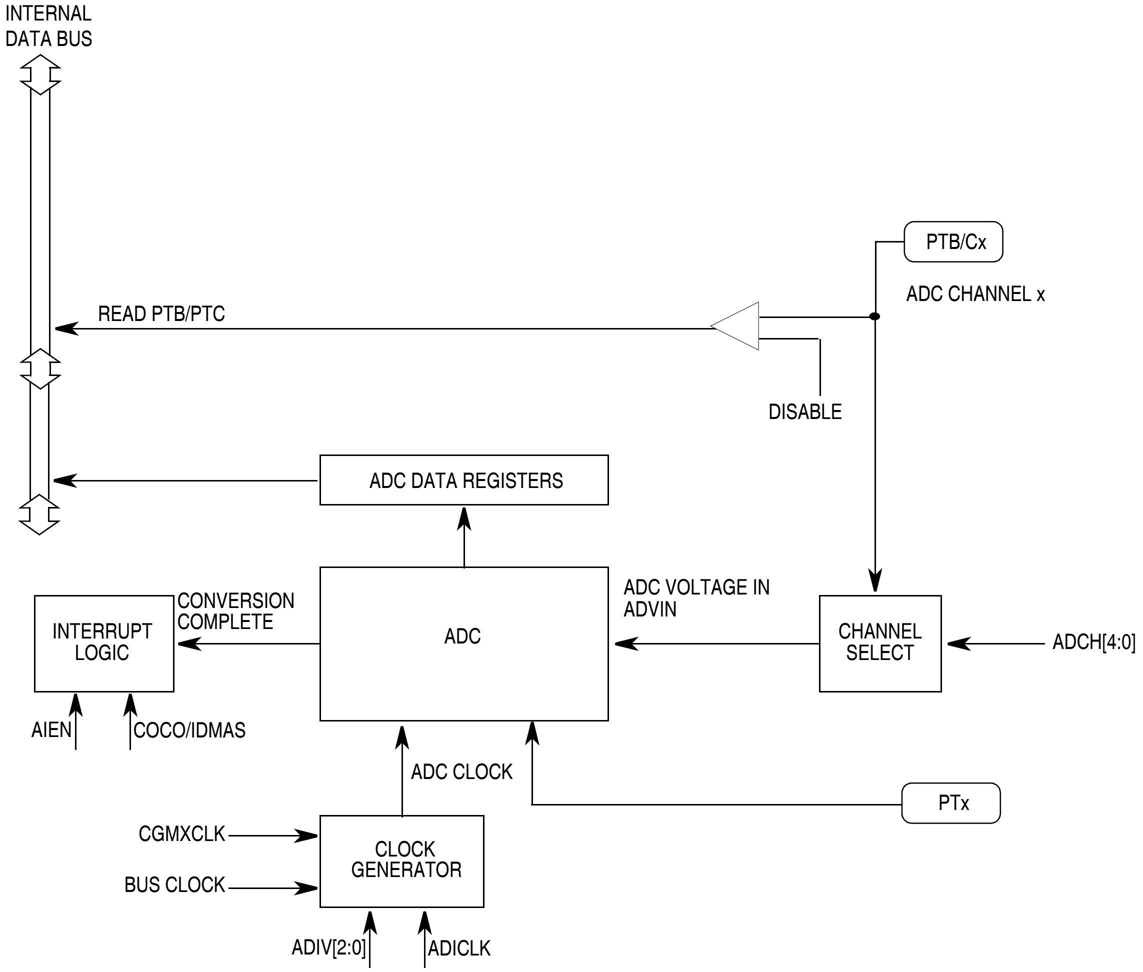


Figure 19-1. ADC Block Diagram

19.4.1 ADC Port I/O Pins

PTC1/ATD9:PTC0/ATD8 and PTB7/ATD7:PTB0/ATD0 are general-purpose I/O pins that are shared 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 logic when that port is selected by the ADC MUX. The remaining ADC channels/port pins are controlled by the port logic and can be used as general-purpose input/output (I/O) pins. Writes to the port register or DDR will not have any affect on the port pin that is selected by the ADC. Read of a port pin which is in use by the ADC will return a logic 0.

19.4.2 Voltage Conversion

When the input voltage to the ADC equals V_{REFH}, the ADC converts the signal to \$3FF (full scale). If the input voltage equals V_{REFL}, the ADC converts it to \$000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. All other input voltages will result in \$3FF if greater than V_{REFH} and \$000 if less than V_{REFL}.

NOTE: *Input voltage should not exceed the analog supply voltages. See* ***21.14 Analog-to-Digital Converter (ADC) Characteristics****.*

19.4.3 Conversion Time

Conversion starts after a write to the ADSCR. A conversion is between 16 and 17 ADC clock cycles, therefore:

$$\text{Conversion time} = \frac{\text{16 to17 ADC cycles}}{\text{ADC frequency}}$$

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

The ADC conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is either the bus clock or CGMXCLK and is selectable by ADICLK located in the ADC clock register. For example, if CGMXCLK is 4 MHz and is selected as the ADC input clock source, the ADC input clock divide-by-2 prescale is selected and the bus frequency is 8 MHz:

$$\text{Conversion time} = \frac{\text{16 to17 ADC cycles}}{\text{4 MHz/2}} = 8 \text{ to } 8.5\ \mu\text{s}$$

$$\text{Number of bus cycles} = 8\ \mu\text{s x 8 MHz} = \text{64 to 68 cycles}$$

NOTE: *The ADC frequency must be between* f_{ADIC} *minimum and* f_{ADIC} *maximum to meet A/D specifications. See* ***21.14 Analog-to-Digital Converter (ADC) Characteristics****.*

Since an ADC cycle may be comprised of several bus cycles (four in the previous example) and the start of a conversion is initiated by a bus cycle write to the ADSCR, from zero to four additional bus cycles may occur before the start of the initial ADC cycle. This results in a fractional ADC cycle and is represented as the “17th” cycle.

19.4.4 Continuous Conversion

In the continuous conversion mode, the ADC data registers ADRH and ADRL 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 for the next several conversions until the next write of the ADC status and control register or the next read of the ADC data register.

19.4.5 Result Justification

The conversion result may be formatted in four different ways:

- Left justified
- Right justified
- Left Justified sign data mode
- 8-bit truncation mode

All four of these modes are controlled using MODE0 and MODE1 bits located in the ADC clock register (ADCR).

Left justification will place the eight most significant bits (MSB) in the corresponding ADC data register high, ADRH. This may be useful if the result is to be treated as an 8-bit result where the two least significant bits (LSB), located in the ADC data register low, ADRL, can be ignored. However, ADRL must be read after ADRH or else the interlocking will prevent all new conversions from being stored.

Right justification will place only the two MSBs in the corresponding ADC data register high, ADRH, and the eight LSBs in ADC data register low, ADRL. This mode of operation typically is used when a 10-bit unsigned result is desired.

Left justified sign data mode is similar to left justified mode with one exception. The MSB of the 10-bit result, AD9 located in ADRH, is complemented. This mode of operation is useful when a result, represented as a signed magnitude from mid-scale, is needed. Finally, 8-bit truncation mode will place the eight MSBs in ADC data register low, ADRL. The two LSBs are dropped. This mode of operation is used when compatibility with 8-bit ADC designs are required. No interlocking between ADRH and ADRL is present.

NOTE: *Quantization error is affected when only the most significant eight bits are used as a result. See* ***Figure 19-2****.*

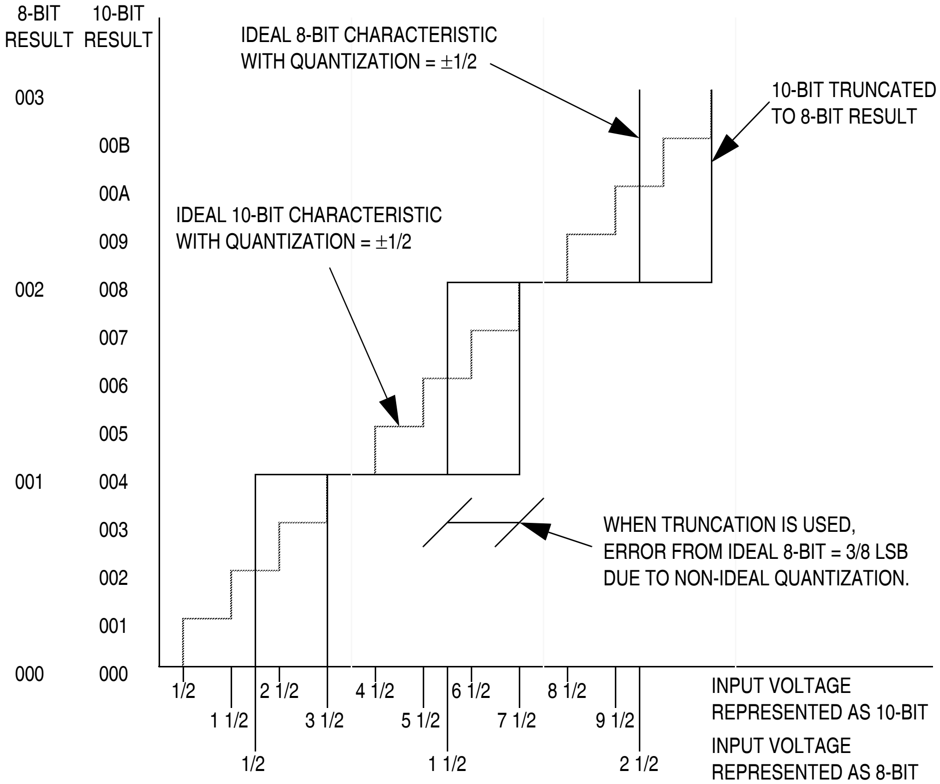


Figure 19-2. 8-Bit Truncation Mode Error

19.4.6 Monotonicity

The conversion process is monotonic and has no missing codes.

19.5 Interrupts

When the AIEN bit is set, the ADC module is capable of generating a CPU interrupt 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.

19.6 Wait Mode

The WAIT instruction can put the MCU in low power-consumption standby 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 ADCH[4:0] in the ADC status and control register before executing the WAIT instruction.

19.7 I/O Signals

The ADC module has 10 input signals that are shared with port B and port C.

19.7.1 ADC Analog Power Pin (V_{DDAD})

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

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

19.7.2 ADC Analog Ground Pin (V_{SSAD})

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

19.7.3 ADC Voltage Reference Pin (V_{REFH})

V_{REFH} is the power supply for setting the reference voltage V_{REFH}. Connect the V_{REFH} pin to the same voltage potential as V_{DDAD}. There will be a finite current associated with V_{REFH}. See **Section 21. Electrical Specifications**.

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

19.7.4 ADC Voltage Reference Low Pin (V_{REFL})

V_{REFL} is the lower reference supply for the ADC. Connect the V_{REFL} pin to the same voltage potential as V_{SSAD}. A finite current will be associated with V_{REFL}. See **Section 21. Electrical Specifications**.

NOTE: *In the 56-pin shrink dual in-line package (SDIP), V_{REFL} and V_{SSAD} are tied together.*

19.7.5 ADC Voltage In (ADVIN)

ADVIN is the input voltage signal from one of the 10 ADC channels to the ADC module.

19.7.6 ADC External Connections

This section describes the ADC external connections: V_{REFH} and V_{REFL}, ANx, and grounding.

19.7.6.1 V_{REFH} and V_{REFL}

Both ac and dc current are drawn through the V_{REFH} and V_{REFL} loop. The AC current is in the form of current spikes required to supply charge to the capacitor array at each successive approximation step. The current flows through the internal resistor string. The best external component to meet both these current demands is a capacitor in the 0.01 μF to 1 μF range with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as close as possible to the package pins. Resistance in the path is not recommended because the dc current will cause a voltage drop which could result in conversion errors.

19.7.6.2 ANx

Empirical data shows that capacitors from the analog inputs to V_{REFL} improve ADC performance. 0.01 μF and 0.1 μF capacitors with good high-frequency characteristics are sufficient. These capacitors must be placed as close as possible to the package pins.

19.7.6.3 Grounding

In cases where separate power supplies are used for analog and digital power, the ground connection between these supplies should be at the V_{SSA} pin. This should be the only ground connection between these supplies if possible. The V_{SSA} pin makes a good single point ground location. Connect the V_{REFL} pin to the same potential as V_{SSAD} at the single point ground location.

19.8 I/O Registers

These I/O registers control and monitor operation of the ADC:

- ADC status and control register, ADSCR
- ADC data registers, ADRH and ARDL
- ADC clock register, ADCLK

19.8.1 ADC Status and Control Register

This section describes the function of the ADC status and control register (ADSCR). Writing ADSCR aborts the current conversion and initiates a new conversion.

Address: $0040

	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 19-3. ADC Status and Control Register (ADSCR)

COCO — Conversions Complete Bit

When 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 ADC status and control register is written or whenever the ADC data register is read.

If AIEN bit is a logic 1, the COCO is a read/write bit. 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 allowed when this bit is cleared. Reset clears the ADCO bit.

1 = Continuous ADC conversion
0 = One ADC conversion

ADCH[4:0] — ADC Channel Select Bits

ADCH4, ADCH3, ADCH2, ADCH1, and ADCH0 form a 5-bit field which is used to select one of 10 ADC channels. The ADC channels are detailed in **Table 19-1**.

NOTE: *Take care to prevent switching noise from corrupting the analog signal when simultaneously using a port pin as both an analog and digital input.*

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 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 19-1** are used to verify the operation of the ADC both in production test and for user applications.

Table 19-1. Mux Channel Select

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	Input Select
0	0	0	0	0	PTB0/ATD0
0	0	0	0	1	PTB1/ATD1
0	0	0	1	0	PTB2/ATD2
0	0	0	1	1	PTB3/ATD3
0	0	1	0	0	PTB4/ATD4
0	0	1	0	1	PTB5/ATD5
0	0	1	1	0	PTB6/ATD6
0	0	1	1	1	PTB7/ATD7
0	1	0	0	0	PTC0/ATD8
0	1	0	0	1	PTC1/ATD9 ***
0	1	0	1	0	Unused *
0	1	0	1	1	Ø
0	1	1	0	0	Ø
0	1	1	0	1	Ø
0	1	1	1	0	Ø
0	1	1	1	1	Ø
1	0	0	0	0	Ø
1	1	0	1	0	Unused *
1	1	0	1	1	Reserved **
1	1	1	0	0	Unused *
1	1	1	0	1	V_{REFH}
1	1	1	1	0	V_{REFL}
1	1	1	1	1	[ADC power off]

* If any unused channels are selected, the resulting ADC conversion will be unknown.

** Used for factory testing.

*** ATD9 is not avialable in the 56-pin SDIP package.

19.8.2 ADC Data Register High

In left justified mode, this 8-bit result register holds the eight MSBs of the 10-bit result. This register is updated each time an ADC single channel conversion completes. Reading ADRH latches the contents of ADRL until ADRL is read. Until ADRL is read, all subsequent ADC results will be lost.

Address: $0041

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
Write:	R	R	R	R	R	R	R	R
Reset:	Unaffected by reset							

R = Reserved

Figure 19-4. ADC Data Register High (ADRH) Left Justified Mode

In right justified mode, this 8-bit result register holds the two MSBs of the 10-bit result. All other bits read as 0. This register is updated each time a single channel ADC conversion completes. Reading ADRH latches the contents of ADRL until ADRL is read. Until ADRL is read, all subsequent ADC results will be lost.

Address: $0041

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	AD9	AD8
Write:	R	R	R	R	R	R	R	R
Reset:	Unaffected by reset							

R = Reserved

Figure 19-5. ADC Data Register High (ADRH) Right Justified Mode

19.8.3 ADC Data Register Low

In left justified mode, this 8-bit result register holds the two LSBs of the 10-bit result. All other bits read as 0. This register is updated each time a single channel ADC conversion completes. Reading ADRH latches the contents of ADRL until ADRL is read. Until ADRL is read, all subsequent ADC results will be lost.

Address: $0042

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AD1	AD0	0	0	0	0	0	0
Write:	R	R	R	R	R	R	R	R
Reset:	Unaffected by reset							

R = Reserved

Figure 19-6. ADC Data Register Low (ADRL)
Left Justified Mode

In right justified mode, this 8-bit result register holds the eight LSBs of the 10-bit result. This register is updated each time an ADC conversion completes. Reading ADRH latches the contents of ADRL until ADRL is read. Until ADRL is read, all subsequent ADC results will be lost.

Address: $0042

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

R = Reserved

Figure 19-7. ADC Data Register Low (ADRL)
Right Justified Mode

In 8-bit mode, this 8-bit result register holds the eight MSBs of the 10-bit result. This register is updated each time an ADC conversion completes. In 8-bit mode, this register contains no interlocking with ADRH.

Address: $0042

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
Write:	R	R	R	R	R	R	R	R
Reset:	Unaffected by reset							

R = Reserved

Figure 19-8. ADC Data Register Low (ADRL) 8-Bit Mode

19.8.4 ADC Clock Register

This register selects the clock frequency for the ADC, selecting between modes of operation.

Address: $0043

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ADIV2	ADIV1	ADIV0	ADICLK	MODE1	MODE0	0	0
Write:								R
Reset:	0	1	1	1	0	0	0	0

R = Reserved

Figure 19-9. ADC Clock Register (ADCLK)

ADIV2:ADIV0 — ADC Clock Prescaler Bits

ADIV2, ADIV1, and ADIV0 form a 3-bit field which selects the divide ratio used by the ADC to generate the internal ADC clock. **Table 19-2** shows the available clock configurations.

Table 19-2. 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	X	ADC input clock /16

X = don't care

ADICLK — ADC Input Clock Select Bit

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

If the external clock (CGMXCLK) is equal to or greater than 1 MHz, CGMXCLK can be used as the clock source for the ADC. If CGMXCLK is less than 1 MHz, use the PLL-generated bus clock as the clock source. As long as the internal ADC clock is at f_{ADIC}, correct operation can be guaranteed. See **21.14 Analog-to-Digital Converter (ADC) Characteristics**.

1 = Internal bus clock
0 = External clock, CGMXCLK

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

MODE1:MODE0 — Modes of Result Justification

MODE1:MODE0 selects between four modes of operation. The manner in which the ADC conversion results will be placed in the ADC data registers is controlled by these modes of operation. Reset returns right-justified mode.

00 = 8-bit truncation mode
01 = Right justified mode
10 = Left justified mode
11 = Left justified sign data mode

Section 20. Power-On Reset (POR)

20.1 Contents

20.2 Introduction

This section describes the power-on reset (POR) module.

20.3 Functional Description

The POR module provides a known, stable signal to the MCU at power-on. This signal tracks V_{DD} until the MCU generates a feedback signal to indicate that it is properly initialized. At this time, the POR drives its output low.

The POR is not a brown-out detector, low-voltage detector, or glitch detector. V_{DD} at the POR must go completely to 0 to reset the MCU. To detect power-loss conditions, use a low-voltage inhibit module (LVI) or other suitable circuit.

Section 21. Electrical Specifications

21.1 Contents

21.2 Introduction

This section contains electrical and timing specifications. These values are design targets and have not yet been fully characterized.

21.3 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. For guaranteed operating conditions, refer to **21.6 DC Electrical Characteristics (V_{DD} = 5.0 Vdc ± 10%)**.*

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
Input high voltage	V_{HI}	V_{DD} + 4 maximum	V
Maximum current per pin excluding V_{DD} and V_{SS}	I	± 25	mA
Storage temperature	T_{STG}	–55 to +150	°C
Maximum current out of V_{SS}	I_{MVSS}	100	mA
Maximum current Into V_{DD}	I_{MVDD}	100	mA

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} be constrained to the range $V_{SS} \leq (V_{In}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either V_{SS} or V_{DD}).*

21.4 Functional Operating Range

Characteristic	Symbol	Value	Unit
Operating temperature range (see notes) MC68HC908MR24CFU MC68HC908MR24VFU	T_A	–40 to 85 –40 to 105	°C
Operating voltage range	V_{DD}	5.0 ± 10%	V

Notes:
See Freescale representative for temperature availability.
C = Extended temperature range (–40°C to +85°C)
V = Automotive temperature range (–40°C to +105°C)

21.5 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance, 64-pin QFP	θ_{JA}	76	°C/W
I/O pin power dissipation	$P_{I/O}$	User determined	W
Power dissipation[1]	P_D	$P_D = (I_{DD} \times V_{DD}) + P_{I/O} = K/(T_J + 273°C)$	W
Constant[2]	K	$P_D \times (T_A + 273°C) + P_D^2 \times \theta_{JA}$	W/°C
Average junction temperature	T_J	$T_A + (P_D \times \theta_{JA})$	°C
Maximum junction temperature	T_{JM}	125	°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.

21.6 DC Electrical Characteristics (V_{DD} = 5.0 Vdc ± 10%)

Characteristic		Min	Typ[2]	Max	Unit
Output high voltage (I_{Load} = –2.0 mA) all I/O pins	V_{OH}	V_{DD} –0.8	—	—	V
Output low voltage (I_{Load} = 1.6 mA) all I/O pins	V_{OL}	—	—	0.4	V
PWM pin output source current ($V_{OH} = V_{DD}$ –0.8 V)	I_{OH}	–7	—	—	mA
PWM pin output sink current (V_{OL} = 0.8 V)	I_{OL}	20	—	—	mA
Input high voltage, all ports, IRQs, RESET, OSC1	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
Input low voltage, all ports, IRQs, RESET, OSC1	V_{IL}	V_{SS}	—	0.3 x V_{DD}	V
V_{DD} supply current Run (3) Wait (4) Quiescent(5)	I_{DD}	— — —	— — —	40 20 700	mA mA µA
I/O ports hi-Z leakage current	I_{IL}	—	—	± 10	µA
Input current	I_{In}	—	—	± 1	µA
Capacitance Ports (as input or output)	C_{OUT} C_{IN}	— —	— —	12 8	pF
Low-voltage inhibit reset [9]	V_{LVR1}	4.33	4.45	4.75	V
Low-voltage reset/recover hysteresis	V_{LVH1}	50	100	—	mV
Low-voltage inhibit reset	V_{LVR2}	3.95	—	4.35	V
Low-voltage reset/recover hysteresis	V_{LVH2}	150	—	250	mV
POR re-arm voltage(6)	V_{POR}	0	—	100	mV
POR rise time ramp rate(8)	R_{POR}	0.035	—	—	V/ms
POR reset voltage	V_{PORRST}	0	700	800	V
Monitor mode entry voltage (on IRQ)	V_{Hi}	V_{DD} + 2	—	V_{DD} + Hi	V

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. Typical values reflect average measurements at midpoint of voltage range, 25°C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{osc} = 8.2 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} = 8.2 MHz); all inputs 0.2 V from rail; no dc loads; less than 100 pF on all outputs. C_L = 20 pF on OSC2; all ports configured as inputs; OSC2 capacitance linearly affects wait I_{DD}; measured with PLL and LVI enabled.
5. Quiescent I_{DD} measured with PLL and LVI disengaged, OCS1 grounded, no port pins sourcing current. It is measured through combination of V_{DD}, V_{DDAD}, and V_{DDA}.
6. Maximum is highest voltage that POR is guaranteed.
7. Maximum is highest voltage that POR is possible.
8. If minimum V_{DD} is not reached before the internal POR is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached
9. The low-voltage inhibit reset is software selectable. Refer to **Section 18. Low-Voltage Inhibit (LVI)**.

21.7 FLASH Memory Characteristics

Characteristic	Symbol/ Description	Min	Max	Units
FLASH pages per row		8	8	Pages
FLASH bytes per page		8	8	Bytes
FLASH read bus clock frequency	f_{Read}(1)	32 K	8.4 M	Hz
FLASH charge pump clock frequency (see **4.5 FLASH Charge Pump Frequency Control**)	f_{Pump}(2)	1.8	2.5	MHz
FLASH block/bulk erase time	t_{Erase}	100	—	ms
FLASH high-voltage kill time	t_{Kill}	200	—	µs
FLASH return to read time	t_{HVD}	50	—	µs
FLASH page program pulses	fls_{Pulses}(3)	—	100	Pulses
FLASH page program step size	t_{PROG}(4)	1.0	1.2	ms
FLASH cumulative program operation per row between erase cycles	t_{ROW}(5)	—	8	Page programming cycles
FLASH HVEN low to MARGIN high time	t_{HVTV}	50	—	µs
FLASH MARGIN high to PGM low time	t_{VTP}	150	—	µs
FLASH row erase endurance(6)	—	100	—	Cycles
FLASH row program endurance(7)	—	100	—	Cycles
FLASH data retention time(8)	—	10	—	Years

1. f_{Read} is defined as the frequency range for which the FLASH memory can be read.
2. f_{Pump} is defined as the charge pump clock frequency required for program, erase, and margin read operations.
3. fls_{Pulses} is defined as the number of pulses used to program the FLASH using the required smart program algorithm.
4. t_{Step} is defined as the amount of time during one page program cycle that HVEN is held high.
5. t_{ROW} is defined as the cumulative time a row can see the program voltage before the row must be erased before further programming.
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 temperature range for at least the minimum time specified.

21.8 Control Timing (V_{DD} = 5.0 Vdc ± 10%)

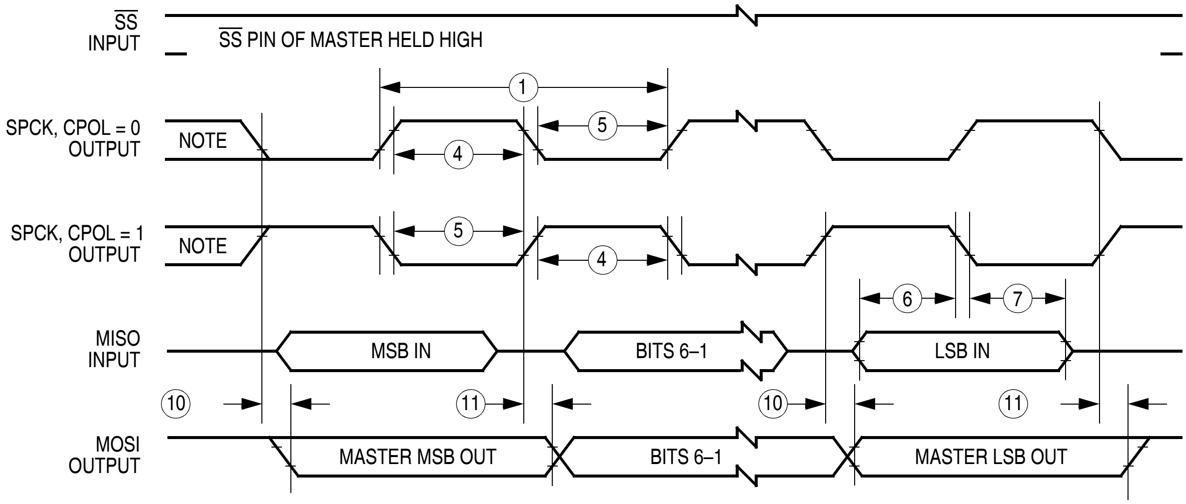
Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation[2] Crystal option External clock option[3]	f_{OSC}	 1 dc[4]	 8 32.8	MHz
Internal operating frequency	f_{OP}	—	8.2	MHz
$\overline{\text{RESET}}$ input pulse width low[5]	t_{IRL}	50	—	ns

1. V_{SS} = 0 Vdc; timing shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted
2. See **21.9 Serial Peripheral Interface Characteristics (V_{DD} = 5.0 Vdc ± 10%)** for more information.
3. No more than 10% duty cycle deviation from 50%.
4. Some modules may require a minimum frequency greater than dc for proper operation; see appropriate table for this information.
5. Minimum pulse width reset is guaranteed to be recognized. It is possible for a smaller pulse width to cause a reset.

21.9 Serial Peripheral Interface Characteristics (V_{DD} = 5.0 Vdc ± 10%)

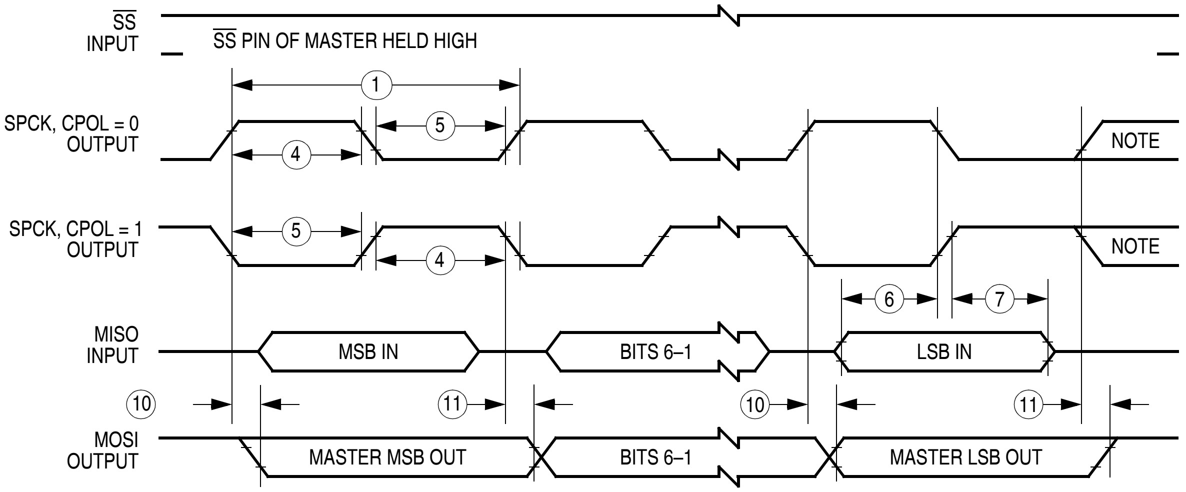
Diagram Number[1]	Characteristic[2]	Symbol	Min	Max	Unit
	Operating frequency Master Slave	$f_{OP(M)}$ $f_{OP(S)}$	$f_{OP}/128$ dc	$f_{OP}/2$ f_{OP}	MHz
1	Cycle time Master Slave	$t_{cyc(M)}$ $t_{cyc(S)}$	2 1	128 —	t_{cyc}
2	Enable lead time	$t_{Lead(S)}$	15	—	ns
3	Enable lag time	$t_{Lag(S)}$	15	—	ns
4	Clock (SPCK) high time Master Slave	$t_{SCKH(M)}$ $t_{SCKH(S)}$	100 50	— —	ns
5	Clock (SPCK) low time Master Slave	$t_{SCKL(M)}$ $t_{SCKL(S)}$	100 50	— —	ns
6	Data setup time (inputs) Master Slave	$t_{SU(M)}$ $t_{SU(S)}$	45 5	— —	ns
7	Data hold time (inputs) Master Slave	$t_{H(M)}$ $t_{H(S)}$	0 15	— —	ns
8	Access time, slave[3] CPHA = 0 CHPA = 1	$t_{A(CP0)}$ $t_{A(CP1)}$	0 0	40 20	ns
9	Disable time, slave[4]	$t_{DIS(S)}$	—	25	ns
10	Data valid time after enable edge Master Slave[5]	$t_{V(M)}$ $t_{V(S)}$	— —	10 40	ns

1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted; assumes 100 pF load on all SPI pins
2. Numbers refer to dimensions in **Figure 21-1** and **Figure 21-2**.
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 SCK pin.

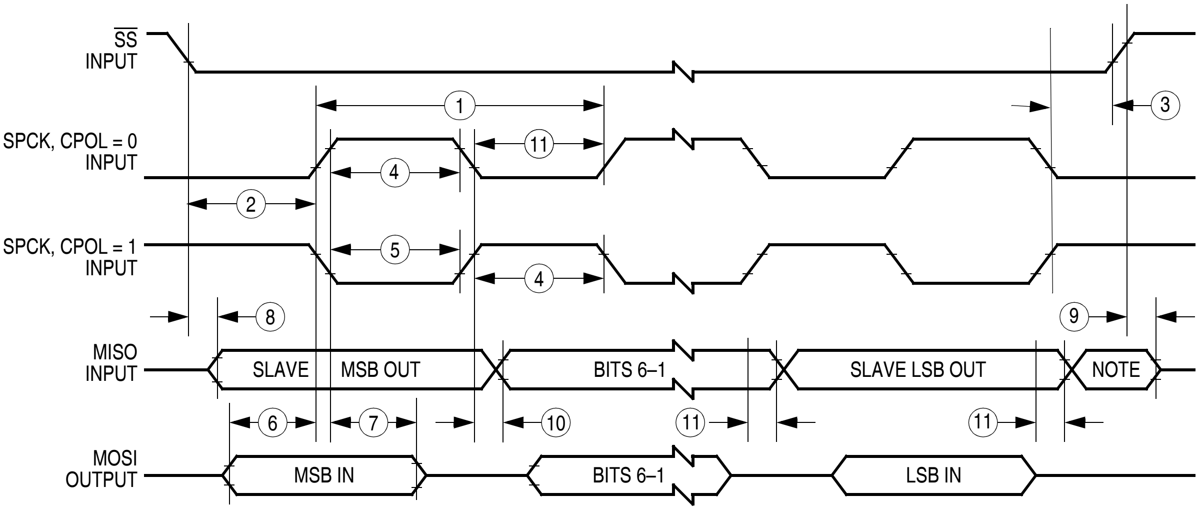
a) SPI Master Timing (CPHA = 0)



Note: This last clock edge is generated internally, but is not seen at the SCK pin.

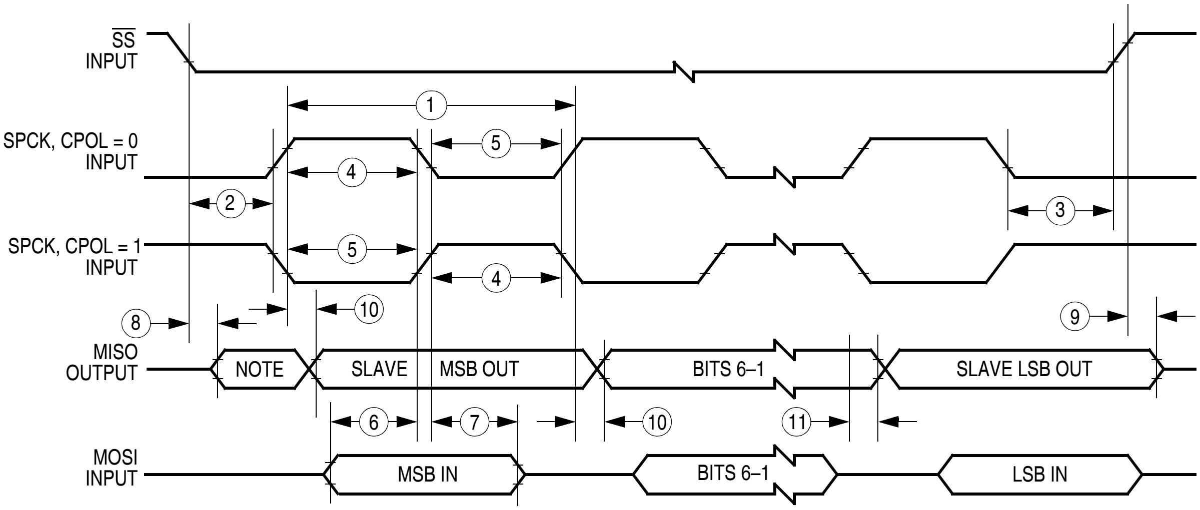
b) SPI Master Timing (CPHA = 1)

Figure 21-1. SPI Master Timing



Note: Not defined, but normally MSB of character just received

a) SPI Slave Timing (CPHA = 0)



Note: Not defined, but normally LSB of character previously transmitted

b) SPI Slave Timing (CPHA = 1)

Figure 21-2. SPI Slave Timing

21.10 TImer Interface Module Characteristics

Characteristic	Symbol	Min	Max	Unit
Input capture pulse width	$t_{TIH,}$ t_{TIL}	125	—	ns
Input clock pulse width	$t_{TCH,}$ t_{TCL}	$(1/f_{OP}) + 5$	—	ns

21.11 Clock Generation Module Component Specifications

Characteristic	Symbol	Min	Typ	Max	Notes
Crystal load capacitance	C_L	—	—	—	Consult crystal manufacturing data
Crystal fixed capacitance	C_1	—	$2 * C_L$	—	Consult crystal manufacturing data
Crystal tuning capacitance	C_2	—	$2 * C_L$	—	Consult crystal manufacturing data
Feedback bias resistor	R_B	—	22 MΩ	—	
Series resistor	R_S	0	330 kΩ	1 MΩ	Not required
Filter capacitor	C_F	—	$C_{FACT}*$ (V_{DDA}/f_{XCLK})	—	
Bypass capacitor	C_{BYP}	—	0.1 μF	—	C_{BYP} must provide low AC impedance from $f = f_{XCLK}/100$ to $100*f_{VCLK}$, so series resistance must be considered

21.12 CGM Operating Conditions

Characteristic	Symbol	Min	Typ	Max
Crystal reference frequency	f_{XCLK}	1 MHz	—	8 MHz
Range nominal multiplier	f_{NOM}	—	4.9152 MHz	—
VCO center-of-range frequency	f_{VRS}	4.9152 MHz	—	32.8 MHz
VCO frequency multiplier	N	1	—	15
VCO center of range multiplier	L	1	—	15
VCO operating frequency	f_{VCLK}	f_{VRSMIN}	—	f_{VRSMAX}

21.13 CGM Acquisition/Lock Time Specifications

Description	Symbol	Min	Typ	Max	Notes
Filter capacitor multiply factor	C_{FACT}	—	0.0154	—	F/sV
Acquisition mode time factor	K_{ACQ}	—	0.1135	—	V
Tracking mode time factor	K_{TRK}	—	0.0174	—	V
Manual mode time to stable	t_{ACQ}	—	$(8*V_{DDA})/$ $(f_{X\ CLK}*K_{ACQ})$	—	If C_F chosen correctly
Manual stable to lock time	t_{AL}	—	$(4*V_{DDA})/$ $(f_{X\ CLK}*K_{TRK})$	—	If C_F chosen correctly
Manual acquisition time	t_{Lock}	—	$t_{ACQ}+t_{AL}$	—	
Tracking mode entry frequency tolerance	Δ_{TRK}	0	—	± 3.6%	
Acquisition mode entry frequency tolerance	Δ_{ACQ}	±6.3%	—	± 7.2%	
Lock entry frequency tolerance	Δ_{Lock}	0	—	± 0.9%	
Lock exit frequency tolerance	Δ_{UNL}	±0.9%	—	± 1.8%	
Reference cycles per acquisition mode measurement	n_{ACQ}	—	32	—	
Reference cycles per tracking mode measurement	n_{TRK}	—	128	—	
Automatic mode time to stable	t_{ACQ}	n_{ACQ}/f_{XCLK}	$(8*V_{DDA})/$ $(f_{X\ CLK}*K_{ACQ})$	—	If C_F chosen correctly
Automatic stable to lock time	t_{AL}	n_{TRK}/f_{XCLK}	$(4*V_{DDA})/$ $(f_{X\ CLK}*K_{TRK})$	—	If C_F chosen correctly
Automatic lock time	t_{Lock}	—	$t_{ACQ}+t_{AL}$	—	
PLL jitter (deviation of average bus frequency over 2 ms)	f_J	0	—	$\pm (f_{XCLK})$ $*(0.025\%)$ $*(N/4)$	N = VCO freq. mult.

21.14 Analog-to-Digital Converter (ADC) Characteristics

Characteristic	Symbol	Min	Typ	Max	Unit	Notes
Supply voltage	V_{DDAD}	4.5	—	5.5	V	V_{DDAD} should be tied to the same potential as V_{DD} via separate traces
Input voltages	V_{ADIN}	0	—	V_{DDAD}	V	$V_{ADIN} <= V_{DDAD}$
Resolution	B_{AD}	10	—	10	Bits	
Absolute accuracy	A_{AD}	—	—	±4	Counts	Includes quantization
ADC internal clock	f_{ADIC}	500 k	—	1.048 M	Hz	$t_{AIC} = 1/f_{ADIC}$
Conversion range	R_{AD}	V_{SSAD}	—	V_{DDAD}	V	
Power-up time	t_{ADPU}	16	—	—	t_{AIC} cycles	
Conversion time	t_{ADC}	16	—	17	t_{AIC} cycles	
Sample time	t_{ADS}	5	—	—	t_{AIC} cycles	
Monotonicity	M_{AD}	Guaranteed				
Zero input reading	Z_{ADI}	000	—	003	Hex	$V_{ADIN} = V_{SSAD}$
Full-scale reading	F_{ADI}	3FC	—	3FF	Hex	$V_{ADIN} = V_{DDAD}$
Input capacitance	C_{ADI}	—	—	30	pF	Not tested
V_{REFH}/V_{REFL} current	I_{VREF}	—	1.6	—	mA	
Absolute accuracy (8-bit truncation mode)	A_{AD}	—	—	±1	LSB	Includes quantization
Quantization error (8-bit truncation mode)	—	—	—	+7/8 –1/8	LSB	

Section 22. Mechanical Specifications

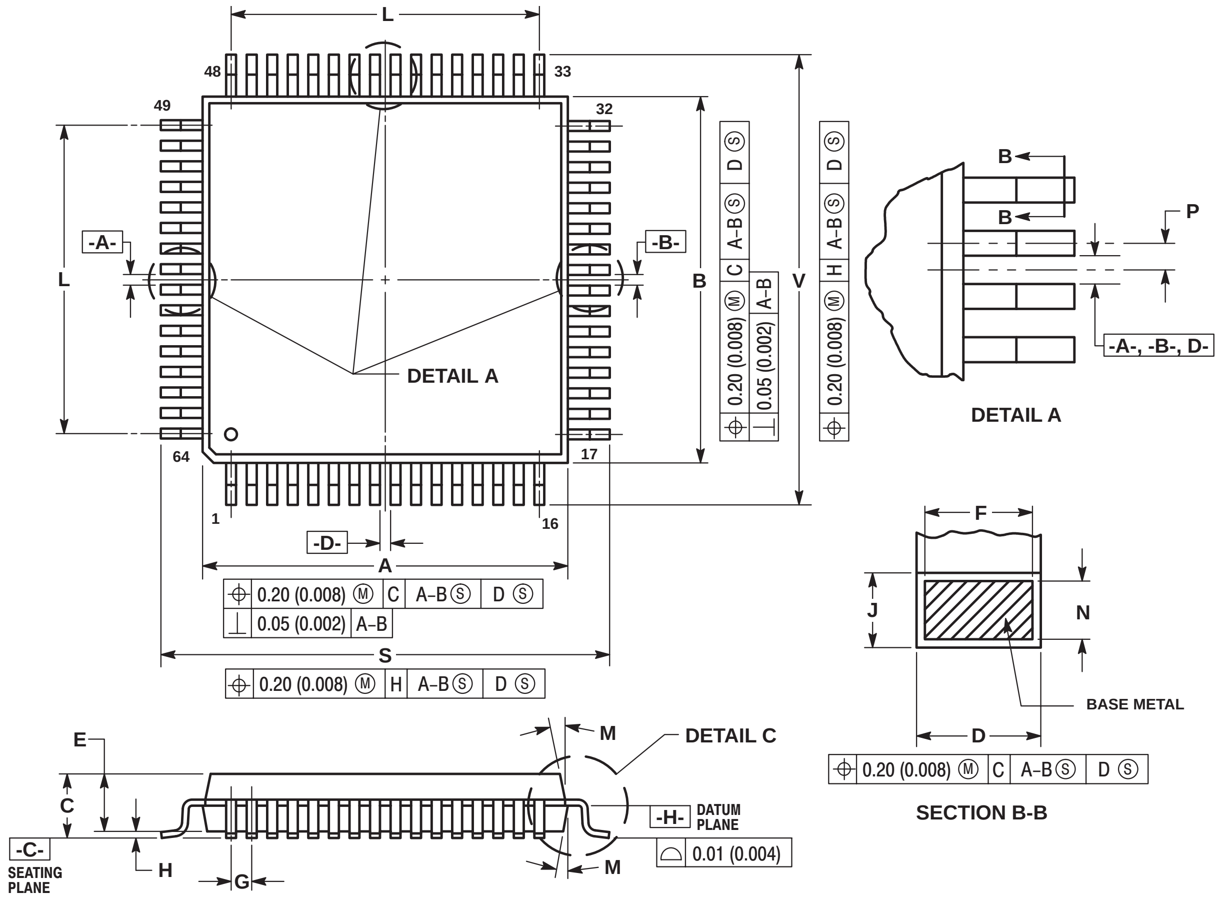
22.1 Contents

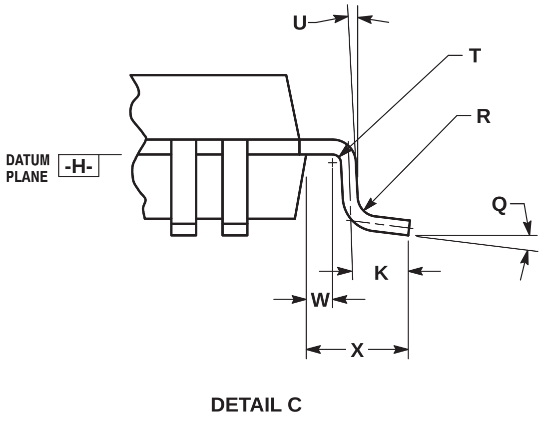
22.2 Introduction

This section gives the dimensions for the 64-lead plastic quad flat pack (QFP) and 56-pin shrink dual in-line package (SDIP).

Figure 22-1 and **Figure 22-2** show the latest package at the time of this publication. To make sure that you have the latest package specifications, please visit the Freescale website at http://freescale.com. Follow Worldwide Web on-line instructions to retrieve the current mechanical specifications.

22.3 64-Pin Plastic Quad Flat Pack (QFP)





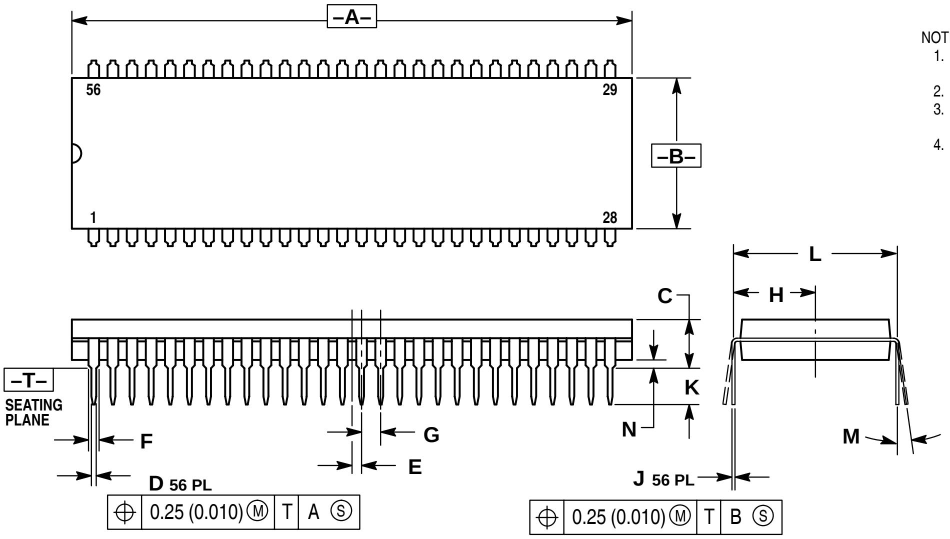
NOTES:

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
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. DATUMS A–B AND -D- TO BE DETERMINED AT DATUM PLANE -H-.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE -C-.
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. 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.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	13.90	14.10	0.547	0.555
B	13.90	14.10	0.547	0.555
C	2.15	2.45	0.085	0.096
D	0.30	0.45	0.012	0.018
E	2.00	2.40	0.079	0.094
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	12.00 REF		0.472 REF	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
P	0.40 BSC		0.016 BSC	
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	16.95	17.45	0.667	0.687
T	0.13	—	0.005	—
U	0°	—	0°	—
V	16.95	17.45	0.667	0.687
W	0.35	0.45	0.014	0.018
X	1.6 REF		0.063 REF	

Figure 22-1. MC68HC908MR24FU

22.4 56-Pin Shrink Dual In-Line Package (SDIP)



NOTES:
1. DIMENSIONING 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)

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	2.035	2.065	51.69	52.45
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
E	0.035 BSC		0.89 BSC	
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

Figure 22-2. MC68HC908MR24B

Section 23. Ordering Information

23.1 Contents

23.2 Introduction

This section contains instructions for ordering the MC68HC908MR24.

23.3 Order Numbers

Table 23-1. Order Numbers

MC Order Number[1]	Operating Temperature Range
68HC908MR24CFU 68HC908MR24VFU	– 40°C to + 85°C – 40°C to + 105°C
68HC908MR24CB 68HC908MR24VB	– 40°C to + 85°C – 40°C to + 105°C

1. FU = quad flat pack
   B = shrink dual in-line package

Glossary

A — See accumulator (A).

accumulator (A) — An 8-bit general-purpose register in the CPU08. The CPU08 uses the accumulator to hold operands and results of arithmetic and logic operations.

acquisition mode — A mode of PLL operation during startup before the PLL locks on a frequency. Also see tracking mode.

address bus — The set of wires that the CPU or DMA uses to read and write memory locations.

addressing mode — The way that the CPU determines the operand address for an instruction. The M68HC08 CPU has 16 addressing modes.

ALU — See arithmetic logic unit (ALU).

arithmetic logic unit (ALU) — The portion of the CPU that contains the logic circuitry to perform arithmetic, logic, and manipulation operations on operands.

asynchronous — Refers to logic circuits and operations that are not synchronized by a common reference signal.

baud rate — The total number of bits transmitted per unit of time.

BCD — See binary-coded decimal (BCD).

binary — Relating to the base 2 number system.

binary-coded decimal (BCD) — A notation that uses 4-bit binary numbers to represent the 10 decimal digits and that retains the same positional structure of a decimal number. For example,

234 (decimal) = 0010 0011 0100 (BCD)

binary number system — The base 2 number system, having two digits, 0 and 1. Binary arithmetic is convenient in digital circuit design because digital circuits have two permissible voltage levels, low and high. The binary digits 0 and 1 can be interpreted to correspond to the two digital voltage levels.

bit — A binary digit. A bit has a value of either logic 0 or logic 1.

branch instruction — An instruction that causes the CPU to continue processing at a memory location other than the next sequential address.

break interrupt — A software interrupt caused by the appearance on the internal address bus of the same value that is written in the break address registers.

break module — A module in the M68HC08 Family. The break module allows software to halt program execution at a programmable point to enter a background routine.

breakpoint — A number written into the break address registers of the break module. When a number appears on the internal address bus that is the same as the number in the break address registers, the CPU executes the software interrupt instruction (SWI).

bus — A set of wires that transfers logic signals.

bus clocks — There are two bus clocks, IT12 and IT23. These clocks are generated by the CGM and distributed throughout the MCU by the SIM. The frequency of the bus clocks, or operating frequency, is f_{OP}. While the frequency of these two clocks is the same, the phase is different.

byte — A set of eight bits.

C — The carry/borrow bit in the condition code register. The CPU08 sets the carry/borrow bit when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some logical operations and data manipulation instructions also clear or set the carry/borrow bit (as in bit test and branch instructions and shifts and rotates).

CCR — See condition code register.

central processor unit (CPU) — The primary functioning unit of any computer system. The CPU controls the execution of instructions.

CGM — See clock generator module (CGM).

clear — To change a bit from logic 1 to logic 0; the opposite of set.

clock — A square wave signal used to synchronize events in a computer.

clock generator module (CGM) — A module in the M68HC08 Family. The CGM generates a base clock signal from which the system clocks are derived. The CGM may include a crystal oscillator circuit and/or phase-locked loop (PLL) circuit.

comparator — A device that compares the magnitude of two inputs. A digital comparator defines the equality or relative differences between two binary numbers.

computer operating properly module (COP) — A counter module in the M68HC08 Family that resets the MCU if allowed to overflow.

condition code register (CCR) — An 8-bit register in the CPU08 that contains the interrupt mask bit and five bits that indicate the results of the instruction just executed.

control bit — One bit of a register manipulated by software to control the operation of the module.

control unit — One of two major units of the CPU. The control unit contains logic functions that synchronize the machine and direct various operations. The control unit decodes instructions and generates the internal control signals that perform the requested operations. The outputs of the control unit drive the execution unit, which contains the arithmetic logic unit (ALU), CPU registers, and bus interface.

COP — See computer operating properly module (COP).

counter clock — The input clock to the TIM counter. This clock is an output of the prescaler sub-module. The frequency of the counter clock is f_{TCNT}, and the period is t_{TCNT}.

CPU — See central processor unit (CPU).

CPU08 — The central processor unit of the M68HC08 Family.

CPU cycles — A CPU clock cycle is one period of the internal bus-rate clock, f_{OP}, normally derived by dividing a crystal oscillator source by two or more so the high and low times will be equal. The length of time required to execute an instruction is measured in CPU clock cycles.

CPU registers — Memory locations that are wired directly into the CPU logic instead of being part of the addressable memory map. The CPU always has direct access to the information in these registers. The CPU registers in an M68HC08 are:

- A, 8-bit accumulator
- H:X, 16-bit index register
- SP, 16-bit stack pointer
- PC, 16-bit program counter
- CCR, condition code register containing the V, H, I, N, Z, and C bits

CSIC — customer-specified integrated circuit

cycle time — The period of the operating frequency: $t_{CYC} = 1/f_{OP}$.

decimal number system — Base 10 numbering system that uses the digits zero through nine.

direct memory access module (DMA) — An M68HC08 Family module that can perform data transfers between any two CPU-addressable locations without CPU intervention. For transmitting or receiving blocks of data to or from peripherals, DMA transfers are faster and more code-efficient than CPU interrupts.

DMA — See direct memory access module (DMA).

DMA service request — A signal from a peripheral to the DMA module that enables the DMA module to transfer data.

duty cycle — A ratio of the amount of time the signal is on versus the time it is off. Duty cycle is usually represented by a percentage.

EEPROM — Electrically erasable, programmable, read-only memory. A non-volatile type of memory that can be electrically reprogrammed.

EPROM — Erasable, programmable, read-only memory. A non-volatile type of memory that can be erased by exposure to an ultraviolet light source and then reprogrammed.

exception — An event such as an interrupt or a reset that stops the sequential execution of the instructions in the main program.

external interrupt module (IRQ) — A module in the M68HC08 Family with both dedicated external interrupt pins and port pins that can be enabled as interrupt pins.

fetch — To copy data from a memory location into the accumulator.

firmware — Instructions and data programmed into non-volatile memory.

free-running counter — A device that counts from zero to a predetermined number, then rolls over to zero and begins counting again.

full-duplex transmission — Communication on a channel in which data can be sent and received simultaneously.

H — The upper byte of the 16-bit index register (H:X) in the CPU08.

H — The half-carry bit in the condition code register of the CPU08. This bit indicates a carry from the low-order four bits of the accumulator value to the high-order four bits. The half-carry bit is required for binary-coded decimal arithmetic operations. The decimal adjust accumulator (DAA) instruction uses the state of the H and C bits to determine the appropriate correction factor.

hexadecimal — Base 16 numbering system that uses the digits 0 through 9 and the letters A through F.

high byte — The most significant eight bits of a word.

illegal address — An address not within the memory map.

illegal opcode — A non-existent opcode.

I — The interrupt mask bit in the condition code register of the CPU08. When I is set, all interrupts are disabled.

index register (H:X) — A 16-bit register in the CPU08. The upper byte of H:X is called H. The lower byte is called X. In the indexed addressing modes, the CPU uses the contents of H:X to determine the effective address of the operand. H:X can also serve as a temporary data storage location.

input/output (I/O) — Input/output interfaces between a computer system and the external world. A CPU reads an input to sense the level of an external signal and writes to an output to change the level on an external signal.

instructions — Operations that a CPU can perform. Instructions are expressed by programmers as assembly language mnemonics. A CPU interprets an opcode and its associated operand(s) and instruction.

interrupt — A temporary break in the sequential execution of a program to respond to signals from peripheral devices by executing a subroutine.

interrupt request — A signal from a peripheral to the CPU intended to cause the CPU to execute a subroutine.

I/O — See input/output (I/0).

IRQ — See external interrupt module (IRQ).

jitter — Short-term signal instability.

latch — A circuit that retains the voltage level (logic 1 or logic 0) written to it for as long as power is applied to the circuit.

latency — The time lag between instruction completion and data movement.

least significant bit (LSB) — The rightmost digit of a binary number.

logic 1 — A voltage level approximately equal to the input power voltage (V_{DD}).

logic 0 — A voltage level approximately equal to the ground voltage (V_{SS}).

low byte — The least significant eight bits of a word.

low-voltage inhibit module (LVI) — A module in the M68HC08 Family that monitors power supply voltage.

LVI — See low-voltage inhibit module (LVI).

M68HC08 — A Freescale family of 8-bit MCUs.

mark/space — The logic 1/logic 0 convention used in formatting data in serial communication.

mask — 1. A logic circuit that forces a bit or group of bits to a desired state. 2. A photomask used in integrated circuit fabrication to transfer an image onto silicon.

mask option — An optional microcontroller feature that the customer chooses to enable or disable.

mask option register (MOR) — An EPROM location containing bits that enable or disable certain MCU features.

MCU — Microcontroller unit. See microcontroller.

memory location — Each M68HC08 memory location holds one byte of data and has a unique address. To store information in a memory location, the CPU places the address of the location on the address bus, the data information on the data bus, and asserts the write signal. To read information from a memory location, the CPU places the address of the location on the address bus and asserts the read signal. In response to the read signal, the selected memory location places its data onto the data bus.

memory map — A pictorial representation of all memory locations in a computer system.

microcontroller — Microcontroller unit (MCU). A complete computer system, including a CPU, memory, a clock oscillator, and input/output (I/O) on a single integrated circuit.

modulo counter — A counter that can be programmed to count to any number from zero to its maximum possible modulus.

monitor ROM — A section of ROM that can execute commands from a host computer for testing purposes.

MOR — See mask option register (MOR).

most significant bit (MSB) — The leftmost digit of a binary number.

multiplexer — A device that can select one of a number of inputs and pass the logic level of that input on to the output.

N — The negative bit in the condition code register of the CPU08. The CPU sets the negative bit when an arithmetic operation, logical operation, or data manipulation produces a negative result.

nibble — A set of four bits (half of a byte).

object code — The output from an assembler or compiler that is itself executable machine code or is suitable for processing to produce executable machine code.

opcode — A binary code that instructs the CPU to perform an operation.

open-drain — An output that has no pullup transistor. An external pullup device can be connected to the power supply to provide the logic 1 output voltage.

operand — Data on which an operation is performed. Usually, a statement consists of an operator and an operand. For example, the operator may be an add instruction, and the operand may be the quantity to be added.

oscillator — A circuit that produces a constant frequency square wave that is used by the computer as a timing and sequencing reference.

OTPROM — One-time programmable read-only memory. A non-volatile type of memory that cannot be reprogrammed.

overflow — A quantity that is too large to be contained in one byte or one word.

page zero — The first 256 bytes of memory (addresses $0000–$00FF).

parity — An error-checking scheme that counts the number of logic 1s in each byte transmitted. In a system that uses odd parity, every byte is expected to have an odd number of logic 1s. In an even parity system, every byte should have an even number of logic 1s. In the transmitter, a parity generator appends an extra bit to each byte to make the number of logic 1s odd for odd parity or even for even parity. A parity checker in the receiver counts the number of logic 1s in each byte. The parity checker generates an error signal if it finds a byte with an incorrect number of logic 1s.

PC — See program counter (PC).

peripheral — A circuit not under direct CPU control.

phase-locked loop (PLL) — An oscillator circuit in which the frequency of the oscillator is synchronized to a reference signal.

PLL — See phase-locked loop (PLL).

pointer — Pointer register. An index register is sometimes called a pointer register because its contents are used in the calculation of the address of an operand and, therefore, points to the operand.

polarity — The two opposite logic levels, logic 1 and logic 0, which correspond to two different voltage levels, V_{DD} and V_{SS}.

polling — Periodically reading a status bit to monitor the condition of a peripheral device.

port — A set of wires for communicating with off-chip devices.

prescaler — A circuit that generates an output signal related to the input signal by a fractional scale factor such as 1/2, 1/8, 1/10, etc.

program — A set of computer instructions that causes a computer to perform a desired operation or operations.

program counter (PC) — A 16-bit register in the CPU08. The PC register holds the address of the next instruction or operand that the CPU will use.

pull — An instruction that copies into the accumulator the contents of a stack RAM location. The stack RAM address is in the stack pointer.

pullup — A transistor in the output of a logic gate that connects the output to the logic 1 voltage of the power supply.

pulse-width — The amount of time a signal is on as opposed to being in its off state.

pulse-width modulation (PWM) — Controlled variation (modulation) of the pulse width of a signal with a constant frequency.

push — An instruction that copies the contents of the accumulator to the stack RAM. The stack RAM address is in the stack pointer.

PWM period — The time required for one complete cycle of a PWM waveform.

PMC — Pulse width modulated motor control module

RAM — Random access memory. All RAM locations can be read or written by the CPU. The contents of a RAM memory location remain valid until the CPU writes a different value or until power is turned off.

RC circuit — A circuit consisting of capacitors and resistors having a defined time constant.

read — To copy the contents of a memory location to the accumulator.

register — A circuit that stores a group of bits.

reserved memory location — A memory location that is used only in special factory test modes. Writing to a reserved location has no effect. Reading a reserved location returns an unpredictable value.

reset — To force a device to a known condition.

ROM — Read-only memory. A type of memory that can be read but cannot be changed (written). The contents of ROM must be specified before manufacturing the MCU.

SCI — See serial communication interface module (SCI).

serial — Pertaining to sequential transmission over a single line.

serial communication interface module (SCI) — A module in the M68HC08 Family that supports asynchronous communication.

serial peripheral interface module (SPI) — A module in the M68HC08 Family that supports synchronous communicaton.

set — To change a bit from logic 0 to logic 1; opposite of clear.

shift register — A chain of circuits that can retain the logic levels (logic 1 or logic 0) written to them and that can shift the logic levels to the right or left through adjacent circuits in the chain.

signed — A binary number notation that accommodates both positive and negative numbers. The most significant bit is used to indicate whether the number is positive or negative, normally logic 0 for positive and logic 1 for negative. The other seven bits indicate the magnitude of the number.

SIM — See system integration module (SIM).

software — Instructions and data that control the operation of a microcontroller.

software interrupt (SWI) — An instruction that causes an interrupt and its associated vector fetch.

SPI — See serial peripheral interface module (SPI).

stack — A portion of RAM reserved for storage of CPU register contents and subroutine return addresses.

stack pointer (SP) — A 16-bit register in the CPU08 containing the address of the next available storage location on the stack.

start bit — A bit that signals the beginning of an asynchronous serial transmission.

status bit — A register bit that indicates the condition of a device.

stop bit — A bit that signals the end of an asynchronous serial transmission.

subroutine — A sequence of instructions to be used more than once in the course of a program. The last instruction in a subroutine is a return from subroutine (RTS) instruction. At each place in the main program where the subroutine instructions are needed, a jump or branch to subroutine (JSR or BSR) instruction is used to call the subroutine. The CPU leaves the flow of the main program to execute the instructions in the subroutine. When the RTS instruction is executed, the CPU returns to the main program where it left off.

synchronous — Refers to logic circuits and operations that are synchronized by a common reference signal.

system integration module (SIM) — One of a number of modules that handle a variety of control functions in the modular M68HC08 Family. The SIM controls mode of operation, resets and interrupts, and system clock distribution.

TIM — See timer interface module (TIM).

timer interface module (TIM) — A module used to relate events in a system to a point in time.

timer — A module used to relate events in a system to a point in time.

toggle — To change the state of an output from a logic 0 to a logic 1 or from a logic 1 to a logic 0.

tracking mode — Mode of low-jitter PLL operation during which the PLL is locked on a frequency. Also see acquisition mode.

two’s complement — A means of performing binary subtraction using addition techniques. The most significant bit of a two’s complement number indicates the sign of the number (1 indicates negative). The two’s complement negative of a number is obtained by inverting each bit in the number and then adding 1 to the result.

unbuffered — Utilizes only one register for data; new data overwrites current data.

unimplemented memory location — A memory location that is not used. Writing to an unimplemented location has no effect. Reading an unimplemented location returns an unpredictable value. Executing an opcode at an unimplemented location causes an illegal address reset.

V — The overflow bit in the condition code register of the CPU08. The CPU08 sets the V bit when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow bit.

variable — A value that changes during the course of program execution.

VCO — See voltage-controlled oscillator.

vector — A memory location that contains the address of the beginning of a subroutine written to service an interrupt or reset.

voltage-controlled oscillator (VCO) — A circuit that produces an oscillating output signal of a frequency that is controlled by a dc voltage applied to a control input.

waveform — A graphical representation in which the amplitude of a wave is plotted against time.

wired-OR — Connection of circuit outputs so that if any output is high, the connection point is high.

word — A set of two bytes (16 bits).

write — The transfer of a byte of data from the CPU to a memory location.

X — The lower byte of the index register (H:X) in the CPU08.

Z — The zero bit in the condition code register of the CPU08. The CPU08 sets the zero bit when an arithmetic operation, logical operation, or data manipulation produces a result of $00.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb- free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb- free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale.s Environmental Products program, go to http://www.freescale.com/epp.

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

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

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. The ARM POWERED logo is a registered trademark of ARM Limited. ARM7TDMI-S is a trademark of ARM Limited. Java and all other Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. The Bluetooth trademarks are owned by their proprietor and used by Freescale Semiconductor, Inc. under license.

© Freescale Semiconductor, Inc. 2005. All rights reserved.

Rev. 4.1
MC68HC908MR24/D
August 1, 2005

freescale™
semiconductor