

Welcome to [E-XFL.COM](https://www.e-xfl.com)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Not For New Designs
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	-
Peripherals	LVD, POR, PWM
Number of I/O	13
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 4x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-SOIC (0.295", 7.50mm Width)
Supplier Device Package	16-SOIC
Purchase URL	<a href="https://www.e-xfl.com/product-detail/nxp-semiconductors/mc908qy8cdwer">https://www.e-xfl.com/product-detail/nxp-semiconductors/mc908qy8cdwer</a>

# 1.4 Pin Assignments

The MC68HC908QB8, MC68HC908QB4, and MC68HC908QY8 are available in 16-pin packages. Figure 1-2 shows the pin assignment for these packages.

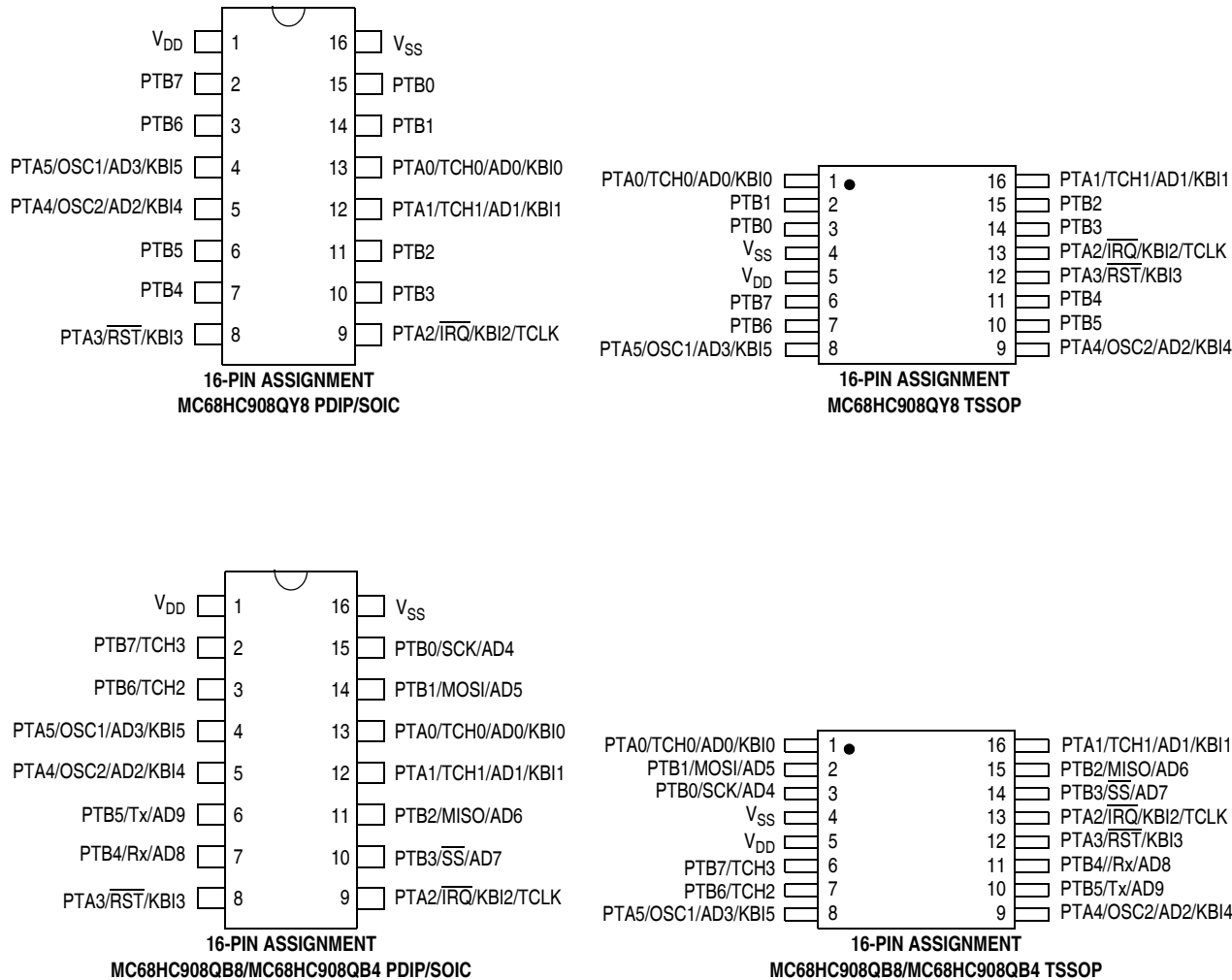


Figure 1-2. MCU Pin Assignments

# 1.5 Pin Functions

Table 1-2 provides a description of the pin functions.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
\$0022	TIM Counter Register Low (TCNTL) <a href="#">See page 185.</a>	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
\$0023	TIM Counter Modulo Register High (TMODH) <a href="#">See page 185.</a>	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
\$0024	TIM Counter Modulo Register Low (TMODL) <a href="#">See page 185.</a>	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
\$0025	TIM Channel 0 Status and Control Register (TSC0) <a href="#">See page 186.</a>	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
\$0026	TIM Channel 0 Register High (TCH0H) <a href="#">See page 189.</a>	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
\$0027	TIM Channel 0 Register Low (TCH0L) <a href="#">See page 189.</a>	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
\$0028	TIM Channel 1 Status and Control Register (TSC1) <a href="#">See page 186.</a>	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
\$0029	TIM Channel 1 Register High (TCH1H) <a href="#">See page 189.</a>	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
\$002A	TIM Channel 1 Register Low (TCH1L) <a href="#">See page 189.</a>	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
\$002B	Reserved									
\$002F										
\$0030	TIM Channel 2 Status and Control Register (TSC2) <a href="#">See page 186.</a>	Read:	CH2F	CH2IE	0	MS2A	ELS2B	ELS2A	TOV2	CH2MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
\$0031	TIM Channel 2 Register High (TCH2H) <a href="#">See page 189.</a>	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
\$0032	TIM Channel 2 Register Low (TCH2L) <a href="#">See page 189.</a>	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
\$0033	TIM Channel 3 Status and Control Register (TSC3) <a href="#">See page 186.</a>	Read:	CH3F	CH3IE	0	MS3A	ELS3B	ELS3A	TOV3	CH3MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
\$0034	TIM Channel 3 Register High (TCH3H) <a href="#">See page 189.</a>	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
\$0035	TIM Channel 3 Register Low (TCH3L) <a href="#">See page 189.</a>	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							

  = Unimplemented     
 R = Reserved     
 U = Unaffected

**Figure 2-2. Control, Status, and Data Registers (Sheet 3 of 5)**

## Memory

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
\$0036	Oscillator Status and Control Register (OSCSR) <a href="#">See page 100.</a>	Read:	OSCOPT1	OSCOPT0	ICFS1	ICFS0	ECFS1	ECFS0	ECGN	ECGST
		Write:								
		Reset:	0	0	0	0	0	0	0	0
\$0037	Reserved									
\$0038	Oscillator Trim Register (OSCTRIM) <a href="#">See page 101.</a>	Read:	TRIM7	TRIM6	TRIM5	TRIM4	TRIM3	TRIM2	TRIM1	TRIM0
		Write:								
		Reset:	1	0	0	0	0	0	0	0
\$0039 ↓ \$003B	Reserved									
\$003C	ADC10 Status and Control Register (ADCSR) <a href="#">See page 46.</a>	Read:	COCO	AIEN	ADCO	ADCH4	ADCH3	ADCH2	ADCH1	ADCH0
		Write:								
		Reset:	0	0	0	1	1	1	1	1
\$003D	ADC10 Data Register High (ADRH) <a href="#">See page 48.</a>	Read:	0	0	0	0	0	0	AD9	AD8
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
\$003E	ADC10 Data Register Low (ADRL) <a href="#">See page 48.</a>	Read:	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
\$003F	ADC10 Clock Register (ADCLK) <a href="#">See page 49.</a>	Read:	ADLPC	ADIV1	ADIV0	ADICLK	MODE1	MODE0	ADLSMP	ACLKEN
		Write:								
		Reset:	0	0	0	0	0	0	0	0
\$FE00	Break Status Register (BSR) <a href="#">See page 195.</a>	Read:	R	R	R	R	R	R	SBSW	R
		Write:							0	
		Reset:							0	
\$FE01	SIM Reset Status Register (SRSR) <a href="#">See page 152.</a>	Read:	POR	PIN	COP	ILOP	ILAD	MODRST	LVI	0
		Write:								
		POR:	1	0	0	0	0	0	0	0
\$FE02	Break Auxiliary Register (BRKAR) <a href="#">See page 195.</a>	Read:	0	0	0	0	0	0	0	BDCOP
		Write:								
		Reset:	0	0	0	0	0	0	0	0
\$FE03	Break Flag Control Register (BFCR) <a href="#">See page 195.</a>	Read:	BCFE	R	R	R	R	R	R	R
		Write:								
		Reset:	0							
\$FE04	Interrupt Status Register 1 (INT1) <a href="#">See page 149.</a>	Read:	IF6	IF5	IF4	IF3	IF2	IF1	0	0
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
\$FE05	Interrupt Status Register 2 (INT2) <a href="#">See page 149.</a>	Read:	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0
\$FE06	Interrupt Status Register 3 (INT3) <a href="#">See page 149.</a>	Read:	IF22	IF21	IF20	IF19	IF18	IF17	IF16	IF15
		Write:	R	R	R	R	R	R	R	R
		Reset:	0	0	0	0	0	0	0	0


  = Unimplemented     
 R = Reserved     
 U = Unaffected

**Figure 2-2. Control, Status, and Data Registers (Sheet 4 of 5)**

### 3.8.2 ADC10 Result High Register (ADRH)

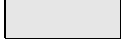
This register holds the MSBs of the result and is updated each time a conversion completes. All other bits read as 0s. Reading ADRH prevents the ADC10 from transferring subsequent conversion results into the result registers until ADRL is read. If ADRL is not read until the after next conversion is completed, then the intermediate conversion result will be lost. In 8-bit mode, this register contains no interlocking with ADRL.

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

 = Unimplemented

**Figure 3-4. ADC10 Data Register High (ADRH), 8-Bit Mode**

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	AD9	AD8
Write:								
Reset:	0	0	0	0	0	0	0	0


 = Unimplemented

**Figure 3-5. ADC10 Data Register High (ADRH), 10-Bit Mode**

### 3.8.3 ADC10 Result Low Register (ADRL)

This register holds the LSBs of the result. This register is updated each time a conversion completes. Reading ADRH prevents the ADC10 from transferring subsequent conversion results into the result registers until ADRL is read. If ADRL is not read until the after next conversion is completed, then the intermediate conversion result will be lost. In 8-bit mode, there is no interlocking with ADRH.

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

 = Unimplemented

**Figure 3-6. ADC10 Data Register Low (ADRL)**

### **ADLSMP — Long Sample Time Configuration**

This bit configures the sample time of the ADC10 to either 3.5 or 23.5 ADCK clock cycles. This adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption in continuous conversion mode if high conversion rates are not required.

1 = Long sample time (23.5 cycles)

0 = Short sample time (3.5 cycles)

### **ACLKEN — Asynchronous Clock Source Enable**

This bit enables the asynchronous clock source as the input clock to generate the internal clock ADCK, and allows operation in stop mode. The asynchronous clock source will operate between 1 MHz and 2 MHz if ADLPC is clear, and between 0.5 MHz and 1 MHz if ADLPC is set.

1 = The asynchronous clock is selected as the input clock source (the clock generator is only enabled during the conversion)

0 = ADICLK specifies the input clock source and conversions will not continue in stop mode

## Configuration Register (CONFIG)

### IRQPUD — $\overline{\text{IRQ}}$ Pin Pullup Control Bit

- 1 = Internal pullup is disconnected
- 0 = Internal pullup is connected between  $\overline{\text{IRQ}}$  pin and  $V_{DD}$

### IRQEN — $\overline{\text{IRQ}}$ Pin Function Selection Bit

- 1 = Interrupt request function active in pin
- 0 = Interrupt request function inactive in pin

### ESCIBDSRC — ESCI Baud Rate Clock Source Bit

ESCIBDSRC controls the clock source used for the ESCI. The setting of the bit affects the frequency at which the ESCI operates.

- 1 = Internal data bus clock used as clock source for ESCI
- 0 = BUSCLKX4 used as clock source for ESCI

### OSCENINSTOP— Oscillator Enable in Stop Mode Bit

OSCENINSTOP, when set, will allow the clock source to continue to generate clocks in stop mode. This function can be used to keep the auto-wakeup running while the rest of the microcontroller stops. When clear, the clock source is disabled when the microcontroller enters stop mode.

- 1 = Oscillator enabled to operate during stop mode
- 0 = Oscillator disabled during stop mode

### RSTEN — $\overline{\text{RST}}$ Pin Function Selection

- 1 = Reset function active in pin
- 0 = Reset function inactive in pin

#### NOTE

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

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVITRIP	SSREC	STOP	COPD
Write:								
Reset:	0	0	0	0	U	0	0	0
POR:	0	0	0	0	0	0	0	0

U = Unaffected

**Figure 5-2. Configuration Register 1 (CONFIG1)**

### COPRS (Out of Stop Mode) — COP Reset Period Selection Bit

- 1 = COP reset short cycle =  $8176 \times \text{BUSCLKX4}$
- 0 = COP reset long cycle =  $262,128 \times \text{BUSCLKX4}$

### COPRS (In Stop Mode) — Auto Wakeup Period Selection Bit, depends on OSCSTOPEN in CONFIG2 and external clock source

- 1 = Auto wakeup short cycle =  $512 \times (\text{INTRCOSC or BUSCLKX4})$
- 0 = Auto wakeup long cycle =  $16,384 \times (\text{INTRCOSC or BUSCLKX4})$

### LVISTOP — LVI Enable in Stop Mode Bit

When the LVIPWRD bit is clear, setting the LVISTOP bit enables the LVI to operate during stop mode. Reset clears LVISTOP.

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

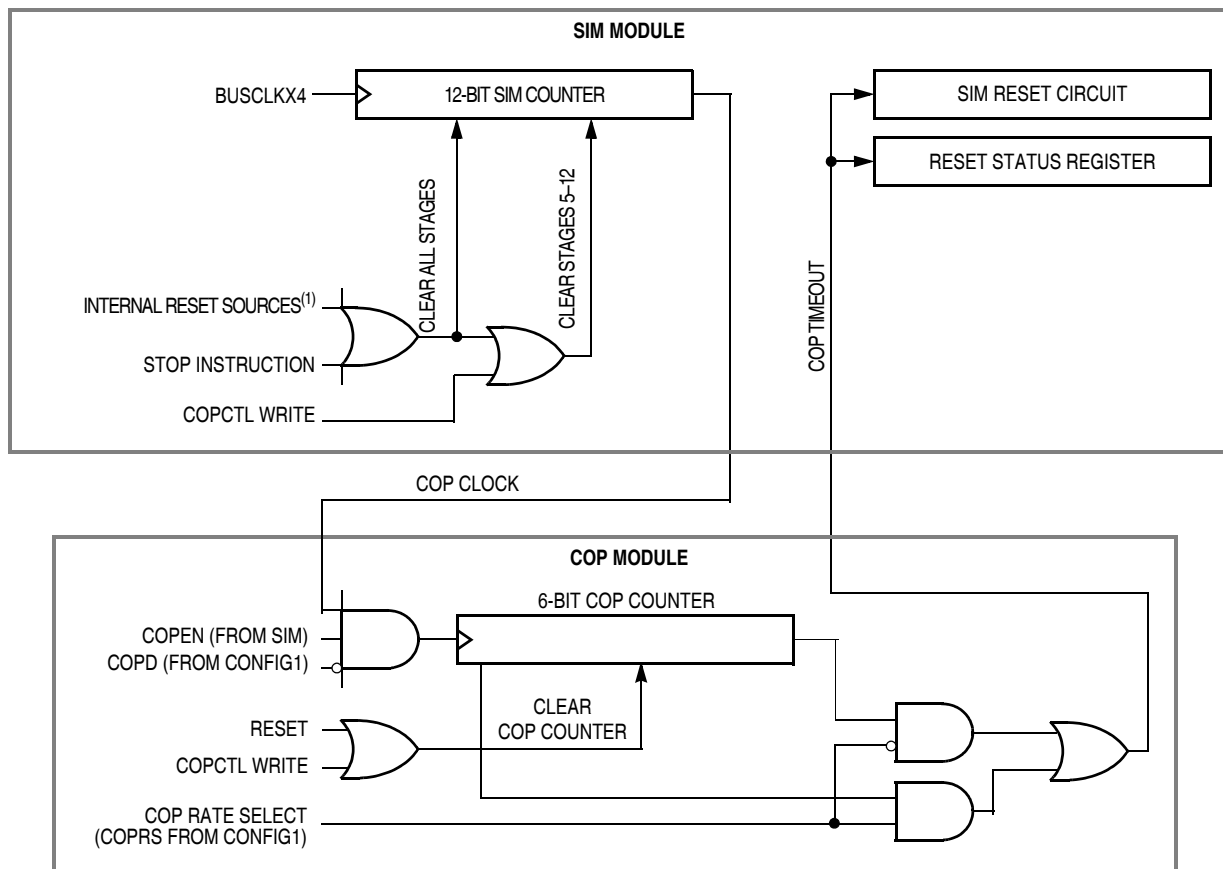
# Chapter 6

## Computer Operating Properly (COP)

### 6.1 Introduction

The computer operating properly (COP) module contains a free-running counter that generates a reset if allowed to overflow. The COP module helps software recover from runaway code. Prevent a COP reset by clearing the COP counter periodically. The COP module can be disabled through the COPD bit in the configuration 1 (CONFIG1) register.

### 6.2 Functional Description



1. See [Chapter 14 System Integration Module \(SIM\)](#) for more details.

**Figure 6-1. COP Block Diagram**



## Computer Operating Properly (COP)

The COP counter is a free-running 6-bit counter preceded by the 12-bit system integration module (SIM) counter. If not cleared by software, the COP counter overflows and generates an asynchronous reset after 8176 or 262,128 BUSCLKX4 cycles; depending on the state of the COP rate select bit, COPRS, in configuration register 1. With a 262,128 BUSCLKX4 cycle overflow option, the internal 12.8-MHz oscillator gives a COP timeout period of 20.48 ms. Writing any value to location \$FFFF before an overflow occurs prevents a COP reset by clearing the COP counter and stages 12–5 of the SIM counter.

### NOTE

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

A COP reset pulls the  $\overline{\text{RST}}$  pin low (if the RSTEN bit is set in the CONFIG1 register) for  $32 \times \text{BUSCLKX4}$  cycles and sets the COP bit in the reset status register (RSR). See [14.8.1 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.*

## 6.3 I/O Signals

The following paragraphs describe the signals shown in [Figure 6-1](#).

### 6.3.1 BUSCLKX4

BUSCLKX4 is the oscillator output signal. BUSCLKX4 frequency is equal to the internal oscillator frequency, the crystal frequency, or the RC-oscillator frequency.

### 6.3.2 STOP Instruction

The STOP instruction clears the SIM counter.

### 6.3.3 COPCTL Write

Writing any value to the COP control register (COPCTL) (see [Figure 6-2](#)) clears the COP counter and clears stages 12–5 of the SIM counter. Reading the COP control register returns the low byte of the reset vector.

### 6.3.4 Power-On Reset

The power-on reset (POR) circuit in the SIM clears the SIM counter  $4096 \times \text{BUSCLKX4}$  cycles after power up.

### 6.3.5 Internal Reset

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

### 6.3.6 COPD (COP Disable)

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

## Low-Voltage Inhibit (LVI)

The LVI module contains a bandgap reference circuit and comparator. When the LVITRIP bit is cleared, the default state at power-on reset,  $V_{TRIPF}$  is configured for the lower  $V_{DD}$  operating range. The actual trip points are specified in [18.5 5-V DC Electrical Characteristics](#) and [18.8 3-V DC Electrical Characteristics](#).

Because the default LVI trip point after power-on reset is configured for low voltage operation, a system requiring high voltage LVI operation must set the LVITRIP bit during system initialization.  $V_{DD}$  must be above the LVI trip rising voltage,  $V_{TRIPR}$ , for the high voltage operating range or the MCU will immediately go into LVI reset.

After an LVI reset occurs, the MCU remains in reset until  $V_{DD}$  rises above  $V_{TRIPR}$ . See [Chapter 14 System Integration Module \(SIM\)](#) for the reset recovery sequence.

The output of the comparator controls the state of the LVIOOUT flag in the LVI status register (LVISR) and can be used for polling LVI operation when the LVI reset is disabled.

The LVI is enabled out of reset. The following bits located in the configuration register can alter the default conditions.

- Setting the LVI power disable bit, LVIPWRD, disables the LVI.
- Setting the LVI reset disable bit, LVIRSTD, prevents the LVI module from generating a reset.
- Setting the LVI enable in stop mode bit, LVISTOP, enables the LVI to operate in stop mode.
- Setting the LVI trip point bit, LVITRIP, configures the trip point voltage ( $V_{TRIPF}$ ) for the higher  $V_{DD}$  operating range.

### 10.3.1 Polled LVI Operation

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

### 10.3.2 Forced Reset Operation

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

### 10.3.3 LVI Hysteresis

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

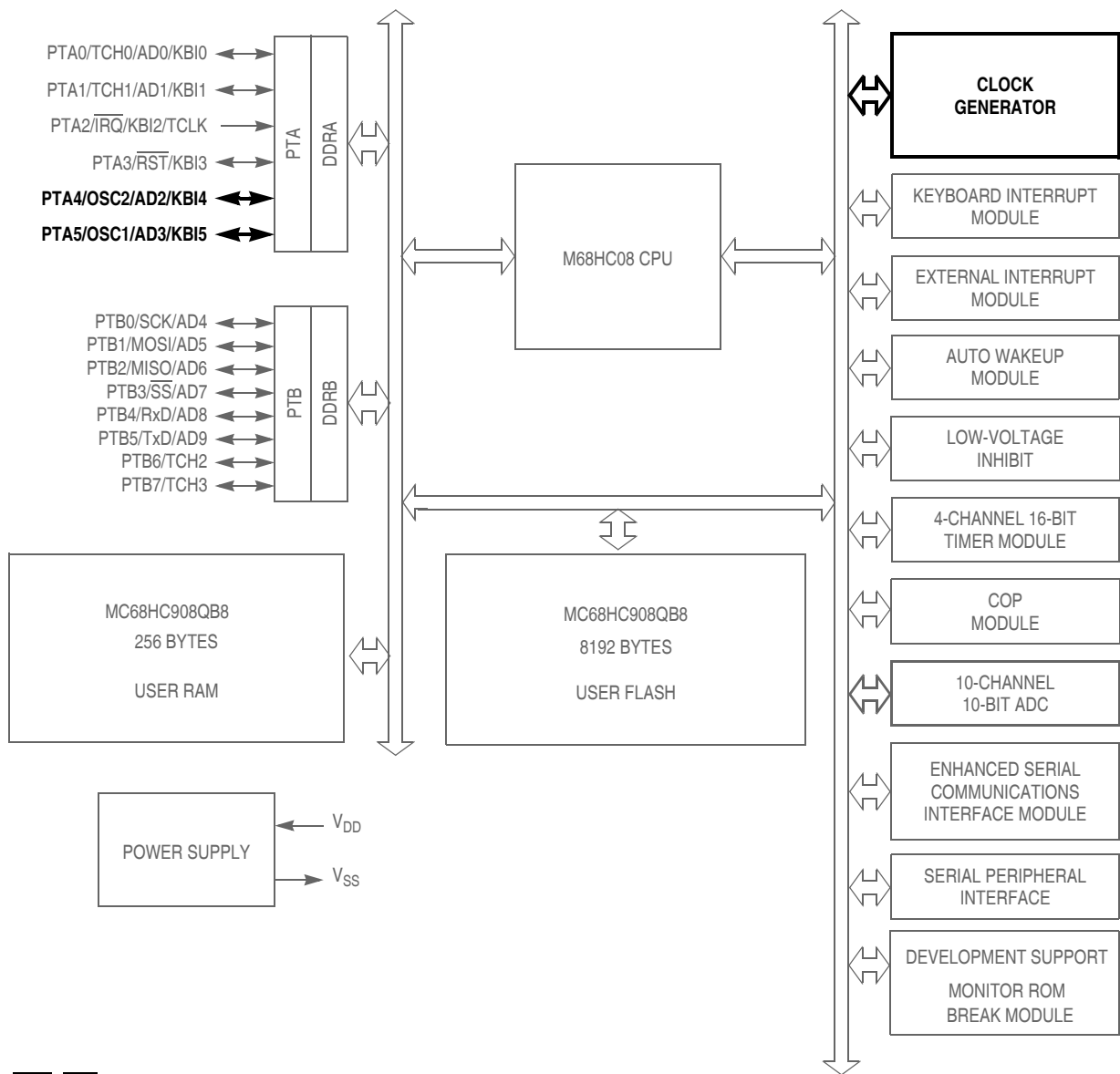
### 10.3.4 LVI Trip Selection

LVITRIP in the configuration register selects the LVI protection range. The default setting out of reset is for the low voltage range. Because LVITRIP is in a write-once configuration register, the protection range cannot be changed after initialization.

#### NOTE

*The MCU is guaranteed to operate at a minimum supply voltage. The trip point ( $V_{TRIPF}$ ) may be lower than this. See the Electrical Characteristics section for the actual trip point voltages.*

# Oscillator Module (OSC)



$\overline{RST}$ ,  $\overline{IRQ}$ : Pins have internal pull up device  
All port pins have programmable pull up device  
PTA[0:5]: Higher current sink and source capability

**Figure 11-1. Block Diagram Highlighting OSC Block and Pins**

## Chapter 13

# Enhanced Serial Communications Interface (ESCI) Module

### 13.1 Introduction

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

The ESCI module shares its pins with general-purpose input/output (I/O) port pins. See [Figure 13-1](#) for port location of these shared pins. The ESCI baud rate clock source is controlled by a bit (ESCIBDSRC) located in the configuration register.

### 13.2 Features

Features include:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- Programmable baud rates
- Programmable 8-bit or 9-bit character length
- Separately enabled transmitter and receiver
- Separate receiver and transmitter interrupt requests
- Programmable transmitter output polarity
- Receiver wakeup methods
  - Idle line
  - Address mark
- 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

**NOTE**

*Writing to the RE bit is not allowed when the enable ESCI bit (ENSCI) is clear. ENSCI is in ESCI control register 1.*

**RWU — Receiver Wakeup Bit**

This read/write bit puts the receiver in a standby state during which receiver interrupts are disabled. The WAKE bit in SCC1 determines whether an idle input or an address mark brings the receiver out of the standby state and clears the RWU bit.

- 1 = Standby state
- 0 = Normal operation

**SBK — Send Break Bit**

Setting and then clearing this read/write bit transmits a break character followed by a 1. The 1 after the break character guarantees recognition of a valid start bit. If SBK remains set, the transmitter continuously transmits break characters with no 1s between them.

- 1 = Transmit break characters
- 0 = No break characters being transmitted

**NOTE**

*Do not toggle the SBK bit immediately after setting the SCTE bit. Toggling SBK before the preamble begins causes the ESCI to send a break character instead of a preamble.*

### 13.8.3 ESCI Control Register 3

ESCI control register 3 (SCC3):

- Stores the ninth ESCI data bit received and the ninth ESCI data bit to be transmitted.
- Enables these interrupts:
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error

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

  = Unimplemented     
 R = Reserved     
 U = Unaffected

**Figure 13-11. ESCI Control Register 3 (SCC3)**

**R8 — Received Bit 8**

When the ESCI is receiving 9-bit characters, R8 is the read-only ninth bit (bit 8) of the received character. R8 is received at the same time that the SCDR receives the other 8 bits.

When the ESCI is receiving 8-bit characters, R8 is a copy of the eighth bit (bit 7).

**T8 — Transmitted Bit 8**

When the ESCI is transmitting 9-bit characters, T8 is the read/write ninth bit (bit 8) of the transmitted character. T8 is loaded into the transmit shift register at the same time that the SCDR is loaded into the transmit shift register.

Table 13-10. ESCI Baud Rate Selection Examples

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

## 13.9.4 Arbitration Mode

If AM[1:0] is set to 10, the arbiter module operates in arbitration mode. On every rising edge of SCI\_TxD (output of the transmit shift register, see [Figure 13-2](#)), the counter is started. When the counter reaches \$38 (ACLK = 0) or \$08 (ACLK = 1), RxD is statically sensed. If in this case, RxD is sensed low (for example, another bus is driving the bus dominant) ALOST is set. As long as ALOST is set, the TxD pin is forced to 1, resulting in a seized transmission.

If SCI\_TxD senses 0 without having sensed a 0 before on RxD, the counter will be reset, arbitration operation will be restarted after the next rising edge of SCI\_TxD.

# Chapter 14

## System Integration Module (SIM)

### 14.1 Introduction

This section describes the system integration module (SIM), which supports up to 24 external and/or internal interrupts. Together with the central processor unit (CPU), the SIM controls all microcontroller unit (MCU) activities. A block diagram of the SIM is shown in [Figure 14-1](#). The SIM is a system state controller that coordinates CPU and exception timing.

The SIM is responsible for:

- Bus clock generation and control for CPU and peripherals
  - Stop/wait/reset/break entry and recovery
  - Internal clock control
- Master reset control, including power-on reset (POR) and computer operating properly (COP) timeout
- Interrupt control:
  - Acknowledge timing
  - Arbitration control timing
  - Vector address generation
- CPU enable/disable timing

**Table 14-1. Signal Name Conventions**

Signal Name	Description
BUSCLKX4	Buffered clock from the internal, RC or XTAL oscillator circuit.
BUSCLKX2	The BUSCLKX4 frequency divided by two. This signal is again divided by two in the SIM to generate the internal bus clocks (bus clock = BUSCLKX4 ÷ 4).
Address bus	Internal address bus
Data bus	Internal data bus
PORRST	Signal from the power-on reset module to the SIM
IRST	Internal reset signal
R/ $\overline{W}$	Read/write signal

### 14.2 $\overline{RST}$ and $\overline{IRQ}$ Pins Initialization

$\overline{RST}$  and  $\overline{IRQ}$  pins come out of reset as PTA3 and PTA2 respectively.  $\overline{RST}$  and  $\overline{IRQ}$  functions can be activated by programing CONFIG2 accordingly. Refer to [Chapter 5 Configuration Register \(CONFIG\)](#).



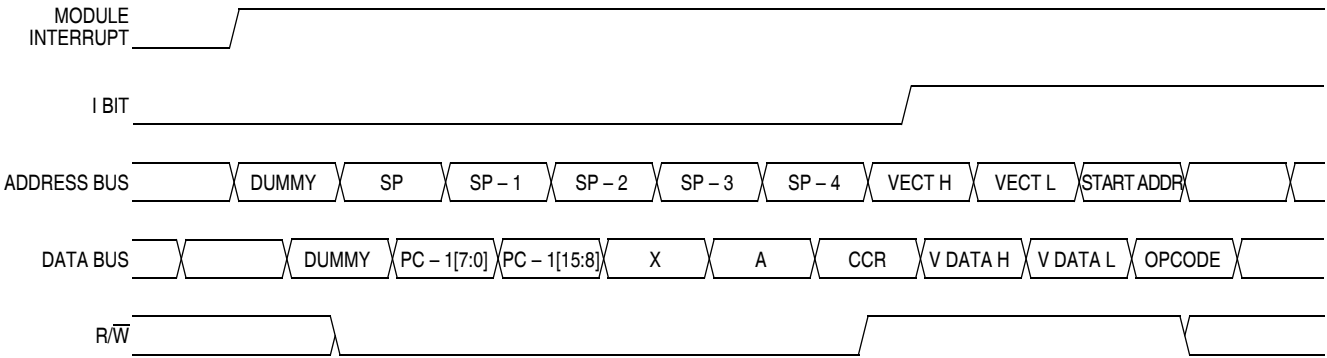


Figure 14-8. Interrupt Entry

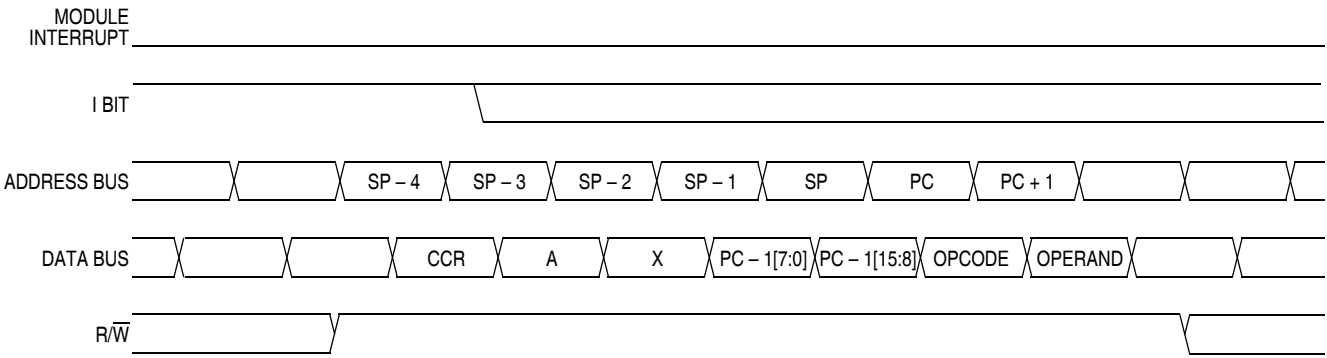


Figure 14-9. Interrupt Recovery

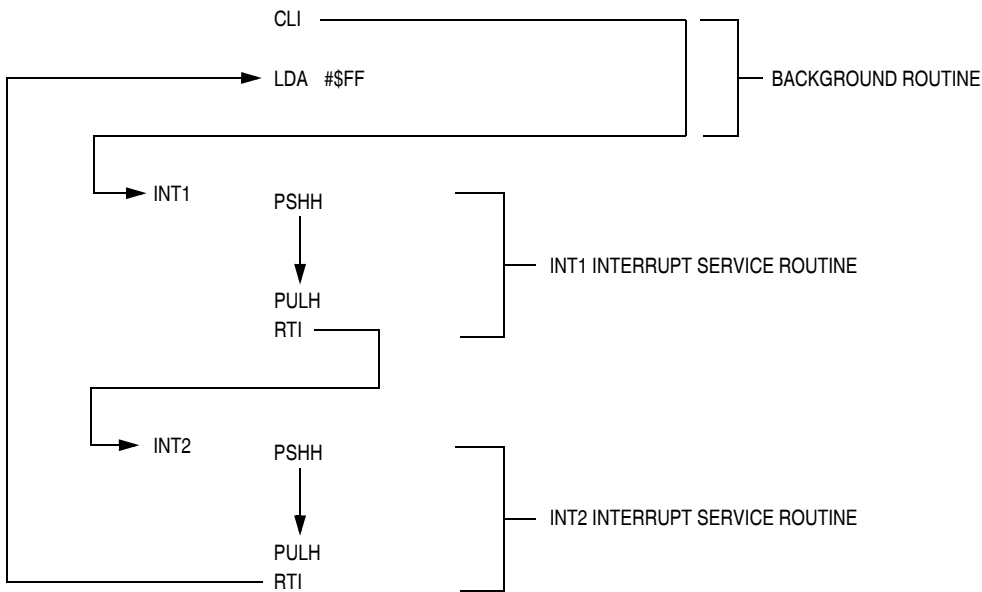
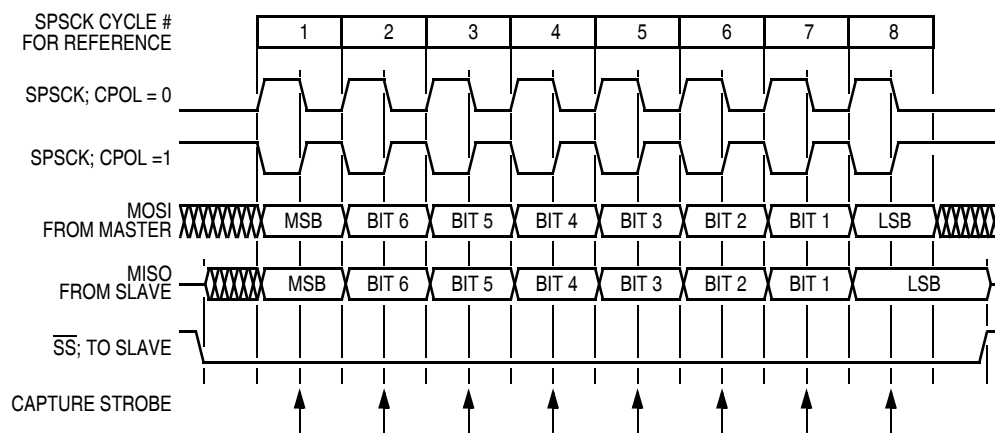


Figure 14-10. Interrupt Recognition Example

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 15-6. Transmission Format (CPHA = 1)**

When CPHA = 1 for a slave, the first edge of the SPSCCK 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. After 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 SPSCCK. Any data written after the first edge is stored in the transmit data register and transferred to the shift register after the current transmission.

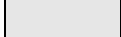
#### 15.3.3.4 Transmission Initiation Latency

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 SPSCCK signal. When CPHA = 0, the SPSCCK signal remains inactive for the first half of the first SPSCCK cycle. When CPHA = 1, the first SPSCCK cycle begins with an edge on the SPSCCK line from its inactive to its active level. The SPI clock rate (selected by SPR1:SPR0) affects the delay from the write to SPDR and the start of the SPI transmission. (See [Figure 15-7](#).) 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. Because the SPI clock is free-running, it is uncertain where the write to the SPDR occurs relative to the slower SPSCCK. This uncertainty causes the variation in the initiation delay shown in [Figure 15-7](#). 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.

### 17.2.2.3 Break Auxiliary Register

The break auxiliary register (BRKAR) contains a bit that enables software to disable the COP while the MCU is in a state of break interrupt with monitor mode.

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

 = Unimplemented

**Figure 17-6. Break Auxiliary Register (BRKAR)**

#### BDCOP — Break Disable COP Bit


This read/write bit disables the COP during a break interrupt. Reset clears the BDCOP bit.

- 1 = COP disabled during break interrupt
- 0 = COP enabled during break interrupt

### 17.2.2.4 Break Status Register

The break status register (BSR) contains a flag to indicate that a break caused an exit from wait mode. This register is only used in emulation mode.

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R	R	R	R	R	R	SBSW	R
Write:							Note <sup>(1)</sup>	
Reset:							0	

 = Reserved

1. Writing a 0 clears SBSW.

**Figure 17-7. Break Status Register (BSR)**

#### SBSW — SIM Break Stop/Wait


SBSW can be read within the break state SWI routine. The user can modify the return address on the stack by subtracting one from it.

- 1 = Wait mode was exited by break interrupt
- 0 = Wait mode was not exited by break interrupt

### 17.2.2.5 Break Flag Control Register

The break control register (BFCR) contains a bit that enables software to clear status bits while the MCU is in a break state.

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

 = Reserved

**Figure 17-8. Break Flag Control Register (BFCR)**

Table 17-7. READSP (Read Stack Pointer) Command

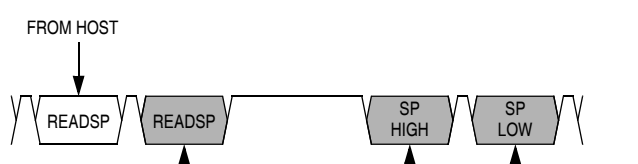
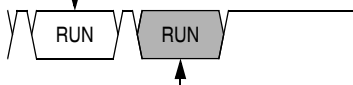
Description	Reads stack pointer
Operand	None
Data Returned	Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order
Opcode	\$0C
<div>Command Sequence</div> <div><p>The diagram illustrates the timing of the READSP command sequence. It shows a sequence of signals: FROM HOST, READSP, ECHO, READSP, SP HIGH, SP LOW, and RETURN. The READSP signal is a pulse that occurs after FROM HOST and before ECHO. The ECHO signal is a pulse that occurs after the first READSP. The SP HIGH and SP LOW signals are pulses that occur after the second READSP. The RETURN signal is a pulse that occurs after SP LOW.</p></div>	

Table 17-8. RUN (Run User Program) Command

Description	Executes PULH and RTI instructions
Operand	None
Data Returned	None
Opcode	\$28
<div>Command Sequence</div> <div><div>FROM HOST</div><div></div><div>ECHO</div></div>	

The MCU executes the SWI and PSHH instructions when it enters monitor mode. The RUN command tells the MCU to execute the PULH and RTI instructions. Before sending the RUN command, the host can modify the stacked CPU registers to prepare to run the host program. The READSP command returns the incremented stack pointer value, SP + 1. The high and low bytes of the program counter are at addresses SP + 5 and SP + 6.

	SP
HIGH BYTE OF INDEX REGISTER	SP + 1
CONDITION CODE REGISTER	SP + 2
ACCUMULATOR	SP + 3
LOW BYTE OF INDEX REGISTER	SP + 4
HIGH BYTE OF PROGRAM COUNTER	SP + 5
LOW BYTE OF PROGRAM COUNTER	SP + 6
	SP + 7

Figure 17-17. Stack Pointer at Monitor Mode Entry

