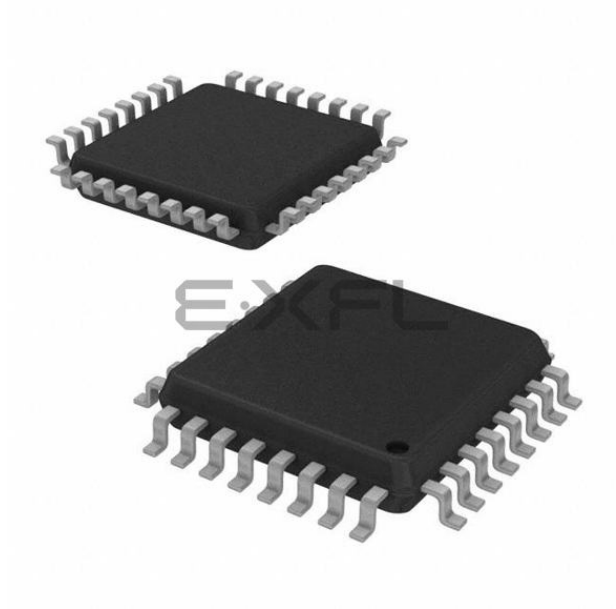


Welcome to E-XFL.COM

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	Obsolete
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	LINbus, SCI, SPI
Peripherals	LVD, POR, PWM
Number of I/O	21
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc908gr16acfje

Table 2-1. Vector Addresses

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

AIEN — ADC Interrupt Enable Bit

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

1 = ADC interrupt enabled
0 = ADC interrupt disabled

ADCO — ADC Continuous Conversion Bit

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

1 = Continuous ADC conversion
0 = One ADC conversion

ADCH4–ADCH0 — ADC Channel Select Bits

ADCH4–ADCH0 form a 5-bit field which is used to select one of 32 ADC channels. Only eight channels, AD7–AD0, are available on this MCU. The channels are detailed in Table 3-1. Care should be taken when using a port pin as both an analog and digital input simultaneously to prevent switching noise from corrupting the analog signal. See Table 3-1.

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

NOTE

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

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

Table 3-1. Mux Channel Select(1)

ADCH4	ADCH3	ADCH2	ADCH1	ADCH0	Input Select
0	0	0	0	0	PTB0/AD0
0	0	0	0	1	PTB1/AD1
0	0	0	1	0	PTB2/AD2
0	0	0	1	1	PTB3/AD3
0	0	1	0	0	PTB4/AD4
0	0	1	0	1	PTB5/AD5
0	0	1	1	0	PTB6/AD6
0	0	1	1	1	PTB7/AD7
0 ↓ 1	1 ↓ 1	0 ↓ 1	0 ↓ 0	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

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

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

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

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

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

4.8.3 Choosing a Filter

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

Figure 4-9 shows two types of filter circuits. In low-cost applications, where stability and reaction time of the PLL are not critical, the three component filter network shown in Figure 4-9 (B) can be replaced by a single capacitor, C_F, as shown in shown in Figure 4-9 (A). Refer to Table 4-5 for recommended filter components at various reference frequencies. For reference frequencies between the values listed in the table, extrapolate to the nearest common capacitor value. In general, a slightly larger capacitor provides more stability at the expense of increased lock time.

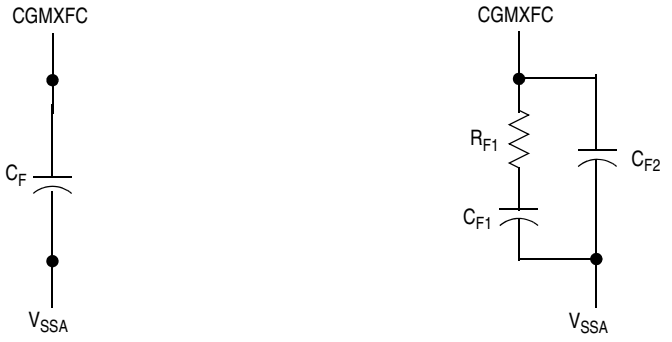


Figure 4-9. PLL Filter

6.3.6 Reset Vector Fetch

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

6.3.7 COPD (COP Disable)

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

6.3.8 COPRS (COP Rate Select)

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

6.4 COP Control Register

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

Address: $FFFF

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

Figure 6-2. COP Control Register (COPCTL)

6.5 Interrupts

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

6.6 Monitor Mode

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

6.7 Low-Power Modes

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

6.7.1 Wait Mode

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

Z — Zero Flag

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

1 = Zero result
0 = Non-zero result

C — Carry/Borrow Flag

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

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

7.4 Arithmetic/Logic Unit (ALU)

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

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

7.5 Low-Power Modes

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

7.5.1 Wait Mode

The WAIT instruction:

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

7.5.2 Stop Mode

The STOP instruction:

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

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

7.6 CPU During Break Interrupts

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

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

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

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

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

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

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

7.8 Opcode Map

See Table 7-2.

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

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

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

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

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

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

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

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

NOTE

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

8.5 IRQ Module During Break Interrupts

The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear the latch during the break state. See Chapter 19 Development Support.

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

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

Chapter 9
Keyboard Interrupt Module (KBI)

9.1 Introduction

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

9.2 Features

Features include:

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

9.3 Functional Description

Writing to the KBIE7–KBIE0 bits in the keyboard interrupt enable register independently enables or disables each port A pin as a keyboard interrupt pin. Enabling a keyboard interrupt pin also enables its internal pullup device. A low level applied to an enabled keyboard interrupt pin latches a keyboard interrupt request.

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

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

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

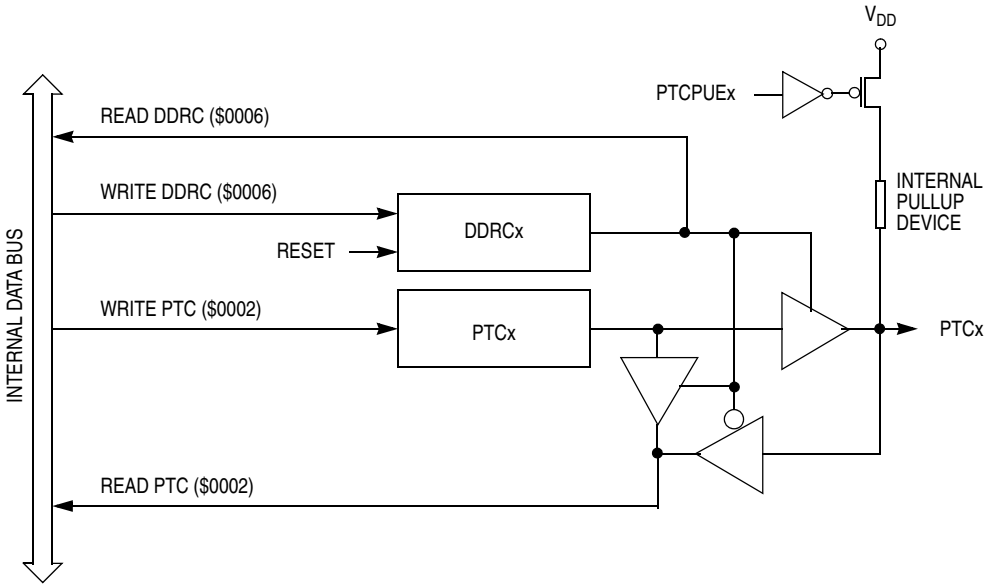


Figure 12-11. Port C I/O Circuit

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

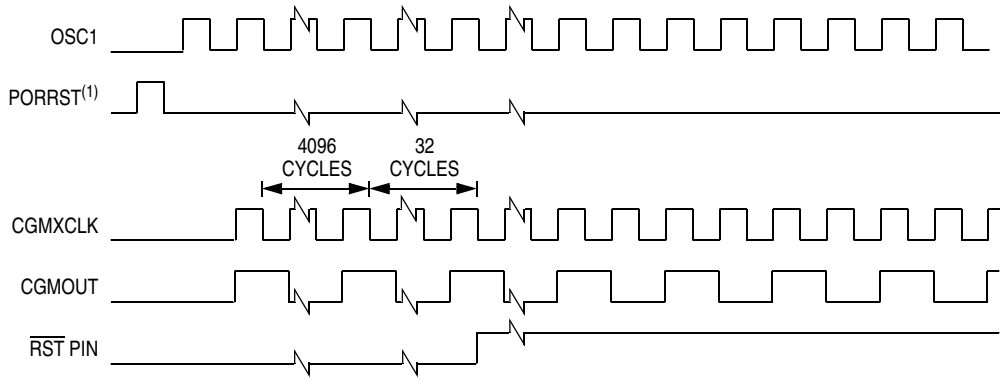
Table 12-4. Port C Pin Functions

PTCPUE Bit	DDRC Bit	PTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PTC	
				Read/Write	Read	Write
1	0	X(1)	Input, V_{DD}(2)	DDRC6–DDRC0	Pin	PTC6–PTC0(3)
0	0	X	Input, Hi-Z(4)	DDRC6–DDRC0	Pin	PTC6–PTC0(3)
X	1	X	Output	DDRC6–DDRC0	PTC6–PTC0	PTC6–PTC0

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

A power-on reset:

- Holds the clocks to the central processor unit (CPU) and modules inactive for an oscillator stabilization delay of 4096 CGMXCLK cycles
- Drives the $\overline{\text{RST}}$ pin low during the oscillator stabilization delay
- Releases the RST pin 32 CGMXCLK cycles after the oscillator stabilization delay
- Releases the CPU to begin the reset vector sequence 64 CGMXCLK cycles after the oscillator stabilization delay
- Sets the POR and LVI bits in the SIM reset status register and clears all other bits in the register



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

Figure 13-1. Power-On Reset Recovery

13.2.3.2 Computer Operating Properly (COP) Reset

A computer operating properly (COP) reset is an internal reset caused by an overflow of the COP counter. A COP reset sets the COP bit in the SIM reset status register.

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

13.2.3.3 Low-Voltage Inhibit (LVI) Reset

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

An LVI reset:

- Holds the clocks to the CPU and modules inactive for an oscillator stabilization delay of 4096 CGMXCLK cycles after the power supply voltage rises to the LVI_{TRIPR} voltage
- Drives the $\overline{\text{RST}}$ pin low for as long as V_{DD} is below the V_{TRIPR} voltage and during the oscillator stabilization delay
- Releases the $\overline{\text{RST}}$ pin 32 CGMXCLK cycles after the oscillator stabilization delay
- Releases the CPU to begin the reset vector sequence 64 CGMXCLK cycles after the oscillator stabilization delay
- Sets the LVI bit in the SIM reset status register

13.2.3.4 Illegal Opcode Reset

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

14.6 ESCI During Break Module Interrupts

The BCFE bit in the break flag control register (SBFCR) enables software to clear status bits during the break state. See Chapter 19 Development Support.

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

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

14.7 I/O Signals

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

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

14.7.1 PTE0/TxD (Transmit Data)

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

14.7.2 PTE1/RxD (Receive Data)

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

14.8 I/O Registers

These I/O registers control and monitor ESCI operation:

- ESCI control register 1, SCC1
- ESCI control register 2, SCC2
- ESCI control register 3, SCC3
- ESCI status register 1, SCS1
- ESCI status register 2, SCS2
- ESCI data register, SCDR
- ESCI baud rate register, SCBR
- ESCI prescaler register, SCPSC
- ESCI arbiter control register, SCIACTL
- ESCI arbiter data register, SCIADAT

PDS2–PDS0 — Prescaler Divisor Select Bits

These read/write bits select the prescaler divisor as shown in Table 14-10. Reset clears PDS2–PDS0.

NOTE

The setting of '000' will bypass not only this prescaler but also the prescaler divisor fine adjust (PDFA). It is not recommended to bypass the prescaler while ENSCI is set, because the switching is not glitch free.

Table 14-10. ESCI Prescaler Division Ratio

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

PSSB4–PSSB0 — Clock Insertion Select Bits

These read/write bits select the number of clocks inserted in each 32 output cycle frame to achieve more timing resolution on the **average** prescaler frequency as shown in Table 14-11. Reset clears PSSB4–PSSB0.

Table 14-11. ESCI Prescaler Divisor Fine Adjust

PSSB[4:3:2:1:0]	Prescaler Divisor Fine Adjust (PDFA)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875

Continued on next page

low while the SIM counter counts out 4096 + 32 CGMXCLK cycles. Thirty-two CGMXCLK cycles later, the CPU is released from reset to allow the reset vector sequence to occur. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

15.3.2.6 Monitor Mode Entry Module Reset (MODRST)

The monitor mode entry module reset (MODRST) asserts its output to the SIM when monitor mode is entered in the condition where the reset vectors are erased ($FF) (see 19.3.1.1 Normal Monitor Mode). When MODRST gets asserted, an internal reset occurs. The SIM actively pulls down the $\overline{\text{RST}}$ pin for all internal reset sources.

15.4 SIM Counter

The SIM counter is used by the power-on reset module (POR) and in stop mode recovery to allow the oscillator time to stabilize before enabling the internal bus (IBUS) clocks. The SIM counter also serves as a prescaler for the computer operating properly module (COP). The SIM counter is 12 bits long.

15.4.1 SIM Counter During Power-On Reset

The power-on reset module (POR) detects power applied to the MCU. At power-on, the POR circuit asserts the signal PORRST. Once the SIM is initialized, it enables the clock generation module (CGM) to drive the bus clock state machine.

15.4.2 SIM Counter During Stop Mode Recovery

The SIM counter also is used for stop mode recovery. The STOP instruction clears the SIM counter. After an interrupt, break, or reset, the SIM senses the state of the short stop recovery bit, SSREC, in the CONFIG1 register. If the SSREC bit is a 1, then the stop recovery is reduced from the normal delay of 4096 CGMXCLK cycles down to 32 CGMXCLK cycles. This is ideal for applications using crystals with the OSCENINSTOP bit set. External crystal applications should use the full stop recovery time, SSREC cleared, if the OSCENINSTOP bit is cleared.

15.4.3 SIM Counter and Reset States

External reset has no effect on the SIM counter. See 15.6.2 Stop Mode for details. The SIM counter is free-running after all reset states. See 15.3.2 Active Resets from Internal Sources for counter control and internal reset recovery sequences.

15.5 Exception Control

Normal, sequential program execution can be changed in three different ways:

- Interrupts:
  - Maskable hardware CPU interrupts
  - Non-maskable software interrupt instruction (SWI)
- Reset
- Break interrupts

In a master SPI with the mode fault enable bit (MODFEN) set, the mode fault flag (MODF) is set if $\overline{SS}$ goes low. A mode fault in a master SPI causes the following events to occur:

- If ERRIE = 1, the SPI generates an SPI receiver/error CPU interrupt request.
- The SPE bit is cleared.
- The SPTE bit is set.
- The SPI state counter is cleared.
- The data direction register of the shared I/O port regains control of port drivers.

NOTE

To prevent bus contention with another master SPI after a mode fault error, clear all SPI bits of the data direction register of the shared I/O port before enabling the SPI.

When configured as a slave (SPMSTR = 0), the MODF flag is set if $\overline{SS}$ goes high during a transmission. When CPHA = 0, a transmission begins when $\overline{SS}$ goes low and ends once the incoming SPSCK goes back to its idle level following the shift of the eighth data bit. When CPHA = 1, the transmission begins when the SPSCK leaves its idle level and $\overline{SS}$ is already low. The transmission continues until the SPSCK returns to its idle level following the shift of the last data bit. See 16.4 Transmission Formats.

NOTE

Setting the MODF flag does not clear the SPMSTR bit. SPMSTR has no function when SPE = 0. Reading SPMSTR when MODF = 1 shows the difference between a MODF occurring when the SPI is a master and when it is a slave.

NOTE

When CPHA = 0, a MODF occurs if a slave is selected ($\overline{SS}$ is low) and later unselected ($\overline{SS}$ is high) even if no SPSCK is sent to that slave. This happens because $\overline{SS}$ low indicates the start of the transmission (MISO driven out with the value of MSB) for CPHA = 0. When CPHA = 1, a slave can be selected and then later unselected with no transmission occurring. Therefore, MODF does not occur since a transmission was never begun.

In a slave SPI (MSTR = 0), MODF generates an SPI receiver/error CPU interrupt request if the ERRIE bit is set. The MODF bit does not clear the SPE bit or reset the SPI in any way. Software can abort the SPI transmission by clearing the SPE bit of the slave.

NOTE

A high on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. Also, the slave SPI ignores all incoming SPSCK clocks, even if it was already in the middle of a transmission.

To clear the MODF flag, read the SPSCR with the MODF bit set and then write to the SPCR register. This entire clearing mechanism must occur with no MODF condition existing or else the flag is not cleared.

16.7 Interrupts

Four SPI status flags can be enabled to generate CPU interrupt requests. See Table 16-1.

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.

16.11.4 $\overline{SS}$ (Slave Select)

The $\overline{SS}$ pin has various functions depending on the current state of the SPI. For an SPI configured as a slave, $\overline{SS}$ is used to select a slave. For CPHA = 0, $\overline{SS}$ is used to define the start of a transmission. (See 16.4 Transmission Formats.) Since it is used to indicate the start of a transmission, $\overline{SS}$ must be toggled high and low between each byte transmitted for the CPHA = 0 format. However, it can remain low between transmissions for the CPHA = 1 format. See Figure 16-13.

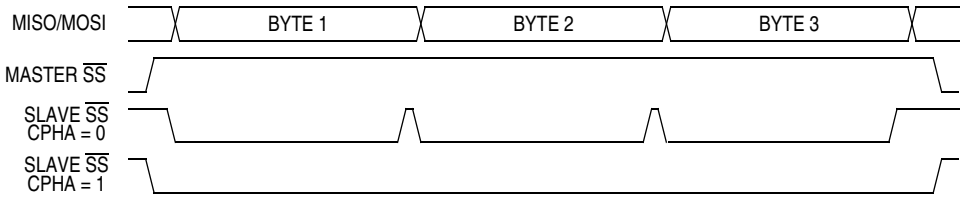


Figure 16-13. CPHA/$\overline{SS}$ Timing

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 $\overline{SS}$ from creating a MODF error. See 16.12.2 SPI Status and Control Register.

NOTE

A high 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 16.6.2 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 MODFEN is 0 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. When MODFEN is 1, $\overline{SS}$ is an input-only pin to the SPI regardless of the state of the data direction register of the shared I/O port.

The CPU can always read the state of the $\overline{SS}$ pin by configuring the appropriate pin as an input and reading the port data register. See Table 16-2.

Table 16-2. SPI Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	Function of $\overline{SS}$ Pin
0	X(1))	X	Not enabled	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	0	X	Slave	Input-only to SPI
1	1	0	Master without MODF	General-purpose I/O; $\overline{SS}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

1. X = Don't care

Table 17-1. Timebase Divider Selection

TBR2	TBR1	TBR0	Divider	
			TBMCLKSEL	
			0	1
0	0	0	32,768	4,194,304
0	0	1	8192	1,048,576
0	1	0	2048	262144
0	1	1	128	16,384
1	0	0	64	8192
1	0	1	32	4096
1	1	0	16	2048
1	1	1	8	1024

As an example, the divider is 16,384 with a 4.9152 MHz crystal, the TBMCLKSEL set for divide-by-128, and TBR2–TBR0 set to {011}. The interrupt period is:

$16{,}384/4.9152 \times 10^6 = 3.33$ ms

NOTE

Do not change TBR2–TBR0 bits while the timebase is enabled (TBON = 1).

17.6 Low-Power Modes

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

17.6.1 Wait Mode

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

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

17.6.2 Stop Mode

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

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

If the timebase functions are not required during stop mode, reduce power consumption by disabling the timebase module before executing the STOP instruction.

into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

18.4.3 Output Compare

With the output compare function, the TIM can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

18.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in 18.4.3 Output Compare. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIM channel registers.

An unsynchronized write to the TIM channel registers to change an output compare value could cause incorrect operation for up to two counter overflow periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that counter overflow period. Also, using a TIM overflow interrupt routine to write a new, smaller output compare value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the output compare value on channel x:

- When changing to a smaller value, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current output compare pulse. The interrupt routine has until the end of the counter overflow period to write the new value.
- When changing to a larger output compare value, enable TIM overflow interrupts and write the new value in the TIM overflow interrupt routine. The TIM overflow interrupt occurs at the end of the current counter overflow period. Writing a larger value in an output compare interrupt routine (at the end of the current pulse) could cause two output compares to occur in the same counter overflow period.

18.4.3.2 Buffered Output Compare

Channels 0 and 1 can be linked to form a buffered output compare channel whose output appears on the TCH0 pin. The TIM channel registers of the linked pair alternately control the output.

Setting the MS0B bit in TIM channel 0 status and control register (TSC0) links channel 0 and channel 1. The output compare value in the TIM channel 0 registers initially controls the output on the TCH0 pin. Writing to the TIM channel 1 registers enables the TIM channel 1 registers to synchronously control the output after the TIM overflows. At each subsequent overflow, the TIM channel registers (0 or 1) that control the output are the ones written to last. TSC0 controls and monitors the buffered output compare function, and TIM channel 1 status and control register (TSC1) is unused. While the MS0B bit is set, the channel 1 pin, TCH1, is available as a general-purpose I/O pin.

NOTE

In buffered output compare operation, do not write new output compare values to the currently active channel registers. User software should track the currently active channel to prevent writing a new value to the active channel. Writing to the active channel registers is the same as generating unbuffered output compares.

18.4.4 Pulse Width Modulation (PWM)

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

As Figure 18-4 shows, the output compare value in the TIM channel registers determines the pulse width of the PWM signal. The time between overflow and output compare is the pulse width. Program the TIM to clear the channel pin on output compare if the polarity of the PWM pulse is 1 (ELSxA = 0). Program the TIM to set the pin if the polarity of the PWM pulse is 0 (ELSxA = 1).

The value in the TIM counter modulo registers and the selected prescaler output determines the frequency of the PWM output. The frequency of an 8-bit PWM signal is variable in 256 increments. Writing $00FF (255) to the TIM counter modulo registers produces a PWM period of 256 times the internal bus clock period if the prescaler select value is $000. See 18.9.1 TIM Status and Control Register.

The value in the TIM channel registers determines the pulse width of the PWM output. The pulse width of an 8-bit PWM signal is variable in 256 increments. Writing $0080 (128) to the TIM channel registers produces a duty cycle of 128/256 or 50%.

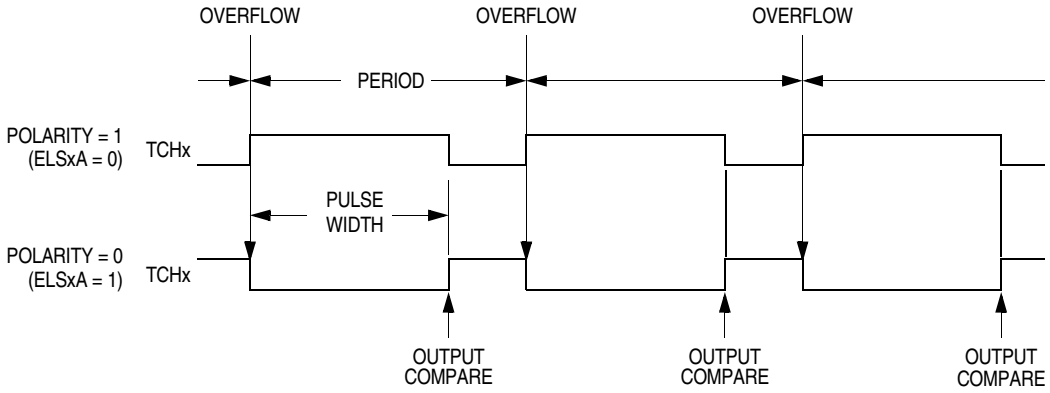


Figure 18-4. PWM Period and Pulse Width

18.4.4.1 Unbuffered PWM Signal Generation

Any output compare channel can generate unbuffered PWM pulses as described in 18.4.4 Pulse Width Modulation (PWM). The pulses are unbuffered because changing the pulse width requires writing the new pulse width value over the old value currently in the TIM channel registers.

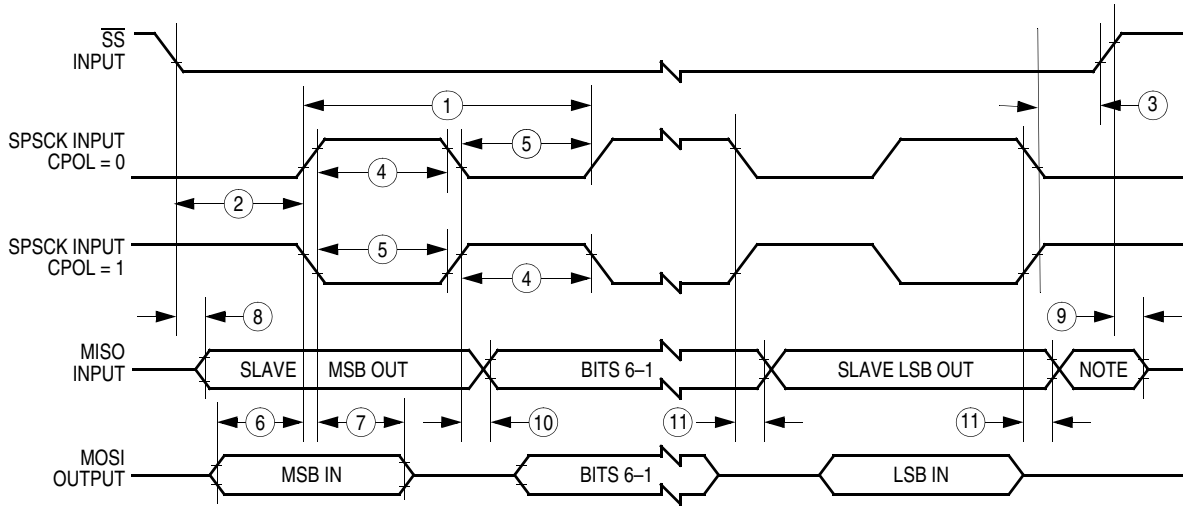
An unsynchronized write to the TIM channel registers to change a pulse width value could cause incorrect operation for up to two PWM periods. For example, writing a new value before the counter reaches the old value but after the counter reaches the new value prevents any compare during that PWM period. Also, using a TIM overflow interrupt routine to write a new, smaller pulse width value may cause the compare to be missed. The TIM may pass the new value before it is written.

Use the following methods to synchronize unbuffered changes in the PWM pulse width on channel x:

- When changing to a shorter pulse width, enable channel x output compare interrupts and write the new value in the output compare interrupt routine. The output compare interrupt occurs at the end of the current pulse. The interrupt routine has until the end of the PWM period to write the new value.

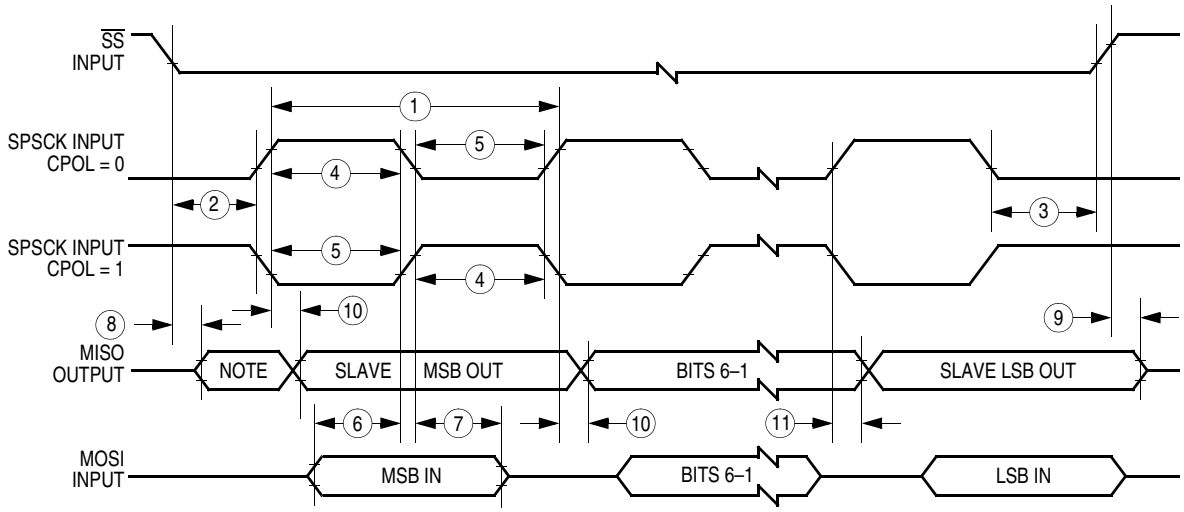
Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Capacitance Ports (as input or output)	C_{Out} C_{In}	— —	— —	12 8	pF
Monitor mode entry voltage	V_{TST}	V_{DD} + 2.5	—	V_{DD} + 4.0	V
Low-voltage inhibit, trip falling voltage	V_{TRIPF}	3.90	4.25	4.50	V
Low-voltage inhibit, trip rising voltage	V_{TRIPR}	4.20	4.35	4.60	V
Low-voltage inhibit reset/recover hysteresis (V_{TRIPF} + V_{HYS} = V_{TRIPR})	V_{HYS}	—	100	—	mV
POR rearm voltage[8]	V_{POR}	0	—	100	mV
POR reset voltage[9]	V_{PORRST}	0	700	800	mV
POR rise time ramp rate[10]	R_{POR}	0.035	—	—	V/ms

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_A (min) to T_A (max), unless otherwise noted
2. Typical values reflect average measurements at midpoint of voltage range, 25°C only.
3. Run (operating) I_{DD} measured using external square wave clock source (f_{OSC} = 32 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects run I_{DD}. Measured with all modules enabled.
4. Wait I_{DD} measured using external square wave clock source (f_{OSC} = 32 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. C_L = 20 pF on OSC2. All ports configured as inputs. OSC2 capacitance linearly affects wait I_{DD}. Measured with CGM and LVI enabled.
5. Stop I_{DD} is measured with OSC1 = V_{SS}. All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All ports configured as inputs. Typical values at midpoint of voltage range, 25°C only.
6. Stop I_{DD} with TBM enabled is measured using an external square wave clock source (f_{OSC} = 32 MHz). All inputs 0.2 V from rail. No dc loads. Less than 100 pF on all outputs. All inputs configured as inputs.
7. Pullups and pulldowns are disabled. Port B leakage is specified in 20.10 5.0-Volt ADC Characteristics.
8. Maximum is highest voltage that POR is guaranteed.
9. Maximum is highest voltage that POR is possible.
10. If minimum V_{DD} is not reached before the internal POR reset is released, $\overline{RST}$ must be driven low externally until minimum V_{DD} is reached.



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 20-3. SPI Slave Timing