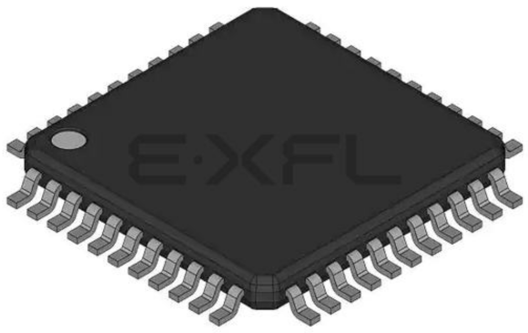


Welcome to E-XFL.COM

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	HC05
Core Size	8-Bit
Speed	2.1MHz
Connectivity	SCI, SPI
Peripherals	POR, WDT
Number of I/O	24
Program Memory Size	16KB (16K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	352 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-QFP
Supplier Device Package	44-QFP (10x10)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc68hc705c9acfb

1.5 Software-Programmable Options (MC68HC05C9A Mode Only)

The C9A option register (OR), shown in Figure 1-4, is enabled only if configured in C9A mode. This register contains the programmable bits for the following options:

- Map two different areas of memory between RAM and EPROM, one of 48 bytes and one of 128 bytes
- Edge-triggered only or edge- and level-triggered external interrupt ($\overline{\text{IRQ}}$ pin and any port B pin configured for interrupt)

This register must be written to by user software during operation of the microcontroller.

$3FDF	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RAM0	RAM1	▨	▨	▨	▨	IRQ	▨
Reset:	0	0	0	0	0	0	1	0

▨ = Unimplemented

Figure 1-4. C9A Option Register

RAM0 — Random Access Memory Control Bit 0

This read/write bit selects between RAM or EPROM in location $0020 to $004F. This bit can be read or written at any time.

1 = RAM selected
0 = EPROM selected

RAM1— Random Access Memory Control Bit 1

This read/write bit selects between RAM or EPROM in location $0100 to $017F. This bit can be read or written at any time.

1 = RAM selected
0 = EPROM selected

IRQ — Interrupt Request Bit

This bit selects between an edge-triggered only or edge- and level- triggered external interrupt pin. This bit is set by reset, but can be cleared by software. This bit can be written only once.

1 = Edge and level interrupt option selected
0 = Edge-only interrupt option selected

1.6 Functional Pin Descriptions

Figure 1-5, Figure 1-6, Figure 1-7, and Figure 1-8 show the pin assignments for the available packages. A functional description of the pins follows.

NOTE

A line over a signal name indicates an active low signal. For example, RESET is active high and $\overline{\text{RESET}}$ is active low.

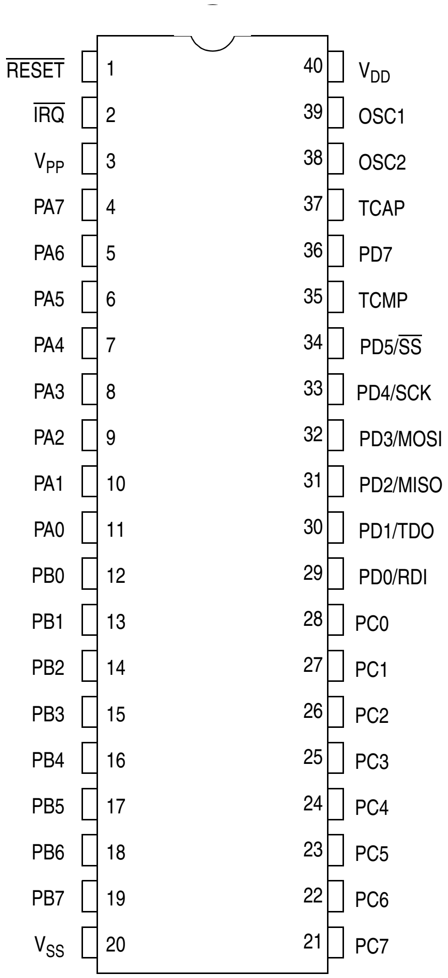
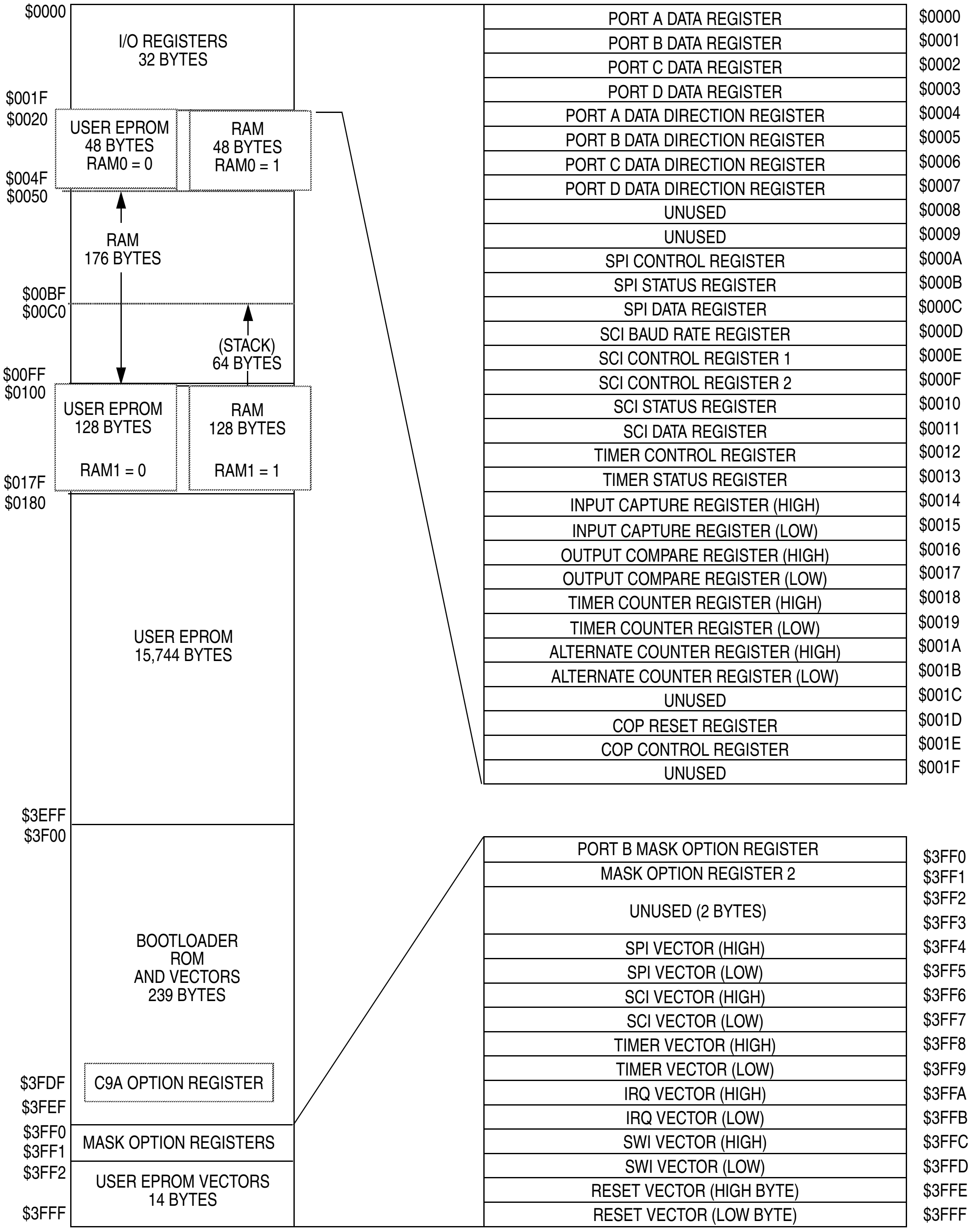


Figure 1-5. 40-Pin PDIP Pin Assignments



PORT A DATA REGISTER	$0000
PORT B DATA REGISTER	$0001
PORT C DATA REGISTER	$0002
PORT D DATA REGISTER	$0003
PORT A DATA DIRECTION REGISTER	$0004
PORT B DATA DIRECTION REGISTER	$0005
PORT C DATA DIRECTION REGISTER	$0006
PORT D DATA DIRECTION REGISTER	$0007
UNUSED	$0008
UNUSED	$0009
SPI CONTROL REGISTER	$000A
SPI STATUS REGISTER	$000B
SPI DATA REGISTER	$000C
SCI BAUD RATE REGISTER	$000D
SCI CONTROL REGISTER 1	$000E
SCI CONTROL REGISTER 2	$000F
SCI STATUS REGISTER	$0010
SCI DATA REGISTER	$0011
TIMER CONTROL REGISTER	$0012
TIMER STATUS REGISTER	$0013
INPUT CAPTURE REGISTER (HIGH)	$0014
INPUT CAPTURE REGISTER (LOW)	$0015
OUTPUT COMPARE REGISTER (HIGH)	$0016
OUTPUT COMPARE REGISTER (LOW)	$0017
TIMER COUNTER REGISTER (HIGH)	$0018
TIMER COUNTER REGISTER (LOW)	$0019
ALTERNATE COUNTER REGISTER (HIGH)	$001A
ALTERNATE COUNTER REGISTER (LOW)	$001B
UNUSED	$001C
COP RESET REGISTER	$001D
COP CONTROL REGISTER	$001E
UNUSED	$001F

PORT B MASK OPTION REGISTER	$3FF0
MASK OPTION REGISTER 2	$3FF1
UNUSED (2 BYTES)	$3FF2 $3FF3
SPI VECTOR (HIGH)	$3FF4
SPI VECTOR (LOW)	$3FF5
SCI VECTOR (HIGH)	$3FF6
SCI VECTOR (LOW)	$3FF7
TIMER VECTOR (HIGH)	$3FF8
TIMER VECTOR (LOW)	$3FF9
IRQ VECTOR (HIGH)	$3FFA
IRQ VECTOR (LOW)	$3FFB
SWI VECTOR (HIGH)	$3FFC
SWI VECTOR (LOW)	$3FFD
RESET VECTOR (HIGH BYTE)	$3FFE
RESET VECTOR (LOW BYTE)	$3FFF

Figure 2-1. C9A Memory Map

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000D	SCI Baud Rate Register BAUD See page 69.	Read:	▨	▨	SCP1	SCP0	▨	SCR2	SCR1	SCR0
		Write:								
		Reset:	—	—	0	0	—	U	U	U
$000E	SCI Control Register 1 (SCCR1) See page 65.	Read:	R8	T8	▨	M	WAKE	▨	▨	▨
		Write:								
		Reset:	U	U	0	U	U	0	0	0
$000F	SCI Control Register 2 (SCCR2) See page 66.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0010	SCI Status Register (SCSR) See page 68.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	▨
		Write:	▨	▨	▨	▨	▨	▨	▨	
		Reset:	1	1	0	0	0	0	0	—
$0011	SCI Data Register (SCDR) See page 65.	Read:	SCD7	SDC6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
		Write:								
		Reset:	Unaffected by reset							
$0012	Timer Control Register (TCR) See page 53.	Read:	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
		Write:				▨	▨	▨		
		Reset:	0	0	0	0	0	0	U	0
$0013	Timer Status Register (TSR) See page 54.	Read:	ICF	OCF	TOF	0	0	0	0	0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	U	U	U	0	0	0	0	0
$0014	Input Capture Register High (ICRH) See page 56.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	Unaffected by reset							
$0015	Input Capture Register Low (ICRL) See page 56.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:	▨	▨	▨	▨	▨	▨	▨	▨
		Reset:	Unaffected by reset							
$0016	Output Compare Register High (OCRH) See page 56.	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
		Reset:	Unaffected by reset							
$0017	Output Compare Register Low (OCRL) See page 56.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Unaffected by reset							
$0018	Timer Register High (TRH) See page 55.	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

▨ = Unimplemented R = Reserved U = Unaffected

Figure 2-4. Input/Output Registers (Sheet 2 of 3)

Chapter 3
Central Processor Unit (CPU)

3.1 Introduction

This section contains the basic programmers model and the registers contained in the CPU.

3.2 CPU Registers

The MCU contains five registers as shown in the programming model of Figure 3-1. The interrupt stacking order is shown in Figure 3-2.

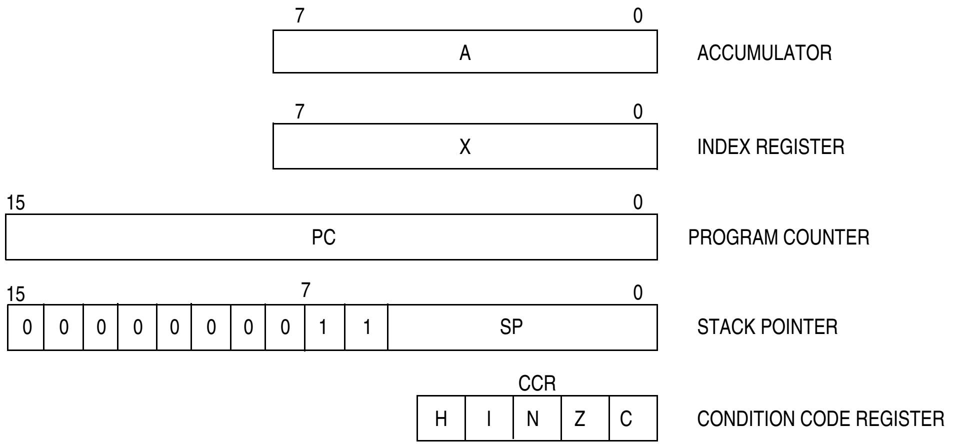


Figure 3-1. Programming Model

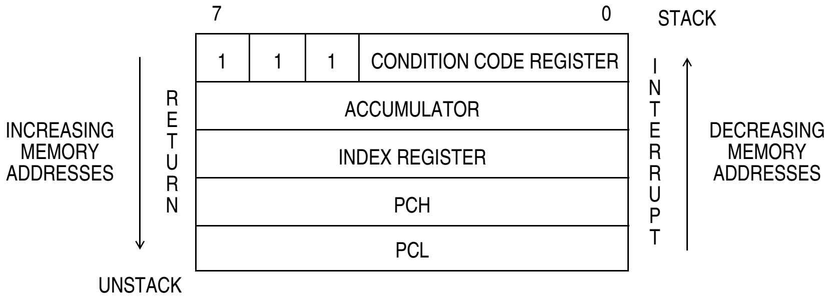


Figure 3-2. Interrupt Stacking Order

Chapter 5
Resets

5.1 Introduction

The MCU can be reset four ways: by the initial power-on reset function, by an active low input to the RESET pin, by the COP, or by the clock monitor. A reset immediately stops the operation of the instruction being executed, initializes some control bits, and loads the program counter with a user-defined reset vector address. Figure 5-1 is a block diagram of the reset sources.

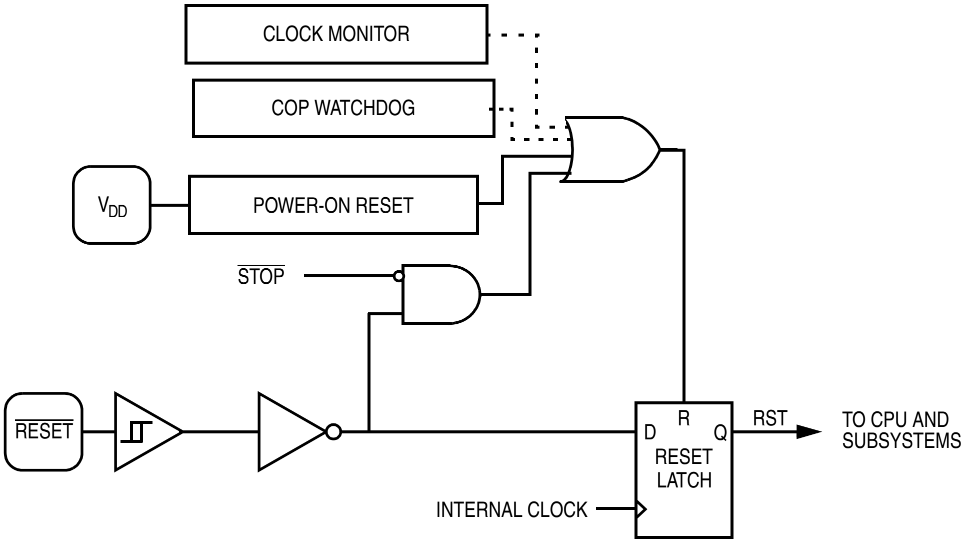


Figure 5-1. Reset Sources

5.2 Power-On Reset (POR)

A power-on-reset occurs when a positive transition is detected on V_{DD}. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{cyc}) oscillator stabilization delay after the oscillator becomes active. (When configured as a C9A, the RESET pin will output a logic 0 during the 4064-cycle delay.) If the RESET pin is low after the end of this 4064-cycle delay, the MCU will remain in the reset condition until RESET is driven high externally.

5.3 RESET Pin

The function of the RESET pin is dependent on whether the device is configured as an MC68HC05C9A or an MC68HC05C12A. When it is in the MC68HC05C12A configuration, the pin is input only. When in MC68HC05C9A configuration the pin is bidirectional. In both cases the MCU is reset when a logic 0 is applied to the RESET pin for a period of one and one-half machine cycles (t_{RL}). For the MC68HC05C9A configuration, the RESET pin will be driven low by a COP, clock monitor, or power-on reset.

$001E	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	COPF	CME	COPE	CM1	CM0
Write:	▨	▨	▨					
Reset:	0	0	0	U	0	0	0	0

▨ = Unimplemented U = Undetermined

Figure 5-5. COP Control Register (COPCR)

COPF — Computer Operating Properly Flag

Reading the COP control register clears COPF.

1 = COP or clock monitor reset has occurred.
0 = No COP or clock monitor reset has occurred.

CME — Clock Monitor Enable Bit

This bit is readable any time, but may be written only once.

1 = Clock monitor enabled
0 = Clock monitor disabled

COPE — COP Enable Bit

This bit is readable any time. COPE, CM1, and CM0 together may be written with a single write, only once, after reset. This bit is cleared by reset.

1 = COP enabled
0 = COP disabled

CM1 — COP Mode Bit 1

Used in conjunction with CM0 to establish the COP timeout period, this bit is readable any time. COPE, CM1, and CM0 together may be written with a single write, only once, after reset. This bit is cleared by reset.

CM0 — COP Mode Bit 0

Used in conjunction with CM1 to establish the COP timeout period, this bit is readable any time. COPE, CM1, and CM0 together may be written with a single write, only once, after reset. This bit is cleared by reset.

Bits 7–5 — Not Used

These bits always read as 0.

Table 5-1. COP Timeout Period

CM1	CM0	$f_{op}/2^{15}$ Divide By	Timeout Period (f_{osc} = 2.0 MHz)	Timeout Period (f_{osc} = 4.0 MHz)
0	0	1	32.77 ms	16.38 ms
0	1	4	131.07 ms	65.54 ms
1	0	16	524.29 ms	262.14 ms
1	1	64	2.097 sec	1.048 sec

5.6 MC68HC05C12A Compatible COP

This COP is implemented with an 18-bit ripple counter. This provides a timeout period of 64 milliseconds at a bus rate (f_{op}) of 2 MHz. If the COP should time out, a system reset will occur and the device will be re-initialized in the same fashion as a power-on reset or reset.

8.2 Timer Operation

The core of the capture/compare timer is a 16-bit free-running counter. The counter provides the timing reference for the input capture and output compare functions. The input capture and output compare functions provide a means to latch the times at which external events occur, to measure input waveforms, and to generate output waveforms and timing delays. Software can read the value in the 16-bit free-running counter at any time without affecting the counter sequence.

Because of the 16-bit timer architecture, the I/O registers for the input capture and output compare functions are pairs of 8-bit registers.

Because the counter is 16 bits long and preceded by a fixed divide-by-4 prescaler, the counter rolls over every 262,144 internal clock cycles. Timer resolution with a 4-MHz crystal is 2 μs.

8.2.1 Input Capture

The input capture function is a means to record the time at which an external event occurs. When the input capture circuitry detects an active edge on the TCAP pin, it latches the contents of the timer registers into the input capture registers. The polarity of the active edge is programmable.

Latching values into the input capture registers at successive edges of the same polarity measures the period of the input signal on the TCAP pin. Latching values into the input capture registers at successive edges of opposite polarity measures the pulse width of the signal.

8.2.2 Output Compare

The output compare function is a means of generating an output signal when the 16-bit counter reaches a selected value. Software writes the selected value into the output compare registers. On every fourth internal clock cycle the output compare circuitry compares the value of the counter to the value written in the output compare registers. When a match occurs, the timer transfers the programmable output level bit (OLVL) from the timer control register to the TCMP pin.

The programmer can use the output compare register to measure time periods, to generate timing delays, or to generate a pulse of specific duration or a pulse train of specific frequency and duty cycle on the TCMP pin.

8.3 Timer I/O Registers

The following I/O registers control and monitor timer operation:

- Timer control register (TCR)
- Timer status register (TSR)
- Timer registers (TRH and TRL)
- Alternate timer registers (ATRH and ATRL)
- Input capture registers (ICRH and ICRL)
- Output compare registers (OCRH and OCRL)

8.3.3 Timer Registers

The timer registers (TRH and TRL), shown in Figure 8-4, contains the current high and low bytes of the 16-bit counter. Reading TRH before reading TRL causes TRL to be latched until TRL is read. Reading TRL after reading the timer status register clears the timer overflow flag (TOF). Writing to the timer registers has no effect.

TRH $0018	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	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	1	1	1	1	1	1	1	1

TRL $0019	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:	1	1	1	1	1	1	0	0

▨ = Unimplemented

Figure 8-4. Timer Registers (TRH and TRL)

8.3.4 Alternate Timer Registers

The alternate timer registers (ATRH and ATRL), shown in Figure 8-5, contain the current high and low bytes of the 16-bit counter. Reading ATRH before reading ATRL causes ATRL to be latched until ATRL is read. Reading ATRL has no effect on the timer overflow flag (TOF). Writing to the alternate timer registers has no effect.

ATRH $001A	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:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	1	1	1	1	1	1	1	1

ATRL $001B	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:	1	1	1	1	1	1	0	0

▨ = Unimplemented

Figure 8-5. Alternate Timer Registers (ATRH and ATRL)

NOTE

To prevent interrupts from occurring between readings of ATRH and ATRL, set the interrupt flag in the condition code register before reading ATRH, and clear the flag after reading ATRL.

The SPI is double buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit in the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave select start logic receives a logic low at the $\overline{SS}$ pin and a clock at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the MOSI line and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the slave's MISO line.

Figure 10-3 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

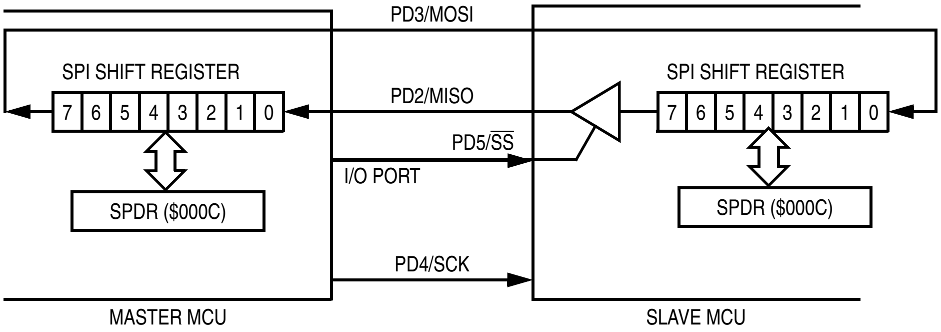


Figure 10-3. Serial Peripheral Interface Master-Slave Interconnection

10.5 SPI Registers

Three registers in the SPI provide control, status, and data storage functions. These registers are called the serial peripheral control register (SPCR), serial peripheral status register (SPSR), and serial peripheral data I/O register (SPDR) and are described in the following paragraphs.

10.5.1 Serial Peripheral Control Register

The SPI control register (SPCR), shown in Figure 10-4, controls these functions:

- Enables SPI interrupts
- Enables the SPI system
- Selects between standard CMOS or open drain outputs for port D (C9A mode only)
- Selects between master mode and slave mode
- Controls the clock/data relationship between master and slave
- Determines the idle level of the clock pin

11.2.4 Extended

Extended instructions use three bytes and can access any address in memory. The first byte is the opcode; the second and third bytes are the high and low bytes of the operand address.

When using the Freescale assembler, the programmer does not need to specify whether an instruction is direct or extended. The assembler automatically selects the shortest form of the instruction.

11.2.5 Indexed, No Offset

Indexed instructions with no offset are 1-byte instructions that can access data with variable addresses within the first 256 memory locations. The index register contains the low byte of the effective address of the operand. The CPU automatically uses $00 as the high byte, so these instructions can address locations $0000–$00FF.

Indexed, no offset instructions are often used to move a pointer through a table or to hold the address of a frequently used random-access memory (RAM) or input/output (I/O) location.

11.2.6 Indexed, 8-Bit Offset

Indexed, 8-bit offset instructions are 2-byte instructions that can access data with variable addresses within the first 511 memory locations. The CPU adds the unsigned byte in the index register to the unsigned byte following the opcode. The sum is the effective address of the operand. These instructions can access locations $0000–$01FE.

Indexed 8-bit offset instructions are useful for selecting the kth element in an n-element table. The table can begin anywhere within the first 256 memory locations and could extend as far as location 510 ($01FE). The k value is typically in the index register, and the address of the beginning of the table is in the byte following the opcode.

11.2.7 Indexed, 16-Bit Offset

Indexed, 16-bit offset instructions are 3-byte instructions that can access data with variable addresses at any location in memory. The CPU adds the unsigned byte in the index register to the two unsigned bytes following the opcode. The sum is the effective address of the operand. The first byte after the opcode is the high byte of the 16-bit offset; the second byte is the low byte of the offset.

Indexed, 16-bit offset instructions are useful for selecting the kth element in an n-element table anywhere in memory.

As with direct and extended addressing, the Freescale assembler determines the shortest form of indexed addressing.

11.2.8 Relative

Relative addressing is only for branch instructions. If the branch condition is true, the CPU finds the effective branch destination by adding the signed byte following the opcode to the contents of the program counter. If the branch condition is not true, the CPU goes to the next instruction. The offset is a signed, two's complement byte that gives a branching range of –128 to +127 bytes from the address of the next location after the branch instruction.

When using the Freescale assembler, the programmer does not need to calculate the offset, because the assembler determines the proper offset and verifies that it is within the span of the branch.

Table 11-6. Instruction Set Summary (Sheet 2 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X	Bit Test Accumulator with Memory Byte	(A) ∧ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff	2 3 4 5 4 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
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? I = 0	—	—	—	—	—	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? N = 1	—	—	—	—	—	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? I = 1	—	—	—	—	—	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? Z = 0	—	—	—	—	—	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? N = 0	—	—	—	—	—	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel* ? 1 = 1	—	—	—	—	—	REL	20	rr	3
BRCLR *n opr rel*	Branch if Bit n Clear	PC ← (PC) + 2 + *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* ? 1 = 0	—	—	—	—	—	REL	21	rr	3
BRSET *n opr rel*	Branch if Bit n Set	PC ← (PC) + 2 + *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	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	5 5 5 5 5 5 5 5
BSR *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) – 1; push (PCH) SP ← (SP) – 1 PC ← (PC) + *rel*	—	—	—	—	—	REL	AD	rr	6
CLC	Clear Carry Bit	C ← 0	—	—	—	—	0	INH	98		2
CLI	Clear Interrupt Mask	I ← 0	—	0	—	—	—	INH	9A		2

Table 11-6. Instruction Set Summary (Sheet 5 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
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		9
RTS	Return from Subroutine	SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	—	—	—	—	—	INH	81		6
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X	Subtract Memory Byte and Carry Bit from Accumulator	A ← (A) – (M) – (C)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A2 B2 C2 D2 E2 F2	ii dd hh ll ee ff ff	2 3 4 5 4 3
SEC	Set Carry Bit	C ← 1	—	—	—	—	1	INH	99		2
SEI	Set Interrupt Mask	I ← 1	—	1	—	—	—	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X	Store Accumulator in Memory	M ← (A)	—	—	↕	↕	—	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	4 5 6 5 4
STOP	Stop Oscillator and Enable IRQ Pin		—	0	—	—	—	INH	8E		2
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X	Store Index Register In Memory	M ← (X)	—	—	↕	↕	—	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X	Subtract Memory Byte from Accumulator	A ← (A) – (M)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A0 B0 C0 D0 E0 F0	ii dd hh ll ee ff ff	2 3 4 5 4 3
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		10
TAX	Transfer Accumulator to Index Register	X ← (A)	—	—	—	—	—	INH	97		2
TST *opr* TSTA TSTX TST *opr*,X TST ,X	Test Memory Byte for Negative or Zero	(M) – $00	—	—	↕	↕	—	DIR INH INH IX1 IX	3D 4D 5D 6D 7D	dd ff	4 3 3 5 4

12.5 5.0-Vdc Electrical Characteristics

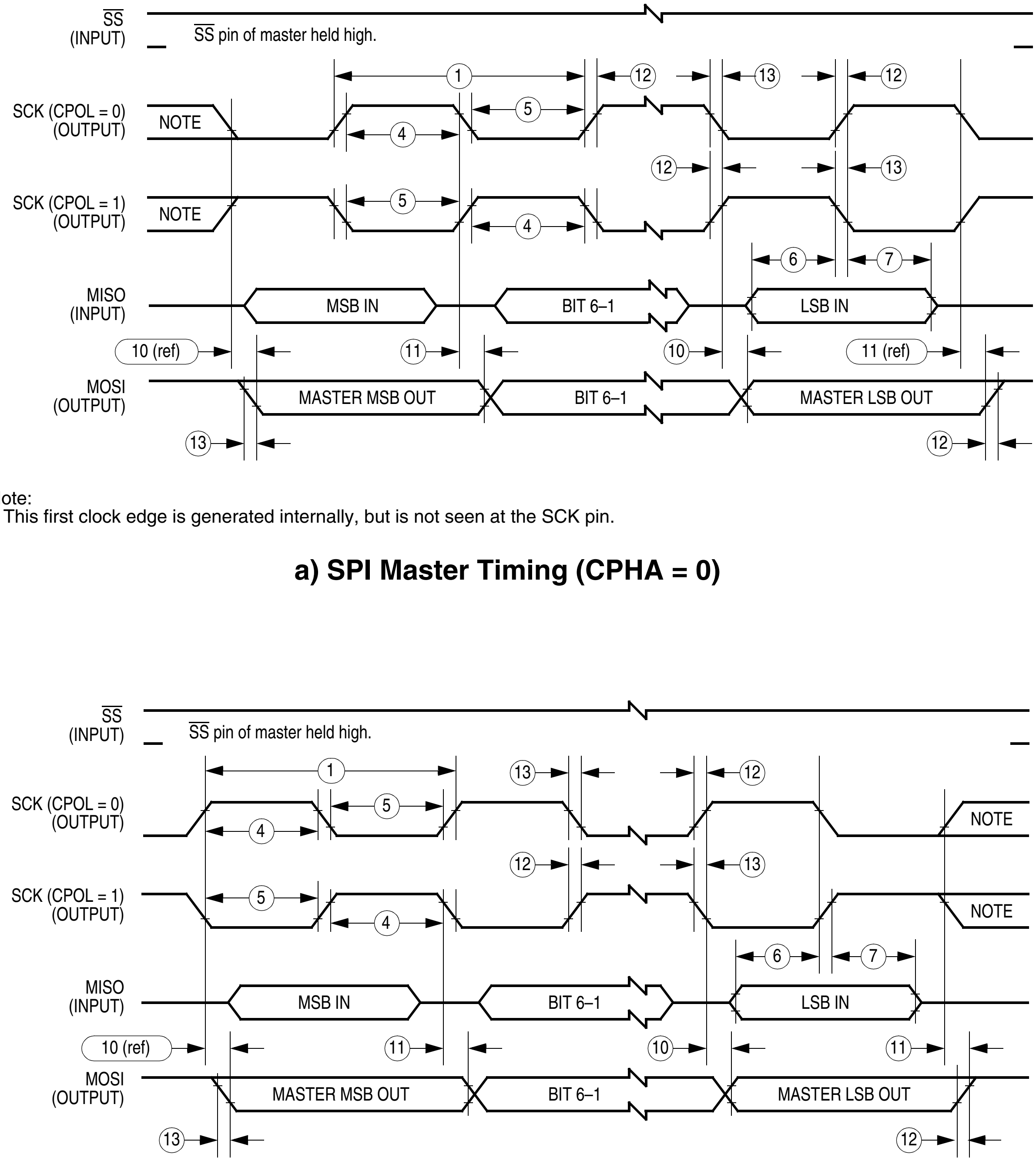
Characteristic[1]	Symbol	Min	Typ[2]	Max	Unit
Output voltage I_{Load} = 10.0 µA I_{Load} = –10.0 µA	V_{OL} V_{OH}	— V_{DD} –0.1	— —	0.1 —	V
Output high voltage (I_{Load} = –0.8 mA) PA7–PA0, PB7–PB0, PC6–PC0, TCMP, PD7, PD0 (I_{Load} = –1.6 mA) PD5–PD1 (I_{Load} = –5.0 mA) PC7	V_{OH}	V_{DD} –0.8 V_{DD} –0.8 V_{DD} –0.8	— — —	— — —	V
Output low voltage (I_{Load} = 1.6 mA) PA7–PA0, PB7–PB0, PC6–PC0, PD7, PD5–PD0, TCMP (I_{Load} = 10 mA) PC7	V_{OL}	— —	— —	0.4 0.4	V
Input high voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input low voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IL}	V_{SS}	—	0.2 × V_{DD}	V
Supply current (4.5–5.5 Vdc @ f_{OP} = 2.1 MHz) Run[3] Wait[4] Stop[5] 25°C –40 to 85 °C	I_{DD}	— — — —	3.5 1.0 1.0 7.0	5.25 3.25 20.0 50.0	mA mA µA µA
I/O ports hi-Z leakage current PA7–PA0, PB7–PB0 (without pullup) PC7–PC0, PD7, PD5–PD0	I_{OZ}	—	—	10	µA
Input current $\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, OSC1, TCAP, PD7, PD5–PD0	I_{In}	—	—	1	µA
Input pullup current[6] PB7–PB0 (with pullup)	I_{In}	5	—	60	µA
Capacitance Ports (as input or output) $\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, OSC1, TCAP, PD7, PD5, PD0	C_{Out} C_{In}	— —	— —	12 8	pF
Programming voltage (25°C)	V_{PP}	15.0	16.0	17.0	V
Programming current (25°C)	I_{PP}	—	—	200	mA

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 to +85 °C, unless otherwise noted
2. Typical values reflect measurements taken on average processed devices at the midpoint of voltage range, 25 °C only.
3. Run (operating) I_{DD} measured using external square wave clock source; all I/O pins configured as inputs, port B = V_{DD}, all other inputs V_{IL} = 0.2 V, V_{IH} = V_{DD}–0.2 V; no DC loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2
4. Wait I_{DD} measured using external square wave clock source; all I/O pins configured as inputs, port B = V_{DD}, all other inputs V_{IL} = 0.2 V, V_{IH} = V_{DD} –0.2 V; no DC loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2. Wait I_{DD} is affected linearly by the OSC2 capacitance.
5. Stop I_{DD} measured with OSC1 = 0.2 V; all I/O pins configured as inputs, port B = V_{DD}, all other inputs V_{IL} = 0.2 V, V_{IH} = V_{DD} –0.2 V.
6. Input pullup current measured with V_{IL} = 0.2 V.

12.7 5.0-Vdc Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation Crystal External clock	f_{OSC}	— DC	4.2 4.2	MHz
Internal pperating frequency (f_{OSC} ÷ 2) Crystal External clock	f_{OP}	— DC	2.1 2.1	MHz
Cycle time	t_{CYC}	480	—	ns
Crystal oscillator startup time	t_{OXOV}	—	100	ms
Stop recovery startup time (crystal oscillator)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ pulse width	t_{RL}	1.5	—	t_{CYC}
Timer Resolution[2] Input capture pulse width Input capture pulse period	t_{RESL} t_{TH}, t_{TL} t_{TLTL}	4.0 125 [3]	— — —	t_{CYC} ns t_{CYC}
Interrupt pulse width low (edge-triggered)	t_{ILIH}	125	—	ns
Interrupt pulse period	t_{ILIL}	[4]	—	t_{CYC}
OSC1 pulse width	t_{OH},t_{OL}	90	—	ns

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 to +85 °C, unless otherwise noted
2. Because a 2-bit prescaler in the timer must count four internal cycles (t_{CYC}), this is the limiting minimum factor in determining the timer resolution.
3. The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{CYC}.
4. The minimum t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 19 t_{CYC}.



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 12-9. SPI Master Timing Diagram