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	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	21
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf24j10t-i-ss

PIC18F45J10 FAMILY

28/40/44-Pin High-Performance, RISC Microcontrollers

Special Microcontroller Features:

- Operating Voltage Range: 2.0V to 3.6V
- 5.5V Tolerant Input (digital pins only)
- On-Chip 2.5V Regulator
- 4x Phase Lock Loop (PLL) available for Crystal and Internal Oscillators
- Self-Programmable under Software Control
- Low-Power, High-Speed CMOS Flash Technology
- C Compiler Optimized Architecture:
  - Optional extended instruction set designed to optimize re-entrant code
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) with Three Breakpoints via Two Pins
- Power-Managed modes with Clock Switching:
  - Run: CPU on, peripherals on
  - Idle: CPU off, peripherals on
  - Sleep: CPU off, peripherals off

Flexible Oscillator Structure:

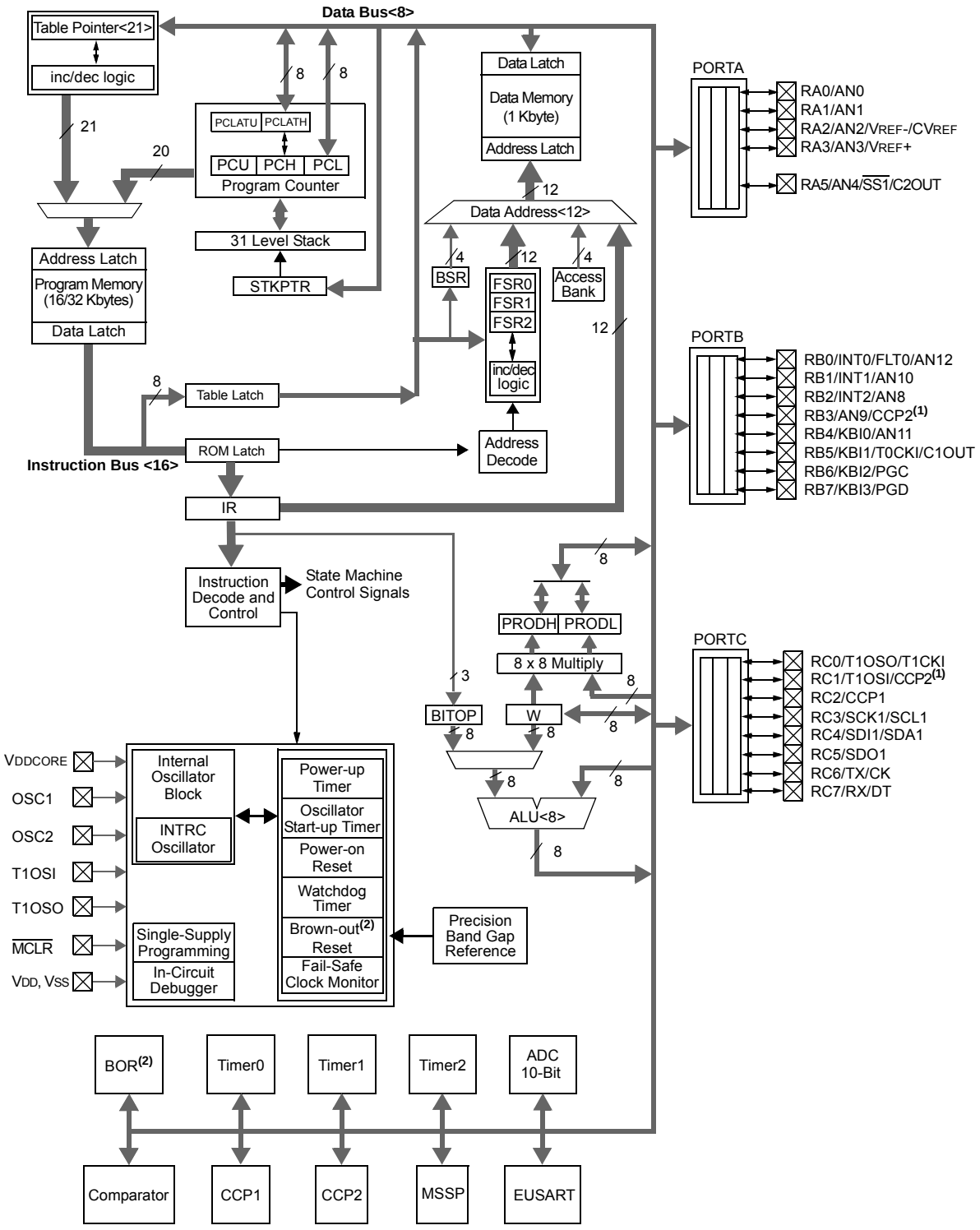
- Two Crystal modes, up to 40 MHz
- Two External Clock modes, up to 40 MHz
- Internal 31 kHz Oscillator
- Secondary Oscillator using Timer1 @ 32 kHz
- Two-Speed Oscillator Start-up
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if peripheral clock stops

Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA (PORTB and PORTC)
- Three Programmable External Interrupts
- Four Input Change Interrupts
- One Capture/Compare/PWM (CCP) module
- One Enhanced Capture/Compare/PWM (ECCP) module:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
- Two Master Synchronous Serial Port (MSSP) modules supporting 3-Wire SPI (all 4 modes) and I^2C™ Master and Slave modes
- One Enhanced Addressable USART module:
  - Supports RS-485, RS-232 and LIN/J2602
  - Auto-wake-up on Start bit
  - Auto-Baud Detect (ABD)
- 10-Bit, up to 13-Channel Analog-to-Digital Converter module (A/D):
  - Auto-acquisition capability
  - Conversion available during Sleep
  - Self-calibration feature
- Dual Analog Comparators with Input Multiplexing

Device	Program Memory		SRAM Data Memory (bytes)	I/O	10-Bit A/D (ch)	CCP/ ECCP (PWM)	MSSP			EUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions						SPI	Master I^2C™			
PIC18F24J10	16K	8192	1024	21	10	2/0	1	Y	Y	1	2	1/2
PIC18F25J10	32K	16384	1024	21	10	2/0	1	Y	Y	1	2	1/2
PIC18F44J10	16K	8192	1024	32	13	1/1	2	Y	Y	1	2	1/2
PIC18F45J10	32K	16384	1024	32	13	1/1	2	Y	Y	1	2	1/2

FIGURE 1-1: PIC18F24J10/25J10 (28-PIN) BLOCK DIAGRAM



Note 1: CCP2 is multiplexed with RC1 when Configuration bit, CCP2MX, is set, or RB3 when CCP2MX is not set.

2: Brown-out Reset is not available in PIC18LF2XJ10/4XJ10 devices.

TABLE 1-2: PIC18F24J10/25J10 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	SPDIP, SOIC, SSOP	QFN			
					PORTA is a bidirectional I/O port.
RA0/AN0	2	27			
RA0			I/O	TTL	Digital I/O.
AN0			I	Analog	Analog Input 0.
RA1/AN1	3	28			
RA1			I/O	TTL	Digital I/O.
AN1			I	Analog	Analog Input 1.
RA2/AN2/VREF-/CVREF	4	1			
RA2			I/O	TTL	Digital I/O.
AN2			I	Analog	Analog Input 2.
VREF-			I	Analog	A/D reference voltage (low) input.
CVREF			O	Analog	Comparator reference voltage output.
RA3/AN3/VREF+	5	2			
RA3			I/O	TTL	Digital I/O.
AN3			I	Analog	Analog Input 3.
VREF+			I	Analog	A/D reference voltage (high) input.
RA5/AN4/$\overline{SS1}$/C2OUT	7	4			
RA5			I/O	TTL	Digital I/O.
AN4			I	Analog	Analog Input 4.
$\overline{SS1}$			I	TTL	SPI slave select input.
C2OUT			O	—	Comparator 2 output.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Default assignment for CCP2 when Configuration bit, CCP2MX, is set.
2: Alternate assignment for CCP2 when Configuration bit, CCP2MX, is cleared.

5.2 Master Clear ($\overline{MCLR}$)

The $\overline{MCLR}$ pin provides a method for triggering a hard external Reset of the device. A Reset is generated by holding the pin low. PIC18 extended microcontroller devices have a noise filter in the $\overline{MCLR}$ Reset path which detects and ignores small pulses.

The $\overline{MCLR}$ pin is not driven low by any internal Resets, including the WDT.

5.3 Power-on Reset (POR)

A Power-on Reset condition is generated on-chip whenever VDD rises above a certain threshold. This allows the device to start in the initialized state when VDD is adequate for operation.

To take advantage of the POR circuitry, tie the $\overline{MCLR}$ pin through a resistor (1 kΩ to 10 kΩ) to VDD. This will eliminate external RC components usually needed to create a Power-on Reset delay. A minimum rise rate for VDD is specified (parameter D004). For a slow rise time, see Figure 5-2.

When the device starts normal operation (i.e., exits the Reset condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not met, the device must be held in Reset until the operating conditions are met.

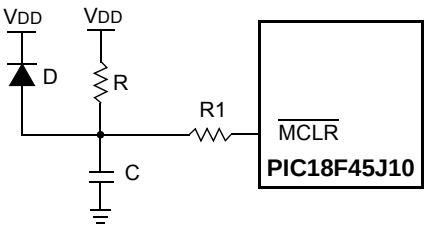
Power-on Reset events are captured by the $\overline{POR}$ bit (RCON<1>). The state of the bit is set to '0' whenever a Power-on Reset occurs; it does not change for any other Reset event. $\overline{POR}$ is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to '1' in software following any Power-on Reset.

5.4 Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)

The PIC18F45J10 family of devices incorporates a simple BOR function when the internal regulator is enabled (ENVREG pin is tied to VDD). Any drop of VDD below VBOR (parameter D005) for greater than time TBOR (parameter 35) will reset the device. A Reset may or may not occur if VDD falls below VBOR for less than TBOR. The chip will remain in Brown-out Reset until VDD rises above VBOR.

Once a BOR has occurred, the Power-up Timer will keep the chip in Reset for TPWRT (parameter 33). If VDD drops below VBOR while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be initialized. Once VDD rises above VBOR, the Power-up Timer will execute the additional time delay.

FIGURE 5-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External Power-on Reset circuit is required only if the VDD power-up slope is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 ≥ 1 kΩ will limit any current flowing into $\overline{MCLR}$ from external capacitor C, in the event of $\overline{MCLR}$ pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

5.4.1 DETECTING BOR

The $\overline{BOR}$ bit always resets to '0' on any Brown-out Reset or Power-on Reset event. This makes it difficult to determine if a Brown-out Reset event has occurred just by reading the state of $\overline{BOR}$ alone. A more reliable method is to simultaneously check the state of both $\overline{POR}$ and $\overline{BOR}$. This assumes that the $\overline{POR}$ bit is reset to '1' in software immediately after any Power-on Reset event. If $\overline{BOR}$ is '0' while $\overline{POR}$ is '1', it can be reliably assumed that a Brown-out Reset event has occurred.

In devices designated with an "LF" part number (such as PIC18**LF**25J10), Brown-out Reset functionality is disabled. In this case, the $\overline{BOR}$ bit cannot be used to determine a Brown-out Reset event. The $\overline{BOR}$ bit is still cleared by a Power-on Reset event.

6.4 Data Addressing Modes

Note:	The execution of some instructions in the core PIC18 instruction set are changed when the PIC18 extended instruction set is enabled. See **Section 6.5 "Data Memory and the Extended Instruction Set"** for more information.

While the program memory can be addressed in only one way – through the program counter – information in the data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST Configuration bit = 1). Its operation is discussed in greater detail in **Section 6.5.1 "Indexed Addressing with Literal Offset"**.

6.4.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all; they either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples include SLEEP, RESET and DAW.

Other instructions work in a similar way but require an additional explicit argument in the opcode. This is known as Literal Addressing mode because they require some literal value as an argument. Examples include ADDLW and MOVLW, which respectively, add or move a literal value to the W register. Other examples include CALL and GOTO, which include a 20-bit program memory address.

6.4.2 DIRECT ADDRESSING

Direct Addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of Direct Addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 6.3.3 "General Purpose Register File"**) or a location in the Access Bank (**Section 6.3.2 "Access Bank"**) as the data source for the instruction.

The Access RAM bit 'a' determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 6.3.1 "Bank Select Register (BSR)"**) are used with the address to determine the complete 12-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFF, include the entire 12-bit address (either source or destination) in their opcodes. In these cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

6.4.3 INDIRECT ADDRESSING

Indirect Addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations to be read or written to. Since the FSRs are themselves located in RAM as Special Function Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for Indirect Addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 6-5.

EXAMPLE 6-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR    FSR0, 100h ;
NEXT    CLRF    POSTINC0   ; Clear INDF
                           ; register then
                           ; inc pointer
        BTFSS   FSR0H, 1   ; All done with
                           ; Bank1?
        BRA     NEXT       ; NO, clear next
CONTINUE                   ; YES, continue
```

7.5 Writing to Flash Program Memory

The minimum programming block is 32 words or 64 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 64 holding registers used by the table writes for programming.

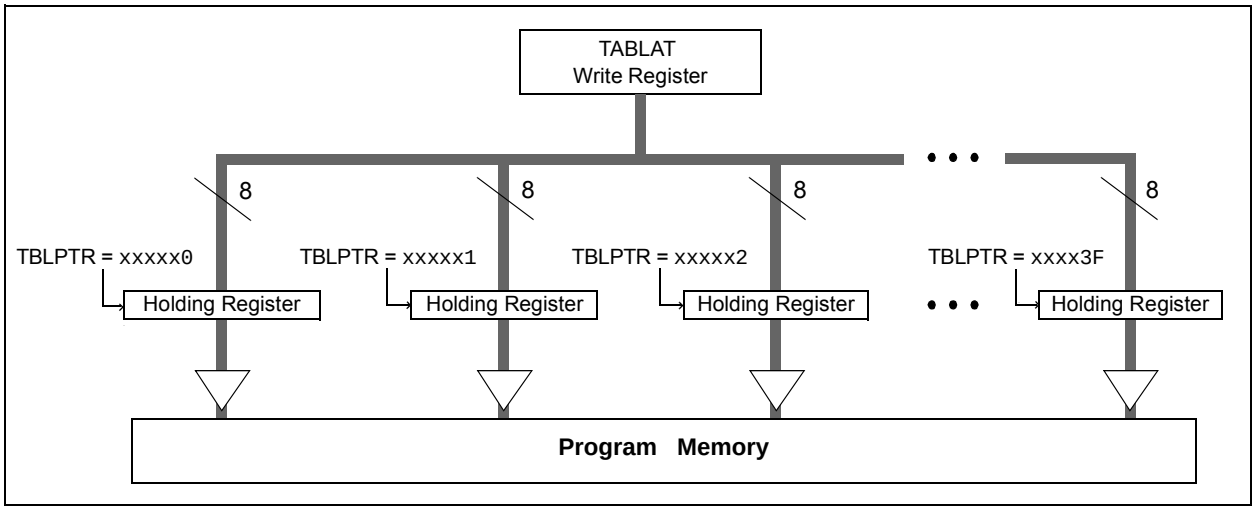
Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction may need to be executed 64 times for each programming operation. All of the table write operations will essentially be short writes because only the holding registers are written. At the end of updating the 64 holding registers, the EECON1 register must be written to in order to start the programming operation with a long write.

The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The EEPROM on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device.

Note:	Unlike previous devices, the PIC18F45J10 family of devices does not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence. In order to maintain the endurance of the cells, each Flash byte should not be programmed more then twice between erase operations. Either a Bulk or Row Erase of the target row is required before attempting to modify the contents a third time.

FIGURE 7-5: TABLE WRITES TO FLASH PROGRAM MEMORY



7.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. If the section of program memory to be written to has been programmed previously, then the memory will need to be erased before the write occurs (see **Section 7.4.1 "Flash Program Memory Erase Sequence"**).
2. Write the 64 bytes into the holding registers with auto-increment.
3. Set the EECON1 register for the write operation:
   - set WREN to enable byte writes.
4. Disable interrupts.
5. Write 55h to EECON2.
6. Write 0AAh to EECON2.
7. Set the WR bit. This will begin the write cycle.
8. The CPU will stall for duration of the write (about 2 ms using internal timer).
9. Re-enable interrupts.
10. Verify the memory (table read).

An example of the required code is shown in Example 7-3.

Note:	Before setting the WR bit, the Table Pointer address needs to be within the intended address range of the 64 bytes in the holding register.

14.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F45J10 family devices all have two CCP (Capture/Compare/PWM) modules. Each module contains a 16-bit register which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register.

In 28-pin devices, the two standard CCP modules (CCP1 and CCP2) operate as described in this chapter. In 40/44-pin devices, CCP1 is implemented as an Enhanced CCP module (ECCP1) with standard Capture and Compare modes and Enhanced PWM modes. The Enhanced CCP implementation is discussed in **Section 15.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**.

The Capture and Compare operations described in this chapter apply to all standard and Enhanced CCP modules.

Note: Throughout this section and **Section 15.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**, references to the register and bit names for CCP modules are referred to generically by the use of 'x' or 'y' in place of the specific module number. Thus, "CCPxCON" might refer to the control register for CCP1, CCP2 or ECCP1. "CCPxCON" is used throughout these sections to refer to the module control register regardless of whether the CCP module is a standard or Enhanced implementation.

REGISTER 14-1: CCPxCON: CCP1/CCP2 CONTROL REGISTER IN 28-PIN DEVICES

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **DCxB<1:0>:** PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight MSbs (DCxB<9:2>) of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>:** CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0001 = Reserved
0010 = Compare mode, toggle output on match (CCPxIF bit is set)
0011 = Reserved
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode: initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)
1001 = Compare mode: initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)
1010 = Compare mode: generate software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)
1011 = Compare mode: trigger special event, reset timer, start A/D conversion on CCPx match (CCPxIF bit is set)
11xx = PWM mode

14.1 CCP Module Configuration

Each Capture/Compare/PWM module is associated with a control register (generically, CCPxCON) and a data register (CCPRx). The data register, in turn, is comprised of two 8-bit registers: CCPRxL (low byte) and CCPRxH (high byte). All registers are both readable and writable.

14.1.1 CCP MODULES AND TIMER RESOURCES

The CCP modules utilize Timers 1 or 2, depending on the mode selected. Timer1 is available to modules in Capture or Compare modes, while Timer2 is available for modules in PWM mode.

TABLE 14-1: ECCP/CCP MODE – TIMER RESOURCE

ECCP/CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

Both modules may be active at any given time and may share the same timer resource if they are configured to operate in the same mode (Capture/Compare or PWM) at the same time. The interactions between the two modules are summarized in Figure 14-1 and Figure 14-2. In Timer1 in Asynchronous Counter mode, the capture operation will not work.

14.1.2 CCP2 PIN ASSIGNMENT

The pin assignment for CCP2 (Capture input, Compare and PWM output) can change, based on device configuration. The CCP2MX Configuration bit determines which pin CCP2 is multiplexed to. By default, it is assigned to RC1 (CCP2MX = 1). If the Configuration bit is cleared, CCP2 is multiplexed with RB3.

Changing the pin assignment of CCP2 does not automatically change any requirements for configuring the port pin. Users must always verify that the appropriate TRIS register is configured correctly for CCP2 operation regardless of where it is located.

TABLE 14-2: INTERACTIONS BETWEEN ECCP1/CCP1 AND CCP2 FOR TIMER RESOURCES

CCP1 Mode	CCP2 Mode	Interaction
Capture	Capture	Each module uses TMR1 as the time base.
Capture	Compare	CCP2 can be configured for the Special Event Trigger to reset TMR1. Automatic A/D conversions on the trigger event can also be done. Operation of ECCP1/CCP1 will be affected.
Compare	Capture	ECCP1/CCP1 can be configured for the Special Event Trigger to reset TMR1. Operation of CCP2 will be affected.
Compare	Compare	Either module can be configured for the Special Event Trigger to reset TMR1. Automatic A/D conversions on the CCP2 trigger event can be done.
Capture	PWM(1)	None
Compare	PWM(1)	None
PWM(1)	Capture	None
PWM(1)	Compare	None
PWM(1)	PWM	Both PWMs will have the same frequency and update rate (TMR2 interrupt).

Note 1: Includes standard and Enhanced PWM operation.

FIGURE 15-2: PWM OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)

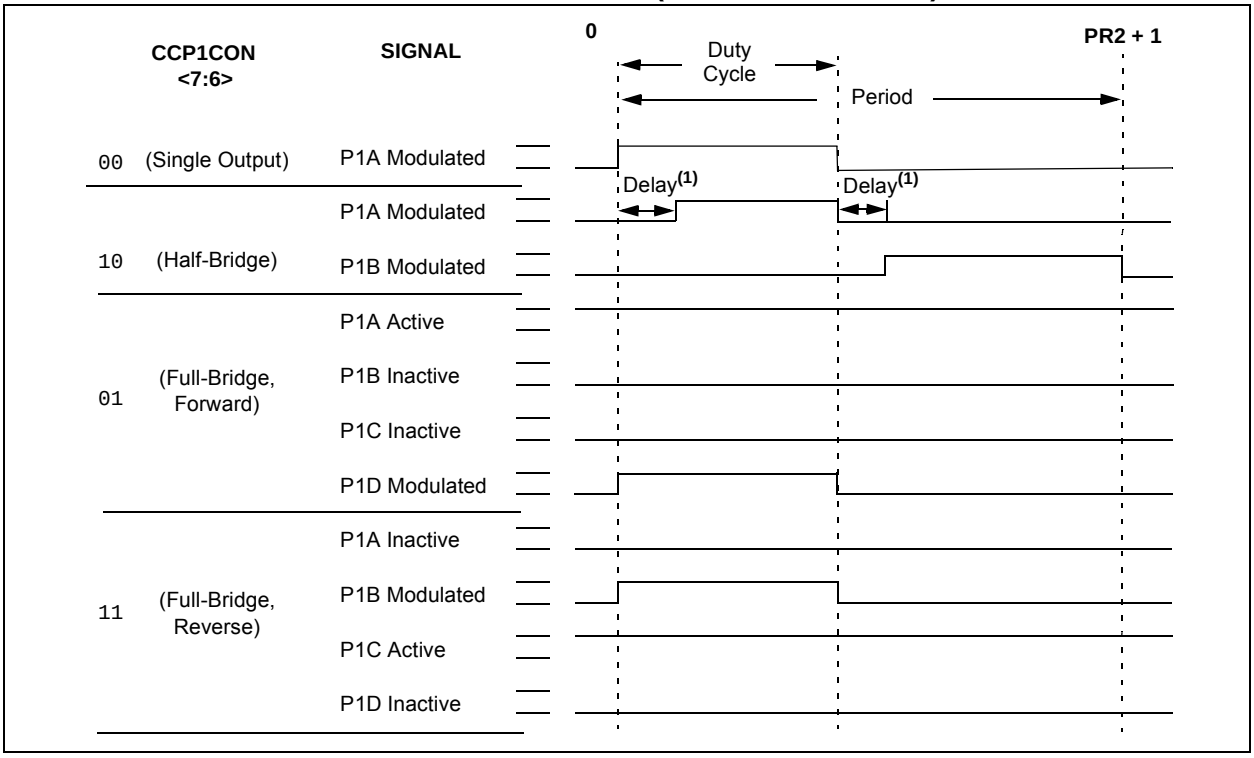
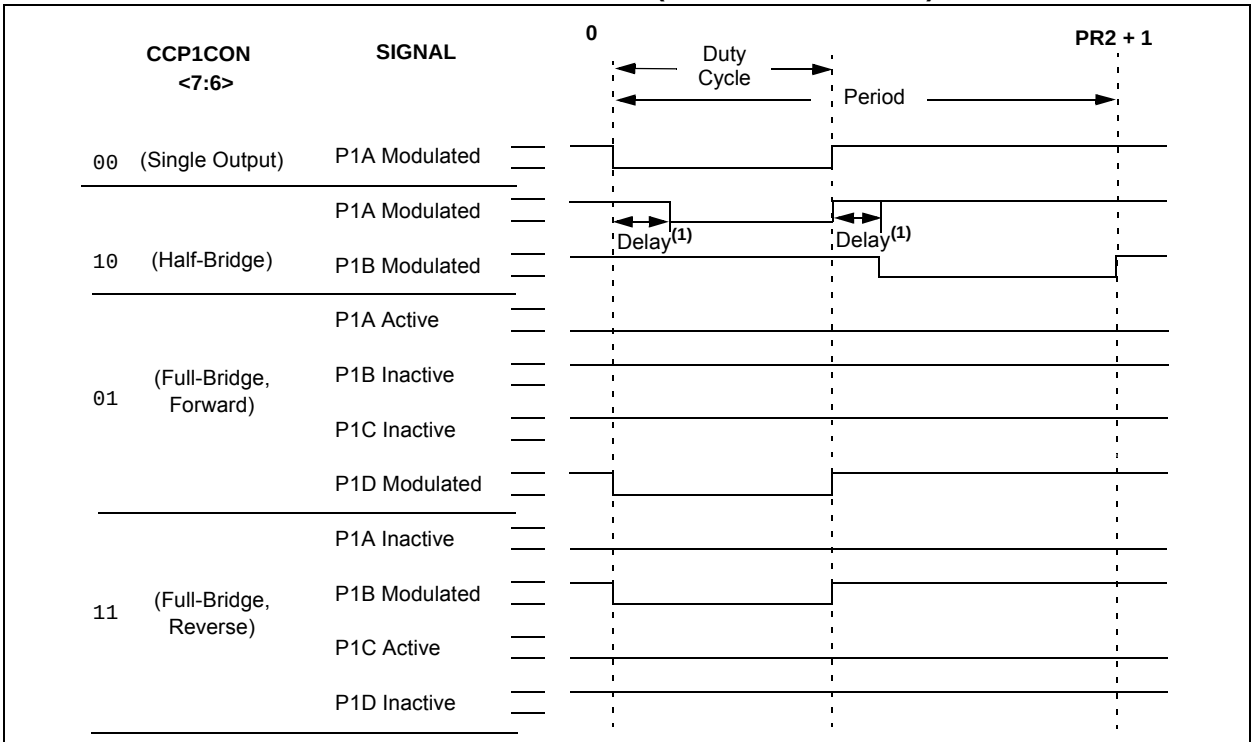


FIGURE 15-3: PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)



Relationships:

- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Duty Cycle = TOSC * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (ECCP1DEL<6:0>)

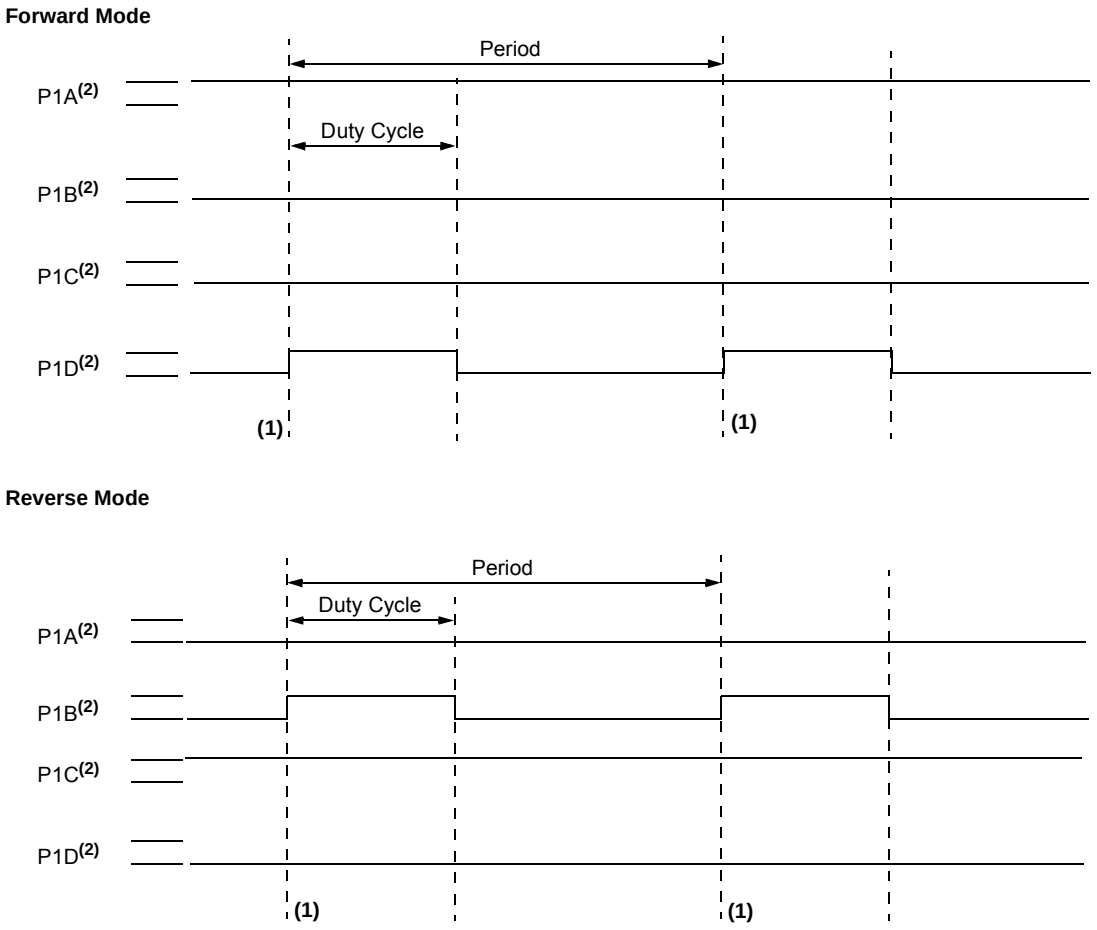
Note 1: Dead-band delay is programmed using the ECCP1DEL register (see **Section 15.4.6 "Programmable Dead-Band Delay"**).

15.4.5 FULL-BRIDGE MODE

In Full-Bridge Output mode, four pins are used as outputs; however, only two outputs are active at a time. In the Forward mode, pin P1A is continuously active and pin P1D is modulated. In the Reverse mode, pin P1C is continuously active and pin P1B is modulated. These are illustrated in Figure 15-6.

P1A, P1B, P1C and P1D outputs are multiplexed with the PORTC<2> and PORTD<7:5> data latches. The TRISC<2> and TRISD<7:5> bits must be cleared to make the P1A, P1B, P1C and P1D pins outputs.

FIGURE 15-6: FULL-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.
2: Output signal is shown as active-high.

16.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

16.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I^2C™)
  - Full Master mode
  - Slave mode (with general address call)

The I^2C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

PIC18F24J10/25J10 (28-pin) devices have one MSSP module designated as MSSP1. PIC18F44J10/45J10 (40/44-pin) devices have two MSSP modules, designated as MSSP1 and MSSP2. Each module operates independently of the other.

Note:	Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names and module I/O signals use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module, when required. Control bit names are not individuated.

16.2 Control Registers

Each MSSP module has three associated control registers. These include a status register (SSPxSTAT) and two control registers (SSPxCON1 and SSPxCON2). The use of these registers and their individual configuration bits differ significantly depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections.

Note:	Disabling the MSSP module by clearing the SSPEN (SSPxCON1<5>) bit may not reset the module. It is recommended to clear the SSPxSTAT, SSPxCON1 and SSPxCON2 registers and select the mode prior to setting the SSPEN bit to enable the MSSP module.

Note:	In devices with more than one MSSP module, it is very important to pay close attention to SSPxCON register names. SSP1CON1 and SSP1CON2 control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

16.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

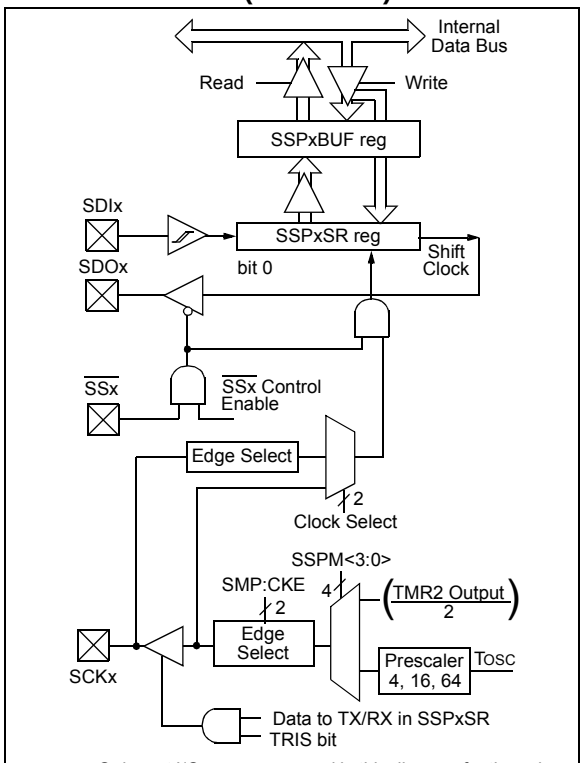
- Serial Data Out (SDOx) – RC5/SDO1 or RD2/PSP2/SDO2
- Serial Data In (SDIx) – RC4/SDI1/SDA1 or RD1/PSP1/SDI2/SDA2
- Serial Clock (SCKx) – RC3/SCK1/SCL1 or RD0/PSP0/SCK2/SCL2

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SSx}$) – RA5/AN4/$\overline{SS1}$/C2OUT or RD3/PSP3/$\overline{SS2}$

Figure 16-1 shows the block diagram of the MSSP module when operating in SPI mode.

FIGURE 16-1: MSSP BLOCK DIAGRAM (SPI MODE)



Note: Only port I/O names are used in this diagram for the sake of brevity. Refer to the text for a full list of multiplexed functions.

16.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCLx output is forced to '0'. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external I^2C master device has already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I^2C bus have deasserted SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 16-12).

FIGURE 16-12: CLOCK SYNCHRONIZATION TIMING

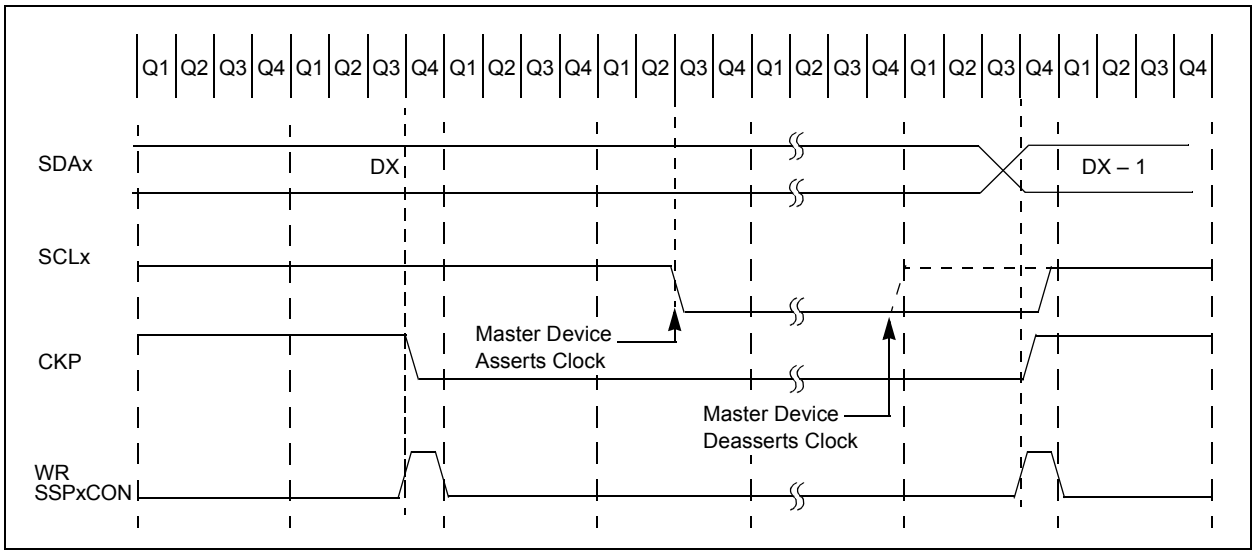
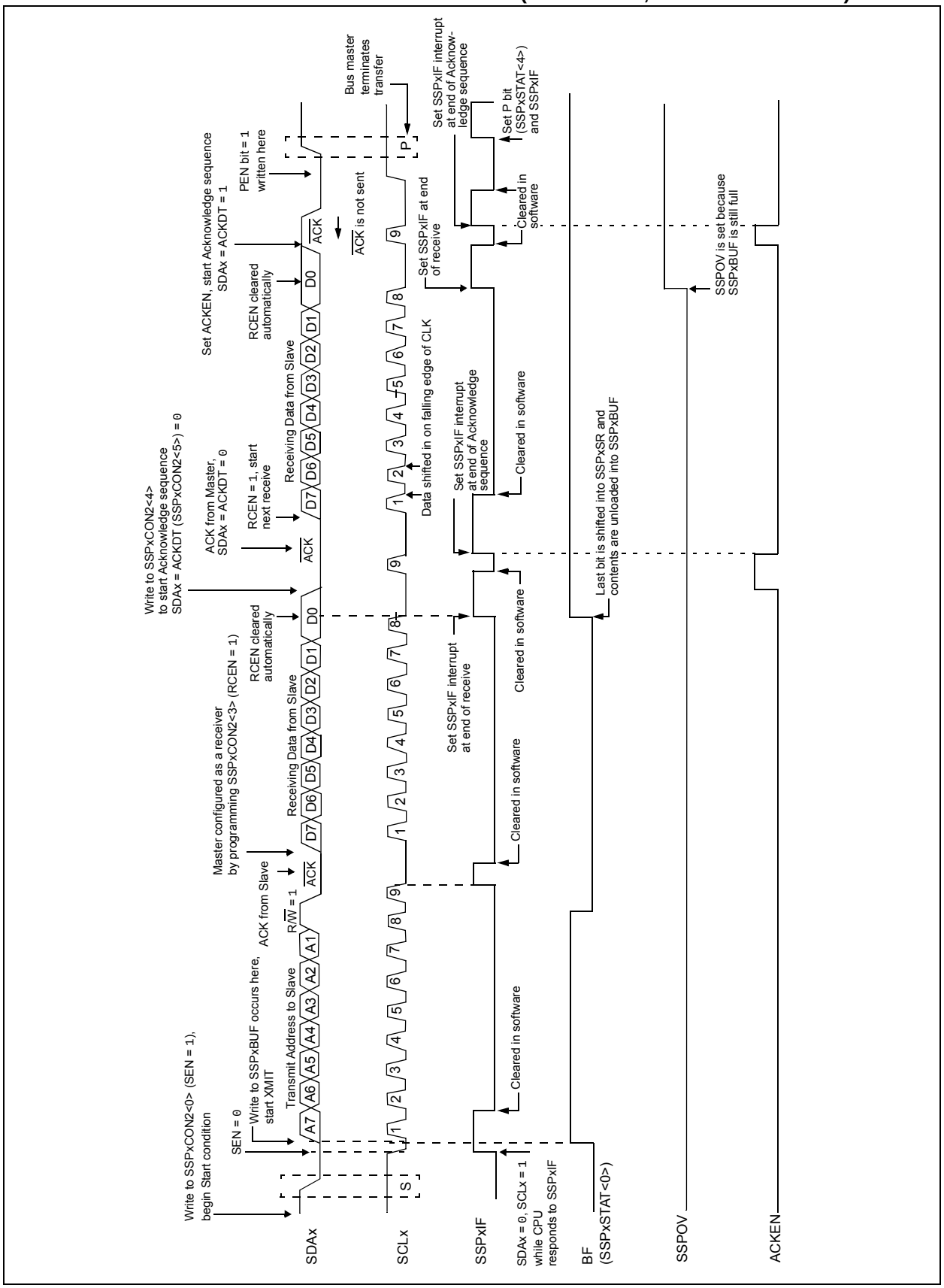


FIGURE 16-22: I²C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESSING)



16.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDAx or SCLx are sampled low at the beginning of the Start condition (Figure 16-26).

b) SCLx is sampled low before SDAx is asserted low (Figure 16-27).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

- the Start condition is aborted;
- the BCLxIF flag is set; and
- the MSSP module is reset to its Idle state (Figure 16-26).

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded from SSPxADD<6:0> and counts down to '0'. If the SCLx pin is sampled low while SDAx is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 16-28). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to 0. If the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx pin is asserted low.

Note:	The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 16-26: BUS COLLISION DURING START CONDITION (SDAx ONLY)

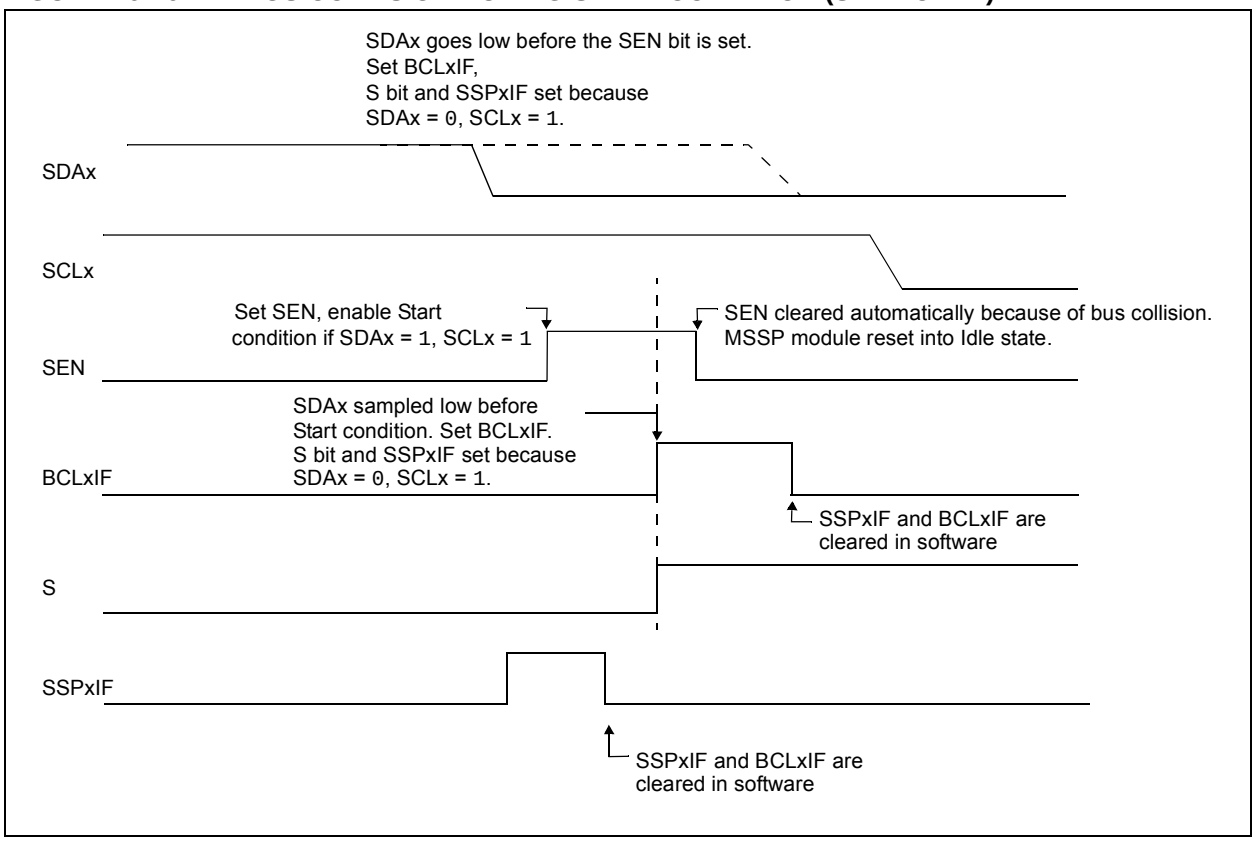


TABLE 17-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

NOTES:

TABLE 22-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS						
ADDLW	k	Add Literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW	k	AND Literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW	k	Inclusive OR Literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR	f, k	Move Literal (12-bit) 2nd Word to FSR(f) 1st Word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB	k	Move Literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW	k	Move Literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW	k	Multiply Literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW	k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW	k	Subtract WREG from Literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR Literal with WREG	1	0000 1010 kkkk kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS						
TBLRD*		Table Read	2	0000 0000 0000 1000	None	
TBLRD*+		Table Read with Post-Increment		0000 0000 0000 1001	None	
TBLRD*-		Table Read with Post-Decrement		0000 0000 0000 1010	None	
TBLRD+*		Table Read with Pre-Increment		0000 0000 0000 1011	None	
TBLWT*		Table Write	2	0000 0000 0000 1100	None	
TBLWT*+		Table Write with Post-Increment		0000 0000 0000 1101	None	
TBLWT*-		Table Write with Post-Decrement		0000 0000 0000 1110	None	
TBLWT+*		Table Write with Pre-Increment		0000 0000 0000 1111	None	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, 'd' = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

DAW — Decimal Adjust W Register

Syntax: DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1] then,
(W<3:0>) + 6 → W<3:0>;
else,
(W<3:0>) → W<3:0>

If [W<7:4> + DC > 9] or [C = 1] then,
(W<7:4>) + 6 + DC → W<7:4>;
else,
(W<7:4>) + DC → W<7:4>

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the eight-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example 1:

```
DAW
```

Before Instruction
W = A5h
C = 0
DC = 0
After Instruction
W = 05h
C = 1
DC = 0

Example 2:

Before Instruction
W = CEh
C = 0
DC = 0
After Instruction
W = 34h
C = 1
DC = 0

DECF — Decrement f

Syntax: DECF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0000	01da	ffff	ffff

Description: Decrement register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default).
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 22.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `DECF CNT, 1, 0`

Before Instruction
CNT = 01h
Z = 0
After Instruction
CNT = 00h
Z = 1

MOVFF **Move f to f**

Syntax: MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register 'f_s' are moved to destination register 'f_d'. Location of source 'f_s' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination 'f_d' can also be anywhere from 000h to FFFh.
Either source or destination can be W (a useful special situation).
MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).
The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 33h
REG2 = 11h
After Instruction
REG1 = 33h
REG2 = 33h

MOVLB **Move Literal to Low Nibble in BSR**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0', regardless of the value of k_7:k_4.

Words: 1

Cycles: 1

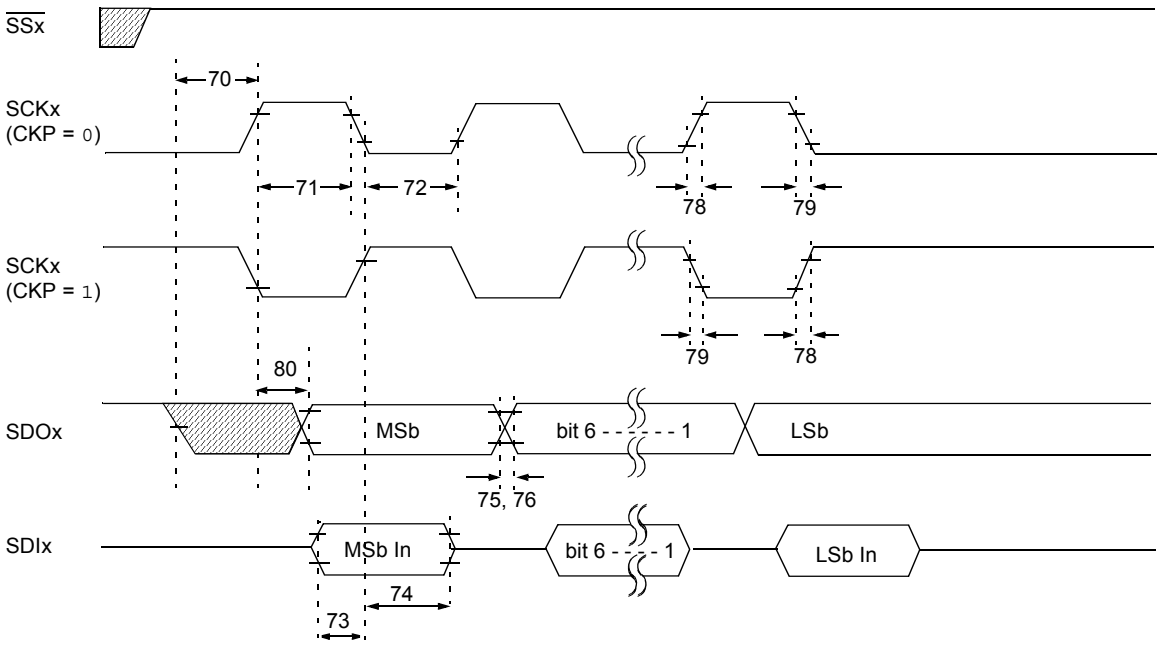
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example: MOVLB 5

Before Instruction
BSR Register = 02h
After Instruction
BSR Register = 05h

FIGURE 24-10: EXAMPLE SPI™ MASTER MODE TIMING (CKE = 0)



Note: Refer to Figure 24-3 for load conditions.

TABLE 24-14: EXAMPLE SPI™ MODE REQUIREMENTS (CKE = 0)

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓ to SCKx ↓ or SCKx ↑ Input	Tcy	—	ns	
73	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 Tcy + 40	—	ns	**(Note 1)**
74	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	40	—	ns	
75	TdoR	SDOx Data Output Rise Time	—	25	ns	
76	TdoF	SDOx Data Output Fall Time	—	25	ns	
78	TscR	SCKx Output Rise Time (Master mode)	—	25	ns	
79	TscF	SCKx Output Fall Time (Master mode)	—	25	ns	
80	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	50	ns	

Note 1: Only if Parameter #71A and #72A are used.