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	Active
Core Processor	PIC
Core Size	8-Bit
Speed	48MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K 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-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf26j11t-i-so

4.0 LOW-POWER MODES

The PIC18F46J11 family devices can manage power consumption through clocking to the CPU and the peripherals. In general, reducing the clock frequency and the amount of circuitry being clocked reduces power consumption.

For managing power in an application, the primary modes of operation are:

- Run Mode
- Idle Mode
- Sleep Mode
- Deep Sleep Mode

Additionally, there is an Ultra Low-Power Wake-up (ULPWU) mode for generating an interrupt-on-change on RA0.

These modes define which portions of the device are clocked and at what speed.

- The Run and Idle modes can use any of the three available clock sources (primary, secondary or internal oscillator blocks).
- The Sleep mode does not use a clock source.

The ULPWU mode on RA0 allows a slow falling voltage to generate an interrupt-on-change on RA0 without excess current consumption. See **Section 4.7 "Ultra Low-Power Wake-up"**.

The power-managed modes include several power-saving features offered on previous PIC® devices, such as clock switching, ULPWU and Sleep mode. In addition, the PIC18F46J11 family devices add a new power-managed Deep Sleep mode.

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires these decisions:

- Will the CPU be clocked?
- If so, which clock source will be used?

The IDLEN bit (OSCCON<7>) controls CPU clocking and the SCS<1:0> bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 4-1.

4.1.1 CLOCK SOURCES

The SCS<1:0> bits allow the selection of one of three clock sources for power-managed modes. They are:

- Primary clock source – Defined by the FOSC<2:0> Configuration bits
- Timer1 clock – Provided by the secondary oscillator
- Postscaled internal clock – Derived from the internal oscillator block

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one clock source to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source.

Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch also may be subject to clock transition delays. These delays are discussed in **Section 4.1.3 "Clock Transitions and Status Indicators"** and subsequent sections.

Entry to the power-managed Idle or Sleep modes is triggered by the execution of a SLEEP instruction. The actual mode that results depends on the status of the IDLEN bit.

Depending on the current mode and the mode being switched to, a change to a power-managed mode does not always require setting all of these bits. Many transitions may be done by changing the oscillator select bits, the IDLEN bit or the DSEN bit prior to issuing a SLEEP instruction.

If the IDLEN and DSEN bits are already configured correctly, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

TABLE 6-3: **NON-ACCESS BANK SPECIAL FUNCTION REGISTER MAP**

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
F5Fh	PMCONH[1]	F3Fh	RTCCFG	F1Fh	—	EFFh	PPSCON	EDFh	—
F5Eh	PMCONL[1]	F3Eh	RTCCAL	F1Eh	—	EFEh	RPINR24	EDEh	RPOR24[1]
F5Dh	PMMODEH[1]	F3Dh	REFOCON	F1Dh	—	EFDh	RPINR23	EDDh	RPOR23[1]
F5Ch	PMMODEL[1]	F3Ch	PADCFG1	F1Ch	—	EFCh	RPINR22	EDCh	RPOR22[1]
F5Bh	PMDOUT2H[1]	F3Bh	—	F1Bh	—	EFBh	RPINR21	EDBh	RPOR21[1]
F5Ah	PMDOUT2L[1]	F3Ah	—	F1Ah	—	EFAh	—	EDAh	RPOR20[1]
F59h	PMDIN2H[1]	F39h	—	F19h	—	EF9h	—	ED9h	RPOR19[1]
F58h	PMDIN2L[1]	F38h	—	F18h	—	EF8h	—	ED8h	RPOR18
F57h	PMEH[1]	F37h	—	F17h	—	EF7h	RPINR17	ED7h	RPOR17
F56h	PMEL[1]	F36h	—	F16h	—	EF6h	RPINR16	ED6h	RPOR16
F55h	PMSTATH[1]	F35h	—	F15h	—	EF5h	—	ED5h	RPOR15
F54h	PMSTATL[1]	F34h	—	F14h	—	EF4h	—	ED4h	RPOR14
F53h	CVRCON	F33h	—	F13h	—	EF3h	—	ED3h	RPOR13
F52h	TCLKCON	F32h	—	F12h	—	EF2h	—	ED2h	RPOR12
F51h	—	F31h	—	F11h	—	EF1h	—	ED1h	RPOR11
F50h	—	F30h	—	F10h	—	EF0h	—	ED0h	RPOR10
F4Fh	DSGPR1[2]	F2Fh	—	F0Fh	—	EEFh	—	ECFh	RPOR9
F4Eh	DSGPR0[2]	F2Eh	—	F0Eh	—	EEEh	RPINR8	ECEh	RPOR8
F4Dh	DSCONH[2]	F2Dh	—	F0Dh	—	EEDh	RPINR7	ECDh	RPOR7
F4Ch	DSCONL[2]	F2Ch	—	F0Ch	—	EECh	RPINR6	ECCh	RPOR6
F4Bh	DSWAKEH[2]	F2Bh	—	F0Bh	—	EEBh	—	ECBh	RPOR5
F4Ah	DSWAKEL[2]	F2Ah	—	F0Ah	—	EEAh	RPINR4	ECAh	RPOR4
F49h	ANCON1	F29h	—	F09h	—	EE9h	RPINR3	EC9h	RPOR3
F48h	ANCON0	F28h	—	F08h	—	EE8h	RPINR2	EC8h	RPOR2
F47h	—	F27h	—	F07h	—	EE7h	RPINR1	EC7h	RPOR1
F46h	—	F26h	—	F06h	—	EE6h	—	EC6h	RPOR0
F45h	—	F25h	—	F05h	—	EE5h	—	EC5h	—
F44h	—	F24h	—	F04h	—	EE4h	—	EC4h	—
F43h	—	F23h	—	F03h	—	EE3h	—	EC3h	—
F42h	ODCON1	F22h	—	F02h	—	EE2h	—	EC2h	—
F41h	ODCON2	F21h	—	F01h	—	EE1h	—	EC1h	—
F40h	ODCON3	F20h	—	F00h	—	EE0h	—	EC0h	—

Note 1: This register is not available on 28-pin devices.
    2: Deep Sleep registers are not available on LF devices.

7.5.3 WRITE VERIFY

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.5.4 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. If the write operation is interrupted by a $\overline{\text{MCLR}}$ Reset or a WDT time-out Reset during normal operation, the user can check the WRERR bit and rewrite the location(s) as needed.

7.6 Flash Program Operation During Code Protection

See **Section 26.6 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

TABLE 7-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					69
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								69
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								69
TABLAT	Program Memory Table Latch								69
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
EECON2	Program Memory Control Register 2 (not a physical register)								71
EECON1	—	—	WPROG	FREE	WRERR	WREN	WR	—	71

Legend: — = unimplemented, read as '0'. Shaded cells are not used during Flash program memory access.

REGISTER 9-6: **PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3 (ACCESS FA4h)**

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF
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   **SSP2IF:** Master Synchronous Serial Port 2 Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 6   **BCL2IF:** Bus Collision Interrupt Flag bit (MSSP2 module)
1 = A bus collision occurred (must be cleared in software)
0 = No bus collision occurred

bit 5   **RC2IF:** EUSART2 Receive Interrupt Flag bit
1 = The EUSART2 receive buffer, RCREG2, is full (cleared when RCREG2 is read)
0 = The EUSART2 receive buffer is empty

bit 4   **TX2IF:** EUSART2 Transmit Interrupt Flag bit
1 = The EUSART2 transmit buffer, TXREG2, is empty (cleared when TXREG2 is written)
0 = The EUSART2 transmit buffer is full

bit 3   **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = TMR4 to PR4 match occurred (must be cleared in software)
0 = No TMR4 to PR4 match occurred

bit 2   **CTMUIF:** Charge Time Measurement Unit Interrupt Flag bit
1 = A CTMU event has occurred (must be cleared in software)
0 = CTMU event has not occurred

bit 1   **TMR3GIF:** Timer3 Gate Event Interrupt Flag bit
1 = A Timer3 gate event completed (must be cleared in software)
0 = No Timer3 gate event completed

bit 0   **RTCCIF:** RTCC Interrupt Flag bit
1 = RTCC interrupt occurred (must be cleared in software)
0 = No RTCC interrupt occurred

11.2.5 ADDRESSABLE PARALLEL SLAVE PORT MODE

In the Addressable Parallel Slave Port mode (PMMODEH<1:0> = `01`), the module is configured with two extra inputs, PMA<1:0>, which are the address lines 1 and 0. This makes the 4-byte buffer space directly addressable as fixed pairs of read and write buffers. As with Legacy Buffered mode, data is output from PMDOUT1L, PMDOUT1H, PMDOUT2L and PMDOUT2H, and is read in on PMDIN1L, PMDIN1H, PMDIN2L and PMDIN2H. Table 11-1 provides the buffer addressing for the incoming address to the input and output registers.

TABLE 11-1: SLAVE MODE BUFFER ADDRESSING

PMA<1:0>	Output Register (Buffer)	Input Register (Buffer)
00	PMDOUT1L (0)	PMDIN1L (0)
01	PMDOUT1H (1)	PMDIN1H (1)
10	PMDOUT2L (2)	PMDIN2L (2)
11	PMDOUT2H((3)	PMDIN2H (3)

**FIGURE 11-6: PARALLEL MASTER/SLAVE CONNECTION ADDRESSED BUFFER EXAMPLE**
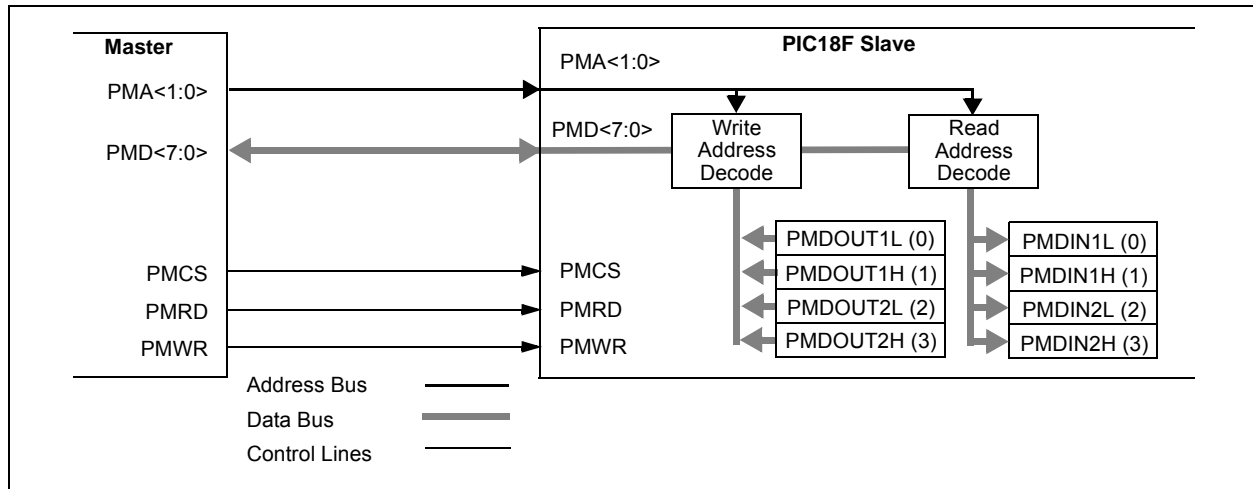
11.2.5.1 READ FROM SLAVE PORT

When chip select is active and a read strobe occurs (PMCS = `1` and PMRD = `1`), the data from one of the four output bytes is presented onto PMD<7:0>. Which byte is read depends on the 2-bit address placed on ADDR<1:0>. Table 11-1 provides the corresponding output registers and their associated address. When an output buffer is read, the corresponding OBxE bit is set. The OBxE flag bit is set when all the buffers are empty. If any buffer is already empty, OBxE = `1`, the next read to that buffer will generate an OBUF event.

**FIGURE 11-7: PARALLEL SLAVE PORT READ WAVEFORMS**
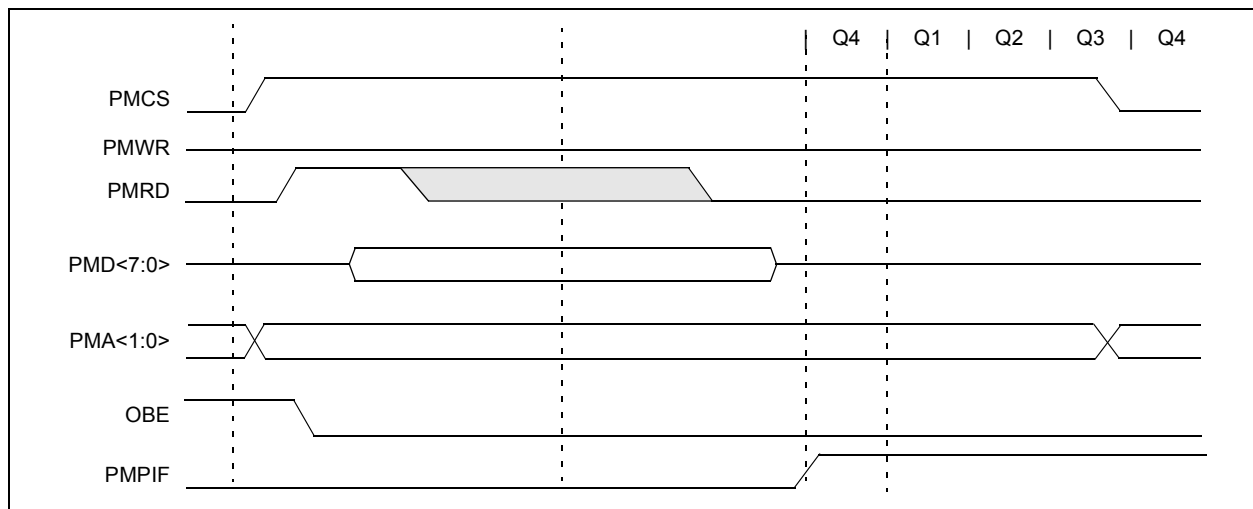
11.4   Application Examples

This section introduces some potential applications for the PMP module.

11.4.1   MULTIPLEXED MEMORY OR PERIPHERAL

Figure 11-27 demonstrates the hookup of a memory or another addressable peripheral in Full Multiplex mode. Consequently, this mode achieves the best pin saving from the microcontroller perspective. However, for this configuration, there needs to be some external latches to maintain the address.

FIGURE 11-27:   EXAMPLE – MULTIPLEXED ADDRESSING APPLICATION
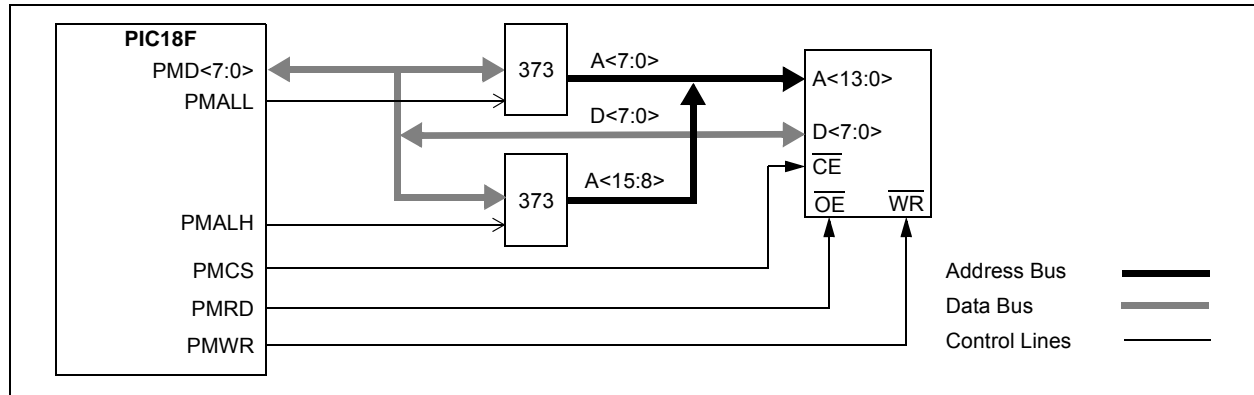
11.4.2   PARTIALLY MULTIPLEXED MEMORY OR PERIPHERAL

Partial multiplexing implies using more pins; however, for a few extra pins, some extra performance can be achieved. Figure 11-28 provides an example of a memory or peripheral that is partially multiplexed with an external latch. If the peripheral has internal latches, as displayed in Figure 11-29, then no extra circuitry is required except for the peripheral itself.

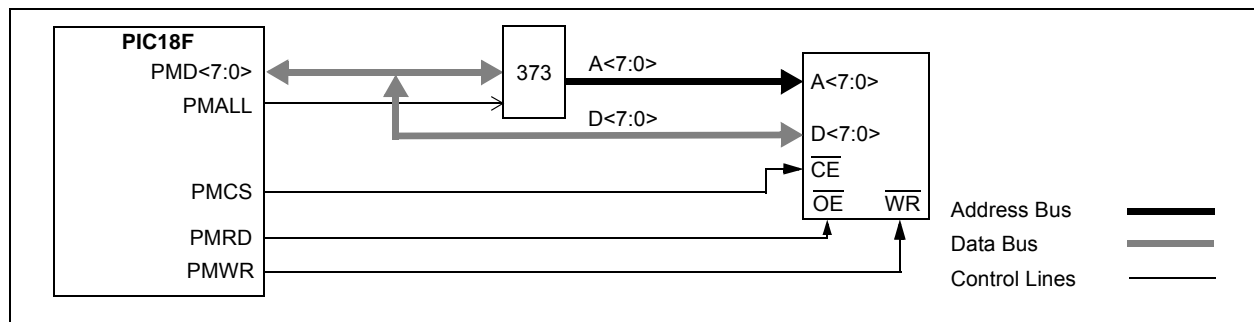**FIGURE 11-28:   EXAMPLE OF A PARTIALLY MULTIPLEXED ADDRESSING APPLICATION**
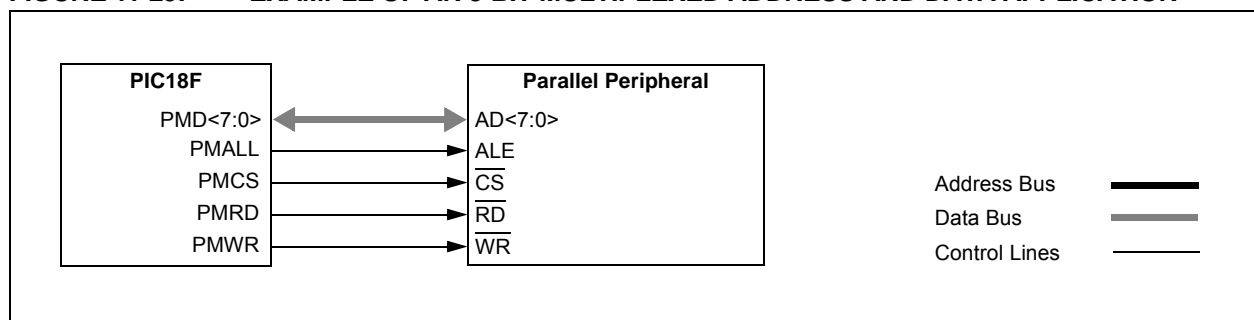

FIGURE 11-29:   EXAMPLE OF AN 8-BIT MULTIPLEXED ADDRESS AND DATA APPLICATION

REGISTER 17-16: ALRMWD: ALARM WEEKDAY VALUE REGISTER (ACCESS F8Fh, PTR 01b)[1]

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
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 7-3   **Unimplemented:** Read as '0'

bit 2-0   **WDAY<2:0>:** Binary Coded Decimal Value of Weekday Digit bits
                  Contains a value from 0 to 6.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-17: ALRMHR: ALARM HOURS VALUE REGISTER (ACCESS F8Eh, PTR 01b)[1]

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
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   **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits
                  Contains a value from 0 to 2.

bit 3-0   **HRONE3:HRONE0:** Binary Coded Decimal Value of Hour's Ones Digit bits
                  Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

PIC18F46J11 FAMILY

TABLE 18-3: REGISTERS ASSOCIATED WITH PWM, TIMER2 AND TIMER4

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
PIR1	PMPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	72
PIE1	PMPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	72
IPR1	PMPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	72
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	71
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	71
TCLKCON	—	—	—	T1RUN	—	—	T3CCP2	T3CCP1	74
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	72
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	72
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	72
TMR2	Timer2 Register								70
PR2	Timer2 Period Register								70
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	70
TMR4	Timer4 Register								73
PR4	Timer4 Period Register								73
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	73
ODCON1	—	—	—	—	—	—	ECCP2OD	ECCP1OD	74

Legend: — = unimplemented, read as '0'. Shaded cells are not used by PWM, Timer2 or Timer4.

18.5.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCPxAS<2:0> bits of the ECCPAS register. A shutdown event may be generated by:

- A logic '0' on the pin that is assigned the FLT0 input function
- Comparator C1
- Comparator C2
- Setting the ECCPxASE bit in firmware

A shutdown condition is indicated by the ECCPxASE (Auto-Shutdown Event Status) bit of the ECCPxAS register. If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

When a shutdown event occurs, two things happen:

The ECCPxASE bit is set to '1'. The ECCPxASE will remain set until cleared in firmware or an auto-restart occurs (see **Section 18.5.5 "Auto-Restart Mode"**).

The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs [PxA/PxC] and [PxB/PxD]. The state of each pin pair is determined by the PSSxAC and PSSxBD bits of the ECCPxAS register. Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

REGISTER 18-2: ECCPxAS: ECCPx AUTO-SHUTDOWN CONTROL REGISTER (ACCESS FBEh/FB8h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
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 **ECCPxASE:** ECCP Auto-Shutdown Event Status bit
1 = A shutdown event has occurred; ECCP outputs are in a shutdown state
0 = ECCP outputs are operating

bit 6-4 **ECCPxAS<2:0>:** ECCP Auto-Shutdown Source Select bits
000 = Auto-shutdown is disabled
001 = Comparator C1OUT output is high
010 = Comparator C2OUT output is high
011 = Either Comparator C1OUT or C2OUT is high
100 = V_{IL} on FLT0 pin
101 = V_{IL} on FLT0 pin or Comparator C1OUT output is high
110 = V_{IL} on FLT0 pin or Comparator C2OUT output is high
111 = V_{IL} on FLT0 pin or Comparator C1OUT or Comparator C2OUT is high

bit 3-2 **PSSxAC<1:0>:** Pins PxA and PxC Shutdown State Control bits
00 = Drive pins PxA and PxC to '0'
01 = Drive pins PxA and PxC to '1'
10 = Pins PxA and PxC tri-state

bit 1-0 **PSSxBD<1:0>:** Pins PxB and PxD Shutdown State Control bits
00 = Drive pins PxB and PxD to '0'
01 = Drive pins PxB and PxD to '1'
10 = Pins PxB and PxD tri-state

Note 1: The auto-shutdown condition is a level-based signal, not an edge-based signal. As long as the level is present, the auto-shutdown will persist.
2: Writing to the ECCPxASE bit is disabled while an auto-shutdown condition persists.
3: Once the auto-shutdown condition has been removed and the PWM restarted (either through firmware or auto-restart), the PWM signal will always restart at the beginning of the next PWM period.

19.3.7 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device can be configured to wake-up from Sleep.

19.3.8 SLAVE SELECT SYNCHRONIZATION

The $\overline{SSx}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with the $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 04h). When the $\overline{SSx}$ pin is low, transmission and reception are enabled and the SDOx pin is driven. When the $\overline{SSx}$ pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note 1: When the SPI is in Slave mode with the $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SSx}$ pin is set to VDD.

2: If the SPI is used in Slave mode with CKE set, then the $\overline{SSx}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SSx}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDOx pin can be connected to the SDIx pin. When the SPI needs to operate as a receiver, the SDOx pin can be configured as an input. This disables transmissions from the SDOx. The SDIx can always be left as an input (SDIx function) since it cannot create a bus conflict.

FIGURE 19-4: **SLAVE SYNCHRONIZATION WAVEFORM**
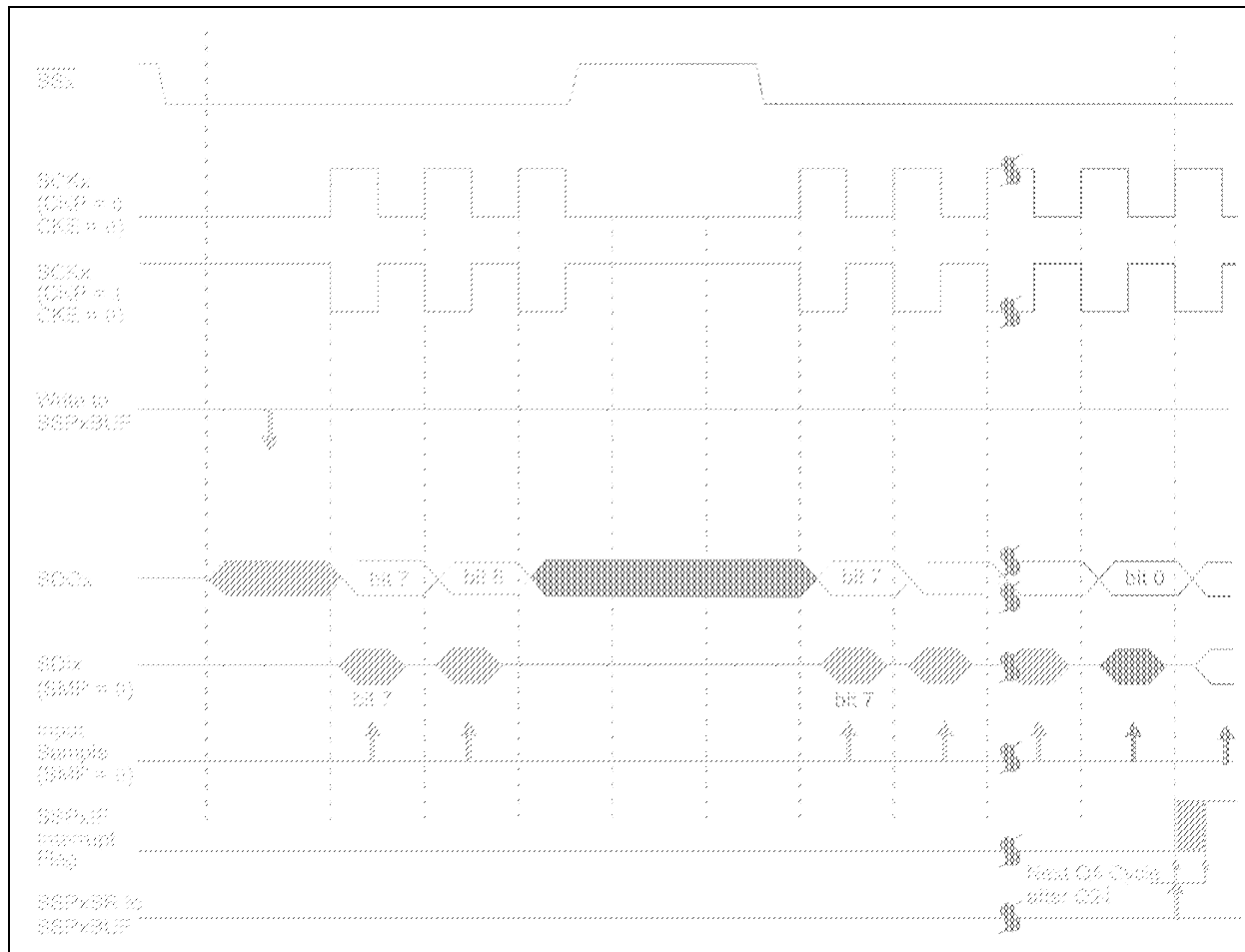
FIGURE 19-12: **I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)**
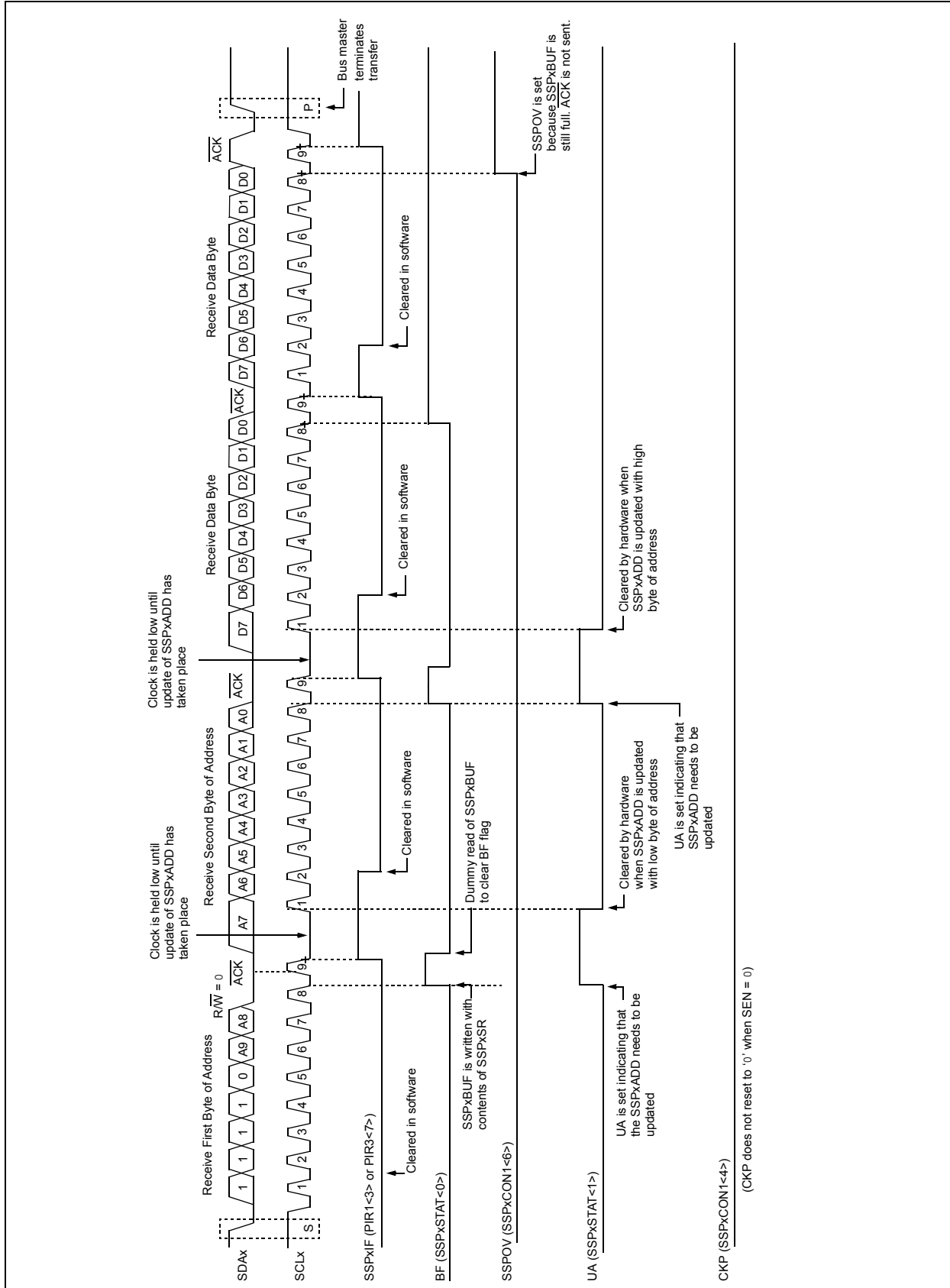
TABLE 20-3:   BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	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	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	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	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0											
	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	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9615.	-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 = 1, BRG16 = 0								
	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	-0.16	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	—	—	—	—	—	—

20.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

20.4.1 EUSART SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep mode.

If two words are written to the TXREGx and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.

b) The second word will remain in the TXREGx register.

c) Flag bit, TXxIF, will not be set.

d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit, TXxIF, will now be set.

e) If enable bit, TXxIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.

2. Clear bits, CREN and SREN.

3. If interrupts are desired, set enable bit, TXxIE.

4. If 9-bit transmission is desired, set bit, TX9.

5. Enable the transmission by setting enable bit, TXEN.

6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.

7. Start transmission by loading data to the TXREGx register.

8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

REGISTER 25-2: CTMUCONL: CTMU CONTROL REGISTER LOW (ACCESS FB2h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x	R/W-x
EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
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 **EDG2POL:** Edge 2 Polarity Select bit

1 = Edge 2 programmed for a positive edge response
0 = Edge 2 programmed for a negative edge response

bit 6-5 **EDG2SEL<1:0>:** Edge 2 Source Select bits

11 = CTED1 pin
10 = CTED2 pin
01 = ECCP1 Special Event Trigger
00 = ECCP2 Special Event Trigger

bit 4 **EDG1POL:** Edge 1 Polarity Select bit

1 = Edge 1 programmed for a positive edge response
0 = Edge 1 programmed for a negative edge response

bit 3-2 **EDG1SEL<1:0>:** Edge 1 Source Select bits

11 = CTED1 pin
10 = CTED2 pin
01 = ECCP1 Special Event Trigger
00 = ECCP2 Special Event Trigger

bit 1 **EDG2STAT:** Edge 2 Status bit

1 = Edge 2 event has occurred
0 = Edge 2 event has not occurred

bit 0 **EDG1STAT:** Edge 1 Status bit

1 = Edge 1 event has occurred
0 = Edge 1 event has not occurred

FIGURE 26-4: **FSCM BLOCK DIAGRAM**
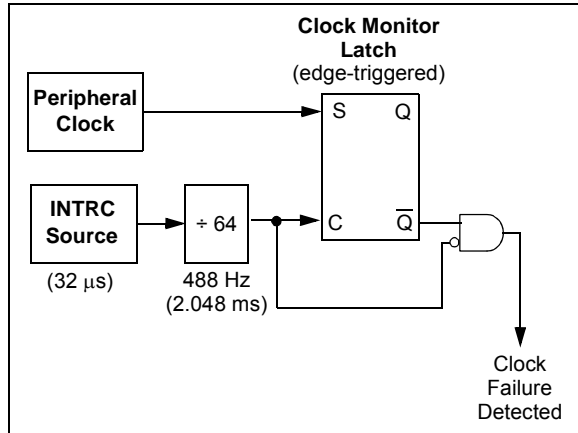
Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while the clock monitor is still set, and a clock failure has been detected (Figure 26-5), the following results:

• The FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>);

• The device clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the Fail-safe condition); and

• The WDT is reset.

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing-sensitive applications. In these cases, it may

be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 4.1.4 "Multiple Sleep Commands"** and **Section 26.4.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

26.5.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTRC clock when a clock failure is detected; this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

FIGURE 26-5: **FSCM TIMING DIAGRAM**
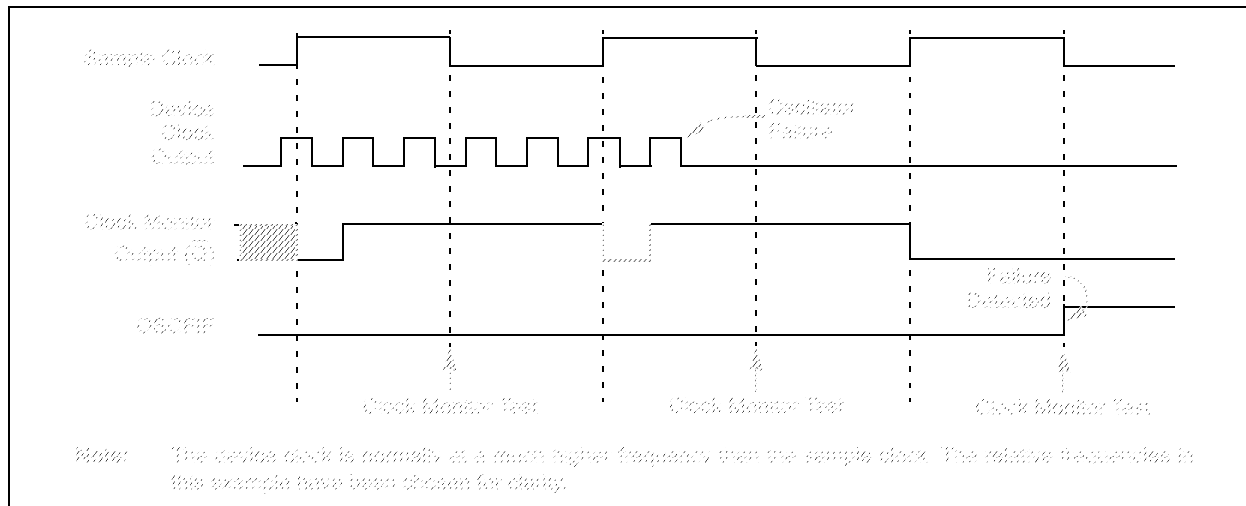
26.7 In-Circuit Serial Programming (ICSP)

PIC18F46J11 family microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

26.8 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ Configuration bit is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use.

Table 26-4 lists the resources required by the background debugger.

TABLE 26-4: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	TOSx registers reserved

BCF	Bit Clear f
Syntax:	BCF f, b {,a}
Operands:	$0 \leq f \leq 255$ $0 \leq b \leq 7$ $a \in [0,1]$
Operation:	$0 \rightarrow f<b>$
Status Affected:	None
Encoding:	`1001` `bbba` `ffff` `ffff`
Description:	Bit 'b' in register 'f' is cleared. 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 $f \leq 95$ (5Fh). See **Section 27.2.3 "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 register 'f'

<u>Example:</u>   BCF   FLAG_REG, 7, 0

Before Instruction
    FLAG_REG = C7h
After Instruction
    FLAG_REG = 47h

BN	Branch if Negative
Syntax:	BN n
Operands:	$-128 \leq n \leq 127$
Operation:	if Negative bit is '1', $(PC) + 2 + 2n \rightarrow PC$
Status Affected:	None
Encoding:	`1110` `0110` `nnnn` `nnnn`
Description:	If the Negative bit is '1', then the program will branch. The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

<u>Example:</u>   HERE   BN   Jump

Before Instruction
    PC   =   address (HERE)
After Instruction
    If Negative   =   1;
        PC   =   address (Jump)
    If Negative   =   0;
        PC   =   address (HERE + 2)

BTG	Bit Toggle f

Syntax:	BTG f, b {,a}
Operands:	$0 \leq f \leq 255$ $0 \leq b < 7$ $a \in [0,1]$
Operation:	$(\overline{f<b>}) \rightarrow f<b>$
Status Affected:	None

Encoding:

0111	bbba	ffff	ffff

Description:

Bit 'b' in data memory location 'f' is inverted.

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 27.2.3 "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 register 'f'

Example: BTG   LATC,   4, 0

Before Instruction:
    LATC   =   0111 0101 [75h]
After Instruction:
    LATC   =   0110 0101 [65h]

BOV	Branch if Overflow

Syntax:	BOV n
Operands:	$-128 \leq n \leq 127$
Operation:	if Overflow bit is '1', $(PC) + 2 + 2n \rightarrow PC$
Status Affected:	None

Encoding:

1110	0100	nnnn	nnnn

Description:

If the Overflow bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE   BOV Jump

Before Instruction
    PC   =   address (HERE)
After Instruction
    If Overflow =   1;
        PC   =   address (Jump)
    If Overflow =   0;
        PC   =   address (HERE + 2)

COMF	Complement f
Syntax:	COMF f {,d {,a}}
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	f̄ → dest
Status Affected:	N, Z
Encoding:	0001	11da	ffff	ffff

Description:	The contents of register 'f' are complemented. 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 27.2.3 "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: COMF   REG, 0, 0

    Before Instruction
        REG   =   13h
    After Instruction
        REG   =   13h
        W   =   ECh

CPFSEQ	Compare f with W, Skip if f = W
Syntax:	CPFSEQ f {,a}
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(f) – (W), skip if (f) = (W) (unsigned comparison)
Status Affected:	None
Encoding:	0110	001a	ffff	ffff

Description:	Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
	If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.
	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 27.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:	1
Cycles:	1(2)
	Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example: HERE   CPFSEQ REG, 0
                   NEQUAL   :
                   EQUAL   :

    Before Instruction
        PC Address   =   HERE
        W   =   ?
        REG   =   ?
    After Instruction
        If REG   =   W;
            PC   =   Address (EQUAL)
        If REG   ≠   W;
            PC   =   Address (NEQUAL)

TBLRD	Table Read

Syntax: TBLRD (*; *+; *-; +*)

Operands: None

Operation:
if TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT,
TBLPTR – No Change;
if TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT,
(TBLPTR) + 1 → TBLPTR;
if TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT,
(TBLPTR) – 1 → TBLPTR;
if TBLRD +*,
(TBLPTR) + 1 → TBLPTR,
(Prog Mem (TBLPTR)) → TABLAT

Status Affected: None

Encoding:

0000	0000	0000	10nn nn=0 * =1 *+ =2 *– =3 +*

Description:
This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used.

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR<0> = 0: Least Significant Byte of Program Memory Word

TBLPTR<0> = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:
- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read Program Memory)	No operation	No operation (Write TABLAT)

TBLRD	Table Read (Continued)

Example 1: TBLRD *+

Before Instruction

TABLAT	=	55h
TBLPTR	=	00A356h
MEMORY(00A356h)	=	34h

After Instruction

TABLAT	=	34h
TBLPTR	=	00A357h

Example 2: TBLRD +*

Before Instruction

TABLAT	=	AAh
TBLPTR	=	01A357h
MEMORY(01A357h)	=	12h
MEMORY(01A358h)	=	34h

After Instruction

TABLAT	=	34h
TBLPTR	=	01A358h