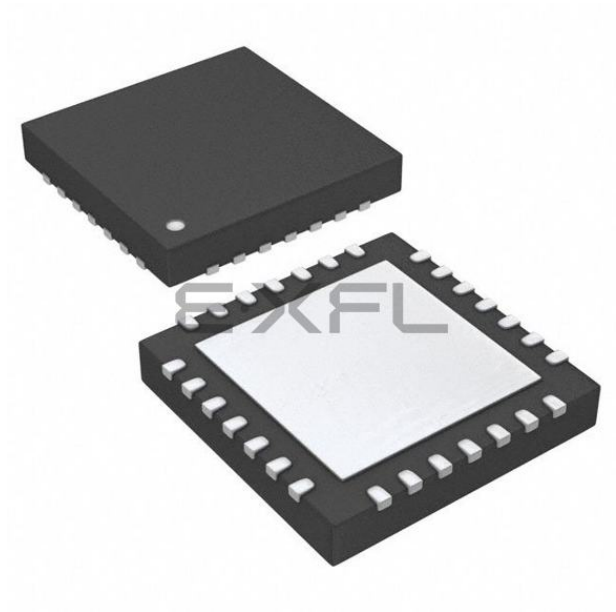


Welcome to E-XFL.COM

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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25k42-e-mv

REGISTER 8-3: PCON1: POWER CONTROL REGISTER 1

U-0	U-0	U-0	U-0	U-0	U-0	R/W/HC-1/u	U-0
—	—	—	—	—	—	$\overline{\text{MEMV}}$	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-m/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	q = Value depends on condition

bit 7-2 **Unimplemented:** Read as '0'

bit 1 $\overline{\textbf{MEMV}}$**:** Memory Violation Flag bit
- 1 = No memory violation Reset occurred or set to '1' by firmware
- 0 = A memory violation Reset occurred (set to '0' in hardware when a Memory Violation occurs)

bit 0 **Unimplemented:** Read as '0'

TABLE 8-4: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BORCON	SBOREN	—	—	—	—	—	—	BORRDY	90
PCON0	STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	95
PCON1	—	—	—	—	—	—	$\overline{\text{MEMV}}$	—	96

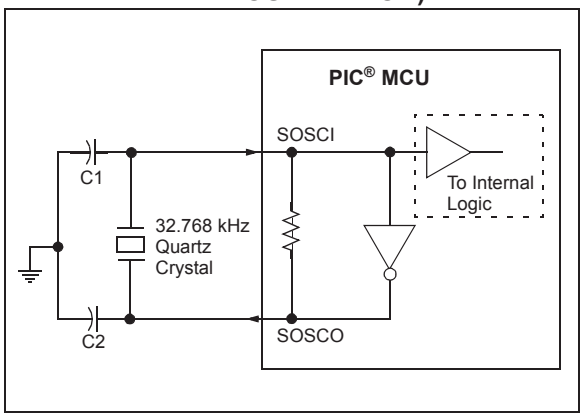
Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Resets.

9.2.1.5 Secondary Oscillator

The secondary oscillator is a separate oscillator block that can be used as an alternate system clock source. The secondary oscillator is optimized for 32.768 kHz, and can be used with an external crystal oscillator connected to the SOSCI and SOSCO device pins, or an external clock source connected to the SOSCIN pin. The secondary oscillator can be selected during run-time using clock switching. Refer to **Section 9.3 "Clock Switching"** for more information.

Two power modes are available for the secondary oscillator. These modes are selected with the SOSCPWR (OSCCON3<6>). Clearing this bit selects the lower Crystal Gain mode which provides lowest microcontroller power consumption. Setting this bit enables a higher Gain mode to support faster crystal start-up or crystals with higher ESR.

FIGURE 9-5: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)



Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

2: Always verify oscillator performance over the V_{DD} and temperature range that is expected for the application.

3: For oscillator design assistance, reference the following Microchip Application Notes:

- AN826, "*Crystal Oscillator Basics and Crystal Selection for PIC® and PIC® Devices*" (DS00826)
- AN849, "*Basic PIC® Oscillator Design*" (DS00849)
- AN943, "*Practical PIC® Oscillator Analysis and Design*" (DS00943)
- AN949, "*Making Your Oscillator Work*" (DS00949)
- TB097, "*Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS*" (DS91097)
- AN1288, "*Design Practices for Low-Power External Oscillators*" (DS01288)

9.5 Register Definitions: Oscillator Control

REGISTER 9-1: OSCCON1: OSCILLATOR CONTROL REGISTER1

U-0	R/W-f/f	R/W-f/f	R/W-f/f	R/W-q/q	R/W-q/q	R/W-q/q	R/W-q/q
—	NOSC<2:0>			NDIV<3:0>			
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	f = determined by Configuration bit setting
		q = Reset value is determined by hardware

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

bit 6-4 **NOSC<2:0>:** New Oscillator Source Request bits(1,2,3)
The setting requests a source oscillator and PLL combination per Table 9-1.
POR value = RSTOSC (Register 5-1).

bit 3-0 **NDIV<3:0>:** New Divider Selection Request bits(2,3)
The setting determines the new postscaler division ratio per Table 9-1.

Note 1: The default value (f/f) is determined by the RSTOSC Configuration bits. See Table 9-2below.
2: If NOSC is written with a reserved value (Table 9-1), the operation is ignored and neither NOSC nor NDIV is written.
3: When CSWEN = 0, this register is read-only and cannot be changed from the POR value.

TABLE 9-2: DEFAULT OSCILLATOR SETTINGS

RSTOSC	SFR Reset Values			Initial Fosc Frequency
	NOSC/COSC	CDIV	OSCFRQ	
111	111	1:1	4 MHz	EXTOSC per FEXTOSC
110	110	4:1		FOSC = 1 MHz (4 MHz/4)
101	101	1:1		LFINTOSC
100	100	1:1		SOSC
011	Reserved			
010	010	1:1	4 MHz	EXTOSC + 4xPLL (1)
001	Reserved			
000	110	1:1	64 MHz	FOSC = 64 MHz

Note 1: EXTOSC must meet the PLL specifications (Table 46-9).

REGISTER 11-5: PIR2: PERIPHERAL INTERRUPT REGISTER 2

R-0/0	R-0/0	R-0/0	R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
I2C1RXIF	SPI1IF	SPI1TXIF	SPI1RXIF	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HS = Hardware set

bit 7 **I2C1RXIF:** I^2C1 Receive Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 6 **SPI1IF:** SPI1 Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 5 **SPI1TXIF:** SPI1 Transmit Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 4 **SPI1RXIF:** SPI1 Receive Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 3 **DMA1AIF:** DMA1 Abort Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 2 **DMA1ORIF:** DMA1 Overrun Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 1 **DMA1DCNTIF:** DMA1 Destination Count Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 0 **DMA1SCNTIF:** DMA1 Source Count Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 11-11: PIR8: PERIPHERAL INTERRUPT REGISTER 8

R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	U-0	U-0	U-0
TMR5GIF	TMR5IF	—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HS = Bit is set in hardware

bit 7 **TMR5GIF:** TMR5 Gate Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 6 **TMR5IF:** TMR5 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 5-0 **Unimplemented:** Read as '0'

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 11-12: PIR9: PERIPHERAL INTERRUPT REGISTER 9

U-0	U-0	U-0	U-0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
—	—	—	—	CLC3IF	CWG3IF	CCP3IF	TMR6IF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **CLC3IF:** CLC3 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 2 **CWG3IF:** CWG3 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 1 **CCP3IF:** CCP3 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 0 **TMR6IF:** TMR6 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

TABLE 23-4: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1/3/5 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TxCON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	316
TxGCON	GE	GPOL	GTM	GSPM	GO/$\overline{\text{DONE}}$	GVAL	—	—	317
TxCLK	—	—	—	CS<4:0>					318
TxGATE	—	—	—	GSS<4:0>					319
TMRxL	Least Significant Byte of the 16-bit TMR3 Register								320
TMRxH	Holding Register for the Most Significant Byte of the 16-bit TMR3 Register								320

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by TIMER1/3/5.

24.5 Operation Examples

Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the T2CON register are cleared).
- The diagrams illustrate any clock except FOSC/4 and show clock-sync delays of at least two full cycles for both ON and T2TMR_ers. When using FOSC/4, the clock-sync delay is at least one instruction period for T2TMR_ers; ON applies in the next instruction period.
- ON and T2TMR_ers are somewhat generalized, and clock-sync delays may produce results that are slightly different than illustrated.
- The PWM Duty Cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module as described in **Section 25.0 “Capture/Compare/PWM Module”** and **Section 26.0 “Pulse-Width Modulation (PWM)”**. The signals are not a part of the T2TMR module.

24.5.7 EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE

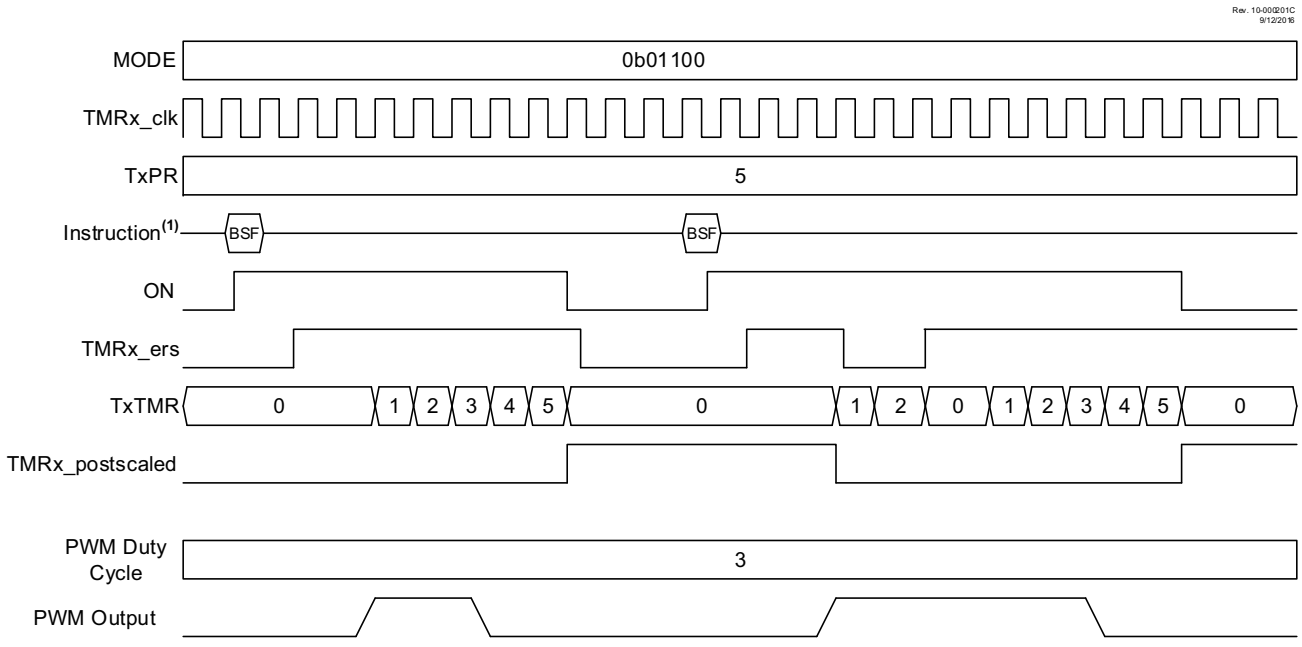
In Edge-Triggered Hardware Limit One-Shot modes the timer starts on the first external signal edge after the ON bit is set and resets on all subsequent edges. Only the first edge after the ON bit is set is needed to start the timer. The counter will resume counting automatically two clocks after all subsequent external Reset edges. Edge triggers are as follows:

- Rising edge Start and Reset (MODE<4:0> = 01100)
- Falling edge Start and Reset (MODE<4:0> = 01101)

The timer resets and clears the ON bit when the timer value matches the T2PR period value. External signal edges will have no effect until after software sets the ON bit. Figure 24-10 illustrates the rising edge hardware limit one-shot operation.

When this mode is used in conjunction with the CCP then the first starting edge trigger, and all subsequent Reset edges, will activate the PWM drive. The PWM drive will deactivate when the timer matches the CCPRx pulse width value and stay deactivated until the timer halts at the T2PR period match unless an external signal edge resets the timer before the match occurs.

FIGURE 24-10: EDGE TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01100))



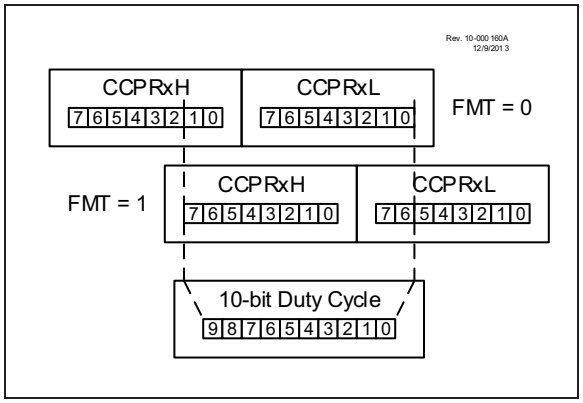
Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

25.4.5 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the CCPRxH:CCPRxL register pair. The alignment of the 10-bit value is determined by the FMT bit of the CCPxCON register (see Figure 25-5). The CCPRxH:CCPRxL register pair can be written to at any time; however the duty cycle value is not latched into the 10-bit buffer until after a match between T2PR and T2TMR.

Equation 25-2 is used to calculate the PWM pulse width. Equation 25-3 is used to calculate the PWM duty cycle ratio.

FIGURE 25-5: PWM 10-BIT ALIGNMENT



EQUATION 25-2: PULSE WIDTH

$$Pulse\ Width = (CCPRxH{:}CCPRxL\ register\ pair) \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

EQUATION 25-3: DUTY CYCLE RATIO

$$Duty\ Cycle\ Ratio = \frac{(CCPRxH{:}CCPRxL\ register\ pair)}{4(T2PR + 1)}$$

CCPRxH:CCPRxL register pair are used to double buffer the PWM duty cycle. This double buffering provides glitchless PWM operation.

The 8-bit timer T2TMR register is concatenated with either the 2-bit internal system clock (FOSC), or two bits of the prescaler, to create the 10-bit time base. The system clock is used if the Timer2 prescaler is set to 1:1.

When the 10-bit time base matches the CCPRxH:CCPRxL register pair, then the CCPx pin is cleared (see Figure 25-4).

25.4.6 PWM RESOLUTION

The resolution determines the number of available duty cycles for a given period. For example, a 10-bit resolution will result in 1024 discrete duty cycles, whereas an 8-bit resolution will result in 256 discrete duty cycles.

The maximum PWM resolution is ten bits when T2PR is 255. The resolution is a function of the T2PR register value as shown by Equation 25-4.

EQUATION 25-4: PWM RESOLUTION

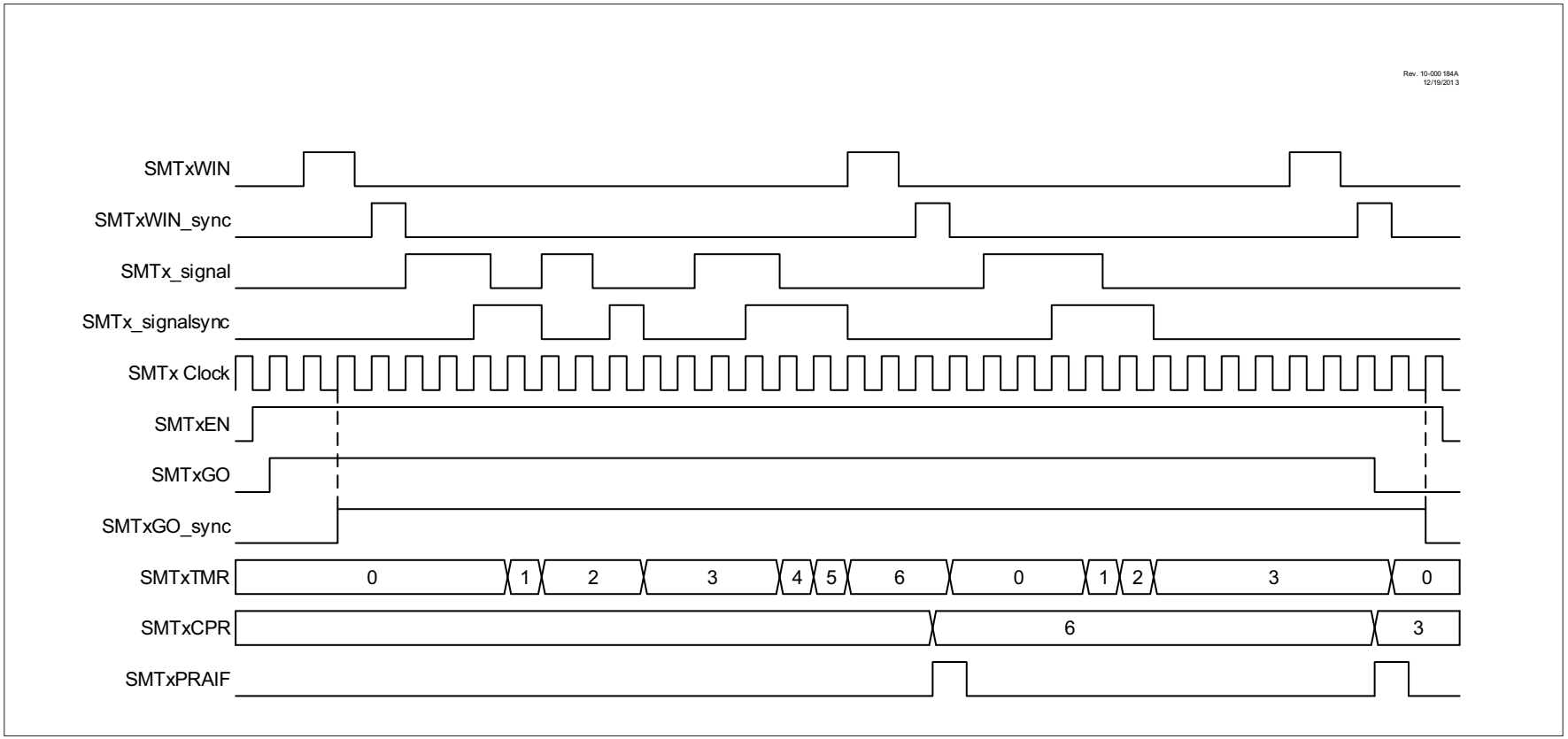
$$Resolution = \frac{log[4(T2PR + 1)]}{log(2)}\ bits$$

Note: If the pulse-width value is greater than the period, the assigned PWM pin(s) will remain unchanged.

27.6.3 PERIOD AND DUTY-CYCLE MODE

In Duty-Cycle mode, either the duty cycle or period (depending on polarity) of the SMT1_signal can be acquired relative to the SMT clock. The CPW register is updated on a falling edge of the signal, and the CPR register is updated on a rising edge of the signal, along with the SMT1TMR resetting to 0x0001. In addition, the GO bit is reset on a rising edge when the SMT is in Single Acquisition mode. See Figure 27-6 and Figure 27-7.

FIGURE 27-12: GATED WINDOWED MEASURE MODE REPEAT ACQUISITION TIMING DIAGRAM



33.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART) WITH PROTOCOL SUPPORT

The Universal Asynchronous Receiver Transmitter (UART) module is a serial I/O communications peripheral. It contains all the clock generators, shift registers and data buffers necessary to perform an input or output serial data transfer, independent of device program execution. The UART, also known as a Serial Communications Interface (SCI), can be configured as a full-duplex asynchronous system or one of several automated protocols. Full-Duplex mode is useful for communications with peripheral systems, such as CRT terminals and personal computers.

Supported protocols include:

- LIN Master and Slave
- DMX mode
- DALI control gear and control device

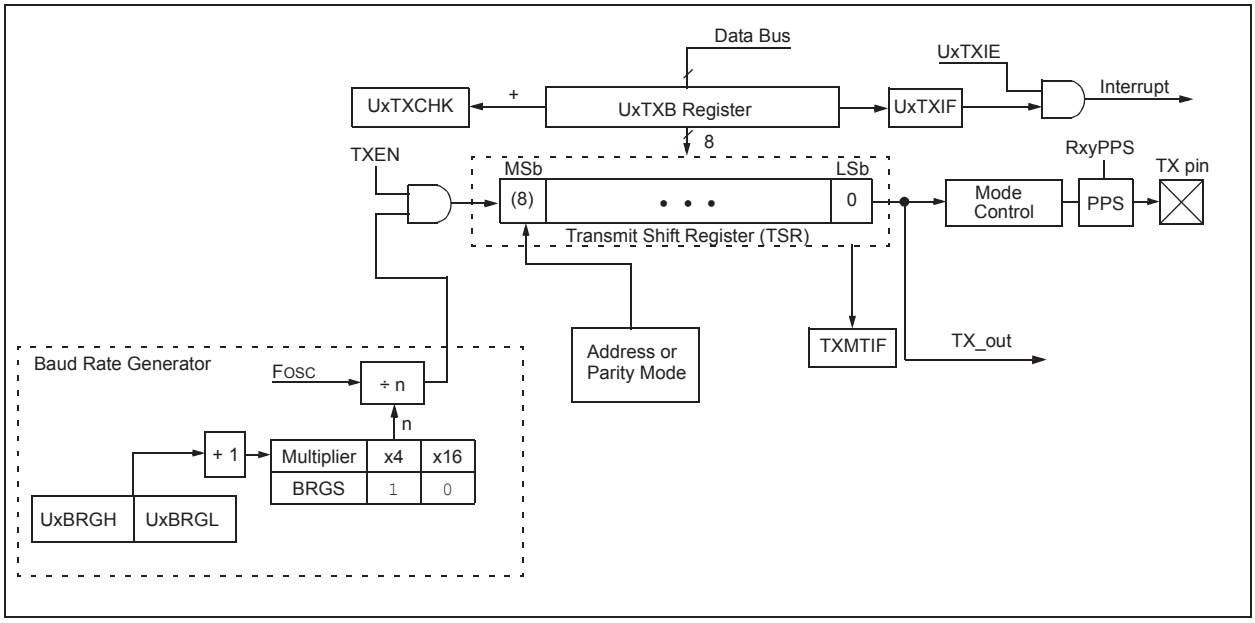
The UART module includes the following capabilities:

- Full-duplex asynchronous transmit and receive
- Two-character input buffer
- One-character output buffer
- Programmable 7-bit or 8-bit character length
- 9th bit Address detection
- 9th bit even or odd parity
- Input buffer overrun error detection
- Received character framing error detection
- Hardware and software flow control
- Automatic checksums
- Programmable 1, 1.5, and 2 Stop bits
- Programmable data polarity
- Manchester encoder/decoder
- Operation in Sleep
- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- Automatic and user timed Break period generation
- RX and TX inactivity timeouts (with Timer2)

Block diagrams of the UART transmitter and receiver are shown in Figure 33-1 and Figure 33-2.

The UART transmit output (TX_out) is available to the TX pin and internally to various peripherals.

FIGURE 33-1: UART TRANSMIT BLOCK DIAGRAM



36.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of VDD, with 1.024V, 2.048V or 4.096V selectable output levels. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel
- ADC positive reference
- Comparator input
- Digital-to-Analog Converter (DAC)

The FVR can be enabled by setting the EN bit of the FVRCON register.

Note:	Fixed Voltage Reference output cannot exceed VDD.

36.1 Independent Gain Amplifiers

The output of the FVR, which is connected to the ADC, Comparators, and DAC, is routed through two independent programmable gain amplifiers. Each amplifier can be programmed for a gain of 1x, 2x or 4x, to produce the three possible voltage levels.

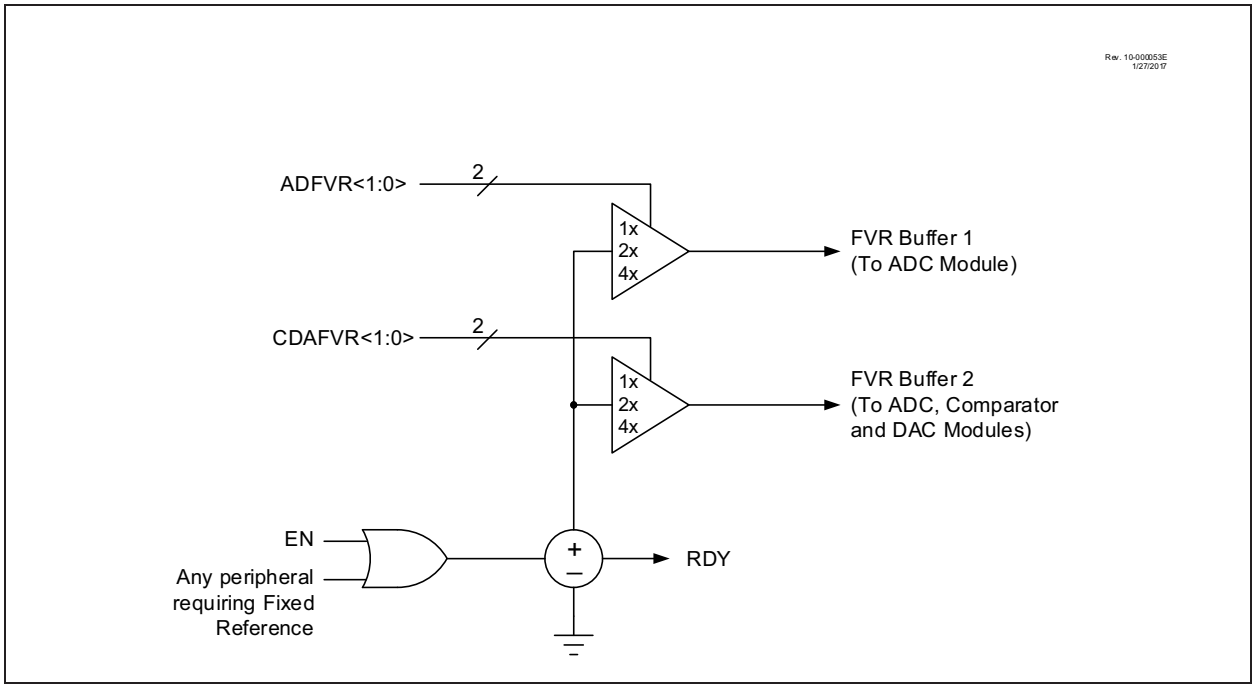
The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference **Section 38.0 "Analog-to-Digital Converter with Computation (ADC2) Module"** for additional information.

The CDAFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the DAC and comparator module. Reference **Section 39.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** and **Section 40.0 "Comparator Module"** for additional information.

36.2 FVR Stabilization Period

When the Fixed Voltage Reference module is enabled, it requires time for the reference and amplifier circuits to stabilize. Once the circuits stabilize and are ready for use, the RDY bit of the FVRCON register will be set.

FIGURE 36-1: VOLTAGE REFERENCE BLOCK DIAGRAM



38.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIR1 register. The ADC Interrupt Enable is the ADIE bit in the PIE1 register. The ADIF bit must be cleared in software.

Note 1: The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.

2: The ADC operates during Sleep only when the FRC oscillator is selected.

This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake-up from Sleep and resume in-line code execution, the ADIE bit of the PIEx register and the GIE bits of the INTCON0 register must both be set. If all these bits are set, the execution will switch to the Interrupt Service Routine.

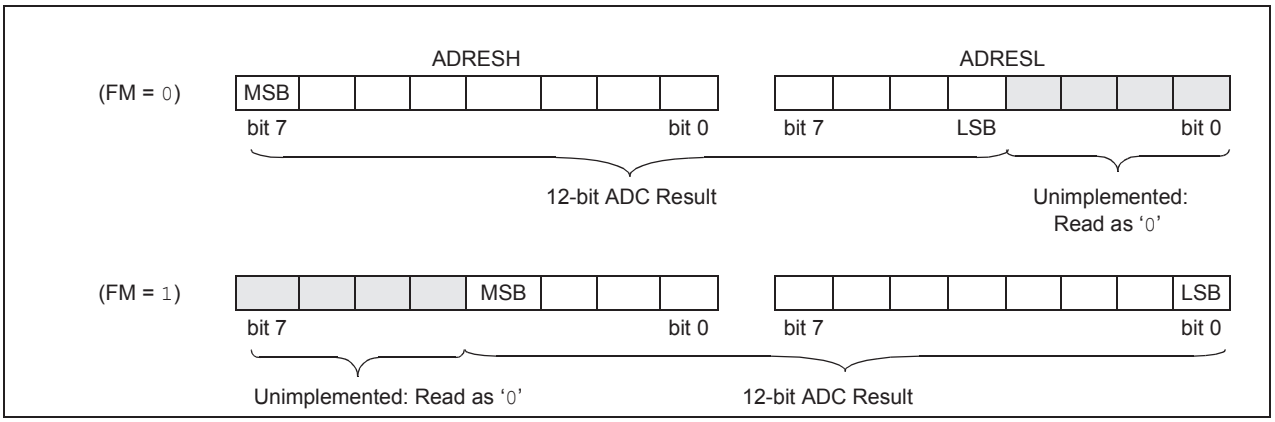
38.1.6 RESULT FORMATTING

The 12-bit ADC conversion result can be supplied in two formats, left justified or right justified. The FM bits of the ADCON0 register controls the output format.

Figure 38-3 shows the two output formats.

Writes to the ADRES register pair are always right justified regardless of the selected format mode. Therefore, data read after writing to ADRES when ADFRM0 = 0 will be shifted left four places.

FIGURE 38-3: 12-BIT ADC CONVERSION RESULT FORMAT



REGISTER 38-22: ADPREVH: ADC PREVIOUS RESULT REGISTER

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
PREV<15:8>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-0 **PREV<15:8>**: Previous ADC Results bits
If ADPSIS = 1:
Upper byte of FLTR at the start of current ADC conversion
If ADPSIS = 0:
Upper bits of ADRES at the start of current ADC conversion(1)

Note 1: If ADPSIS = 0, ADPREVH and ADPREVL are formatted the same way as ADRES is, depending on the FM bit.

REGISTER 38-23: ADPREVL: ADC PREVIOUS RESULT REGISTER

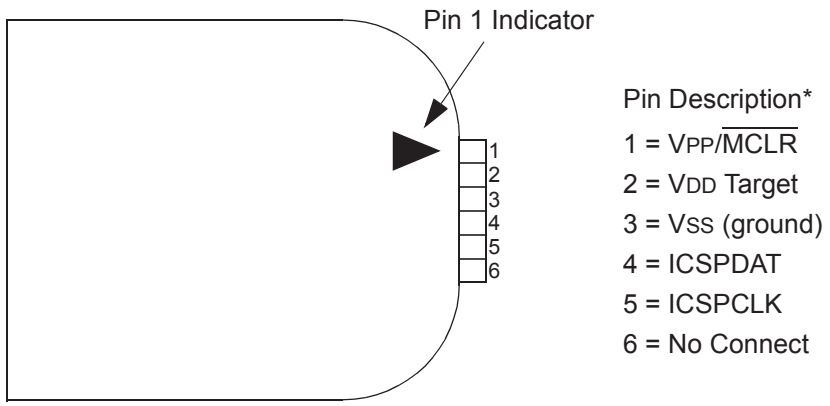
R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
PREV<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-0 **PREV<7:0>**: Previous ADC Results bits
If ADPSIS = 1:
Lower byte of FLTR at the start of current ADC conversion
If ADPSIS = 0:
Lower bits of ADRES at the start of current ADC conversion(1)

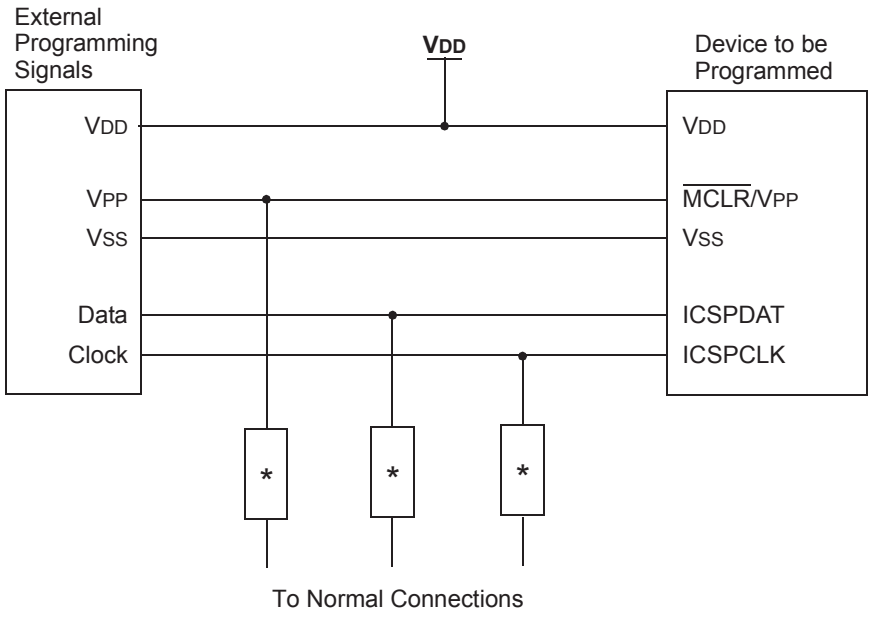
Note 1: If ADPSIS = 0, ADPREVH and ADPREVL are formatted the same way as ADRES is, depending on the FM bit.

FIGURE 42-2: PICkit™ PROGRAMMER STYLE CONNECTOR INTERFACE



* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

FIGURE 42-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING



* Isolation devices (as required).

ADDWFC **ADD W and CARRY bit to f**

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

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) + (f) + (C) → dest

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

Encoding:

0010	00da	ffff	ffff

Description: Add W, the CARRY flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
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 43.2.3 "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: ADDWFC REG, 0, 1

Before Instruction
CARRY bit = 1
REG = 02h
W = 4Dh
After Instruction
CARRY bit = 0
REG = 02h
W = 50h

ANDLW **AND literal with W**

Syntax: ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

0000	1011	kkkk	kkkk

Description: The contents of W are AND'ed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ANDLW 05Fh

Before Instruction
W = A3h
After Instruction
W = 03h

MOVFFL **Move f to f (Long Range)**

Syntax: MOVFFL f_s,f_d

Operands: $0 \le f_s \le 16383$
$0 \le f_d \le 16383$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word	0000	0000	0110	$f_sf_sf_sf_s$
2nd word	1111	$f_sf_sf_sf_s$	$f_sf_sf_sf_s$	$f_sf_sf_df_d$
3rd word	1111	$f_df_df_df_d$	$f_df_df_df_d$	$f_df_df_df_d$

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 16 Kbyte data space (0000h to 3FFFh). Either source or destination can be W (a useful special situation).
MOVFFL is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).
The MOVFFL instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

Words: 3

Cycles: 3

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
Decode	Read register 'f_s' (src)	Process data	No operation
Decode	No operation No dummy read	No operation	Write register 'f_d' (dest)

Example: MOVFFL 2000h, 200Ah

Before Instruction
Contents of 2000h = 33h
Contents of 200Ah = 11h
After Instruction
Contents of 2000h = 33h
Contents of 200Ah = 33h

MOVLB **Move literal to BSR**

Syntax: MOVLW k

Operands: $0 \le k \le 63$

Operation: $k \rightarrow$ BSR

Status Affected: None

Encoding:

0000	0001	00kk	kkkk

Description: The 6-bit literal 'k' is loaded into the Bank Select Register (BSR<5:0>). The value of BSR<7:6> always remains '0'.

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

TABLE 43-2: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDULNK	k	Add FSR2 with (k) & return	2	1110	1000	11kk	kkkk	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word	2	1111	ffff	ffff	ffff	
MOVSFL	z_s, f_d	Opcode 1st word		0000	0000	0000	0010	None
		Move z_s (source) to 2nd word	3	1111	xxxz	zzzz	zzff	
		f_d (full destination) 3rd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word		1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word	2	1111	xxxx	xzzz	zzzz	
PUSHL	k	Push literal to POSTDEC2	1	1110	1010	kkkk	kkkk	None
SUBULNK	k	Subtract (k) from FSR2 & return	2	1110	1001	11kk	kkkk	None

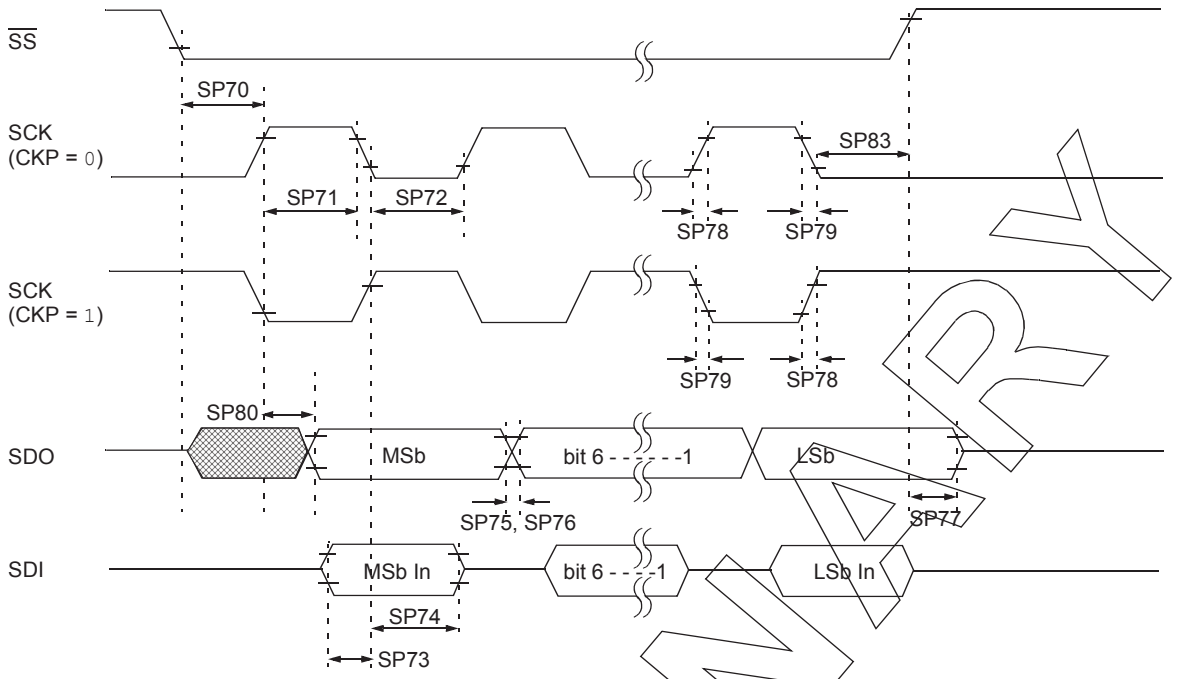
Note 1: If Program Counter (PC) is modified or a conditional test is true, the instruction requires an additional cycle. The extra cycle is executed as a NOP.

2: Some instructions are multi word instructions. The second/third words of these instructions will be decoded 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.

3: Only available when extended instruction set is enabled.

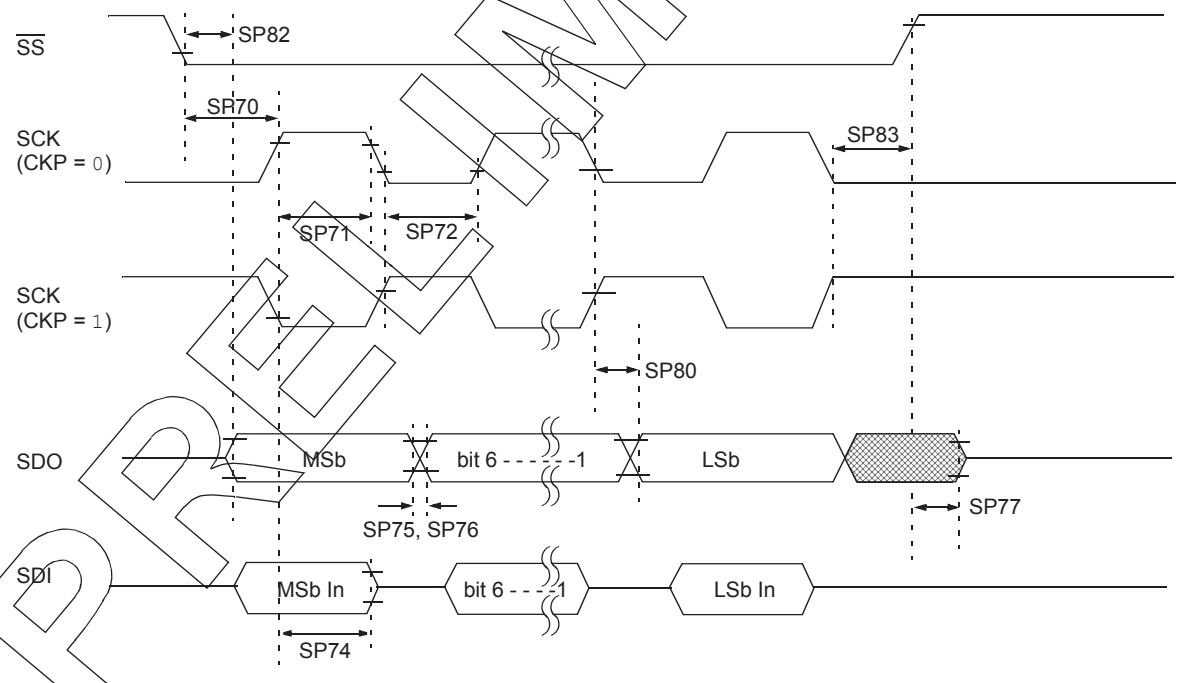
4: f_s and f_d do not cover the full memory range. 2 MSBs of bank selection are forced to ‘b00 to limit the range of these instructions to lower 4k addressing space.

FIGURE 46-16: SPI SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 46-4 for load conditions.

FIGURE 46-17: SPI SLAVE MODE TIMING (CKE = 1)



Note: Refer to Figure 46-4 for load conditions.