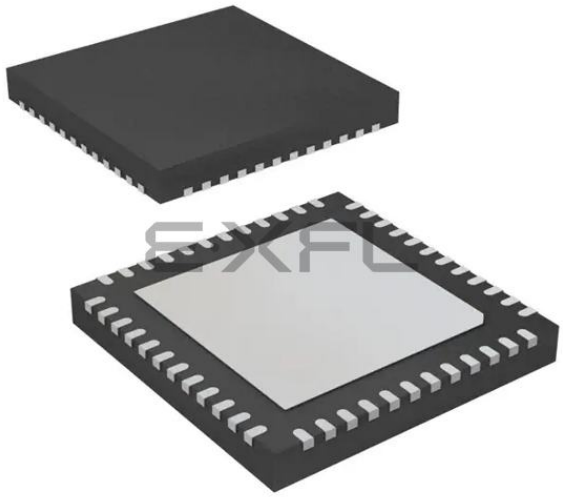


Welcome to E-XFL.COM

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, LVD, POR, PWM, WDT
Number of I/O	44
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 43x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-UFQFN Exposed Pad
Supplier Device Package	48-UQFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f55k42-i-mv

Pin Allocation Tables

TABLE 1: 28-PIN ALLOCATION TABLE (PIC18(L)F2XK42)

I/O	28-Pin SPDIP/SOIC/SSOP	28-Pin (U)QFN	ADC	Voltage Reference	DAC	Comparators	Zero Cross Detect	I^2C	SPI	UART	DSM	Timers/SMT	CCP and PWM	CWG	CLC	NCO	Clock Reference (CLKR)	Interrupt-on-Change	Basic
RA0	2	27	ANA0	—	—	C1IN0- C2IN0-	—	—	—	—	—	—	—	—	CLCIN0(1)	—	—	IOCA0	—
RA1	3	28	ANA1	—	—	C1IN1- C2IN1-	—	—	—	—	—	—	—	—	CLCIN1(1)	—	—	IOCA1	—
RA2	4	1	ANA2	VREF-	DAC1OUT1	C1IN0+ C2IN0+	—	—	—	—	—	—	—	—	—	—	—	IOCA2	—
RA3	5	2	ANA3	VREF+	—	C1IN1+	—	—	—	—	MDCARL(1)	—	—	—	—	—	—	IOCA3	—
RA4	6	3	ANA4	—	—	—	—	—	—	—	MDCARH(1)	T0CKI(1)	—	—	—	—	—	IOCA4	—
RA5	7	4	ANA5	—	—	—	—	—	SS1(1)	—	MDSRC(1)	—	—	—	—	—	—	IOCA5	—
RA6	10	7	ANA6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA6	OSC2 CLKOUT
RA7	9	6	ANA7	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA7	OSC1 CLKIN
RB0	21	18	ANB0	—	—	C2IN1+	ZCD	—	—	—	—	—	CCP4(1)	CWG1IN(1)	—	—	—	INT0(1) IOCB0	—
RB1	22	19	ANB1	—	—	C1IN3- C2IN3-	—	SCL2(3,4)	—	—	—	—	—	CWG2IN(1)	—	—	—	INT1(1) IOCB1	—
RB2	23	20	ANB2	—	—	—	—	SDA2(3,4)	—	—	—	—	—	CWG3IN(1)	—	—	—	INT2(1) IOCB2	—
RB3	24	21	ANB3	—	—	C1IN2- C2IN2-	—	—	—	—	—	—	—	—	—	—	—	IOCB3	—
RB4	25	22	ANB4 ADCACT(1)	—	—	—	—	—	—	—	—	T5G(1)	—	—	—	—	—	IOCB4	—
RB5	26	23	ANB5	—	—	—	—	—	—	—	—	T1G(1)	CCP3(1)	—	—	—	—	IOCB5	—
RB6	27	24	ANB6	—	—	—	—	—	—	CTS2(1)	—	—	—	—	CLCIN2(1)	—	—	IOCB6	ICSPCLK
RB7	28	25	ANB7	—	DAC1OUT2	—	—	—	—	RX2(1)	—	T6IN(1)	—	—	CLCIN3(1)	—	—	IOCB7	ICSPDAT

Note 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins.

2: All output signals shown in this row are PPS remappable.

3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.

4: These pins can be configured for I^2C and SMB™ 3.0/2.0 logic levels; The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST as selected by the INLVL register, instead of the I^2C specific or SMBus input buffer thresholds.

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18(L)F26/27/45/46/47/55/56/57K42 MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18(L)F26/27/45/46/47/55/56/57K42 family of 8-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins (see **Section 2.2 "Power Supply Pins"**)
- $\overline{\text{MCLR}}$ pin (see **Section 2.3 "Master Clear (MCLR) Pin"**)

These pins must also be connected if they are being used in the end application:

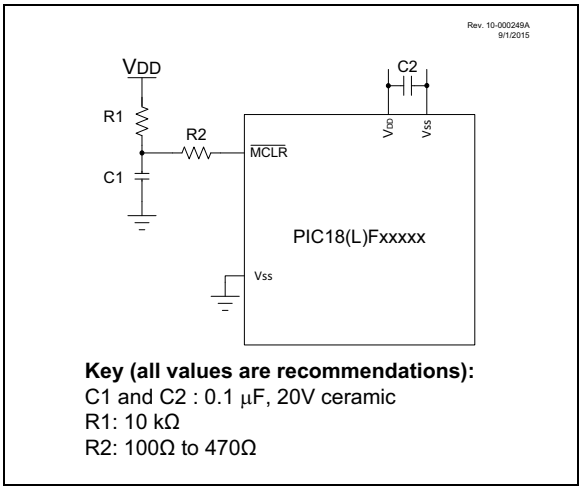
- ICSPCLK/ICSPDAT pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes (see **Section 2.4 "ICSP™ Pins"**)
- OSCI and OSCO pins when an external oscillator source is used (see **Section 2.5 "External Oscillator Pins"**)

Additionally, the following pins may be required:

- VREF+/VREF- pins are used when external voltage reference for analog modules is implemented

The minimum mandatory connections are shown in Figure 2-1.

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS



Key (all values are recommendations):
C1 and C2 : 0.1 µF, 20V ceramic
R1: 10 kΩ
R2: 100Ω to 470Ω

2.2 Power Supply Pins

2.2.1 DECOUPLING CAPACITORS

The use of decoupling capacitors on every pair of power supply pins (VDD and VSS) is required.

Consider the following criteria when using decoupling capacitors:

- **Value and type of capacitor:** A 0.1 µF (100 nF), 10-20V capacitor is recommended. The capacitor should be a low-ESR device, with a resonance frequency in the range of 200 MHz and higher. Ceramic capacitors are recommended.
- **Placement on the printed circuit board:** The decoupling capacitors should be placed as close to the pins as possible. It is recommended to place the capacitors on the same side of the board as the device. If space is constricted, the capacitor can be placed on another layer on the PCB using a via; however, make sure that the trace length from the pin to the capacitor is no greater than 0.25 inch (6 mm).
- **Handling high-frequency noise:** If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01 µF to 0.001 µF. Place this second capacitor next to each primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible (e.g., 0.1 µF in parallel with 0.001 µF).
- **Maximizing performance:** On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

2.2.2 TANK CAPACITORS

On boards with power traces running longer than six inches in length, it is suggested to use a tank capacitor for integrated circuits, including microcontrollers, to supply a local power source. The value of the tank capacitor should be determined based on the trace resistance that connects the power supply source to the device, and the maximum current drawn by the device in the application. In other words, select the tank capacitor so that it meets the acceptable voltage sag at the device. Typical values range from 4.7 µF to 47 µF.

3.1.1 PRIORITY LOCK

The System arbiter grants memory access to the peripheral selections (DMAx, Scanner) when the PRLOCKED bit (PRLOCK Register) is set.

Priority selections are locked by setting the PRLOCKED bit of the PRLOCK register. Setting and clearing this bit requires a special sequence as an extra precaution against inadvertent changes. Examples of setting and clearing the PRLOCKED bit are shown in Example 3-1 and Example 3-2.

EXAMPLE 3-1: PRIORITY LOCK SEQUENCE

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Set PRLOCKED bit to grant memory
access to peripherals
BSF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

EXAMPLE 3-2: PRIORITY UNLOCK SEQUENCE

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Clear PRLOCKED bit to allow changing
priority settings
BCF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

3.2 Memory Access Scheme

The user can assign priorities to both system level and peripheral selections based on which the system arbiter grants memory access. Let us consider the following priority scenarios between ISR, MAIN, and Peripherals.

Note:	It is always required that the ISR priority be higher than Main priority.

3.2.1 ISR PRIORITY > MAIN PRIORITY > PERIPHERAL PRIORITY

When the Peripheral Priority (DMAx, Scanner) is lower than ISR and MAIN Priority, and the peripheral requires:

1. Access to the Program Flash Memory, then the peripheral waits for an instruction cycle in which the CPU does not need to access the PFM (such as a branch instruction) and uses that cycle to do its own Program Flash Memory access, unless a PFM Read/Write operation is in progress.
2. Access to the SFR/GPR, then the peripheral waits for an instruction cycle in which the CPU does not need to access the SFR/GPR (such as `MOVLW`, `CALL`, `NOP`) and uses that cycle to do its own SFR/GPR access.
3. Access to the Data EEPROM, then the peripheral has access to Data EEPROM unless a Data EEPROM Read/Write operation is being performed.

This results in the lowest throughput for the peripheral to access the memory, and does so without any impact on execution times.

3.2.2 PERIPHERAL PRIORITY > ISR PRIORITY > MAIN PRIORITY

When the Peripheral Priority (DMAx, Scanner) is higher than ISR and MAIN Priority, the CPU operation is stalled when the peripheral requests memory.

The CPU is held in its current state until the peripheral completes its operation. Since the peripheral requests access to the bus, the peripheral cannot be disabled until it completes its operation.

This results in the highest throughput for the peripheral to access the memory, but has the cost of stalling other execution while it occurs.

5.0 DEVICE CONFIGURATION

Device configuration consists of the Configuration Words, User ID, Device ID, Rev ID, Device Information Area (DIA), (see **Section 5.7 "Device Information Area"**), and the Device Configuration Information (DCI) regions, (see **Section 5.8 "Device Configuration Information"**).

5.1 Configuration Words

There are six Configuration Word bits that allow the user to setup the device with several choices of oscillators, Resets and memory protection options. These are implemented as Configuration Word 1 through Configuration Word 6 at 300000h through 30000Bh.

5.7.1 MICROCHIP UNIQUE IDENTIFIER (MUI)

The PIC18(L)F26/27/45/46/47/55/56/57K42 devices are individually encoded during final manufacturing with a Microchip Unique Identifier, or MUI. The MUI cannot be user-erased. This feature allows for manufacturing traceability of Microchip Technology devices in applications where this is a required. It may also be used by the application manufacturer for a number of functions that require unverified unique identification, such as:

- Tracking the device
- Unique serial number

The MUI consists of six program words. When read together, these fields form a unique identifier. The MUI is stored in nine read-only locations, located between 3F0000h to 3F000Fh in the DIA space. Table 5-3 lists the addresses of the identifier words.

Note:	For applications that require verified unique identification, contact your Microchip Technology sales office to create a Serialized Quick Turn Programming^{SM} option.

5.7.2 EXTERNAL UNIQUE IDENTIFIER (EUI)

The EUI data is stored at locations 3F0010h to 3F0023h in the Program Memory region. This region is an optional space for placing application specific information. The data is coded per customer requirements during manufacturing.

Note:	Data is stored in this address range on receiving a request from the customer. The customer may contact the local sales representative, or Field Applications Engineer, and provide them the unique identifier information that is supposed to be stored in this region.

5.7.3 ANALOG-TO-DIGITAL CONVERSION DATA OF THE TEMPERATURE SENSOR

The purpose of the Temperature Sensor module is to provide a temperature-dependent voltage that can be measured by an analog module, see **Section 35.0 “Temperature Indicator Module”**.

The DIA table contains the internal ADC measurement values of the Temperature sensor for Low and High range at fixed points of reference. The values are measured during test and are unique to each device. The measurement data is stored in the DIA memory region as hexadecimal numbers corresponding to the ADC conversion result. The calibration data can be used to plot the approximate sensor output voltage, V_{TSENSE} vs. Temperature curve without having to make calibration measurements in the application. For more information on the operation of the Temperature Sensor, refer to **Section 35.0 “Temperature Indicator Module”**.

- **TSLR2**: Address 3F0026h to 3F0027h store the measurements for the low-range setting of the Temperature Sensor at V_{DD} = 3V.
- **TSHR2**: Address 3F002Ch to 3F002Dh store the measurements for the High Range setting of the Temperature Sensor at V_{DD} = 3V.
- The stored measurements are made by the device ADC using the internal V_{REF} = 2.048V.

7.2.2 INTERNAL CLOCK SOURCES

The device may be configured to use the internal oscillator block as the system clock by performing one of the following actions:

- Program the RSTOSC<2:0> bits in Configuration Words to select the INTOSC clock source, which will be used as the default system clock upon a device Reset.
- Write the NOSC<2:0> bits in the OSCCON1 register to switch the system clock source to the internal oscillator during run-time. See **Section 7.3 "Clock Switching"** for more information.

In INTOSC mode, OSC1/CLKIN is available for general purpose I/O, provided that FEXTOSC is configured to 'oscillator is not enabled'. OSC2/CLKOUT is available for general purpose I/O or CLKOUT.

The function of the OSC2/CLKOUT pin is determined by the CLKOUTEN bit in Configuration Words.

The internal oscillator block has two independent oscillators that can produce two internal system clock sources.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory-calibrated and operates from 1 to 64 MHz. The frequency of HFINTOSC can be selected through the OSCFRQ Frequency Selection register, and fine-tuning can be done via the OSCTUNE register.
2. The **LFINTOSC** (Low-Frequency Internal Oscillator) is factory-calibrated and operates at 31 kHz.

7.2.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a precision digitally-controlled internal clock source that produces a stable clock up to 64 MHz. The HFINTOSC can be enabled through one of the following methods:

- Programming the RSTOSC<2:0> bits in Configuration Word 1 to '110' (FOSC = 1 MHz) or '000' (FOSC = 64 MHz) to set the oscillator upon device Power-up or Reset.
- Write to the NOSC<2:0> bits of the OSCCON1 register during run-time. See **Section 7.3 "Clock Switching"** for more information.

The HFINTOSC frequency can be selected by setting the FRQ<3:0> bits of the OSCFRQ register.

The NDIV<3:0> bits of the OSCCON1 register allow for division of the HFINTOSC output from a range between 1:1 and 1:512.

7.2.2.2 MFINTOSC

The module provides two (500 kHz and 31.25 kHz) constant clock outputs. These clocks are digital divisors of the HFINTOSC clock. Dynamic divider logic is used to provide constant MFINTOSC clock rates for all settings of HFINTOSC.

The MFINTOSC cannot be used to drive the system but it is used to clock certain modules such as the Timers and WWDT.

REGISTER 7-7: OSCEN: OSCILLATOR MANUAL ENABLE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0
EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—

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 **EXTOEN:** External Oscillator Manual Request Enable bit
1 = EXTOSC is explicitly enabled, operating as specified by FEXTOSC
0 = EXTOSC could be enabled by requesting peripheral

bit 6 **HFOEN:** HFINTOSC Oscillator Manual Request Enable bit
1 = HFINTOSC is explicitly enabled, operating as specified by OSCFRQ (Register 7-5)
0 = HFINTOSC could be enabled by requesting peripheral

bit 5 **MFOEN:** MFINTOSC (500 kHz/31.25 kHz) Oscillator Manual Request Enable bit (Derived from HFINTOSC)
1 = MFINTOSC is explicitly enabled
0 = MFINTOSC could be enabled by requesting peripheral

bit 4 **LFOEN:** LFINTOSC (31 kHz) Oscillator Manual Request Enable bit
1 = LFINTOSC is explicitly enabled
0 = LFINTOSC could be enabled by requesting peripheral

bit 3 **SOSCEN:** Secondary Oscillator Manual Request Enable bit
1 = Secondary Oscillator is explicitly enabled, operating as specified by SOSCPWR
0 = Secondary Oscillator could be enabled by requesting peripheral

bit 2 **ADOEN:** ADC Oscillator Manual Request Enable bit
1 = ADC oscillator is explicitly enabled
0 = ADC oscillator could be enabled by requesting peripheral

bit 1-0 **Unimplemented:** Read as '0'

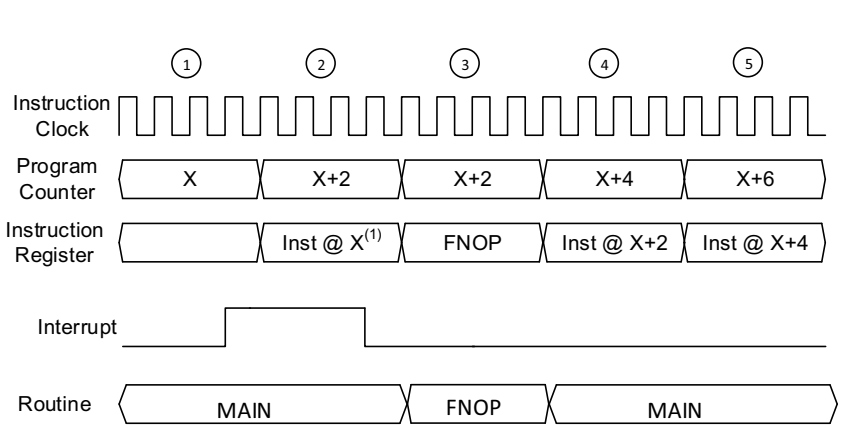
9.7.1 ABORTING INTERRUPTS

If the last instruction before the interrupt controller vectors to the ISR from main routine clears the GIE, PIE or PIR bit associated with the interrupt, the controller executes one force NOP cycle before it returns to the main routine.

Figure 9-10 illustrates the sequence of events when a peripheral interrupt is asserted and then cleared on the last executed instruction cycle.

If the GIE, PIE or PIR bit associated with the interrupt is cleared prior to vectoring to the ISR, then the controller continues executing the main routine.

FIGURE 9-10: INTERRUPT TIMING DIAGRAM - ABORTING INTERRUPTS



Note 1: Inst @ X clears the interrupt flag, Example BCF INTCON0, GIE.

REGISTER 9-9: PIR6: PERIPHERAL INTERRUPT REGISTER 6(1)

R/W/HS-0/0	R/W/HS-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
TMR3GIF	TMR3IF	U2IF(2)	U2EIF(3)	U2TXIF(4)	U2RXIF(4)	I2C2EIF(5)	I2C2IF(6)
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 **TMR3GIF:** TMR3 Gate Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 6 **TMR3IF:** TMR3 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 5 **U2IF:** UART2 Interrupt Flag bit(2)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 4 **U2EIF:** UART2 Framing Error Interrupt Flag bit(3)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 3 **U2TXIF:** UART2 Transmit Interrupt Flag bit(4)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 2 **U2RXIF:** UART2 Receive Interrupt Flag bit(4)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 1 **I2C2EIF:** I^2C2 Error Interrupt Flag bit(5)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 0 **I2C2IF:** I^2C2 Interrupt Flag bit(6)
1 = Interrupt has occurred
0 = Interrupt event has not occurred

Note 1: 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.
2: UxIF is a read-only bit. To clear the interrupt condition, all bits in the UxUIR register must be cleared.
3: UxEIF is a read-only bit. To clear the interrupt condition, all bits in the UxERRIR register must be cleared.
4: UxTXIF and UxRXIF are read-only bits and cannot be set/cleared by the software.
5: I2CxEIF is a read-only bit. To clear the interrupt condition, all bits in the I2CxERR register must be cleared.
6: I2CxIF is a read-only bit. To clear the interrupt condition, all bits in the I2CxPIR register must be cleared.

REGISTER 9-15: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SMT1PWAIE	SMT1PRAIE	SMT1IE	C1IE	ADTIE	ADIE	ZCDIE	INT0IE
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 **SMT1PWAIE:** SMT1 Pulse Width Acquisition Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **SMT1PRAIE:** SMT1 Period Acquisition Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **SMT1IE:** SMT1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **C1IE:** C1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **ADTIE:** ADC Threshold Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **ADIE:** ADC Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **ZCDIE:** ZCD Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **INT0IE:** External Interrupt 0 Enable bit
1 = Enabled
0 = Disabled

12.0 8x8 HARDWARE MULTIPLIER

12.1 Introduction

All PIC18 devices include an 8x8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 12-1.

12.2 Operation

Example 12-1 shows the instruction sequence for an 8x8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 12-2 shows the sequence to do an 8x8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 12-1: 8x8 UNSIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

EXAMPLE 12-2: 8x8 SIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG2
```

TABLE 12-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 40 MHz	@ 10 MHz	@ 4 MHz
8x8 unsigned	Without hardware multiply	13	69	4.3 µs	6.9 µs	27.6 µs	69 µs
	Hardware multiply	1	1	62.5 ns	100 ns	400 ns	1 µs
8x8 signed	Without hardware multiply	33	91	5.7 µs	9.1 µs	36.4 µs	91 µs
	Hardware multiply	6	6	375 ns	600 ns	2.4 µs	6 µs
16x16 unsigned	Without hardware multiply	21	242	15.1 µs	24.2 µs	96.8 µs	242 µs
	Hardware multiply	28	28	1.8 µs	2.8 µs	11.2 µs	28 µs
16x16 signed	Without hardware multiply	52	254	15.9 µs	25.4 µs	102.6 µs	254 µs
	Hardware multiply	35	40	2.5 µs	4.0 µs	16.0 µs	40 µs

13.0 NONVOLATILE MEMORY (NVM) CONTROL

Nonvolatile Memory (NVM) is separated into two types: Program Flash Memory (PFM) and Data EEPROM Memory.

PFM, Data EEPROM, User IDs and Configuration bits can all be accessed using the REG<1:0> bits of the NVMCON1 register.

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

NVM can be protected in two ways, by either code protection or write protection. Code protection ($\overline{CP}$ and $\overline{CPD}$ bits in Configuration Word 5L) disables access, reading and writing to both PFM and Data EEPROM Memory via external device programmers. Code protection does not affect the self-write and erase functionality. Code protection can only be reset by a device programmer performing a Bulk Erase to the device, clearing all nonvolatile memory, Configuration bits and User IDs.

Write protection prohibits self-write and erase to a portion or all of the PFM, as defined by the WRT bits of Configuration Word 4H. Write protection does not affect a device programmer's ability to read, write or erase the device.

TABLE 13-1: NVM ORGANIZATION AND ACCESS INFORMATION

Memory	PC<20:0> ICSP™ Addr<21:0> TBLPTR<21:0> NVMADDR<9:0>	Execution	User Access		
		CPU Execution	REG	TABLAT	NVMDAT
Program Flash Memory (PFM)	00 0000h ••• 01 FFFFh	Read	10	Read/ Write(1)	—(3)
User IDs(2)	20 0000h ••• 20 000Fh	No Access	x1	Read/ Write	—(3)
Reserved	20 0010h 2F FFFFh	No Access	—(3)		
Configuration	30 0000h ••• 30 0009h	No Access	x1	Read/ Write(1)	—(3)
Reserved	30 000Ah 30 FFFFh	No Access	—(3)		
User Data Memory (Data EEPROM)	31 0000h ••• 31 03FFh	No Access	00	—(3)	Read/ Write(1)
Reserved	31 0400h 3E FFFFh	No Access	—(3)		
Device Information Area (DIA)	3F 0000h ••• 3F 003Fh	No Access	x1	Read	—(3)
Reserved	3F 0040h 3F FF09h	No Access	—(3)		
Device Configuration Information (DCI)	3F FF00h ••• 3F FF09h	No Access	x1	Read	—(3)
Reserved	3F FF0Ah 3F FFFBh	No Access	—(3)		
Revision ID/ Device ID	3F FFFCh ••• 3F FFFFh	No Access	x1	Read	—(3)

Note 1: Subject to Memory Write Protection settings.
2: User IDs are eight words ONLY. There is no code protection, table read protection or write protection implemented for this region.
3: Reads as '0', writes clear the WR bit and WRERR bit is set.

Depending on the priority of the DMA with respect to CPU execution (Refer to **Section 3.2 "Memory Access Scheme"** for more information), the DMA Controller can move data through two methods:

- Stalling the CPU execution until it has completed its transfers (DMA has higher priority over the CPU in this mode of operation)
- Utilizing unused CPU cycles for DMA transfers (CPU has higher priority over the DMA in this mode of operation). Unused CPU cycles are referred to as bubbles which are instruction cycles available for use by the DMA to perform read and write operations. In this way, the effective bandwidth for handling data is increased; at the same time, DMA operations can proceed without causing a processor stall.

15.4 DMA Interface

The DMA module transfers data from the source to the destination one byte at a time, this smallest data movement is called a DMA data transaction. A DMA Message refers to one or more DMA data transactions.

Each DMA data transaction consists of two separate actions:

- Reading the Source Address Memory and storing the value in the DMA Buffer register
- Writing the contents of the DMA Buffer register to the Destination Address Memory

Note:	DMA data movement is a two-cycle operation.

The XIP bit (DMAxCON0 register) is a status bit to indicate whether or not the data in the DMAxBUF register has been written to the destination address. If the bit is set then data is waiting to be written to the destination. If clear, it means that either data has been written to the destination or that no source read has occurred.

The DMA has read access to PFM, Data EEPROM, and SFR/GPR space, and write access to SFR/GPR space. Based on these memory access capabilities, the DMA can support the following memory transactions:

TABLE 15-1: DMA MEMORY ACCESS

Read Source	Write Destination
Program Flash Memory	GPR
Program Flash Memory	SFR
Data EE	GPR
Data EE	SFR
GPR	GPR
SFR	GPR
GPR	SFR
SFR	SFR

Note:	Even though the DMA module has access to all memory and peripherals that are also available to the CPU, it is recommended that the DMA does not access any register that is part of the System arbitration. The DMA, as a system arbitration client should not be read or written by itself or by another DMA instantiation.

The following sections discuss the various control interfaces required for DMA data transfers.

15.4.1 DMA ADDRESSING

The start addresses for the source read and destination write operations are set using the DMAxSSA <21:0> and DMAxDSA <15:0> registers, respectively.

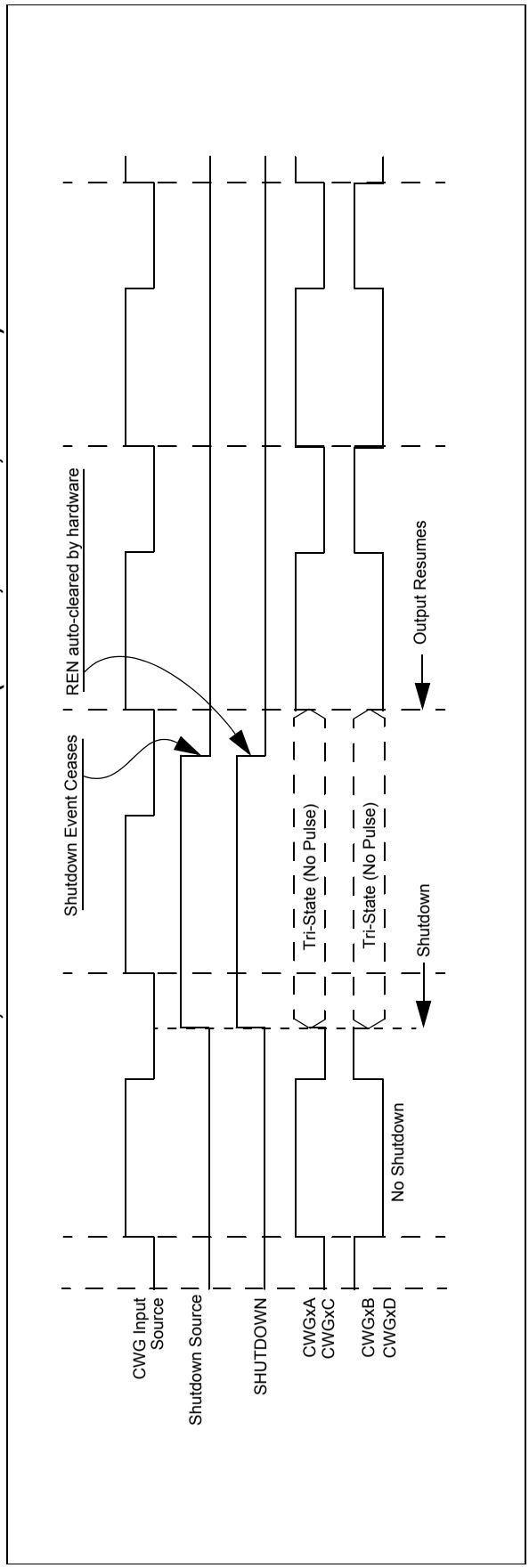
When the DMA Message transfers are in progress, the DMAxSPTR <21:0> and DMAxDPTR <15:0> registers contain the current address pointers for each source read and destination write operation, these registers are modified after each transaction based on the Address mode selection bits.

The SMODE and DMODE bits in the DMAxCON1 control register determine the address modes of operation by controlling how the DMAxSPTR <21:0> and DMAxDPTR <15:0> bits are updated after every DMA data transaction combination (Figure 15-2).

Each address can be separately configured to:

- Remain unchanged
- Increment by 1
- Decrement by 1

FIGURE 26-16: SHUTDOWN FUNCTIONALITY, AUTO-RESTART ENABLED (REN = 1, LSAC = 01, LSBD = 01)

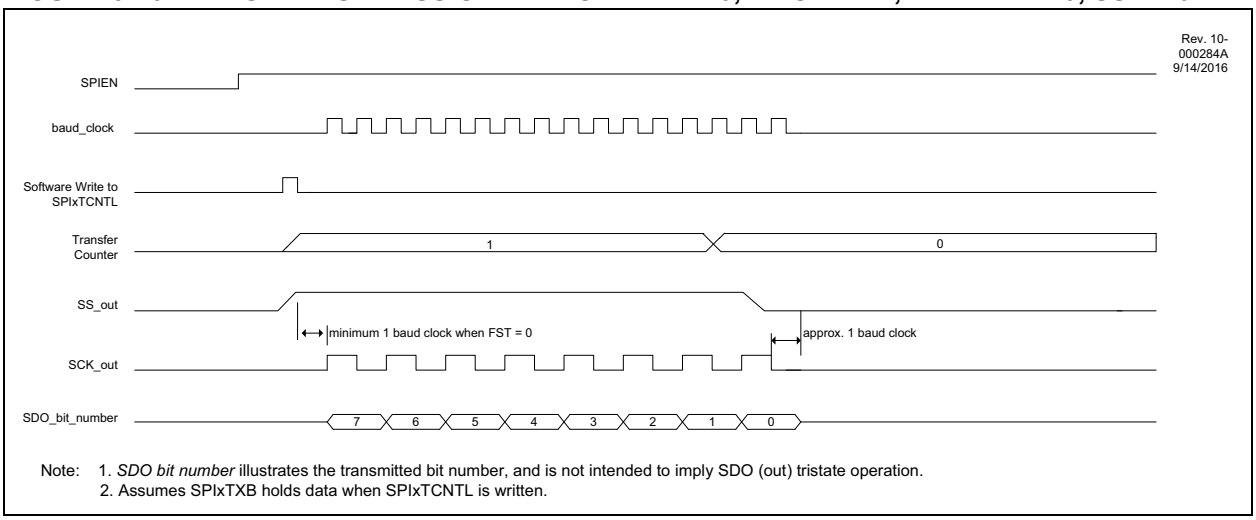


32.5.5 MASTER MODE SLAVE SELECT CONTROL

32.5.5.1 Hardware Slave Select Control

This SPI module allows for direct hardware control of a Slave Select output. The Slave Select output SS(out) is controlled both directly, through the SSET bit of SPIxCON2, as well indirectly by the hardware while the transfer counter is non-zero (see **Section 32.4 "Transfer Counter"**). SS(out) is steered by the PPS registers to pins (see **Section 17.2 "PPS Outputs"**) and its polarity is controlled by the SSP bit of SPIxCON1. Setting the SSET bit will also assert SS(out). Clearing the SSET bit will leave SS(out) to be controlled by the Transfer Counter. When the Transfer Counter is loaded, the SPI module will automatically assert the SS. When the Transfer Counter decrements to zero, the SPI module will deassert SS either one baud period after the final SCK pulse of the final transfer (if CKE/SMP = 0/1) or one half baud period otherwise (see Figure 32-6).

FIGURE 32-6: SPI MASTER SS OPERATION- CKE = 0, BMODE = 1, TCWIDTH = 0, SSP = 0



Note: 1. *SDO bit number* illustrates the transmitted bit number, and is not intended to imply SDO (out) tristate operation.
2. Assumes SPIxTXB holds data when SPIxTCNTL is written.

32.5.5.2 Software Slave Select Control

Slave Select can also be controlled through software via a general purpose I/O pin. In this case, ensure that the pin in question is configured as a GPIO through PPS (see **Section 17.2 "PPS Outputs"**), and ensure that the pin is set as an output (clear the appropriate bit in the appropriate TRIS register). In this case, SSET will not affect the slave select, the Transfer Counter will not automatically control the slave select output, and all setting and clearing of the slave select output line must be directly controlled by software.

REGISTER 33-16: I2CxADB0: I²C ADDRESS DATA BUFFER 0 REGISTER(1)

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADB7	ADB6	ADB5	ADB4	ADB3	ADB2	ADB1	ADB0
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 HC = Hardware clear

bit 7-0

MODE<2:0> = 00x

ADB<7:1>: Address Data byte

Received matching 7-bit slave address data

R/$\overline{W}$: Read/not-Write Data bit

Received read/write value from 7-bit address byte

MODE<2:0> = 01x

ADB<7:0>: Address Data byte

Received matching lower 8-bits of 10-bit slave address data

MODE<2:0> = 100

Unused in this mode; bit state is a don't care

MODE<2:0> = 101

ADB<7:0>: Low Address Data byte

Low 10-bit address value copied to transmit shift register

MODE<2:0> = 11x

ADB<7:1>: Address Data byte

Received matching 7-bit slave address

R/$\overline{W}$: Read/not-Write Data bit

Received read/write value received 7-bit slave address byte

Note 1: This register is read only except in master, 10-bit Address mode (MODE<2:0> = 101).

REGISTER 36-4: ADCON3: ADC CONTROL REGISTER 3

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W/HC-0	R/W-0/0	R/W-0/0	R/W-0/0
—	CALC<2:0>			SOI	TMD<2:0>		
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	HC = Bit is cleared by hardware

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

bit 6-4 **CALC<2:0>:** ADC Error Calculation Mode Select bits

CALC	DSEN = 0 Single-Sample Mode	DSEN = 1 CVD Double-Sample Mode(1)	Application
111	Reserved	Reserved	Reserved
110	Reserved	Reserved	Reserved
101	FLTR-STPT	FLTR-STPT	Average/filtered value vs. setpoint
100	PREV-FLTR	PREV-FLTR	First derivative of filtered value(3) (negative)
011	Reserved	Reserved	Reserved
010	RES-FLTR	(RES-PREV)-FLTR	Actual result vs. averaged/ filtered value
001	RES-STPT	(RES-PREV)-STPT	Actual result vs.setpoint
000	RES-PREV	RES-PREV	First derivative of single measurement(2)
			Actual CVD result in CVD mode(2)

bit 3 **SOI:** ADC Stop-on-Interrupt bit

If CONT = 1:
1 = GO is cleared when the threshold conditions are met, otherwise the conversion is retriggered
0 = GO is not cleared by hardware, must be cleared by software to stop retriggers

bit 2-0 **TMD<2:0>:** Threshold Interrupt Mode Select bits

111 = Interrupt regardless of threshold test results
110 = Interrupt if ERR>UTH
101 = Interrupt if ERR≤UTH
100 = Interrupt if ERR<LTH or ERR>UTH
011 = Interrupt if ERR>LTH and ERR<UTH
010 = Interrupt if ERR≥LTH
001 = Interrupt if ERR<LTH
000 = Never interrupt

Note 1: When PSIS = 0, the value of (RES-PREV) is the value of (S2-S1) from Table 36-2.
2: When PSIS = 0
3: When PSIS = 1.

REGISTER 36-35: ADACT: ADC AUTO CONVERSION TRIGGER CONTROL REGISTER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ACT<4:0>				
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 7-5 **Unimplemented**: Read as '0'

bit 4-0 **ADACT<4:0>**: Auto-Conversion Trigger Select Bits

11111 = Reserved, do not use
•
•
•
11110 = Reserved, do not use
11101 = Software write to ADPCH
11100 = Reserved, do not use
11011 = Software read of ADRESH
11010 = Software read of ADERRH
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = Logical OR of all Interrupt-on-change Interrupt Flags
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = PWM8_out
10000 = PWM7_out
01111 = PWM6_out
01110 = PWM5_out
01101 = CCP4_trigger
01100 = CCP3_trigger
01011 = CCP2_trigger
01010 = CCP1_trigger
01001 = SMT1_trigger
01000 = TMR6_postscaled
00111 = TMR5_overflow
00110 = TMR4_postscaled
00101 = TMR3_overflow
00100 = TMR2_postscaled
00011 = TMR1_overflow
00010 = TMR0_overflow
00001 = Pin selected by ADACTPPS
00000 = External Trigger Disabled

43.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

43.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

43.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

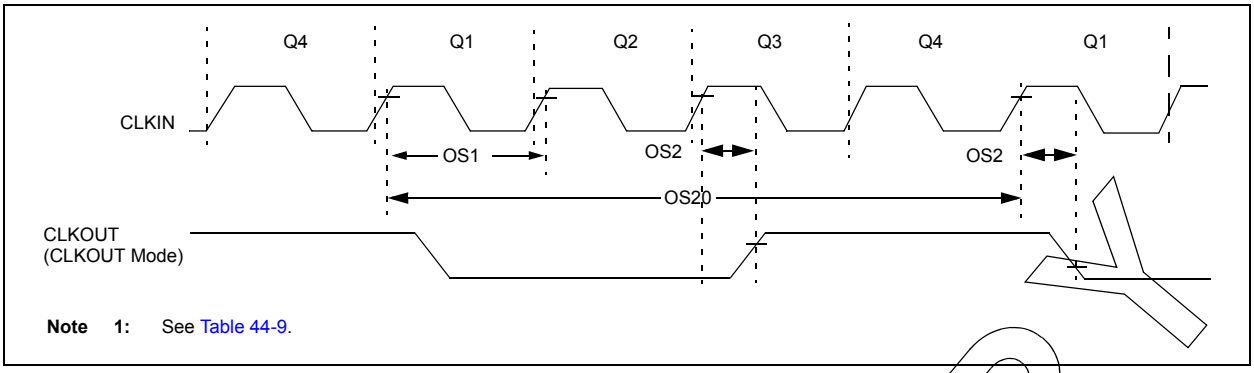
43.9 PICkit 3 In-Circuit Debugger/ Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

43.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

FIGURE 44-5: CLOCK TIMING



Note 1: See Table 44-9.

TABLE 44-9: EXTERNAL CLOCK/OSCILLATOR TIMING REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
ECL Oscillator							
OS1	F_{ECL}	Clock Frequency	—	—	500	kHz	
OS2	T_{ECL_DC}	Clock Duty Cycle	40	—	60	%	
ECM Oscillator							
OS3	F_{ECM}	Clock Frequency	—	—	8	MHz	
OS4	T_{ECM_DC}	Clock Duty Cycle	40	—	60	%	
ECH Oscillator							
OS5	F_{ECH}	Clock Frequency	—	—	64	MHz	
OS6	T_{ECH_DC}	Clock Duty Cycle	40	—	60	%	
LP Oscillator							
OS7	F_{LP}	Clock Frequency	—	—	100	kHz	**Note 4**
XT Oscillator							
OS8	F_{XT}	Clock Frequency	—	—	4	MHz	**Note 4**
HS Oscillator							
OS9	F_{HS}	Clock Frequency	—	—	20	MHz	**Note 4**
Secondary Oscillator							
OS10	F_{SEC}	Clock Frequency	32.4	32.768	33.1	kHz	
System Oscillator							
OS20	F_{OSC}	System Clock Frequency	—	—	64	MHz	**(Note 2, Note 3)**

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min" values with an external clock applied to OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

2: The system clock frequency (FOSC) is selected by the "main clock switch controls" as described in **Section 10.0 "Power-Saving Operation Modes"**.

3: The system clock frequency (FOSC) must meet the voltage requirements defined in the **Section 44.2 "Standard Operating Conditions"**.

4: LP, XT and HS oscillator modes require an appropriate crystal or resonator to be connected to the device. For clocking the device with the external square wave, one of the EC mode selections must be used.