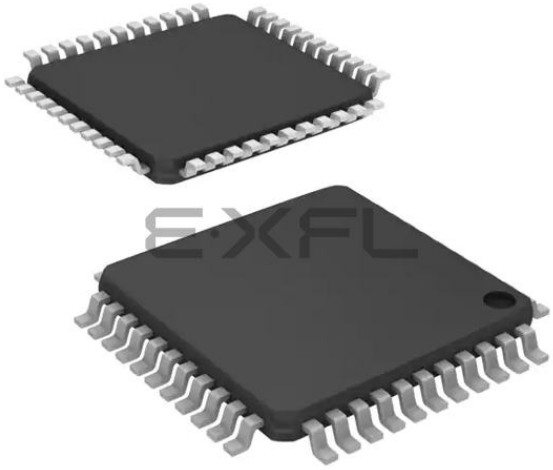


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

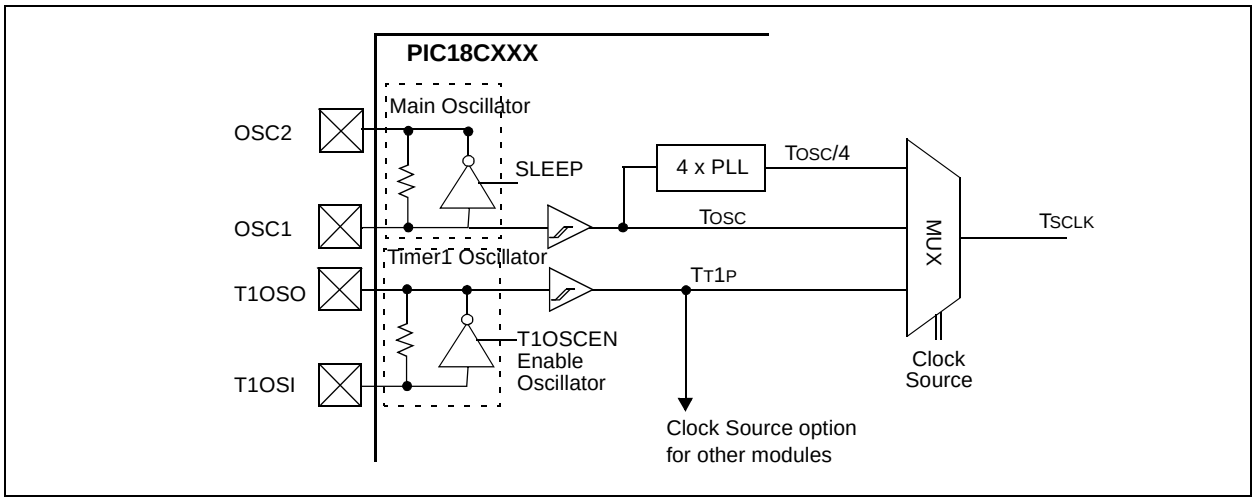
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	33
Program Memory Size	16KB (8K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lc442t-i-pt

2.6 Oscillator Switching Feature

The PIC18CXX2 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low frequency clock source. For the PIC18CXX2 devices, this alternate clock source is the Timer1 oscillator. If a low frequency crystal (32 kHz, for example) has been attached to the Timer1 oscillator pins and the Timer1 oscillator has been enabled, the device can switch to a low power execution mode. Figure 2-7 shows a block diagram of the system clock sources. The clock switching feature is enabled by programming the Oscillator Switching Enable ($\overline{\text{OSCSEN}}$) bit in Configuration Register1H to a '0'. Clock switching is disabled in an erased device. See Section 9.0 for further details of the Timer1 oscillator. See Section 18.0 for Configuration Register details.

FIGURE 2-7: DEVICE CLOCK SOURCES



2.6.1 SYSTEM CLOCK SWITCH BIT

The system clock source switching is performed under software control. The system clock switch bit, SCS (OSCCON<0>) controls the clock switching. When the SCS bit is'0', the system clock source comes from the main oscillator that is selected by the FOSC configuration bits in Configuration Register1H. When the SCS bit is set, the system clock source will come from the Timer1 oscillator. The SCS bit is cleared on all forms of RESET.

Note:	The Timer1 oscillator must be enabled and operating to switch the system clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 control register (T1CON). If the Timer1 oscillator is not enabled, then any write to the SCS bit will be ignored (SCS bit forced cleared) and the main oscillator will continue to be the system clock source.

REGISTER 2-1: OSCCON REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-1
—	—	—	—	—	—	—	SCS
bit 7							bit 0

bit 7-1 **Unimplemented:** Read as '0'

bit 0 **SCS:** System Clock Switch bit
When $\overline{\text{OSCSEN}}$ configuration bit = '0' and T1OSCEN bit is set:
1 = Switch to Timer1 oscillator/clock pin
0 = Use primary oscillator/clock input pin
When $\overline{\text{OSCSEN}}$ and T1OSCEN are in other states:
bit is forced clear

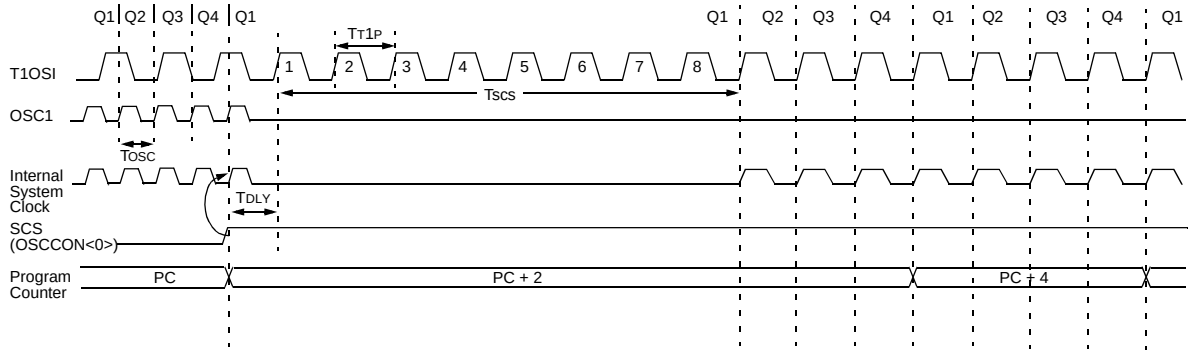
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.6.2 OSCILLATOR TRANSITIONS

The PIC18CXX2 devices contain circuitry to prevent "glitches" when switching between oscillator sources. Essentially, the circuitry waits for eight rising edges of the clock source that the processor is switching to. This ensures that the new clock source is stable and that it's pulse width will not be less than the shortest pulse width of the two clock sources.

A timing diagram indicating the transition from the main oscillator to the Timer1 oscillator is shown in Figure 2-8. The Timer1 oscillator is assumed to be running all the time. After the SCS bit is set, the processor is frozen at the next occurring Q1 cycle. After eight synchronization cycles are counted from the Timer1 oscillator, operation resumes. No additional delays are required after the synchronization cycles.

FIGURE 2-8: TIMING DIAGRAM FOR TRANSITION FROM OSC1 TO TIMER1 OSCILLATOR

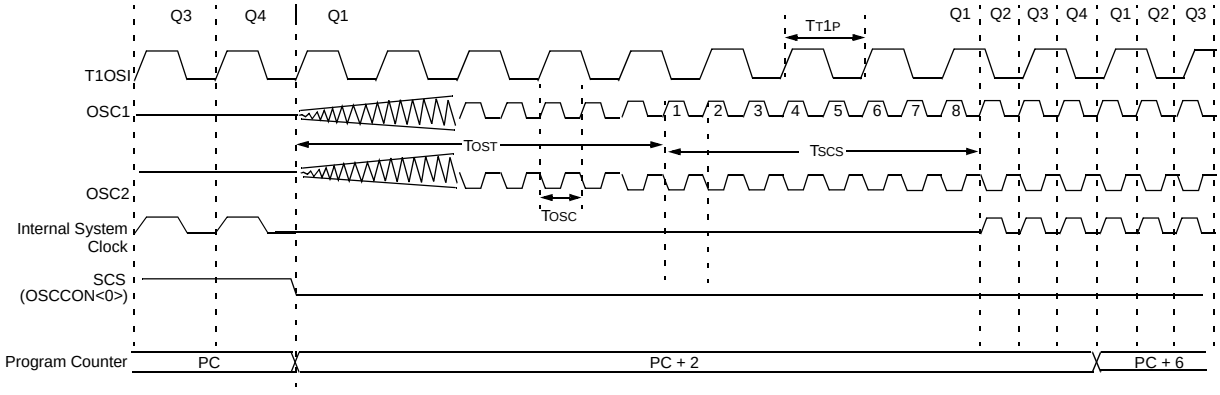


Note 1: Delay on internal system clock is eight oscillator cycles for synchronization.

The sequence of events that takes place when switching from the Timer1 oscillator to the main oscillator will depend on the mode of the main oscillator. In addition to eight clock cycles of the main oscillator, additional delays may take place.

If the main oscillator is configured for an external crystal (HS, XT, LP), then the transition will take place after an oscillator start-up time (TOST) has occurred. A timing diagram indicating the transition from the Timer1 oscillator to the main oscillator for HS, XT and LP modes is shown in Figure 2-9.

FIGURE 2-9: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (HS, XT, LP)



Note 1: TOST = 1024TOSC (drawing not to scale).

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TRISE	242	442	252	452	0000 -111	0000 -111	uuuu -uuu
TRISD	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISC	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISB	242	442	252	452	1111 1111	1111 1111	uuuu uuuu
TRISA(5, 7)	242	442	252	452	-111 1111(5)	-111 1111(5)	-uuu uuuu(5)
LATE	242	442	252	452	---- -xxx	---- -uuu	---- -uuu
LATD	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5, 7)	242	442	252	452	-xxx xxxx(5)	-uuu uuuu(5)	-uuu uuuu(5)
PORTE	242	442	252	452	---- -000	---- -000	---- -uuu
PORTD	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	242	442	252	452	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5, 7)	242	442	252	452	-x0x 0000(5)	-u0u 0000(5)	-uuu uuuu(5)

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 3-2 for RESET value for specific condition.

5: Bit 6 of PORTA, LATA, and TRISA are enabled in ECIO and RCIO oscillator modes only. In all other oscillator modes, they are disabled and read '0'.

6: The long write enable is only reset on a POR or $\overline{\text{MCLR}}$ Reset.

7: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read as '0'.

5.2.3 INTERRUPTS

The long write must be terminated by a RESET or any interrupt.

The interrupt source must have its interrupt enable bit set. When the source sets its interrupt flag, programming will terminate. This will occur, regardless of the settings of interrupt priority bits, the GIE/GIEH bit, or the PIE/GIEL bit.

Depending on the states of interrupt priority bits, the GIE/GIEH bit or the PIE/GIEL bit, program execution can either be vectored to the high or low priority Interrupt Service Routine (ISR), or continue execution from where programming commenced.

In either case, the interrupt flag will not be cleared when programming is terminated and will need to be cleared by the software.

TABLE 5-2: LONG WRITE EXECUTION, INTERRUPT ENABLE BITS AND INTERRUPT RESULTS

GIE/ GIEH	PIE/ GIEL	Priority	Interrupt Enable	Interrupt Flag	Action
X	X	X	0 (default)	X	Long write continues even if interrupt flag becomes set.
X	X	X	1	0	Long write continues, will resume operations when the interrupt flag is set.
0 (default)	0 (default)	X	1	1	Terminates long write, executes next instruction. Interrupt flag not cleared.
0 (default)	1	1 high priority (default)	1	1	Terminates long write, executes next instruction. Interrupt flag not cleared.
1	0 (default)	0 low	1	1	Terminates long write, executes next instruction. Interrupt flag not cleared.
0 (default)	1	0 low	1	1	Terminates long write, branches to low priority interrupt vector. Interrupt flag can be cleared by ISR.
1	0 (default)	1 high priority (default)	1	1	Terminates long write, branches to high priority interrupt vector. Interrupt flag can be cleared by ISR.

5.2.4 UNEXPECTED TERMINATION OF WRITE OPERATIONS

If a write is terminated by an unplanned event such as loss of power, an unexpected RESET, or an interrupt that was not disabled, the memory location just programmed should be verified and reprogrammed if needed.

EXAMPLE 6-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF     ARG1L, W
    MULWF    ARG2L       ; ARG1L * ARG2L ->
                         ; PRODH:PRODL
    MOVFF    PRODH, RES1 ;
    MOVFF    PRODL, RES0 ;
;
    MOVF     ARG1H, W
    MULWF    ARG2H       ; ARG1H * ARG2H ->
                         ; PRODH:PRODL
    MOVFF    PRODH, RES3 ;
    MOVFF    PRODL, RES2 ;
;
    MOVF     ARG1L, W
    MULWF    ARG2H       ; ARG1L * ARG2H ->
                         ; PRODH:PRODL
    MOVF     PRODL, W    ;
    ADDWF    RES1,  F    ; Add cross
    MOVF     PRODH, W    ; products
    ADDWFC   RES2,  F    ;
    CLRF     WREG,  F    ;
    ADDWFC   RES3,  F    ;
;
    MOVF     ARG1H, W    ;
    MULWF    ARG2L       ; ARG1H * ARG2L ->
                         ; PRODH:PRODL
    MOVF     PRODL, W    ;
    ADDWF    RES1,  F    ; Add cross
    MOVF     PRODH, W    ; products
    ADDWFC   RES2,  F    ;
    CLRF     WREG,  F    ;
    ADDWFC   RES3,  F    ;
```

Example 6-4 shows the sequence to do a 16 x 16 signed multiply. Equation 6-2 shows the algorithm used. The 32-bit result is stored in four registers, RES3:RES0. To account for the sign bits of the arguments, each argument pairs' Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EQUATION 6-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

RES3:RES0

= ARG1H:ARG1L • ARG2H:ARG2L

= (ARG1H • ARG2H • 2^{16}) +
(ARG1H • ARG2L • 2^{8}) +
(ARG1L • ARG2H • 2^{8}) +
(ARG1L • ARG2L) +
(-1 • ARG2H<7> • ARG1H:ARG1L • 2^{16}) +
(-1 • ARG1H<7> • ARG2H:ARG2L • 2^{16})

EXAMPLE 6-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
    MOVF     ARG1L, W
    MULWF    ARG2L       ; ARG1L * ARG2L ->
                         ; PRODH:PRODL
    MOVFF    PRODH, RES1 ;
    MOVFF    PRODL, RES0 ;
;
    MOVF     ARG1H, W
    MULWF    ARG2H       ; ARG1H * ARG2H ->
                         ; PRODH:PRODL
    MOVFF    PRODH, RES3 ;
    MOVFF    PRODL, RES2 ;
;
    MOVF     ARG1L, W
    MULWF    ARG2H       ; ARG1L * ARG2H ->
                         ; PRODH:PRODL
    MOVF     PRODL, W    ;
    ADDWF    RES1,  F    ; Add cross
    MOVF     PRODH, W    ; products
    ADDWFC   RES2,  F    ;
    CLRF     WREG,  F    ;
    ADDWFC   RES3,  F    ;
;
    MOVF     ARG1H, W    ;
    MULWF    ARG2L       ; ARG1H * ARG2L ->
                         ; PRODH:PRODL
    MOVF     PRODL, W    ;
    ADDWF    RES1,  F    ; Add cross
    MOVF     PRODH, W    ; products
    ADDWFC   RES2, F     ;
    CLRF     WREG, F     ;
    ADDWFC   RES3, F     ;
;
    BTFSS    ARG2H, 7    ; ARG2H:ARG2L neg?
    BRA      SIGN_ARG1   ; no, check ARG1
    MOVF     ARG1L, W    ;
    SUBWF    RES2        ;
    MOVF     ARG1H, W    ;
    SUBWFB   RES3
;
SIGN_ARG1
    BTFSS    ARG1H, 7    ; ARG1H:ARG1L neg?
    BRA      CONT_CODE   ; no, done
    MOVF     ARG2L, W    ;
    SUBWF    RES2        ;
    MOVF     ARG2H, W    ;
    SUBWFB   RES3
;
CONT_CODE
     :
```

7.5 RCON Register

The RCON register contains the bit which is used to enable prioritized interrupts (IPEN).

REGISTER 7-10: RCON REGISTER

R/W-0	R/W-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	LWRT	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **IPEN:** Interrupt Priority Enable bit

1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (16CXXX compatibility mode)

bit 6 **LWRT:** Long Write Enable bit

For details of bit operation, see Register 4-3

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

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit

For details of bit operation, see Register 4-3

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit

For details of bit operation, see Register 4-3

bit 2 **$\overline{PD}$:** Power-down Detection Flag bit

For details of bit operation, see Register 4-3

bit 1 **$\overline{POR}$:** Power-on Reset Status bit

For details of bit operation, see Register 4-3

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit

For details of bit operation, see Register 4-3

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

8.3 PORTC, TRISC and LATC Registers

PORTC is an 8-bit wide, bi-directional port. The corresponding Data Direction Register is TRISC. Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., put the contents of the output latch on the selected pin).

Note:	On a Power-on Reset, these pins are configured as digital inputs.

The Data Latch register (LATC) is also memory mapped. Read-modify-write operations on the LATC register reads and writes the latched output value for PORTC.

PORTC is multiplexed with several peripheral functions (Table 8-5). PORTC pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTC pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

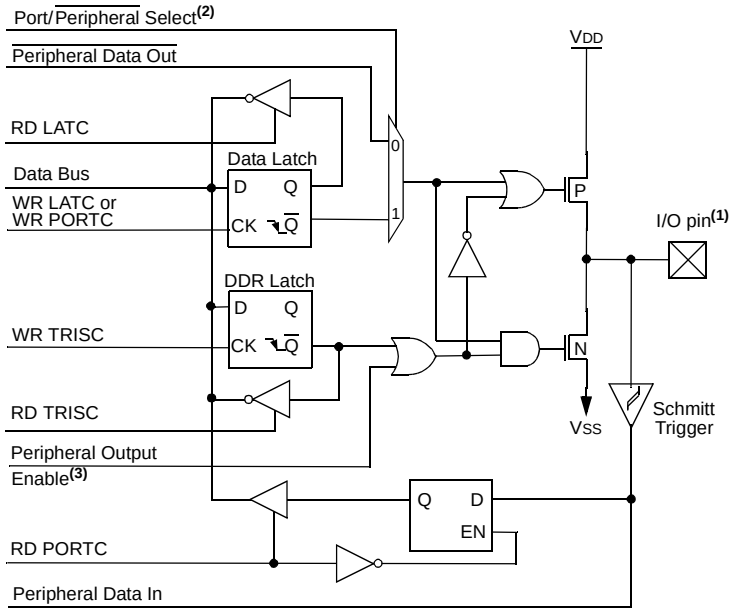
The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register, without concern due to peripheral overrides.

RC1 is normally configured by the configuration bit CCP2MX as the default peripheral pin for the CCP2 module (default/erased state, CCP2MX = '1').

EXAMPLE 8-3: INITIALIZING PORTC

```
CLRF    PORTC    ; Initialize PORTC by
                 ; clearing output
                 ; data latches
CLRF    LATC     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW 0xCF       ; Value used to
                 ; initialize data
                 ; direction
MOVWF TRISC      ; Set RC<3:0> as inputs
                 ; RC<5:4> as outputs
                 ; RC<7:6> as inputs
```

FIGURE 8-7: PORTC BLOCK DIAGRAM (PERIPHERAL OUTPUT OVERRIDE)



Note 1: I/O pins have diode protection to VDD and VSS.

2: Port/Peripheral select signal selects between port data (input) and peripheral output.

3: Peripheral Output Enable is only active if peripheral select is active.

12.0 TIMER3 MODULE

The Timer3 module timer/counter has the following features:

- 16-bit timer/counter
  (two 8-bit registers: TMR3H and TMR3L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module trigger

Figure 12-1 is a simplified block diagram of the Timer3 module.

Register 12-1 shows the Timer3 control register. This register controls the operating mode of the Timer3 module and sets the CCP clock source.

Register 10-1 shows the Timer1 control register. This register controls the operating mode of the Timer1 module, as well as contains the Timer1 oscillator enable bit (T1OSCEN), which can be a clock source for Timer3.

REGISTER 12-1: T3CON: TIMER3 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

bit 7 **RD16:** 16-bit Read/Write Mode Enable
1 = Enables register Read/Write of Timer3 in one 16-bit operation
0 = Enables register Read/Write of Timer3 in two 8-bit operations

bit 6-3 **T3CCP2:T3CCP1:** Timer3 and Timer1 to CCPx Enable bits
1x = Timer3 is the clock source for compare/capture CCP modules
01 = Timer3 is the clock source for compare/capture of CCP2,
Timer1 is the clock source for compare/capture of CCP1
00 = Timer1 is the clock source for compare/capture CCP modules

bit 5-4 **T3CKPS1:T3CKPS0**: Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 2 **$\overline{\text{T3SYNC}}$:** Timer3 External Clock Input Synchronization Control bit
(Not usable if the system clock comes from Timer1/Timer3.)
When TMR3CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR3CS = 0:
This bit is ignored. Timer3 uses the internal clock when TMR3CS = 0.

bit 1 **TMR3CS:** Timer3 Clock Source Select bit
1 = External clock input from Timer1 oscillator or T1CKI
(on the rising edge after the first falling edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR3ON:** Timer3 On bit
1 = Enables Timer3
0 = Stops Timer3

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

14.3.4 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 14-2) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

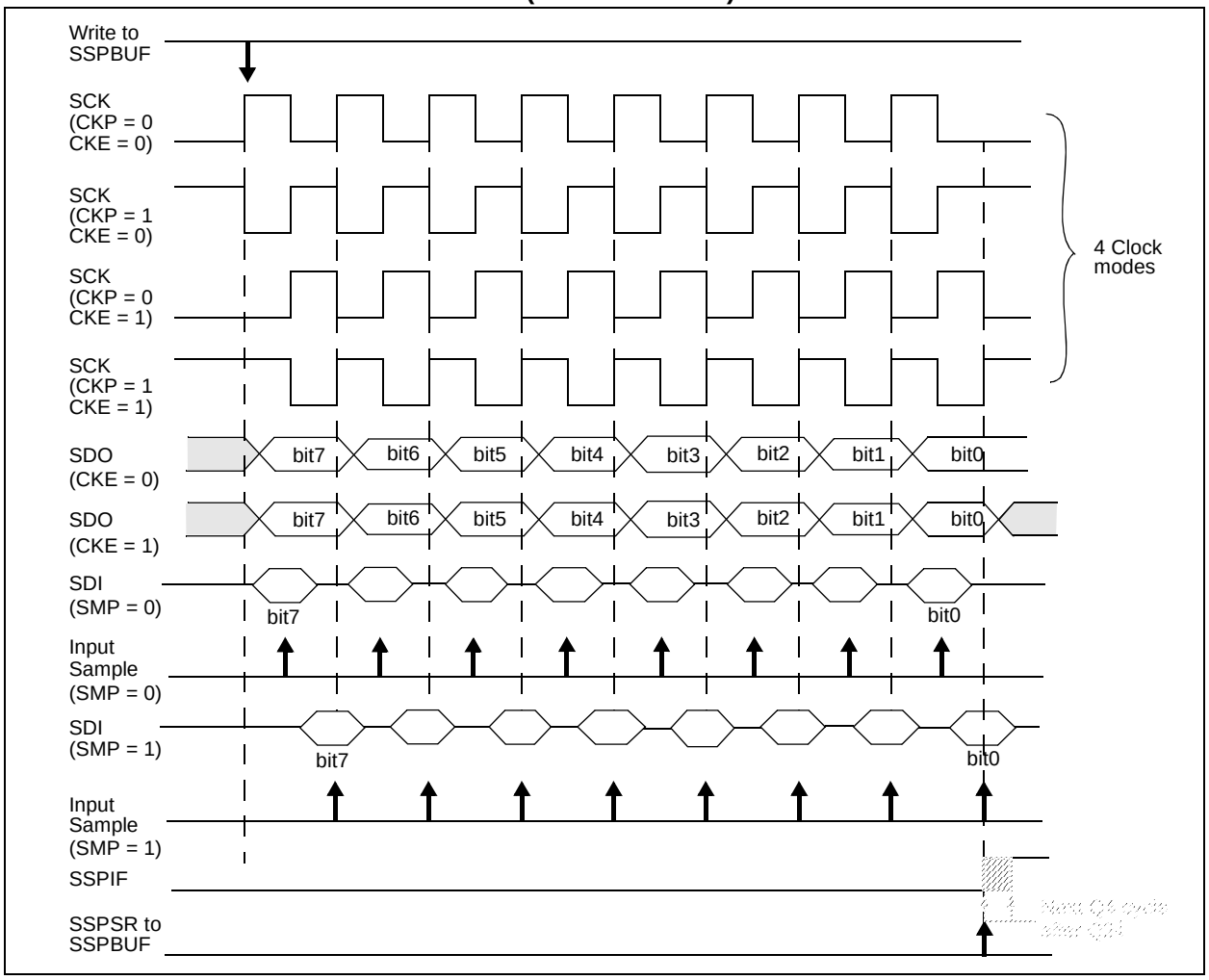
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 14-3, Figure 14-5, and Figure 14-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

This allows a maximum data rate (at 40 MHz) of 10.00 Mbps.

Figure 14-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

FIGURE 14-3: SPI MODE WAVEFORM (MASTER MODE)



The analog reference voltage is software selectable to either the device's positive and negative supply voltage (V_{DD} and V_{SS}) or the voltage level on the RA3/AN3/V_{REF+} pin and RA2/AN2/V_{REF-}.

The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

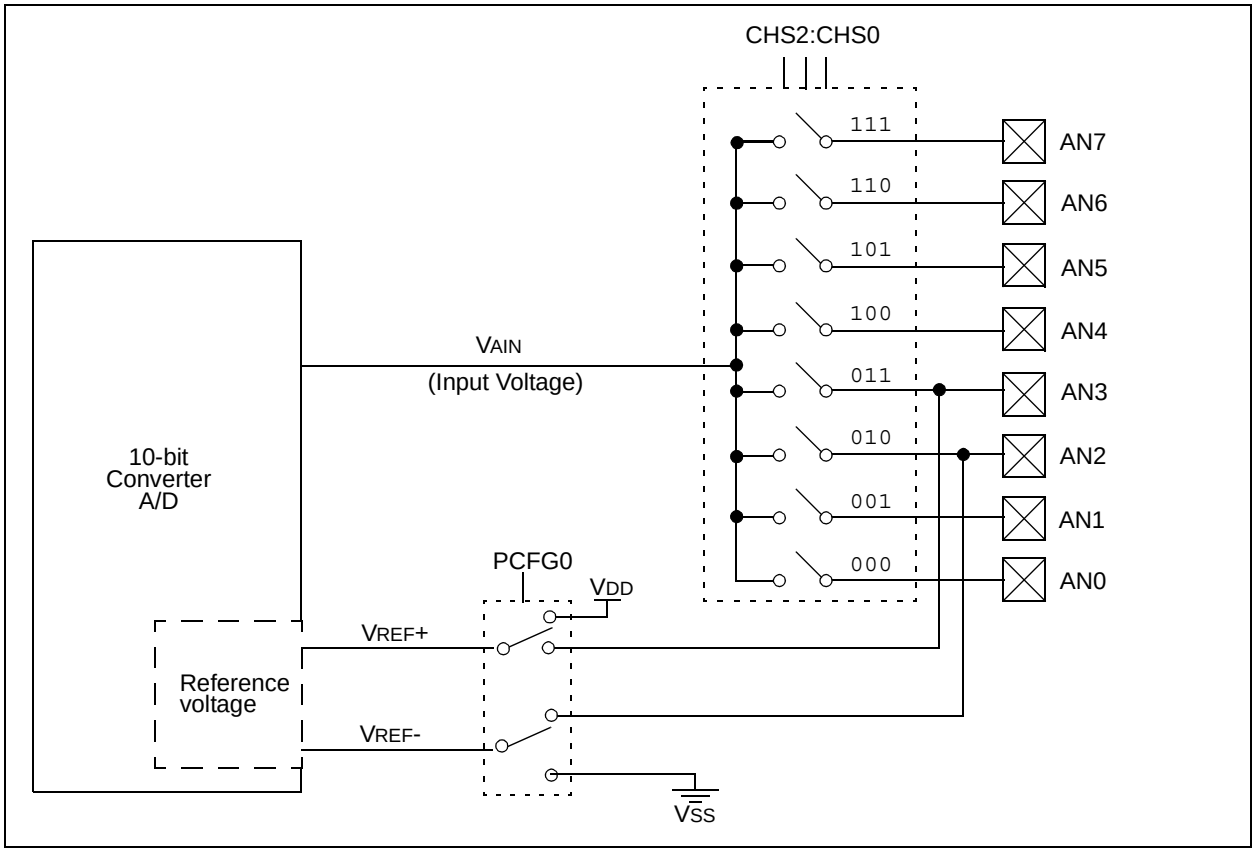
The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

A device RESET forces all registers to their RESET state. This forces the A/D module to be turned off and any conversion is aborted.

Each port pin associated with the A/D converter can be configured as an analog input (RA3 can also be a voltage reference) or as a digital I/O.

The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH/ADRESL registers, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared, and A/D interrupt flag bit ADIF is set. The block diagram of the A/D module is shown in Figure 16-1.

FIGURE 16-1: A/D BLOCK DIAGRAM



16.4 A/D Conversions

Figure 16-3 shows the operation of the A/D converter after the GO bit has been set. Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D result register pair will NOT be updated with the partially completed A/D conversion sample. That is, the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers). After the A/D conversion is aborted, a 2TAD wait is required before the next acquisition is started. After this 2TAD wait, acquisition on the selected channel is automatically started.

Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

16.5 Use of the CCP2 Trigger

An A/D conversion can be started by the "special event trigger" of the CCP2 module. This requires that the CCP2M3:CCP2M0 bits (CCP2CON<3:0>) be programmed as 1011 and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit will be set, starting the A/D conversion and the Timer1 (or Timer3) counter will be reset to zero. Timer1 (or Timer3) is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving ADRESH/ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition done before the "special event trigger" sets the GO/$\overline{\text{DONE}}$ bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), the "special event trigger" will be ignored by the A/D module, but will still reset the Timer1 (or Timer3) counter.

FIGURE 16-3: A/D CONVERSION TAD CYCLES

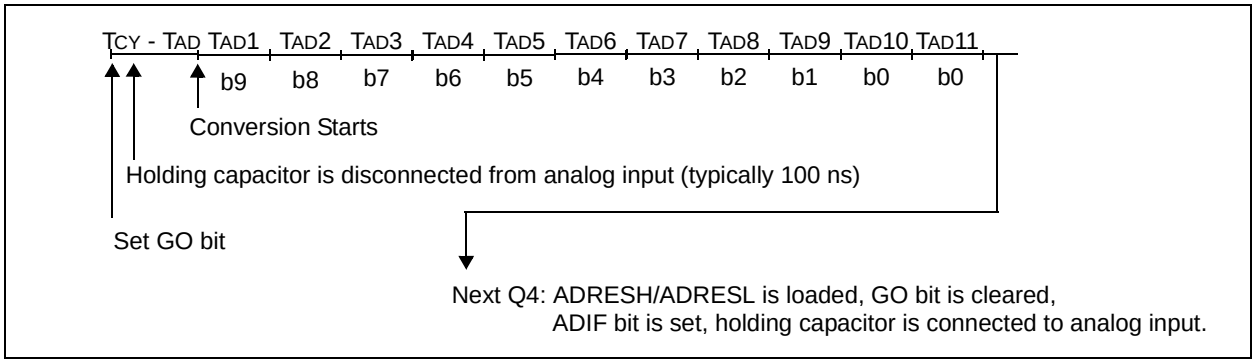
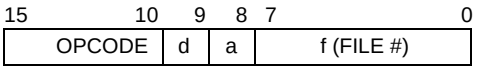


FIGURE 19-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations

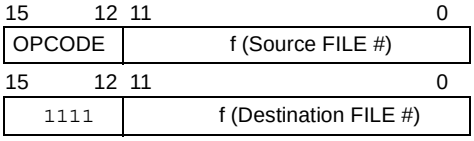
Example Instruction



ADDWF MYREG, W, B

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

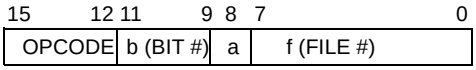
Byte to Byte move operations (2-word)



MOVFF MYREG1, MYREG2

f = 12-bit file register address

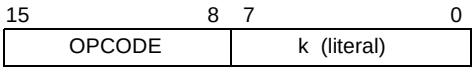
Bit-oriented file register operations



BSF MYREG, bit, B

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Literal operations

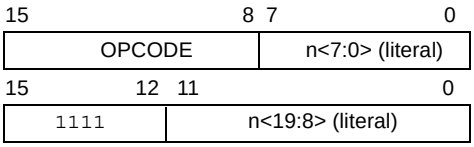


MOVLW 0x7F

k = 8-bit immediate value

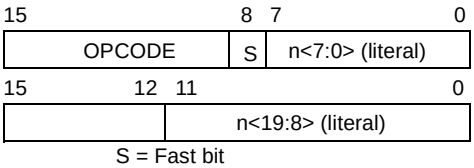
Control operations

CALL, GOTO and Branch operations



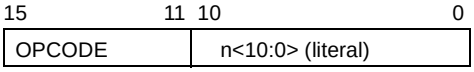
GOTO Label

n = 20-bit immediate value

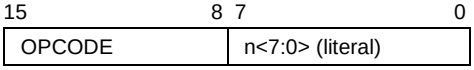


CALL MYFUNC

S = Fast bit



BRA MYFUNC



BC MYFUNC

ANDWF **AND WREG with f**

Syntax: [*label*] ANDWF f [,d [,a]

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

Operation: (WREG) .AND. (f) → dest

Status Affected: N,Z

Encoding:

0001	01da	ffff	ffff

Description: The contents of WREG are AND'ed with register 'f'. If 'd' is 0, the result is stored in WREG. If 'd' is 1, the result is stored back in register 'f' (default). If 'a' is 0, the Access Bank will be selected. If 'a' is 1, the BSR will not be overridden (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ANDWF REG, 0, 0`

Before Instruction

```
WREG = 0x17
REG  = 0xC2
```

After Instruction

```
WREG = 0x02
REG  = 0xC2
```

BC **Branch if Carry**

Syntax: [*label*] BC n

Operands: $-128 \le n \le 127$

Operation: if carry bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0010	nnnn	nnnn

Description: If the Carry 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 BC 5`

Before Instruction

```
PC = address (HERE)
```

After Instruction

```
If Carry = 1;
   PC    = address (HERE+12)
If Carry = 0;
   PC    = address (HERE+2)
```

CLRF **Clear f**

Syntax: [*label*] CLRF f [,a]

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

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: CLRF FLAG_REG,1

Before Instruction
FLAG_REG = 0x5A
After Instruction
FLAG_REG = 0x00

CLRWDT **Clear Watchdog Timer**

Syntax: [*label*] CLRWDT

Operands: None

Operation: 000h → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: CLRWDT

Before Instruction
WDT counter = ?
After Instruction
WDT counter = 0x00
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

INCFSZ **Increment f, skip if 0**

Syntax: [*label*] INCFSZ f [,d [,a]

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

Operation: (f) + 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0011	11da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f' (default).
If the result is 0, the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

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	Write to destination

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    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```

Before Instruction
```
PC      =   Address (HERE)
```
After Instruction
```
CNT     =   CNT + 1
If CNT  =   0;
    PC  =   Address(ZERO)
If CNT  ≠   0;
    PC  =   Address(NZERO)
```

INFSNZ **Increment f, skip if not 0**

Syntax: [*label*] INFSNZ f [,d [,a]

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

Operation: (f) + 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f' (default).
If the result is not 0, the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

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	Write to destination

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    INFSNZ  REG, 1, 0
ZERO
NZERO
```

Before Instruction
```
PC      =   Address (HERE)
```
After Instruction
```
REG     =   REG + 1
If REG  ≠   0;
    PC  =   Address (NZERO)
If REG  =   0;
    PC  =   Address (ZERO)
```

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL,
if s = 1
(WS) → WREG,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged.

Status Affected: GIE/GIEH,PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from Interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers, WREG, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	pop PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example:

```
RETFIE  1
```

After Interrupt

```
PC                  =    TOS
W                   =    WS
BSR                 =    BSRS
STATUS              =    STATUSS
GIE/GIEH, PEIE/GIEL=     1
```

RETLW **Return Literal to WREG**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → WREG,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	1100	kkkk	kkkk

Description: WREG is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	pop PC from stack, Write to WREG
No operation	No operation	No operation	No operation

Example:

```
   CALL TABLE  ; WREG contains table
               ; offset value
               ; WREG now has
               ; table value
   :
TABLE
   ADDWF PCL   ; WREG = offset
   RETLW k0    ; Begin table
   RETLW k1    ;
   :
   :
   RETLW kn    ; End of table
```

Before Instruction

```
WREG    =   0x07
```

After Instruction

```
WREG    =   value of kn
```

FIGURE 21-1: PIC18CXX2 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL, EXTENDED)

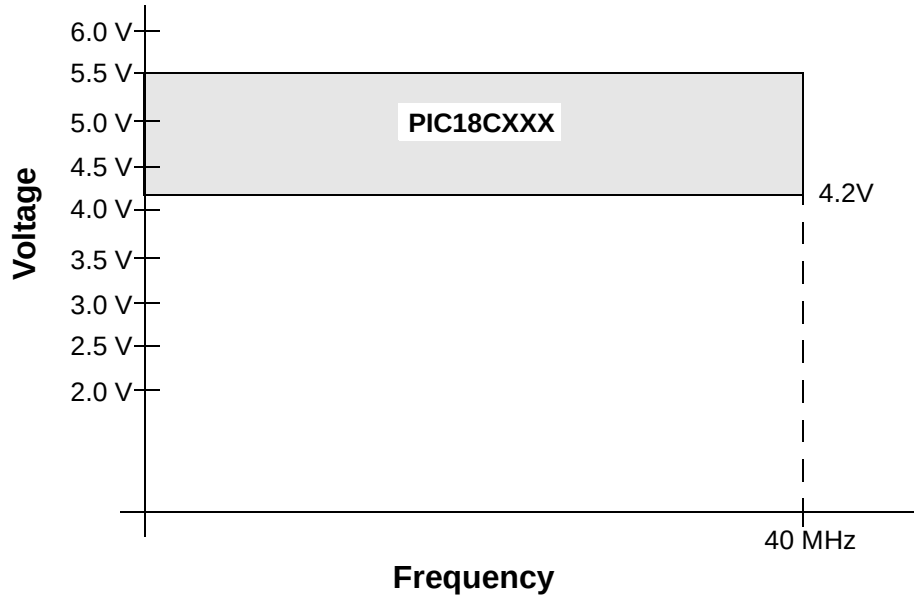
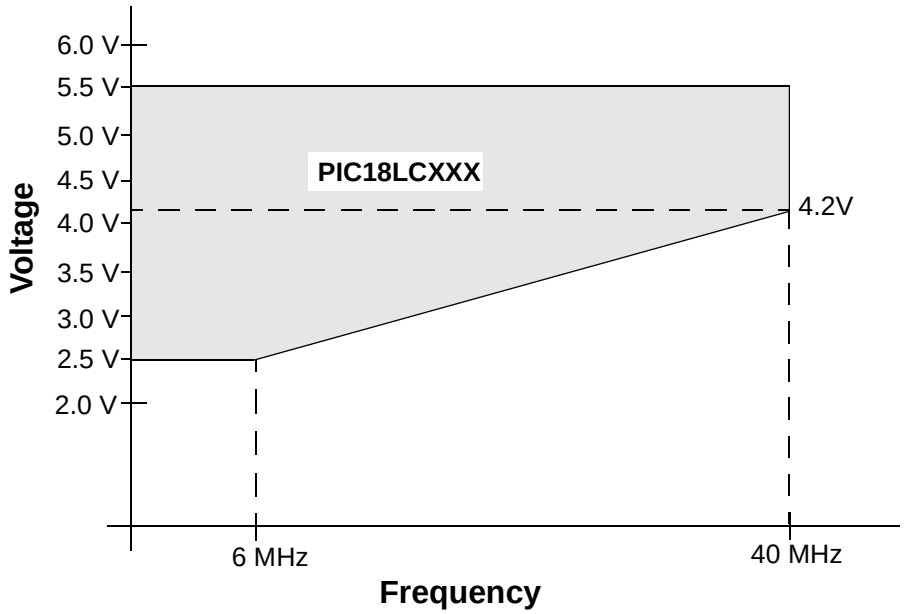


FIGURE 21-2: PIC18LCXX2 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



F_{MAX} = (20.0 MHz/V) ($V_{DDAPPMIN}$ - 2.5 V) + 6 MHz

Note: $V_{DDAPPMIN}$ is the minimum voltage of the PIC® device in the application.

21.2 DC Characteristics: PIC18CXX2 (Industrial, Extended) PIC18LCXX2 (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		I/O ports:				
D030		with TTL buffer	Vss	0.15VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger buffer	Vss	0.2VDD	V	
		RC3 and RC4	Vss	0.3VDD	V	
D032		$\overline{MCLR}$	Vss	0.2VDD	V	
D032A		OSC1 (in XT, HS and LP modes) and T1OSI	Vss	0.3VDD	V	
D033		OSC1 (in RC and EC mode)(1)	Vss	0.2VDD	V	
	VIH	**Input High Voltage**				
		I/O ports:				
D040		with TTL buffer	0.25VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		with Schmitt Trigger buffer	0.8VDD	VDD	V	
		RC3 and RC4	0.7VDD	VDD	V	
D042		$\overline{MCLR}$, OSC1 (EC mode)	0.8VDD	VDD	V	
D042A		OSC1 (in XT, HS and LP modes) and T1OSI	0.7VDD	VDD	V	
D043		OSC1 (RC mode)(1)	0.9VDD	VDD	V	
	IIL	**Input Leakage Current(2,3)**				
D060		I/O ports	—	±1	µA	VSS ≤ VPIN ≤ VDD, Pin at hi-impedance
D061		$\overline{MCLR}$	—	±5	µA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	±5	µA	Vss ≤ VPIN ≤ VDD
	IPU	**Weak Pull-up Current**				
D070	IPURB	PORTB weak pull-up current	50	400	µA	VDD = 5V, VPIN = VSS

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC MCU be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

21.2 DC Characteristics: PIC18CXX2 (Industrial, Extended) PIC18LCXX2 (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**				
D080		I/O ports	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C
D080A			—	0.6	V	IOL = 7.0 mA, VDD = 4.5V, -40°C to +125°C
D083		OSC2/CLKOUT (RC mode)	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C to +85°C
D083A			—	0.6	V	IOL = 1.2 mA, VDD = 4.5V, -40°C to +125°C
	VOH	**Output High Voltage(3)**				
D090		I/O ports	VDD - 0.7	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C to +85°C
D090A			VDD - 0.7	—	V	IOH = -2.5 mA, VDD = 4.5V, -40°C to +125°C
D092		OSC2/CLKOUT (RC mode)	VDD - 0.7	—	V	IOH = -1.3 mA, VDD = 4.5V, -40°C to +85°C
D092A			VDD - 0.7	—	V	IOH = -1.0 mA, VDD = 4.5V, -40°C to +125°C
D150	VOD	**Open Drain High Voltage**	—	8.5	V	RA4 pin
		Capacitive Loading Specs on Output Pins				
D101	CIO	All I/O pins and OSC2 (in RC mode)	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	In I^2C mode

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC MCU be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

FIGURE 22-21: TYPICAL, MINIMUM AND MAXIMUM V_{OH} vs. I_{OH} (V_{DD} = 3V, -40°C TO +125°C)

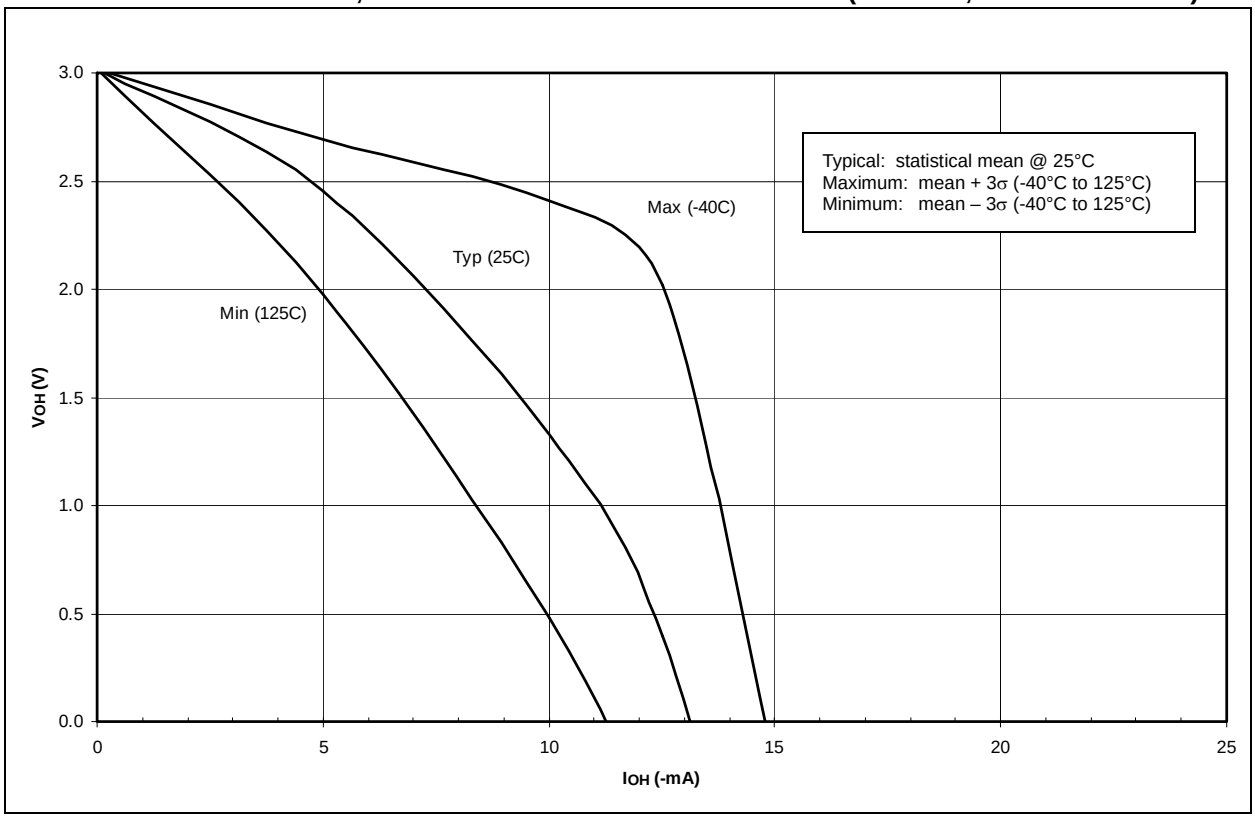


FIGURE 22-22: TYPICAL AND MAXIMUM V_{OL} vs. I_{OL} (V_{DD} = 5V, -40°C TO +125°C)