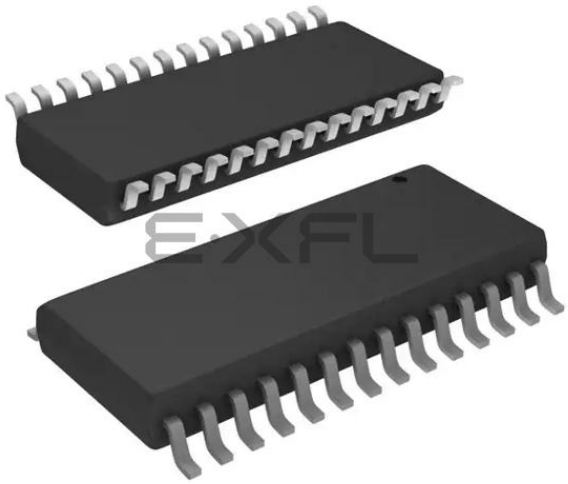


Welcome to E-XFL.COM

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	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	24
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2331-i-so

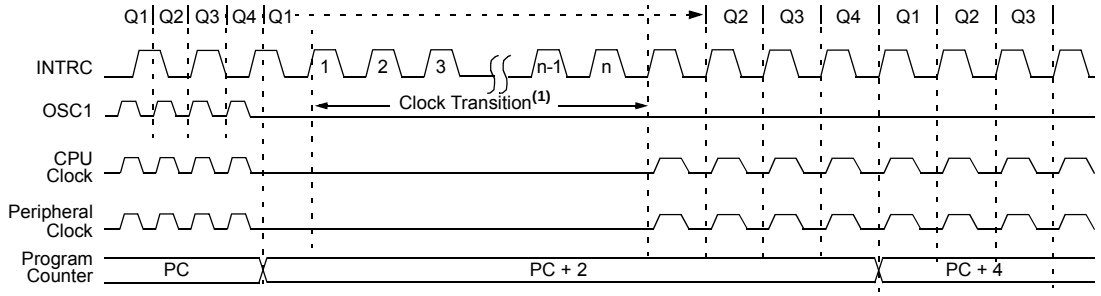
If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the device clocks.

If the IRCF bits are changed from all clear (thus, enabling the INTOSC output), or if INTSRC is set, the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the device continue while the INTOSC source stabilizes, after an interval of TIOBST.

If the IRCF bits were previously at a non-zero value, or if INTSRC was set before setting SCS1 and the INTOSC source was already stable, the IOFS bit will remain set.

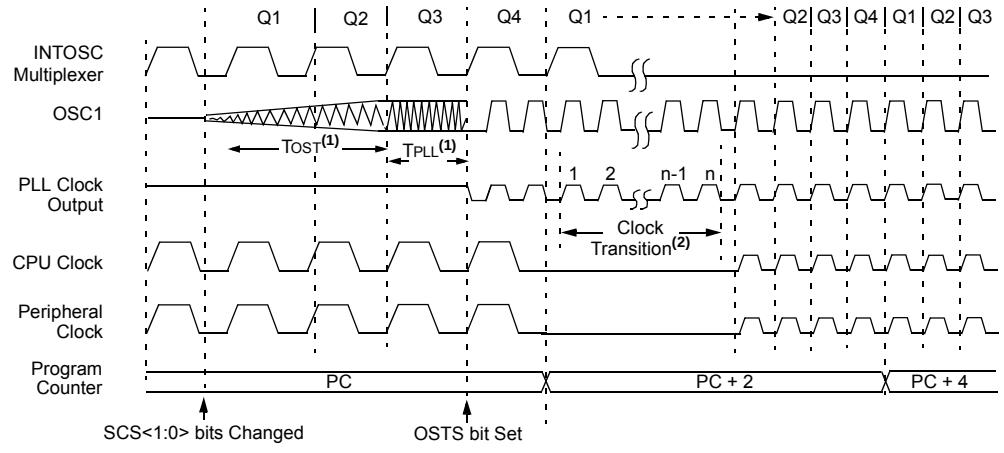
On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 4-4). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

FIGURE 4-3: TRANSITION TIMING TO RC_RUN MODE



Note 1: Clock transition typically occurs within 2-4 TOSC.

FIGURE 4-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.
2: Clock transition typically occurs within 2-4 TOSC.

6.0 MEMORY ORGANIZATION

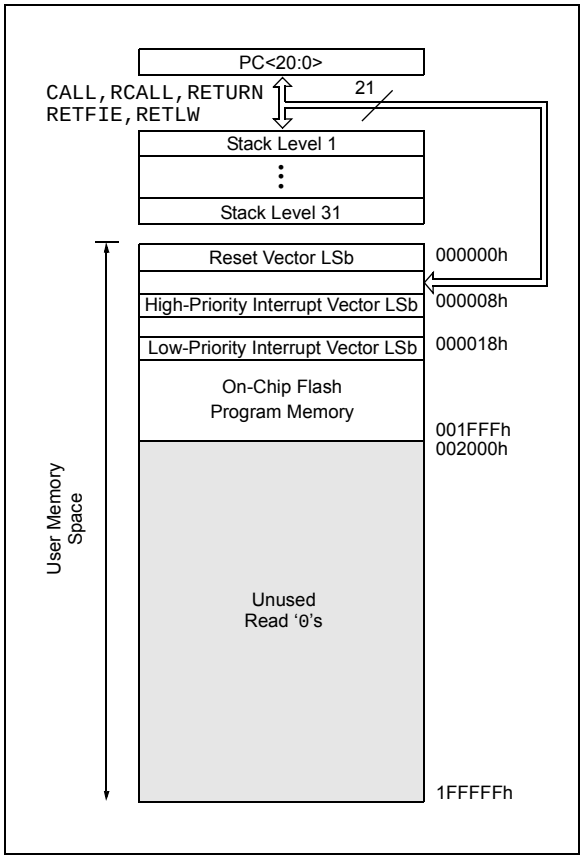
There are three memory types in enhanced MCU devices. These memory types are:

- Program Memory
- Data RAM
- Data EEPROM

As Harvard architecture devices, the data and program memories use separate buses, enabling concurrent access of the two memory spaces. The data EEPROM, for practical purposes, can be regarded as a peripheral device, since it is addressed and accessed through a set of control registers.

Additional detailed information on the operation of the Flash program memory is provided in **Section 8.0 "Flash Program Memory"**. Data EEPROM is discussed separately in **Section 7.0 "Data EEPROM Memory"**.

FIGURE 6-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F2331/4331



6.1 Program Memory Organization

PIC18 microcontrollers implement a 21-bit program counter that can address a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all '0's (a NOP instruction).

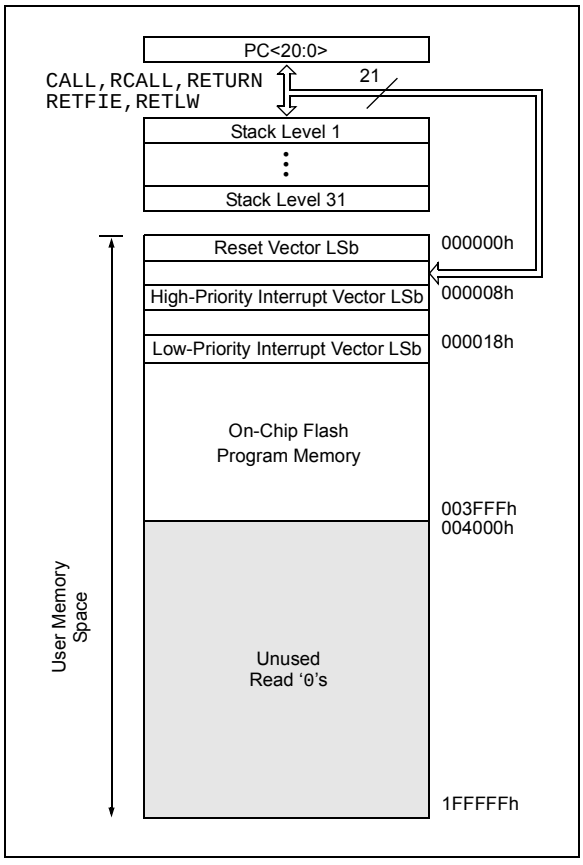
The PIC18F2331/4331 devices each have 8 Kbytes of Flash memory and can store up to 4,096 single-word instructions.

The PIC18F2431/4431 devices each have 16 Kbytes of Flash memory and can store up to 8,192 single-word instructions.

PIC18 devices have two interrupt vectors. The Reset vector address is at 000000h and the interrupt vector addresses are at 000008h and 000018h.

The program memory maps for PIC18F2331/4331 and PIC18F2431/4431 devices are shown in Figure 6-1 and Figure 6-2, respectively.

FIGURE 6-2: PROGRAM MEMORY MAP AND STACK FOR PIC18F2431/4431



7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADR register, clear the EEPGD control bit (EECON1<7>) and then set control bit, RD (EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation, or until it is written to by the user (during a write operation).The basic process is shown in Example 7-1.

7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register. The sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not begin if this sequence is not exactly followed (write 55h to EECON2, write 0AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution (i.e., runaway programs). The WREN bit should be kept clear at all times, except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, EECON1, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. The WREN bit must be set on a previous instruction. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Interrupt Flag bit (EEIF) is set. The user may either enable this interrupt or poll this bit. EEIF must be cleared by software.

7.5 Write Verify

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

7.6 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

EXAMPLE 7-1: DATA EEPROM READ

```
        MOVLW   DATA_EE_ADDR      ;
        MOVWF   EEADR             ; Data Memory Address to read
        BCF     EECON1, EEPGD     ; Point to DATA memory
        BSF     EECON1, RD        ; EEPROM Read
        MOVF    EEDATA, W         ; W = EEDATA
```

EXAMPLE 7-2: DATA EEPROM WRITE

```
                MOVLW   DATA_EE_ADDR      ;
                MOVWF   EEADR             ; Data Memory Address to write
                MOVLW   DATA_EE_DATA      ;
                MOVWF   EEDATA            ; Data Memory Value to write
                BCF     EECON1, EEPGD     ; Point to DATA memory
                BCF     EECON1, CFGS      ; Access EEPROM
                BSF     EECON1, WREN      ; Enable writes
                BCF     INTCON, GIE       ; Disable Interrupts
                MOVLW   55h               ;
Required        MOVWF   EECON2            ; Write 55h
Sequence        MOVLW   0AAh              ;
                MOVWF   EECON2            ; Write 0AAh
                BSF     EECON1, WR        ; Set WR bit to begin write
                BTFSC   EECON1, WR        ; Wait for write to complete

                GOTO    $-2               ;
                BSF     INTCON, GIE       ; Enable interrupts
```

7.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect bits in Configuration Words. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal data EEPROM, regardless of the state of the code-protect Configuration bit. Refer to **Section 23.0 "Special Features of the CPU"** for additional information.

7.8 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been implemented. On power-up, the WREN bit is cleared. In addition, writes to the EEPROM memory are blocked during the Power-up Timer period (TPWRT, Parameter 33).

The write/initiate sequence, and the WREN bit together, help prevent an accidental write during Brown-out Reset, power glitch or software malfunction.

7.9 Using the Data EEPROM

The data EEPROM is a high-endurance, byte-addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than Specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 7-3.

Note:	If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See Specification D124.

EXAMPLE 7-3: DATA EEPROM REFRESH ROUTINE

```
            CLRF    EEADR           ; Start at address 0
            BCF     EECON1, CFGS    ; Set for memory
            BCF     EECON1, EEPGD   ; Set for Data EEPROM
            BCF     INTCON, GIE     ; Disable interrupts
            BSF     EECON1, WREN    ; Enable writes
LOOP                                ; Loop to refresh array
            BSF     EECON1, RD      ; Read current address
            MOVLW   55h             ;
            MOVWF   EECON2          ; Write 55h
Required    MOVLW   0AAh            ;
Sequence    MOVWF   EECON2          ; Write 0AAh
            BSF     EECON1, WR      ; Set WR bit to begin write
            BTFSC   EECON1, WR      ; Wait for write to complete
            BRA     $-2
            INCFSZ  EEADR, F        ; Increment address
            BRA     LOOP            ; Not zero, do it again

            BCF     EECON1, WREN    ; Disable writes
            BSF     INTCON, GIE     ; Enable interrupts
```

10.0 INTERRUPTS

The PIC18F2331/2431/4331/4431 devices have multiple interrupt sources and an interrupt priority feature that allows each interrupt source to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 000008h and the low-priority interrupt vector is at 000018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

There are thirteen registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3
- PIE1, PIE2, PIE3
- IPR1, IPR2, IPR3

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, each interrupt source has three bits to control its operation. The functions of these bits are:

- Flag bit to indicate that an interrupt event occurred
- Enable bit that allows program execution to branch to the interrupt vector address when the flag bit is set
- Priority bit to select high priority or low priority (most interrupt sources have priority bits)

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 000008h in Compatibility mode.

When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High-priority interrupt sources can interrupt a low-priority interrupt. Low-priority interrupts are not processed while high-priority interrupts are in progress.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INTx pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

Note:	Do not use the `MOVFF` instruction to modify any of the Interrupt Control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

10.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Request (Flag) Registers (PIR1, PIR2 and PIR3).

Note 1: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE (INTCON<7>).

2: User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

REGISTER 10-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1

U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **ADIF:** A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5 **RCIF:** EUSART Receive Interrupt Flag bit
1 = The EUSART receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The EUSART receive buffer is empty

bit 4 **TXIF:** EUSART Transmit Interrupt Flag bit
1 = The EUSART transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The EUSART transmit buffer is full

bit 3 **SSPIF:** Synchronous Serial Port Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

bit 1 **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

14.0 TIMER2 MODULE

The Timer2 module has the following features:

- 8-bit Timer register (TMR2)
- 8-bit Period register (PR2)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match with PR2
- SSP module optional use of TMR2 output to generate clock shift

Timer2 has a control register, shown in Register 14-1. TMR2 can be shut off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption. Figure 14-1 is a simplified block diagram of the Timer2 module. Register 14-1 shows the Timer2 Control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

14.1 Timer2 Operation

Timer2 can be used as the PWM time base for the PWM mode of the CCP module. The TMR2 register is readable and writable, and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T2CKPS<1:0> (T2CON<1:0>). The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt, latched in flag bit, TMR2IF (PIR1<1>).

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, while the PR2 register initializes at FFh.

The prescaler and postscaler counters are cleared when any of the following occurs:

- A write to the TMR2 register
- A write to the T2CON register
- Any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 14-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

TABLE 15-1: REGISTERS ASSOCIATED WITH TIMER5

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	56
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	56
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	56
TMR5H	Timer5 Register High Byte								57
TMR5L	TImer5 Register Low Byte								57
PR5H	Timer5 Period Register High Byte								57
PR5L	Timer5 Period Register Low Byte								57
T5CON	T5SEN	$\overline{\text{RESEN}}$	T5MOD	T5PS1	T5PS0	$\overline{\text{T5SYNC}}$	TMR5CS	TMR5ON	56
CAP1CON	—	CAP1REN	—	—	CAP1M3	CAP1M2	CAP1M1	CAP1M0	59
DFLTCON	—	FLT4EN	FLT3EN	FLT2EN	FLT1EN	FLTCK2	FLTCK1	FLTCK0	59

Legend: — = unimplemented. Shaded cells are not used by the Timer5 module.

17.2.2 QEI MODES

Position measurement resolution depends on how often the Position Counter register, POSCNT, is incremented. There are two QEI Update modes to measure the rotor's position: QEI x2 and QEI x4.

TABLE 17-4: QEI MODES

QEIM<2:0>	Mode/ Reset	Description
000	—	QEI disabled.(1)
001	x2 update/ index pulse	Two clocks per QEA pulse. INDX resets POSCNT.
010	x2 update/ period match	Two clocks per QEA pulse. POSCNT is reset by the period match (MAXCNT).
011	—	Unused.
100	—	Unused.
101	x4 update/ index pulse	Four clocks per QEA and QEB pulse pair. INDX resets POSCNT.
110	x4 update/ period match	Four clocks per QEA and QEB pulse pair. POSCNT is reset by the period match (MAXCNT).
111	—	Unused.

Note 1: QEI module is disabled. The position counter and the velocity measurement functions are fully disabled in this mode.

17.2.2.1 QEI x2 Update Mode

QEI x2 Update mode is selected by setting the QEI Mode Select bits (QEIM<2:0>) to '001' or '010'. In this mode, the QEI logic detects every edge on the QEA input only. Every rising and falling edge on the QEA signal clocks the position counter.

The position counter can be reset by either an input on the INDX pin (QEIM<2:0> = 001), or by a period match, even when the POSCNT register pair equals MAXCNT (QEIM<2:0> = 010).

17.2.2.2 QEI x4 Update Mode

QEI x4 Update mode provides for a finer resolution of the rotor position, since the counter increments or decrements more frequently for each QEA/QEB input pulse pair than in QEI x2 mode. This mode is selected by setting the QEI mode select bits to '101' or '110'. In QEI x4, the phase measurement is made on the rising and the falling edges of both QEA and QEB inputs. The position counter is clocked on every QEA and QEB edge.

Like QEI x2 mode, the position counter can be reset by an input on the pin (QEIM<2:0> = 101), or by the period match event (QEIM<2:0> = 010).

17.2.3 QEI OPERATION

The Position Counter register pair (POSCNTH: POSCNTL) acts as an integrator, whose value is proportional to the position of the sensor rotor that corresponds to the number of active edges detected. POSCNT can either increment or decrement, depending on a number of selectable factors which are decoded by the QEI logic block. These include the Count mode selected, the phase relationship of QEA to QEB ("lead/lag"), the direction of rotation and if a Reset event occurs. The logic is detailed in the sections that follow.

17.2.3.1 Edge and Phase Detect

In the first step, the active edges of QEA and QEB are detected, and the phase relationship between them is determined. The position counter is changed based on the selected QEI mode.

In QEI x2 Update mode, the position counter increments or decrements on every QEA edge based on the phase relationship of the QEA and QEB signals.

In QEI x4 Update mode, the position counter increments or decrements on every QEA and QEB edge based on the phase relationship of the QEA and QEB signals. For example, if QEA leads QEB, the position counter is incremented by '1'. If QEB lags QEA, the position counter is decremented by '1'.

17.2.3.2 Direction of Count

The QEI control logic generates a signal that sets the UP/$\overline{\text{DOWN}}$ bit (QEICON<5>); this, in turn, determines the direction of the count. When QEA leads QEB, UP/$\overline{\text{DOWN}}$ is set (= 1) and the position counter increments on every active edge. When QEA lags QEB, UP/$\overline{\text{DOWN}}$ is cleared and the position counter decrements on every active edge.

TABLE 17-5: DIRECTION OF ROTATION

Current Signal Detected	Previous Signal Detected				Pos. Cntrl.(1)
	Rising		Falling		
	QEA	QEB	QEA	QEB	
QEA Rising				x	INC
		x			DEC
QEA Falling				x	DEC
		x			INC
QEB Rising	x				INC
			x		DEC
QEB Falling			x		INC
	x				DEC

Note 1: When UP/$\overline{\text{DOWN}}$ = 1, the position counter is incremented. When UP/$\overline{\text{DOWN}}$ = 0, the position counter is decremented.

17.3 Noise Filters

The Motion Feedback Module includes three noise rejection filters on RA2/AN2/VREF-/CAP1/INDX, RA3/AN3/VREF+/CAP2/QEA and RA4/AN4/CAP3/QEB. The filter block also includes a fourth filter for the T5CKI pin. They are intended to help reduce spurious noise spikes which may cause the input signals to become corrupted at the inputs. The filter ensures that the input signals are not permitted to change until a stable value has been registered for three consecutive sampling clock cycles.

The filters are controlled using the Digital Filter Control (DFLTCON) register (see Register 17-3). The filters can be individually enabled or disabled by setting or clearing the corresponding FLTxEN bit in the DFLTCON register. The sampling frequency, which must be the same for all three noise filters, can be programmed by the FLTCK<2:0> Configuration bits. TCY is used as the clock reference to the clock divider block.

The noise filters can either be added or removed from the input capture, or QEI signal path, by setting or clearing the appropriate FLTxEN bit, respectively. Each capture channel provides for individual enable control of the filter output. The FLT4EN bit enables or disables the noise filter available on the T5CKI input in the Timer5 module.

The filter network for all channels is disabled on Power-on and Brown-out Resets, as the DFLTCON register is cleared on Resets. The operation of the filter is shown in the timing diagram in Figure 17-14.

REGISTER 17-3: DFLTCON: DIGITAL FILTER CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	FLT4EN	FLT3EN(1)	FLT2EN(1)	FLT1EN(1)	FLTCK2	FLTCK1	FLTCK0
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **FLT4EN:** Noise Filter Output Enable bit (T5CKI input)
1 = Enabled
0 = Disabled

bit 5 **FLT3EN:** Noise Filter Output Enable bit (CAP3/QEB input)(1)
1 = Enabled
0 = Disabled

bit 4 **FLT2EN:** Noise Filter Output Enable bit (CAP2/QEA input)(1)
1 = Enabled
0 = Disabled

bit 3 **FLT1EN:** Noise Filter Output Enable bit (CAP1/INDX Input)(1)
1 = Enabled
0 = Disabled

bit 2-0 **FLTCK<2:0>:** Noise Filter Clock Divider Ratio bits
111 = Unused
110 = 1:128
101 = 1:64
100 = 1:32
011 = 1:16
010 = 1:4
001 = 1:2
000 = 1:1

Note 1: The noise filter output enables are functional in both QEI and IC Operating modes.

Note: The noise filter is intended for random high-frequency filtering and not continuous high-frequency filtering.

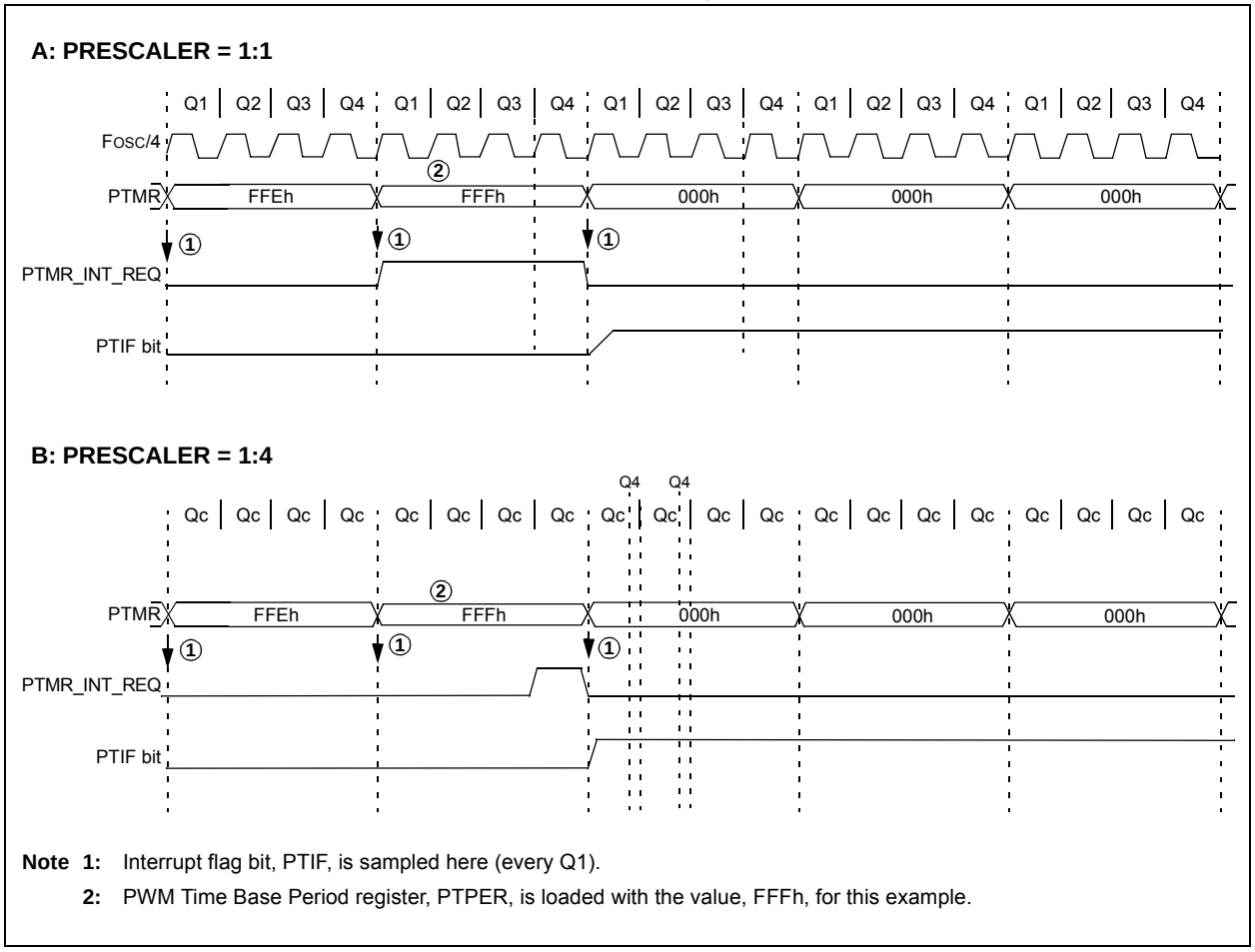
18.4.2 INTERRUPTS IN SINGLE-SHOT MODE

When the PWM time base is in the Single-Shot mode (PTMOD<1:0> = 01), an interrupt event is generated when a match with the PTPER register occurs. The PWM Time Base register (PTMR) is reset to zero on the following input clock edge and the PTEN bit is cleared. The postscaler selection bits have no effect in this Timer mode.

18.4.3 INTERRUPTS IN CONTINUOUS UP/DOWN COUNT MODE

In the Continuous Up/Down Count mode (PTMOD<1:0> = 10), an interrupt event is generated each time the value of the PTMR register becomes zero and the PWM time base begins to count upwards. The postscaler selection bits may be used in this mode of the timer to reduce the frequency of the interrupt events. Figure 18-7 shows the interrupts in Continuous Up/Down Count mode.

FIGURE 18-6: PWM TIME BASE INTERRUPT TIMING, SINGLE-SHOT MODE



Note 1: Interrupt flag bit, PTIF, is sampled here (every Q1).

2: PWM Time Base Period register, PTPER, is loaded with the value, FFFh, for this example.

18.11 PWM Output and Polarity Control

There are three device Configuration bits associated with the PWM module that provide PWM output pin control defined in the CONFIG3L Configuration register. They are:

- HPOL
- LPOL
- PWMPIN

These three Configuration bits work in conjunction with the three PWM Enable bits (PWMEN<2:0>) in the PWMCON0 register. The Configuration bits and PWM enable bits ensure that the PWM pins are in the correct states after a device Reset occurs.

18.11.1 OUTPUT PIN CONTROL

The PWMEN<2:0> control bits enable each PWM output pin as required in the application.

All PWM I/O pins are general purpose I/O. When a pair of pins are enabled for PWM output, the PORT and TRIS registers controlling the pins are disabled. Refer to Figure 18-23 for details.

18.11.2 OUTPUT POLARITY CONTROL

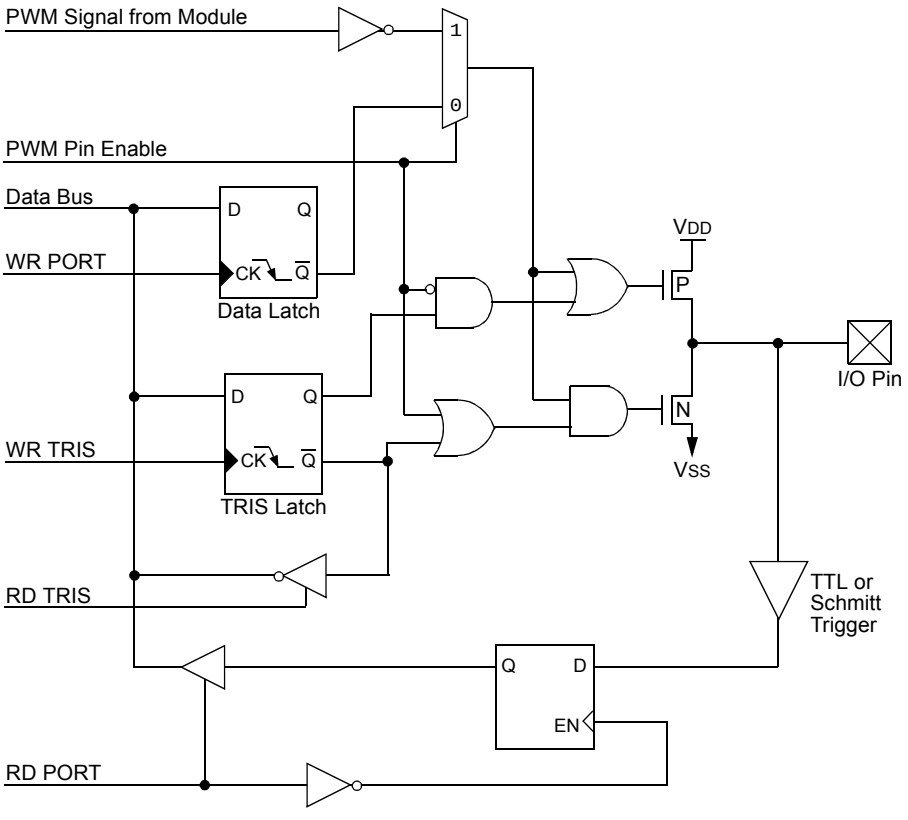
The polarity of the PWM I/O pins is set during device programming via the HPOL and LPOL Configuration bits in the CONFIG3L Configuration register. The HPOL Configuration bit sets the output polarity for the high side PWM outputs: PWM1, PWM3, PWM5 and PWM7. The polarity is active-low when HPOL is cleared (= 0), and active-high when it is set (= 1).

The LPOL Configuration bit sets the output polarity for the low side PWM outputs: PWM0, PWM2, PWM4 and PWM6. As with HPOL, they are active-low when LPOL is cleared and active-high when it is set.

All output signals generated by the PWM module are referenced to the polarity control bits, including those generated by Fault inputs or manual override (see **Section 18.10 "PWM Output Override"**).

The default polarity Configuration bits have the PWM I/O pins in active-high output polarity.

FIGURE 18-23: PWM I/O PIN BLOCK DIAGRAM



Note: I/O pin has protection diodes to VDD and VSS. PWM polarity selection logic not shown for clarity.

TABLE 18-6: REGISTERS ASSOCIATED WITH THE POWER CONTROL PWM MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	56
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	56
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	56
PTCON0	PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0	58
PTCON1	PTEN	PTDIR	—	—	—	—	—	—	58
PTMRL(1)	PWM Time Base Register (lower 8 bits)								58
PTMRH(1)	**UNUSED**				PWM Time Base Register (upper 4 bits)				58
PTPERL(1)	PWM Time Base Period Register (lower 8 bits)								58
PTPERH(1)	**UNUSED**				PWM Time Base Period Register (upper 4 bits)				58
SEVTCMPL(1)	PWM Special Event Compare Register (lower 8 bits)								58
SEVTCMPH(1)	**UNUSED**				PWM Special Event Compare Register (upper 4 bits)				58
PWMCON0	—	PWMEN2	PWMEN1	PWMEN0	PMOD3(2)	PMOD2	PMOD1	PMOD0	58
PWMCON1	SEVOPS3	SEVOPS2	SEVOPS1	SEVOPS0	SEVTDIR	—	UDIS	OSYNC	58
DTCON	DTPS1	DTPS0	DT5	DT4	DT3	DT2	DT1	DT0	58
FLTCONFIG	BRFEN	FLTBS(2)	FLTBMOD(2)	FLTBEN(2)	FLTCON	FLTAS	FLTAMOD	FLTAEN	58
OVDCOND	POVD7(2)	POVD6(2)	POVD5	POVD4	POVD3	POVD2	POVD1	POVD0	58
OVDCONS	POUT7(2)	POUT6(2)	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0	58
PDC0L(1)	PWM Duty Cycle #0L Register (lower 8 bits)								58
PDC0H(1)	**UNUSED**		PWM Duty Cycle #0H Register (upper 6 bits)						58
PDC1L(1)	PWM Duty Cycle #1L register (lower 8 bits)								58
PDC1H(1)	**UNUSED**		PWM Duty Cycle #1H Register (upper 6 bits)						58
PDC2L(1)	PWM Duty Cycle #2L Register (lower 8 bits)								58
PDC2H(1)	**UNUSED**		PWM Duty Cycle #2H Register (upper 6 bits)						58
PDC3L(1,2)	PWM Duty Cycle #3L Register (lower 8 bits)								58
PDC3H(1,2)	**UNUSED**		PWM Duty Cycle #3H Register (upper 6 bits)						58

Legend: — = Unimplemented, read as '0'. Shaded cells are not used with the power control PWM.

Note 1: Double-buffered register pairs. Refer to text for explanation of how these registers are read and written to.

2: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear. Reset values shown are for PIC18F4331/4431 devices.

TABLE 20-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

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

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

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

20.5.2 EUSART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep, or any Idle mode and bit SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this Low-Power mode. Once the word is received, the RSR register will transfer the data to the RCREG register. If the RCIE enable bit is set, the interrupt generated will wake the chip from Low-Power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. If interrupts are desired, set enable bit, RCIE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RCIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCIE, was set.
6. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit, CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 20-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
RCREG	EUSART Receive Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

24.2 Instruction Set

ADDLW **ADD Literal to W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

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

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal ‘k’ and 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: ADDLW 0x15

Before Instruction
W = 0x10
After Instruction
W = 0x25

ADDWF **ADD W to f**

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

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

Operation: $(W) + (f) \rightarrow$ dest

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

Encoding:

0010	01da	ffff	ffff

Description: Add W to register, ‘f’. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register, ‘f’. If ‘a’ is ‘0’, the Access Bank will be selected. If ‘a’ is ‘1’, the BSR is used.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ADDWF REG, W

Before Instruction
W = 0x17
REG = 0xC2
After Instruction
W = 0xD9
REG = 0xC2

MULLW **Multiply Literal with W**

Syntax: [*label*] MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal, 'k'. The 16-bit result is placed in PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged.
None of the Status flags are affected. Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0xC4`

Before Instruction
W = 0xE2
PRODH = ?
PRODL = ?

After Instruction
W = 0xE2
PRODH = 0xAD
PRODL = 0x08

MULWF **Multiply W with f**

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

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

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location, 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged.
None of the Status flags are affected. Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected. 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.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: `MULWF REG`

Before Instruction
W = 0xC4
REG = 0xB5
PRODH = ?
PRODL = ?

After Instruction
W = 0xC4
REG = 0xB5
PRODH = 0x8A
PRODL = 0x94

POP **Pop Top of Return Stack**

Syntax: [*label*] POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

0000	0000	0000	0110

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack. This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

Example:

```
POP
GOTO    NEW
```

Before Instruction
TOS = 0x0031A2
Stack (1 level down) = 0x014332

After Instruction
TOS = 0x014332
PC = NEW

PUSH **Push Top of Return Stack**

Syntax: [*label*] PUSH

Operands: None

Operation: (PC + 2) → TOS

Status Affected: None

Encoding:

0000	0000	0000	0101

Description: The PC + 2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows to implement a software stack by modifying TOS, and then push it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC + 2 onto return stack	No operation	No operation

Example:

```
PUSH
```

Before Instruction
TOS = 0x00345A
PC = 0x000124

After Instruction
PC = 0x000126
TOS = 0x000126
Stack (1 level down) = 0x00345A

FIGURE 26-10: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)

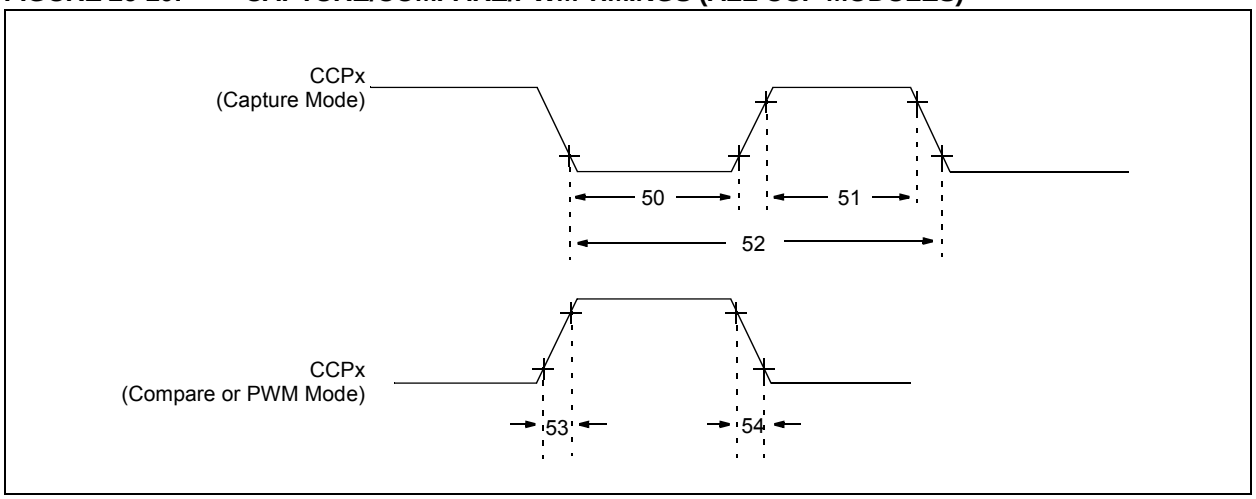


TABLE 26-10: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)

Param No.	Symbol	Characteristic			Min	Max	Units	Conditions
50	TccL	CCPx Input Low Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	20	—	ns	
51	TccH	CCPx Input High Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	20	—	ns	
52	TccP	CCPx Input Period			$\frac{3\ T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time		PIC18FXX31	—	25	ns	
				PIC18LFXX31	—	45	ns	
54	TccF	CCPx Output Fall Time		PIC18FXX31	—	25	ns	
				PIC18LFXX31	—	45	ns	

FIGURE 26-11: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)

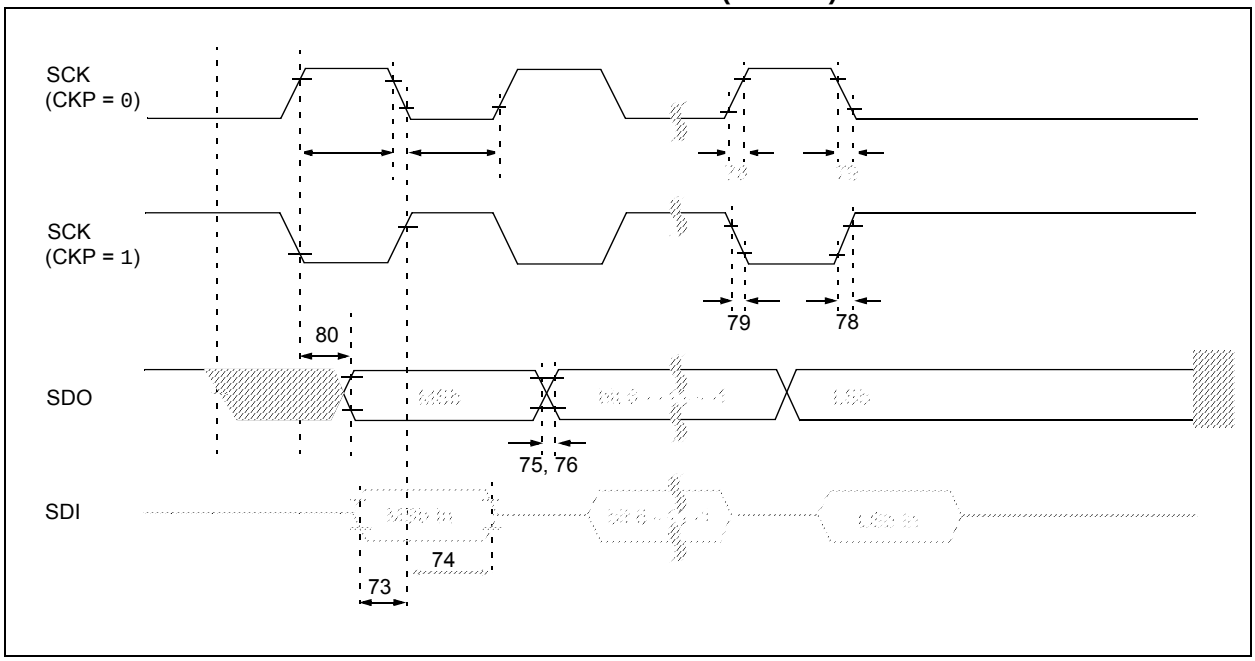


TABLE 26-11: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2		1.5 TCY + 40	—	ns	
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
78	TscR	SCK Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
79	TscF	SCK Output Fall Time		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	