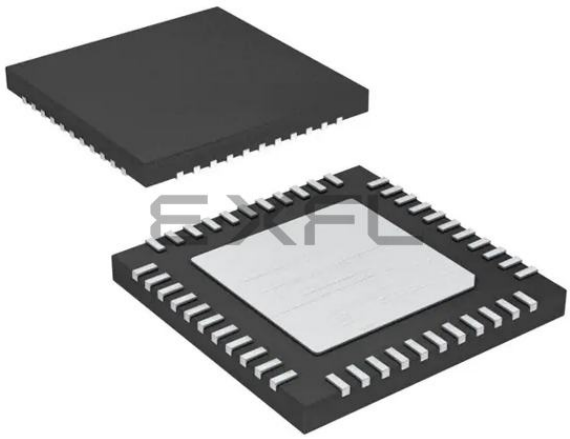


Welcome to E-XFL.COM

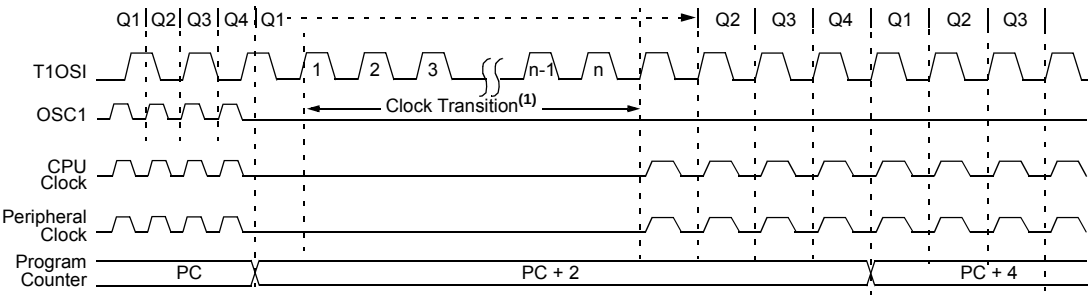
What is "Embedded - Microcontrollers"?

"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"

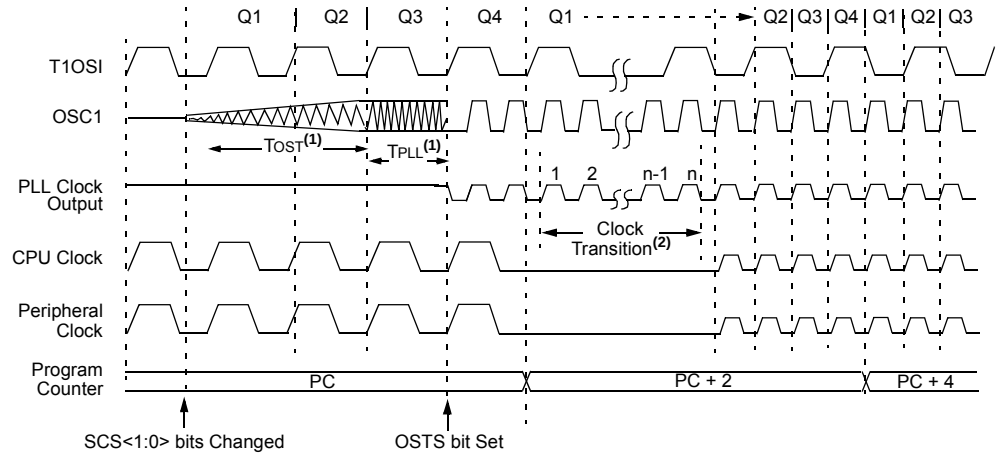
Details	
Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	36
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 9x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf4431t-i-ml

FIGURE 4-1: TRANSITION TIMING FOR ENTRY TO SEC_RUN MODE



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

FIGURE 4-2: TRANSITION TIMING FROM SEC_RUN MODE TO PRI_RUN MODE (HSPLL)



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.

4.2.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator block using the INTOSC multiplexer. In this mode, the primary clock is shut down. When using the INTRC source, this mode provides the best power conservation of all the Run modes, while still executing code. It works well for user applications which are not highly timing-sensitive or do not require high-speed clocks at all times.

If the primary clock source is the internal oscillator block (either INTRC or INTOSC), there are no distinguishable differences between PRI_RUN and RC_RUN modes during execution. However, a clock switch delay will occur during entry to and exit from RC_RUN mode. Therefore, if the primary clock source is the internal oscillator block, the use of RC_RUN mode is not recommended.

This mode is entered by setting the SCS1 bit to '1'. Although it is ignored, it is recommended that the SCS0 bit also be cleared; this is to maintain software compatibility with future devices. When the clock source is switched to the INTOSC multiplexer (see Figure 4-3), the primary oscillator is shut down and the OSTS bit is cleared. The IRCF bits may be modified at any time to immediately change the clock speed.

Note:	Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/FOSC specifications are violated.

6.4 Instructions in Program Memory

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). Figure 6-5 shows an example of how instruction words are stored in the program memory. To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '0'.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction 2 in Figure 6-5 shows how the instruction, 'GOTO 000006h', is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 24.0 "Instruction Set Summary"** provides further details of the instruction set.

6.4.1 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LSFR. In all cases, the second word of the instructions always has '1111' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the four MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence, immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 6-4 shows how this works.

Note:	For information on two-word instructions in the extended instruction set, see **Section 24.2 "Instruction Set".**

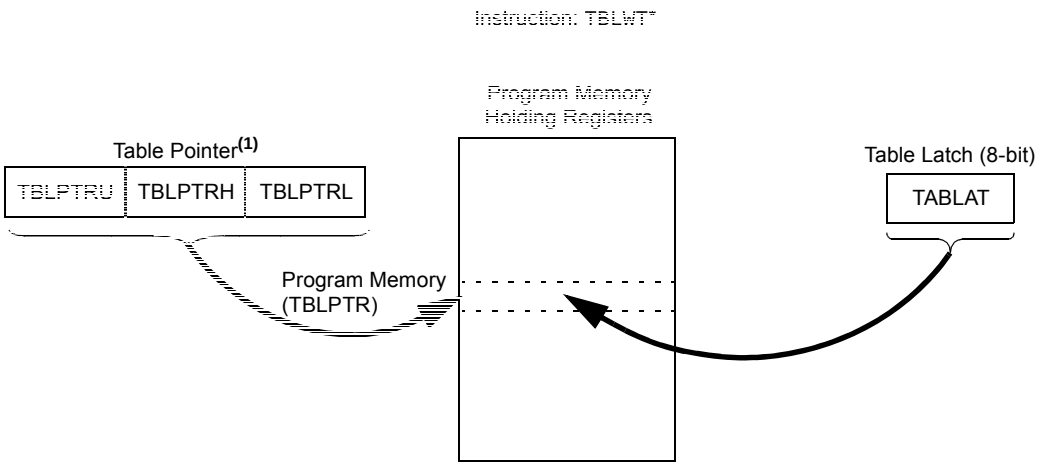
FIGURE 6-5: INSTRUCTIONS IN PROGRAM MEMORY

			LSB = 1	LSB = 0	Word Address ↓
	Program Memory Byte Locations →				000000h
					000002h
					000004h
					000006h
Instruction 1:	MOVLW	055h	0Fh	55h	000008h
Instruction 2:	GOTO	000006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

EXAMPLE 6-4: TWO-WORD INSTRUCTIONS

CASE 1:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
CASE 2:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

FIGURE 8-2: TABLE WRITE OPERATION



Note 1: The Table Pointer actually points to one of eight holding registers, the address of which is determined by TBLPTRL<2:0>. The process for physically writing data to the program memory array is discussed in Section 8.5 "Writing to Flash Program Memory".

8.2 Control Registers

Several control registers are used in conjunction with the TBLRD and TBLWT instructions. These include the:

- EECON1 register
- EECON2 register
- TABLAT register
- TBLPTR registers

8.2.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit, EEPGD, determines if the access will be to program or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit, CFGS, determines if the access will be to the Configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access Configuration registers, regardless of EEPGD. (See **Section 23.0 "Special Features of the CPU"**.) When CFGS is clear, the EEPGD bit selects either program Flash or data EEPROM memory.

The FREE bit controls program memory erase operations. When the FREE bit is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

A write operation is allowed when the WREN bit (EECON1<2>) is set. On power-up, the WREN bit is clear. The WRERR bit (EECON1<3>) is set in hardware when the WR bit (EECON1<1>) is set and cleared when the internal programming timer expires and the write operation is complete.

Note:	During normal operation, the WRERR may read as '1'. This can indicate that a write operation was prematurely terminated by a Reset or a write operation was attempted improperly.

The WR control bit initiates write operations. The bit cannot be cleared, only set, in software. The bit is cleared in hardware at the completion of the write operation.

Note:	The EEIF interrupt flag bit (PIR2<4>) is set when the write is complete. It must be cleared in software.

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers, RES<3:0>.

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES<3:0> &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

EXAMPLE 9-3: 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          ;
   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          ;
   ADDWFC RES3, F       ;
```

Example 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers, RES<3:0>. To account for the sign bits of the arguments, each argument pair's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES<3:0> &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H<7> \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H<7> \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

EXAMPLE 9-4: 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          ;
    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          ;
    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
    :
```

REGISTER 10-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

R/W-1	U-0	U-0	R/W-1	U-0	R/W-1	U-0	R/W-1
OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP
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 **OSCFIP:** Oscillator Fail Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **EEIP:** Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as '0'

bit 2 **LVDIP:** Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 0 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP
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 7-5 **Unimplemented:** Read as '0'

bit 4 **PTIP:** PWM Time Base Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **IC3DRIP:** IC3 Interrupt Priority/Direction Change Interrupt Priority bit
IC3 Enabled (CAP3CON<3:0>):
1 = IC3 interrupt high priority
0 = IC3 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = Change of direction interrupt high priority
0 = Change of direction interrupt low priority

bit 2 **IC2QEIP:** IC2 Interrupt Priority/QEI Interrupt Priority bit
IC2 Enabled (CAP2CON<3:0>):
1 = IC2 interrupt high priority
0 = IC2 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = High priority
0 = Low priority

bit 1 **IC1IP:** IC1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR5IP:** Timer5 Interrupt Priority bit
1 = High priority
0 = Low priority

11.0 I/O PORTS

Depending on the device selected and features enabled, there are up to five ports available. Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

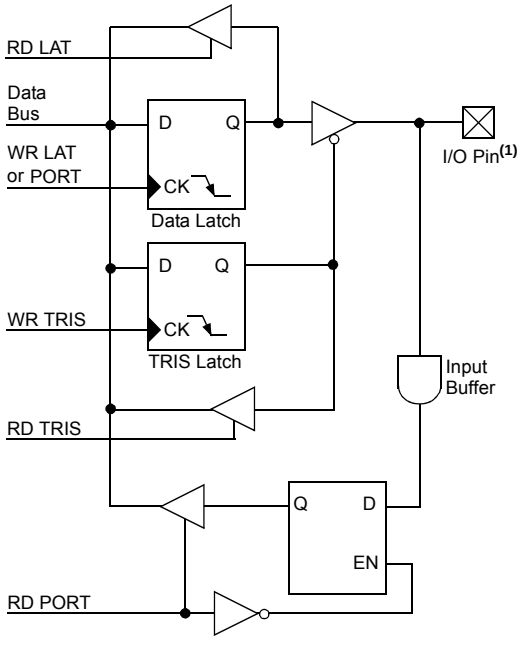
Each port has three registers for its operation. These registers are:

- TRIS register (Data Direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (Data Latch)

The Data Latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A simplified model of a generic I/O port without the interfaces to other peripherals is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



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

11.1 PORTA, TRISA and LATA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Reading the PORTA register reads the status of the pins, whereas writing to it, will write to the port latch.

The Data Latch register (LATA) is also memory mapped. Read-modify-write operations on the LATA register read and write the latched output value for PORTA.

The RA<4:2> pins are multiplexed with three input capture pins and Quadrature Encoder Interface pins. Pins, RA6 and RA7, are multiplexed with the main oscillator pins. They are enabled as oscillator or I/O pins by the selection of the main oscillator in Configuration Register 1H (see **Section 23.1 "Configuration Bits"** for details). When they are not used as port pins, RA6 and RA7 and their associated TRIS and LAT bits are read as '0'.

The other PORTA pins are multiplexed with analog inputs, the analog VREF+ and VREF- inputs and the comparator voltage reference output. The operation of pins RA<3:0> and RA5 as A/D Converter inputs is selected by clearing/setting the control bits in the ANSEL0 and ANSEL1 registers.

Note 1: On a Power-on Reset, RA<5:0> are configured as analog inputs and read as '0'.

2: RA5 I/F is available only on 40-pin devices (PIC18F4331/4431).

The TRISA register controls the direction of the RA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

EXAMPLE 11-1: INITIALIZING PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x3F    ; Configure A/D
MOVWF   ANSEL0  ; for digital inputs
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

17.2.3.3 Reset and Update Events

The position counter will continue to increment or decrement until one of the following events takes place. The type of event and the direction of rotation when it happens determines if a register Reset or update occurs.

1. An index pulse is detected on the INDX input (QEIM<2:0> = 001).

   If the encoder is traveling in the **forward** direction, POSCNT is reset (00h) on the next clock edge after the index marker, INDX, has been detected. The position counter resets on the QEA or QEB edge once the INDX rising edge has been detected.

   If the encoder is traveling in the **reverse** direction, the value in the MAXCNT register is loaded into POSCNT on the next quadrature pulse edge (QEA or QEB) after the falling edge on INDX has been detected.

2. A POSTCNT/MAXCNT period match occurs (QEIM<2:0> = 010).

   If the encoder is traveling in the **forward** direction, POSCNT is reset (00h) on the next clock edge when POSCNT = MAXCNT. An interrupt event is triggered on the next TCY after the Reset (see Figure 17-10)

   If the encoder is traveling in the **reverse** direction and the value of POSCNT reaches 00h, POSCNT is loaded with the contents of the MAXCNT register on the next clock edge. An interrupt event is triggered on the next TCY after the load operation (see Figure 17-10).

The value of the position counter is not affected during QEI mode changes, nor when the QEI is disabled altogether.

17.2.4 QEI INTERRUPTS

The position counter interrupt occurs and the interrupt flag (IC2QEIF) is set, based on the following events:

- A POSCNT/MAXCNT period match event (QEIM<2:0> = 010 or 110)
- A POSCNT rollover (FFFFh to 0000h) in Period mode only (QEIM<2:0> = 010 or 110)
- An index pulse detected on INDX

The interrupt timing diagrams for IC2QEIF are shown in Figure 17-10 and Figure 17-11.

When the direction has changed, the direction change interrupt flag (IC3DRIF) is set on the following TCY clock (see Figure 17-10).

If the position counter rolls over in Index mode, the QERR bit will be set.

17.2.5 QEI SAMPLE TIMING

The quadrature input signals, QEA and QEB, may vary in quadrature frequency. The minimum quadrature input period, TQEI, is 16 TCY.

The position count rate, FPOS, is directly proportional to the rotor's RPM, line count D and QEI Update mode (x2 versus x4); that is,

EQUATION 17-1:

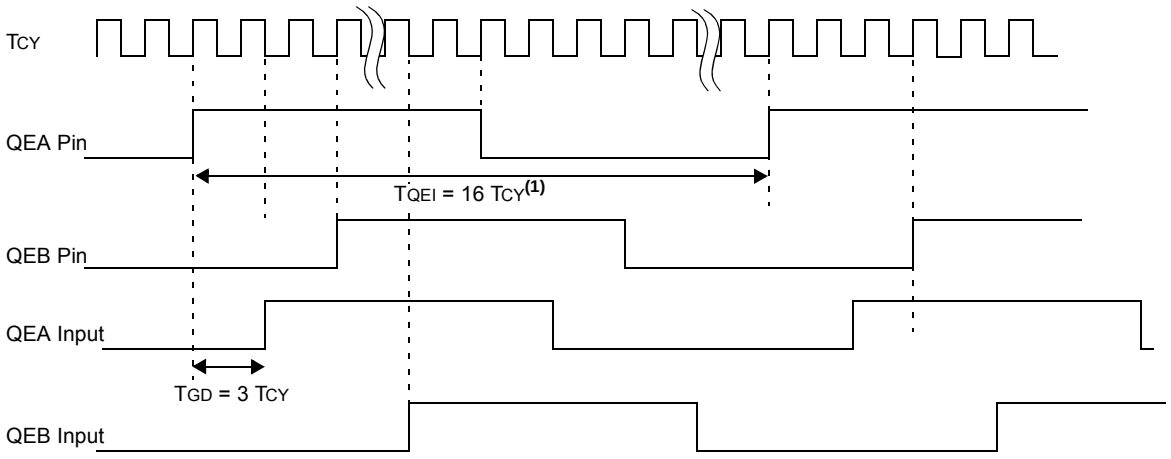
$$F_{POS} = \frac{4D \bullet RPM}{60}$$

Note:	The number of incremental lines in the position encoder is typically set at D = 1024 and the QEI Update mode = x4.

The maximum position count rate (i.e., x4 QEI Update mode, D = 1024) with FCY = 10 MIPS is equal to 2.5 MHz, which corresponds to FQEI of 625 kHz.

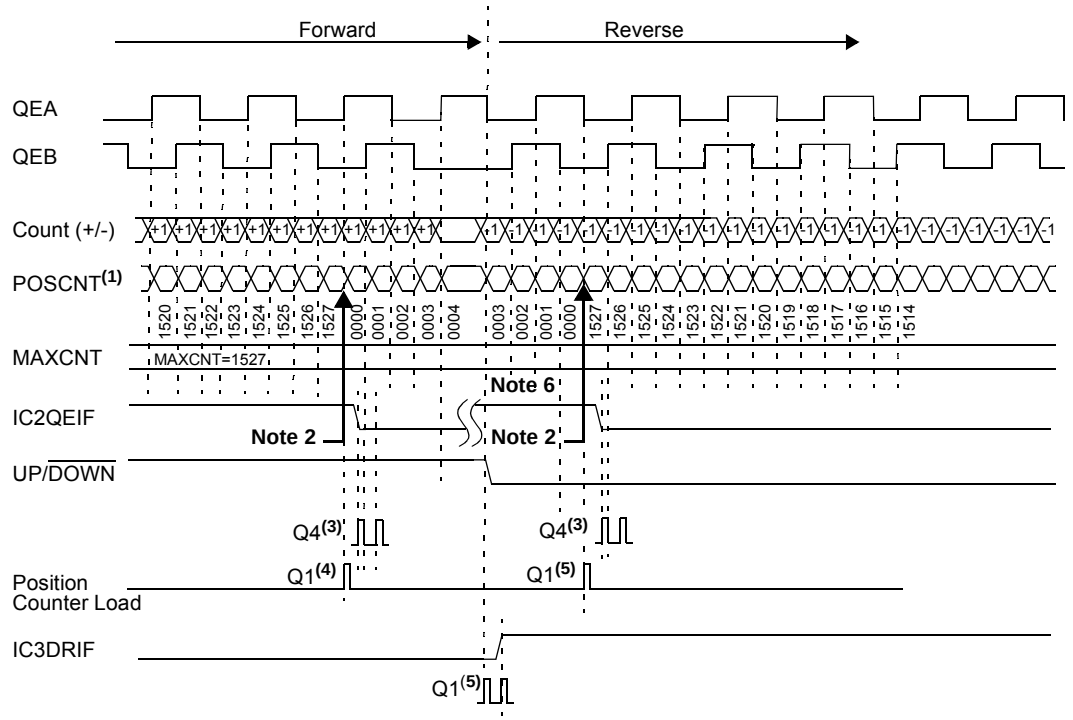
Figure 17-9 shows QEA and QEB quadrature input timing when sampled by the noise filter.

FIGURE 17-9: QEI INPUTS WHEN SAMPLED BY THE FILTER (DIVIDE RATIO = 1:1)



Note 1: The module design allows a quadrature frequency of up to FQEI = FCY/16.

FIGURE 17-10: QEI MODULE RESET TIMING ON PERIOD MATCH



Note 1: The POSCNT register is shown in QEI x4 Update mode (POSCNT increments on every rising and every falling edge of QEA and QEB input signals). Asynchronous external QEA and QEB inputs are synchronized to the TCY clock by the input sampling FF in the noise filter (see Figure 17-14).

2: When POSCNT = MAXCNT, POSCNT is reset to '0' on the next QEA rising edge. POSCNT is set to MAXCNT when POSCNT = 0 (when decrementing), which occurs on the next QEA falling edge.

3: IC2QEIF is generated on the Q4 rising edge.

4: Position counter is loaded with '0' (which is a rollover event in this case) on POSCNT = MAXCNT.

5: Position counter is loaded with MAXCNT value (1527h) on underflow.

6: IC2QEIF must be cleared in software.

18.4.4 INTERRUPTS IN DOUBLE UPDATE MODE

This mode is available in Continuous Up/Down Count mode. In the Double Update mode (PTMOD<1:0> = 11), an interrupt event is generated each time the PTMR register is equal to zero and each time the PTMR matches with PTPER register. Figure 18-8 shows the interrupts in Continuous Up/Down Count mode with double updates.

The Double Update mode provides two additional functions to the user in Center-Aligned mode.

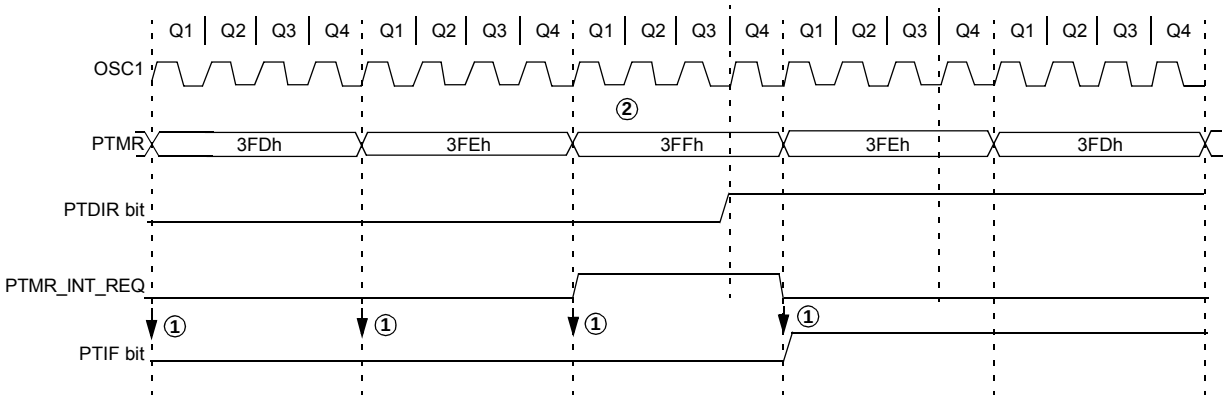
1. The control loop bandwidth is doubled because the PWM duty cycles can be updated twice per period.
2. Asymmetrical center-aligned PWM waveforms can be generated, which are useful for minimizing output waveform distortion in certain motor control applications.

Note:	Do not change the PTMOD bits while PTEN is active; it will yield unexpected results. To change the PWM Timer mode of operation, first clear the PTEN bit, load the PTMOD bits with the required data and then set PTEN.

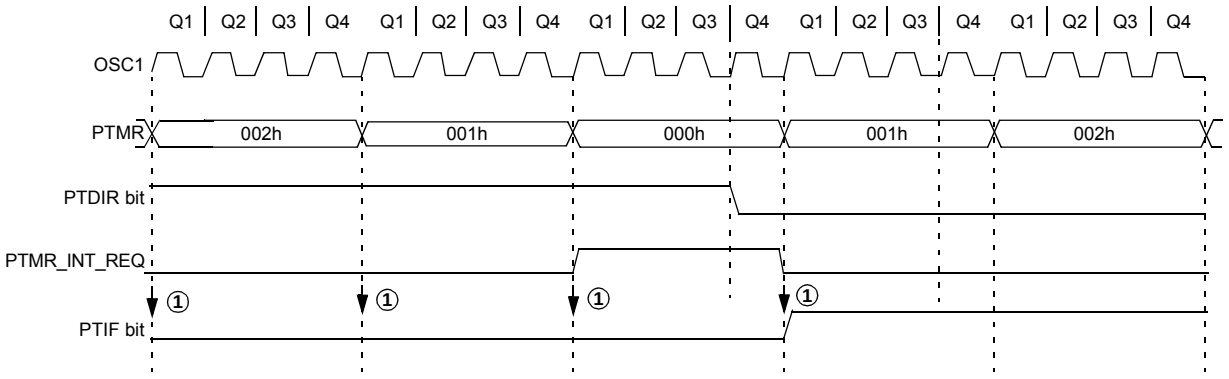
FIGURE 18-8: PWM TIME BASE INTERRUPT, CONTINUOUS UP/DOWN COUNT MODE WITH DOUBLE UPDATES

A: PRESCALER = 1:1

Case 1: PTMR Counting Upwards



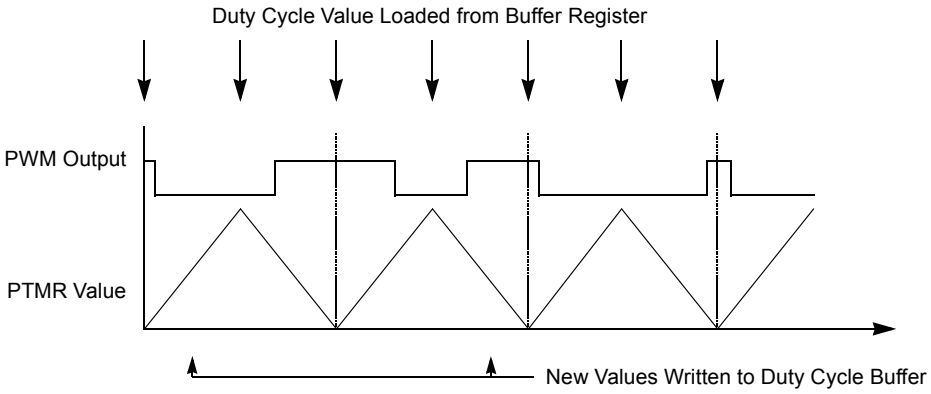
Case 2: PTMR Counting Downwards



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

2: PWM Time Base Period register, PTPER, is loaded with the value, 3FFh, for this example.

FIGURE 18-14: DUTY CYCLE UPDATE TIMES IN CONTINUOUS UP/DOWN COUNT MODE WITH DOUBLE UPDATES

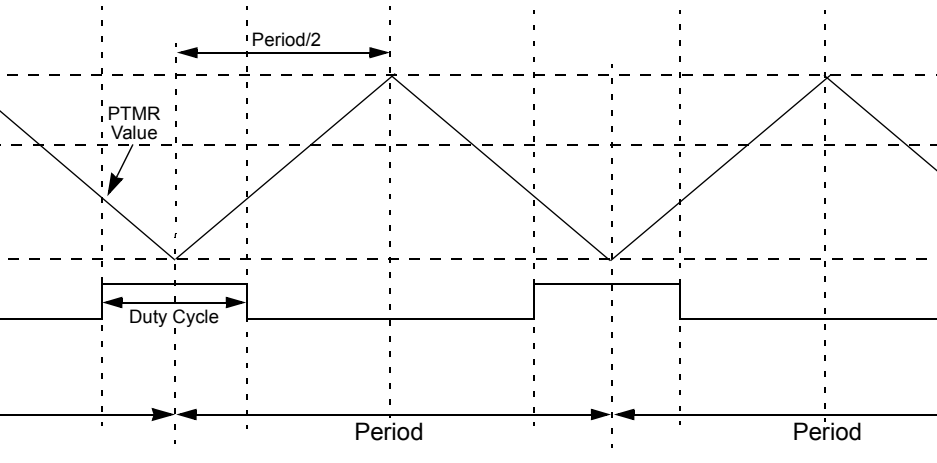


18.6.4 CENTER-ALIGNED PWM

Center-aligned PWM signals are produced by the module when the PWM time base is configured in a Continuous Up/Down Count mode (see Figure 18-15). The PWM compare output is driven to the active state when the value of the Duty Cycle register matches the value of PTMR and the PWM time base is counting downwards (PTDIR = 1). The PWM compare output will be driven to the inactive state when the PWM time base is counting upwards (PTDIR = 0) and the value in the PTMR register matches the duty cycle value. If the value in a particular Duty Cycle register is zero, then the output on the corresponding PWM pin will be inactive for the entire PWM period. In addition, the output on the PWM pin will be active for the entire PWM period if the value in the Duty Cycle register is equal to or greater than the value in the PTPER register.

Note:	When the PWM is started in Center-Aligned mode, the PWM Time Base Period register (PTPER) is loaded into the PWM Time Base register (PTMR) and the PTMR is configured automatically to start down counting. This is done to ensure that all the PWM signals don't start at the same time.

FIGURE 18-15: START OF CENTER-ALIGNED PWM



NOTES:

21.0 10-BIT HIGH-SPEED ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The high-speed Analog-to-Digital (A/D) Converter module allows conversion of an analog signal to a corresponding 10-bit digital number.

The A/D module supports up to 5 input channels on PIC18F2331/2431 devices, and up to 9 channels on the PIC18F4331/4431 devices.

This high-speed 10-bit A/D module offers the following features:

- Up to 200K samples per second
- Two sample and hold inputs for dual-channel simultaneous sampling
- Selectable Simultaneous or Sequential Sampling modes
- 4-word data buffer for A/D results
- Selectable data acquisition timing
- Selectable A/D event trigger
- Operation in Sleep using internal oscillator

These features lend themselves to many applications including motor control, sensor interfacing, data acquisition and process control. In many cases, these features will reduce the software overhead associated with standard A/D modules.

The module has 9 registers:

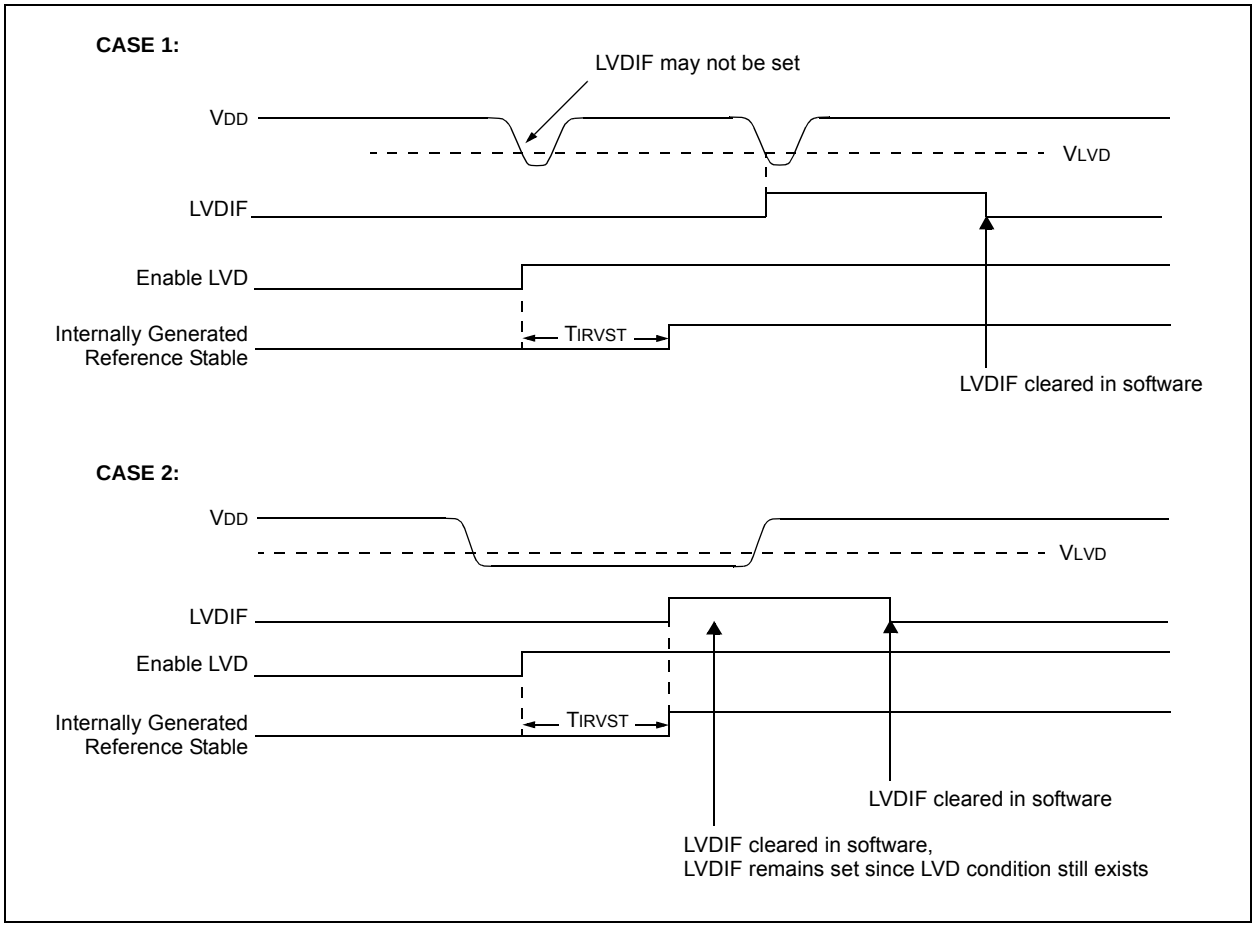
- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)
- A/D Control Register 3 (ADCON3)
- A/D Channel Select Register (ADCHS)
- Analog I/O Select Register 0 (ANSEL0)
- Analog I/O Select Register 1 (ANSEL1)

22.4 LVD Start-up Time

The internal reference voltage of the LVD module, specified in electrical specification Parameter D420, may be used by other internal circuitry, such as the Programmable Brown-out Reset. If the LVD, or other circuits using the voltage reference, are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low-voltage condition can be reliably detected. This start-up time, TIRVST, is an interval that is independent of device clock speed. It is specified in electrical specification Parameter 36.

The LVD interrupt flag is not enabled until TIRVST has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval (refer to Figure 22-2).

FIGURE 22-2: LOW-VOLTAGE DETECT WAVEFORMS



REGISTER 23-5: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

R/P-1	U	U	R/P-1	R/P-1	R/P-1	U	R/P-1
MCLRE(1)	—	—	EXCLKMX(1)	PWM4MX(1)	SSPMX(1)	—	FLTAMX(1)
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Pin Enable bit(1)
- 1 = $\overline{\text{MCLR}}$ pin is enabled; RE3 input pin is disabled
- 0 = RE3 input pin is enabled; $\overline{\text{MCLR}}$ is disabled

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **EXCLKMX:** TMR0/T5CKI External Clock MUX bit(1)
- 1 = TMR0/T5CKI external clock input is multiplexed with RC3
- 0 = TMR0/T5CKI external clock input is multiplexed with RD0

bit 3 **PWM4MX:** PWM4 MUX bit(1)
- 1 = PWM4 output is multiplexed with RB5
- 0 = PWM4 output is multiplexed with RD5

bit 2 **SSPMX:** SSP I/O MUX bit(1)
- 1 = SCK/SCL clocks and SDA/SDI data are multiplexed with RC5 and RC4, respectively. SDO output is multiplexed with RC7.
- 0 = SCK/SCL clocks and SDA/SDI data are multiplexed with RD3 and RD2, respectively. SDO output is multiplexed with RD1.

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

bit 0 **FLTAMX:** $\overline{\text{FLTA}}$ MUX bit(1)
- 1 = $\overline{\text{FLTA}}$ input is multiplexed with RC1
- 0 = $\overline{\text{FLTA}}$ input is multiplexed with RD4

Note 1: Unimplemented in PIC18F2331/2431 devices; maintain this bit set.

REGISTER 23-15: WDTCON: WATCHDOG TIMER CONTROL REGISTER

R-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
WDTW	—	—	—	—	—	—	SWDTEN(1)
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 **WDTW:** Watchdog Timer Window bit
1 = WDT count is in fourth quadrant
0 = WDT count is not in fourth quadrant

bit 6-1 **Unimplemented**: Read as '0'

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit(1)
1 = WDT is turned on
0 = WDT is turned off

Note 1: If the WDTEN Configuration bit = 1, then WDT is always enabled, irrespective of this control bit. If WDTEN Configuration bit = 0, then it is possible to turn WDT on/off with this control bit.

TABLE 23-2: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG2H	—	—	$\overline{\text{WINEN}}$	WDTPS3	WDTPS2	WDTPS2	WDTPS0	WDTEN
RCON	IPEN	—	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
WDTCON	WDTW	—	—	—	—	—	—	SWDTEN

Legend: Shaded cells are not used by the Watchdog Timer.

TBLRD **Table Read**

Syntax: [*label*] TBLRD (*; *+; *-; +*)

Operands: None

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

Status Affected: None

Encoding:

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

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used.
The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

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

TBLRD **Table Read (cont'd)**

Example 1:
```
TBLRD   *+ ;
```
Before Instruction
TABLAT = 0x55
TBLPTR = 0x00A356
MEMORY(0x00A356) = 0x34
After Instruction
TABLAT = 0x34
TBLPTR = 0x00A357

Example 2:
```
TBLRD   +* ;
```
Before Instruction
TABLAT = 0xAA
TBLPTR = 0x01A357
MEMORY(0x01A357) = 0x12
MEMORY(0x01A358) = 0x34
After Instruction
TABLAT = 0x34
TBLPTR = 0x01A358

NOTES:

APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

Not Applicable

APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a baseline device (i.e., PIC16C5X) to an enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

Not Currently Available

Q

R