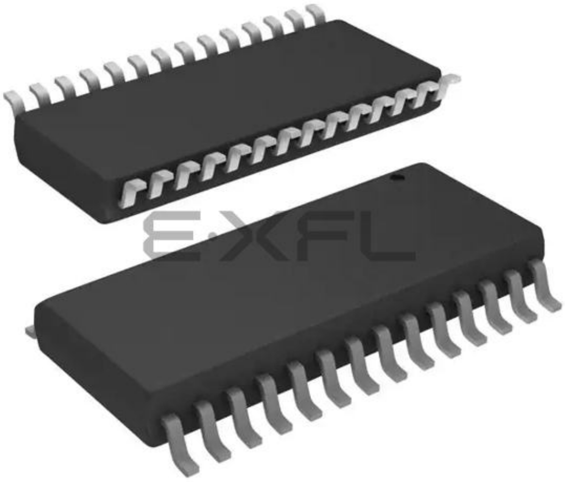
Microchip Technology - PIC18LF2431T-I/SO Datasheet



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	I^2C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	24
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 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/pic18lf2431t-i-so

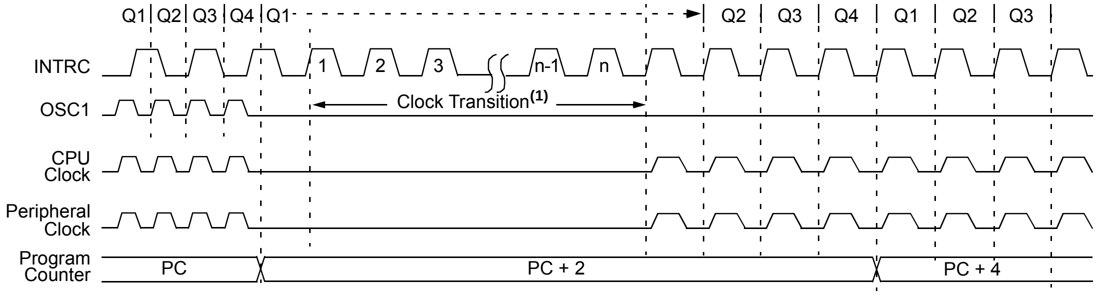
If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the device clocks.

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.

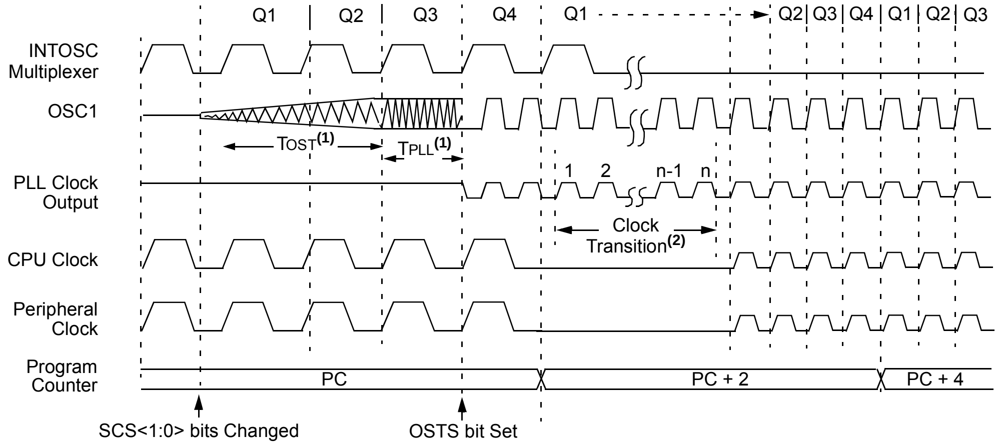
On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 4-4). When the clock switch is complete, the IOFS bit is cleared, 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.

NOTES:

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
```

REGISTER 10-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	U-0	R/W-0
OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF
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 **OSCFIF:** Oscillator Fail Interrupt Flag bit

1 = Device oscillator failed, clock input has changed to INTOSC (must be cleared in software)
0 = Device clock operating

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EEIF:** EEPROM or Flash Write Operation Interrupt Flag bit

1 = The write operation is complete (must be cleared in software)
0 = The write operation is not complete or has not been started

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **LVDIF:** Low-Voltage Detect Interrupt Flag bit

1 = The supply voltage has fallen below the specified LVD voltage (must be cleared in software)
0 = The supply voltage is greater than the specified LVD voltage

bit 1 **Unimplemented:** Read as ‘0’

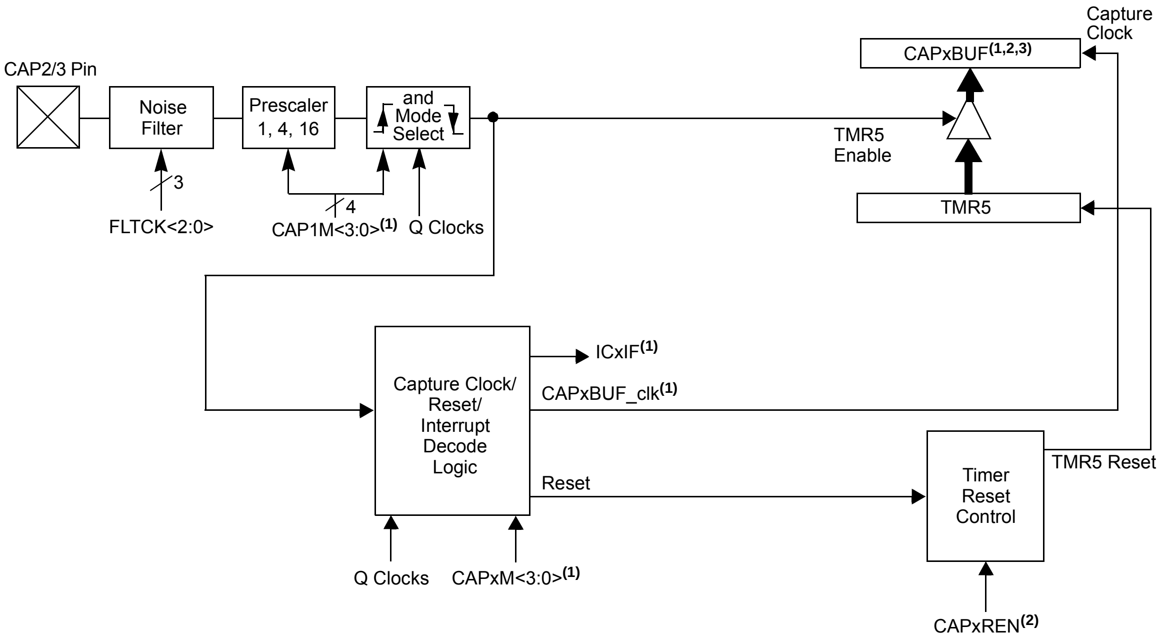
bit 0 **CCP2IF:** CCP2 Interrupt Flag bit

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:
Not used in this mode.

FIGURE 17-3: INPUT CAPTURE BLOCK DIAGRAM FOR IC2 AND IC3



Note 1: IC2 and IC3 are denoted as x = 2 and 3.

2: CAP2BUF is enabled as POSCNT when QEI mode is active.

3: CAP3BUF is enabled as MAXCNT when QEI mode is active.

17.1.5 ENTERING INPUT CAPTURE MODE AND CAPTURE TIMING

The following is a summary of functional operation upon entering any of the Input Capture modes:

1. After the module is configured for one of the Capture modes by setting the Capture Mode Select bits (CAPxM<3:0>), the first detected edge captures the Timer5 value and stores it in the CAPxBUF register. The timer is then reset (depending on the setting of CAPxREN bit) and starts to increment according to its settings (see Figure 17-4, Figure 17-5 and Figure 17-6).
2. On all edges, the capture logic performs the following:
   a) Input Capture mode is decoded and the active edge is identified.
   b) The CAPxREN bit is checked to determine whether Timer5 is reset or not.
   c) On every active edge, the Timer5 value is recorded in the Input Capture Buffer (CAPxBUF).
   d) Reset Timer5 after capturing the value of the timer when the CAPxREN bit is enabled. Timer5 is reset on every active capture edge in this case.
   e) On all continuing capture edge events, repeat steps (a) through (d) until the operational mode is terminated, either by user firmware, POR or BOR.
   f) The timer value is not affected when switching into and out of various Input Capture modes.

17.1.6 TIMER5 RESET

Every input capture trigger can optionally reset (TMR5). The Capture Reset Enable bit, CAPxREN, gates the automatic Reset of the time base of the capture event with this enable Reset signal. All capture events reset the selected timer when CAPxREN is set. Resets are disabled when CAPxREN is cleared (see Figure 17-4, Figure 17-5 and Figure 17-6).

Note: The CAPxREN bit has no effect in Pulse-Width Measurement mode.

17.1.7 IC INTERRUPTS

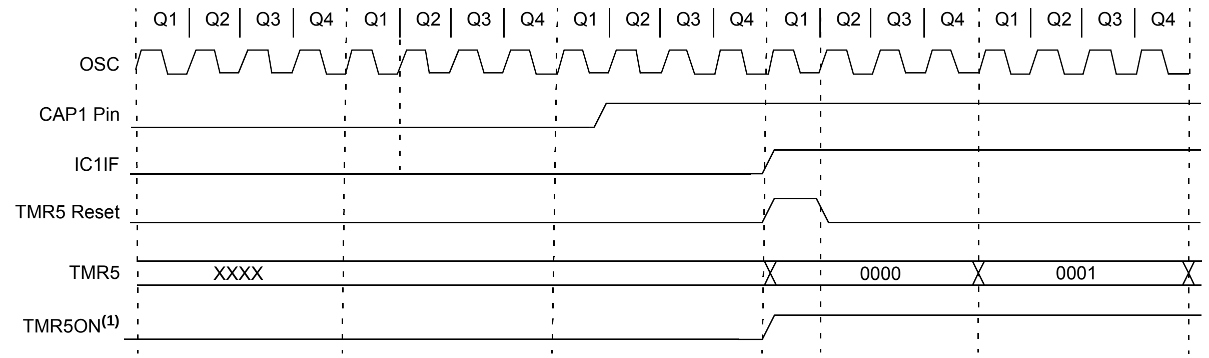
There are four operating modes for which the IC module can generate an interrupt and set one of the Interrupt Capture Flag bits (IC1IF, IC2QEIF or IC3DRIF). The interrupt flag that is set depends on the channel in which the event occurs. The modes are:

- Edge Capture (CAPxM<3:0> = `0001`, `0010`, `0011` or `0100`)
- Period Measurement Event (CAPxM<3:0> = `0101`)
- Pulse-Width Measurement Event (CAPxM<3:0> = `0110` or `0111`)
- State Change Event (CAPxM<3:0> = `1000`)

Note: The Special Event Trigger is generated only in the Special Event Trigger mode on the CAP1 input (CAP1M<3:0> = `1110` and `1111`). IC1IF interrupt is not set in this mode.

The timing of interrupt and Special Event Trigger events is shown in Figure 17-7. Any active edge is detected on the rising edge of Q2 and propagated on the rising edge of Q4 rising edge. If an active edge happens to occur any later than this (on the falling edge of Q2, for example), then it will be recognized on the next Q2 rising edge.

FIGURE 17-7: CAPx INTERRUPTS AND IC1 SPECIAL EVENT TRIGGER



Note 1: Timer5 is only reset and enabled (assuming TMR5ON = `0` and T5MOD = `1`) when the Special Event Trigger Reset is enabled for the Timer5 Reset input. The TMR5ON bit is asserted and Timer5 is reset on the Q1 rising edge following the event capture. With the Special Event Trigger Reset disabled, Timer5 cannot be reset by the Special Event Trigger Reset on the CAP1 input. In order for the Special Event Trigger Reset to work as the Reset trigger to Timer5, IC1 must be configured in the Special Event Trigger mode (CAP1M<3:0> = `1110` or `1111`).

18.4.4 INTERRUPTS IN DOUBLE UPDATE MODE

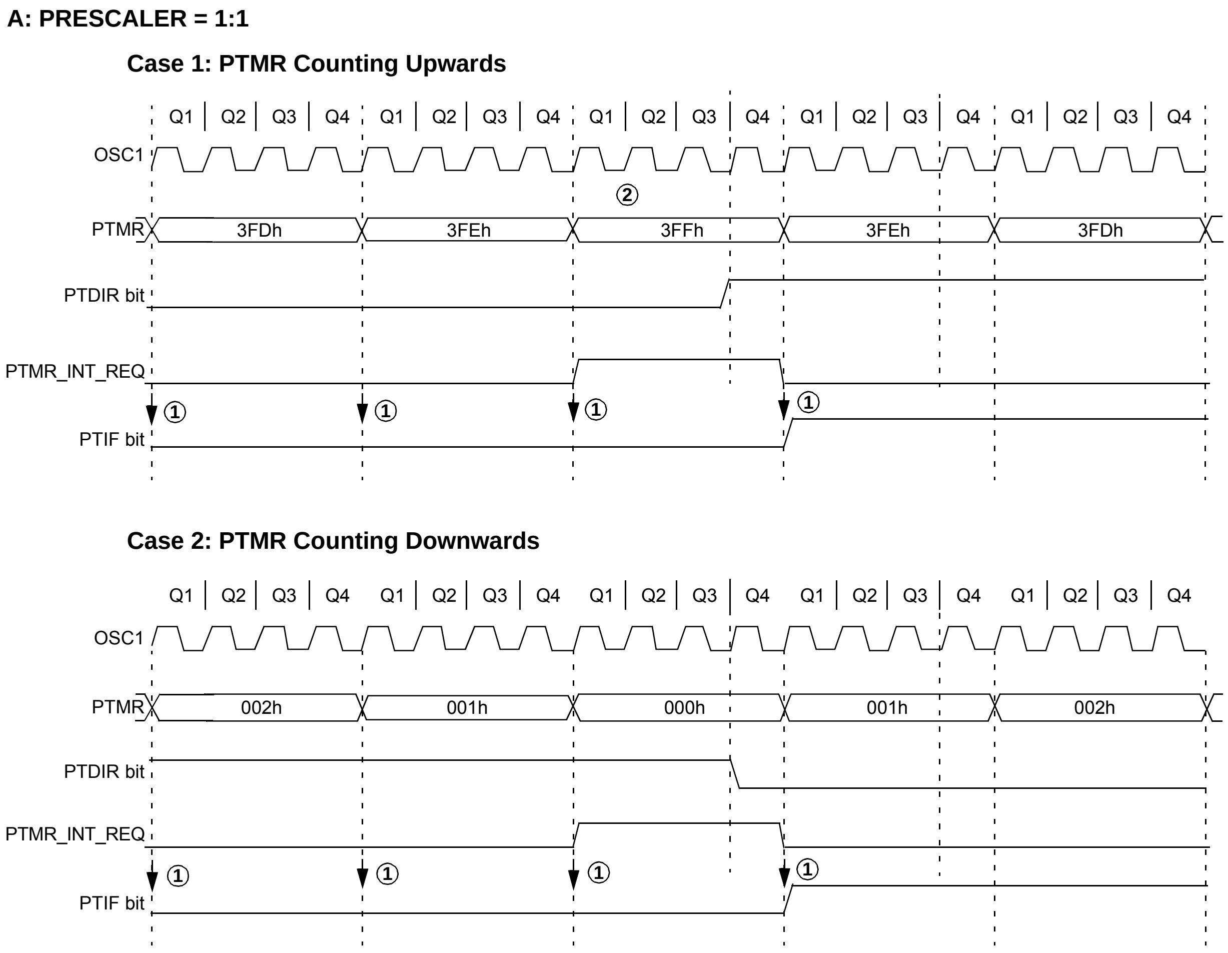
This mode is available in Continuous Up/Down Count mode. In the Double Update mode (PTMOD<1:0> = 11), an interrupt event is generated each time the PTMR register is equal to zero and each time the PTMR matches with PTPER register. Figure 18-8 shows the interrupts in Continuous Up/Down Count mode with 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
Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
OSC1
②
PTMR
3FDh
3FEh
3FFh
3FEh
3FDh
PTDIR bit
PTMR_INT_REQ
①
PTIF bit
Case 2: PTMR Counting Downwards
Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
OSC1
PTMR
002h
001h
000h
001h
002h
PTDIR bit
PTMR_INT_REQ
①
PTIF bit

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 18-21: PWM OUTPUT OVERRIDE EXAMPLE #1

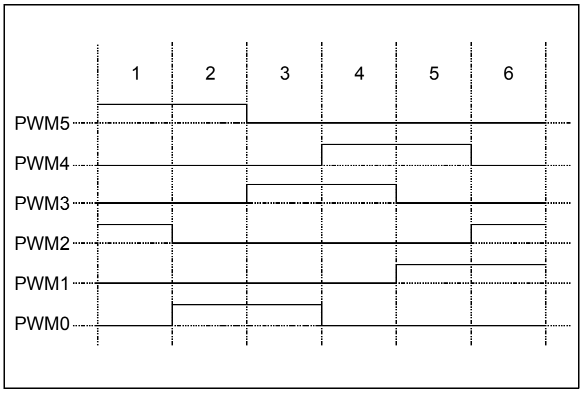


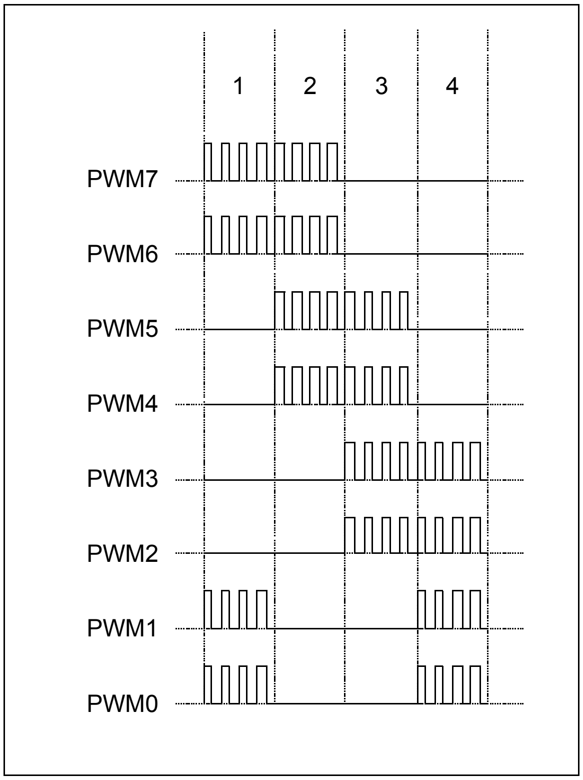
TABLE 18-4: PWM OUTPUT OVERRIDE EXAMPLE #1

State	OVDCOND (POVD)	OVDCONS (POUT)
1	00000000b	00100100b
2	00000000b	00100001b
3	00000000b	00001001b
4	00000000b	00011000b
5	00000000b	00010010b
6	00000000b	00000110b

TABLE 18-5: PWM OUTPUT OVERRIDE EXAMPLE #2

State	OVDCOND (POVD)	OVDCONS (POUT)
1	11000011b	00000000b
2	11110000b	00000000b
3	00111100b	00000000b
4	00001111b	00000000b

FIGURE 18-22: PWM OUTPUT OVERRIDE EXAMPLE #2



18.13 PWM Update Lockout

For a complex PWM application, the user may need to write up to four Duty Cycle registers and the PWM Time Base Period register, PTPER, at a given time. In some applications, it is important that all buffer registers be written before the new duty cycle and period values are loaded for use by the module.

A PWM update lockout feature may optionally be enabled so the user may specify when new duty cycle buffer values are valid. The PWM update lockout feature is enabled by setting the control bit, UDIS, in the PWMCON1 register. This bit affects all Duty Cycle Buffer registers and the PWM Time Base Period register, PTPER.

To perform a PWM update lockout:

1. Set the UDIS bit.
2. Write all Duty Cycle registers and PTPER, if applicable.
3. Clear the UDIS bit to re-enable updates.
4. With this, when UDIS bit is cleared, the buffer values will be loaded to the actual registers. This makes a synchronous loading of the registers.

18.14 PWM Special Event Trigger

The PWM module has a Special Event Trigger capability that allows A/D conversions to be synchronized to the PWM time base. The A/D sampling and conversion time may be programmed to occur at any point within the PWM period. The Special Event Trigger allows the user to minimize the delay between the time when A/D conversion results are acquired and the time when the duty cycle value is updated.

The PWM 16-bit Special Event Trigger register, SEVTCMP (high and low), and five control bits in the PWMCON1 register are used to control its operation.

The PTMR value for which a Special Event Trigger should occur is loaded into the SEVTCMP register pair. The SEVTDIR bit in the PWMCON1 register specifies the counting phase when the PWM time base is in a Continuous Up/Down Count mode.

If the SEVTDIR bit is cleared, the Special Event Trigger will occur on the upward counting cycle of the PWM time base. If SEVTDIR is set, the Special Event Trigger will occur on the downward count cycle of the PWM time base. The SEVTDIR bit has effect only when the PWM timer is in the Continuous Up/Down Count mode.

18.14.1 SPECIAL EVENT TRIGGER ENABLE

The PWM module will always produce Special Event Trigger pulses. This signal may optionally be used by the A/D module. Refer to **Section 21.0 “10-Bit High-Speed Analog-to-Digital Converter (A/D) Module”** for details.

18.14.2 SPECIAL EVENT TRIGGER POSTSCALER

The PWM Special Event Trigger has a postscaler that allows a 1:1 to 1:16 postscale ratio. The postscaler is configured by writing the SEVOPS<3:0> control bits in the PWMCON1 register.

The Special Event Trigger output postscaler is cleared on any write to the SEVTCMP register pair, or on any device Reset.

FIGURE 19-2: SPI MODE TIMING, MASTER MODE

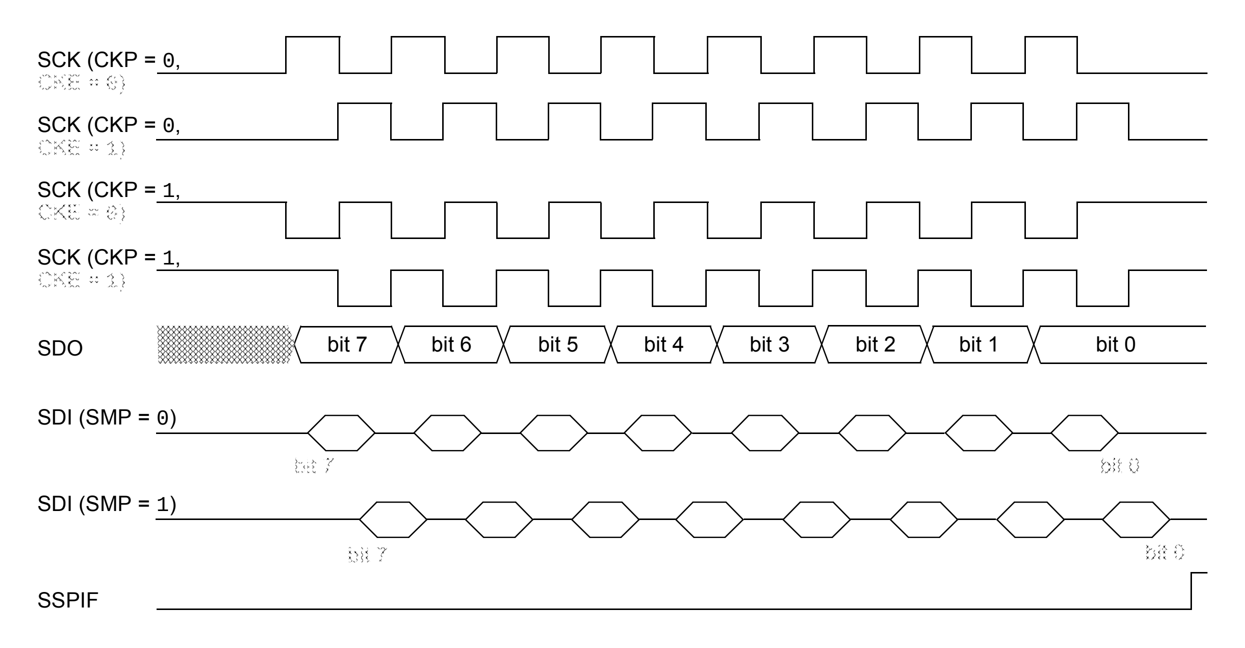
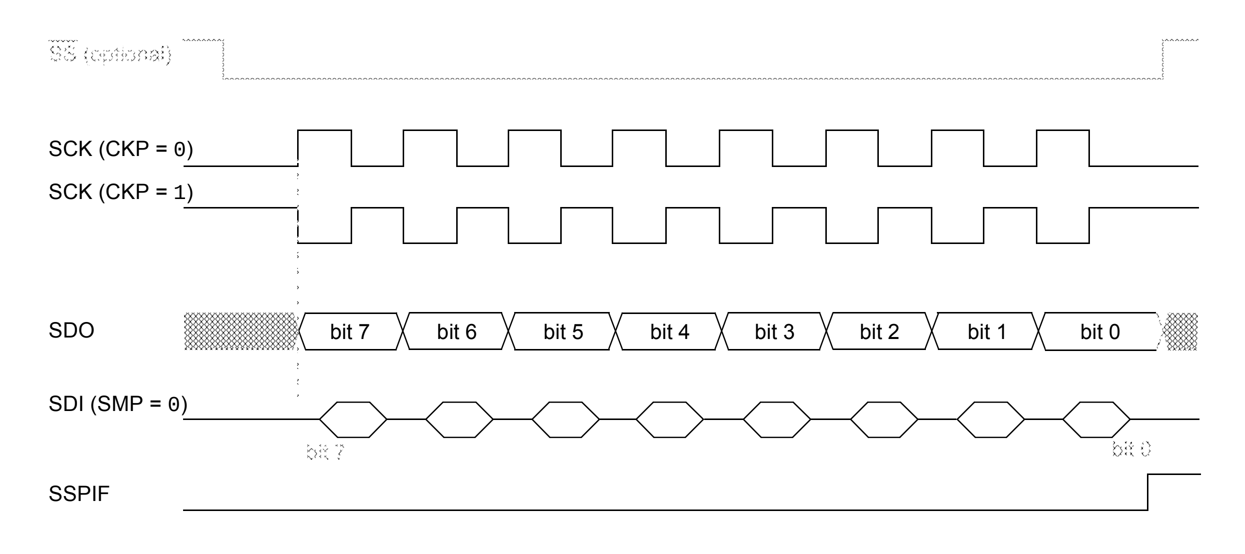


FIGURE 19-3: SPI MODE TIMING (SLAVE MODE WITH CKE = 0)



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.

REGISTER 21-4: ADCON3: A/D CONTROL REGISTER 3

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRS1	ADRS0	—	SSRC4(1)	SSRC3(1)	SSRC2(1)	SSRC1(1)	SSRC0(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 7-6 **ADRS<1:0>**: A/D Result Buffer Depth Interrupt Select Control for Continuous Loop Mode bits

The ADRS bits are ignored in Single-Shot mode.
00 = Interrupt is generated when each word is written to the buffer
01 = Interrupt is generated when the 2nd and 4th words are written to the buffer
10 = Interrupt is generated when the 4th word is written to the buffer
11 = Unimplemented

bit 5 **Unimplemented:** Read as ‘0’

bit 4-0 **SSRC<4:0>:** A/D Trigger Source Select bits(1)

00000 = All triggers disabled
xxxx1 = External interrupt RC3/INT0 starts A/D sequence
xxx1x = Timer5 starts A/D sequence
xx1xx = Input Capture 1 (IC1) starts A/D sequence
x1xxx = CCP2 compare match starts A/D sequence
1xxxx = Power Control PWM module rising edge starts A/D sequence

Note 1: The SSRC<4:0> bits can be set such that any of the triggers will start a conversion (e.g., SSRC<4:0> = 00101 will trigger the A/D conversion sequence when RC3/INT0 or Input Capture 1 event occurs).

TABLE 23-1: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300000h	CONFIG1L	—	—	—	—	—	—	—	—	`---- ----`
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	`11-- 1111`
300002h	CONFIG2L	—	—	—	—	BORV1	BORV0	BOREN	$\overline{\text{PWRTEN}}$	`---- 1111`
300003h	CONFIG2H	—	—	$\overline{\text{WINEN}}$	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	`--11 1111`
300004h	CONFIG3L	—	—	$\overline{\text{T1OSCMX}}$	HPOL	LPOL	PWMPIN	—	—	`--11 11--`
300005h	CONFIG3H	MCLRE(1)	—	—	EXCLKMX(1)	PWM4MX(1)	SSPMX(1)	—	FLTAMX(1)	`1--1 11-1`
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN	`1--- -1-1`
300007h	CONFIG4H	—	—	—	—	—	—	—	—	`---- ----`
300008h	CONFIG5L	—	—	—	—	CP3(1)	CP2(1)	CP1	CP0	`---- 1111`
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	`11-- ----`
30000Ah	CONFIG6L	—	—	—	—	WRT3(1)	WRT2(1)	WRT1	WRT0	`---- 1111`
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	`111- ----`
30000Ch	CONFIG7L	—	—	—	—	EBTR3(1)	EBTR2(1)	EBTR1	EBTR0	`---- 1111`
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	`-1-- ----`
3FFFFEh	DEVID1(2)	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	`xxxx xxxx`(2)
3FFFFFh	DEVID2(2)	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	`0000 0101`

Legend: x = unknown, u = unchanged, - = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.

2: See Register 23-13 for DEVID1 values. DEVID registers are read-only and cannot be programmed by the user.

REGISTER 23-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

R/P-1	R/P-1	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
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 **IESO:** Internal External Switchover bit

1 = Internal External Switchover mode enabled
0 = Internal External Switchover mode disabled

bit 6 **FCMEN:** Fail-Safe Clock Monitor Enable bit

1 = Fail-Safe Clock Monitor enabled
0 = Fail-Safe Clock Monitor disabled

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **FOSC<3:0>:** Oscillator Selection bits

11xx = External RC oscillator, CLKO function on RA6
1001 = Internal oscillator block, CLKO function on RA6 and port function on RA7 (INTIO1)
1000 = Internal oscillator block, port function on RA6 and port function on RA7 (INTIO2)
0111 = External RC oscillator, port function on RA6
0110 = HS oscillator, PLL enabled (clock frequency = 4 x FOSC1)
0101 = EC oscillator, port function on RA6 (ECIO)
0100 = EC oscillator, CLKO function on RA6 (EC)
0010 = HS oscillator
0001 = XT oscillator
0000 = LP oscillator

COMF Complement f

Syntax:	[*label*] COMF f [,d [,a]]
Operands:	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	$(\overline{f}) \rightarrow$ dest
Status Affected:	N, Z
Encoding:	`0001` `11da` `ffff` `ffff`
Description:	The contents of register, ‘f’, are complemented. 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, 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 to destination

Example: `COMF REG, W`

Before Instruction
REG = 0x13

After Instruction
REG = 0x13
W = 0xEC

CPFSEQ Compare f with W, Skip if f = W

Syntax:	[*label*] CPFSEQ f [,a]
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	(f) – (W), skip if (f) = (W) (unsigned comparison)
Status Affected:	None
Encoding:	`0110` `001a` `ffff` `ffff`
Description:	Compares the contents of data memory location, ‘f’, to the contents of W by performing an unsigned subtraction. If ‘f’ = W, then the fetched instruction is discarded and a `NOP` is executed instead, making this 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.
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	No operation

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      CPFSEQ REG
NEQUAL    :
EQUAL     :
```

Before Instruction
PC Address = `HERE`
W = ?
REG = ?

After Instruction
If REG = W;
PC = Address (`EQUAL`)
If REG ≠ W;
PC = Address (`NEQUAL`)

RLNCF Rotate Left f (No Carry)

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

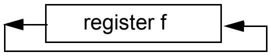
Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

`0100`	`01da`	`ffff`	`ffff`

Description: The contents of register, ‘f’, are rotated one bit to the left. If ‘d’ is ‘`0`’, the result is placed in W. If ‘d’ is ‘`1`’, the result is stored back in register, ‘f’. If ‘a’ is ‘`0`’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘`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 to destination

Example: `RLNCF REG`

Before Instruction
REG = `1010 1011`

After Instruction
REG = `0101 0111`

RRCF Rotate Right f through Carry

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

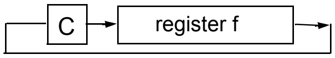
Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

`0011`	`00da`	`ffff`	`ffff`

Description: The contents of register, ‘f’, are rotated one bit to the right through the Carry Flag. If ‘d’ is ‘`0`’, the result is placed in W. If ‘d’ is ‘`1`’, the result is placed back in register, ‘f’. If ‘a’ is ‘`0`’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘`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 to destination

Example: `RRCF REG, W`

Before Instruction
REG = `1110 0110`
C = `0`

After Instruction
REG = `1110 0110`
W = `0111 0011`
C = `0`

25.7 MPLAB SIM Software Simulator

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

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

25.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

25.9 MPLAB ICD 3 In-Circuit Debugger System

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

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

25.10 PICkit 3 In-Circuit Debugger/ Programmer and PICkit 3 Debug Express

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

The PICkit 3 Debug Express include the PICkit 3, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

TABLE 26-1: MEMORY PROGRAMMING REQUIREMENTS

DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial -40°C ≤ T_A ≤ +125°C for extended				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Internal Program Memory Programming Specifications[1]					
D110	V_{PP}	Voltage on $\overline{MCLR}$/V_{PP} pin	9.00	—	13.25	V	**(Note 3)**
D112	I_{PP}	Current into $\overline{MCLR}$/V_{PP} pin	—	—	300	μA	
D113	I_{DDP}	Supply Current during Programming	—	—	1	mA	
		Data EEPROM Memory					
D120	E_D	Byte Endurance	100K	1M	—	E/W	-40°C to +85°C
D121	V_{DRW}	V_{DD} for Read/Write	V_{MIN}	—	5.5	V	Using EECON to read/write V_{MIN} = Minimum operating voltage
D122	T_{DEW}	Erase/Write Cycle Time	—	4	—	ms	
D123	T_{RETD}	Characteristic Retention	40	—	—	Year	Provided no other specifications are violated
D124	T_{REF}	Number of Total Erase/Write Cycles before Refresh[2]	1M	10M	—	E/W	-40°C to +85°C
		Program Flash Memory					
D130	E_P	Cell Endurance	10K	100K	—	E/W	-40°C to +85°C
D131	V_{PR}	V_{DD} for Read	V_{MIN}	—	5.5	V	V_{MIN} = Minimum operating voltage
D132	V_{IE}	V_{DD} for Block Erase	4.5	—	5.5	V	Using ICSP™ port
D132A	V_{IW}	V_{DD} for Externally Timed Erase or Write	4.5	—	5.5	V	Using ICSP port
D132B	V_{PEW}	V_{DD} for Self-Timed Write	V_{MIN}	—	5.5	V	V_{MIN} = Minimum operating voltage
D133	T_{IE}	ICSP™ Block Erase Cycle Time	—	4	—	ms	V_{DD} > 4.5V
D133A	T_{IW}	ICSP Erase or Write Cycle Time (externally timed)	1	—	—	ms	V_{DD} > 4.5V
D133A	T_{IW}	Self-Timed Write Cycle Time	—	2	—	ms	
D134	T_{RETD}	Characteristic Retention	40	100	—	Year	Provided no other specifications are violated

† Data in “Typ” column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: These specifications are for programming the on-chip program memory through the use of table write instructions.

2: Refer to **Section 7.9 “Using the Data EEPROM”** for a more detailed discussion on data EEPROM endurance.

3: Required only if Single-Supply Programming is disabled.

FIGURE 26-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)

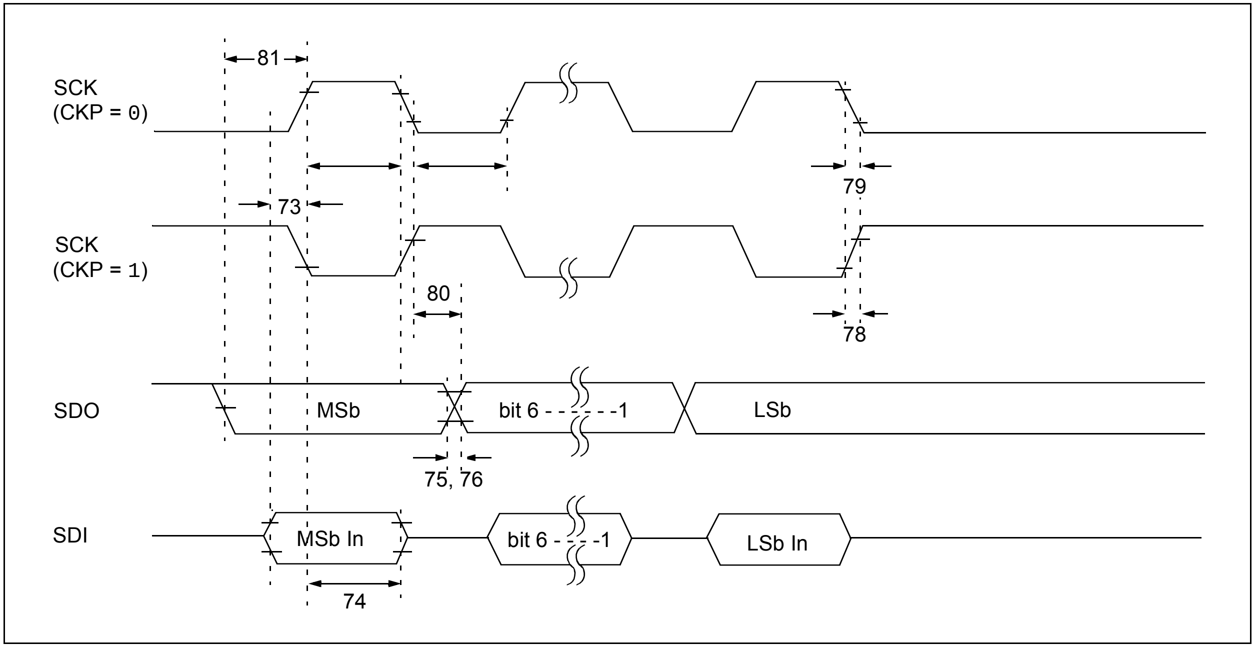
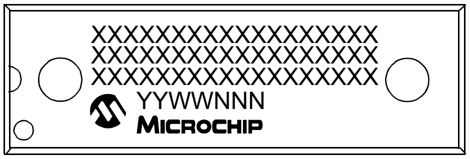


TABLE 26-12: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

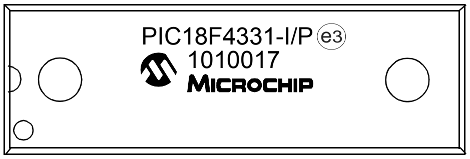
Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2		1.5 TCY + 40	—	ns	
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO 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	
81	TdoV2scH, TdoV2scL	SDO Data Output Setup to SCK Edge		TCY	—	ns	

27.1 Package Marking Information (Continued)

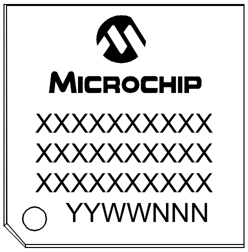
40-Lead PDIP



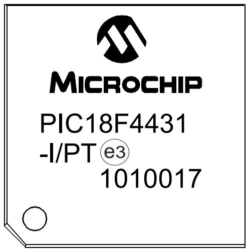
Example



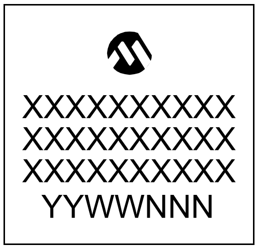
44-Lead TQFP



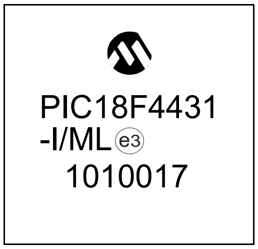
Example



44-Lead QFN



Example



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