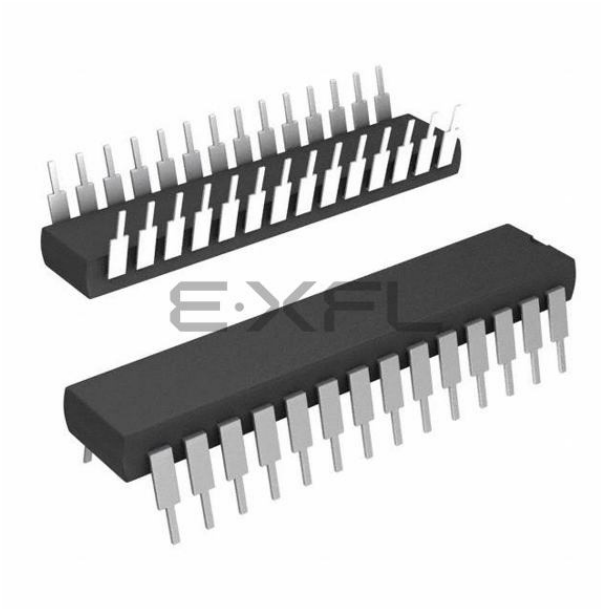
# Microchip Technology - PIC18F258-I/SP Datasheet



**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"

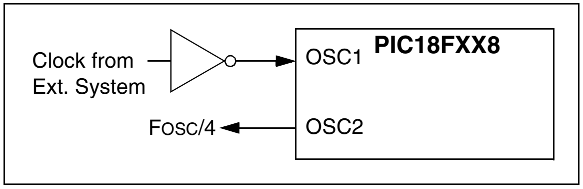
| Details | |
|---|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 40MHz |
| Connectivity | CANbus, I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, LVD, POR, PWM, WDT |
| Number of I/O | 22 |
| Program Memory Size | 32KB (16K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 1.5K x 8 |
| Voltage - Supply (Vcc/Vdd) | 4.2V ~ 5.5V |
| Data Converters | A/D 5x10b |
| Oscillator Type | External |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Through Hole |
| Package / Case | 28-DIP (0.300", 7.62mm) |
| Supplier Device Package | 28-SPDIP |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18f258-i-sp |

## 2.4 External Clock Input

The EC and ECIO Oscillator modes require an external clock source to be connected to the OSC1 pin. The feedback device between OSC1 and OSC2 is turned off in these modes to save current. There is no oscillator start-up time required after a Power-on Reset or after a recovery from Sleep mode.

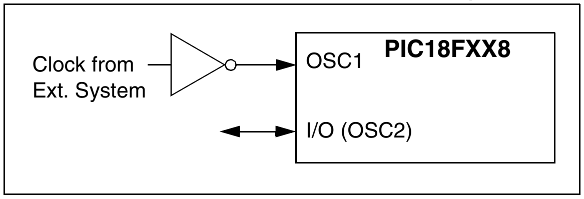
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-3 shows the pin connections for the EC Oscillator mode.

**FIGURE 2-3: EXTERNAL CLOCK INPUT OPERATION (EC OSC CONFIGURATION)**



The ECIO Oscillator mode functions like the EC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. Figure 2-4 shows the pin connections for the ECIO Oscillator mode.

**FIGURE 2-4: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)**



## 2.5 HS4 (PLL)

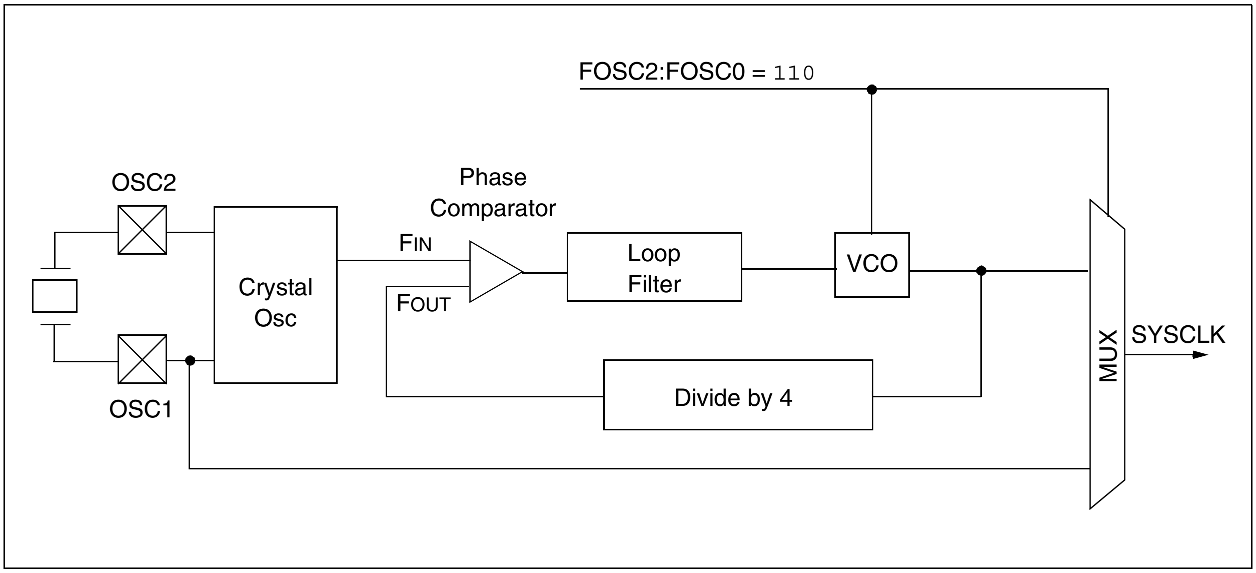
A Phase Locked Loop circuit is provided as a programmable option for users that want to multiply the frequency of the incoming crystal oscillator signal by 4. For an input clock frequency of 10 MHz, the internal clock frequency will be multiplied to 40 MHz. This is useful for customers who are concerned with EMI due to high-frequency crystals.

The PLL can only be enabled when the oscillator configuration bits are programmed for HS mode. If they are programmed for any other mode, the PLL is not enabled and the system clock will come directly from OSC1.

The PLL is one of the modes of the FOSC2:FOSC0 configuration bits. The oscillator mode is specified during device programming.

A PLL lock timer is used to ensure that the PLL has locked before device execution starts. The PLL lock timer has a time-out referred to as TPLL.

**FIGURE 2-5: PLL BLOCK DIAGRAM**



### TABLE 4-2: REGISTER FILE SUMMARY (CONTINUED)

| File Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Details on Page: |
|---|---|---|---|---|---|---|---|---|---|---|
| CANSTATRO1 | OPMODE2 | OPMODE1 | OPMODE0 | — | ICODE2 | ICODE1 | ICODE0 | — | xxx- xxx- | 33, 202 |
| RXB1D7 | RXB1D77 | RXB1D76 | RXB1D75 | RXB1D74 | RXB1D73 | RXB1D72 | RXB1D71 | RXB1D70 | xxxx xxxx | 34, 214 |
| RXB1D6 | RXB1D67 | RXB1D66 | RXB1D65 | RXB1D64 | RXB1D63 | RXB1D62 | RXB1D61 | RXB1D60 | xxxx xxxx | 34, 214 |
| RXB1D5 | RXB1D57 | RXB1D56 | RXB1D55 | RXB1D54 | RXB1D53 | RXB1D52 | RXB1D51 | RXB1D50 | xxxx xxxx | 34, 214 |
| RXB1D4 | RXB1D47 | RXB1D46 | RXB1D45 | RXB1D44 | RXB1D43 | RXB1D42 | RXB1D41 | RXB1D40 | xxxx xxxx | 34, 214 |
| RXB1D3 | RXB1D37 | RXB1D36 | RXB1D35 | RXB1D34 | RXB1D33 | RXB1D32 | RXB1D31 | RXB1D30 | xxxx xxxx | 34, 214 |
| RXB1D2 | RXB1D27 | RXB1D26 | RXB1D25 | RXB1D24 | RXB1D23 | RXB1D22 | RXB1D21 | RXB1D20 | xxxx xxxx | 34, 214 |
| RXB1D1 | RXB1D17 | RXB1D16 | RXB1D15 | RXB1D14 | RXB1D13 | RXB1D12 | RXB1D11 | RXB1D10 | xxxx xxxx | 34, 214 |
| RXB1D0 | RXB1D07 | RXB1D06 | RXB1D05 | RXB1D04 | RXB1D03 | RXB1D02 | RXB1D01 | RXB1D00 | xxxx xxxx | 34, 214 |
| RXB1DLC | — | RXRTR | RB1 | RB0 | DLC3 | DLC2 | DLC1 | DLC0 | -xxx xxxx | 34, 213 |
| RXB1EIDL | EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 | xxxx xxxx | 34, 213 |
| RXB1EIDH | EID15 | EID14 | EID13 | EID12 | EID11 | EID10 | EID9 | EID8 | xxxx xxxx | 34, 212 |
| RXB1SIDL | SID2 | SID1 | SID0 | SRR | EXID | — | EID17 | EID16 | xxxx x-xx | 34, 212 |
| RXB1SIDH | SID10 | SID9 | SID8 | SID7 | SID6 | SID5 | SID4 | SID3 | xxxx xxxx | 34, 212 |
| RXB1CON | RXFUL | RXM1 | RXM0 | — | RXRTRRO | FILHIT2 | FILHIT1 | FILHIT0 | 000- 0000 | 34, 211 |
| CANSTATRO2 | OPMODE2 | OPMODE1 | OPMODE0 | — | ICODE2 | ICODE1 | ICODE0 | — | xxx- xxx- | 33, 202 |
| TXB0D7 | TXB0D77 | TXB0D76 | TXB0D75 | TXB0D74 | TXB0D73 | TXB0D72 | TXB0D71 | TXB0D70 | xxxx xxxx | 34, 208 |
| TXB0D6 | TXB0D67 | TXB0D66 | TXB0D65 | TXB0D64 | TXB0D63 | TXB0D62 | TXB0D61 | TXB0D60 | xxxx xxxx | 34, 208 |
| TXB0D5 | TXB0D57 | TXB0D56 | TXB0D55 | TXB0D54 | TXB0D53 | TXB0D52 | TXB0D51 | TXB0D50 | xxxx xxxx | 34, 208 |
| TXB0D4 | TXB0D47 | TXB0D46 | TXB0D45 | TXB0D44 | TXB0D43 | TXB0D42 | TXB0D41 | TXB0D40 | xxxx xxxx | 34, 208 |
| TXB0D3 | TXB0D37 | TXB0D36 | TXB0D35 | TXB0D34 | TXB0D33 | TXB0D32 | TXB0D31 | TXB0D30 | xxxx xxxx | 34, 208 |
| TXB0D2 | TXB0D27 | TXB0D26 | TXB0D25 | TXB0D24 | TXB0D23 | TXB0D22 | TXB0D21 | TXB0D20 | xxxx xxxx | 34, 208 |
| TXB0D1 | TXB0D17 | TXB0D16 | TXB0D15 | TXB0D14 | TXB0D13 | TXB0D12 | TXB0D11 | TXB0D10 | xxxx xxxx | 34, 208 |
| TXB0D0 | TXB0D07 | TXB0D06 | TXB0D05 | TXB0D04 | TXB0D03 | TXB0D02 | TXB0D01 | TXB0D00 | xxxx xxxx | 34, 208 |
| TXB0DLC | — | TXRTR | — | — | DLC3 | DLC2 | DLC1 | DLC0 | -x-- xxxx | 34, 209 |
| TXB0EIDL | EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 | xxxx xxxx | 34, 208 |
| TXB0EIDH | EID15 | EID14 | EID13 | EID12 | EID11 | EID10 | EID9 | EID8 | xxxx xxxx | 34, 207 |
| TXB0SIDL | SID2 | SID1 | SID0 | — | EXIDE | — | EID17 | EID16 | xxx- x-xx | 34, 207 |
| TXB0SIDH | SID10 | SID9 | SID8 | SID7 | SID6 | SID5 | SID4 | SID3 | xxxx xxxx | 35, 207 |
| TXB0CON | — | TXABT | TXLARB | TXERR | TXREQ | — | TXPRI1 | TXPRI0 | -000 0-00 | 35, 206 |
| CANSTATRO3 | OPMODE2 | OPMODE1 | OPMODE0 | — | ICODE2 | ICODE1 | ICODE0 | — | xxx- xxx- | 33, 202 |
| TXB1D7 | TXB1D77 | TXB1D76 | TXB1D75 | TXB1D74 | TXB1D73 | TXB1D72 | TXB1D71 | TXB1D70 | xxxx xxxx | 35, 208 |
| TXB1D6 | TXB1D67 | TXB1D66 | TXB1D65 | TXB1D64 | TXB1D63 | TXB1D62 | TXB1D61 | TXB1D60 | xxxx xxxx | 35, 208 |
| TXB1D5 | TXB1D57 | TXB1D56 | TXB1D55 | TXB1D54 | TXB1D53 | TXB1D52 | TXB1D51 | TXB1D50 | xxxx xxxx | 35, 208 |
| TXB1D4 | TXB1D47 | TXB1D46 | TXB1D45 | TXB1D44 | TXB1D43 | TXB1D42 | TXB1D41 | TXB1D40 | xxxx xxxx | 35, 208 |
| TXB1D3 | TXB1D37 | TXB1D36 | TXB1D35 | TXB1D34 | TXB1D33 | TXB1D32 | TXB1D31 | TXB1D30 | xxxx xxxx | 35, 208 |
| TXB1D2 | TXB1D27 | TXB1D26 | TXB1D25 | TXB1D24 | TXB1D23 | TXB1D22 | TXB1D21 | TXB1D20 | xxxx xxxx | 35, 208 |
| TXB1D1 | TXB1D17 | TXB1D16 | TXB1D15 | TXB1D14 | TXB1D13 | TXB1D12 | TXB1D11 | TXB1D10 | xxxx xxxx | 35, 208 |
| TXB1D0 | TXB1D07 | TXB1D06 | TXB1D05 | TXB1D04 | TXB1D03 | TXB1D02 | TXB1D01 | TXB1D00 | xxxx xxxx | 35, 208 |
| TXB1DLC | — | TXRTR | — | — | DLC3 | DLC2 | DLC1 | DLC0 | -x-- xxxx | 35, 209 |
| TXB1EIDL | EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 | xxxx xxxx | 35, 208 |
| TXB1EIDH | EID15 | EID14 | EID13 | EID12 | EID11 | EID10 | EID9 | EID8 | xxxx xxxx | 35, 207 |
| TXB1SIDL | SID2 | SID1 | SID0 | — | EXIDE | — | EID17 | EID16 | xxx- x-xx | 35, 207 |
| TXB1SIDH | SID10 | SID9 | SID8 | SID7 | SID6 | SID5 | SID4 | SID3 | xxxx xxxx | 35, 207 |
| TXB1CON | — | TXABT | TXLARB | TXERR | TXREQ | — | TXPRI1 | TXPRI0 | 0000 0000 | 35, 206 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition

**Note 1:** These registers or register bits are not implemented on the PIC18F248 and PIC18F258 and read as '0's.

**2:** Bit 21 of the TBLPTRU allows access to the device configuration bits.

**3:** RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator mode only and read '0' in all other oscillator modes.

## 6.4 Erasing Flash Program Memory

The minimum erase block is 32 words or 64 bytes. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the microcontroller itself, a block of 64 bytes of program memory is erased. The Most Significant 16 bits of the TBLPTR<21:6> point to the block being erased. TBLPTR<5:0> are ignored.

The EECON1 register commands the erase operation. The EEPGD bit must be set to point to the Flash program memory. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation.

For protection, the write initiate sequence for EECON2 must be used.

A long write is necessary for erasing the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

### 6.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

The sequence of events for erasing a block of internal program memory location is:

1. Load Table Pointer with address of row being erased.
2. Set the EECON1 register for the erase operation:
   - set the EEPGD bit to point to program memory;
   - clear the CFGS bit to access program memory;
   - set the WREN bit to enable writes;
   - set the FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the $\overline{\text{WR}}$ bit. This will begin the row erase cycle.
7. The CPU will stall for duration of the erase (about 2 ms using internal timer).
8. Re-enable interrupts.

**EXAMPLE 6-2: ERASING A FLASH PROGRAM MEMORY ROW**

```
               MOVLW   upper (CODE_ADDR)      ; load TBLPTR with the base
               MOVWF   TBLPTRU                ; address of the memory block
               MOVLW   high (CODE_ADDR)
               MOVWF   TBLPTRH
               MOVLW   low (CODE_ADDR)
               MOVWF   TBLPTRL
     ERASE_ROW
               BSF     EECON1, EEPGD          ; point to FLASH program memory
               BCF     EECON1, CFGS           ; access FLASH program memory
               BSF     EECON1, WREN           ; enable write to memory
               BSF     EECON1, FREE           ; enable Row Erase operation
               BCF     INTCON, GIE            ; disable interrupts
               MOVLW   55h
               MOVWF   EECON2                 ; write 55H
Required       MOVLW   0AAh
Sequence       MOVWF   EECON2                 ; write 0AAH
               BSF     EECON1, WR             ; start erase (CPU stall)
               NOP                            ; NOP needed for proper code execution
               BSF     INTCON, GIE            ; re-enable interrupts
```

### 15.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP1IE (PIE registers) clear to avoid false interrupts and should clear the flag bit CCP1IF, following any such change in operating mode.

### 15.2.4 CCP1 PRESCALER

There are four prescaler settings specified by bits CCP1M3:CCP1M0. Whenever the CCP1 module is turned off, or the CCP1 module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared; therefore, the first capture may be from a non-zero prescaler. Example 15-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

### 15.2.5 CAN MESSAGE TIME-STAMP

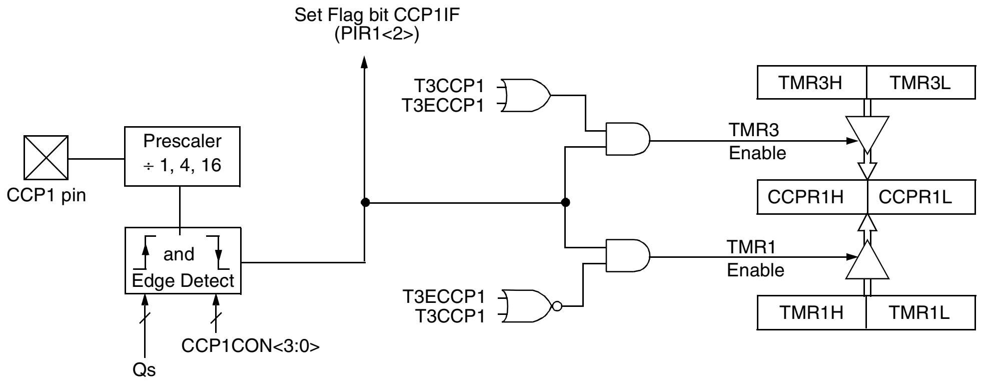
The CAN capture event occurs when a message is received in either of the receive buffers. The CAN module provides a rising edge to the CCP1 module to cause a capture event. This feature is provided to time-stamp the received CAN messages.

This feature is enabled by setting the CANCAP bit of the CAN I/O control register (CIOCON<4>). The message receive signal from the CAN module then takes the place of the events on RC2/CCP1.

**EXAMPLE 15-1: CHANGING BETWEEN CAPTURE PRESCALERS**

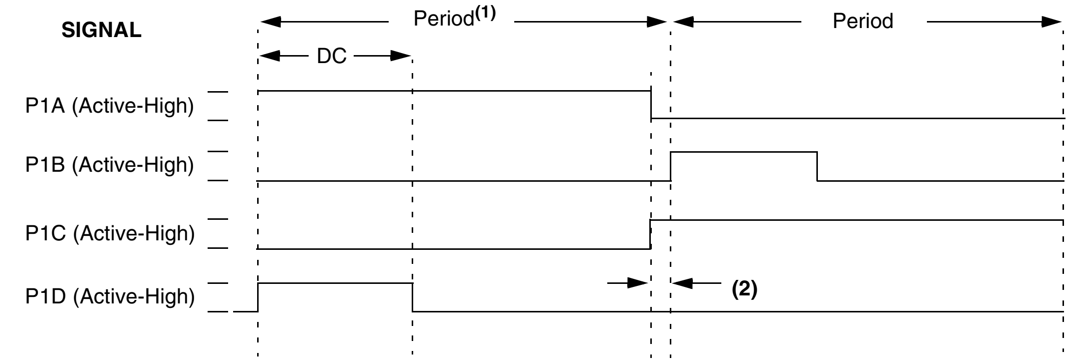
```
CLRF    CCP1CON, F      ; Turn CCP module off
MOVLW   NEW_CAPT_PS     ; Load WREG with the
                        ; new prescaler mode
                        ; value and CCP ON
MOVWF   CCP1CON         ; Load CCP1CON with
                        ; this value
```

**FIGURE 15-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**



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

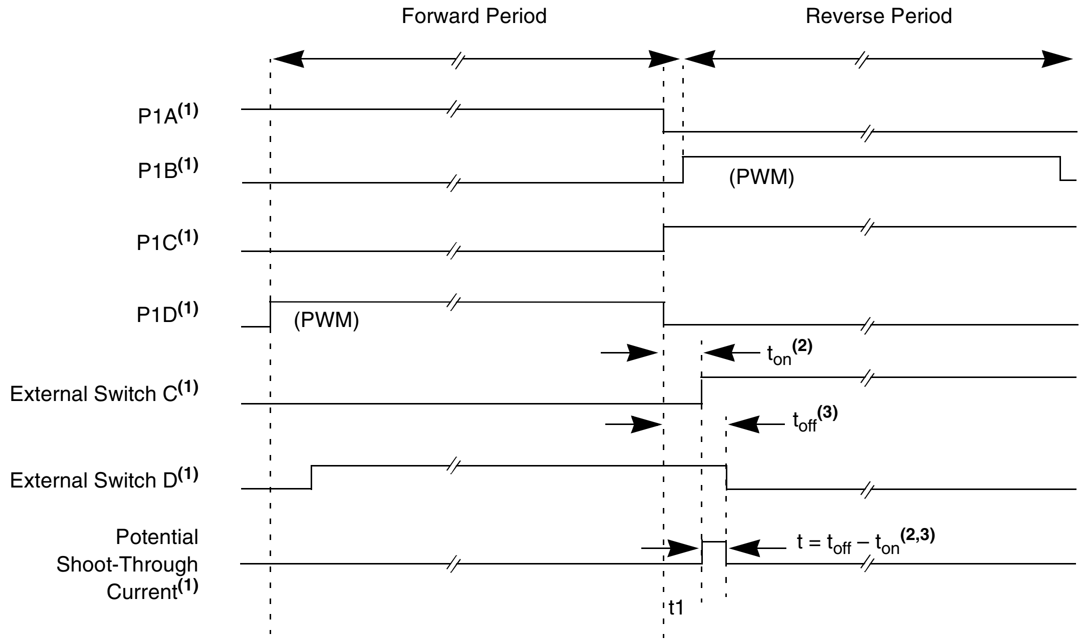
### FIGURE 16-7: PWM DIRECTION CHANGE



**Note 1:** The direction bit in the ECCP1 Control Register (ECCP1CON.EPWM1M1) is written any time during the PWM cycle.

**2:** The P1A and P1C signals switch at intervals of TOSC, 4 TOSC or 16 TOSC, depending on the Timer2 prescaler value earlier when changing direction. The modulated P1B and P1D signals are inactive at this time.

### FIGURE 16-8: PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



**Note 1:** All signals are shown as active-high.

**2:** $t_{on}$ is the turn-on delay of power switch and driver.

**3:** $t_{off}$ is the turn-off delay of power switch and driver.

### REGISTER 19-26: RXMnSIDL: RECEIVE ACCEPTANCE MASK n STANDARD IDENTIFIER MASK, LOW BYTE REGISTERS

| R/W-x | R/W-x | R/W-x | U-0 | U-0 | U-0 | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| SID2 | SID1 | SID0 | — | — | — | EID17 | EID16 |
| bit 7 | | | | | | | bit 0 |

bit 7-5 **SID2:SID0:** Standard Identifier Mask bits or Extended Identifier Mask bits EID20:EID18

bit 4-2 **Unimplemented:** Read as ‘0’

bit 1-0 **EID17:EID16:** Extended Identifier Mask bits

| **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 |

### REGISTER 19-27: RXMnEIDH: RECEIVE ACCEPTANCE MASK n EXTENDED IDENTIFIER MASK, HIGH BYTE REGISTERS

| R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| EID15 | EID14 | EID13 | EID12 | EID11 | EID10 | EID9 | EID8 |
| bit 7 | | | | | | | bit 0 |

bit 7-0 **EID15:EID8:** Extended Identifier Mask bits

| **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 |

### REGISTER 19-28: RXMnEIDL: RECEIVE ACCEPTANCE MASK n EXTENDED IDENTIFIER MASK, LOW BYTE REGISTERS

| R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| EID7 | EID6 | EID5 | EID4 | EID3 | EID2 | EID1 | EID0 |
| bit 7 | | | | | | | bit 0 |

bit 7-0 **EID7:EID0:** Extended Identifier Mask bits

| **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 |

### REGISTER 19-30: BRGCON2: BAUD RATE CONTROL REGISTER 2

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
| SEG2PHTS | SAM | SEG1PH2 | SEG1PH1 | SEG1PH0 | PRSEG2 | PRSEG1 | PRSEG0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **SEG2PHTS:** Phase Segment 2 Time Select bit

1 = Freely programmable
0 = Maximum of PHEG1 or Information Processing Time (IPT), whichever is greater

bit 6 **SAM:** Sample of the CAN bus Line bit

1 = Bus line is sampled three times prior to the sample point
0 = Bus line is sampled once at the sample point

bit 5-3 **SEG1PH2:SEG1PH0:** Phase Segment 1 bits

111 = Phase Segment 1 Time = 8 x TQ
110 = Phase Segment 1 Time = 7 x TQ
101 = Phase Segment 1 Time = 6 x TQ
100 = Phase Segment 1 Time = 5 x TQ
011 = Phase Segment 1 Time = 4 x TQ
010 = Phase Segment 1 Time = 3 x TQ
001 = Phase Segment 1 Time = 2 x TQ
000 = Phase Segment 1 Time = 1 x TQ

bit 2-0 **PRSEG2:PRSEG0:** Propagation Time Select bits

111 = Propagation Time = 8 x TQ
110 = Propagation Time = 7 x TQ
101 = Propagation Time = 6 x TQ
100 = Propagation Time = 5 x TQ
011 = Propagation Time = 4 x TQ
010 = Propagation Time = 3 x TQ
001 = Propagation Time = 2 x TQ
000 = Propagation Time = 1 x TQ

| **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 |

**Note:** This register is accessible in Configuration mode only.

### 19.2.6 CAN INTERRUPT REGISTERS

The registers in this section are the same as described in **Section 8.0 "Interrupts"**. They are duplicated here for convenience.

**REGISTER 19-33: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| IRXIF | WAKIF | ERRIF | TXB2IF | TXB1IF | TXB0IF | RXB1IF | RXB0IF |
| bit 7 | | | | | | | bit 0 |

bit 7 **IRXIF:** CAN Invalid Received Message Interrupt Flag bit
1 = An invalid message has occurred on the CAN bus
0 = No invalid message on CAN bus

bit 6 **WAKIF:** CAN bus Activity Wake-up Interrupt Flag bit
1 = Activity on CAN bus has occurred
0 = No activity on CAN bus

bit 5 **ERRIF:** CAN bus Error Interrupt Flag bit
1 = An error has occurred in the CAN module (multiple sources)
0 = No CAN module errors

bit 4 **TXB2IF:** CAN Transmit Buffer 2 Interrupt Flag bit
1 = Transmit Buffer 2 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 2 has not completed transmission of a message

bit 3 **TXB1IF:** CAN Transmit Buffer 1 Interrupt Flag bit
1 = Transmit Buffer 1 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 1 has not completed transmission of a message

bit 2 **TXB0IF:** CAN Transmit Buffer 0 Interrupt Flag bit
1 = Transmit Buffer 0 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 0 has not completed transmission of a message

bit 1 **RXB1IF:** CAN Receive Buffer 1 Interrupt Flag bit
1 = Receive Buffer 1 has received a new message
0 = Receive Buffer 1 has not received a new message

bit 0 **RXB0IF:** CAN Receive Buffer 0 Interrupt Flag bit
1 = Receive Buffer 0 has received a new message
0 = Receive Buffer 0 has not received a new message

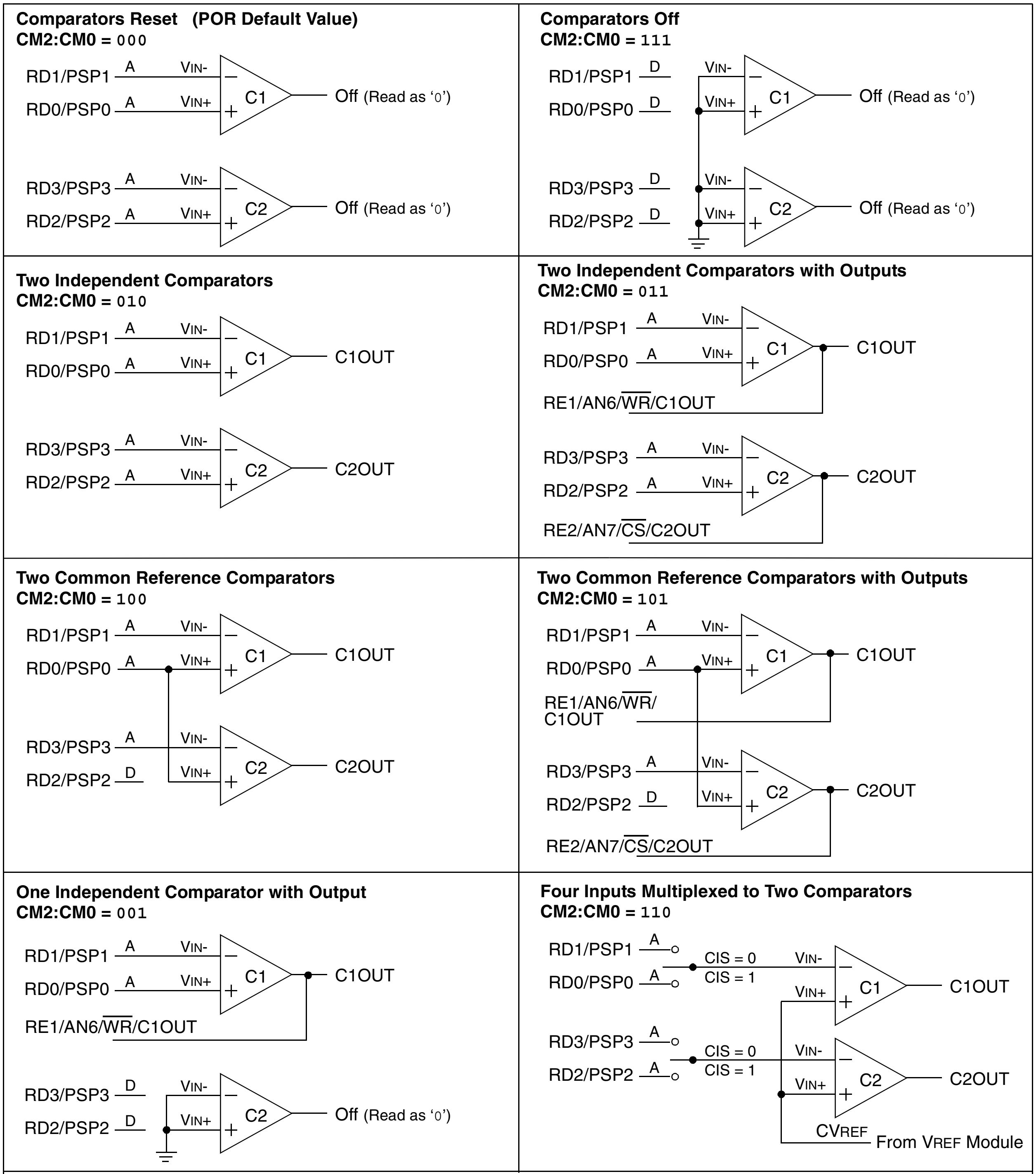
| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

## 21.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 21-1 shows the eight possible modes. The TRISD register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in **Section 27.0 “Electrical Characteristics”**.

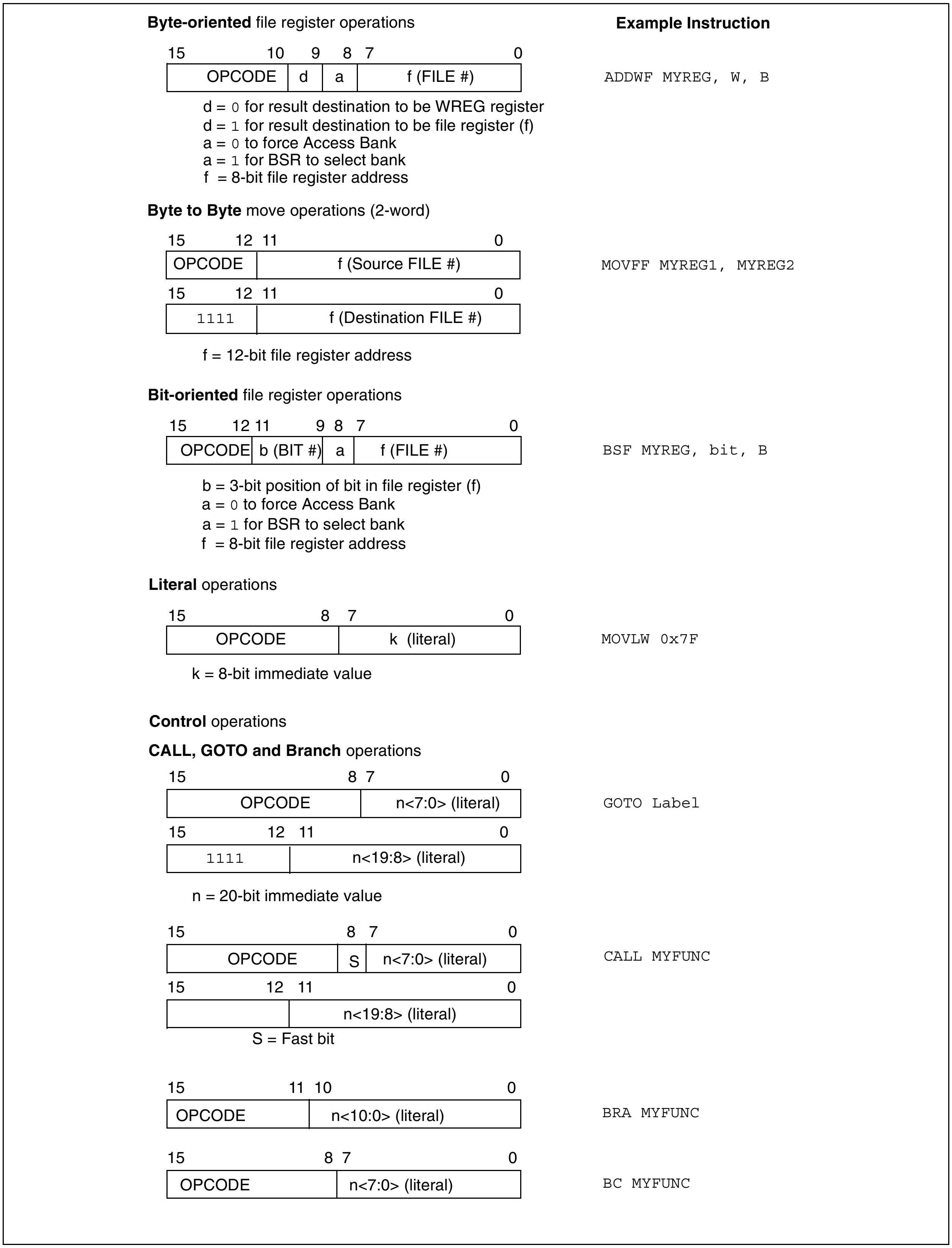
| **Note:** | Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur. |
|---|---|

**FIGURE 21-1: COMPARATOR I/O OPERATING MODES**



A = Analog Input, port reads zeros always
D = Digital Input
CIS (CMCON<3>) is the Comparator Input Switch

**FIGURE 25-1: GENERAL FORMAT FOR INSTRUCTIONS**



**Byte-oriented** file register operations

**Example Instruction**

`ADDWF MYREG, W, B`

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

**Byte to Byte** move operations (2-word)

`MOVFF MYREG1, MYREG2`

f = 12-bit file register address

**Bit-oriented** file register operations

`BSF MYREG, bit, B`

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

**Literal** operations

`MOVLW 0x7F`

k = 8-bit immediate value

**Control** operations

**CALL, GOTO and Branch** operations

`GOTO Label`

n = 20-bit immediate value

`CALL MYFUNC`

S = Fast bit

`BRA MYFUNC`

`BC MYFUNC`

## CLRF Clear f

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

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

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 101a | ffff | ffff |

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

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register ‘f’ | Process Data | Write register ‘f’ |

Example: `CLRF FLAG_REG`

Before Instruction

| | | |
|---|---|---|
| FLAG_REG | = | 0x5A |

After Instruction

| | | |
|---|---|---|
| FLAG_REG | = | 0x00 |

## CLRWDT Clear Watchdog Timer

Syntax: [ *label* ] CLRWDT

Operands: None

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

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

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0100 |

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

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | Process Data | No operation |

Example: `CLRWDT`

Before Instruction

| | | |
|---|---|---|
| WDT Counter | = | ? |

After Instruction

| | | |
|---|---|---|
| WDT Counter | = | 0x00 |
| WDT Postscaler | = | 0 |
| $\overline{\text{TO}}$ | = | 1 |
| PD | = | 1 |

| **MOVFF** | **Move f to f** |
|---|---|
| Syntax: | [ *label* ] MOVFF $f_s$,$f_d$ |
| Operands: | $0 \le f_s \le 4095$<br>$0 \le f_d \le 4095$ |
| Operation: | $(f_s) \rightarrow f_d$ |
| Status Affected: | None |

| Encoding: | | | | |
|---|---|---|---|---|
| 1st word (source) | 1100 | ffff | ffff | $ffff_s$ |
| 2nd word (destin.) | 1111 | ffff | ffff | $ffff_d$ |

Description: The contents of source register '$f_s$' are moved to destination register '$f_d$'. Location of source '$f_s$' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination '$f_d$' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).
`MOVFF` is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The `MOVFF` instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

The `MOVFF` instruction should not be used to modify interrupt settings while any interrupt is enabled (see page 77).

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' (src) | Process Data | No operation |
| Decode | No operation<br>No dummy read | No operation | Write register 'f' (dest) |

Example: `MOVFF REG1, REG2`

Before Instruction

| | | |
|---|---|---|
| REG1 | = | 0x33 |
| REG2 | = | 0x11 |

After Instruction

| | | |
|---|---|---|
| REG1 | = | 0x33 |
| REG2 | = | 0x33 |

| **MOVLB** | **Move Literal to Low Nibble in BSR** |
|---|---|
| Syntax: | [ *label* ] MOVLB k |
| Operands: | $0 \le k \le 255$ |
| Operation: | $k \rightarrow$ BSR |
| Status Affected: | None |

| Encoding: | | | | |
|---|---|---|---|---|
| | 0000 | 0001 | kkkk | kkkk |

Description: The 8-bit literal 'k' is loaded into the Bank Select Register (BSR).

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process Data | Write literal 'k' to BSR |

Example: `MOVLB 5`

Before Instruction

| | | |
|---|---|---|
| BSR register | = | 0x02 |

After Instruction

| | | |
|---|---|---|
| BSR register | = | 0x05 |

### NEGF Negate f

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

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

Operation: $(\overline{f}) + 1 \rightarrow f$

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

Encoding:

| | | | |
|---|---|---|---|
| `0110` | `110a` | `ffff` | `ffff` |

Description: Location ‘f’ is negated using two’s complement. The result is placed in the data memory location ‘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 register ‘f’ |

Example: `NEGF REG, 1`

Before Instruction
REG = `0011 1010` [0x3A]
After Instruction
REG = `1100 0110` [0xC6]

### NOP No Operation

Syntax: [ *label* ] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| `0000`<br>`1111` | `0000`<br>`xxxx` | `0000`<br>`xxxx` | `0000`<br>`xxxx` |

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | No operation | No operation |

Example:

None.

### SUBLW Subtract W from Literal

| | |
|---|---|
| Syntax: | [ *label* ] SUBLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | k – (W) → W |
| Status Affected: | N, OV, C, DC, Z |

Encoding:

| | | | |
|---|---|---|---|
| `0000` | `1000` | `kkkk` | `kkkk` |

| | |
|---|---|
| Description: | W is subtracted from the eight-bit literal ‘k’. 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 1: `SUBLW 0x02`

Before Instruction

| | | | |
|---|---|---|---|
| W | = | 1 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| W | = | 1 | |
| C | = | 1 | ; result is positive |
| Z | = | 0 | |
| N | = | 0 | |

Example 2: `SUBLW 0x02`

Before Instruction

| | | | |
|---|---|---|---|
| W | = | 2 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| W | = | 0 | |
| C | = | 1 | ; result is zero |
| Z | = | 1 | |
| N | = | 0 | |

Example 3: `SUBLW 0x02`

Before Instruction

| | | | |
|---|---|---|---|
| W | = | 3 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| W | = | FF | ; (2’s complement) |
| C | = | 0 | ; result is negative |
| Z | = | 0 | |
| N | = | 1 | |

### SUBWF Subtract W from f

| | |
|---|---|
| Syntax: | [ *label* ] SUBWF f [,d [,a]] |
| Operands: | 0 ≤ f ≤ 255<br>d ∈ [0,1]<br>a ∈ [0,1] |
| Operation: | (f) – (W) → dest |
| Status Affected: | N, OV, C, DC, Z |

Encoding:

| | | | |
|---|---|---|---|
| `0101` | `11da` | `ffff` | `ffff` |

| | |
|---|---|
| Description: | Subtract W from register ‘f’ (2’s complement method). If ‘d’ is ‘`0`’, the result is stored in W. If ‘d’ is ‘`1`’, the result is stored back in register ‘f’ (default). 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 (default). |
| Words: | 1 |
| Cycles: | 1 |

Q Cycle Activity:

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

Example 1: `SUBWF REG`

Before Instruction

| | | | |
|---|---|---|---|
| REG | = | 3 | |
| W | = | 2 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| REG | = | 1 | |
| W | = | 2 | |
| C | = | 1 | ; result is positive |
| Z | = | 0 | |
| N | = | 0 | |

Example 2: `SUBWF REG, W`

Before Instruction

| | | | |
|---|---|---|---|
| REG | = | 2 | |
| W | = | 2 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| REG | = | 2 | |
| W | = | 0 | |
| C | = | 1 | ; result is zero |
| Z | = | 1 | |
| N | = | 0 | |

Example 3: `SUBWF REG`

Before Instruction

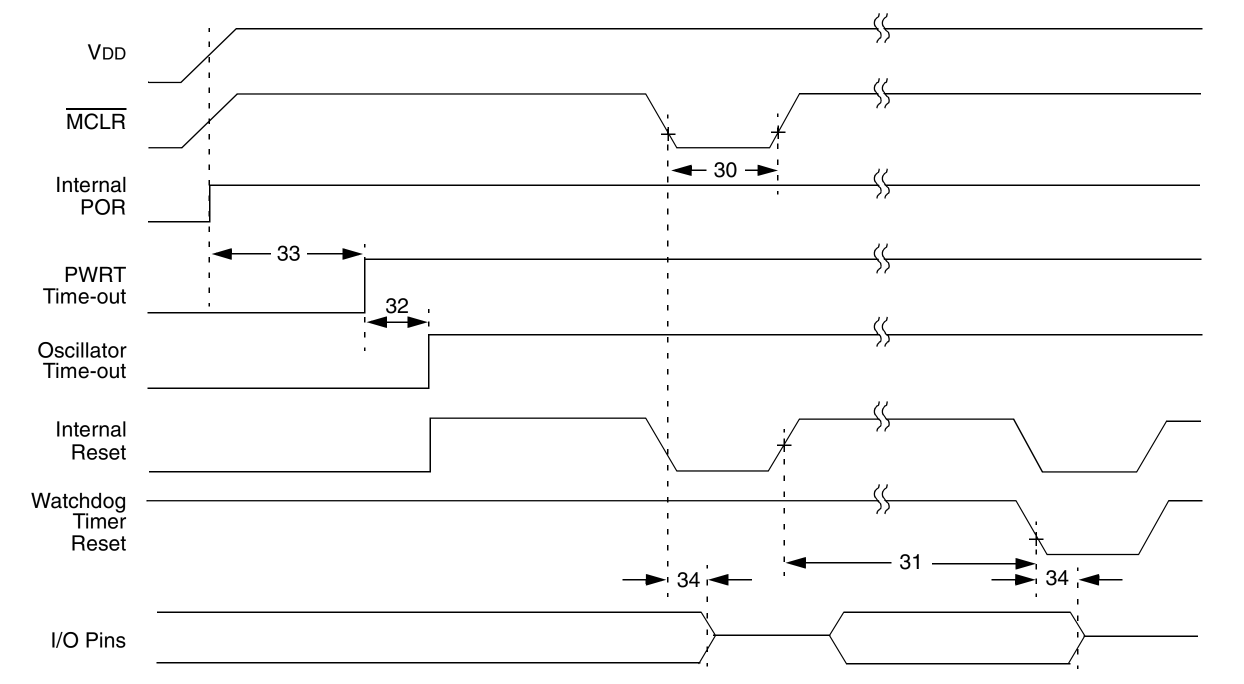
| | | | |
|---|---|---|---|
| REG | = | 0x01 | |
| W | = | 0x02 | |
| C | = | ? | |

After Instruction

| | | | |
|---|---|---|---|
| REG | = | 0xFFh | ;(2’s complement) |
| W | = | 0x02 | |
| C | = | 0x00 | ; result is negative |
| Z | = | 0x00 | |
| N | = | 0x01 | |

NOTES:

FIGURE 27-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



**Note:** Refer to Figure 27-5 for load conditions.

FIGURE 27-9: BROWN-OUT RESET AND LOW-VOLTAGE DETECT TIMING

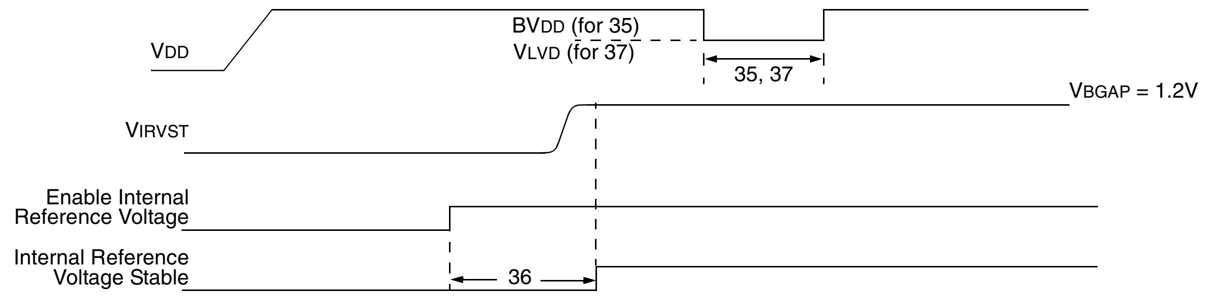
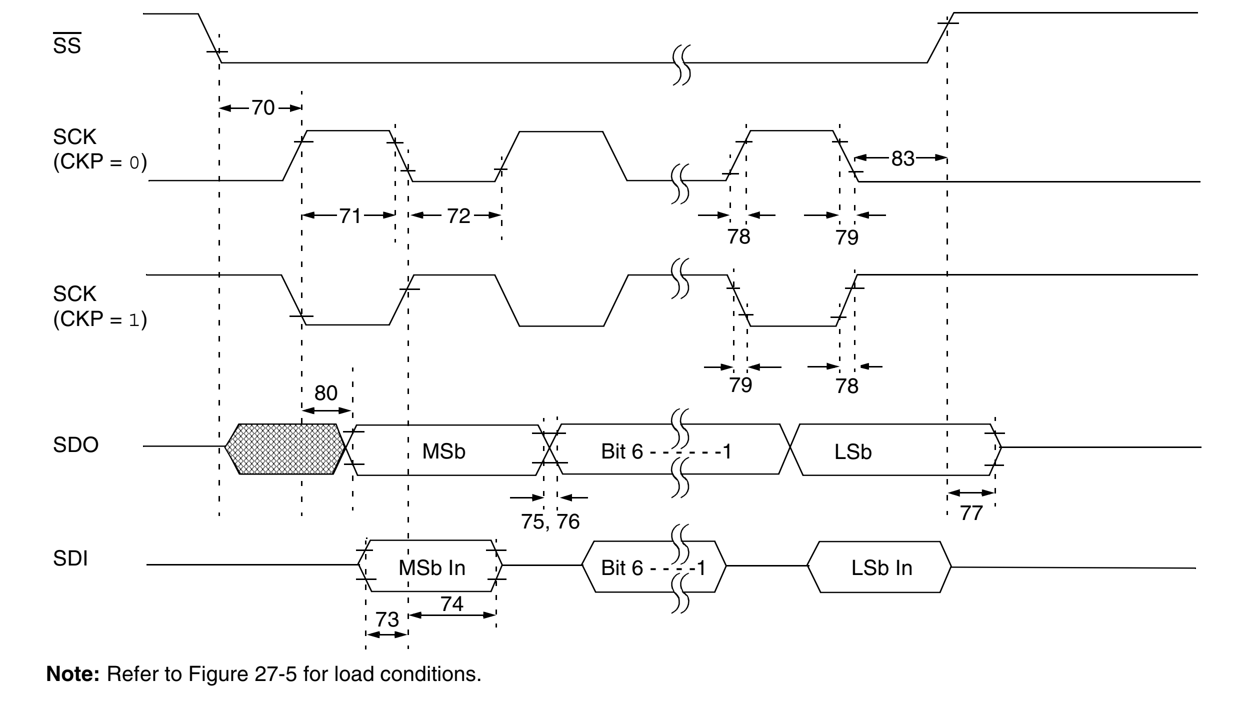

VDD
BVDD (for 35)
VLVD (for 37)
35, 37
VBGAP = 1.2V
VIRVST
Enable Internal Reference Voltage
Internal Reference Voltage Stable
36

TABLE 27-9: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER, BROWN-OUT RESET AND LOW-VOLTAGE DETECT REQUIREMENTS

| Param No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 30 | TmcL | $\overline{\text{MCLR}}$ Pulse Width (low) | 2 | — | — | μs | |
| 31 | TWDT | Watchdog Timer Time-out Period (no prescaler) | 7 | 18 | 33 | ms | |
| 32 | TOST | Oscillation Start-up Timer Period | 1024 TOSC | — | 1024 TOSC | — | TOSC = OSC1 period |
| 33 | TPWRT | Power-up Timer Period | 28 | 72 | 132 | ms | |
| 34 | TIOZ | I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset | — | 2 | — | μs | |
| 35 | TBOR | Brown-out Reset Pulse Width | 200 | — | — | μs | For VDD ≤ BVDD (see D005) |
| 36 | TIRVST | Time for Internal Reference Voltage to become stable | — | 20 | 50 | μs | |
| 37 | TLVD | Low-Voltage Detect Pulse Width | 200 | — | — | μs | For VDD ≤ VLVD (see D420) |

**FIGURE 27-15: EXAMPLE SPI™ SLAVE MODE TIMING (CKE = 0)**



**Note:** Refer to Figure 27-5 for load conditions.

**TABLE 27-15: EXAMPLE SPI™ MODE REQUIREMENTS, SLAVE MODE TIMING (CKE = 0)**

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 70 | TssL2scH,<br>TssL2scL | $\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input | | TCY | — | ns | |
| 71 | TscH | SCK Input High Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 71A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 72 | TscL | SCK Input Low Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 72A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 73 | TdiV2scH,<br>TdiV2scL | Setup Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 73A | TB2B | Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2 | | 1.5 TCY + 40 | — | ns | **(Note 2)** |
| 74 | TscH2diL,<br>TscL2diL | Hold Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 75 | TdoR | SDO Data Output Rise Time | PIC18**F**XX8 | — | 25 | ns | |
| | | | PIC18**LF**XX8 | | 45 | ns | |
| 76 | TdoF | SDO Data Output Fall Time | | — | 25 | ns | |
| 77 | TssH2doZ | $\overline{SS}$ ↑ to SDO Output High-Impedance | | 10 | 50 | ns | |
| 78 | TscR | SCK Output Rise Time (Master mode) | PIC18**F**XX8 | — | 25 | ns | |
| | | | PIC18**LF**XX8 | | 45 | ns | |
| 79 | TscF | SCK Output Fall Time (Master mode) | | — | 25 | ns | |
| 80 | TscH2doV,<br>TscL2doV | SDO Data Output Valid after SCK Edge | PIC18**F**XX8 | — | 50 | ns | |
| | | | PIC18**LF**XX8 | | 100 | ns | |
| 83 | TscH2ssH,<br>TscL2ssH | $\overline{SS}$ ↑ after SCK Edge | | 1.5 TCY + 40 | — | ns | |

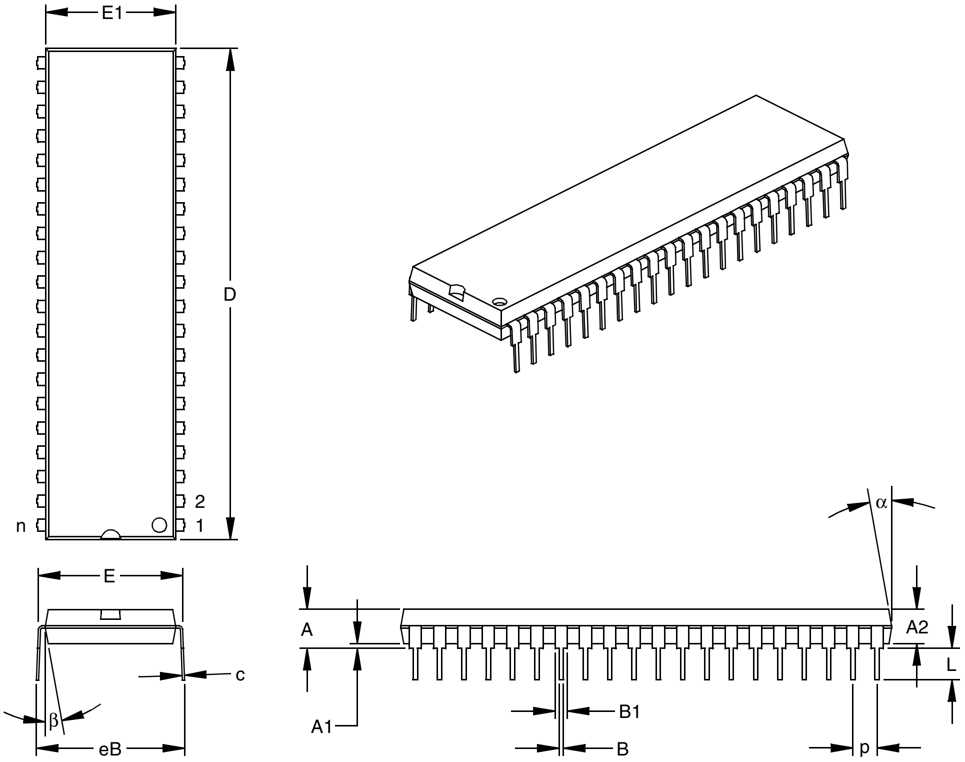
**Note 1:** Requires the use of parameter #73A.

**2:** Only if parameter #71A and #72A are used.

NOTES:

## 40-Lead Plastic Dual In-line (P) – 600 mil Body (PDIP)

| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 40 | | | 40 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .160 | .175 | .190 | 4.06 | 4.45 | 4.83 |
| Molded Package Thickness | A2 | .140 | .150 | .160 | 3.56 | 3.81 | 4.06 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .595 | .600 | .625 | 15.11 | 15.24 | 15.88 |
| Molded Package Width | E1 | .530 | .545 | .560 | 13.46 | 13.84 | 14.22 |
| Overall Length | D | 2.045 | 2.058 | 2.065 | 51.94 | 52.26 | 52.45 |
| Tip to Seating Plane | L | .120 | .130 | .135 | 3.05 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .030 | .050 | .070 | 0.76 | 1.27 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing **§** | eB | .620 | .650 | .680 | 15.75 | 16.51 | 17.27 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

***** Controlling Parameter

**§** Significant Characteristic

**Notes:**

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.

JEDEC Equivalent: MO-011

Drawing No. C04-016

Code Examples
16 x 16 Signed Multiply Routine ................................ 76
16 x 16 Unsigned Multiply Routine ............................ 76
8 x 8 Signed Multiply Routine .................................... 75
8 x 8 Unsigned Multiply Routine ................................ 75
Changing Between Capture Prescalers.................... 125
Data EEPROM Read ................................................ 61
Data EEPROM Refresh Routine................................ 62
Data EEPROM Write ................................................ 61
Erasing a Flash Program Memory Row ...................... 70
Fast Register Stack.................................................... 40
How to Clear RAM (Bank 1) Using
Indirect Addressing ............................................ 55
Initializing PORTA...................................................... 93
Initializing PORTB...................................................... 96
Initializing PORTC.................................................... 100
Initializing PORTD.................................................... 102
Initializing PORTE.................................................... 104
Loading the SSPBUF Register ................................ 146
Reading a Flash Program
Memory Word .................................................... 69
Saving Status, WREG and BSR
Registers in RAM ............................................... 92
WIN and ICODE Bits Usage in
Interrupt Service Routine to
Access TX/RX Buffers ..................................... 203
Writing to Flash Program Memory ........................ 72–73
Code Protection ............................................................. 265
COMF ............................................................................ 298
Comparator Module ....................................................... 249
Analog Input Connection Considerations................. 253
Associated Registers ............................................... 254
Configuration............................................................ 250
Effects of a Reset..................................................... 253
External Reference Signal ....................................... 251
Internal Reference Signal ........................................ 251
Interrupts.................................................................. 252
Operation ................................................................. 251
Operation During Sleep ........................................... 253
Outputs .................................................................... 251
Reference ................................................................ 251
Response Time........................................................ 251
Comparator Specifications............................................. 340
Comparator Voltage Reference Module .......................... 255
Accuracy/Error ......................................................... 256
Associated Registers ............................................... 257
Configuring............................................................... 255
Connection Considerations...................................... 256
Effects of a Reset..................................................... 256
Operation During Sleep ........................................... 256
Compare (CCP Module) ................................................. 126
CCP1 Pin Configuration........................................... 126
CCPR1 and ECCPR1 Registers .............................. 126
Registers Associated with Capture,
Compare, Timer1 and Timer3.......................... 127
Software Interrupt .................................................... 126
Special Event Trigger....................... 115, 121, 126, 248
Timer1/Timer3 Mode Selection................................ 126
Compare (ECCP Module) ............................................... 133
Registers Associated with Enhanced
Capture, Compare, Timer1 and Timer3........... 133
Special Event Trigger............................................... 133
Compatible 10-Bit Analog-to-Digital
Converter (A/D) Module. *See* A/D.
Configuration Mode (CAN Module)................................. 226
CPFSEQ ......................................................................... 298
CPFSGT ......................................................................... 299
CPFSLT.......................................................................... 299
Crystal Oscillator
Capacitor Selection .................................................. 18

D

Data EEPROM Memory.................................................... 59
Associated Registers................................................ 63
EEADR Register....................................................... 59
EECON1 Register .................................................... 59
EECON2 Register .................................................... 59
Operation During Code-Protect ................................ 62
Protection Against Spurious Writes ........................... 62
Reading .................................................................... 61
Usage ....................................................................... 62
Write Verify ............................................................... 62
Writing to .................................................................. 61
Data Memory .................................................................. 44
General Purpose Registers ....................................... 44
Special Function Registers........................................ 44
Data Memory Map
PIC18F248/448 ......................................................... 45
PIC18F258/458 ......................................................... 46
DAW .............................................................................. 300
DC and AC Characteristics
Graphs and Tables .................................................. 361
DC Characteristics.......................................................... 332
EEPROM and Enhanced Flash ............................... 339
PIC18FXX8 (Ind., Ext.) and
PIC18LFXX8 (Ind.) .......................................... 336
DCFSNZ ........................................................................ 301
DECF ............................................................................. 300
DECFSZ ........................................................................ 301
Demonstration Boards
PICDEM 1................................................................ 326
PICDEM 17.............................................................. 327
PICDEM 18R ........................................................... 327
PICDEM 2 Plus........................................................ 326
PICDEM 3................................................................ 326
PICDEM 4................................................................ 326
PICDEM LIN ............................................................ 327
PICDEM USB .......................................................... 327
PICDEM.net Internet/Ethernet................................. 326
Development Support .................................................... 323
Device Differences.......................................................... 385
Device Migrations ........................................................... 386
Device Overview................................................................. 7
Features ...................................................................... 7
Direct Addressing ............................................................. 56
Disable Mode (CAN Module) .......................................... 226

E

Electrical Characteristics ................................................ 329
Enhanced Capture/Compare/PWM (ECCP)..................... 131
Auto-Shutdown ........................................................ 142
Capture Mode. *See* Capture
(ECCP Module).
Compare Mode. *See* Compare
(ECCP Module).
ECCPR1H Register................................................. 132
ECCPR1L Register.................................................. 132
Interaction of CCP1 and
ECCP1 Modules .............................................. 132
Pin Assignments for Various Modes........................ 132
PWM Mode. *See* PWM (ECCP Module).
Timer Resources ..................................................... 132