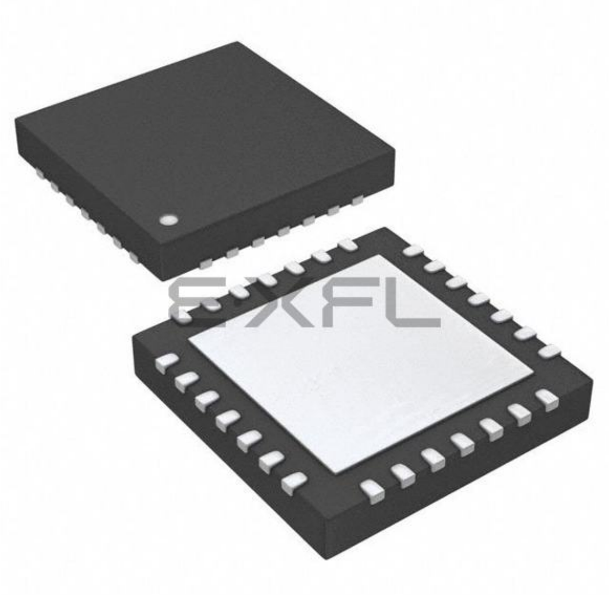


Welcome to E-XFL.COM

## 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, HLVD, POR, PWM, WDT |
| Number of I/O | 25 |
| 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 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-VQFN Exposed Pad |
| Supplier Device Package | 28-QFN (6x6) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18f2580t-i-ml |

### TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2480/2580/4480/4580)

| 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: |
|---|---|---|---|---|---|---|---|---|---|---|
| TOSU | — | — | — | Top-of-Stack Upper Byte (TOS<20:16>) | | | | | ---0 0000 | 55, 68 |
| TOSH | Top-of-Stack High Byte (TOS<15:8>) | | | | | | | | 0000 0000 | 55, 68 |
| TOSL | Top-of-Stack Low Byte (TOS<7:0>) | | | | | | | | 0000 0000 | 55, 68 |
| STKPTR | STKFUL | STKUNF | — | Return Stack Pointer | | | | | 00-0 0000 | 55, 69 |
| PCLATU | — | — | bit 21(1) | Holding Register for PC<20:16> | | | | | ---0 0000 | 55, 68 |
| PCLATH | Holding Register for PC<15:8> | | | | | | | | 0000 0000 | 55, 68 |
| PCL | PC Low Byte (PC<7:0>) | | | | | | | | 0000 0000 | 55, 68 |
| TBLPTRU | — | — | bit 21 | Program Memory Table Pointer Upper Byte (TBLPTR<20:16>) | | | | | --00 0000 | 55, 109 |
| TBLPTRH | Program Memory Table Pointer High Byte (TBLPTR<15:8>) | | | | | | | | 0000 0000 | 55, 109 |
| TBLPTRL | Program Memory Table Pointer Low Byte (TBLPTR<7:0>) | | | | | | | | 0000 0000 | 55, 109 |
| TABLAT | Program Memory Table Latch | | | | | | | | 0000 0000 | 55, 109 |
| PRODH | Product Register High Byte | | | | | | | | xxxx xxxx | 55, 117 |
| PRODL | Product Register Low Byte | | | | | | | | xxxx xxxx | 55, 117 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 55, 121 |
| INTCON2 | $\overline{\text{RBPU}}$ | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP | 1111 -1-1 | 55, 122 |
| INTCON3 | INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF | 11-0 0-00 | 55, 123 |
| INDF0 | Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register) | | | | | | | | N/A | 55, 96 |
| POSTINC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| POSTDEC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| PREINC0 | Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| PLUSW0 | Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register), value of FSR0 offset by W | | | | | | | | N/A | 55, 97 |
| FSR0H | — | — | — | — | Indirect Data Memory Address Pointer 0 High | | | | ---- xxxx | 55, 96 |
| FSR0L | Indirect Data Memory Address Pointer 0 Low Byte | | | | | | | | xxxx xxxx | 55, 96 |
| WREG | Working Register | | | | | | | | xxxx xxxx | 55 |
| INDF1 | Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register) | | | | | | | | N/A | 55, 96 |
| POSTINC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| POSTDEC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| PREINC1 | Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) | | | | | | | | N/A | 55, 97 |
| PLUSW1 | Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register), value of FSR1 offset by W | | | | | | | | N/A | 55, 97 |
| FSR1H | — | — | — | — | Indirect Data Memory Address Pointer 1 High | | | | ---- xxxx | 55, 96 |
| FSR1L | Indirect Data Memory Address Pointer 1 Low Byte | | | | | | | | xxxx xxxx | 55, 96 |
| BSR | — | — | — | — | Bank Select Register | | | | ---- 0000 | 56, 73 |
| INDF2 | Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register) | | | | | | | | N/A | 56, 96 |
| POSTINC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register) | | | | | | | | N/A | 56, 97 |
| POSTDEC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register) | | | | | | | | N/A | 56, 97 |
| PREINC2 | Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) | | | | | | | | N/A | 56, 97 |
| PLUSW2 | Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register), value of FSR2 offset by W | | | | | | | | N/A | 56, 97 |

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

**Note 1:** Bit 21 of the PC is only available in Test mode and Serial Programming modes.

**2:** The SBOREN bit is only available when CONFIG2L<1:0> = 01; otherwise, it is disabled and reads as '0'. See **Section 5.4 "Brown-out Reset (BOR)"**.

**3:** These registers and/or bits are not implemented on PIC18F2X80 devices and are read as '0'. Reset values are shown for PIC18F4X80 devices; individual unimplemented bits should be interpreted as '—'.

**4:** The PLLEN bit is only available in specific oscillator configuration; otherwise, it is disabled and reads as '0'. See **Section 3.6.4 "PLL in INTOSC Modes"**.

**5:** The RE3 bit is only available when Master Clear Reset is disabled (CONFIG3H<7> = 0); otherwise, RE3 reads as '0'. This bit is read-only.

**6:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.

**7:** CAN bits have multiple functions depending on the selected mode of the CAN module.

**8:** This register reads all '0's until the ECAN™ technology is set up in Mode 1 or Mode 2.

**9:** These registers are available on PIC18F4X80 devices only.

### 6.6.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

The use of Indexed Literal Offset Addressing mode effectively changes how the lower half of Access RAM (00h to 7Fh) is mapped. Rather than containing just the contents of the bottom half of Bank 0, this mode maps the contents from Bank 0 and a user-defined "window" that can be located anywhere in the data memory space. The value of FSR2 establishes the lower boundary of the addresses mapped into the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described (see **Section 6.3.2 "Access Bank"**). An example of Access Bank remapping in this addressing mode is shown in Figure 6-10.

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is '1') will continue to use Direct Addressing as before. Any indirect or indexed operation that explicitly uses any of the indirect file operands (including FSR2) will continue to operate as standard Indirect Addressing. Any instruction that uses the Access Bank, but includes a register address of greater than 05Fh, will use Direct Addressing and the normal Access Bank map.

### 6.6.4 BSR IN INDEXED LITERAL OFFSET MODE

Although the Access Bank is remapped when the extended instruction set is enabled, the operation of the BSR remains unchanged. Direct Addressing using the BSR to select the data memory bank operates in the same manner as previously described.

**FIGURE 6-10: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING**

**Example Situation:**

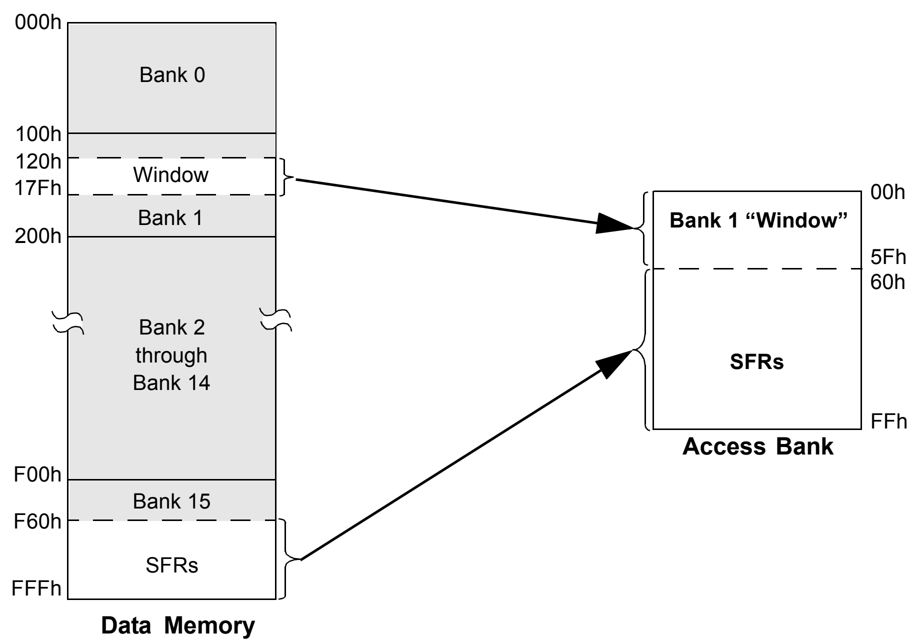
`ADDWF f, d, a`

FSR2H:FSR2L = 120h

Locations in the region from the FSR2 Pointer (120h) to the pointer plus 05Fh (17Fh) are mapped to the bottom of the Access RAM (000h-05Fh).

Special File Registers at F60h through FFFh are mapped to 60h through FFh, as usual.

Bank 0 addresses below 5Fh are not available in this mode. They can still be addressed by using the BSR.



**NOTES:**

## 8.3 Reading the Data EEPROM Memory

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

The basic process is shown in Example 8-1.

## 8.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register. The sequence in Example 8-2 must be followed to initiate the write cycle.

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

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

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

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

## 8.5 Write Verify

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

**EXAMPLE 8-1: DATA EEPROM READ**

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

**EXAMPLE 8-2: DATA EEPROM WRITE**

```
          MOVLW   DATA_EE_ADDR     ;
          MOVWF   EEADR            ; Data Memory Address to write
          MOVLW   DATA_EE_DATA     ;
          MOVWF   EEDATA           ; Data Memory Value to write
          BCF     EECON1, EEPGD    ; Point to DATA memory
          BCF     EECON1, CFGS     ; Access EEPROM
          BSF     EECON1, WREN     ; Enable writes

          BCF     INTCON, GIE      ; Disable Interrupts
          MOVLW   55h              ;
Required  MOVWF   EECON2           ; Write 55h
Sequence  MOVLW   0AAh             ;
          MOVWF   EECON2           ; Write 0AAh
          BSF     EECON1, WR       ; Set WR bit to begin write
          BTFSC   EECON1, WR       ; Wait for write to complete
          BRA     $-2
          BSF     INTCON, GIE      ; Enable Interrupts

                                   ; User code execution
          BCF     EECON1, WREN     ; Disable writes on write complete (EEIF set)
```

## 10.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Priority registers (IPR1, IPR2). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

**REGISTER 10-10: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP |
| 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 **PSPIP:** Parallel Slave Port Read/Write Interrupt Priority bit(1)
1 = High priority
0 = Low priority

bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RCIP:** EUSART Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TXIP:** EUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSPIP:** Master Synchronous Serial Port Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

**Note 1:** This bit is reserved on PIC18F2X80 devices; always maintain this bit set.

**TABLE 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page: |
|---|---|---|---|---|---|---|---|---|---|
| PORTD(1) | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 58 |
| LATD(1) | LATD Output Latch Register | | | | | | | | 58 |
| TRISD(1) | PORTD Data Direction Register | | | | | | | | 58 |
| TRISE(1) | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 58 |
| ECCP1CON(1) | EPWM1M1 | EPWM1M0 | EDC1B1 | EDC1B0 | ECCP1M3 | ECCP1M2 | ECCP1M1 | ECCP1M0 | 57 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by PORTD.

**Note 1:** These registers are available on PIC18F4X80 devices only.

### TABLE 16-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

| 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 | 55 |
| RCON | IPEN | SBOREN(3) | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ | 56 |
| IPR1 | PSPIP | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 58 |
| PIR1 | PSPIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 58 |
| PIE1 | PSPIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 58 |
| IPR2 | OSCFIP | CMIP(2) | — | EEIP | BCLIP | HLVDIP | TMR3IP | ECCP1IP(2) | 58 |
| PIR2 | OSCFIF | CMIF(2) | — | EEIF | BCLIF | HLVDIF | TMR3IF | ECCP1IF(2) | 58 |
| PIE2 | OSCFIE | CMIE(2) | — | EEIE | BCLIE | HLVDIE | TMR3IE | ECCP1IE(2) | 57 |
| TRISB | PORTB Data Direction Register | | | | | | | | 58 |
| TRISC | PORTC Data Direction Register | | | | | | | | 58 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | 56 |
| TMR1H | Timer1 Register High Byte | | | | | | | | 56 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | TMR1CS | TMR1ON | 56 |
| TMR3H | Timer3 Register High Byte | | | | | | | | 57 |
| TMR3L | Timer3 Register Low Byte | | | | | | | | 57 |
| T3CON | RD16 | T3ECCP1(1) | T3CKPS1 | T3CKPS0 | T3CCP1(1) | $\overline{T3SYNC}$ | TMR3CS | TMR3ON | 57 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | 57 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | 57 |
| CCP1CON | — | — | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 57 |
| ECCPR1L(1) | Enhanced Capture/Compare/PWM Register 1 Low Byte | | | | | | | | 57 |
| ECCPR1H(1) | Enhanced Capture/Compare/PWM Register 1 High Byte | | | | | | | | 57 |
| ECCP1CON(1) | EPWM1M1 | EPWM1M0 | EDC1B1 | EDC1B0 | ECCP1M3 | ECCP1M2 | ECCP1M1 | ECCP1M0 | 57 |

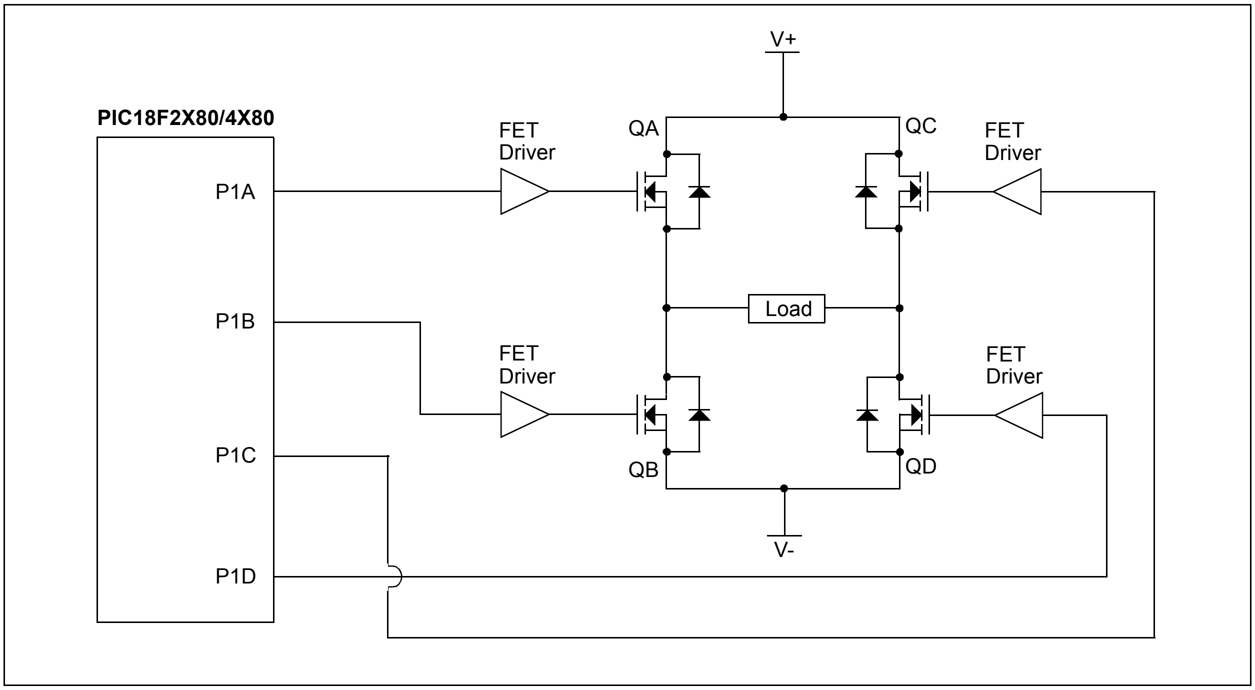
**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by capture, compare, Timer1 or Timer3.

**Note 1:** These bits or registers are available on PIC18F4X80 devices only.

**2:** These bits are available on PIC18F4X80 devices and reserved on PIC18F2X80 devices.

**3:** The SBOREN bit is only available when CONFIG2L<1:0> = 01; otherwise, it is disabled and reads as '0'. See **Section 5.4 "Brown-out Reset (BOR)"**.

**FIGURE 17-7: EXAMPLE OF FULL-BRIDGE OUTPUT APPLICATION**




#### 17.4.5.1 Direction Change in Full-Bridge Output Mode

In the Full-Bridge Output mode, the EPWM1M1 bit in the CCP1CON register allows the user to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will assume the new direction on the next PWM cycle.

Just before the end of the current PWM period, the modulated outputs (P1B and P1D) are placed in their inactive state, while the unmodulated outputs (P1A and P1C) are switched to drive in the opposite direction. This occurs in a time interval of (4 TOSC * (Timer2 Prescale Value) before the next PWM period begins. The Timer2 prescaler will be either 1, 4 or 16, depending on the value of the T2CKPS bits (T2CON<1:0>). During the interval from the switch of the unmodulated outputs to the beginning of the next period, the modulated outputs (P1B and P1D) remain inactive. This relationship is shown in Figure 17-8.

Note that in the Full-Bridge Output mode, the CCP1 module does not provide any dead-band delay. In general, since only one output is modulated at all times, dead-band delay is not required. However, there is a situation where a dead-band delay might be required. This situation occurs when both of the following conditions are true:

1. The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
2. The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn-on time.

Figure 17-9 shows an example where the PWM direction changes from forward to reverse at a near 100% duty cycle. At time t1, the outputs, P1A and P1D, become inactive, while output, P1C, becomes active. In this example, since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current may flow through power devices, QC and QD (see Figure 17-7), for the duration of 't'. The same phenomenon will occur to power devices, QA and QB, for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, one of the following requirements must be met:

1. Reduce PWM for a PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

Other options to prevent shoot-through current may exist.

### 18.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 18-2) is to broadcast data by the software protocol.

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

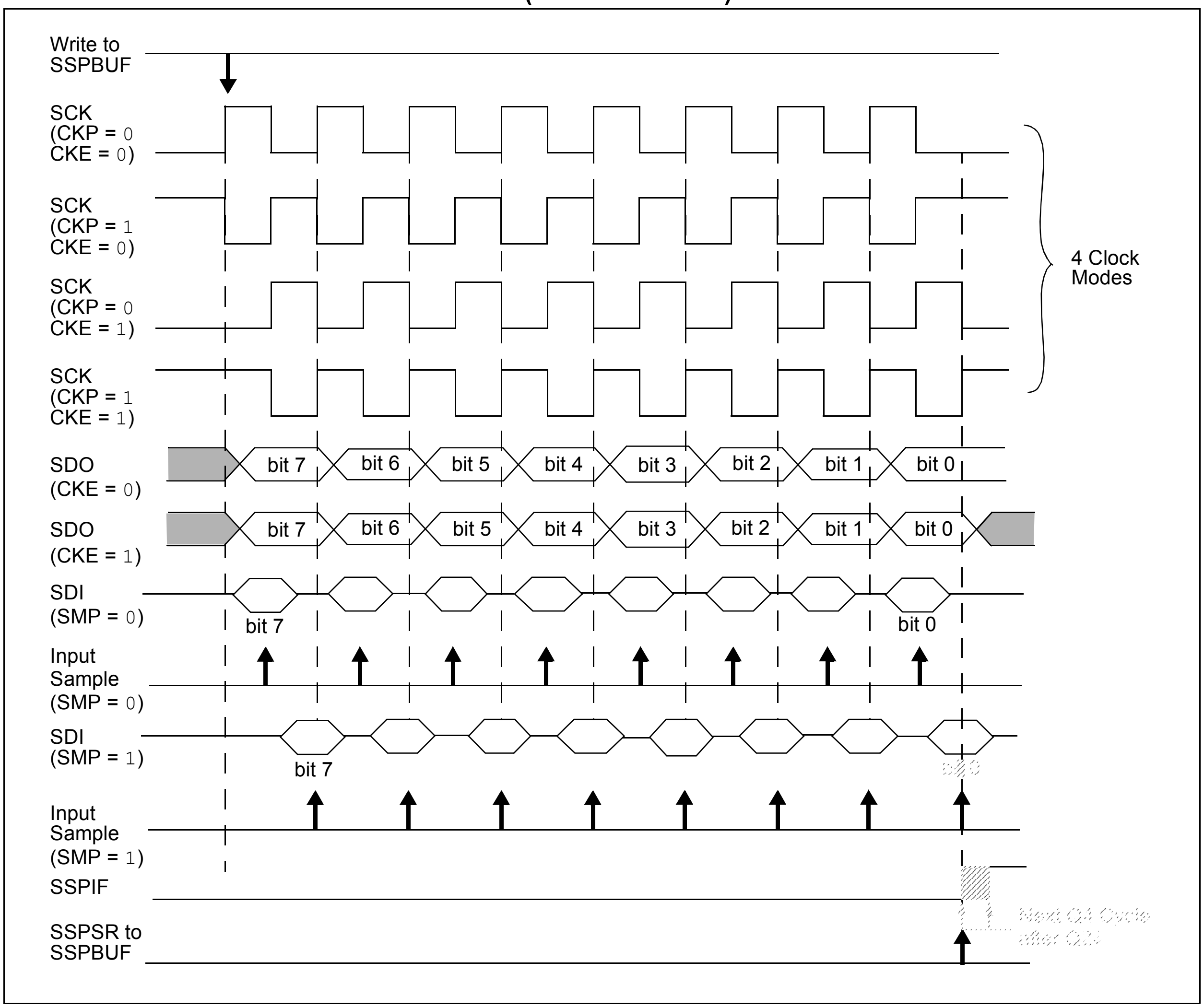
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 18-3, Figure 18-5 and Figure 18-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

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

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

Figure 18-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

**FIGURE 18-3: SPI MODE WAVEFORM (MASTER MODE)**



### REGISTER 18-3: SSPSTAT: MSSP STATUS REGISTER (I²C™ MODE)

| R/W-0 | R/W-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| SMP | CKE | D/$\overline{A}$ | P(1) | S(1) | R/$\overline{W}$(2,3) | UA | BF |
| 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 **SMP:** Slew Rate Control bit
In Master or Slave mode:
1 = Slew rate control disabled for Standard Speed mode (100 kHz and 1 MHz)
0 = Slew rate control enabled for High-Speed mode (400 kHz)

bit 6 **CKE:** SMBus Select bit
In Master or Slave mode:
1 = Enable SMBus specific inputs
0 = Disable SMBus specific inputs

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit
In Master mode:
Reserved.
In Slave mode:
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit(1)
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last

bit 3 **S:** Start bit(1)
1 = Indicates that a Start bit has been detected last
0 = Start bit was not detected last

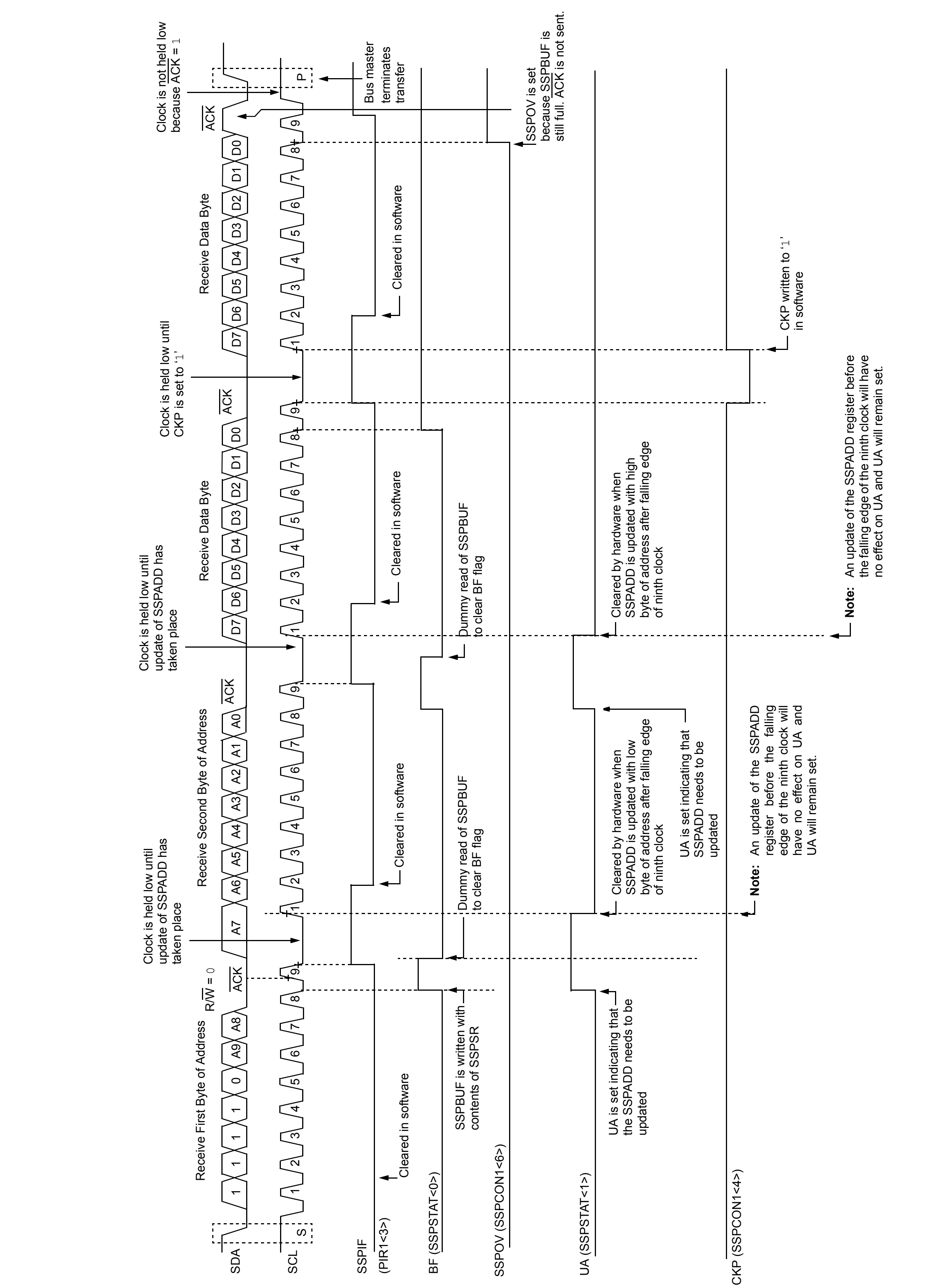
bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit (I²C mode only)(2,3)
In Slave mode:
1 = Read
0 = Write
In Master mode:
1 = Transmit is in progress
0 = Transmit is not in progress

bit 1 **UA:** Update Address bit (10-Bit Slave mode only)
1 = Indicates that the user needs to update the address in the SSPADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit
In Receive mode:
1 = Receive complete, SSPBUF is full
0 = Receive is not complete, SSPBUF is empty
In Transmit mode:
1 = Data transmit in progress (does not include the $\overline{\text{ACK}}$ and Stop bits), SSPBUF is full
0 = Data transmit complete (does not include the $\overline{\text{ACK}}$ and Stop bits), SSPBUF is empty

**Note 1:** This bit is cleared on Reset and when SSPEN is cleared.
**2:** This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not $\overline{\text{ACK}}$ bit.
**3:** ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Idle mode.

**FIGURE 18-14: I²C™ SLAVE MODE TIMING SEN = 1 (RECEPTION, 10-BIT ADDRESS)**



The analog reference voltage is software-selectable to either the device's positive and negative supply voltage (AVDD and AVSS), or the voltage level on the RA3/AN3/VREF+ and RA2/AN2/VREF-/CVREF pins.

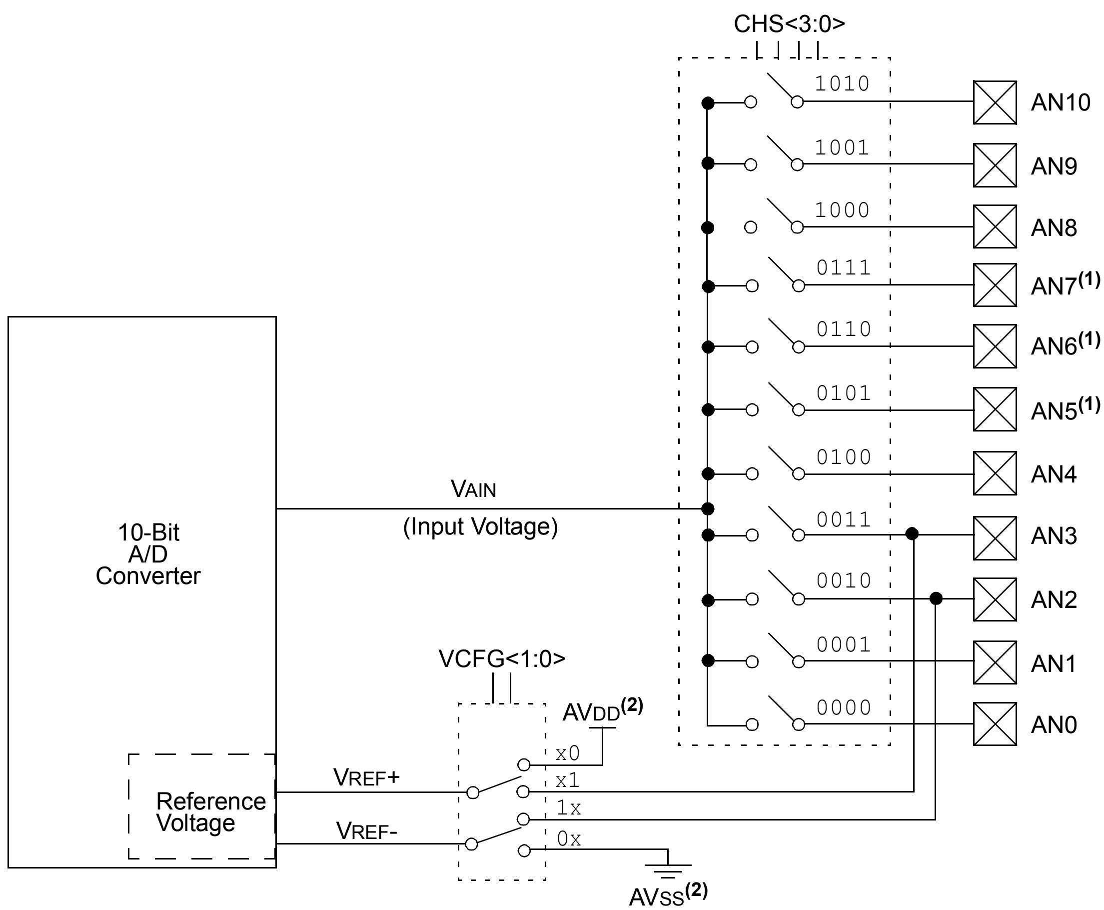
The A/D Converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

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

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

Each port pin associated with the A/D Converter can be configured as an analog input, or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH/ADRESL registers, the GO/$\overline{\text{DONE}}$ bit (ADCON0 register) is cleared and A/D Interrupt Flag bit, ADIF, is set. The block diagram of the A/D module is shown in Figure 20-1.

**FIGURE 20-1: A/D BLOCK DIAGRAM**

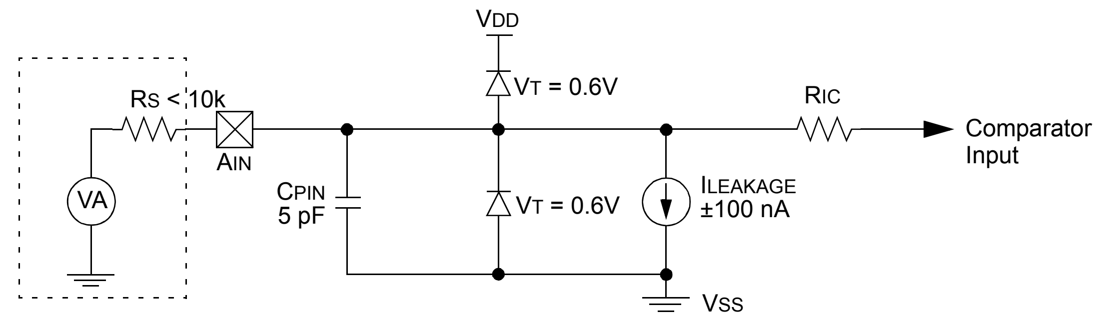



**Note 1:** Channels, AN5 through AN7, are not available on PIC18F2X80 devices.
**2:** I/O pins have diode protection to VDD and VSS.

## 21.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 21-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

**FIGURE 21-4: COMPARATOR ANALOG INPUT MODEL**




**Legend:**
- CPIN = Input Capacitance
- VT = Threshold Voltage
- ILEAKAGE = Leakage Current at the pin due to various junctions
- RIC = Interconnect Resistance
- RS = Source Impedance
- VA = Analog Voltage

**TABLE 21-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page: |
|---|---|---|---|---|---|---|---|---|---|
| CMCON(3) | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 57 |
| CVRCON(3) | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 57 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 58 |
| IPR2 | OSCFIP | CMIP(2) | — | EEIP | BCLIP | HLVDIP | TMR3IP | ECCP1IP | 57 |
| PIR2 | OSCFIF | CMIF(2) | — | EEIF | BCLIF | HLVDIF | TMR3IF | ECCP1IF | 58 |
| PIE2 | OSCFIE | CMIE(2) | — | EEIE | BCLIE | HLVDIE | TMR3IE | ECCP1IE | 58 |
| PORTA | RA7(1) | RA6(1) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 58 |
| LATA | LATA7(1) | LATA6(1) | LATA Data Output Register | | | | | | 58 |
| TRISA | TRISA7(1) | TRISA6(1) | PORTA Data Direction Register | | | | | | 58 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the comparator module.

**Note 1:** PORTA pins are enabled based on oscillator configuration.

**2:** These bits are available in PIC18F4X80 devices and reserved in PIC18F2X80 devices.

**3:** These registers are unimplemented on PIC18F2X80 devices.

**REGISTER 24-3: ECANCON: ENHANCED CAN CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| MDSEL1[(1)] | MDSEL0[(1)] | FIFOWM[(2)] | EWIN4 | EWIN3 | EWIN2 | EWIN1 | EWIN0 |
| 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 **MDSEL<1:0>:** Mode Select bits[(1)]
`00` = Legacy mode (Mode 0, default)
`01` = Enhanced Legacy mode (Mode 1)
`10` = Enhanced FIFO mode (Mode 2)
`11` = Reserved

bit 5 **FIFOWM:** FIFO High Water Mark bit[(2)]
`1` = Will cause FIFO interrupt when one receive buffer remains
`0` = Will cause FIFO interrupt when four receive buffers remain[(3)]

bit 4-0 **EWIN<4:0>:** Enhanced Window Address bits
These bits map the group of 16 banked CAN SFRs into Access Bank addresses 0F60-0F6Dh. The exact group of registers to map is determined by the binary value of these bits.

Mode 0:
**Unimplemented:** Read as ‘0’

Mode 1, 2:
`00000` = Acceptance Filters 0, 1, 2 and BRGCON2, 3
`00001` = Acceptance Filters 3, 4, 5 and BRGCON1, CIOCON
`00010` = Acceptance Filter Masks, Error and Interrupt Control
`00011` = Transmit Buffer 0
`00100` = Transmit Buffer 1
`00101` = Transmit Buffer 2
`00110` = Acceptance Filters 6, 7, 8
`00111` = Acceptance Filters 9, 10, 11
`01000` = Acceptance Filters 12, 13, 14
`01001` = Acceptance Filters 15
`01010-01110` = Reserved
`01111` = RXINT0, RXINT1
`10000` = Receive Buffer 0
`10001` = Receive Buffer 1
`10010` = TX/RX Buffer 0
`10011` = TX/RX Buffer 1
`10100` = TX/RX Buffer 2
`10101` = TX/RX Buffer 3
`10110` = TX/RX Buffer 4
`10111` = TX/RX Buffer 5
`11000-11111` = Reserved

**Note 1:** These bits can only be changed in Configuration mode. See Register 24-1 to change to Configuration mode.
**2:** This bit is used in Mode 2 only.
**3:** If FIFO is configured to contain four or less buffers, then the FIFO interrupt will trigger.

### REGISTER 24-21: RXERRCNT: RECEIVE ERROR COUNT REGISTER

| R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| REC7 | REC6 | REC5 | REC4 | REC3 | REC2 | REC1 | REC0 |
| 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 7-0 **REC<7:0>:** Receive Error Counter bits

This register contains the receive error value as defined by the CAN specifications. When RXERRCNT > 127, the module will go into an error-passive state. RXERRCNT does not have the ability to put the module in “bus-off” state.

### EXAMPLE 24-5: READING A CAN MESSAGE

```
; Need to read a pending message from RXB0 buffer.
; To receive any message, filter, mask and RXM1:RXM0 bits in RXB0CON registers must be
; programmed correctly.
;
; Make sure that there is a message pending in RXB0.
BTFSS   RXB0CON, RXFUL               ; Does RXB0 contain a message?
BRA     NoMessage                    ; No. Handle this situation...
; We have verified that a message is pending in RXB0 buffer.
; If this buffer can receive both Standard or Extended Identifier messages,
; identify type of message received.
BTFSS   RXB0SIDL, EXID               ; Is this Extended Identifier?
BRA     StandardMessage              ; No. This is Standard Identifier message.
                                     ; Yes. This is Extended Identifier message.
; Read all 29-bits of Extended Identifier message.
...
; Now read all data bytes
MOVFF   RXB0DO, MY_DATA_BYTE1
...
; Once entire message is read, mark the RXB0 that it is read and no longer FULL.
BCF     RXB0CON, RXFUL               ; This will allow CAN Module to load new messages
                                     ; into this buffer.
...
```

### 24.2.4 CAN BAUD RATE REGISTERS

This section describes the CAN Baud Rate registers.

| | |
|---|---|
| **Note:** | These registers are writable in Configuration mode only. |

**REGISTER 24-52: BRGCON1: BAUD RATE CONTROL REGISTER 1**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| SJW1 | SJW0 | BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 |
| 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 **SJW<1:0>:** Synchronized Jump Width bits
`11` = Synchronization jump width time = 4 x $T_Q$
`10` = Synchronization jump width time = 3 x $T_Q$
`01` = Synchronization jump width time = 2 x $T_Q$
`00` = Synchronization jump width time = 1 x $T_Q$

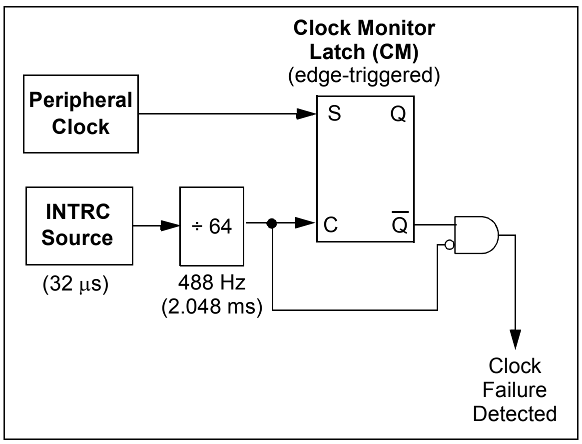
bit 5-0 **BRP<5:0>:** Baud Rate Prescaler bits
`111111` = $T_Q$ = (2 x 64)/FOSC
`111110` = $T_Q$ = (2 x 63)/FOSC
:
:
`000001` = $T_Q$ = (2 x 2)/FOSC
`000000` = $T_Q$ = (2 x 1)/FOSC

## 25.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 25-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor (CM) latch. The CM is set on the falling edge of the device clock source, but cleared on the rising edge of the sample clock.

**FIGURE 25-3: FSCM BLOCK DIAGRAM**




Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 25-4). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>);
- the device clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the Fail-Safe condition); and
- the WDT is reset.

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing-sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 4.1.4 “Multiple Sleep Commands”** and **Section 25.3.1 “Special Considerations for Using Two-Speed Start-up”** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF<2:0> bits prior to entering Sleep mode.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

### 25.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF<2:0> bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

### 25.4.2 EXITING FAIL-SAFE OPERATION

The Fail-Safe condition is terminated by either a device Reset or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as OST or PLL timer). The INTOSC multiplexer provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

| BNC | Branch if Not Carry |
|---|---|
| Syntax: | BNC n |
| Operands: | $-128 \leq n \leq 127$ |
| Operation: | if Carry bit is '0',<br>(PC) + 2 + 2n → PC |
| Status Affected: | None |
| Encoding: | 1110 \| 0011 \| nnnn \| nnnn |
| Description: | If the Carry bit is '0', then the program will branch.<br>The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction. |
| Words: | 1 |
| Cycles: | 1(2) |

Q Cycle Activity:

If Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | Write to PC |
| No operation | No operation | No operation | No operation |

If No Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | No operation |

Example: `HERE BNC Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Carry = 0;
PC = address (Jump)
If Carry = 1;
PC = address (HERE + 2)

| BNN | Branch if Not Negative |
|---|---|
| Syntax: | BNN n |
| Operands: | $-128 \leq n \leq 127$ |
| Operation: | if Negative bit is '0',<br>(PC) + 2 + 2n → PC |
| Status Affected: | None |
| Encoding: | 1110 \| 0111 \| nnnn \| nnnn |
| Description: | If the Negative bit is '0', then the program will branch.<br>The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction. |
| Words: | 1 |
| Cycles: | 1(2) |

Q Cycle Activity:

If Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | Write to PC |
| No operation | No operation | No operation | No operation |

If No Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | No operation |

Example: `HERE BNN Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 0;
PC = address (Jump)
If Negative = 1;
PC = address (HERE + 2)

## SUBWFB — Subtract W from f with Borrow

| | |
|---|---|
| Syntax: | SUBWFB f {,d {,a}} |
| Operands: | $0 \le f \le 255$<br>$d \in [0,1]$<br>$a \in [0,1]$ |
| Operation: | $(f) - (W) - (\overline{C}) \rightarrow$ dest |
| Status Affected: | N, OV, C, DC, Z |
| Encoding: | `0101` `10da` `ffff` `ffff` |
| Description: | Subtract W and the Carry flag (borrow) 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'.<br>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.<br>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details. |
| Words: | 1 |
| Cycles: | 1 |

Q Cycle Activity:

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

Example 1: `SUBWFB REG, 1, 0`

Before Instruction

```
REG  =  19h   (0001 1001)
W    =  0Dh   (0000 1101)
C    =  1
```

After Instruction

```
REG  =  0Ch   (0000 1011)
W    =  0Dh   (0000 1101)
C    =  1
Z    =  0
N    =  0     ; result is positive
```

Example 2: `SUBWFB REG, 0, 0`

Before Instruction

```
REG  =  1Bh   (0001 1011)
W    =  1Ah   (0001 1010)
C    =  0
```

After Instruction

```
REG  =  1Bh   (0001 1011)
W    =  00h
C    =  1
Z    =  1     ; result is zero
N    =  0
```

Example 3: `SUBWFB REG, 1, 0`

Before Instruction

```
REG  =  03h   (0000 0011)
W    =  0Eh   (0000 1101)
C    =  1
```

After Instruction

```
REG  =  F5h   (1111 0100)
              ; [2's comp]
W    =  0Eh   (0000 1101)
C    =  0
Z    =  0
N    =  1     ; result is negative
```

## SWAPF — Swap f

| | |
|---|---|
| Syntax: | SWAPF f {,d {,a}} |
| Operands: | $0 \le f \le 255$<br>$d \in [0,1]$<br>$a \in [0,1]$ |
| Operation: | (f<3:0>) $\rightarrow$ dest<7:4>,<br>(f<7:4>) $\rightarrow$ dest<3:0> |
| Status Affected: | None |
| Encoding: | `0011` `10da` `ffff` `ffff` |
| Description: | The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in register 'f'.<br>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.<br>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details. |
| Words: | 1 |
| Cycles: | 1 |

Q Cycle Activity:

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

Example: `SWAPF REG, 1, 0`

Before Instruction

```
REG  =  53h
```

After Instruction

```
REG  =  35h
```

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F2480/2580/4480/4580 (Industrial, Extended) PIC18LF2480/2580/4480/4580 (Industrial) (Continued)

<table>
<tr><td colspan="2">PIC18LF2480/2580/4480/4580<br>(Industrial)</td><td colspan="6">Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial</td></tr>
<tr><td colspan="2">PIC18F2480/2580/4480/4580<br>(Industrial, Extended)</td><td colspan="6">Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended</td></tr>
<tr><th>Param No.</th><th>Device</th><th>Typ</th><th>Max</th><th>Units</th><th colspan="3">Conditions</th></tr>
<tr><td rowspan="4">D026<br>(ΔIAD)</td><td rowspan="4">A/D Converter</td><td>1.0</td><td>2.0</td><td>μA</td><td>-40°C to +85°C</td><td>VDD = 2.0V</td><td rowspan="4">A/D on, not converting</td></tr>
<tr><td>1.0</td><td>2.0</td><td>μA</td><td>-40°C to +85°C</td><td>VDD = 3.0V</td></tr>
<tr><td>1.0</td><td>2.0</td><td>μA</td><td>-40°C to +85°C</td><td rowspan="2">VDD = 5.0V</td></tr>
<tr><td>2.0</td><td>8.0</td><td>μA</td><td>-40°C to +125°C</td></tr>
</table>

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

**3:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula, Ir = VDD/2REXT (mA), with REXT in kΩ.

**4:** Standard low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.