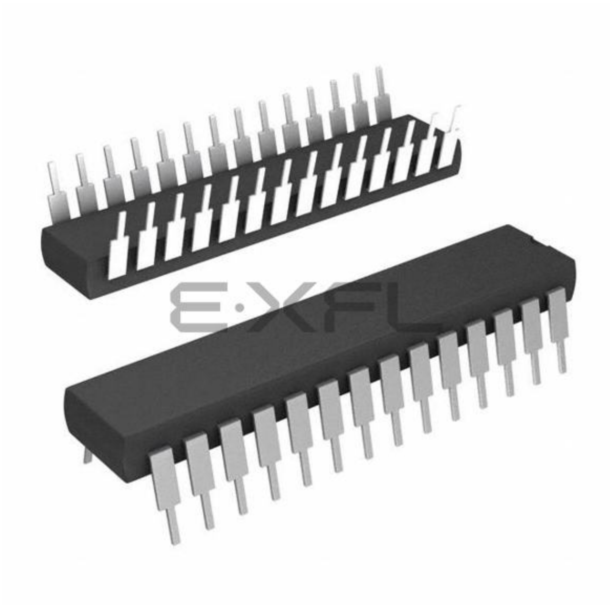





| Details | |
|---|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 48MHz |
| Connectivity | I²C, SPI, UART/USART, USB |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 24 |
| Program Memory Size | 24KB (12K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 4.2V ~ 5.5V |
| Data Converters | A/D 10x10b |
| Oscillator Type | Internal |
| 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/pic18f2455-i-sp |

### 3.4.3 RC_IDLE MODE

In RC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the internal oscillator block using the INTOSC multiplexer. This mode allows for controllable power conservation during Idle periods.

From RC_RUN, this mode is entered by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, first set IDLEN, then set the SCS1 bit and execute `SLEEP`. Although its value is ignored, it is recommended that SCS0 also be cleared; this is to maintain software compatibility with future devices. The INTOSC multiplexer may be used to select a higher clock frequency by modifying the IRCF bits before executing the `SLEEP` instruction. When the clock source is switched to the INTOSC multiplexer, the primary oscillator is shut down and the OSTS bit is cleared.

If the IRCF bits are set to any non-zero value, or the INTSRC bit is set, the INTOSC output is enabled. The IOFS bit becomes set after the INTOSC output becomes stable, after an interval of $T_{IOBST}$ (parameter 39, Table 28-12). Clocks to the peripherals continue while the INTOSC source stabilizes. If the IRCF bits were previously at a non-zero value, or INTSRC was set before the `SLEEP` instruction was executed and the INTOSC source was already stable, the IOFS bit will remain set. If the IRCF bits and INTSRC are all clear, the INTOSC output will not be enabled, the IOFS bit will remain clear and there will be no indication of the current clock source.

When a wake event occurs, the peripherals continue to be clocked from the INTOSC multiplexer. After a delay of $T_{CSD}$ following the wake event, the CPU begins executing code being clocked by the INTOSC multiplexer. The IDLEN and SCS bits are not affected by the wake-up. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

## 3.5 Exiting Idle and Sleep Modes

An exit from Sleep mode or any of the Idle modes is triggered by an interrupt, a Reset or a WDT time-out. This section discusses the triggers that cause exits from power-managed modes. The clocking subsystem actions are discussed in each of the power-managed modes (see **Section 3.2 “Run Modes”**, **Section 3.3 “Sleep Mode”** and **Section 3.4 “Idle Modes”**).

### 3.5.1 EXIT BY INTERRUPT

Any of the available interrupt sources can cause the device to exit from an Idle mode or Sleep mode to a Run mode. To enable this functionality, an interrupt source must be enabled by setting its enable bit in one of the INTCON or PIE registers. The exit sequence is initiated when the corresponding interrupt flag bit is set.

On all exits from Idle or Sleep modes by interrupt, code execution branches to the interrupt vector if the GIE/GIEH bit (INTCON<7>) is set. Otherwise, code execution continues or resumes without branching (see **Section 9.0 “Interrupts”**).

A fixed delay of interval $T_{CSD}$ following the wake event is required when leaving Sleep and Idle modes. This delay is required for the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

### 3.5.2 EXIT BY WDT TIME-OUT

A WDT time-out will cause different actions depending on which power-managed mode the device is in when the time-out occurs.

If the device is not executing code (all Idle modes and Sleep mode), the time-out will result in an exit from the power-managed mode (see **Section 3.2 “Run Modes”** and **Section 3.3 “Sleep Mode”**). If the device is executing code (all Run modes), the time-out will result in a WDT Reset (see **Section 25.2 “Watchdog Timer (WDT)”**).

The WDT timer and postscaler are cleared by executing a `SLEEP` or `CLRWDT` instruction, the loss of a currently selected clock source (if the Fail-Safe Clock Monitor is enabled) and modifying the IRCF bits in the OSCCON register if the internal oscillator block is the device clock source.

### 3.5.3 EXIT BY RESET

Normally, the device is held in Reset by the Oscillator Start-up Timer (OST) until the primary clock becomes ready. At that time, the OSTS bit is set and the device begins executing code. If the internal oscillator block is the new clock source, the IOFS bit is set instead.

The exit delay time from Reset to the start of code execution depends on both the clock sources before and after the wake-up and the type of oscillator if the new clock source is the primary clock. Exit delays are summarized in Table 3-2.

Code execution can begin before the primary clock becomes ready. If either the Two-Speed Start-up (see **Section 25.3 “Two-Speed Start-up”**) or Fail-Safe Clock Monitor (see **Section 25.4 “Fail-Safe Clock Monitor”**) is enabled, the device may begin execution as soon as the Reset source has cleared. Execution is clocked by the INTOSC multiplexer driven by the internal oscillator block. Execution is clocked by the internal oscillator block until either the primary clock becomes ready or a power-managed mode is entered before the primary clock becomes ready; the primary clock is then shut down.

## 10.5 PORTE, TRISE and LATE Registers

Depending on the particular PIC18F2455/2550/4455/4550 device selected, PORTE is implemented in two different ways.

For 40/44-pin devices, PORTE is a 4-bit wide port. Three pins (RE0/AN5/CK1SPP, RE1/AN6/CK2SPP and RE2/AN7/OESPP) are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. When selected as an analog input, these pins will read as '0's.

The corresponding Data Direction register is TRISE. Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., put the contents of the output latch on the selected pin).

In addition to port data, the PORTE register (Register 10-1) also contains the RDPU control bit (PORTE<7>); this enables or disables the weak pull-ups on PORTD.

TRISE controls the direction of the RE pins, even when they are being used as analog inputs. The user must make sure to keep the pins configured as inputs when using them as analog inputs.

| **Note:** | On a Power-on Reset, RE2:RE0 are configured as analog inputs. |
|---|---|

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

The fourth pin of PORTE ($\overline{MCLR}$/VPP/RE3) is an input only pin. Its operation is controlled by the MCLRE Configuration bit. When selected as a port pin (MCLRE = 0), it functions as a digital input only pin; as such, it does not have TRIS or LAT bits associated with its operation. Otherwise, it functions as the device's Master Clear input. In either configuration, RE3 also functions as the programming voltage input during programming.

| **Note:** | On a Power-on Reset, RE3 is enabled as a digital input only if Master Clear functionality is disabled. |
|---|---|

**EXAMPLE 10-5: INITIALIZING PORTE**

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0Ah     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   03h     ; Value used to
                ; initialize data
                ; direction
MOVLW   07h     ; Turn off
MOVWF   CMCON   ; comparators
MOVWF   TRISC   ; Set RE<0> as inputs
                ; RE<1> as outputs
                ; RE<2> as inputs
```

### 10.5.1 PORTE IN 28-PIN DEVICES

For 28-pin devices, PORTE is only available when Master Clear functionality is disabled (MCLRE = 0). In these cases, PORTE is a single bit, input only port comprised of RE3 only. The pin operates as previously described.

**REGISTER 10-1: PORTE REGISTER**

| R/W-0 | U-0 | U-0 | U-0 | R/W-x | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| RDPU[3] | — | — | — | RE3[1,2] | RE2[3] | RE1[3] | RE0[3] |
| 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 | **RDPU:** PORTD Pull-up Enable bit<br>1 = PORTD pull-ups are enabled by individual port latch values<br>0 = All PORTD pull-ups are disabled |
| bit 6-4 | **Unimplemented:** Read as '0' |
| bit 3-0 | **RE3:RE0:** PORTE Data Input bits[1,2,3] |

**Note 1:** implemented only when Master Clear functionality is disabled (MCLRE Configuration bit = 0); otherwise, read as '0'.

**2:** RE3 is the only PORTE bit implemented on both 28-pin and 40/44-pin devices. All other bits are implemented only when PORTE is implemented (i.e., 40/44-pin devices).

**3:** Unimplemented in 28-pin devices; read as '0'.

## 12.1 Timer1 Operation

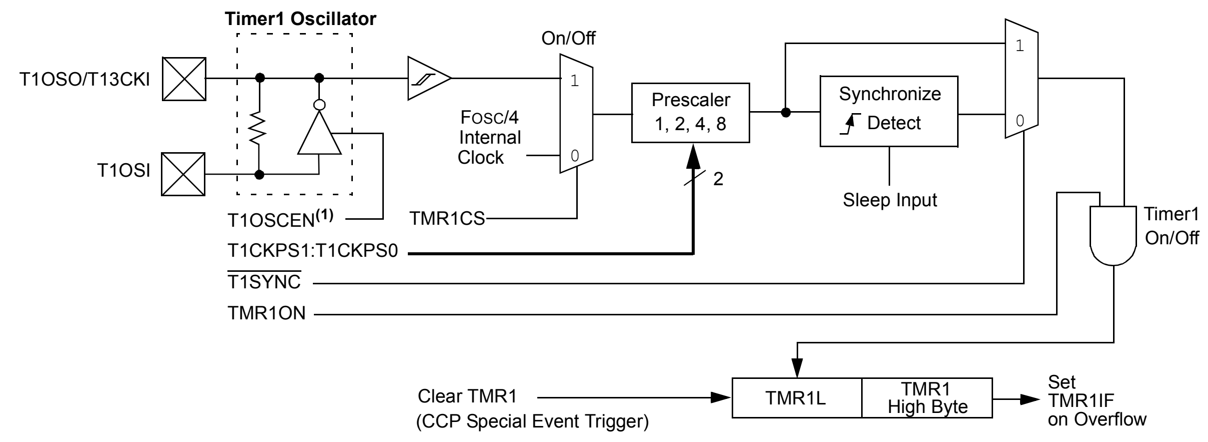
Timer1 can operate in one of these modes:

- Timer
- Synchronous Counter
- Asynchronous Counter

The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>). When TMR1CS is cleared (= 0), Timer1 increments on every internal instruction cycle (FOSC/4). When the bit is set, Timer1 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

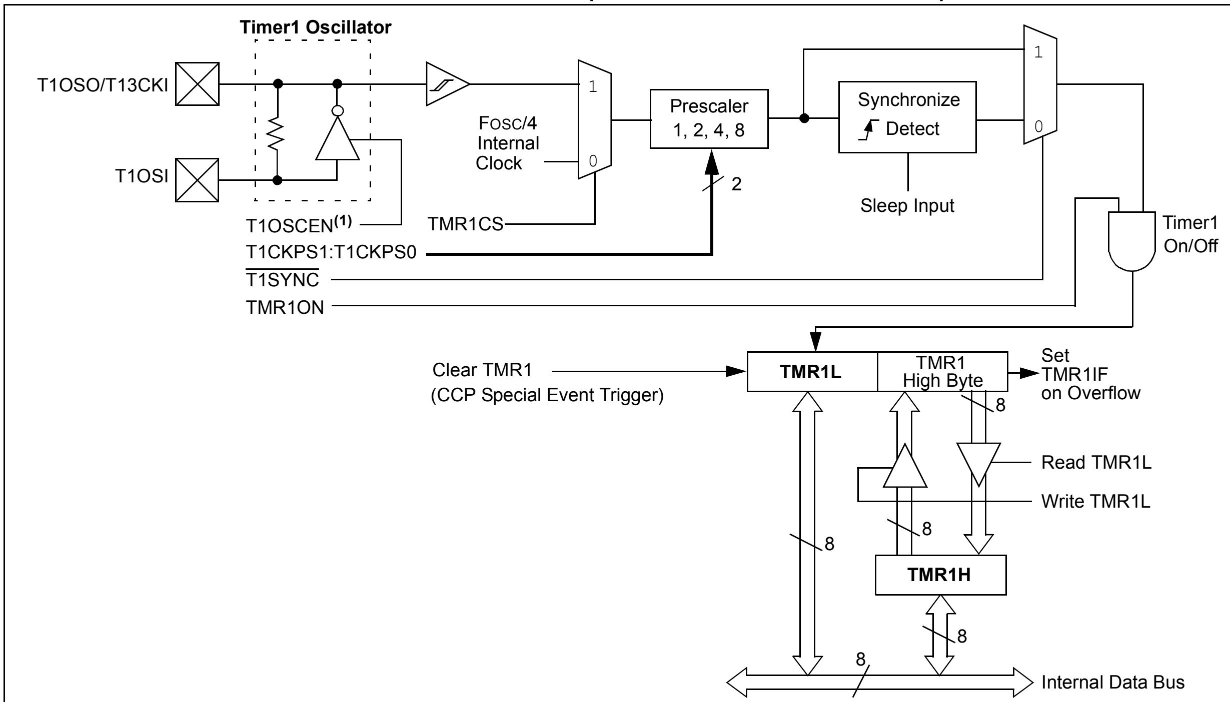
When Timer1 is enabled, the RC1/T1OSI/$\overline{\text{UOE}}$ and RC0/T1OSO/T13CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

**FIGURE 12-1: TIMER1 BLOCK DIAGRAM**




**Note 1:** When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

**FIGURE 12-2: TIMER1 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)**




**Note 1:** When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

The CCPRxH register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPRxH and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCPx pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

**EQUATION 15-3:**

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

**Note:** If the PWM duty cycle value is longer than the PWM period, the CCPx pin will not be cleared.

**TABLE 15-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz**

| PWM Frequency | 2.44 kHz | 9.77 kHz | 39.06 kHz | 156.25 kHz | 312.50 kHz | 416.67 kHz |
|---|---|---|---|---|---|---|
| Timer Prescaler (1, 4, 16) | 16 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | FFh | FFh | FFh | 3Fh | 1Fh | 17h |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.58 |

### 15.4.3 PWM AUTO-SHUTDOWN (CCP1 ONLY)

The PWM auto-shutdown features of the Enhanced CCP module are also available to CCP1 in 28-pin devices. The operation of this feature is discussed in detail in **Section 16.4.7 "Enhanced PWM Auto-Shutdown"**.

Auto-shutdown features are not available for CCP2.

### 15.4.4 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCPx module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPRxL register and CCPxCON<5:4> bits.
3. Make the CCPx pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCPx module for PWM operation.

**TABLE 17-1: DIFFERENTIAL OUTPUTS TO TRANSCEIVER**

| VPO | VMO | Bus State |
|---|---|---|
| 0 | 0 | Single-Ended Zero |
| 0 | 1 | Differential '0' |
| 1 | 0 | Differential '1' |
| 1 | 1 | Illegal Condition |

**TABLE 17-2: SINGLE-ENDED INPUTS FROM TRANSCEIVER**

| VP | VM | Bus State |
|---|---|---|
| 0 | 0 | Single-Ended Zero |
| 0 | 1 | Low Speed |
| 1 | 0 | High Speed |
| 1 | 1 | Error |

The $\overline{\text{UOE}}$ signal toggles the state of the external transceiver. This line is pulled low by the device to enable the transmission of data from the SIE to an external device.

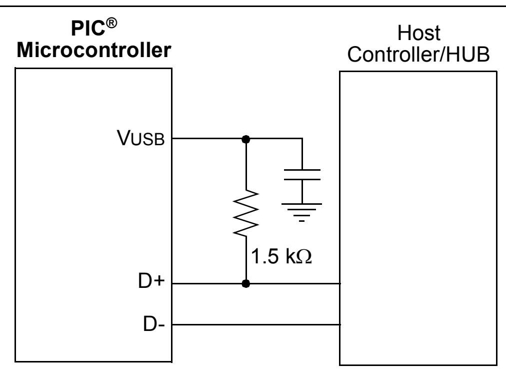
### 17.2.2.3 Internal Pull-up Resistors

The PIC18FX455/X550 devices have built-in pull-up resistors designed to meet the requirements for low-speed and full-speed USB. The UPUEN bit (UCFG<4>) enables the internal pull-ups. Figure 17-1 shows the pull-ups and their control.

### 17.2.2.4 External Pull-up Resistors

External pull-up may also be used if the internal resistors are not used. The VUSB pin may be used to pull up D+ or D-. The pull-up resistor must be 1.5 kΩ (±5%) as required by the USB specifications. Figure 17-3 shows an example.

**FIGURE 17-3: EXTERNAL CIRCUITRY**




**Note:** The above setting shows a typical connection for a full-speed configuration using an on-chip regulator and an external pull-up resistor.

### 17.2.2.5 Ping-Pong Buffer Configuration

The usage of ping-pong buffers is configured using the PPB1:PPB0 bits. Refer to **Section 17.4.4 "Ping-Pong Buffering"** for a complete explanation of the ping-pong buffers.

### 17.2.2.6 USB Output Enable Monitor

The USB $\overline{\text{OE}}$ monitor provides indication as to whether the SIE is listening to the bus or actively driving the bus. This is enabled by default when using an external transceiver or when UCFG<6> = 1.

The USB $\overline{\text{OE}}$ monitoring is useful for initial system debugging, as well as scope triggering during eye pattern generation tests.

### 17.2.2.7 Eye Pattern Test Enable

An automatic eye pattern test can be generated by the module when the UCFG<7> bit is set. The eye pattern output will be observable based on module settings, meaning that the user is first responsible for configuring the SIE clock settings, pull-up resistor and Transceiver mode. In addition, the module has to be enabled.

Once UTEYE is set, the module emulates a switch from a receive to transmit state and will start transmitting a J-K-J-K bit sequence (K-J-K-J for full speed). The sequence will be repeated indefinitely while the Eye Pattern Test mode is enabled.

Note that this bit should never be set while the module is connected to an actual USB system. This test mode is intended for board verification to aid with USB certification tests. It is intended to show a system developer the noise integrity of the USB signals which can be affected by board traces, impedance mismatches and proximity to other system components. It does not properly test the transition from a receive to a transmit state. Although the eye pattern is not meant to replace the more complex USB certification test, it should aid during first order system debugging.

### 17.5.1 USB INTERRUPT STATUS REGISTER (UIR)

The USB Interrupt Status register (Register 17-7) contains the flag bits for each of the USB status interrupt sources. Each of these sources has a corresponding interrupt enable bit in the UIE register. All of the USB status flags are ORed together to generate the USBIF interrupt flag for the microcontroller's interrupt funnel.

Once an interrupt bit has been set by the SIE, it must be cleared by software by writing a '0'. The flag bits can also be set in software which can aid in firmware debugging.

When the USB module is in the Low-Power Suspend mode (UCON<1> = 1), the SIE does not get clocked. When in this state, the SIE cannot process packets, and therefore, cannot detect new interrupt conditions other than the Activity Detect Interrupt, ACTVIF. The ACTVIF bit is typically used by USB firmware to detect when the microcontroller should bring the USB module out of the Low-Power Suspend mode (UCON<1> = 0).

**REGISTER 17-7: UIR: USB INTERRUPT STATUS REGISTER**

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | SOFIF | STALLIF | IDLEIF(1) | TRNIF(2) | ACTVIF(3) | UERRIF(4) | URSTIF |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **Unimplemented:** Read as '0'

bit 6 **SOFIF:** Start-Of-Frame Token Interrupt bit
1 = A Start-Of-Frame token received by the SIE
0 = No Start-Of-Frame token received by the SIE

bit 5 **STALLIF:** A STALL Handshake Interrupt bit
1 = A STALL handshake was sent by the SIE
0 = A STALL handshake has not been sent

bit 4 **IDLEIF:** Idle Detect Interrupt bit(1)
1 = Idle condition detected (constant Idle state of 3 ms or more)
0 = No Idle condition detected

bit 3 **TRNIF:** Transaction Complete Interrupt bit(2)
1 = Processing of pending transaction is complete; read USTAT register for endpoint information
0 = Processing of pending transaction is not complete or no transaction is pending

bit 2 **ACTVIF:** Bus Activity Detect Interrupt bit(3)
1 = Activity on the D+/D- lines was detected
0 = No activity detected on the D+/D- lines

bit 1 **UERRIF:** USB Error Condition Interrupt bit(4)
1 = An unmasked error condition has occurred
0 = No unmasked error condition has occurred.

bit 0 **URSTIF:** USB Reset Interrupt bit
1 = Valid USB Reset occurred; 00h is loaded into UADDR register
0 = No USB Reset has occurred

**Note 1:** Once an Idle state is detected, the user may want to place the USB module in Suspend mode.
**2:** Clearing this bit will cause the USTAT FIFO to advance (valid only for IN, OUT and SETUP tokens).
**3:** This bit is typically unmasked only following the detection of a UIDLE interrupt event.
**4:** Only error conditions enabled through the UEIE register will set this bit. This bit is a status bit only and cannot be set or cleared by the user.

## 17.6 USB Power Modes

Many USB applications will likely have several different sets of power requirements and configuration. The most common power modes encountered are Bus Power Only, Self-Power Only and Dual Power with Self-Power Dominance. The most common cases are presented here.

### 17.6.1 BUS POWER ONLY

In Bus Power Only mode, all power for the application is drawn from the USB (Figure 17-10). This is effectively the simplest power method for the device.

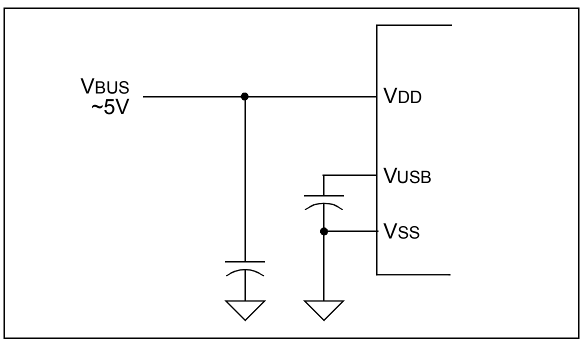
In order to meet the inrush current requirements of the USB 2.0 specifications, the total effective capacitance appearing across VBUS and ground must be no more than 10 μF. If not, some kind of inrush limiting is required. For more details, see Section 7.2.4 of the USB 2.0 specification.

According to the USB 2.0 specification, all USB devices must also support a Low-Power Suspend mode. In the USB Suspend mode, devices must consume no more than 2.5 mA from the 5V VBUS line of the USB cable.

The host signals the USB device to enter the Suspend mode by stopping all USB traffic to that device for more than 3 ms. This condition will cause the IDLEIF bit in the UIR register to become set.

During the USB Suspend mode, the D+ or D- pull-up resistor must remain active, which will consume some of the allowed suspend current: 2.5 mA budget.

**FIGURE 17-10: BUS POWER ONLY**



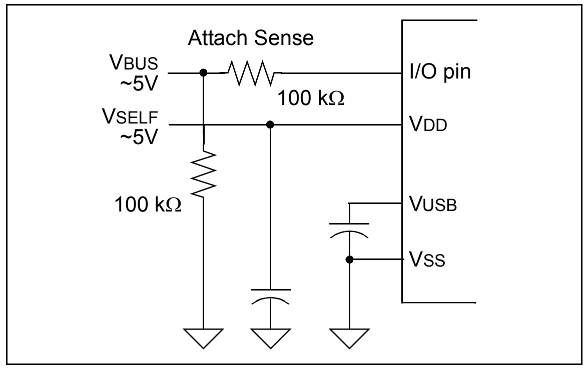

### 17.6.2 SELF-POWER ONLY

In Self-Power Only mode, the USB application provides its own power, with very little power being pulled from the USB. Figure 17-11 shows an example. Note that an attach indication is added to indicate when the USB has been connected and the host is actively powering VBUS.

In order to meet compliance specifications, the USB module (and the D+ or D- pull-up resistor) should not be enabled until the host actively drives VBUS high. One of the I/O pins may be used for this purpose.

The application should never source any current onto the 5V VBUS pin of the USB cable.

**FIGURE 17-11: SELF-POWER ONLY**



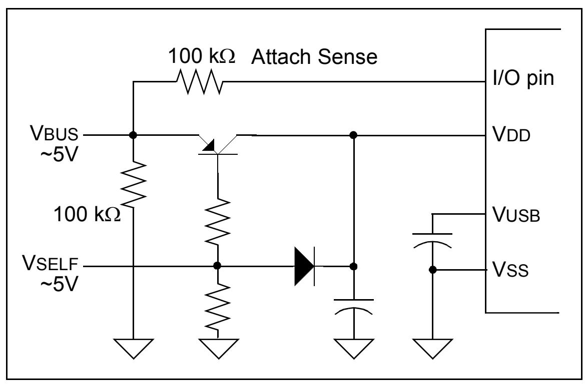

### 17.6.3 DUAL POWER WITH SELF-POWER DOMINANCE

Some applications may require a dual power option. This allows the application to use internal power primarily, but switch to power from the USB when no internal power is available. Figure 17-12 shows a simple Dual Power with Self-Power Dominance example, which automatically switches between Self-Power Only and USB Bus Power Only modes.

Dual power devices also must meet all of the special requirements for inrush current and Suspend mode current and must not enable the USB module until VBUS is driven high. For descriptions of those requirements, see **Section 17.6.1 "Bus Power Only"** and **Section 17.6.2 "Self-Power Only"**.

Additionally, dual power devices must never source current onto the 5V VBUS pin of the USB cable.

**FIGURE 17-12: DUAL POWER EXAMPLE**



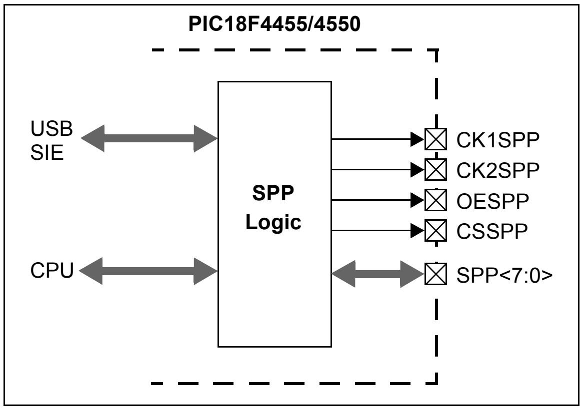

| | |
|---|---|
| **Note:** | Users should keep in mind the limits for devices drawing power from the USB. According to USB specification 2.0, this cannot exceed 100 mA per low-power device or 500 mA per high-power device. |

## 18.0 STREAMING PARALLEL PORT

| Note: | The Streaming Parallel Port is only available on 40/44-pin devices. |
|---|---|

PIC18F4455/4550 USB devices provide a Streaming Parallel Port as a high-speed interface for moving data to and from an external system. This parallel port operates as a master port, complete with chip select and clock outputs to control the movement of data to slave devices. Data can be channelled either directly to the USB SIE or to the microprocessor core. Figure 18-1 shows a block view of the SPP data path.

**FIGURE 18-1: SPP DATA PATH**




In addition, the SPP can provide time multiplexed addressing information along with the data by using the second strobe output. Thus, the USB endpoint number can be written in conjunction with the data for that endpoint.

## 18.1 SPP Configuration

The operation of the SPP is controlled by two registers: SPPCON and SPPCFG. The SPPCON register (Register 18-1) controls the overall operation of the parallel port and determines if it operates under USB or microcontroller control. The SPPCFG register (Register 18-2) controls timing configuration and pin outputs.

### 18.1.1 ENABLING THE SPP

To enable the SPP, set the SPPEN bit (SPPCON<0>). In addition, the TRIS bits for the corresponding SPP pins must be properly configured. At a minimum:

- Bits TRISD<7:0> must be set (= 1)
- Bits TRISE<2:1> must be cleared (= 0)

If CK1SPP is to be used:

- Bit TRISE<0> must be cleared (= 0)

If CSPP is to be used:

- Bit TRISB<4> must be cleared (= 0)

**REGISTER 18-1: SPPCON: SPP CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | SPPOWN | SPPEN |
| 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 7-2 **Unimplemented:** Read as ‘0’

bit 1 **SPPOWN:** SPP Ownership bit
1 = USB peripheral controls the SPP
0 = Microcontroller directly controls the SPP

bit 0 **SPPEN:** SPP Enable bit
1 = SPP is enabled
0 = SPP is disabled

**REGISTER 18-2: SPPCFG: SPP CONFIGURATION REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CLKCFG1 | CLKCFG0 | CSEN | CLK1EN | WS3 | WS2 | WS1 | WS0 |
| 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 **CLKCFG1:CLKCFG0:** SPP Clock Configuration bits
- `1x` = CLK1 toggles on read or write of an Odd endpoint address; CLK2 toggles on read or write of an Even endpoint address
- `01` = CLK1 toggles on write; CLK2 toggles on read
- `00` = CLK1 toggles only on endpoint address write; CLK2 toggles on data read or write

bit 5 **CSEN:** SPP Chip Select Pin Enable bit
- `1` = RB4 pin is controlled by the SPP module and functions as SPP CS output
- `0` = RB4 functions as a digital I/O port

bit 4 **CLK1EN:** SPP CLK1 Pin Enable bit
- `1` = RE0 pin is controlled by the SPP module and functions as SPP CLK1 output
- `0` = RE0 functions as a digital I/O port

bit 3-0 **WS3:WS0:** SPP Wait States bits
- `1111` = 30 additional wait states
- `1110` = 28 additional wait states
- • •
- • •
- `0001` = 2 additional wait states
- `0000` = 0 additional wait states

### 18.1.2 CLOCKING DATA

The SPP has four control outputs:

- Two separate clock outputs (CK1SPP and CK2SPP)
- Output enable (OESPP)
- Chip select (CSSPP)

Together, they allow for several different configurations for controlling the flow of data to slave devices. When all control outputs are used, the three main options are:

- CLK1 clocks endpoint address information while CLK2 clocks data
- CLK1 clocks write operations while CLK2 clocks reads
- CLK1 clocks Odd address data while CLK2 clocks Even address data

Additional control options are derived by disabling the CK1SPP and CSSPP outputs. These are enabled or disabled with the CLK1EN and CSEN bits, respectively, located in Register 18-2.

### 18.1.3 WAIT STATES

The SPP is designed with the capability of adding wait states to read and write operations. This allows access to parallel devices that require extra time for access.

Wait state clocking is based on the data source clock. If the SPP is configured to operate as a USB endpoint, then wait states are based on the USB clock. Likewise, if the SPP is configured to operate from the microcontroller, then wait states are based on the instruction rate (FOSC/4).

The WS3:WS0 bits set the wait states used by the SPP, with a range of no wait states to 30 wait states, in multiples of two. The wait states are added symmetrically to all transactions, with one-half added following each of the two clock cycles normally required for the transaction. Figure 18-3 and Figure 18-4 show signalling examples with 4 wait states added to each transaction.

### 18.1.4 SPP PULL-UPS

The SPP data lines (SPP<7:0>) are equipped with internal pull-ups for applications that may leave the port in a high-impedance condition. The pull-ups are enabled using the control bit, RDPU (PORTE<7>).

### 19.4.14 SLEEP OPERATION

While in Sleep mode, the $I^2C$ module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

### 19.4.15 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

### 19.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the $I^2C$ bus may be taken when the P bit (SSPSTAT<4>) is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the MSSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed in hardware with the result placed in the BCLIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

### 19.4.17 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA, by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF, and reset the $I^2C$ port to its Idle state (Figure 19-27).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine, and if the $I^2C$ bus is free, the user can resume communication by asserting a Start condition.

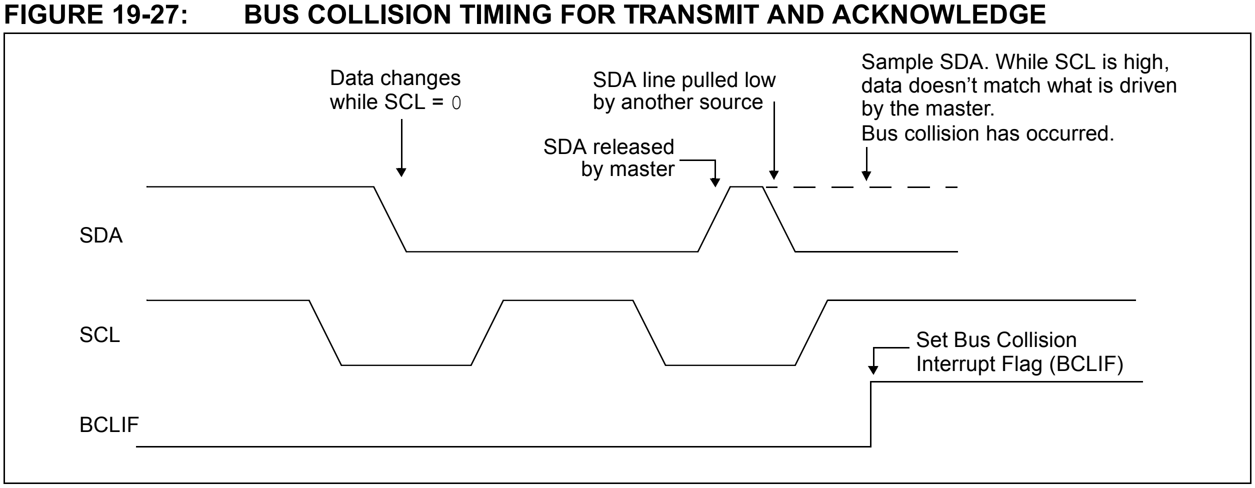
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine, and if the $I^2C$ bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL pins. If a Stop condition occurs, the SSPIF bit will be set.

A write to the SSPBUF bit will start the transmission of data at the first data bit regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the $I^2C$ bus can be taken when the P bit is set in the SSPSTAT register, or the bus is Idle and the S and P bits are cleared.

**FIGURE 19-27: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**



**EXAMPLE 20-1: CALCULATING BAUD RATE ERROR**

```
For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:
Desired Baud Rate      =  FOSC/(64 ([SPBRGH:SPBRG] + 1))
Solving for SPBRGH:SPBRG:
                  X    =  ((FOSC/Desired Baud Rate)/64) – 1
                       =  ((16000000/9600)/64) – 1
                       =  [25.042] = 25
Calculated Baud Rate   =  16000000/(64 (25 + 1))
                       =  9615
Error                  =  (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
                       =  (9615 – 9600)/9600 = 0.16%
```

**TABLE 20-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---|---|---|---|---|---|---|---|---|---|
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 55 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 55 |
| BAUDCON | ABDOVF | RCIDL | RXDTP | TXCKP | BRG16 | — | WUE | ABDEN | 55 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 55 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 55 |

**Legend:** — = unimplemented, read as ‘0’. Shaded cells are not used by the BRG.

**FIGURE 20-6: EUSART RECEIVE BLOCK DIAGRAM**

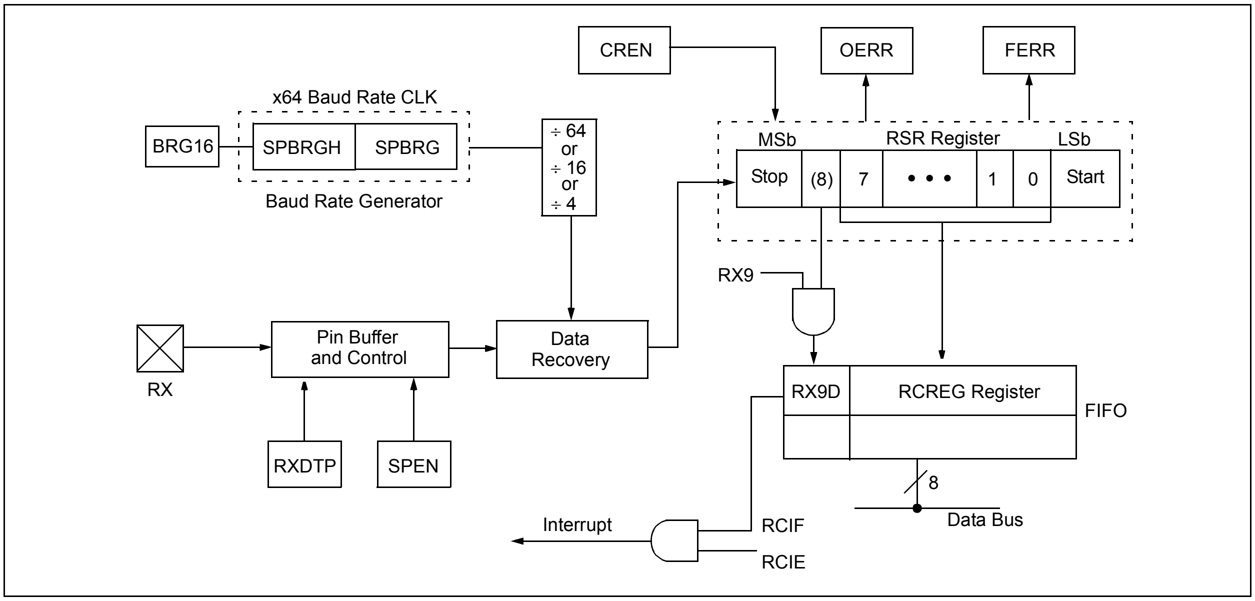
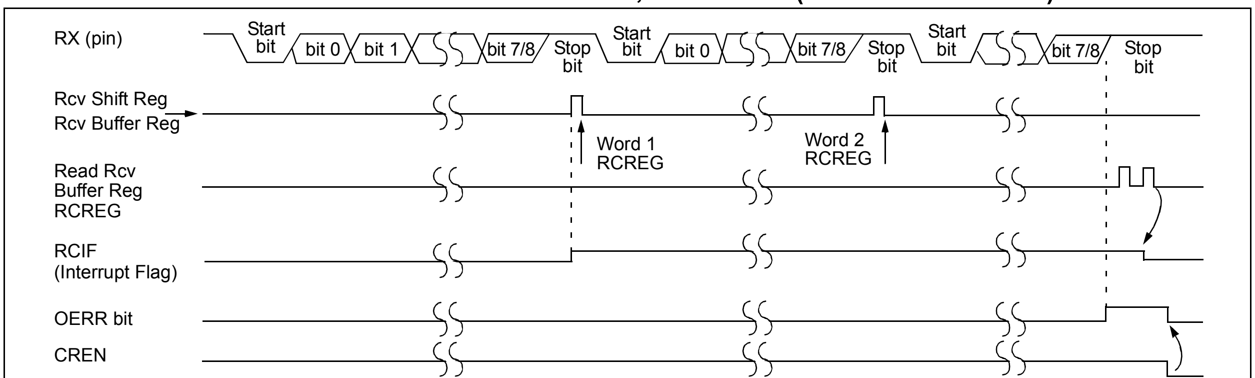



**FIGURE 20-7: ASYNCHRONOUS RECEPTION, RXDTP = 0 (RX NOT INVERTED)**




**Note:** This timing diagram shows three words appearing on the RX input. The RCREG (Receive Buffer) is read after the third word causing the OERR (Overrun) bit to be set.

**TABLE 20-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS 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 | 53 |
| PIR1 | SPPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 56 |
| PIE1 | SPPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 56 |
| IPR1 | SPPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 56 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 55 |
| RCREG | EUSART Receive Register | | | | | | | | 55 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 55 |
| BAUDCON | ABDOVF | RCIDL | RXDTP | TXCKP | BRG16 | — | WUE | ABDEN | 55 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 55 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 55 |

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

**Note 1:** Reserved in 28-pin devices; always maintain these bits clear.

## 23.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable reference voltage. Although its primary purpose is to provide a reference for the analog comparators, it may also be used independently of them.

A block diagram of the module is shown in Figure 23-1. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device VDD/VSS or an external voltage reference.

### 23.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 23-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR3:CVR0), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = ((CVR3:CVR0)/24) x CVRSRC

If CVRR = 0:
CVREF = (CVRSRC/4) + (((CVR3:CVR0)/32) x CVRSRC)

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF- that are multiplexed with RA2 and RA3. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 28-3 in **Section 28.0 "Electrical Characteristics"**).

**REGISTER 23-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CVREN | CVROE(1) | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 |
| 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 **CVREN**: Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RA2/AN2/VREF-/CVREF pin
0 = CVREF voltage is disconnected from the RA2/AN2/VREF-/CVREF pin

bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)

bit 4 **CVRSS**: Comparator VREF Source Selection bit
1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)
0 = Comparator reference source, CVRSRC = VDD – VSS

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ (CVR3:CVR0) ≤ 15)
When CVRR = 1:
CVREF = ((CVR3:CVR0)/24) • (CVRSRC)
When CVRR = 0:
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

**Note 1:** CVROE overrides the TRISA<2> bit setting.

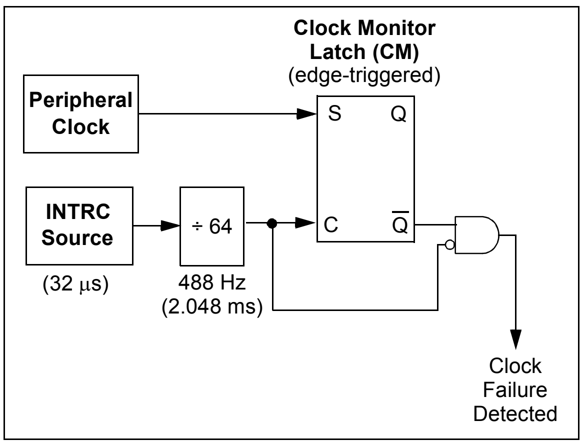
**NOTES:**

## 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 latch (CM). 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 shut-down. See **Section 3.1.4 "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 IRCF2:IRCF0 immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting IRCF2:IRCF0 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 IRCF2:IRCF0 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 Monitor 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 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.

**CLRF** **Clear f**

| | |
|---|---|
| Syntax: | CLRF f {,a} |
| Operands: | $0 \leq f \leq 255$<br>$a \in [0,1]$ |
| Operation: | $000h \rightarrow f$,<br>$1 \rightarrow Z$ |
| Status Affected: | Z |
| Encoding: | `0110` `101a` `ffff` `ffff` |
| Description: | Clears the contents of the specified register.<br>If 'a' is '0', the Access Bank is selected.<br>If 'a' is '1', the BSR is used to select the GPR bank (default).<br>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (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 register 'f' |

Example: `CLRF FLAG_REG,1`

Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

**CLRWDT** **Clear Watchdog Timer**

| | |
|---|---|
| Syntax: | CLRWDT |
| Operands: | None |
| Operation: | $000h \rightarrow$ WDT,<br>$000h \rightarrow$ WDT postscaler,<br>$1 \rightarrow \overline{TO}$,<br>$1 \rightarrow \overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Encoding: | `0000` `0000` `0000` `0100` |
| Description: | `CLRWDT` instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits, $\overline{TO}$ and $\overline{PD}$, 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 = 00h
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

### GOTO — Unconditional Branch

Syntax: GOTO k

Operands: $0 \le k \le 1048575$

Operation: k → PC<20:1>

Status Affected: None

Encoding:

| | | | | |
|---|---|---|---|---|
| 1st word (k<7:0>) | 1110 | 1111 | $k_7$kkk | kkkk$_0$ |
| 2nd word(k<19:8>) | 1111 | $k_{19}$kkk | kkkk | kkkk$_8$ |

Description: `GOTO` allows an unconditional branch anywhere within the entire 2-Mbyte memory range. The 20-bit value ‘k’ is loaded into PC<20:1>. `GOTO` is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal ‘k’<7:0>, | No operation | Read literal ‘k’<19:8>, Write to PC |
| No operation | No operation | No operation | No operation |

Example: `GOTO THERE`

After Instruction
PC = Address `(THERE)`

### INCF — Increment f

Syntax: INCF f {,d {,a}}

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

Operation: (f) + 1 → dest

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

Encoding:

| | | | |
|---|---|---|---|
| 0010 | 10da | ffff | ffff |

Description: The contents of register ‘f’ are incremented. If ‘d’ is ‘`0`’, the result is placed in W. If ‘d’ is ‘`1`’, the result is placed back in register ‘f’ (default).
If ‘a’ is ‘`0`’, the Access Bank is selected. If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank (default).
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: `INCF CNT, 1, 0`

Before Instruction
CNT = FFh
Z = 0
C = ?
DC = ?

After Instruction
CNT = 00h
Z = 1
C = 1
DC = 1

| MOVFF | Move f to f | | | |
|---|---|---|---|---|
| Syntax: | 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:<br>1st word (source)<br>2nd word (destin.) | 1100<br>1111 | ffff<br>ffff | ffff<br>ffff | $ffff_s$<br>$ffff_d$ |

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.

Words: 2

Cycles: 2

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 = 33h
REG2 = 11h
After Instruction
REG1 = 33h
REG2 = 33h

| MOVLB | Move Literal to Low Nibble in BSR | | | |
|---|---|---|---|---|
| Syntax: | MOVLW k | | | |
| Operands: | $0 \le k \le 255$ | | | |
| Operation: | $k \rightarrow$ BSR | | | |
| Status Affected: | None | | | |
| Encoding: | 0000 | 0001 | kkkk | kkkk |

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '`0`' regardless of the value of $k_7$:$k_4$.

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 = 02h
After Instruction
BSR Register = 05h

**NOTES:**

**NOTES:**