# PIC16F87/88
# Data Sheet

## 18/20-Pin Enhanced FLASH Microcontrollers with nanoWatt Technology

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products.



DNV Certification, Inc. USA — ANSI-RAB QMS ACCREDITED — DNV MSC The Netherlands Accredited by the RvA — DNV — ISO 9001 / QS-9000 REGISTERED FIRM

*Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.*

# PIC16F87/88

## 18/20-Pin Enhanced FLASH MCUs with nanoWatt Technology

### Low Power Features:

- Power Managed modes:
  - Primary RUN XT, RC oscillator, 87 μA, 1 MHz, 2V
  - RC_RUN 7 μA, 31.25 kHz, 2V
  - SEC_RUN 14 μA, 32 kHz, 2V
  - SLEEP 0.2 μA, 2V
- Timer1 oscillator 1.3 μA, 32 kHz, 2V
- Watchdog Timer 0.7 μA, 2V
- Two-Speed Oscillator Start-up

### Oscillators:

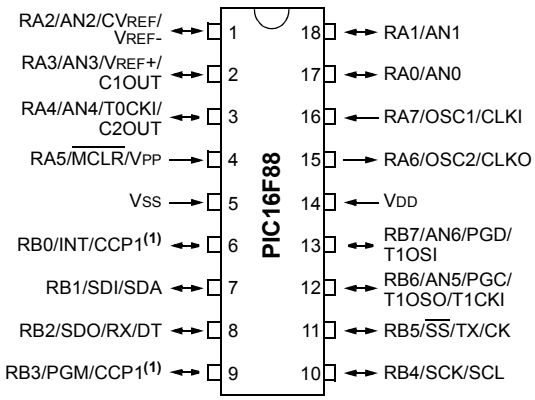
- Three Crystal modes:
  - LP, XT, HS up to 20 MHz
- Two External RC modes
- One External Clock mode:
  - ECIO up to 20 MHz
- Internal oscillator block:
  - 8 user selectable frequencies: 31 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz

### Peripheral Features:

- Capture, Compare, PWM (CCP) module:
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit, 7-channel Analog-to-Digital Converter
- Synchronous Serial Port (SSP) with SPI™ (Master/Slave) and $I^2C$™ (Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection:
  - RS-232 operation using internal oscillator (no external crystal required)
- Dual Analog Comparator module:
  - Programmable on-chip voltage reference
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

### Pin Diagram

**18-Pin DIP, SOIC**




**Note 1:** The CCP1 pin is determined by CCPMX in Configuration Word1 register.

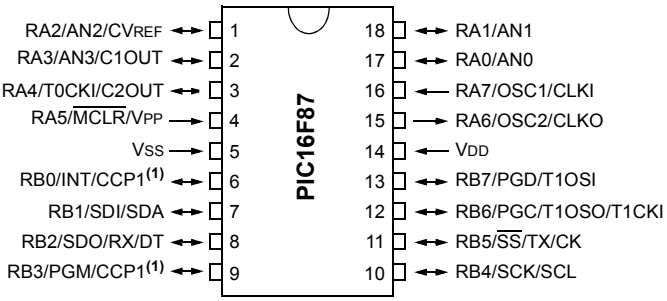
### Special Microcontroller Features:

- 100,000 erase/write cycles Enhanced FLASH program memory typical
- 1,000,000 typical erase/write cycles EEPROM data memory typical
- EEPROM Data Retention: > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Processor read/write access to program memory
- Low Voltage Programming
- In-Circuit Debugging via two pins
- Extended Watchdog Timer (WDT):
  - Programmable period from 1 ms to 268s
- Wide operating voltage range: 2.0V to 5.5V

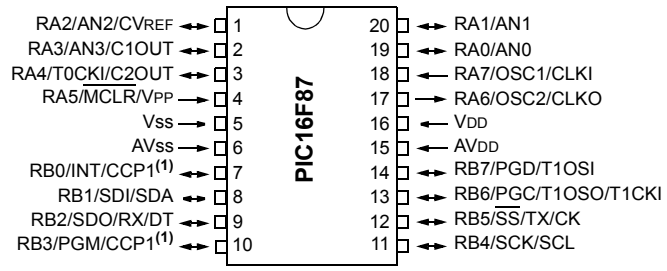
| Device | Program Memory | | Data Memory | | I/O Pins | 10-bit A/D (ch) | CCP (PWM) | USART | Comparators | SSP | Timers 8/16-bit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | FLASH (bytes) | # Single Word Instructions | SRAM (bytes) | EEPROM (bytes) | | | | | | | |
| PIC16F87 | 7168 | 4096 | 368 | 256 | 16 | 0 | 1 | Y | 2 | Y | 2/1 |
| PIC16F88 | 7168 | 4096 | 368 | 256 | 16 | 1 | 1 | Y | 2 | Y | 2/1 |

## Pin Diagrams

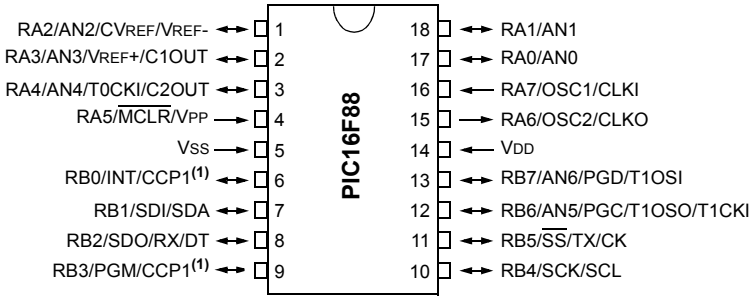
**18-Pin DIP, SOIC**



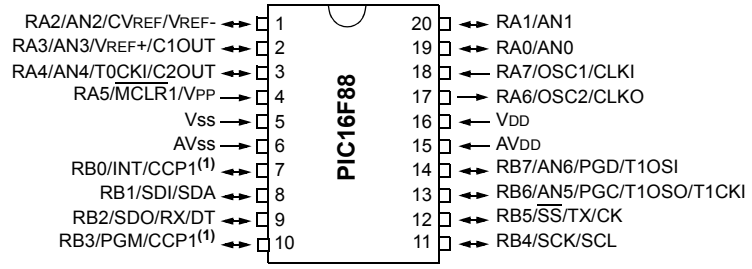

**20-Pin SSOP**




**18-Pin DIP & SOIC**




**20-Pin SSOP**

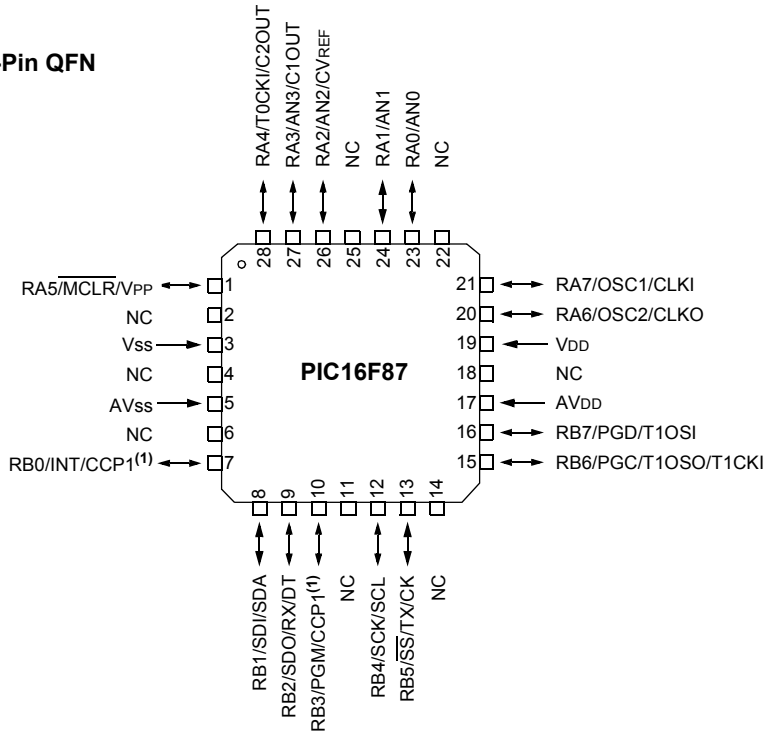



**Note 1:** The CCP1 pin is determined by CCPMX in Configuration Word1 register.

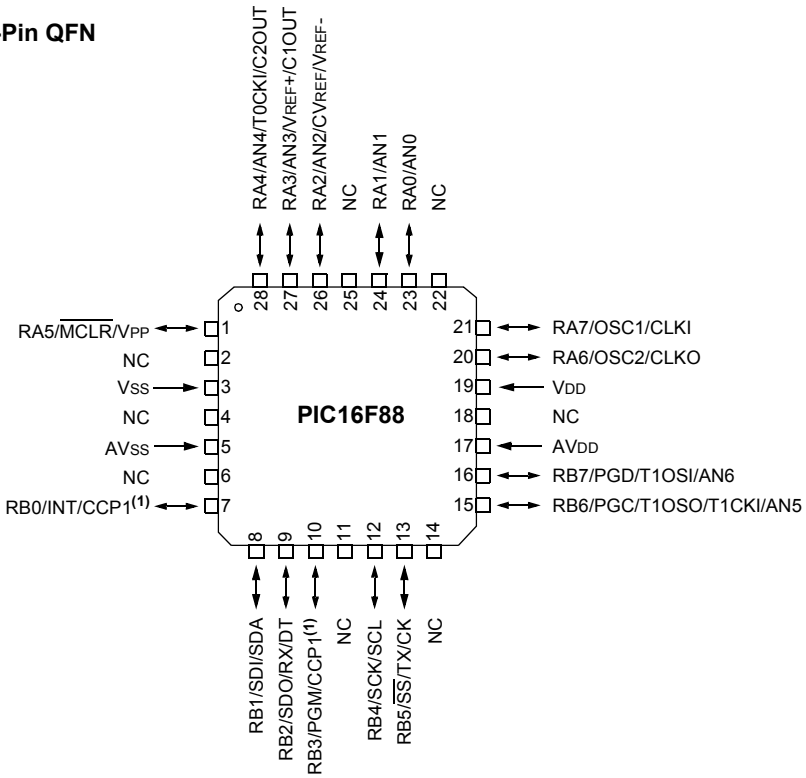
 

## Pin Diagrams (Cont'd)

**28-Pin QFN**




**28-Pin QFN**




**Note 1:** The CCP1 pin is determined by CCPMX in Configuration Word1 register.

## Table of Contents



## TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

### Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

### Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

### Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

## 1.0 DEVICE OVERVIEW

This document contains device specific information for the operation of the PIC16F87/88 devices. Additional information may be found in the PICmicro™ Mid-Range MCU Reference Manual (DS33023), which may be downloaded from the Microchip web site. This Reference Manual should be considered a complementary document to this data sheet, and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

The PIC16F87/88 belongs to the Mid-Range family of the PICmicro® devices. Block diagrams of the devices are shown in Figure 1-1 and Figure 1-2. These devices contain features that are new to the PIC16 product line:

- Low Power modes: The first PIC16 device to have Low Power modes that extend past SLEEP mode. RC_RUN allows the core and peripherals to be clocked from the INTRC, while SEC_RUN allows the core and peripherals to be clocked from the Low Power Timer1. Refer to Section 4.7 for further details.
- Internal RC oscillator with eight selectable frequencies, including 31.25 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, and 8 MHz. The INTRC can be configured as a primary or secondary clock source. Refer to Section 4.5 for further details.
- The Timer1 module current consumption has been greatly reduced from 20 μA (previous PIC16 devices) to 1.3 μA typical (32 kHz at 2V), which is ideal for real-time clock applications. Refer to Section 7.0 for further details.
- Extended Watchdog Timer (WDT) that can have a programmable period from 1 ms to 268s. The WDT has its own 16-bit prescaler. Refer to Section 15.12 for further details.
- Two-Speed Start-up: When the oscillator is configured for LP, XT, or HS, this feature will clock the device from the INTRC while the oscillator is warming up. This, in turn, will enable almost immediate code execution. Refer to Section 15.12.4 for further details.
- Fail-Safe Clock Monitor: This feature will allow the device to continue operation if the primary or secondary clock source fails, by switching over to the INTRC.
- The A/D module has a new register for PIC16 devices named ANSEL. This register allow easier configuration of Analog or Digital I/O pins.

**TABLE 1-1: AVAILABLE MEMORY IN PIC16F87/88 DEVICES**

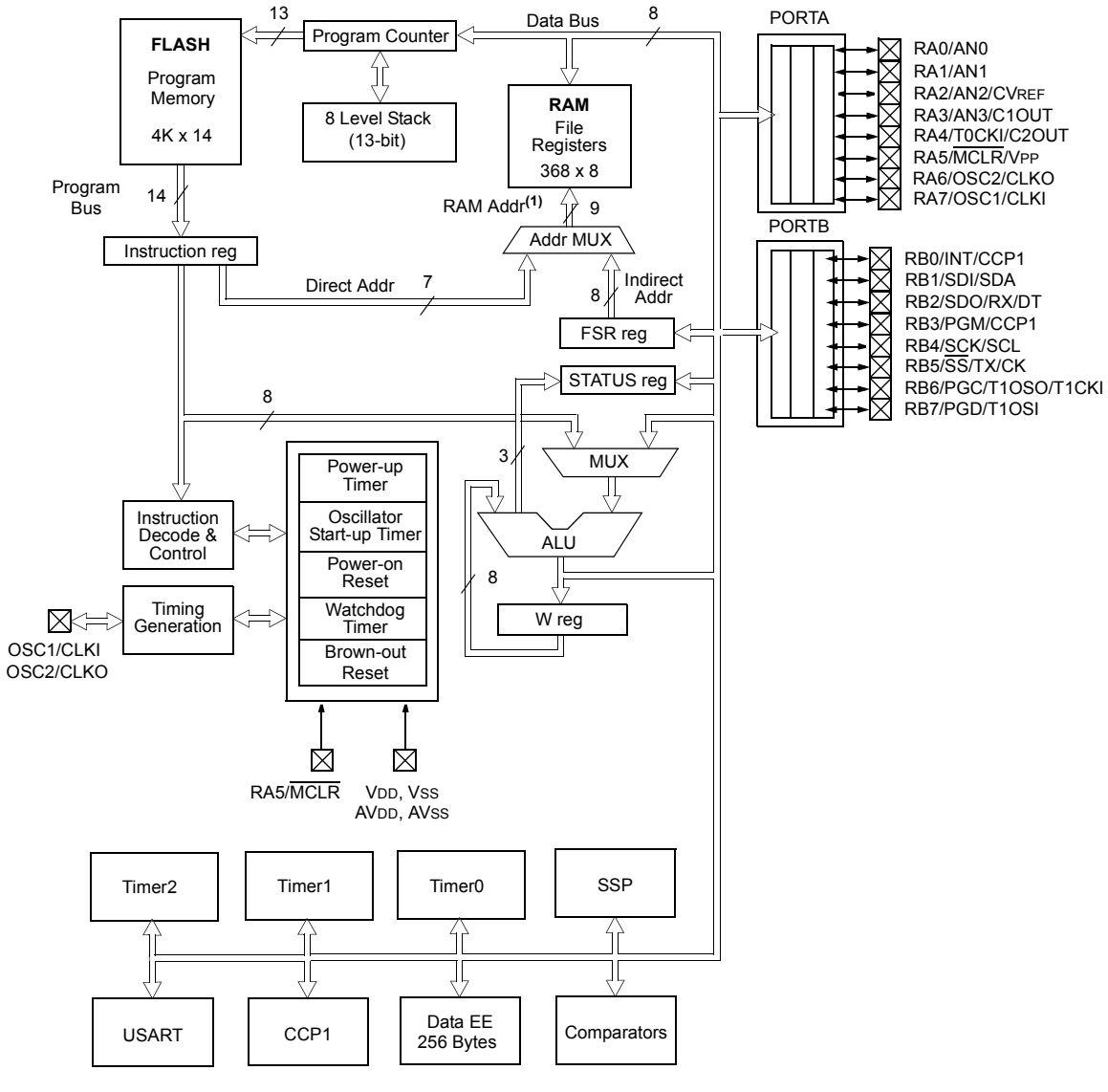
| Device | Program FLASH | Data Memory | Data EEPROM |
|---|---|---|---|
| PIC16F87/88 | 4K x 14 | 368 x 8 | 256 x 8 |

There are 16 I/O pins that are user configurable on a pin-to-pin basis. Some pins are multiplexed with other device functions. These functions include:

- External Interrupt
- Change on PORTB Interrupt
- Timer0 Clock Input
- Low Power Timer1 Clock/Oscillator
- Capture/Compare/PWM
- 10-bit, 7-channel A/D Converter (PIC16F88 only)
- SPI/$I^2C$
- Two Analog Comparators
- USART
- $\overline{MCLR}$ (RA5) can be configured as an Input

Table 1-2 details the pinout of the device with descriptions and details for each pin.

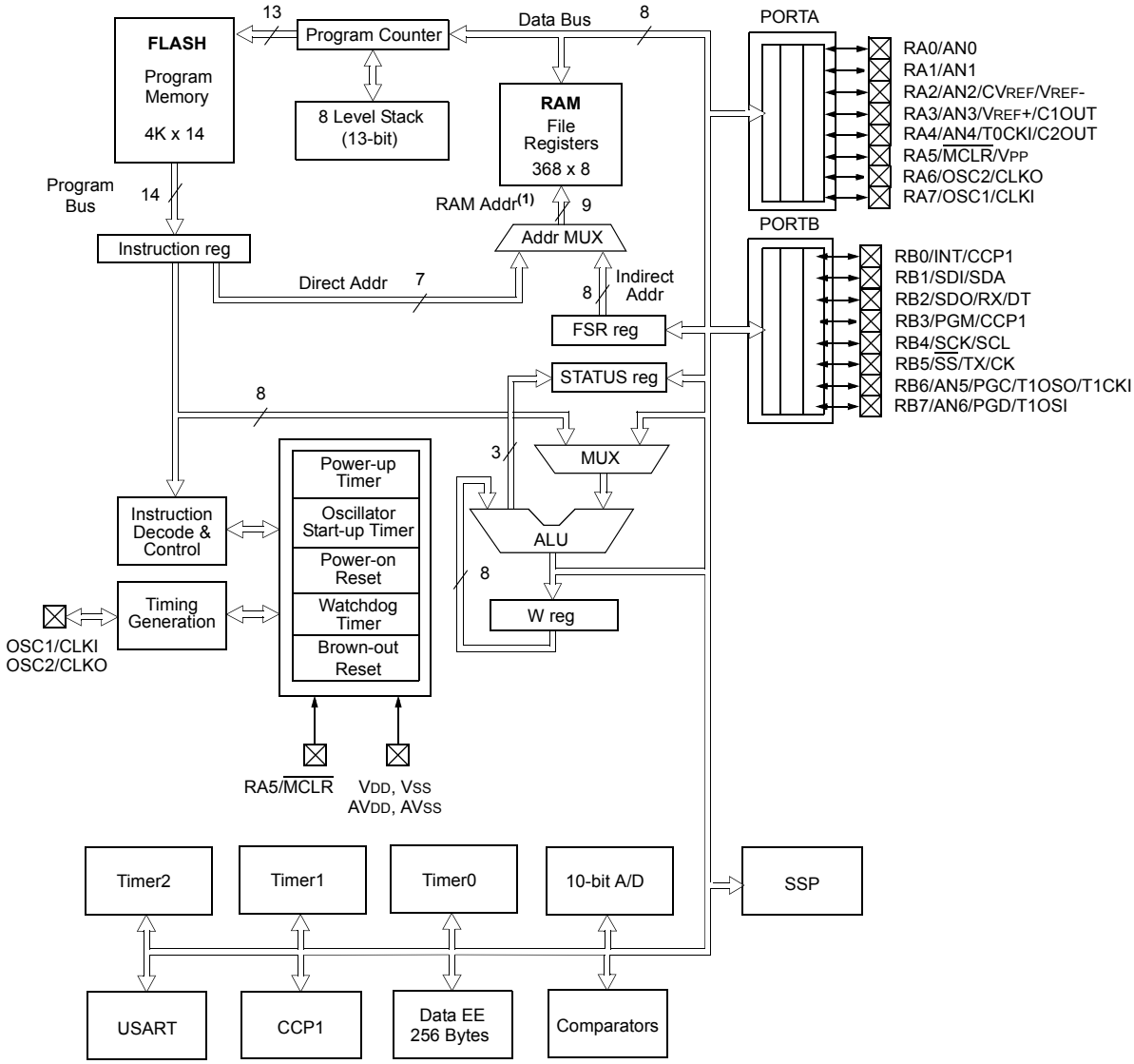
**FIGURE 1-1: PIC16F87 DEVICE BLOCK DIAGRAM**




**Note 1:** Higher order bits are from the STATUS register.

**FIGURE 1-2: PIC16F88 DEVICE BLOCK DIAGRAM**




**Note 1:** Higher order bits are from the STATUS register.

**TABLE 1-2: PIC16F87/88 PINOUT DESCRIPTION**

| Pin Name | PDIP/ SOIC Pin# | SSOP Pin# | QFN Pin# | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | | | | | | PORTA is a bi-directional I/O port. |
| RA0/AN0 | 17 | 19 | 23 | | | |
| RA0 | | | | I/O | TTL | Bi-directional I/O pin. |
| AN0 | | | | I | Analog | Analog input channel 0. |
| RA1/AN1 | 18 | 20 | 24 | | | |
| RA1 | | | | I/O | TTL | Bi-directional I/O pin. |
| AN1 | | | | I | Analog | Analog input channel 1. |
| RA2/CVREF/AN2/VREF- | 1 | 1 | 26 | | | |
| RA2 | | | | I/O | TTL | Bi-directional I/O pin. |
| CVREF | | | | O | | Comparator VREF output. |
| AN2 | | | | I | Analog | Analog input channel 2. |
| VREF-(4) | | | | I | Analog | A/D reference voltage (Low) input. |
| RA3/AN3/VREF+/C1OUT | 2 | 2 | 27 | | | |
| RA3 | | | | I/O | TTL | Bi-directional I/O pin. |
| AN3 | | | | I | Analog | Analog input channel 3. |
| VREF+(4) | | | | I | Analog | A/D reference voltage (High) input. |
| C1OUT | | | | O | | Comparator1 output. |
| RA4/AN4/T0CKI/C2OUT | 3 | 3 | 28 | | | |
| RA4 | | | | I/O | ST | Bi-directional I/O pin. |
| AN4(4) | | | | I | Analog | Analog input channel 4. |
| T0CKI | | | | I | ST | Clock input to the TMR0 timer/counter. |
| C2OUT | | | | O | | Comparator2 output. |
| RA5/$\overline{MCLR}$/VPP | 4 | 4 | 1 | | | |
| RA5 | | | | I | ST | Input pin. |
| $\overline{MCLR}$ | | | | I | ST | Master Clear (Reset). Input/programming voltage input. This pin is an active low RESET to the device. |
| VPP | | | | P | – | Programming voltage input. |
| RA6/OSC2/CLKO | 15 | 17 | 20 | | | |
| RA6 | | | | I/O | ST | Bi-directional I/O pin. |
| OSC2 | | | | O | – | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. |
| CLKO | | | | O | – | In RC mode, this pin outputs CLKO signal, which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate. |
| RA7/OSC1/CLKI | 16 | 18 | 21 | | | |
| RA7 | | | | I/O | ST | Bi-directional I/O pin. |
| OSC1 | | | | I | ST/CMOS(3) | Oscillator crystal input. |
| CLKI | | | | I | – | External clock source input. |

Legend: I = Input O = Output I/O = Input/Output P = Power
– = Not used TTL = TTL Input ST = Schmitt Trigger Input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.
**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.
**3:** This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.
**4:** PIC16F88 devices only.

**TABLE 1-2: PIC16F87/88 PINOUT DESCRIPTION (CONTINUED)**

| Pin Name | PDIP/ SOIC Pin# | SSOP Pin# | QFN Pin# | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | | | | | | PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs. |
| RB0/INT/CCP1 | 6 | 7 | 7 | | | |
| RB0 | | | | I/O | TTL | Bi-directional I/O pin. |
| INT | | | | I | ST(1) | External interrupt pin. |
| CCP1 | | | | I/O | ST | Capture input, Compare output, PWM output. |
| RB1/SDI/SDA | 7 | 8 | 8 | | | |
| RB1 | | | | I/O | TTL | Bi-directional I/O pin. |
| SDI | | | | I | ST | SPI Data in. |
| SDA | | | | I/O | ST | $I^2C$ Data. |
| RB2/SDO/RX/DT | 8 | 9 | 9 | | | |
| RB2 | | | | I/O | TTL | Bi-directional I/O pin. |
| SDO | | | | O | ST | SPI Data out. |
| RX | | | | I | | USART asynchronous receive. |
| DT | | | | I/O | | USART synchronous detect. |
| RB3/CCP1/PGM | 9 | 10 | 10 | | | |
| RB3 | | | | I/O | TTL | Bi-directional I/O pin. |
| CCP1 | | | | I/O | ST | Capture input, Compare output, PWM output. |
| PGM | | | | I | ST | Low Voltage ICSP programming enable pin. |
| RB4/SCK/SCL | 10 | 11 | 12 | | | |
| RB4 | | | | I/O | TTL | Bi-directional I/O pin. Interrupt-on-change pin. |
| SCK | | | | I/O | ST | Synchronous serial clock input/output for SPI. |
| SCL | | | | I | ST | Synchronous serial clock Input for $I^2C$. |
| RB5/$\overline{SS}$/TX/CK | 11 | 12 | 13 | | | |
| RB5 | | | | I/O | TTL | Bi-directional I/O pin. Interrupt-on-change pin. |
| $\overline{SS}$ | | | | I | TTL | Slave select for SPI in Slave mode. |
| TX | | | | O | | USART asynchronous transmit. |
| CK | | | | I/O | | USART synchronous clock. |
| RB6/T1OSO/T1CKI/ PGC/AN5 | 12 | 13 | 15 | | | |
| RB6 | | | | I/O | TTL | Bi-directional I/O pin. Interrupt-on-change pin. |
| T1OSO | | | | O | ST | Timer1 Oscillator output. |
| T1CKI | | | | I | ST | Timer1 external clock input. |
| PGC | | | | I/O | ST(2) | In-circuit debugger and programming clock pin. |
| AN5(4) | | | | I | | Analog input channel 5. |
| RB7/T1OSI/PGD/AN6 | 13 | 14 | 16 | | | |
| RB7 | | | | I/O | TTL | Bi-directional I/O pin. Interrupt-on-change pin. |
| T1OSI | | | | I | ST | Timer1 Oscillator input. |
| PGD | | | | I | ST(2) | In-circuit debugger and ICSP programming data pin. |
| AN6(4) | | | | I | | Analog input channel 6. |
| VSS | 5 | 5, 6 | 3, 5 | P | – | Ground reference for logic and I/O pins. |
| VDD | 14 | 15, 16 | 17, 19 | P | – | Positive supply for logic and I/O pins. |

Legend: I = Input O = Output I/O = Input/Output P = Power
– = Not used TTL = TTL Input ST = Schmitt Trigger Input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.
**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.
**3:** This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.
**4:** PIC16F88 devices only.

**NOTES:**

## 2.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F87/88. These are the program memory and the data memory. Each block has its own bus, so access to each block can occur during the same oscillator cycle.

The data memory can be further broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The PIC16F87/88's 256 bytes of data EEPROM memory have the address range 00h-FFh. More details on the EEPROM memory can be found in Section 3.0.

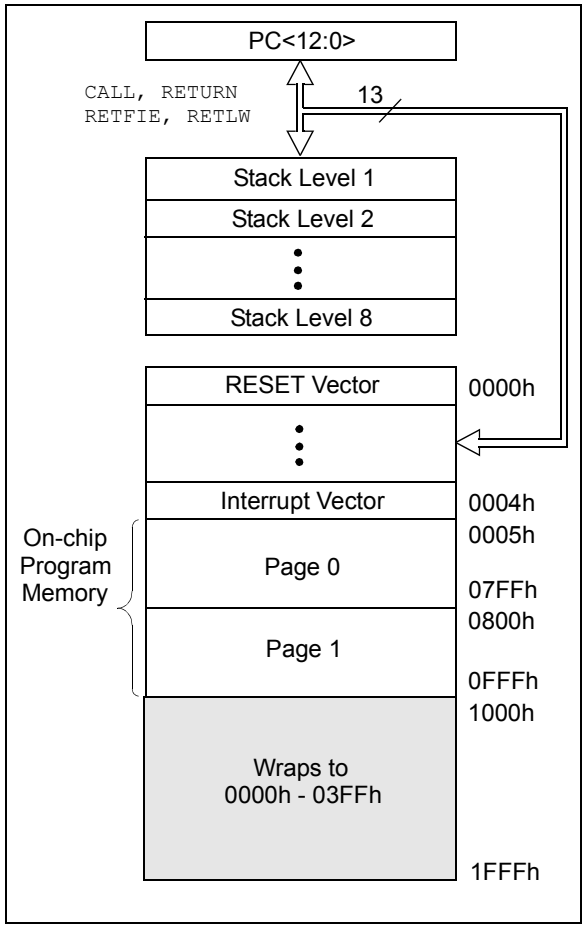
Additional information on device memory may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023).

### 2.1 Program Memory Organization

The PIC16F87/88 devices have a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F87/88, the first 4K x 14 (0000h-0FFFh) is physically implemented (see Figure 2-1). Accessing a location above the physically implemented address will cause a wraparound. For example, the same instruction will be accessed at locations 020h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h.

The RESET vector is at 0000h and the interrupt vector is at 0004h.

**FIGURE 2-1: PROGRAM MEMORY MAP AND STACK FOR PIC16F87/88**




### 2.2 Data Memory Organization

The Data Memory is partitioned into multiple banks that contain the General Purpose Registers and the Special Function Registers. Bits RP1 (STATUS<6>) and RP0 (STATUS<5>) are the bank select bits.

| RP1:RP0 | Bank |
|---|---|
| 00 | 0 |
| 01 | 1 |
| 10 | 2 |
| 11 | 3 |

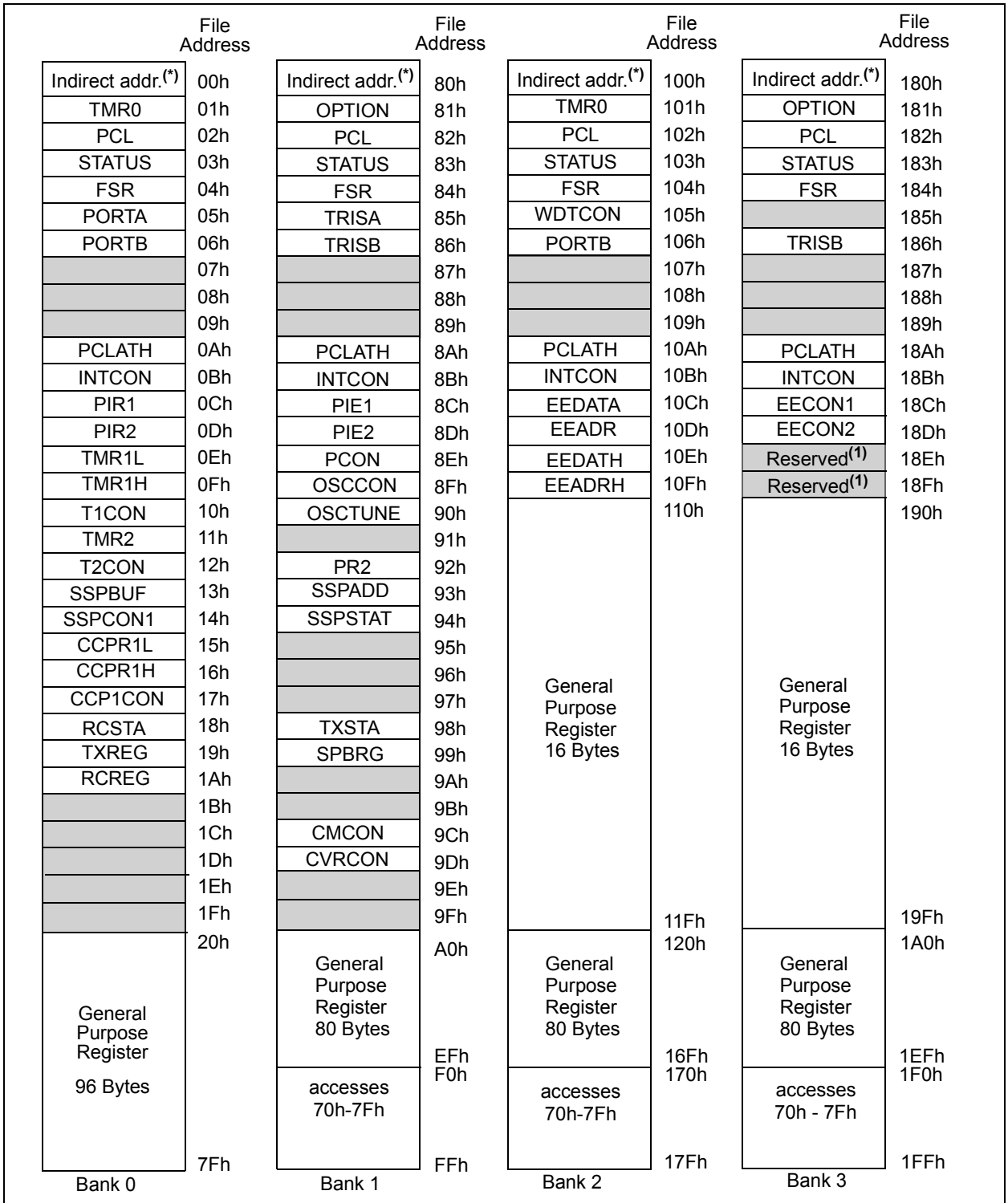
Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers, implemented as static RAM. All implemented banks contain SFRs. Some "high use" SFRs from one bank may be mirrored in another bank for code reduction and quicker access (e.g., the STATUS register is in Banks 0 - 3).

| **Note:** | EEPROM Data Memory description can be found in Section 3.0 of this data sheet. |
|---|---|

### 2.2.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly, or indirectly through the File Select Register FSR.

**FIGURE 2-2: PIC16F87 REGISTER FILE MAP**



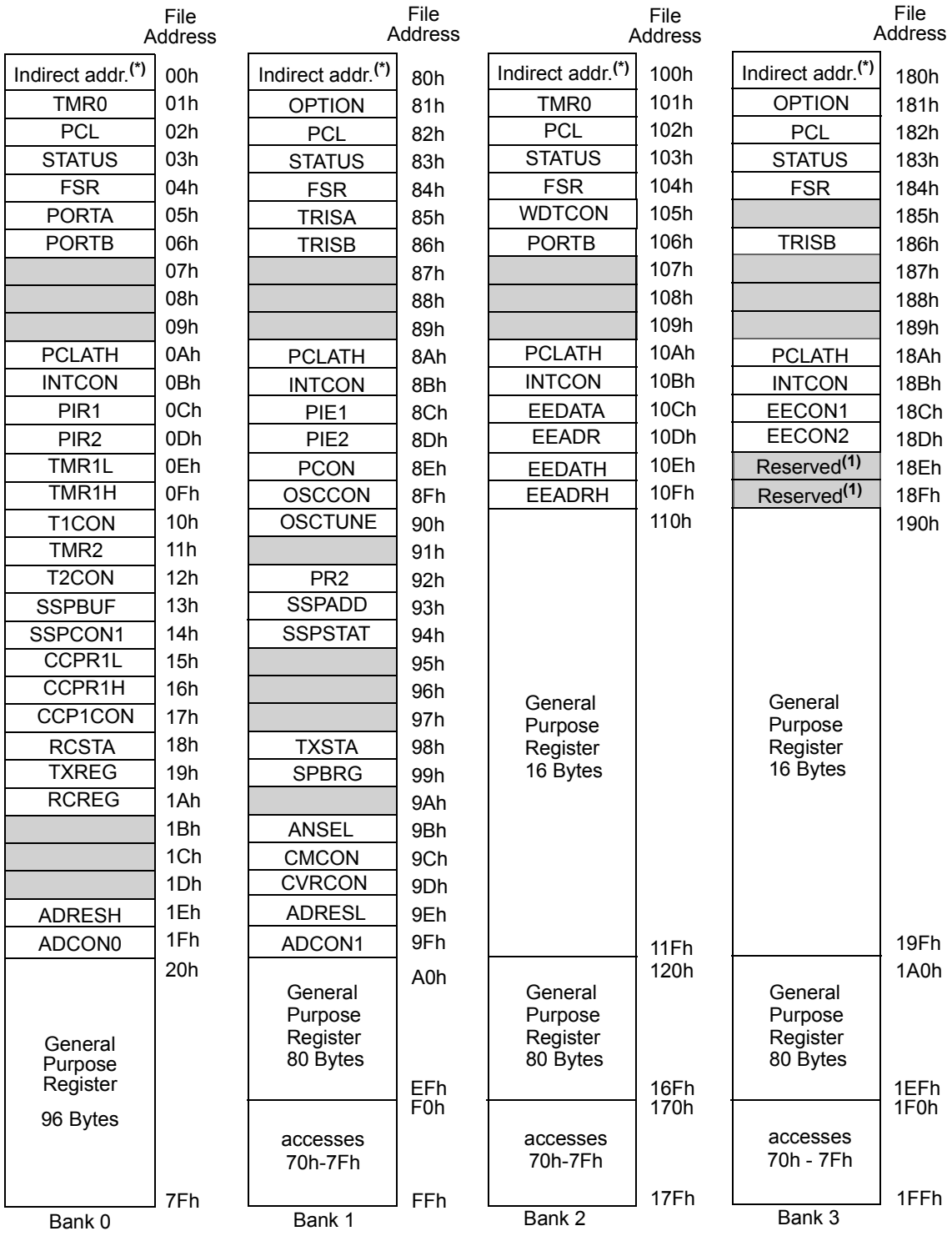

Unimplemented data memory locations, read as '0'.

* Not a physical register.

**Note 1:** This register is reserved, maintain this register clear.

**FIGURE 2-3: PIC16F88 REGISTER FILE MAP**




Unimplemented data memory locations, read as '0'.

* Not a physical register.

**Note 1:** This register is reserved, maintain this register clear.

### 2.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 2-1.

The Special Function Registers can be classified into two sets: core (CPU) and peripheral. Those registers associated with the core functions are described in detail in this section. Those related to the operation of the peripheral features are described in detail in the peripheral feature section.

**TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR |
|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 0** | | | | | | | | | | |
| 00h(2) | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 |
| 01h | TMR0 | Timer0 Module Register | | | | | | | | xxxx xxxx |
| 02h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 |
| 03h(2) | STATUS | IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 0001 1xxx |
| 04h(2) | FSR | Indirect Data Memory Address Pointer | | | | | | | | xxxx xxxx |
| 05h | PORTA - 87<br>PORTA - 88 | PORTA Data Latch when written; PORTA pins when read | | | | | | | | xxxx 0000<br>xxx0 0000 |
| 06h | PORTB - 87<br>PORTB - 88 | PORTB Data Latch when written; PORTB pins when read | | | | | | | | xxxx xxxx<br>00xx xxxx |
| 07h | — | Unimplemented | | | | | | | | — |
| 08h | — | Unimplemented | | | | | | | | — |
| 09h | — | Unimplemented | | | | | | | | — |
| 0Ah(1,2) | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the Program Counter | | | | | ---0 0000 |
| 0Bh(2) | INTCON | GIE | PEIE | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 |
| 0Dh | PIR2 | OSFIF | CMIF | — | EEIF | — | — | — | — | 00-0 ---- |
| 0Eh | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx |
| 0Fh | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx |
| 10h | T1CON | — | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | TMR1CS | TMR1ON | -000 0000 |
| 11h | TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 |
| 12h | T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 |
| 13h | SSPBUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx |
| 14h | SSPCON | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 |
| 15h | CCPR1L | Capture/Compare/PWM Register1 (LSB) | | | | | | | | xxxx xxxx |
| 16h | CCPR1H | Capture/Compare/PWM Register1 (MSB) | | | | | | | | xxxx xxxx |
| 17h | CCP1CON | — | — | CCP1X | CCP1Y | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x |
| 19h | TXREG | USART Transmit Data Register | | | | | | | | 0000 0000 |
| 1Ah | RCREG | USART Receive Data Register | | | | | | | | 0000 0000 |
| 1Bh | — | Unimplemented | | | | | | | | — |
| 1Ch | — | Unimplemented | | | | | | | | — |
| 1Dh | — | Unimplemented | | | | | | | | — |
| 1Eh | ADRESH(4) | A/D Result Register High Byte | | | | | | | | xxxx xxxx |
| 1Fh | ADCON0(4) | ADCS1 | ADCS0 | CHS2 | CHS1 | CHS0 | GO/$\overline{DONE}$ | — | ADON | 0000 00-0 |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

**3:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

**4:** PIC16F88 device only.

**TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR |
|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 1** | | | | | | | | | | |
| 80h(2) | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 |
| 81h | OPTION | $\overline{RBPU}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 |
| 82h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 |
| 83h(2) | STATUS | IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 0001 1xxx |
| 84h(2) | FSR | Indirect Data Memory Address Pointer | | | | | | | | xxxx xxxx |
| 85h | TRISA | TRISA7 | TRISA6 | TRISA5(3) | PORTA Data Direction Register (TRISA<4:0>) | | | | | 1111 1111 |
| 86h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 |
| 87h | — | Unimplemented | | | | | | | | — |
| 88h | — | Unimplemented | | | | | | | | — |
| 89h | — | Unimplemented | | | | | | | | — |
| 8Ah(1,2) | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the Program Counter | | | | | ---0 0000 |
| 8Bh(2) | INTCON | GIE | PEIE | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 |
| 8Dh | PIE2 | OSFIE | CMIE | — | EEIE | — | — | — | — | 00-0 ---- |
| 8Eh | PCON | — | — | — | — | — | — | $\overline{POR}$ | $\overline{BOR}$ | ---- --qq |
| 8Fh | OSCCON | — | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 | -000 0000 |
| 90h | OSCTUNE | — | — | TUN5 | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 | --00 0000 |
| 91h | — | Unimplemented | | | | | | | | — |
| 92h | PR2 | Timer2 Period Register | | | | | | | | 1111 1111 |
| 93h | SSPADD | Synchronous Serial Port ($I^2C$ mode) Address Register | | | | | | | | 0000 0000 |
| 94h | SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 |
| 95h | — | Unimplemented | | | | | | | | — |
| 96h | — | Unimplemented | | | | | | | | — |
| 97h | — | Unimplemented | | | | | | | | — |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 |
| 9Ah | — | Unimplemented | | | | | | | | — |
| 9Bh | ANSEL(4) | — | ANS6 | ANS5 | ANS4 | ANS3 | ANS2 | ANS1 | ANS0 | -111 1111 |
| 9Ch | CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0000 |
| 9Dh | CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 |
| 9Eh | ADRESL(4) | A/D Result Register Low Byte | | | | | | | | xxxx xxxx |
| 9Fh | ADCON1(4) | ADFM | ADCS2 | VCFG1 | VCFG0 | — | — | — | — | 0000 ---- |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

**3:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

**4:** PIC16F88 device only.

**TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR |
|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 2** | | | | | | | | | | |
| 100h(2) | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 |
| 101h | TMR0 | Timer0 Module Register | | | | | | | | xxxx xxxx |
| 102h(2) | PCL | Program Counter's (PC) Least Significant Byte | | | | | | | | 0000 0000 |
| 103h(2) | STATUS | IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 0001 1xxx |
| 104h(2) | FSR | Indirect Data Memory Address Pointer | | | | | | | | xxxx xxxx |
| 105h | WDTCON | — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | SWDTEN | ---0 1000 |
| 106h | PORTB | PORTB Data Latch when written; PORTB pins when read | | | | | | | | xxxx xxxx |
| 107h | — | Unimplemented | | | | | | | | — |
| 108h | — | Unimplemented | | | | | | | | — |
| 109h | — | Unimplemented | | | | | | | | — |
| 10Ah(1,2) | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the Program Counter | | | | | ---0 0000 |
| 10Bh(2) | INTCON | GIE | PEIE | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x |
| 10Ch | EEDATA | EEPROM Data Register Low Byte | | | | | | | | xxxx xxxx |
| 10Dh | EEADR | EEPROM Address Register Low Byte | | | | | | | | xxxx xxxx |
| 10Eh | EEDATH | — | — | EEPROM Data Register High Byte | | | | | | --xx xxxx |
| 10Fh | EEADRH | — | — | — | — | EEPROM Address Register High Byte | | | | ---- xxxx |
| **Bank 3** | | | | | | | | | | |
| 180h(2) | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 |
| 181h | OPTION | RBPU | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 |
| 182h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 |
| 183h(2) | STATUS | IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 0001 1xxx |
| 184h(2) | FSR | Indirect Data Memory Address Pointer | | | | | | | | xxxx xxxx |
| 185h | — | Unimplemented | | | | | | | | — |
| 186h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 |
| 187h | — | Unimplemented | | | | | | | | — |
| 188h | — | Unimplemented | | | | | | | | — |
| 189h | — | Unimplemented | | | | | | | | — |
| 18Ah(1,2) | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the Program Counter | | | | | ---0 0000 |
| 18Bh(2) | INTCON | GIE | PEIE | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x |
| 18Ch | EECON1 | EEPGD | — | — | FREE | WRERR | WREN | WR | RD | x--x x000 |
| 18Dh | EECON2 | EEPROM Control Register2 (not a physical register) | | | | | | | | ---- ---- |
| 18Eh | — | Reserved, maintain clear | | | | | | | | 0000 0000 |
| 18Fh | — | Reserved, maintain clear | | | | | | | | 0000 0000 |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

**3:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

**4:** PIC16F88 device only.

#### 2.2.2.1 STATUS Register

The STATUS register, shown in Register 2-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as '`000u u1uu`' (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C or DC bits from the STATUS register. For other instructions not affecting any status bits, see Section 16.0, "Instruction Set Summary".

| **Note:** | The C and DC bits operate as a $\overline{borrow}$ and digit $\overline{borrow}$ bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples. |
|---|---|

**REGISTER 2-1: STATUS REGISTER (ADDRESS 03h, 83h, 103h, 183h)**

| R/W-0 | R/W-0 | R/W-0 | R-1 | R-1 | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| bit 7 | | | | | | | bit 0 |

bit 7 **IRP:** Register Bank Select bit (used for indirect addressing)
`1` = Bank 2, 3 (100h - 1FFh)
`0` = Bank 0, 1 (00h - FFh)

bit 6-5 **RP<1:0>:** Register Bank Select bits (used for direct addressing)
`11` = Bank 3 (180h - 1FFh)
`10` = Bank 2 (100h - 17Fh)
`01` = Bank 1 (80h - FFh)
`00` = Bank 0 (00h - 7Fh)
Each bank is 128 bytes

bit 4 **$\overline{TO}$:** Time-out bit
`1` = After power-up, `CLRWDT` instruction, or `SLEEP` instruction
`0` = A WDT time-out occurred

bit 3 **$\overline{PD}$:** Power-down bit
`1` = After power-up or by the `CLRWDT` instruction
`0` = By execution of the `SLEEP` instruction

bit 2 **Z:** Zero bit
`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit carry/$\overline{borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions)**(1)**
`1` = A carry-out from the 4th low order bit of the result occurred
`0` = No carry-out from the 4th low order bit of the result

bit 0 **C:** Carry/$\overline{borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions)**(1,2)**
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

**Note 1:** For borrow, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand.

**2:** For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

| 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 |

### 2.2.2.2 OPTION Register

The OPTION register is a readable and writable register that contains various control bits to configure the TMR0 prescaler/WDT postscaler (single assignable register known also as the prescaler), the External INT Interrupt, TMR0, and the weak pull-ups on PORTB.

| **Note:** | To achieve a 1:1 prescaler assignment for the TMR0 register, assign the prescaler to the Watchdog Timer. Although the prescaler can be assigned to either the WDT or Timer0, but not both, a new divide counter is implemented in the WDT circuit to give multiple WDT time-out selection. This allows TMR0 and WDT to each have their own scaler. Refer to Section 15.12 for further details. |
|---|---|

**REGISTER 2-2: OPTION REGISTER (ADDRESS 81h, 181h)**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{RBPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **$\overline{\text{RBPU}}$**: PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG:** Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5 **T0CS:** TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE:** TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin

bit 3 **PSA:** Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

| Bit Value | TMR0 Rate | WDT Rate |
|---|---|---|
| 000 | 1 : 2 | 1 : 1 |
| 001 | 1 : 4 | 1 : 2 |
| 010 | 1 : 8 | 1 : 4 |
| 011 | 1 : 16 | 1 : 8 |
| 100 | 1 : 32 | 1 : 16 |
| 101 | 1 : 64 | 1 : 32 |
| 110 | 1 : 128 | 1 : 64 |
| 111 | 1 : 256 | 1 : 128 |

| 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 |

### 2.2.2.3 INTCON Register

The INTCON Register is a readable and writable register that contains various enable and flag bits for the TMR0 register overflow, RB Port change and External RB0/INT pin interrupts.

**Note:** Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

**REGISTER 2-3: INTCON: INTERRUPT CONTROL REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-x |
|---|---|---|---|---|---|---|---|
| GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF |
| bit 7 | | | | | | | bit 0 |

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE:** RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INTF:** RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

| 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 |

### 2.2.2.4 PIE1 Register

This register contains the individual enable bits for the peripheral interrupts.

| **Note:** | Bit PEIE (INTCON<6>) must be set to enable any peripheral interrupt. |
|---|---|

**REGISTER 2-4: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1**

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | ADIE(1) | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **ADIE(1):** A/D Converter Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **RCIE:** USART Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TXIE:** USART Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **SSPIE:** Synchronous Serial Port (SSP) Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

**Note 1:** This bit is only implemented on the PIC16F88. The bit will read ‘0’ on the PIC16F87.

| 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 |

### 2.2.2.5 PIR1 Register

This register contains the individual flag bits for the Peripheral interrupts.

| **Note:** | Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. |
|---|---|

**REGISTER 2-5: PIR1: PERIPHERAL INTERRUPT STATUS REGISTER 1**

| U-0 | R/W-0 | R-0 | R-0 | R-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | ADIF(1) | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **ADIF(1):** A/D Converter Interrupt Flag bit (only on PIC16F86)
1 = The A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5 **RCIF:** USART Receive Interrupt Flag bit
1 = The USART receive buffer is full (cleared by reading RCREG)
0 = The USART receive buffer is not full

bit 4 **TXIF:** USART Transmit Interrupt Flag bit
1 = The USART transmit buffer is empty (cleared by writing to TXREG)
0 = The USART transmit buffer is full

bit 3 **SSPIF:** Synchronous Serial Port (SSP) Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode

bit 1 **TMR2IF:** TMR2 to PR2 Interrupt Flag bit
1 = A TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = The TMR1 register overflowed (must be cleared in software)
0 = The TMR1 register did not overflow

**Note 1:** This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

| 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 |

### 2.2.2.6 PIE2 Register

The PIE2 register contains the individual enable bit for the EEPROM write operation interrupt.

**REGISTER 2-6: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2**

| R/W-0 | R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| OSFIE | CMIE | — | EEIE | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

bit 7 **OSFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **CMIE:** Comparator Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 4 **EEIE:** EEPROM Write Operation Interrupt Enable bit
1 = Enabled
0 = Disabled

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

### 2.2.2.7 PIR2 Register

The PIR2 register contains the flag bit for the EEPROM write operation interrupt.

| **Note:** | Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. |
|---|---|

**REGISTER 2-7: PIR2: PERIPHERAL INTERRUPT STATUS REGISTER 2**

| R/W-0 | R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| OSFIF | CMIF | — | EEIF | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

bit 7 **OSFIF:** Oscillator Fail Interrupt Flag bit
1 = System oscillator failed, clock input has changed to INTRC (must be cleared in software)
0 = System clock operating

bit 6 **CMIF:** Comparator Interrupt Flag bit
1 = Comparator input has changed (must be cleared in software)
0 = Comparator input has not changed

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

bit 4 **EEIF:** EEPROM Write Operation Interrupt Flag bit
1 = The write operation completed (must be cleared in software)
0 = The write operation is not complete or has not been started

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

2.2.2.8 PCON Register

| | |
|---|---|
| **Note:** | Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. |

The Power Control (PCON) register contains a flag bit to allow differentiation between a Power-on Reset (POR), a Brown-out Reset, an external $\overline{MCLR}$ Reset and WDT Reset.

| | |
|---|---|
| **Note:** | $\overline{BOR}$ is unknown on Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if $\overline{BOR}$ is clear, indicating a brown-out has occurred. The $\overline{BOR}$ status bit is a 'don't care' and is not necessarily predictable if the brown-out circuit is disabled (by clearing the BOREN bit in the Configuration word). |

**REGISTER 2-8: PCON: POWER CONTROL REGISTER (ADDRESS 8Eh)**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-x |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | $\overline{POR}$ | $\overline{BOR}$ |
| bit 7 | | | | | | | bit 0 |

bit 7-2 **Unimplemented:** Read as '0'

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

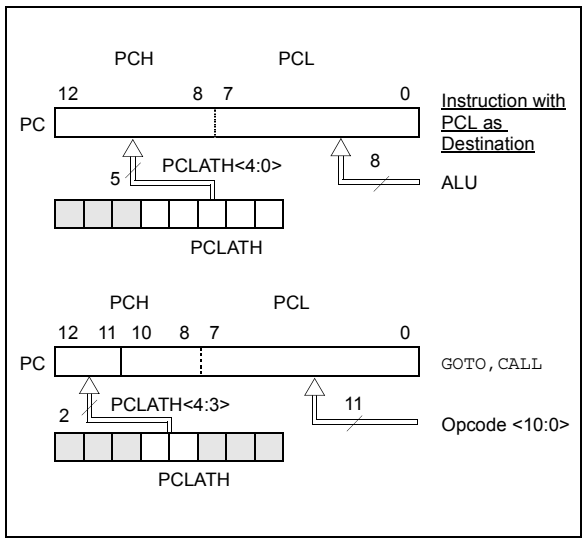
bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = No Brown-out Reset occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

| 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 |

## 2.3 PCL and PCLATH

The program counter (PC) is 13-bits wide. The low byte comes from the PCL register, which is a readable and writable register. The upper bits (PC<12:8>) are not readable, but are indirectly writable through the PCLATH register. On any RESET, the upper bits of the PC will be cleared. Figure 2-4 shows the two situations for the loading of the PC. The upper example in the figure shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in the figure shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

**FIGURE 2-4: LOADING OF PC IN DIFFERENT SITUATIONS**




### 2.3.1 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When doing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to the application note, *"Implementing a Table Read"* (AN556).

### 2.3.2 STACK

The PIC16F87/88 family has an 8-level deep x 13-bit wide hardware stack. The stack space is not part of either program or data space and the stack pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed, or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

**Note 1:** There are no status bits to indicate stack overflow or stack underflow conditions.

**2:** There are no instructions/mnemonics called PUSH or POP. These are actions that occur from the execution of the CALL, RETURN, RETLW and RETFIE instructions, or the vectoring to an interrupt address.

## 2.4 Program Memory Paging

All PIC16F87/88 devices are capable of addressing a continuous 8K word block of program memory. The CALL and GOTO instructions provide only 11-bits of address to allow branching within any 2K program memory page. When doing a CALL or GOTO instruction, the upper 2 bits of the address are provided by PCLATH<4:3>. When doing a CALL or GOTO instruction, the user must ensure that the page select bits are programmed so that the desired program memory page is addressed. If a return from a CALL instruction (or interrupt) is executed, the entire 13-bit PC is popped off the stack. Therefore, manipulation of the PCLATH<4:3> bits is not required for the return instructions (which POPs the address from the stack).

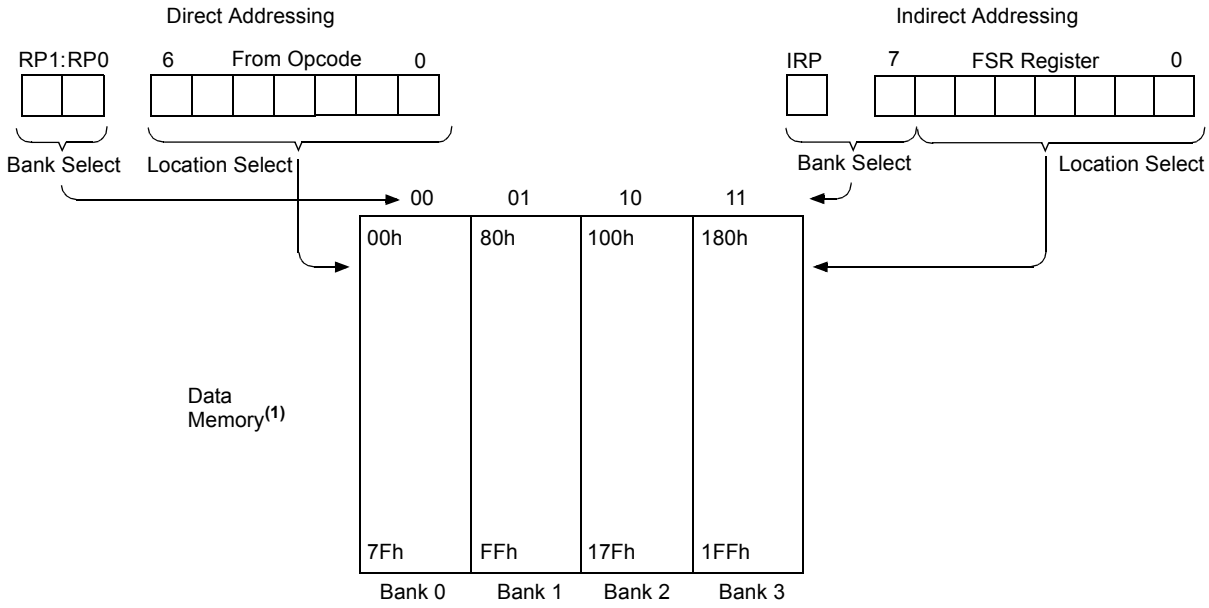
**Note:** The contents of the PCLATH register are unchanged after a RETURN or RETFIE instruction is executed. The user must rewrite the contents of the PCLATH register for any subsequent subroutine calls or GOTO instructions.

Example 2-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that PCLATH is saved and restored by the Interrupt Service Routine (if interrupts are used).

**EXAMPLE 2-1: CALL OF A SUBROUTINE IN PAGE 1 FROM PAGE 0**

```
        ORG 0x500
        BCF PCLATH,4
        BSF PCLATH,3    ;Select page 1
                        ;(800h-FFFh)
        CALL SUB1_P1    ;Call subroutine in
        :               ;page 1 (800h-FFFh)
        :
        ORG 0x900       ;page 1 (800h-FFFh)
SUB1_P1
        :               ;called subroutine
                        ;page 1 (800h-FFFh)
        :
        RETURN          ;return to
                        ;Call subroutine
                        ;in page 0
                        ;(000h-7FFh)
```

**FIGURE 2-5: DIRECT/INDIRECT ADDRESSING**




**Note 1:** For register file map detail, see Figure 2-2 or Figure 2-3.

## 3.0 DATA EEPROM AND FLASH PROGRAM MEMORY

The Data EEPROM and FLASH Program memory is readable and writable during normal operation (over the full VDD range). This memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers. There are six SFRs used to read and write this memory:

- EECON1
- EECON2
- EEDATA
- EEDATH
- EEADR
- EEADRH

When interfacing the data memory block, EEDATA holds the 8-bit data for read/write, and EEADR holds the address of the EEPROM location being accessed. The PIC16F87/88 devices have 256 bytes of data EEPROM, with an address range from 00h to 0FFh. When writing to unimplemented locations, the charge pump will be turned off.

When interfacing the program memory block, the EEDATA and EEDATH registers form a two-byte word that holds the 14-bit data for read/write, and the EEADR and EEADRH registers form a two-byte word that holds the 13-bit address of the EEPROM location being accessed. The PIC16F87/88 devices have 4K words of program FLASH, with an address range from 0000h to 0FFFh. Addresses above the range of the respective device will wraparound to the beginning of program memory.

The EEPROM data memory allows single byte read and write. The FLASH program memory allows single word reads and four-word block writes. Program memory writes must first start with a 32-word block erase, then write in 4-word blocks. A byte write in data EEPROM memory automatically erases the location and writes the new data (erase before write).

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device for byte or word operations.

When the device is code protected, the CPU may continue to read and write the data EEPROM memory. Depending on the settings of the write protect bits, the device may or may not be able to write certain blocks of the program memory; however, reads of the program memory are allowed. When code protected, the device programmer can no longer access data or program memory; this does NOT inhibit internal reads or writes.

### 3.1 EEADR and EEADRH

The EEADRH:EEADR register pair can address up to a maximum of 256 bytes of data EEPROM, or up to a maximum of 8K words of program EEPROM. When selecting a data address value, only the LSByte of the address is written to the EEADR register. When selecting a program address value, the MSByte of the address is written to the EEADRH register and the LSByte is written to the EEADR register.

If the device contains less memory than the full address reach of the address register pair, the Most Significant bits of the registers are not implemented. For example, if the device has 128 bytes of data EEPROM, the Most Significant bit of EEADR is not implemented on access to data EEPROM.

### 3.2 EECON1 and EECON2 Registers

EECON1 is the control register for memory accesses.

Control bit EEPGD determines if the access will be a program or data memory access. When clear, as it is when reset, any subsequent operations will operate on the data memory. When set, any subsequent operations will operate on the program memory.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write or erase operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write (or erase) operation is interrupted by a $\overline{\text{MCLR}}$, or a WDT Time-out Reset during normal operation. In these situations, following RESET, the user can check the WRERR bit and rewrite the location. The data and address will be unchanged in the EEDATA and EEADR registers.

Interrupt flag bit, EEIF in the PIR2 register, is set when write is complete. It must be cleared in software.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the EEPROM write sequence.

**REGISTER 3-1: EECON1: EEPROM ACCESS CONTROL REGISTER 1 (ADDRESS 18Ch)**

| R/W-x | U-0 | U-0 | R/W-x | R/W-x | R/W-0 | R/S-0 | R/S-0 |
|---|---|---|---|---|---|---|---|
| EEPGD | — | — | FREE | WRERR | WREN | WR | RD |
| bit 7 | | | | | | | bit 0 |

bit 7 **EEPGD**: Program/Data EEPROM Select bit
1 = Accesses program memory
0 = Accesses data memory
Reads '0' after a POR; this bit cannot be changed while a write operation is in progress.

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

bit 4 **FREE:** EEPROM Forced Row Erase bit
1 = Erase the program memory row addressed by EEADRH:EEADR on the next WR command
0 = Perform write only

bit 3 **WRERR:** EEPROM Error Flag bit
1 = A write operation is prematurely terminated
(any $\overline{\text{MCLR}}$ or any WDT Reset during normal operation)
0 = The write operation completed

bit 2 **WREN:** EEPROM Write Enable bit
1 = Allows write cycles
0 = Inhibits write to the EEPROM

bit 1 **WR:** Write Control bit
1 = Initiates a write cycle. The bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.
0 = Write cycle to the EEPROM is complete

bit 0 **RD:** Read Control bit
1 = Initiates an EEPROM read, RD is cleared in hardware. The RD bit can only be set (not cleared) in software.
0 = Does not initiate an EEPROM read

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

## 3.3 Reading 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 in the very next cycle, in the EEDATA register; therefore, it can be read in the next instruction (see Example 3-1). EEDATA will hold this value until another read, or until it is written to by the user (during a write operation).

The steps to reading the EEPROM data memory are:

1. Write the address to EEADR. Make sure that the address is not larger than the memory size of the device.
2. Clear the EEPGD bit to point to EEPROM data memory.
3. Set the RD bit to start the read operation.
4. Read the data from the EEDATA register.

**EXAMPLE 3-1: DATA EEPROM READ**

```
BANKSEL EEADR          ; Select Bank of EEADR
MOVF    ADDR,W         ;
MOVWF   EEADR          ; Data Memory Address
                       ; to read
BANKSEL EECON1         ; Select Bank of EECON1
BCF     EECON1,EEPGD   ; Point to Data memory
BSF     EECON1,RD      ; EE Read
BANKSEL EEDATA         ; Select Bank of EEDATA
MOVF    EEDATA,W       ; W = EEDATA
```

## 3.4 Writing to Data EEPROM Memory

To write an EEPROM data location, the user must first write the address to the EEADR register and the data to the EEDATA register. Then, the user must follow a specific write sequence to initiate the write for each byte.

The write will not initiate if the write sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. We strongly recommend that interrupts be disabled during this code segment (see Example 3-2).

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

After a write sequence has been initiated, clearing the WREN bit will not affect this write cycle. The WR bit will be inhibited from being set unless the WREN bit is set. At the completion of the write cycle, the WR bit is cleared in hardware and the EE Write Complete Interrupt Flag bit (EEIF) is set. The user can either enable this interrupt, or poll this bit. EEIF must be cleared by software.

The steps to write to EEPROM data memory are:

1. If step 10 is not implemented, check the WR bit to see if a write is in progress.
2. Write the address to EEADR. Make sure that the address is not larger than the memory size of the device.
3. Write the 8-bit data value to be programmed in the EEDATA register.
4. Clear the EEPGD bit to point to EEPROM data memory.
5. Set the WREN bit to enable program operations.
6. Disable interrupts (if enabled).
7. Execute the special five instruction sequence:
   - Write 55h to EECON2 in two steps (first to W, then to EECON2)
   - Write AAh to EECON2 in two steps (first to W, then to EECON2)
   - Set the WR bit
8. Enable interrupts (if using interrupts).
9. Clear the WREN bit to disable program operations.
10. At the completion of the write cycle, the WR bit is cleared and the EEIF interrupt flag bit is set (EEIF must be cleared by firmware). If step 1 is not implemented, then firmware should check for EEIF to be set, or WR to clear, to indicate the end of the program cycle.

**EXAMPLE 3-2: DATA EEPROM WRITE**

```
        BANKSEL EECON1       ; Select Bank of
                             ; EECON1
        BTFSC   EECON1,WR    ; Wait for write
        GOTO    $-1          ; to complete
        BANKSEL EEADR        ; Select Bank of
                             ; EEADR
        MOVF    ADDR,W       ;
        MOVWF   EEADR        ; Data Memory
                             ; Address to write
        MOVF    VALUE,W      ;
        MOVWF   EEDATA       ; Data Memory Value
                             ; to write
        BANKSEL EECON1       ; Select Bank of
                             ; EECON1
        BCF     EECON1,EEPGD; Point to DATA
                             ; memory
        BSF     EECON1,WREN  ; Enable writes

        BCF     INTCON,GIE   ; Disable INTs.
Required
Sequence
        MOVLW   55h          ;
        MOVWF   EECON2       ; Write 55h
        MOVLW   AAh          ;
        MOVWF   EECON2       ; Write AAh
        BSF     EECON1,WR    ; Set WR bit to
                             ; begin write
        BSF     INTCON,GIE   ; Enable INTs.
        BCF     EECON1,WREN  ; Disable writes
```

## 3.5 Reading FLASH Program Memory

To read a program memory location, the user must write two bytes of the address to the EEADR and EEADRH registers, set the EEPGD control bit (EECON1<7>), and then set control bit, RD (EECON1<0>). Once the read control bit is set, the program memory FLASH controller will use the second instruction cycle to read the data. This causes the second instruction immediately following the "BSF EECON1,RD" instruction to be ignored. The data is available in the very next cycle, in the EEDATA and EEDATH registers; therefore, it can be read as two bytes in the following instructions. EEDATA and EEDATH registers will hold this value until another read, or until it is written to by the user (during a write operation).

**EXAMPLE 3-3: FLASH PROGRAM READ**

```
BANKSEL EEADRH        ; Select Bank of EEADRH
MOVF    ADDRH, W      ;
MOVWF   EEADRH        ; MS Byte of Program
                      ; Address to read
MOVF    ADDRL, W      ;
MOVWF   EEADR         ; LS Byte of Program
                      ; Address to read
BANKSEL EECON1        ; Select Bank of EECON1
BSF     EECON1, EEPGD ; Point to PROGRAM
                      ; memory
BSF     EECON1, RD    ; EE Read
                      ;
NOP                   ; Any instructions
                      ; here are ignored as
NOP                   ; program memory is
                      ; read in second cycle
                      ; after BSF EECON1,RD
BANKSEL EEDATA        ; Select Bank of EEDATA
MOVF    EEDATA, W     ; DATAL = EEDATA
MOVWF   DATAL         ;
MOVF    EEDATH, W     ; DATAH = EEDATH
MOVWF   DATAH         ;
```

## 3.6 Erasing FLASH Program Memory

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

When initiating an erase sequence from the microcontroller itself, a block of 32 words of program memory is erased. The Most Significant 11 bits of the EEADRH:EEADR point to the block being erased. EEADR< 4:0> are ignored.

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

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

After the "BSF EECON1,WR" instruction, the processor requires two cycles to setup the erase operation. The user must place two NOP instructions after the WR bit is set. The processor will halt internal operations for the typical 2 ms, only during the cycle in which the erase takes place. This is not SLEEP mode, as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the EECON1 write instruction.

### 3.6.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

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

1. Load EEADRH:EEADR with address of row being erased.
2. Set EEPGD bit to point to program memory, set WREN bit to enable writes, and set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write AAh to EECON2.
6. Set the WR bit. This will begin the row erase cycle.
7. The CPU will stall for duration of the erase.

**EXAMPLE 3-4: ERASING A FLASH PROGRAM MEMORY ROW**

```
          BANKSEL EEADRH          ; Select Bank of EEADRH
          MOVF    ADDRH, W        ;
          MOVWF   EEADRH          ; MS Byte of Program Address to Erase
          MOVF    ADDRL, W        ;
          MOVWF   EEADR           ; LS Byte of Program Address to Erase
ERASE_ROW
          BANKSEL EECON1          ; Select Bank of EECON1
          BSF     EECON1, EEPGD   ; Point to PROGRAM memory
          BSF     EECON1, WREN    ; Enable Write to memory
          BSF     EECON1, FREE    ; Enable Row Erase operation
;
          BCF     INTCON, GIE     ; Disable interrupts (if using)
          MOVLW   55h             ;
          MOVWF   EECON2          ; Write 55h
          MOVLW   AAh             ;
          MOVWF   EECON2          ; Write AAh
          BSF     EECON1, WR      ; Start Erase (CPU stall)
          NOP                     ; Any instructions here are ignored as processor
                                  ; halts to begin Erase sequence
          NOP                     ; processor will stop here and wait for Erase complete
                                  ; after Erase processor continues with 3rd instruction
          BCF     EECON1, WREN    ; Disable writes
          BSF     INTCON, GIE     ; Enable interrupts (if using)
```

## 3.7 Writing to FLASH Program Memory

FLASH program memory may only be written to if the destination address is in a segment of memory that is not write protected, as defined in bits WRT1:WRT0 of the device configuration word (Register 15-1). FLASH program memory must be written in four-word blocks. A block consists of four words with sequential addresses, with a lower boundary defined by an address, where EEADR<1:0> = 00. At the same time, all block writes to program memory are done as write only operations. The program memory must first be erased. The write operation is edge-aligned, and cannot occur across boundaries.

To write to the program memory, the data must first be loaded into the buffer registers. There are four 14-bit buffer registers and they are addressed by the low 2 bits of EEADR.

Loading data into the buffer registers is accomplished via the EEADR, EEADT, EECON1 and EECON2 registers as follows:

- Set EECON1 PGD, and WREN
- Write address to EEADRH:EEADR
- Write data to EEDATA:EEDATH
- Write 55, AA to EECON2
- Set WR bit in EECON1

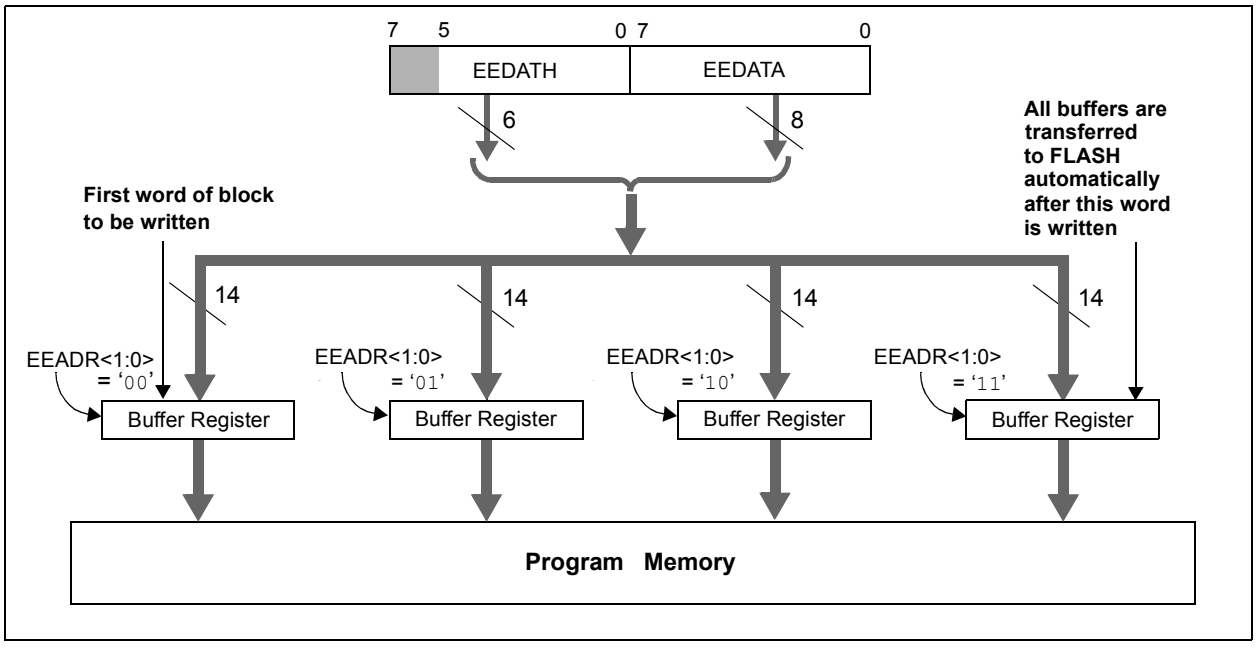
There are 4 buffer register words and all four locations **MUST** be written to with correct data.

After the "BSF EECON1,WR" instruction, if EEADR = xxxxxx11, then a short write will occur. This short write only transfers the data to the buffer register. The WR bit will be cleared in hardware after 1 cycle. The core will not halt and there will be no EEWHLT signal generated.

After the "BSF EECON1,WR" instruction, if EEADR = xxxxxx11, then a long write will occur. This will simultaneously transfer the data from EEDATH:EEDATA to the buffer registers and begin the write of all four words. The processor will execute the next instruction and then ignore the subsequent instruction. The user should place NOP instructions into the second words. The processor will then halt internal operations for typically 2 msec in which the write takes place. This is not a SLEEP mode, as the clocks and peripherals will continue to run. After the write cycle, the processor will resume operation with the 3rd instruction after the EECON1 write instruction.

After each long write, the 4 buffer registers will be reset to 3FFF.

**FIGURE 3-1: BLOCK WRITES TO FLASH PROGRAM MEMORY**



An example of the complete four-word write sequence is shown in Example 3-5. The initial address is loaded into the EEADRH:EEADR register pair; the four words of data are loaded using indirect addressing, assuming that a row erase sequence has already been performed.

**EXAMPLE 3-5: WRITING TO FLASH PROGRAM MEMORY**

```
; This write routine assumes the following:

; 1. The 32 words in the erase block have already been erased.
; 2. A valid starting address (the least significant bits = '00') is loaded into EEADRH:EEADR
; 3. This example is starting at 0x100, this is an application dependent setting.
; 4. The 8 bytes (4 words) of data are loaded, starting at an address in RAM called ARRAY.
; 5. This is an example only, location of data to program is application dependent.
; 6. word_block is located in data memory.

        BANKSEL  EECON1           ;prepare for WRITE procedure
        BSF      EECON1,EEPGD     ;point to program memory
        BSF      EECON1,WREN      ;allow write cycles

        BANKSEL  word_block
        MOVLW    .4
        MOVWF    word_block       ;prepare for 4 words to be written

        BANKSEL  EEADRH           ;Start writing at 0x100
        MOVLW    0x01
        MOVWF    EEADRH           ;load HIGH address
        MOVLW    0x00
        MOVWF    EEADR            ;load LOW address
        BANKSEL  ARRAY
        MOVLW    ARRAY            ;initialize FSR to start of data
        MOVWF    FSR
LOOP
        BANKSEL  EEDATA
        MOVF     INDF,W           ;indirectly load EEDATA
        MOVWF    EEDATA
        INCF     FSR,F            ;increment data pointer
        MOVF     INDF,W           ;indirectly load EEDATH
        MOVWF    EEDATH
        INCF     FSR,F            ;increment data pointer

        BANKSEL  EECON1
        MOVLW    0x55             ;required sequence
        MOVWF    EECON2
        MOVLW    0xAA
        MOVWF    EECON2
        BSF      EECON1,WR        ;set WR bit to begin write
        NOP                       ;instructions here are ignored as processor
        NOP

        BANKSEL  EEADR
        INCF     EEADR,f          ;load next word address
        BANKSEL  word_block
        DECFSZ   word_block,f     ;have 4 words been written?
        GOTO     loop             ;NO, continue with writing

        BANKSEL  EECON1
        BCF      EECON1,WREN      ;YES, 4 words complete, disable writes
        BSF      INTCON,GIE       ;enable interrupts
```

Required Sequence

## 3.8 Protection Against Spurious Write

There are conditions when the device should not write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, WREN is cleared. Also, the Power-up Timer (72 ms duration) prevents an EEPROM write.

The write initiate sequence and the WREN bit together, help prevent an accidental write during brown-out, power glitch, or software malfunction.

## 3.9 Operation During Code Protect

When the data EEPROM is code protected, the microcontroller can read and write to the EEPROM normally. However, all external access to the EEPROM is disabled. External write access to the program memory is also disabled.

When program memory is code protected, the microcontroller can read and write to program memory normally, as well as execute instructions. Writes by the device may be selectively inhibited to regions of the memory, depending on the setting of bits WRT1:WRT0 of the configuration word (see Section 15.1 for additional information). External access to the memory is also disabled.

**TABLE 3-1: REGISTERS/BITS ASSOCIATED WITH DATA EEPROM AND FLASH PROGRAM MEMORIES**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10Ch | EEDATA | EEPROM/FLASH Data Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Dh | EEADR | EEPROM/FLASH Address Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Eh | EEDATH | — | — | EEPROM/FLASH Data Register High Byte | | | | | | --xx xxxx | --uu uuuu |
| 10Fh | EEADRH | — | — | — | — | — | EEPROM/FLASH Address Register High Byte | | | ---- -xxx | ---- -uuu |
| 18Ch | EECON1 | EEPGD | — | — | FREE | WRERR | WREN | WR | RD | x--x x000 | x--x q000 |
| 18Dh | EECON2 | EEPROM Control Register2 (not a physical register) | | | | | | | | ---- ---- | ---- ---- |
| 0Dh | PIR2 | OSFIF | CMIF | — | EEIF | — | — | — | — | 00-0 ---- | 00-0 ---- |
| 8Dh | PIE2 | OSFIE | CMIE | — | EEIE | — | — | — | — | 00-0 ---- | 00-0 ---- |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends upon condition.
Shaded cells are not used by Data EEPROM or FLASH Program Memory.

# 4.0 OSCILLATOR CONFIGURATIONS

## 4.1 Oscillator Types

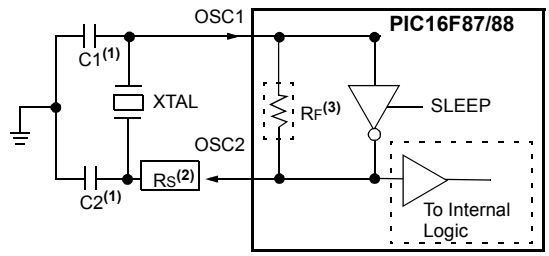
The PIC16F87/88 can be operated in eight different Oscillator modes. The user can program three configuration bits (FOSC2:FOSC0) to select one of these eight modes (modes 5 - 8 are new PIC16 oscillator configurations):

1. LP Low Power Crystal
2. XT Crystal/Resonator
3. HS High Speed Crystal/Resonator
4. RC External Resistor/Capacitor with FOSC/4 output on RA6
5. RCIO External Resistor/Capacitor with I/O on RA6
6. INTIO1 Internal Oscillator with FOSC/4 output on RA6 and I/O on RA7
7. INTIO2 Internal Oscillator with I/O on RA6 and RA7
8. ECIO External Clock with I/O on RA6

## 4.2 Crystal Oscillator/Ceramic Resonators

In XT, LP or HS modes, a crystal or ceramic resonator is connected to the OSC1/CLKI and OSC2/CLKO pins to establish oscillation (see Figure 4-1 and Figure 4-2). The PIC16F87/88 oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications.

**FIGURE 4-1: CRYSTAL OPERATION (HS, XT, OR LP OSC CONFIGURATION)**




**Note 1:** See Table 4-1 for typical values of C1 and C2.

**2:** A series resistor (RS) may be required for AT strip cut crystals.

**3:** RF varies with the crystal chosen (typically between 2 MΩ to 10 MΩ).

**TABLE 4-1: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR (FOR DESIGN GUIDANCE ONLY)**

| Osc Type | Crystal Freq | Typical Capacitor Values Tested: | |
|---|---|---|---|
| | | C1 | C2 |
| LP | 32 kHz | 33 pF | 33 pF |
| | 200 kHz | 15 pF | 15 pF |
| XT | 200 kHz | 56 pF | 56 pF |
| | 1 MHz | 15 pF | 15 pF |
| | 4 MHz | 15 pF | 15 pF |
| HS | 4 MHz | 15 pF | 15 pF |
| | 8 MHz | 15 pF | 15 pF |
| | 20 MHz | 15 pF | 15 pF |

**Capacitor values are for design guidance only.**

These capacitors were tested with the crystals listed below for basic start-up and operation. These values were not optimized.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

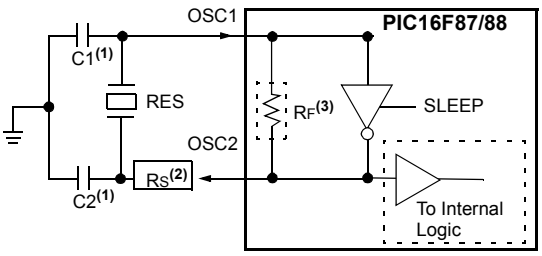
**Note 1:** Higher capacitance increases the stability of oscillator, but also increases the start-up time.

**2:** Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

**3:** Rs may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification.

**4:** Always verify oscillator performance over the VDD and temperature range that is expected for the application.

**FIGURE 4-2: CERAMIC RESONATOR OPERATION (HS OR XT OSC CONFIGURATION)**




**Note 1:** See Table 4-2 for typical values of C1 and C2.

**2:** A series resistor ($R_S$) may be required.

**3:** $R_F$ varies with the resonator chosen (typically between 2 MΩ to 10 MΩ).

**TABLE 4-2: CERAMIC RESONATORS (FOR DESIGN GUIDANCE ONLY)**

| Typical Capacitor Values Used: | | | |
|---|---|---|---|
| **Mode** | **Freq** | **OSC1** | **OSC2** |
| XT | 455 kHz<br>2.0 MHz<br>4.0 MHz | 56 pF<br>47 pF<br>33 pF | 56 pF<br>47 pF<br>33 pF |
| HS | 8.0 MHz<br>16.0 MHz | 27 pF<br>22 pF | 27 pF<br>22 pF |

**Capacitor values are for design guidance only.**

These capacitors were tested with the resonators listed below for basic start-up and operation. These values were not optimized.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected $V_{DD}$ and temperature range for the application.

See the notes following this table for additional information.

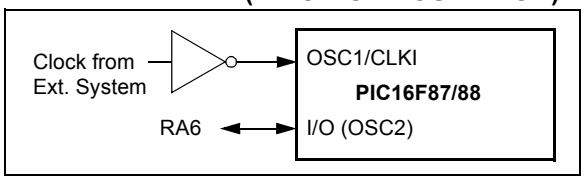
**Note:** When using resonators with frequencies above 3.5 MHz, the use of HS mode, rather than XT mode, is recommended. HS mode may be used at any $V_{DD}$ for which the controller is rated. If HS is selected, it is possible that the gain of the oscillator will overdrive the resonator. Therefore, a series resistor should be placed between the OSC2 pin and the resonator. As a good starting point, the recommended value of $R_S$ is 330Ω.

## 4.3 External Clock Input

The ECIO Oscillator mode requires an external clock source to be connected to the OSC1 pin. There is no oscillator start-up time required after a Power-on Reset, or after an exit from SLEEP mode.

In the ECIO Oscillator mode, the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 4-3 shows the pin connections for the ECIO Oscillator mode.

**FIGURE 4-3: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)**



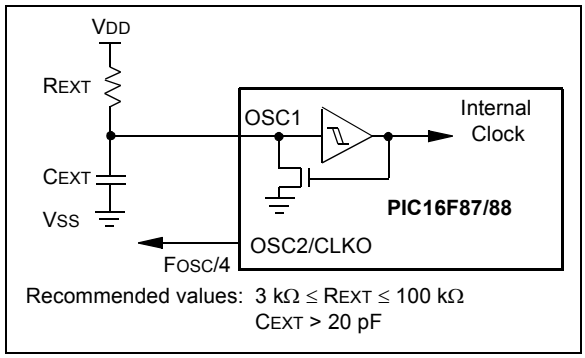

 

## 4.4 RC Oscillator

For timing insensitive applications, the "RC" and "RCIO" device options offer additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values, and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal manufacturing variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 4-4 shows how the R/C combination is connected.

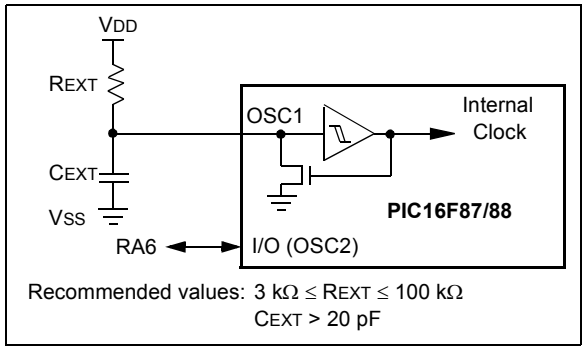
In the RC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic.

**FIGURE 4-4: RC OSCILLATOR MODE**




The RCIO Oscillator mode (Figure 4-5) functions like the RC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6).

**FIGURE 4-5: RCIO OSCILLATOR MODE**




## 4.5 Internal Oscillator Block

The PIC16F87/88 devices include an internal oscillator block, which generates two different clock signals; either can be used as the system's clock source. This can eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

The main output (INTOSC) is an 8 MHz clock source, which can be used to directly drive the system clock. It also drives the INTOSC postscaler, which can provide a range of six clock frequencies from 125 kHz to 4 MHz.

The other clock source is the internal RC oscillator (INTRC), which provides a 31.25 kHz (32 μs nominal period) output. The INTRC oscillator is enabled by selecting the INTRC as the system clock source, or when any of the following are enabled:

- Power-up Timer
- Watchdog Timer
- Two-Speed Start-up
- Fail-Safe Clock Monitor

These features are discussed in greater detail in Section 15.0 ("Special Features of the CPU").

The clock source frequency (INTOSC direct, INTRC direct, or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (page 41).

| **Note:** | Throughout this data sheet, when referring *specifically* to a generic clock source, the term "INTRC" may also be used to refer to the Clock modes using the internal oscillator block. This is regardless of whether the actual frequency used is INTOSC (8 MHz), the INTOSC postscaler, or INTRC (31.25 kHz). |
|---|---|

### 4.5.1 INTRC MODES

Using the internal oscillator as the clock source can eliminate the need for up to two external oscillator pins, after which it can be used for digital I/O. Two distinct configurations are available:

- In INTIO1 mode, the OSC2 pin outputs FOSC/4, while OSC1 functions as RA7 for digital input and output.
- In INTIO2 mode, OSC1 functions as RA7 and OSC2 functions as RA6, both for digital input and output.

### 4.5.2 OSCTUNE REGISTER

The internal oscillator's output has been calibrated at the factory, but can be adjusted in the application. This is done by writing to the OSCTUNE register (Register 4-1). The tuning sensitivity is constant throughout the tuning range. See Section 18.0 ("Electrical Characteristics") for further details.

When the OSCTUNE register is modified, the INTRC frequency will begin shifting to the new frequency. The INTRC clock will reach the new frequency within 8 clock cycles (approximately 8 * 32 μs = 256 μs). Code execution continues during this shift. There is no indication that the shift has occurred. Operation of features that depend on the 31.25 kHz INTRC clock source frequency, such as the WDT and peripherals, will also be affected by the change in frequency.

**REGISTER 4-1: OSCTUNE: OSCILLATOR TUNING REGISTER**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | TUN5 | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 |
| bit 7 | | | | | | | bit 0 |

bit 7-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** Frequency Tuning bits

011111 = Maximum frequency
011110 =
•
•
•
000001 =
000000 = Center frequency. Oscillator Module is running at the calibrated frequency.
111111 =
•
•
•
100000 = Minimum frequency

| 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 |

## 4.6 Clock Sources and Oscillator Switching

The PIC16F87/88 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low frequency clock source. PIC16F87/88 devices offer three alternate clock sources. When enabled, these give additional options for switching to the various Power Managed Operating modes.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator block (INTRC)

The **primary oscillators** include the external Crystal and Resonator modes, the external RC modes, the external Clock mode and the internal oscillator block. The particular mode is defined on POR by the contents of Configuration Word 1. The details of these modes are covered earlier in this chapter.

The **secondary oscillators** are those external sources not connected to the OSC1 or OSC2 pins. These sources may continue to operate even after the controller is placed in a Power Managed mode.

PIC16F87/88 devices offer only the Timer1 oscillator as a secondary oscillator. This oscillator continues to run when a SLEEP instruction is executed, and is often the time-base for functions, such as a real-time clock.

Most often, a 32.768 kHz watch crystal is connected between the RB6/T1OS0 and RB7/T1OSI pins. Like the LP mode oscillator circuit, loading capacitors are also connected from each pin to ground. The Timer1 oscillator is discussed in greater detail in Section 7.6.

In addition to being a primary clock source, the **internal oscillator block** is available as a Power Managed mode clock source. The 31.25 kHz INTRC source is also used as the clock source for several special features, such as the WDT, Fail-Safe Clock Monitor, Power-up Timer, and Two-Speed Start-up.

The clock sources for the PIC16F87/88 devices are shown in Figure 4-6. See Section 7.0 for further details of the Timer1 oscillator. See Section 15.1 for Configuration register details.

### 4.6.1 OSCILLATOR CONTROL REGISTER

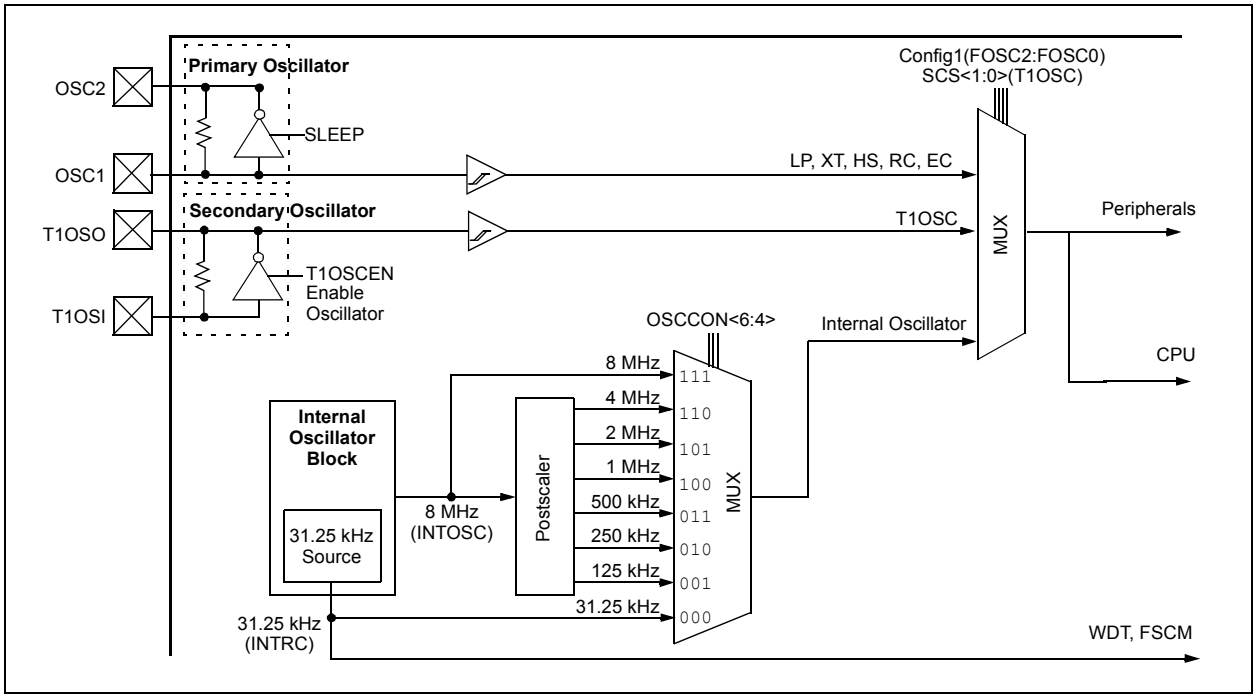
The OSCCON register (Register 4-2) controls several aspects of the system clock's operation, both in full power operation and in Power Managed modes.

The System Clock Select bits, SCS1:SCS0, select the clock source that is used when the device is operating in Power Managed modes. When the bits are cleared (= 00), the system clock source comes from the main oscillator that is selected by the FOSC2:FOSC0 configuration bits in Configuration Register 1. When the bits are set in any other manner, the system clock source is provided by the Timer1 oscillator (SCS1:SCS0 = 01), or from the internal oscillator block (SCS1:SCS0 = 10). After a RESET, SCS<1:0> are always set to '00'.

The Internal Oscillator Select bits, IRCF2:IRCF0, select the frequency output of the internal oscillator block that is used to drive the system clock. The choices are the INTRC source (31.25 kHz), the INTOSC source (8 MHz), or one of the six frequencies derived from the INTOSC postscaler (125 kHz to 4 MHz). Changing the configuration of these bits has an immediate change on the internal oscillator's output.

The OSTS and IOFS bits indicate the status of the primary oscillator and INTOSC source; these bits are set when their respective oscillators are stable. In particular, OSTS indicates that the Oscillator Start-up Timer has timed out.

**FIGURE 4-6: PIC16F87/88 CLOCK DIAGRAM**




### 4.6.2 MODIFYING THE IRCF BITS

The IRCF bits can be modified at any time, regardless of which clock source is currently being used as the system clock. The internal oscillator allows users to change the frequency during RUN time. This is achieved by modifying the IRCF bits in the OSCCON register. The sequence of events that occur after the IRCF bits are modified is dependent upon the initial value of the IRCF bits before they are modified. The system clock, in either case, will switch to the new internal oscillator frequency after **eight** falling edges of the new clock. If the INTRC (31.25 kHz) is running and the IRCF bits are modified to any of the other high frequency values, a 1 ms clock switch delay is turned on. Code execution continues at a higher than expected frequency while the new frequency stabilizes. Time sensitive code should wait for the IOFS bit in the OSCCON register to become set before continuing. This bit can be monitored to ensure that the frequency is stable before using the system clock in time critical applications.

If the IRCF bits are modified while the internal oscillator is running at any other frequency than INTRC (31.25 kHz), there is no need for a 1 ms clock switch delay. The new INTOSC frequency will be stable immediately after the **eight** falling edges. The IOFS bit will remain set after clock switching occurs.

Caution must be taken when modifying the IRCF bits using `BCF` or `BSF` instructions. It is possible to modify the IRCF bits to a frequency that may be out of the $V_{DD}$ specification range; for example, $V_{DD}$ = 2.0V and IRCF = `111` (8 MHz).

### 4.6.3 CLOCK TRANSITION SEQUENCE WHEN THE IRCF BITS ARE MODIFIED

The following sequence is performed when the IRCF bits are changed and the system clock is the internal oscillator.

1. The IRCF bits are modified.
2. The clock switching circuitry waits for a falling edge of the current clock, at which point CLKO is held low.
3. The clock switching circuitry then waits for **eight** falling edges of requested clock, after which it switches CLKO to this new clock source.
4. If the INTRC (31.25 kHz) is enabled, the IOFS bit is clear to indicate that the clock is unstable and a 1 ms delay is started. If the internal oscillator frequency is anything other than INTRC (31.25 kHz), this step is skipped. After the appropriate number of clock periods have passed, the IOFS bit is set to indicate to the internal oscillator that the frequency is stable.
5. Oscillator switch over is complete.

**REGISTER 4-2: OSCCON: OSCILLATOR CONTROL REGISTER**

| U-0 | R/W-0 | R/W-0 | R/W0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| — | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 6-4 **IRCF<2:0>:** Internal RC Oscillator Frequency Select bits

000 = 31.25 kHz
001 = 125 kHz
010 = 250 kHz
011 = 500 kHz
100 = 1 MHz
101 = 2 MHz
110 = 4 MHz
111 = 8 MHz

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit

1 = Device is running from the primary system clock
0 = Device is running from T1OSC or INTRC as a secondary system clock

bit 2 **IOFS:** INTOSC Frequency Stable bit

1 = Frequency is stable
0 = Frequency is not stable

bit 1-0 **SCS<1:0>:** Oscillator Mode Select bits

00 = Oscillator mode defined by FOSC<2:0>
01 = T1OSC is used for system clock
10 = Internal RC is used for system clock
11 = Reserved

| 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 |

### 4.6.4 OSCILLATOR DELAY UPON POWER-UP AND WAKE-UP

The Oscillator Start-up Timer (OST) is used to ensure that a stable system clock is provided to the device. The OST is activated following a POR, or a wake-up from SLEEP mode, when the system clock is configured for one of the primary oscillator modes (LP, XT, and HS).

Table 4-3 shows examples where the oscillator delay is invoked.

**TABLE 4-3: OSCILLATOR DELAY EXAMPLES**

| Switch From | Switch To | Frequency | Oscillator Delay | Comments |
|---|---|---|---|---|
| SLEEP | INTRC<br>T1OSC | 31.25 kHz<br>32.768 kHz | 5 μs - 10 μs (approx.)<br>CPU Start-up(1) | Following a wake-up from SLEEP mode or POR, CPU start-up is invoked to allow the CPU to become ready for code execution. |
| SLEEP | INTOSC | 125 kHz - 8 MHz | | |
| INTRC/SLEEP | EC, RC | 0 - 20 MHz | | |
| INTRC<br>(31.25 kHz) | EC, RC | 0 - 20 MHz | | |
| SLEEP | LP, XT, HS | 32.768 kHz - 20 MHz | 1024 Clock Cycles<br>(OST) | Following a change from INTRC, an OST of 1024 cycles must occur. |
| INTRC<br>(31.25 kHz) | INTOSC | 125 kHz - 8 MHz | 1 ms | Refer to Section 4.6.2 for further details. |

**Note 1:** The 5 μs - 10 μs start-up delay is based on a 1 MHz System Clock.

### 4.6.5 CLOCK SWITCHING

Clock switching will occur for the following reasons:

- The FCMEN bit is set, the device is running from the primary oscillator, and the primary oscillator fails.
- The FCMEN bit is set, the device is running from the T1OSC and T1OSC fails.
- Following a wake-up due to a RESET or a POR, when the device is configured for Two-Speed mode, switching will occur between the INTRC and the system clock defined by the FOSC<2:0> bits.
- A wake-up from SLEEP occurs due to interrupt or WDT wake-up and Two-Speed Start-up is enabled. If the primary clock is XT, HS, or LP, the clock will switch between the INTRC and the primary system clock after 1024 clock (OST) and 8 clocks of the primary oscillator. This is conditional upon the SCS bits being set equal to '00'.

| | |
|---|---|
| **Note:** | Because the SCS bits are cleared on any RESET, no clock switching will occur on a RESET unless the Two-Speed Start-up is enabled and the primary clock is XT, HS, or LP. The device will wait for the primary clock to become stable before execution begins (Two-Speed Start-up disabled). |

### 4.6.6 CLOCK TRANSITION DELAYS

When a clock transition is requested, the CLKO signal will continue to provide the current clock at its output throughout the transition period. After this transition period, the requested clock will start to drive the CLKO signal. The transition delay comprises the time to detect clock source stability plus eight cycles (of the new clock). For internal RC oscillators, the transition delay is eight clocks. When the Primary oscillator is configured for any oscillator (LP, XT, or HS), the transition delay is 1024 plus eight clocks. When the primary oscillator is configured for an external clock, the transition delay is eight clocks.

If an attempt is made to switch to the same clock source already in use, the clock transition sequence will not take place.

### 4.6.7 CLOCK TRANSITION AND THE WATCHDOG

When clock switching is performed, the Watchdog Timer is disabled because the Watchdog Ripple Counter is used as the Oscillator Start-up Timer.

| | |
|---|---|
| **Note:** | The OST is only used when switching to XT, HS, and LP Oscillator modes. |

Once the clock transition is complete (i.e., new oscillator selection switch has occurred), the Watchdog Counter is re-enabled with the Counter Reset. This allows the user to synchronize the Watchdog Timer to the start of execution at the new clock frequency.

## 4.7 Power Managed Modes

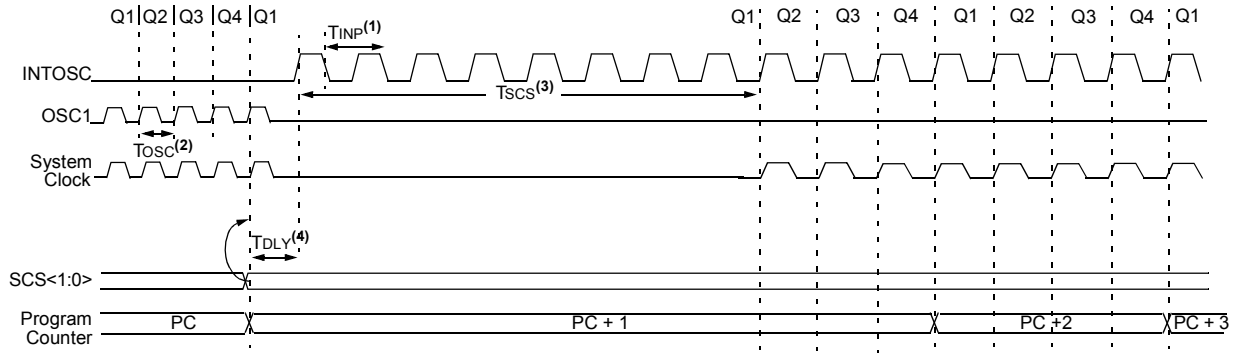
### 4.7.1 RC_RUN MODE

When SCS bits are configured to run from the INTRC, a clock transition is generated if the system clock is not already using the INTRC. The event will clear the OSTS bit, switch the system clock from the primary system clock (if SCS<1:0> = 00) determined by the value contained in the configuration bits, or from the T1OSC (if SCS<1:0> = 01) to the INTRC clock option, and shut down the primary system clock to conserve power. Clock switching will not occur if the primary system clock is already configured as INTRC.

If the system clock does not come from the INTRC (31.25 kHz) when the SCS bits are changed, and the IRCF bits in the OSCCON register are configured for a frequency other than INTRC, the frequency may not be stable immediately. The IOFS bit (OSCCON<2>) will be set when the INTOSC or postscaler frequency is stable, after approximately 1 ms.

After a clock switch has been executed, the OSTS bit is cleared, indicating a Low Power mode, and the device does not run from the primary system clock. The internal Q clocks are held in the Q1 state until eight falling edge clocks are counted on the INTRC oscillator. After the eight clock periods have transpired, the clock input to the Q clocks is released and operation resumes (see Figure 4-7).

**FIGURE 4-7: TIMING DIAGRAM FOR XT, HS, LP, EC AND EXTRC TO RC_RUN MODE**




**Note 1:** TINP = 32 μs typical.
**2:** TOSC = 50 ns minimum.
**3:** TSCS = 8 TINP.
**4:** TDLY = 1 TINP.

### 4.7.2 SEC_RUN MODE

The core and peripherals can be configured to be clocked by T1OSC using a 32.768 kHz crystal. The crystal must be connected to the T1OSO and T1OSI pins. This is the same configuration as the low power timer circuit (see Section 7.6). When SCS bits are configured to run from T1OSC, a clock transition is generated. It will clear the OSTS bit, switch the system clock from either the primary system clock, or INTRC, depending on the value of SCS<1:0> and FOSC<2:0>, to the external low power Timer1 oscillator input (T1OSC), and shut down the primary system clock to conserve power.

After a clock switch has been executed, the internal Q clocks are held in the Q1 state until eight falling edge clocks are counted on the T1OSC. After the eight clock periods have transpired, the clock input to the Q clocks is released and operation resumes (see Figure 4-8). In addition, T1RUN (In T1CON) is set to indicate that T1OSC is being used as the system clock.

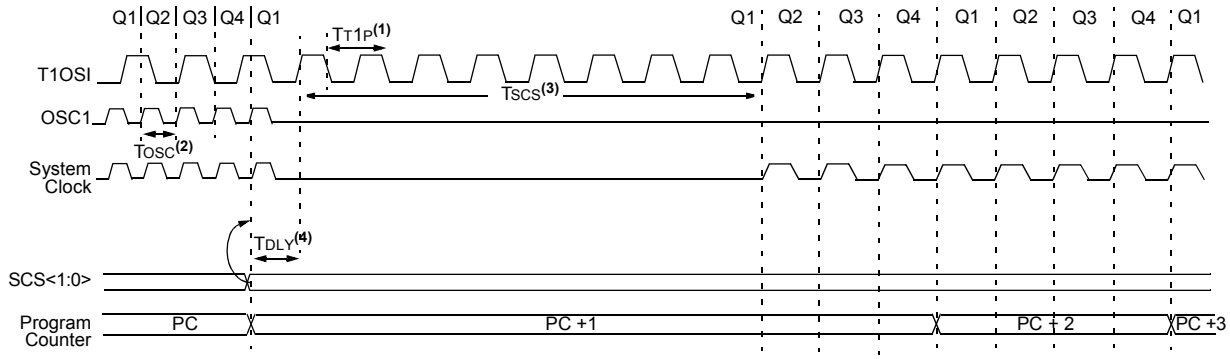
**Note 1:** The T1OSCEN bit must be enabled and it is the user's responsibility to ensure T1OSC is stable before clock switching to the T1OSC input clock can occur.

**2:** When T1OSCEN = 0, the following possible effects result.

| Original SCS<1:0> | Modified SCS<1:0> | Final SCS<1:0> |
|---|---|---|
| 00 | 01 | 00 - no change |
| 00 | 11 | 10 - INTRC |
| 10 | 11 | 10 - no change |
| 10 | 01 | 00 - OSC defined by FOSC<2:0> |

A clock switching event will occur if the final state of the SCS bits is different from the original.

**FIGURE 4-8: TIMING DIAGRAM FOR SWITCHING TO SEC_RUN MODE**




**Note 1:** TT1P = 30.52 μs.
**2:** TOSC = 50 ns minimum.
**3:** TSCS = 8 TT1P
**4:** TDLY = 1 TT1P.

### 4.7.3 SEC_RUN/RC_RUN TO PRIMARY CLOCK SOURCE

When switching from a SEC_RUN or RC_RUN mode back to the primary system clock, following a change of SCS<1:0> to '00', the sequence of events that take place will depend upon the value of the FOSC bits in the Configuration register. If the external oscillator is configured as a crystal (HS, XT, or LP), then the transition will take place after 1024 clock cycles. This is necessary because the crystal oscillator had been powered down until the time of the transition. In order to provide the system with a reliable clock when the changeover has occurred, the clock will not be released to the changeover circuit until the 1024 count has expired.

During the Oscillator Start-up Time, the system clock comes from the current system clock. Instruction execution and/or peripheral operation continues using the currently selected oscillator as the CPU clock source, until the necessary clock count has expired to ensure that the primary system clock is stable.

**Note 1:** When the device is configured to use T1OSC, the act of clearing the T1OSCEN bit in the T1CON register will cause SCS<0> to be cleared, which causes the SCS<1:0> bits to revert to '00' or '10', depending on what SCS<1> is. The T1OSCEN bit will be cleared immediately; however, T1OSC will be enabled and instruction execution will continue until the OST time-out for the main system clock is complete. At that time, the system clock will switch from the T1OSC to the primary clock or the INTRC. Following this, the T1 oscillator will be shut down.

**2:** If it is desired not to run time critical application code while running from the secondary clock source, the OSTS bit should be monitored until the Oscillator Start-up Timer has completed. OSTS = 1 indicates that the Oscillator Start-up Timer has timed out and the system clock comes from the primary clock source.

Following the Oscillator Start-up Time, the internal Q clocks are held in the Q1 state until eight falling edge clocks are counted from the primary system clock. The clock input to the Q clocks is then released, and operation resumes with primary system clock determined by the FOSC bits (see Figure 4-10).

**Note:** If the primary system clock is either RC or EC, an internal delay timer (5 - 10 μs) will suspend operation after exiting Secondary Clock mode to allow the CPU to become ready for code execution.

#### 4.7.3.1 Returning to Primary Clock Source Sequence

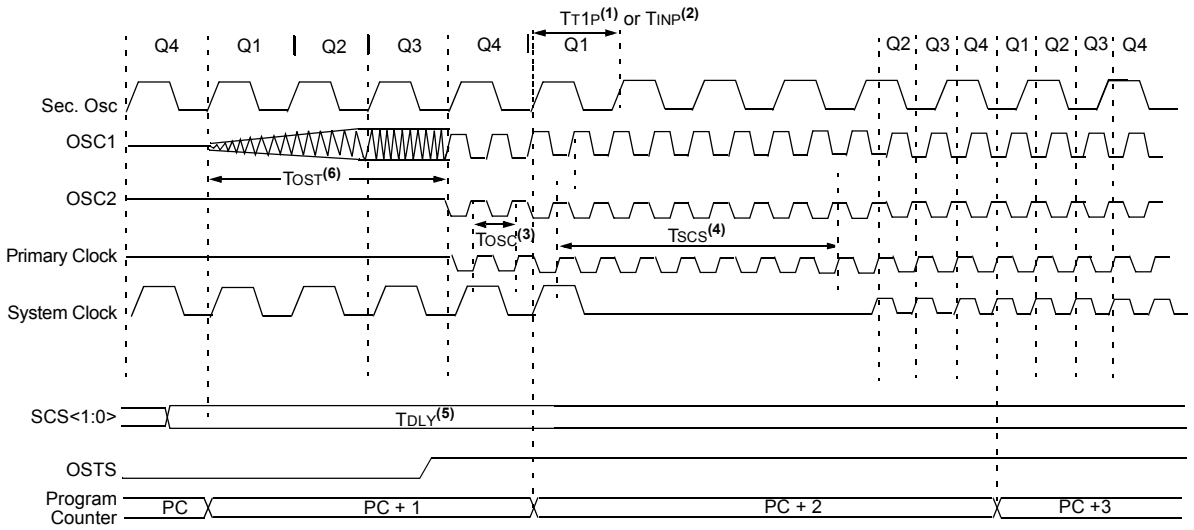
Changing back to the primary oscillator from SEC_RUN or RC_RUN can be accomplished by either changing SCS<1:0> to '00', or clearing the T1OSCEN bit in the T1CON register (if T1OSC was the secondary clock).

The sequence of events that follows is the same for both modes:

1. If the primary system clock is configured as EC, RC, or INTRC, then the OST time-out is skipped. Skip to step 3.
2. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active, waiting for 1024 clocks of the primary system clock.
3. On the following Q1, the device holds the system clock in Q1.
4. The device stays in Q1 while eight falling edges of the primary system clock are counted.
5. Once the eight counts transpire, the device begins to run from the primary oscillator.
6. If the secondary clock was INTRC and the primary is not INTRC, the INTRC will be shut down to save current, providing that the INTRC is not being used for any other function, such as WDT, or Fail-Safe Clock Monitoring.
7. If the secondary clock was T1OSC, the T1OSC will continue to run if T1OSCEN is still set, otherwise the T1 oscillator will be shut down.

**FIGURE 4-9: TIMING FOR TRANSITION BETWEEN SEC_RUN/RC_RUN AND PRIMARY CLOCK**




**Note 1:** TT1P = 30.52 μs.
**2:** TINP = 32 μs typical.
**3:** TOSC = 50 ns minimum.
**4:** TSCS = 8 TINP OR 8 TT1P.
**5:** TDLY = 1 TINP OR 1 TT1P.
**6:** Refer to parameter “D033” in Section 18.0.

#### 4.7.3.2 Returning to Primary Oscillator with a RESET

A RESET will clear SCS<1:0> back to '00'. The sequence for starting the primary oscillator following a RESET is the same for all forms of RESET, including POR. There is no transition sequence from the secondary system clock to the primary system clock. Instead, the device will reset the state of the OSCCON register and default to the primary system clock. The sequence of events that take place after this will depend upon the value of the FOSC bits in the Configuration register. If the external oscillator is configured as a crystal (HS, XT, or LP), the CPU will be held in the Q1 state until 1024 clock cycles have transpired on the primary clock. This is necessary because the crystal oscillator had been powered down until the time of the transition.

During the Oscillator Start-up Time, the system clock does not come from the low power oscillator. Instruction execution and/or peripheral operation is suspended and the secondary, low power, oscillator is disabled.

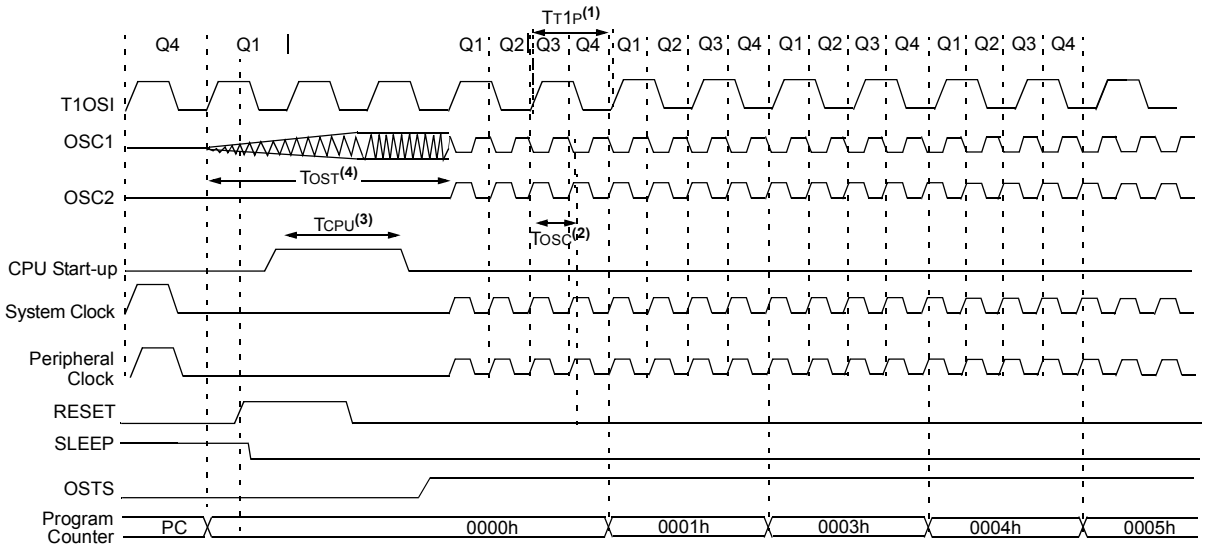
| **Note:** | If Two-Speed Clock Start-up mode is enabled, the INTRC will act as the system clock until the OST timer has timed out. |
|---|---|

If the primary system clock is either RC, EC, or INTRC, the CPU will begin operating on the first Q1 cycle following the wake-up event. This means that there is no Oscillator Start-up Time required because the primary clock is already stable; however, there is a delay between the wake-up event and the following Q2. An internal delay timer of 5 - 10 μs will suspend operation after the RESET to allow the CPU to become ready for code execution. The CPU and peripheral clock will be held in the first Q1 following the exit from low power. The clocks will be released on the next falling edge of the input system clock. The CPU will advance the system clock into the Q2 state following two rising edges of the incoming clock on OSC1. The extra clock transition is required following a RESET to allow the system clock to synchronize to the asynchronous nature of the RESET source (see Figure 4-11).

The sequence of events is as follows:

1. A device RESET is asserted from one of many sources (WDT, BOR, $\overline{MCLR}$, etc.).
2. The device resets and the CPU start-up timer is enabled if in SLEEP mode. The device is held in RESET until the CPU start-up time-out is complete.
3. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active waiting for 1024 clocks of the primary system clock. While waiting for the OST, the device will be held in RESET. The OST and CPU start-up timers run in parallel.
4. After both the CPU start-up and OST timers have timed out, the device will wait for one additional clock cycle and instruction execution will begin.

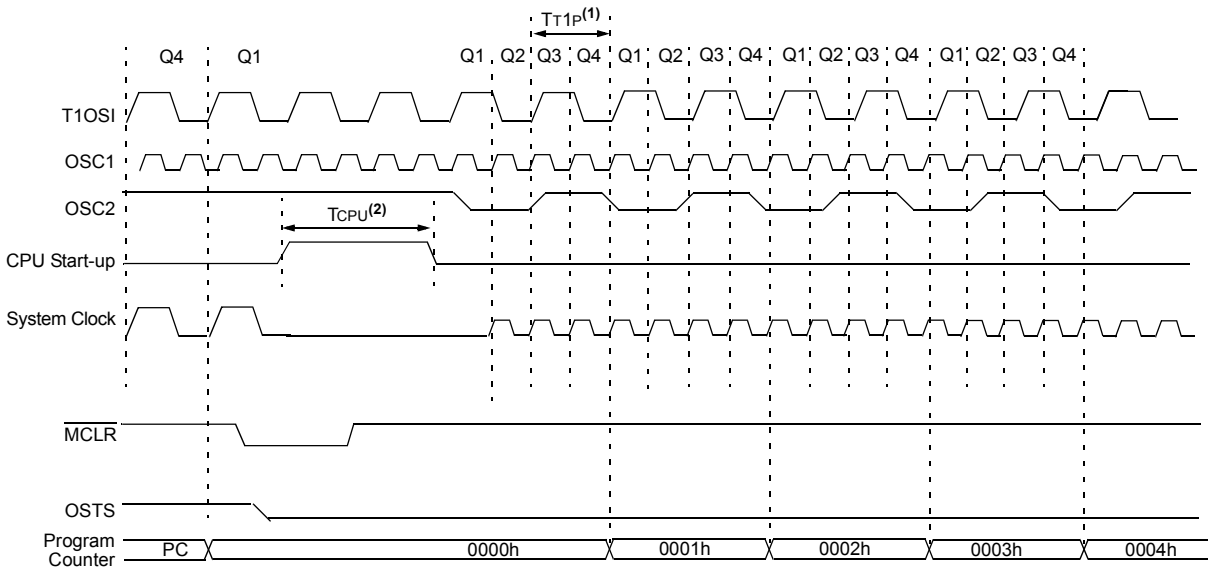
**FIGURE 4-10: TIMING LP CLOCK TO PRIMARY SYSTEM CLOCK AFTER RESET (HS, XT, LP)**




**Note 1:** TT1P = 30.52 μs.
**2:** TOSC = 50 ns minimum.
**3:** TCPU = 5 - 10 μs (1 MHz System Clock).
**4:** Refer to parameter "D033" in Section 18.0.

**FIGURE 4-11: TIMING LP CLOCK TO PRIMARY SYSTEM CLOCK AFTER RESET (EC, RC, INTRC)**




**Note 1:** TT1P = 30.52 μs.
**2:** TCPU = 5 - 10 μs (1 MHz System Clock).

**TABLE 4-4: CLOCK SWITCHING MODES**

| Current System Clock | SCS bits <1:0> Modified to: | OSTS bit | Delay | IOFS bit | T1RUN bit | New System Clock | Comments |
|---|---|---|---|---|---|---|---|
| LP, XT, HS, T1OSC, EC, RC | 10 (INTRC) | 0 | 8 Clocks of INTRC | 1 | 0 | INTRC or INTOSC or INTOSC Postscaler | The internal RC oscillator frequency is dependant upon the IRCF bits. |
| LP, XT, HS, INTRC, EC, RC | 01 (T1OSC) | 0 | 8 Clocks of T1OSC | N/A | 1 | T1OSC | T1OSCEN bit must be enabled. |
| INTRC T1OSC | 00 FOSC<2:0> = EC or FOSC<2:0> = RC | 1 | 8 Clocks of EC or RC | N/A | 0 | EC or RC | |
| INTRC T1OSC | 00 FOSC<2:0> = LP, XT, HS | 0 | 1024 Clocks (OST) + 8 Clocks of LP, XT, HS | N/A | 0 | LP, XT, HS | During the 1024 clocks, program execution is clocked from the secondary oscillator until the primary oscillator becomes stable. |
| LP, XT, HS | 00 (Due to RESET) LP, XT, HS | 1 | 1024 Clocks (OST) | N/A | 0 | LP, XT, HS | When a RESET occurs, there is no clock transition sequence. Instruction execution and/or peripheral operation is suspended unless Two-Speed Start-up mode is enabled, after which the INTRC will act as the system clock until the OST timer has timed out. |

### 4.7.4 EXITING SLEEP WITH AN INTERRUPT

Any interrupt, such as WDT or INT0, will cause the part to leave the SLEEP mode.

The SCS bits are unaffected by a SLEEP command and are the same before and after entering and leaving SLEEP. The clock source used after an exit from SLEEP is determined by the SCS bits.

#### 4.7.4.1 Sequence of Events

**If SCS<1:0> = 00:**

1. The device is held in SLEEP until the CPU start-up time-out is complete.
2. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active waiting for 1024 clocks of the primary system clock. While waiting for the OST, the device will be held in SLEEP unless Two-Speed Start-up is enabled. The OST and CPU start-up timers run in parallel. Refer to Section 15.12.4 for details on Two-Speed Start-up.
3. After both the CPU start-up and OST timers have timed out, the device will exit SLEEP and begin instruction execution with the primary clock defined by the FOSC bits.

**If SCS<1:0> = 01 or 10:**

1. The device is held in SLEEP until the CPU start-up time-out is complete.
2. After the CPU start-up timer has timed out, the device will exit SLEEP and begin instruction execution with the selected Oscillator mode.

**Note:** If a user changes SCS<1:0> just before entering SLEEP mode, the system clock used when exiting SLEEP mode could be different than the system clock used when entering SLEEP mode.

As an example, if SCS<1:0> = 01 and T1OSC is the system clock, and the following instructions are executed:

```
BCF        OSCON,SCS0

SLEEP
```

then a clock change event is executed. If the primary oscillator is XS, LP, or HS, the core will continue to run off T1OSC and execute the SLEEP command.

When SLEEP is exited, the part will resume operation with the primary oscillator after the start-up.

**NOTES:**

## 5.0 I/O PORTS

Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Additional information on I/O ports may be found in the PICmicro™ Mid-Range Reference Manual (DS33023).

### 5.1 PORTA and the TRISA Register

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

| **Note:** | On a Power-on Reset, the pins PORTA<4:0> are configured as analog inputs and read as '0'. |
|---|---|

Reading the PORTA register, reads the status of the pins, whereas writing to it, will write to the port latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified, and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input and with analog input to become the RA4/AN4/T0CKI/C2OUT pin. The RA4/AN4/T0CKI/C2OUT pin is a Schmitt Trigger input and full CMOS output driver.

Pin RA5 is multiplexed with the Master Clear module input. The RA5/$\overline{MCLR}$/VPP pin is a Schmitt Trigger input.

Pin RA6 is multiplexed with the oscillator module input and external oscillator output. Pin RA7 is multiplexed with the oscillator module input and external oscillator input. Pin RA6/OSC2/CLKO and pin RA7/OSC1/CLKI are Schmitt Trigger inputs and full CMOS output drivers.

Pins RA<1:0> are multiplexed with analog inputs. Pins RA<3:2> are multiplexed with analog inputs, comparator outputs, and VREF inputs. Pins RA<3:0> have TTL inputs and full CMOS output drivers.

**EXAMPLE 5-1: INITIALIZING PORTA**

```
BANKSEL  PORTA     ; select bank of PORTA
CLRF     PORTA     ; Initialize PORTA by
                   ; clearing output
                   ; data latches
BANKSEL  ADCON1    ; Select Bank of ADCON1
MOVLW    0x06      ; Configure all pins
MOVWF    ADCON1    ; as digital inputs
MOVLW    0xFF      ; Value used to
                   ; initialize data
                   ; direction
MOVWF    TRISA     ; Set RA<7:0> as inputs
```

**TABLE 5-1: PORTA FUNCTIONS**

| Name | Bit# | Buffer | Function |
|---|---|---|---|
| RA0/AN0 | bit 0 | TTL | Input/output or analog input. |
| RA1/AN1 | bit 1 | TTL | Input/output or analog input. |
| RA2/AN2/CVREF/VREF-(2) | bit 2 | TTL | Input/output or analog input or VREF- or Comparator VREF output. |
| RA3/AN3/VREF+(2)/C1OUT | bit 3 | TTL | Input/output or analog input or VREF+ or Comparator output. |
| RA4/AN4(2)/T0CKI/C2OUT | bit 4 | ST | Input/output, analog input or TMR0 external input or Comparator output. |
| RA5/$\overline{MCLR}$/VPP | bit 5 | ST | Input, Master Clear (Reset) or Programming voltage input. |
| RA6/OSC2/CLKO | bit 6 | ST | Input/output, connects to Crystal or Resonator, Oscillator output or 1/4 the frequency of OSC1, and denotes the instruction cycle in RC mode. |
| RA7/OSC1/CLKI | bit 7 | ST/CMOS(1) | Input/output, connects to Crystal or Resonator or Oscillator input. |

Legend: TTL = TTL input, ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

**2:** PIC16F88 only.

**TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 05h | PORTA | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xxxx 0000(1)<br>xxx0 0000(2) | uuuu 0000(1)<br>uuu0 0000(2) |
| 85h | TRISA | TRISA7 | TRISA6 | TRISA5(3) | PORTA Data Direction Register | | | | | 1111 1111 | 1111 1111 |
| 9Fh | ADCON1 | ADFM | ADCS2 | VCFG1 | VCFG0 | — | — | — | — | 0000 ---- | 0000 ---- |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTA.

**Note 1:** This value applies only to the PIC16F87.

**2:** This value applies only to the PIC16F88.

**3:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

**FIGURE 5-1: BLOCK DIAGRAM OF RA0/AN0:RA1/AN1 PINS**

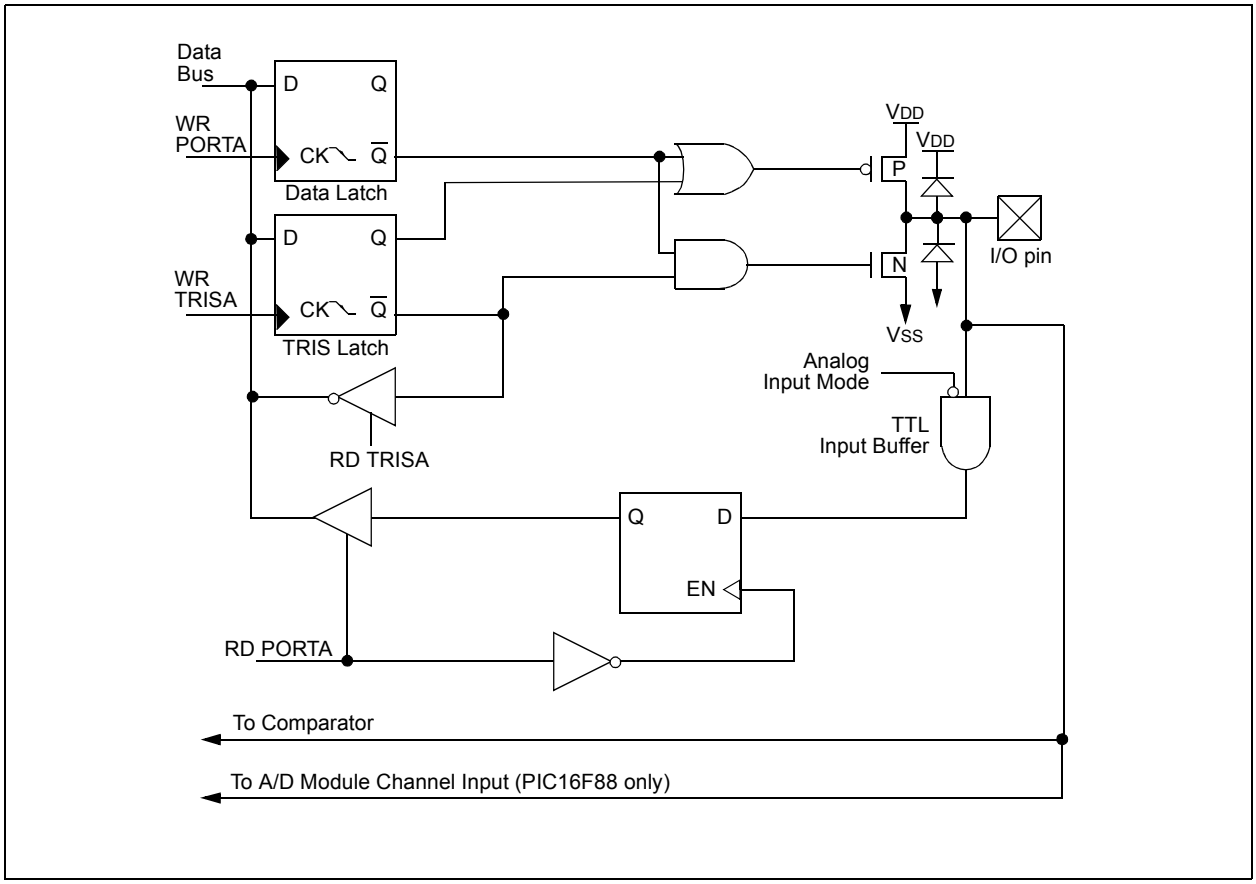


**FIGURE 5-2: BLOCK DIAGRAM OF RA3/AN3/VREF+/C1OUT PIN**

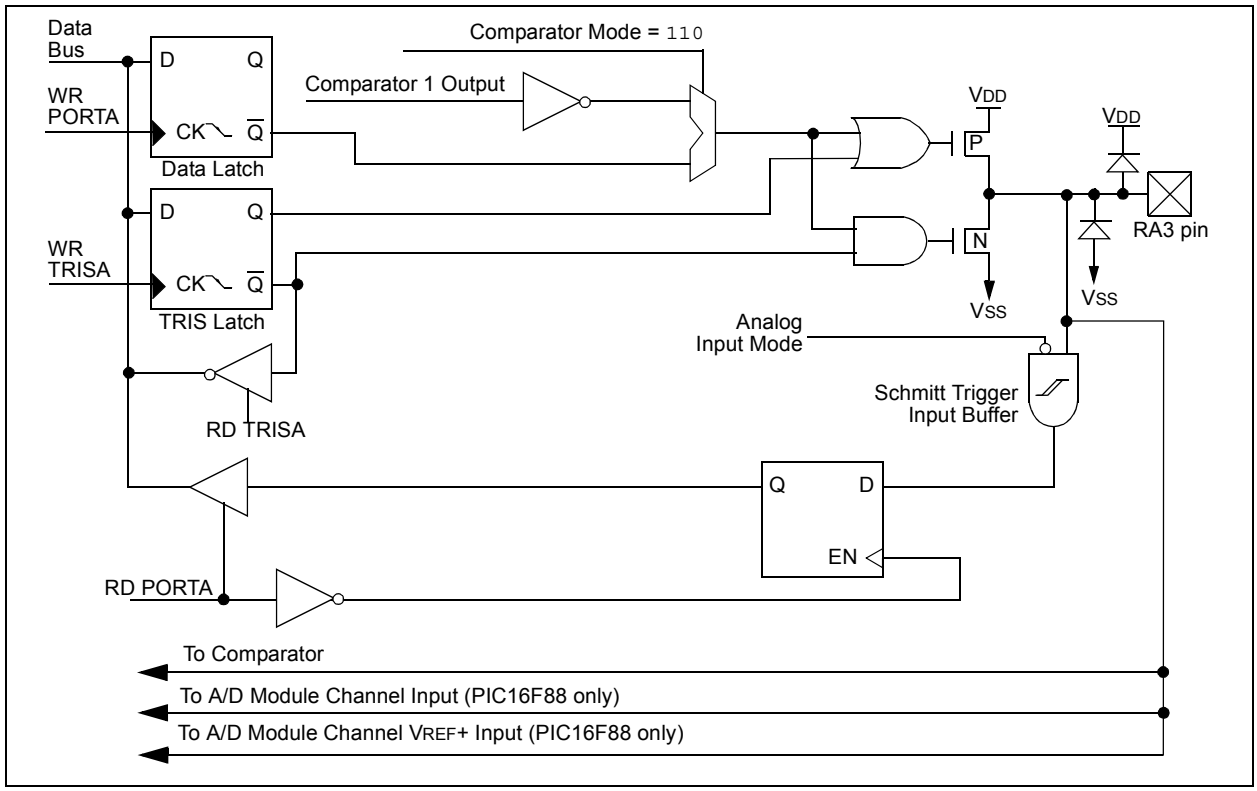



**FIGURE 5-3: BLOCK DIAGRAM OF RA2/AN2/CVREF/VREF- Pin**

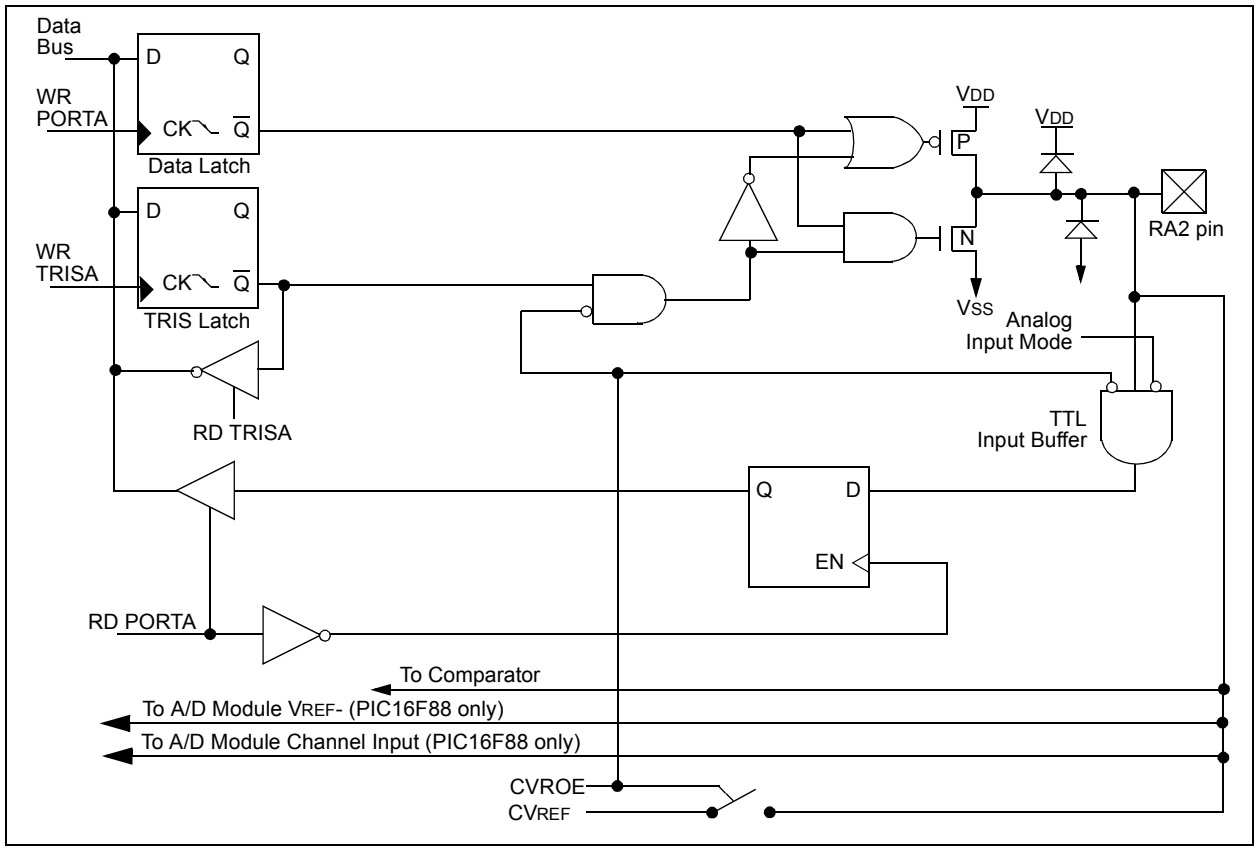


**FIGURE 5-4: BLOCK DIAGRAM OF RA4/T0CKI/C2OUT PIN**

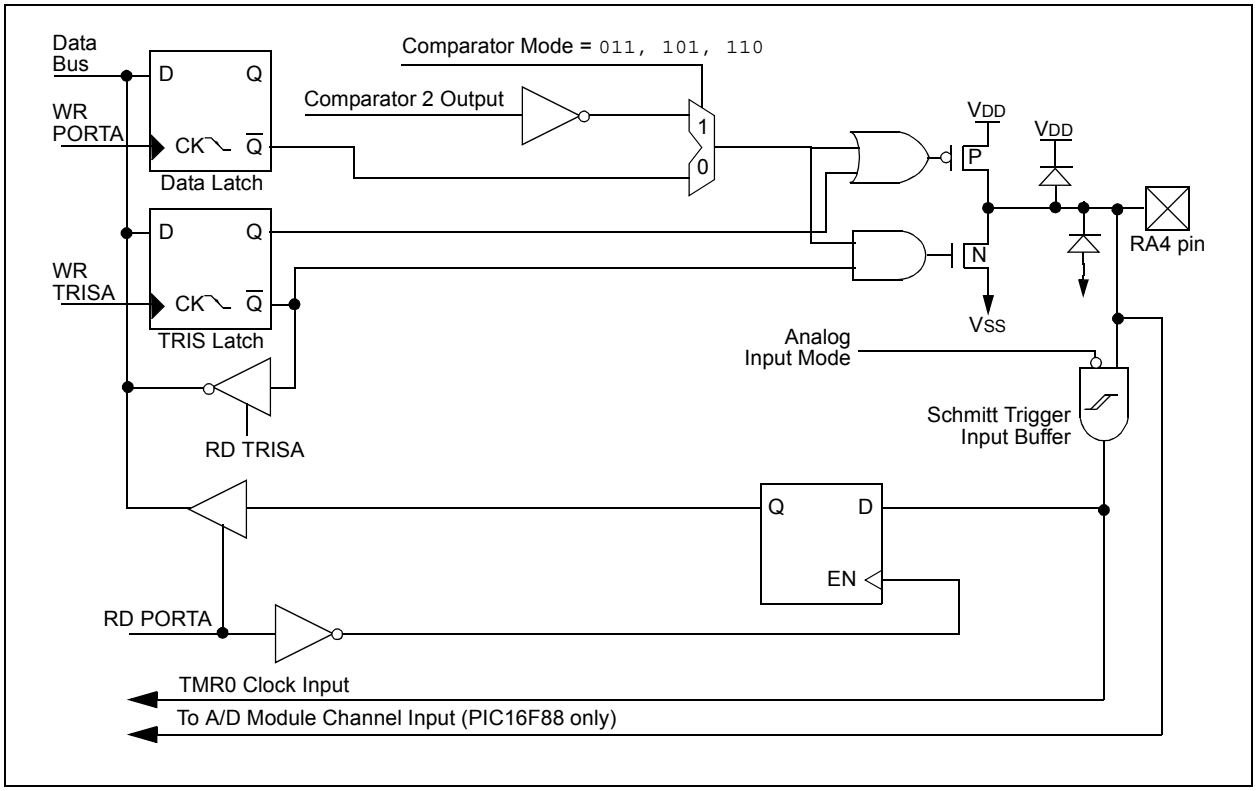



**FIGURE 5-5: BLOCK DIAGRAM OF RA5/$\overline{\text{MCLR}}$/VPP PIN**

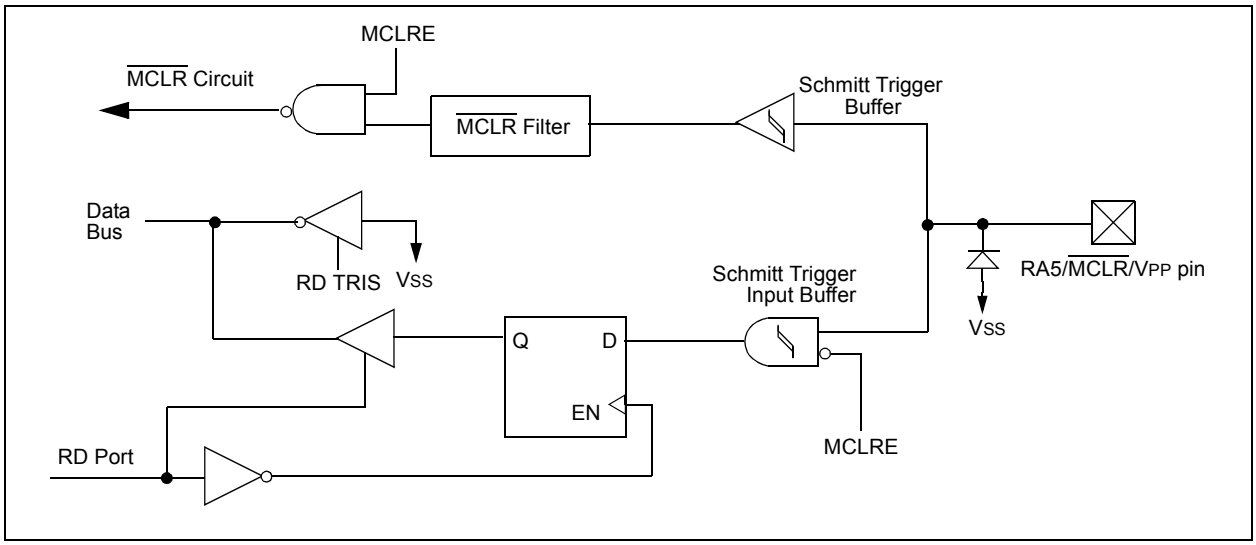
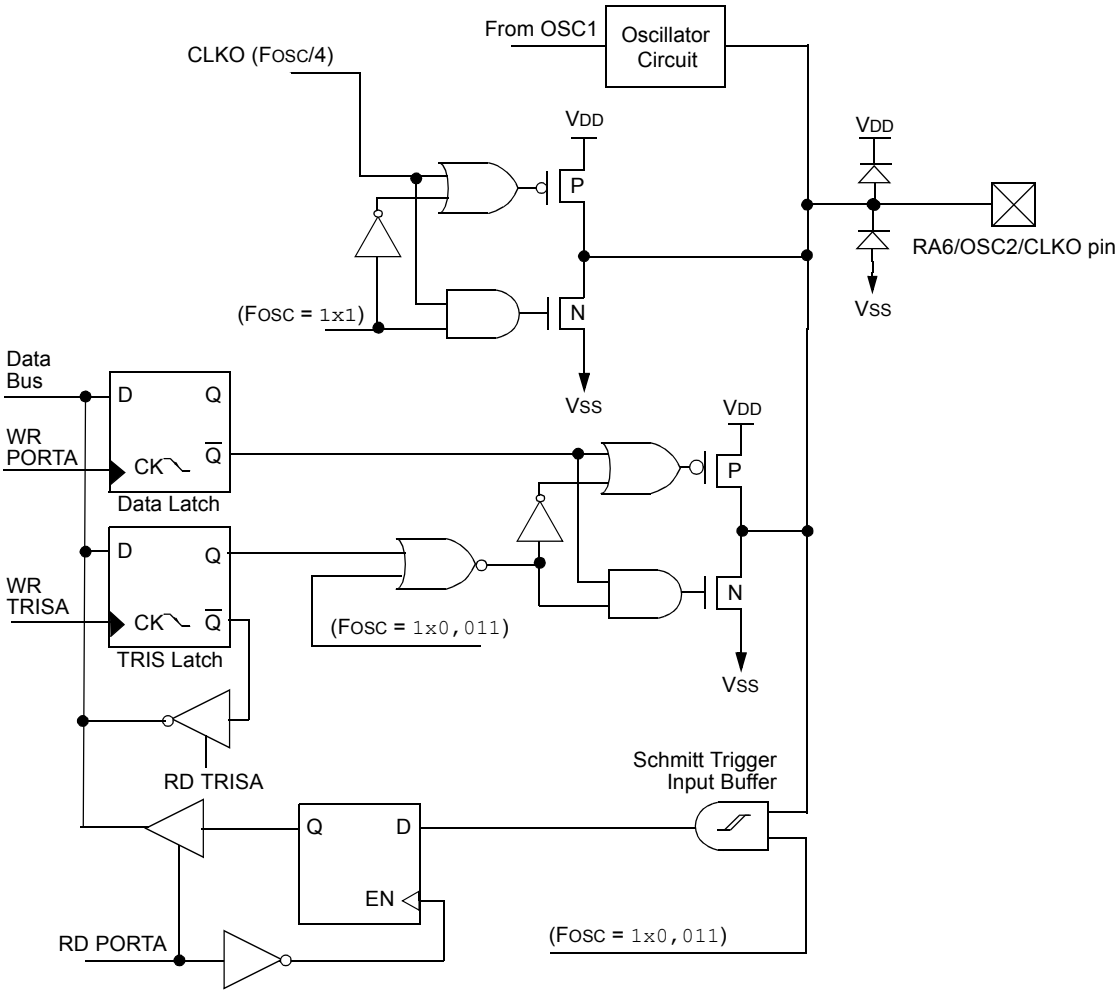


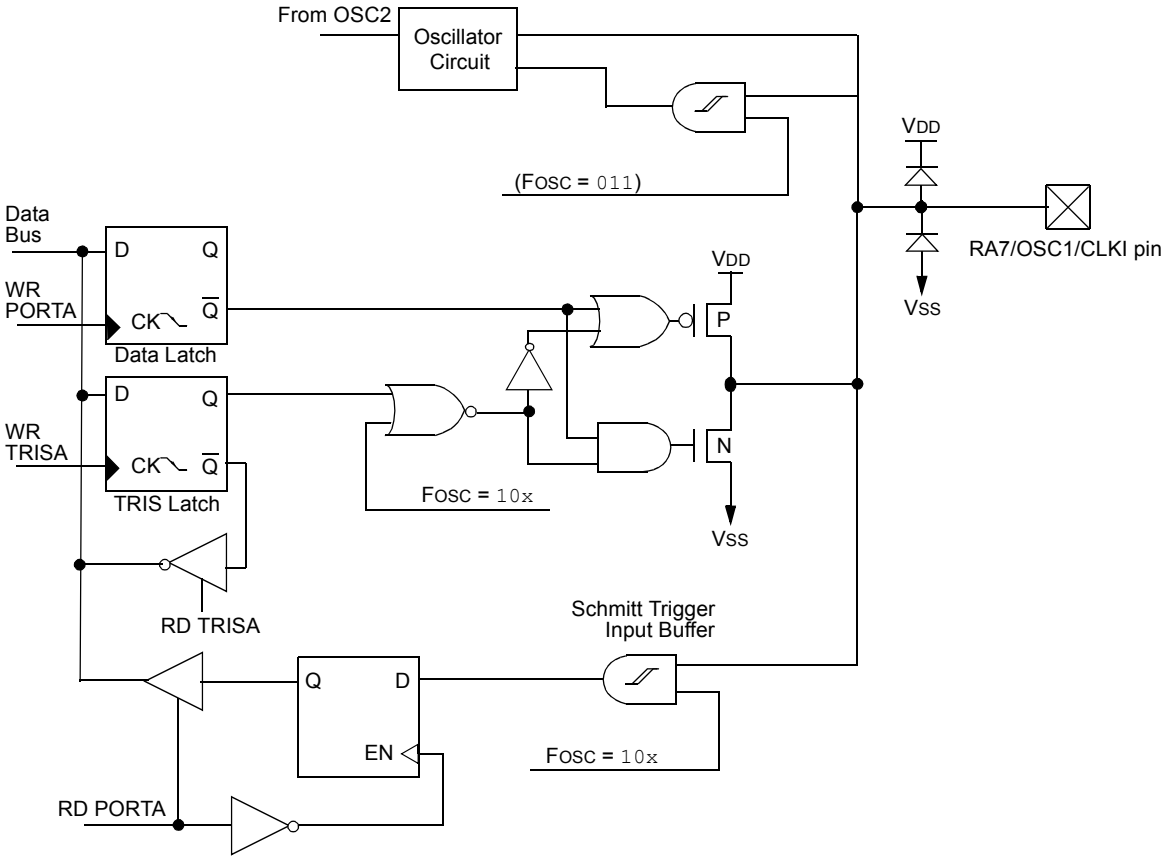

 

**FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKO PIN**




**Note 1:** I/O pins have protection diodes to VDD and VSS.
**2:** CLKO signal is 1/4 of the FOSC frequency.

**FIGURE 5-7: BLOCK DIAGRAM OF RA7/OSC1/CLKI PIN**




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

## 5.2 PORTB and the TRISB Register

PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{\text{RBPU}}$ (OPTION<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of PORTB's pins, RB7:RB4, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB7:RB4 pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB7:RB4 are OR'd together to generate the RB Port Change Interrupt with flag bit RBIF (INTCON<0>).

This interrupt can wake the device from SLEEP. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of PORTB. This will end the mismatch condition.
b) Clear flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

RB0/INT is an external interrupt input pin and is configured using the INTEDG bit (OPTION<6>).

PORTB is multiplexed with several peripheral functions (see Table 5-3). PORTB pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTB pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. Since the TRIS bit override is in effect while the peripheral is enabled, read-modify-write instructions (`BSF, BCF, XORWF`) with TRISB as destination should be avoided. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

**TABLE 5-3: PORTB FUNCTIONS**

| Name | Bit# | Buffer | Function |
|---|---|---|---|
| RB0/INT/CCP1 | bit 0 | TTL/ST(1) | Input/output pin or external interrupt input.<br>Capture input/Compare output/PWM output pin.<br>Internal software programmable weak pull-up. |
| RB1/SDI/SDA | bit 1 | TTL/ST(5) | Input/output pin, SPI Data input pin or $I^2C$ Data I/O pin.<br>Internal software programmable weak pull-up. |
| RB2/SDO/RX/DT | bit 2 | TTL/ST(4) | Input/output pin, SPI Data output pin.<br>USART Asynchronous Receive or Synchronous Data.<br>Internal software programmable weak pull-up. |
| RB3/CCP1/PGM(3) | bit 3 | TTL/ST(2) | Input/output pin, Capture input/Compare output/PWM output pin or programming in LVP mode. Internal software programmable weak pull-up. |
| RB4/SCK/SCL | bit 4 | TTL/ST(5) | Input/output pin or SPI and $I^2C$ clock pin (with interrupt-on-change).<br>Internal software programmable weak pull-up. |
| RB5/$\overline{SS}$/TX/CK | bit 5 | TTL | Input/output pin or SPI Slave select pin (with interrupt-on-change).<br>USART Asynchronous Transmit or Synchronous Clock.<br>Internal software programmable weak pull-up. |
| RB6/PGC/T1OSO/ T1CKI/AN5 | bit 6 | TTL/ST(2) | Input/output pin, Analog input(6), Timer1 Oscillator output pin, Timer1 Clock input pin or Serial Programming Clock (with interrupt-on-change).<br>Internal software programmable weak pull-up. |
| RB7/PGD/T1OSI/AN6 | bit 7 | TTL/ST(2) | Input/output pin, Analog input(6), Timer1 Oscillator input pin or Serial Programming Data (with interrupt-on-change).<br>Internal software programmable weak pull-up. |

Legend: TTL = TTL input, ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.

**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.

**3:** Low Voltage ICSP Programming (LVP) is enabled by default, which disables the RB3 I/O function. LVP must be disabled to enable RB3 as an I/O pin and allow maximum compatibility to the other 18-pin mid-range devices.

**4:** This buffer is a Schmitt Trigger input when configured for CCP or SSP mode.

**5:** This buffer is a Schmitt Trigger input when configured for SPI or $I^2C$ mode.

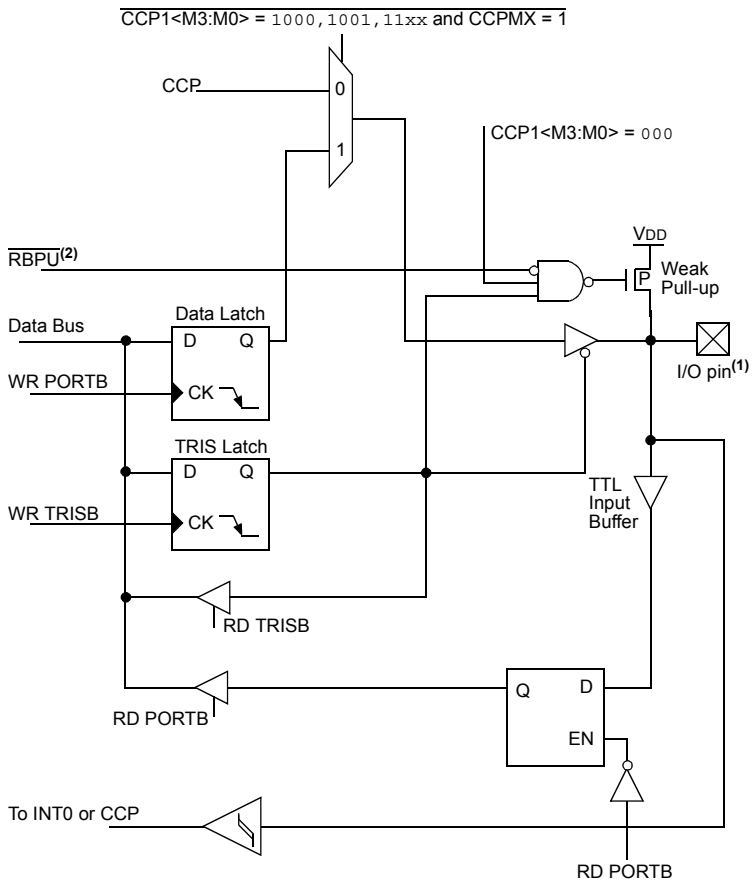
**6:** PIC16F88 only.

**TABLE 5-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 06h, 106h | PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | uuuu uuuu |
| 86h, 186h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 81h, 181h | OPTION | RBPU | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |

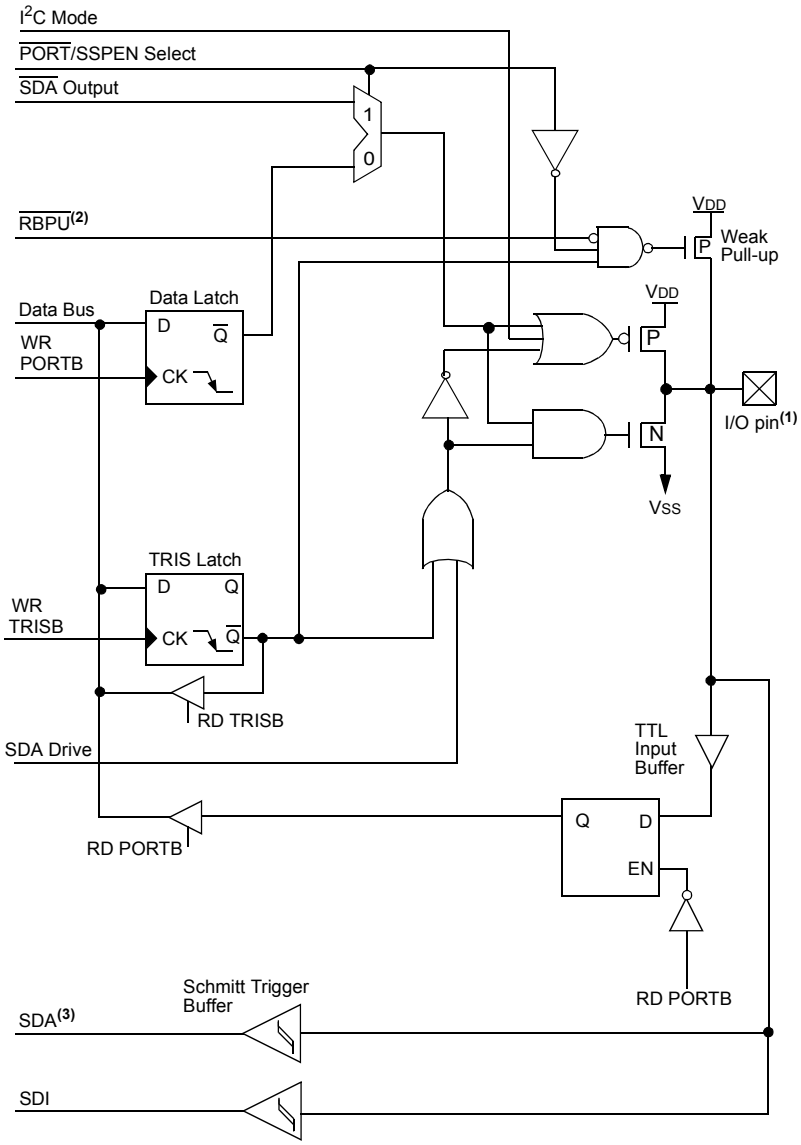
Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

**FIGURE 5-8: BLOCK DIAGRAM OF RB0 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.

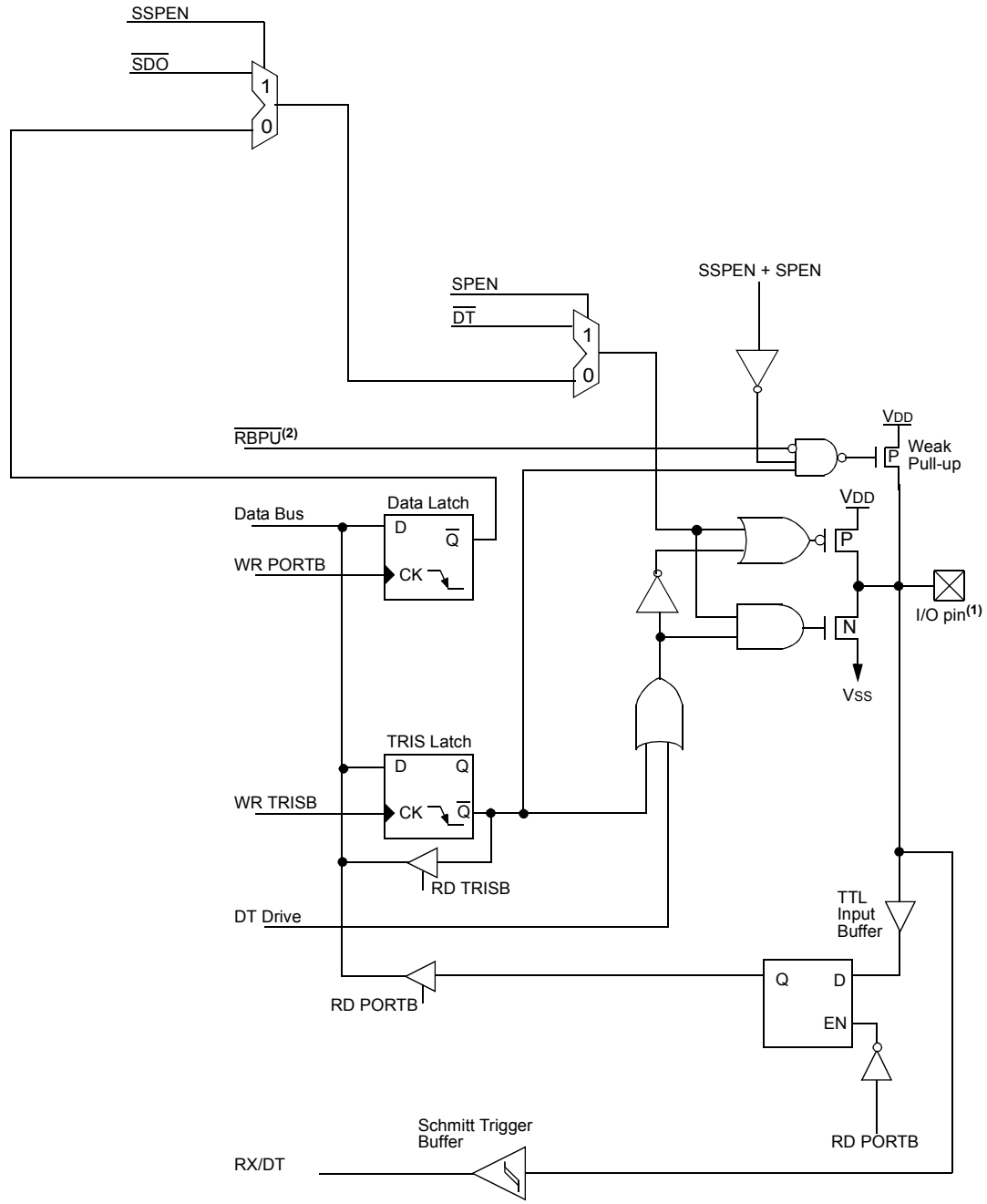
**FIGURE 5-9: BLOCK DIAGRAM OF RB1 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.
**3:** The SDA Schmitt conforms to the I2C specification.

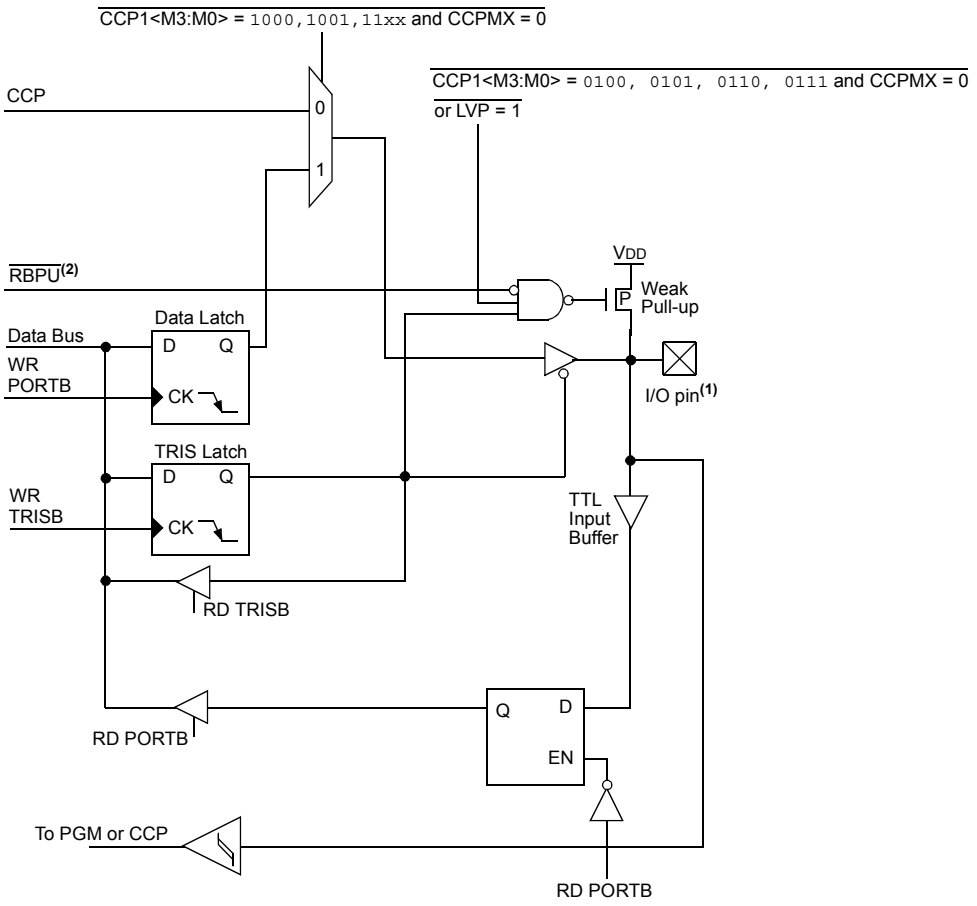
 

**FIGURE 5-10: BLOCK DIAGRAM OF RB2 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.

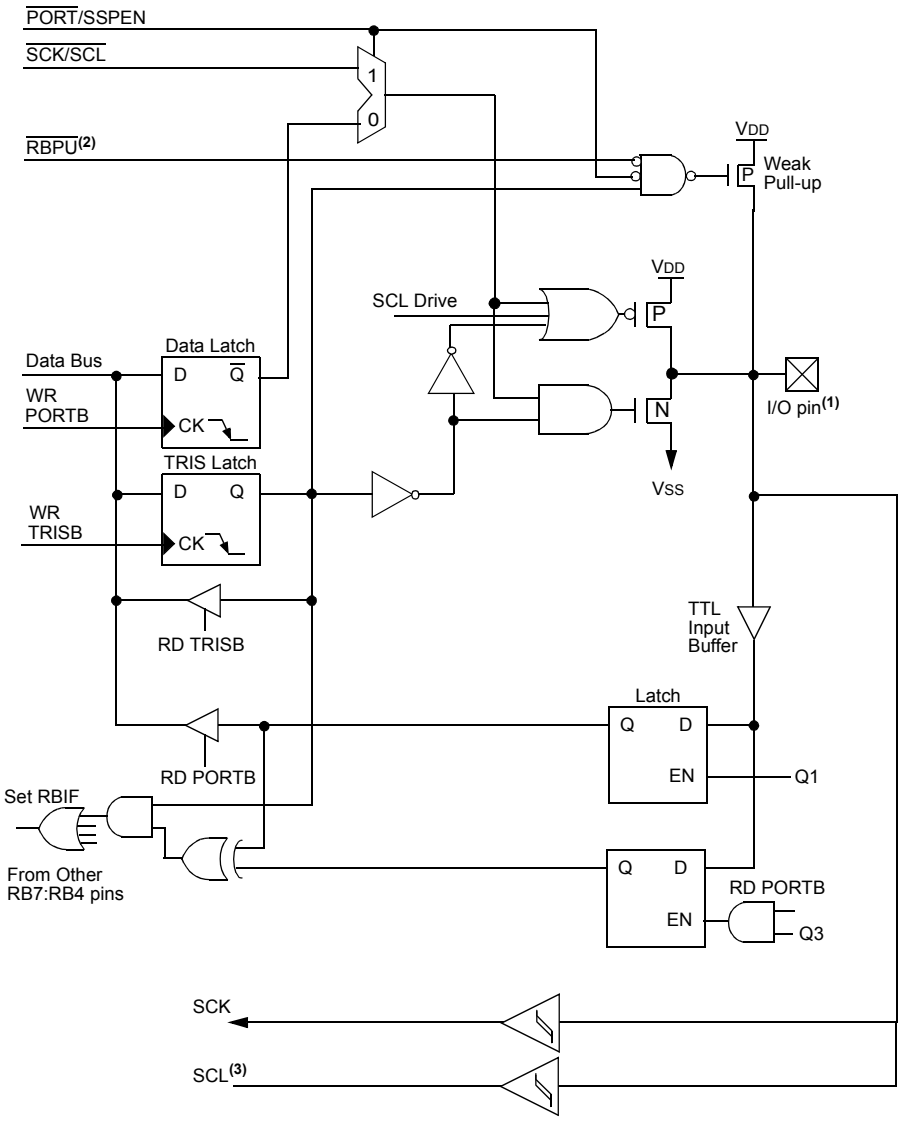
**FIGURE 5-11: BLOCK DIAGRAM OF RB3 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.

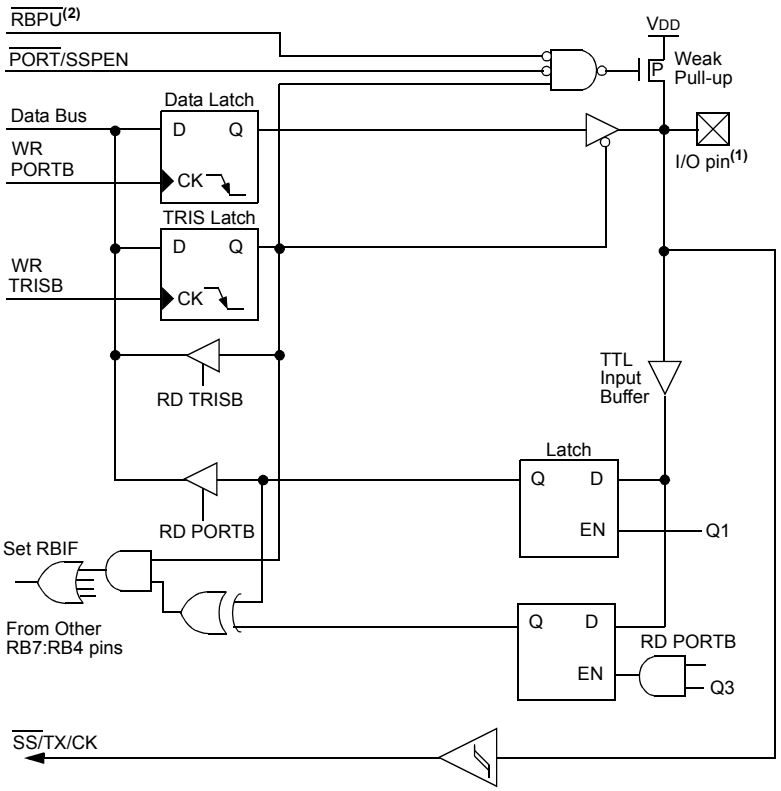
 

**FIGURE 5-12: BLOCK DIAGRAM OF RB4 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.
**3:** The SCL Schmitt conforms to the $I^2C$ specification.

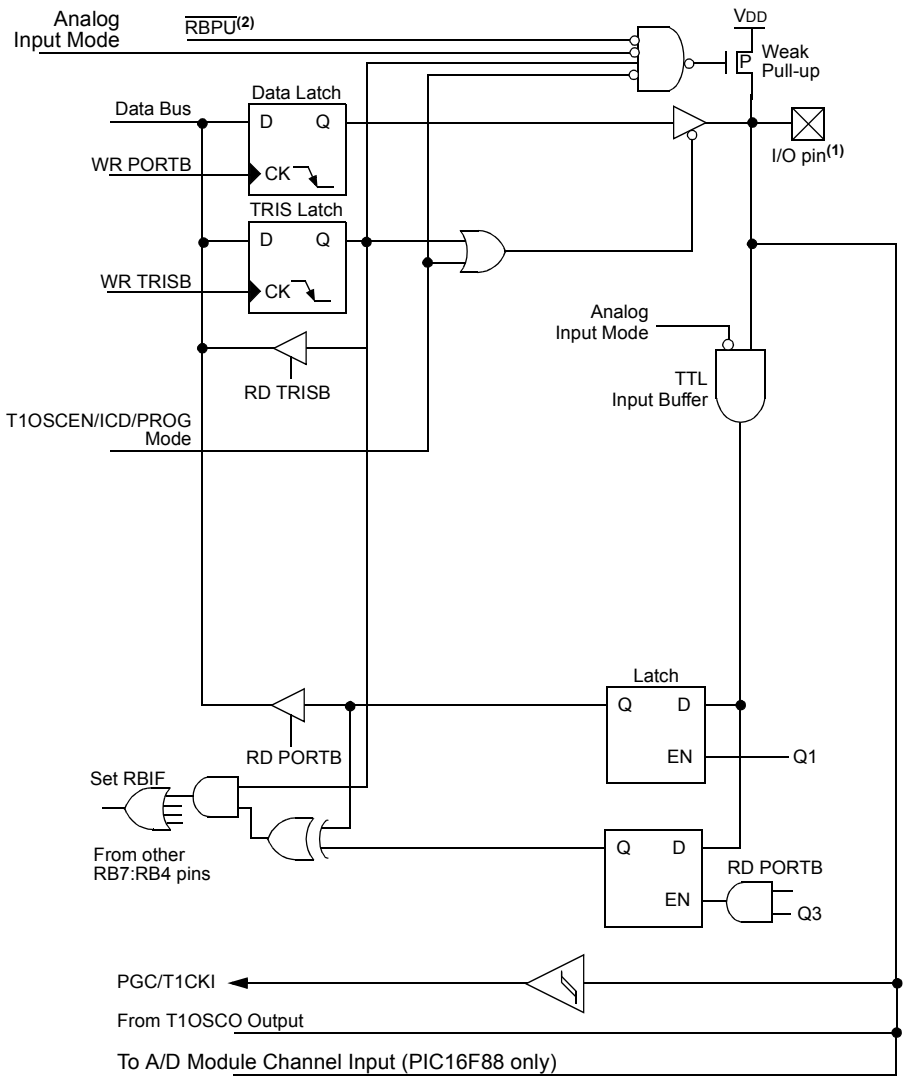
**FIGURE 5-13: BLOCK DIAGRAM OF RB5 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.

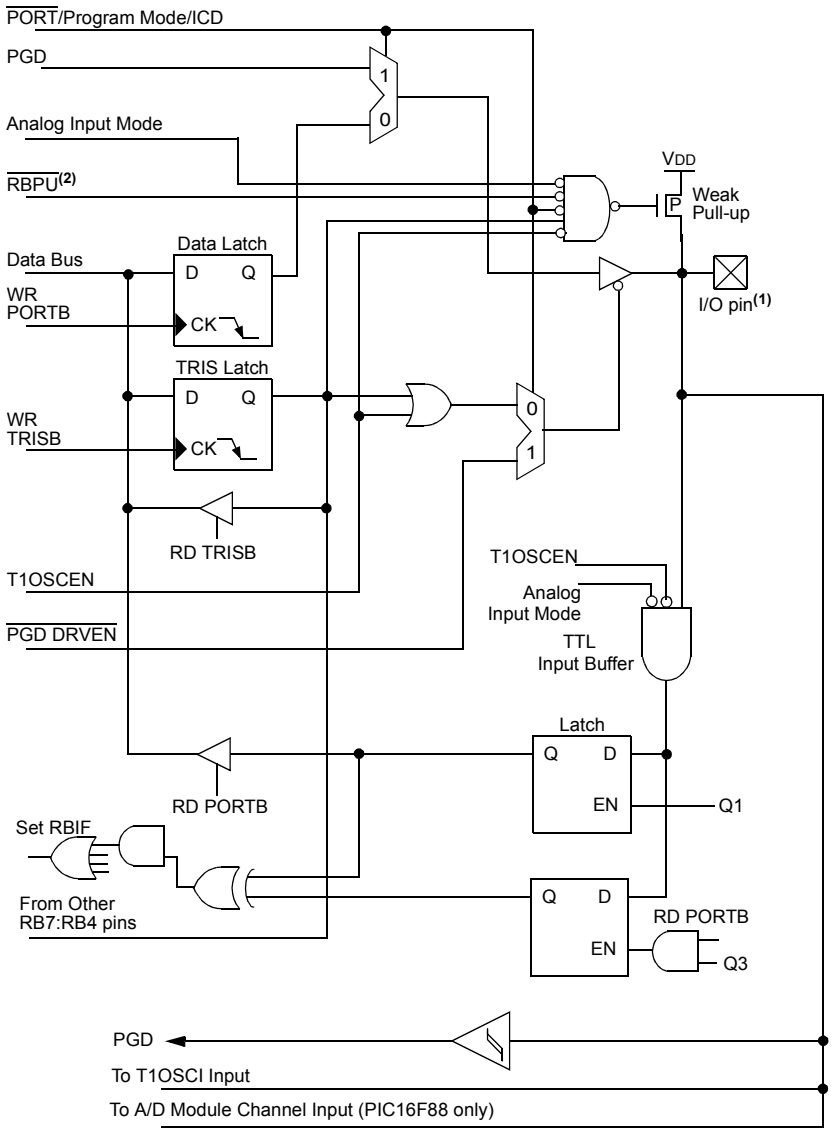
 

**FIGURE 5-14: BLOCK DIAGRAM OF RB6 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.

**FIGURE 5-15: BLOCK DIAGRAM OF RB7 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit.

## 6.0 TIMER0 MODULE

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt on overflow from FFh to 00h
- Edge select for external clock

Additional information on the Timer0 module is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

Figure 6-1 is a block diagram of the Timer0 module and the prescaler shared with the WDT.

### 6.1 Timer0 Operation

Timer0 operation is controlled through the OPTION register (see Register 2-2). Timer mode is selected by clearing bit T0CS (OPTION<5>). In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

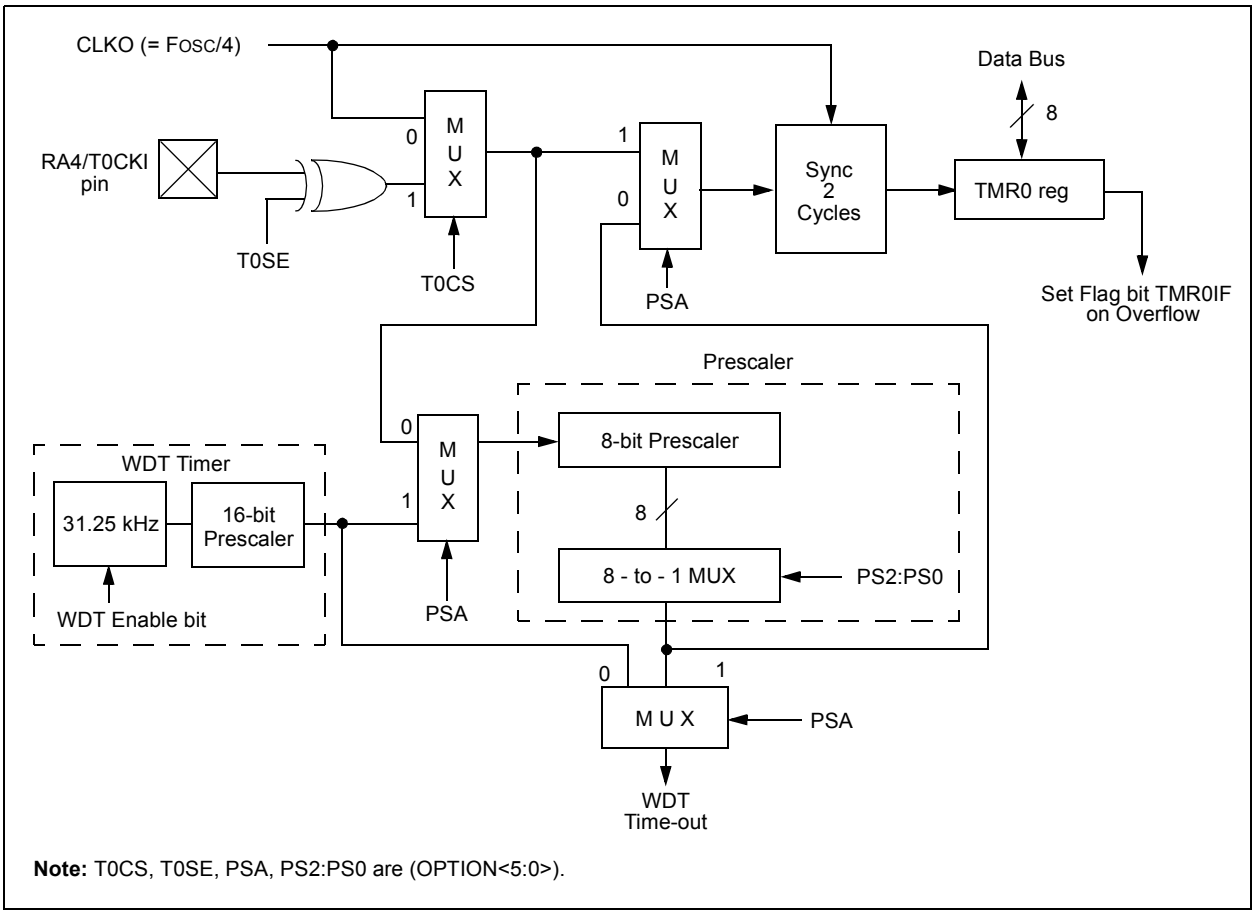
Counter mode is selected by setting bit T0CS (OPTION<5>). In Counter mode, Timer0 will increment, either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (OPTION<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.3.

The prescaler is mutually, exclusively shared between the Timer0 module and the Watchdog Timer. The prescaler is not readable or writable. Section 6.4 details the operation of the prescaler.

### 6.2 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit TMR0IF (INTCON<2>). The interrupt can be masked by clearing bit TMR0IE (INTCON<5>). Bit TMR0IF must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from SLEEP, since the timer is shut-off during SLEEP.

**FIGURE 6-1: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER**




**Note:** T0CS, T0SE, PSA, PS2:PS0 are (OPTION<5:0>).

## 6.3 Using Timer0 with an External Clock

When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI, with the internal phase clocks, is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks. Therefore, it is necessary for T0CKI to be high for at least 2 TOSC (and a small RC delay of 20 ns) and low for at least 2 TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

## 6.4 Prescaler

There is only one prescaler available, which is mutually exclusively shared between the Timer0 module and the Watchdog Timer. A prescaler assignment for the Timer0 module means that the prescaler cannot be used by the Watchdog Timer, and vice-versa. This prescaler is not readable or writable (see Figure 6-1).

**Note:** Although the prescaler can be assigned to either the WDT or Timer0, but not both, a new divide counter is implemented in the WDT circuit to give multiple WDT time-out selection. This allows TMR0 and WDT to each have their own scaler. Refer to Section 15.12 for further details.

The PSA and PS2:PS0 bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF 1, MOVWF 1, BSF 1,x`....etc.) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

**Note:** Writing to TMR0 when the prescaler is assigned to Timer0, will clear the prescaler count but will not change the prescaler assignment.

**REGISTER 6-1: OPTION_REG REGISTER**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{RBPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **$\overline{\text{RBPU}}$**

bit 6 **INTEDG**

bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

| Bit Value | TMR0 Rate | WDT Rate |
|---|---|---|
| 000 | 1 : 2 | 1 : 1 |
| 001 | 1 : 4 | 1 : 2 |
| 010 | 1 : 8 | 1 : 4 |
| 011 | 1 : 16 | 1 : 8 |
| 100 | 1 : 32 | 1 : 16 |
| 101 | 1 : 64 | 1 : 32 |
| 110 | 1 : 128 | 1 : 64 |
| 111 | 1 : 256 | 1 : 128 |

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

**Note:** To avoid an unintended device RESET, the instruction sequence shown in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be followed even if the WDT is disabled.

**EXAMPLE 6-1: CHANGING THE PRESCALER ASSIGNMENT FROM WDT TO TIMER0**

```
CLRWDT                ; Clear WDT and prescaler
BANKSEL  OPTION       ; Select Bank of OPTION
MOVLW    b'xxxx0xxx'  ; Select TMR0, new prescale
MOVWF    OPTION       ; value and clock source
```

**TABLE 6-1: REGISTERS ASSOCIATED WITH TIMER0**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 01h,101h | TMR0 | Timer0 Module Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Bh,8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 81h,181h | OPTION | $\overline{\text{RBPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by Timer0.

**NOTES:**

## 7.0 TIMER1 MODULE

The Timer1 module is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L), which are readable and writable. The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR1 interrupt, if enabled, is generated on overflow, which is latched in interrupt flag bit TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing TMR1 interrupt enable bit TMR1IE (PIE1<0>).

The Timer1 oscillator can be used as a secondary clock source in Low Power modes. When the T1RUN bit is set, the Timer1 oscillator is providing the system clock. If the Fail-Safe Clock Monitor is enabled, and the Timer1 oscillator fails while providing the system clock, polling the T1RUN bit will indicate whether the clock is being provided by the Timer1 oscillator or another source.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components or code overhead.

### 7.1 Timer1 Operation

Timer1 can operate in one of three modes:

- as a Timer
- as a Synchronous Counter
- as an Asynchronous Counter

The Operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

In Timer mode, Timer1 increments every instruction cycle. In Counter mode, it increments on every rising edge of the external clock input.

Timer1 can be enabled/disabled by setting/clearing control bit TMR1ON (T1CON<0>).

Timer1 also has an internal “RESET input”. This RESET can be generated by the CCP1 module as the special event trigger (see Section 9.1). Register 7-1 shows the Timer1 Control register.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RB6/T1OSO/T1CKI/PGC and RB7/T1OSI/PGD pins become inputs. That is, the TRISB<7:6> value is ignored and these pins read as ‘0’.

Additional information on timer modules is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

**REGISTER 7-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)**

| U-0 | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **T1RUN**: Timer1 System Clock Status bit
1 = System clock is derived from Timer1 oscillator
0 = System clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut-off (the oscillator inverter is turned off to eliminate power drain)

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Control bit
TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RB6/T1OSO/T1CKI/PGC (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

| 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 |

## 7.2 Timer1 Operation in Timer Mode

Timer mode is selected by clearing the TMR1CS (T1CON<1>) bit. In this mode, the input clock to the timer is FOSC/4. The synchronize control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) has no effect, since the internal clock is always in sync.

## 7.3 Timer1 Counter Operation

Timer1 may operate in Asynchronous or Synchronous mode, depending on the setting of the TMR1CS bit.

When Timer1 is being incremented via an external source, increments occur on a rising edge. After Timer1 is enabled in Counter mode, the module must first have a falling edge before the counter begins to increment.

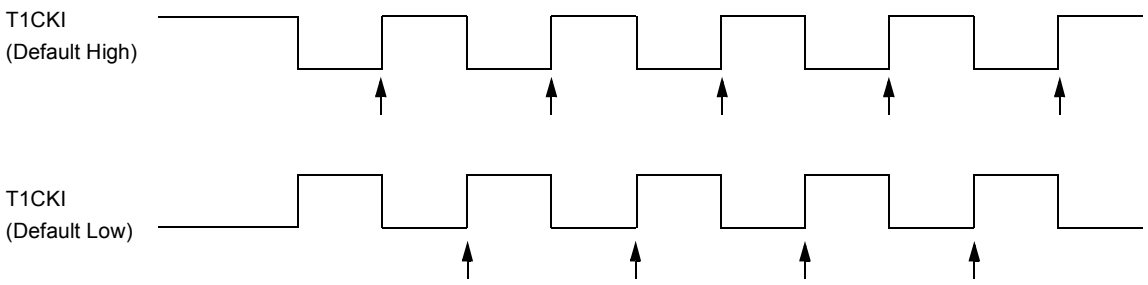
## 7.4 Timer1 Operation in Synchronized Counter Mode

Counter mode is selected by setting bit TMR1CS. In this mode, the timer increments on every rising edge of clock input on pin RB7/T1OSI/PGD, when bit T1OSCEN is set, or on pin RB6/T1OSO/T1CKI/PGC, when bit T1OSCEN is cleared.

If $\overline{\text{T1SYNC}}$ is cleared, then the external clock input is synchronized with internal phase clocks. The synchronization is done after the prescaler stage. The prescaler stage is an asynchronous ripple counter.

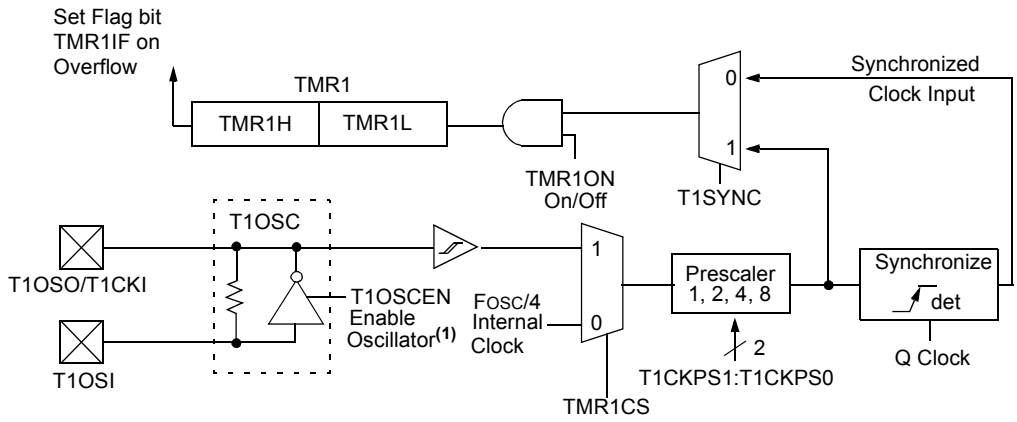
In this configuration, during SLEEP mode, Timer1 will not increment even if the external clock is present, since the synchronization circuit is shut-off. The prescaler, however, will continue to increment.

**FIGURE 7-1: TIMER1 INCREMENTING EDGE**




**Note:** Arrows indicate counter increments.

**FIGURE 7-2: TIMER1 BLOCK DIAGRAM**




**Note 1:** When the T1OSCEN bit is cleared, the inverter is turned off. This eliminates power drain.

## 7.5 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during SLEEP and can generate an interrupt on overflow that will wake-up the processor. However, special precautions in software are needed to read/write the timer (Section 7.5.1).

In Asynchronous Counter mode, Timer1 cannot be used as a time-base for capture or compare operations.

### 7.5.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers while the register is incrementing. This may produce an unpredictable value in the timer register.

Reading the 16-bit value requires some care. The example codes provided in Example 7-1 and Example 7-2 demonstrate how to write to and read Timer1 while it is running in Asynchronous mode.

**EXAMPLE 7-1: WRITING A 16-BIT FREE-RUNNING TIMER**

```
; All interrupts are disabled
CLRF    TMR1L       ; Clear Low byte, Ensures no rollover into TMR1H
MOVLW   HI_BYTE     ; Value to load into TMR1H
MOVWF   TMR1H, F    ; Write High byte
MOVLW   LO_BYTE     ; Value to load into TMR1L
MOVWF   TMR1H, F    ; Write Low byte
; Re-enable the Interrupt (if required)
CONTINUE            ; Continue with your code
```

**EXAMPLE 7-2: READING A 16-BIT FREE-RUNNING TIMER**

```
; All interrupts are disabled
MOVF    TMR1H, W    ; Read high byte
MOVWF   TMPH
MOVF    TMR1L, W    ; Read low byte
MOVWF   TMPL
MOVF    TMR1H, W    ; Read high byte
SUBWF   TMPH,  W    ; Sub 1st read with 2nd read
BTFSC   STATUS,Z    ; Is result = 0
GOTO    CONTINUE    ; Good 16-bit read
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
MOVF    TMR1H, W    ; Read high byte
MOVWF   TMPH
MOVF    TMR1L, W    ; Read low byte
MOVWF   TMPL        ; Re-enable the Interrupt (if required)
CONTINUE            ; Continue with your code
```

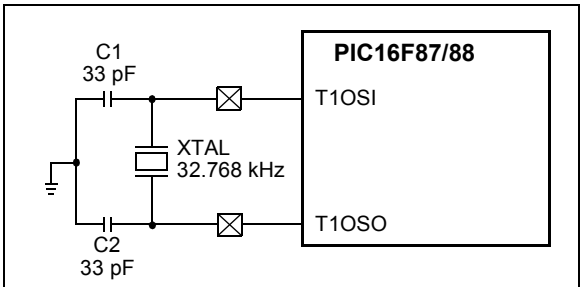
 

## 7.6 Timer1 Oscillator

A crystal oscillator circuit is built between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit T1OSCEN (T1CON<3>). The oscillator is a low power oscillator rated up to 32.768 kHz. It will continue to run during all Power Managed modes. It is primarily intended for a 32 kHz crystal. The circuit for a typical LP oscillator is shown in Figure 7-3. Table 7-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper oscillator start-up.

**FIGURE 7-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR**




**Note:** See the Notes with Section 7-1 for additional information about capacitor selection.

**TABLE 7-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR**

| Osc Type | Freq | C1 | C2 |
|---|---|---|---|
| LP | 32 kHz | 33 pF | 33 pF |

**Note 1:** Microchip suggests this value as a starting point in validating the oscillator circuit.

**2:** Higher capacitance increases the stability of the oscillator, but also increases the start-up time.

**3:** Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

**4:** Capacitor values are for design guidance only.

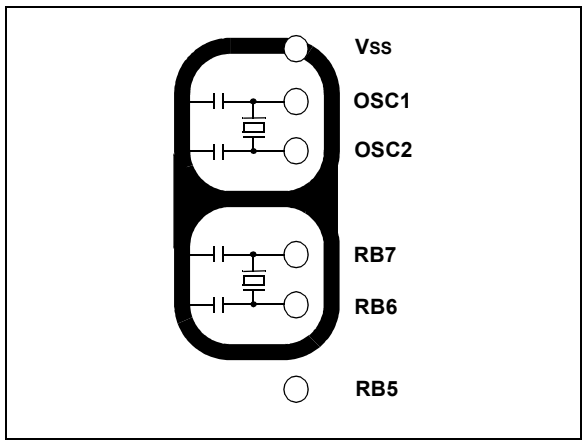
## 7.7 Timer1 Oscillator Layout Considerations

The Timer1 oscillator circuit draws very little power during operation. Due to the low power nature of the oscillator, it may also be sensitive to rapidly changing signals in close proximity.

The oscillator circuit, shown in Figure 7-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high speed circuit must be located near the oscillator, a grounded guard ring around the oscillator circuit, as shown in Figure 7-4, may be helpful when used on a single sided PCB, or in addition to a ground plane.

**FIGURE 7-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING**




## 7.8 Resetting Timer1 Using a CCP Trigger Output

If the CCP1 module is configured in Compare mode to generate a “special event trigger" signal (CCP1M3:CCP1M0 = 1011), the signal will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

**Note:** The special event triggers from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this RESET operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer1.

## 7.9 Resetting Timer1 Register Pair (TMR1H, TMR1L)

TMR1H and TMR1L registers are not reset to 00h on a POR, or any other RESET, except by the CCP1 special event triggers.

T1CON register is reset to 00h on a Power-on Reset or a Brown-out Reset, which shuts off the timer and leaves a 1:1 prescale. In all other RESETS, the register is unaffected.

## 7.10 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

## 7.11 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in Section 7.6, above), gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time-base, and several lines of application code to calculate the time. When operating in SLEEP mode and using a battery or super capacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCisr`, shown in Example 7-3, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow triggers the interrupt and calls the routine, which increments the seconds counter by one; additional counters for minutes and hours are incremented as the previous counter overflow.

Since the register pair is 16-bits wide, counting up to overflow the register directly from a 32.768 kHz clock would take 2 seconds. To force the overflow at the required one-second intervals, it is necessary to preload it; the simplest method is to set the MSbit of TMR1H with a `BSF` instruction. Note that the TMR1L register is never pre-loaded or altered; doing so may introduce cumulative error over many cycles.

For this method to be accurate, Timer1 must operate in Asynchronous mode, and the Timer1 Overflow Interrupt must be enabled (PIE1<0> = 1), as shown in the routine, `RTCinit`. The Timer1 oscillator must also be enabled and running at all times.

**EXAMPLE 7-3: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE**

```
RTCinit     banksel  TMR1H
            movlw    0x80          ; Preload TMR1 register pair
            movwf    TMR1H         ; for 1 second overflow
            clrf     TMR1L
            movlw    b'00001111'   ; Configure for external clock,
            movwf    T1CON         ; Asynchronous operation, external oscillator
            clrf     secs          ; Initialize timekeeping registers
            clrf     mins
            movlw    .12
            movwf    hours
            banksel  PIE1
            bsf      PIE1, TMR1IE  ; Enable Timer1 interrupt
            return
RTCisr      banksel  TMR1H
            bsf      TMR1H,7       ; Preload for 1 sec overflow
            bcf      PIR1,TMR1IF   ; Clear interrupt flag
            incf     secs,F        ; Increment seconds
            movf     secs,w
            sublw    .60
            btfss    STATUS,Z      ; 60 seconds elapsed?
            return                 ; No, done
            clrf     seconds       ; Clear seconds
            incf     mins,f        ; Increment minutes
            movf     mins,w
            sublw    .60
            btfss    STATUS,Z      ; 60 seconds elapsed?
            return                 ; No, done
            clrf     mins          ; Clear minutes
            incf     hours,f       ; Increment hours
            movf     hours,w
            sublw    .24
            btfss    STATUS,Z      ; 24 hours elapsed?
            return                 ; No, done
            clrf     hours         ; Clear hours
            return                 ; Done
```

**TABLE 7-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh, 18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 0Eh | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Fh | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10h | T1CON | — | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYNC | TMR1CS | TMR1ON | -000 0000 | -uuu uuuu |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

**NOTES:**

## 8.0 TIMER2 MODULE

Timer2 is an 8-bit timer with a prescaler and a postscaler. It can be used as the PWM time-base for the PWM mode of the CCP1 module. The TMR2 register is readable and writable, and is cleared on any device RESET.

The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits T2CKPS1:T2CKPS0 (T2CON<1:0>).

The Timer2 module has an 8-bit period register, PR2. Timer2 increments from 00h until it matches PR2 and then resets to 00h on the next increment cycle. PR2 is a readable and writable register. The PR2 register is initialized to FFh upon RESET.

The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit TMR2IF (PIR1<1>)).

Timer2 can be shut-off by clearing control bit TMR2ON (T2CON<2>) to minimize power consumption.

Register 8-1 shows the Timer2 control register.

Additional information on timer modules is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

### 8.1 Timer2 Prescaler and Postscaler

The prescaler and postscaler counters are cleared when any of the following occurs:

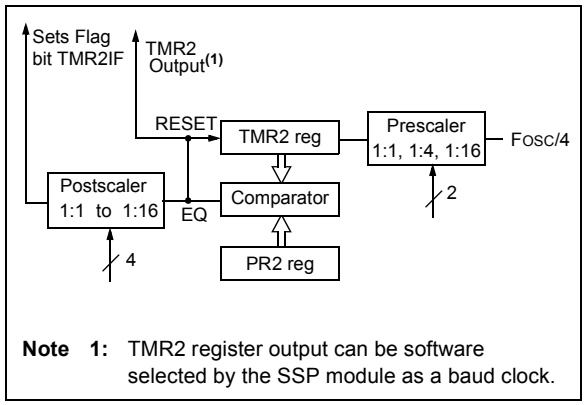
- A write to the TMR2 register
- A write to the T2CON register
- Any device RESET (Power-on Reset, $\overline{\text{MCLR}}$, WDT Reset, or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

### 8.2 Output of TMR2

The output of TMR2 (before the postscaler) is fed to the Synchronous Serial Port module, which optionally uses it to generate a shift clock.

**FIGURE 8-1: TIMER2 BLOCK DIAGRAM**




**Note 1:** TMR2 register output can be software selected by the SSP module as a baud clock.

**REGISTER 8-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)**

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits

0000 = 1:1 Postscale
0001 = 1:2 Postscale
0010 = 1:3 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit

1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits

00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

| 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 |

**TABLE 8-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh, 18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 11h | TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 12h | T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | -000 0000 |
| 92h | PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer2 module.

## 9.0 CAPTURE/COMPARE/PWM (CCP) MODULE

The Capture/Compare/PWM (CCP) module contains a 16-bit register that can operate as a:

- 16-bit capture register
- 16-bit compare register
- PWM master/slave duty cycle register.

Table 9-1 shows the timer resources of the CCP module modes.

Capture/Compare/PWM Register1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. The special event trigger is generated by a compare match which will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

The CCP module's input/output pin (CCP1) can be configured as RB0 or RB3. This selection is set in bit 12 (CCPMX) of the configuration word.

Additional information on the CCP module is available in the PICmicro™ Mid-Range MCU Reference Manual, (DS33023) and in Application Note AN594, "*Using the CCP Modules*" (DS00594).

**TABLE 9-1: CCP MODE - TIMER RESOURCE**

| CCP Mode | Timer Resource |
|---|---|
| Capture | Timer1 |
| Compare | Timer1 |
| PWM | Timer2 |

**REGISTER 9-1: CCP1CON: CAPTURE/COMPARE/PWM CONTROL REGISTER 1 (ADDRESS 17h)**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | CCP1X | CCP1Y | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 |
| bit 7 | | | | | | | bit 0 |

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCP1X:CCP1Y:** PWM Least Significant bits

Capture mode:
Unused
Compare mode:
Unused
PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0 **CCP1M<3:0>:** CCP1 Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCP1 module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCP1IF bit is set)
1001 = Compare mode, clear output on match (CCP1IF bit is set)
1010 = Compare mode, generate software interrupt on match (CCP1IF bit is set, CCP1 pin is unaffected)
1011 = Compare mode, trigger special event (CCP1IF bit is set, CCP1 pin is unaffected); CCP1 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

| 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 |

## 9.1 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on CCP1 pin. An event is defined as:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

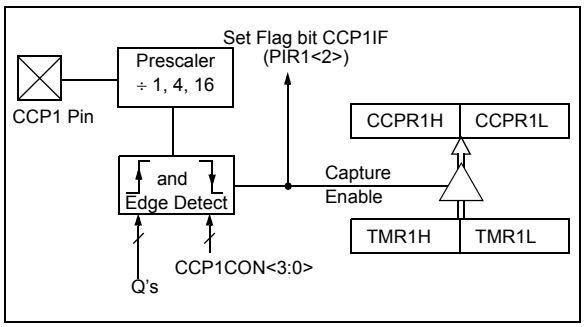
An event is selected by control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit CCP1IF (PIR1<2>) is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten by the new captured value.

### 9.1.1 CCP PIN CONFIGURATION

In Capture mode, the CCP1 pin should be configured as an input by setting the TRISB<x> bit.

| | |
|---|---|
| **Note 1:** | If the CCP1 pin is configured as an output, a write to the port can cause a capture condition. |
| **2:** | The TRISB bit (0 or 3) is dependent upon the setting of configuration bit 12 (CCPMX). |

**FIGURE 9-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**




### 9.1.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode for the CCP module to use the capture feature. In Asynchronous Counter mode, the capture operation may not work.

### 9.1.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP1IE (PIE1<2>) clear to avoid false interrupts and should clear the flag bit CCP1IF, following any such change in Operating mode.

### 9.1.4 CCP PRESCALER

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

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 9-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

**EXAMPLE 9-1: CHANGING BETWEEN CAPTURE PRESCALERS**

```
CLRF    CCP1CON     ;Turn CCP module off
MOVLW   NEW_CAPT_PS ;Load the W reg with
                    ;the new prescaler
                    ;move value and CCP ON
MOVWF   CCP1CON     ;Load CCP1CON with this
                    ;value
```

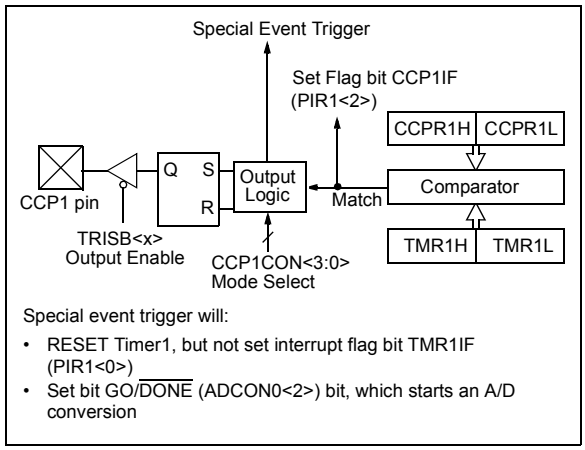
## 9.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the CCP1 pin is:

- Driven High
- Driven Low
- Remains Unchanged

The action on the pin is based on the value of control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). At the same time, interrupt flag bit CCP1IF is set.

**FIGURE 9-2: COMPARE MODE OPERATION BLOCK DIAGRAM**




### 9.2.1 CCP PIN CONFIGURATION

The user must configure the CCP1 pin as an output by clearing the TRISB<x> bit.

**Note 1:** Clearing the CCP1CON register will force the CCP1 compare output latch to the default low level. This is not the data latch.

**2:** The TRISB bit (0 or 3) is dependent upon the setting of configuration bit 12 (CCPMX).

### 9.2.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

### 9.2.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

### 9.2.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated that may be used to initiate an action.

The special event trigger output of CCP1 resets the TMR1 register pair and starts an A/D conversion (if the A/D module is enabled). This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

**Note:** The special event trigger from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

**TABLE 9-2: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND TIMER1**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh,8Bh 10BH,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 86h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 0Eh | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Fh | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10h | T1CON | — | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYNC | TMR1CS | TMR1ON | -000 0000 | -uuu uuuu |
| 15h | CCPR1L | Capture/Compare/PWM Register1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 16h | CCPR1H | Capture/Compare/PWM Register1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 17h | CCP1CON | — | — | CCP1X | CCP1Y | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by Capture and Timer1.

## 9.3 PWM Mode

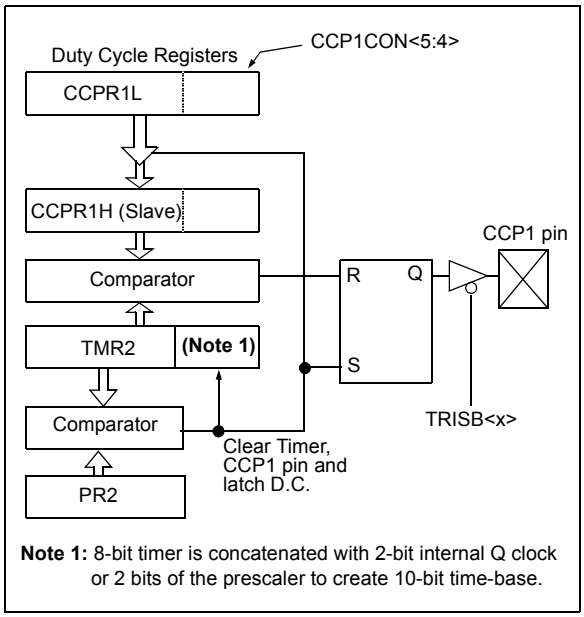
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTB data latch, the TRISB<x> bit must be cleared to make the CCP1 pin an output.

| **Note:** | Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTB I/O data latch. |
|---|---|

Figure 9-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step by step procedure on how to setup the CCP module for PWM operation, see Section 9.3.3.

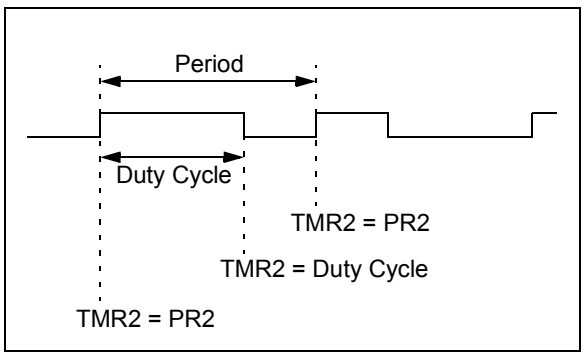
**FIGURE 9-3: SIMPLIFIED PWM BLOCK DIAGRAM**




**Note 1:** 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time-base.

A PWM output (Figure 9-4) has a time-base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

**FIGURE 9-4: PWM OUTPUT**




### 9.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula.

**EQUATION 9-1:**

$$\text{PWM period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

PWM frequency is defined as 1 / [PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

| **Note:** | The Timer2 postscaler (see Section 8.0) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output. |
|---|---|

### 9.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time.

**EQUATION 9-2:**

$$\text{PWM duty cycle} = (CCPR1L{:}CCP1CON\langle 5{:}4\rangle) \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read only register.

The CCPR1H 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 CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the following formula.

**EQUATION 9-3:**

$$\text{Resolution} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{ bits}$$

**Note:** If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

### 9.3.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP 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 CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISB<x> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

**Note:** The TRISB bit (0 or 3) is dependant upon the setting of configuration bit 12 (CCPMX).

**TABLE 9-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 20 MHz**

| PWM Frequency | 1.22 kHz | 4.88 kHz | 19.53 kHz | 78.12 kHz | 156.3 kHz | 208.3 kHz |
|---|---|---|---|---|---|---|
| Timer Prescaler (1, 4, 16) | 16 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 5.5 |

**TABLE 9-4: REGISTERS ASSOCIATED WITH PWM AND TIMER2**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh,8Bh 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 86h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 11h | TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 92h | PR2 | Timer2 Module Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 12h | T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | -000 0000 |
| 15h | CCPR1L | Capture/Compare/PWM Register1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 16h | CCPR1H | Capture/Compare/PWM Register1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 17h | CCP1CON | — | — | CCP1X | CCP1Y | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.

**NOTES:**

# 10.0 SYNCHRONOUS SERIAL PORT (SSP) MODULE

## 10.1 SSP Module Overview

The Synchronous Serial Port (SSP) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be Serial EEPROMs, shift registers, display drivers, A/D converters, etc. The SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit ($I^2C$)

An overview of $I^2C$ operations and additional information on the SSP module can be found in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

Refer to Application Note AN578, *"Use of the SSP Module in the $I^2C$ Multi-Master Environment"* (DS00578).

## 10.2 SPI Mode

This section contains register definitions and operational characteristics of the SPI module.

SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO) RB2/SDO/RX/DT
- Serial Data In (SDI) RB1/SDI/SDA
- Serial Clock (SCK) RB4/SCK/SCL

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) RB5/$\overline{SS}$/TX/CK

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON register (SSPCON<5:0>) and the SSPSTAT register (SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (IDLE state of SCK)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

**REGISTER 10-1: SSPSTAT: SYNCHRONOUS SERIAL PORT STATUS REGISTER (ADDRESS 94h)**

| R/W-0 | R/W-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| bit 7 | | | | | | | bit 0 |

bit 7 **SMP:** SPI Data Input Sample Phase bit

SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time (Microwire®)

SPI Slave mode:
This bit must be cleared when SPI is used in Slave mode

$I^2C$ mode:
This bit must be maintained clear

bit 6 **CKE:** SPI Clock Edge Select bit

SPI mode, CKP = 0:
1 = Data transmitted on rising edge of SCK (Microwire alternate)
0 = Data transmitted on falling edge of SCK

SPI mode, CKP = 1:
1 = Data transmitted on falling edge of SCK (Microwire alternate)
0 = Data transmitted on rising edge of SCK

$I^2C$ mode:
This bit must be maintained clear

bit 5 **D/$\overline{A}$:** Data/Address bit ($I^2C$ mode only)

In $I^2C$ Slave mode:
1 = Indicates that the last byte received was data
0 = Indicates that the last byte received was address

bit 4 **P:** STOP bit(1) ($I^2C$ mode only)
1 = Indicates that a STOP bit has been detected last
0 = STOP bit was not detected last

bit 3 **S:** START bit(1) ($I^2C$ mode only)
1 = Indicates that a START bit has been detected last (this bit is '0' on RESET)
0 = START bit was not detected last

bit 2 **R/$\overline{W}$:** Read/Write Information bit ($I^2C$ mode only)
Holds the R/W bit information following the last address match, and is only valid from address match to the next START bit, STOP bit, or $\overline{ACK}$ bit
1 = Read
0 = Write

bit 1 **UA:** Update Address bit (10-bit $I^2C$ 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

Receive (SPI and $I^2C$ modes):
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty

Transmit (In $I^2C$ mode only):
1 = Transmit in progress, SSPBUF is full (8 bits)
0 = Transmit complete, SSPBUF is empty

**Note 1:** This bit is cleared when the SSP module is disabled (i.e., the SSPEN bit is cleared).

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

**REGISTER 10-2: SSPCON: SYNCHRONOUS SERIAL PORT CONTROL REGISTER 1 (ADDRESS 14h)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **WCOL:** Write Collision Detect bit
1 = An attempt to write the SSPBUF register failed because the SSP module is busy (must be cleared in software)
0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit
In SPI mode:
1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.
0 = No overflow
In $I^2C$ mode:
1 = A byte is received while the SSPBUF register is still holding the previous byte. SSPOV is a "don't care" in Transmit mode. SSPOV must be cleared in software in either mode.
0 = No overflow

bit 5 **SSPEN:** Synchronous Serial Port Enable bit**(1)**
In SPI mode:
1 = Enables serial port and configures SCK, SDO, and SDI as serial port pins
0 = Disables serial port and configures these pins as I/O port pins
In $I^2C$ mode:
1 = Enables the serial port and configures the SDA and SCL pins as serial port pins
0 = Disables serial port and configures these pins as I/O port pins

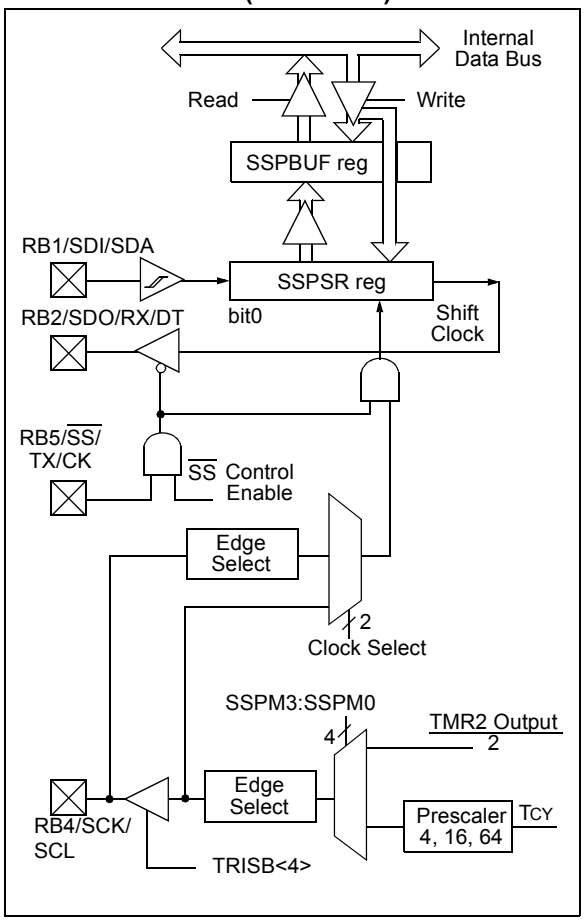
bit 4 **CKP:** Clock Polarity Select bit
In SPI mode:
1 = Transmit happens on falling edge, receive on rising edge. IDLE state for clock is a high level.
0 = Transmit happens on rising edge, receive on falling edge. IDLE state for clock is a low level.
In $I^2C$ Slave mode:
SCK release control
1 = Enable clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)

bit 3-0 **SSPM<3:0>:** Synchronous Serial Port Mode Select bits
0000 = SPI Master mode, clock = OSC/4
0001 = SPI Master mode, clock = OSC/16
0010 = SPI Master mode, clock = OSC/64
0011 = SPI Master mode, clock = TMR2 output/2
0100 = SPI Slave mode, clock = SCK pin. $\overline{SS}$ pin control enabled.
0101 = SPI Slave mode, clock = SCK pin. $\overline{SS}$ pin control disabled. $\overline{SS}$ can be used as I/O pin.
0110 = $I^2C$ Slave mode, 7-bit address
0111 = $I^2C$ Slave mode, 10-bit address
1011 = $I^2C$ firmware controlled Master mode (Slave IDLE)
1110 = $I^2C$ Slave mode, 7-bit address with START and STOP bit interrupts enabled
1111 = $I^2C$ Slave mode, 10-bit address with START and STOP bit interrupts enabled
1000, 1001, 1010, 1100, 1101 = **Reserved**

**Note 1:** In both modes, when enabled, these pins must be properly configured as input or output.

| 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 |

**FIGURE 10-1: SSP BLOCK DIAGRAM (SPI MODE)**




To enable the serial port, SSP enable bit, SSPEN (SSPCON<5>), must be set. To reset or reconfigure SPI mode, clear bit SSPEN, re-initialize the SSPCON register, and then set bit SSPEN. This configures the SDI, SDO, SCK, and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, they must have their data direction bits (in the TRISB register) appropriately programmed. That is:

- SDI must have TRISB<1> set
- SDO must have TRISB<2> cleared
- SCK (Master mode) must have TRISB<4> cleared
- SCK (Slave mode) must have TRISB<4> set
- $\overline{SS}$ must have TRISB<5> set

**Note 1:** When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.

**2:** If the SPI is used in Slave mode with CKE = 1, then the $\overline{SS}$ pin control must be enabled.

**3:** When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 0100), the state of $\overline{SS}$ pin can affect the state read back from the TRISB<5> bit. The Peripheral OE signal from the SSP module into PORTB controls the state that is read back from the TRISB<5> bit. If Read-Modify-Write instructions, such as BSF, are performed on the TRISB register while the $\overline{SS}$ pin is high, this will cause the TRISB<5> bit to be set, thus disabling the SDO output.

**TABLE 10-1: REGISTERS ASSOCIATED WITH SPI OPERATION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh,8Bh 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 86h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 13h | SSPBUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 14h | SSPCON | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 0000 0000 |
| 94h | SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the SSP in SPI mode.

**FIGURE 10-2: SPI MODE TIMING, MASTER MODE**

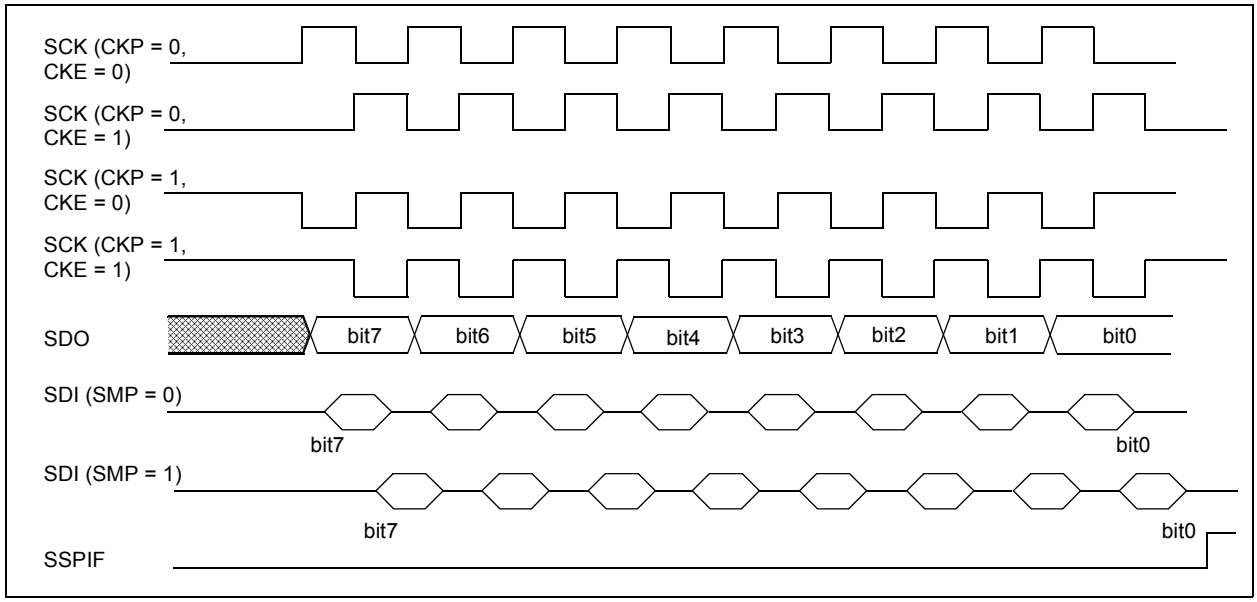



**FIGURE 10-3: SPI MODE TIMING (SLAVE MODE WITH CKE = 0)**

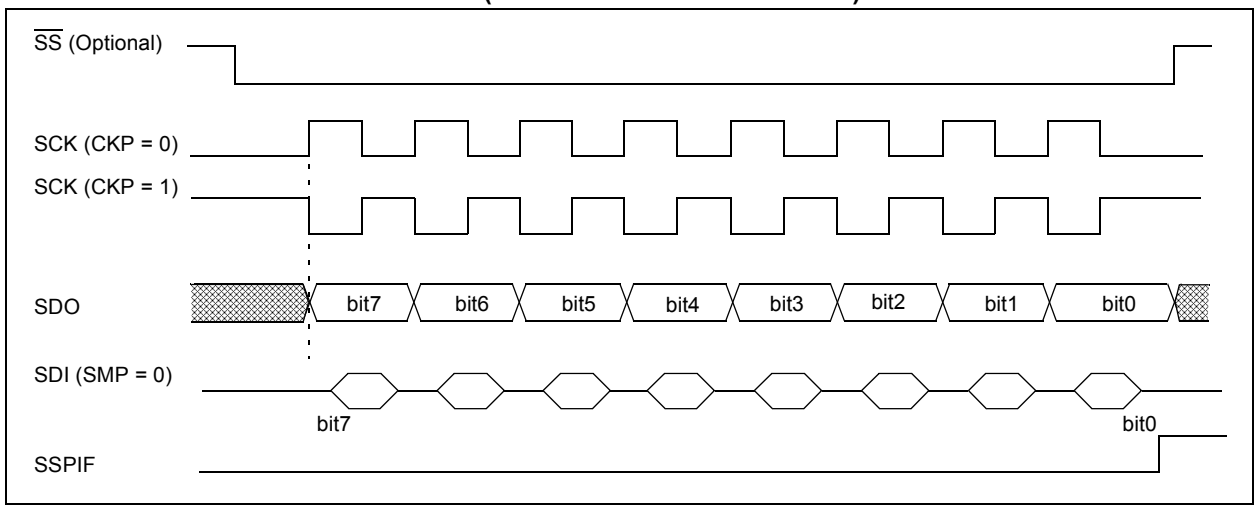
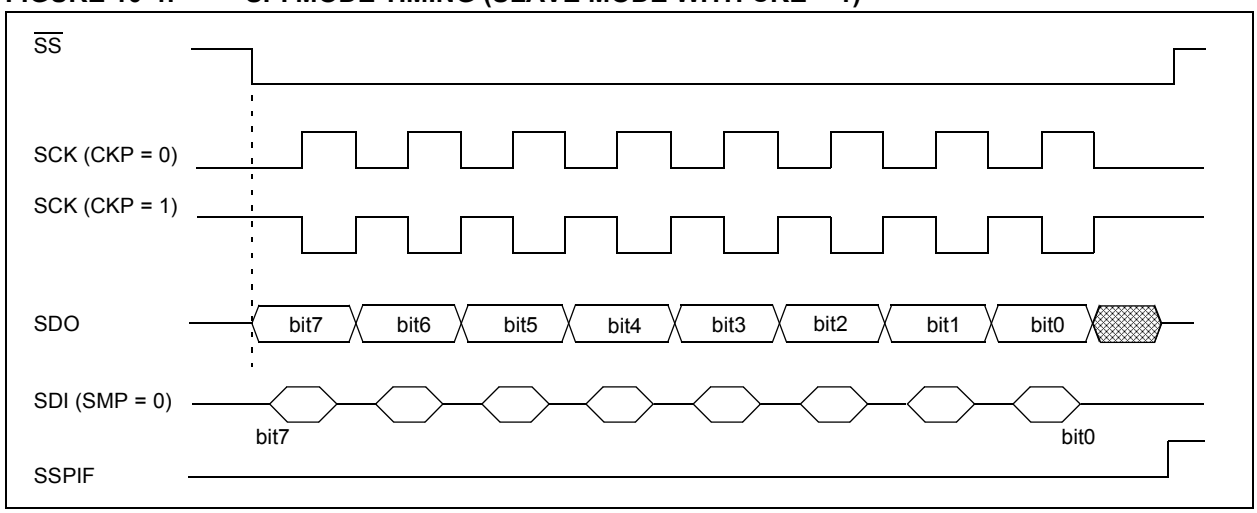



**FIGURE 10-4: SPI MODE TIMING (SLAVE MODE WITH CKE = 1)**



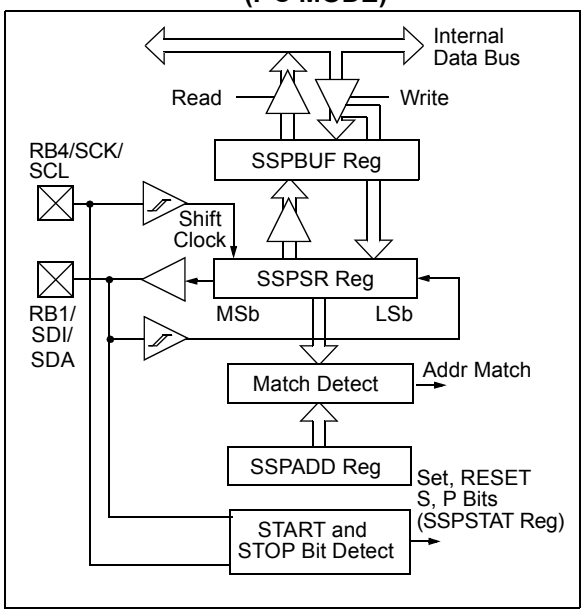
## 10.3 SSP I²C Mode Operation

The SSP module in I²C mode fully implements all slave functions, except general call support, and provides interrupts on START and STOP bits in hardware to facilitate firmware implementations of the master functions. The SSP module implements the Standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer. These are the RB4/SCK/SCL pin, which is the clock (SCL), and the RB1/SDI/SDA pin, which is the data (SDA). The user must configure these pins as inputs or outputs through the TRISB<4,1> bits.

The SSP module functions are enabled by setting SSP Enable bit SSPEN (SSPCON<5>).

**FIGURE 10-5: SSP BLOCK DIAGRAM (I²C MODE)**




The SSP module has five registers for I²C operation:

- SSP Control Register (SSPCON)
- SSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- SSP Shift Register (SSPSR) - Not directly accessible
- SSP Address Register (SSPADD)

The SSPCON register allows control of the I²C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I²C modes to be selected:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address), with START and STOP bit interrupts enabled to support firmware Master mode
- I²C Slave mode (10-bit address), with START and STOP bit interrupts enabled to support firmware Master mode
- I²C Firmware controlled Master operation with START and STOP bit interrupts enabled, Slave is IDLE

Selection of any I²C mode, with the SSPEN bit set, forces the SCL and SDA pins to be open drain, provided these pins are programmed to inputs by setting the appropriate TRISB bits. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I²C module.

Additional information on SSP I²C operation may be found in the PICmicro™ Mid-Range MCU Reference Manual (DS33023).

### 10.3.1 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISB<4,1> set). The SSP module will override the input state with the output data, when required (slave-transmitter).

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) pulse, and then load the SSPBUF register with the received value currently in the SSPSR register.

Either or both of the following conditions will cause the SSP module not to give this $\overline{ACK}$ pulse:

a) The buffer full bit BF (SSPSTAT<0>) was set before the transfer was received.

b) The overflow bit SSPOV (SSPCON<6>) was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit SSPIF (PIR1<3>) is set. Table 10-2 shows what happens when a data transfer byte is received, given the status of bits BF and SSPOV. The shaded cells show the condition where user software did not properly clear the overflow condition. Flag bit BF is cleared by reading the SSPBUF register while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I²C specification, as well as the requirement of the SSP module, are shown in timing parameter #100 and parameter #101.

#### 10.3.1.1 Addressing

Once the SSP module has been enabled, it waits for a START condition to occur. Following the START condition, the eight bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match, and the BF and SSPOV bits are clear, the following events occur:

a) The SSPSR register value is loaded into the SSPBUF register.
b) The buffer full bit, BF is set.
c) An $\overline{ACK}$ pulse is generated.
d) SSP interrupt flag bit, SSPIF (PIR1<3>), is set (interrupt is generated if enabled) - on the falling edge of the ninth SCL pulse.

In 10-bit Address mode, two address bytes need to be received by the slave device. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{W}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '`1111 0 A9 A8 0`', where `A9` and `A8` are the two MSbs of the address.

The sequence of events for 10-bit address is as follows, with steps 7- 9 for slave-transmitter:

1. Receive first (high) byte of address (bits SSPIF, BF, and bit UA (SSPSTAT<1>) are set).
2. Update the SSPADD register with second (low) byte of address (clears bit UA and releases the SCL line).
3. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
4. Receive second (low) byte of address (bits SSPIF, BF, and UA are set).
5. Update the SSPADD register with the first (high) byte of Address, if match releases SCL line, this will clear bit UA.
6. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
7. Receive Repeated START condition.
8. Receive first (high) byte of address (bits SSPIF and BF are set).
9. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.

#### 10.3.1.2 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register.

When the address byte overflow condition exists, then a no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is indicated if either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON<6>) is set.

An SSP interrupt is generated for each data transfer byte. Flag bit SSPIF (PIR1<3>) must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

#### 10.3.1.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit, and pin RB4/SCK/SCL is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then, pin RB4/SCK/SCL should be enabled by setting bit CKP (SSPCON<4>). The master device must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master device by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 10-7).

An SSP interrupt is generated for each data transfer byte. Flag bit SSPIF must be cleared in software, and the SSPSTAT register is used to determine the status of the byte. Flag bit SSPIF is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. When the $\overline{ACK}$ is latched by the slave device, the slave logic is reset (resets SSPSTAT register) and the slave device then monitors for another occurrence of the START bit. If the SDA line was low ($\overline{ACK}$), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then, pin RB4/SCK/SCL should be enabled by setting bit CKP.

**TABLE 10-2: DATA TRANSFER RECEIVED BYTE ACTIONS**

| Status Bits as Data Transfer is Received | | SSPSR → SSPBUF | Generate $\overline{\text{ACK}}$ Pulse | Set bit SSPIF (SSP Interrupt Occurs if Enabled) |
|---|---|---|---|---|
| BF | SSPOV | | | |
| 0 | 0 | Yes | Yes | Yes |
| 1 | 0 | No | No | Yes |
| 1 | 1 | No | No | Yes |
| 0 | 1 | No | No | Yes |

**Note 1:** Shaded cells show the conditions where the user software did not properly clear the overflow condition.

**FIGURE 10-6: I²C WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)**

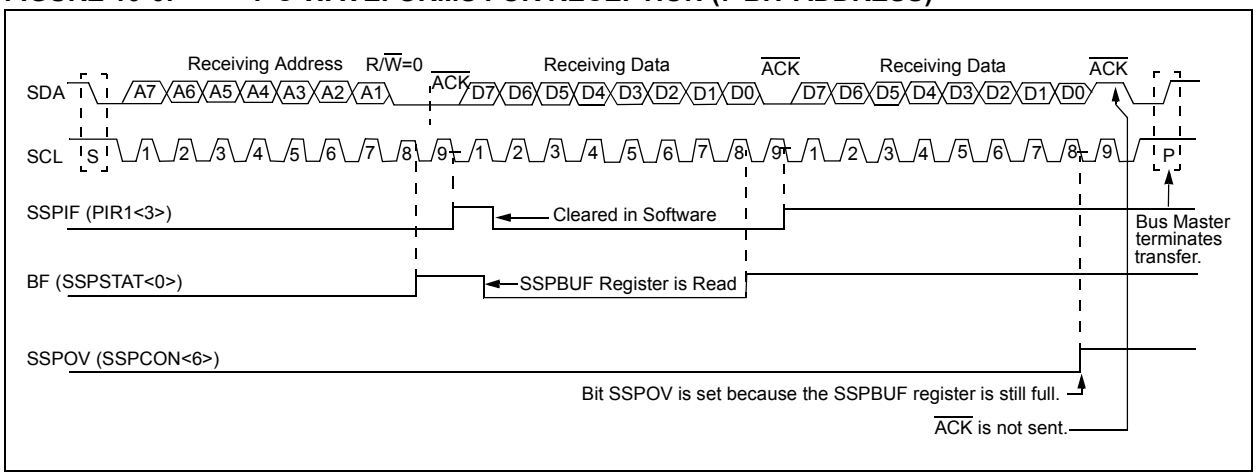
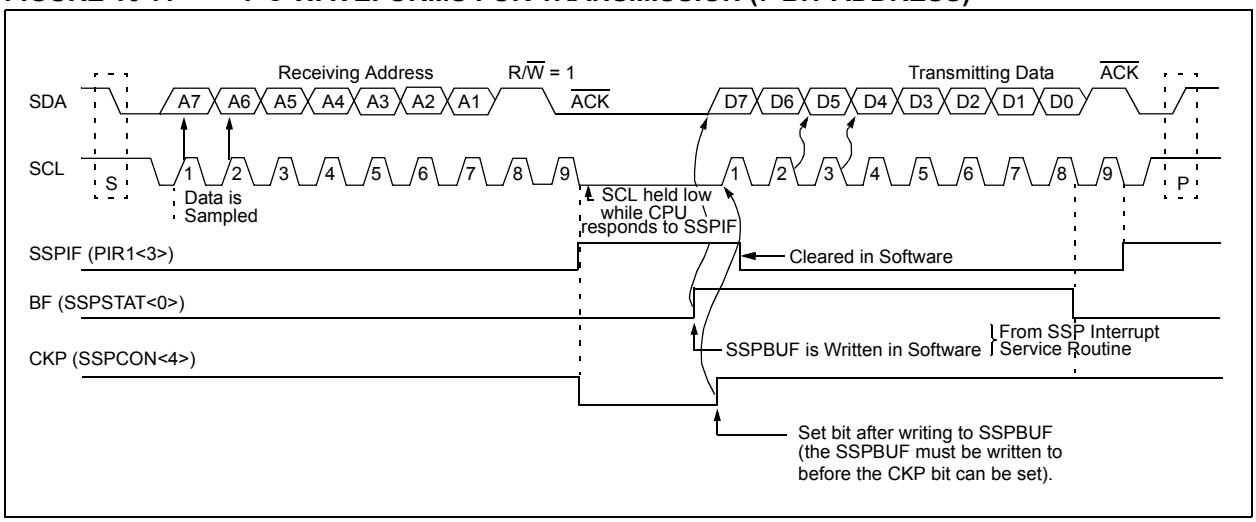



**FIGURE 10-7: I²C WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)**



### 10.3.2 MASTER MODE OPERATION

Master mode operation is supported in firmware using interrupt generation on the detection of the START and STOP conditions. The STOP (P) and START (S) bits are cleared from a RESET, or when the SSP module is disabled. The STOP (P) and START (S) bits will toggle based on the START and STOP conditions. Control of the $I^2C$ bus may be taken when the P bit is set, or the bus is IDLE and both the S and P bits are clear.

In Master mode operation, the SCL and SDA lines are manipulated in firmware by clearing the corresponding TRISB<4,1> bit(s). The output level is always low, irrespective of the value(s) in PORTB<4,1>. So, when transmitting data, a '1' data bit must have the TRISB<1> bit set (input) and a '0' data bit must have the TRISB<1> bit cleared (output). The same scenario is true for the SCL line with the TRISB<4> bit. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the $I^2C$ module.

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP Interrupt if enabled):

- START condition
- STOP condition
- Data transfer byte transmitted/received

Master mode operation can be done with either the Slave mode IDLE (SSPM3:SSPM0 = 1011), or with the Slave mode active. When both Master mode operation and Slave modes are used, the software needs to differentiate the source(s) of the interrupt.

For more information on Master mode operation, see Application Note AN554, *"Software Implementation of $I^2C$ Bus Master"*.

### 10.3.3 MULTI-MASTER MODE OPERATION

In Multi-Master mode operation, 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 SSP module is disabled. The STOP (P) and START (S) bits will toggle based on the START and STOP conditions. Control of the $I^2C$ bus may be taken when bit P (SSPSTAT<4>) is set, or the bus is IDLE and both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the STOP condition occurs.

In Multi-Master mode operation, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRISB<4,1>). There are two stages where this arbitration can be lost:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the Slave device continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to re-transfer the data at a later time.

For more information on Multi-Master mode operation, see Application Note AN578, *"Use of the SSP Module in the of $I^2C$ Multi-Master Environment"*.

**TABLE 10-3: REGISTERS ASSOCIATED WITH $I^2C$ OPERATION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 13h | SSPBUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 93h | SSPADD | Synchronous Serial Port ($I^2C$ mode) Address Register | | | | | | | | 0000 0000 | 0000 0000 |
| 14h | SSPCON | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 0000 0000 |
| 94h | SSPSTAT | SMP(1) | CKE(1) | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |
| 86h | TRISB | PORTB Data Direction register | | | | | | | | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'.
Shaded cells are not used by SSP module in SPI mode.

**Note 1:** Maintain these bits clear in $I^2C$ mode.

**NOTES:**

## 11.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The USART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous - Master (half-duplex)
- Synchronous - Slave (half-duplex)

Bit SPEN (RCSTA<7>) and bits TRISB<5,2> have to be set in order to configure pins RB5/$\overline{SS}$/TX/CK and RB2/SDO/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter.

The USART module also has a multi-processor communication capability, using 9-bit address detection.

**REGISTER 11-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R-1 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D |
| bit 7 | | | | | | | bit 0 |

bit 7 **CSRC:** Clock Source Select bit

Asynchronous mode:
Don't care

Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9**: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN**: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

**Note:** SREN/CREN overrides TXEN in Sync mode.

bit 4 **SYNC**: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

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

bit 2 **BRGH**: High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode

bit 1 **TRMT**: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data, can be Parity bit

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

**REGISTER 11-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-0 | R-0 | R-x |
|---|---|---|---|---|---|---|---|
| SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| bit 7 | | | | | | | bit 0 |

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port enabled (configures RB2/SDO/RX/DT and RB5/$\overline{SS}$/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care
Synchronous mode - Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete
Synchronous mode - Slave:
Don't care

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received, and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D:** 9th bit of Received Data (can be parity bit, but must be calculated by user firmware)

| 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 |

## 11.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free-running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 11-1 shows the formula for computation of the baud rate for different USART modes, which only apply in Master mode (internal clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRG register can be calculated using the formula in Table 11-1. From this, the error in baud rate can be determined.

It may be advantageous to use the high baud rate (BRGH = 1), even for slower baud clocks. This is because the FOSC/(16(X + 1)) equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

### 11.1.1 USART AND INTRC OPERATION

The PIC16F87/88 has an 8 MHz INTRC that can be used as the system clock, thereby eliminating the need for external components to provide the clock source. When the INTRC provides the system clock, the USART module will also use the INTRC as its system clock. Table 11-1 shows some of the INTRC frequencies that can be used to generate the USART's baud rate.

### 11.1.2 LOW POWER MODE OPERATION

The system clock is used to generate the desired baud rate; however, when a Low Power mode is entered, the low power clock source may be operating at a different frequency than in full power execution. In SLEEP mode, no clocks are present. This may require the value in SPBRG to be adjusted.

### 11.1.3 SAMPLING

The data on the RB2/SDO/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

**TABLE 11-1: BAUD RATE FORMULA**

| SYNC | BRGH = 0 (Low Speed) | BRGH = 1 (High Speed) |
|---|---|---|
| 0 | (Asynchronous) Baud Rate = FOSC/(64(X+1)) | Baud Rate = FOSC/(16(X+1)) |
| 1 | (Synchronous) Baud Rate = FOSC/(4(X+1)) | N/A |

X = value in SPBRG (0 to 255)

**TABLE 11-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

**TABLE 11-3: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)**

| BAUD RATE (K) | Fosc = 20 MHz | | | Fosc = 16 MHz | | | Fosc = 10 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | — | — | — |
| 1.2 | 1.221 | +1.75 | 255 | 1.202 | +0.17 | 207 | 1.202 | +0.17 | 129 |
| 2.4 | 2.404 | +0.17 | 129 | 2.404 | +0.17 | 103 | 2.404 | +0.17 | 64 |
| 9.6 | 9.766 | +1.73 | 31 | 9.615 | +0.16 | 25 | 9.766 | +1.73 | 15 |
| 19.2 | 19.531 | + 1.72 | 15 | 19.231 | +0.16 | 12 | 19.531 | +1.72 | 7 |
| 28.8 | 31.250 | +8.51 | 9 | 27.778 | -3.55 | 8 | 31.250 | +8.51 | 4 |
| 33.6 | 34.722 | +3.34 | 8 | 35.714 | +6.29 | 6 | 31.250 | -6.99 | 4 |
| 57.6 | 62.500 | +8.51 | 4 | 62.500 | +8.51 | 3 | 52.083 | -9.58 | 2 |
| HIGH | 1.221 | — | 255 | 0.977 | — | 255 | 0.610 | — | 255 |
| LOW | 312.500 | — | 0 | 250.000 | — | 0 | 156.250 | — | 0 |

| BAUD RATE (K) | Fosc = 4 MHz | | | Fosc = 3.6864 MHz | | |
|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | 0.300 | 0 | 207 | 0.3 | 0 | 191 |
| 1.2 | 1.202 | +0.17 | 51 | 1.2 | 0 | 47 |
| 2.4 | 2.404 | +0.17 | 25 | 2.4 | 0 | 23 |
| 9.6 | 8.929 | +6.99 | 6 | 9.6 | 0 | 5 |
| 19.2 | 20.833 | +8.51 | 2 | 19.2 | 0 | 2 |
| 28.8 | 31.250 | +8.51 | 1 | 28.8 | 0 | 1 |
| 33.6 | — | — | — | — | — | — |
| 57.6 | 62.500 | +8.51 | 0 | 57.6 | 0 | 0 |
| HIGH | 0.244 | — | 255 | 0.225 | — | 255 |
| LOW | 62.500 | — | 0 | 57.6 | — | 0 |

**TABLE 11-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)**

| BAUD RATE (K) | Fosc = 20 MHz | | | Fosc = 16 MHz | | | Fosc = 10 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | — | — | — |
| 1.2 | — | — | — | — | — | — | — | — | — |
| 2.4 | — | — | — | — | — | — | 2.441 | +1.71 | 255 |
| 9.6 | 9.615 | +0.16 | 129 | 9.615 | +0.16 | 103 | 9.615 | +0.16 | 64 |
| 19.2 | 19.231 | +0.16 | 64 | 19.231 | +0.16 | 51 | 19.531 | +1.72 | 31 |
| 28.8 | 29.070 | +0.94 | 42 | 29.412 | +2.13 | 33 | 28.409 | -1.36 | 21 |
| 33.6 | 33.784 | +0.55 | 36 | 33.333 | -0.79 | 29 | 32.895 | -2.10 | 18 |
| 57.6 | 59.524 | +3.34 | 20 | 58.824 | +2.13 | 16 | 56.818 | -1.36 | 10 |
| HIGH | 4.883 | — | 255 | 3.906 | — | 255 | 2.441 | — | 255 |
| LOW | 1250.000 | — | 0 | 1000.000 | — | 0 | 625.000 | — | 0 |

| BAUD RATE (K) | Fosc = 4 MHz | | | Fosc = 3.6864 MHz | | |
|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — |
| 1.2 | 1.202 | +0.17 | 207 | 1.2 | 0 | 191 |
| 2.4 | 2.404 | +0.17 | 103 | 2.4 | 0 | 95 |
| 9.6 | 9.615 | +0.16 | 25 | 9.6 | 0 | 23 |
| 19.2 | 19.231 | +0.16 | 12 | 19.2 | 0 | 11 |
| 28.8 | 27.798 | -3.55 | 8 | 28.8 | 0 | 7 |
| 33.6 | 35.714 | +6.29 | 6 | 32.9 | -2.04 | 6 |
| 57.6 | 62.500 | +8.51 | 3 | 57.6 | 0 | 3 |
| HIGH | 0.977 | — | 255 | 0.9 | — | 255 |
| LOW | 250.000 | — | 0 | 230.4 | — | 0 |

**TABLE 11-5: INTRC BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)**

| BAUD RATE (K) | Fosc = 8 MHz | | | Fosc = 4 MHz | | | Fosc = 2 MHz | | | Fosc = 1 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | NA | — | — | 0.300 | 0 | 207 | 0.300 | 0 | 103 | 0.300 | 0 | 51 |
| 1.2 | 1.202 | +0.16 | 103 | 1.202 | +0.16 | 51 | 1.202 | +0.16 | 25 | 1.202 | +0.16 | 12 |
| 2.4 | 2.404 | +0.16 | 51 | 2.404 | +0.16 | 25 | 2.404 | +0.16 | 12 | 2.232 | -6.99 | 6 |
| 9.6 | 9.615 | +0.16 | 12 | 8.929 | -6.99 | 6 | 10.417 | +8.51 | 2 | NA | — | — |
| 19.2 | 17.857 | -6.99 | 6 | 20.833 | +8.51 | 2 | NA | — | — | NA | — | — |
| 28.8 | 31.250 | +8.51 | 3 | 31.250 | +8.51 | 1 | 31.250 | +8.51 | 0 | NA | — | — |
| 38.4 | 41.667 | +8.51 | 2 | NA | — | — | NA | — | — | NA | — | — |
| 57.6 | 62.500 | +8.51 | 1 | 62.500 | 8.51 | 0 | NA | — | — | NA | — | — |

**TABLE 11-6: INTRC BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)**

| BAUD RATE (K) | Fosc = 8 MHz | | | Fosc = 4 MHz | | | Fosc = 2 MHz | | | Fosc = 1 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | NA | — | — | NA | — | — | NA | — | — | 0.300 | 0 | 207 |
| 1.2 | NA | — | — | 1.202 | +0.16 | 207 | 1.202 | +0.16 | 103 | 1.202 | +0.16 | 51 |
| 2.4 | 2.404 | +0.16 | 207 | 2.404 | +0.16 | 103 | 2.404 | +0.16 | 51 | 2.404 | +0.16 | 25 |
| 9.6 | 9.615 | +0.16 | 51 | 9.615 | +0.16 | 25 | 9.615 | +0.16 | 12 | 8.929 | -6.99 | 6 |
| 19.2 | 19.231 | +0.16 | 25 | 19.231 | +0.16 | 12 | 17.857 | -6.99 | 6 | 20.833 | +8.51 | 2 |
| 28.8 | 29.412 | +2.12 | 16 | 27.778 | -3.55 | 8 | 31.250 | +8.51 | 3 | 31.250 | +8.51 | 1 |
| 38.4 | 38.462 | +0.16 | 12 | 35.714 | -6.99 | 6 | 41.667 | +8.51 | 2 | NA | — | — |
| 57.6 | 55.556 | -3.55 | 8 | 62.500 | +8.51 | 3 | 62.500 | +8.51 | 1 | 62.500 | +8.51 | 0 |

## 11.2 USART Asynchronous Mode

In this mode, the USART uses standard non-return-to-zero (NRZ) format (one START bit, eight or nine data bits, and one STOP bit). The most common data format is 8-bits. An on-chip, dedicated, 8-bit baud rate generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The transmitter and receiver are functionally independent, but use the same data format and baud rate. The baud rate generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during SLEEP.

Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

### 11.2.1 USART ASYNCHRONOUS TRANSMITTER

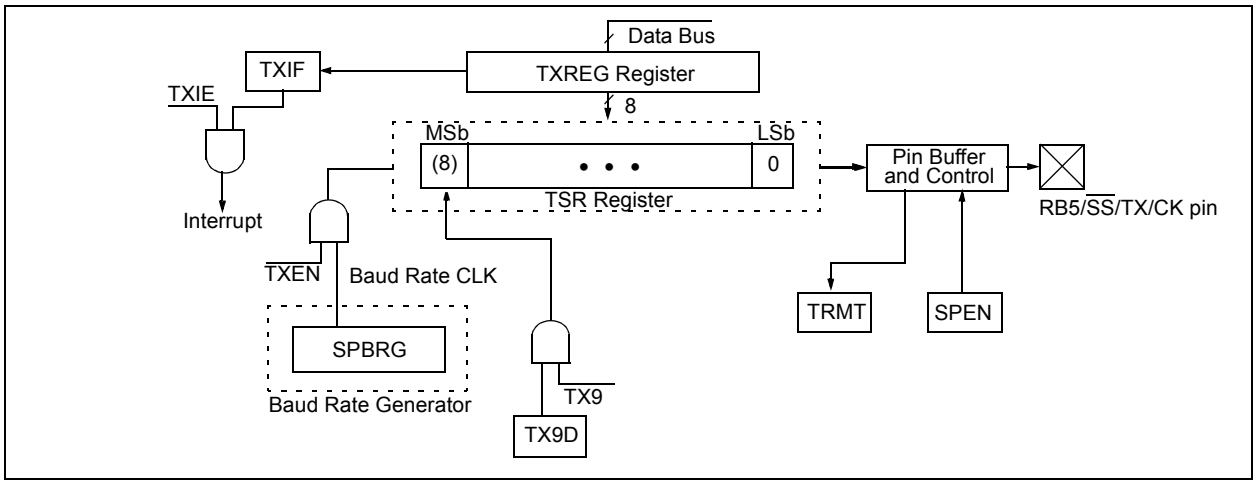
The USART transmitter block diagram is shown in Figure 11-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the read/write transmit buffer, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the STOP bit has been transmitted from the previous load. As soon as the STOP bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit TXIF (PIR1<4>) is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXIE ( PIE1<4>). Flag bit TXIF will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit TRMT is a read only bit, which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

| | |
|---|---|
| **Note 1:** | The TSR register is not mapped in data memory, so it is not available to the user. |
| **2:** | Flag bit TXIF is set when enable bit TXEN is set. TXIF is cleared by loading TXREG. |

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the baud rate generator (BRG) has produced a shift clock (Figure 11-2). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally, when transmission is first started, the TSR register is empty. At that point, transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 11-3). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result, the RB5/$\overline{SS}$/TX/CK pin will revert to hi-impedance.

In order to select 9-bit transmission, transmit bit TX9 (TXSTA<6>) should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit may be loaded in the TSR register.

**FIGURE 11-1: USART TRANSMIT BLOCK DIAGRAM**



When setting up an Asynchronous Transmission, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH (Section 11.1).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set transmit bit TX9.
5. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

**FIGURE 11-2: ASYNCHRONOUS MASTER TRANSMISSION**

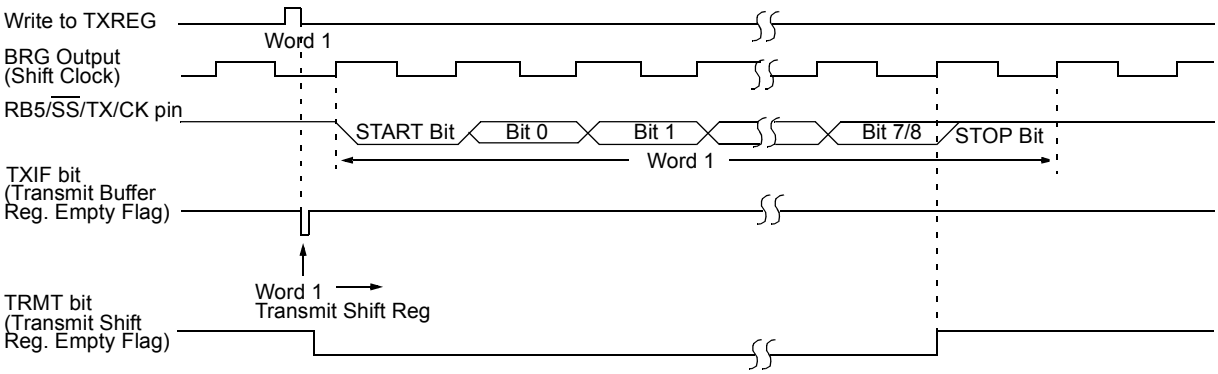
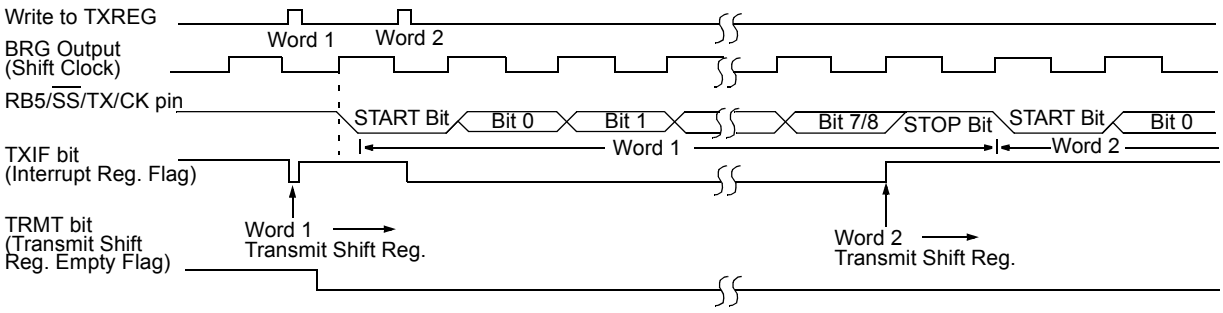



**FIGURE 11-3: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)**




**Note:** This timing diagram shows two consecutive transmissions.

**TABLE 11-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | -000 000x | -000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | — | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| 19h | TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

### 11.2.2 USART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 11-4. The data is received on the RB2/SDO/RX/DT pin and drives the data recovery block. The data recovery block is actually a high speed shifter, operating at x16 times the baud rate; whereas, the main receive serial shifter operates at the bit rate or at FOSC.

Once Asynchronous mode is selected, reception is enabled by setting bit CREN (RCSTA<4>).

The heart of the receiver is the receive (serial) shift register (RSR). After sampling the STOP bit, the received data in the RSR is transferred to the RCREG register (if it is empty). If the transfer is complete, flag bit RCIF (PIR1<5>) is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>). Flag bit RCIF is a read only bit, which is cleared by the hardware. It is cleared when the RCREG register has been read and is empty. The RCREG is a double buffered register (i.e., it is a two-deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting to the RSR register. On the detection of the STOP bit of the third byte, if the RCREG register is still full, the overrun error bit OERR (RCSTA<1>) will be set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Overrun bit OERR has to be cleared in software. This is done by resetting the receive logic (CREN is cleared and then set). If bit OERR is set, transfers from the RSR register to the RCREG register are inhibited, and no further data will be received. It is, therefore, essential to clear error bit OERR if it is set. Framing error bit FERR (RCSTA<2>) is set if a STOP bit is detected as clear. Bit FERR and the 9th receive bit are buffered the same way as the receive data. Reading the RCREG will load bits RX9D and FERR with new values, therefore, it is essential for the user to read the RCSTA register before reading the RCREG register, in order not to lose the old FERR and RX9D information.

**FIGURE 11-4: USART RECEIVE BLOCK DIAGRAM**

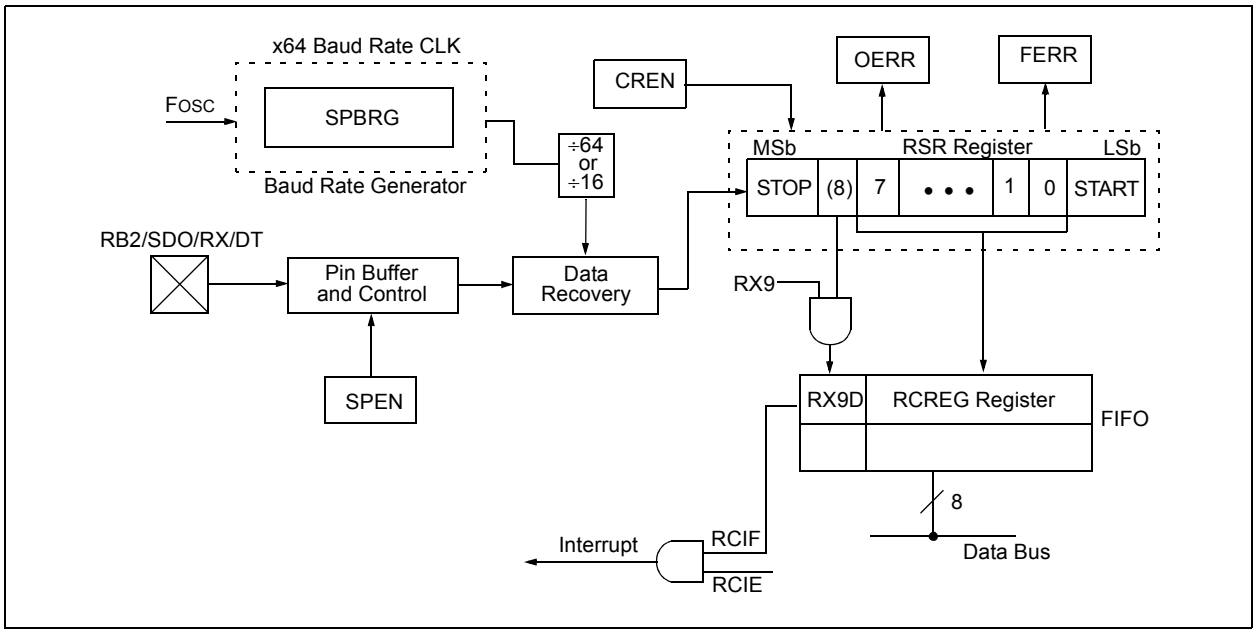
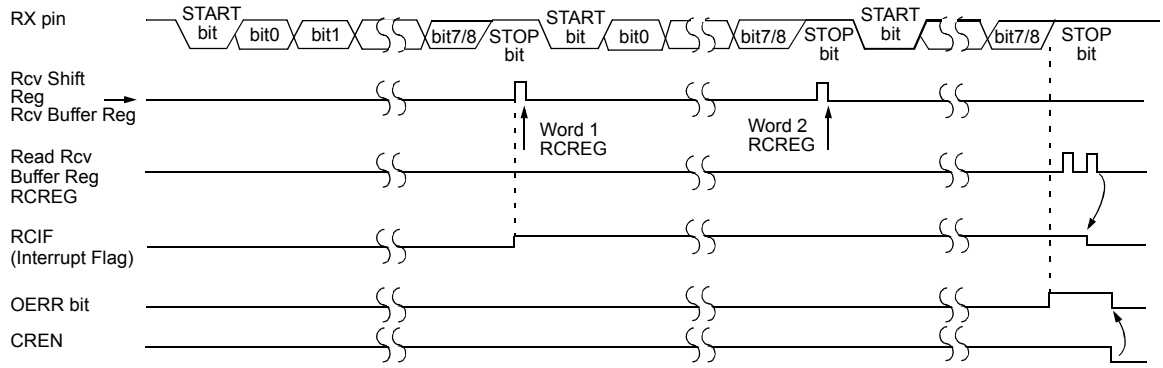


**FIGURE 11-5: ASYNCHRONOUS RECEPTION**




**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.

When setting up an Asynchronous Reception, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH (Section 11.1).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE is set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

**TABLE 11-8: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | — | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| 1Ah | RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

### 11.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

When setting up an Asynchronous Reception with Address Detect Enabled:

- Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH.
- Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
- If interrupts are desired, then set enable bit RCIE.
- Set bit RX9 to enable 9-bit reception.
- Set ADDEN to enable address detect.
- Enable the reception by setting enable bit CREN.
- Flag bit RCIF will be set when reception is complete, and an interrupt will be generated if enable bit RCIE was set.
- Read the RCSTA register to get the ninth bit and determine if any error occurred during reception.
- Read the 8-bit received data by reading the RCREG register, to determine if the device is being addressed.
- If any error occurred, clear the error by clearing enable bit CREN.
- If the device has been addressed, clear the ADDEN bit to allow data bytes and address bytes to be read into the receive buffer, and interrupt the CPU.

**FIGURE 11-6: USART RECEIVE BLOCK DIAGRAM**

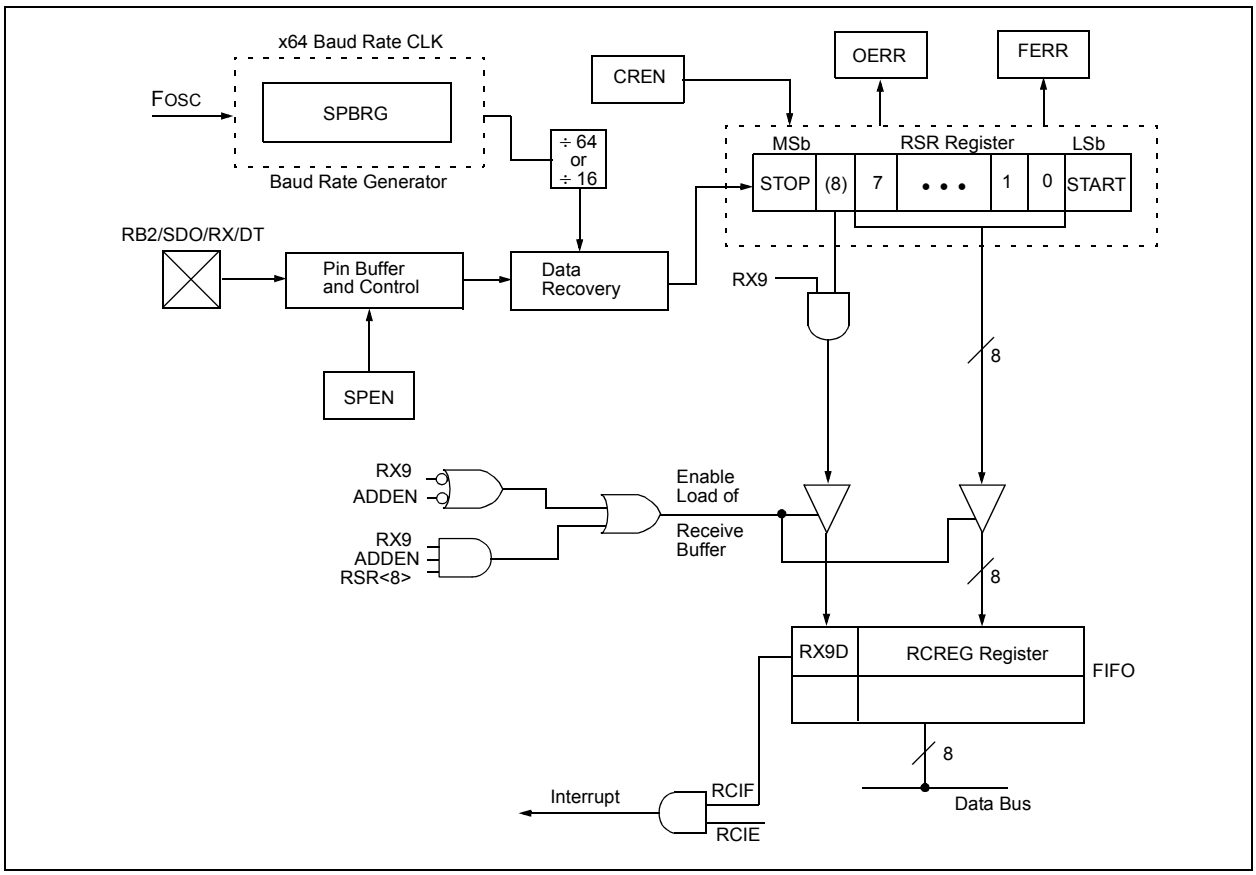
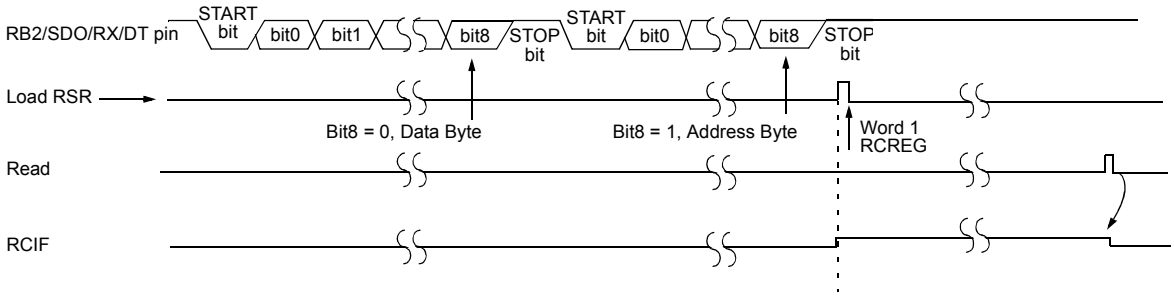


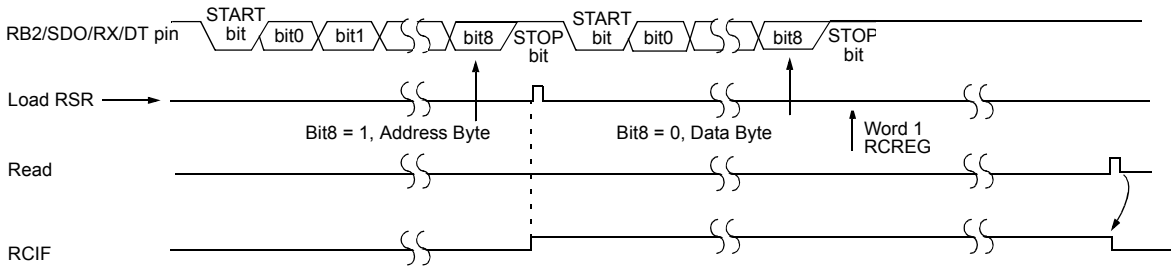

 

**FIGURE 11-7: ASYNCHRONOUS RECEPTION WITH ADDRESS DETECT**




**Note:** This timing diagram shows a data byte followed by an address byte. The data byte is not read into the RCREG (receive buffer) because ADDEN = 1.

**FIGURE 11-8: ASYNCHRONOUS RECEPTION WITH ADDRESS BYTE FIRST**




**Note:** This timing diagram shows a data byte followed by an address byte. The data byte is not read into the RCREG (receive buffer) because ADDEN was not updated and still = 0.

**TABLE 11-9: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 1Ah | RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

## 11.3 USART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the RB5/$\overline{SS}$/TX/CK and RB2/SDO/RX/DT I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTA<7>).

### 11.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 11-6. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the read/write transmit buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and interrupt bit TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. TRMT is a read only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the CK line. Data out is stable around the falling edge of the synchronous clock (Figure 11-9). The transmission can also be started by first loading the TXREG register and then setting bit TXEN (Figure 11-10). This is advantageous when slow baud rates are selected, since the BRG is kept in RESET when bits TXEN, CREN and SREN are clear. Setting enable bit TXEN will start the BRG, creating a shift clock immediately. Normally, when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. The DT and CK pins will revert to hi-impedance. If either bit CREN or bit SREN is set during a transmission, the transmission is aborted and the DT pin reverts to a hi-impedance state (for a reception). The CK pin will remain an output if bit CSRC is set (internal clock). The transmitter logic, however, is not reset, although it is disconnected from the pins. In order to reset the transmitter, the user has to clear bit TXEN. If bit SREN is set (to interrupt an on-going transmission and receive a single word), then after the single word is received, bit SREN will be cleared and the serial port will revert back to transmitting, since bit TXEN is still set. The DT line will immediately switch from Hi-Impedance Receive mode to transmit and start driving. To avoid this, bit TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit should be set and the ninth bit should be written to bit TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG can result in an immediate transfer of the data to the TSR register (if the TSR is empty). If the TSR was empty and the TXREG was written before writing the “new” TX9D, the “present” value of bit TX9D is loaded.

Steps to follow when setting up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (Section 11.1).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

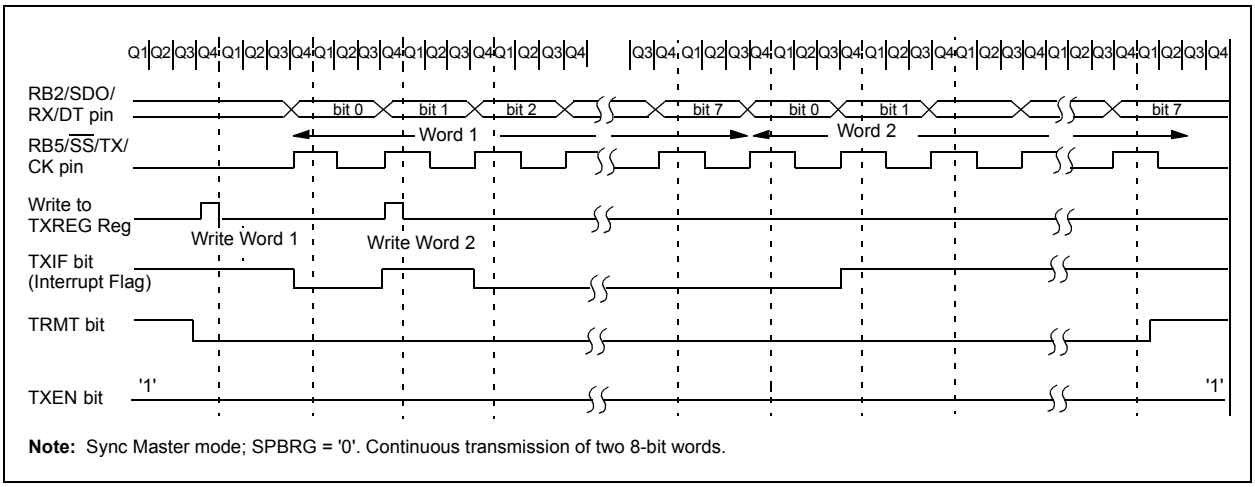
 

**TABLE 11-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | — | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| 19h | TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

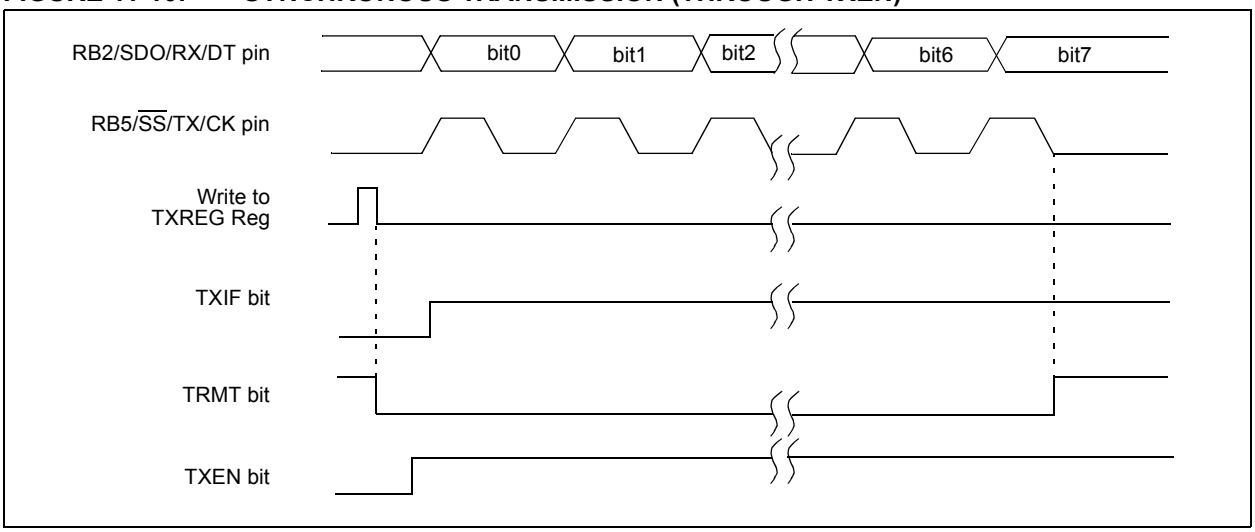
Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

**FIGURE 11-9: SYNCHRONOUS TRANSMISSION**




**Note:** Sync Master mode; SPBRG = '0'. Continuous transmission of two 8-bit words.

**FIGURE 11-10: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



### 11.3.2 USART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either enable bit SREN (RCSTA<5>), or enable bit CREN (RCSTA<4>). Data is sampled on the RB2/SDO/RX/DT pin on the falling edge of the clock. If enable bit SREN is set, then only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set, CREN takes precedence. After clocking the last bit, the received data in the Receive Shift Register (RSR) is transferred to the RCREG register (if it is empty). When the transfer is complete, interrupt flag bit RCIF (PIR1<5>) is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>). Flag bit RCIF is a read only bit, which is reset by the hardware. In this case, it is reset when the RCREG register has been read and is empty. The RCREG is a double buffered register (i.e., it is a two-deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting into the RSR register. On the clocking of the last bit of the third byte, if the RCREG register is still full, then overrun error bit OERR (RCSTA<1>) is set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Bit OERR has to be cleared in software (by clearing bit CREN). If bit OERR is set, transfers from the RSR to the RCREG are inhibited, so it is essential to clear bit OERR if it is set. The ninth receive bit is buffered the same way as the receive data. Reading the RCREG register will load bit RX9D with a new value, therefore, it is essential for the user to read the RCSTA register before reading RCREG, in order not to lose the old RX9D information.

When setting up a Synchronous Master Reception:

1. Initialize the SPBRG register for the appropriate baud rate (Section 11.1).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, then set enable bit RCIE.
5. If 9-bit reception is desired, then set bit RX9.
6. If a single reception is required, set bit SREN. For continuous reception, set bit CREN.
7. Interrupt flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
8. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

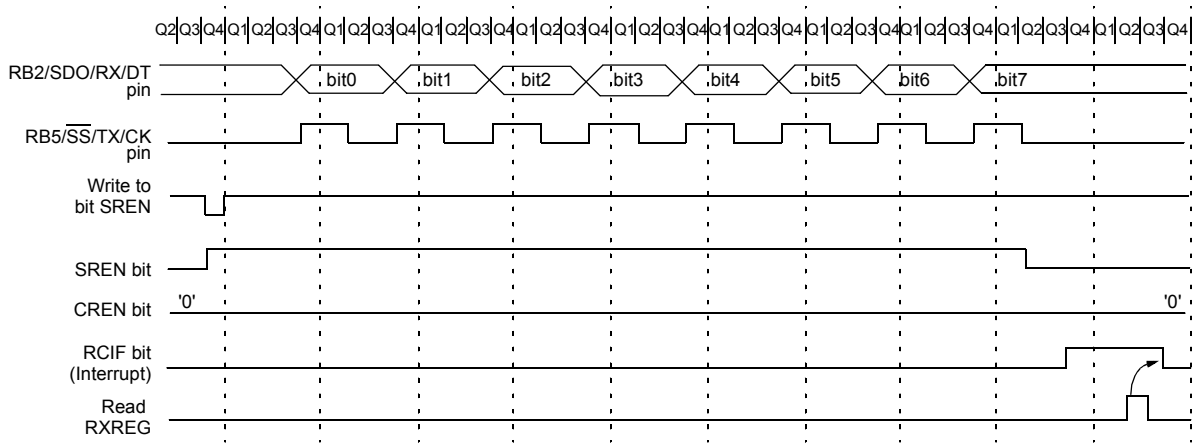
**TABLE 11-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | — | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| 1Ah | RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

**FIGURE 11-11: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**




**Note:** Timing diagram demonstrates Sync Master mode with bit SREN = '1' and bit BRG = '0'.

## 11.4 USART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the RB5/$\overline{SS}$/TX/CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in SLEEP mode. Slave mode is entered by clearing bit CSRC (TXSTA<7>).

### 11.4.1 USART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes is identical, except in the case of the SLEEP mode.

If two words are written to the TXREG and then the `SLEEP` instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREG register.
c) Flag bit TXIF will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will now be set.
e) If enable bit TXIE is set, the interrupt will wake the chip from SLEEP and if the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

When setting up a Synchronous Slave Transmission, follow these steps:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

**TABLE 11-12: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 19h | TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

### 11.4.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the SLEEP mode. Bit SREN is a “don't care” in Slave mode.

If receive is enabled by setting bit CREN prior to the SLEEP instruction, then a word may be received during SLEEP. On completely receiving the word, the RSR register will transfer the data to the RCREG register and if enable bit RCIE bit is set, the interrupt generated will wake the chip from SLEEP. If the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

When setting up a Synchronous Slave Reception, follow these steps:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete and an interrupt will be generated, if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

**TABLE 11-13: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh,18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | R0IF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 18h | RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 1Ah | RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 98h | TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| 99h | SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous slave reception.

## 12.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has seven inputs for 18/20 pin devices (PIC16F88 only).

The conversion of an analog input signal results in a corresponding 10-bit digital number. The A/D module has a high and low voltage reference input that is software selectable to some combination of VDD, VSS, RA2, or RA3.

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 A/D module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- Analog Select Register (ANSEL)

The ADCON0 register, shown in Register 12-2, controls the operation of the A/D module. The ANSEL register, shown in Register 12-1 and the ADCON1 register, shown in Register 12-3, configure the functions of the port pins. The port pins can be configured as analog inputs (RA3/RA2 can also be voltage references) or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

**REGISTER 12-1: ANSEL REGISTER (PIC16F88 DEVICE ONLY)**

| U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | ANS6 | ANS5 | ANS4 | ANS3 | ANS2 | ANS1 | ANS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 6-0 **ANS<6:0>:** Analog Input Select bits

Bits select input function on corresponding AN<6:0> pins

1 = Analog I/O (see notes below)

0 = Digital I/O

**Note 1:** Setting a pin to an analog input disables the digital input buffer. The corresponding TRIS bit should be set to Input mode when using pins as analog inputs. Only AN2 is an analog I/O, all other ANx pins are analog inputs.

**2:** See the Block Diagrams for the Analog I/O pins to see how ANSEL interacts with the CHS bits of the ADCON0 register.

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

**REGISTER 12-2: ADCON0: A/D CONTROL REGISTER 0 (ADDRESS 1Fh)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADCS1 | ADCS0 | CHS2 | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | — | ADON |
| bit 7 | | | | | | | bit 0 |

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits

If ADSC2 = 0:
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

If ADSC2 = 1:
00 = FOSC/4
01 = FOSC/16
10 = FOSC/64
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS<2:0>:** Analog Channel Select bits

000 = Channel 0 (RA0/AN0)
001 = Channel 1 (RA1/AN1)
010 = Channel 2 (RA2/AN2)
011 = Channel 3 (RA3/AN3)
100 = Channel 4 (RA4/AN4)
101 = Channel 5 (RB6/AN5)
110 = Channel 6 (RB7/AN6)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

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

bit 0 **ADON:** A/D On bit

1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

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

**REGISTER 12-3: ADCON1 REGISTER (PIC16F88 DEVICE ONLY)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| ADFM | ADCS2 | VCFG1 | VCFG0 | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

bit 7 **ADFM:** A/D Result Format Select bits

1 = Right justified. Six Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six Least Significant bits of ADRESH are read as '0'.

bit 6 **ADCS2:** A/D Clock Divide by 2 Select bits

1 = A/D clock source is divided by 2 when system clock is used
0 = Disabled

bit 5-4 **VCFG<1:0>:** A/D Voltage Reference Configuration bits

| Logic State | VREF+ | VREF- |
|---|---|---|
| 00 | AVDD | AVSS |
| 01 | AVDD | VREF- |
| 10 | VREF+ | AVSS |
| 11 | VREF+ | VREF- |

**Note:** The ANSEL bits for AN3 and AN2 inputs must be configured as analog inputs for the VREF+ and VREF- external pins to be used.

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

The ADRESH:ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the A/D result register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared, and A/D interrupt flag bit ADIF is set. The block diagram of the A/D module is shown in Figure 12-1.

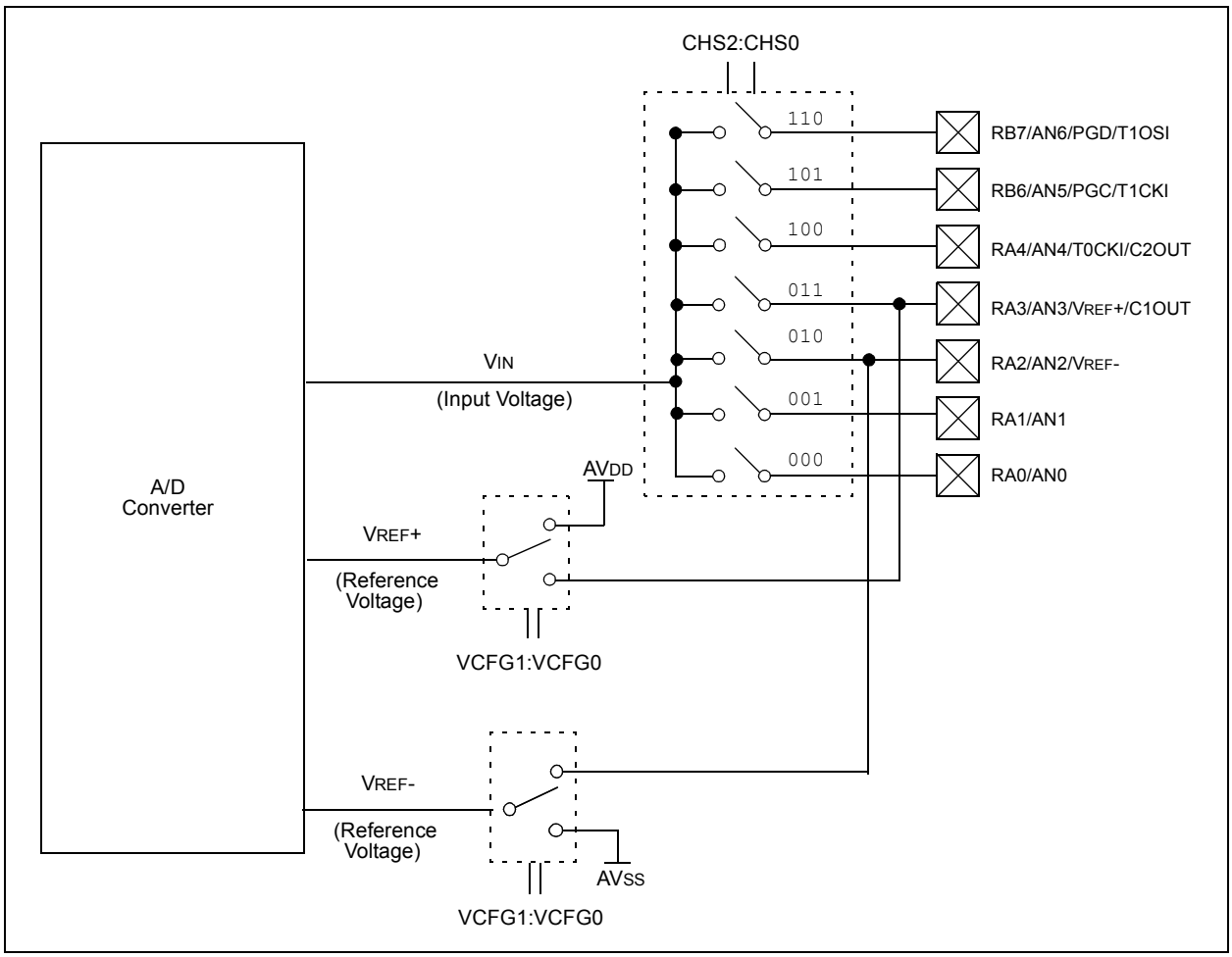
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs.

To determine sample time, see Section 12.1. After this sample time has elapsed the A/D conversion can be started.

These steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
   - Configure analog/digital I/O (ANSEL)
   - Configure voltage reference (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D conversion clock (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared (with interrupts disabled); OR
   - Waiting for the A/D interrupt
6. Read A/D Result register pair (ADRESH:ADRESL), clear bit ADIF if required.
7. For next conversion, go to step 1 or step 2 as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2 TAD is required before the next acquisition starts.

**FIGURE 12-1: A/D BLOCK DIAGRAM**



## 12.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 12-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), see Figure 12-2. **The maximum recommended impedance for analog sources is 2.5 kΩ.** As the impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 12-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

To calculate the minimum acquisition time, TACQ, see the PICmicro™ Mid-Range Reference Manual (DS33023).

**EQUATION 12-1: ACQUISITION TIME**

$$
\begin{aligned}
T_{ACQ} &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\
&= T_{AMP} + T_C + T_{COFF} \\
&= 2\ \mu s + T_C + [(\text{Temperature} - 25°C)(0.05\ \mu s/°C)] \\
T_C &= C_{HOLD}(R_{IC} + R_{SS} + R_S)\ \ln(1/2047) \\
&= -120\ pF\ (1\ k\Omega + 7\ k\Omega + 10\ k\Omega)\ \ln(0.0004885) \\
&= 16.47\ \mu s \\
T_{ACQ} &= 2\ \mu s + 16.47\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C) \\
&= 19.72\ \mu s
\end{aligned}
$$

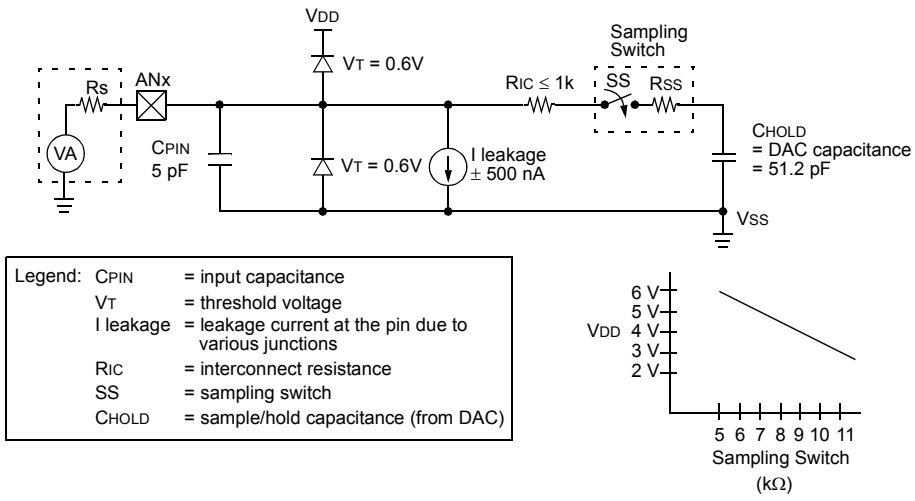
**Note 1:** The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

**2:** The charge holding capacitor (CHOLD) is not discharged after each conversion.

**3:** The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

**4:** After a conversion has completed, a 2.0 TAD delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

**FIGURE 12-2: ANALOG INPUT MODEL**



## 12.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 9.0 TAD per 8-bit conversion. The source of the A/D conversion clock is software selectable. The seven possible options for TAD are:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal A/D module RC oscillator (2 - 6 μs)

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time as small as possible, but no less than 1.6 μs and not greater than 6.4 μs.

Table 12-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

## 12.3 Configuring Analog Port Pins

The ADCON1, ANSEL, TRISA, and TRISB registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS<2:0> bits and the TRIS bits.

**Note 1:** When reading the port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

**2:** Analog levels on any pin that is defined as a digital input (including the RA4:RA0 and RB7:RB6 pins), may cause the input buffer to consume current out of the device specification.

**TABLE 12-1: TAD vs. MAXIMUM DEVICE OPERATING FREQUENCIES (STANDARD DEVICES (C))**

| AD Clock Source (TAD) | | | Maximum Device Frequency |
|---|---|---|---|
| **Operation** | **ADCS<2>** | **ADCS<1:0>** | **Max.** |
| 2 TOSC | 0 | 00 | 1.25 MHz |
| 4 TOSC | 1 | 00 | 2.5 MHz |
| 8 TOSC | 0 | 01 | 5 MHz |
| 16 TOSC | 1 | 01 | 10 MHz |
| 32 TOSC | 0 | 10 | 20 MHz |
| 64 TOSC | 1 | 10 | 20 MHz |
| RC(1,2,3) | X | 11 | **(Note 1)** |

**Note 1:** The RC source has a typical TAD time of 4 μs, but can vary between 2 - 6 μs.

**2:** When the device frequencies are greater than 1 MHz, the RC A/D conversion clock source is only recommended for SLEEP operation.

**3:** For extended voltage devices (LF), please refer to the Electrical Characteristics (Section 18.0 and Section 18.4).

## 12.4 A/D Conversions

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D result register pair will NOT be updated with the partially completed A/D conversion sample. That is, the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers). After the A/D conversion is aborted, a 2 TAD wait is required before the next acquisition is started. After this 2 TAD wait, acquisition on the selected channel is automatically started. The GO/$\overline{\text{DONE}}$ bit can then be set to start the conversion.

In Figure 12-3, after the GO bit is set, the first time segment has a minimum of TCY and a maximum of TAD.

| **Note:** | The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D. |
|---|---|

### 12.4.1 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16-bits wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D Format Select bit (ADFM) controls this justification. Figure 12-4 shows the operation of the A/D result justification. The extra bits are loaded with '0's. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

**FIGURE 12-3: A/D CONVERSION TAD CYCLES**

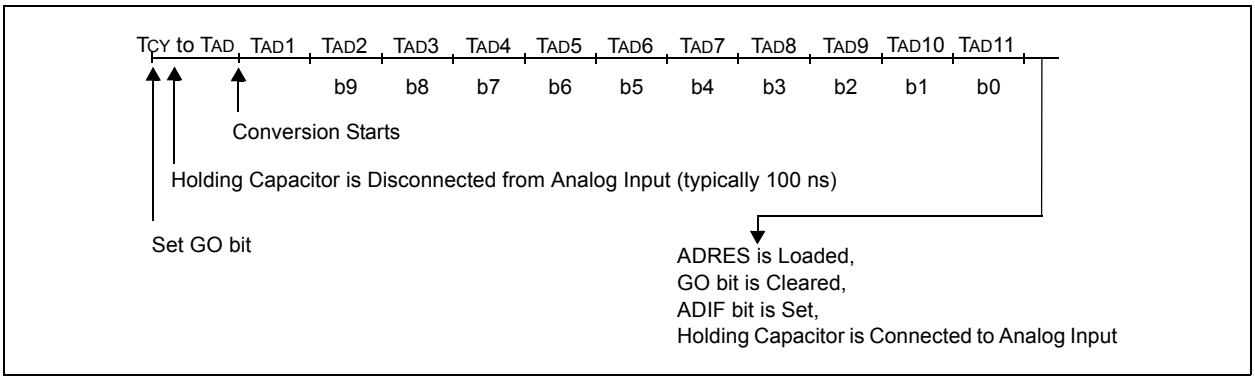
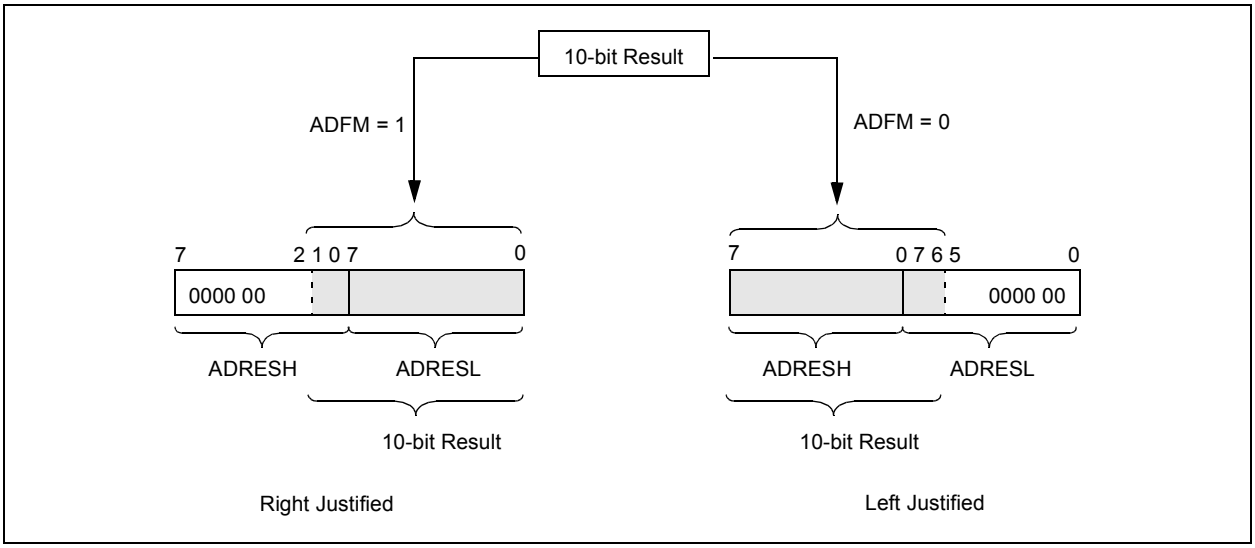



**FIGURE 12-4: A/D RESULT JUSTIFICATION**



## 12.5 A/D Operation During SLEEP

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/DONE bit will be cleared and the result loaded into the ADRES register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

| Note: | For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To perform an A/D conversion in SLEEP, ensure the SLEEP instruction immediately follows the instruction that sets the GO/DONE bit. |
|---|---|

## 12.6 Effects of a RESET

A device RESET forces all registers to their RESET state. The A/D module is disabled and any conversion in progress is aborted. All A/D input pins are configured as analog inputs.

The value that is in the ADRESH:ADRESL registers is not modified for a Power-on Reset. The ADRESH:ADRESL registers will contain unknown data after a Power-on Reset.

## 12.7 Use of the CCP Trigger

An A/D conversion can be started by the "special event trigger" of the CCP module. This requires that the CCP1M3:CCP1M0 bits (CCP1CON<3:0>) be programmed as '1011' and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/DONE bit will be set, starting the A/D conversion and the Timer1 counter will be reset to zero. Timer1 is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving the ADRESH:ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition done before the "special event trigger" sets the GO/DONE bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), then the "special event trigger" will be ignored by the A/D module, but will still reset the Timer1 counter.

**TABLE 12-2: REGISTERS/BITS ASSOCIATED WITH A/D**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh 10Bh, 18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | -000 0000 | -000 0000 |
| 8Ch | PIE1 | — | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | -000 0000 | -000 0000 |
| 1Eh | ADRESH(1) | A/D Result Register High Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 9Eh | ADRESL(1) | A/D Result Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 1Fh | ADCON0(1) | ADCS1 | ADCS0 | CHS2 | CHS1 | CHS0 | GO/DONE | — | ADON | 0000 00-0 | 0000 00-0 |
| 9Fh | ADCON1(1) | ADFM | ADCS2 | VCFG1 | VCFG0 | — | — | — | — | 0000 ---- | 0000 ---- |
| 9Bh | ANSEL(1) | — | AN6 | AN5 | AN4 | AN3 | AN2 | AN1 | AN0 | -111 1111 | -111 1111 |
| 05h | PORTA-87<br>PORTA-88 | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xxxx 0000<br>xxx0 0000 | uuuu 0000<br>uuu0 0000 |
| 05h, 106h | PORTB-87<br>PORTB-88 | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx<br>00xx xxxx | uuuu uuuu<br>00uu uuuu |
| 85h | TRISA | TRISA7 | TRISA6 | TRISA5(1) | PORTA Data Direction Register | | | | | 1111 1111 | 1111 1111 |
| 86h, 186h | TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

**Note 1:** PIC16F88 only.

**2:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

## 13.0 COMPARATOR MODULE

The comparator module contains two analog comparators. The inputs to the comparators are multiplexed with I/O port pins RA0 through RA3, while the outputs are multiplexed to pins RA3 and RA4. The on-chip Voltage Reference (Section 14.0) can also be an input to the comparators.

The CMCON register (Register 13-1) controls the comparator input and output multiplexers. A block diagram of the various comparator configurations is shown in Figure 13-1.

**REGISTER 13-1: CMCON REGISTER**

| R-0 | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **C2OUT:** Comparator 2 Output bit
When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
When C2INV = 1:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-

bit 6 **C1OUT:** Comparator 1 Output bit
When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
When C1INV = 1:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-

bit 5 **C2INV:** Comparator 2 Output Inversion bit
1 = C2 output inverted
0 = C2 output not inverted

bit 4 **C1INV:** Comparator 1 Output Inversion bit
1 = C1 output inverted
0 = C1 output not inverted

bit 3 **CIS:** Comparator Input Switch bit
When CM2:CM0 = 001:
1 = C1 VIN- connects to RA3
0 = C1 VIN- connects to RA0
When CM2:CM0 = 010:
1 = C1 VIN- connects to RA3
C2 VIN- connects to RA2
0 = C1 VIN- connects to RA0
C2 VIN- connects to RA1

bit 2-0 **CM<2:0>:** Comparator Mode bits

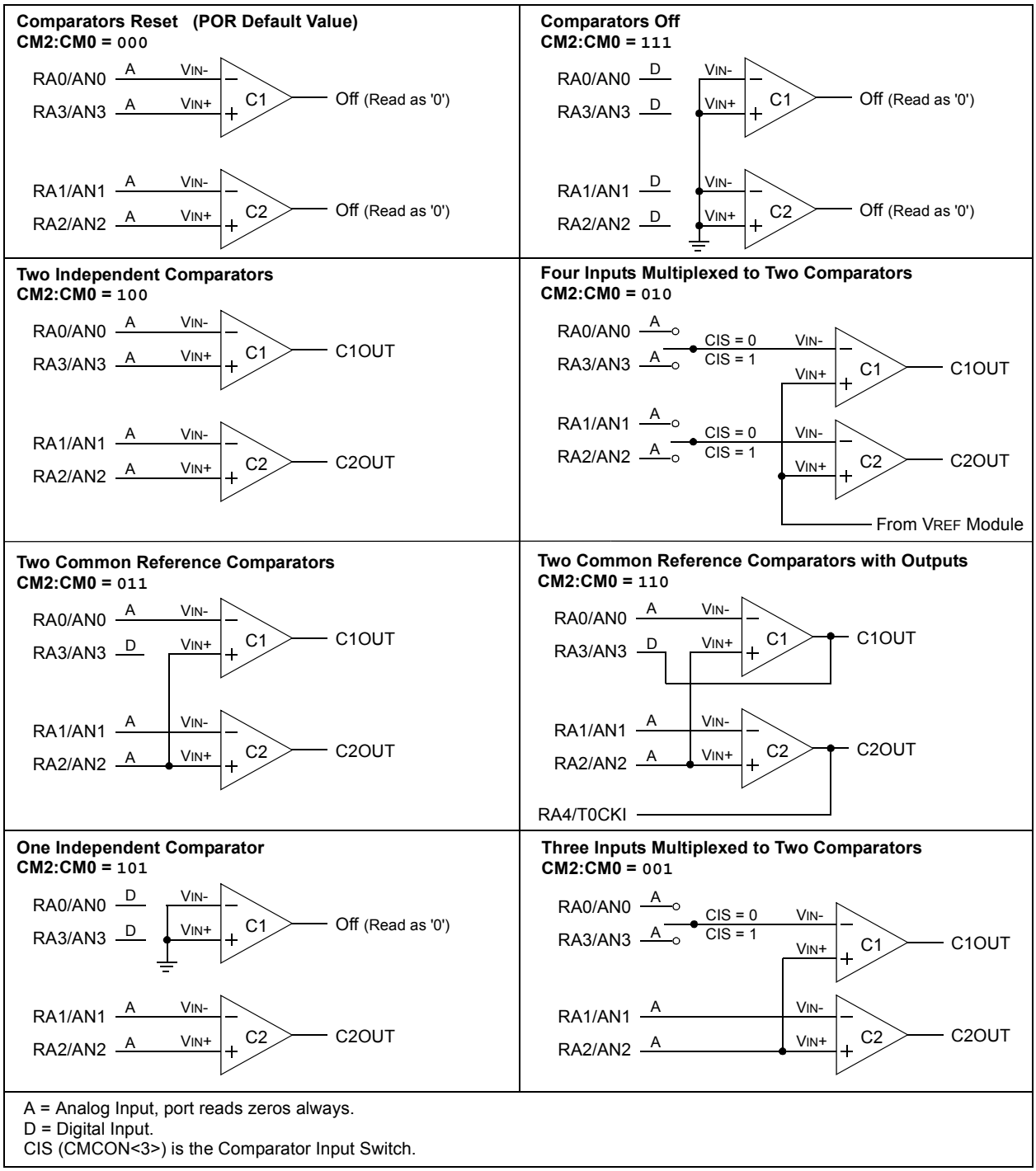
| Legend: | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| - n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

## 13.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 13-1 shows the eight possible modes. The TRISA register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in the Electrical Specifications (Section 18.0).

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

**FIGURE 13-1: COMPARATOR I/O OPERATING MODES**




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

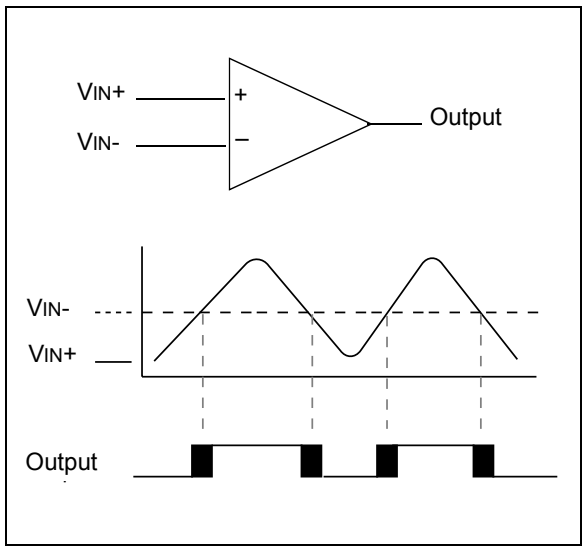
## 13.2 Comparator Operation

A single comparator is shown in Figure 13-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 13-2 represent the uncertainty due to input offsets and response time.

## 13.3 Comparator Reference

An external or internal reference signal may be used depending on the Comparator Operating mode. The analog signal present at VIN- is compared to the signal at VIN+, and the digital output of the comparator is adjusted accordingly (Figure 13-2).

**FIGURE 13-2: SINGLE COMPARATOR**




### 13.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same, or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD, and can be applied to either pin of the comparator(s).

### 13.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference for the comparators. Section 14.0 contains a detailed description of the Comparator Voltage Reference module that provides this signal. The internal reference signal is used when comparators are in mode CM<2:0> = 110 (Figure 13-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

## 13.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise, the maximum delay of the comparators should be used (Section 18.0).

## 13.5 Comparator Outputs

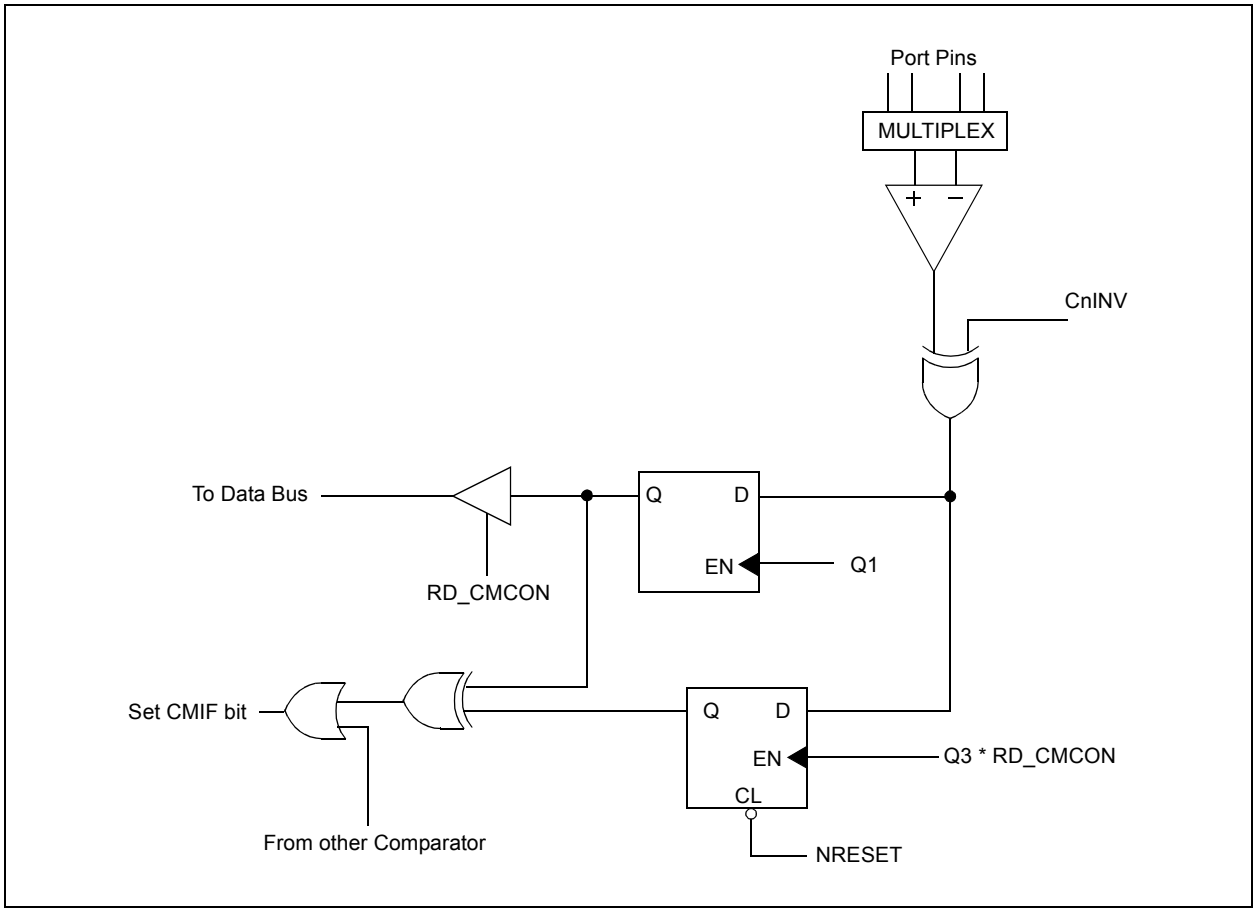
The comparator outputs are read through the CMCON register. These bits are read only. The comparator outputs may also be directly output to the RA3 and RA4 I/O pins. When enabled, multiplexors in the output path of the RA3 and RA4 pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 13-3 shows the comparator output block diagram.

The TRISA bits will still function as an output enable/disable for the RA3 and RA4 pins while in this mode.

The polarity of the comparator outputs can be changed using the C2INV and C1INV bits (CMCON<4:5>).

| | |
|---|---|
| **Note 1:** | When reading the PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert an analog input, according to the Schmitt Trigger input specification. |
| **2:** | Analog levels, on any pin defined as a digital input, may cause the input buffer to consume more current than is specified. |

**FIGURE 13-3: COMPARATOR OUTPUT BLOCK DIAGRAM**




## 13.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR registers) is the comparator interrupt flag. The CMIF bit must be reset by clearing it ('0'). Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

The CMIE bit (PIE registers) and the PEIE bit (INTCON register) must be set to enable the interrupt. In addition, the GIE bit must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

| **Note:** | If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR registers) interrupt flag may not get set. |
|---|---|

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit CMIF.

A mismatch condition will continue to set flag bit CMIF. Reading CMCON will end the mismatch condition, and allow flag bit CMIF to be cleared.

## 13.7 Comparator Operation During SLEEP

When a comparator is active and the device is placed in SLEEP mode, the comparator remains active and the interrupt is functional, if enabled. This interrupt will wake-up the device from SLEEP mode when enabled. While the comparator is powered up, higher SLEEP currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in SLEEP mode, turn off the comparators, CM<2:0> = 111, before entering SLEEP. If the device wakes up from SLEEP, the contents of the CMCON register are not affected.

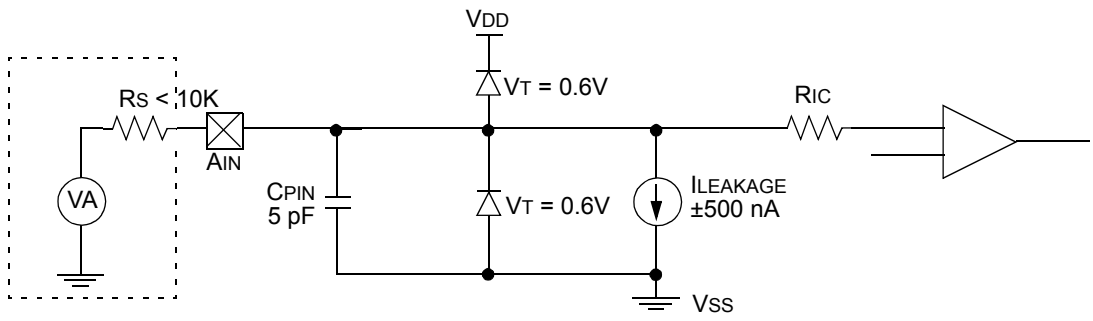
## 13.8 Effects of a RESET

A device RESET forces the CMCON register to its RESET state, causing the comparator module to be in the Comparator Off mode, CM<2:0> = 111.

## 13.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 13-4. 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 latchup 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 13-4: 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 13-1: REGISTERS ASSOCIATED WITH THE COMPARATOR MODULE**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 9Ch | CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 0000 0111 |
| 9Dh | CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 000- 0000 |
| 0Bh, 8Bh, 10Bh, 18Bh | INTCON | GIE | PEIE | TMR0IE | INTIE | RBIE | TMR0IF | INTIF | RBIF | 0000 000x | 0000 000u |
| 0Dh | PIR2 | OSFIF | CMIF | — | EEIF | — | — | — | — | 00-0 ---- | 00-0 ---- |
| 8Dh | PIE2 | OSFIE | CMIE | — | EEIE | — | — | — | — | 00-0 ---- | 00-0 ---- |
| 05h | PORTA-87<br>PORTA-88 | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xxxx 0000<br>xxx0 0000 | uuuu 0000<br>uuu0 0000 |
| 85h | TRISA | TRISA7 | TRISA6 | TRISA5(1) | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are unused by the comparator module.

**Note 1:** Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

## 14.0 COMPARATOR VOLTAGE REFERENCE MODULE

The Comparator Voltage Reference Generator is a 16-tap resistor ladder network that provides a fixed voltage reference when the comparators are in mode '110'. A programmable register controls the function of the reference generator. Register 14-1 lists the bit functions of the CVRCON register.

As shown in Figure 14-1, the resistor ladder is segmented to provide two ranges of $CV_{REF}$ values and has a power-down function to conserve power when the reference is not being used. The comparator reference supply voltage (also referred to as $CV_{RSRC}$) comes directly from $V_{DD}$. It should be noted, however, that the voltage at the top of the ladder is $CV_{RSRC} - V_{SAT}$, where $V_{SAT}$ is the saturation voltage of the power switch transistor. This reference will only be as accurate as the values of $CV_{RSRC}$ and $V_{SAT}$.

The output of the reference generator may be connected to the RA2/AN2/$V_{REF}$-/$CV_{REF}$ pin. This can be used as a simple D/A function by the user, if a very high impedance load is used. The primary purpose of this function is to provide a test path for testing the reference generator function.

**REGISTER 14-1: CVRCON CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **CVREN:** Comparator Voltage Reference Enable bit
1 = $CV_{REF}$ circuit powered on
0 = $CV_{REF}$ circuit powered down

bit 6 **CVROE:** Comparator $V_{REF}$ Output Enable bit
1 = $CV_{REF}$ voltage level is output on RA2/AN2/$V_{REF}$-/$CV_{REF}$ pin
0 = $CV_{REF}$ voltage level is disconnected from RA2/AN2/$V_{REF}$-/$CV_{REF}$ pin

bit 5 **CVRR:** Comparator $V_{REF}$ Range Selection bit
1 = 0.00 $CV_{RSRC}$ to 0.75 $CV_{RSRC}$, with $CV_{RSRC}$/24 step size
0 = 0.25 $CV_{RSRC}$ to 0.75 $CV_{RSRC}$, with $CV_{RSRC}$/32 step size

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

bit 3-0 **CVR<3:0>:** Comparator $V_{REF}$ Value Selection $0 \le VR3:VR0 \le 15$ bits
When CVRR = 1:
$CV_{REF} = (VR<3:0>/ 24) \bullet (CV_{RSRC})$
When CVRR = 0:
$CV_{REF} = 1/4 \bullet (CV_{RSRC}) + (VR3:VR0/ 32) \bullet (CV_{RSRC})$

| 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 |

**FIGURE 14-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM**

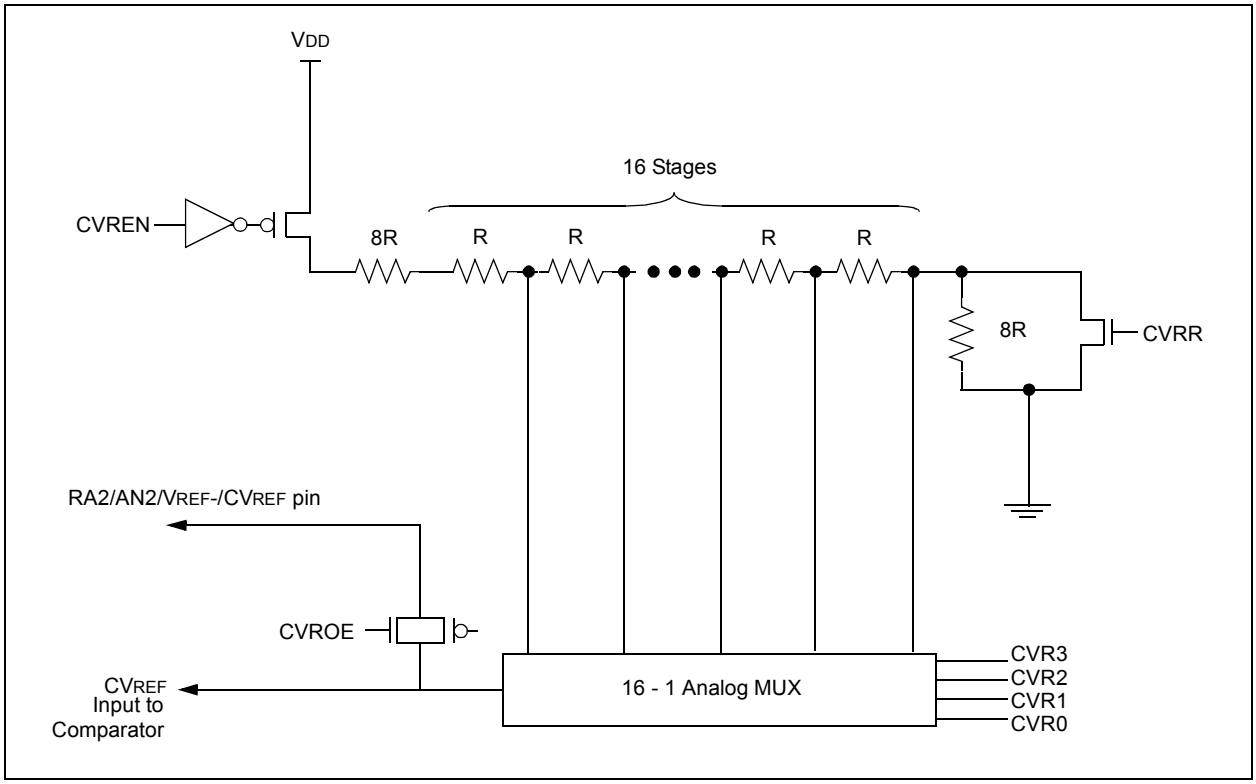



**TABLE 14-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 9Dh | CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 000- 0000 |
| 9Ch | CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0000 | 0000 0000 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are not used with the comparator voltage reference.

## 15.0 SPECIAL FEATURES OF THE CPU

These devices have a host of features intended to maximize system reliability, minimize cost through elimination of external components, provide Power Saving Operating modes and offer code protection:

- RESET
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Two-Speed Start-up
- Fail-Safe Clock Monitor
- SLEEP
- Code Protection
- ID Locations
- In-Circuit Serial Programming

There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in RESET until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only. It is designed to keep the part in RESET while the power supply stabilizes, and is enabled or disabled using a configuration bit. With these two timers on-chip, most applications need no external RESET circuitry.

SLEEP mode is designed to offer a very low current Power-down mode. The user can wake-up from SLEEP through external RESET, Watchdog Timer Wake-up, or through an interrupt.

Several oscillator options are also made available to allow the part to fit the application. The RC oscillator option saves system cost while the LP crystal option saves power. Configuration bits are used to select the desired Oscillator mode.

Additional information on special features is available in the PICmicro® Mid-Range Reference Manual (DS33023).

### 15.1 Configuration Bits

The configuration bits can be programmed (read as ‘0’), or left unprogrammed (read as ‘1’), to select various device configurations. These bits are mapped in program memory locations 2007h and 2008h.

The user will note that address 2007h is beyond the user program memory space, which can be accessed only during programming.

## REGISTER 15-1: CONFIGURATION WORD 1 REGISTER (ADDRESS 2007h)

| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CP | CCPMX | RESV | WRT1 | WRT0 | CPD | LVP | BOREN | MCLRE | FOSC2 | $\overline{\text{PWRTEN}}$ | WDTEN | FOSC1 | FOSC0 |
| bit 13 | | | | | | | | | | | | | bit 0 |

bit 13 **CP:** FLASH Program Memory Code Protection bits
1 = Code protection off
0 = 0000h to 0FFFh code protected (All protected)

bit 12 **CCPMX:** CCP1 Pin Selection bit
1 = CCP1 function on RB0
0 = CCP1 function on RB3

bit 11 **DEBUG:** In-Circuit Debugger Mode bit
1 = In-Circuit Debugger disabled, RB6 and RB7 are general purpose I/O pins
0 = In-Circuit Debugger enabled, RB6 and RB7 are dedicated to the debugger

bit 10-9 **WRT<1:0>:** FLASH Program Memory Write Enable bits
11 = Write protection off
10 = 0000h to 00FFh write protected, 0100h to 0FFFh may be modified by EECON control
01 = 0000h to 07FFh write protected, 0800h to 0FFFh may be modified by EECON control
00 = 0000h to 0FFFh write protected

bit 8 **CPD:** Data EE Memory Code Protection bit
1 = Code protection off
0 = Data EE memory code protected

bit 7 **LVP:** Low Voltage Programming Enable bit
1 = RB3/PGM pin has PGM function, low voltage programming enabled
0 = RB3 is digital I/O, HV on $\overline{\text{MCLR}}$ must be used for programming

bit 6 **BOREN:** Brown-out Reset Enable bit
1 = BOR enabled
0 = BOR disabled

bit 5 **MCLRE:** RA5/$\overline{\text{MCLR}}$ Pin Function Select bit
1 = RA5/$\overline{\text{MCLR}}$ pin function is $\overline{\text{MCLR}}$
0 = RA5/$\overline{\text{MCLR}}$ pin function is digital I/O, $\overline{\text{MCLR}}$ internally tied to $V_{DD}$

bit 3 **$\overline{\text{PWRTEN}}$:** Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 2 **WDTEN:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 4, 1-0 **FOSC<2:0>:** Oscillator Selection bits
111 = EXTRC oscillator; CLKO function on RA6/OSC2/CLKO
110 = EXTRC oscillator; Port I/O function on RA6/OSC2/CLKO
101 = INTRC oscillator; CLKO function on RA6/OSC2/CLKO
100 = INTRC oscillator; Port I/O function on RA6/OSC2/CLKO
011 = EXTCLK; Port I/O function on RA6/OSC2/CLKO
010 = HS oscillator
001 = XT oscillator
000 = LP oscillator

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

**REGISTER 15-2: CONFIGURATION WORD 2 REGISTER (ADDRESS 2008h)**

| U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | U-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — | — | — | — | — | IESO | FCMEN |
| bit 13 | | | | | | | | | | | | | bit 0 |

bit 13-2 **Unimplemented:** Read as '1'

bit 1 **IESO:** Internal External Switch Over bit
1 = Internal External Switch Over mode enabled
0 = Internal External Switch Over mode disabled

bit 0 **FCMEN:** Fail Clock Monitor Enable bit
1 = Fail-Safe Clock Monitor enabled
0 = Fail-Safe Clock Monitor disabled

| 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 |

## 15.2 RESET

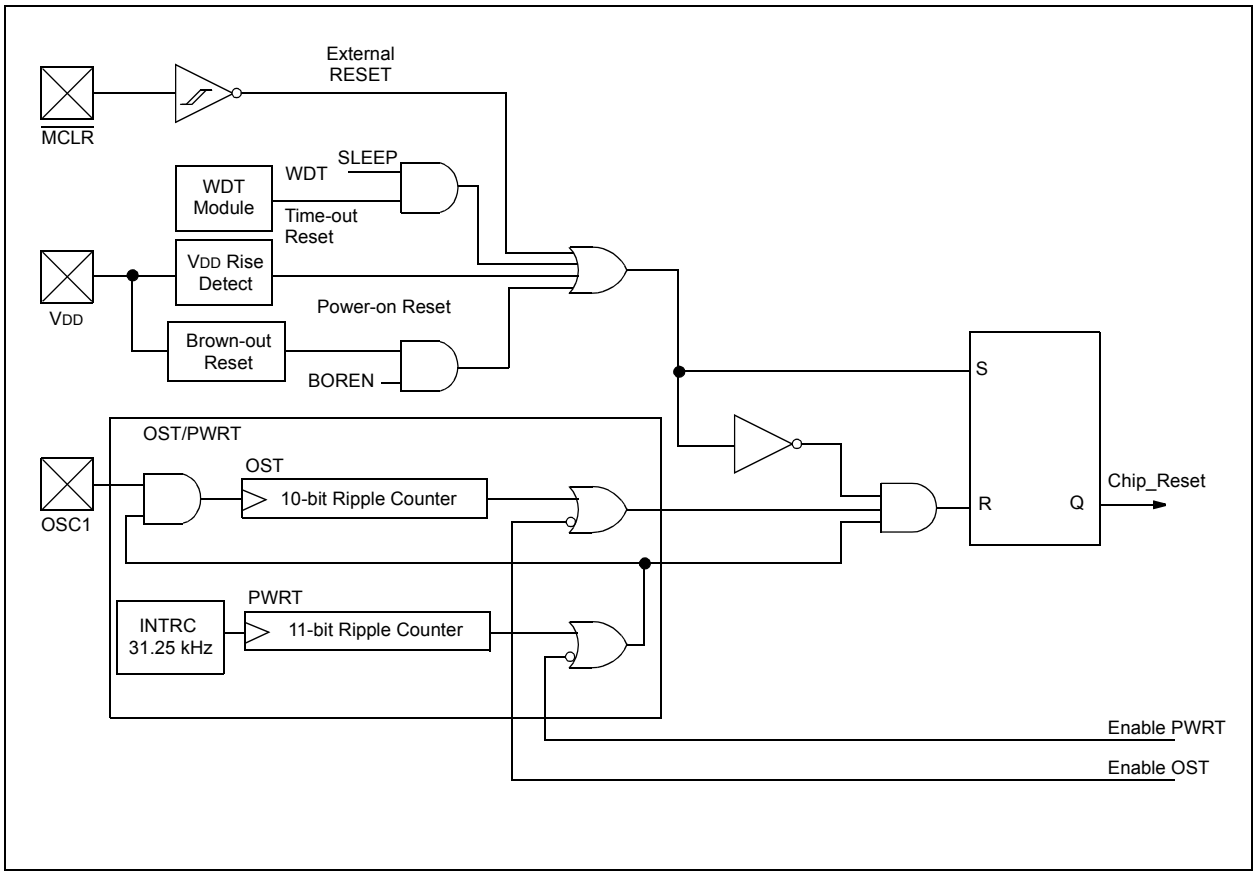
The PIC16F87/88 differentiates between various kinds of RESET:

- Power-on Reset (POR)
- $\overline{MCLR}$ Reset during normal operation
- $\overline{MCLR}$ Reset during SLEEP
- WDT Reset during normal operation
- WDT Wake-up during SLEEP
- Brown-out Reset (BOR)

Some registers are not affected in any RESET condition. Their status is unknown on POR and unchanged in any other RESET. Most other registers are reset to a "RESET state" on Power-on Reset (POR), on the $\overline{MCLR}$ and WDT Reset, on $\overline{MCLR}$ Reset during SLEEP, and Brown-out Reset (BOR). They are not affected by a WDT wake-up, which is viewed as the resumption of normal operation. The $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different RESET situations, as indicated in Table 15-3. These bits are used in software to determine the nature of the RESET. Upon a POR, BOR, or wake-up from SLEEP, the CPU requires approximately 5 - 10 μs to become ready for code execution. This delay runs in parallel with any other timers. See Table 15-4 for a full description of RESET states of all registers.

A simplified block diagram of the on-chip RESET circuit is shown in Figure 15-1.

**FIGURE 15-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



## 15.3 MCLR

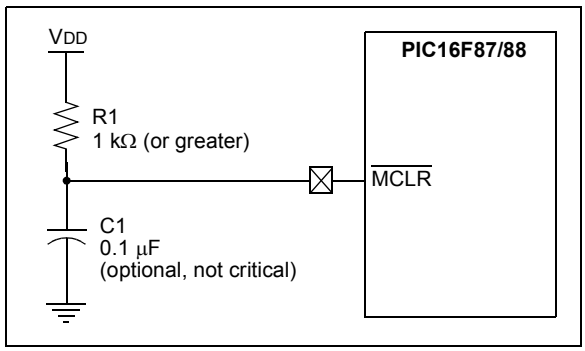
PIC16F87/88 devices have a noise filter in the MCLR Reset path. The filter will detect and ignore small pulses.

It should be noted that a WDT Reset does not drive MCLR pin low.

The behavior of the ESD protection on the MCLR pin has been altered from previous devices of this family. Voltages applied to the pin that exceed its specification, can result in both MCLR and excessive current beyond the device specification, during the ESD event. For this reason, Microchip recommends that the MCLR pin no longer be tied directly to VDD. The use of an RC network, as shown in Figure 15-2, is suggested.

The RA5/MCLR pin can be configured for MCLR (default), or as an I/O pin (RA5). This is configured through the MCLRE bit in Configuration Word 1.

**FIGURE 15-2: RECOMMENDED MCLR CIRCUIT**




## 15.4 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.2V - 1.7V). To take advantage of the POR, tie the MCLR pin to VDD, as described in Section 15.3. A maximum rise time for VDD is specified. See Section 18.0, "Electrical Characteristics" for details.

When the device starts normal operation (exits the RESET condition), device operating parameters (voltage, frequency, temperature,...) must be met to ensure operation. If these conditions are not met, the device must be held in RESET until the operating conditions are met. For more information, see Application Note, AN607 *"Power-up Trouble Shooting"* (DS00607).

## 15.5 Power-up Timer (PWRT)

The Power-up Timer (PWRT) of the PIC16F87/88 is a counter that uses the INTRC oscillator as the clock input. This yields a count of 72 ms. While the PWRT is counting, the device is held in RESET.

The power-up time delay depends on the INTRC, and will vary from chip-to-chip due to temperature and process variation. See DC parameter #33 for details.

The PWRT is enabled by clearing configuration bit PWRTEN.

## 15.6 Oscillator Start-up Timer (OST)

The Oscillator Start-up Timer (OST) provides 1024 oscillator cycles (from OSC1 input) delay after the PWRT delay is over (if enabled). This helps to ensure that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP and HS modes and only on Power-on Reset, or wake-up from SLEEP.

## 15.7 Brown-out Reset (BOR)

The configuration bit, BOREN, can enable or disable the Brown-out Reset circuit. If VDD falls below VBOR (parameter D005, about 4V) for longer than TBOR (parameter #35, about 100 μs), the brown-out situation will reset the device. If VDD falls below VBOR for less than TBOR, a RESET may not occur.

Once the brown-out occurs, the device will remain in Brown-out Reset until VDD rises above VBOR. The Power-up Timer (if enabled) will keep the device in RESET for TPWRT (parameter #33, about 72 ms). If VDD should fall below VBOR during TPWRT, the Brown-out Reset process will restart when VDD rises above VBOR, with the Power-up Timer Reset. Unlike previous PIC16 devices, the PWRT is no longer automatically enabled when the Brown-out Reset circuit is enabled. The PWRTEN and BOREN configuration bits are independent of each other.

## 15.8 Time-out Sequence

On power-up, the time-out sequence is as follows: the PWRT delay starts (if enabled) when a POR occurs. Then, OST starts counting 1024 oscillator cycles when PWRT ends (LP, XT, HS). When the OST ends, the device comes out of RESET.

If $\overline{MCLR}$ is kept low long enough, all delays will expire. Bringing $\overline{MCLR}$ high will begin execution immediately. This is useful for testing purposes, or to synchronize more than one PIC16F87/88 device operating in parallel.

Table 15-3 shows the RESET conditions for the STATUS, PCON and PC registers, while Table 15-4 shows the RESET conditions for all the registers.

## 15.9 Power Control/Status Register (PCON)

The Power Control/Status Register, PCON, has two bits to indicate the type of RESET that last occurred.

Bit0 is Brown-out Reset Status bit, $\overline{BOR}$. Bit $\overline{BOR}$ is unknown on a Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if bit $\overline{BOR}$ cleared, indicating a Brown-out Reset occurred. When the Brown-out Reset is disabled, the state of the $\overline{BOR}$ bit is unpredictable.

Bit1 is $\overline{POR}$ (Power-on Reset Status bit). It is cleared on a Power-on Reset and unaffected otherwise. The user must set this bit following a Power-on Reset.

**TABLE 15-1: TIME-OUT IN VARIOUS SITUATIONS**

| Oscillator Configuration | Power-up | | Brown-out Reset | | Wake-up from SLEEP |
|---|---|---|---|---|---|
| | $\overline{PWRTE}$ = 0 | $\overline{PWRTE}$ = 1 | $\overline{PWRTE}$ = 0 | $\overline{PWRTE}$ = 1 | |
| XT, HS, LP | TPWRT + 1024 • TOSC | 1024 • TOSC | TPWRT + 1024 • TOSC | 1024 • TOSC | 1024 • TOSC |
| EXTRC, INTRC | TPWRT | 5 - 10 μs(1) | TPWRT | 5 - 10 μs(1) | 5 - 10 μs(1) |
| T1OSC | — | — | — | — | 5 - 10 μs(1) |

**Note 1:** CPU start-up is always invoked on POR, BOR and wake-up from SLEEP. The 5 μs - 10 μs delay is based on a 1 MHz System Clock.

**TABLE 15-2: STATUS BITS AND THEIR SIGNIFICANCE**

| $\overline{POR}$ | $\overline{BOR}$ | $\overline{TO}$ | $\overline{PD}$ | |
|---|---|---|---|---|
| 0 | x | 1 | 1 | Power-on Reset |
| 0 | x | 0 | x | Illegal, $\overline{TO}$ is set on $\overline{POR}$ |
| 0 | x | x | 0 | Illegal, $\overline{PD}$ is set on $\overline{POR}$ |
| 1 | 0 | 1 | 1 | Brown-out Reset |
| 1 | 1 | 0 | 1 | WDT Reset |
| 1 | 1 | 0 | 0 | WDT Wake-up |
| 1 | 1 | u | u | $\overline{MCLR}$ Reset during normal operation |
| 1 | 1 | 1 | 0 | $\overline{MCLR}$ Reset during SLEEP or Interrupt Wake-up from SLEEP |

Legend: u = unchanged, x = unknown

**TABLE 15-3: RESET CONDITION FOR SPECIAL REGISTERS**

| Condition | Program Counter | STATUS Register | PCON Register |
|---|---|---|---|
| Power-on Reset | 000h | 0001 1xxx | ---- --0x |
| $\overline{MCLR}$ Reset during normal operation | 000h | 000u uuuu | ---- --uu |
| $\overline{MCLR}$ Reset during SLEEP | 000h | 0001 0uuu | ---- --uu |
| WDT Reset | 000h | 0000 1uuu | ---- --uu |
| WDT Wake-up | PC + 1 | uuu0 0uuu | ---- --uu |
| Brown-out Reset | 000h | 0001 1uuu | ---- --u0 |
| Interrupt Wake-up from SLEEP | PC + 1(1) | uuu1 0uuu | ---- --uu |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'

**Note 1:** When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

**TABLE 15-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

| Register | Power-on Reset, Brown-out Reset | $\overline{MCLR}$ Reset, WDT Reset | Wake-up via WDT or Interrupt |
|---|---|---|---|
| W | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF | N/A | N/A | N/A |
| TMR0 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCL | 0000h | 0000h | PC + 1(2) |
| STATUS | 0001 1xxx | 000q quuu(3) | uuuq quuu(3) |
| FSR | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTA | xxx0 0000 | uuu0 0000 | uuuu uuuu |
| PORTB | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCLATH | ---0 0000 | ---0 0000 | ---u uuuu |
| INTCON | 0000 000x | 0000 000u | uuuu uuuu(1) |
| PIR1 | -000 0000 | -000 0000 | -uuu uuuu(1) |
| PIR2 | 00-0 ---- | 00-0 ---- | uu-u ----(1) |
| TMR1L | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR1H | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T1CON | -000 0000 | -uuu uuuu | -uuu uuuu |
| TMR2 | 0000 0000 | 0000 0000 | uuuu uuuu |
| T2CON | -000 0000 | -000 0000 | -uuu uuuu |
| SSPBUF | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| SSPCON | 0000 0000 | 0000 0000 | uuuu uuuu |
| CCPR1L | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR1H | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP1CON | --00 0000 | --00 0000 | --uu uuuu |
| RCSTA | 0000 000x | 0000 000x | uuuu uuuu |
| TXREG | 0000 0000 | 0000 0000 | uuuu uuuu |
| RCREG | 0000 0000 | 0000 0000 | uuuu uuuu |
| ADRESH | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADCON0 | 0000 00-0 | 0000 00-0 | uuuu uu-u |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition, r = reserved, maintain clear

**Note 1:** One or more bits in INTCON, PIR1 and PR2 will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

**3:** See Table 15-3 for RESET value for specific condition.

**TABLE 15-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Reset, WDT Reset | Wake-up via WDT or Interrupt |
|---|---|---|---|
| OPTION | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISB | 1111 1111 | 1111 1111 | uuuu uuuu |
| PIE1 | -000 0000 | -000 0000 | -uuu uuuu |
| PIE2 | 00-0 ---- | 00-0 ---- | uu-u ---- |
| PCON | ---- --qq | ---- --uu | ---- --uu |
| OSCCON | -000 0000 | -000 0000 | -uuu uuuu |
| OSCTUNE | --00 0000 | --00 0000 | --uu uuuu |
| PR2 | 1111 1111 | 1111 1111 | 1111 1111 |
| SSPADD | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSPSTAT | 0000 0000 | 0000 0000 | uuuu uuuu |
| TXSTA | 0000 -010 | 0000 -010 | uuuu -u1u |
| SPBRG | 0000 0000 | 0000 0000 | uuuu uuuu |
| ANSEL | -111 1111 | -111 1111 | -111 1111 |
| CMCON | 0000 0000 | 0000 0000 | uuuu uuuu |
| CVRCON | 000- 0000 | 000- 0000 | uuu- uuuu |
| WDTCON | ---0 1000 | ---0 1000 | ---u 1uuu |
| ADRESL | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADCON1 | 0000 ---- | 0000 ---- | uuuu ---- |
| EEDATA | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| EEADR | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| EEDATH | --xx xxxx | --uu uuuu | --uu uuuu |
| EEADRH | ---- -xxx | ---- -uuu | ---- -uuu |
| EECON1 | x--x x000 | u--x u000 | u--u uuuu |
| EECON2 | ---- ---- | ---- ---- | ---- ---- |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition, r = reserved, maintain clear

**Note 1:** One or more bits in INTCON, PIR1 and PR2 will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

**3:** See Table 15-3 for RESET value for specific condition.

**FIGURE 15-3: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO $V_{DD}$ THROUGH PULL-UP RESISTOR)**

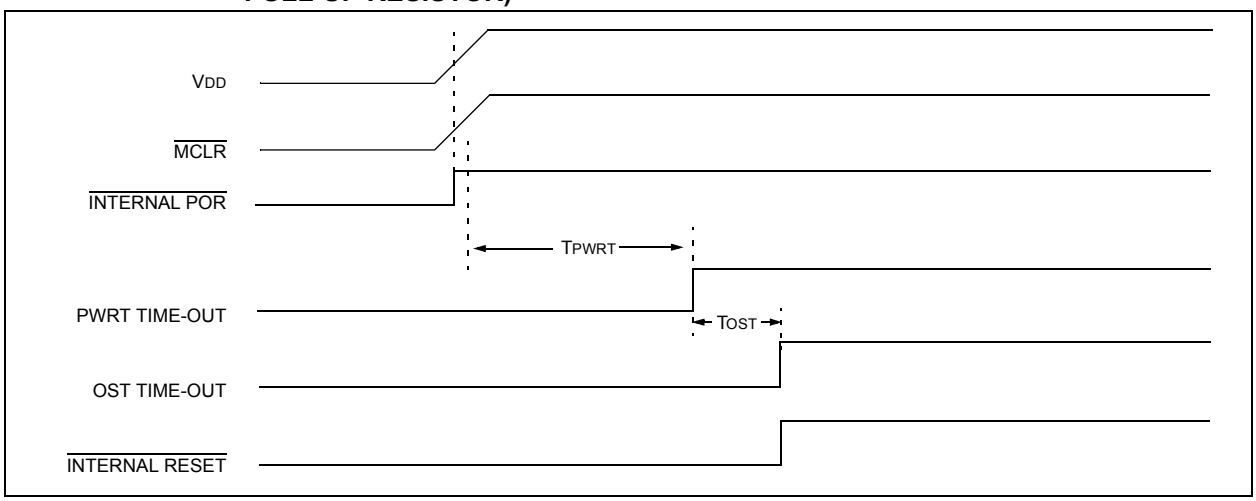


**FIGURE 15-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD THROUGH RC NETWORK): CASE 1**

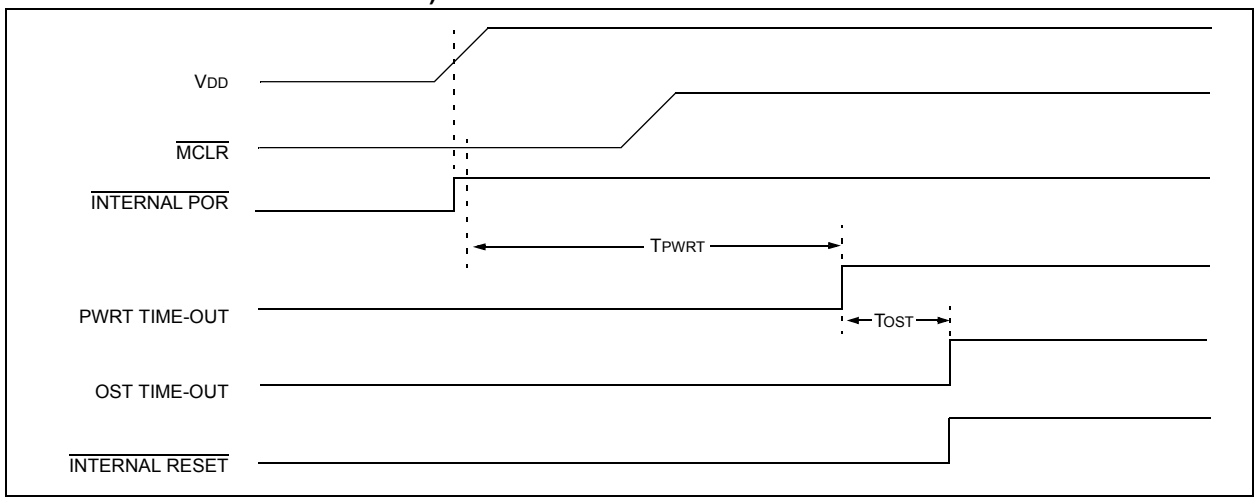



**FIGURE 15-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD THROUGH RC NETWORK): CASE 2**

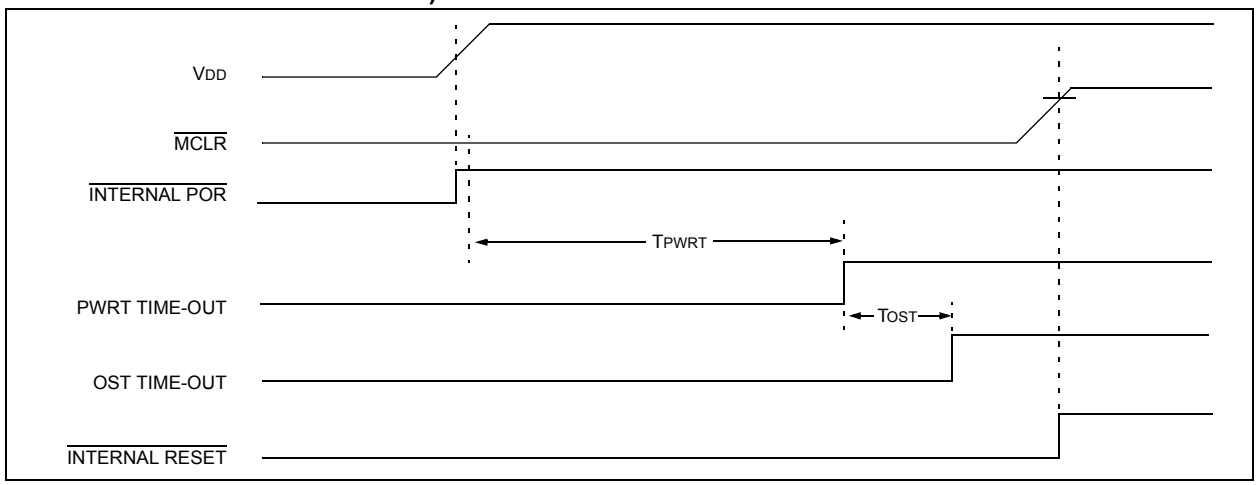
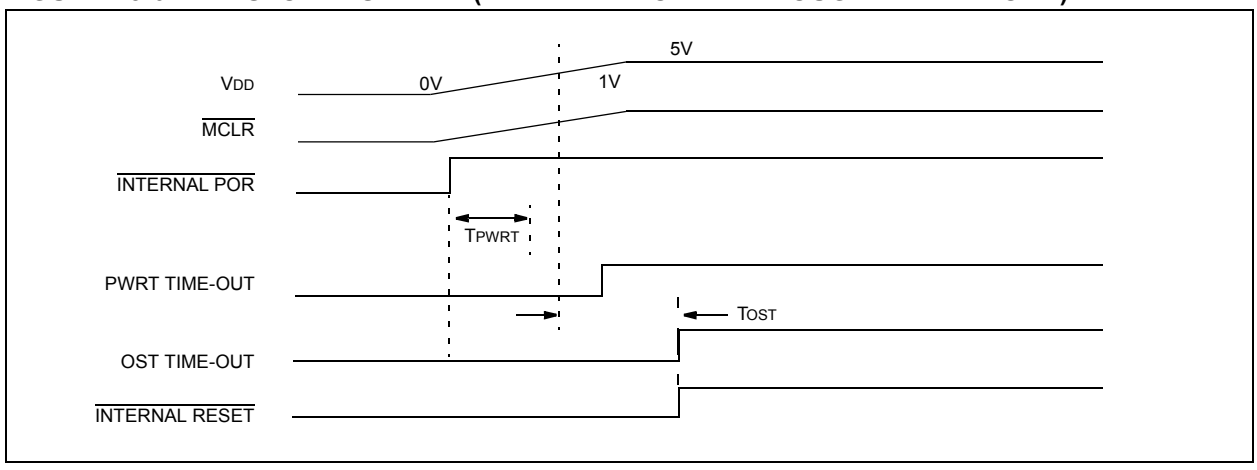



**FIGURE 15-6: SLOW RISE TIME (MCLR TIED TO VDD THROUGH RC NETWORK)**



## 15.10 Interrupts

The PIC16F87/88 has up to 12 sources of interrupt. The Interrupt Control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

| **Note:** | Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit or the GIE bit. |
|---|---|

A global interrupt enable bit, GIE (INTCON<7>), enables (if set) all unmasked interrupts, or disables (if cleared) all interrupts. When bit GIE is enabled, and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt bits are set regardless of the status of the GIE bit. The GIE bit is cleared on RESET.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine, as well as sets the GIE bit, which re-enables interrupts.

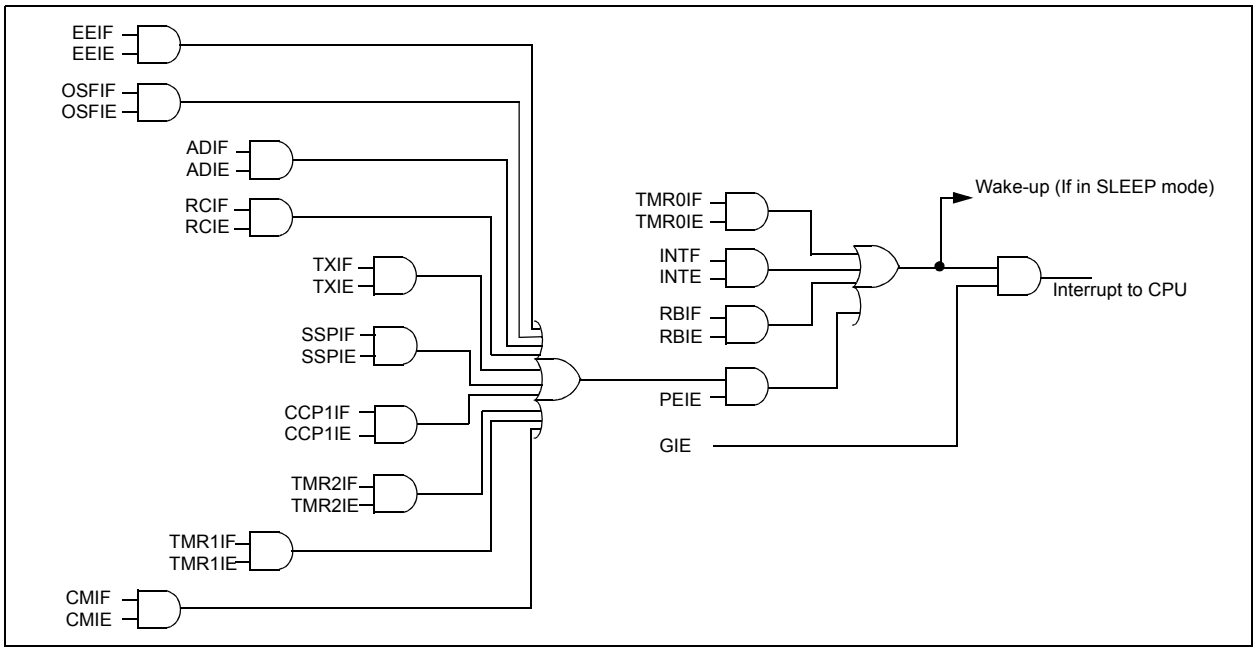
The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flags are contained in the Special Function Register, PIR1. The corresponding interrupt enable bits are contained in Special Function Register, PIE1, and the peripheral interrupt enable bit is contained in Special Function Register, INTCON.

When an interrupt is serviced, the GIE bit is cleared to disable any further interrupt, the return address is pushed onto the stack, and the PC is loaded with 0004h. Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends on when the interrupt event occurs, relative to the current Q cycle. The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit, PEIE bit, or the GIE bit.

**FIGURE 15-7: INTERRUPT LOGIC**



### 15.10.1 INT INTERRUPT

External interrupt on the RB0/INT pin is edge-triggered, either rising, if bit INTEDG (OPTION<6>) is set, or falling, if the INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, flag bit INTF (INTCON<1>) is set. This interrupt can be disabled by clearing enable bit INTE (INTCON<4>). Flag bit INTF must be cleared in software in the Interrupt Service Routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from SLEEP, if bit INTE was set prior to going into SLEEP. The status of global interrupt enable bit GIE decides whether or not the processor branches to the interrupt vector, following wake-up. See Section 15.13 for details on SLEEP mode.

### 15.10.2 TMR0 INTERRUPT

An overflow (FFh $\rightarrow$ 00h) in the TMR0 register will set flag bit TMR0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit TMR0IE (INTCON<5>), see Section 6.0.

### 15.10.3 PORTB INTCON CHANGE

An input change on PORTB<7:4> sets flag bit RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit RBIE (INTCON<4>), see Section 3.2.

## 15.11 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers during an interrupt (i.e., W, STATUS registers). This will have to be implemented in software, as shown in Example 15-1.

For the PIC16F87/88 devices, the register W_TEMP must be defined in both banks 0 and 1 and must be defined at the same offset from the bank base address (i.e., if W_TEMP is defined at 20h in bank 0, it must also be defined at A0h in bank 1). The register STATUS_TEMP is only defined in bank 0.

**EXAMPLE 15-1: SAVING STATUS AND W REGISTERS IN RAM**

```
    MOVWF   W_TEMP              ;Copy W to TEMP register
    SWAPF   STATUS,W            ;Swap status to be saved into W
    CLRF    STATUS              ;bank 0, regardless of current bank, Clears IRP,RP1,RP0
    MOVWF   STATUS_TEMP         ;Save status to bank zero STATUS_TEMP register
    :
    :(ISR)                      ;Insert user code here
    :
    SWAPF   STATUS_TEMP,W       ;Swap STATUS_TEMP register into W
                                ;(sets bank to original state)
    MOVWF   STATUS              ;Move W into STATUS register
    SWAPF   W_TEMP,F            ;Swap W_TEMP
    SWAPF   W_TEMP,W            ;Swap W_TEMP into W
```

## 15.12 Watchdog Timer (WDT)

For PIC16F87/88 devices, the WDT has been modified from previous PIC16 devices. The new WDT is code and functionally backward compatible with previous PIC16 WDT modules, and allows the user to have a scaler value for the WDT and TMR0 at the same time. In addition, the WDT time-out value can be extended to 268 seconds, using the prescaler with the postscaler when PSA is set to '1'.

### 15.12.1 WDT OSCILLATOR

The WDT derives its time-base from the 31.25 kHz INTRC; therefore, the accuracy of the 31.25 kHz will be the same accuracy for the WDT time-out period.

A new prescaler has been added to the path between the internal RC and the multiplexors used to select the path for the WDT. This prescaler is 16-bits and can be programmed to divide the internal RC by 128 to 65536, giving the time-base used for the WDT a nominal range of 1 ms to 2.097s.

### 15.12.2 WDT CONTROL

The WDTEN bit is located in Configuration Word 1 and when this bit is set, the WDT runs continuously.

The SWDTEN bit is in the WDTCON register. When the WDTEN bit in the Configuration Word 1 register is set, the SWDTEN bit has no effect. If WDTEN is clear, then the SWDTEN bit can be used to enable and disable the WDT. Setting the bit will enable it and clearing the bit will disable it.

The PSA and PS<2:0> bits (OPTION_REG) have the same function as in previous versions of the PIC16 family of microcontrollers.

### 15.12.3 RESET STATE

The value of WDTCON is '---0 1000' on all RESETS. This gives a nominal time-base of 16.38 ms, which is compatible with the time-base generated with previous PIC16 microcontroller versions.

| **Note:** | When the OST is invoked, the WDT is held in RESET, because the WDT Ripple Counter is used by the OST to perform the oscillator delay count. When the OST count has expired, the WDT will begin counting (if enabled). |
|---|---|

**FIGURE 15-8: WATCHDOG TIMER BLOCK DIAGRAM**

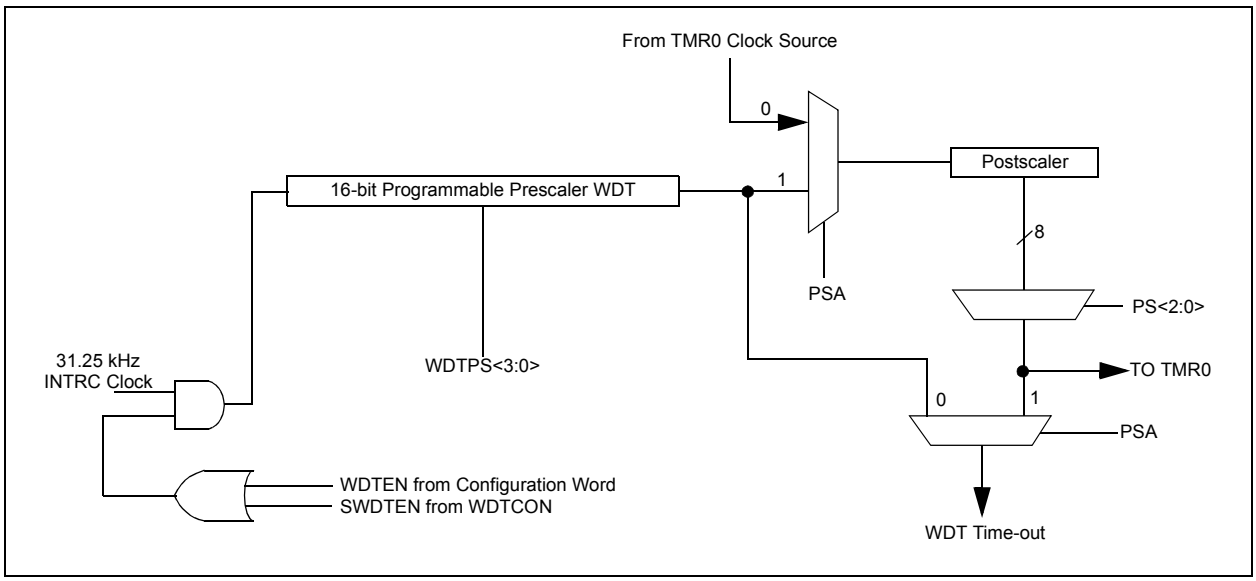



**TABLE 15-5: PRESCALER/POSTSCALER BIT STATUS**

| Conditions | Prescaler | Postscaler (PSA = 1) |
|---|---|---|
| WDTEN = 0 | Cleared | Cleared |
| CLRWDT command | | |
| OSC FAIL detected | | |
| Exit SLEEP + System Clock = T1OSC, EXTRC, INTRC, EXTCLK | | |
| Exit SLEEP + System Clock = XT, HS, LP | Cleared at end of OST | Cleared at end of OST |

**REGISTER 15-3: WDTCON REGISTER**

| U-0 | U-0 | U-0 | R/W-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | SWDTEN |
| bit 7 | | | | | | | bit 0 |

bit 7-5 **Unimplemented:** Read as '0'

bit 4-1 **WDTPS<3:0>:** Watchdog Timer Period Select bits

| Bit Value | | Prescale Rate |
|---|---|---|
| 0000 | = | 1:32 |
| 0001 | = | 1:64 |
| 0010 | = | 1:128 |
| 0011 | = | 1:256 |
| 0100 | = | 1:512 |
| 0101 | = | 1:1024 |
| 0110 | = | 1:2048 |
| 0111 | = | 1:4096 |
| 1000 | = | 1:8192 |
| 1001 | = | 1:16394 |
| 1010 | = | 1:32768 |
| 1011 | = | 1:65536 |

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit(1)

1 = WDT is turned on
0 = WDT is turned off

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

| 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 |

**TABLE 15-6: SUMMARY OF WATCHDOG TIMER REGISTERS**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 81h,181h | OPTION | $\overline{\text{RBPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 |
| 2007h | Configuration bits | LVP | BOREN | MVCLRE | FOSC2 | $\overline{\text{PWRTEN}}$ | WDTEN | FOSC1 | FOSC0 |
| 105h | WDTCON | — | — | — | WDTPS3 | $\overline{\text{WDTPS2}}$ | WSTPS1 | WDTPS0 | SWDTEN |

Legend: Shaded cells are not used by the Watchdog Timer.
**Note 1:** See Register 15-1 for operation of these bits.

### 15.12.4 TWO-SPEED CLOCK START-UP MODE

Two-Speed Start-up minimizes the latency between oscillator start-up and code execution that may be selected with the IESO (Internal/External Switch Over) bit in Configuration Word 2. This mode is achieved by initially using the INTRC for code execution until the primary oscillator is stable.

In this mode, upon

- POR and after the Power-up Timer has expired (if $\overline{\text{PWRTEN}}$ = 0),
- or following a wake-up from SLEEP,
- or a RESET when running from T1OSC or INTRC (after a RESET, SCS<1:0> are always set to '00').

the system will begin execution with the INTRC oscillator. This results in almost immediate code execution with a minimum of delay.

| **Note:** | Following any RESET, the IRCF bits are zeroed and the frequency selection is forced to 31.25 kHz. The user can modify the IRCF bits to select a higher internal oscillator frequency. |
|---|---|

If the primary oscillator is configured to be anything other than XT, LP, or HS, then Two-Speed Start-up is disabled, because the primary oscillator doesn't require any time to become stable after POR, or an exit from SLEEP.

If the IRCF bits of the OSCCON register are configured to a non-zero value prior to entering SLEEP mode, the secondary system clock frequency will come from the output of the INTOSC. The IOFS bit in the OSCCON register will be clear until the INTOSC is stable. This will allow the user to determine when the internal oscillator can be used for time critical applications.

Checking the state of the OSTS bit will confirm whether the primary clock configuration is engaged. If not, the OSTS bit will remain clear.

When the device is auto-configured in INTRC mode following a POR or wake-up from SLEEP, the rules for entering other Oscillator modes still apply, meaning the SCS<1:0> bits in OSCCON can be modified before the OST time-out has occurred. This would allow the application to wake-up from SLEEP, perform a few instructions using the INTRC as the clock source and go back to SLEEP without waiting for the primary oscillator to become stable.

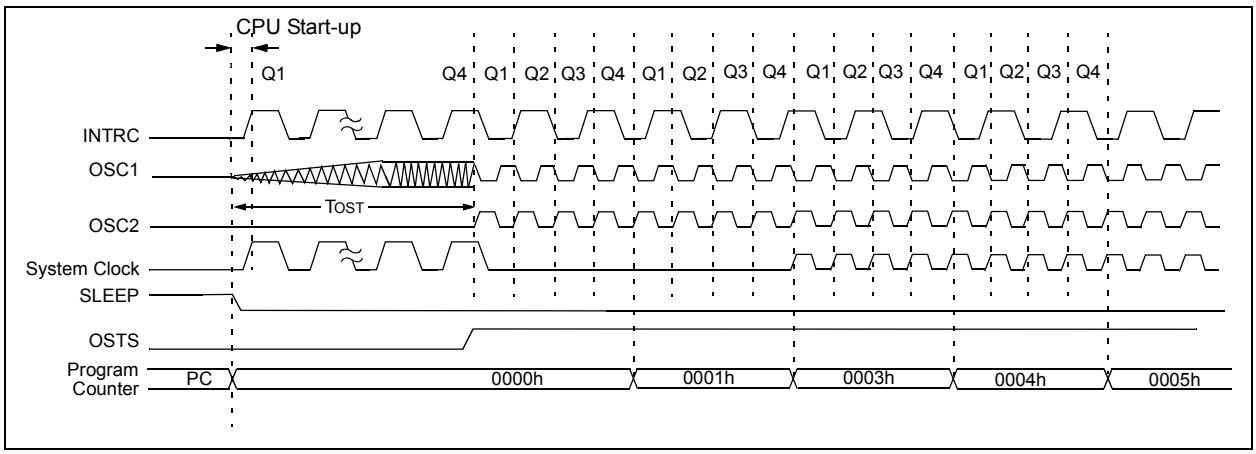
| **Note:** | Executing a SLEEP instruction will abort the Oscillator Start-up Time and will cause the OSTS bit to remain clear. |
|---|---|

#### 15.12.4.1 Two-Speed Start-up Sequence

1. Wake-up from SLEEP, RESET, or POR.
2. OSCON bits configured to run from INTRC (31.25 kHz).
3. Instructions begin execution by INTRC (31.25 kHz).
4. OST enabled to count 1024 clock cycles.
5. OST timed out, wait for falling edge of INTRC.
6. OSTS is set.
7. System clock held low for eight falling edges of new clock (LP, XT, or HS).
8. System clock is switched to primary source (LP, XT, or HS).

The software may read the OSTS bit to determine when the switch over takes place so that any software timing edges can be adjusted.

**FIGURE 15-9: TWO-SPEED START-UP**

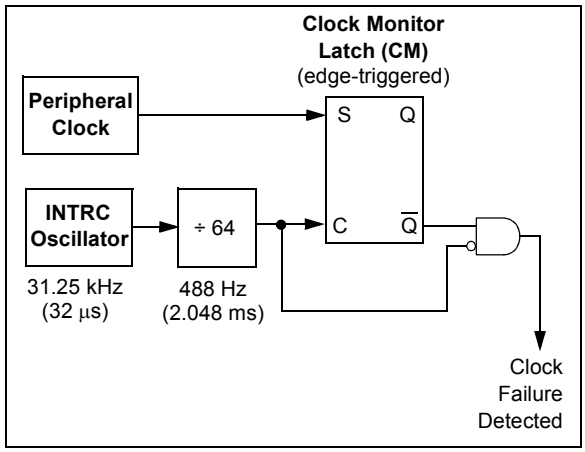


### 15.12.5 FAIL-SAFE OPTION

The Fail-Safe Clock Monitor (FSCM) is designed to allow the device to continue to operate even in the event of an oscillator failure.

**FIGURE 15-10: FSCM BLOCK DIAGRAM**




The FSCM function is enabled by setting the FCMEN bit in Configuration Word 2.

In the event of an oscillator failure, the FSCM will generate an Oscillator Fail interrupt and will switch the system clock over to the internal oscillator. The system will continue to come from the internal oscillator until the Fail-Safe condition is exited. The Fail-Safe condition is exited with either a RESET, the execution of a SLEEP instruction, or a write to the SCS bits.

The frequency of the internal oscillator will depend upon the value contained in the IRCF bits. Another clock source can be selected via the IRCF and the SCS bits of the OSCCON register.

The FSCM sample clock is generated by dividing the INTRC clock by 64. This will allow enough time between FSCM sample clocks for a system clock edge to occur.

On the rising edge of the postscaled clock, the monitoring latch (CM = 0) will be cleared. On a falling edge of the primary or secondary system clock, the monitoring latch will be set (CM = 1). In the event that a falling edge of the postscaled clock occurs, and the monitoring latch is not set, a clock failure has been detected.

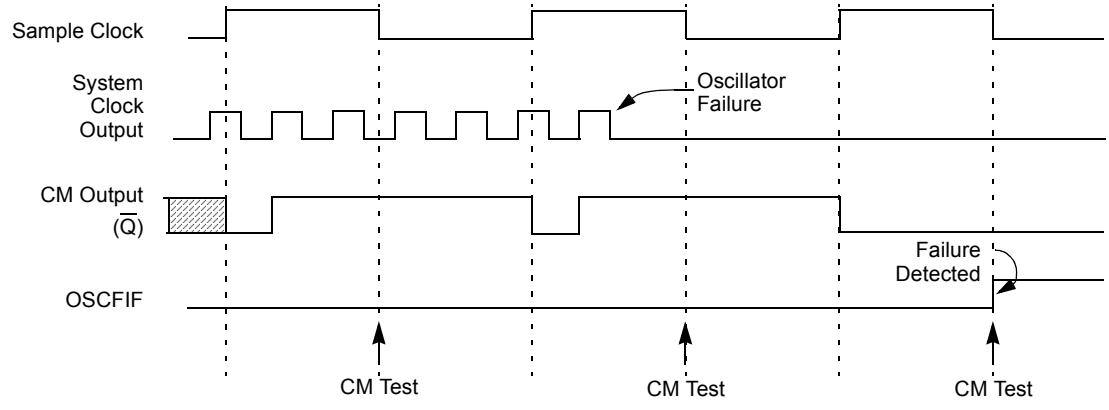
While in Fail-Safe mode, a RESET will exit the Fail-Safe condition. If the primary clock source is configured for a crystal, the OST timer will wait for the 1024 clock cycles for the OST time-out, and the device will continue running from the internal oscillator until the OST is complete. A SLEEP instruction, or a write to the SCS bits (where SCS bits do not = 00), can be performed to put the device into a Low Power mode.

If RESET occurs while in Fail-Safe mode and the primary clock source is EC, or RC, then the device will immediately switch back to EC or RC mode.

#### 15.12.5.1 Fail-Safe in Low Power Mode

A change of SCS<1:0>, or SLEEP instruction will end the Fail-Safe condition. The system clock will default to the source selected by the SCS bits, which is either T1OSC, INTRC, or none (SLEEP mode). However, the FSCM will continue to monitor the system clock. If the secondary clock fails, the device will immediately switch to the internal oscillator clock. If OSFIE is set, an interrupt will be generated.

**FIGURE 15-11: FSCM TIMING DIAGRAM**




**Note:** The system clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

#### 15.12.5.2 FSCM and the Watchdog Timer

When a clock failure is detected, SCS<1:0> will be forced to '10', which will reset the WDT (if enabled).

#### 15.12.5.3 FSCM Following POR or SLEEP

The FSCM is intended to detect oscillator failure at any point after the device has exited POR or SLEEP. However, following a POR or a wake-up from SLEEP, the primary clock will require a start-up time if the primary clock is configured as an oscillator (HS, XT, LP). The amount of time required to ensure a stable oscillator is undetermined and could be considerably longer than the FSCM sample clock time. Therefore, following a Power-on Reset, or following a wake-up from SLEEP, if the primary clock is configured as a crystal input, the INTRC clock is configured as the system clock until the primary clock, determined by fuse bits FOSC<2:0>, becomes stable. That is, if the intended clock is not valid after POR or wake-up is exited, the device will fetch the RESET vector or next instruction, using the INTRC clock until the primary clock becomes stable. This is the same as Two-Speed Start-up mode. If the primary clock is configured as anything else (RC, INTRC, or EC), the FSCM will monitor the system clock immediately following POR or wake-up from SLEEP.

| | |
|---|---|
| **Note:** | If the primary clock is configured as a crystal (HS, XT, LP) and the oscillator fails to operate following an exit from SLEEP or a POR, there is no way for the user to determine that the oscillator has failed. The user can monitor the OSTS bit in the OSCCON register and use a timing routine to determine if the oscillator time-out is taking too long, but no oscillator fail interrupt will take place. |

## 15.13 Power-down Mode (SLEEP)

Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{PD}$ bit (STATUS<3>) is cleared, the $\overline{TO}$ (STATUS<4>) bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, place all I/O pins at either $V_{DD}$ or $V_{SS}$, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D and disable external clocks. Pull all I/O pins that are hi-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input should also be at $V_{DD}$ or $V_{SS}$ for lowest current consumption. The contribution from on-chip pull-ups on PORTB should also be considered.

The $\overline{MCLR}$ pin must be at a logic high level ($V_{IHMC}$).

### 15.13.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

1. External RESET input on $\overline{MCLR}$ pin.
2. Watchdog Timer wake-up (if WDT was enabled).
3. Interrupt from INT pin, RB port change or a peripheral interrupt.

External $\overline{MCLR}$ Reset will cause a device RESET. All other events are considered a continuation of program execution and cause a "wake-up". The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of the device RESET. The $\overline{PD}$ bit, which is set on power-up, is cleared when SLEEP is invoked. The $\overline{TO}$ bit is cleared if a WDT time-out occurred and caused wake-up.

The following peripheral interrupts can wake the device from SLEEP:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. CCP Capture mode interrupt.
3. Special event trigger (Timer1 in Asynchronous mode using an external clock).
4. SSP (START/STOP) bit detect interrupt.
5. SSP transmit or receive in Slave mode (SPI/$I^2C$).
6. A/D conversion (when A/D clock source is RC).
7. EEPROM write operation completion.
8. Comparator output changes state.
9. USART RX or TX (Synchronous Slave mode).

Other peripherals cannot generate interrupts, since during SLEEP, no on-chip clocks are present.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up occurs regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

### 15.13.2 WAKE-UP USING INTERRUPTS

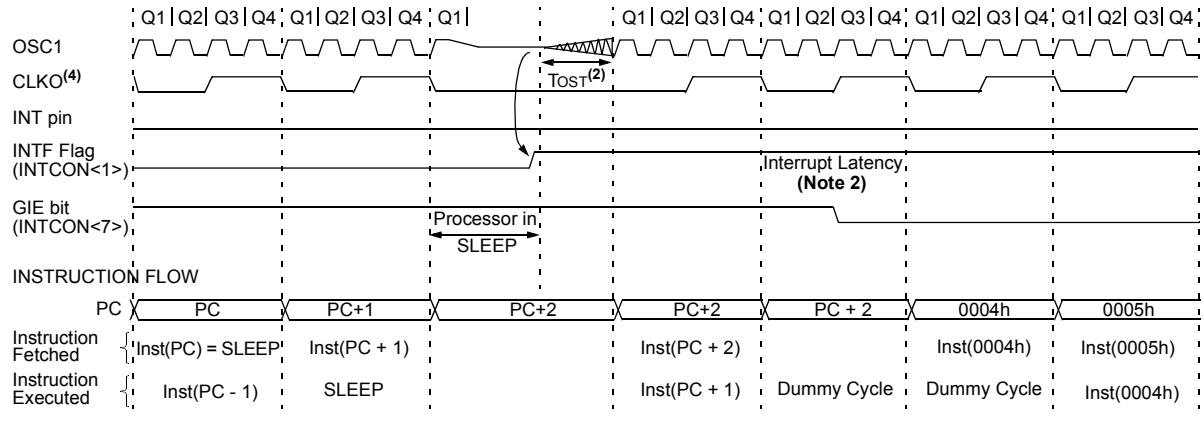
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will not be cleared, the $\overline{TO}$ bit will not be set and the $\overline{PD}$ bit will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake-up from SLEEP. The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

**FIGURE 15-12: WAKE-UP FROM SLEEP THROUGH INTERRUPT**




**Note 1:** XT, HS or LP Oscillator mode assumed.
**2:** TOST = 1024 TOSC (drawing not to scale). This delay will not be there for RC Osc mode.
**3:** GIE = '1' assumed. In this case, after wake-up, the processor jumps to the interrupt routine. If GIE = '0', execution will continue in-line.
**4:** CLKO is not available in these Osc modes, but shown here for timing reference.

## 15.14 In-Circuit Debugger

When the DEBUG bit in the configuration word is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® ICD. When the microcontroller has this feature enabled, some of the resources are not available for general use. Table 15-7 shows which features are consumed by the background debugger.

**TABLE 15-7: DEBUGGER RESOURCES**

| | |
|---|---|
| I/O pins | RB6, RB7 |
| Stack | 1 level |
| Program Memory | Address 0000h must be NOP |
| | Last 100h words |
| Data Memory | 0x070 (0x0F0, 0x170, 0x1F0)<br>0x1EB - 0x1EF |

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{MCLR}$/VPP, VDD, GND, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip, or one of the third party development tool companies.

## 15.15 Program Verification/Code Protection

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

## 15.16 ID Locations

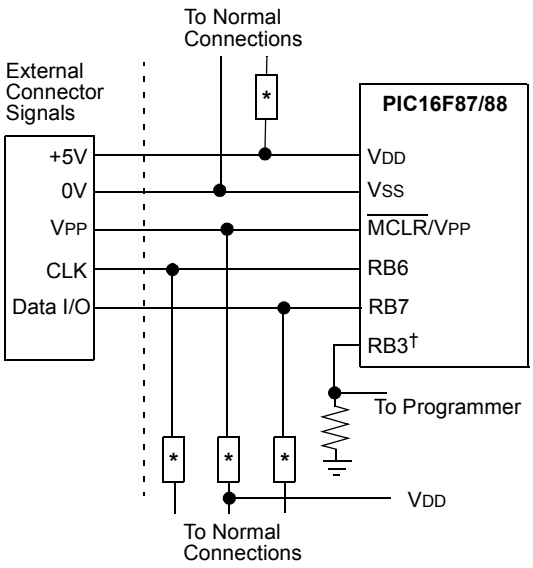
Four memory locations (2000h - 2003h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are not accessible during normal execution, but are readable and writable during program/verify. It is recommended that only the four Least Significant bits of the ID location are used.

## 15.17 In-Circuit Serial Programming

PIC16F87/88 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground, and the programming voltage (see Figure 15-13 for an example). This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

For general information of serial programming, please refer to the In-Circuit Serial Programming™ (ICSP™) Guide (DS30277).

**FIGURE 15-13: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION**



## 15.18 Low Voltage ICSP Programming

The LVP bit of the configuration word enables low voltage ICSP programming. This mode allows the microcontroller to be programmed via ICSP, using a VDD source in the operating voltage range. This only means that VPP does not have to be brought to VIHH, but can instead be left at the normal operating voltage. In this mode, the RB3/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. During programming, VDD is applied to the MCLR pin. To enter Programming mode, VDD must be applied to the RB3/PGM pin, provided the LVP bit is set. The LVP bit defaults to on ('1') from the factory.

**Note 1:** The High Voltage Programming mode is always available, regardless of the state of the LVP bit, by applying VIHH to the MCLR pin.

**2:** While in Low Voltage ICSP mode, the RB3 pin can no longer be used as a general purpose I/O pin.

**3:** When using Low Voltage ICSP Programming (LVP) and the pull-ups on PORTB are enabled, bit 3 in the TRISB register must be cleared to disable the pull-up on RB3 and ensure the proper operation of the device.

**4:** RB3 should not be allowed to float if LVP is enabled. An external pull-down device should be used to default the device to normal Operating mode. If RB3 floats high, the PIC16F87/88 device will enter Programming mode.

**5:** LVP mode is enabled by default on all devices shipped from Microchip. It can be disabled by clearing the LVP bit in the CONFIG register.

**6:** Disabling LVP will provide maximum compatibility to other PIC16CXXX devices.

If Low Voltage Programming mode is not used, the LVP bit can be programmed to a '0' and RB3/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed when programming is entered with VIHH on MCLR. The LVP bit can only be charged when using high voltage on MCLR.

It should be noted, that once the LVP bit is programmed to '0', only the High Voltage Programming mode is available and only High Voltage Programming mode can be used to program the device.

When using low voltage ICSP, the part must be supplied at 4.5V to 5.5V, if a bulk erase will be executed. This includes reprogramming of the code protect bits from an on-state to an off-state. For all other cases of low voltage ICSP, the part may be programmed at the normal operating voltage. This means calibration values, unique user IDs, or user code can be reprogrammed or added.

**NOTES:**

## 16.0 INSTRUCTION SET SUMMARY

The PIC16 instruction set is highly orthogonal and is comprised of three basic categories:

• **Byte-oriented** operations

• **Bit-oriented** operations

• **Literal and control** operations

Each PIC16 instruction is a 14-bit word divided into an **opcode,** which specifies the instruction type, and one or more **operands,** which further specify the operation of the instruction. The formats for each of the categories are presented in Figure 16-1, while the various opcode fields are summarized in Table 16-1.

Table 16-2 lists the instructions recognized by the MPASM™ assembler. A complete description of each instruction is also available in the PICmicro™ Mid-Range Reference Manual (DS33023).

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator, which selects the bit affected by the operation, while 'f' represents the address of the file in which the bit is located.

For **literal and control** operations, 'k' represents an eight- or eleven-bit constant or literal value

One instruction cycle consists of four oscillator periods; for an oscillator frequency of 4 MHz, this gives a normal instruction execution time of 1 μs. All instructions are executed within a single instruction cycle, unless a conditional test is true, or the program counter is changed as a result of an instruction. When this occurs, the execution takes two instruction cycles, with the second cycle executed as a NOP.

| **Note:** | To maintain upward compatibility with future PIC16F87/88 products, do not use the OPTION and TRIS instructions. |
|---|---|

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

## 16.1 READ-MODIFY-WRITE OPERATIONS

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified, and the result is stored according to either the instruction, or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a "clrf PORTB" instruction will read PORTB, clear all the data bits, then write the result back to PORTB. This example would have the unintended result that the condition that sets the RBIF flag would be cleared.

**TABLE 16-1: OPCODE FIELD DESCRIPTIONS**

| Field | Description |
|---|---|
| f | Register file address (0x00 to 0x7F) |
| W | Working register (accumulator) |
| b | Bit address within an 8-bit file register |
| k | Literal field, constant data or label |
| x | Don't care location (= 0 or 1).<br>The assembler will generate code with x = 0.<br>It is the recommended form of use for compatibility with all Microchip software tools. |
| d | Destination select; d = 0: store result in W,<br>d = 1: store result in file register f.<br>Default is d = 1. |
| PC | Program Counter |
| TO | Time-out bit |
| PD | Power-down bit |

**FIGURE 16-1: GENERAL FORMAT FOR INSTRUCTIONS**

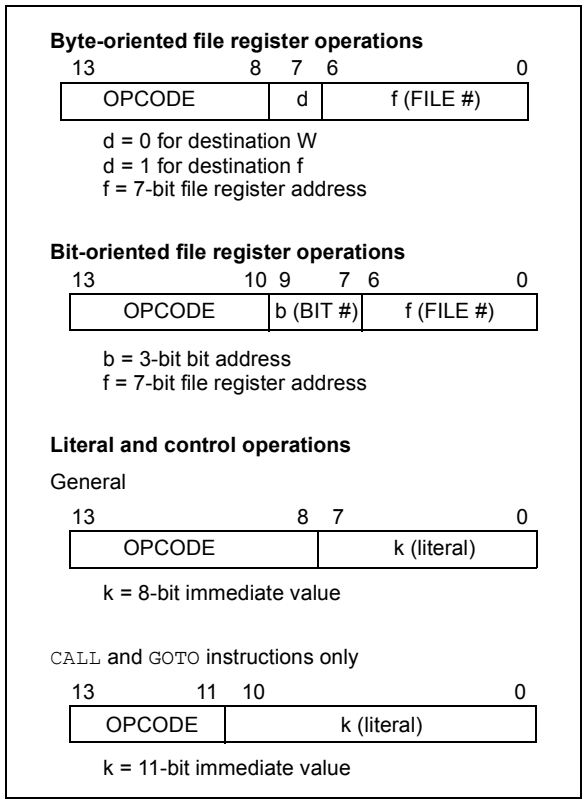


**TABLE 16-2: PIC16F87/88 INSTRUCTION SET**

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode MSb | | | LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| **BYTE-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| **ADDWF** | **f, d** | Add W and f | 1 | 00 | 0111 | dfff | ffff | C,DC,Z | 1,2 |
| **ANDWF** | **f, d** | AND W with f | 1 | 00 | 0101 | dfff | ffff | Z | 1,2 |
| **CLRF** | **f** | Clear f | 1 | 00 | 0001 | lfff | ffff | Z | 2 |
| **CLRW** | **-** | Clear W | 1 | 00 | 0001 | 0xxx | xxxx | Z | |
| **COMF** | **f, d** | Complement f | 1 | 00 | 1001 | dfff | ffff | Z | 1,2 |
| **DECF** | **f, d** | Decrement f | 1 | 00 | 0011 | dfff | ffff | Z | 1,2 |
| **DECFSZ** | **f, d** | Decrement f, Skip if 0 | 1(2) | 00 | 1011 | dfff | ffff | | 1,2,3 |
| **INCF** | **f, d** | Increment f | 1 | 00 | 1010 | dfff | ffff | Z | 1,2 |
| **INCFSZ** | **f, d** | Increment f, Skip if 0 | 1(2) | 00 | 1111 | dfff | ffff | | 1,2,3 |
| **IORWF** | **f, d** | Inclusive OR W with f | 1 | 00 | 0100 | dfff | ffff | Z | 1,2 |
| **MOVF** | **f, d** | Move f | 1 | 00 | 1000 | dfff | ffff | Z | 1,2 |
| **MOVWF** | **f** | Move W to f | 1 | 00 | 0000 | lfff | ffff | | |
| **NOP** | **-** | No Operation | 1 | 00 | 0000 | 0xx0 | 0000 | | |
| **RLF** | **f, d** | Rotate Left f through Carry | 1 | 00 | 1101 | dfff | ffff | C | 1,2 |
| **RRF** | **f, d** | Rotate Right f through Carry | 1 | 00 | 1100 | dfff | ffff | C | 1,2 |
| **SUBWF** | **f, d** | Subtract W from f | 1 | 00 | 0010 | dfff | ffff | C,DC,Z | 1,2 |
| **SWAPF** | **f, d** | Swap nibbles in f | 1 | 00 | 1110 | dfff | ffff | | 1,2 |
| **XORWF** | **f, d** | Exclusive OR W with f | 1 | 00 | 0110 | dfff | ffff | Z | 1,2 |
| **BIT-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| **BCF** | **f, b** | Bit Clear f | 1 | 01 | 00bb | bfff | ffff | | 1,2 |
| **BSF** | **f, b** | Bit Set f | 1 | 01 | 01bb | bfff | ffff | | 1,2 |
| **BTFSC** | **f, b** | Bit Test f, Skip if Clear | 1 (2) | 01 | 10bb | bfff | ffff | | 3 |
| **BTFSS** | **f, b** | Bit Test f, Skip if Set | 1 (2) | 01 | 11bb | bfff | ffff | | 3 |
| **LITERAL AND CONTROL OPERATIONS** | | | | | | | | | |
| **ADDLW** | **k** | Add literal and W | 1 | 11 | 111x | kkkk | kkkk | C,DC,Z | |
| **ANDLW** | **k** | AND literal with W | 1 | 11 | 1001 | kkkk | kkkk | Z | |
| **CALL** | **k** | Call subroutine | 2 | 10 | 0kkk | kkkk | kkkk | | |
| **CLRWDT** | **-** | Clear Watchdog Timer | 1 | 00 | 0000 | 0110 | 0100 | $\overline{TO}$,$\overline{PD}$ | |
| **GOTO** | **k** | Go to address | 2 | 10 | 1kkk | kkkk | kkkk | | |
| **IORLW** | **k** | Inclusive OR literal with W | 1 | 11 | 1000 | kkkk | kkkk | Z | |
| **MOVLW** | **k** | Move literal to W | 1 | 11 | 00xx | kkkk | kkkk | | |
| **RETFIE** | **-** | Return from interrupt | 2 | 00 | 0000 | 0000 | 1001 | | |
| **RETLW** | **k** | Return with literal in W | 2 | 11 | 01xx | kkkk | kkkk | | |
| **RETURN** | **-** | Return from Subroutine | 2 | 00 | 0000 | 0000 | 1000 | | |
| **SLEEP** | **-** | Go into Standby mode | 1 | 00 | 0000 | 0110 | 0011 | $\overline{TO}$,$\overline{PD}$ | |
| **SUBLW** | **k** | Subtract W from literal | 1 | 11 | 110x | kkkk | kkkk | C,DC,Z | |
| **XORLW** | **k** | Exclusive OR literal with W | 1 | 11 | 1010 | kkkk | kkkk | Z | |

**Note 1:** When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

**Note:** Additional information on the mid-range instruction set is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

## 16.2 Instruction Descriptions

| ADDLW | Add Literal and W |
|---|---|
| Syntax: | [ *label* ] ADDLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | (W) + k → (W) |
| Status Affected: | C, DC, Z |
| Description: | The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register. |

| ADDWF | Add W and f |
|---|---|
| Syntax: | [ *label* ] ADDWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) + (f) → (destination) |
| Status Affected: | C, DC, Z |
| Description: | Add the contents of the W register with register 'f'. If 'd' = '0', the result is stored in the W register. If 'd' = '1', the result is stored back in register 'f'. |

| ANDLW | AND Literal with W |
|---|---|
| Syntax: | [ *label* ] ANDLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | (W) .AND. (k) → (W) |
| Status Affected: | Z |
| Description: | The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register. |

| ANDWF | AND W with f |
|---|---|
| Syntax: | [ *label* ] ANDWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) .AND. (f) → (destination) |
| Status Affected: | Z |
| Description: | AND the W register with register 'f'. If 'd' = '0', the result is stored in the W register. If 'd' = '1', the result is stored back in register 'f'. |

| BCF | Bit Clear f |
|---|---|
| Syntax: | [ *label* ] BCF f,b |
| Operands: | $0 \le f \le 127$<br>$0 \le b \le 7$ |
| Operation: | 0 → (f<b>) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is cleared. |

| BSF | Bit Set f |
|---|---|
| Syntax: | [ *label* ] BSF f,b |
| Operands: | $0 \le f \le 127$<br>$0 \le b \le 7$ |
| Operation: | 1 → (f<b>) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is set. |

| BTFSS | Bit Test f, Skip if Set |
|---|---|
| Syntax: | [ *label* ] BTFSS f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b < 7 |
| Operation: | skip if (f<b>) = 1 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' = '0', the next instruction is executed.<br>If bit 'b' = '1', then the next instruction is discarded and a NOP is executed instead, making this a 2 TCY instruction. |

| BTFSC | Bit Test, Skip if Clear |
|---|---|
| Syntax: | [ *label* ] BTFSC f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | skip if (f<b>) = 0 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' = '1', the next instruction is executed.<br>If bit 'b', in register 'f', = '0', the next instruction is discarded, and a NOP is executed instead, making this a 2 TCY instruction. |

| CALL | Call Subroutine |
|---|---|
| Syntax: | [ *label* ] CALL k |
| Operands: | 0 ≤ k ≤ 2047 |
| Operation: | (PC) + 1 → TOS,<br>k → PC<10:0>,<br>(PCLATH<4:3>) → PC<12:11> |
| Status Affected: | None |
| Description: | Call subroutine. First, return address (PC+1) is pushed onto the stack. The eleven-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle instruction. |

| CLRF | Clear f |
|---|---|
| Syntax: | [ *label* ] CLRF f |
| Operands: | 0 ≤ f ≤ 127 |
| Operation: | 00h → (f)<br>1 → Z |
| Status Affected: | Z |
| Description: | The contents of register 'f' are cleared and the Z bit is set. |

| CLRW | Clear W |
|---|---|
| Syntax: | [ *label* ] CLRW |
| Operands: | None |
| Operation: | 00h → (W)<br>1 → Z |
| Status Affected: | Z |
| Description: | W register is cleared. Zero bit (Z) is set. |

| CLRWDT | Clear Watchdog Timer |
|---|---|
| Syntax: | [ *label* ] CLRWDT |
| Operands: | None |
| Operation: | 00h → WDT<br>0 → WDT prescaler,<br>1 → $\overline{TO}$<br>1 → $\overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Description: | CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set. |

| COMF | Complement f |
|---|---|
| Syntax: | [ *label* ] COMF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) → (destination) |
| Status Affected: | Z |
| Description: | The contents of register 'f' are complemented. If 'd' = '0', the result is stored in W. If 'd' = '1', the result is stored back in register 'f'. |

| DECF | Decrement f |
|---|---|
| Syntax: | [ *label* ] DECF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - 1 → (destination) |
| Status Affected: | Z |
| Description: | Decrement register 'f'. If 'd' = '0', the result is stored in the W register. If 'd' = '1', the result is stored back in register 'f'. |

| DECFSZ | Decrement f, Skip if 0 |
|---|---|
| Syntax: | [ *label* ] DECFSZ f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - 1 → (destination);<br>skip if result = 0 |
| Status Affected: | None |
| Description: | The contents of register 'f' are decremented. If 'd' = '0', the result is placed in the W register. If 'd' = '1', the result is placed back in register 'f'.<br>If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2 TCY instruction. |

| GOTO | Unconditional Branch |
|---|---|
| Syntax: | [ *label* ] GOTO k |
| Operands: | $0 \le k \le 2047$ |
| Operation: | k → PC<10:0><br>PCLATH<4:3> → PC<12:11> |
| Status Affected: | None |
| Description: | GOTO is an unconditional branch. The eleven bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction. |

| INCF | Increment f |
|---|---|
| Syntax: | [ *label* ] INCF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) + 1 → (destination) |
| Status Affected: | Z |
| Description: | The contents of register 'f' are incremented. If 'd' = '0', the result is placed in the W register. If 'd' = '1', the result is placed back in register 'f'. |

| INCFSZ | Increment f, Skip if 0 |
|---|---|
| Syntax: | [ *label* ] INCFSZ f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) + 1 → (destination),<br>skip if result = 0 |
| Status Affected: | None |
| Description: | The contents of register 'f' are incremented. If 'd' = '0', the result is placed in the W register. If 'd' = '1', the result is placed back in register 'f'.<br>If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2 TCY instruction. |

| IORLW | Inclusive OR Literal with W |
|---|---|
| Syntax: | [ *label* ] IORLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | (W) .OR. k $\rightarrow$ (W) |
| Status Affected: | Z |
| Description: | The contents of the W register are OR'd with the eight-bit literal 'k'. The result is placed in the W register. |

| IORWF | Inclusive OR W with f |
|---|---|
| Syntax: | [ *label* ] IORWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) .OR. (f) $\rightarrow$ (destination) |
| Status Affected: | Z |
| Description: | Inclusive OR the W register with register 'f'. If 'd' = '0', the result is placed in the W register. If 'd' = '1', the result is placed back in register 'f'. |

| MOVF | Move f |
|---|---|
| Syntax: | [ *label* ] MOVF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) $\rightarrow$ (destination) |
| Status Affected: | Z |
| Description: | The contents of register 'f' are moved to a destination dependant upon the status of 'd'. If 'd' = '0', the destination is W register. If 'd' = '1', the destination is file register 'f' itself. 'd' = '1' is useful to test a file register, since status flag Z is affected. |

| MOVLW | Move Literal to W |
|---|---|
| Syntax: | [ *label* ] MOVLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | k $\rightarrow$ (W) |
| Status Affected: | None |
| Description: | The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as '0's. |

| MOVWF | Move W to f |
|---|---|
| Syntax: | [ *label* ] MOVWF f |
| Operands: | $0 \le f \le 127$ |
| Operation: | (W) $\rightarrow$ (f) |
| Status Affected: | None |
| Description: | Move data from W register to register 'f'. |

| NOP | No Operation |
|---|---|
| Syntax: | [ *label* ] NOP |
| Operands: | None |
| Operation: | No operation |
| Status Affected: | None |
| Description: | No operation. |

**RETFIE** **Return from Interrupt**

Syntax: [ *label* ] RETFIE

Operands: None

Operation: TOS → PC,
1 → GIE

Status Affected: None

**RETLW** **Return with Literal in W**

Syntax: [ *label* ] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W);
TOS → PC

Status Affected: None

Description: The W register is loaded with the eight-bit literal ‘k’. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.

**RETURN** **Return from Subroutine**

Syntax: [ *label* ] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

**RLF** **Rotate Left f through Carry**

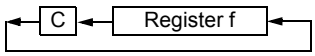
Syntax: [ *label* ] RLF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: See description below

Status Affected: C

Description: The contents of register ‘f’ are rotated one bit to the left through the Carry Flag. If ‘d’ = ‘0’, the result is placed in the W register. If ‘d’ = ‘1’, the result is stored back in register ‘f’.




**RRF** **Rotate Right f through Carry**

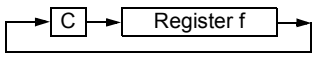
Syntax: [ *label* ] RRF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: See description below

Status Affected: C

Description: The contents of register ‘f’ are rotated one bit to the right through the Carry Flag. If ‘d’ = ‘0’, the result is placed in the W register. If ‘d’ = ‘1’, the result is placed back in register ‘f’.




**SLEEP**

Syntax: [ *label* ] SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT prescaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

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

Description: The power-down status bit, $\overline{PD}$ is cleared. Time-out status bit, $\overline{TO}$ is set. Watchdog Timer and its prescaler are cleared.
The processor is put into SLEEP mode with the oscillator stopped.

| SUBLW | Subtract W from Literal |
|---|---|
| Syntax: | [ *label* ] SUBLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | k - (W) → (W) |
| Status Affected: | C, DC, Z |
| Description: | The W register is subtracted (2's complement method) from the eight-bit literal 'k'. The result is placed in the W register. |

| SUBWF | Subtract W from f |
|---|---|
| Syntax: | [ *label* ] SUBWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - (W) → (destination) |
| Status Affected: | C, DC, Z |
| Description: | Subtract (2's complement method) W register from register 'f'. If 'd' = '0', the result is stored in the W register. If 'd' = '1', the result is stored back in register 'f'. |

| SWAPF | Swap Nibbles in f |
|---|---|
| Syntax: | [ *label* ] SWAPF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f<3:0>) → (destination<7:4>),<br>(f<7:4>) → (destination<3:0>) |
| Status Affected: | None |
| Description: | The upper and lower nibbles of register 'f' are exchanged. If 'd' = '0', the result is placed in W register. If 'd' = '1', the result is placed in register 'f'. |

| XORLW | Exclusive OR Literal with W |
|---|---|
| Syntax: | [ *label* ] XORLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | (W) .XOR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register. |

| XORWF | Exclusive OR W with f |
|---|---|
| Syntax: | [ *label* ] XORWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) .XOR. (f) → (destination) |
| Status Affected: | Z |
| Description: | Exclusive OR the contents of the W register with register 'f'. If 'd' = 0, the result is stored in the W register. If 'd' = '1', the result is stored back in register 'f'. |

## 17.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C17 and MPLAB C18 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - ICEPIC™ In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD
- Device Programmers
  - PRO MATE® II Universal Device Programmer
  - PICSTART® Plus Entry-Level Development Programmer
- Low Cost Demonstration Boards
  - PICDEM™ 1 Demonstration Board
  - PICDEM 2 Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 17 Demonstration Board
  - KEELOQ® Demonstration Board

### 17.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8-bit microcontroller market. The MPLAB IDE is a Windows®-based application that contains:

- An interface to debugging tools
  - simulator
  - programmer (sold separately)
  - emulator (sold separately)
  - in-circuit debugger (sold separately)
- A full-featured editor
- A project manager
- Customizable toolbar and key mapping
- A status bar
- On-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or ‘C’)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
  - machine code

The ability to use MPLAB IDE with multiple debugging tools allows users to easily switch from the cost-effective simulator to a full-featured emulator with minimal retraining.

### 17.2 MPASM Assembler

The MPASM assembler is a full-featured universal macro assembler for all PICmicro MCU’s.

The MPASM assembler has a command line interface and a Windows shell. It can be used as a stand-alone application on a Windows 3.x or greater system, or it can be used through MPLAB IDE. The MPASM assembler generates relocatable object files for the MPLINK object linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, an absolute LST file that contains source lines and generated machine code, and a COD file for debugging.

The MPASM assembler features include:

- Integration into MPLAB IDE projects.
- User-defined macros to streamline assembly code.
- Conditional assembly for multi-purpose source files.
- Directives that allow complete control over the assembly process.

### 17.3 MPLAB C17 and MPLAB C18 C Compilers

The MPLAB C17 and MPLAB C18 Code Development Systems are complete ANSI ‘C’ compilers for Microchip’s PIC17CXXX and PIC18CXXX family of microcontrollers, respectively. These compilers provide powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compilers provide symbol information that is compatible with the MPLAB IDE memory display.

## 17.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can also link relocatable objects from pre-compiled libraries, using directives from a linker script.

The MPLIB object librarian is a librarian for pre-compiled code to be used with the MPLINK object linker. When a routine from a library is called from another source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. The MPLIB object librarian manages the creation and modification of library files.

The MPLINK object linker features include:

- Integration with MPASM assembler and MPLAB C17 and MPLAB C18 C compilers.
- Allows all memory areas to be defined as sections to provide link-time flexibility.

The MPLIB object librarian features include:

- Easier linking because single libraries can be included instead of many smaller files.
- Helps keep code maintainable by grouping related modules together.
- Allows libraries to be created and modules to be added, listed, replaced, deleted or extracted.

## 17.5 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC-hosted environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user-defined key press, to any of the pins. The execution can be performed in single step, execute until break, or trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and the MPLAB C18 C compilers and the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent multi-project software development tool.

## 17.6 MPLAB ICE High Performance Universal In-Circuit Emulator with MPLAB IDE

The MPLAB ICE universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers (MCUs). Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE in-circuit emulator system has been designed as a real-time emulation system, with advanced features that are generally found on more expensive development tools. The PC platform and Microsoft® Windows environment were chosen to best make these features available to you, the end user.

## 17.7 ICEPIC In-Circuit Emulator

The ICEPIC low cost, in-circuit emulator is a solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X and PIC16CXXX families of 8-bit One-Time-Programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules, or daughter boards. The emulator is capable of emulating without target application circuitry being present.

## 17.8 MPLAB ICD In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD, is a powerful, low cost, run-time development tool. This tool is based on the FLASH PICmicro MCUs and can be used to develop for this and other PICmicro microcontrollers. The MPLAB ICD utilizes the in-circuit debugging capability built into the FLASH devices. This feature, along with Microchip's In-Circuit Serial Programming™ protocol, offers cost-effective in-circuit FLASH debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time.

## 17.9 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable $V_{DD}$ and $V_{PP}$ supplies, which allow it to verify programmed memory at $V_{DD}$ min and $V_{DD}$ max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In stand-alone mode, the PRO MATE II device programmer can read, verify, or program PICmicro devices. It can also set code protection in this mode.

## 17.10 PICSTART Plus Entry Level Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient.

The PICSTART Plus development programmer supports all PICmicro devices with up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

## 17.11 PICDEM 1 Low Cost PICmicro Demonstration Board

The PICDEM 1 demonstration board is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The user can program the sample microcontrollers provided with the PICDEM 1 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The user can also connect the PICDEM 1 demonstration board to the MPLAB ICE in-circuit emulator and download the firmware to the emulator for testing. A prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs connected to PORTB.

## 17.12 PICDEM 2 Low Cost PIC16CXX Demonstration Board

The PICDEM 2 demonstration board is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 2 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a serial EEPROM to demonstrate usage of the $I^2C$™ bus and separate headers for connection to an LCD module and a keypad.

## 17.13 PICDEM 3 Low Cost PIC16CXXX Demonstration Board

The PICDEM 3 demonstration board is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with an LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 3 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer with an adapter socket, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 3 demonstration board to test firmware. A prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM 3 demonstration board is a LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM 3 demonstration board provides an additional RS-232 interface and Windows software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals.

## 17.14 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. All necessary hardware is included to run basic demo programs, which are supplied on a 3.5-inch disk. A programmed sample is included and the user may erase it and program it with the other sample programs using the PRO MATE II device programmer, or the PICSTART Plus development programmer, and easily debug and test the sample code. In addition, the PICDEM 17 demonstration board supports downloading of programs to and executing out of external FLASH memory on board. The PICDEM 17 demonstration board is also usable with the MPLAB ICE in-circuit emulator, or the PICMASTER emulator and all of the sample programs can be run and modified using either emulator. Additionally, a generous prototype area is available for user hardware.

## 17.15 KEELOQ Evaluation and Programming Tools

KEELOQ evaluation and programming tools support Microchip's HCS Secure Data Products. The HCS evaluation kit includes a LCD display to show changing codes, a decoder to decode transmissions and a programming interface to program test transmitters.

**TABLE 17-1: DEVELOPMENT TOOLS FROM MICROCHIP**

| | | PIC12CXXX | PIC14000 | PIC16C5X | PIC16C6X | PIC16CXXX | PIC16F62X | PIC16C7X | PIC16C7XX | PIC16C8X/ PIC16F8X | PIC16F8XX | PIC16C9XX | PIC17C4X | PIC17C7XX | PIC18CXX2 | PIC18FXXX | 24CXX/ 25CXX/ 93CXX | HCSXXX | MCRFXXX | MCP2510 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Software Tools | MPLAB® Integrated Development Environment | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | |
| | MPLAB® C17 C Compiler | | | | | | | | | | | | ✓ | ✓ | | | | | | |
| | MPLAB® C18 C Compiler | | | | | | | | | | | | | | ✓ | ✓ | | | | |
| | MPASM™ Assembler/ MPLINK™ Object Linker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Emulators | MPLAB® ICE In-Circuit Emulator | ✓ | ✓ | ✓ | ✓ | ✓ | ✓** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | |
| | ICEPIC™ In-Circuit Emulator | ✓ | | ✓ | ✓ | ✓ | | ✓ | ✓ | ✓ | | ✓ | | | | | | | | |
| Debugger | MPLAB® ICD In-Circuit Debugger | | | | ✓* | | | ✓* | | ✓ | | | | | | ✓ | | | | |
| Programmers | PICSTART® Plus Entry Level Development Programmer | ✓ | ✓ | ✓ | ✓ | ✓ | ✓** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | |
| | PRO MATE® II Universal Device Programmer | ✓ | ✓ | ✓ | ✓ | ✓ | ✓** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Demo Boards and Eval Kits | PICDEM™ 1 Demonstration Board | | | ✓ | | ✓ | | ✓† | | ✓ | | | ✓ | | | | | | | |
| | PICDEM™ 2 Demonstration Board | | | | ✓† | | | ✓† | | | | | | | ✓ | ✓ | | | | |
| | PICDEM™ 3 Demonstration Board | | | | | | | | | | | ✓ | | | | | | | | |
| | PICDEM™ 14A Demonstration Board | | ✓ | | | | | | | | | | | | | | | | | |
| | PICDEM™ 17 Demonstration Board | | | | | | | | | | | | | ✓ | | | | | | |
| | KEELOQ® Evaluation Kit | | | | | | | | | | | | | | | | ✓ | ✓ | ✓ | |
| | KEELOQ® Transponder Kit | | | | | | | | | | | | | | | | | ✓ | | |
| | microID™ Programmer's Kit | | | | | | | | | | | | | | | | | | ✓ | |
| | 125 kHz microID™ Developer's Kit | | | | | | | | | | | | | | | | | | ✓ | |
| | 125 kHz Anticollision microID™ Developer's Kit | | | | | | | | | | | | | | | | | | ✓ | |
| | 13.56 MHz Anticollision microID™ Developer's Kit | | | | | | | | | | | | | | | | | | ✓ | |
| | MCP2510 CAN Developer's Kit | | | | | | | | | | | | | | | | | | | ✓ |

* Contact the Microchip Technology Inc. web site at www.microchip.com for information on how to use the MPLAB® ICD In-Circuit Debugger (DV164001) with PIC16C62, 63, 64, 65, 72, 73, 74, 76, 77.

** Contact Microchip Technology Inc. for availability date.

† Development tool is available on select devices.

**NOTES:**

## 18.0 ELECTRICAL CHARACTERISTICS

**Absolute Maximum Ratings †**

Ambient temperature under bias ..... -55°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on any pin with respect to $V_{SS}$ (except $V_{DD}$ and $\overline{MCLR}$) ..... -0.3V to ($V_{DD}$ + 0.3V)
Voltage on $V_{DD}$ with respect to $V_{SS}$ ..... -0.3 to +7.5V
Voltage on $\overline{MCLR}$ with respect to $V_{SS}$ **(Note 2)** ..... -0.3 to +14V
Total power dissipation **(Note 1)** ..... 1W
Maximum current out of $V_{SS}$ pin ..... 200 mA
Maximum current into $V_{DD}$ pin ..... 200 mA
Input clamp current, $I_{IK}$ ($V_I$ < 0 or $V_I$ > $V_{DD}$) ..... ± 20 mA
Output clamp current, $I_{OK}$ ($V_O$ < 0 or $V_O$ > $V_{DD}$) ..... ± 20 mA
Maximum output current sunk by any I/O pin ..... 25 mA
Maximum output current sourced by any I/O pin ..... 25 mA
Maximum current sunk by PORTA ..... 100 mA
Maximum current sourced by PORTA ..... 100 mA
Maximum current sunk by PORTB ..... 100 mA
Maximum current sourced by PORTB ..... 100 mA

**Note 1:** Power dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OI} \times I_{OL})$

**2:** Voltage spikes at the $\overline{MCLR}$ pin may cause latchup. A series resistor of greater than 1 kΩ should be used to pull $\overline{MCLR}$ to $V_{DD}$, rather than tying the pin directly to $V_{DD}$.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

**FIGURE 18-1: PIC16F87/88 VOLTAGE-FREQUENCY GRAPH**

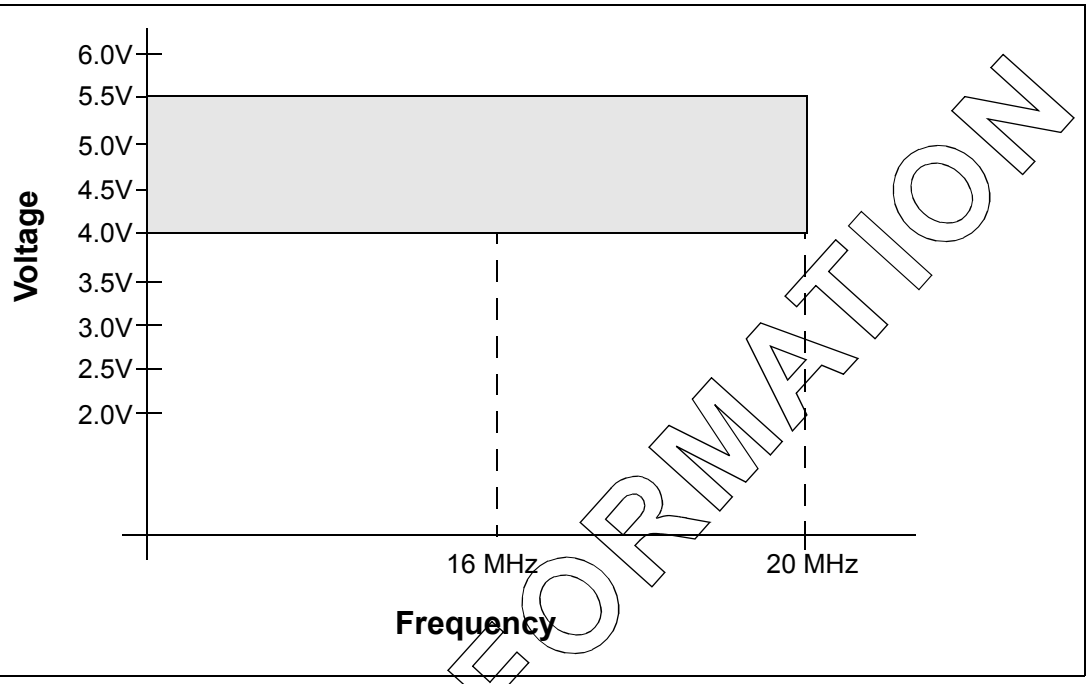
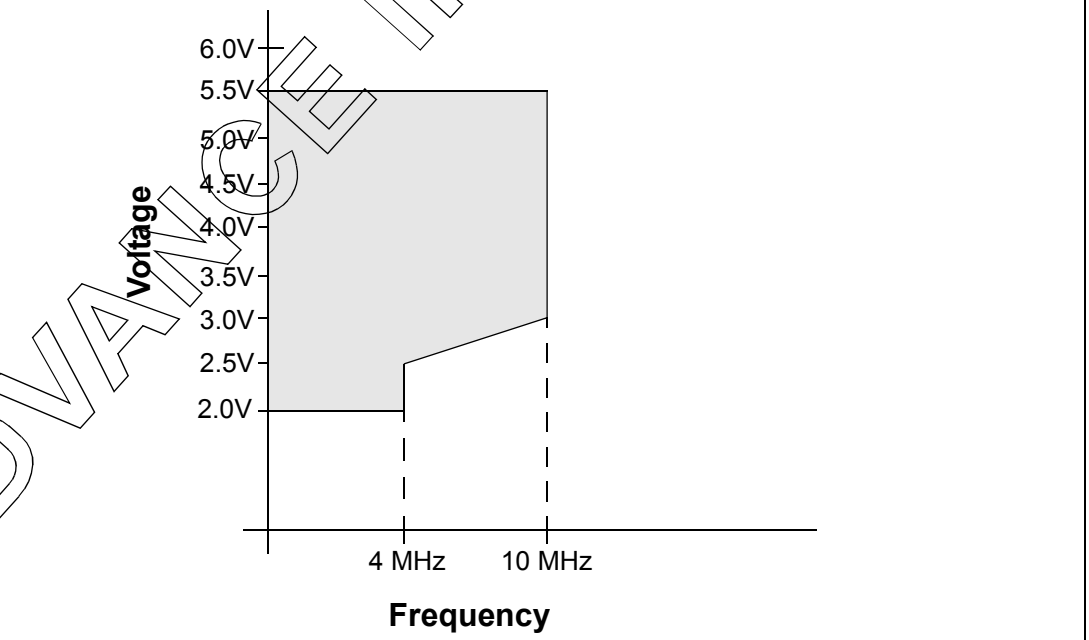



**FIGURE 18-2: PIC16LF87/88 VOLTAGE-FREQUENCY GRAPH**




FMAX = (12 MHz/V) (VDDAPPMIN - 2.5V) + 4 MHz

**Note 1:** VDDAPPMIN is the minimum voltage of the PICmicro® device in the application.

**2:** FMAX has a maximum frequency of 10 MHz.



## 18.1 DC Characteristics: Supply Voltage PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | |
| **PIC16F87/88** (Industrial, Extended) | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Conditions** |
| | VDD | **Supply Voltage** | | | | | |
| D001 | | PIC16LF87/88 | 2.0 | — | 5.5 | V | HS, XT, RC and LP Osc mode |
| D001 | | PIC16F87/88 | 4.0 | — | 5.5 | V | |
| D002 | VDR | **RAM Data Retention Voltage(1)** | 1.5 | — | — | V | |
| D003 | VPOR | **VDD Start Voltage** to ensure internal Power-on Reset signal | — | — | 0.7 | V | See Section 15.4, "Power-on Reset (POR)" for details |
| D004 | SVDD | **VDD Rise Rate** to ensure internal Power-on Reset signal | 0.05 | — | — | V/ms | See Section 15.4, "Power-on Reset (POR)" for details |
| | VBOR | **Brown-out Reset Voltage** | | | | | |
| D005 | | PIC16LF87/88 | 3.65 | — | 4.35 | V | |
| D005 | | PIC16F87/88 | 3.65 | — | 4.35 | V | FMAX = 14 MHz(2) |

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

**Note 1:** This is the limit to which VDD can be lowered in SLEEP mode, or during a device RESET, without losing RAM data.

**2:** When BOR is enabled, the device will operate correctly until the VBOR voltage trip point is reached.

## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Power-down Current (IPD)(1)** | | | | | | |
| | PIC16LF87/88 | 0.2 | TBD | µA | -40°C | VDD = 2.0V | |
| | | 0.2 | TBD | µA | 25°C | | |
| | | 0.3 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 0.3 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 0.3 | TBD | µA | 25°C | | |
| | | 0.4 | TBD | µA | 85°C | | |
| | All devices | 0.4 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 0.5 | TBD | µA | 25°C | | |
| | | 0.6 | TBD | µA | 85°C | | |

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;

$\overline{MCLR}$ = 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.



## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| | |
|---|---|
| **PIC16LF87/88** (Industrial) | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial |
| **PIC16F87/88** (Industrial, Extended) | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended |

| Param No. | Device | Typ | Max | Units | Conditions | | |
|---|---|---|---|---|---|---|---|
| | PIC16LF87/88 | 8 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 32 kHz (LP Oscillator) |
| | | 10 | TBD | µA | 25°C | | |
| | | 14 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 17 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 16 | TBD | µA | 25°C | | |
| | | 15 | TBD | µA | 85°C | | |
| | All devices | 34 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 28 | TBD | µA | 25°C | | |
| | | 25 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 85 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (RC Oscillator) |
| | | 87 | TBD | µA | 25°C | | |
| | | 83 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 200 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 165 | TBD | µA | 25°C | | |
| | | 150 | TBD | µA | 85°C | | |
| | All devices | 408 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 338 | TBD | µA | 25°C | | |
| | | 300 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 233 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (RC Oscillator) |
| | | 240 | TBD | µA | 25°C | | |
| | | 243 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 466 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 429 | TBD | µA | 25°C | | |
| | | 416 | TBD | µA | 85°C | | |
| | All devices | 972 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 874 | TBD | µA | 25°C | | |
| | | 835 | TBD | µA | 85°C | | |

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;

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.

## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | All devices | 1.4 | TBD | mA | -40°C | VDD = 4.0V | FOSC = 20 MHz (HS Oscillator) |
| | | 1.3 | TBD | mA | 25°C | | |
| | | 1.0 | TBD | mA | 85°C | | |
| | All devices | 2.4 | TBD | mA | -40°C | VDD = 5.0V | |
| | | 1.8 | TBD | mA | 25°C | | |
| | | 1.6 | TBD | mA | 85°C | | |

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;

$\overline{MCLR}$ = 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.



## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC16LF87/88 | 7 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 31.25 kHz (**RC_RUN** mode, Internal RC Oscillator) |
| | | 7 | TBD | µA | 25°C | | |
| | | 8 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 16 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 14 | TBD | µA | 25°C | | |
| | | 13 | TBD | µA | 85°C | | |
| | All devices | 35 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 28 | TBD | µA | 25°C | | |
| | | 25 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 111 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**RC_RUN** mode, Internal RC Oscillator) |
| | | 116 | TBD | µA | 25°C | | |
| | | 122 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 164 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 162 | TBD | µA | 25°C | | |
| | | 165 | TBD | µA | 85°C | | |
| | All devices | 278 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 266 | TBD | µA | 25°C | | |
| | | 266 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 288 | TBD | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**RC_RUN** mode, Internal RC Oscillator) |
| | | 294 | TBD | µA | 25°C | | |
| | | 299 | TBD | µA | 85°C | | |
| | PIC16LF87/88 | 441 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 428 | TBD | µA | 25°C | | |
| | | 428 | TBD | µA | 85°C | | |
| | All devices | 791 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 752 | TBD | µA | 25°C | | |
| | | 747 | TBD | µA | 85°C | | |

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;

$\overline{MCLR}$ = 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.

## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC16LF87/88 | 847 | TBD | µA | -40°C | VDD = 3.0V | FOSC = 8 MHz (**RC_RUN** mode, Internal RC Oscillator) |
| | | 796 | TBD | µA | 25°C | | |
| | | 784 | TBD | µA | 85°C | | |
| | All devices | 1.6 | TBD | mA | -40°C | VDD = 5.0V | |
| | | 1.5 | TBD | mA | 25°C | | |
| | | 1.4 | TBD | mA | 85°C | | |
| | PIC16LF87/88 | 13 | TBD | µA | -10°C | VDD = 2.0V | FOSC = 32 kHz (**SEC_RUN** mode, Timer1 as clock) |
| | | 14 | TBD | µA | 25°C | | |
| | | 16 | TBD | µA | 70°C | | |
| | PIC16LF87/88 | 34 | TBD | µA | -10°C | VDD = 3.0V | |
| | | 31 | TBD | µA | 25°C | | |
| | | 28 | TBD | µA | 70°C | | |
| | All devices | 72 | TBD | µA | -10°C | VDD = 5.0V | |
| | | 65 | TBD | µA | 25°C | | |
| | | 59 | TBD | µA | 70°C | | |

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;
$\overline{MCLR}$ = 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.



## 18.2 DC Characteristics: Power-down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)** | | | | | | |
| D022 (ΔIWDT) | **Watchdog Timer** | 1.3 | TBD | µA | -40°C | VDD = 2.0V | |
| | | 0.7 | TBD | µA | 25°C | | |
| | | 0.2 | TBD | µA | 85°C | | |
| | | 1.0 | TBD | µA | -40°C | VDD = 3.0V | |
| | | 1.4 | TBD | µA | 25°C | | |
| | | 2.4 | TBD | µA | 85°C | | |
| | | 1.9 | TBD | µA | -40°C | VDD = 5.0V | |
| | | 2.0 | TBD | µA | 25°C | | |
| | | 3.0 | TBD | µA | 85°C | | |
| D022A (ΔIBOR) | **Brown-out Reset** | 85 | TBD | µA | -40°C to +85°C | VDD = 5.0V | |
| D025 (ΔIOSCB) | **Timer1 Oscillator** | 1.3 | TBD | µA | -10°C | VDD = 2.0V | 32 kHz on Timer1 |
| | | 1.3 | TBD | µA | 25°C | | |
| | | 1.4 | TBD | µA | 70°C | | |
| | | 1.6 | TBD | µA | -10°C | VDD = 3.0V | |
| | | 1.6 | TBD | µA | 25°C | | |
| | | 1.7 | TBD | µA | 70°C | | |
| | | 2.8 | TBD | µA | -10°C | VDD = 5.0V | |
| | | 2.8 | TBD | µA | 25°C | | |
| | | 3.0 | TBD | µA | 70°C | | |
| D026 (ΔIAD) | **A/D Converter** | 44 | TBD | µA | | VDD = 2.0V | A/D on, not converting |
| | | 53 | TBD | µA | | VDD = 3.0V | |
| | | 61 | TBD | µA | | VDD = 5.0V | |

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;
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.

## 18.3 DC Characteristics: Internal RC Accuracy PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **PIC16LF87/88** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | | |
| **PIC16F87/88** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | | |
| **Param No.** | **Device** | **Min** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz(1)** | | | | | | | |
| | PIC16LF87/88 | TBD | +/-1 | TBD | % | 25°C | VDD = 2.0V | |
| | | TBD | +/-1 | TBD | % | 25°C | VDD = 3.0V | |
| | All devices | TBD | +/-1 | TBD | % | 25°C | VDD = 5.0V | |
| | **INTRC Accuracy @ Freq = 31.25 kHz(2)** | | | | | | | |
| | PIC16LF87/88 | 28.125 | 31.25 | 34.375 | kHz | 25°C | VDD = 2.0V | |
| | | 28.125 | 31.25 | 34.375 | kHz | 25°C | VDD = 3.0V | |
| | All devices | 28.125 | 31.25 | 34.375 | kHz | 25°C | VDD = 5.0V | |
| | **INTRC Stability(3)** | | | | | | | |
| | PIC16LF87/88 | TBD | 1 | TBD | % | 25°C | VDD = 2.0V | |
| | | TBD | 1 | TBD | % | 25°C | VDD = 3.0V | |
| | All devices | TBD | 1 | TBD | % | 25°C | VDD = 5.0V | |

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

**Note 1:** Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.

**2:** INTRC is used to calibrate INTOSC.

**3:** Change of INTRC frequency as VDD changes.



## 18.4 DC Characteristics: PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended<br>Operating voltage VDD range as described in DC Specification, Section 18.0. | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage** | | | | | |
| | | I/O ports: | | | | | |
| D030 | | with TTL buffer | VSS | — | 0.15 VDD | V | For entire VDD range |
| D030A | | | VSS | — | 0.8V | V | 4.5V ≤ VDD ≤ 5.5V |
| D031 | | with Schmitt Trigger buffer | VSS | — | 0.2 VDD | V | |
| D032 | | $\overline{MCLR}$, OSC1 (in RC mode) | VSS | — | 0.2 VDD | V | **(Note 1)** |
| D033 | | OSC1 (in XT and LP mode) | VSS | — | 0.3V | V | |
| | | OSC1 (in HS mode) | VSS | — | 0.3 VDD | V | |
| | | Ports RB1 and RB4: | | | | | |
| D034 | | with Schmitt Trigger buffer | VSS | — | 0.3 VDD | V | **For entire** VDD range |
| | VIH | **Input High Voltage** | | | | | |
| | | I/O ports: | | | | | |
| D040 | | with TTL buffer | 2.0 | — | VDD | V | 4.5V ≤ VDD ≤ 5.5V |
| D040A | | | 0.25 VDD + 0.8V | — | VDD | V | For entire VDD range |
| D041 | | with Schmitt Trigger buffer | 0.8 VDD | — | VDD | V | For entire VDD range |
| D042 | | $\overline{MCLR}$ | 0.8 VDD | — | VDD | V | |
| D042A | | OSC1 (in XT and LP mode) | 1.6V | — | VDD | V | |
| | | OSC1 (in HS mode) | 0.7 VDD | — | VDD | V | |
| D043 | | OSC1 (in RC mode) | 0.9 VDD | — | VDD | V | **(Note 1)** |
| | | Ports RB1 and RB4: | | | | | |
| D044 | | with Schmitt Trigger buffer | 0.7 VDD | — | VDD | V | **For entire** VDD range |
| D070 | IPURB | **PORTB Weak Pull-up Current** | 50 | 250 | 400 | µA | VDD = 5V, VPIN = VSS |
| | IIL | **Input Leakage Current (Notes 2, 3)** | | | | | |
| D060 | | I/O ports | — | — | ±1 | µA | Vss ≤ VPIN ≤ VDD, pin at hi-impedance |
| D061 | | $\overline{MCLR}$ | — | — | ±5 | µA | Vss ≤ VPIN ≤ VDD |
| D063 | | OSC1 | — | — | ±5 | µA | Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC16F87/88 be driven with external clock in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**3:** Negative current is defined as current sourced by the pin.

## 18.4 DC Characteristics: PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended<br>Operating voltage VDD range as described in DC Specification, Section 18.0. | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | VOL | **Output Low Voltage** | | | | | |
| D080 | | I/O ports | — | — | 0.6 | V | IOL = 8.5 mA, VDD = 4.5V, -40°C to +125°C |
| D083 | | OSC2/CLKO (RC osc config) | — | — | 0.6 | V | IOL = 1.6 mA, VDD = 4.5V, -40°C to +125°C |
| | VOH | **Output High Voltage** | | | | | |
| D090 | | I/O ports **(Note 3)** | VDD – 0.7 | — | — | V | IOH = -3.0 mA, VDD = 4.5V, -40°C to +125°C |
| D092 | | OSC2/CLKO (RC osc config) | VDD – 0.7 | — | — | V | IOH = -1.3 mA, VDD = 4.5V, -40°C to +125°C |
| | | **Capacitive Loading Specs on Output Pins** | | | | | |
| D100 | COSC2 | OSC2 pin | — | — | 15 | pF | In XT, HS and LP modes when external clock is used to drive OSC1 |
| D101 | CIO | All I/O pins and OSC2 (in RC mode) | — | — | 50 | pF | |
| D102 | CB | SCL, SDA in $I^2C$ mode | — | — | 400 | pF | |
| | | **Data EEPROM Memory** | | | | | |
| D120 | ED | Endurance | 100K<br>10K | 1M<br>100K | —<br>— | E/W<br>E/W | -40°C to 85°C<br>+85°C to +125°C |
| D121 | VDRW | VDD for read/write | VMIN | — | 5.5 | V | Using EECON to read/write, VMIN = min. operating voltage |
| D122 | TDEW | Erase/write cycle time | — | 4 | 8 | ms | |
| | | **Program FLASH Memory** | | | | | |
| D130 | EP | Endurance | 10K<br>1K | 100K<br>10K | —<br>— | E/W<br>E/W | -40°C to 85°C<br>+85°C to +125°C |
| D131 | VPR | VDD for read | VMIN | — | 5.5 | V | |
| D132A | | VDD for erase/write | VMIN | — | 5.5 | V | Using EECON to read/write, VMIN = min. operating voltage |
| D133 | TPE | Erase cycle time | — | 2 | 4 | ms | |
| D134 | TPW | Write cycle time | — | 2 | 4 | ms | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC16F87/88 be driven with external clock in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**3:** Negative current is defined as current sourced by the pin.



**TABLE 18-1: COMPARATOR SPECIFICATIONS**

| Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +85°C, unless otherwise stated. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Characteristics** | **Sym** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| D300 | Input Offset Voltage | VIOFF | — | ± 5.0 | ± 10 | mV | |
| D301 | Input Common Mode Voltage* | VICM | 0 | - | VDD – 1.5 | V | |
| D302 | Common Mode Rejection Ratio* | CMRR | 55 | - | — | dB | |
| 300<br>300A | Response Time(1)* | TRESP | — | 150 | 400<br>600 | ns<br>ns | PIC16F87/88<br>PIC16LF87/88 |
| 301 | Comparator Mode Change to Output Valid* | TMC2OV | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at (VDD – 1.5)/2 while the other input transitions from VSS to VDD.

**TABLE 18-2: VOLTAGE REFERENCE SPECIFICATIONS**

| Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +85°C, unless otherwise stated. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Spec No.** | **Characteristics** | **Sym** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| D310 | Resolution | VRES | VDD/24 | — | VDD/32 | LSb | |
| D311 | Absolute Accuracy | VRAA | —<br>— | —<br>— | 1/4<br>1/2 | LSb<br>LSb | Low Range (VRR = 1)<br>High Range (VRR = 0) |
| D312 | Unit Resistor Value (R)* | VRUR | — | 2 k | — | Ω | |
| 310 | Settling Time(1)* | TSET | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while VRR = 1 and VR<3:0> transitions from 0000 to 1111.

## 18.5 Timing Parameter Symbology

The timing parameter symbols have been created using one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST ($I^2C$ specifications only)
4. Ts ($I^2C$ specifications only)

| | | | |
|---|---|---|---|
| **T** | | | |
| F | Frequency | T | Time |

Lowercase letters (pp) and their meanings:

| | | | |
|---|---|---|---|
| **pp** | | | |
| cc | CCP1 | osc | OSC1 |
| ck | CLKO | rd | $\overline{RD}$ |
| cs | $\overline{CS}$ | rw | $\overline{RD}$ or $\overline{WR}$ |
| di | SDI | sc | SCK |
| do | SDO | ss | $\overline{SS}$ |
| dt | Data in | t0 | T0CKI |
| io | I/O port | t1 | T1CKI |
| mc | $\overline{MCLR}$ | wr | $\overline{WR}$ |

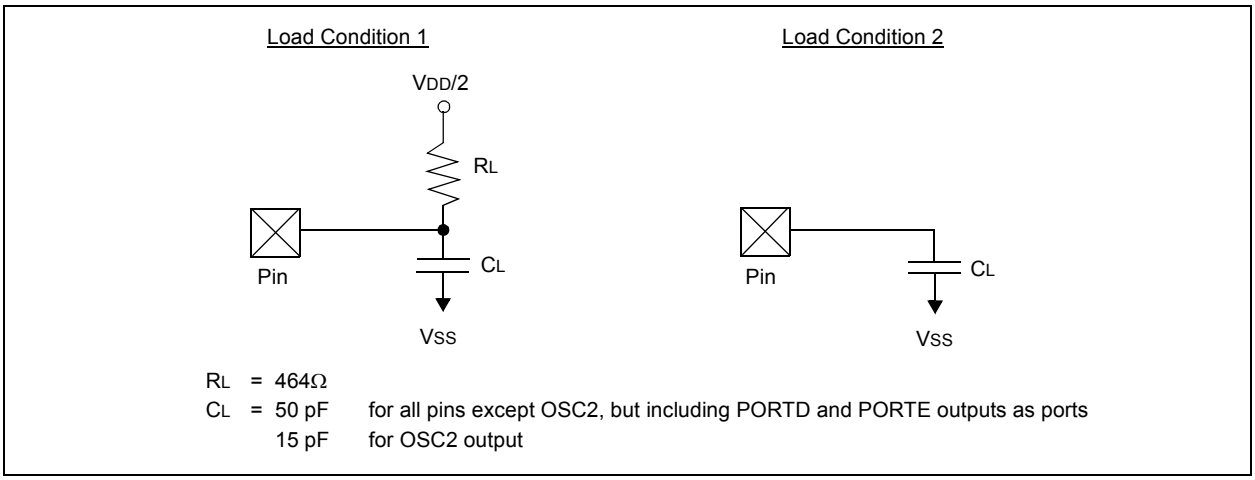
Uppercase letters and their meanings:

| | | | |
|---|---|---|---|
| **S** | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (Hi-impedance) | V | Valid |
| L | Low | Z | Hi-impedance |
| **$I^2C$ only** | | | |
| AA | output access | High | High |
| BUF | Bus free | Low | Low |

TCC:ST ($I^2C$ specifications only)

| | | | |
|---|---|---|---|
| **CC** | | | |
| HD | Hold | SU | Setup |
| **ST** | | | |
| DAT | DATA input hold | STO | STOP condition |
| STA | START condition | | |

**FIGURE 18-3: LOAD CONDITIONS**




RL = 464Ω
CL = 50 pF for all pins except OSC2, but including PORTD and PORTE outputs as ports
15 pF for OSC2 output

**FIGURE 18-4: EXTERNAL CLOCK TIMING**

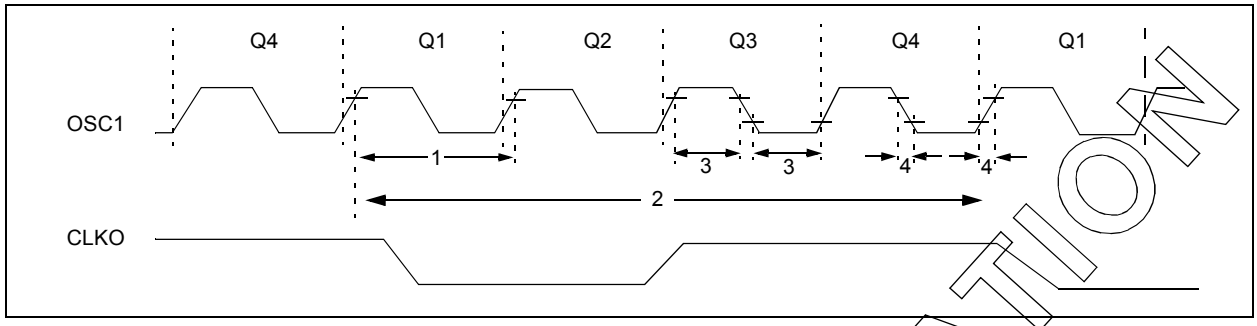


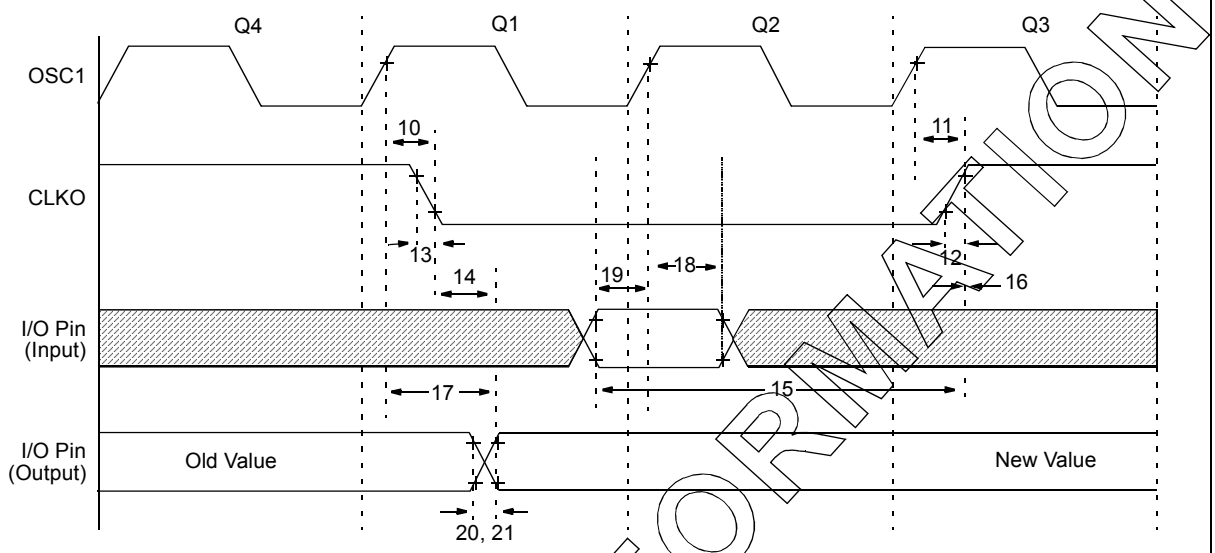

**TABLE 18-3: EXTERNAL CLOCK TIMING REQUIREMENTS**

| Parameter No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| | FOSC | External CLKI Frequency (Note 1) | DC | — | 4 | MHz | XT and RC Osc mode |
| | | | DC | — | 20 | MHz | HS Osc mode |
| | | | DC | — | 200 | kHz | LP Osc mode |
| | | Oscillator Frequency (Note 1) | DC | — | 4 | MHz | RC Osc mode |
| | | | 0.1 | — | 4 | MHz | XT Osc mode |
| | | | 4 | — | 20 | MHz | HS Osc mode |
| | | | 5 | — | 200 | kHz | LP Osc mode |
| 1 | TOSC | External CLKI Period (Note 1) | 250 | — | — | ns | XT and RC Osc mode |
| | | | 50 | — | — | ns | HS Osc mode |
| | | | 5 | — | — | µs | LP Osc mode |
| | | Oscillator Period (Note 1) | 250 | — | — | ns | RC Osc mode |
| | | | 250 | — | 10,000 | ns | XT Osc mode |
| | | | 100 | — | 250 | ns | HS Osc mode |
| | | | 50 | — | 250 | ns | HS Osc mode |
| | | | 5 | — | — | µs | LP Osc mode |
| 2 | TCY | Instruction Cycle Time (Note 1) | 200 | TCY | DC | ns | TCY = 4/FOSC |
| 3 | TosL, TosH | External Clock in (OSC1) High or Low Time | 100 | — | — | ns | XT oscillator |
| | | | 2.5 | — | — | µs | LP oscillator |
| | | | 15 | — | — | ns | HS oscillator |
| 4 | TosR, TosF | External Clock in (OSC1) Rise or Fall Time | — | — | 25 | ns | XT oscillator |
| | | | — | — | 50 | ns | LP oscillator |
| | | | — | — | 15 | ns | HS oscillator |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Instruction cycle period (TCY) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions, with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the "max." cycle time limit is "DC" (no clock) for all devices.

**FIGURE 18-5: CLKO AND I/O TIMING**




**Note:** Refer to Figure 18-3 for load conditions.

**TABLE 18-4: CLKO AND I/O TIMING REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 10* | TosH2ckL | OSC1↑ to CLKO↓ | | — | 75 | 200 | ns | **(Note 1)** |
| 11* | TosH2ckH | OSC1↑ to CLKO↑ | | — | 75 | 200 | ns | **(Note 1)** |
| 12* | TckR | CLKO rise time | | — | 35 | 100 | ns | **(Note 1)** |
| 13* | TckF | CLKO fall time | | — | 35 | 100 | ns | **(Note 1)** |
| 14* | TckL2ioV | CLKO ↓ to Port out valid | | — | — | 0.5 TCY + 20 | ns | **(Note 1)** |
| 15* | TioV2ckH | Port in valid before CLKO ↑ | | TOSC + 200 | — | — | ns | **(Note 1)** |
| 16* | TckH2ioI | Port in hold after CLKO ↑ | | 0 | — | — | ns | **(Note 1)** |
| 17* | TosH2ioV | OSC1↑ (Q1 cycle) to Port out valid | | — | 100 | 255 | ns | |
| 18* | TosH2ioI | OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time) | PIC16**F**87/88 | 100 | — | — | ns | |
| | | | PIC16**LF**87/88 | 200 | — | — | ns | |
| 19* | TioV2osH | Port input valid to OSC1↑ (I/O in setup time) | | 0 | — | — | ns | |
| 20* | TIOR | Port output rise time | PIC16**F**87/88 | — | 10 | 40 | ns | |
| | | | PIC16**LF**87/88 | — | — | 145 | ns | |
| 21* | TIOF | Port output fall time | PIC16**F**87/88 | — | 10 | 40 | ns | |
| | | | PIC16**LF**87/88 | — | — | 145 | ns | |
| 22††* | TINP | INT pin high or low time | | TCY | — | — | ns | |
| 23††* | TRBP | RB7:RB4 change INT high or low time | | TCY | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

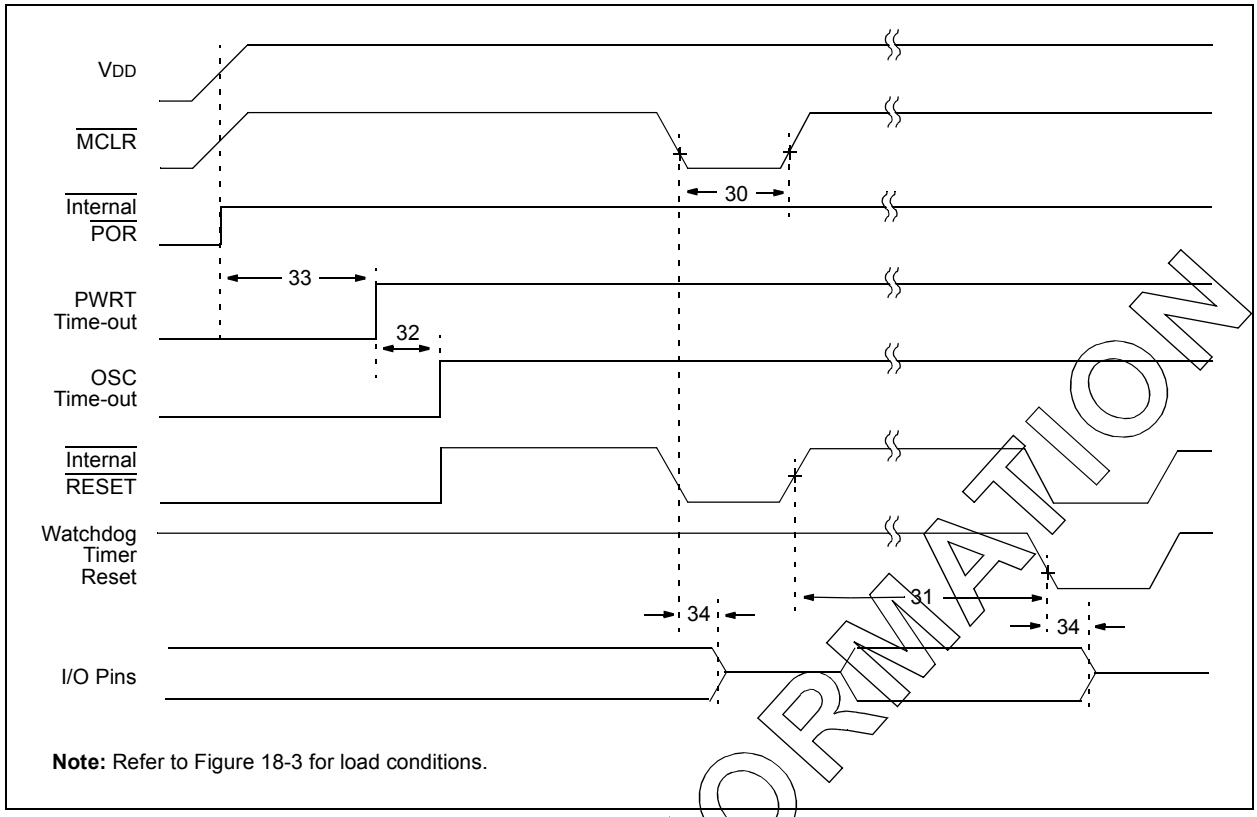
†† These parameters are asynchronous events, not related to any internal clock edges.

**Note 1:** Measurements are taken in RC mode where CLKO output is 4 x TOSC.



**FIGURE 18-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**




**Note:** Refer to Figure 18-3 for load conditions.

**FIGURE 18-7: BROWN-OUT RESET TIMING**

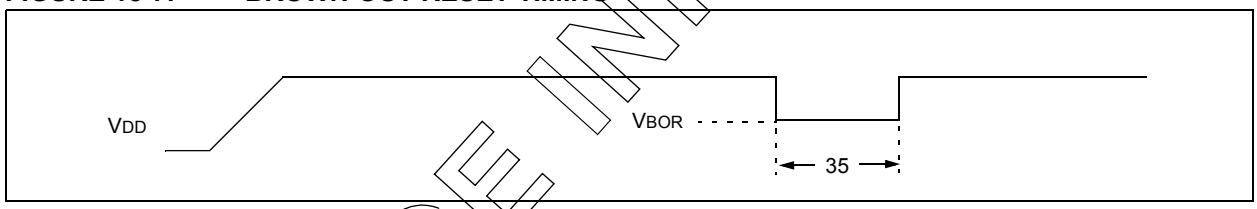


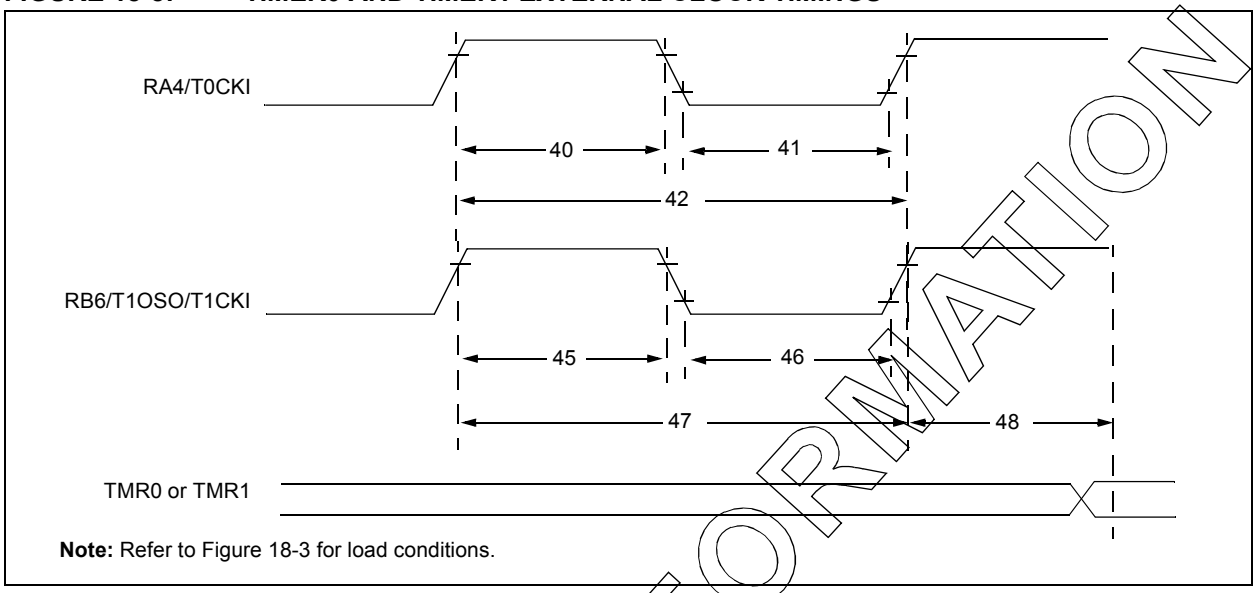

**TABLE 18-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER, AND BROWN-OUT RESET REQUIREMENTS**

| Parameter No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 30 | TmcL | MCLR Pulse Width (Low) | 2 | — | — | μs | VDD = 5V, -40°C to +85°C |
| 31* | TWDT | Watchdog Timer Time-out Period (No Prescaler) | TBD | 16 | TBD | ms | VDD = 5V, -40°C to +85°C |
| 32 | TOST | Oscillation Start-up Timer Period | — | 1024 TOSC | — | — | TOSC = OSC1 period |
| 33* | TPWRT | Power-up Timer Period | TBD | 72 | TBD | ms | VDD = 5V, -40°C to +85°C |
| 34 | TIOZ | I/O Hi-impedance from MCLR Low or Watchdog Timer Reset | — | — | 2.1 | μs | |
| 35 | TBOR | Brown-out Reset Pulse Width | 100 | — | — | μs | VDD ≤ VBOR (D005) |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 18-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**




**Note:** Refer to Figure 18-3 for load conditions.



**TABLE 18-6: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

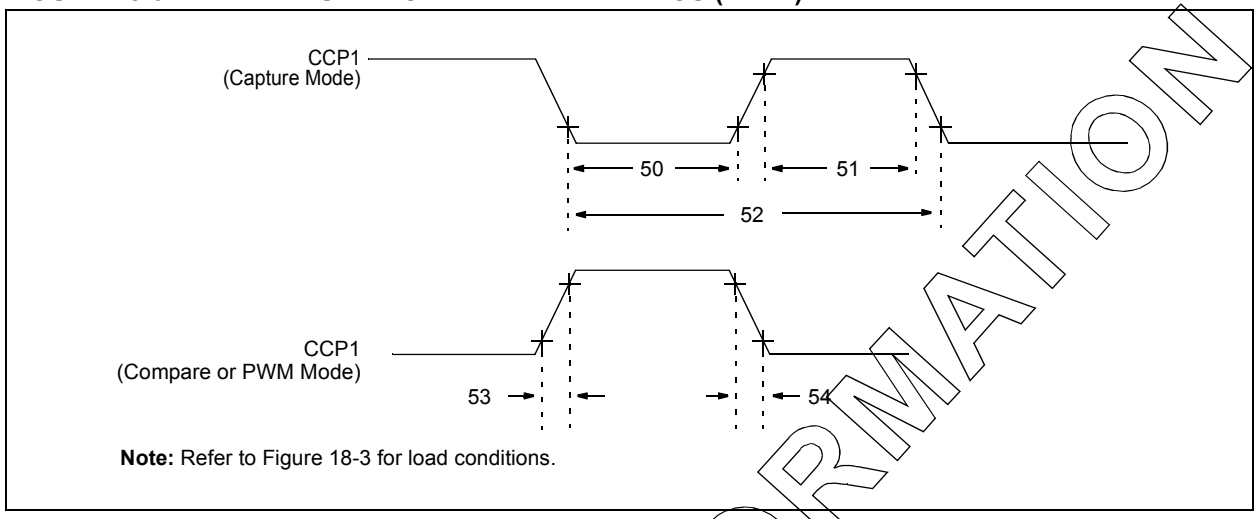
| Param No. | Symbol | Characteristic | | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|---|
| 40* | Tt0H | T0CKI High Pulse Width | | No Prescaler | 0.5 TCY + 20 | — | — | ns | Must also meet parameter 42 |
| | | | | With Prescaler | 10 | — | — | ns | |
| 41* | Tt0L | T0CKI Low Pulse Width | | No Prescaler | 0.5 TCY + 20 | — | — | ns | Must also meet parameter 42 |
| | | | | With Prescaler | 10 | — | — | ns | |
| 42* | Tt0P | T0CKI Period | | No Prescaler | TCY + 40 | — | — | ns | |
| | | | | With Prescaler | Greater of: 20 or TCY + 40 / N | — | — | ns | N = prescale value (2, 4, ..., 256) |
| 45* | Tt1H | T1CKI High Time | Synchronous, Prescaler = 1 | | 0.5 TCY + 20 | — | — | ns | Must also meet parameter 47 |
| | | | Synchronous, Prescaler = 2,4,8 | PIC16**F**87/88 | 15 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 25 | — | — | ns | |
| | | | Asynchronous | PIC16**F**87/88 | 30 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 50 | — | — | ns | |
| 46* | Tt1L | T1CKI Low Time | Synchronous, Prescaler = 1 | | 0.5 TCY + 20 | — | — | ns | Must also meet parameter 47 |
| | | | Synchronous, Prescaler = 2,4,8 | PIC16**F**87/88 | 15 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 25 | — | — | ns | |
| | | | Asynchronous | PIC16**F**87/88 | 30 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 50 | — | — | ns | |
| 47* | Tt1P | T1CKI Input Period | Synchronous | PIC16**F**87/88 | Greater of: 30 or TCY + 40 / N | — | — | ns | N = prescale value (1, 2, 4, 8) |
| | | | | PIC16**LF**87/88 | Greater of: 50 or TCY + 40 / N | | | | N = prescale value (1, 2, 4, 8) |
| | | | Asynchronous | PIC16**F**87/88 | 60 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 100 | — | — | ns | |
| | Ft1 | Timer1 Oscillator Input Frequency Range (Oscillator enabled by setting bit T1OSCEN) | | | DC | — | 32.768 | kHz | |
| 48 | TCKEZtmr1 | Delay from External Clock Edge to Timer Increment | | | 2 TOSC | — | 7 TOSC | — | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 18-9: CAPTURE/COMPARE/PWM TIMINGS (CCP1)**




**Note:** Refer to Figure 18-3 for load conditions.

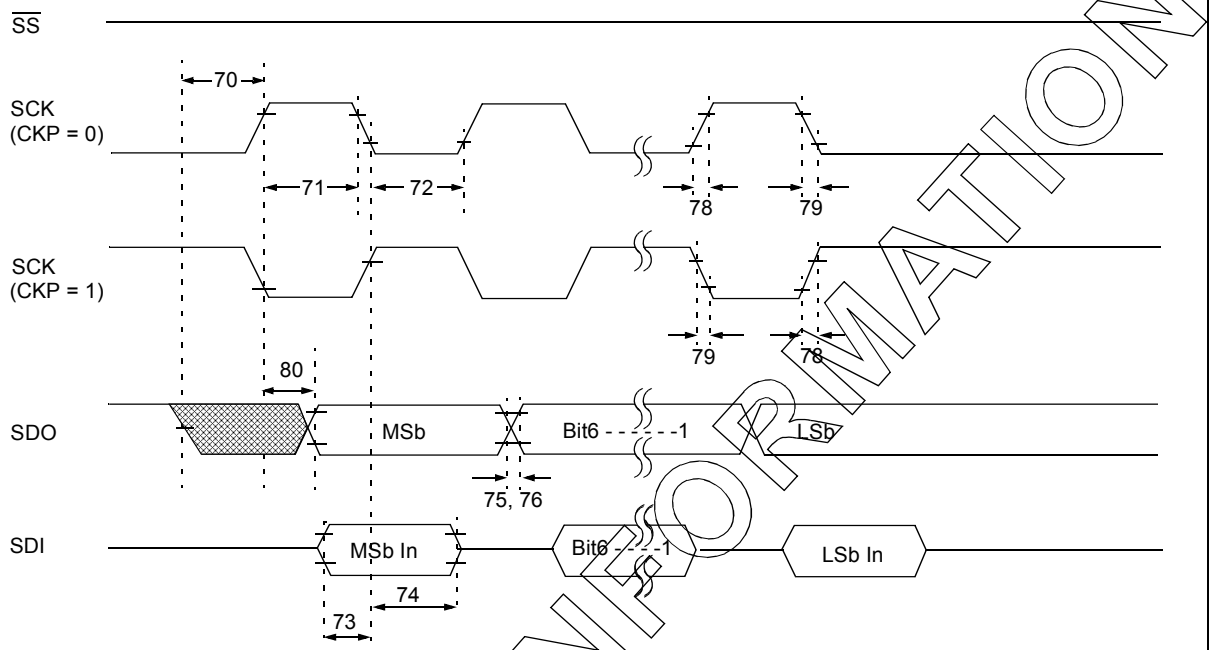
**TABLE 18-7: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP1)**

| Param No. | Symbol | Characteristic | | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|---|
| 50* | TccL | CCP1 Input Low Time | No Prescaler | | 0.5 TCY + 20 | — | — | ns | |
| | | | With Prescaler | PIC16**F**87/88 | 10 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 20 | — | — | ns | |
| 51* | TccH | CCP1 Input High Time | No Prescaler | | 0.5 TCY + 20 | — | — | ns | |
| | | | With Prescaler | PIC16**F**87/88 | 10 | — | — | ns | |
| | | | | PIC16**LF**87/88 | 20 | — | — | ns | |
| 52* | TccP | CCP1 Input Period | | | $\frac{3\ T_{CY} + 40}{N}$ | — | — | ns | N = prescale value (1,4 or 16) |
| 53* | TccR | CCP1 Output Rise Time | | PIC16**F**87/88 | — | 10 | 25 | ns | |
| | | | | PIC16**LF**87/88 | — | 25 | 50 | ns | |
| 54* | TccF | CCP1 Output Fall Time | | PIC16**F**87/88 | — | 10 | 25 | ns | |
| | | | | PIC16**LF**87/88 | — | 25 | 45 | ns | |

* These parameters are characterized but not tested.

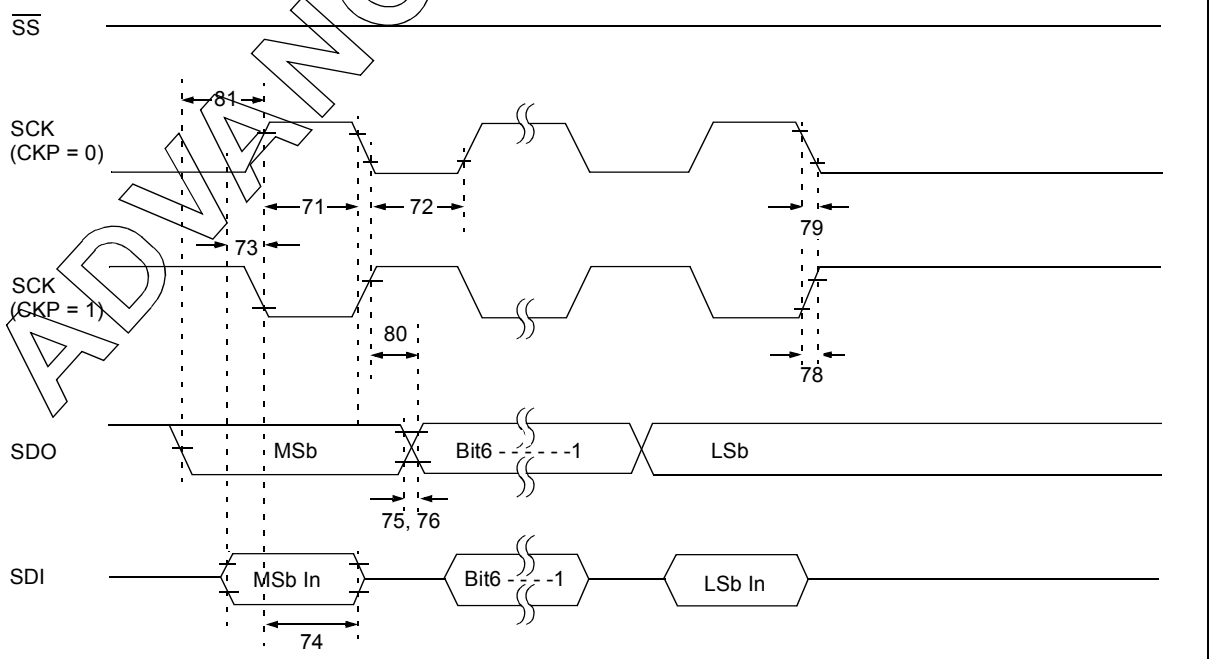
† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 18-10: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)**




**Note:** Refer to Figure 18-3 for load conditions.

**FIGURE 18-11: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)**

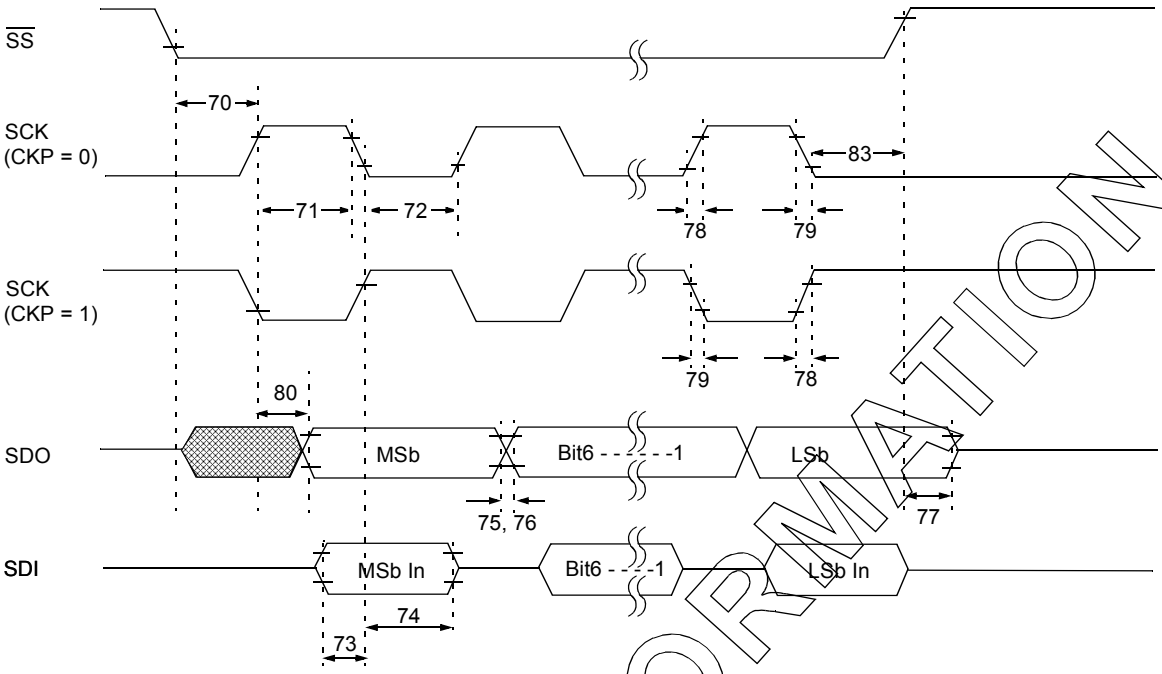



**Note:** Refer to Figure 18-3 for load conditions.



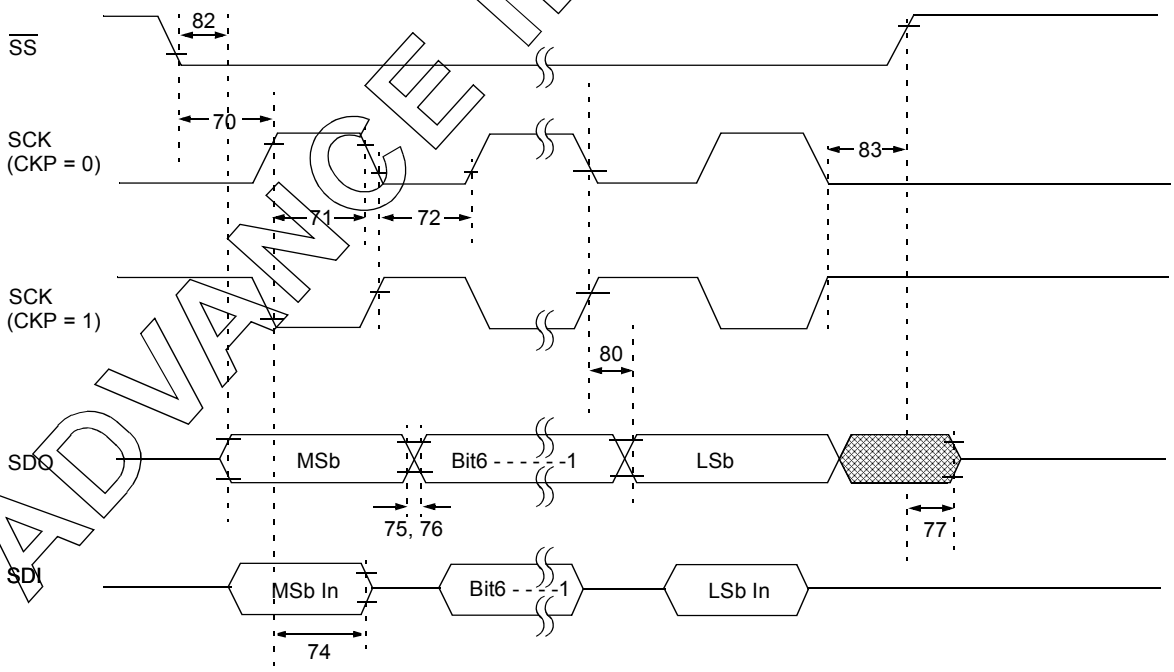
 

**FIGURE 18-12: SPI SLAVE MODE TIMING (CKE = 0)**




**Note:** Refer to Figure 18-3 for load conditions.

**FIGURE 18-13: SPI SLAVE MODE TIMING (CKE = 1)**




**Note:** Refer to Figure 18-3 for load conditions.

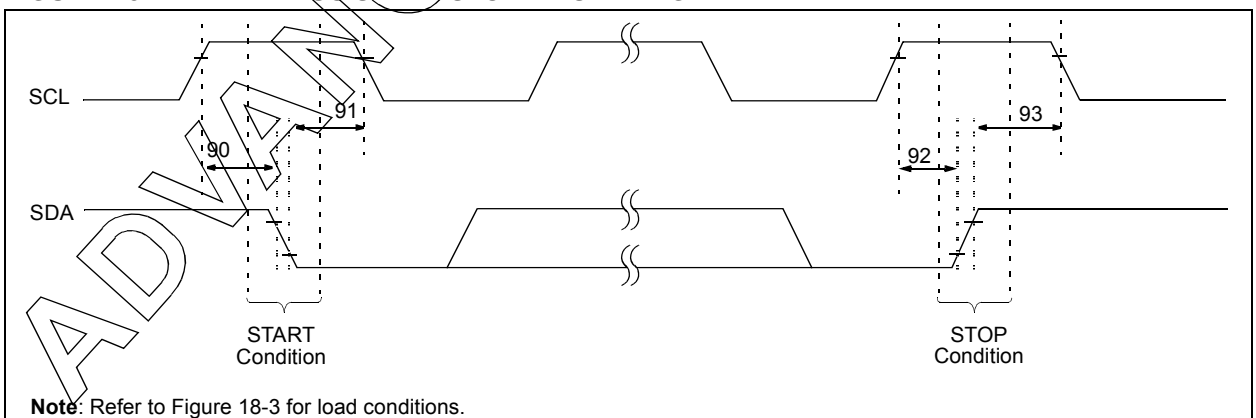
**TABLE 18-8: SPI MODE REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 70* | TssL2scH, TssL2scL | $\overline{SS}\downarrow$ to SCK↓ or SCK↑ input | | TCY | — | — | ns | |
| 71* | TscH | SCK input high time (Slave mode) | | TCY + 20 | — | — | ns | |
| 72* | TscL | SCK input low time (Slave mode) | | TCY + 20 | — | — | ns | |
| 73* | TdiV2scH, TdiV2scL | Setup time of SDI data input to SCK edge | | 100 | — | — | ns | |
| 74* | TscH2diL, TscL2diL | Hold time of SDI data input to SCK edge | | 100 | — | — | ns | |
| 75* | TdoR | SDO data output rise time | PIC16**F**87/88<br>PIC16**LF**87/88 | —<br>— | 10<br>25 | 25<br>50 | ns<br>ns | |
| 76* | TdoF | SDO data output fall time | | — | 10 | 25 | ns | |
| 77* | TssH2doZ | $\overline{SS}\uparrow$ to SDO output hi-impedance | | 10 | — | 50 | ns | |
| 78* | TscR | SCK output rise time (Master mode) | PIC16**F**87/88<br>PIC16**LF**87/88 | —<br>— | 10<br>25 | 25<br>50 | ns<br>ns | |
| 79* | TscF | SCK output fall time (Master mode) | | — | 10 | 25 | ns | |
| 80* | TscH2doV, TscL2doV | SDO data output valid after SCK edge | PIC16**F**87/88<br>PIC16**LF**87/88 | —<br>— | —<br>— | 50<br>145 | ns<br>ns | |
| 81* | TdoV2scH, TdoV2scL | SDO data output setup to SCK edge | | TCY | — | — | ns | |
| 82* | TssL2doV | SDO data output valid after $\overline{SS}\downarrow$ edge | | — | — | 50 | ns | |
| 83* | TscH2ssH, TscL2ssH | $\overline{SS}\uparrow$ after SCK edge | | 1.5 TCY + 40 | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 18-14: $I^2C$ BUS START/STOP BITS TIMING**




**Note**: Refer to Figure 18-3 for load conditions.



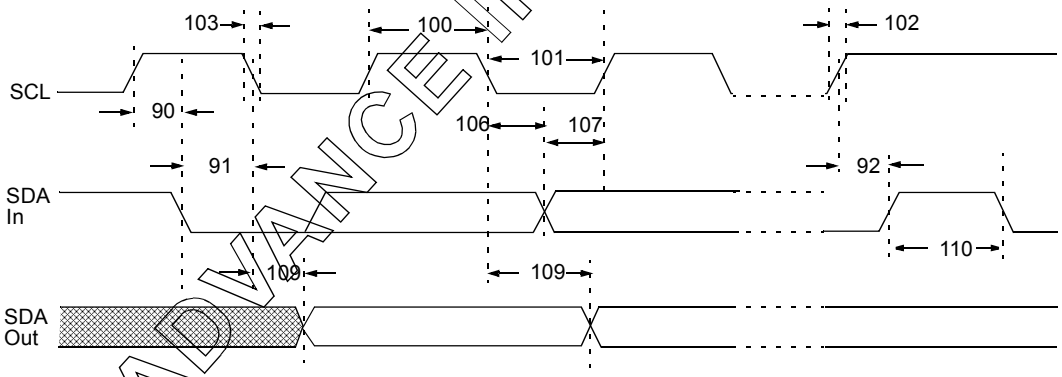
 

**TABLE 18-9: I²C BUS START/STOP BITS REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 90* | TSU:STA | START condition Setup time | 100 kHz mode | 4700 | — | — | ns | Only relevant for Repeated START condition |
| | | | 400 kHz mode | 600 | — | — | | |
| 91* | THD:STA | START condition Hold time | 100 kHz mode | 4000 | — | — | ns | After this period the first clock pulse is generated |
| | | | 400 kHz mode | 600 | — | — | | |
| 92* | TSU:STO | STOP condition Setup time | 100 kHz mode | 4700 | — | — | ns | |
| | | | 400 kHz mode | 600 | — | — | | |
| 93 | THD:STO | STOP condition Hold time | 100 kHz mode | 4000 | — | — | ns | |
| | | | 400 kHz mode | 600 | — | — | | |

* These parameters are characterized but not tested.

**FIGURE 18-15: I²C BUS DATA TIMING**




**Note:** Refer to Figure 18-3 for load conditions.

**TABLE 18-10: I²C BUS DATA REQUIREMENTS**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 100* | THIGH | Clock high time | 100 kHz mode | 4.0 | — | µs | |
| | | | 400 kHz mode | 0.6 | — | µs | |
| | | | SSP Module | 1.5 TCY | — | | |
| 101* | TLOW | Clock low time | 100 kHz mode | 4.7 | — | µs | |
| | | | 400 kHz mode | 1.3 | — | µs | |
| | | | SSP Module | 1.5 TCY | — | | |
| 102* | TR | SDA and SCL rise time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | CB is specified to be from 10 - 400 pF |
| 103* | TF | SDA and SCL fall time | 100 kHz mode | — | 300 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | CB is specified to be from 10 - 400 pF |
| 90* | TSU:STA | START condition setup time | 100 kHz mode | 4.7 | — | µs | Only relevant for Repeated START condition |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 91* | THD:STA | START condition hold time | 100 kHz mode | 4.0 | — | µs | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 106* | THD:DAT | Data input hold time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | 0.9 | µs | |
| 107* | TSU:DAT | Data input setup time | 100 kHz mode | 250 | — | ns | **(Note 2)** |
| | | | 400 kHz mode | 100 | — | ns | |
| 92* | TSU:STO | STOP condition setup time | 100 kHz mode | 4.7 | — | µs | |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 109* | TAA | Output valid from clock | 100 kHz mode | — | 3500 | ns | **(Note 1)** |
| | | | 400 kHz mode | — | — | ns | |
| 110* | TBUF | Bus free time | 100 kHz mode | 4.7 | — | µs | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | µs | |
| | CB | Bus capacitive loading | | — | 400 | pF | |

* These parameters are characterized but not tested.

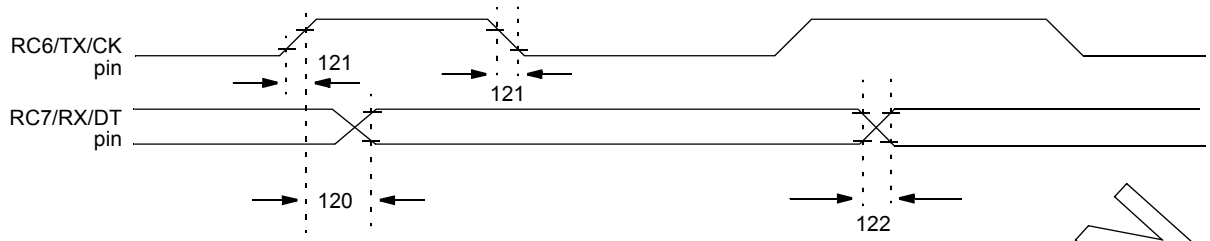
**Note 1:** As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of START or STOP conditions.

**2:** A Fast mode (400 kHz) I²C bus device can be used in a Standard mode (100 kHz) I²C bus system, but the requirement TSU:DAT ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCL line is released.



**FIGURE 18-16: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**



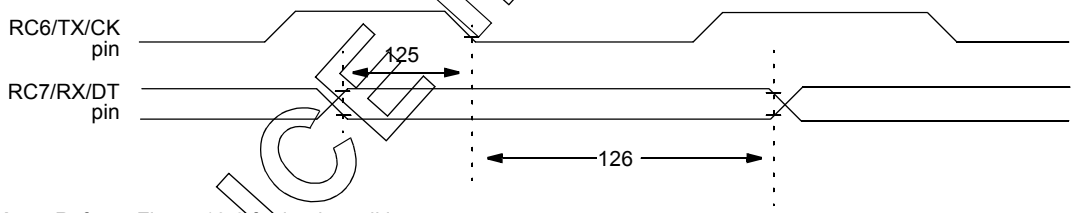

**Note:** Refer to Figure 18-3 for load conditions.

**TABLE 18-11: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 120 | TckH2dtV | SYNC XMIT (MASTER & SLAVE) Clock high to data out valid | PIC16**F**87/88 | | | 80 | ns | |
| | | | PIC16**LF**87/88 | | | 100 | ns | |
| 121 | Tckrf | Clock out rise time and fall time (Master mode) | PIC16**F**87/88 | — | — | 45 | ns | |
| | | | PIC16**LF**87/88 | — | — | 50 | ns | |
| 122 | Tdtrf | Data out rise time and fall time | PIC16**F**87/88 | — | — | 45 | ns | |
| | | | PIC16**LF**87/88 | — | — | 50 | ns | |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 18-17: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**




**Note:** Refer to Figure 18-3 for load conditions.

**TABLE 18-12: USART SYNCHRONOUS RECEIVE REQUIREMENTS**

| Parameter No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 125 | TdtV2ckL | SYNC RCV (MASTER & SLAVE) Data setup before CK↓ (DT setup time) | 15 | — | — | ns | |
| 126 | TckL2dtl | Data hold after CK↓ (DT hold time) | 15 | — | — | ns | |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**TABLE 18-13: A/D CONVERTER CHARACTERISTICS: PIC16F87/88 (INDUSTRIAL, EXTENDED) PIC16LF87/88 (INDUSTRIAL)**

| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| A01 | NR | Resolution | — | — | 10 bits | bit | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A03 | EIL | Integral linearity error | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A04 | EDL | Differential linearity error | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A06 | EOFF | Offset error | — | — | < ± 2 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A07 | EGN | Gain error | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A10 | — | Monotonicity(3) | — | guaranteed | — | — | VSS ≤ VAIN ≤ VREF |
| A20 | VREF | Reference Voltage | 2.5<br>2.2 | —<br>— | VDD + 0.3<br>VDD + 0.3 | V<br>V | -40°C to +85°C<br>0°C to +85°C |
| A21 | VREF+ | Reference voltage high | AVDD – 2.5V | | AVDD + 0.3V | V | |
| A22 | VREF- | Reference voltage low | AVSS – 0.3V | | VREF+ – 2.0V | V | |
| A25 | VAIN | Analog input voltage | VSS – 0.3V | — | VREF + 0.3V | V | |
| A30 | ZAIN | Recommended impedance of analog voltage source | — | — | 2.5 | kΩ | See **(Note 4)** |
| A50 | IREF | VREF input current(2) | —<br>— | —<br>— | 5<br>500 | μA<br>μA | During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD, see Section 12.1.<br>During A/D Conversion cycle. |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

**2:** VREF current is from RA3 pin or VDD pin, whichever is selected as reference input.

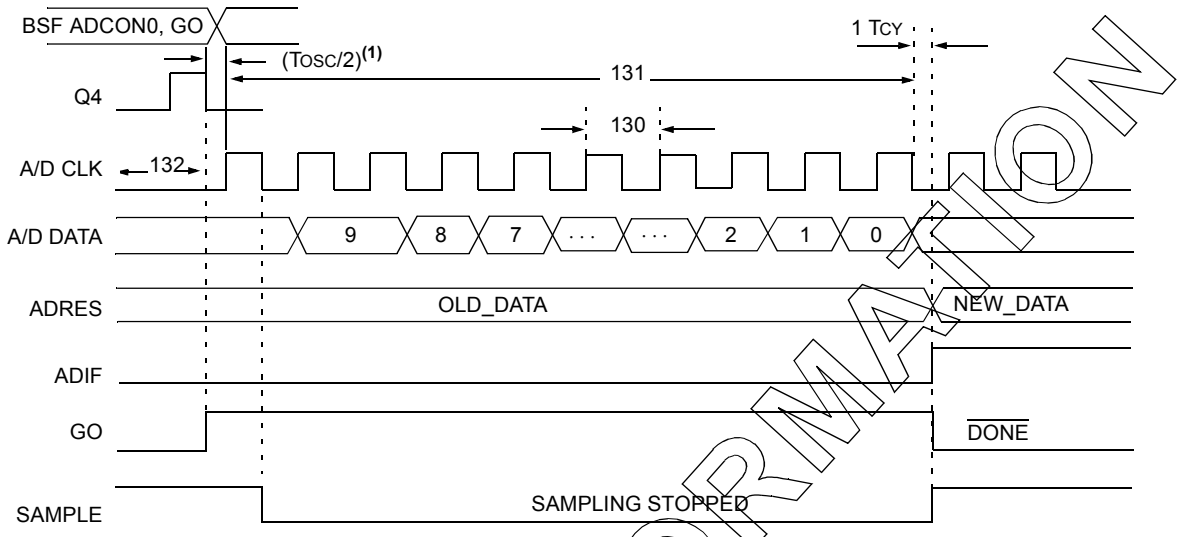
**3:** The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

**4:** The maximum allowed impedance for analog voltage source is 10 kΩ. This requires higher acquisition times.



**FIGURE 18-18: A/D CONVERSION TIMING**




**Note 1:** If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

**TABLE 18-14: A/D CONVERSION REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 130 | TAD | A/D clock period | PIC16F87/88 | 1.6 | — | — | µs | TOSC based, VREF ≥ 3.0V |
| | | | PIC16LF87/88 | 3.0 | — | — | µs | TOSC based, VREF ≥ 2.0V |
| | | | PIC16F87/88 | 2.0 | 4.0 | 6.0 | µs | A/D RC mode |
| | | | PIC16LF87/88 | 3.0 | 6.0 | 9.0 | µs | A/D RC mode |
| 131 | TCNV | Conversion time (not including S/H time) **(Note 1)** | | | — | 12 | TAD | |
| 132 | TACQ | Acquisition time | | **(Note 2)**<br>10* | 40<br>— | —<br>— | µs<br>µs | The minimum time is the amplifier settling time. This may be used if the "new" input voltage has not changed by more than 1 LSb (i.e., 20.0 mV @ 5.12V) from the last sampled voltage (as stated on CHOLD). |
| 134 | TGO | Q4 to A/D clock start | | — | TOSC/2 | — | — | If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed. |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

§ This specification ensured by design.

**Note 1:** ADRES register may be read on the following TCY cycle.

**2:** See Section 12.1 for minimum conditions.

**NOTES:**

## 19.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

No Graphs and Tables are available at this time.

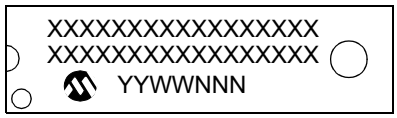
**NOTES:**

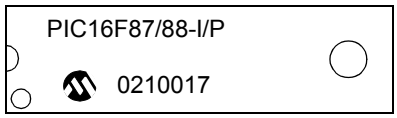
## 20.0 PACKAGING INFORMATION

### 20.1 Package Marking Information

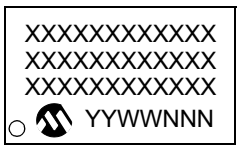
18-Lead PDIP



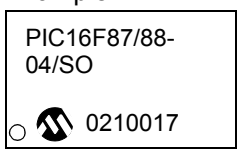

Example



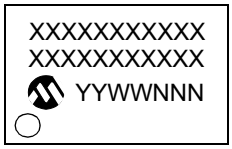

18-Lead SOIC




Example




20-Lead SSOP




Example


28-Lead QFN

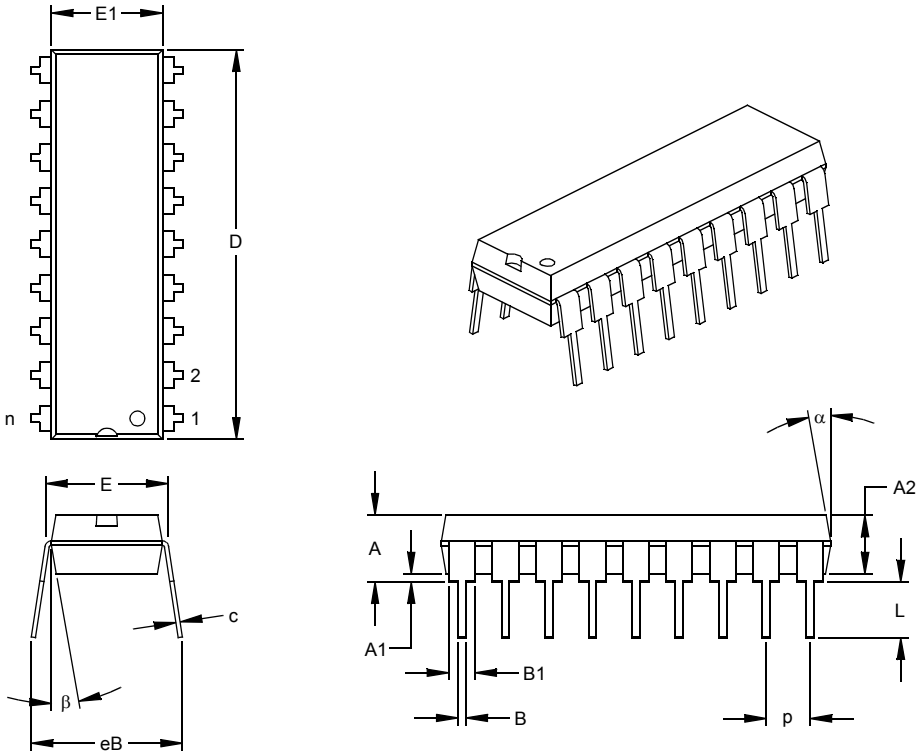

Example


| | | |
|---|---|---|
| **Legend:** | XX...X | Customer specific information* |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week ‘01’) |
| | NNN | Alphanumeric traceability code |

**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

* Standard PICmicro device marking consists of Microchip part number, year code, week code, and traceability code. For PICmicro device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

## 18-Lead Plastic Dual In-line (P) – 300 mil (PDIP)

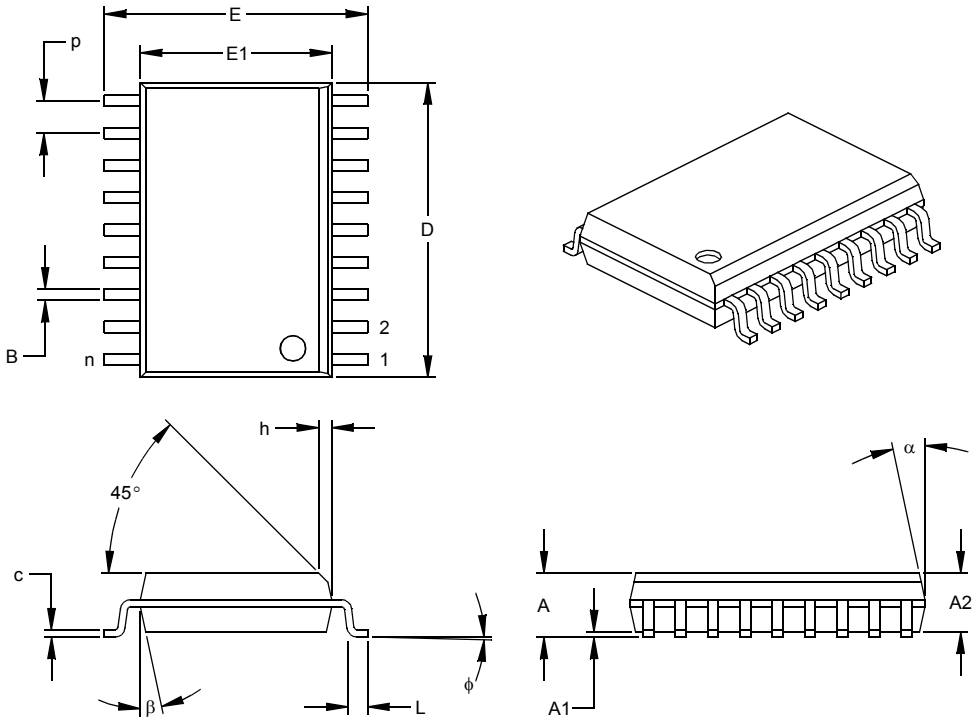



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .155 | .170 | 3.56 | 3.94 | 4.32 |
| Molded Package Thickness | A2 | .115 | .130 | .145 | 2.92 | 3.30 | 3.68 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Molded Package Width | E1 | .240 | .250 | .260 | 6.10 | 6.35 | 6.60 |
| Overall Length | D | .890 | .898 | .905 | 22.61 | 22.80 | 22.99 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .045 | .058 | .070 | 1.14 | 1.46 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing § | eB | .310 | .370 | .430 | 7.87 | 9.40 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-001
Drawing No. C04-007

## 18-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)

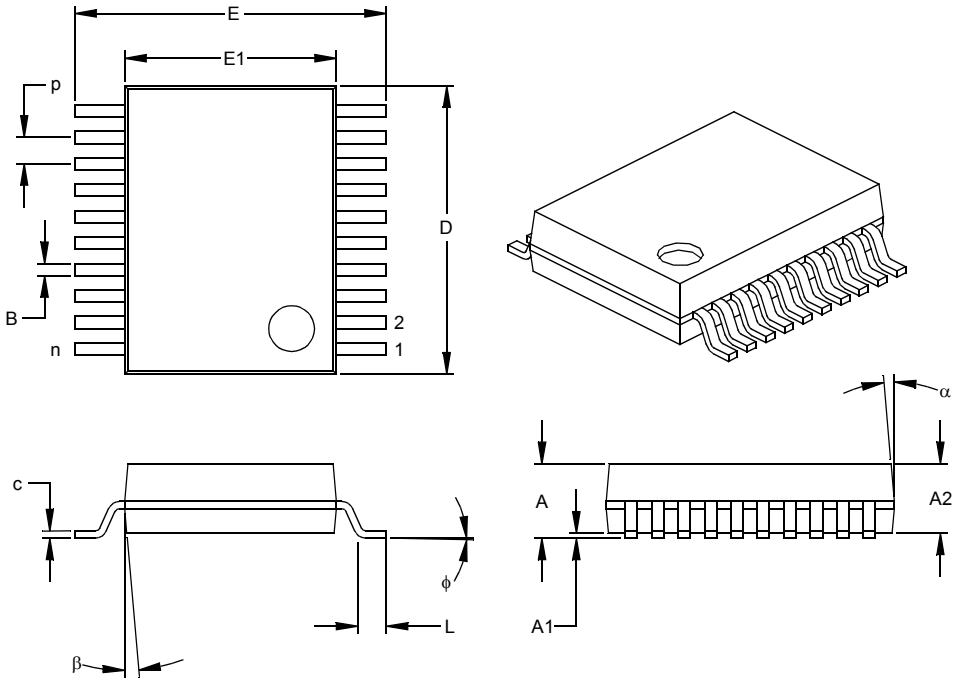



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .291 | .295 | .299 | 7.39 | 7.49 | 7.59 |
| Overall Length | D | .446 | .454 | .462 | 11.33 | 11.53 | 11.73 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .012 | 0.23 | 0.27 | 0.30 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-013
Drawing No. C04-051

### 20-Lead Plastic Shrink Small Outline (SS) – 209 mil, 5.30 mm (SSOP)

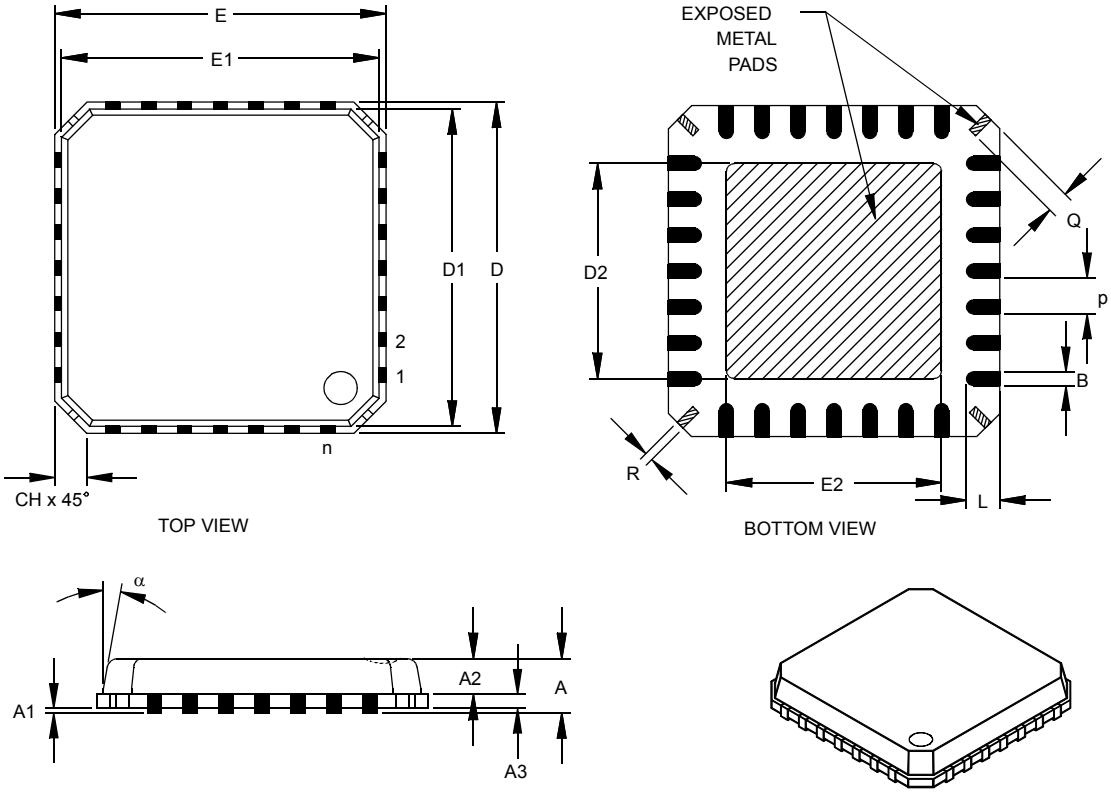



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 20 | | | 20 | |
| Pitch | p | | .026 | | | 0.65 | |
| Overall Height | A | .068 | .073 | .078 | 1.73 | 1.85 | 1.98 |
| Molded Package Thickness | A2 | .064 | .068 | .072 | 1.63 | 1.73 | 1.83 |
| Standoff § | A1 | .002 | .006 | .010 | 0.05 | 0.15 | 0.25 |
| Overall Width | E | .299 | .309 | .322 | 7.59 | 7.85 | 8.18 |
| Molded Package Width | E1 | .201 | .207 | .212 | 5.11 | 5.25 | 5.38 |
| Overall Length | D | .278 | .284 | .289 | 7.06 | 7.20 | 7.34 |
| Foot Length | L | .022 | .030 | .037 | 0.56 | 0.75 | 0.94 |
| Lead Thickness | c | .004 | .007 | .010 | 0.10 | 0.18 | 0.25 |
| Foot Angle | φ | 0 | 4 | 8 | 0.00 | 101.60 | 203.20 |
| Lead Width | B | .010 | .013 | .015 | 0.25 | 0.32 | 0.38 |
| Mold Draft Angle Top | α | 0 | 5 | 10 | 0 | 5 | 10 |
| Mold Draft Angle Bottom | β | 0 | 5 | 10 | 0 | 5 | 10 |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MO-150
Drawing No. C04-072

## 28-Lead Plastic Quad Flat No Lead Package (ML) 6x6 mm Body (QFN)



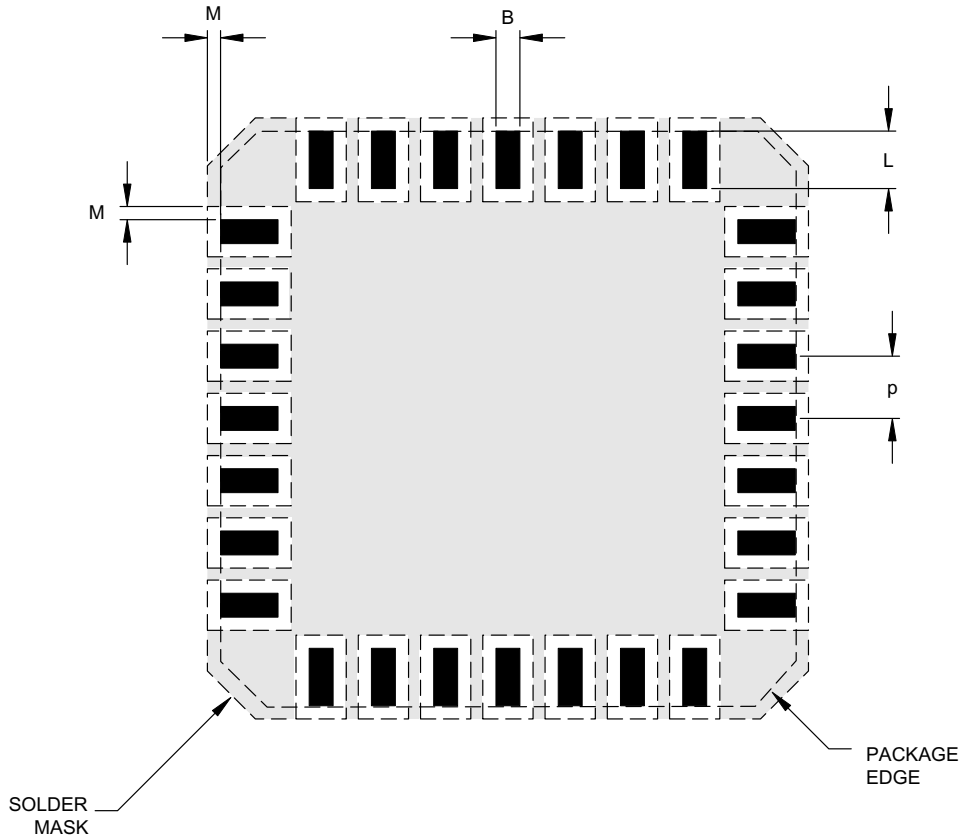

| Units | | INCHES | | | MILLIMETERS* | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .026 BSC | | | 0.65 BSC | |
| Overall Height | A | | .033 | .039 | | 0.85 | 1.00 |
| Molded Package Thickness | A2 | | .026 | .031 | | 0.65 | 0.80 |
| Standoff | A1 | .000 | .0004 | .002 | 0.00 | 0.01 | 0.05 |
| Base Thickness | A3 | | .008 REF. | | | 0.20 REF. | |
| Overall Width | E | | .236 BSC | | | 6.00 BSC | |
| Molded Package Width | E1 | | .226 BSC | | | 5.75 BSC | |
| Exposed Pad Width | E2 | .140 | .146 | .152 | 3.55 | 3.70 | 3.85 |
| Overall Length | D | | .236 BSC | | | 6.00 BSC | |
| Molded Package Length | D1 | | .226 BSC | | | 5.75 BSC | |
| Exposed Pad Length | D2 | .140 | .146 | .152 | 3.55 | 3.70 | 3.85 |
| Lead Width | B | .009 | .011 | .014 | 0.23 | 0.28 | 0.35 |
| Lead Length | L | .020 | .024 | .030 | 0.50 | 0.60 | 0.75 |
| Tie Bar Width | R | .005 | .007 | .010 | 0.13 | 0.17 | 0.23 |
| Tie Bar Length | Q | .012 | .016 | .026 | 0.30 | 0.40 | 0.65 |
| Chamfer | CH | .009 | .017 | .024 | 0.24 | 0.42 | 0.60 |
| Mold Draft Angle Top | α | | | 12° | | | 12° |

*Controlling Parameter

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC equivalent: M0-220

Drawing No. C04-114

### 28-Lead Plastic Quad Flat No Lead Package (ML) 6x6 mm Body (QFN)
### Land Pattern and Solder Mask




| Units | | INCHES | | | MILLIMETERS* | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Pitch | p | | .026 BSC | | | 0.65 BSC | |
| Pad Width | B | .009 | .011 | .014 | 0.23 | 0.28 | 0.35 |
| Pad Length | L | .020 | .024 | .030 | 0.50 | 0.60 | 0.75 |
| Pad to Solder Mask | M | .005 | | .006 | 0.13 | | 0.15 |

*Controlling Parameter

Drawing No. C04-2114

## APPENDIX A: REVISION HISTORY

| Version | Date | Revision Description |
|---|---|---|
| A | November 2002 | This is a new data sheet. |

## APPENDIX B: DEVICE DIFFERENCES

The differences between the devices in this data sheet are listed in Table B-1.

**TABLE B-1: DIFFERENCES BETWEEN THE PIC16F87 AND PIC16F88**

| Features | PIC16F87 | PIC16F88 |
|---|---|---|
| Analog-to-Digital Converter | N/A | 10-bit, 7-channel |

**NOTES:**

# INDEX

## D



## E



## F



## I



## K



## L



## M

## O



## P



## R



## S

## T



## U



## V



## W

**NOTES:**

## ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® or Microsoft® Internet Explorer. Files are also available for FTP download from our FTP site.

### Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

**www.microchip.com**

The file transfer site is available by using an FTP service to connect to:

**ftp://ftp.microchip.com**

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

## SYSTEMS INFORMATION AND UPGRADE HOT LINE

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits.The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

092002

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC16F87/88 Literature Number: DS30487A

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?

## PIC16F87/88 PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

| PART NO. | X | /XX | XXX |
|---|---|---|---|
| Device | Temperature Range | Package | Pattern |

| | |
|---|---|
| Device | PIC16F87: Standard VDD range<br>PIC16F87T: (Tape and Reel)<br>PIC16LF87: Extended VDD range |
| Temperature Range | - = 0°C to +70°C<br>I = -40°C to +85°C |
| Package | P = PDIP<br>SO = SOIC<br>SS = SSOP<br>ML = QFN |
| Pattern | QTP, SQTP, ROM Code (factory specified) or Special Requirements. Blank for OTP and Windowed devices. |

**Examples:**

a) PIC16F87-I/P = Industrial temp., PDIP package, Extended VDD limits.

b) PIC16F87-I/SO = Industrial temp., SOIC package, normal VDD limits.

**Note 1:** F = CMOS FLASH
LF = Low Power CMOS FLASH

**2:** T = in tape and reel - SOIC, SSOP packages only.

### Sales and Support

**Data Sheets**

Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 792-7277
3. The Microchip Worldwide Site (www.microchip.com)

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

**New Customer Notification System**

Register on our web site (www.microchip.com/cn) to receive the most current information on our products.

# WORLDWIDE SALES AND SERVICE

## AMERICAS

**Corporate Office**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

**Rocky Mountain**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-4338

**Atlanta**
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034 Fax: 770-640-0307

**Boston**
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

**Chicago**
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

**Dallas**
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

**Detroit**
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

**Kokomo**
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

**Los Angeles**
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

**San Jose**
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

**Toronto**
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

## ASIA/PACIFIC

**Australia**
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

**China - Beijing**
Microchip Technology Consulting (Shanghai) Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

**China - Chengdu**
Microchip Technology Consulting (Shanghai) Co., Ltd., Chengdu Liaison Office
Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200 Fax: 86-28-86766599

**China - Fuzhou**
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

**China - Shanghai**
Microchip Technology Consulting (Shanghai) Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

**China - Shenzhen**
Microchip Technology Consulting (Shanghai) Co., Ltd., Shenzhen Liaison Office
Rm. 15-16, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-82350361 Fax: 86-755-82366086

**China - Hong Kong SAR**
Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

**India**
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

**Japan**
Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

**Korea**
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

**Singapore**
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

**Taiwan**
Microchip Technology (Barbados) Inc.,
Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

## EUROPE

**Austria**
Microchip Technology Austria GmbH
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

**Denmark**
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

**France**
Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

**Germany**
Microchip Technology GmbH
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

**Italy**
Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

**United Kingdom**
Microchip Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

11/15/02