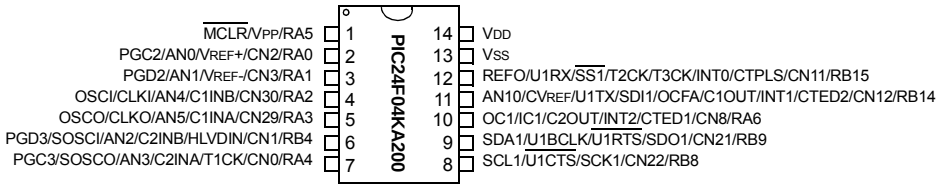
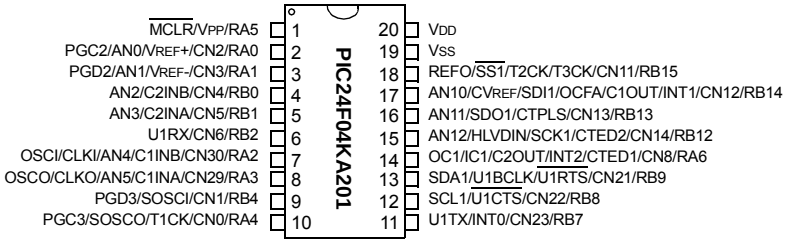
## Pin Diagrams

### 14-Pin PDIP, TSSOP(1)



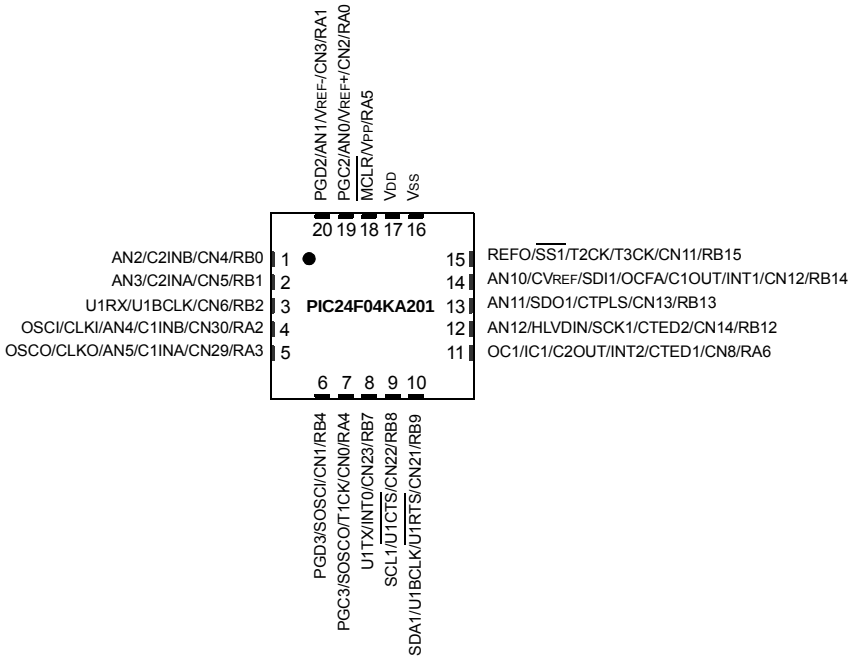

### 20-Pin PDIP, SSOP, SOIC(1)




**Note 1:** All device pins have a maximum voltage of 3.6V and are not 5V tolerant.

## Pin Diagrams (Continued)

**20-Pin QFN(1,2)**




**Note 1:** Connecting the bottom pad to Vss is recommended.
**2:** All device pins have a maximum voltage of 3.6V and are not 5V tolerant.

## 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@microchip.com**. 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., DS30000000A is version A of document DS30000000).

### 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)

When contacting a sales office, 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** to receive the most current information on all of our products.

**NOTES:**

# 1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC24F04KA200
- PIC24F04KA201

The PIC24F04KA201 family introduces a new line of extreme low-power Microchip devices: a 16-bit microcontroller family with a broad peripheral feature set and enhanced computational performance. It also offers a new migration option for those high-performance applications, which may be outgrowing their 8-bit platforms, but do not require the numerical processing power of a digital signal processor.

## 1.1 Core Features

### 1.1.1 16-BIT ARCHITECTURE

Central to all PIC24F devices is the 16-bit modified Harvard architecture, first introduced with Microchip's dsPIC® digital signal controllers. The PIC24F CPU core offers a wide range of enhancements, such as:

- 16-bit data and 24-bit address paths with the ability to move information between data and memory spaces
- Linear addressing of up to 12 Mbytes (program space) and 64 Kbytes (data)
- A 16-element working register array with built-in software stack support
- A 17 x 17 hardware multiplier with support for integer math
- Hardware support for 32-bit by 16-bit division
- An instruction set that supports multiple addressing modes and is optimized for high-level languages, such as C
- Operational performance up to 16 MIPS

### 1.1.2 POWER-SAVING TECHNOLOGY

The PIC24F04KA200 and PIC24F04KA201 devices incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **On-the-Fly Clock Switching:** The device clock can be changed under software control to the Timer1 source or the internal, low-power RC oscillator during operation, allowing users to incorporate power-saving ideas into their software designs.
- **Doze Mode Operation:** When timing-sensitive applications, such as serial communications, require the uninterrupted operation of peripherals, the CPU clock speed can be selectively reduced, allowing incremental power savings without missing a beat.
- **Instruction-Based Power-Saving Modes:** There are three instruction-based power-saving modes:
  - Idle Mode: The core is shut down while leaving the peripherals active.
  - Sleep Mode: The core and peripherals that require the system clock are shut down, leaving the peripherals that use their own clock, or the clock from other devices, active.
  - Deep Sleep Mode: The core, peripherals (except DSWDT), Flash and SRAM are shut down.

### 1.1.3 OSCILLATOR OPTIONS AND FEATURES

The PIC24F04KA201 family offers five different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes using crystals or ceramic resonators.
- Two External Clock modes offering the option of a divide-by-2 clock output.
- Two fast internal oscillators (FRCs): One with a nominal 8 MHz output and the other with nominal 500 kHz output. These outputs can also be divided under software control to provide clock speed as low as 31 kHz or 2 kHz.
- A Phase Locked Loop (PLL) frequency multiplier, available to the External Oscillator modes and the 8 MHz FRC oscillator, which allows clock speeds of up to 32 MHz.
- A separate internal RC oscillator (LPRC) with a fixed 31 kHz output, which provides a low-power option for timing-insensitive applications.

The internal oscillator block also provides a stable reference source for the Fail-Safe Clock Monitor (FSCM). This option constantly monitors the main clock source against a reference signal provided by the internal oscillator and enables the controller to switch to the internal oscillator, allowing for continued low-speed operation or a safe application shutdown.

### 1.1.4 EASY MIGRATION

Regardless of the memory size, all the devices share the same rich set of peripherals, allowing for a smooth migration path as applications grow and evolve.

The consistent pinout scheme used throughout the entire family also helps in migrating to the next larger device. This is true when moving between devices with the same pin count, or even jumping from 14-pin to 20-pin devices. The PIC24F16KA102 family is directly compatible for migration to larger program and data memory.

The PIC24F family is pin compatible with devices in the dsPIC33 family, and shares some compatibility with the pinout schema for PIC18 and dsPIC30. This extends the ability of applications to grow from the relatively simple, to the powerful and complex.

## 1.2 Other Special Features

- **Communications:** The PIC24F04KA201 family incorporates a range of serial communication peripherals to handle a range of application requirements. There is an $I^2C$™ module that supports both the Master and Slave modes of operation. It also comprises a UART with built-in IrDA® encoders/decoders and an SPI module.
- **10-Bit A/D Converter:** This module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period, and faster sampling speed. The 16-deep result buffer can be used either in Sleep to reduce power, or in Active mode to improve throughput.
- **Charge Time Measurement Unit (CTMU) Interface:** The PIC24F04KA201 family includes the new CTMU interface module, which can be used for capacitive touch sensing using Microchip's mTouch™ technology, proximity sensing and also for precision time measurement and pulse generation.

## 1.3 Details on Individual Family Members

Devices in the PIC24F04KA201 family are available in 14-pin and 20-pin packages. The general block diagram for all devices is displayed in Figure 1-1.

The devices are different from each other in two ways:

1. Number of ADC channels (9 channels on 20-pin parts, 7 channels on 14-pin parts).
2. Available I/O pins and ports (12 pins on two ports for 14-pin devices and 18 pins on two ports for 20-pin devices).

All other features for devices in this family are identical; these are summarized in Table 1-1.

A list of the pin features available on the PIC24F04KA201 family devices, sorted by function, is provided in Table 1-2.

| **Note:** | Table 1-1 provides the pin location of individual peripheral features and not how they are multiplexed on the same pin. This information is provided in the pinout diagrams on pages 2 and 3 of the data sheet. Multiplexed features are sorted by the priority given to a feature, with the highest priority peripheral being listed first. |
|---|---|

## 1.4 Differences from PIC24F16KA102 Family

The PIC24F04KA201 family architecture is very similar to that of the PIC24F16KA102 family. The PIC24F04KA201 family is a subset of the PIC24F16KA102 devices.

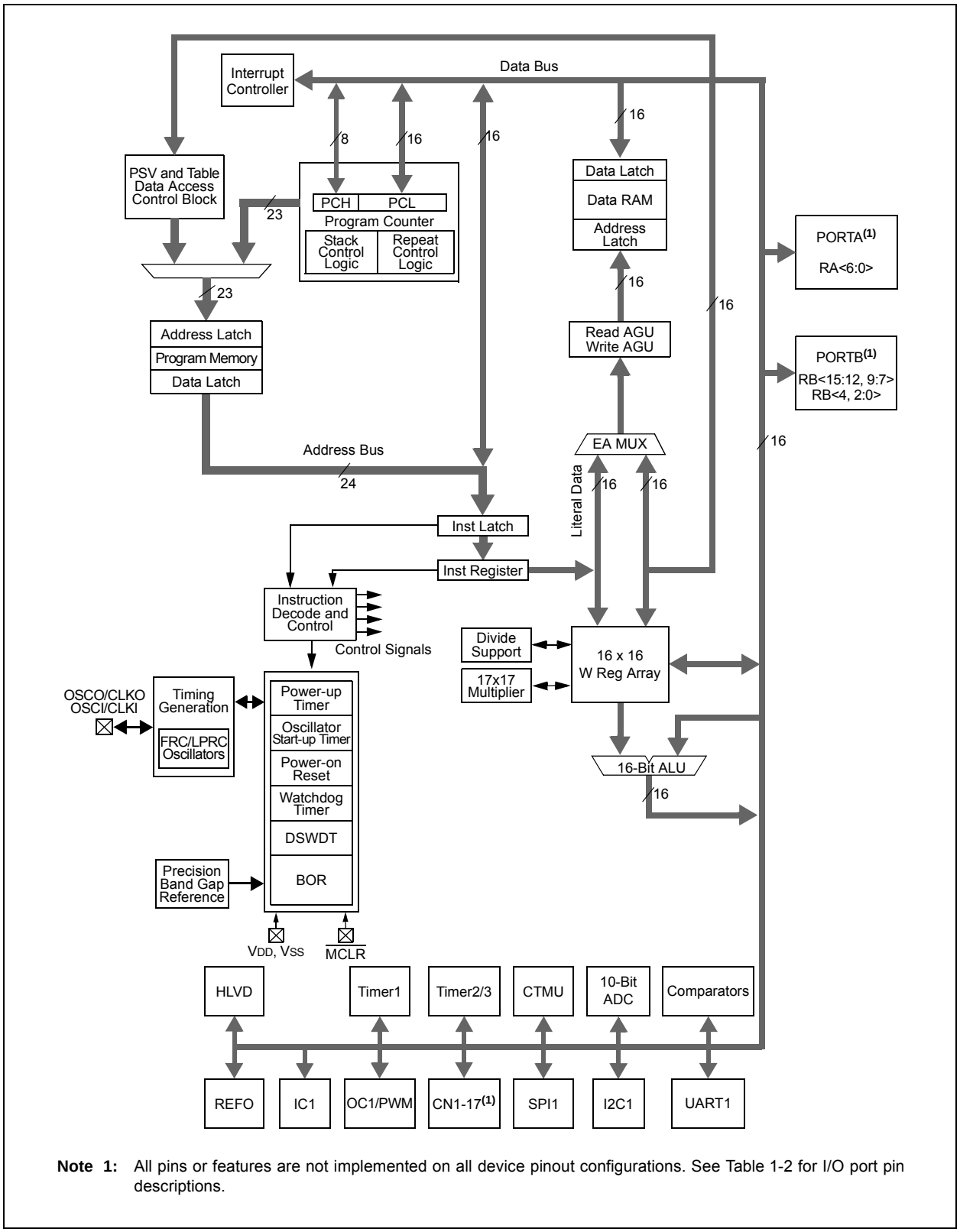
The PIC24F16KA102 family has the following additional features:

- Larger Program Memory
- Larger Data Memory
- CRC Module
- Debugging Capabilities through ICSP™
- Additional I/O on 20-Pin Devices (up to 24 I/O pins)
- Data EEPROM memory
- Boot Segment and General Segments for Program Code (with available code protection)
- One Additional UART (2 total)

**TABLE 1-1: DEVICE FEATURES FOR THE PIC24F04KA201 FAMILY**

| Features | PIC24F04KA200 | PIC24F04KA201 |
|---|---|---|
| Operating Frequency | DC – 32 MHz | |
| Program Memory (bytes) | 4K | |
| Program Memory (instructions) | 1408 | |
| Data Memory (bytes) | 512 | |
| Interrupt Sources (soft vectors/NMI traps) | 25 (21/4) | |
| I/O Ports | PORTA<6:0><br>PORTB<15:14, 9:8, 4> | PORTA<6:0><br>PORTB<15:12, 9:7, 4, 2:0> |
| Total I/O Pins | 12 | 18 |
| Timers: Total Number (16-bit)<br>32-Bit (from paired 16-bit timers) | 3<br>1 | |
| Input Capture Channels | 1 | |
| Output Compare/PWM Channels | 1 | |
| Input Change Notification Interrupt | 11 | 17 |
| Serial Communications: UART<br>SPI (3-wire/4-wire)<br>$I^2C$™ | 1<br>1<br>1 | |
| 10-Bit Analog-to-Digital Module (input channels) | 7 | 9 |
| Analog Comparators | 2 | |
| Resets (and delays) | POR, BOR, RESET Instruction, $\overline{MCLR}$, WDT, Illegal Opcode, REPEAT Instruction, Hardware Traps, Configuration Word Mismatch (PWRT, OST, PLL Lock) | |
| Instruction Set | 76 Base Instructions, Multiple Addressing Mode Variations | |
| Packages | 14-Pin PDIP/TSSOP | 20-Pin PDIP/SSOP/SOIC/QFN |

**FIGURE 1-1: PIC24F04KA201 FAMILY GENERAL BLOCK DIAGRAM**




**Note 1:** All pins or features are not implemented on all device pinout configurations. See Table 1-2 for I/O port pin descriptions.

**TABLE 1-2: PIC24F04KA201 FAMILY PINOUT DESCRIPTIONS**

| Function | Pin Number | | | I/O | Input Buffer | Description |
|---|---|---|---|---|---|---|
| | 14-Pin PDIP/TSSOP/ SOIC | 20-Pin PDIP/SSOP/ SOIC | 20-Pin QFN | | | |
| AN0 | 2 | 2 | 19 | I | ANA | A/D Analog Inputs |
| AN1 | 3 | 3 | 20 | I | ANA | |
| AN2 | 6 | 4 | 1 | I | ANA | |
| AN3 | 7 | 5 | 2 | I | ANA | |
| AN4 | 4 | 7 | 4 | I | ANA | |
| AN5 | 5 | 8 | 5 | I | ANA | |
| AN10 | 11 | 17 | 14 | I | ANA | |
| AN11 | — | 16 | 13 | I | ANA | |
| AN12 | — | 15 | 12 | I | ANA | |
| U1BCLK | 9 | 13 | 10 | O | — | UART1 IrDA® Baud Clock |
| C1INA | 5 | 8 | 5 | I | ANA | Comparator 1 Input A (Positive input) |
| C1INB | 4 | 7 | 4 | I | ANA | Comparator 1 Input B (Negative input option 1) |
| C1OUT | 11 | 17 | 14 | O | — | Comparator 1 Output |
| C2INA | 7 | 5 | 2 | I | ANA | Comparator 2 Input A (Positive input) |
| C2INB | 6 | 4 | 1 | I | ANA | Comparator 2 Input B (Negative input option 1) |
| C2OUT | 10 | 14 | 11 | O | — | Comparator 2 Output |
| CLKI | 4 | 7 | 4 | I | ANA | Main Clock Input Connection |
| CLKO | 5 | 8 | 5 | O | — | System Clock Output |
| CN0 | 7 | 10 | 7 | I | ST | Interrupt-on-Change Inputs |
| CN1 | 6 | 9 | 6 | I | ST | |
| CN2 | 2 | 2 | 19 | I | ST | |
| CN3 | 3 | 3 | 20 | I | ST | |
| CN4 | — | 4 | 1 | I | ST | |
| CN5 | — | 5 | 2 | I | ST | |
| CN6 | — | 6 | 3 | I | ST | |
| CN8 | 10 | 14 | 11 | I | ST | |
| CN11 | 12 | 18 | 15 | I | ST | |
| CN12 | 11 | 17 | 14 | I | ST | |
| CN13 | — | 16 | 13 | I | ST | |
| CN14 | — | 15 | 12 | I | ST | |
| CN21 | 9 | 13 | 10 | I | ST | |
| CN22 | 8 | 12 | 9 | I | ST | |
| CN23 | — | 11 | 8 | I | ST | |
| CN29 | 5 | 8 | 5 | I | ST | |
| CN30 | 4 | 7 | 4 | I | ST | |
| CVREF | 11 | 17 | 14 | O | ANA | Comparator Voltage Reference Output |
| CTED1 | 10 | 14 | 11 | I | ST | CTMU Trigger Edge Input 1 |
| CTED2 | 11 | 15 | 12 | I | ST | CTMU Trigger Edge Input 2 |
| CTPLS | 12 | 16 | 13 | O | — | CTMU Pulse Output |
| IC1 | 10 | 14 | 11 | I | ST | Input Capture 1 Input |
| INT0 | 12 | 11 | 8 | I | ST | External Interrupt Inputs |
| INT1 | 11 | 17 | 14 | I | ST | |
| INT2 | 10 | 14 | 11 | I | ST | |

**Legend:** ST = Schmitt Trigger input buffer, ANA = Analog level input/output, $I^2C$™ = $I^2C$/SMBus input buffer

**TABLE 1-2: PIC24F04KA201 FAMILY PINOUT DESCRIPTIONS (CONTINUED)**

| Function | Pin Number | | | I/O | Input Buffer | Description |
|---|---|---|---|---|---|---|
| | 14-Pin PDIP/TSSOP/ SOIC | 20-Pin PDIP/SSOP/ SOIC | 20-Pin QFN | | | |
| HLVDIN | 6 | 15 | 12 | I | ANA | HLVD Voltage Input |
| $\overline{MCLR}$ | 1 | 1 | 18 | I | ST | Master Clear (device Reset) Input |
| OC1 | 10 | 14 | 11 | O | — | Output Compare/PWM Outputs |
| OCFA | 11 | 17 | 14 | I | — | Output Compare Fault A |
| OSCI | 4 | 7 | 4 | I | ANA | Main Oscillator Input Connection |
| OSCO | 5 | 8 | 5 | O | ANA | Main Oscillator Output Connection |
| PGC2 | 2 | 2 | 19 | I/O | ST | In-Circuit Debugger and ICSP Programming Clock |
| PGD2 | 3 | 3 | 20 | I/O | ST | In-Circuit Debugger and ICSP Programming Data |
| PGC3 | 7 | 10 | 7 | I/O | ST | In-Circuit Debugger and ICSP Programming Clock |
| PGD3 | 6 | 9 | 6 | I/O | ST | In-Circuit Debugger and ICSP Programming Data |
| RA0 | 2 | 2 | 19 | I/O | ST | PORTA Digital I/O |
| RA1 | 3 | 3 | 20 | I/O | ST | |
| RA2 | 4 | 7 | 4 | I/O | ST | |
| RA3 | 5 | 8 | 5 | I/O | ST | |
| RA4 | 7 | 10 | 7 | I/O | ST | |
| RA5 | 1 | 1 | 18 | I/O | ST | |
| RA6 | 10 | 14 | 11 | I/O | ST | |
| RB0 | — | 4 | 1 | I/O | ST | PORTB Digital I/O |
| RB1 | — | 5 | 2 | I/O | ST | |
| RB2 | — | 6 | 3 | I/O | ST | |
| RB4 | 6 | 9 | 6 | I/O | ST | |
| RB8 | 8 | 12 | 9 | I/O | ST | |
| RB9 | 9 | 13 | 10 | I/O | ST | |
| RB12 | — | 15 | 12 | I/O | ST | |
| RB13 | — | 16 | 13 | I/O | ST | |
| RB14 | 11 | 17 | 14 | I/O | ST | |
| RB15 | 12 | 18 | 15 | I/O | ST | |
| REFO | 12 | 18 | 15 | O | — | Reference Clock Output |
| SCK1 | 8 | 15 | 12 | I/O | ST | SPI1 Serial Clock Input/Output |
| SCL1 | 8 | 12 | 9 | I/O | $I^2C$ | I2C1 Synchronous Serial Clock Input/Output |
| SDA1 | 9 | 13 | 10 | I/O | $I^2C$ | I2C1 Data Input/Output |
| SDI1 | 11 | 17 | 14 | I | ST | SPI1 Serial Data Input |
| SDO1 | 9 | 16 | 13 | O | — | SPI1 Serial Data Output |
| SOSCI | 6 | 9 | 6 | I | ANA | Secondary Oscillator Input |
| SOSCO | 7 | 10 | 7 | O | ANA | Secondary Oscillator Output |
| $\overline{SS1}$ | 12 | 18 | 15 | I/O | ST | Slave Select Input/Frame Select Output (SPI1) |
| T1CK | 7 | 10 | 7 | I | ST | Timer1 Clock |
| T2CK | 12 | 18 | 15 | I | ST | Timer2 Clock |
| T3CK | 12 | 18 | 15 | I | ST | Timer3 Clock |
| $\overline{U1CTS}$ | 8 | 12 | 9 | I | ST | UART1 Clear to Send Input |
| $\overline{U1RTS}$ | 9 | 13 | 10 | O | — | UART1 Request to Send Output |
| U1RX | 12 | 6 | 3 | I | ST | UART1 Receive |
| U1TX | 11 | 11 | 8 | O | — | UART1 Transmit Output |

**Legend:** ST = Schmitt Trigger input buffer, ANA = Analog level input/output, $I^2C$™ = $I^2C$/SMBus input buffer

**TABLE 1-2: PIC24F04KA201 FAMILY PINOUT DESCRIPTIONS (CONTINUED)**

| Function | Pin Number | | | I/O | Input Buffer | Description |
|---|---|---|---|---|---|---|
| | 14-Pin PDIP/TSSOP/ SOIC | 20-Pin PDIP/SSOP/ SOIC | 20-Pin QFN | | | |
| VDD | 14 | 20 | 17 | P | — | Positive Supply for Peripheral Digital Logic and I/O Pins |
| VPP | 1 | 1 | 18 | P | — | Programming Mode Entry Voltage |
| VREF- | 3 | 3 | 20 | I | ANA | A/D and Comparator Reference Voltage (low) Input |
| VREF+ | 2 | 2 | 19 | I | ANA | A/D and Comparator Reference Voltage (high) Input |
| VSS | 13 | 19 | 16 | P | — | Ground Reference for Logic and I/O Pin |

**Legend:** ST = Schmitt Trigger input buffer, ANA = Analog level input/output, $I^2C$™ = $I^2C$/SMBus input buffer

**NOTES:**

## 2.0 GUIDELINES FOR GETTING STARTED WITH 16-BIT MICROCONTROLLERS

### 2.1 Basic Connection Requirements

Getting started with the PIC24F04KA201 family of 16-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins
  (see **Section 2.2 "Power Supply Pins"**)
- All AVDD and AVSS pins, regardless of whether or not the analog device features are used
  (see **Section 2.2 "Power Supply Pins"**)
- MCLR pin
  (see **Section 2.3 "Master Clear (MCLR) Pin"**)
- ENVREG/DISVREG and VCAP/VDDCORE pins (PIC24F devices only)
  (see **Section 2.4 "Voltage Regulator Pins (ENVREG/DISVREG and VCAP/VDDCORE)"**)

These pins must also be connected if they are being used in the end application:

- PGECx/PGEDx pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes
  (see **Section 2.5 "ICSP Pins"**)
- OSCI and OSCO pins when an external oscillator source is used
  (see **Section 2.6 "External Oscillator Pins"**)

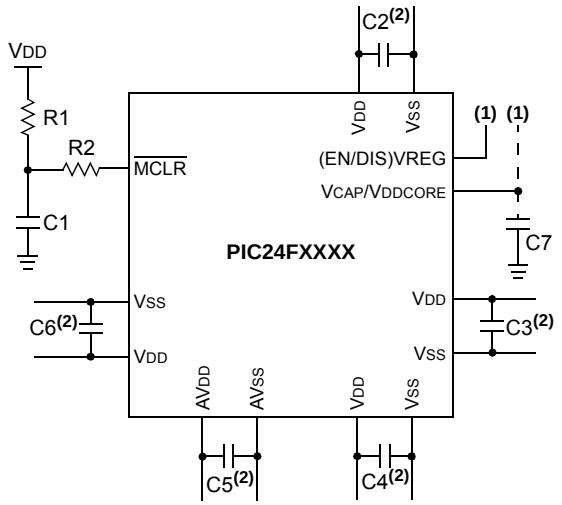
Additionally, the following pins may be required:

- VREF+/VREF- pins used when external voltage reference for analog modules is implemented

| **Note:** | The AVDD and AVSS pins must always be connected, regardless of whether any of the analog modules are being used. |
|---|---|

The minimum mandatory connections are shown in Figure 2-1.

**FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS**




**Key (all values are recommendations):**

C1 through C6: 0.1 µF, 20V ceramic

C7: 10 µF, 6.3V or greater, tantalum or ceramic

R1: 10 kΩ

R2: 100Ω to 470Ω

**Note 1:** See **Section 2.4 "Voltage Regulator Pins (ENVREG/DISVREG and VCAP/VDDCORE)"** for an explanation of ENVREG/DISVREG pin connections.

**2:** The example shown is for a PIC24F device with five VDD/VSS and AVDD/AVSS pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.

## 2.2 Power Supply Pins

### 2.2.1 DECOUPLING CAPACITORS

The use of decoupling capacitors on every pair of power supply pins, such as VDD, VSS, AVDD and AVSS, is required.

Consider the following criteria when using decoupling capacitors:

- **Value and type of capacitor:** A 0.1 μF (100 nF), 10-20V capacitor is recommended. The capacitor should be a low-ESR device with a resonance frequency in the range of 200 MHz and higher. Ceramic capacitors are recommended.
- **Placement on the printed circuit board:** The decoupling capacitors should be placed as close to the pins as possible. It is recommended to place the capacitors on the same side of the board as the device. If space is constricted, the capacitor can be placed on another layer on the PCB using a via; however, ensure that the trace length from the pin to the capacitor is no greater than 0.25 inch (6 mm).
- **Handling high-frequency noise:** If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01 μF to 0.001 μF. Place this second capacitor next to each primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible (e.g., 0.1 μF in parallel with 0.001 μF).
- **Maximizing performance:** On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

### 2.2.2 TANK CAPACITORS

On boards with power traces running longer than six inches in length, it is suggested to use a tank capacitor for integrated circuits including microcontrollers to supply a local power source. The value of the tank capacitor should be determined based on the trace resistance that connects the power supply source to the device and the maximum current drawn by the device in the application. In other words, select the tank capacitor so that it meets the acceptable voltage sag at the device. Typical values range from 4.7 μF to 47 μF.

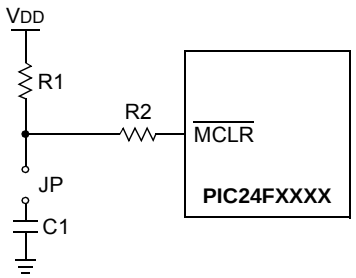
## 2.3 Master Clear ($\overline{\text{MCLR}}$) Pin

The $\overline{\text{MCLR}}$ pin provides two specific device functions: device Reset, and device programming and debugging. If programming and debugging are not required in the end application, a direct connection to VDD may be all that is required. The addition of other components, to help increase the application's resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 2-1. Other circuit designs may be implemented depending on the application's requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the $\overline{\text{MCLR}}$ pin. Consequently, specific voltage levels (VIH and VIL) and fast signal transitions must not be adversely affected. Therefore, specific values of R1 and C1 will need to be adjusted based on the application and PCB requirements. For example, it is recommended that the capacitor, C1, be isolated from the $\overline{\text{MCLR}}$ pin during programming and debugging operations by using a jumper (Figure 2-2). The jumper is replaced for normal run-time operations.

Any components associated with the $\overline{\text{MCLR}}$ pin should be placed within 0.25 inch (6 mm) of the pin.

**FIGURE 2-2: EXAMPLE OF $\overline{\text{MCLR}}$ PIN CONNECTIONS**




**Note 1:** R1 ≤ 10 kΩ is recommended. A suggested starting value is 10 kΩ. Ensure that the $\overline{\text{MCLR}}$ pin VIH and VIL specifications are met.

**2:** R2 ≤ 470Ω will limit any current flowing into $\overline{\text{MCLR}}$ from the external capacitor, C, in the event of $\overline{\text{MCLR}}$ pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS). Ensure that the $\overline{\text{MCLR}}$ pin VIH and VIL specifications are met.

## 2.4 Voltage Regulator Pins (ENVREG/DISVREG and VCAP/VDDCORE)

**Note:** This section applies only to PIC24F devices with an on-chip voltage regulator.

The on-chip voltage regulator enable/disable pin (ENVREG or DISVREG, depending on the device family) must always be connected directly to either a supply voltage or to ground. The particular connection is determined by whether or not the regulator is to be used:

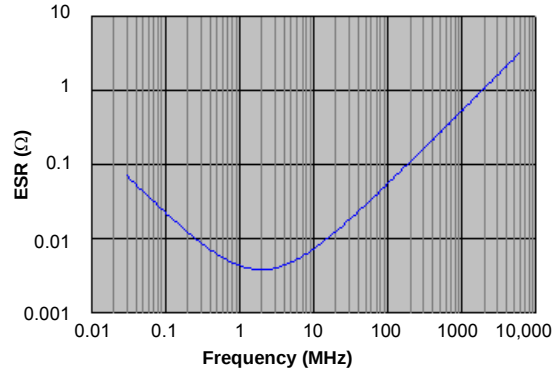
- For ENVREG, tie to VDD to enable the regulator or to ground to disable the regulator
- For DISVREG, tie to ground to enable the regulator or to VDD to disable the regulator

When the regulator is enabled, a low-ESR (<5Ω) capacitor is required on the VCAP/VDDCORE pin to stabilize the voltage regulator output voltage. The VCAP/VDDCORE pin must not be connected to VDD and must use a capacitor of 10 μF connected to ground. The type can be ceramic or tantalum. A suitable example is the Murata GRM21BF50J106ZE01 (10 μF, 6.3V) or equivalent. Designers may use Figure 2-3 to evaluate ESR equivalence of candidate devices.

The placement of this capacitor should be close to VCAP/VDDCORE. It is recommended that the trace length not exceed 0.25 inch (6 mm). Refer to **Section 26.0 "Electrical Characteristics"** for additional information.

When the regulator is disabled, the VCAP/VDDCORE pin must be tied to a voltage supply at the VDDCORE level. Refer to **Section 26.0 "Electrical Characteristics"** for information on VDD and VDDCORE.

**FIGURE 2-3: FREQUENCY vs. ESR PERFORMANCE FOR SUGGESTED VCAP**




**Note:** Data for Murata GRM21BF50J106ZE01 shown. Measurements at 25°C, 0V DC bias.

## 2.5 ICSP Pins

The PGECx and PGEDx pins are used for In-Circuit Serial Programming (ICSP) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

Pull-up resistors, series diodes and capacitors on the PGECx and PGEDx pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits and pin input voltage high (VIH) and input low (VIL) requirements.

For device emulation, ensure that the "Communication Channel Select" (i.e., PGECx/PGEDx pins) programmed into the device matches the physical connections for the ICSP to the MPLAB® ICD 2, MPLAB ICD 3 or REAL ICE™ emulator.

For more information on the ICD 2, ICD 3 and REAL ICE emulator connection requirements, refer to the following documents that are available on the Microchip web site.

- *"MPLAB® ICD 2 In-Circuit Debugger User's Guide"* (DS51331)
- *"Using MPLAB® ICD 2"* (poster) (DS51265)
- *"MPLAB® ICD 2 Design Advisory"* (DS51566)
- *"Using MPLAB® ICD 3"* (poster) (DS51765)
- *"MPLAB® ICD 3 Design Advisory"* (DS51764)
- *"MPLAB® REAL ICE™ In-Circuit Emulator User's Guide"* (DS51616)
- *"Using MPLAB® REAL ICE™ In-Circuit Emulator"* (poster) (DS51749)

## 2.6 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 8.0 "Oscillator Configuration"** for details).

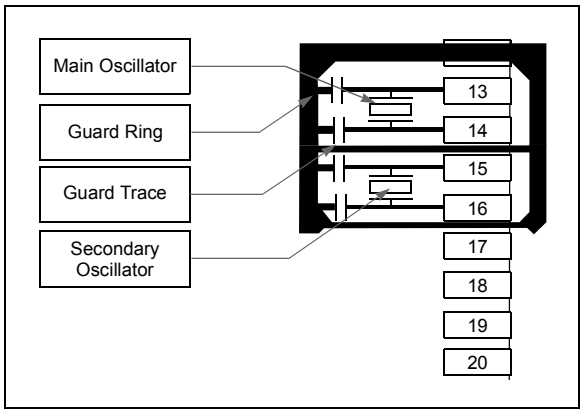
The oscillator circuit should be placed on the same side of the board as the device. Place the oscillator circuit close to the respective oscillator pins with no more than 0.5 inch (12 mm) between the circuit components and the pins. The load capacitors should be placed next to the oscillator itself, on the same side of the board.

Use a grounded copper pour around the oscillator circuit to isolate it from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed. A suggested layout is shown in Figure 2-4.

For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC™ and PICmicro® Devices"*
- *AN849, "Basic PICmicro® Oscillator Design"*
- *AN943, "Practical PICmicro® Oscillator Analysis and Design"*
- *AN949, "Making Your Oscillator Work"*

**FIGURE 2-4: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT**




## 2.7 Configuration of Analog and Digital Pins During ICSP Operations

If the MPLAB ICD 2, ICD 3 or REAL ICE emulator is selected as a debugger, it automatically initializes all of the A/D input pins (ANx) as "digital" pins by setting all bits in the AD1PCFGL register.

The bits in this register that correspond to the A/D pins that are initialized by the MPLAB ICD 2, ICD 3 or REAL ICE emulator must not be cleared by the user application firmware; otherwise, communication errors will result between the debugger and the device.

If your application needs to use certain A/D pins as analog input pins during the debug session, the user application must clear the corresponding bits in the AD1PCFGL register during initialization of the ADC module.

When the MPLAB ICD 2, ICD 3 or REAL ICE emulator is used as a programmer, the user application firmware must correctly configure the AD1PCFGL register. Automatic initialization of this register is only done during debugger operation. Failure to correctly configure the register(s) will result in all A/D pins being recognized as analog input pins, resulting in the port value being read as a logic '0', which may affect user application functionality.

## 2.8 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to VSS on unused pins and drive the output to logic low.

## 3.0 CPU

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the CPU, refer to the *"PIC24F Family Reference Manual"*, **Section 2. "CPU"** (DS39703). |

The PIC24F CPU has a 16-bit (data) modified Harvard architecture with an enhanced instruction set and a 24-bit instruction word with a variable length opcode field. The Program Counter (PC) is 23 bits wide and addresses up to 4M instructions of user program memory space. A single-cycle instruction prefetch mechanism is used to help maintain throughput and provides predictable execution. All instructions execute in a single cycle, with the exception of instructions that change the program flow, the double-word move (MOV.D) instruction and the table instructions. Overhead-free program loop constructs are supported using the REPEAT instructions, which are interruptible at any point.

PIC24F devices have sixteen, 16-bit working registers in the programmer's model. Each of the working registers can act as a data, address or address offset register. The 16$^{th}$ working register (W15) operates as a Software Stack Pointer (SSP) for interrupts and calls.

The upper 32 Kbytes of the data space memory map can optionally be mapped into program space at any 16K word boundary of program memory defined by the 8-bit Program Space Visibility Page Address (PSVPAG) register. The program to data space mapping feature lets any instruction access program space as if it were data space.

The Instruction Set Architecture (ISA) has been significantly enhanced beyond that of the PIC18, but maintains an acceptable level of backward compatibility. All PIC18 instructions and addressing modes are supported, either directly, or through simple macros. Many of the ISA enhancements have been driven by compiler efficiency needs.

The core supports Inherent (no operand), Relative, Literal, Memory Direct and three groups of addressing modes. All modes support Register Direct and various Register Indirect modes. Each group offers up to seven addressing modes. Instructions are associated with predefined addressing modes depending upon their functional requirements.

For most instructions, the core is capable of executing a data (or program data) memory read, a working register (data) read, a data memory write and a program (instruction) memory read per instruction cycle. As a result, three parameter instructions can be supported, allowing trinary operations (that is, A + B = C) to be executed in a single cycle.

A high-speed, 17-bit by 17-bit multiplier has been included to significantly enhance the core arithmetic capability and throughput. The multiplier supports Signed, Unsigned and Mixed mode, 16-bit by 16-bit or 8-bit by 8-bit integer multiplication. All multiply instructions execute in a single cycle.

The 16-bit ALU has been enhanced with integer divide assist hardware that supports an iterative non-restoring divide algorithm. It operates in conjunction with the REPEAT instruction looping mechanism and a selection of iterative divide instructions to support 32-bit (or 16-bit), divided by 16-bit integer signed and unsigned division. All divide operations require 19 cycles to complete but are interruptible at any cycle boundary.

The PIC24F has a vectored exception scheme with up to eight sources of non-maskable traps and up to 118 interrupt sources. Each interrupt source can be assigned to one of seven priority levels.

A block diagram of the CPU is illustrated in Figure 3-1.

### 3.1 Programmer's Model

Figure 3-2 displays the programmer's model for the PIC24F. All registers in the programmer's model are memory mapped and can be manipulated directly by instructions.

Table 3-1 provides a description of each register. All registers associated with the programmer's model are memory mapped.

**FIGURE 3-1: PIC24F CPU CORE BLOCK DIAGRAM**

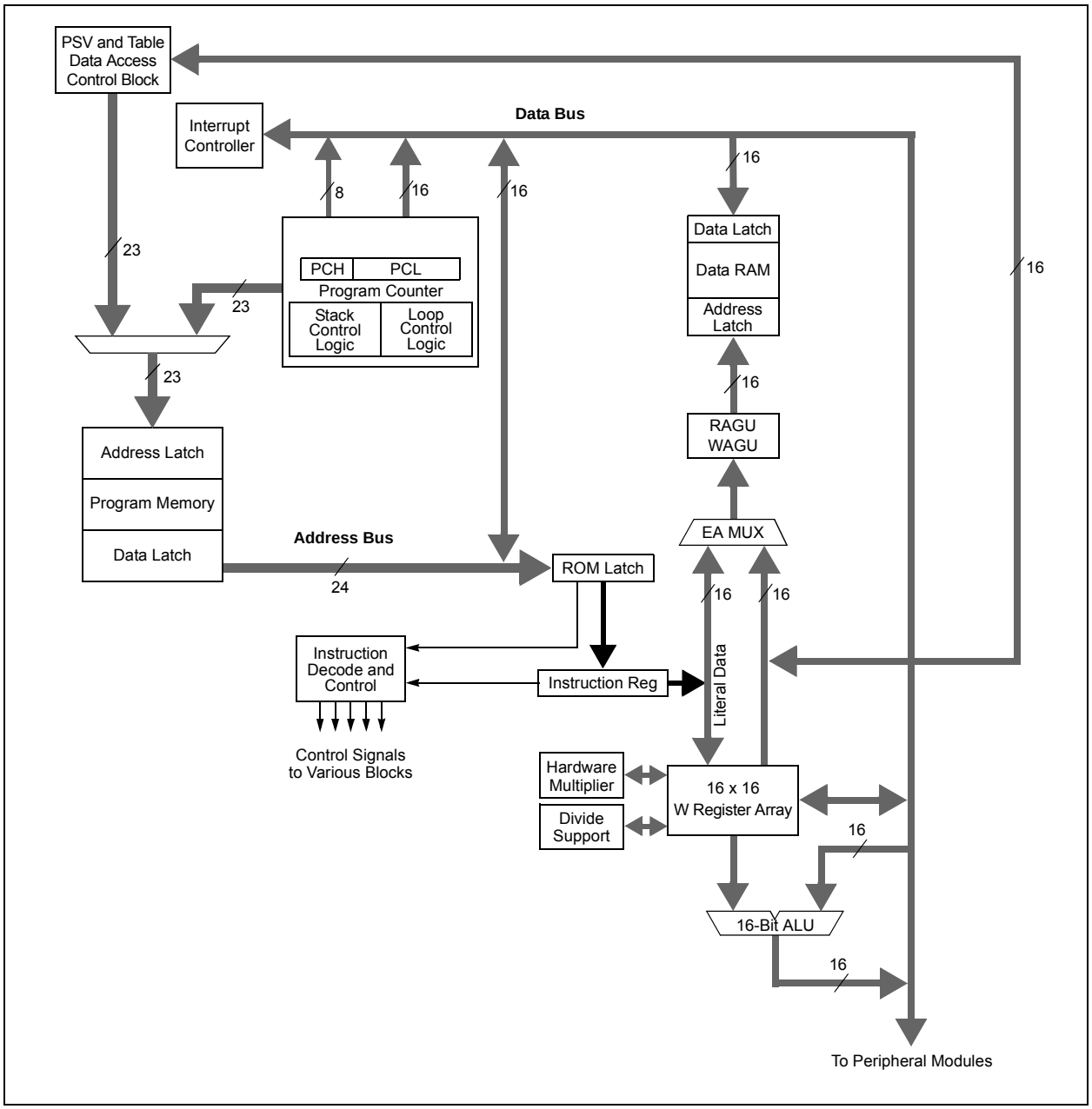
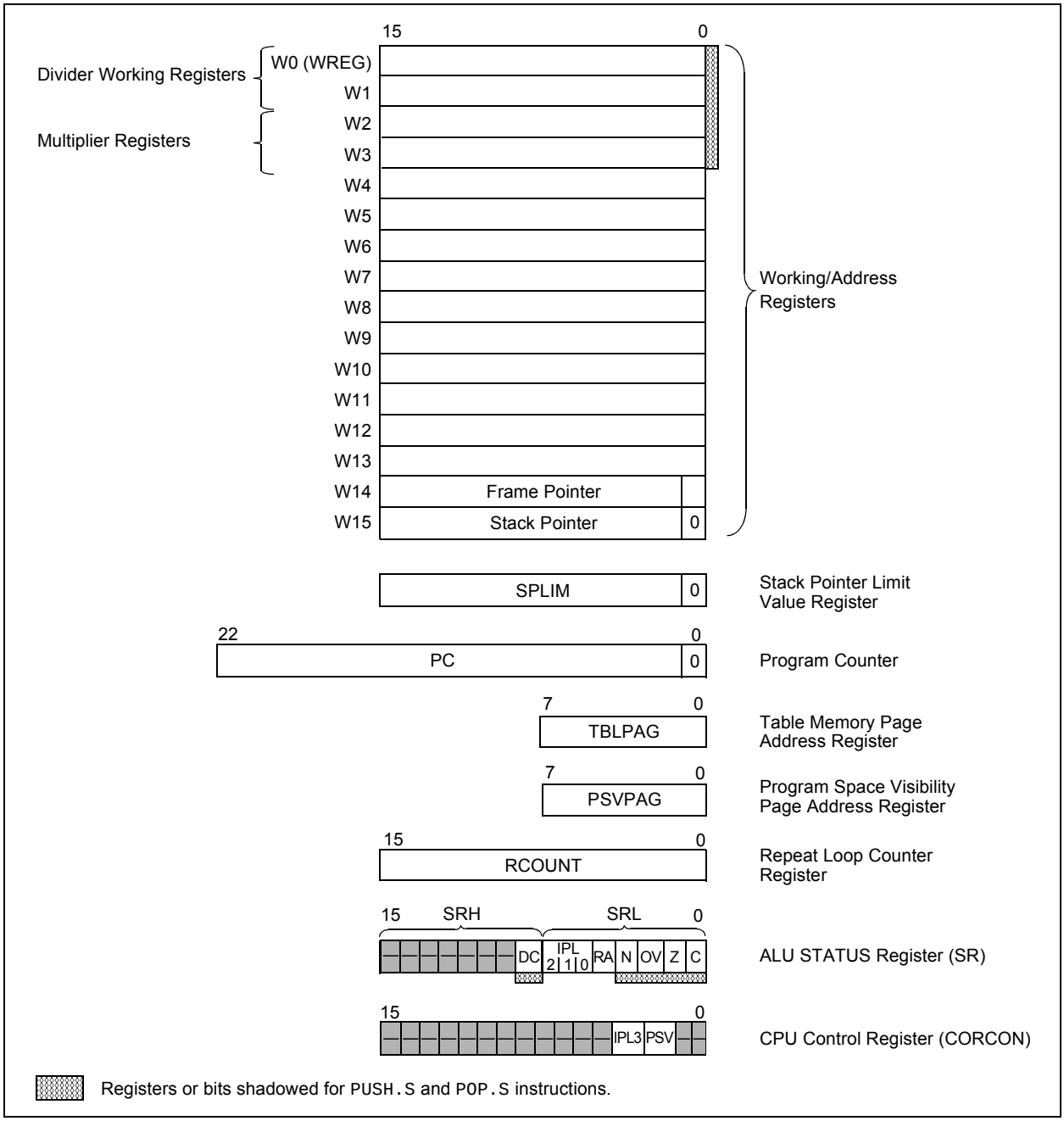



**TABLE 3-1: CPU CORE REGISTERS**

| Register(s) Name | Description |
|---|---|
| W0 through W15 | Working Register Array |
| PC | 23-Bit Program Counter |
| SR | ALU STATUS Register |
| SPLIM | Stack Pointer Limit Value Register |
| TBLPAG | Table Memory Page Address Register |
| PSVPAG | Program Space Visibility Page Address Register |
| RCOUNT | Repeat Loop Counter Register |
| CORCON | CPU Control Register |

**FIGURE 3-2: PROGRAMMER'S MODEL**



## 3.2 CPU Control Registers

**REGISTER 3-1: SR: ALU STATUS REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0, HSC |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | DC |
| bit 15 | | | | | | | bit 8 |

| R/W-0, HSC(1) | R/W-0, HSC(1) | R/W-0, HSC(1) | R-0, HSC | R/W-0, HSC | R/W-0, HSC | R/W-0, HSC | R/W-0, HSC |
|---|---|---|---|---|---|---|---|
| IPL2(2) | IPL1(2) | IPL0(2) | RA | N | OV | Z | C |
| bit 7 | | | | | | | bit 0 |

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

bit 15-9 **Unimplemented:** Read as '0'

bit 8 **DC:** ALU Half Carry/Borrow bit
1 = A carry-out from the 4th low-order bit (for byte-sized data) or 8th low-order bit (for word-sized data) of the result occurred
0 = No carry-out from the 4th or 8th low-order bit of the result has occurred

bit 7-5 **IPL<2:0>:** CPU Interrupt Priority Level Status bits(1,2)
111 = CPU interrupt priority level is 7 (15); user interrupts disabled
110 = CPU interrupt priority level is 6 (14)
101 = CPU Interrupt priority Level is 5 (13)
100 = CPU interrupt priority level is 4 (12)
011 = CPU interrupt priority level is 3 (11)
010 = CPU interrupt priority level is 2 (10)
001 = CPU interrupt priority level is 1 (9)
000 = CPU interrupt priority level is 0 (8)

bit 4 **RA:** REPEAT Loop Active bit
1 = REPEAT loop in progress
0 = REPEAT loop not in progress

bit 3 **N:** ALU Negative bit
1 = Result was negative
0 = Result was non-negative (zero or positive)

bit 2 **OV:** ALU Overflow bit
1 = Overflow occurred for signed (2's complement) arithmetic in this arithmetic operation
0 = No overflow has occurred

bit 1 **Z:** ALU Zero bit
1 = An operation, which effects the Z bit, has set it at some time in the past
0 = The most recent operation, which effects the Z bit, has cleared it (i.e., a non-zero result)

bit 0 **C:** ALU Carry/$\overline{\text{Borrow}}$ bit
1 = A carry-out from the Most Significant bit (MSb) of the result occurred
0 = No carry-out from the Most Significant bit (MSb) of the result occurred

**Note 1:** The IPL Status bits are read-only when NSTDIS (INTCON1<15>) = 1.
**2:** The IPL Status bits are concatenated with the IPL3 bit (CORCON<3>) to form the CPU Interrupt Priority Level (IPL). The value in parentheses indicates the IPL when IPL3 = 1.

**REGISTER 3-2: CORCON: CPU CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | R/C-0, HSC | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | IPL3(1) | PSV | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-4 **Unimplemented:** Read as '0'

bit 3 **IPL3:** CPU Interrupt Priority Level Status bit(1)
1 = CPU interrupt priority level is greater than 7
0 = CPU interrupt priority level is 7 or less

bit 2 **PSV:** Program Space Visibility in Data Space Enable bit
1 = Program space visible in data space
0 = Program space not visible in data space

bit 1-0 **Unimplemented:** Read as '0'

**Note 1:** User interrupts are disabled when IPL3 = 1.

## 3.3 Arithmetic Logic Unit (ALU)

The PIC24F ALU is 16 bits wide and is capable of addition, subtraction, bit shifts and logic operations. Unless otherwise mentioned, arithmetic operations are 2's complement in nature. Depending on the operation, the ALU may affect the values of the Carry (C), Zero (Z), Negative (N), Overflow (OV) and Digit Carry (DC) Status bits in the SR register. The C and DC Status bits operate as Borrow and Digit Borrow bits, respectively, for subtraction operations.

The ALU can perform 8-bit or 16-bit operations, depending on the mode of the instruction that is used. Data for the ALU operation can come from the W register array, or data memory, depending on the addressing mode of the instruction. Likewise, output data from the ALU can be written to the W register array or a data memory location.

The PIC24F CPU incorporates hardware support for both multiplication and division. This includes a dedicated hardware multiplier and support hardware division for 16-bit divisor.

### 3.3.1 MULTIPLIER

The ALU contains a high-speed, 17-bit x 17-bit multiplier. It supports unsigned, signed or mixed sign operation in several multiplication modes:

- 16-bit x 16-bit signed
- 16-bit x 16-bit unsigned
- 16-bit signed x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit unsigned
- 16-bit unsigned x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit signed
- 8-bit unsigned x 8-bit unsigned

### 3.3.2 DIVIDER

The divide block supports 32-bit/16-bit and 16-bit/16-bit signed and unsigned integer divide operations with the following data sizes:

1. 32-bit signed/16-bit signed divide
2. 32-bit unsigned/16-bit unsigned divide
3. 16-bit signed/16-bit signed divide
4. 16-bit unsigned/16-bit unsigned divide

The quotient for all divide instructions ends up in W0 and the remainder in W1. Sixteen-bit signed and unsigned DIV instructions can specify any W register for both the 16-bit divisor (Wn), and any W register (aligned) pair (W(m + 1):Wm) for the 32-bit dividend. The divide algorithm takes one cycle per bit of divisor, so both 32-bit/16-bit and 16-bit/16-bit instructions take the same number of cycles to execute.

### 3.3.3 MULTI-BIT SHIFT SUPPORT

The PIC24F ALU supports both single bit and single-cycle, multi-bit arithmetic and logic shifts. Multi-bit shifts are implemented using a shifter block, capable of performing up to a 15-bit arithmetic right shift, or up to a 15-bit left shift, in a single cycle. All multi-bit shift instructions only support Register Direct Addressing for both the operand source and result destination.

A full summary of instructions that use the shift operation is provided below in Table 3-2.

**TABLE 3-2: INSTRUCTIONS THAT USE THE SINGLE AND MULTI-BIT SHIFT OPERATION**

| Instruction | Description |
|---|---|
| ASR | Arithmetic shift right source register by one or more bits. |
| SL | Shift left source register by one or more bits. |
| LSR | Logical shift right source register by one or more bits. |

## 4.0 MEMORY ORGANIZATION

As with Harvard architecture devices, the PIC24F microcontrollers feature separate program and data memory space and busing. This architecture also allows the direct access of program memory from the data space during code execution.

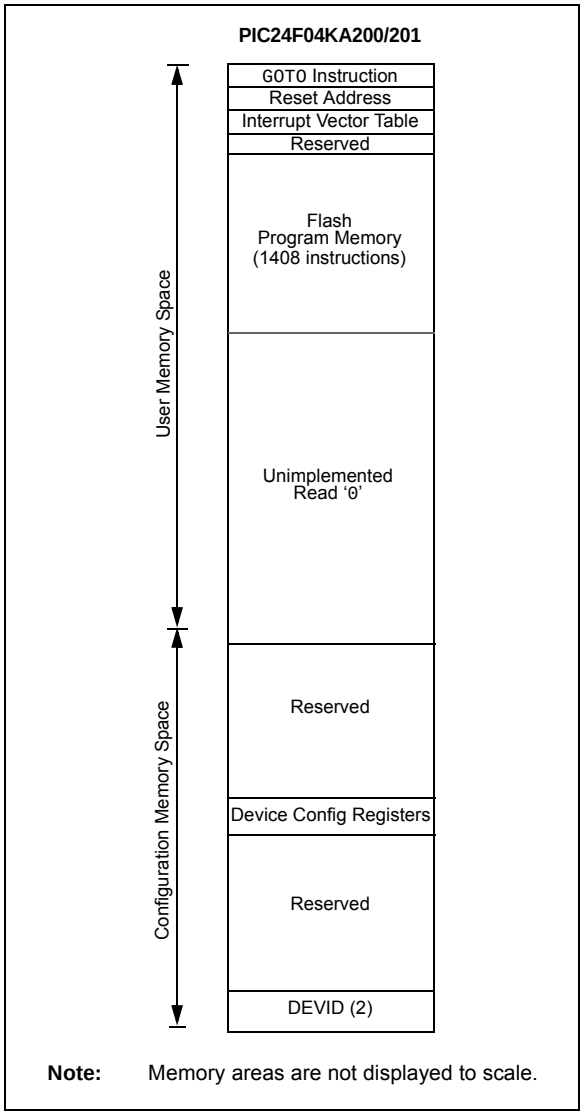
### 4.1 Program Address Space

The program address memory space of the PIC24F devices is 4M instructions. The space is addressable by a 24-bit value derived from either the 23-bit Program Counter (PC) during program execution, or from a table operation or data space remapping, as described in **Section 4.3 "Interfacing Program and Data Memory Spaces"**.

The user access to the program memory space is restricted to the lower half of the address range (000000h to 7FFFFFh). The exception is the use of TBLRD/TBLWT operations, which use TBLPAG<7> to permit access to the Configuration bits and Device ID sections of the configuration memory space.

The memory map for the PIC24F04KA201 family of devices is displayed in Figure 4-1.

**FIGURE 4-1: PROGRAM SPACE MEMORY MAP FOR PIC24F04KA201 FAMILY DEVICES**




**Note:** Memory areas are not displayed to scale.

### 4.1.1 PROGRAM MEMORY ORGANIZATION

The program memory space is organized in word-addressable blocks. Although it is treated as 24 bits wide, it is more appropriate to think of each address of the program memory as a lower and upper word, with the upper byte of the upper word being unimplemented. The lower word always has an even address, while the upper word has an odd address (Figure 4-2).

Program memory addresses are always word-aligned on the lower word, and addresses are incremented or decremented by two during code execution. This arrangement also provides compatibility with data memory space addressing and makes it possible to access data in the program memory space.

### 4.1.2 HARD MEMORY VECTORS

All PIC24F devices reserve the addresses between 00000h and 000200h for hard coded program execution vectors. A hardware Reset vector is provided to redirect code execution from the default value of the PC on device Reset to the actual start of code. A GOTO instruction is programmed by the user at 000000h with the actual address for the start of code at 000002h.

PIC24F devices also have two interrupt vector tables, located from 000004h to 0000FFh and 000104h to 0001FFh. These vector tables allow each of the many device interrupt sources to be handled by separate ISRs. **Section 7.1 "Interrupt Vector (IVT) Table"** discusses the interrupt vector tables more in detail.

### 4.1.3 DEVICE CONFIGURATION WORDS

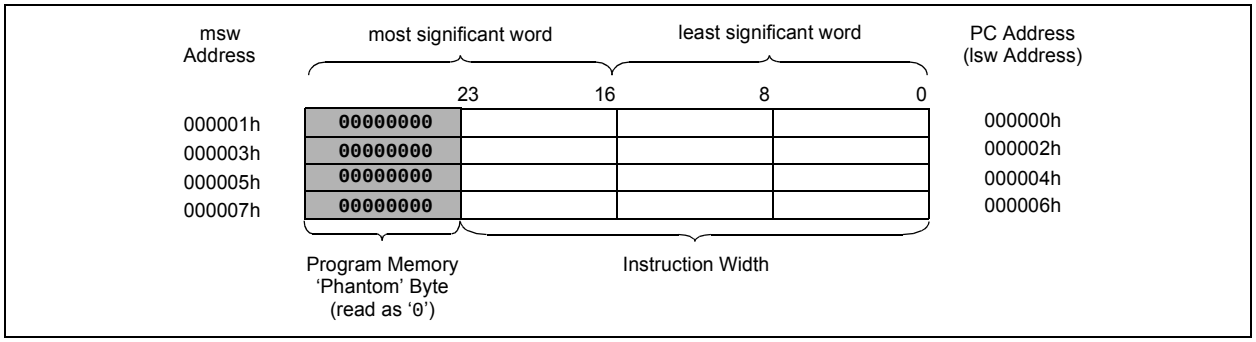
Table 4-1 provides the addresses of the device Configuration Words for the PIC24F04KA201 family. Their location in the memory map is displayed in Figure 4-1.

Refer to **Section 23.1 "Configuration Bits"** for more information on device Configuration Words.

**TABLE 4-1: DEVICE CONFIGURATION WORDS FOR PIC24F04KA201 FAMILY DEVICES**

| Configuration Word | Configuration Word Addresses |
|---|---|
| FBS | F80000 |
| FGS | F80004 |
| FOSCSEL | F80006 |
| FOSC | F80008 |
| FWDT | F8000A |
| FPOR | F8000C |
| FICD | F8000E |
| FDS | F80010 |

**FIGURE 4-2: PROGRAM MEMORY ORGANIZATION**



## 4.2 Data Address Space

The PIC24F core has a separate, 16-bit wide data memory space, addressable as a single linear range. The data space is accessed using two Address Generation Units (AGUs), one each for read and write operations. The data space memory map is displayed in Figure 4-3.

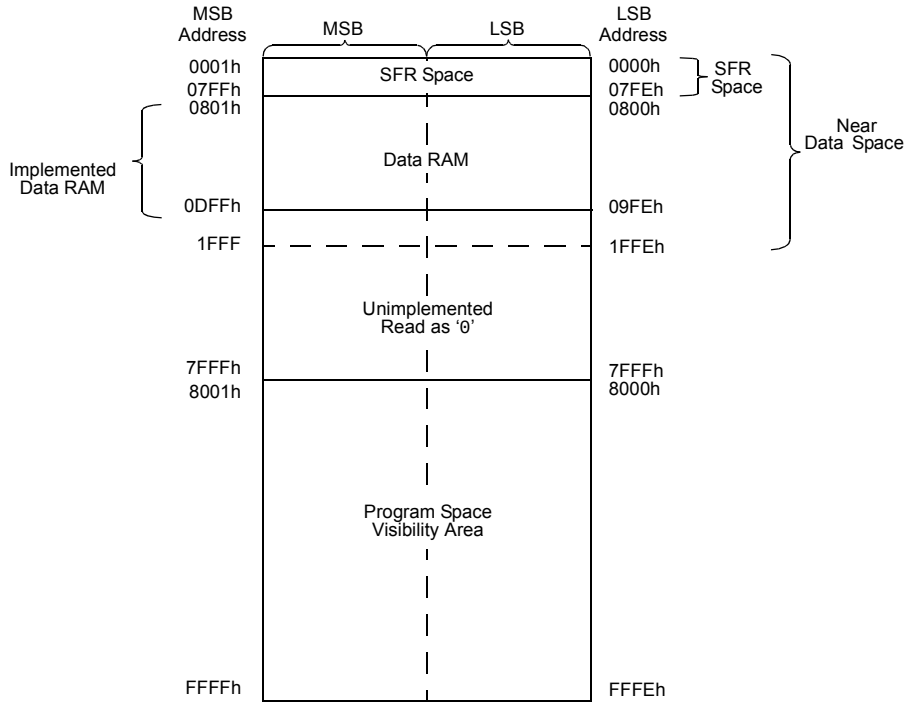
All Effective Addresses (EAs) in the data memory space are 16 bits wide and point to bytes within the data space. This gives a data space address range of 64 Kbytes or 32K words. The lower half of the data memory space (that is, when EA<15> = 0) is used for implemented memory addresses, while the upper half (EA<15> = 1) is reserved for the Program Space Visibility (PSV) area (see **Section 4.3.3 "Reading Data From Program Memory Using Program Space Visibility"**).

PIC24F04KA201 family devices implement a total of 768 words of data memory. Should an EA point to a location outside of this area, an all zero word or byte will be returned.

### 4.2.1 DATA SPACE WIDTH

The data memory space is organized in byte-addressable, 16-bit wide blocks. Data is aligned in data memory and registers as 16-bit words, but all the data space EAs resolve to bytes. The Least Significant Bytes (LSBs) of each word have even addresses, while the Most Significant Bytes (MSBs) have odd addresses.

**FIGURE 4-3: DATA SPACE MEMORY MAP FOR PIC24F04KA201 FAMILY DEVICES**




**Note:** Data memory areas are not shown to scale.

### 4.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PIC® devices and improve data space memory usage efficiency, the PIC24F instruction set supports both word and byte operations. As a consequence of byte accessibility, all EA calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

Data byte reads will read the complete word, which contains the byte, using the LSB of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, the data memory and the registers are organized as two parallel, byte-wide entities with shared (word) address decode but separate write lines. Data byte writes only write to the corresponding side of the array or register, which matches the byte address.

All word accesses must be aligned to an even address. Misaligned word data fetches are not supported, so care must be taken when mixing byte and word operations, or translating from 8-bit MCU code. If a misaligned read or write is attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed; if it occurred on a write, the instruction will be executed, but the write will not occur. In either case, a trap is then executed, allowing the system and/or user to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the LSB. The MSB is not modified.

A sign-extend instruction (SE) is provided to allow the users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MSB of any W register by executing a zero-extend (ZE) instruction on the appropriate address.

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions operate only on words.

### 4.2.3 NEAR DATA SPACE

The 8-Kbyte area between 0000h and 1FFFh is referred to as the near data space. Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. The remainder of the data space is addressable indirectly. Additionally, the whole data space is addressable using MOV instructions, which support Memory Direct Addressing (MDA) with a 16-bit address field. For PIC24F04KA201 family devices, the entire implemented data memory lies in Near Data Space (NDS).

### 4.2.4 SFR SPACE

The first 2 Kbytes of the near data space, from 0000h to 07FFh, are primarily occupied with Special Function Registers (SFRs). These are used by the PIC24F core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control and are generally grouped together by that module. Much of the SFR space contains unused addresses; these are read as '0'. The SFR space, where the SFRs are actually implemented, is provided in Table 4-2. Each implemented area indicates a 32-byte region where at least one address is implemented as an SFR. A complete listing of implemented SFRs, including their addresses, is provided in Table 4-3 through Table 4-21.

**TABLE 4-2: IMPLEMENTED REGIONS OF SFR DATA SPACE**

| | SFR Space Address | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **xx00** | **xx20** | **xx40** | **xx60** | **xx80** | **xxA0** | **xxC0** | **xxE0** |
| **000h** | Core | | | ICN | Interrupts | | | — |
| **100h** | Timers | | Capture | — | Compare | — | — | — |
| **200h** | $I^2C$™ | UART | SPI | | — | — | I/O | |
| **300h** | ADC/CMTU | | — | — | — | — | — | — |
| **400h** | — | — | — | — | — | — | — | — |
| **500h** | — | — | — | — | — | — | — | — |
| **600h** | — | Comp | — | — | — | | | |
| **700h** | — | — | System/DS/HLVD | NVM/PMD | — | — | — | — |

**Legend:** — = No implemented SFRs in this block.

**TABLE 4-3: CPU CORE REGISTERS MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WREG0 | 0000 | Working Register 0 | | | | | | | | | | | | | | | | 0000 |
| WREG1 | 0002 | Working Register 1 | | | | | | | | | | | | | | | | 0000 |
| WREG2 | 0004 | Working Register 2 | | | | | | | | | | | | | | | | 0000 |
| WREG3 | 0006 | Working Register 3 | | | | | | | | | | | | | | | | 0000 |
| WREG4 | 0008 | Working Register 4 | | | | | | | | | | | | | | | | 0000 |
| WREG5 | 000A | Working Register 5 | | | | | | | | | | | | | | | | 0000 |
| WREG6 | 000C | Working Register 6 | | | | | | | | | | | | | | | | 0000 |
| WREG7 | 000E | Working Register 7 | | | | | | | | | | | | | | | | 0000 |
| WREG8 | 0010 | Working Register 8 | | | | | | | | | | | | | | | | 0000 |
| WREG9 | 0012 | Working Register 9 | | | | | | | | | | | | | | | | 0000 |
| WREG10 | 0014 | Working Register 10 | | | | | | | | | | | | | | | | 0000 |
| WREG11 | 0016 | Working Register 11 | | | | | | | | | | | | | | | | 0000 |
| WREG12 | 0018 | Working Register 12 | | | | | | | | | | | | | | | | 0000 |
| WREG13 | 001A | Working Register 13 | | | | | | | | | | | | | | | | 0000 |
| WREG14 | 001C | Working Register 14 | | | | | | | | | | | | | | | | 0000 |
| WREG15 | 001E | Working Register 15 | | | | | | | | | | | | | | | | 0800 |
| SPLIM | 0020 | Stack Pointer Limit Value Register | | | | | | | | | | | | | | | | xxxx |
| PCL | 002E | Program Counter Low Byte Register | | | | | | | | | | | | | | | | 0000 |
| PCH | 0030 | — | — | — | — | — | — | — | — | Program Counter Register High Byte | | | | | | | | 0000 |
| TBLPAG | 0032 | — | — | — | — | — | — | — | — | Table Memory Page Address Register | | | | | | | | 0000 |
| PSVPAG | 0034 | — | — | — | — | — | — | — | — | Program Space Visibility Page Address Register | | | | | | | | 0000 |
| RCOUNT | 0036 | REPEAT Loop Counter Register | | | | | | | | | | | | | | | | xxxx |
| SR | 0042 | — | — | — | — | — | — | — | DC | IPL2 | IPL1 | IPL0 | RA | N | OV | Z | C | 0000 |
| CORCON | 0044 | — | — | — | — | — | — | — | — | — | — | — | — | IPL3 | PSV | — | — | 0000 |
| DISICNT | 0052 | — | — | Disable Interrupts Counter Register | | | | | | | | | | | | | | xxxx |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-4: ICN REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNEN1 | 0060 | — | CN14IE(1) | CN13IE(1) | CN12IE | CN11IE | — | — | CN8IE | — | CN6IE(1) | CN5IE(1) | CN4IE(1) | CN3IE | CN2IE | CN1IE | CN0IE | 0000 |
| CNEN2 | 0062 | — | CN30IE | CN29IE | — | — | — | — | — | CN23IE(1) | CN22IE | CN21IE | — | — | — | — | — | 0000 |
| CNPU1 | 0068 | — | CN14PUE(1) | CN13PUE(1) | CN12PUE | CN11PUE | — | — | CN8PUE | — | CN6PUE(1) | CN5PUE(1) | CN4PUE(1) | CN3PUE | CN2PUE | CN1PUE | CN0PUE | 0000 |
| CNPU2 | 006A | — | CN30PUE | CN29PUE | — | — | — | — | — | CN23PUE(1) | CN22PUE | CN21PUE | — | — | — | — | — | 0000 |
| CNPD1 | 0070 | — | CN14PDE(1) | CN13PDE(1) | CN12PDE | CN11PDE | — | — | CN8PDE | — | CN6PDE(1) | CN5PDE(1) | CN4PDE(1) | CN3PDE | CN2PDE | CN1PDE | CN0PDE | 0000 |
| CNPD2 | 0072 | — | CN30PDE | CN29PDE | — | — | — | — | — | CN23PDE(1) | CN22PDE | CN21PDE | — | — | — | — | — | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**Note 1:** These bits are not implemented on 14-pin devices.

**TABLE 4-5: INTERRUPT CONTROLLER REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTCON1 | 0080 | NSTDIS | — | — | — | — | — | — | — | — | — | — | MATHERR | ADDRERR | STKERR | OSCFAIL | — | 0000 |
| INTCON2 | 0082 | ALTIVT | DISI | — | — | — | — | — | — | — | — | — | — | — | INT2EP | INT1EP | INT0EP | 0000 |
| IFS0 | 0084 | NVMIF | — | AD1IF | U1TXIF | U1RXIF | SPI1IF | SPF1IF | T3IF | T2IF | — | — | — | T1IF | OC1IF | IC1IF | INT0IF | 0000 |
| IFS1 | 0086 | — | — | INT2IF | — | — | — | — | — | — | — | — | INT1IF | CNIF | CMIF | MI2C1IF | SI2C1IF | 0000 |
| IFS3 | 008A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 |
| IFS4 | 008C | — | — | CTMUIF | — | — | — | — | HLVDIF | — | — | — | — | — | — | U1ERIF | — | 0000 |
| IEC0 | 0094 | NVMIE | — | AD1IE | U1TXIE | U1RXIE | SPI1IE | SPF1IE | T3IE | T2IE | — | — | — | T1IE | OC1IE | IC1IE | INT0IE | 0000 |
| IEC1 | 0096 | — | — | INT2IE | — | — | — | — | — | — | — | — | INT1IE | CNIE | CMIE | MI2C1IE | SI2C1IE | 0000 |
| IEC3 | 009A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 |
| IEC4 | 009C | — | — | CTMUIE | — | — | — | — | HLVDIE | — | — | — | — | — | — | U1ERIE | — | 0000 |
| IPC0 | 00A4 | — | T1IP2 | T1IP1 | T1IP0 | — | OC1IP2 | OC1IP1 | OC1IP0 | — | IC1IP2 | IC1IP1 | IC1IP0 | — | INT0IP2 | INT0IP1 | INT0IP0 | 4444 |
| IPC1 | 00A6 | — | T2IP2 | T2IP1 | T2IP0 | — | — | — | — | — | — | — | — | — | — | — | — | 4444 |
| IPC2 | 00A8 | — | U1RXIP2 | U1RXIP1 | U1RXIP0 | — | SPI1IP2 | SPI1IP1 | SPI1IP0 | — | SPF1IP2 | SPF1IP1 | SPF1IP0 | — | T3IP2 | T3IP1 | T3IP0 | 4444 |
| IPC3 | 00AA | — | NVMIP2 | NVMIP1 | NVMIP0 | — | — | — | — | — | AD1IP2 | AD1IP1 | AD1IP0 | — | U1TXIP2 | U1TXIP1 | U1TXIP0 | 4044 |
| IPC4 | 00AC | — | CNIP2 | CNIP1 | CNIP0 | — | CMIP2 | CMIP1 | CMIP0 | — | MI2C1P2 | MI2C1P1 | MI2C1P0 | — | SI2C1P2 | SI2C1P1 | SI2C1P0 | 4444 |
| IPC5 | 00AE | — | — | — | — | — | — | — | — | — | — | — | — | — | INT1IP2 | INT1IP1 | INT1IP0 | 0004 |
| IPC7 | 00B2 | — | — | — | — | — | — | — | — | — | INT2IP2 | INT2IP1 | INT2IP0 | — | — | — | — | 4440 |
| IPC16 | 00C4 | — | — | — | — | — | — | — | — | — | U1ERIP2 | U1ERIP1 | U1ERIP0 | — | — | — | — | 4440 |
| IPC18 | 00C8 | — | — | — | — | — | — | — | — | — | — | — | — | — | HLVDIP2 | HLVDIP1 | HLVDIP0 | 0004 |
| IPC19 | 00CA | — | — | — | — | — | — | — | — | — | CTMUIP2 | CTMUIP1 | CTMUIP0 | — | — | — | — | 0040 |
| INTTREG | 00E0 | CPUIRQ | — | VHOLD | — | ILR3 | ILR2 | ILR1 | ILR0 | — | VECNUM6 | VECNUM5 | VECNUM4 | VECNUM3 | VECNUM2 | VECNUM1 | VECNUM0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-6: TIMER REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMR1 | 0100 | Timer1 Register | | | | | | | | | | | | | | | | 0000 |
| PR1 | 0102 | Timer1 Period Register | | | | | | | | | | | | | | | | FFFF |
| T1CON | 0104 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | — | TSYNC | TCS | — | 0000 |
| TMR2 | 0106 | Timer2 Register | | | | | | | | | | | | | | | | 0000 |
| TMR3HLD | 0108 | Timer3 Holding Register (for 32-bit timer operations only) | | | | | | | | | | | | | | | | 0000 |
| TMR3 | 010A | Timer3 Register | | | | | | | | | | | | | | | | 0000 |
| PR2 | 010C | Timer2 Period Register | | | | | | | | | | | | | | | | FFFF |
| PR3 | 010E | Timer3 Period Register | | | | | | | | | | | | | | | | FFFF |
| T2CON | 0110 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | T32 | — | TCS | — | 0000 |
| T3CON | 0112 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | — | — | TCS | — | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-7: INPUT CAPTURE REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IC1BUF | 0140 | Input Capture 1 Register | | | | | | | | | | | | | | | | FFFF |
| IC1CON | 0142 | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI1 | ICI0 | ICOV | ICBNE | ICM2 | ICM1 | ICM0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-8: OUTPUT COMPARE REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OC1RS | 0180 | Output Compare 1 Secondary Register | | | | | | | | | | | | | | | | FFFF |
| OC1R | 0182 | Output Compare 1 Register | | | | | | | | | | | | | | | | FFFF |
| OC1CON | 0184 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM2 | OCM1 | OCM0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-9: I²C™ REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| I2C1RCV | 0200 | — | — | — | — | — | — | — | — | I2C1 Receive Register | | | | | | | | 0000 |
| I2C1TRN | 0202 | — | — | — | — | — | — | — | — | I2C1 Transmit Register | | | | | | | | 00FF |
| I2C1BRG | 0204 | — | — | — | — | — | — | — | I2C1 Baud Rate Generator Register | | | | | | | | | 0000 |
| I2C1CON | 0206 | I2CEN | — | I2CSIDL | SCLREL | IPMIEN | A10M | DISSLW | SMEN | GCEN | STREN | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 1000 |
| I2C1STAT | 0208 | ACKSTAT | TRSTAT | — | — | — | BCL | GCSTAT | ADD10 | IWCOL | I2COV | D/$\overline{A}$ | P | S | R/$\overline{W}$ | RBF | TBF | 0000 |
| I2C1ADD | 020A | — | — | — | — | — | — | I2C1 Address Register | | | | | | | | | | 0000 |
| I2C1MSK | 020C | — | — | — | — | — | — | AMSK9 | AMSK8 | AMSK7 | AMSK6 | AMSK5 | AMSK4 | AMSK3 | AMSK2 | AMSK1 | AMSK0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-10: UART REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U1MODE | 0220 | UARTEN | — | USIDL | IREN | RTSMD | — | UEN1 | UEN0 | WAKE | LPBACK | ABAUD | RXINV | BRGH | PDSEL1 | PDSEL0 | STSEL | 0000 |
| U1STA | 0222 | UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0110 |
| U1TXREG | 0224 | — | — | — | — | — | — | — | UART1 Transmit Register | | | | | | | | | 0000 |
| U1RXREG | 0226 | — | — | — | — | — | — | — | UART1 Receive Register | | | | | | | | | 0000 |
| U1BRG | 0228 | Baud Rate Generator Prescaler Register | | | | | | | | | | | | | | | | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-11: SPI REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPI1STAT | 0240 | SPIEN | — | SPISIDL | — | — | SPIBEC2 | SPIBEC1 | SPIBEC0 | SRMPT | SPIROV | SRXMPT | SISEL2 | SISEL1 | SISEL0 | SPITBF | SPIRBF | 0000 |
| SPI1CON1 | 0242 | — | — | — | DISSCK | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 | 0000 |
| SPI1CON2 | 0244 | FRMEN | SPIFSD | SPIFPOL | — | — | — | — | — | — | — | — | — | — | — | SPIFE | SPIBEN | 0000 |
| SPI1BUF | 0248 | SPI1 Transmit/Receive Buffer | | | | | | | | | | | | | | | | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-12: PORTA REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5(1) | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TRISA | 02C0 | — | — | — | — | — | — | — | — | — | TRISA6 | — | TRISA4 | TRISA3(2,3) | TRISA2(2) | TRISA1 | TRISA0 | 00DF |
| PORTA | 02C2 | — | — | — | — | — | — | — | — | — | RA6 | RA5 | RA4 | RA3(2,3) | RA2(2) | RA1 | RA0 | xxxx |
| LATA | 02C4 | — | — | — | — | — | — | — | — | — | LATA6 | — | LATA4 | LATA3(2,3) | LATA2(2) | LATA1 | LATA0 | xxxx |
| ODCA | 02C6 | — | — | — | — | — | — | — | — | — | ODA6 | — | ODA4 | ODA3(2,3) | ODA2(2) | ODA1 | ODA0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**Note 1:** Bit available only when MCLRE = 0.
**2:** Bits are available only when the primary oscillator is disabled (POSCMD1:POSCMD0 = 00); otherwise read as '0'.
**3:** Bits are available only when the primary oscillator is disabled or EC mode is selected (POSCMD1:POSCMD0 = 00 or 11) and CLKO is disabled (OSCIOFNC = 0); otherwise read as '0'.

**TABLE 4-13: PORTB REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TRISB | 02C8 | TRISB15 | TRISB14 | TRISB13(1) | TRISB12(1) | — | — | TRISB9 | TRISB8 | TRISB7 | — | — | TRISB4 | — | TRISB2(1) | TRISB1(1) | TRISB0(1) | FFFF |
| PORTB | 02CA | RB15 | RB14 | RB13(1) | RB12(1) | — | — | RB9 | RB8 | RB7 | — | — | RB4 | — | RB2(1) | RB1(1) | RB0(1) | xxxx |
| LATB | 02CC | LATB15 | LATB14 | LATB13(1) | LATB12(1) | — | — | LATB9 | LATB8 | LATB7 | — | — | LATB4 | — | LATB2(1) | LATB1(1) | LATB0(1) | xxxx |
| ODCB | 02CE | ODB15 | ODB14 | ODB13(1) | ODB12(1) | — | — | ODB9 | ODB8 | ODB7 | — | — | ODB4 | — | ODB2(1) | ODB1(1) | ODB(1)0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**Note 1:** These bits are not implemented on 14-pin devices.

**TABLE 4-14: PAD CONFIGURATION REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PADCFG1 | 02FC | — | — | — | — | — | — | — | — | — | — | — | SMBUSDEL | OC1TRIS | — | — | — | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-15: ADC REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADC1BUF0 | 0300 | ADC Data Buffer 0 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF1 | 0302 | ADC Data Buffer 1 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF2 | 0304 | ADC Data Buffer 2 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF3 | 0306 | ADC Data Buffer 3 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF4 | 0308 | ADC Data Buffer 4 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF5 | 030A | ADC Data Buffer 5 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF6 | 030C | ADC Data Buffer 6 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF7 | 030E | ADC Data Buffer 7 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF8 | 0310 | ADC Data Buffer 8 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUF9 | 0312 | ADC Data Buffer 9 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFA | 0314 | ADC Data Buffer 10 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFB | 0316 | ADC Data Buffer 11 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFC | 0318 | ADC Data Buffer 12 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFD | 031A | ADC Data Buffer 13 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFE | 031C | ADC Data Buffer 14 | | | | | | | | | | | | | | | | xxxx |
| ADC1BUFF | 031E | ADC Data Buffer 15 | | | | | | | | | | | | | | | | xxxx |
| AD1CON1 | 0320 | ADON | — | ADSIDL | — | — | — | FORM1 | FORM0 | SSRC2 | SSRC1 | SSRC0 | — | — | ASAM | SAMP | DONE | 0000 |
| AD1CON2 | 0322 | VCFG2 | VCFG1 | VCFG0 | OFFCAL | — | CSCNA | — | — | BUFS | — | SMPI3 | SMPI2 | SMPI1 | SMPI0 | BUFM | ALTS | 0000 |
| AD1CON3 | 0324 | ADRC | — | — | SAMC4 | SAMC3 | SAMC2 | SAMC1 | SAMC0 | — | — | ADCS5 | ADCS4 | ADCS3 | ADCS2 | ADCS1 | ADCS0 | 0000 |
| AD1CHS | 0328 | CH0NB | — | — | — | CH0SB3 | CH0SB2 | CH0SB1 | CH0SB0 | CH0NA | — | — | CH0SA4 | CH0SA3 | CH0SA2 | CH0SA1 | CH0SA0 | 0000 |
| AD1PCFG | 032C | — | — | — | PCFG12(1) | PCFG11(1) | PCFG10 | — | — | — | — | PCFG5 | PCFG4 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 0000 |
| AD1CSSL | 0330 | — | — | — | CSSL12(1) | CSSL11(1) | CSSL10 | — | — | — | — | CSSL5 | CSSL4 | CSSL3 | CSSL2 | CSSL1 | CSSL0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**Note 1:** These bits are not implemented on 14-pin devices.

**TABLE 4-16: CTMU REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CTMUCON | 033C | CTMUEN | — | CTMUSIDL | TGEN | EDGEN | EDGSEQEN | IDISSEN | CTTRIG | EDG2POL | EDG2SEL1 | EDG2SEL0 | EDG1POL | EDG1SEL1 | EDG1SEL0 | EDG2STAT | EDG1STAT | 0000 |
| CTMUICON | 033E | ITRIM5 | ITRIM4 | ITRIM3 | ITRIM2 | ITRIM1 | ITRIM0 | IRNG1 | IRNG0 | — | — | — | — | — | — | — | — | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-17: DUAL COMPARATOR REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMSTAT | 0630 | CMSIDL | — | — | — | — | — | C2EVT | C1EVT | — | — | — | — | — | — | C2OUT | C1OUT | 0000 |
| CVRCON | 0632 | — | — | — | — | — | — | — | — | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 0000 |
| CM1CON | 0634 | CON | COE | CPOL | CLPWR | — | — | CEVT | COUT | EVPOL1 | EVPOL0 | — | CREF | — | — | CCH1 | CCH0 | 0000 |
| CM2CON | 0636 | CON | COE | CPOL | CLPWR | — | — | CEVT | COUT | EVPOL1 | EVPOL0 | — | CREF | — | — | CCH1 | CCH0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-18: CLOCK CONTROL REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RCON | 0740 | TRAPR | IOPUWR | SBOREN | — | — | DPSLP | — | PMSLP | EXTR | SWR | SWDTEN | WDTO | SLEEP | IDLE | BOR | POR | (Note 1) |
| OSCCON | 0742 | — | COSC2 | COSC1 | COSC0 | — | NOSC2 | NOSC1 | NOSC0 | CLKLOCK | — | LOCK | — | CF | — | SOSCEN | OSWEN | (Note 2) |
| CLKDIV | 0744 | ROI | DOZE2 | DOZE1 | DOZE0 | DOZEN | RCDIV2 | RCDIV1 | RCDIV0 | — | — | — | — | — | — | — | — | 3140 |
| OSCTUN | 0748 | — | — | — | — | — | — | — | — | — | — | TUN5 | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 | 0000 |
| REFOCON | 074E | ROEN | — | ROSSLP | ROSEL | RODIV3 | RODIV2 | RODIV1 | RODIV0 | — | — | — | — | — | — | — | — | 0000 |
| HLVDCON | 0756 | HLVDEN | — | HLSIDL | — | — | — | — | — | VDIR | BGVST | IRVST | — | HLVDL3 | HLVDL2 | HLVDL1 | HLVDL0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**Note 1:** RCON register Reset values are dependent on type of Reset.
**2:** OSCCON register Reset values are dependent on configuration fuses and by type of Reset.

**TABLE 4-19: DEEP SLEEP REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets(1) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DSCON | 0758 | DSEN | — | — | — | — | — | — | — | — | — | — | — | — | — | DSBOR | RELEASE | 0000 |
| DSWSRC | 075A | — | — | — | — | — | — | — | DSINT0 | DSFLT | — | — | DSWDT | — | DSMCLR | — | DSPOR | 0000 |
| DSGPR0 | 075C | Deep Sleep General Purpose Register 0 | | | | | | | | | | | | | | | | 0000 |
| DSGPR1 | 075E | Deep Sleep General Purpose Register 1 | | | | | | | | | | | | | | | | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
**Note 1:** The Deep Sleep registers are only reset on a VDD POR event.

**TABLE 4-20: NVM REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NVMCON | 0760 | WR | WREN | WRERR | PGMONLY | — | — | — | — | — | ERASE | NVMOP5 | NVMOP4 | NVMOP3 | NVMOP2 | NVMOP1 | NVMOP0 | 0000(1) |
| NVMKEY | 0766 | — | — | — | — | — | — | — | — | NVMKEY7 | NVMKEY6 | NVMKEY5 | NVMKEY4 | NVMKEY3 | NVMKEY2 | NVMKEY1 | NVMKEY0 | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
**Note 1:** Reset value shown is for POR only. Value on other Reset states is dependent on the state of memory write or erase operations at the time of Reset.

**TABLE 4-21: PMD REGISTER MAP**

| File Name | Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMD1 | 0770 | — | — | T3MD | T2MD | T1MD | — | — | — | I2C1MD | — | U1MD | — | SPI1MD | — | — | ADC1MD | 0000 |
| PMD2 | 0772 | — | — | — | — | — | — | — | IC1MD | — | — | — | — | — | — | — | OC1MD | 0000 |
| PMD3 | 0774 | — | — | — | — | — | CMPMD | — | — | — | — | — | — | — | — | — | — | 0000 |
| PMD4 | 0776 | — | — | — | — | — | — | — | — | — | — | — | — | REFOMD | CTMUMD | HLVDMD | — | 0000 |

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

### 4.2.5 SOFTWARE STACK

In addition to its use as a working register, the W15 register in PIC24F devices is also used as a Software Stack Pointer. The pointer always points to the first available free word and grows from lower to higher addresses. It pre-decrements for stack pops and post-increments for stack pushes, as depicted in Figure 4-4.

For a PC push during any CALL instruction, the MSB of the PC is zero-extended before the push, ensuring that the MSB is always clear.

| **Note:** | A PC push during exception processing will concatenate the SRL register to the MSB of the PC prior to the push. |
|---|---|

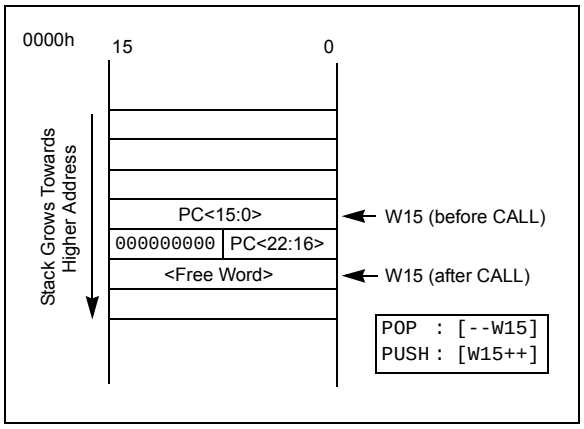
The Stack Pointer Limit Value (SPLIM) register, associated with the Stack Pointer, sets an upper address boundary for the stack. SPLIM is uninitialized at Reset. As is the case for the Stack Pointer, SPLIM<0> is forced to '0' as all stack operations must be word-aligned. Whenever an EA is generated using W15 as a source or destination pointer, the resulting address is compared with the value in SPLIM. If the contents of the Stack Pointer (W15) and the SPLIM register are equal, and a push operation is performed, a stack error trap will not occur. The stack error trap will occur on a subsequent push operation.

Thus, for example, if it is desirable to cause a stack error trap when the stack grows beyond address 09FF in RAM, initialize the SPLIM with the value, 09FD.

Similarly, a Stack Pointer underflow (stack error) trap is generated when the Stack Pointer address is found to be less than 0800h. This prevents the stack from interfering with the Special Function Register (SFR) space.

| **Note:** | A write to the SPLIM register should not be immediately followed by an indirect read operation using W15. |
|---|---|

**FIGURE 4-4: CALL STACK FRAME**




## 4.3 Interfacing Program and Data Memory Spaces

The PIC24F architecture uses a 24-bit wide program space and 16-bit wide data space. The architecture is also a modified Harvard scheme, meaning that data can also be present in the program space. To use this data successfully, it must be accessed in a way that preserves the alignment of information in both spaces.

Apart from the normal execution, the PIC24F architecture provides two methods by which the program space can be accessed during operation:

- Using table instructions to access individual bytes or words anywhere in the program space
- Remapping a portion of the program space into the data space, PSV

Table instructions allow an application to read or write small areas of the program memory. This makes the method ideal for accessing data tables that need to be updated from time to time. It also allows access to all bytes of the program word. The remapping method allows an application to access a large block of data on a read-only basis, which is ideal for look ups from a large table of static data. It can only access the least significant word (lsw) of the program word.

### 4.3.1 ADDRESSING PROGRAM SPACE

Since the address ranges for the data and program spaces are 16 and 24 bits, respectively, a method is needed to create a 23-bit or 24-bit program address from 16-bit data registers. The solution depends on the interface method to be used.

For table operations, the 8-bit Table Memory Page Address register (TBLPAG) is used to define a 32K word region within the program space. This is concatenated with a 16-bit EA to arrive at a full 24-bit program space address. In this format, the Most Significant bit (MSb) of TBLPAG is used to determine if the operation occurs in the user memory (TBLPAG<7> = 0) or the configuration memory (TBLPAG<7> = 1).

For remapping operations, the 8-bit Program Space Visibility Page Address register (PSVPAG) is used to define a 16K word page in the program space. When the MSb of the EA is '1', PSVPAG is concatenated with the lower 15 bits of the EA to form a 23-bit program space address. Unlike the table operations, this limits remapping operations strictly to the user memory area.

See Table 4-22 and Figure 4-5 to know how the program EA is created for table operations and remapping accesses from the data EA. Here, P<23:0> refers to a program space word, whereas D<15:0> refers to a data space word.

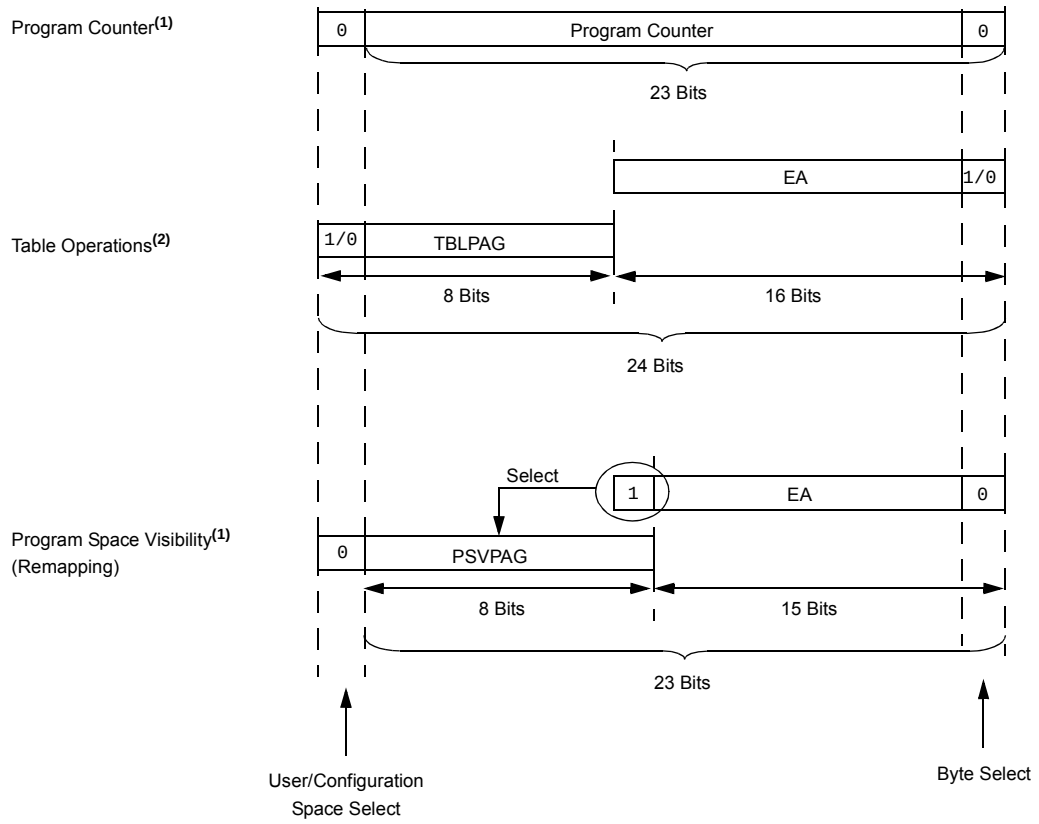
**TABLE 4-22: PROGRAM SPACE ADDRESS CONSTRUCTION**

| Access Type | Access Space | Program Space Address | | | | |
|---|---|---|---|---|---|---|
| | | <23> | <22:16> | <15> | <14:1> | <0> |
| Instruction Access (Code Execution) | User | 0 | PC<22:1> | | | 0 |
| | | 0xx xxxx xxxx xxxx xxxx xxx0 | | | | |
| TBLRD/TBLWT (Byte/Word Read/Write) | User | TBLPAG<7:0> | | Data EA<15:0> | | |
| | | 0xxx xxxx | | xxxx xxxx xxxx xxxx | | |
| | Configuration | TBLPAG<7:0> | | Data EA<15:0> | | |
| | | 1xxx xxxx | | xxxx xxxx xxxx xxxx | | |
| Program Space Visibility (Block Remap/Read) | User | 0 | PSVPAG<7:0>(2) | | Data EA<14:0>(1) | |
| | | 0 | xxxx xxxx | | xxx xxxx xxxx xxxx | |

**Note 1:** Data EA<15> is always '1' in this case, but is not used in calculating the program space address. Bit 15 of the address is PSVPAG<0>.

**2:** PSVPAG can have only one value ('00' to access program memory) on the PIC24F04KA201 family.

**FIGURE 4-5: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION**




**Note 1:** The LSb of program space addresses is always fixed as '0' in order to maintain word alignment of data in the program and data spaces.

**2:** Table operations are not required to be word-aligned. Table read operations are permitted in the configuration memory space.

### 4.3.2 DATA ACCESS FROM PROGRAM MEMORY USING TABLE INSTRUCTIONS

The TBLRDL and TBLWTL instructions offer a direct method of reading or writing the lower word of any address within the program memory without going through data space. The TBLRDH and TBLWTH instructions are the only method to read or write the upper 8 bits of a program space word as data.

The PC is incremented by 2 for each successive 24-bit program word. This allows program memory addresses to directly map to data space addresses. Program memory can thus be regarded as two 16-bit word-wide address spaces, residing side by side, each with the same address range. TBLRDL and TBLWTL access the space which contains the least significant data word, and TBLRDH and TBLWTH access the space which contains the upper data byte.

Two table instructions are provided to move byte or word-sized (16-bit) data to and from program space. Both function as either byte or word operations.

1. TBLRDL (Table Read Low): In Word mode, it maps the lower word of the program space location (P<15:0>) to a data address (D<15:0>).

   In Byte mode, either the upper or lower byte of the lower program word is mapped to the lower byte of a data address. The upper byte is selected when byte select is '1'; the lower byte is selected when it is '0'.

2. TBLRDH (Table Read High): In Word mode, it maps the entire upper word of a program address (P<23:16>) to a data address. Note that D<15:8>, the 'phantom' byte, will always be '0'.

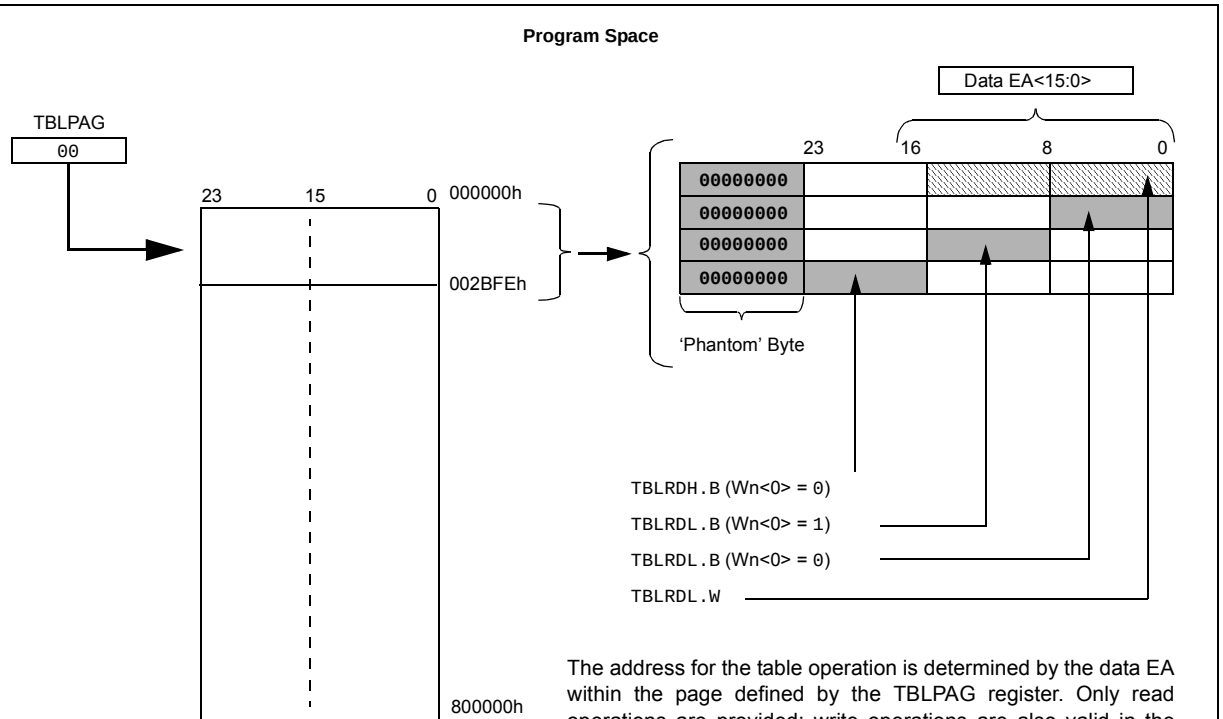
   In Byte mode, it maps the upper or lower byte of the program word to D<7:0> of the data address, as above. Note that the data will always be '0' when the upper 'phantom' byte is selected (byte select = 1).

In a similar fashion, two table instructions, TBLWTH and TBLWTL, are used to write individual bytes or words to a program space address. The details of their operation are explained in **Section 5.0 "Flash Program Memory"**.

For all table operations, the area of program memory space to be accessed is determined by the Table Memory Page Address register (TBLPAG). TBLPAG covers the entire program memory space of the device, including user and configuration spaces. When TBLPAG<7> = 0, the table page is located in the user memory space. When TBLPAG<7> = 1, the page is located in configuration space.

| **Note:** | Only table read operations will execute in the configuration memory space, and only then, in implemented areas such as the Device ID. Table write operations are not allowed. |
|---|---|

**FIGURE 4-6: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS**




The address for the table operation is determined by the data EA within the page defined by the TBLPAG register. Only read operations are provided; write operations are also valid in the user memory area.

### 4.3.3 READING DATA FROM PROGRAM MEMORY USING PROGRAM SPACE VISIBILITY

The upper 32 Kbytes of data space may optionally be mapped into an 8K word page (in PIC24F08KA1XX devices) and a 16K word page (in PIC24F16KA1XX devices) of the program space. This provides transparent access of stored constant data from the data space without the need to use special instructions (i.e., TBLRDL/H).

Program space access through the data space occurs if the MSb of the data space EA is '1', and PSV is enabled by setting the PSV bit in the CPU Control (CORCON<2>) register. The location of the program memory space to be mapped into the data space is determined by the Program Space Visibility Page Address register (PSVPAG). This 8-bit register defines any one of 256 possible pages of 16K words in program space. In effect, PSVPAG functions as the upper 8 bits of the program memory address, with the 15 bits of the EA functioning as the lower bits.

By incrementing the PC by 2 for each program memory word, the lower 15 bits of data space addresses directly map to the lower 15 bits in the corresponding program space addresses.

Data reads from this area add an additional cycle to the instruction being executed, since two program memory fetches are required.

Although each data space address, 8000h and higher, maps directly into a corresponding program memory address (see Figure 4-7), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits of any program space locations used as data should be programmed with '1111 1111' or '0000 0000' to force a NOP. This prevents possible issues should the area of code ever be accidentally executed.

| **Note:** | PSV access is temporarily disabled during table reads/writes. |
|---|---|

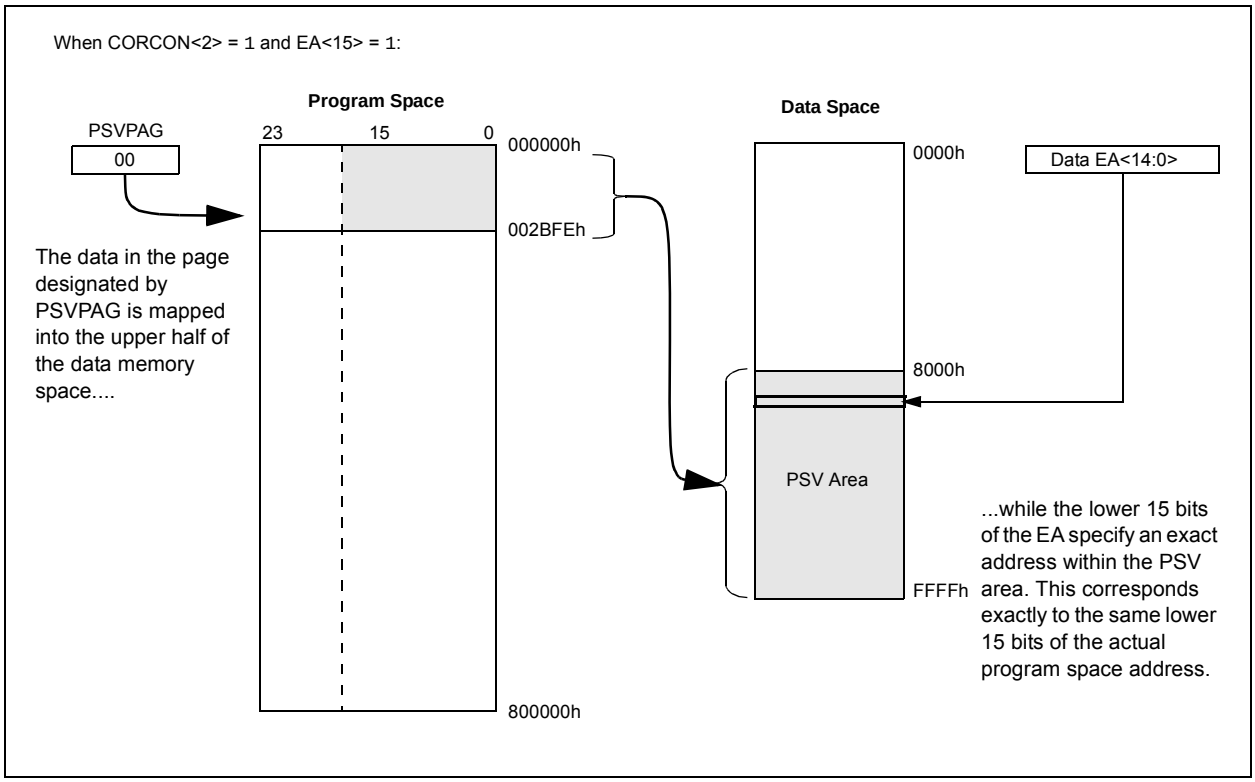
For operations that use PSV and are executed outside a REPEAT loop, the MOV and MOV.D instructions will require one instruction cycle in addition to the specified execution time. All other instructions will require two instruction cycles in addition to the specified execution time.

For operations that use PSV, which are executed inside a REPEAT loop, there will be some instances that require two instruction cycles in addition to the specified execution time of the instruction:

- Execution in the first iteration
- Execution in the last iteration
- Execution prior to exiting the loop due to an interrupt
- Execution upon re-entering the loop after an interrupt is serviced

Any other iteration of the REPEAT loop will allow the instruction accessing data, using PSV, to execute in a single cycle.

**FIGURE 4-7: PROGRAM SPACE VISIBILITY OPERATION**



**NOTES:**

## 5.0 FLASH PROGRAM MEMORY

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Flash programming, refer to the *"PIC24F Family Reference Manual"*, **Section 4. "Program Memory"** (DS39715).

The PIC24F04KA201 family of devices contains internal Flash program memory for storing and executing application code. The memory is readable, writable and erasable when operating with VDD over 1.8V.

Flash memory can be programmed in three ways:

- In-Circuit Serial Programming™ (ICSP™)
- Run-Time Self Programming (RTSP)
- Enhanced In-Circuit Serial Programming (Enhanced ICSP)

ICSP allows a PIC24FXXXX device to be serially programmed while in the end application circuit. This is simply done with two lines for the programming clock and programming data (which are named PGCx and PGDx, respectively), and three other lines for power (VDD), ground (VSS) and Master Clear/Program Mode Entry Voltage (MCLR/VPP). 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 custom firmware to be programmed.

Real-Time Streaming Protocol (RTSP) is accomplished using TBLRD (table read) and TBLWT (table write) instructions. With RTSP, the user may write program memory data in blocks of 32 instructions (96 bytes) at a time, and erase program memory in blocks of 32, 64 and 128 instructions (96,192 and 384 bytes) at a time.

The NVMOP<1:0> (NVMCON<1:0>) bits decide the erase block size.

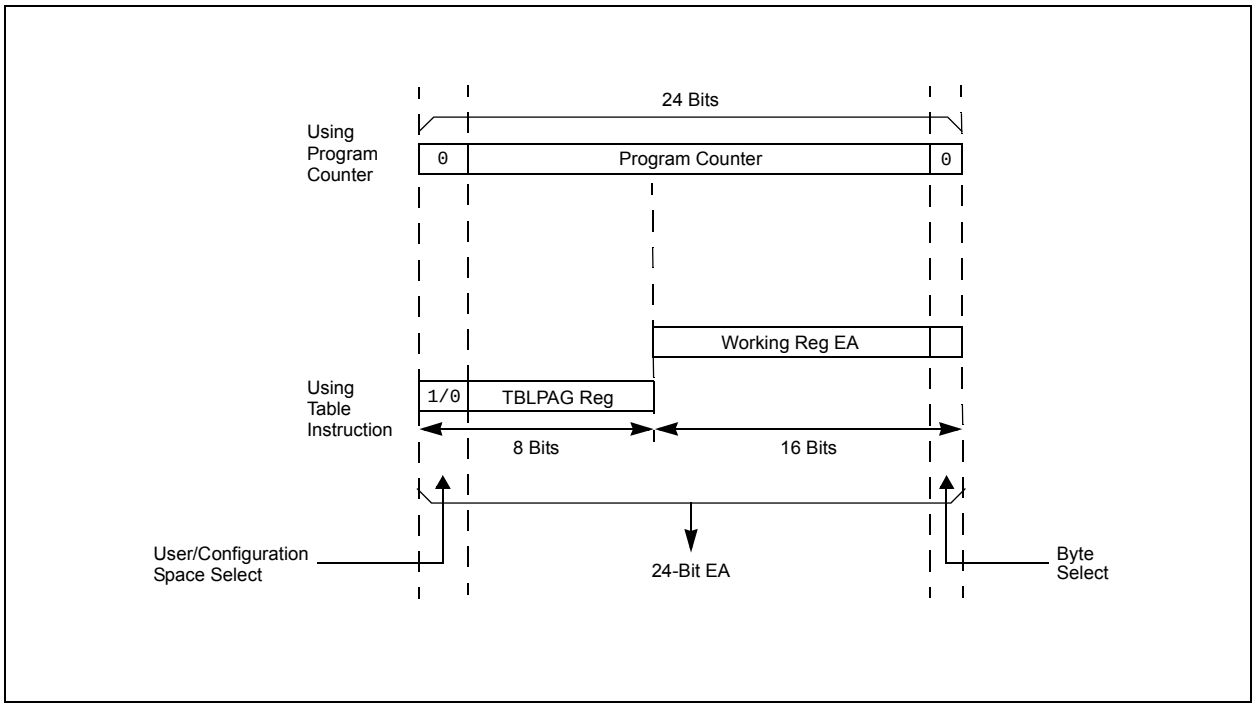
### 5.1 Table Instructions and Flash Programming

Regardless of the method used, Flash memory programming is done with the table read and write instructions. These allow direct read and write access to the program memory space from the data memory while the device is in normal operating mode. The 24-bit target address in the program memory is formed using the TBLPAG<7:0> bits and the Effective Address (EA) from a W register, specified in the table instruction, as depicted in Figure 5-1.

The TBLRDL and the TBLWTL instructions are used to read or write to bits<15:0> of program memory. TBLRDL and TBLWTL can access program memory in both Word and Byte modes.

The TBLRDH and TBLWTH instructions are used to read or write to bits<23:16> of program memory. TBLRDH and TBLWTH can also access program memory in Word or Byte mode.

**FIGURE 5-1: ADDRESSING FOR TABLE REGISTERS**



## 5.2 RTSP Operation

The PIC24F Flash program memory array is organized into rows of 32 instructions or 96 bytes. RTSP allows the user to erase blocks of 1 row, 2 rows and 4 rows (32, 64 and 128 instructions) at a time and to program one row at a time. It is also possible to program single words.

The 1-row (96 bytes), 2-row (192 bytes) and 4-row (384 bytes) erase blocks and single row write block (96 bytes) are edge-aligned, from the beginning of program memory.

When data is written to program memory using TBLWT instructions, the data is not written directly to memory. Instead, data written using table writes is stored in holding latches until the programming sequence is executed.

Any number of TBLWT instructions can be executed and a write will be successfully performed. However, 32 TBLWT instructions are required to write the full row of memory.

The basic sequence for RTSP programming is to set up a Table Pointer, then do a series of TBLWT instructions to load the buffers. Programming is performed by setting the control bits in the NVMCON register.

Data can be loaded in any order and the holding registers can be written to multiple times before performing a write operation. Subsequent writes, however, will wipe out any previous writes.

| **Note:** | Writing to a location multiple times without erasing it is not recommended. |
|---|---|

All of the table write operations are single-word writes (two instruction cycles), because only the buffers are written. A programming cycle is required for programming each row.

## 5.3 Enhanced In-Circuit Serial Programming

Enhanced ICSP uses an on-board bootloader, known as the program executive, to manage the programming process. Using an SPI data frame format, the program executive can erase, program and verify program memory. For more information on Enhanced ICSP, see the device programming specification.

## 5.4 Control Registers

There are two SFRs used to read and write the program Flash memory: NVMCON and NVMKEY.

The NVMCON register (Register 5-1) controls the blocks that need to be erased, which memory type is to be programmed and when the programming cycle starts.

NVMKEY is a write-only register that is used for write protection. To start a programming or erase sequence, the user must consecutively write 55h and AAh to the NVMKEY register. Refer to **Section 5.5 “Programming Operations”** for further details.

## 5.5 Programming Operations

A complete programming sequence is necessary for programming or erasing the internal Flash in RTSP mode. During a programming or erase operation, the processor stalls (waits) until the operation is finished. Setting the WR bit (NVMCON<15>) starts the operation and the WR bit is automatically cleared when the operation is finished.

**REGISTER 5-1: NVMCON: FLASH MEMORY CONTROL REGISTER**

| R/SO-0, HC | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| WR | WREN | WRERR | PGMONLY | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | ERASE | NVMOP5(1) | NVMOP4(1) | NVMOP3(1) | NVMOP2(1) | NVMOP1(1) | NVMOP0(1) |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **WR:** Write Control bit
- 1 = Initiates a Flash memory program or erase operation. The operation is self-timed and the bit is cleared by hardware once the operation is complete
- 0 = Program or erase operation is complete and inactive

bit 14 **WREN:** Write Enable bit
- 1 = Enable Flash program/erase operations
- 0 = Inhibit Flash program/erase operations

bit 13 **WRERR:** Write Sequence Error Flag bit
- 1 = An improper program or erase sequence attempt or termination has occurred (bit is set automatically on any set attempt of the WR bit)
- 0 = The program or erase operation completed normally

bit 12 **PGMONLY:** Program Only Enable bit

bit 11-7 **Unimplemented:** Read as '0'

bit 6 **ERASE:** Erase/Program Enable bit
- 1 = Perform the erase operation specified by NVMOP<5:0> on the next WR command
- 0 = Perform the program operation specified by NVMOP<5:0> on the next WR command

bit 5-0 **NVMOP<5:0>:** Programming Operation Command Byte bits(1)

Erase Operations (when ERASE bit is '1'):
- 1010xx = Erase entire boot block (including code-protected boot block)(2)
- 1001xx = Erase entire memory (including boot block, configuration block, general block)(2)
- 011010 = Program/erase 4 rows of Flash memory(3)
- 011001 = Program/erase 2 rows of Flash memory(3)
- 011000 = Program/erase 1 row of Flash memory(3)
- 0101xx = Erase entire configuration block (except code protection bits)
- 0011xx = Erase entire general memory block programming operations

**Note 1:** All other combinations of NVMOP<5:0> are no operation.

**2:** Available in ICSP™ mode only. Refer to device programming specification.

**3:** The address in the Table Pointer decides which rows will be erased.

### 5.5.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

The user can program one row of Flash program memory at a time by erasing the programmable row. The general process is:

1. Read a row of program memory (32 instructions) and store in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase a row (see Example 5-1):
   a) Set the NVMOP bits (NVMCON<5:0>) to '011000' to configure for row erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
   b) Write the starting address of the block to be erased into the TBLPAG and W registers.
   c) Write 55h to NVMKEY.
   d) Write AAh to NVMKEY.
   e) Set the WR bit (NVMCON<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 32 instructions from data RAM into the program memory buffers (see Example 5-1).
5. Write the program block to Flash memory:
   a) Set the NVMOP bits to '011000' to configure for row programming. Clear the ERASE bit and set the WREN bit.
   b) Write 55h to NVMKEY.
   c) Write AAh to NVMKEY.
   d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.

For protection against accidental operations, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs, as displayed in Example 5-5.

**EXAMPLE 5-1: ERASING A PROGRAM MEMORY ROW – ASSEMBLY LANGUAGE CODE**

```
; Set up NVMCON for row erase operation
        MOV     #0x4058, W0                 ;
        MOV     W0, NVMCON                  ; Initialize NVMCON
; Init pointer to row to be ERASED
        MOV     #tblpage(PROG_ADDR), W0     ;
        MOV     W0, TBLPAG                  ; Initialize PM Page Boundary SFR
        MOV     #tbloffset(PROG_ADDR), W0   ; Initialize in-page EA[15:0] pointer
        TBLWTL  W0, [W0]                    ; Set base address of erase block
        DISI    #5                          ; Block all interrupts
                                              for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY                  ; Write the 55 key
        MOV     #0xAA, W1                   ;
        MOV     W1, NVMKEY                  ; Write the AA key
        BSET    NVMCON, #WR                 ; Start the erase sequence
        NOP                                 ; Insert two NOPs after the erase
        NOP                                 ; command is asserted
```

**EXAMPLE 5-2: ERASING A PROGRAM MEMORY ROW – 'C' LANGUAGE CODE**

```
// C example using MPLAB C30

int __attribute__ ((space(auto_psv))) progAddr = &progAddr;  // Variable located in Pgm Memory
    unsigned int offset;

//Set up pointer to the first memory location to be written

    TBLPAG = __builtin_tblpage(&progAddr);                   // Initialize PM Page Boundary SFR
    offset = &progAddr & 0xFFFF;                             // Initialize lower word of address

    __builtin_tblwtl(offset, 0x0000);                        // Set base address of erase block
                                                             // with dummy latch write

    NVMCON = 0x4058;                                         // Initialize NVMCON

    asm("DISI #5");                                          // Block all interrupts for next 5 instructions
    __builtin_write_NVM();                                   // C30 function to perform unlock
                                                             // sequence and set WR
```

**EXAMPLE 5-3: LOADING THE WRITE BUFFERS – ASSEMBLY LANGUAGE CODE**

```
; Set up NVMCON for row programming operations
        MOV     #0x4004, W0                 ;
        MOV     W0, NVMCON                  ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
        MOV     #0x0000, W0                 ;
        MOV     W0, TBLPAG                  ; Initialize PM Page Boundary SFR
        MOV     #0x6000, W0                 ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
        MOV     #LOW_WORD_0, W2             ;
        MOV     #HIGH_BYTE_0, W3            ;
        TBLWTL  W2, [W0]                    ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                  ; Write PM high byte into program latch
; 1st_program_word
        MOV     #LOW_WORD_1, W2             ;
        MOV     #HIGH_BYTE_1, W3            ;
        TBLWTL  W2, [W0]                    ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                  ; Write PM high byte into program latch
;  2nd_program_word
        MOV     #LOW_WORD_2, W2             ;
        MOV     #HIGH_BYTE_2, W3            ;
        TBLWTL  W2, [W0]                    ; Write PM low word into program latch
        TBLWTH  W3, [W0++]                  ; Write PM high byte into program latch
        •
        •
        •
; 32nd_program_word
        MOV     #LOW_WORD_31, W2            ;
        MOV     #HIGH_BYTE_31, W3           ;
        TBLWTL  W2, [W0]                    ; Write PM low word into program latch
        TBLWTH  W3, [W0]                    ; Write PM high byte into program latch
```

**EXAMPLE 5-4: LOADING THE WRITE BUFFERS – 'C' LANGUAGE CODE**

```
// C example using MPLAB C30

   #define NUM_INSTRUCTION_PER_ROW 64
int __attribute__ ((space(auto_psv))) progAddr = &progAddr; // Variable located in Pgm Memory
   unsigned int offset;
   unsigned int i;
   unsigned int progData[2*NUM_INSTRUCTION_PER_ROW];        // Buffer of data to write

   //Set up NVMCON for row programming
   NVMCON = 0x4001;                                         // Initialize NVMCON

   //Set up pointer to the first memory location to be written
   TBLPAG = __builtin_tblpage(&progAddr);                   // Initialize PM Page Boundary SFR
   offset = &progAddr & 0xFFFF;                             // Initialize lower word of address

   //Perform TBLWT instructions to write necessary number of latches
   for(i=0; i < 2*NUM_INSTRUCTION_PER_ROW; i++)
   {
       __builtin_tblwtl(offset, progData[i++]);             // Write to address low word
       __builtin_tblwth(offset, progData[i]);               // Write to upper byte
       offset = offset + 2;                                 // Increment address
   }
```

**EXAMPLE 5-5: INITIATING A PROGRAMMING SEQUENCE – ASSEMBLY LANGUAGE CODE**

```
        DISI    #5                      ; Block all interrupts
                                          for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY              ; Write the 55 key
        MOV     #0xAA, W1               ;
        MOV     W1, NVMKEY              ; Write the AA key
        BSET    NVMCON, #WR             ; Start the erase sequence
        NOP                             ; 2 NOPs required after setting WR
        NOP                             ;
        BTSC    NVMCON, #15             ; Wait for the sequence to be completed
        BRA     $-2                     ;
```

**EXAMPLE 5-6: INITIATING A PROGRAMMING SEQUENCE – 'C' LANGUAGE CODE**

```
// C example using MPLAB C30

asm("DISI #5");              // Block all interrupts for next 5 instructions

__builtin_write_NVM();       // Perform unlock sequence and set WR
```

**EXAMPLE 5-7: PROGRAMMING A SINGLE WORD OF FLASH PROGRAM MEMORY**

```
; Setup a pointer to data Program Memory
MOV     #tblpage(PROG_ADDR), W0         ;
MOV     W0, TBLPAG                      ;Initialize PM Page Boundary SFR
MOV     #tbloffset(PROG_ADDR), W0       ;Initialize a register with program memory address
MOV     #LOW_WORD_N, W2                 ;
MOV     #HIGH_BYTE_N, W3                ;
TBLWTL  W2, [W0]                        ; Write PM low word into program latch
TBLWTH  W3, [W0++]                      ; Write PM high byte into program latch
; Setup NVMCON for programming one word to data Program Memory
MOV     #0x4003, W0                     ;
MOV     W0, NVMCON                      ; Set NVMOP bits to 0011
DISI    #5                              ; Disable interrupts while the KEY sequence is written
MOV     #0x55, W0                       ; Write the key sequence
MOV     W0, NVMKEY
MOV     #0xAA, W0
MOV     W0, NVMKEY
BSET    NVMCON, #WR                     ; Start the write cycle
```

**NOTES:**

## 6.0 RESETS

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Resets, refer to the *"PIC24F Family Reference Manual"*, **Section 40. "Reset with Programmable Brown-out Reset"** (DS39728). |

The Reset module combines all Reset sources and controls the device Master Reset Signal, SYSRST. The following is a list of device Reset sources:

- POR: Power-on Reset
- MCLR: Pin Reset
- SWR: RESET Instruction
- WDTR: Watchdog Timer Reset
- BOR: Brown-out Reset
- Low-Power BOR/Deep Sleep BOR
- TRAPR: Trap Conflict Reset
- IOPUWR: Illegal Opcode Reset
- UWR: Uninitialized W Register Reset

Figure 6-1 displays a simplified block diagram of the Reset module.

Any active source of Reset will make the SYSRST signal active. Many registers associated with the CPU and peripherals are forced to a known Reset state. Most registers are unaffected by a Reset; their status is unknown on Power-on Reset (POR) and unchanged by all other Resets.

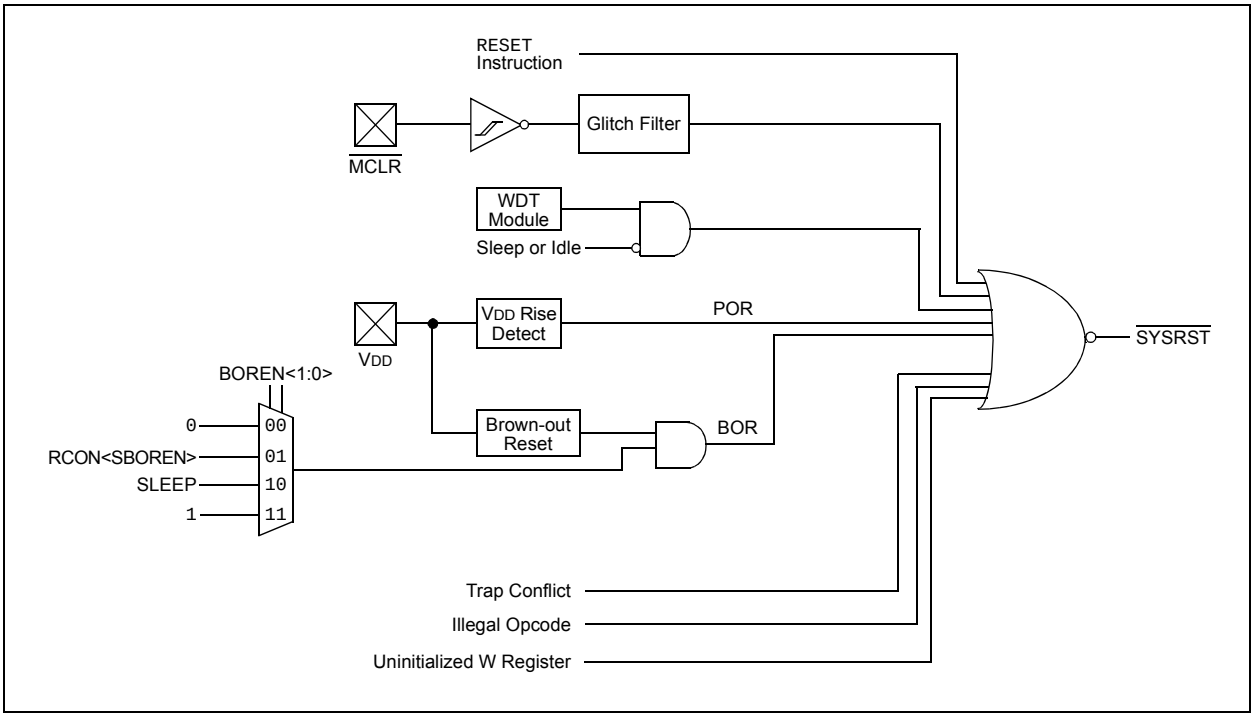
| | |
|---|---|
| **Note:** | Refer to the specific peripheral or CPU section of this manual for register Reset states. |

All types of device Reset will set a corresponding status bit in the RCON register to indicate the type of Reset (see Register 6-1). A POR will clear all bits except for the BOR and POR bits (RCON<1:0>) which are set. The user may set or clear any bit at any time during code execution. The RCON bits only serve as status bits. Setting a particular Reset status bit in software will not cause a device Reset to occur.

The RCON register also has other bits associated with the Watchdog Timer (WDT) and device power-saving states. The function of these bits is discussed in other sections of this manual.

| | |
|---|---|
| **Note:** | The status bits in the RCON register should be cleared after they are read so that the next RCON register value after a device Reset will be meaningful. |

**FIGURE 6-1: RESET SYSTEM BLOCK DIAGRAM**



**REGISTER 6-1: RCON: RESET CONTROL REGISTER(1)**

| R/W-0, HS | R/W-0, HS | R/W-0 | U-0 | U-0 | R/C-0, HS | U-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| TRAPR | IOPUWR | SBOREN | — | — | DPSLP | — | PMSLP |
| bit 15 | | | | | | | bit 8 |

| R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-1, HS | R/W-1, HS |
|---|---|---|---|---|---|---|---|
| EXTR | SWR | SWDTEN(2) | WDTO | SLEEP | IDLE | BOR | POR |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **TRAPR:** Trap Reset Flag bit
1 = A Trap Conflict Reset has occurred
0 = A Trap Conflict Reset has not occurred

bit 14 **IOPUWR:** Illegal Opcode or Uninitialized W Access Reset Flag bit
1 = An illegal opcode detection, an illegal address mode or uninitialized W register used as an Address Pointer caused a Reset
0 = An illegal opcode or uninitialized W Reset has not occurred

bit 13 **SBOREN:** Software Enable/Disable of BOR bit
1 = BOR is turned on in software
0 = BOR is turned off in software

bit 12-11 **Unimplemented:** Read as '0'

bit 10 **DPSLP:** Deep Sleep Mode Flag bit
1 = Deep Sleep has occurred
0 = Deep Sleep has not occurred

bit 9 **Unimplemented:** Read as '0'

bit 8 **PMSLP:** Program Memory Power During Sleep/Idle bit
1 = Program memory bias voltage remains powered during Sleep/Idle
0 = Program memory bias voltage is powered down during Sleep/Idle

bit 7 **EXTR:** External Reset ($\overline{MCLR}$) Pin bit
1 = A Master Clear (pin) Reset has occurred
0 = A Master Clear (pin) Reset has not occurred

bit 6 **SWR:** Software Reset (Instruction) Flag bit
1 = A RESET instruction has been executed
0 = A RESET instruction has not been executed

bit 5 **SWDTEN:** Software Enable/Disable of WDT bit(2)
1 = WDT is enabled
0 = WDT is disabled

bit 4 **WDTO:** Watchdog Timer Time-out Flag bit
1 = WDT time-out has occurred
0 = WDT time-out has not occurred

bit 3 **SLEEP:** Wake-up from Sleep Flag bit
1 = Device has been in Sleep mode
0 = Device has not been in Sleep mode

bit 2 **IDLE:** Wake-up from Idle Flag bit
1 = Device has been in Idle mode
0 = Device has not been in Idle mode

bit 1 **BOR:** Brown-out Reset Flag bit
1 = A Brown-out Reset has occurred (the BOR is also set after a POR)
0 = A Brown-out Reset has not occurred

**REGISTER 6-1: RCON: RESET CONTROL REGISTER(1) (CONTINUED)**

bit 0 **POR:** Power-on Reset Flag bit
1 = A Power-up Reset has occurred
0 = A Power-up Reset has not occurred

**Note 1:** All of the Reset status bits may be set or cleared in software. Setting one of these bits in software does not cause a device Reset.

**2:** If the FWDTEN Configuration bit is '1' (unprogrammed), the WDT is always enabled, regardless of the SWDTEN bit setting.

**TABLE 6-1: RESET FLAG BIT OPERATION**

| Flag Bit | Setting Event | Clearing Event |
|---|---|---|
| TRAPR (RCON<15>) | Trap Conflict Event | POR |
| IOPUWR (RCON<14>) | Illegal Opcode or Uninitialized W Register Access | POR |
| CM (RCON<9>) | Configuration Mismatch Reset | POR |
| EXTR (RCON<7>) | $\overline{\text{MCLR}}$ Reset | POR |
| SWR (RCON<6>) | RESET Instruction | POR |
| WDTO (RCON<4>) | WDT Time-out | PWRSAV Instruction, POR |
| SLEEP (RCON<3>) | PWRSAV #SLEEP Instruction | POR |
| IDLE (RCON<2>) | PWRSAV #IDLE Instruction | POR |
| BOR (RCON<1>) | POR, BOR | — |
| POR (RCON<0>) | POR | — |
| DPSLP (RCON<10>) | PWRSAV #SLEEP instruction with DSCON <DSEN> set | POR |

**Note:** All Reset flag bits may be set or cleared by the user software.

## 6.1 Clock Source Selection at Reset

If clock switching is enabled, the system clock source at device Reset is chosen as shown in Table 6-2. If clock switching is disabled, the system clock source is always selected according to the oscillator Configuration bits. Refer to **Section 8.0 "Oscillator Configuration"** for further details.

**TABLE 6-2: OSCILLATOR SELECTION vs. TYPE OF RESET (CLOCK SWITCHING ENABLED)**

| Reset Type | Clock Source Determinant |
|---|---|
| POR | FNOSC Configuration bits (FNOSC<10:8>) |
| BOR | |
| $\overline{\text{MCLR}}$ | COSC Control bits (OSCCON<14:12>) |
| WDTO | |
| SWR | |

## 6.2 Device Reset Times

The Reset times for various types of device Reset are summarized in Table 6-3. Note that the system Reset signal, $\overline{SYSRST}$, is released after the POR and PWRT delay times expire.

The time at which the device actually begins to execute code will also depend on the system oscillator delays, which include the Oscillator Start-up Timer (OST) and the PLL lock time. The OST and PLL lock times occur in parallel with the applicable $\overline{SYSRST}$ delay times.

The FSCM delay determines the time at which the FSCM begins to monitor the system clock source after the $\overline{SYSRST}$ signal is released.

**TABLE 6-3: RESET DELAY TIMES FOR VARIOUS DEVICE RESETS**

| Reset Type | Clock Source | $\overline{SYSRST}$ Delay | System Clock Delay | Notes |
|---|---|---|---|---|
| POR(6) | EC | TPOR + TPWRT | — | 1, 2 |
| | FRC, FRCDIV | TPOR + TPWRT | TFRC | 1, 2, 3 |
| | LPRC | TPOR + TPWRT | TLPRC | 1, 2, 3 |
| | ECPLL | TPOR + TPWRT | TLOCK | 1, 2, 4 |
| | FRCPLL | TPOR + TPWRT | TFRC + TLOCK | 1, 2, 3, 4 |
| | XT, HS, SOSC | TPOR+ TPWRT | TOST | 1, 2, 5 |
| | XTPLL, HSPLL | TPOR + TPWRT | TOST + TLOCK | 1, 2, 4, 5 |
| BOR | EC | TPWRT | — | 2 |
| | FRC, FRCDIV | TPWRT | TFRC | 2, 3 |
| | LPRC | TPWRT | TLPRC | 2, 3 |
| | ECPLL | TPWRT | TLOCK | 2, 4 |
| | FRCPLL | TPWRT | TFRC + TLOCK | 2, 3, 4 |
| | XT, HS, SOSC | TPWRT | TOST | 2, 5 |
| | XTPLL, HSPLL | TPWRT | TFRC + TLOCK | 2, 3, 4 |
| All Others | Any Clock | — | — | None |

**Note 1:** TPOR = Power-on Reset delay.

**2:** TPWRT = 64 ms nominal if the Power-up Timer is enabled; otherwise, it is zero.

**3:** TFRC and TLPRC = RC Oscillator start-up times.

**4:** TLOCK = PLL lock time.

**5:** TOST = Oscillator Start-up Timer (OST). A 10-bit counter waits 1024 oscillator periods before releasing oscillator clock to the system.

**6:** If Two-Speed Start-up is enabled, regardless of the Primary Oscillator selected, the device starts with FRC, and in such cases, FRC start-up time is valid.

**Note:** For detailed operating frequency and timing specifications, see **Section 26.0 "Electrical Characteristics"**.

## 6.3 Brown-out Reset (BOR)

The PIC24F04KA201 family devices implement a BOR circuit, which provides the user several configuration and power-saving options. The BOR is controlled by the <BORV1:BORV0> and (BOREN<1:0>) Configuration bits (FPOR<6:5,1:0>). There are a total of four BOR configurations, which are provided in Table 6.3.1.

The BOR threshold is set by the BORV<1:0> bits. If BOR is enabled (any values of BOREN<1:0>, except '00'), any drop of VDD below the set threshold point will reset the device. The chip will remain in BOR until VDD rises above threshold.

If the Power-up Timer is enabled, it will be invoked after VDD rises above the threshold; it, then, will keep the chip in Reset for an additional time delay, TPWRT, if VDD drops below the threshold while the power-up timer is running. The chip goes back into a BOR and the Power-up Timer will be initialized. Once VDD rises above the threshold, the Power-up Timer will execute the additional time delay.

BOR and the Power-up Timer are independently configured. Enabling the BOR Reset does not automatically enable the PWRT.

### 6.3.1 SOFTWARE ENABLED BOR

When BOREN<1:0> = 01, the BOR can be enabled or disabled by the user in software. This is done with the control bit, SBOREN (RCON<13>). Setting SBOREN enables the BOR to function as previously described. Clearing the SBOREN disables the BOR entirely. The SBOREN bit operates only in this mode; otherwise, it is read as '0'.

Placing BOR under software control gives the user the additional flexibility of tailoring the application to its environment without having to reprogram the device to change the BOR configuration. It also allows the user to tailor the incremental current that the BOR consumes. While the BOR current is typically very small, it may have some impact in low-power applications.

| | |
|---|---|
| **Note:** | Even when the BOR is under software control, the BOR Reset voltage level is still set by the BORV1:BORV0 Configuration bits. It can not be changed in software. |

### 6.3.2 DETECTING BOR

When BOR is enabled, the BOR bit (RCON<1>) is always reset to '1' on any BOR or POR event. This makes it difficult to determine if a BOR event has occurred just by reading the state of BOR alone. A more reliable method is to simultaneously check the state of both POR and BOR. This assumes that the POR and BOR bits are reset to '0' in the software immediately after any POR event. If the BOR bit is '1' while POR is '0', it can be reliably assumed that a BOR event has occurred.

| | |
|---|---|
| **Note:** | Even when the device exits from Deep Sleep mode, both the POR and BOR are set. |

### 6.3.3 DISABLING BOR IN SLEEP MODE

When BOREN<1:0> = 10, BOR remains under hardware control and operates as previously described. However, whenever the device enters Sleep mode, BOR is automatically disabled. When the device returns to any other operating mode, BOR is automatically re-enabled.

This mode allows for applications to recover from brown-out situations, while actively executing code, when the device requires BOR protection the most. At the same time, it saves additional power in Sleep mode by eliminating the small incremental BOR current.

### 6.3.4 DEEP SLEEP BOR (DSBOR)

Deep Sleep BOR is a very low-power BOR circuitry. Due to low current consumption, accuracy may vary. DSBOR occurs anywhere between 1.55V and 1.95V.

DSBOR is selected in configuration through the BORV<1:0> (FPOR<6:5>) bits = 00.

DSBOR re-arms the POR anywhere between 1.55V and 1.95V; however, below 1.55V, the POR is asserted.

### 6.3.5 POR AND LONG OSCILLATOR START-UP TIMES

The oscillator start-up circuitry and its associated delay timers are not linked to the device Reset delays that occur at power-up. Some crystal circuits (especially low-frequency crystals) will have a relatively long start-up time. Therefore, one or more of the following conditions is possible after $\overline{\text{SYSRST}}$ is released:

- The oscillator circuit has not begun to oscillate.
- The Oscillator Start-up Timer has not expired (if a crystal oscillator is used).
- The PLL has not achieved a lock (if PLL is used).

The device will not begin to execute code until a valid clock source has been released to the system. Therefore, the oscillator and PLL start-up delays must be considered when the Reset delay time must be known.

### 6.3.6 FAIL-SAFE CLOCK MONITOR (FSCM) AND DEVICE RESETS

If the FSCM is enabled, it will begin to monitor the system clock source when $\overline{\text{SYSRST}}$ is released. If a valid clock source is not available at this time, the device will automatically switch to the FRC Oscillator and the user can switch to the desired crystal oscillator in the Trap Service Routine (TSR).

## 6.4 Special Function Register Reset States

Most of the Special Function Registers (SFRs) associated with the PIC24F CPU and peripherals are reset to a particular value at a device Reset. The SFRs are grouped by their peripheral or CPU function and their Reset values are specified in each section of this manual.

The Reset value for each SFR does not depend on the type of Reset with the exception of four registers. The Reset value for the Reset Control register, RCON, will depend on the type of device Reset. The Reset value for the Oscillator Control register, OSCCON, will depend on the type of Reset and the programmed values of the FNOSC bits in Flash Configuration Word (FOSCSEL); see Table 6-2. The RCFGCAL and NVMCON registers are only affected by a POR.

## 6.5 Deep Sleep BOR (DSBOR)

Deep Sleep BOR is a very low-power BOR circuitry, used when the device is in Deep Sleep mode. Due to low-current consumption, accuracy may vary.

The DSBOR trip point is around 2.0V. DSBOR is enabled by configuring FDS <DSLPBOR> = 1. DSLPBOR will re-arm the POR to ensure the device will reset if $V_{DD}$ drops below the POR threshold.

## 7.0 INTERRUPT CONTROLLER

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Interrupt Controller, refer to the *"PIC24F Family Reference Manual"*, **Section 8. "Interrupts"** (DS39707). |

The PIC24F interrupt controller reduces the numerous peripheral interrupt request signals to a single interrupt request signal to the CPU. It has the following features:

- Up to eight processor exceptions and software traps
- Seven user-selectable priority levels
- Interrupt Vector Table (IVT) with up to 118 vectors
- Unique vector for each interrupt or exception source
- Fixed priority within a specified user priority level
- Fixed interrupt entry and return latencies

### 7.1 Interrupt Vector (IVT) Table

The IVT is displayed in Figure 7-1. The IVT resides in the program memory, starting at location 000004h. The IVT contains 126 vectors, consisting of eight non-maskable trap vectors, plus, up to 118 sources of interrupt. In general, each interrupt source has its own vector.

Each interrupt vector contains a 24-bit wide address. The value programmed into each interrupt vector location is the starting address of the associated Interrupt Service Routine (ISR).

Interrupt vectors are prioritized in terms of their natural priority; this is linked to their position in the vector table. All other things being equal, lower addresses have a higher natural priority. For example, the interrupt associated with vector 0 will take priority over interrupts at any other vector address.

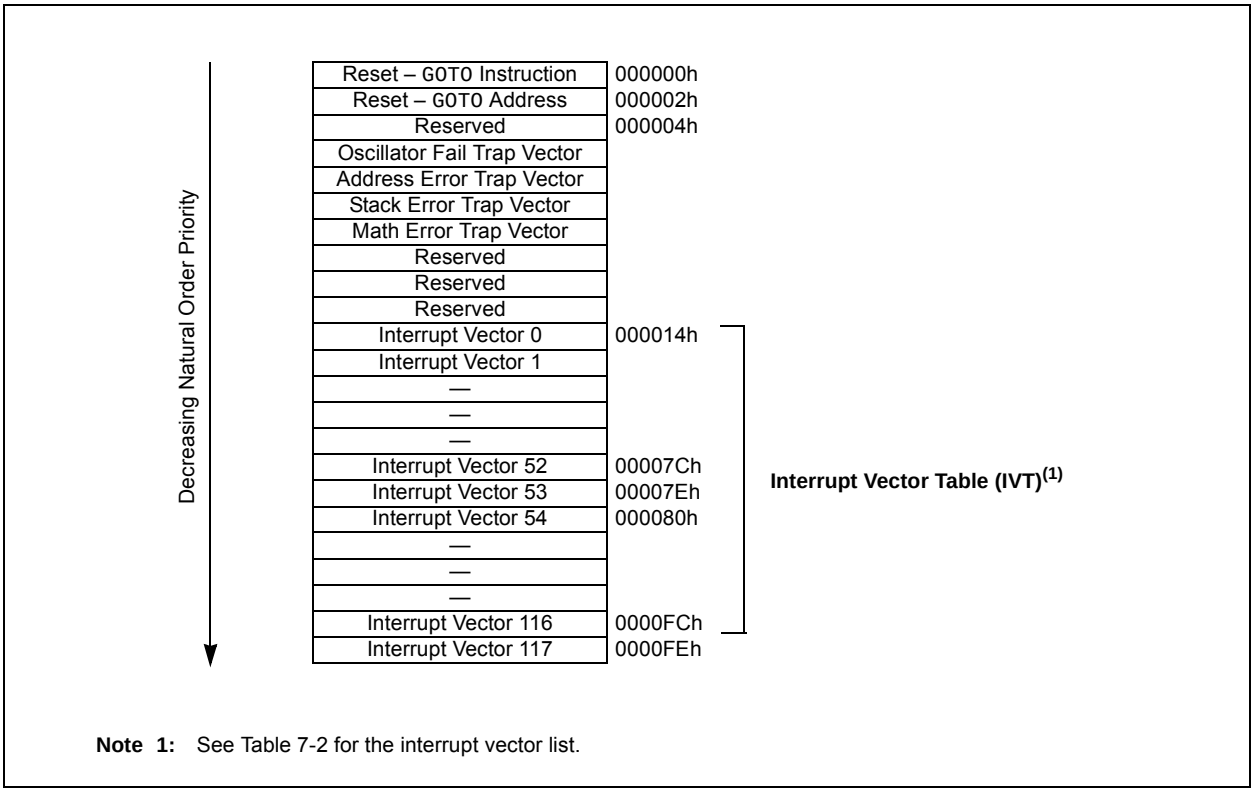
PIC24F04KA201 family devices implement non-maskable traps and unique interrupts; these are summarized in Table 7-1 and Table 7-2.

### 7.2 Reset Sequence

A device Reset is not a true exception because the interrupt controller is not involved in the Reset process. The PIC24F devices clear their registers in response to a Reset, which forces the Program Counter (PC) to zero. The microcontroller then begins program execution at location 000000h. The user programs a GOTO instruction at the Reset address, which redirects the program execution to the appropriate start-up routine.

| | |
|---|---|
| **Note:** | Any unimplemented or unused vector locations in the IVT should be programmed with the address of a default interrupt handler routine that contains a RESET instruction. |

**FIGURE 7-1: PIC24F INTERRUPT VECTOR TABLE**




**Note 1:** See Table 7-2 for the interrupt vector list.

**TABLE 7-1: TRAP VECTOR DETAILS**

| Vector Number | IVT Address | AIVT Address | Trap Source |
|---|---|---|---|
| 0 | 000004h | 000104h | Reserved |
| 1 | 000006h | 000106h | Oscillator Failure |
| 2 | 000008h | 000108h | Address Error |
| 3 | 00000Ah | 00010Ah | Stack Error |
| 4 | 00000Ch | 00010Ch | Math Error |
| 5 | 00000Eh | 00010Eh | Reserved |
| 6 | 000010h | 000110h | Reserved |
| 7 | 000012h | 000112h | Reserved |

**TABLE 7-2: IMPLEMENTED INTERRUPT VECTORS**

| Interrupt Source | Vector Number | IVT Address | AIVT Address | Interrupt Bit Locations | | |
|---|---|---|---|---|---|---|
| | | | | Flag | Enable | Priority |
| ADC1 Conversion Done | 13 | 00002Eh | 00012Eh | IFS0<13> | IEC0<13> | IPC3<6:4> |
| Comparator Event | 18 | 000038h | 000138h | IFS1<2> | IEC1<2> | IPC4<10:8> |
| CTMU | 77 | 0000AEh | 0001AEh | IFS4<13> | IEC4<13> | IPC19<6:4> |
| External Interrupt 0 | 0 | 000014h | 000114h | IFS0<0> | IEC0<0> | IPC0<2:0> |
| External Interrupt 1 | 20 | 00003Ch | 00013Ch | IFS1<4> | IEC1<4> | IPC5<2:0> |
| External Interrupt 2 | 29 | 00004Eh | 00014Eh | IFS1<13> | IEC1<13> | IPC7<6:4> |
| I2C1 Master Event | 17 | 000036h | 000136h | IFS1<1> | IEC1<1> | IPC4<6:4> |
| I2C1 Slave Event | 16 | 000034h | 000134h | IFS1<0> | IEC1<0> | IPC4<2:0> |
| Input Capture1 | 1 | 000016h | 000116h | IFS0<1> | IEC0<1> | IPC0<6:4> |
| Input Change Notification | 19 | 00003Ah | 00013Ah | IFS1<3> | IEC1<3> | IPC4<14:12> |
| HLVD High/Low-Voltage Detect | 72 | 0000A4h | 0001A4h | IFS4<8> | IEC4<8> | IPC17<2:0> |
| NVM – NVM Write Complete | 15 | 000032h | 000132h | IFS0<15> | IEC0<15> | IPC3<14:12> |
| Output Compare 1 | 2 | 000018h | 000118h | IFS0<2> | IEC0<2> | IPC0<10:8> |
| SPI1 Error | 9 | 000026h | 000126h | IFS0<9> | IEC0<9> | IPC2<6:4> |
| SPI1 Event | 10 | 000028h | 000128h | IFS0<10> | IEC0<10> | IPC2<10:8> |
| Timer1 | 3 | 00001Ah | 00011Ah | IFS0<3> | IEC0<3> | IPC0<14:12> |
| Timer2 | 7 | 000022h | 000122h | IFS0<7> | IEC0<7> | IPC1<14:12> |
| Timer3 | 8 | 000024h | 000124h | IFS0<8> | IEC0<8> | IPC2<2:0> |
| UART1 Error | 65 | 000096h | 000196h | IFS4<1> | IEC4<1> | IPC16<6:4> |
| UART1 Receiver | 11 | 00002Ah | 00012Ah | IFS0<11> | IEC0<11> | IPC2<14:12> |
| UART1 Transmitter | 12 | 00002Ch | 00012Ch | IFS0<12> | IEC0<12> | IPC3<2:0> |

## 7.3 Interrupt Control and Status Registers

The PIC24F04KA201 family of devices implements a total of 23 registers for the interrupt controller:

- INTCON1
- INTCON2
- IFS0, IFS1, IFS3 and IFS4
- IEC0, IEC1, IEC3 and IEC4
- IPC0 through IPC5, IPC7 and IPC15 through IPC19
- INTTREG

Global interrupt control functions are controlled from INTCON1 and INTCON2. INTCON1 contains the Interrupt Nesting Disable (NSTDIS) bit, as well as the control and status flags for the processor trap sources. The INTCON2 register controls the external interrupt request signal behavior and the use of the AIV table.

The IFSx registers maintain all of the interrupt request flags. Each source of interrupt has a status bit, which is set by the respective peripherals, or external signal, and is cleared via software.

The IECx registers maintain all of the interrupt enable bits. These control bits are used to individually enable interrupts from the peripherals or external signals.

The IPCx registers are used to set the interrupt priority level for each source of interrupt. Each user interrupt source can be assigned to one of eight priority levels.

The INTTREG register contains the associated interrupt vector number and the new CPU interrupt priority level, which are latched into the Vector Number (VECNUM<6:0>) and the Interrupt Level (ILR<3:0>) bit fields in the INTTREG register. The new interrupt priority level is the priority of the pending interrupt.

The interrupt sources are assigned to the IFSx, IECx and IPCx registers in the same sequence listed in Table 7-2. For example, the INT0 (External Interrupt 0) is depicted as having a vector number and a natural order priority of 0. Thus, the INT0IF status bit is found in IFS0<0>, the INT0IE enable bit in IEC0<0> and the INT0IP<2:0> priority bits in the first position of IPC0 (IPC0<2:0>).

Although they are not specifically part of the interrupt control hardware, two of the CPU control registers contain bits that control interrupt functionality. The ALU STATUS register (SR) contains the IPL<2:0> bits (SR<7:5>). These indicate the current CPU interrupt priority level. The user may change the current CPU priority level by writing to the IPL bits.

The CORCON register contains the IPL3 bit, which together with IPL<2:0>, also indicates the current CPU priority level. IPL3 is a read-only bit so that the trap events cannot be masked by the user's software.

All interrupt registers are described in Register 7-1 through Register 7-18, in the following sections.

**REGISTER 7-1: SR: ALU STATUS REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-0, HSC |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | DC(1) |
| bit 15 | | | | | | | bit 8 |

| R/W-0, HSC | R/W-0, HSC | R/W-0, HSC | R-0, HSC | R/W-0, HSC | R/W-0, HSC | R/W-0, HSC | R/W-0, HSC |
|---|---|---|---|---|---|---|---|
| IPL2(2,3) | IPL1(2,3) | IPL0(2,3) | RA(1) | N(1) | OV(1) | Z(1) | C(1) |
| bit 7 | | | | | | | bit 0 |

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

bit 15-9 **Unimplemented:** Read as '0'

bit 7-5 **IPL<2:0>:** CPU Interrupt Priority Level Status bits(2,3)

111 = CPU interrupt priority level is 7 (15); user interrupts disabled
110 = CPU interrupt priority level is 6 (14)
101 = CPU interrupt priority level is 5 (13)
100 = CPU interrupt priority level is 4 (12)
011 = CPU interrupt priority level is 3 (11)
010 = CPU interrupt priority level is 2 (10)
001 = CPU interrupt priority level is 1 (9)
000 = CPU interrupt priority level is 0 (8)

**Note 1:** See Register 3-1 for the description of these bits, which are not dedicated to interrupt control functions.

**2:** The IPL bits are concatenated with the IPL3 bit (CORCON<3>) to form the CPU interrupt priority level. The value in parentheses indicates the interrupt priority level if IPL3 = 1.

**3:** The IPL Status bits are read-only when NSTDIS (INTCON1<15>) = 1.

**Note:** Bit 8 and bits 4 through 0 are described in **Section 3.0 "CPU"**.

**REGISTER 7-2: CORCON: CPU CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | R/C-0, HSC | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | IPL3(2) | PSV(1) | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-4 **Unimplemented:** Read as '0'

bit 3 **IPL3:** CPU Interrupt Priority Level Status bit(2)
1 = CPU interrupt priority level is greater than 7
0 = CPU interrupt priority level is 7 or less

bit 1-0 **Unimplemented:** Read as '0'

**Note 1:** See Register 3-1 for the description of this bit, which is not dedicated to interrupt control functions.
**2:** The IPL3 bit is concatenated with the IPL<2:0> bits (SR<7:5>) to form the CPU interrupt priority level.

**Note:** Bit 2 is described in **Section 3.0 "CPU"**.

**REGISTER 7-3: INTCON1: INTERRUPT CONTROL REGISTER 1**

| R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| NSTDIS | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | MATHERR | ADDRERR | STKERR | OSCFAIL | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **NSTDIS:** Interrupt Nesting Disable bit
1 = Interrupt nesting is disabled
0 = Interrupt nesting is enabled

bit 14-5 **Unimplemented:** Read as ‘0’

bit 4 **MATHERR:** Arithmetic Error Trap Status bit
1 = Overflow trap has occurred
0 = Overflow trap has not occurred

bit 3 **ADDRERR:** Address Error Trap Status bit
1 = Address error trap has occurred
0 = Address error trap has not occurred

bit 2 **STKERR:** Stack Error Trap Status bit
1 = Stack error trap has occurred
0 = Stack error trap has not occurred

bit 1 **OSCFAIL:** Oscillator Failure Trap Status bit
1 = Oscillator failure trap has occurred
0 = Oscillator failure trap has not occurred

bit 0 **Unimplemented:** Read as ‘0’

**REGISTER 7-4: INTCON2: INTERRUPT CONTROL REGISTER2**

| R/W-0 | R-0, HSC | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| ALTIVT | DISI | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | INT2EP | INT1EP | INT0EP |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ALTIVT:** Enable Alternate Interrupt Vector Table bit
1 = Use Alternate Interrupt Vector Table
0 = Use standard (default) vector table

bit 14 **DISI:** DISI Instruction Status bit
1 = DISI instruction is active
0 = DISI instruction is not active

bit 13-3 **Unimplemented:** Read as '0'

bit 2 **INT2EP:** External Interrupt 2 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 1 **INT1EP:** External Interrupt 1 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 0 **INT0EP:** External Interrupt 0 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

**REGISTER 7-5: IFS0: INTERRUPT FLAG STATUS REGISTER 0**

| R/W-0, HS | U-0 | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS |
|---|---|---|---|---|---|---|---|
| NVMIF | — | AD1IF | U1TXIF | U1RXIF | SPI1IF | SPF1IF | T3IF |
| bit 15 | | | | | | | bit 8 |

| R/W-0, HS | U-0 | U-0 | U-0 | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0, HS |
|---|---|---|---|---|---|---|---|
| T2IF | — | — | — | T1IF | OC1IF | IC1IF | INT0IF |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **NVMIF:** NVM Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 14 **Unimplemented:** Read as '0'

bit 13 **AD1IF:** A/D Conversion Complete Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12 **U1TXIF:** UART1 Transmitter Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 11 **U1RXIF:** UART1 Receiver Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 10 **SPI1IF:** SPI1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 9 **SPF1IF:** SPI1 Fault Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 8 **T3IF:** Timer3 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 7 **T2IF:** Timer2 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 6-4 **Unimplemented:** Read as '0'

bit 3 **T1IF:** Timer1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **OC1IF:** Output Compare Channel 1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **IC1IF:** Input Capture Channel 1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **INT0IF:** External Interrupt 0 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

**REGISTER 7-6: IFS1: INTERRUPT FLAG STATUS REGISTER 1**

| U-0 | U-0 | R/W-0, HS | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | INT2IF | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0, HS | R/W-0, HS | R/W-0, HS | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | INT1IF | CNIF | CMIF | MI2C1IF | SI2C1IF |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **INT2IF:** External Interrupt 2 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-5 **Unimplemented:** Read as '0'

bit 4 **INT1IF:** External Interrupt 1 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 3 **CNIF:** Input Change Notification Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **CMIF:** Comparator Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **MI2C1IF:** Master I2C1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **SI2C1IF:** Slave I2C1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

**REGISTER 7-7: IFS4: INTERRUPT FLAG STATUS REGISTER 4**

| U-0 | U-0 | R/W-0, HS | U-0 | U-0 | U-0 | U-0 | R/W-0, HS |
|---|---|---|---|---|---|---|---|
| — | — | CTMUIF | — | — | — | — | HLVDIF |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0, HS | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | U1ERIF | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **CTMUIF:** CTMU Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-9 **Unimplemented:** Read as '0'

bit 8 **HLVDIF:** High/Low-Voltage Detect Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

bit 1 **U1ERIF:** UART1 Error Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

**REGISTER 7-8: IEC0: INTERRUPT ENABLE CONTROL REGISTER 0**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| NVMIE | — | AD1IE | U1TXIE | U1RXIE | SPI1IE | SPF1IE | T3IE |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| T2IE | — | — | — | T1IE | OC1IE | IC1IE | INT0IE |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **NVMIE:** NVM Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 14 **Unimplemented:** Read as '0'

bit 13 **AD1IE:** A/D Conversion Complete Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12 **U1TXIE:** UART1 Transmitter Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 11 **U1RXIE:** UART1 Receiver Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 10 **SPI1IE:** SPI1 Transfer Complete Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 9 **SPF1IE:** SPI1 Fault Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 8 **T3IE:** Timer3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 7 **T2IE:** Timer2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 6-4 **Unimplemented:** Read as '0'

bit 3 **T1IE:** Timer1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **OC1IE:** Output Compare Channel 1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **IC1IE:** Input Capture Channel 1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **INT0IE:** External Interrupt 0 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

**REGISTER 7-9: IEC1: INTERRUPT ENABLE CONTROL REGISTER 1**

| U-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | INT2IE | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | INT1IE | CNIE | CMIE | MI2C1IE | SI2C1IE |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **INT2IE:** External Interrupt 2 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12-5 **Unimplemented:** Read as '0'

bit 4 **INT1IE:** External Interrupt 1 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 3 **CNIE:** Input Change Notification Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **CMIE:** Comparator Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **MI2C1IE:** Master I2C1 Event Interrupt Enable Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **SI2C1IE:** Slave I2C1 Event Interrupt Enable Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

**REGISTER 7-10: IEC4: INTERRUPT ENABLE CONTROL REGISTER 4**

| U-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | CTMUIE | — | — | — | — | HLVDIE |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | U1ERIE | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as ‘0’

bit 13 **CTMUIE:** CTMU Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12-9 **Unimplemented:** Read as ‘0’

bit 8 **HLVDIE:** High/Low-Voltage Detect Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 7-2 **Unimplemented:** Read as ‘0’

bit 1 **U1ERIE:** UART1 Error Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **Unimplemented:** Read as ‘0’

**REGISTER 7-11: IPC0: INTERRUPT PRIORITY CONTROL REGISTER 0**

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | T1IP2 | T1IP1 | T1IP0 | — | OC1IP2 | OC1IP1 | OC1IP0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | IC1IP2 | IC1IP1 | IC1IP0 | — | INT0IP2 | INT0IP1 | INT0IP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **T1IP<2:0>:** Timer1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **OC1IP<2:0>:** Output Compare Channel 1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **IC1IP<2:0>:** Input Capture Channel 1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **INT0IP<2:0>:** External Interrupt 0 Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-12: IPC1: INTERRUPT PRIORITY CONTROL REGISTER 1**

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | T2IP2 | T2IP1 | T2IP0 | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **T2IP<2:0>:** Timer2 Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11-0 **Unimplemented:** Read as '0'

**REGISTER 7-13: IPC2: INTERRUPT PRIORITY CONTROL REGISTER 2**

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | U1RXIP2 | U1RXIP1 | U1RXIP0 | — | SPI1IP2 | SPI1IP1 | SPI1IP0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | SPF1IP2 | SPF1IP1 | SPF1IP0 | — | T3IP2 | T3IP1 | T3IP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **Unimplemented:** Read as ‘0’

bit 14-12 **U1RXIP<2:0>:** UART1 Receiver Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as ‘0’

bit 10-8 **SPI1IP<2:0>:** SPI1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 7 **Unimplemented:** Read as ‘0’

bit 6-4 **SPF1IP<2:0>:** SPI1 Fault Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **T3IP<2:0>:** Timer3 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-14: IPC3: INTERRUPT PRIORITY CONTROL REGISTER 3**

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | NVMIP2 | NVMIP1 | NVMIP0 | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | AD1IP2 | AD1IP1 | AD1IP0 | — | U1TXIP2 | U1TXIP1 | U1TXIP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **Unimplemented:** Read as ‘0’

bit 14-12 **NVMIP<2:0>:** NVM Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11-7 **Unimplemented:** Read as ‘0’

bit 6-4 **AD1IP<2:0>:** A/D Conversion Complete Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **U1TXIP<2:0>:** UART1 Transmitter Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-15: IPC4: INTERRUPT PRIORITY CONTROL REGISTER 4**

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | CNIP2 | CNIP1 | CNIP0 | — | CMIP2 | CMIP1 | CMIP0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | MI2C1P2 | MI2C1P1 | MI2C1P0 | — | SI2C1P2 | SI2C1P1 | SI2C1P0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **Unimplemented:** Read as ‘0’

bit 14-12 **CNIP<2:0>:** Input Change Notification Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as ‘0’

bit 10-8 **CMIP<2:0>:** Comparator Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 7 **Unimplemented:** Read as ‘0’

bit 6-4 **MI2C1P<2:0>:** Master I2C1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **SI2C1P<2:0>:** Slave I2C1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-16: IPC5: INTERRUPT PRIORITY CONTROL REGISTER 5**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | INT1IP2 | INT1IP1 | INT1IP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-3 **Unimplemented:** Read as ‘0’

bit 2-0 **INT1IP<2:0>:** External Interrupt 1 Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-17: IPC7: INTERRUPT PRIORITY CONTROL REGISTER 7**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | INT2IP2 | INT2IP1 | INT2IP0 | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-7 **Unimplemented:** Read as ‘0’

bit 6-4 **INT2IP<2:0>:** External Interrupt 2 Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3-0 **Unimplemented:** Read as ‘0’

**REGISTER 7-18: IPC16: INTERRUPT PRIORITY CONTROL REGISTER 16**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | U1ERIP2 | U1ERIP1 | U1ERIP0 | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-7 **Unimplemented:** Read as ‘0’

bit 6-4 **U1ERIP<2:0>:** UART1 Error Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3-0 **Unimplemented:** Read as ‘0’

**REGISTER 7-19: IPC18: INTERRUPT PRIORITY CONTROL REGISTER 18**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | HLVDIP2 | HLVDIP1 | HLVDIP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-3 **Unimplemented:** Read as '0'

bit 2-0 **HLVDIP<2:0>:** High/Low-Voltage Detect Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

**REGISTER 7-20: IPC19: INTERRUPT PRIORITY CONTROL REGISTER 19**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | CTMUIP2 | CTMUIP1 | CTMUIP0 | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-7 **Unimplemented:** Read as '0'

bit 6-4 **CTMUIP<2:0>:** CTMU Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

**REGISTER 7-21: INTTREG: INTERRUPT CONTROL AND STATUS REGISTER**

| R-0 | U-0 | R/W-0 | U-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| CPUIRQ | — | VHOLD | — | ILR<3:0> | | | |
| bit 15 | | | | | | | bit 8 |

| U-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| — | VECNUM<6:0> | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **CPUIRQ:** Interrupt Request from Interrupt Controller CPU bit
1 = An interrupt request has occurred but has not yet been Acknowledged by the CPU (this will happen when the CPU priority is higher than the interrupt priority)
0 = No interrupt request is left unacknowledged

bit 14 **Unimplemented:** Read as '0'

bit 13 **VHOLD:** Allows Vector Number Capture and Changes what Interrupt is Stored in VECNUM bit
1 = VECNUM will contain the value of the highest priority pending interrupt, instead of the current interrupt
0 = VECNUM will contain the value of the last Acknowledged interrupt (last interrupt that has occurred with higher priority than the CPU, even if other interrupts are pending)

bit 12 **Unimplemented:** Read as '0'

bit 11-8 **ILR<3:0>:** New CPU Interrupt Priority Level bits
1111 = CPU Interrupt Priority Level is 15
•
•
•
0001 = CPU Interrupt Priority Level is 1
0000 = CPU Interrupt Priority Level is 0

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

bit 6-0 **VECNUM<6:0>:** Vector Number of Pending Interrupt bits
0111111 = Interrupt Vector pending is number 135
•
•
•
0000001 = Interrupt Vector pending is number 9
0000000 = Interrupt Vector pending is number 8

## 7.4 Interrupt Setup Procedures

### 7.4.1 INITIALIZATION

To configure an interrupt source:

1. Set the NSTDIS Control bit (INTCON1<15>) if nested interrupts are not desired.
2. Select the user-assigned priority level for the interrupt source by writing the control bits in the appropriate IPCx register. The priority level will depend on the specific application and type of interrupt source. If multiple priority levels are not desired, the IPCx register control bits for all enabled interrupt sources may be programmed to the same non-zero value.

| **Note:** | At a device Reset, the IPCx registers are initialized, such that all user interrupt sources are assigned to priority level 4. |
|---|---|

3. Clear the interrupt flag status bit associated with the peripheral in the associated IFSx register.
4. Enable the interrupt source by setting the interrupt enable control bit associated with the source in the appropriate IECx register.

### 7.4.2 INTERRUPT SERVICE ROUTINE

The method that is used to declare an ISR and initialize the IVT with the correct vector address depends on the programming language (i.e., C or assembler) and the language development toolsuite that is used to develop the application. In general, the user must clear the interrupt flag in the appropriate IFSx register for the source of the interrupt that the ISR handles. Otherwise, the ISR will be re-entered immediately after exiting the routine. If the ISR is coded in assembly language, it must be terminated using a `RETFIE` instruction to unstack the saved PC value, SRL value and old CPU priority level.

### 7.4.3 TRAP SERVICE ROUTINE (TSR)

A Trap Service Routine (TSR) is coded like an ISR, except that the appropriate trap status flag in the INTCON1 register must be cleared to avoid re-entry into the TSR.

### 7.4.4 INTERRUPT DISABLE

All user interrupts can be disabled using the following procedure:

1. Push the current SR value onto the software stack using the `PUSH` instruction.
2. Force the CPU to priority level 7 by inclusive ORing the value OEh with SRL.

To enable user interrupts, the `POP` instruction may be used to restore the previous SR value.

Only user interrupts with a priority level of 7 or less can be disabled. Trap sources (level 8-15) cannot be disabled.

The `DISI` instruction provides a convenient way to disable interrupts of priority levels 1-6 for a fixed period. Level 7 interrupt sources are not disabled by the `DISI` instruction.

## 8.0 OSCILLATOR CONFIGURATION

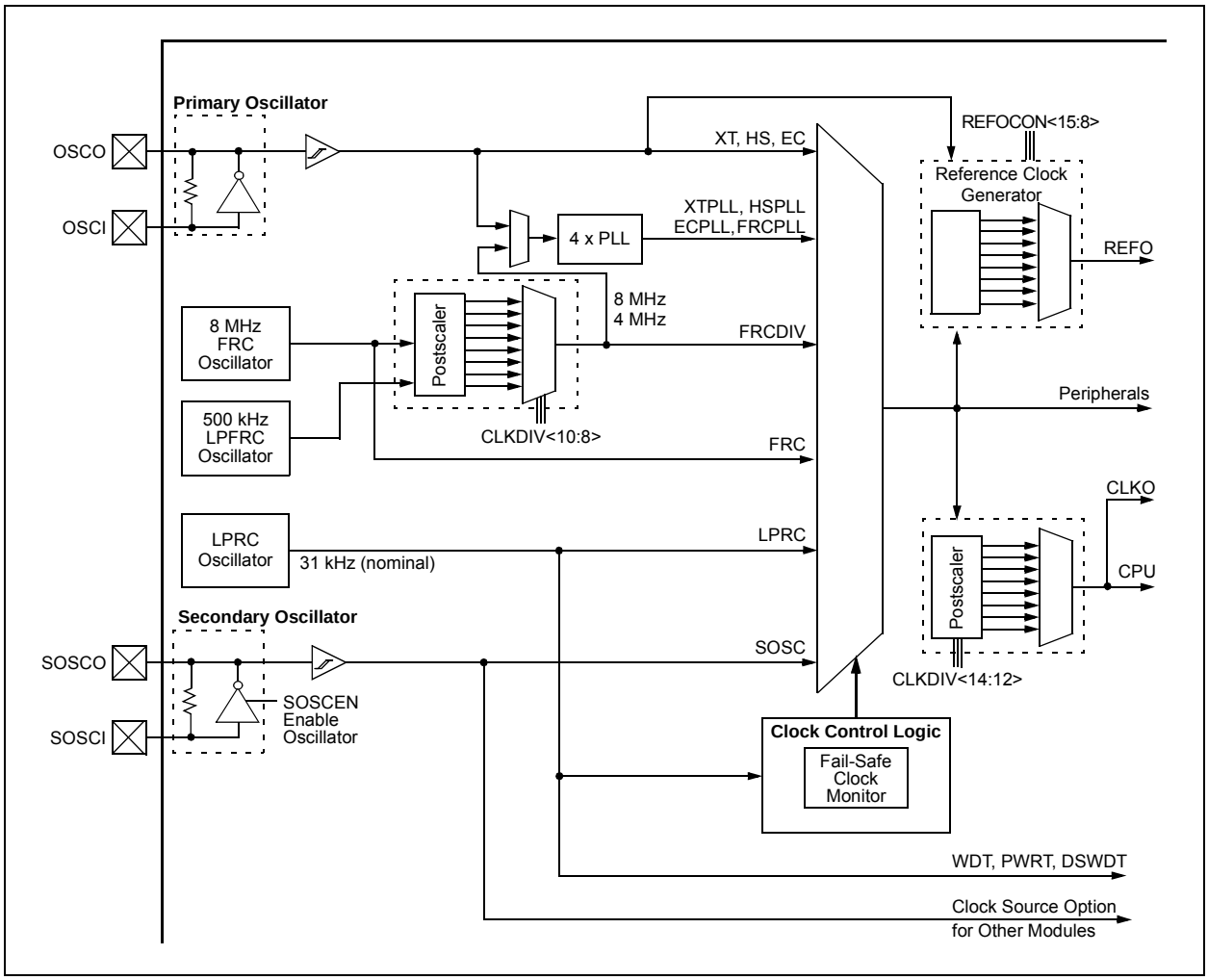
**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Oscillator Configuration, refer to the *"PIC24F Family Reference Manual"*, **Section 38. "Oscillator with 500 kHz Low-Power FRC"** (DS39726).

The oscillator system for the PIC24F04KA201 family of devices has the following features:

- A total of five external and internal oscillator options as clock sources, providing 11 different clock modes.
- On-chip 4x Phase Locked Loop (PLL) to boost internal operating frequency on select internal and external oscillator sources.
- Software-controllable switching between various clock sources.
- Software-controllable postscaler for selective clocking of CPU for system power savings.
- System frequency range declaration bits for EC mode. When using an external clock source, the current consumption is reduced by setting the declaration bits to the expected frequency range.
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and permits safe application recovery or shutdown.

Figure 8-1 provides a simplified diagram of the oscillator system.

**FIGURE 8-1: PIC24F04KA201 FAMILY CLOCK DIAGRAM**



## 8.1 CPU Clocking Scheme

The system clock source can be provided by one of four sources:

- Primary Oscillator (POSC) on the OSCI and OSCO pins
- Secondary Oscillator (SOSC) on the SOSCI and SOSCO pins

  The PIC24F04KA201 family devices consist of two types of secondary oscillator:

  - High-Power Secondary Oscillator
  - Low-Power Secondary Oscillator

  These can be selected by using the SOSCSEL (FOSC<5>) bit.
- Fast Internal RC (FRC) Oscillator
  - 8 MHz FRC Oscillator
  - 500 kHz Lower Power FRC Oscillator
- Low-Power Internal RC (LPRC) Oscillator

The primary oscillator and 8 MHz FRC sources have the option of using the internal 4x PLL. The frequency of the FRC clock source can optionally be reduced by the programmable clock divider. The selected clock source generates the processor and peripheral clock sources.

The processor clock source is divided by two to produce the internal instruction cycle clock, FCY. In this document, the instruction cycle clock is also denoted by FOSC/2. The internal instruction cycle clock, FOSC/2, can be provided on the OSCO I/O pin for some operating modes of the primary oscillator.

## 8.2 Initial Configuration on POR

The oscillator source (and operating mode) that is used at a device Power-on Reset (POR) event is selected using Configuration bit settings. The oscillator Configuration bit settings are located in the Configuration registers in the program memory (refer to **Section 23.1 "Configuration Bits"** for further details). The Primary Oscillator Configuration bits, POSCMD<1:0> (FOSC<1:0>), and the Initial Oscillator Select Configuration bits, FNOSC<2:0> (FOSCSEL<2:0>), select the oscillator source that is used at a POR. The FRC primary oscillator with postscaler (FRCDIV) is the default (unprogrammed) selection. The secondary oscillator, or one of the internal oscillators, may be chosen by programming these bit locations. The EC mode frequency range Configuration bits, POSCFREQ<1:0> (FOSC<4:3>), optimize power consumption when running in EC mode. The default configuration is "frequency range is greater than 8 MHz".

The Configuration bits allow users to choose between the various clock modes, shown in Table 8-1.

### 8.2.1 CLOCK SWITCHING MODE CONFIGURATION BITS

The FCKSM Configuration bits (FOSC<7:6>) are used jointly to configure device clock switching and the FSCM. Clock switching is enabled only when FCKSM1 is programmed ('0'). The FSCM is enabled only when FCKSM<1:0> are both programmed ('00').

**TABLE 8-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION**

| Oscillator Mode | Oscillator Source | POSCMD<1:0> | FNOSC<2:0> | Note |
|---|---|---|---|---|
| 8 MHz FRC Oscillator with Postscaler (FRCDIV) | Internal | 11 | 111 | **1, 2** |
| 500 MHz FRC Oscillator with Postscaler (LPFRCDIV) | Internal | 11 | 110 | **1** |
| Low-Power RC Oscillator (LPRC) | Internal | 11 | 101 | **1** |
| Secondary (Timer1) Oscillator (SOSC) | Secondary | 00 | 100 | **1** |
| Primary Oscillator (HS) with PLL Module (HSPLL) | Primary | 10 | 011 | |
| Primary Oscillator (EC) with PLL Module (ECPLL) | Primary | 00 | 011 | |
| Primary Oscillator (HS) | Primary | 10 | 010 | |
| Primary Oscillator (XT) | Primary | 01 | 010 | |
| Primary Oscillator (EC) | Primary | 00 | 010 | |
| 8 MHz FRC Oscillator with PLL Module (FRCPLL) | Internal | 11 | 001 | **1** |
| 8 MHz FRC Oscillator (FRC) | Internal | 11 | 000 | **1** |

**Note 1:** OSCO pin function is determined by the OSCIOFNC Configuration bit.

**2:** This is the default oscillator mode for an unprogrammed (erased) device.

## 8.3 Control Registers

The operation of the oscillator is controlled by three Special Function Registers (SFRs):

- OSCCON
- CLKDIV
- OSCTUN

The OSCCON register (Register 8-1) is the main control register for the oscillator. It controls clock source switching and allows the monitoring of clock sources.

The Clock Divider register (Register 8-2) controls the features associated with Doze mode, as well as the postscaler for the FRC oscillator.

The FRC Oscillator Tune register (Register 8-3) allows the user to fine tune the FRC oscillator over a range of approximately ±12%. Each bit increment or decrement changes the factory calibrated frequency of the FRC oscillator by a fixed amount.

**REGISTER 8-1: OSCCON: OSCILLATOR CONTROL REGISTER**

| U-0 | R-0, HSC | R-0, HSC | R-0, HSC | U-0 | R/W-x(1) | R/W-x(1) | R/W-x(1) |
|---|---|---|---|---|---|---|---|
| — | COSC2 | COSC1 | COSC0 | — | NOSC2 | NOSC1 | NOSC0 |
| bit 15 | | | | | | | bit 8 |

| R/SO-0, HSC | U-0 | R-0, HSC(2) | U-0 | R/CO-0, HS | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CLKLOCK | — | LOCK | — | CF | — | SOSCEN | OSWEN |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | CO = Clear Only bit | SO = Set Only bit | |
|---|---|---|---|
| | HS = Hardware Settable bit | HSC = Hardware Settable/Clearable bit | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ | |
| -n = Value at POR | ‘1’ = Bit is set | ‘0’ = Bit is cleared | x = Bit is unknown |

bit 15 **Unimplemented:** Read as ‘0’

bit 14-12 **COSC<2:0>:** Current Oscillator Selection bits

111 = 8 MHz Fast RC Oscillator with Postscaler (FRCDIV)
110 = 500 kHz Low-Power Fast RC Oscillator (FRC) with Postscaler (LPFRCDIV)
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)
011 = Primary Oscillator with PLL module (XTPLL, HSPLL, ECPLL)
010 = Primary Oscillator (XT, HS, EC)
001 = 8 MHz FRC Oscillator with Postscaler and PLL module (FRCPLL)
000 = 8 MHz FRC Oscillator (FRC)

bit 11 **Unimplemented:** Read as ‘0’

bit 10-8 **NOSC<2:0>:** New Oscillator Selection bits(1)

111 = 8 MHz Fast RC Oscillator with Postscaler (FRCDIV)
110 = 500 kHz Low-Power Fast RC Oscillator (FRC) with Postscaler (LPFRCDIV)
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)
011 = Primary Oscillator with PLL module (XTPLL, HSPLL, ECPLL)
010 = Primary Oscillator (XT, HS, EC)
001 = 8 MHz FRC Oscillator with Postscaler and PLL module (FRCPLL)
000 = 8 MHz FRC Oscillator (FRC)

**Note 1:** Reset values for these bits are determined by the FNOSC Configuration bits.
**2:** Also resets to ‘0’ during any valid clock switch or whenever a non-PLL Clock mode is selected.

**REGISTER 8-1: OSCCON: OSCILLATOR CONTROL REGISTER (CONTINUED)**

bit 7 **CLKLOCK:** Clock Selection Lock Enabled bit
If FSCM is enabled (FCKSM1 = 1):
1 = Clock and PLL selections are locked
0 = Clock and PLL selections are not locked and may be modified by setting the OSWEN bit
If FSCM is disabled (FCKSM1 = 0):
Clock and PLL selections are never locked and may be modified by setting the OSWEN bit.

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

bit 5 **LOCK:** PLL Lock Status bit(2)
1 = PLL module is in lock or PLL module start-up timer is satisfied
0 = PLL module is out of lock, PLL start-up timer is running or PLL is disabled

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

bit 3 **CF:** Clock Fail Detect bit
1 = FSCM has detected a clock failure
0 = No clock failure has been detected

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

bit 1 **SOSCEN:** 32 kHz Secondary Oscillator (SOSC) Enable bit
1 = Enable secondary oscillator
0 = Disable secondary oscillator

bit 0 **OSWEN:** Oscillator Switch Enable bit
1 = Initiate an oscillator switch to clock source specified by NOSC<2:0> bits
0 = Oscillator switch is complete

**Note 1:** Reset values for these bits are determined by the FNOSC Configuration bits.
**2:** Also resets to '0' during any valid clock switch or whenever a non-PLL Clock mode is selected.

**REGISTER 8-2: CLKDIV: CLOCK DIVIDER REGISTER**

| R/W-0 | R/W-0 | R/W-1 | R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-1 |
|---|---|---|---|---|---|---|---|
| ROI | DOZE2 | DOZE1 | DOZE0 | DOZEN(1) | RCDIV2 | RCDIV1 | RCDIV0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ROI:** Recover on Interrupt bit
1 = Interrupts clear the DOZEN bit and reset the CPU and peripheral clock ratio to 1:1
0 = Interrupts have no effect on the DOZEN bit

bit 14-12 **DOZE<2:0>:** CPU and Peripheral Clock Ratio Select bits
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

bit 11 **DOZEN:** DOZE Enable bit(1)
1 = DOZE<2:0> bits specify the CPU and peripheral clock ratio
0 = CPU and peripheral clock ratio set to 1:1

bit 10-8 **RCDIV<2:0>:** FRC Postscaler Select bits
When OSCCON (COSC<2:0>) = 111:
111 = 31.25 kHz (divide by 256)
110 = 125 kHz (divide by 64)
101 = 250 kHz (divide by 32)
100 = 500 kHz (divide by 16)
011 = 1 MHz (divide by 8)
010 = 2 MHz (divide by 4)
001 = 4 MHz (divide by 2) (default)
000 = 8 MHz (divide by 1)
When OSCCON (COSC<2:0>) = 110:
111 = 1.95 kHz (divide by 256)
110 = 7.81 kHz (divide by 64)
101 = 15.62 kHz (divide by 32)
100 = 31.25 kHz (divide by 16)
011 = 62.5 kHz (divide by 8)
010 = 125 kHz (divide by 4)
001 = 250 kHz (divide by 2) (default)
000 = 500 kHz (divide by 1)

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

**Note 1:** This bit is automatically cleared when the ROI bit is set and an interrupt occurs.

**REGISTER 8-3: OSCTUN: FRC OSCILLATOR TUNE REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | TUN5(1) | TUN4(1) | TUN3(1) | TUN2(1) | TUN1(1) | TUN0(1) |
| bit 7 | | | | | | | bit 0 |

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

bit 15-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** FRC Oscillator Tuning bits(1)

011111 = Maximum frequency deviation
011110
.
.
.
000001
000000 = Center frequency, oscillator is running at factory calibrated frequency
111111
.
.
.
100001
100000 = Minimum frequency deviation

**Note 1:** Increments or decrements of TUN<5:0> may not change the FRC frequency in equal steps over the FRC tuning range and may not be monotonic.

## 8.4 Clock Switching Operation

With few limitations, applications are free to switch between any of the four clock sources (POSC, SOSC, FRC and LPRC) under software control and at any time. To limit the possible side effects that could result from this flexibility, PIC24F devices have a safeguard lock built into the switching process.

| | |
|---|---|
| **Note:** | The primary oscillator mode has three different submodes (XT, HS and EC), which are determined by the POSCMDx Configuration bits. While an application can switch to and from primary oscillator mode in software, it cannot switch between the different primary submodes without reprogramming the device. |

### 8.4.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit in the FOSC Configuration register must be programmed to '0'. (Refer to **Section 23.1 "Configuration Bits"** for further details.) If the FCKSM1 Configuration bit is unprogrammed ('1'), the clock switching function and FSCM function are disabled. This is the default setting.

The NOSCx control bits (OSCCON<10:8>) do not control the clock selection when clock switching is disabled. However, the COSCx bits (OSCCON<14:12>) will reflect the clock source selected by the FNOSCx Configuration bits.

The OSWEN control bit (OSCCON<0>) has no effect when clock switching is disabled; it is held at '0' at all times.

### 8.4.2 OSCILLATOR SWITCHING SEQUENCE

At a minimum, performing a clock switch requires this basic sequence:

1. If desired, read the COSCx bits (OSCCON<14:12>), to determine the current oscillator source.
2. Perform the unlock sequence to allow a write to the OSCCON register high byte.
3. Write the appropriate value to the NOSCx bits (OSCCON<10:8>) for the new oscillator source.
4. Perform the unlock sequence to allow a write to the OSCCON register low byte.
5. Set the OSWEN bit to initiate the oscillator switch.

Once the basic sequence is completed, the system clock hardware responds automatically as follows:

1. The clock switching hardware compares the COSCx bits with the new value of the NOSCx bits. If they are the same, then the clock switch is a redundant operation. In this case, the OSWEN bit is cleared automatically and the clock switch is aborted.
2. If a valid clock switch has been initiated, the LOCK (OSCCON<5>) and CF (OSCCON<3>) bits are cleared.
3. The new oscillator is turned on by the hardware if it is not currently running. If a crystal oscillator must be turned on, the hardware will wait until the OST expires. If the new source is using the PLL, then the hardware waits until a PLL lock is detected (LOCK = 1).
4. The hardware waits for 10 clock cycles from the new clock source and then performs the clock switch.
5. The hardware clears the OSWEN bit to indicate a successful clock transition. In addition, the NOSCx bits value is transferred to the COSCx bits.
6. The old clock source is turned off at this time, with the exception of LPRC (if WDT or FSCM with LPRC as a clock source is enabled) or SOSC (if SOSCEN remains enabled).

| | |
|---|---|
| **Note 1:** | The processor will continue to execute code throughout the clock switching sequence. Timing-sensitive code should not be executed during this time. |
| **2:** | Direct clock switches between any primary oscillator mode with PLL and FRCPLL mode are not permitted. This applies to clock switches in either direction. In these instances, the application must switch to FRC mode as a transition clock source between the two PLL modes. |

The following code sequence for a clock switch is recommended:

1. Disable interrupts during the OSCCON register unlock and write sequence.
2. Execute the unlock sequence for the OSCCON high byte by writing 78h and 9Ah to OSCCON<15:8> in two back-to-back instructions.
3. Write new oscillator source to the NOSCx bits in the instruction immediately following the unlock sequence.
4. Execute the unlock sequence for the OSCCON low byte by writing 46h and 57h to OSCCON<7:0> in two back-to-back instructions.
5. Set the OSWEN bit in the instruction immediately following the unlock sequence.
6. Continue to execute code that is not clock-sensitive (optional).
7. Invoke an appropriate amount of software delay (cycle counting) to allow the selected oscillator and/or PLL to start and stabilize.
8. Check to see if OSWEN is '0'. If it is, the switch was successful. If OSWEN is still set, then check the LOCK bit to determine the cause of failure.

The core sequence for unlocking the OSCCON register and initiating a clock switch is provided in Example 8-1.

**EXAMPLE 8-1: BASIC CODE SEQUENCE FOR CLOCK SWITCHING**

```
;Place the new oscillator selection in W0
;OSCCONH (high byte) Unlock Sequence
MOV         #OSCCONH, w1
MOV         #0x78, w2
MOV         #0x9A, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Set new oscillator selection
MOV.b       WREG, OSCCONH
;OSCCONL (low byte) unlock sequence
MOV         #OSCCONL, w1
MOV         #0x46, w2
MOV         #0x57, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Start oscillator switch operation
BSET        OSCCON,#0
```

## 8.5 Reference Clock Output

In addition to the CLKO output (FOSC/2) available in certain oscillator modes, the device clock in the PIC24F04KA201 family devices can also be configured to provide a reference clock output signal to a port pin. This feature is available in all oscillator configurations and allows the user to select a greater range of clock submultiples to drive external devices in the application.

This reference clock output is controlled by the REFOCON register (Register 8-4). Setting the ROEN bit (REFOCON<15>) makes the clock signal available on the REFO pin. The RODIV bits (REFOCON<11:8>) enable the selection of 16 different clock divider options.

The ROSSLP and ROSEL bits (REFOCON<13:12>) control the availability of the reference output during Sleep mode. The ROSEL bit determines if the oscillator on OSC1 and OSC2, or the current system clock source, is used for the reference clock output. The ROSSLP bit determines if the reference source is available on REFO when the device is in Sleep mode.

To use the reference clock output in Sleep mode, both the ROSSLP and ROSEL bits must be set. The device clock must also be configured for one of the primary modes (EC, HS or XT); otherwise, if the ROSEL bit is not also set, the oscillator on OSC1 and OSC2 will be powered down when the device enters Sleep mode. Clearing the ROSEL bit allows the reference output frequency to change as the system clock changes during any clock switches.

**REGISTER 8-4: REFOCON: REFERENCE OSCILLATOR CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ROEN | — | ROSSLP | ROSEL | RODIV3 | RODIV2 | RODIV1 | RODIV0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ROEN:** Reference Oscillator Output Enable bit
1 = Reference oscillator enabled on REFO pin
0 = Reference oscillator disabled

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **ROSSLP:** Reference Oscillator Output Stop in Sleep bit
1 = Reference oscillator continues to run in Sleep
0 = Reference oscillator is disabled in Sleep

bit 12 **ROSEL:** Reference Oscillator Source Select bit
1 = Primary oscillator used as the base clock(1)
0 = System clock used as the base clock; base clock reflects any clock switching of the device

bit 11-8 **RODIV3:RODIV0:** Reference Oscillator Divisor Select bits
1111 = Base clock value divided by 32,768
1110 = Base clock value divided by 16,384
1101 = Base clock value divided by 8,192
1100 = Base clock value divided by 4,096
1011 = Base clock value divided by 2,048
1010 = Base clock value divided by 1,024
1001 = Base clock value divided by 512
1000 = Base clock value divided by 256
0111 = Base clock value divided by 128
0110 = Base clock value divided by 64
0101 = Base clock value divided by 32
0100 = Base clock value divided by 16
0011 = Base clock value divided by 8
0010 = Base clock value divided by 4
0001 = Base clock value divided by 2
0000 = Base clock value

bit 7-0 **Unimplemented:** Read as ‘0’

**Note 1:** The crystal oscillator must be enabled using the FOSC<2:0> bits; the crystal maintains the operation in Sleep mode.

**NOTES:**

## 9.0 POWER-SAVING FEATURES

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the *"PIC24F Family Reference Manual"*, **Section 39. "Power-Saving Features with Deep Sleep"** (DS39727).

The PIC24F04KA201 family of devices provides the ability to manage power consumption by selectively managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. All PIC24F devices manage power consumption in four different ways:

- Clock frequency
- Instruction-based Sleep, Idle and Deep Sleep modes
- Software controlled Doze mode
- Selective peripheral control in software

Combinations of these methods can be used to selectively tailor an application's power consumption, while still maintaining critical application features, such as timing-sensitive communications.

### 9.1 Clock Frequency and Clock Switching

PIC24F devices allow for a wide range of clock frequencies to be selected under application control. If the system clock configuration is not locked, users can choose low-power or high-precision oscillators by simply changing the NOSC bits. The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 8.0 "Oscillator Configuration"**.

### 9.2 Instruction-Based Power-Saving Modes

PIC24F devices have two special power-saving modes that are entered through the execution of a special PWRSAV instruction. Sleep mode stops clock operation and halts all code execution; Idle mode halts the CPU and code execution, but allows peripheral modules to continue operation. Deep Sleep mode stops clock operation, code execution and all peripherals except DSWDT. It also freezes I/O states and removes power to SRAM and Flash memory.

The assembly syntax of the PWRSAV instruction is shown in Example 9-1.

**Note:** SLEEP_MODE and IDLE_MODE are constants defined in the assembler include file for the selected device.

Sleep and Idle modes can be exited as a result of an enabled interrupt, WDT time-out or a device Reset. When the device exits these modes, it is said to "wake-up".

#### 9.2.1 SLEEP MODE

Sleep mode has these features:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption will be reduced to a minimum provided that no I/O pin is sourcing current.
- The I/O pin directions and states are frozen.
- The Fail-Safe Clock Monitor does not operate during Sleep mode since the system clock source is disabled.
- The LPRC clock will continue to run in Sleep mode if the WDT with LPRC as a clock source is enabled.
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode.
- Some device features or peripherals may continue to operate in Sleep mode. This includes items such as the input change notification on the I/O ports, or peripherals that use an external clock input. Any peripheral that requires the system clock source for its operation will be disabled in Sleep mode.

The device will wake-up from Sleep mode on any of these events:

- On any interrupt source that is individually enabled
- On any form of device Reset
- On a WDT time-out

On wake-up from Sleep, the processor will restart with the same clock source that was active when Sleep mode was entered.

**EXAMPLE 9-1: PWRSAV INSTRUCTION SYNTAX**

```
PWRSAV     #SLEEP_MODE    ; Put the device into SLEEP mode
PWRSAV     #IDLE_MODE     ; Put the device into IDLE mode
BSET       DSCON, #DSEN   ; Enable Deep Sleep
PWRSAV     #SLEEP_MODE    ; Put the device into Deep SLEEP mode
```

### 9.2.2 IDLE MODE

Idle mode has these features:

- The CPU will stop executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 9.4 "Selective Peripheral Module Control"**).
- If the WDT or FSCM is enabled, the LPRC will also remain active.

The device will wake from Idle mode on any of these events:

- Any interrupt that is individually enabled
- Any device Reset
- A WDT time-out

On wake-up from Idle, the clock is reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the PWRSAV instruction or the first instruction in the ISR.

### 9.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a PWRSAV instruction will be held off until entry into Sleep or Idle mode has completed. The device will then wake-up from Sleep or Idle mode.

### 9.2.4 DEEP SLEEP MODE

In PIC24F04KA201 family devices, Deep Sleep mode is intended to provide the lowest levels of power consumption available, without requiring the use of external switches to completely remove all power from the device. Entry into Deep Sleep mode is completely under software control. Exit from Deep Sleep mode can be triggered from any of the following events:

- POR event
- $\overline{\text{MCLR}}$ event
- External Interrupt 0
- Deep Sleep Watchdog Timer (DSWDT) time-out

The device has a dedicated Deep Sleep Brown-out Reset (DSBOR) and a Deep Sleep Watchdog Timer Reset (DSWDT) for monitoring voltage and time-out events. The DSBOR and DSWDT are independent of the standard BOR and WDT used with other power-managed modes (Sleep, Idle and Doze).

#### 9.2.4.1 Entering Deep Sleep Mode

Deep Sleep mode is entered by setting the DSEN bit in the DSCON register, and then executing a SLEEP instruction (PWRSAV #SLEEP_MODE) within one instruction cycle to minimize the chance that Deep Sleep will be spuriously entered.

If the PWRSAV command is not given within one instruction cycle, the DSEN bit will be cleared by the hardware and must be set again by the software before entering Deep Sleep mode. The DSEN bit is also automatically cleared when exiting the Deep Sleep mode.

| **Note:** | To re-enter Deep Sleep after a Deep Sleep wake-up, allow a delay of at least 3 TCY after clearing the RELEASE bit. |
|---|---|

The sequence to enter Deep Sleep mode is:

1. If the application requires the Deep Sleep WDT, enable it and configure its clock source (see **Section 9.2.4.5 "Deep Sleep WDT"** for details).
2. If the application requires Deep Sleep BOR, enable it by programming the DSBOREN Configuration bit (FDS<6>).
3. If needed, save any critical application context data by writing it to the DSGPR0 and DSGPR1 registers (optional).
4. Enable Deep Sleep mode by setting the DSEN bit (DSCON<15>).
5. Enter Deep Sleep mode by issuing 3 NOP commands and then a PWRSAV #0 instruction.

Any time the DSEN bit is set, all bits in the DSWSRC register will be automatically cleared.

#### 9.2.4.2 Exiting Deep Sleep Mode

Deep Sleep mode exits on any one of the following events:

- POR event on VDD supply. If there is no DSBOR circuit to re-arm the VDD supply POR circuit, the external VDD supply must be lowered to the natural arming voltage of the POR circuit.
- DSWDT time-out. When the DSWDT timer times out, the device exits Deep Sleep.
- Assertion ('0') of the $\overline{\text{MCLR}}$ pin.
- Assertion of the INT0 pin (if the interrupt was enabled before Deep Sleep mode was entered). The polarity configuration is used to determine the assertion level ('0' or '1') of the pin that will cause an exit from Deep Sleep mode. Exiting from Deep Sleep mode requires a change on the INT0 pin while in Deep Sleep mode.

| **Note:** | Any interrupt pending when entering Deep Sleep mode is cleared, |
|---|---|

Exiting Deep Sleep mode generally does not retain the state of the device and is equivalent to a Power-on Reset (POR) of the device. Exceptions to this include the DSGPRx registers and DSWSRC.

Wake-up events that occur from the time Deep Sleep exits until the time the POR sequence completes are ignored, and are not be captured in the DSWAKE register.

The sequence for exiting Deep Sleep mode is:

1. After a wake-up event, the device exits Deep Sleep and performs a POR. The DSEN bit is cleared automatically. Code execution resumes at the Reset vector.
2. To determine if the device exited Deep Sleep, read the Deep Sleep bit, DPSLP (RCON<10>). This bit will be set if there was an exit from Deep Sleep mode. If the bit is set, clear it.
3. Determine the wake-up source by reading the DSWAKE register.
4. Determine if a DSBOR event occurred during Deep Sleep mode by reading the DSBOR bit (DSCON<1>).
5. If application context data has been saved, read it back from the DSGPR0 and DSGPR1 registers.
6. Clear the RELEASE bit (DSCON<0>).

#### 9.2.4.3 Saving Context Data with the DSGPR0/DSGPR1 Registers

As exiting Deep Sleep mode causes a POR, most Special Function Registers reset to their default POR values. In addition, because $V_{DDCORE}$ power is not supplied in Deep Sleep mode, information in data RAM may be lost when exiting this mode.

Applications which require critical data to be saved prior to Deep Sleep may use the Deep Sleep General Purpose registers, DSGPR0 and DSGPR1, or data EEPROM (if available). Unlike other SFRs, the contents of these registers are preserved while the device is in Deep Sleep mode. After exiting Deep Sleep, software can restore the data by reading the registers and clearing the RELEASE bit (DSCON<0>).

#### 9.2.4.4 I/O Pins During Deep Sleep

During Deep Sleep, the general purpose I/O pins retain their previous states and the Secondary Oscillator (SOSC) will remain running, if enabled. Pins that are configured as inputs (TRIS bit set) prior to entry into Deep Sleep remain high-impedance during Deep Sleep. Pins that are configured as outputs (TRIS bit clear) prior to entry into Deep Sleep remain as output pins during Deep Sleep. While in this mode, they continue to drive the output level determined by their corresponding LAT bit at the time of entry into Deep Sleep.

Once the device wakes back up, all I/O pins continue to maintain their previous states, even after the device has finished the POR sequence and is executing application code again. Pins configured as inputs during Deep Sleep remain high-impedance and pins configured as outputs continue to drive their previous value. After waking up, the TRIS and LAT registers, and the SOSCEN bit (OSCCON<1>) are reset. If firmware modifies any of these bits or registers, the I/O will not immediately go to the newly configured states. Once the firmware clears the RELEASE bit (DSCON<0>) the I/O pins are “released”. This causes the I/O pins to take the states configured by their respective TRIS and LAT bit values.

This means that keeping the SOSC running after waking up requires the SOSCEN bit to be set before clearing RELEASE.

If the Deep Sleep BOR (DSBOR) is enabled, and a DSBOR or a true POR event occurs during Deep Sleep, the I/O pins will be immediately released similar to clearing the RELEASE bit. All previous state information will be lost, including the general purpose DSGPR0 and DSGPR1 contents.

If a $\overline{\text{MCLR}}$ Reset event occurs during Deep Sleep, the DSGPRx, DSCON and DSWAKE registers will remain valid and the RELEASE bit will remain set. The state of the SOSC also will be retained. The I/O pins, however, will be reset to their $\overline{\text{MCLR}}$ Reset state. Since RELEASE is still set, changes to the SOSCEN bit (OSCCON<1>) cannot take effect until the RELEASE bit is cleared.

In all other Deep Sleep wake-up cases, application firmware must clear the RELEASE bit in order to reconfigure the I/O pins.

#### 9.2.4.5 Deep Sleep WDT

To enable the DSWDT in Deep Sleep mode, program the Configuration bit, DSWDTEN (FDS<7>). The device Watchdog Timer (WDT) need not be enabled for the DSWDT to function. Entry into Deep Sleep mode automatically resets the DSWDT.

The DSWDT clock source is selected by the DSWDTOSC Configuration bit (FDS<4>). The postscaler options are programmed by the DSWDTPS<3:0> Configuration bits (FDS<3:0>). The minimum time-out period that can be achieved is 2.1 ms and the maximum is 25.7 days. For more details on the FDS Configuration register and DSWDT configuration options, refer to **Section 23.0 "Special Features"**.

#### 9.2.4.6 Switching Clocks in Deep Sleep Mode

The DSWDT may run from either SOSC or the LPRC clock source. This allows the DSWDT to run without requiring both the LPRC and SOSC to be enabled together, reducing power consumption.

Under certain circumstances, it is possible for the DSWDT clock source to be off when entering Deep Sleep mode. In this case, the clock source is turned on automatically (if DSWDT is enabled), without the need for software intervention. However, this can cause a delay in the start of the DSWDT counters. In order to avoid this delay when using SOSC as a clock source, the application can activate SOSC prior to entering Deep Sleep mode.

#### 9.2.4.7 Checking and Clearing the Status of Deep Sleep

Upon entry into Deep Sleep mode, the status bit, DPSLP (RCON<10>), becomes set and must be cleared by the software.

On power-up, the software should read this status bit to determine if the Reset was due to an exit from Deep Sleep mode and clear the bit if it is set. Of the four possible combinations of DPSLP and POR bit states, three cases can be considered:

- Both the DPSLP and POR bits are cleared. In this case, the Reset was due to some event other than a Deep Sleep mode exit.
- The DPSLP bit is clear, but the POR bit is set. This is a normal POR.
- Both the DPSLP and POR bits are set. This means that Deep Sleep mode was entered, the device was powered down and Deep Sleep mode was exited.

#### 9.2.4.8 Power-on Resets (PORs)

VDD voltage is monitored to produce PORs. Since exiting from Deep Sleep functionally looks like a POR, the technique described in **Section 9.2.4.7 "Checking and Clearing the Status of Deep Sleep"** should be used to distinguish between Deep Sleep and a true POR event.

When a true POR occurs, the entire device including all Deep Sleep logic, (Deep Sleep registers, DSWDT, etc.) is reset.

#### 9.2.4.9 Summary of Deep Sleep Sequence

To review, these are the necessary steps involved in invoking and exiting Deep Sleep mode:

1. Device exits Reset and begins to execute its application code.
2. If DSWDT functionality is required, program the appropriate Configuration bit.
3. Select the appropriate clock(s) for the DSWDT (optional).
4. Enable and configure the DSWDT (optional).
5. Write context data to the DSGPRx registers (optional).
6. Enable the INT0 interrupt (optional).
7. Set the DSEN bit in the DSCON register.
8. Enter Deep Sleep by issuing a `PWRSV #SLEEP_MODE` command.
9. Device exits Deep Sleep when a wake-up event occurs.
10. The DSEN bit is automatically cleared.
11. Read and clear the DPSLP status bit in RCON, and the DSWAKE status bits.
12. Read the DSGPRx registers (optional).
13. Once all state related configurations are complete, clear the RELEASE bit.
14. Application resumes normal operation.

**REGISTER 9-1: DSCON: DEEP SLEEP CONTROL REGISTER(1)**

| R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| DSEN | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/C-0, HS |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | DSBOR(2) | RELEASE |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **DSEN:** Deep Sleep Enable bit
1 = Enters Deep Sleep on execution of PWRSAV #0
0 = Enters normal Sleep on execution of PWRSAV #0

bit 14-2 **Unimplemented:** Read as '0'

bit 1 **DSBOR:** Deep Sleep BOR Event bit(2)
1 = The DSBOR was active and a BOR event was detected during Deep Sleep
0 = The DSBOR was not active, or was active but did not detect a BOR event during Deep Sleep

bit 0 **RELEASE:** I/O Pin State Release bit
1 = Upon waking from Deep Sleep, I/O pins maintain their states previous to Deep Sleep entry
0 = Release I/O pins from their state previous to Deep Sleep entry, and allow their respective TRIS and LAT bits to control their states

**Note 1:** All register bits are reset only in the case of a POR event outside of Deep Sleep mode.
**2:** Unlike all other events, a Deep Sleep BOR event will NOT cause a wake-up from Deep Sleep; this re-arms POR.

**REGISTER 9-2: DSWSRC: DEEP SLEEP WAKE-UP SOURCE REGISTER(1)**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0, HS |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | DSINT0 |
| bit 15 | | | | | | | bit 8 |

| R/W-0, HS | U-0 | U-0 | R/W-0, HS | U-0 | R/W-0, HS | U-0 | R/W-0, HS |
|---|---|---|---|---|---|---|---|
| DSFLT | — | — | DSWDT | — | DSMCLR | — | DSPOR(2,3) |
| bit 7 | | | | | | | bit 0 |

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

bit 15-9 **Unimplemented:** Read as '0'

bit 8 **DSINT0:** Interrupt-on-Change bit
1 = Interrupt-on-change was asserted during Deep Sleep
0 = Interrupt-on-change was not asserted during Deep Sleep

bit 7 **DSFLT:** Deep Sleep Fault Detected bit
1 = A Fault occurred during Deep Sleep, and some Deep Sleep configuration settings may have been corrupted
0 = No Fault was detected during Deep Sleep

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

bit 4 **DSWDT:** Deep Sleep Watchdog Timer Time-out bit
1 = The Deep Sleep Watchdog Timer timed out during Deep Sleep
0 = The Deep Sleep Watchdog Timer did not time out during Deep Sleep

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

bit 2 **DSMCLR:** $\overline{\text{MCLR}}$ Event bit
1 = The $\overline{\text{MCLR}}$ pin was active and was asserted during Deep Sleep
0 = The $\overline{\text{MCLR}}$ pin was not active, or was active, but not asserted during Deep Sleep

bit 1 **Unimplemented:** Read as '0'

bit 0 **DSPOR:** Power-on Reset Event bit(2,3)
1 = The VDD supply POR circuit was active and a POR event was detected
0 = The VDD supply POR circuit was not active, or was active but did not detect a POR event

**Note 1:** All register bits are cleared when the DSCON<DSEN> bit is set.

**2:** All register bits are reset only in the case of a POR event outside Deep Sleep mode, except the DSPOR bit, which does not reset on a POR event that is caused due to a Deep Sleep exit.

**3:** Unlike the other bits in this register, this bit can be set outside of Deep Sleep.

#### 9.2.5 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a PWRSAV instruction will be held off until entry into Sleep or Idle mode is completed. The device will then wake-up from Sleep or Idle mode.

## 9.3 Doze Mode

Generally, changing clock speed and invoking one of the power-saving modes are the preferred strategies for reducing power consumption. There may be circumstances, however, where this is not practical. For example, it may be necessary for an application to maintain uninterrupted synchronous communication, even while it is doing nothing else. Reducing system clock speed may introduce communication errors, while using a power-saving mode may stop communications completely.

Doze mode is a simple and effective alternative method to reduce power consumption while the device is still executing code. In this mode, the system clock continues to operate from the same source and at the same speed. Peripheral modules continue to be clocked at the same speed while the CPU clock speed is reduced. Synchronization between the two clock domains is maintained, allowing the peripherals to access the SFRs while the CPU executes code at a slower rate.

Doze mode is enabled by setting the DOZEN bit (CLKDIV<11>). The ratio between peripheral and core clock speed is determined by the DOZE<2:0> bits (CLKDIV<14:12>). There are eight possible configurations, from 1:1 to 1:128, with 1:1 being the default.

It is also possible to use Doze mode to selectively reduce power consumption in event driven applications. This allows clock-sensitive functions, such as synchronous communications, to continue without interruption while the CPU Idles, waiting for something to invoke an interrupt routine. Enabling the automatic return to full-speed CPU operation on interrupts is enabled by setting the ROI bit (CLKDIV<15>). By default, interrupt events have no effect on Doze mode operation.

## 9.4 Selective Peripheral Module Control

Idle and Doze modes allow users to substantially reduce power consumption by slowing or stopping the CPU clock. Even so, peripheral modules still remain clocked, and thus, consume power. There may be cases where the application needs what these modes do not provide: the allocation of power resources to CPU processing with minimal power consumption from the peripherals.

PIC24F devices address this requirement by allowing peripheral modules to be selectively disabled, reducing or eliminating their power consumption. This can be done with two control bits:

- The Peripheral Enable bit, generically named, “XXXEN”, located in the module’s main control SFR.
- The Peripheral Module Disable (PMD) bit, generically named, “XXXMD”, located in one of the PMD Control registers.

Both bits have similar functions in enabling or disabling its associated module. Setting the PMD bit for a module disables all clock sources to that module, reducing its power consumption to an absolute minimum. In this state, the control and status registers associated with the peripheral will also be disabled, so writes to those registers will have no effect and read values will be invalid. Many peripheral modules have a corresponding PMD bit.

In contrast, disabling a module by clearing its XXXEN bit disables its functionality, but leaves its registers available to be read and written to. Power consumption is reduced, but not by as much as the PMD bits are used. Most peripheral modules have an enable bit; exceptions include capture and compare modules.

To achieve more selective power savings, peripheral modules can also be selectively disabled when the device enters Idle mode. This is done through the control bit of the generic name format, “XXXIDL”. By default, all modules that can operate during Idle mode will do so. Using the disable on Idle feature disables the module while in Idle mode, allowing further reduction of power consumption during Idle mode, enhancing power savings for extremely critical power applications.

**NOTES:**

## 10.0 I/O PORTS

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the I/O Ports, refer to the *"PIC24F Family Reference Manual"*, **Section 12. "I/O Ports with Peripheral Pin Select (PPS)"** (DS39711). Note that the PIC24F04KA201 family devices do not support Peripheral Pin Select features.

All of the device pins (except VDD and VSS) are shared between the peripherals and the parallel I/O ports. All I/O input ports feature Schmitt Trigger inputs for improved noise immunity.

### 10.1 Parallel I/O (PIO) Ports

A parallel I/O port that shares a pin with a peripheral is, in general, subservient to the peripheral. The peripheral's output buffer data and control signals are provided to a pair of multiplexers. The multiplexers select whether the peripheral or the associated port has ownership of the output data and control signals of the I/O pin. The logic also prevents "loop through", in which a port's digital output can drive the input of a peripheral that shares the same pin. Figure 10-1 displays how ports are shared with other peripherals and the associated I/O pin to which they are connected.

When a peripheral is enabled and the peripheral is actively driving an associated pin, the use of the pin as a general purpose output pin is disabled. The I/O pin may be read, but the output driver for the parallel port bit will be disabled. If a peripheral is enabled, but the peripheral is not actively driving a pin, that pin may be driven by a port.

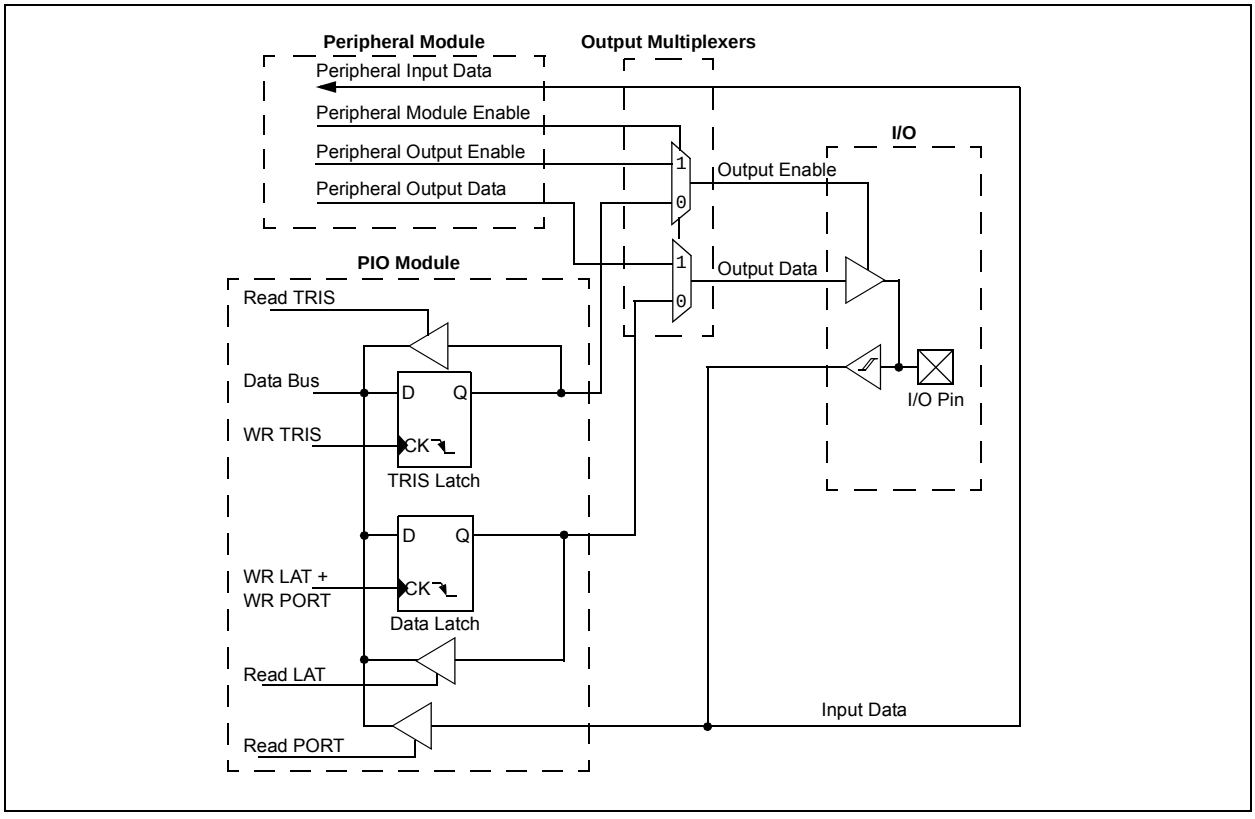
All port pins have three registers directly associated with their operation as digital I/O. The Data Direction register (TRISx) determines whether the pin is an input or an output. If the data direction bit is a '1', then the pin is an input. All port pins are defined as inputs after a Reset. Reads from the Data Latch register (LATx), read the latch. Writes to the latch, write the latch. Reads from the port (PORTx), read the port pins, while writes to the port pins, write the latch.

Any bit and its associated data and control registers that are not valid for a particular device will be disabled. That means the corresponding LATx and TRISx registers and the port pin will read as zeros.

When a pin is shared with another peripheral or function that is defined as an input only, it is nevertheless regarded as a dedicated port because there is no other competing source of outputs.

**Note:** The I/O pins retain their state during Deep Sleep. They will retain this state at wake-up until the software restore bit (RELEASE) is cleared.

**FIGURE 10-1: BLOCK DIAGRAM OF A TYPICAL SHARED PORT STRUCTURE**



### 10.1.1 OPEN-DRAIN CONFIGURATION

In addition to the PORT, LAT and TRIS registers for data control, each port pin can also be individually configured for either digital or open-drain output. This is controlled by the Open-Drain Control register, ODCx, associated with each port. Setting any of the bits configures the corresponding pin to act as an open-drain output.

The maximum open-drain voltage allowed is the same as the maximum $V_{IH}$ specification.

## 10.2 Configuring Analog Port Pins

The use of the AD1PCFG and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bit set (input). If the TRIS bit is cleared (output), the digital output level ($V_{OH}$ or $V_{OL}$) will be converted.

When reading the PORT register, all pins configured as analog input channels will read as cleared (a low level). Analog levels on any pin that is defined as a digital input (including the ANx pins) may cause the input buffer to consume current that exceeds the device specifications.

### 10.2.1 I/O PORT WRITE/READ TIMING

One instruction cycle is required between a port direction change or port write operation and a read operation of the same port. Typically, this instruction would be a NOP.

## 10.3 Input Change Notification

The input change notification function of the I/O ports allows the PIC24F04KA201 family of devices to generate interrupt requests to the processor in response to a change of state on selected input pins. This feature is capable of detecting input change of states even in Sleep mode, when the clocks are disabled. Depending on the device pin count, there are up to 17 external signals (11 on 14-pin devices) that may be selected (enabled) for generating an interrupt request on a change of state.

There are six control registers associated with the CN module. The CNEN1 and CNEN2 registers contain the interrupt enable control bits for each of the CN input pins. Setting any of these bits enables a CN interrupt for the corresponding pins.

Each CN pin also has a weak pull-up/pull-down connected to it. The pull-ups act as a current source that is connected to the pin and the pull-downs act as a current sink to eliminate the need for external resistors when push button or keypad devices are connected.

On any pin, only the pull-up resistor or the pull-down resistor should be enabled, but not both of them. If the push button or the keypad is connected to $V_{DD}$, enable the pull-down, or if they are connected to $V_{SS}$, enable the pull-up resistors. The pull-ups are enabled separately using the CNPU1 and CNPU2 registers, which contain the control bits for each of the CN pins.

Setting any of the control bits enables the weak pull-ups for the corresponding pins. The pull-downs are enabled separately using the CNPD1 and CNPD2 registers, which contain the control bits for each of the CN pins. Setting any of the control bits enables the weak pull-downs for the corresponding pins.

When the internal pull-up is selected, the pin uses $V_{DD}$ as the pull-up source voltage. When the internal pull-down is selected, the pins are pulled down to $V_{SS}$ by an internal resistor. Make sure that there is no external pull-up source/pull-down sink when the internal pull-ups/pull-downs are enabled.

| **Note:** | Pull-ups and pull-downs on change notification pins should always be disabled whenever the port pin is configured as a digital output. |
|---|---|

**EXAMPLE 10-1: PORT WRITE/READ EXAMPLE**

```
MOV   0xFF00, W0;          //Configure PORTB<15:8> as inputs and PORTB<7:0> as outputs
MOV   W0, TRISBB;
NOP;                       //Delay 1 cycle
BTSS  PORTB, #13;          //Next Instruction

Equivalent 'C' Code
TRISB = 0xFF00;            //Configure PORTB<15:8> as inputs and PORTB<7:0> as outputs
NOP();                     //Delay 1 cycle
if(PORTBbits.RB13 == 1)    // execute following code if PORTB pin 13 is set.
{
}
```

## 11.0 TIMER1

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Timers, refer to the *"PIC24F Family Reference Manual"*, **Section 14. "Timers"** (DS39704). |

The Timer1 module is a 16-bit timer which can operate as a free-running, interval timer/counter. Timer1 can operate in three modes:

- 16-Bit Timer
- 16-Bit Synchronous Counter
- 16-Bit Asynchronous Counter

Timer1 also supports these features:

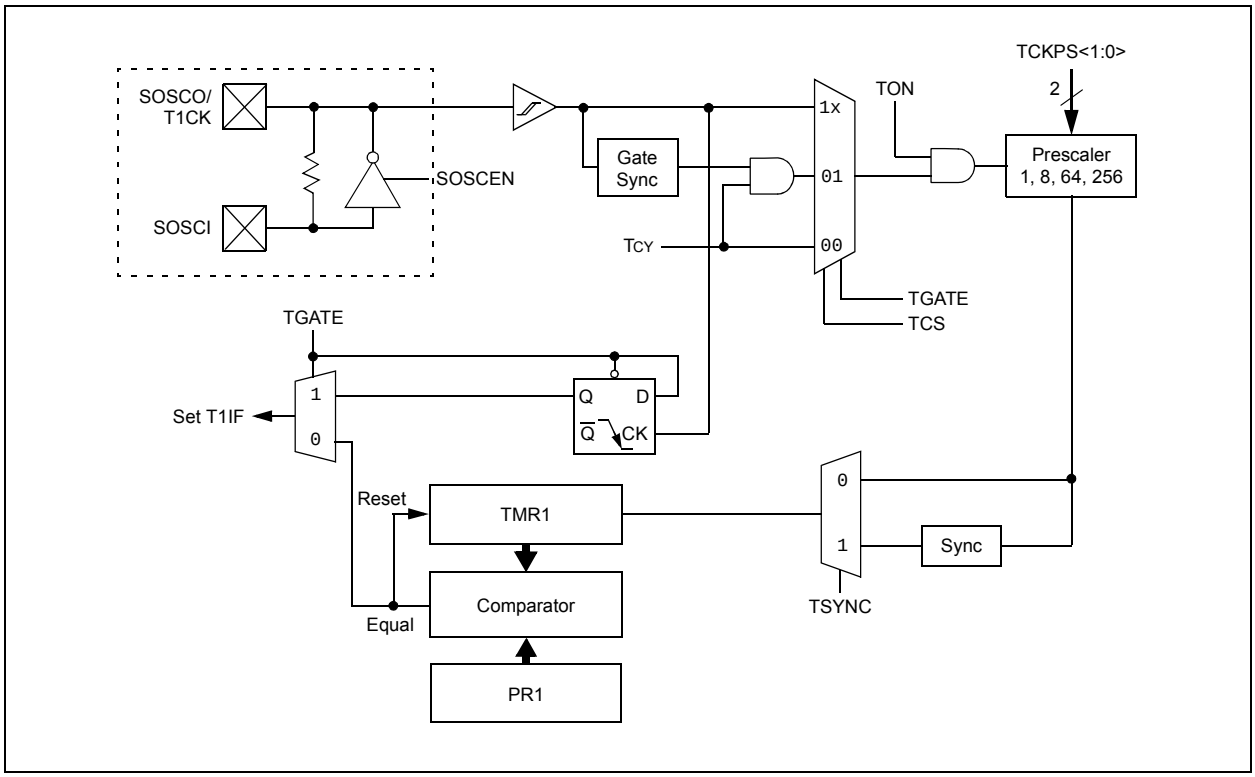
- Timer Gate Operation
- Selectable Prescaler Settings
- Timer Operation during CPU Idle and Sleep modes
- Interrupt on 16-Bit Period Register Match or Falling Edge of External Gate Signal

Figure 11-1 presents a block diagram of the 16-bit Timer1 module.

To configure Timer1 for operation:

1. Set the TON bit (= 1).
2. Select the timer prescaler ratio using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Set or clear the TSYNC bit to configure synchronous or asynchronous operation.
5. Load the timer period value into the PR1 register.
6. If interrupts are required, set the interrupt enable bit, T1IE. Use the priority bits, T1IP<2:0>, to set the interrupt priority.

**FIGURE 11-1: 16-BIT TIMER1 MODULE BLOCK DIAGRAM**



**REGISTER 11-1: T1CON: TIMER1 CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| TON | — | TSIDL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | TGATE | TCKPS1 | TCKPS0 | — | TSYNC | TCS | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **TON:** Timer1 On bit
1 = Starts 16-bit Timer1
0 = Stops 16-bit Timer1

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timer1 Gated Time Accumulation Enable bit
When TCS = 1:
This bit is ignored.
When TCS = 0:
1 = Gated time accumulation enabled
0 = Gated time accumulation disabled

bit 5-4 **TCKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

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

bit 2 **TSYNC:** Timer1 External Clock Input Synchronization Select bit
When TCS = 1:
1 = Synchronize external clock input
0 = Do not synchronize external clock input
When TCS = 0:
This bit is ignored.

bit 1 **TCS:** Timer1 Clock Source Select bit
1 = External clock from T1CK pin (on the rising edge)
0 = Internal clock (FOSC/2)

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

## 12.0 TIMER2/3

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Timers, refer to the *"PIC24F Family Reference Manual"*, **Section 14. "Timers"** (DS39704). |

The Timer2/3 module is a 32-bit timer, which can also be configured as two independent 16-bit timers with selectable operating modes.

As a 32-bit timer, Timer2/3 operates in three modes:

- Two independent 16-bit timers (Timer2 and Timer3) with all 16-bit operating modes (except Asynchronous Counter mode)
- Single 32-bit timer
- Single 32-bit synchronous counter

They also support these features:

- Timer gate operation
- Selectable prescaler settings
- Timer operation during Idle and Sleep modes
- Interrupt on a 32-bit Period register match
- ADC Event Trigger

Individually, both of the 16-bit timers can function as synchronous timers or counters. They also offer the features listed above, except for the ADC event trigger (this is implemented only with Timer3). The operating modes and enabled features are determined by setting the appropriate bit(s) in the T2CON and T3CON registers. T2CON and T3CON are provided in generic form in Register 12-1 and Register 12-2, respectively.

For 32-bit timer/counter operation, Timer2 is the least significant word (lsw) and Timer3 is the most significant word (msw) of the 32-bit timer.

| | |
|---|---|
| **Note:** | For 32-bit operation, T3CON control bits are ignored. Only T2CON control bits are used for setup and control. Timer2 clock and gate inputs are utilized for the 32-bit timer modules, but an interrupt is generated with the Timer3 interrupt flags. |

To configure Timer2/3 for 32-bit operation:

1. Set the T32 bit (T2CON<3> = 1).
2. Select the prescaler ratio for Timer2 using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value. PR3 will contain the msw of the value while PR2 contains the lsw.
5. If interrupts are required, set the interrupt enable bit, T3IE; use the priority bits, T3IP<2:0>, to set the interrupt priority.

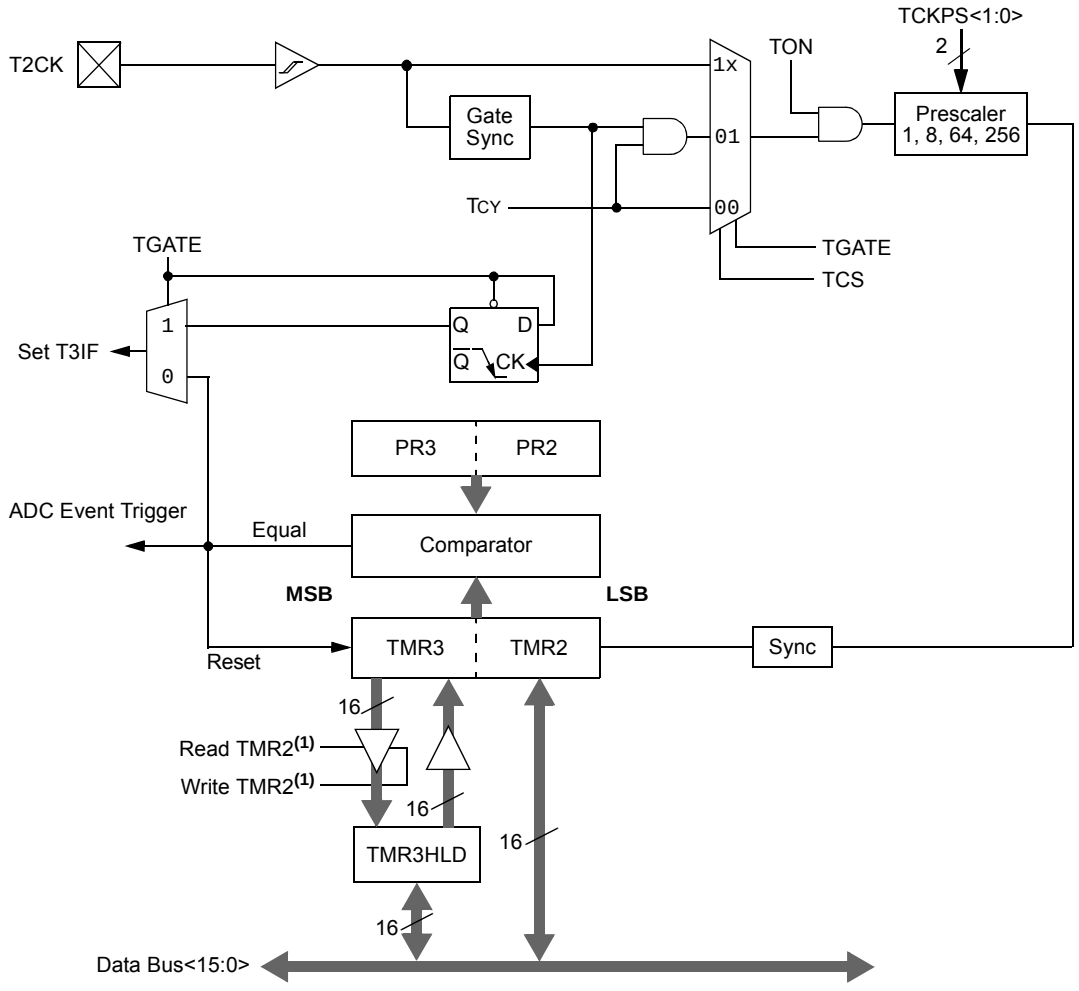
   While Timer2 controls the timer, the interrupt appears as a Timer3 interrupt.
6. Set the TON bit (= 1).

The timer value, at any point, is stored in the register pair, TMR<3:2>. TMR3 always contains the msw of the count, while TMR2 contains the lsw.

To configure any of the timers for individual 16-bit operation:

1. Clear the T32 bit in T2CON<3>.
2. Select the timer prescaler ratio using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value into the PRx register.
5. If interrupts are required, set the interrupt enable bit, TxIE; use the priority bits, TxIP<2:0>, to set the interrupt priority.
6. Set the TON bit (TxCON<15> = 1).

**FIGURE 12-1: TIMER2/3 (32-BIT) BLOCK DIAGRAM**




**Note 1:** The 32-Bit Timer Configuration (T32) bit must be set for 32-bit timer/counter operation. All control bits are respective to the T2CON register.

**FIGURE 12-2: TIMER2 (16-BIT SYNCHRONOUS) BLOCK DIAGRAM**

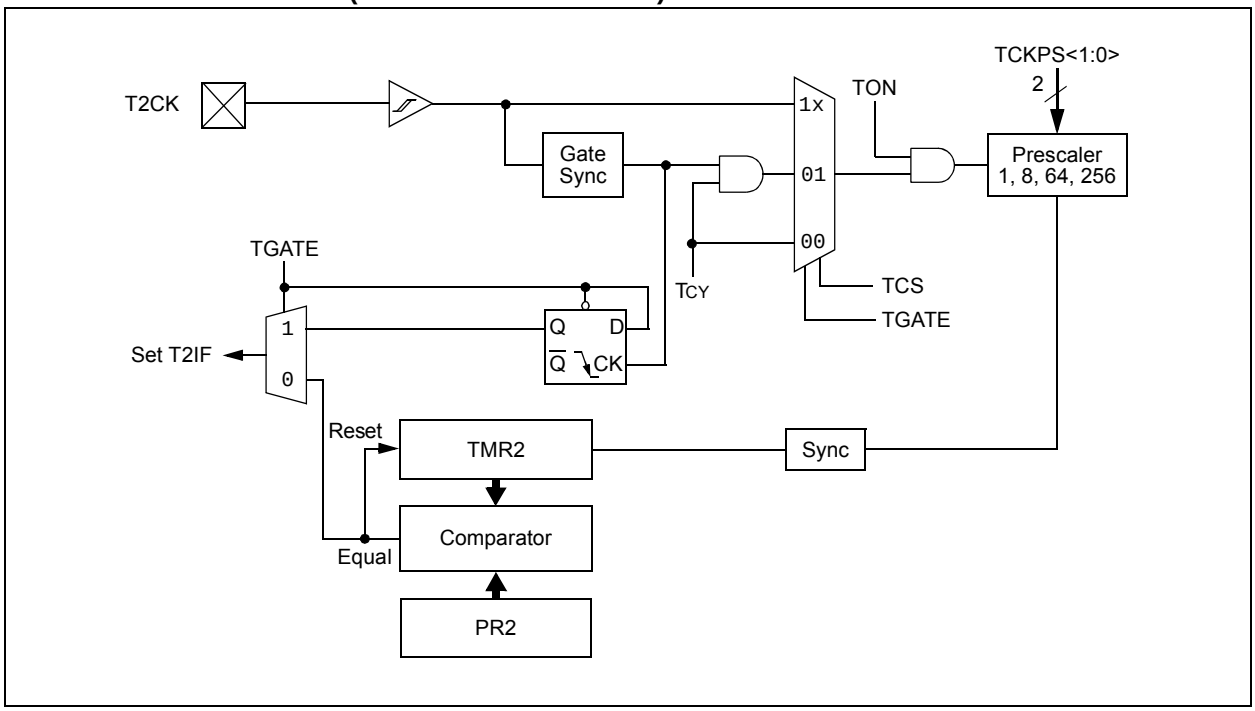
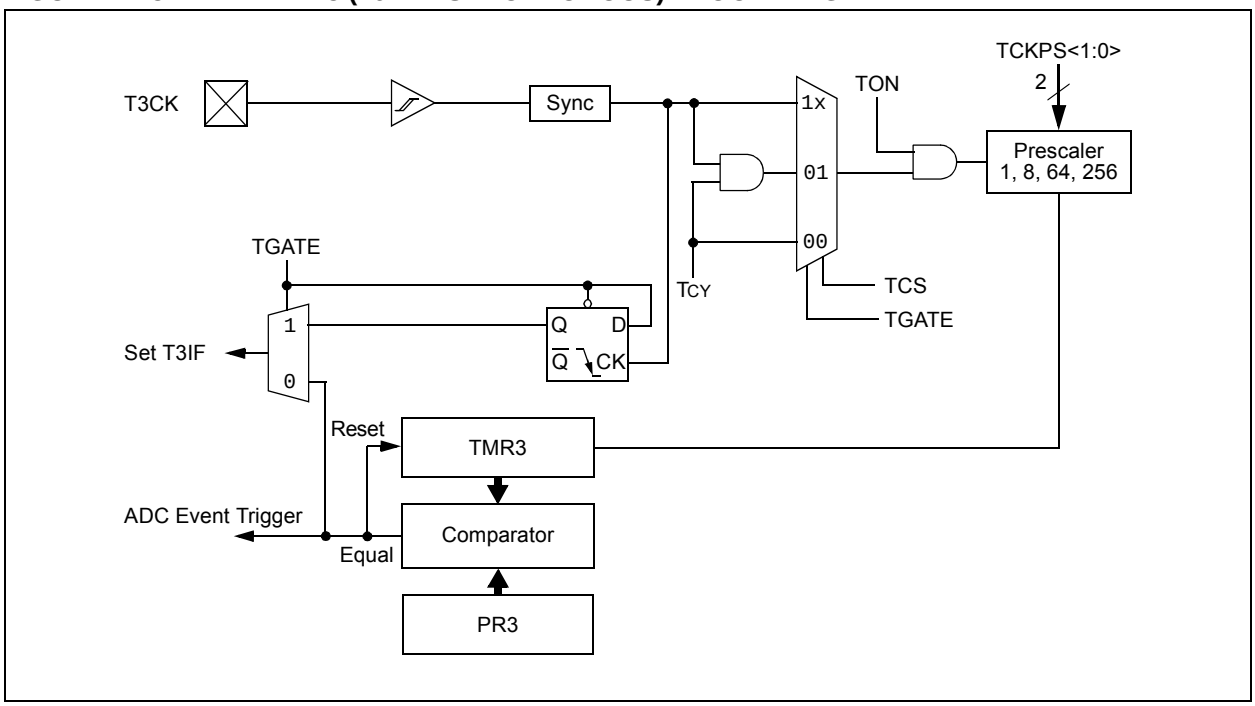



**FIGURE 12-3: TIMER3 (16-BIT SYNCHRONOUS) BLOCK DIAGRAM**



**REGISTER 12-1: T2CON: TIMER2 CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| TON | — | TSIDL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | TGATE | TCKPS1 | TCKPS0 | T32(1) | — | TCS | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **TON:** Timer2 On bit
When T2CON<3> = 1:
1 = Starts 32-bit Timer2/3
0 = Stops 32-bit Timer2/3
When T2CON<3> = 0:
1 = Starts 16-bit Timer2
0 = Stops 16-bit Timer2

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timer2 Gated Time Accumulation Enable bit
When TCS = 1:
This bit is ignored.
When TCS = 0:
1 = Gated time accumulation enabled
0 = Gated time accumulation disabled

bit 5-4 **TCKPS<1:0>:** Timer2 Input Clock Prescale Select bits
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3 **T32:** 32-Bit Timer Mode Select bit(1)
1 = Timer2 and Timer3 form a single 32-bit timer
0 = Timer2 and Timer3 act as two 16-bit timers

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

bit 1 **TCS:** Timer2 Clock Source Select bit
1 = External clock from pin, T2CK (on the rising edge)
0 = Internal clock (FOSC/2)

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

**Note 1:** In 32-bit mode, the T3CON control bits do not affect 32-bit timer operation.

**REGISTER 12-2: T3CON: TIMER3 CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| TON(1) | — | TSIDL(1) | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | TGATE(1) | TCKPS1(1) | TCKPS0(1) | — | — | TCS(1) | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **TON:** Timer3 On bit(1)
1 = Starts 16-bit Timer3
0 = Stops 16-bit Timer3

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit(1)
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timer3 Gated Time Accumulation Enable bit(1)
When TCS = 1:
This bit is ignored.
When TCS = 0:
1 = Gated time accumulation enabled
0 = Gated time accumulation disabled

bit 5-4 **TCKPS<1:0>:** Timer3 Input Clock Prescale Select bits(1)
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **TCS:** Timer3 Clock Source Select bit(1)
1 = External clock from the T3CK pin (on the rising edge)
0 = Internal clock (FOSC/2)

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

**Note 1:** When 32-bit operation is enabled (T2CON<3> = 1), these bits have no effect on Timer3 operation; all timer functions are set through T2CON.

**NOTES:**

## 13.0 INPUT CAPTURE

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Input Capture, refer to the *"PIC24F Family Reference Manual"*, **Section 15. "Input Capture"** (DS39701).

The input capture module is used to capture a timer value from one of two selectable time bases upon an event on an input pin.

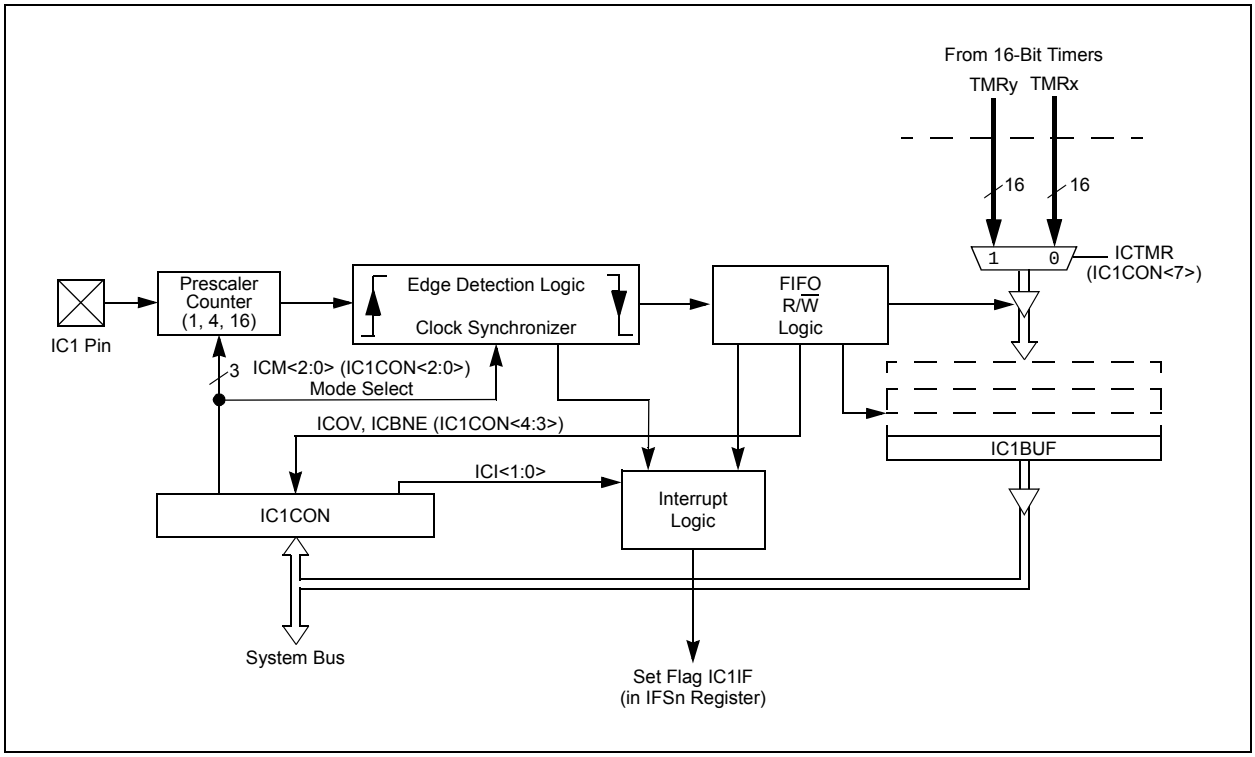
The input capture features are quite useful in applications requiring frequency (Time Period) and pulse measurement. Figure 13-1 depicts a simplified block diagram of the input capture module.

The PIC24F04KA201 family devices have one input capture channel. The input capture module has multiple operating modes, which are selected via the IC1CON register. The operating modes include:

- Capture timer value on every falling edge of input applied at the IC1 pin
- Capture timer value on every rising edge of input applied at the IC1 pin
- Capture timer value on every 4th rising edge of input applied at the IC1 pin
- Capture timer value on every 16th rising edge of input applied at the IC1 pin
- Capture timer value on every rising and every falling edge of input applied at the IC1 pin
- Device wake-up from capture pin during CPU Sleep and Idle modes

The input capture module has a four-level FIFO buffer. The number of capture events required to generate a CPU interrupt can be selected by the user.

**FIGURE 13-1: INPUT CAPTURE BLOCK DIAGRAM**



## 13.1 Input Capture Registers

**REGISTER 13-1: IC1CON: INPUT CAPTURE 1 CONTROL REGISTER**

| U-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | ICSIDL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R-0, HC | R-0, HC | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ICTMR | ICI1 | ICI0 | ICOV | ICBNE | ICM2 | ICM1 | ICM0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **ICSIDL:** Input Capture 1 Module Stop in Idle Control bit
1 = Input capture module will halt in CPU Idle mode
0 = Input capture module will continue to operate in CPU Idle mode

bit 12-8 **Unimplemented:** Read as '0'

bit 7 **ICTMR:** Input Capture 1 Timer Select bit
1 = TMR2 contents are captured on capture event
0 = TMR3 contents are captured on capture event

bit 6-5 **ICI<1:0>:** Select Number of Captures per Interrupt bits
11 = Interrupt on every fourth capture event
10 = Interrupt on every third capture event
01 = Interrupt on every second capture event
00 = Interrupt on every capture event

bit 4 **ICOV:** Input Capture 1 Overflow Status Flag bit (read-only)
1 = Input capture overflow occurred
0 = No input capture overflow occurred

bit 3 **ICBNE:** Input Capture 1 Buffer Empty Status bit (read-only)
1 = Input capture buffer is not empty, at least one more capture value can be read
0 = Input capture buffer is empty

bit 2-0 **ICM<2:0>:** Input Capture 1 Mode Select bits
111 = Input capture functions as interrupt pin only when device is in Sleep or Idle mode (rising edge detect only, all other control bits are not applicable)
110 = Unused (module disabled)
101 = Capture mode, every 16th rising edge
100 = Capture mode, every 4th rising edge
011 = Capture mode, every rising edge
010 = Capture mode, every falling edge
001 = Capture mode, every edge (rising and falling) – ICI<1:0> bits do not control interrupt generation for this mode
000 = Input capture module turned off

## 14.0 OUTPUT COMPARE

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Output Compare, refer to the *"PIC24F Family Reference Manual"*, **Section 16. "Output Compare"** (DS39706).

### 14.1 Setup for Single Output Pulse Generation

When the OCM control bits (OC1CON<2:0>) are set to '100', the selected output compare channel initializes the OC1 pin to the low state and generates a single output pulse.

To generate a single output pulse, the following steps are required (these steps assume the timer source is initially turned off, but this is not a requirement for the module operation):

1. Determine the instruction clock cycle time. Take into account the frequency of the external clock to the timer source (if one is used) and the timer prescaler settings.
2. Calculate time to the rising edge of the output pulse relative to the TMRy start value (0000h).
3. Calculate the time to the falling edge of the pulse based on the desired pulse width and the time to the rising edge of the pulse.
4. Write the values computed in steps 2 and 3 above into the Output Compare 1 register, OC1R, and the Output Compare 1 Secondary register, OC1RS, respectively.
5. Set Timer Period register, PRy, to value equal to or greater than the value in OC1RS, the Output Compare 1 Secondary register.
6. Set the OCM bits to '100' and the OCTSEL (OC1CON<3>) bit to the desired timer source. The OC1 pin state will now be driven low.
7. Set the TON (TyCON<15>) bit to '1', which enables the compare time base to count.
8. Upon the first match between TMRy and OC1R, the OC1 pin will be driven high.
9. When the incrementing timer, TMRy, matches the Output Compare 1 Secondary register, OC1RS, the second and trailing edge (high-to-low) of the pulse is driven onto the OC1 pin. No additional pulses are driven onto the OC1 pin and it remains low. As a result of the second compare match event, the OC1IF interrupt flag bit is set, which will result in an interrupt if it is enabled, by setting the OC1IE bit. For further information on peripheral interrupts, refer to **Section 7.0 "Interrupt Controller"**.
10. To initiate another single pulse output, change the Timer and Compare register settings, if needed, and then issue a write to set the OCM bits to '100'. Disabling and re-enabling of the timer and clearing the TMRy register are not required, but may be advantageous for defining a pulse from a known event time boundary.

The output compare module does not have to be disabled after the falling edge of the output pulse. Another pulse can be initiated by rewriting the value of the OC1CON register.

### 14.2 Setup for Continuous Output Pulse Generation

When the OCM control bits (OC1CON<2:0>) are set to '101', the selected output compare channel initializes the OC1 pin to the low state and generates output pulses on each and every compare match event.

For the user to configure the module for the generation of a continuous stream of output pulses, the following steps are required (these steps assume the timer source is initially turned off, but this is not a requirement for the module operation):

1. Determine the instruction clock cycle time. Take into account the frequency of the external clock to the timer source (if one is used) and the timer prescaler settings.
2. Calculate time to the rising edge of the output pulse relative to the TMRy start value (0000h).
3. Calculate the time to the falling edge of the pulse based on the desired pulse width and the time to the rising edge of the pulse.
4. Write the values computed in step 2 and 3 above into the Output Compare 1 register, OC1R, and the Output Compare 1 Secondary register, OC1RS, respectively.
5. Set the Timer Period register, PRy, to a value equal to or greater than the value in OC1RS.
6. Set the OCM bits to '101' and the OCTSEL bit to the desired timer source. The OC1 pin state will now be driven low.
7. Enable the compare time base by setting the TON (TyCON<15>) bit to '1'.
8. Upon the first match between TMRy and OC1R, the OC1 pin will be driven high.
9. When the compare time base, TMRy, matches the OC1RS, the second and trailing edge (high-to-low) of the pulse is driven onto the OC1 pin.
10. As a result of the second compare match event, the OC1IF interrupt flag bit is set.
11. When the compare time base and the value in its respective Timer Period register match, the TMRy register resets to 0x0000 and resumes counting.
12. Steps 8 through 11 are repeated and a continuous stream of pulses is generated indefinitely. The OC1IF flag is set on each OC1RS/TMRy compare match event.

## 14.3 Pulse-Width Modulation (PWM) Mode

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

1. Set the PWM period by writing to the selected Timer Period register (PRy).
2. Set the PWM duty cycle by writing to the OC1RS register.
3. Write the OC1R register with the initial duty cycle.
4. Enable interrupts, if required, for the timer and output compare modules. The output compare interrupt is required for PWM Fault pin utilization.
5. Configure the output compare module for one of two PWM Operation modes by writing to the Output Compare Mode bits, OCM<2:0> (OC1CON<2:0>).
6. Set the TMRy prescale value and enable the time base by setting TON (TxCON<15>) = 1.

**Note:** The OC1R register should be initialized before the output compare module is first enabled. The OC1R register becomes a read-only Duty Cycle register when the module is operated in the PWM modes. The value held in OC1R will become the PWM duty cycle for the first PWM period. The contents of the Output Compare 1 Secondary register, OC1RS, will not be transferred into OC1R until a time base period match occurs.

### 14.3.1 PWM PERIOD

The PWM period is specified by writing to PRy, the Timer Period register. The PWM period can be calculated using Equation 14-1.

**EQUATION 14-1: CALCULATING THE PWM PERIOD(1)**

PWM Period = [(PRy) + 1] • TCY • (Timer Prescale Value)

where:

PWM Frequency = 1/[PWM Period]

**Note 1:** Based on TCY = 2 * TOSC, Doze mode and PLL are disabled.

**Note:** A PRy value of N will produce a PWM period of N + 1 time base count cycles. For example, a value of 7 written into the PRy register will yield a period consisting of 8 time base cycles.

### 14.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the OC1RS register. The OC1RS register can be written to at any time, but the duty cycle value is not latched into OC1R until a match between PRy and TMRy occurs (i.e., the period is complete). This provides a double buffer for the PWM duty cycle and is essential for glitchless PWM operation. In PWM mode, OC1R is a read-only register.

Some important boundary parameters of the PWM duty cycle include:

- If the Output Compare 1 register, OC1R, is loaded with 0000h, the OC1 pin will remain low (0% duty cycle).
- If OC1R is greater than PRy (Timer Period register), the pin will remain high (100% duty cycle).
- If OC1R is equal to PRy, the OC1 pin will be low for one time base count value and high for all other count values.

See Example 14-1 for PWM mode timing details. Table 14-1 provides an example of PWM frequencies and resolutions for a device operating at 10 MIPS.

**EQUATION 14-2: CALCULATION FOR MAXIMUM PWM RESOLUTION(1)**

$$\text{Maximum PWM Resolution (bits)} = \frac{\log_{10}\left(\frac{F_{CY}}{F_{PWM} \bullet (\text{Timer Prescale Value})}\right)}{\log_{10}(2)} \text{ bits}$$

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

**EXAMPLE 14-1: PWM PERIOD AND DUTY CYCLE CALCULATIONS(1)**

1. Find the Timer Period register value for a desired PWM frequency of 52.08 kHz, where FOSC = 8 MHz with PLL (32 MHz device clock rate) and a Timer2 prescaler setting of 1:1.

   TCY = 2 * TOSC = 62.5 ns

   PWM Period = 1/PWM Frequency = 1/52.08 kHz = 19.2 μs

   PWM Period = (PR2 + 1) • TCY • (Timer 2 Prescale Value)

   19.2 μs = (PR2 + 1) • 62.5 ns • 1

   PR2 = 306

2. Find the maximum resolution of the duty cycle that can be used with a 52.08 kHz frequency and a 32 MHz device clock rate:

   PWM Resolution = $\log_{10}(FCY/FPWM)/\log_{10}2$) bits

   = ($\log_{10}$(16 MHz/52.08 kHz)/$\log_{10}$2) bits

   = 8.3 bits

**Note 1:** Based on TCY = 2 * TOSC, Doze mode and PLL are disabled.

**TABLE 14-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 4 MIPS (FCY = 4 MHz)(1)**

| PWM Frequency | 7.6 Hz | 61 Hz | 122 Hz | 977 Hz | 3.9 kHz | 31.3 kHz | 125 kHz |
|---|---|---|---|---|---|---|---|
| Timer Prescaler Ratio | 8 | 1 | 1 | 1 | 1 | 1 | 1 |
| Period Register Value | FFFFh | FFFFh | 7FFFh | 0FFFh | 03FFh | 007Fh | 001Fh |
| Resolution (bits) | 16 | 16 | 15 | 12 | 10 | 7 | 5 |

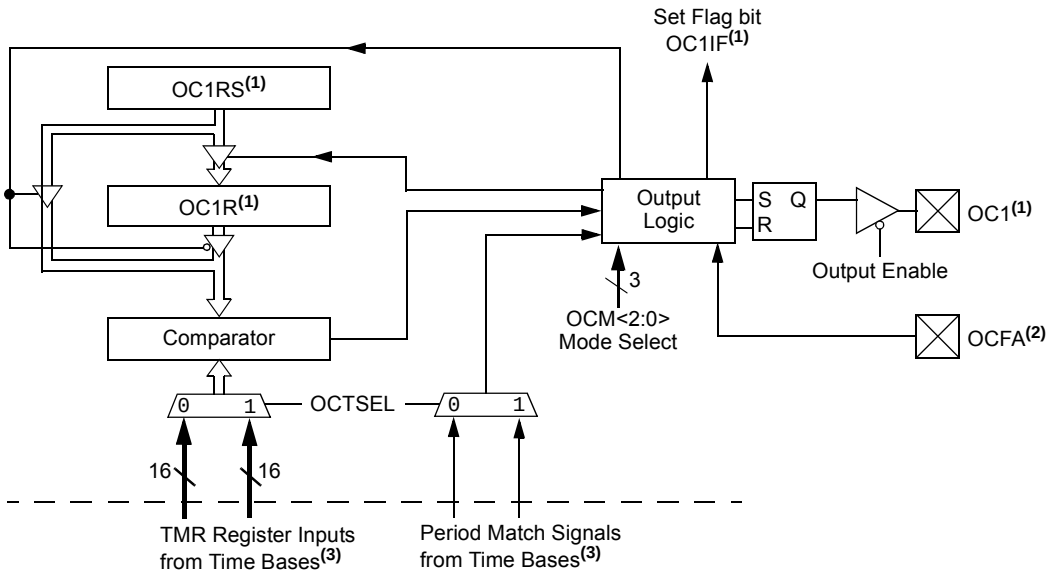
**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

**TABLE 14-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 16 MIPS (FCY = 16 MHz)(1)**

| PWM Frequency | 30.5 Hz | 244 Hz | 488 Hz | 3.9 kHz | 15.6 kHz | 125 kHz | 500 kHz |
|---|---|---|---|---|---|---|---|
| Timer Prescaler Ratio | 8 | 1 | 1 | 1 | 1 | 1 | 1 |
| Period Register Value | FFFFh | FFFFh | 7FFFh | 0FFFh | 03FFh | 007Fh | 001Fh |
| Resolution (bits) | 16 | 16 | 15 | 12 | 10 | 7 | 5 |

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

**FIGURE 14-1: OUTPUT COMPARE MODULE BLOCK DIAGRAM**




**Note 1:** Where 'x' is depicted, reference is made to the registers associated with the respective Output Compare Channel 1.

**2:** OCFA pin controls OC1 channel.

**3:** Each output compare channel can use one of two selectable time bases. Refer to the device data sheet for the time bases associated with the module.

## 14.4 Output Compare Register

**REGISTER 14-1: OC1CON: OUTPUT COMPARE 1 CONTROL REGISTER**

| U-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | OCSIDL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R-0, HC | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | OCFLT | OCTSEL | OCM2 | OCM1 | OCM0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **OCSIDL:** Stop Output Compare 1 in Idle Mode Control bit
1 = Output Compare 1 will halt in CPU Idle mode
0 = Output Compare 1 will continue to operate in CPU Idle mode

bit 12-5 **Unimplemented:** Read as '0'

bit 4 **OCFLT:** PWM Fault Condition Status bit
1 = PWM Fault condition has occurred (cleared in HW only)
0 = No PWM Fault condition has occurred (this bit is only used when OCM<2:0> = 111)

bit 3 **OCTSEL:** Output Compare 1 Timer Select bit
1 = Timer3 is the clock source for Output Compare 1
0 = Timer2 is the clock source for Output Compare 1
Refer to the device data sheet for specific time bases available to the output compare module.

bit 2-0 **OCM<2:0>:** Output Compare 1 Mode Select bits
111 = PWM mode on OC1, Fault pin; OFCA enabled(1)
110 = PWM mode on OC1, Fault pin; OFCA disabled(1)
101 = Initialize OC1 pin low, generate continuous output pulses on OC1 pin
100 = Initialize OC1 pin low, generate single output pulse on OC1 pin
011 = Compare event toggles OC1 pin
010 = Initialize OC1 pin high, compare event forces OC1 pin low
001 = Initialize OC1 pin low, compare event forces OC1 pin high
000 = Output compare channel is disabled

**Note 1:** OCFA pin controls OC1 channel.

**REGISTER 14-2: PADCFG1: PAD CONFIGURATION CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | SMBUSDEL(2) | OC1TRIS(1) | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-5 **Unimplemented:** Read as ‘0’

bit 3 **OC1TRIS:** OC1 Output Tri-State Select bit(1)
1 = OC1 output will not be active on the pin; OCPWM1 can still be used for internal triggers
0 = OC1 output will be active on the pin based on the OCPWM1 module settings

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

**Note 1:** To enable the actual OC1 output, the OCPWM1 module has to be enabled.
**2:** Bit 4 is described in **Section 16.0 “Inter-Integrated Circuit (I2C™)”**.

## 15.0 SERIAL PERIPHERAL INTERFACE (SPI)

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Serial Peripheral Interface, refer to the *"PIC24F Family Reference Manual"*, **Section 23. "Serial Peripheral Interface (SPI)"** (DS39699). |

The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial data EEPROMs, shift registers, display drivers, A/D Converters, etc. The SPI module is compatible with Motorola's SPI and SIOP interfaces.

The module supports operation in two buffer modes. In Standard mode, data is shifted through a single serial buffer. In Enhanced Buffer mode, data is shifted through an 8-level FIFO buffer.

| | |
|---|---|
| **Note:** | Do not perform read-modify-write operations (such as bit-oriented instructions) on the SPI1BUF register in either Standard or Enhanced Buffer mode. |

The module also supports a basic framed SPI protocol while operating in either Master or Slave mode. A total of four framed SPI configurations are supported.

The SPI serial interface consists of four pins:

- SDI1: Serial Data Input
- SDO1: Serial Data Output
- SCK1: Shift Clock Input or Output
- $\overline{SS1}$: Active-Low Slave Select or Frame Synchronization I/O Pulse

The SPI module can be configured to operate using 2, 3 or 4 pins. In the 3-pin mode, $\overline{SS1}$ is not used. In the 2-pin mode, both SDO1 and $\overline{SS1}$ are not used.

Block diagrams of the module in Standard and Enhanced Buffer modes are displayed in Figure 15-1 and Figure 15-2.

The devices of the PIC24F04KA201 family offer one SPI module on a device.

| | |
|---|---|
| **Note:** | In this section, the SPI module is referred to as SPI1, or separately as SPI1. Special Function Registers (SFRs) will follow a similar notation. For example, SPI1CON1 or SPI1CON2 refers to the control register for the SPI1 module. |

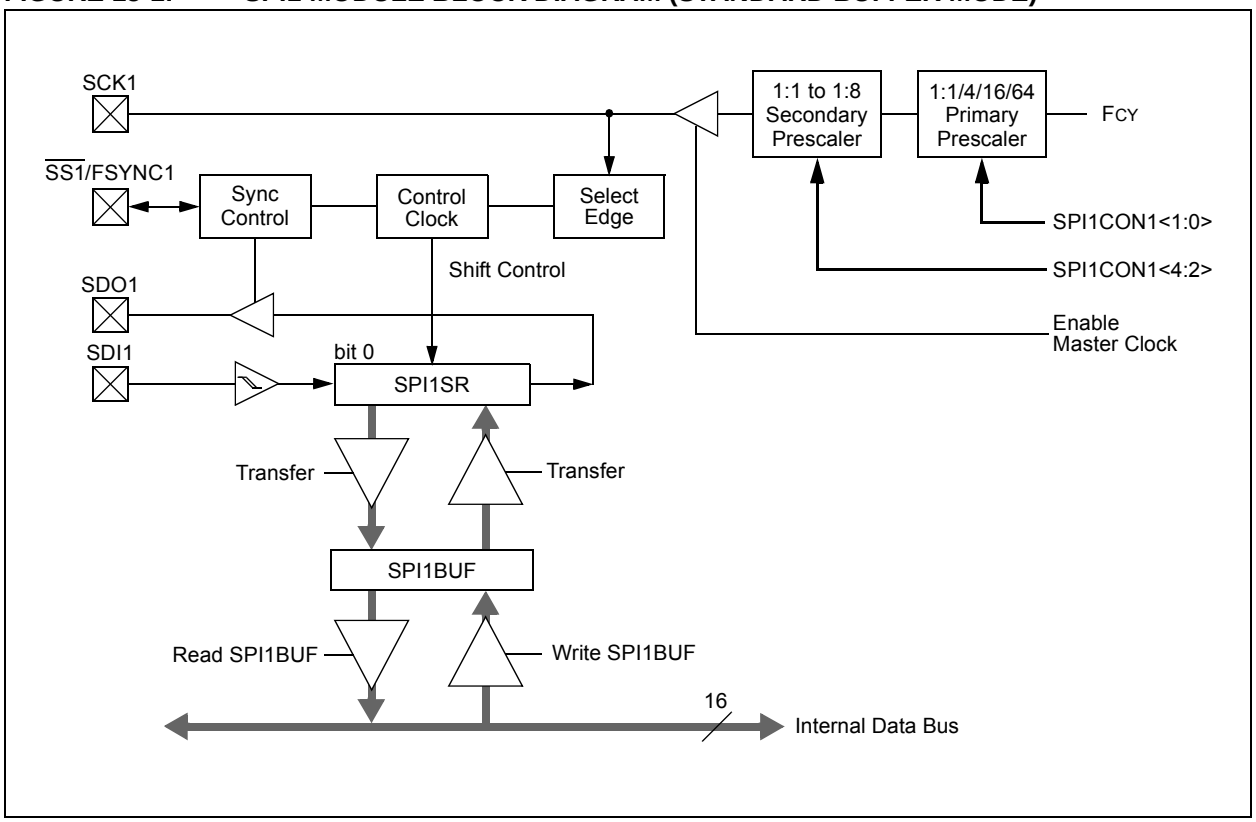
To set up the SPI module for the Standard Master mode of operation:

1. If using interrupts:
   a) Clear the respective SPI1IF bit in the IFS0 register.
   b) Set the respective SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IPx bits in the IPC2 register to set the interrupt priority.
2. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 1.
3. Clear the SPIROV bit (SPI1STAT<6>).
4. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).
5. Write the data to be transmitted to the SPI1BUF register. Transmission (and reception) will start as soon as data is written to the SPI1BUF register.

To set up the SPI module for the Standard Slave mode of operation:

1. Clear the SPI1BUF register.
2. If using interrupts:
   a) Clear the respective SPI1IF bit in the IFS0 register.
   b) Set the respective SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IP bits in the IPC2 register to set the interrupt priority.
3. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 0.
4. Clear the SMP bit.
5. If the CKE bit is set, then the SSEN bit (SPI1CON1<7>) must be set to enable the $\overline{SS1}$ pin.
6. Clear the SPIROV bit (SPI1STAT<6>).
7. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).

**FIGURE 15-1: SPI1 MODULE BLOCK DIAGRAM (STANDARD BUFFER MODE)**



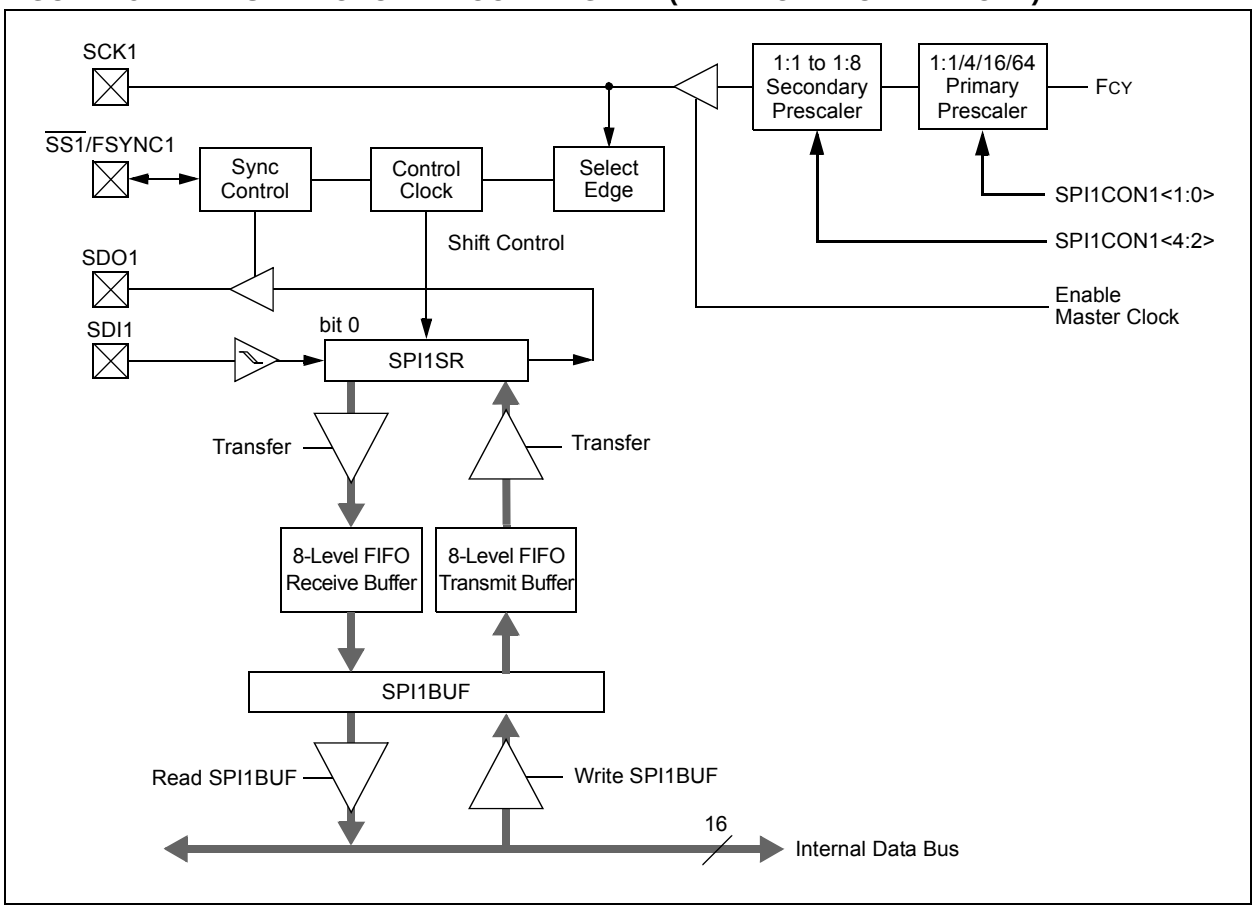
To set up the SPI module for the Enhanced Buffer Master (EBM) mode of operation:

1. If using interrupts:
   a) Clear the respective SPI1IF bit in the IFS0 register.
   b) Set the respective SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IPx bits in the IPC2 register.
2. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 1.
3. Clear the SPIROV bit (SPI1STAT<6>).
4. Select Enhanced Buffer mode by setting the SPIBEN bit (SPI1CON2<0>).
5. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).
6. Write the data to be transmitted to the SPI1BUF register. Transmission (and reception) will start as soon as data is written to the SPI1BUF register.

To set up the SPI module for the Enhanced Buffer Slave mode of operation:

1. Clear the SPI1BUF register.
2. If using interrupts:
   a) Clear the respective SPI1IF bit in the IFS0 register.
   b) Set the respective SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IPx bits in the IPC2 register to set the interrupt priority.
3. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 0.
4. Clear the SMP bit.
5. If the CKE bit is set, then the SSEN bit must be set, thus enabling the $\overline{SS1}$ pin.
6. Clear the SPIROV bit (SPI1STAT<6>).
7. Select Enhanced Buffer mode by setting the SPIBEN bit (SPI1CON2<0>).
8. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).

**FIGURE 15-2: SPI1 MODULE BLOCK DIAGRAM (ENHANCED BUFFER MODE)**



**REGISTER 15-1: SPI1STAT: SPI1 STATUS AND CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | R-0, HSC | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| SPIEN | — | SPISIDL | — | — | SPIBEC2 | SPIBEC1 | SPIBEC0 |
| bit 15 | | | | | | | bit 8 |

| R-0,HSC | R/C-0, HS | R/W-0, HSC | R/W-0 | R/W-0 | R/W-0 | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| SRMPT | SPIROV | SRXMPT | SISEL2 | SISEL1 | SISEL0 | SPITBF | SPIRBF |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **SPIEN:** SPI1 Enable bit
1 = Enables module and configures SCK1, SDO1, SDI1 and $\overline{SS1}$ as serial port pins
0 = Disables module

bit 14 **Unimplemented:** Read as '0'

bit 13 **SPISIDL:** Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 12-11 **Unimplemented:** Read as '0'

bit 10-8 **SPIBEC<2:0>:** SPI1 Buffer Element Count bits (valid in Enhanced Buffer mode)
Master mode:
Number of SPI transfers pending.
Slave mode:
Number of SPI transfers unread.

bit 7 **SRMPT:** Shift Register (SPI1SR) Empty bit (valid in Enhanced Buffer mode)
1 = SPI1 Shift register is empty and ready to send or receive
0 = SPI1 Shift register is not empty

bit 6 **SPIROV:** Receive Overflow Flag bit
1 = A new byte/word is completely received and discarded. The user software has not read the previous data in the SPI1BUF register.
0 = No overflow has occurred

bit 5 **SRXMPT:** Receive FIFO Empty bit (valid in Enhanced Buffer mode)
1 = Receive FIFO is empty
0 = Receive FIFO is not empty

bit 4-2 **SISEL<2:0>:** SPI1 Buffer Interrupt Mode bits (valid in Enhanced Buffer mode)
111 = Interrupt when SPI1 transmit buffer is full (SPITBF bit is set)
110 = Interrupt when last bit is shifted into SPI1SR; as a result, the TX FIFO is empty
101 = Interrupt when the last bit is shifted out of SPI1SR; now the transmit is complete
100 = Interrupt when one data byte is shifted into the SPI1SR; as a result, the TX FIFO has one open spot
011 = Interrupt when SPI1 receive buffer is full (SPIRBF bit set)
010 = Interrupt when SPI1 receive buffer is 3/4 or more full
001 = Interrupt when data is available in receive buffer (SRMPT bit is set)
000 = Interrupt when the last data in the receive buffer is read; as a result, the buffer is empty (SRXMPT bit is set)

**REGISTER 15-1: SPI1STAT: SPI1 STATUS AND CONTROL REGISTER (CONTINUED)**

bit 1 **SPITBF:** SPI1 Transmit Buffer Full Status bit

1 = Transmit not yet started, SPI1TXB is full
0 = Transmit started, SPI1TXB is empty

In Standard Buffer mode:
Automatically set in hardware when CPU writes SPI1BUF location, loading SPI1TXB.
Automatically cleared in hardware when SPI1 module transfers data from SPI1TXB to SPI1SR.

In Enhanced Buffer mode:
Automatically set in hardware when CPU writes SPI1BUF location, loading the last available buffer location.
Automatically cleared in hardware when a buffer location is available for a CPU write.

bit 0 **SPIRBF:** SPI1 Receive Buffer Full Status bit

1 = Receive complete, SPI1RXB is full
0 = Receive is not complete, SPI1RXB is empty

In Standard Buffer mode:
Automatically set in hardware when SPI1 transfers data from SPI1SR to SPI1RXB.
Automatically cleared in hardware when core reads SPI1BUF location, reading SPI1RXB.

In Enhanced Buffer mode:
Automatically set in hardware when SPI1 transfers data from SPI1SR to buffer, filling the last unread buffer location.
Automatically cleared in hardware when a buffer location is available for a transfer from SPI1SR.

**REGISTER 15-2: SPI1CON1: SPI1 CONTROL REGISTER 1**

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | DISSCK | DISSDO | MODE16 | SMP | CKE(1) |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-13 **Unimplemented:** Read as '0'

bit 12 **DISSCK:** Disable SCK1 pin bit (SPI Master modes only)
1 = Internal SPI clock is disabled, pin functions as I/O
0 = Internal SPI clock is enabled

bit 11 **DISSDO:** Disables SDO1 pin bit
1 = SDO1 pin is not used by module; pin functions as I/O
0 = SDO1 pin is controlled by the module

bit 10 **MODE16:** Word/Byte Communication Select bit
1 = Communication is word-wide (16 bits)
0 = Communication is byte-wide (8 bits)

bit 9 **SMP:** SPI1 Data Input Sample Phase bit
Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
Slave mode:
SMP must be cleared when SPI1 is used in Slave mode.

bit 8 **CKE:** SPI1 Clock Edge Select bit(1)
1 = Serial output data changes on transition from active clock state to Idle clock state (see bit 6)
0 = Serial output data changes on transition from Idle clock state to active clock state (see bit 6)

bit 7 **SSEN:** Slave Select Enable bit (Slave mode)
1 = $\overline{SS1}$ pin used for Slave mode
0 = $\overline{SS1}$ pin not used by module; pin controlled by port function

bit 6 **CKP:** Clock Polarity Select bit
1 = Idle state for clock is a high level; active state is a low level
0 = Idle state for clock is a low level; active state is a high level

bit 5 **MSTEN:** Master Mode Enable bit
1 = Master mode
0 = Slave mode

bit 4-2 **SPRE<2:0>:** Secondary Prescale bits (Master mode)
111 = Secondary prescale 1:1
110 = Secondary prescale 2:1
.
.
.
000 = Secondary prescale 8:1

**Note 1:** The CKE bit is not used in the Framed SPI modes. The user should program this bit to '0' for the Framed SPI modes (FRMEN = 1).

**REGISTER 15-2: SPI1CON1: SPI1 CONTROL REGISTER 1 (CONTINUED)**

bit 1-0 **PPRE<1:0>:** Primary Prescale bits (Master mode)
11 = Primary prescale 1:1
10 = Primary prescale 4:1
01 = Primary prescale 16:1
00 = Primary prescale 64:1

**Note 1:** The CKE bit is not used in the Framed SPI modes. The user should program this bit to '0' for the Framed SPI modes (FRMEN = 1).

**REGISTER 15-3: SPI1CON2: SPI1 CONTROL REGISTER 2**

| R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| FRMEN | SPIFSD | SPIFPOL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | SPIFE | SPIBEN |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **FRMEN:** Framed SPI1 Support bit
1 = Framed SPI1 support enabled
0 = Framed SPI1 support disabled

bit 14 **SPIFSD:** Frame Sync Pulse Direction Control on $\overline{SS1}$ Pin bit
1 = Frame sync pulse input (slave)
0 = Frame sync pulse output (master)

bit 13 **SPIFPOL:** Frame Sync Pulse Polarity bit (Frame mode only)
1 = Frame sync pulse is active-high
0 = Frame sync pulse is active-low

bit 12-2 **Unimplemented:** Read as '0'

bit 1 **SPIFE:** Frame Sync Pulse Edge Select bit
1 = Frame sync pulse coincides with first bit clock
0 = Frame sync pulse precedes first bit clock

bit 0 **SPIBEN:** Enhanced Buffer Enable bit
1 = Enhanced Buffer enabled
0 = Enhanced Buffer disabled (Legacy mode)

**EQUATION 15-1: RELATIONSHIP BETWEEN DEVICE AND SPI CLOCK SPEED(1)**

$$FSCK = \frac{FCY}{\text{Primary Prescaler} * \text{Secondary Prescaler}}$$

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

**TABLE 15-1: SAMPLE SCK FREQUENCIES(1,2)**

| FCY = 16 MHz | | Secondary Prescaler Settings | | | | |
|---|---|---|---|---|---|---|
| | | 1:1 | 2:1 | 4:1 | 6:1 | 8:1 |
| Primary Prescaler Settings | 1:1 | Invalid | 8000 | 4000 | 2667 | 2000 |
| | 4:1 | 4000 | 2000 | 1000 | 667 | 500 |
| | 16:1 | 1000 | 500 | 250 | 167 | 125 |
| | 64:1 | 250 | 125 | 63 | 42 | 31 |
| **FCY = 5 MHz** | | | | | | |
| Primary Prescaler Settings | 1:1 | 5000 | 2500 | 1250 | 833 | 625 |
| | 4:1 | 1250 | 625 | 313 | 208 | 156 |
| | 16:1 | 313 | 156 | 78 | 52 | 39 |
| | 64:1 | 78 | 39 | 20 | 13 | 10 |

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.
**2:** SCK1 frequencies indicated in kHz.

## 16.0 INTER-INTEGRATED CIRCUIT (I²C™)

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Inter-Integrated Circuit, refer to the *"PIC24F Family Reference Manual"*, **Section 24. "Inter-Integrated Circuit (I²C™)"** (DS39702).

The Inter-Integrated Circuit (I²C™) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial data EEPROMs, display drivers, A/D Converters, etc.

The I²C module supports these features:

- Independent master and slave logic
- 7-bit and 10-bit device addresses
- General call address, as defined in the I²C protocol
- Clock stretching to provide delays for the processor to respond to a slave data request
- Both 100 kHz and 400 kHz bus specifications
- Configurable address masking
- Multi-Master modes to prevent loss of messages in arbitration
- Bus Repeater mode, allowing the acceptance of all messages as a slave regardless of the address
- Automatic SCL

Figure 16-1 illustrates a block diagram of the module.

### 16.1 Pin Remapping Options

The I²C module is tied to a fixed pin. To allow flexibility with peripheral multiplexing, the I2C1 module in 20-pin devices can be reassigned to the alternate pins, designated as SCL1 and SDA1 during device configuration.

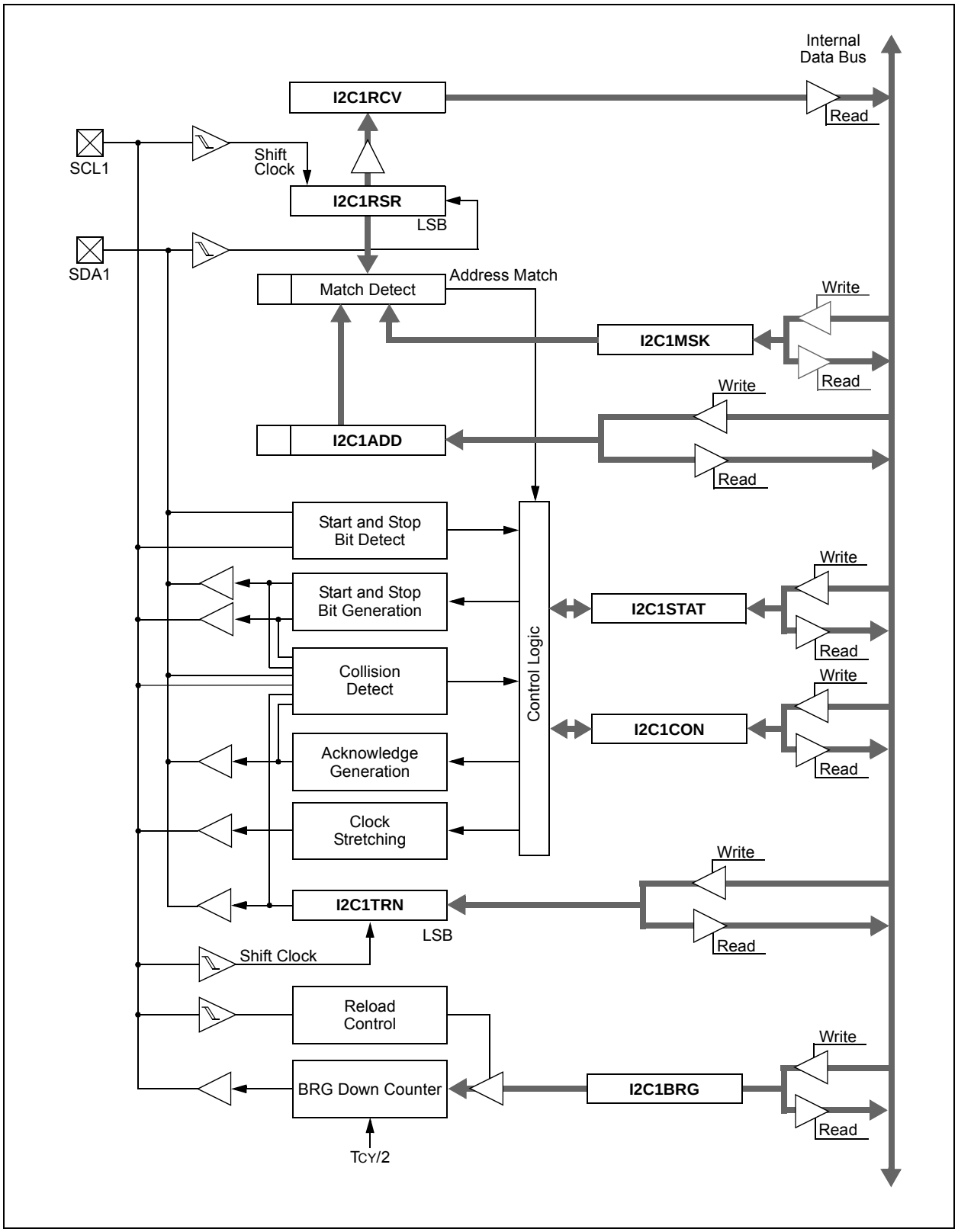
Pin assignment is controlled by the I2C1SEL Configuration bit. Programming this bit (= 0) multiplexes the module to the SCL1 and SDA1 pins.

### 16.2 Communicating as a Master in a Single Master Environment

The details of sending a message in Master mode depends on the communications protocol for the device being communicated with. Typically, the sequence of events is as follows:

1. Assert a Start condition on SDA1 and SCL1.
2. Send the I²C device address byte to the slave with a write indication.
3. Wait for and verify an Acknowledge from the slave.
4. Send the first data byte (sometimes known as the command) to the slave.
5. Wait for and verify an Acknowledge from the slave.
6. Send the serial memory address low byte to the slave.
7. Repeat steps 4 and 5 until all data bytes are sent.
8. Assert a Repeated Start condition on SDA1 and SCL1.
9. Send the device address byte to the slave with a read indication.
10. Wait for and verify an Acknowledge from the slave.
11. Enable master reception to receive serial memory data.
12. Generate an ACK or NACK condition at the end of a received byte of data.
13. Generate a Stop condition on SDA1 and SCL1.

**FIGURE 16-1: I²C™ BLOCK DIAGRAM**



## 16.3 Setting Baud Rate When Operating as a Bus Master

To compute the Baud Rate Generator (BRG) reload value, use Equation 16-1.

**EQUATION 16-1: COMPUTING BAUD RATE RELOAD VALUE(1)**

$$FSCL = \frac{FCY}{I2C1BRG + 1 + \frac{FCY}{10,000,000}}$$

or

$$I2C1BRG = \left(\frac{FCY}{FSCL} - \frac{FCY}{10,000,000}\right) - 1$$

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

## 16.4 Slave Address Masking

The I2C1MSK register (Register 16-3) designates address bit positions as “don’t care” for both 7-Bit and 10-Bit Addressing modes. Setting a particular bit location (= 1) in the I2C1MSK register causes the slave module to respond whether the corresponding address bit value is ‘0’ or ‘1’. For example, when I2C1MSK is set to ‘00100000’, the slave module will detect both addresses: ‘0000000’ and ‘00100000’.

To enable address masking, the Intelligent Peripheral Management Interface (IPMI) must be disabled by clearing the IPMIEN bit (I2C1CON<11>).

**Note:** As a result of changes in the I²C protocol, the addresses in Table 16-2 are reserved and will not be Acknowledged in Slave mode. This includes any address mask settings that include any of these addresses.

**TABLE 16-1: I²C™ CLOCK RATES(1)**

| Required System FSCL | FCY | I2C1BRG Value | | Actual FSCL |
|---|---|---|---|---|
| | | (Decimal) | (Hexadecimal) | |
| 100 kHz | 16 MHz | 157 | 9D | 100 kHz |
| 100 kHz | 8 MHz | 78 | 4E | 100 kHz |
| 100 kHz | 4 MHz | 39 | 27 | 99 kHz |
| 400 kHz | 16 MHz | 37 | 25 | 404 kHz |
| 400 kHz | 8 MHz | 18 | 12 | 404 kHz |
| 400 kHz | 4 MHz | 9 | 9 | 385 kHz |
| 400 kHz | 2 MHz | 4 | 4 | 385 kHz |
| 1 MHz | 16 MHz | 13 | D | 1.026 MHz |
| 1 MHz | 8 MHz | 6 | 6 | 1.026 MHz |
| 1 MHz | 4 MHz | 3 | 3 | 0.909 MHz |

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

**TABLE 16-2: I²C™ RESERVED ADDRESSES(1)**

| Slave Address | R/$\overline{W}$ Bit | Description |
|---|---|---|
| 0000 000 | 0 | General Call Address(2) |
| 0000 000 | 1 | Start Byte |
| 0000 001 | x | Cbus Address |
| 0000 010 | x | Reserved |
| 0000 011 | x | Reserved |
| 0000 1xx | x | HS Mode Master Code |
| 1111 1xx | x | Reserved |
| 1111 0xx | x | 10-Bit Slave Upper Byte(3) |

**Note 1:** The address bits listed here will never cause an address match, independent of the address mask settings.

**2:** Address will be Acknowledged only if GCEN = 1.

**3:** Match on this address can only occur on the upper byte in 10-Bit Addressing mode.

**REGISTER 16-1: I2C1CON: I2C1 CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | R/W-1 HC | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| I2CEN | — | I2CSIDL | SCLREL | IPMIEN | A10M | DISSLW | SMEN |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0, HC | R/W-0, HC | R/W-0, HC | R/W-0, HC | R/W-0, HC |
|---|---|---|---|---|---|---|---|
| GCEN | STREN | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **I2CEN:** I2C1 Enable bit
1 = Enables the I2C1 module and configures the SDA1 and SCL1 pins as serial port pins
0 = Disables the I2C1 module; all $I^2C$™ pins are controlled by port functions

bit 14 **Unimplemented:** Read as '0'

bit 13 **I2CSIDL:** Stop in Idle Mode bit
1 = Discontinues module operation when device enters an Idle mode
0 = Continues module operation in Idle mode

bit 12 **SCLREL:** SCL1 Release Control bit (when operating as $I^2C$ slave)
1 = Releases SCL1 clock
0 = Holds SCL1 clock low (clock stretch)
If STREN = 1:
Bit is R/$\overline{W}$ (i.e., software may write '0' to initiate stretch and write '1' to release clock). Hardware clear at beginning of slave transmission. Hardware clear at end of slave reception.
If STREN = 0:
Bit is R/S (i.e., software may only write '1' to release clock). Hardware clear at beginning of slave transmission.

bit 11 **IPMIEN:** Intelligent Peripheral Management Interface (IPMI) Enable bit
1 = IPMI Support mode is enabled; all addresses Acknowledged
0 = IPMI Support mode is disabled

bit 10 **A10M:** 10-Bit Slave Addressing bit
1 = I2C1ADD is a 10-bit slave address
0 = I2C1ADD is a 7-bit slave address

bit 9 **DISSLW:** Disable Slew Rate Control bit
1 = Slew rate control disabled
0 = Slew rate control enabled

bit 8 **SMEN:** SMBus Input Levels bit
1 = Enables I/O pin thresholds compliant with the SMBus specification
0 = Disables the SMBus input thresholds

bit 7 **GCEN:** General Call Enable bit (when operating as $I^2C$ slave)
1 = Enables interrupt when a general call address is received in the I2C1RSR (module is enabled for reception)
0 = General call address disabled

bit 6 **STREN:** SCL1 Clock Stretch Enable bit (when operating as $I^2C$ slave)
Used in conjunction with the SCLREL bit.
1 = Enables software or receive clock stretching
0 = Disables software or receive clock stretching

**REGISTER 16-1: I2C1CON: I2C1 CONTROL REGISTER (CONTINUED)**

bit 5 **ACKDT:** Acknowledge Data bit (when operating as $I^2C$ master; applicable during master receive)
Value that will be transmitted when the software initiates an Acknowledge sequence.
1 = Sends NACK during Acknowledge
0 = Sends ACK during Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit
(when operating as $I^2C$ master; applicable during master receive)
1 = Initiates Acknowledge sequence on SDA1 and SCL1 pins and transmits ACKDT data bit; hardware clear at end of master Acknowledge sequence
0 = Acknowledge sequence not in progress

bit 3 **RCEN:** Receive Enable bit (when operating as $I^2C$ master)
1 = Enables Receive mode for $I^2C$; hardware clear at end of eighth bit of master receive data byte
0 = Receive sequence not in progress

bit 2 **PEN:** Stop Condition Enable bit (when operating as $I^2C$ master)
1 = Initiates Stop condition on SDA1 and SCL1 pins; hardware clear at end of master Stop sequence
0 = Stop condition not in progress

bit 1 **RSEN:** Repeated Start Condition Enable bit (when operating as $I^2C$ master)
1 = Initiates Repeated Start condition on SDA1 and SCL1 pins; hardware clear at end of master Repeated Start sequence
0 = Repeated Start condition not in progress

bit 0 **SEN:** Start Condition Enable bit (when operating as $I^2C$ master)
1 = Initiates Start condition on SDA1 and SCL1 pins; hardware clear at end of master Start sequence
0 = Start condition not in progress

**REGISTER 16-2: I2C1STAT: I2C1 STATUS REGISTER**

| R-0, HSC | R-0, HSC | U-0 | U-0 | U-0 | R/C-0, HS | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| ACKSTAT | TRSTAT | — | — | — | BCL | GCSTAT | ADD10 |
| bit 15 | | | | | | | bit 8 |

| R/C-0, HS | R/C-0, HS | R-0, HSC | R/C-0, HSC | R/C-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| IWCOL | I2COV | D/$\overline{A}$ | P | S | R/$\overline{W}$ | RBF | TBF |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ACKSTAT:** Acknowledge Status bit
1 = NACK was detected last
0 = ACK was detected last
Hardware set or clear at end of Acknowledge.

bit 14 **TRSTAT:** Transmit Status bit
(When operating as $I^2C$™ master; applicable to master transmit operation.)
1 = Master transmit is in progress (8 bits + ACK)
0 = Master transmit is not in progress
Hardware set at beginning of master transmission; hardware clear at end of slave Acknowledge.

bit 13-11 **Unimplemented:** Read as '0'

bit 10 **BCL:** Master Bus Collision Detect bit
1 = A bus collision has been detected during a master operation
0 = No collision
Hardware set at detection of bus collision.

bit 9 **GCSTAT:** General Call Status bit
1 = General call address was received
0 = General call address was not received
Hardware set when address matches general call address; hardware clear at Stop detection.

bit 8 **ADD10:** 10-Bit Address Status bit
1 = 10-bit address was matched
0 = 10-bit address was not matched
Hardware set at match of 2nd byte of matched 10-bit address; hardware clear at Stop detection.

bit 7 **IWCOL:** Write Collision Detect bit
1 = An attempt to write to the I2C1TRN register failed because the $I^2C$ module is busy
0 = No collision
Hardware set at occurrence of write to I2C1TRN while busy (cleared by software).

bit 6 **I2COV:** Receive Overflow Flag bit
1 = A byte was received while the I2C1RCV register is still holding the previous byte
0 = No overflow
Hardware set at attempt to transfer I2C1RSR to I2C1RCV (cleared by software).

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit (when operating as $I^2C$ slave)
1 = Indicates that the last byte received was data
0 = Indicates that the last byte received was the device address
Hardware clear at device address match; hardware set by write to I2C1TRN or by reception of slave byte.

bit 4 **P:** Stop bit
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

**REGISTER 16-2: I2C1STAT: I2C1 STATUS REGISTER (CONTINUED)**

bit 3 **S:** Start bit
1 = Indicates that a Start (or Repeated Start) bit has been detected last
0 = Start bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

bit 2 **R/$\overline{W}$:** Read/$\overline{Write}$ Information bit (when operating as $I^2C$ slave)
1 = Read – indicates data transfer is output from slave
0 = Write – indicates data transfer is input to slave
Hardware set or clear after reception of $I^2C$ device address byte.

bit 1 **RBF:** Receive Buffer Full Status bit
1 = Receive complete, I2C1RCV is full
0 = Receive not complete, I2C1RCV is empty
Hardware set when I2C1RCV is written with received byte; hardware clear when software reads I2C1RCV.

bit 0 **TBF:** Transmit Buffer Full Status bit
1 = Transmit in progress, I2C1TRN is full
0 = Transmit complete, I2C1TRN is empty
Hardware set when software writes to I2C1TRN; hardware clear at completion of data transmission.

**REGISTER 16-3: I2C1MSK: I2C1 SLAVE MODE ADDRESS MASK REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | AMSK9 | AMSK8 |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| AMSK7 | AMSK6 | AMSK5 | AMSK4 | AMSK3 | AMSK2 | AMSK1 | AMSK0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-10 **Unimplemented:** Read as '0'

bit 9-0 **AMSK<9:0>:** Mask for Address Bit x Select bits
1 = Enable masking for bit x of incoming message address; bit match not required in this position
0 = Disable masking for bit x; bit match required in this position

**REGISTER 16-4: PADCFG1: PAD CONFIGURATION CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | SMBUSDEL | OC1TRIS(1,2) | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-5 **Unimplemented:** Read as '0'

bit 4 **SMBUSDEL:** SMBus SDA Input Delay Select bit
1 = The $I^2C$™ module is configured for a longer SMBus input delay (nominal 300 ns delay)
0 = The $1^2C$ module is configured for a legacy input delay (nominal 150 ns delay)

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

**Note 1:** To enable the actual OC1 output, the OCPWM1 module has to be enabled.
**2:** Bit 3 is described in related chapters.

## 17.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Universal Asynchronous Receiver Transmitter, refer to the *"PIC24F Family Reference Manual"*, **Section 21. "UART"** (DS39708).

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in this PIC24F device family. The UART is a full-duplex asynchronous system that can communicate with peripheral devices, such as personal computers, LIN, RS-232 and RS-485 interfaces. This module also supports a hardware flow control option with the $\overline{\text{U1CTS}}$ and $\overline{\text{U1RTS}}$ pins, and also includes an IrDA® encoder and decoder.

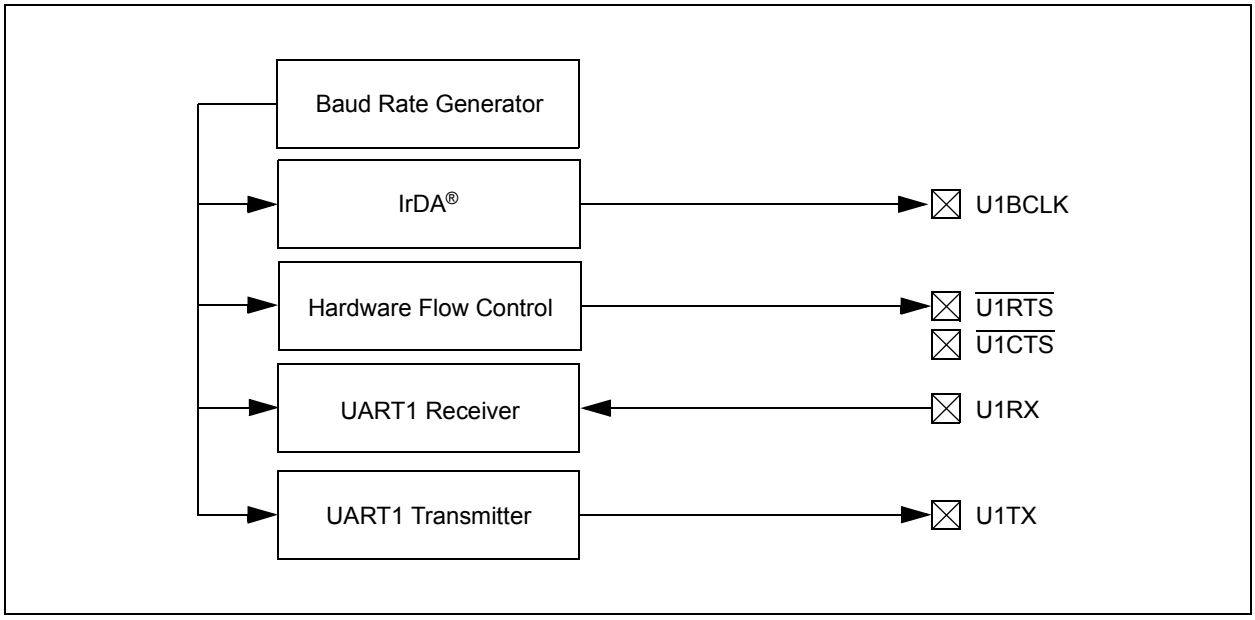
The primary features of the UART module are:

- Full-Duplex, 8-Bit or 9-Bit Data Transmission through the U1TX and U1RX pins
- Even, Odd or No Parity Options (for 8-bit data)
- One or Two Stop bits
- Hardware Flow Control Option with $\overline{\text{U1CTS}}$ and $\overline{\text{U1RTS}}$ pins
- Fully Integrated Baud Rate Generator (IBRG) with 16-Bit Prescaler
- Baud Rates Ranging from 1 Mbps to 15 bps at 16 MIPS
- 4-Deep, First-In-First-Out (FIFO) Transmit Data Buffer
- 4-Deep FIFO Receive Data Buffer
- Parity, Framing and Buffer Overrun Error Detection
- Support for 9-Bit mode with Address Detect (9th bit = 1)
- Transmit and Receive Interrupts
- Loopback mode for Diagnostic Support
- Support for Sync and Break Characters
- Supports Automatic Baud Rate Detection
- IrDA Encoder and Decoder Logic
- 16x Baud Clock Output for IrDA Support

A simplified block diagram of the UART is displayed in Figure 17-1. The UART module consists of these important hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver

**FIGURE 17-1: UART SIMPLIFIED BLOCK DIAGRAM**



## 17.1 UART Baud Rate Generator (BRG)

The UART module includes a dedicated 16-bit Baud Rate Generator (BRG). The U1BRG register controls the period of a free-running, 16-bit timer. Equation 17-1 provides the formula for computation of the baud rate with BRGH = 0.

**EQUATION 17-1: UART BAUD RATE WITH BRGH = 0(1)**

$$\text{Baud Rate} = \frac{FCY}{16 \bullet (U1BRG + 1)}$$

$$U1BRG = \frac{FCY}{16 \bullet \text{Baud Rate}} - 1$$

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

Example 17-1 provides the calculation of the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600

The maximum baud rate (BRGH = 0) possible is FCY/16 (for U1BRG = 0) and the minimum baud rate possible is FCY/(16 * 65536).

Equation 17-2 provides the formula for computation of the baud rate with BRGH = 1.

**EQUATION 17-2: UART BAUD RATE WITH BRGH = 1(1)**

$$\text{Baud Rate} = \frac{FCY}{4 \bullet (U1BRG + 1)}$$

$$U1BRG = \frac{FCY}{4 \bullet \text{Baud Rate}} - 1$$

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

The maximum baud rate (BRGH = 1) possible is FCY/4 (for U1BRG = 0) and the minimum baud rate possible is FCY/(4 * 65536).

Writing a new value to the U1BRG register causes the BRG timer to be reset (cleared). This ensures the BRG does not wait for a timer overflow before generating the new baud rate.

**EXAMPLE 17-1: BAUD RATE ERROR CALCULATION (BRGH = 0)(1)**

Desired Baud Rate = FCY/(16 (U1BRG + 1))

Solving for UxBRG value:

U1BRG = ((FCY/Desired Baud Rate)/16) – 1
U1BRG = ((4000000/9600)/16) – 1
U1BRG = 25

Calculated Baud Rate = 4000000/(16 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate) / Desired Baud Rate
= (9615 – 9600)/9600
= 0.16%

**Note 1:** Based on FCY = FOSC/2, Doze mode and PLL are disabled.

## 17.2 Transmitting in 8-Bit Data Mode

1. Set up the UART:
   a) Write appropriate values for data, parity and Stop bits.
   b) Write appropriate baud rate value to the U1BRG register.
   c) Set up transmit and receive interrupt enable and priority bits.
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt two cycles after being set).
4. Write data byte to lower byte of U1TXREG word. The value will be immediately transferred to the Transmit Shift Register (TSR), and the serial bit stream will start shifting out with the next rising edge of the baud clock.
5. Alternately, the data byte may be transferred while UTXEN = 0, and then, the user may set UTXEN. This will cause the serial bit stream to begin immediately because the baud clock will start from a cleared state.
6. A transmit interrupt will be generated as per interrupt control bit, UTXISEL1.

## 17.3 Transmitting in 9-Bit Data Mode

1. Set up the UART (as described in **Section 17.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt two cycles after being set).
4. Write U1TXREG as a 16-bit value only.
5. A word write to U1TXREG triggers the transfer of the 9-bit data to the TSR. The serial bit stream will start shifting out with the first rising edge of the baud clock.
6. A transmit interrupt will be generated as per the setting of control bit, UTXISEL1.

## 17.4 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an auto-baud Sync byte.

1. Configure the UART for the desired mode.
2. Set UTXEN and UTXBRK – sets up the Break character.
3. Load the U1TXREG with a dummy character to initiate transmission (value is ignored).
4. Write '55h' to U1TXREG – loads the Sync character into the transmit FIFO.
5. After the Break has been sent, the UTXBRK bit is reset by hardware. The Sync character now transmits.

## 17.5 Receiving in 8-Bit or 9-Bit Data Mode

1. Set up the UART (as described in **Section 17.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. A receive interrupt will be generated when one or more data characters have been received as per interrupt control bit, URXISEL1.
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read U1RXREG.

The act of reading the UxRXREG character will move the next character to the top of the receive FIFO, including a new set of PERR and FERR values.

## 17.6 Operation of $\overline{\text{U1CTS}}$ and $\overline{\text{U1RTS}}$ Control Pins

UART1 Clear to Send ($\overline{\text{U1CTS}}$) and Request to Send ($\overline{\text{U1RTS}}$) are the two hardware-controlled pins that are associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control modes. They are implemented to control the transmission and reception between the Data Terminal Equipment (DTE). The UEN<1:0> bits in the U1MODE register configure these pins.

## 17.7 Infrared Support

The UART module provides two types of infrared UART support: one is the IrDA clock output to support an external IrDA encoder and decoder device (legacy module support), and the other is the full implementation of the IrDA encoder and decoder.

As the IrDA modes require a 16x baud clock, they will only work when the BRGH bit (U1MODE<3>) is '0'.

### 17.7.1 EXTERNAL IrDA SUPPORT – IrDA CLOCK OUTPUT

To support external IrDA encoder and decoder devices, the U1BCLK pin (same as the $\overline{\text{U1RTS}}$ pin) can be configured to generate the 16x baud clock. When UEN<1:0> = 11, the U1BCLK pin will output the 16x baud clock if the UART module is enabled; it can be used to support the IrDA codec chip.

### 17.7.2 BUILT-IN IrDA ENCODER AND DECODER

The UART has full implementation of the IrDA encoder and decoder as part of the UART module. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit (U1MODE<12>). When enabled (IREN = 1), the receive pin (U1RX) acts as the input from the infrared receiver. The transmit pin (U1TX) acts as the output to the infrared transmitter.

**REGISTER 17-1: U1MODE: UART1 MODE REGISTER**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0(2) | R/W-0(2) |
|---|---|---|---|---|---|---|---|
| UARTEN | — | USIDL | IREN(1) | RTSMD | — | UEN1 | UEN0 |
| bit 15 | | | | | | | bit 8 |

| R/C-0, HC | R/W-0 | R/W-0, HC | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| WAKE | LPBACK | ABAUD | RXINV | BRGH | PDSEL1 | PDSEL0 | STSEL |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **UARTEN:** UART1 Enable bit
1 = UART1 is enabled; all UART1 pins are controlled by UART1 as defined by UEN<1:0>
0 = UART1 is disabled; all UART1 pins are controlled by port latches; UART1 power consumption is minimal

bit 14 **Unimplemented:** Read as '0'

bit 13 **USIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12 **IREN:** IrDA® Encoder and Decoder Enable bit(1)
1 = IrDA encoder and decoder enabled
0 = IrDA encoder and decoder disabled

bit 11 **RTSMD:** Mode Selection for $\overline{\text{U1RTS}}$ Pin bit
1 = $\overline{\text{U1RTS}}$ pin in Simplex mode
0 = $\overline{\text{U1RTS}}$ pin in Flow Control mode

bit 10 **Unimplemented:** Read as '0'

bit 9-8 **UEN<1:0>:** UART1 Enable bits(2)
11 = U1TX, U1RX and U1BCLK pins are enabled and used; $\overline{\text{U1CTS}}$ pin controlled by port latches
10 = U1TX, U1RX, $\overline{\text{U1CTS}}$ and $\overline{\text{U1RTS}}$ pins are enabled and used
01 = U1TX, U1RX and $\overline{\text{U1RTS}}$ pins are enabled and used; $\overline{\text{U1CTS}}$ pin controlled by port latches
00 = U1TX and UxRX pins are enabled and used; $\overline{\text{U1CTS}}$ and $\overline{\text{U1RTS}}$/U1BCLK pins controlled by port latches

bit 7 **WAKE:** Wake-up on Start Bit Detect During Sleep Mode Enable bit
1 = UART1 will continue to sample the U1RX pin; interrupt generated on falling edge, bit cleared in hardware on following rising edge
0 = No wake-up enabled

bit 6 **LPBACK:** UART1 Loopback Mode Select bit
1 = Enable Loopback mode
0 = Loopback mode is disabled

bit 5 **ABAUD:** Auto-Baud Enable bit
1 = Enable baud rate measurement on the next character – requires reception of a Sync field (55h); cleared in hardware upon completion
0 = Baud rate measurement disabled or completed

bit 4 **RXINV:** Receive Polarity Inversion bit
1 = U1RX Idle state is '0'
0 = U1RX Idle state is '1'

**Note 1:** This feature is only available for the 16x BRG mode (BRGH = 0).
**2:** Bit availability depends on pin availability.

**REGISTER 17-1: U1MODE: UART1 MODE REGISTER (CONTINUED)**

bit 3 **BRGH:** High Baud Rate Enable bit
1 = BRG generates 4 clocks per bit period (4x baud clock, High-Speed mode)
0 = BRG generates 16 clocks per bit period (16x baud clock, Standard mode)

bit 2-1 **PDSEL<1:0>:** Parity and Data Selection bits
11 = 9-bit data, no parity
10 = 8-bit data, odd parity
01 = 8-bit data, even parity
00 = 8-bit data, no parity

bit 0 **STSEL:** Stop Bit Selection bit
1 = Two Stop bits
0 = One Stop bit

**Note 1:** This feature is only available for the 16x BRG mode (BRGH = 0).
**2:** Bit availability depends on pin availability.

**REGISTER 17-2: U1STA: UART1 STATUS AND CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0, HC | R/W-0 | R-0, HSC | R-1, HSC |
|---|---|---|---|---|---|---|---|
| UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R-1, HSC | R-0, HSC | R-0, HSC | R/C-0, HS | R-0, HSC |
|---|---|---|---|---|---|---|---|
| URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | C = Clearable bit | HC = Hardware Clearable bit | |
|---|---|---|---|
| | HS = Hardware Settable bit | HSC = Hardware Settable/Clearable bit | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 15,13 **UTXISEL<1:0>:** Transmission Interrupt Mode Selection bits
- 11 = Reserved; do not use
- 10 = Interrupt when a character is transferred to the Transmit Shift Register (TSR), and as a result, the transmit buffer becomes empty
- 01 = Interrupt when the last character is shifted out of the Transmit Shift Register; all transmit operations are completed
- 00 = Interrupt when a character is transferred to the Transmit Shift Register (this implies there is at least one character open in the transmit buffer)

bit 14 **UTXINV:** IrDA® Encoder Transmit Polarity Inversion bit

If IREN = 0:
- 1 = U1TX Idle '0'
- 0 = U1TX Idle '1'

If IREN = 1:
- 1 = U1TX Idle '1'
- 0 = U1TX Idle '0'

bit 12 **Unimplemented:** Read as '0'

bit 11 **UTXBRK:** Transmit Break bit
- 1 = Send Sync Break on next transmission – Start bit, followed by twelve '0' bits, followed by Stop bit; cleared by hardware upon completion
- 0 = Sync Break transmission disabled or completed

bit 10 **UTXEN:** Transmit Enable bit
- 1 = Transmit enabled, U1TX pin controlled by UART1
- 0 = Transmit disabled, any pending transmission is aborted and buffer is reset. U1TX pin controlled by the PORT register.

bit 9 **UTXBF:** Transmit Buffer Full Status bit (read-only)
- 1 = Transmit buffer is full
- 0 = Transmit buffer is not full, at least one more character can be written

bit 8 **TRMT:** Transmit Shift Register Empty bit (read-only)
- 1 = Transmit Shift Register is empty and transmit buffer is empty (the last transmission has completed)
- 0 = Transmit Shift Register is not empty, a transmission is in progress or queued

bit 7-6 **URXISEL<1:0>:** Receive Interrupt Mode Selection bits
- 11 = Interrupt is set on RSR transfer, making the receive buffer full (i.e., has 4 data characters)
- 10 = Interrupt is set on RSR transfer, making the receive buffer 3/4 full (i.e., has 3 data characters)
- 0x = Interrupt is set when any character is received and transferred from the RSR to the receive buffer. Receive buffer has one or more characters.

**REGISTER 17-2: U1STA: UART1 STATUS AND CONTROL REGISTER (CONTINUED)**

bit 5 **ADDEN:** Address Character Detect bit (bit 8 of received data = 1)
1 = Address Detect mode enabled. If 9-bit mode is not selected, this does not take effect.
0 = Address Detect mode disabled

bit 4 **RIDLE:** Receiver Idle bit (read-only)
1 = Receiver is Idle
0 = Receiver is active

bit 3 **PERR:** Parity Error Status bit (read-only)
1 = Parity error has been detected for the current character (character at the top of the receive FIFO)
0 = Parity error has not been detected

bit 2 **FERR:** Framing Error Status bit (read-only)
1 = Framing error has been detected for the current character (character at the top of the receive FIFO)
0 = Framing error has not been detected

bit 1 **OERR:** Receive Buffer Overrun Error Status bit (clear/read-only)
1 = Receive buffer has overflowed
0 = Receive buffer has not overflowed (clearing a previously set OERR bit (1 → 0 transition) will reset the receiver buffer and the RSR to the empty state)

bit 0 **URXDA:** Receive Buffer Data Available bit (read-only)
1 = Receive buffer has data; at least one more character can be read
0 = Receive buffer is empty

**REGISTER 17-3: U1TXREG: UART1 TRANSMIT REGISTER**

| U-x | U-x | U-x | U-x | U-x | U-x | U-x | W-x |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | UTX8 |
| bit 15 | | | | | | | bit 8 |

| W-x | W-x | W-x | W-x | W-x | W-x | W-x | W-x |
|---|---|---|---|---|---|---|---|
| UTX7 | UTX6 | UTX5 | UTX4 | UTX3 | UTX2 | UTX1 | UTX0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-9 **Unimplemented:** Read as ‘0’
bit 8 **UTX8:** Data of the Transmitted Character bit (in 9-bit mode)
bit 7-0 **UTX<7:0>:** Data of the Transmitted Character bits

**REGISTER 17-4: U1RXREG: UART1 RECEIVE REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-0, HSC |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | URX8 |
| bit 15 | | | | | | | bit 8 |

| R-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| URX7 | URX6 | URX5 | URX4 | URX3 | URX2 | URX1 | URX0 |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | HSC = Hardware Settable/Clearable bit | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ | |
| -n = Value at POR | ‘1’ = Bit is set | ‘0’ = Bit is cleared | x = Bit is unknown |

bit 15-9 **Unimplemented:** Read as ‘0’
bit 8 **URX8:** Data of the Received Character bit (in 9-bit mode)
bit 7-0 **URX<7:0>:** Data of the Received Character bits

## 18.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

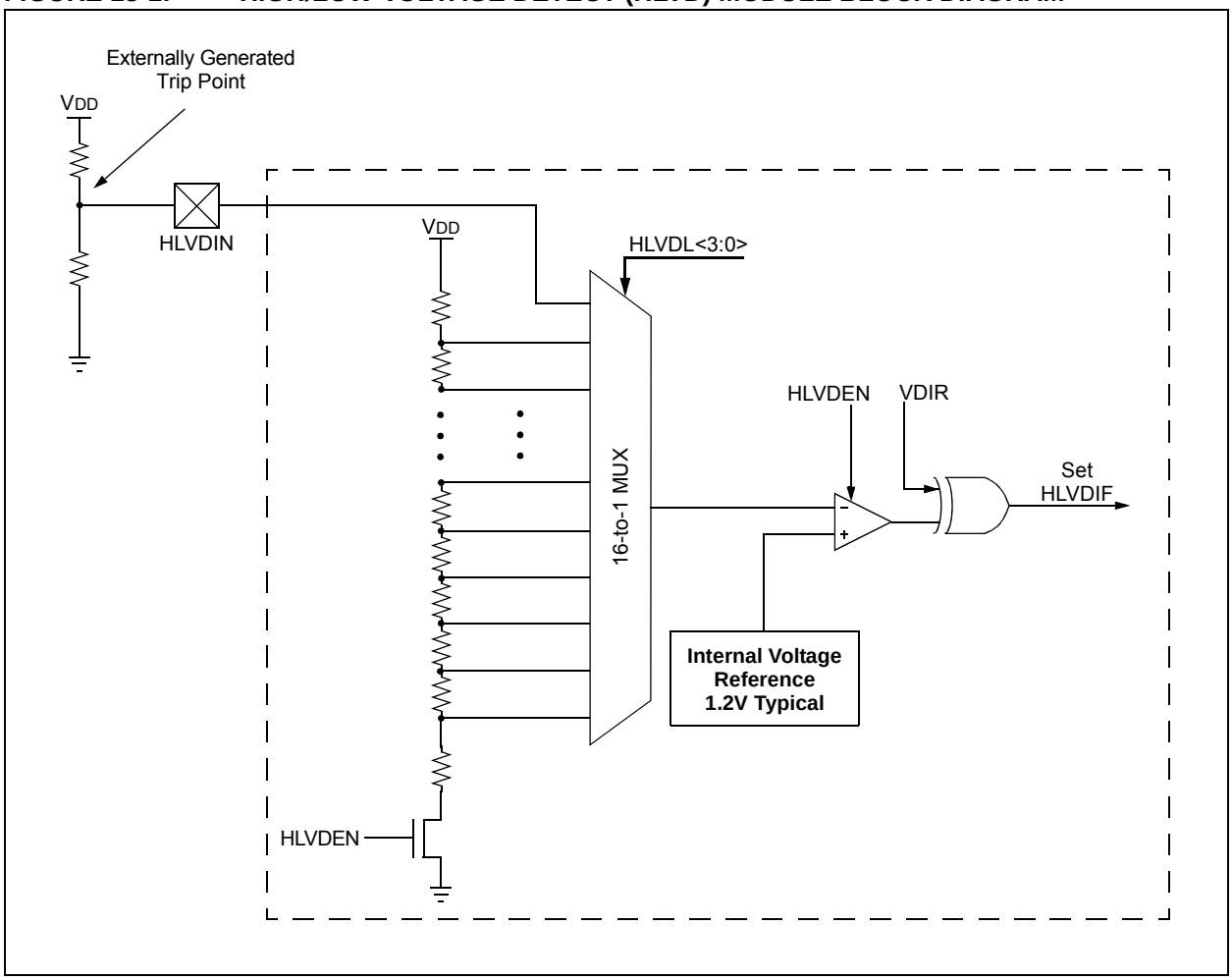
| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the High/Low-Voltage Detect, refer to the *"PIC24F Family Reference Manual"*, **Section 36. "High-Level Integration with Programmable High/Low-Voltage Detect (HLVD)"** (DS39725). |

The High/Low-Voltage Detect module (HLVD) is a programmable circuit that allows the user to specify both the device voltage trip point and the direction of change.

An interrupt flag is set if the device experiences an excursion past the trip point in the direction of change. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to the interrupt.

The HLVD Control register (see Register 18-1) completely controls the operation of the HLVD module. This allows the circuitry to be "turned off" by the user under software control, which minimizes the current consumption for the device.

**FIGURE 18-1: HIGH/LOW-VOLTAGE DETECT (HLVD) MODULE BLOCK DIAGRAM**



**REGISTER 18-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| HLVDEN | — | HLSIDL | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| VDIR | BGVST | IRVST | — | HLVDL3 | HLVDL2 | HLVDL1 | HLVDL0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **HLVDEN:** High/Low-Voltage Detect Power Enable bit
1 = HLVD enabled
0 = HLVD disabled

bit 14 **Unimplemented:** Read as '0'

bit 13 **HLSIDL:** HLVD Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-8 **Unimplemented:** Read as '0'

bit 7 **VDIR:** Voltage Change Direction Select bit
1 = Event occurs when voltage equals or exceeds trip point (HLVDL<3:0>)
0 = Event occurs when voltage equals or falls below trip point (HLVDL<3:0>)

bit 6 **BGVST:** Band Gap Voltage Stable Flag bit
1 = Indicates that the band gap voltage is stable
0 = Indicates that the band gap voltage is unstable

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit
1 = Indicates that the internal reference voltage is stable and the high-voltage detect logic generates the interrupt flag at the specified voltage range
0 = Indicates that the internal reference voltage is unstable and the high-voltage detect logic will not generate the interrupt flag at the specified voltage range, and the HLVD interrupt should not be enabled

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

bit 3-0 **HLVDL<3:0>:** High/Low-Voltage Detection Limit bits
1111 = External analog input is used (input comes from the HLVDIN pin)
1110 = Trip point 1(1)
1101 = Trip point 2(1)
1100 = Trip point 3(1)
.
.
.
0000 = Trip point 15(1)

**Note 1:** For actual trip point, refer to **Section 26.0 "Electrical Characteristics"**.

## 19.0 10-BIT HIGH-SPEED A/D CONVERTER

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the 10-Bit High-Speed A/D Converter, refer to the *"PIC24F Family Reference Manual"*, **Section 17. "10-Bit A/D Converter"** (DS39705).

The 10-bit A/D Converter has the following key features:

- Successive Approximation (SAR) conversion
- Conversion speeds of up to 500 ksps
- 9 analog input pins
- External voltage reference input pins
- Internal band gap reference inputs
- Automatic Channel Scan mode
- Selectable conversion trigger source
- 16-word conversion result buffer
- Selectable Buffer Fill modes
- Four result alignment options
- Operation during CPU Sleep and Idle modes

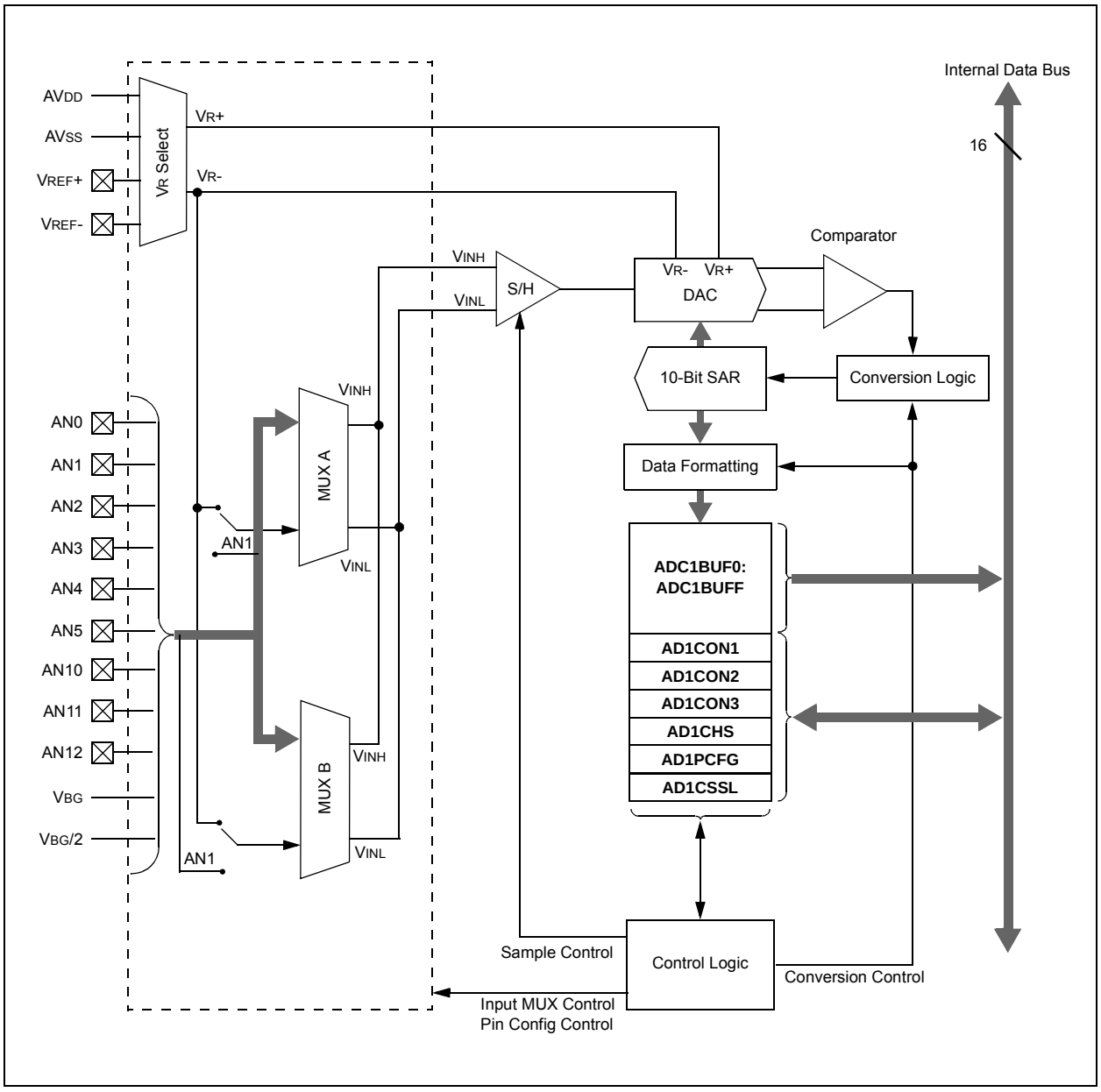
On all PIC24F04KA201 family devices, the 10-bit A/D Converter has nine analog input pins, designated AN0 through AN5 and AN10 through AN12. In addition, there are two analog input pins for external voltage reference connections ($V_{REF}$+ and $V_{REF}$-). These voltage reference inputs may be shared with other analog input pins.

A block diagram of the A/D Converter is displayed in Figure 19-1.

To perform an A/D conversion:

1. Configure the A/D module:
   a) Configure port pins as analog inputs and/or select band gap reference inputs (AD1PCFG<15:13>, AD1PCFG<9:6>).
   b) Select voltage reference source to match expected range on analog inputs (AD1CON2<15:13>).
   c) Select the analog conversion clock to match the desired data rate with the processor clock (AD1CON3<7:0>).
   d) Select the appropriate sample/conversion sequence (AD1CON1<7:5> and AD1CON3<12:8>).
   e) Select how conversion results are presented in the buffer (AD1CON1<9:8>).
   f) Select interrupt rate (AD1CON2<5:2>).
   g) Turn on A/D module (AD1CON1<15>).
2. Configure A/D interrupt (if required):
   a) Clear the AD1IF bit.
   b) Select A/D interrupt priority.

**FIGURE 19-1: 10-BIT HIGH-SPEED A/D CONVERTER BLOCK DIAGRAM**



**REGISTER 19-1: AD1CON1: A/D CONTROL REGISTER 1**

| R/W-0 | U-0 | R/W-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADON(1) | — | ADSIDL | — | — | — | FORM1 | FORM0 |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 | R/W-0, HSC | R/W-0, HSC |
|---|---|---|---|---|---|---|---|
| SSRC2 | SSRC1 | SSRC0 | — | — | ASAM | SAMP | DONE |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ADON:** A/D Operating Mode bit(1)
1 = A/D Converter module is operating
0 = A/D Converter is off

bit 14 **Unimplemented:** Read as '0'

bit 13 **ADSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-10 **Unimplemented:** Read as '0'

bit 9-8 **FORM<1:0>:** Data Output Format bits
11 = Signed fractional (sddd dddd dd00 0000)
10 = Fractional (dddd dddd dd00 0000)
01 = Signed integer (ssss sssd dddd dddd)
00 = Integer (0000 00dd dddd dddd)

bit 7-5 **SSRC<2:0>:** Conversion Trigger Source Select bits
111 = Internal counter ends sampling and starts conversion (auto-convert)
110 = CTMU event ends sampling and starts conversion
101 = Reserved
100 = Reserved
011 = Reserved
010 = Timer3 compare ends sampling and starts conversion
001 = Active transition on INT0 pin ends sampling and starts conversion
000 = Clearing SAMP bit ends sampling and starts conversion

bit 4-3 **Unimplemented:** Read as '0'

bit 2 **ASAM:** A/D Sample Auto-Start bit
1 = Sampling begins immediately after last conversion completes; SAMP bit is auto-set
0 = Sampling begins when SAMP bit is set

bit 1 **SAMP:** A/D Sample Enable bit
1 = A/D sample/hold amplifier is sampling input
0 = A/D sample/hold amplifier is holding

bit 0 **DONE:** A/D Conversion Status bit
1 = A/D conversion is done
0 = A/D conversion is not done

**Note 1:** Values of ADC1BUFn registers will not retain their values once the ADON bit is cleared. Read out the conversion values from the buffer before disabling the module.

**REGISTER 19-2: AD1CON2: A/D CONTROL REGISTER 2**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| VCFG2 | VCFG1 | VCFG0 | OFFCAL(1) | — | CSCNA | — | — |
| bit 15 | | | | | | | bit 8 |

| R-0, HSC | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| BUFS | — | SMPI3 | SMPI2 | SMPI1 | SMPI0 | BUFM | ALTS |
| bit 7 | | | | | | | bit 0 |

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

bit 15-13 **VCFG<2:0>:** Voltage Reference Configuration bits

| VCFG<2:0> | VR+ | VR- |
|---|---|---|
| 000 | AVDD | AVSS |
| 001 | External VREF+ pin | AVSS |
| 010 | AVDD | External VREF- pin |
| 011 | External VREF+ pin | External VREF- pin |
| 1xx | AVDD | AVSS |

bit 12 **OFFCAL:** Offset Calibration bit(1)
1 = Converts to get the offset calibration value
0 = Coverts to get the actual input value

bit 11 **Unimplemented:** Read as '0'

bit 10 **CSCNA:** Scan Input Selections for CH0+ S/H Input for MUX A Input Multiplexer Setting bit
1 = Scan inputs
0 = Do not scan inputs

bit 9-8 **Unimplemented:** Read as '0'

bit 7 **BUFS:** Buffer Fill Status bit (valid only when BUFM = 1)
1 = A/D is currently filling buffer, 08-0F, user should access data in 00-07
0 = A/D is currently filling buffer, 00-07, user should access data in 08-0F

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

bit 5-2 **SMPI<3:0>:** Sample/Convert Sequences Per Interrupt Selection bits
1111 = Interrupts at the completion of conversion for each 16th sample/convert sequence
1110 = Interrupts at the completion of conversion for each 15th sample/convert sequence
.
.
.
0001 = Interrupts at the completion of conversion for each 2nd sample/convert sequence
0000 = Interrupts at the completion of conversion for each sample/convert sequence

bit 1 **BUFM:** Buffer Mode Select bit
1 = Buffer configured as two 8-word buffers (ADC1BUFn<15:8> and ADC1BUFn<7:0>)
0 = Buffer configured as one 16-word buffer (ADC1BUFn<15:0>)

bit 0 **ALTS:** Alternate Input Sample Mode Select bit
1 = Uses MUX A input multiplexer settings for first sample, then alternates between MUX B and MUX A input multiplexer settings for all subsequent samples
0 = Always uses MUX A input multiplexer settings

**Note 1:** When the OFFCAL bit is set, inputs are disconnected and tied to AVSS. This sets the inputs of the A/D to zero. Then, the user can perform a conversion. Use of the Calibration mode is not affected by AD1PCFG contents nor channel input selection. Any analog input switches are disconnected from the A/D converter in this mode. The conversion result is stored by the user software and used to compensate subsequent conversions. This can be done by adding the two's complement of the result obtained with the OFFCAL bit set to all normal A/D conversions.

**REGISTER 19-3: AD1CON3: A/D CONTROL REGISTER 3**

| R/W-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADRC | — | — | SAMC4 | SAMC3 | SAMC2 | SAMC1 | SAMC0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | ADCS5 | ADCS4 | ADCS3 | ADCS2 | ADCS1 | ADCS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ADRC:** A/D Conversion Clock Source bit
1 = A/D internal RC clock
0 = Clock derived from system clock

bit 14-13 **Unimplemented:** Read as ‘0’

bit 12-8 **SAMC<4:0>:** Auto-Sample Time bits
11111 = 31 TAD
.
.
.
00001 = 1 TAD
00000 = 0 TAD (not recommended)

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **ADCS<5:0>:** A/D Conversion Clock Select bits
111111 = 32 • TCY
.
.
.
000001 = TCY
000000 = TCY/2

**REGISTER 19-4: AD1CHS: A/D INPUT SELECT REGISTER**

| R/W-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CH0NB | — | — | — | CH0SB3 | CH0SB2 | CH0SB1 | CH0SB0 |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CH0NA | — | — | CH0SA4 | CH0SA3 | CH0SA2 | CH0SA1 | CH0SA0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **CH0NB:** Channel 0 Negative Input Select for MUX B Multiplexer Setting bit
1 = Channel 0 negative input is AN1
0 = Channel 0 negative input is VR-

bit 14-12 **Unimplemented:** Read as '0'

bit 11-8 **CH0SB<3:0>:** Channel 0 Positive Input Select for MUX B Multiplexer Setting bits
1111 = Channel 0 positive input is band gap reference (VBG)
1110 = Channel 0 positive input is band gap, divided by two, reference (VBG/2)
1101 = No channels connected (actual ADC MUX switch activates but input floats); used for CTMU
1100 = Channel 0 positive input is AN12
1011 = Channel 0 positive input is AN11
1010 = Channel 0 positive input is AN10
1001 = Reserved
1000 = Reserved
0110 = AVDD
0110 = AVSS
0101 = Channel 0 positive input is AN5
0100 = Channel 0 positive input is AN4
0010 = Channel 0 positive input is AN3
0010 = Channel 0 positive input is AN2
0001 = Channel 0 positive input is AN1
0000 = Channel 0 positive input is AN0

bit 7 **CH0NA:** Channel 0 Negative Input Select for MUX A Multiplexer Setting bit
1 = Channel 0 negative input is AN1
0 = Channel 0 negative input is VR-

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

bit 4-0 **CH0SA<4:0>:** Channel 0 Positive Input Select for Sample A bits
1111 = Channel 0 positive input is band gap reference (VBG)
1110 = Channel 0 positive input is band gap, divided by two, reference (VBG/2)
1101 = No channels connected (actual ADC MUX switch activates but input floats); used for CTMU
1100 = Channel 0 positive input is AN12
1011 = Channel 0 positive input is AN11
1010 = Channel 0 positive input is AN10
1001 = Reserved
1000 = Reserved
0110 = AVDD
0110 = AVSS
0101 = Channel 0 positive input is AN5
0100 = Channel 0 positive input is AN4
0010 = Channel 0 positive input is AN3
0010 = Channel 0 positive input is AN2
0001 = Channel 0 positive input is AN1
0000 = Channel 0 positive input is AN0

**REGISTER 19-5: AD1PCFG: A/D PORT CONFIGURATION REGISTER**

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | PCFG12 | PCFG11 | PCFG10 | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | PCFG5 | PCFG4 | PCFG3 | PCFG2 | PCFG1 | PCFG0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-13 **Unimplemented:** Read as '0'

bit 12-10 **PCFG<12:10>:** Analog Input Pin Configuration Control bits
1 = Pin for corresponding analog channel is configured in Digital mode; I/O port read enabled
0 = Pin configured in Analog mode; I/O port read disabled; A/D samples pin voltage

bit 9-6 **Unimplemented:** Read as '0'

bit 5-0 **PCFG<5:0>:** Analog Input Pin Configuration Control bits
1 = Pin for corresponding analog channel is configured in Digital mode; I/O port read enabled
0 = Pin configured in Analog mode; I/O port read disabled; A/D samples pin voltage

**REGISTER 19-6: AD1CSSL: A/D INPUT SCAN SELECT REGISTER (LOW)**

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | CSSL12 | CSSL11 | CSSL10 | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | CSSL5 | CSSL4 | CSSL3 | CSSL2 | CSSL1 | CSSL0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-13 **Unimplemented:** Read as '0'

bit 12-10 **CSSL<12:10>:** A/D Input Pin Scan Selection bits
1 = Corresponding analog channel selected for input scan
0 = Analog channel omitted from input scan

bit 9-6 **Unimplemented:** Read as '0'

bit 5-0 **CSSL<5:0>:** A/D Input Pin Scan Selection bits
1 = Corresponding analog channel selected for input scan
0 = Analog channel omitted from input scan

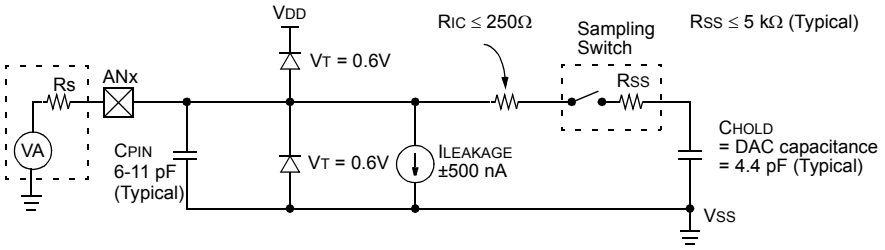
**EQUATION 19-1: A/D CONVERSION CLOCK PERIOD(1)**

$$ADCS = \frac{TAD}{TCY} - 1$$

$$TAD = TCY \bullet (ADCS + 1)$$

**Note 1:** Based on TCY = 2 * TOSC, Doze mode and PLL are disabled.

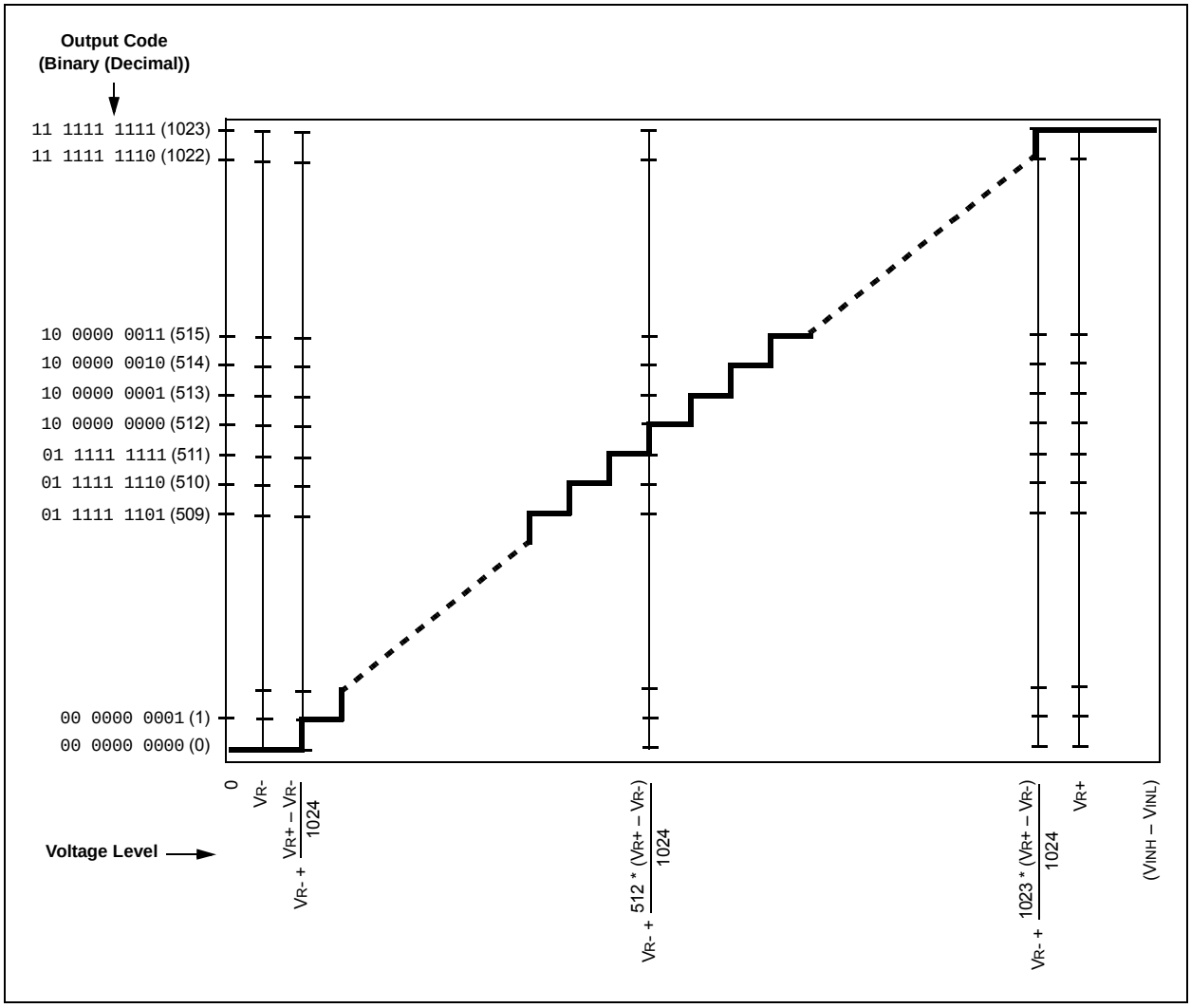
**FIGURE 19-2: 10-BIT A/D CONVERTER ANALOG INPUT MODEL**




| Legend: | | |
|---|---|---|
| | CPIN | = Input Capacitance |
| | VT | = Threshold Voltage |
| | ILEAKAGE | = Leakage Current at the pin due to various junctions |
| | RIC | = Interconnect Resistance |
| | RSS | = Sampling Switch Resistance |
| | CHOLD | = Sample/Hold Capacitance (from DAC) |

**Note:** CPIN value depends on device package and is not tested. Effect of CPIN negligible if Rs ≤ 5 kΩ.

**FIGURE 19-3: A/D TRANSFER FUNCTION**



**NOTES:**

## 20.0 COMPARATOR MODULE

| | |
|---|---|
| **Note:** | This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Comparator module, refer to the *"PIC24F Family Reference Manual"*, **Section 19. "Comparator Module"** (DS39710). |

The comparator module provides two dual input comparators. The inputs to the comparator can be configured to use any one of four external analog inputs, as well as a voltage reference input from either the internal band gap reference divided by 2 ($V_{BG}/2$) or the comparator voltage reference generator.

The comparator outputs may be directly connected to the CxOUT pins. When the respective COE equals '1', the I/O pad logic makes the unsynchronized output of the comparator available on the pin.

A simplified block diagram of the module is displayed in Figure 20-1. Diagrams of the possible individual comparator configurations are displayed in Figure 20-2.

Each comparator has its own control register, CMxCON (Register 20-1), for enabling and configuring its operation. The output and event status of all three comparators is provided in the CMSTAT register (Register 20-2).

**FIGURE 20-1: COMPARATOR MODULE BLOCK DIAGRAM**

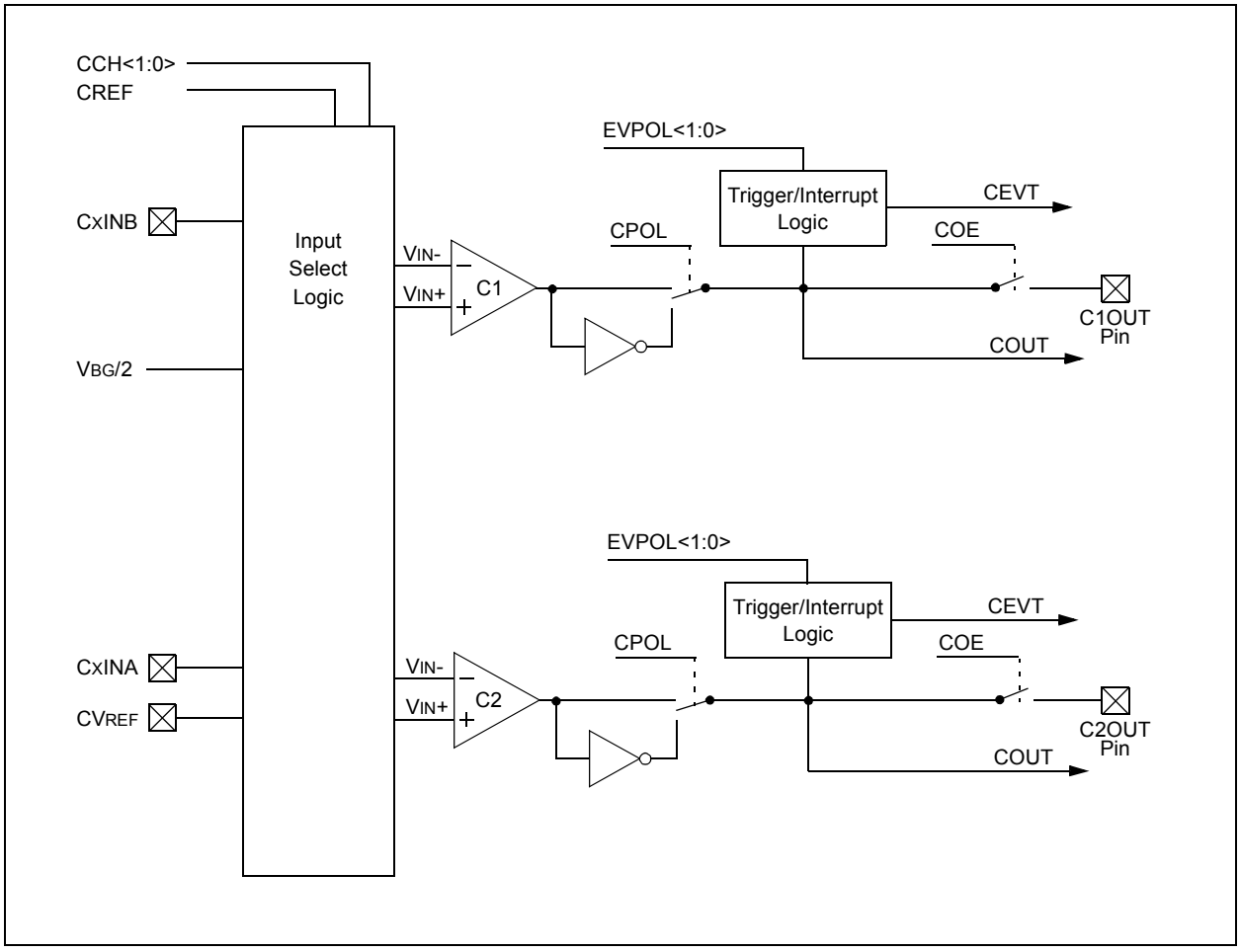
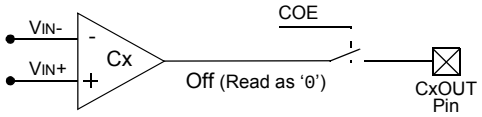
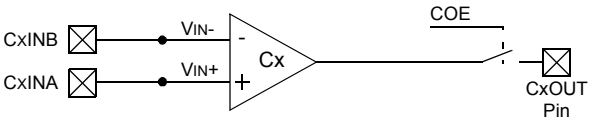


**FIGURE 20-2: INDIVIDUAL COMPARATOR CONFIGURATIONS**

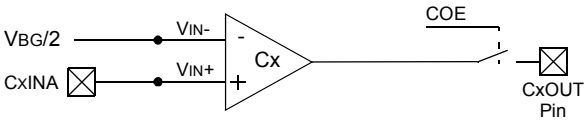
**Comparator Off**
**CON = 0, CREF = x, CCH<1:0> = xx**



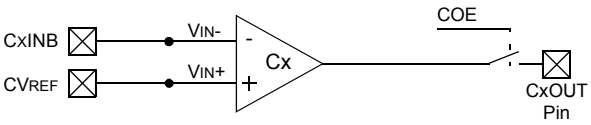

**Comparator CxINB > CxINA Compare**
**CON = 1, CREF = 0, CCH<1:0> = 00**



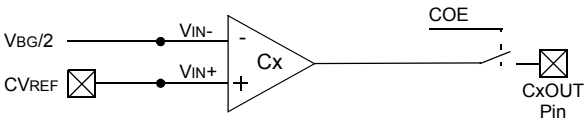

**Comparator VBG > CxINA Compare**
**CON = 1, CREF = 0, CCH<1:0> = 11**




**Comparator CxINB > CVREF Compare**
**CON = 1, CREF = 1, CCH<1:0> = 00**




**Comparator VBG > CVREF Compare**
**CON = 1, CREF = 1, CCH<1:0> = 11**



**REGISTER 20-1: CMxCON: COMPARATOR x CONTROL REGISTERS**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 | R-0 |
|---|---|---|---|---|---|---|---|
| CON | COE | CPOL | CLPWR | — | — | CEVT | COUT |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | U-0 | R/W-0 | U-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| EVPOL1 | EVPOL0 | — | CREF | — | — | CCH1 | CCH0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **CON:** Comparator Enable bit
1 = Comparator is enabled
0 = Comparator is disabled

bit 14 **COE:** Comparator Output Enable bit
1 = Comparator output is present on the CxOUT pin
0 = Comparator output is internal only

bit 13 **CPOL:** Comparator Output Polarity Select bit
1 = Comparator output is inverted
0 = Comparator output is not inverted

bit 12 **CLPWR:** Comparator Low-Power Mode Select bit
1 = Comparator operates in Low-Power mode
0 = Comparator does not operate in Low-Power mode

bit 11-10 **Unimplemented:** Read as '0'

bit 9 **CEVT:** Comparator Event bit
1 = Comparator event defined by EVPOL<1:0> has occurred; subsequent triggers and interrupts are disabled until the bit is cleared
0 = Comparator event has not occurred

bit 8 **COUT:** Comparator Output bit
When CPOL = 0:
1 = VIN+ > VIN-
0 = VIN+ < VIN-
When CPOL = 1:
1 = VIN+ < VIN-
0 = VIN+ > VIN-

bit 7-6 **EVPOL<1:0>:** Trigger/Event/Interrupt Polarity Select bits
11 = Trigger/event/interrupt generated on any change of the comparator output (while CEVT = 0)
10 = Trigger/event/interrupt generated on transition of the comparator output:
If CPOL = 0 (non-inverted polarity):
High-to-low transition only.
If CPOL = 1 (inverted polarity):
Low-to-high transition only.
01 = Trigger/event/interrupt generated on transition of comparator output:
If CPOL = 0 (non-inverted polarity):
Low-to-high transition only.
If CPOL = 1 (inverted polarity):
High-to-low transition only.
00 = Trigger/event/interrupt generation is disabled

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

bit 4 **CREF:** Comparator Reference Select bits (non-inverting input)
1 = Non-inverting input connects to internal CVREF voltage
0 = Non-inverting input connects to CxINA pin

bit 3-2 **Unimplemented:** Read as '0'

bit 1-0 **CCH<1:0>:** Comparator Channel Select bits
11 = Inverting input of comparator connects to VBG/2
00 = Inverting input of comparator connects to CxINB pin

**REGISTER 20-2: CMSTAT: COMPARATOR MODULE STATUS REGISTER**

| R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| CMIDL | — | — | — | — | — | C2EVT | C1EVT |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-0, HSC | R-0, HSC |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | C2OUT | C1OUT |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **CMIDL:** Comparator Stop in Idle Mode bit
1 = When device enters Idle mode, the module does not generate interrupts; it is still enabled
0 = Continue operation of all enabled comparators in Idle mode

bit 14-10 **Unimplemented:** Read as '0'

bit 9 **C2EVT:** Comparator 2 Event Status bit (read-only)
Shows the current event status of Comparator 2 (CM2CON<9>).

bit 8 **C1EVT:** Comparator 1 Event Status bit (read-only)
Shows the current event status of Comparator 1 (CM1CON<9>).

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

bit 1 **C2OUT:** Comparator 2 Output Status bit (read-only)
Shows the current output of Comparator 2 (CM2CON<8>).

bit 0 **C1OUT:** Comparator 1 Output Status bit (read-only)
Shows the current output of Comparator 1 (CM1CON<8>).

## 21.0 COMPARATOR VOLTAGE REFERENCE

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Comparator Voltage Reference, refer to the *"PIC24F Family Reference Manual"*, **Section 20. "Comparator Voltage Reference Module"** (DS39709).

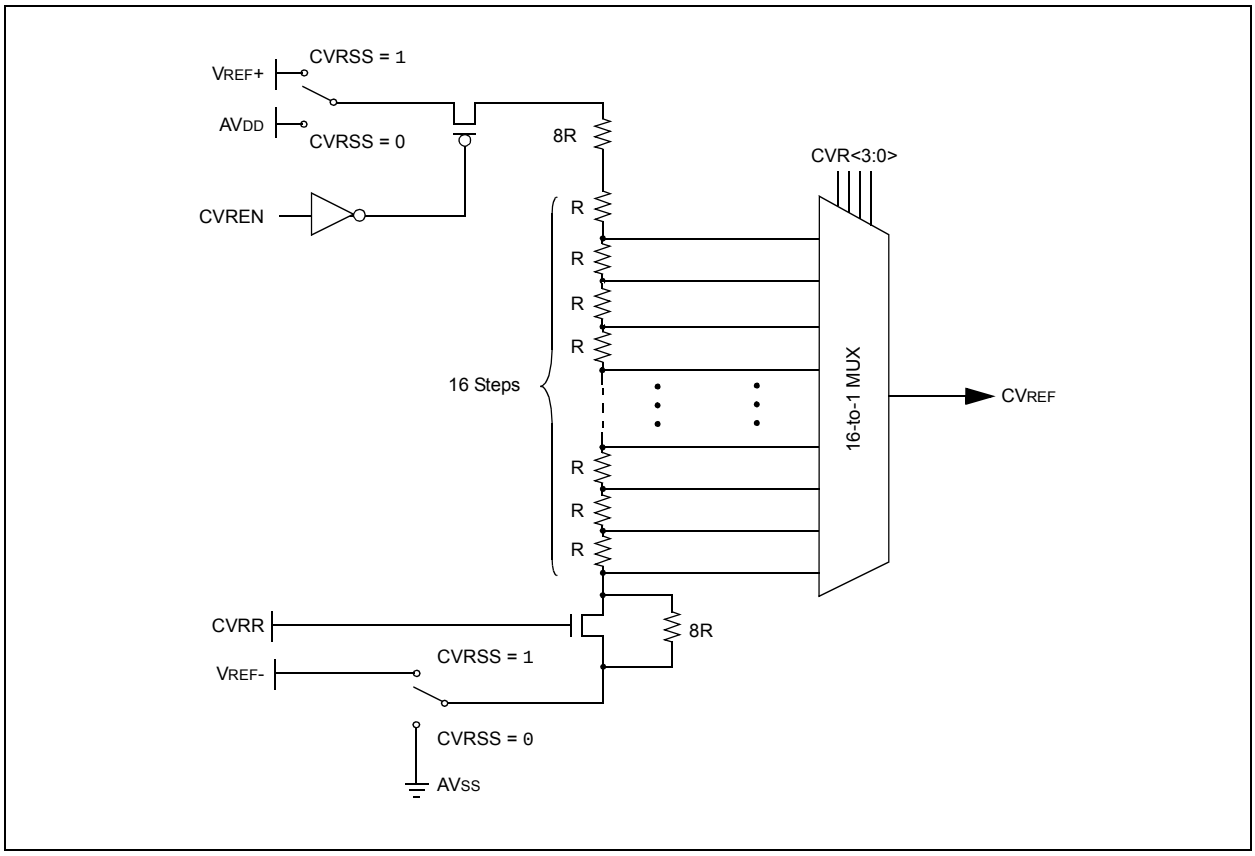
### 21.1 Configuring the Comparator Voltage Reference

The comparator voltage reference module is controlled through the CVRCON register (Register 21-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR<3:0>), with one range offering finer resolution.

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

The settling time of the comparator voltage reference must be considered when changing the CVREF output.

**FIGURE 21-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM**



**REGISTER 21-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 |
| bit 7 | | | | | | | bit 0 |

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

bit 15-8 **Unimplemented:** Read as '0'

bit 7 **CVREN:** Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE:** Comparator VREF Output Enable bit
1 = CVREF voltage level is output on CVREF pin
0 = CVREF voltage level is disconnected from CVREF pin

bit 5 **CVRR:** Comparator VREF Range Selection bit
1 = CVRSRC range should be 0 to 0.625 CVRSRC with CVRSRC/24 step size
0 = CVRSRC range should be 0.25 to 0.719 CVRSRC with CVRSRC/32 step size

bit 4 **CVRSS:** Comparator VREF Source Selection bit
1 = Comparator reference source CVRSRC = VREF+ – VREF-
0 = Comparator reference source CVRSRC = AVDD – AVSS

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

## 22.0 CHARGE TIME MEASUREMENT UNIT (CTMU)

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Charge Measurement Unit, refer to the *"PIC24F Family Reference Manual"*, **Section 11. "Charge Time Measurement Unit (CTMU)"** (DS39724).

The Charge Time Measurement Unit (CTMU) is a flexible analog module that provides charge measurement, accurate differential time measurement between pulse sources and asynchronous pulse generation. Its key features include:

- Four edge input trigger sources
- Polarity control for each edge source
- Control of edge sequence
- Control of response to edges
- Time measurement resolution of one nanosecond
- Accurate current source suitable for capacitive measurement

Together with other on-chip analog modules, the CTMU can be used to precisely measure time, measure capacitance, measure relative changes in capacitance, or generate output pulses that are independent of the system clock. The CTMU module is ideal for interfacing with capacitive-based touch sensors.

The CTMU is controlled through two registers, CTMUCON and CTMUICON. CTMUCON enables the module, and controls edge source selection, edge source polarity selection, and edge sequencing. The CTMUICON register selects the current range of current source and trims the current.

### 22.1 Measuring Capacitance

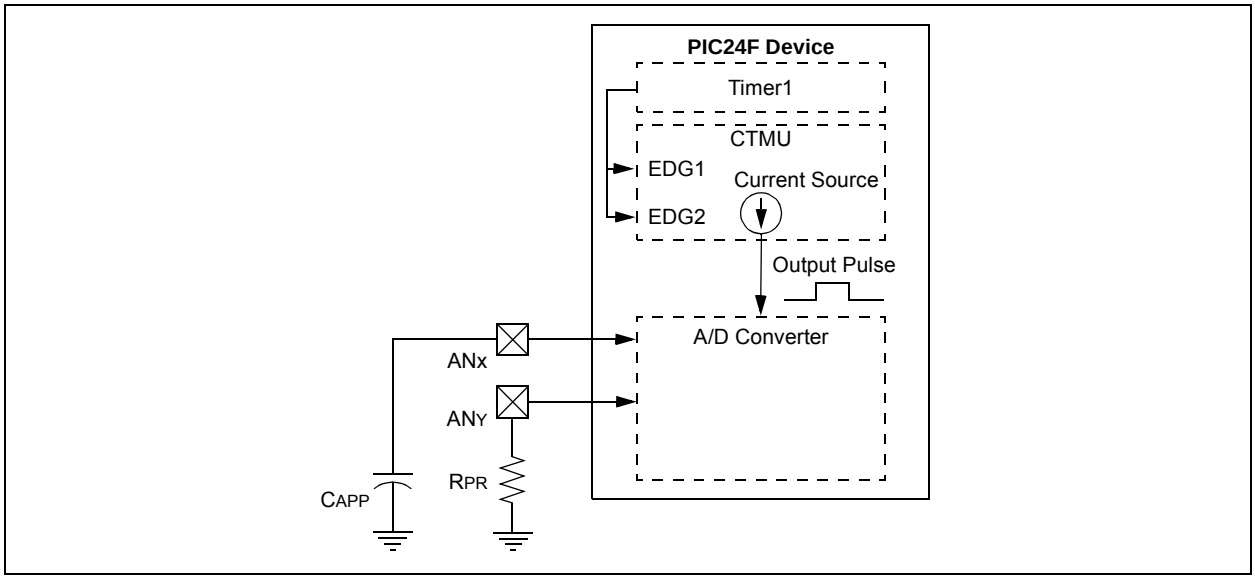
The CTMU module measures capacitance by generating an output pulse with a width equal to the time between edge events on two separate input channels. The pulse edge events to both input channels can be selected from four sources: two internal peripheral modules (OC1 and Timer1) and two external pins (CTEDG1 and CTEDG2). This pulse is used with the module's precision current source to calculate capacitance according to the relationship:

$$C = I \cdot \frac{dV}{dT}$$

For capacitance measurements, the A/D Converter samples an external capacitor (CAPP) on one of its input channels after the CTMU output's pulse. A precision resistor (RPR) provides current source calibration on a second A/D channel. After the pulse ends, the converter determines the voltage on the capacitor. The actual calculation of capacitance is performed in software by the application.

Figure 22-1 displays the external connections used for capacitance measurements, and how the CTMU and A/D modules are related in this application. This example also shows the edge events coming from Timer1, but other configurations using external edge sources are possible. A detailed discussion on measuring capacitance and time with the CTMU module is provided in the "*PIC24F Family Reference Manual*".

**FIGURE 22-1: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR CAPACITANCE MEASUREMENT**



## 22.2 Measuring Time

Time measurements on the pulse width can be similarly performed using the A/D module's internal capacitor (CAD) and a precision resistor for current calibration. Figure 22-2 displays the external connections used for time measurements, and how the CTMU and A/D modules are related in this application. This example also shows both edge events coming from the external CTEDG pins, but other configurations using internal edge sources are possible.

## 22.3 Pulse Generation and Delay

The CTMU module can also generate an output pulse with edges that are not synchronous with the device's system clock. More specifically, it can generate a pulse with a programmable delay from an edge event input to the module.

When the module is configured for pulse generation delay by setting the TGEN bit (CTMUCON<12>), the internal current source is connected to the B input of Comparator 2. A capacitor (CDELAY) is connected to the Comparator 2 pin, C2INB, and the comparator voltage reference, CVREF, is connected to C2INA. CVREF is then configured for a specific trip point. The module begins to charge CDELAY when an edge event is detected. When CDELAY charges above the CVREF trip point, a pulse is output on CTPLS. The length of the pulse delay is determined by the value of CDELAY and the CVREF trip point.

Figure 22-3 shows the external connections for pulse generation, as well as the relationship of the different analog modules required. While CTEDG1 is shown as the input pulse source, other options are available. A detailed discussion on pulse generation with the CTMU module is provided in the "*PIC24F Family Reference Manual*".

**FIGURE 22-2: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR TIME MEASUREMENT**

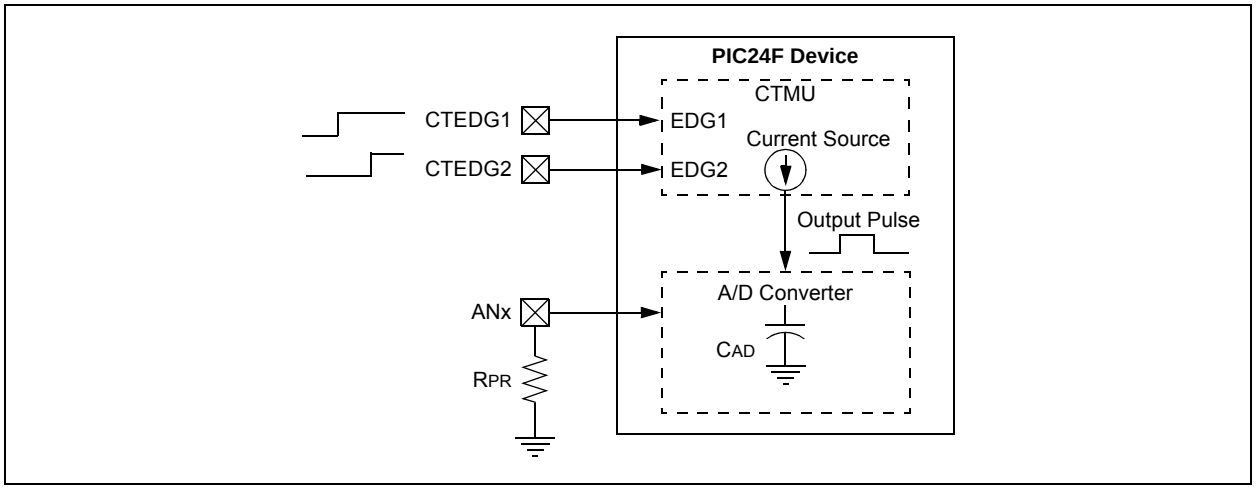
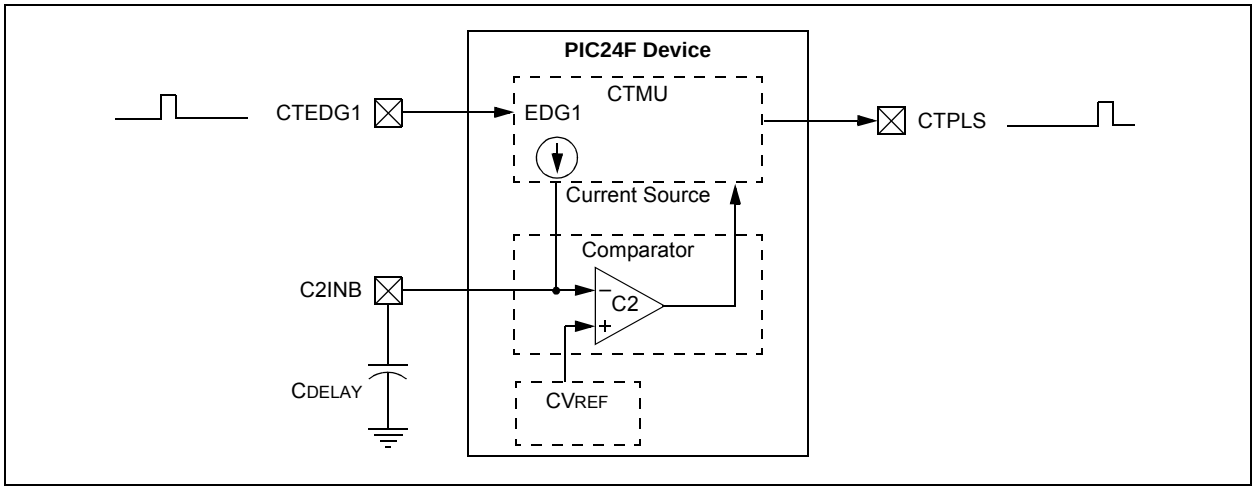



**FIGURE 22-3: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR PULSE DELAY GENERATION**



**REGISTER 22-1: CTMUCON: CTMU CONTROL REGISTER**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CTMUEN | — | CTMUSIDL | TGEN | EDGEN | EDGSEQEN | IDISSEN | CTTRIG |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| EDG2POL | EDG2SEL1 | EDG2SEL0 | EDG1POL | EDG1SEL1 | EDG1SEL0 | EDG2STAT | EDG1STAT |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **CTMUEN:** CTMU Enable bit
1 = Module is enabled
0 = Module is disabled

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **CTMUSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12 **TGEN:** Time Generation Enable bit
1 = Enables edge delay generation
0 = Disables edge delay generation

bit 10 **EDGEN:** Edge Enable bit
1 = Edges are not blocked
0 = Edges are blocked

bit 10 **EDGSEQEN:** Edge Sequence Enable bit
1 = Edge 1 event must occur before Edge 2 event can occur
0 = No edge sequence is needed

bit 9 **IDISSEN:** Analog Current Source Control bit
1 = Analog current source output is grounded
0 = Analog current source output is not grounded

bit 8 **CTTRIG:** Trigger Control bit
1 = Trigger output is enabled
0 = Trigger output is disabled

bit 7 **EDG2POL:** Edge 2 Polarity Select bit
1 = Edge 2 programmed for a positive edge response
0 = Edge 2 programmed for a negative edge response

bit 6-5 **EDG2SEL<1:0>:** Edge 2 Source Select bits
11 = CTED1 pin
10 = CTED2 pin
01 = OC1 module
00 = Timer1 module

bit 4 **EDG1POL:** Edge 1 Polarity Select bit
1 = Edge 1 programmed for a positive edge response
0 = Edge 1 programmed for a negative edge response

**REGISTER 22-1: CTMUCON: CTMU CONTROL REGISTER (CONTINUED)**

bit 3-2 **EDG1SEL<1:0>:** Edge 1 Source Select bits
11 = CTED1 pin
10 = CTED2 pin
01 = OC1 module
00 = Timer1 module

bit 1 **EDG2STAT:** Edge 2 Status bit
1 = Edge 2 event has occurred
0 = Edge 2 event has not occurred

bit 0 **EDG1STAT:** Edge 1 Status bit
1 = Edge 1 event has occurred
0 = Edge 1 event has not occurred

**REGISTER 22-2: CTMUICON: CTMU CURRENT CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ITRIM5 | ITRIM4 | ITRIM3 | ITRIM2 | ITRIM1 | ITRIM0 | IRNG1 | IRNG0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 15-10 **ITRIM<5:0>:** Current Source Trim bits
011111 = Maximum positive change from nominal current
011110
.
.
.
000001 = Minimum positive change from nominal current
000000 = Nominal current output specified by IRNG<1:0>
111111 = Minimum negative change from nominal current
.
.
.
100010
100001 = Maximum negative change from nominal current

bit 9-8 **IRNG<1:0>:** Current Source Range Select bits
11 = 100 × Base current
10 = 10 × Base current
01 = Base current level (0.55 μA nominal)
00 = Current source disabled

bit 7-0 **Unimplemented:** Read as ‘0’

## 23.0 SPECIAL FEATURES

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on the Watchdog Timer, High-Level Device Integration and Programming Diagnostics, refer to the individual sections of the *"PIC24F Family Reference Manual" provided below*:

- **Section 9. "Watchdog Timer (WDT)"** (DS39697)
- **Section 36. "High-Level Integration with Programmable High/Low-Voltage Detect (HLVD)"** (DS39725)
- **Section 33. "Programming and Diagnostics"** (DS39716)

PIC24F04KA201 family devices include several features intended to maximize application flexibility and reliability, and minimize cost through elimination of external components. These are:

- Flexible Configuration
- Watchdog Timer (WDT)
- Code Protection
- In-Circuit Serial Programming™ (ICSP™)
- In-Circuit Emulation

### 23.1 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 starting at program memory location, F80000h. A complete list is provided in Table 23-1. A detailed explanation of the various bit functions is provided in Register 23-1 through Register 23-7.

The address, F80000h, is beyond the user program memory space. In fact, it belongs to the configuration memory space (800000h-FFFFFFh), which can only be accessed using table reads and table writes.

**TABLE 23-1: CONFIGURATION REGISTERS LOCATIONS**

| Configuration Register | Address |
|---|---|
| FGS | F80004 |
| FOSCSEL | F80006 |
| FOSC | F80008 |
| FWDT | F8000A |
| FPOR | F8000C |
| FICD | F8000E |
| FDS | F80010 |

**REGISTER 23-1: FGS: GENERAL SEGMENT CONFIGURATION REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/C-1 | R/C-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | GSS0 | GWRP |
| bit 7 | | | | | | | bit 0 |

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

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

bit 1 **GSS0:** General Segment Code Flash Code Protection bit
1 = No protection
0 = Standard security enabled

bit 0 **GWRP**: General Segment Code Flash Write Protection bit
1 = General segment may be written
0 = General segment is write-protected

**REGISTER 23-2: FOSCSEL: OSCILLATOR SELECTION CONFIGURATION REGISTER**

| R/P-1 | U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| IESO | — | — | — | — | FNOSC2 | FNOSC1 | FNOSC0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **IESO:** Internal External Switchover bit
1 = Internal External Switchover mode enabled (Two-Speed Start-up enabled)
0 = Internal External Switchover mode disabled (Two-Speed Start-up disabled)

bit 6-3 **Unimplemented:** Read as '0'

bit 2-0 **FNOSC<2:0>:** Oscillator Selection bits
000 = Fast RC oscillator (FRC)
001 = Fast RC oscillator with divide-by-N with PLL module (FRCDIV+PLL)
010 = Primary oscillator (XT, HS, EC)
011 = Primary oscillator with PLL module (HS+PLL, EC+PLL)
100 = Secondary oscillator (SOSC)
101 = Low-Power RC oscillator (LPRC)
110 = 500 kHz Low-Power FRC oscillator with divide-by-N (LPFRCDIV)
111 = 8 MHz FRC oscillator with divide-by-N (FRCDIV)

**REGISTER 23-3: FOSC: OSCILLATOR CONFIGURATION REGISTER**

| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| FCKSM1 | FCKSM0 | SOSCSEL | POSCFREQ1 | POSCFREQ0 | OSCIOFNC | POSCMD1 | POSCMD0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **FCKSM<1:0>:** Clock Switching and Monitor Selection Configuration bits
1x = Clock switching is disabled, Fail-Safe Clock Monitor is disabled
01 = Clock switching is enabled, Fail-Safe Clock Monitor is disabled
00 = Clock switching is enabled, Fail-Safe Clock Monitor is enabled

bit 5 **SOSCSEL:** Secondary Oscillator Select Bit
1 = Secondary oscillator configured for high-power operation
0 = Secondary oscillator configured for low-power operation

bit 4-3 **POSCFREQ<1:0>:** Primary Oscillator Frequency Range Configuration bits
11 = Primary oscillator/external clock input frequency greater than 8 MHz
10 = Primary oscillator/external clock input frequency between 100 kHz and 8 MHz
01 = Primary oscillator/external clock input frequency less than 100 kHz
00 = Reserved; do not use

bit 2 **OSCIOFNC:** CLKO Enable Configuration bit
1 = CLKO output signal active on the OSCO pin; primary oscillator must be disabled or configured for the External Clock mode (EC) for the CLKO to be active (POSCMD<1:0> = 11 or 00)
0 = CLKO output disabled

bit 1-0 **POSCMD<1:0>:** Primary Oscillator Configuration bits
11 = Primary oscillator disabled
10 = HS Oscillator mode selected
01 = XT Oscillator mode selected
00 = External Clock mode selected

**REGISTER 23-4: FWDT: WATCHDOG TIMER CONFIGURATION REGISTER**

| R/P-1 | R/P-1 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| FWDTEN | WINDIS | — | FWPSA | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **FWDTEN:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled (control is placed on the SWDTEN bit)

bit 6 **WINDIS:** Windowed Watchdog Timer Disable bit
1 = Standard WDT selected; windowed WDT disabled
0 = Windowed WDT enabled

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

bit 4 **FWPSA:** WDT Prescaler bit
1 = WDT prescaler ratio of 1:128
0 = WDT prescaler ratio of 1:32

bit 3-0 **WDTPS<3:0>:** Watchdog Timer Postscale Select bits
1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

**REGISTER 23-5: FPOR: RESET CONFIGURATION REGISTER**

| R/P-1 | R/P-1 | R/P-1 | U-0 | R/P-1 | U-0 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| MCLRE(1) | BORV1(2) | BORV0(2) | — | PWRTEN | — | BOREN1 | BOREN0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Pin Enable bit(1)
1 = $\overline{\text{MCLR}}$ pin enabled; RA5 input pin disabled
0 = RA5 input pin enabled; $\overline{\text{MCLR}}$ disabled

bit 6-5 **BORV<1:0>:** Brown-out Reset Enable bits(2)
11 = Brown-out Reset set to lowest voltage
10 = Brown-out Reset
01 = Brown-out Reset set to highest voltage
00 = Low-power Brown-out Reset occurs around 2.0V

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

bit 3 **PWRTEN:** Power-up Timer Enable bit
0 = PWRT disabled
1 = PWRT enabled

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

bit 1-0 **BOREN<1:0>:** Brown-out Reset Enable bits
11 = Brown-out Reset enabled in hardware; SBOREN bit disabled
10 = Brown-out Reset enabled only while device is active and disabled in Sleep; SBOREN bit disabled
01 = Brown-out Reset controlled with the SBOREN bit setting
00 = Brown-out Reset disabled in hardware; SBOREN bit disabled

**Note 1:** The MCLRE fuse can only be changed when using the VPP-Based ICSP™ mode entry. This prevents a user from accidentally locking out the device from the low-voltage test entry.
**2:** Refer to the electrical specifications for BOR voltages.

**REGISTER 23-6: FICD: IN-CIRCUIT DEBUGGER CONFIGURATION REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | FICD1 | FICD0 |
| bit 7 | | | | | | | bit 0 |

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

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

bit 1-0 **FICD<1:0:>** ICD Pin Select bits
10 = PGC2/PGD2 are used for programming the device
01 = PGC3/PGD3 are used for programming the device
00, 11 = Reserved; do not use

**REGISTER 23-7: FDS: DEEP SLEEP CONFIGURATION REGISTER**

| R/P-1 | R/P-1 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| DSWDTEN | DSLPBOR | — | — | DSWDTPS3 | DSWDTPS2 | DSWDTPS1 | DSWDTPS0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **DSWDTEN:** Deep Sleep Watchdog Timer Enable bit
1 = DSWDT enabled
0 = DSWDT disabled

bit 6 **DSLPBOR:** Deep Sleep/Low-Power BOR Enable bit (does not affect operation in non Deep Sleep modes)
1 = Deep Sleep BOR enabled in Deep Sleep
0 = Deep Sleep BOR disabled in Deep Sleep

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

bit 3-0 **DSWDTPS<3:0>:** Deep Sleep Watchdog Timer Postscale Select bits
The DSWDT prescaler is 32; this creates an approximate base time unit of 1 ms.
1111 = 1:2,147,483,648 (25.7 days) nominal
1110 = 1:536,870,912 (6.4 days) nominal
1101 = 1:134,217,728 (38.5 hours) nominal
1100 = 1:33,554,432 (9.6 hours) nominal
1011 = 1:8,388,608 (2.4 hours) nominal
1010 = 1:2,097,152 (36 minutes) nominal
1001 = 1:524,288 (9 minutes) nominal
1000 = 1:131,072 (135 seconds) nominal
0111 = 1:32,768 (34 seconds) nominal
0110 = 1:8,192 (8.5 seconds) nominal
0101 = 1:2,048 (2.1 seconds) nominal
0100 = 1:512 (528 ms) nominal
0011 = 1:128 (132 ms) nominal
0010 = 1:32 (33 ms) nominal
0001 = 1:8 (8.3 ms) nominal
0000 = 1:2 (2.1 ms) nominal

**REGISTER 23-8: DEVID: DEVICE ID REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 23 | | | | | | | bit 16 |

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| FAMID7 | FAMID6 | FAMID5 | FAMID4 | FAMID3 | FAMID2 | FAMID1 | FAMID0 |
| bit 15 | | | | | | | bit 8 |

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| DEV7 | DEV6 | DEV5 | DEV4 | DEV3 | DEV2 | DEV1 | DEV0 |
| bit 7 | | | | | | | bit 0 |

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

bit 23-16 **Unimplemented:** Read as ‘0’

bit 15-8 **FAMID<7:0>:** Device Family Identifier bits
00001011 = PIC24F04KA201 family

bit 7-0 **DEV<7:0>:** Individual Device Identifier bits
00000000 = PIC24F04KA201
00000010 = PIC24F04KA200

**REGISTER 23-9: DEVREV: DEVICE REVISION REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 23 | | | | | | | bit 16 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | R | R | R | R |
|---|---|---|---|---|---|---|---|
| — | — | — | — | REV3 | REV2 | REV1 | REV0 |
| bit 7 | | | | | | | bit 0 |

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

bit 23-4 **Unimplemented:** Read as ‘0’

bit 3-0 **REV<3:0>:** Minor Revision Identifier bits

## 23.2 Watchdog Timer (WDT)

For the PIC24F04KA201 family of devices, the WDT is driven by the LPRC oscillator. When the WDT is enabled, the clock source is also enabled.

The nominal WDT clock source from LPRC is 31 kHz. This feeds a prescaler that can be configured for either 5-bit (divide-by-32) or 7-bit (divide-by-128) operation. The prescaler is set by the FWPSA Configuration bit. With a 31 kHz input, the prescaler yields a nominal WDT time-out period (TWDT) of 1 ms in 5-bit mode or 4 ms in 7-bit mode.

A variable postscaler divides down the WDT prescaler output and allows for a wide range of time-out periods. The postscaler is controlled by the Configuration bits, WDTPS<3:0> (FWDT<3:0>), which allow the selection of a total of 16 settings, from 1:1 to 1:32,768. Using the prescaler and postscaler, time-out periods ranging from 1 ms to 131 seconds can be achieved.

The WDT, prescaler and postscaler are reset:

- On any device Reset
- On the completion of a clock switch, whether invoked by software (i.e., setting the OSWEN bit after changing the NOSC bits) or by hardware (i.e., Fail-Safe Clock Monitor)
- When a PWRSAV instruction is executed (i.e., Sleep or Idle mode is entered)
- When the device exits Sleep or Idle mode to resume normal operation
- By a CLRWDT instruction during normal execution

If the WDT is enabled, it will continue to run during Sleep or Idle modes. When the WDT time-out occurs, the device will wake the device and code execution will continue from where the PWRSAV instruction was executed. The corresponding SLEEP or IDLE bits (RCON<3:2>) will need to be cleared in software after the device wakes up.

The WDT Flag bit, WDTO (RCON<4>), is not automatically cleared following a WDT time-out. To detect subsequent WDT events, the flag must be cleared in software.

| **Note:** | The CLRWDT and PWRSAV instructions clear the prescaler and postscaler counts when executed. |
|---|---|

### 23.2.1 WINDOWED OPERATION

The Watchdog Timer has an optional Fixed Window mode of operation. In this Windowed mode, CLRWDT instructions can only reset the WDT during the last 1/4 of the programmed WDT period. A CLRWDT instruction executed before that window causes a WDT Reset, similar to a WDT time-out.

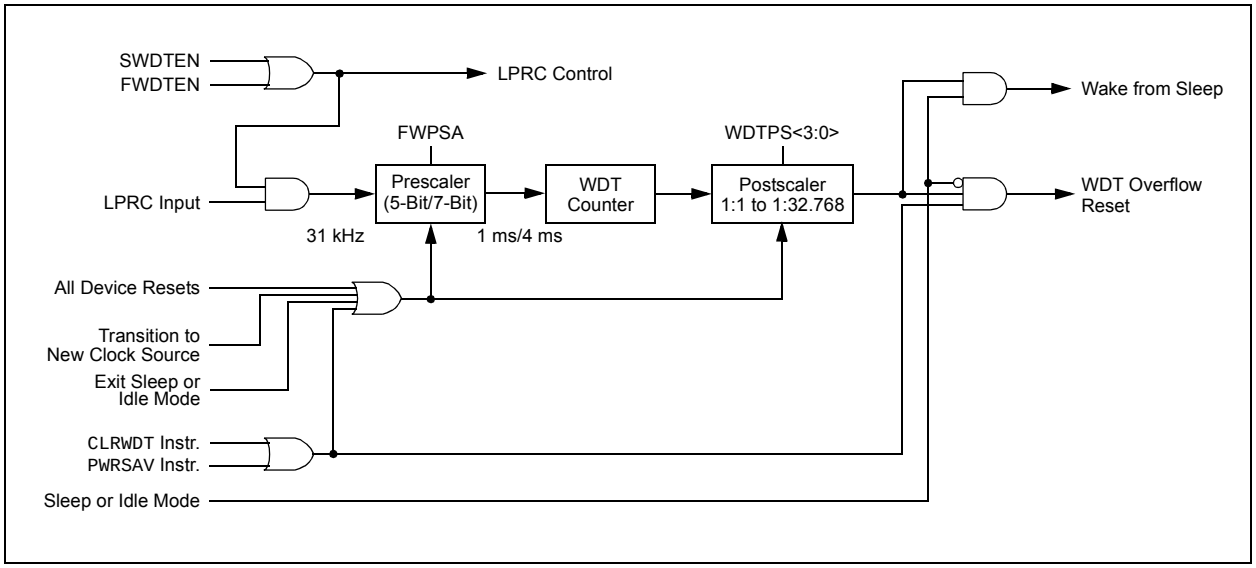
Windowed WDT mode is enabled by programming the Configuration bit, WINDIS (FWDT<6>), to '0'.

### 23.2.2 CONTROL REGISTER

The WDT is enabled or disabled by the FWDTEN Configuration bit. When the FWDTEN Configuration bit is set, the WDT is always enabled.

The WDT can be optionally controlled in software when the FWDTEN Configuration bit has been programmed to '0'. The WDT is enabled in software by setting the SWDTEN control bit (RCON<5>). The SWDTEN control bit is cleared on any device Reset. The software WDT option allows the user to enable the WDT for critical code segments and disable the WDT during non-critical segments for maximum power savings.

**FIGURE 23-1: WDT BLOCK DIAGRAM**



## 23.3 Deep Sleep Watchdog Timer (DSWDT)

In PIC24F04KA201 family devices, in addition to the WDT module, a DSWDT module is present which runs while the device is in Deep Sleep, if enabled. It is driven by either the SOSC or LPRC oscillator. The clock source is selected by the Configuration bit, DSWCKSEL (FDS<4>).

The DSWDT can be configured to generate a time-out at 2.1 ms to 25.7 days by selecting the respective postscaler. The postscaler can be selected by the Configuration bits, DSWDTPS<3:0> (FDS<3:0>). When the DSWDT is enabled, the clock source is also enabled.

DSWDT is one of the sources that can wake-up the device from Deep Sleep mode.

## 23.4 Program Verification and Code Protection

For all devices in the PIC24F04KA201 family, code protection for the general segment is controlled by the Configuration bit, GSS0. This bit inhibits external reads and writes to the program memory space; this has no direct effect in normal execution mode.

Write protection is controlled by the GWRP bit for the general segment in the Configuration Word. When this bit is programmed to '0', internal write and erase operations to program memory are blocked.

## 23.5 In-Circuit Serial Programming

PIC24F04KA201 family microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock (PGCx) and data (PGDx) and three other lines for power, ground and the programming voltage. 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.

**NOTES:**

## 24.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

### 24.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

## 24.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16 and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

## 24.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

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, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

## 24.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a 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 object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

## 24.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

## 24.6 MPLAB X SIM Software Simulator

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

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

## 24.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

## 24.8 MPLAB ICD 3 In-Circuit Debugger System

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

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

## 24.9 PICkit 3 In-Circuit Debugger/ Programmer

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

## 24.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at $V_{DDMIN}$ and $V_{DDMAX}$ for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

## 24.11 Demonstration/Development Boards, Evaluation Kits and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

## 24.12 Third-Party Development Tools

Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

## 25.0 INSTRUCTION SET SUMMARY

| | |
|---|---|
| **Note:** | This chapter is a brief summary of the PIC24F instruction set architecture and is not intended to be a comprehensive reference source. |

The PIC24F instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from previous PIC MCU instruction sets. Most instructions are a single program memory word. Only three instructions require two program memory locations.

Each single-word instruction is a 24-bit word divided into an 8-bit opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction. The instruction set is highly orthogonal and is grouped into four basic categories:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- Control operations

Table 25-1 lists the general symbols used in describing the instructions. The PIC24F instruction set summary in Table 25-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand, which is typically a register ‘Wb’ without any address modifier
- The second source operand, which is typically a register ‘Ws’ with or without an address modifier
- The destination of the result, which is typically a register ‘Wd’ with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register specified by the value, ‘f’
- The destination, which could either be the file register, ‘f’, or the W0 register, which is denoted as ‘WREG’

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of ‘Ws’ or ‘f’)
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register ‘Wb’)

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of ‘k’)
- The W register or file register where the literal value is to be loaded (specified by ‘Wb’ or ‘f’)

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand, which is a register ‘Wb’ without any address modifier
- The second source operand, which is a literal value
- The destination of the result (only if not the same as the first source operand), which is typically a register ‘Wd’ with or without an address modifier

The control instructions may use some of the following operands:

- A program memory address
- The mode of the table read and table write instructions

All instructions are a single word, except for certain double-word instructions, which were made double-word instructions so that all of the required information is available in these 48 bits. In the second word, the 8 MSbs are ‘0’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

Most single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter (PC) is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a NOP. Notable exceptions are the BRA (unconditional/computed branch), indirect CALL/GOTO, all table reads and writes, and RETURN/RETFIE instructions, which are single-word instructions but take two or three cycles.

Certain instructions that involve skipping over the subsequent instruction require either two or three cycles if the skip is performed, depending on whether the instruction being skipped is a single-word or two-word instruction. Moreover, double-word moves require two cycles. The double-word instructions execute in two instruction cycles.

**TABLE 25-1: SYMBOLS USED IN OPCODE DESCRIPTIONS**

| Field | Description |
|---|---|
| #text | Means literal defined by "text" |
| (text) | Means "content of text" |
| [text] | Means "the location addressed by text" |
| { } | Optional field or operation |
| <n:m> | Register bit field |
| .b | Byte mode selection |
| .d | Double-Word mode selection |
| .S | Shadow register select |
| .w | Word mode selection (default) |
| bit4 | 4-bit bit selection field (used in word addressed instructions) ∈ {0...15} |
| C, DC, N, OV, Z | MCU Status bits: Carry, Digit Carry, Negative, Overflow, Sticky Zero |
| Expr | Absolute address, label or expression (resolved by the linker) |
| f | File register address ∈ {0000h...1FFFh} |
| lit1 | 1-bit unsigned literal ∈ {0,1} |
| lit4 | 4-bit unsigned literal ∈ {0...15} |
| lit5 | 5-bit unsigned literal ∈ {0...31} |
| lit8 | 8-bit unsigned literal ∈ {0...255} |
| lit10 | 10-bit unsigned literal ∈ {0...255} for Byte mode, {0:1023} for Word mode |
| lit14 | 14-bit unsigned literal ∈ {0...16384} |
| lit16 | 16-bit unsigned literal ∈ {0...65535} |
| lit23 | 23-bit unsigned literal ∈ {0...8388608}; LSB must be '0' |
| None | Field does not require an entry, may be blank |
| PC | Program Counter |
| Slit10 | 10-bit signed literal ∈ {-512...511} |
| Slit16 | 16-bit signed literal ∈ {-32768...32767} |
| Slit6 | 6-bit signed literal ∈ {-16...16} |
| Wb | Base W register ∈ {W0..W15} |
| Wd | Destination W register ∈ { Wd, [Wd], [Wd++], [Wd--], [++Wd], [--Wd] } |
| Wdo | Destination W register ∈<br>{ Wnd, [Wnd], [Wnd++], [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] } |
| Wm,Wn | Dividend, Divisor working register pair (direct addressing) |
| Wn | One of 16 working registers ∈ {W0..W15} |
| Wnd | One of 16 destination working registers ∈ {W0..W15} |
| Wns | One of 16 source working registers ∈ {W0..W15} |
| WREG | W0 (working register used in file register instructions) |
| Ws | Source W register ∈ { Ws, [Ws], [Ws++], [Ws--], [++Ws], [--Ws] } |
| Wso | Source W register ∈ { Wns, [Wns], [Wns++], [Wns--], [++Wns], [--Wns], [Wns+Wb] } |

**TABLE 25-2: INSTRUCTION SET OVERVIEW**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| ADD | ADD | f | f = f + WREG | 1 | 1 | C, DC, N, OV, Z |
| | ADD | f,WREG | WREG = f + WREG | 1 | 1 | C, DC, N, OV, Z |
| | ADD | #lit10,Wn | Wd = lit10 + Wd | 1 | 1 | C, DC, N, OV, Z |
| | ADD | Wb,Ws,Wd | Wd = Wb + Ws | 1 | 1 | C, DC, N, OV, Z |
| | ADD | Wb,#lit5,Wd | Wd = Wb + lit5 | 1 | 1 | C, DC, N, OV, Z |
| ADDC | ADDC | f | f = f + WREG + (C) | 1 | 1 | C, DC, N, OV, Z |
| | ADDC | f,WREG | WREG = f + WREG + (C) | 1 | 1 | C, DC, N, OV, Z |
| | ADDC | #lit10,Wn | Wd = lit10 + Wd + (C) | 1 | 1 | C, DC, N, OV, Z |
| | ADDC | Wb,Ws,Wd | Wd = Wb + Ws + (C) | 1 | 1 | C, DC, N, OV, Z |
| | ADDC | Wb,#lit5,Wd | Wd = Wb + lit5 + (C) | 1 | 1 | C, DC, N, OV, Z |
| AND | AND | f | f = f .AND. WREG | 1 | 1 | N, Z |
| | AND | f,WREG | WREG = f .AND. WREG | 1 | 1 | N, Z |
| | AND | #lit10,Wn | Wd = lit10 .AND. Wd | 1 | 1 | N, Z |
| | AND | Wb,Ws,Wd | Wd = Wb .AND. Ws | 1 | 1 | N, Z |
| | AND | Wb,#lit5,Wd | Wd = Wb .AND. lit5 | 1 | 1 | N, Z |
| ASR | ASR | f | f = Arithmetic Right Shift f | 1 | 1 | C, N, OV, Z |
| | ASR | f,WREG | WREG = Arithmetic Right Shift f | 1 | 1 | C, N, OV, Z |
| | ASR | Ws,Wd | Wd = Arithmetic Right Shift Ws | 1 | 1 | C, N, OV, Z |
| | ASR | Wb,Wns,Wnd | Wnd = Arithmetic Right Shift Wb by Wns | 1 | 1 | N, Z |
| | ASR | Wb,#lit5,Wnd | Wnd = Arithmetic Right Shift Wb by lit5 | 1 | 1 | N, Z |
| BCLR | BCLR | f,#bit4 | Bit Clear f | 1 | 1 | None |
| | BCLR | Ws,#bit4 | Bit Clear Ws | 1 | 1 | None |
| BRA | BRA | C,Expr | Branch if Carry | 1 | 1 (2) | None |
| | BRA | GE,Expr | Branch if Greater than or Equal | 1 | 1 (2) | None |
| | BRA | GEU,Expr | Branch if Unsigned Greater than or Equal | 1 | 1 (2) | None |
| | BRA | GT,Expr | Branch if Greater than | 1 | 1 (2) | None |
| | BRA | GTU,Expr | Branch if Unsigned Greater than | 1 | 1 (2) | None |
| | BRA | LE,Expr | Branch if Less than or Equal | 1 | 1 (2) | None |
| | BRA | LEU,Expr | Branch if Unsigned Less than or Equal | 1 | 1 (2) | None |
| | BRA | LT,Expr | Branch if Less than | 1 | 1 (2) | None |
| | BRA | LTU,Expr | Branch if Unsigned Less than | 1 | 1 (2) | None |
| | BRA | N,Expr | Branch if Negative | 1 | 1 (2) | None |
| | BRA | NC,Expr | Branch if Not Carry | 1 | 1 (2) | None |
| | BRA | NN,Expr | Branch if Not Negative | 1 | 1 (2) | None |
| | BRA | NOV,Expr | Branch if Not Overflow | 1 | 1 (2) | None |
| | BRA | NZ,Expr | Branch if Not Zero | 1 | 1 (2) | None |
| | BRA | OV,Expr | Branch if Overflow | 1 | 1 (2) | None |
| | BRA | Expr | Branch Unconditionally | 1 | 2 | None |
| | BRA | Z,Expr | Branch if Zero | 1 | 1 (2) | None |
| | BRA | Wn | Computed Branch | 1 | 2 | None |
| BSET | BSET | f,#bit4 | Bit Set f | 1 | 1 | None |
| | BSET | Ws,#bit4 | Bit Set Ws | 1 | 1 | None |
| BSW | BSW.C | Ws,Wb | Write C bit to Ws<Wb> | 1 | 1 | None |
| | BSW.Z | Ws,Wb | Write Z bit to Ws<Wb> | 1 | 1 | None |
| BTG | BTG | f,#bit4 | Bit Toggle f | 1 | 1 | None |
| | BTG | Ws,#bit4 | Bit Toggle Ws | 1 | 1 | None |
| BTSC | BTSC | f,#bit4 | Bit Test f, Skip if Clear | 1 | 1 (2 or 3) | None |
| | BTSC | Ws,#bit4 | Bit Test Ws, Skip if Clear | 1 | 1 (2 or 3) | None |

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| BTSS | BTSS | f,#bit4 | Bit Test f, Skip if Set | 1 | 1 (2 or 3) | None |
| | BTSS | Ws,#bit4 | Bit Test Ws, Skip if Set | 1 | 1 (2 or 3) | None |
| BTST | BTST | f,#bit4 | Bit Test f | 1 | 1 | Z |
| | BTST.C | Ws,#bit4 | Bit Test Ws to C | 1 | 1 | C |
| | BTST.Z | Ws,#bit4 | Bit Test Ws to Z | 1 | 1 | Z |
| | BTST.C | Ws,Wb | Bit Test Ws<Wb> to C | 1 | 1 | C |
| | BTST.Z | Ws,Wb | Bit Test Ws<Wb> to Z | 1 | 1 | Z |
| BTSTS | BTSTS | f,#bit4 | Bit Test then Set f | 1 | 1 | Z |
| | BTSTS.C | Ws,#bit4 | Bit Test Ws to C, then Set | 1 | 1 | C |
| | BTSTS.Z | Ws,#bit4 | Bit Test Ws to Z, then Set | 1 | 1 | Z |
| CALL | CALL | lit23 | Call Subroutine | 2 | 2 | None |
| | CALL | Wn | Call Indirect Subroutine | 1 | 2 | None |
| CLR | CLR | f | f = 0x0000 | 1 | 1 | None |
| | CLR | WREG | WREG = 0x0000 | 1 | 1 | None |
| | CLR | Ws | Ws = 0x0000 | 1 | 1 | None |
| CLRWDT | CLRWDT | | Clear Watchdog Timer | 1 | 1 | WDTO, Sleep |
| COM | COM | f | $f = \overline{f}$ | 1 | 1 | N, Z |
| | COM | f,WREG | $WREG = \overline{f}$ | 1 | 1 | N, Z |
| | COM | Ws,Wd | $Wd = \overline{Ws}$ | 1 | 1 | N, Z |
| CP | CP | f | Compare f with WREG | 1 | 1 | C, DC, N, OV, Z |
| | CP | Wb,#lit5 | Compare Wb with lit5 | 1 | 1 | C, DC, N, OV, Z |
| | CP | Wb,Ws | Compare Wb with Ws (Wb – Ws) | 1 | 1 | C, DC, N, OV, Z |
| CP0 | CP0 | f | Compare f with 0x0000 | 1 | 1 | C, DC, N, OV, Z |
| | CP0 | Ws | Compare Ws with 0x0000 | 1 | 1 | C, DC, N, OV, Z |
| CPB | CPB | f | Compare f with WREG, with Borrow | 1 | 1 | C, DC, N, OV, Z |
| | CPB | Wb,#lit5 | Compare Wb with lit5, with Borrow | 1 | 1 | C, DC, N, OV, Z |
| | CPB | Wb,Ws | Compare Wb with Ws, with Borrow ($Wb – Ws – \overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| CPSEQ | CPSEQ | Wb,Wn | Compare Wb with Wn, Skip if = | 1 | 1 (2 or 3) | None |
| CPSGT | CPSGT | Wb,Wn | Compare Wb with Wn, Skip if > | 1 | 1 (2 or 3) | None |
| CPSLT | CPSLT | Wb,Wn | Compare Wb with Wn, Skip if < | 1 | 1 (2 or 3) | None |
| CPSNE | CPSNE | Wb,Wn | Compare Wb with Wn, Skip if ≠ | 1 | 1 (2 or 3) | None |
| DAW | DAW | Wn | Wn = Decimal Adjust Wn | 1 | 1 | C |
| DEC | DEC | f | f = f –1 | 1 | 1 | C, DC, N, OV, Z |
| | DEC | f,WREG | WREG = f –1 | 1 | 1 | C, DC, N, OV, Z |
| | DEC | Ws,Wd | Wd = Ws – 1 | 1 | 1 | C, DC, N, OV, Z |
| DEC2 | DEC2 | f | f = f – 2 | 1 | 1 | C, DC, N, OV, Z |
| | DEC2 | f,WREG | WREG = f – 2 | 1 | 1 | C, DC, N, OV, Z |
| | DEC2 | Ws,Wd | Wd = Ws – 2 | 1 | 1 | C, DC, N, OV, Z |
| DISI | DISI | #lit14 | Disable Interrupts for k Instruction Cycles | 1 | 1 | None |
| DIV | DIV.SW | Wm,Wn | Signed 16/16-bit Integer Divide | 1 | 18 | N, Z, C, OV |
| | DIV.SD | Wm,Wn | Signed 32/16-bit Integer Divide | 1 | 18 | N, Z, C, OV |
| | DIV.UW | Wm,Wn | Unsigned 16/16-bit Integer Divide | 1 | 18 | N, Z, C, OV |
| | DIV.UD | Wm,Wn | Unsigned 32/16-bit Integer Divide | 1 | 18 | N, Z, C, OV |
| EXCH | EXCH | Wns,Wnd | Swap Wns with Wnd | 1 | 1 | None |
| FF1L | FF1L | Ws,Wnd | Find First One from Left (MSb) Side | 1 | 1 | C |
| FF1R | FF1R | Ws,Wnd | Find First One from Right (LSb) Side | 1 | 1 | C |

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| GOTO | GOTO | Expr | Go to Address | 2 | 2 | None |
| | GOTO | Wn | Go to Indirect | 1 | 2 | None |
| INC | INC | f | f = f + 1 | 1 | 1 | C, DC, N, OV, Z |
| | INC | f,WREG | WREG = f + 1 | 1 | 1 | C, DC, N, OV, Z |
| | INC | Ws,Wd | Wd = Ws + 1 | 1 | 1 | C, DC, N, OV, Z |
| INC2 | INC2 | f | f = f + 2 | 1 | 1 | C, DC, N, OV, Z |
| | INC2 | f,WREG | WREG = f + 2 | 1 | 1 | C, DC, N, OV, Z |
| | INC2 | Ws,Wd | Wd = Ws + 2 | 1 | 1 | C, DC, N, OV, Z |
| IOR | IOR | f | f = f .IOR. WREG | 1 | 1 | N, Z |
| | IOR | f,WREG | WREG = f .IOR. WREG | 1 | 1 | N, Z |
| | IOR | #lit10,Wn | Wd = lit10 .IOR. Wd | 1 | 1 | N, Z |
| | IOR | Wb,Ws,Wd | Wd = Wb .IOR. Ws | 1 | 1 | N, Z |
| | IOR | Wb,#lit5,Wd | Wd = Wb .IOR. lit5 | 1 | 1 | N, Z |
| LNK | LNK | #lit14 | Link Frame Pointer | 1 | 1 | None |
| LSR | LSR | f | f = Logical Right Shift f | 1 | 1 | C, N, OV, Z |
| | LSR | f,WREG | WREG = Logical Right Shift f | 1 | 1 | C, N, OV, Z |
| | LSR | Ws,Wd | Wd = Logical Right Shift Ws | 1 | 1 | C, N, OV, Z |
| | LSR | Wb,Wns,Wnd | Wnd = Logical Right Shift Wb by Wns | 1 | 1 | N, Z |
| | LSR | Wb,#lit5,Wnd | Wnd = Logical Right Shift Wb by lit5 | 1 | 1 | N, Z |
| MOV | MOV | f,Wn | Move f to Wn | 1 | 1 | None |
| | MOV | [Wns+Slit10],Wnd | Move [Wns+Slit10] to Wnd | 1 | 1 | None |
| | MOV | f | Move f to f | 1 | 1 | N, Z |
| | MOV | f,WREG | Move f to WREG | 1 | 1 | N, Z |
| | MOV | #lit16,Wn | Move 16-bit Literal to Wn | 1 | 1 | None |
| | MOV.b | #lit8,Wn | Move 8-bit Literal to Wn | 1 | 1 | None |
| | MOV | Wn,f | Move Wn to f | 1 | 1 | None |
| | MOV | Wns,[Wns+Slit10] | Move Wns to [Wns+Slit10] | 1 | 1 | None |
| | MOV | Wso,Wdo | Move Ws to Wd | 1 | 1 | None |
| | MOV | WREG,f | Move WREG to f | 1 | 1 | N, Z |
| | MOV.D | Wns,Wd | Move Double from W(ns):W(ns+1) to Wd | 1 | 2 | None |
| | MOV.D | Ws,Wnd | Move Double from Ws to W(nd+1):W(nd) | 1 | 2 | None |
| MUL | MUL.SS | Wb,Ws,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Signed(Ws) | 1 | 1 | None |
| | MUL.SU | Wb,Ws,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Unsigned(Ws) | 1 | 1 | None |
| | MUL.US | Wb,Ws,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Signed(Ws) | 1 | 1 | None |
| | MUL.UU | Wb,Ws,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(Ws) | 1 | 1 | None |
| | MUL.SU | Wb,#lit5,Wnd | {Wnd+1, Wnd} = Signed(Wb) * Unsigned(lit5) | 1 | 1 | None |
| | MUL.UU | Wb,#lit5,Wnd | {Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(lit5) | 1 | 1 | None |
| | MUL | f | W3:W2 = f * WREG | 1 | 1 | None |
| NEG | NEG | f | f = $\overline{f}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| | NEG | f,WREG | WREG = $\overline{f}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| | NEG | Ws,Wd | Wd = $\overline{Ws}$ + 1 | 1 | 1 | C, DC, N, OV, Z |
| NOP | NOP | | No Operation | 1 | 1 | None |
| | NOPR | | No Operation | 1 | 1 | None |
| POP | POP | f | Pop f from Top-of-Stack (TOS) | 1 | 1 | None |
| | POP | Wdo | Pop from Top-of-Stack (TOS) to Wdo | 1 | 1 | None |
| | POP.D | Wnd | Pop from Top-of-Stack (TOS) to W(nd):W(nd+1) | 1 | 2 | None |
| | POP.S | | Pop Shadow Registers | 1 | 1 | All |
| PUSH | PUSH | f | Push f to Top-of-Stack (TOS) | 1 | 1 | None |
| | PUSH | Wso | Push Wso to Top-of-Stack (TOS) | 1 | 1 | None |
| | PUSH.D | Wns | Push W(ns):W(ns+1) to Top-of-Stack (TOS) | 1 | 2 | None |
| | PUSH.S | | Push Shadow Registers | 1 | 1 | None |

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| PWRSAV | PWRSAV | #lit1 | Go into Sleep or Idle mode | 1 | 1 | WDTO, Sleep |
| RCALL | RCALL | Expr | Relative Call | 1 | 2 | None |
| | RCALL | Wn | Computed Call | 1 | 2 | None |
| REPEAT | REPEAT | #lit14 | Repeat Next Instruction lit14 + 1 times | 1 | 1 | None |
| | REPEAT | Wn | Repeat Next Instruction (Wn) + 1 times | 1 | 1 | None |
| RESET | RESET | | Software Device Reset | 1 | 1 | None |
| RETFIE | RETFIE | | Return from Interrupt | 1 | 3 (2) | None |
| RETLW | RETLW | #lit10,Wn | Return with Literal in Wn | 1 | 3 (2) | None |
| RETURN | RETURN | | Return from Subroutine | 1 | 3 (2) | None |
| RLC | RLC | f | f = Rotate Left through Carry f | 1 | 1 | C, N, Z |
| | RLC | f,WREG | WREG = Rotate Left through Carry f | 1 | 1 | C, N, Z |
| | RLC | Ws,Wd | Wd = Rotate Left through Carry Ws | 1 | 1 | C, N, Z |
| RLNC | RLNC | f | f = Rotate Left (No Carry) f | 1 | 1 | N, Z |
| | RLNC | f,WREG | WREG = Rotate Left (No Carry) f | 1 | 1 | N, Z |
| | RLNC | Ws,Wd | Wd = Rotate Left (No Carry) Ws | 1 | 1 | N, Z |
| RRC | RRC | f | f = Rotate Right through Carry f | 1 | 1 | C, N, Z |
| | RRC | f,WREG | WREG = Rotate Right through Carry f | 1 | 1 | C, N, Z |
| | RRC | Ws,Wd | Wd = Rotate Right through Carry Ws | 1 | 1 | C, N, Z |
| RRNC | RRNC | f | f = Rotate Right (No Carry) f | 1 | 1 | N, Z |
| | RRNC | f,WREG | WREG = Rotate Right (No Carry) f | 1 | 1 | N, Z |
| | RRNC | Ws,Wd | Wd = Rotate Right (No Carry) Ws | 1 | 1 | N, Z |
| SE | SE | Ws,Wnd | Wnd = Sign-Extended Ws | 1 | 1 | C, N, Z |
| SETM | SETM | f | f = FFFFh | 1 | 1 | None |
| | SETM | WREG | WREG = FFFFh | 1 | 1 | None |
| | SETM | Ws | Ws = FFFFh | 1 | 1 | None |
| SL | SL | f | f = Left Shift f | 1 | 1 | C, N, OV, Z |
| | SL | f,WREG | WREG = Left Shift f | 1 | 1 | C, N, OV, Z |
| | SL | Ws,Wd | Wd = Left Shift Ws | 1 | 1 | C, N, OV, Z |
| | SL | Wb,Wns,Wnd | Wnd = Left Shift Wb by Wns | 1 | 1 | N, Z |
| | SL | Wb,#lit5,Wnd | Wnd = Left Shift Wb by lit5 | 1 | 1 | N, Z |
| SUB | SUB | f | f = f – WREG | 1 | 1 | C, DC, N, OV, Z |
| | SUB | f,WREG | WREG = f – WREG | 1 | 1 | C, DC, N, OV, Z |
| | SUB | #lit10,Wn | Wn = Wn – lit10 | 1 | 1 | C, DC, N, OV, Z |
| | SUB | Wb,Ws,Wd | Wd = Wb – Ws | 1 | 1 | C, DC, N, OV, Z |
| | SUB | Wb,#lit5,Wd | Wd = Wb – lit5 | 1 | 1 | C, DC, N, OV, Z |
| SUBB | SUBB | f | f = f – WREG – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBB | f,WREG | WREG = f – WREG – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBB | #lit10,Wn | Wn = Wn – lit10 – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBB | Wb,Ws,Wd | Wd = Wb – Ws – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBB | Wb,#lit5,Wd | Wd = Wb – lit5 – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| SUBR | SUBR | f | f = WREG – f | 1 | 1 | C, DC, N, OV, Z |
| | SUBR | f,WREG | WREG = WREG – f | 1 | 1 | C, DC, N, OV, Z |
| | SUBR | Wb,Ws,Wd | Wd = Ws – Wb | 1 | 1 | C, DC, N, OV, Z |
| | SUBR | Wb,#lit5,Wd | Wd = lit5 – Wb | 1 | 1 | C, DC, N, OV, Z |
| SUBBR | SUBBR | f | f = WREG – f – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBBR | f,WREG | WREG = WREG – f – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBBR | Wb,Ws,Wd | Wd = Ws – Wb – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| | SUBBR | Wb,#lit5,Wd | Wd = lit5 – Wb – ($\overline{C}$) | 1 | 1 | C, DC, N, OV, Z |
| SWAP | SWAP.b | Wn | Wn = Nibble Swap Wn | 1 | 1 | None |
| | SWAP | Wn | Wn = Byte Swap Wn | 1 | 1 | None |
| TBLRDH | TBLRDH | Ws,Wd | Read Prog<23:16> to Wd<7:0> | 1 | 2 | None |

**TABLE 25-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Assembly Mnemonic | Assembly Syntax | | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| TBLRDL | TBLRDL | Ws,Wd | Read Prog<15:0> to Wd | 1 | 2 | None |
| TBLWTH | TBLWTH | Ws,Wd | Write Ws<7:0> to Prog<23:16> | 1 | 2 | None |
| TBLWTL | TBLWTL | Ws,Wd | Write Ws to Prog<15:0> | 1 | 2 | None |
| ULNK | ULNK | | Unlink Frame Pointer | 1 | 1 | None |
| XOR | XOR | f | f = f .XOR. WREG | 1 | 1 | N, Z |
| | XOR | f,WREG | WREG = f .XOR. WREG | 1 | 1 | N, Z |
| | XOR | #lit10,Wn | Wd = lit10 .XOR. Wd | 1 | 1 | N, Z |
| | XOR | Wb,Ws,Wd | Wd = Wb .XOR. Ws | 1 | 1 | N, Z |
| | XOR | Wb,#lit5,Wd | Wd = Wb .XOR. lit5 | 1 | 1 | N, Z |
| ZE | ZE | Ws,Wnd | Wnd = Zero-Extend Ws | 1 | 1 | C, Z, N |

**NOTES:**

## 26.0 ELECTRICAL CHARACTERISTICS

This section provides an overview of the PIC24F04KA201 family electrical characteristics. Additional information will be provided in future revisions of this document as it becomes available.

Absolute maximum ratings for the PIC24F04KA201 family are listed below. Exposure to these maximum rating conditions for extended periods may affect device reliability. Functional operation of the device at these, or any other conditions above the parameters indicated in the operation listings of this specification, is not implied.

### Absolute Maximum Ratings(†)

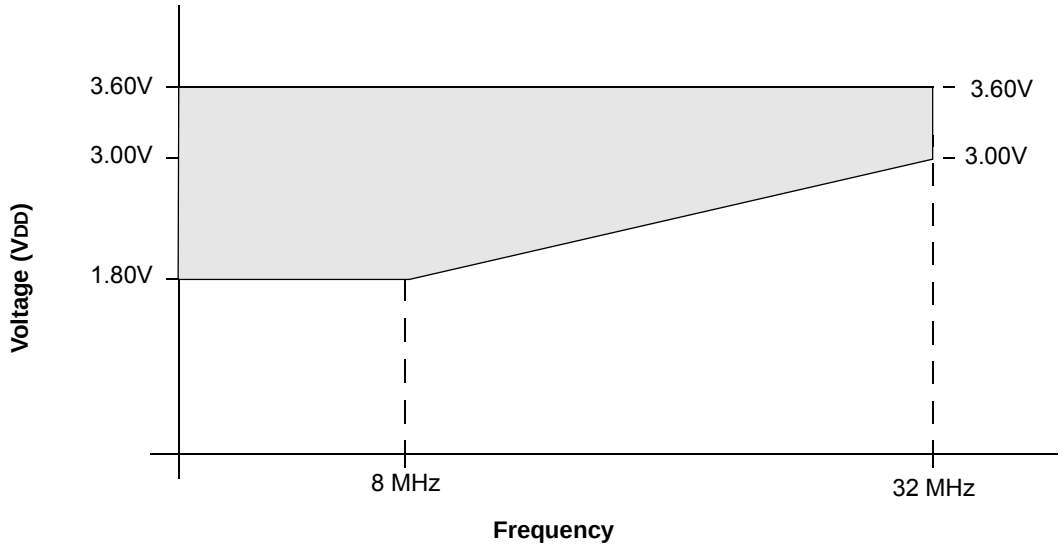
Ambient temperature under bias ..... -40°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on $V_{DD}$ with respect to $V_{SS}$ ..... -0.3V to +5.0V
Voltage on any combined analog and digital pin with respect to $V_{SS}$ ..... -0.3V to ($V_{DD}$ + 0.3V)
Voltage on any digital only pin with respect to $V_{SS}$ ..... -0.3V to ($V_{DD}$ + 0.3V)
Voltage on $\overline{MCLR}/V_{PP}$ pin with respect to $V_{SS}$ ..... -0.3V to +9.0V
Maximum current out of $V_{SS}$ pin ..... 300 mA
Maximum current into $V_{DD}$ pin (1) ..... 250 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 all ports ..... 200 mA
Maximum current sourced by all ports (1) ..... 200 mA

**Note 1:** Maximum allowable current is a function of device maximum power dissipation (see Table 26-1).

†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.

## 26.1 DC Characteristics

**FIGURE 26-1: PIC24F04KA201 FAMILY VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)**




**Note:** For frequencies between 8 MHz and 32 MHz, FMAX = 20 MHz *(VDD – 1.8) + 8 MHz.

**TABLE 26-1: THERMAL OPERATING CONDITIONS**

| Rating | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Operating Junction Temperature Range | TJ | -40 | — | +125 | °C |
| Operating Ambient Temperature Range | TA | -40 | — | +85 | °C |
| Power Dissipation:<br>Internal Chip Power Dissipation:<br>PINT = VDD x (IDD – Σ IOH)<br>I/O Pin Power Dissipation:<br>PI/O = Σ ({VDD – VOH} x IOH) + Σ (VOL x IOL) | PD | PINT + PI/O | | | W |
| Maximum Allowed Power Dissipation | PDMAX | (TJ – TA)/θJA | | | W |

**TABLE 26-2: THERMAL PACKAGING CHARACTERISTICS**

| Characteristic | Symbol | Typ | Max | Unit | Notes |
|---|---|---|---|---|---|
| Package Thermal Resistance, 14-Pin PDIP | θJA | 62.4 | — | °C/W | 1 |
| Package Thermal Resistance, 20-Pin PDIP | θJA | 60 | — | °C/W | 1 |
| Package Thermal Resistance, 14-Pin SSOP | θJA | 108 | — | °C/W | 1 |
| Package Thermal Resistance, 20-Pin SSOP | θJA | 71 | — | °C/W | 1 |
| Package Thermal Resistance, 14-Pin SOIC | θJA | 75 | — | °C/W | 1 |
| Package Thermal Resistance, 20-Pin SOIC | θJA | 80.2 | — | °C/W | 1 |
| Package Thermal Resistance, 14-Pin QFN | θJA | 43 | — | °C/W | 1 |
| Package Thermal Resistance, 20-Pin QFN | θJA | 32 | — | °C/W | 1 |

**Note 1:** Junction to ambient thermal resistance, Theta-JA (θJA) numbers are achieved by package simulations.

**TABLE 26-3: DC CHARACTERISTICS: TEMPERATURE AND VOLTAGE SPECIFICATIONS**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| DC10 | VDD | **Supply Voltage** | 1.8 | — | 3.6 | V | |
| DC12 | VDR | **RAM Data Retention Voltage(2)** | 1.5 | — | — | V | |
| DC16 | VPOR | **VDD Start Voltage** to Ensure Internal Power-on Reset Signal | VSS | — | 0.7 | V | |
| DC17 | SVDD | **VDD Rise Rate** to Ensure Internal Power-on Reset Signal | 0.05 | — | — | V/ms | 0-3.3V in 0.1s<br>0-2.5V in 60 ms |

**Note 1:** Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** This is the limit to which VDD can be lowered without losing RAM data.

**TABLE 26-4: HIGH/LOW–VOLTAGE DETECT CHARACTERISTICS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | | **Min** | **Typ** | **Max** | **Units** | **Conditions** |
| DC18 | VHLVD | HLVD Voltage on VDD Transition | HLVDL<3:0> = 0000 | — | 1.85 | 1.94 | V | |
| | | | HLVDL<3:0> = 0001 | 1.81 | 1.90 | 2.00 | V | |
| | | | HLVDL<3:0> = 0010 | 1.85 | 1.95 | 2.05 | V | |
| | | | HLVDL<3:0> = 0011 | 1.90 | 2.00 | 2.10 | V | |
| | | | HLVDL<3:0> = 0100 | 1.95 | 2.05 | 2.15 | V | |
| | | | HLVDL<3:0> = 0101 | 2.06 | 2.17 | 2.28 | V | |
| | | | HLVDL<3:0> = 0110 | 2.12 | 2.23 | 2.34 | V | |
| | | | HLVDL<3:0> = 0111 | 2.24 | 2.36 | 2.48 | V | |
| | | | HLVDL<3:0> = 1000 | 2.31 | 2.43 | 2.55 | V | |
| | | | HLVDL<3:0> = 1001 | 2.47 | 2.60 | 2.73 | V | |
| | | | HLVDL<3:0> = 1010 | 2.64 | 2.78 | 2.92 | V | |
| | | | HLVDL<3:0> = 1011 | 2.74 | 2.88 | 3.02 | V | |
| | | | HLVDL<3:0> = 1100 | 2.85 | 3.00 | 3.15 | V | |
| | | | HLVDL<3:0> = 1101 | 2.96 | 3.12 | 3.28 | V | |
| | | | HLVDL<3:0> = 1110 | 3.22 | 3.39 | 3.56 | V | |

**TABLE 26-5: BOR TRIP POINTS**

| Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | | **Min** | **Typ** | **Max** | **Units** | **Conditions** |
| DC19 | | BOR Voltage on VDD Transition | BOR = 00 | 1.55 | 2 | 2.00 | V | Valid for LPBOR and DSBOR |
| | | | BOR = 01 | 2.92 | 3 | 3.25 | V | |
| | | | BOR = 10 | 2.63 | 2.7 | 2.92 | V | |
| | | | BOR = 11 | 1.75 | 1.78 | 2.01 | V | |

**TABLE 26-6: DC CHARACTERISTICS: OPERATING CURRENT (IDD)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | |
|---|---|---|---|---|---|---|
| **Parameter No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | | |
| **IDD Current** | | | | | | |
| DC20 | 195 | 330 | µA | -40°C | 1.8V | 0.5 MIPS, FOSC = 1 MHz |
| DS20a | | 330 | | +25°C | | |
| DC20b | | 330 | | +60°C | | |
| DC20c | | 330 | | +85°C | | |
| DC20d | 365 | 590 | µA | -40°C | 3.3V | |
| DC20e | | 590 | | +25°C | | |
| DC20f | | 645 | | +60°C | | |
| DC20g | | 720 | | +85°C | | |
| DC22 | 363 | 600 | µA | -40°C | 1.8V | 1 MIPS, FOSC = 2 MHz |
| DC22a | | 600 | | +25°C | | |
| DC22b | | 600 | | +60°C | | |
| DC22c | | 600 | | +85°C | | |
| DC22d | 695 | 1100 | µA | -40°C | 3.3V | |
| DC22e | | 1100 | | +25°C | | |
| DC22f | | 1100 | | +60°C | | |
| DC22g | | 1100 | | +85°C | | |
| DC23 | 11 | 18 | mA | -40°C | 3.3V | 16 MIPS, FOSC = 32 MHz |
| DC23a | | 18 | | +25°C | | |
| DC23b | | 18 | | +60°C | | |
| DC23c | | 18 | | +85°C | | |
| DC27 | 2.25 | 3.40 | mA | -40°C | 2.5V | FRC (4 MIPS), FOSC = 8 MHz |
| DC27a | | 3.40 | | +25°C | | |
| DC27b | | 3.40 | | +60°C | | |
| DC27c | | 3.40 | | +85°C | | |
| DC27d | 3.05 | 4.60 | mA | -40°C | 3.3V | |
| DC27e | | 4.60 | | +25°C | | |
| DC27f | | 4.60 | | +60°C | | |
| DC27g | | 4.60 | | +85°C | | |

**Note 1:** Data in "Typical" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Operating Parameters:
- EC mode with clock input driven with a square wave rail-to-rail
- I/O configured as outputs driven low
- $\overline{MCLR}$ – VDD
- WDT FSCM disabled
- SRAM, program and data memory active
- All PMD bits set except for modules being measured

**TABLE 26-6: DC CHARACTERISTICS: OPERATING CURRENT (IDD) (CONTINUED)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | |
|---|---|---|---|---|---|---|
| **Parameter No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | | |
| **IDD Current** | | | | | | |
| DC31 | 8 | 28 | µA | -40°C | 1.8V | LPRC (31 kHz) |
| DC31a | | 28 | | +25°C | | |
| DC31b | | 28 | | +60°C | | |
| DC31c | | 28 | | +85°C | | |
| DC31d | 15 | 55 | µA | -40°C | 3.3V | |
| DC31e | | 55 | | +25°C | | |
| DC31f | | 55 | | +60°C | | |
| DC31g | | 55 | | +85°C | | |

**Note 1:** Data in "Typical" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Operating Parameters:
- EC mode with clock input driven with a square wave rail-to-rail
- I/O configured as outputs driven low
- $\overline{MCLR}$ – VDD
- WDT FSCM disabled
- SRAM, program and data memory active
- All PMD bits set except for modules being measured

**TABLE 26-7: DC CHARACTERISTICS: IDLE CURRENT (IIDLE)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | |
|---|---|---|---|---|---|---|
| **Param No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | | |
| **Idle Current (IIDLE): Core Off, Clock On Base Current, PMD Bits are Set(2)** | | | | | | |
| DC40 | 48 | 100 | µA | -40°C | 1.8V | 0.5 MIPS, FOSC = 1 MHz |
| DC40a | | 100 | | +25°C | | |
| DC40b | | 100 | | +60°C | | |
| DC40c | | 100 | | +85°C | | |
| DC40d | 106 | 215 | µA | -40°C | 3.3V | |
| DC40e | | 215 | | +25°C | | |
| DC40f | | 215 | | +60°C | | |
| DC40g | | 215 | | +85°C | | |

**Note 1:** Data in "Typical" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Operating Parameters:
- Core off
- EC mode with clock input driven with a square wave rail-to-rail
- I/O configured as outputs driven low
- $\overline{MCLR}$ – VDD
- WDT FSCM disabled
- SRAM, program and data memory active
- All PMD bits set except for modules being measured

**TABLE 26-7: DC CHARACTERISTICS: IDLE CURRENT (IIDLE) (CONTINUED)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | |
|---|---|---|---|---|---|---|
| **Param No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | | |
| **Idle Current (IIDLE): Core Off, Clock On Base Current, PMD Bits are Set(2)** | | | | | | |
| DC42 | 94 | 200 | µA | -40°C | 1.8V | 1 MIPS, FOSC = 2 MHz |
| DC42a | | 200 | | +25°C | | |
| DC42b | | 200 | | +60°C | | |
| DC42c | | 200 | | +85°C | | |
| DC42d | 160 | 395 | µA | -40°C | 3.3V | |
| DC42e | | 395 | | +25°C | | |
| DC42f | | 395 | | +60°C | | |
| DC42g | | 395 | | +85°C | | |
| DC43 | 3.1 | 6.0 | mA | -40°C | 3.3V | 16 MIPS, FOSC = 32 MHz |
| DC43a | | 6.0 | | +25°C | | |
| DC43b | | 6.0 | | +60°C | | |
| DC43c | | 6.0 | | +85°C | | |
| DC44 | 0.56 | 0.74 | mA | -40°C | 1.8V | FRC (4 MIPS), FOSC = 8 MHz |
| DC44a | | 0.74 | | +25°C | | |
| DC44b | | 0.74 | | +60°C | | |
| DC44c | | 0.74 | | +85°C | | |
| DC44d | 0.95 | 1.50 | mA | -40°C | 3.3V | |
| DC44e | | 1.50 | | +25°C | | |
| DC44f | | 1.50 | | +60°C | | |
| DC44g | | 1.50 | | +85°C | | |
| DC50 | 2 | 18 | µA | -40°C | 1.8V | LPRC (31 kHz) |
| DC50a | | 18 | | +25°C | | |
| DC50b | | 18 | | +60°C | | |
| DC50c | | 18 | | +85°C | | |
| DC50d | 4 | 40 | µA | -40°C | 3.3V | |
| DC50e | | 40 | | +25°C | | |
| DC50f | | 40 | | +60°C | | |
| DC50g | | 40 | | +85°C | | |

**Note 1:** Data in "Typical" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Operating Parameters:
- Core off
- EC mode with clock input driven with a square wave rail-to-rail
- I/O configured as outputs driven low
- MCLR – VDD
- WDT FSCM disabled
- SRAM, program and data memory active
- All PMD bits set except for modules being measured

**TABLE 26-8: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | |
|---|---|---|---|---|---|
| Parameter No. | Typical(1) | Max | Units | Conditions | |
| **Power-Down Current (IPD): PMD Bits are Set, PMSLP Bit is '0'(2)** | | | | | |
| DC60 | 0.025 | 0.200 | µA | -40°C | 1.8V | Base Power-Down Current (Sleep)(3) |
| DC60a | | 0.200 | | +25°C | | |
| DC60b | | 0.870 | | +60°C | | |
| DC60c | | 1.350 | | +85°C | | |
| DC60d | 0.105 | 0.540 | µA | -40°C | 3.3V | |
| DC60e | | 0.540 | | +25°C | | |
| DC60f | | 1.680 | | +60°C | | |
| DC60g | | 2.450 | | +85°C | | |
| DC70 | 0.020 | 0.150 | µA | -40°C | 1.8V | Base Deep Sleep Current |
| DC70a | | 0.150 | | +25°C | | |
| DC70b | | 0.430 | | +60°C | | |
| DC70c | | 0.630 | | +85°C | | |
| DC70d | 0.035 | 0.300 | µA | -40°C | 3.3V | |
| DC70e | | 0.300 | | +25°C | | |
| DC70f | | 0.700 | | +60°C | | |
| DC70g | | 0.980 | | +85°C | | |
| DC61 | 0.55 | 0.65 | µA | -40°C | 1.8V | Watchdog Timer Current: WDT(3,4) |
| DC61a | | 0.65 | | +25°C | | |
| DC61b | | 0.65 | | +60°C | | |
| DC61c | | 0.65 | | +85°C | | |
| DC61d | 0.87 | 0.95 | µA | -40°C | 3.3V | |
| DC61e | | 0.95 | | +25°C | | |
| DC61f | | 0.95 | | +60°C | | |
| DC61g | | 0.95 | | +85°C | | |

**Note 1:** Data in the Typical column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. WDT, etc., are all switched off.

**3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

**4:** Current applies to Sleep only.

**5:** Current applies to Deep Sleep only.

**TABLE 26-8: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD) (CONTINUED)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | |
|---|---|---|---|---|---|
| **Parameter No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | |
| **Power-Down Current (IPD): PMD Bits are Set, PMSLP Bit is '0'(2)** | | | | | |
| DC62 | 0.450 | 0.650 | µA | -40°C | 1.8V, Timer1 w/32 kHz Crystal: T132 (SOSC – LP)(3) |
| DC62a | | 0.650 | | +25°C | |
| DC62b | | 0.650 | | +60°C | |
| DC62c | | 0.650 | | +85°C | |
| DC62d | 0.730 | 0.980 | µA | -40°C | 3.3V, Timer1 w/32 kHz Crystal: T132 (SOSC – LP)(3) |
| DC62e | | 0.980 | | +25°C | |
| DC62f | | 0.980 | | +60°C | |
| DC62g | | 0.980 | | +85°C | |
| DC64 | 5.5 | 7.10 | µA | -40°C | 1.8V, HLVD(3,4) |
| DC64a | | 7.10 | | +25°C | |
| DC64b | | 7.80 | | +60°C | |
| DC64c | | 8.30 | | +85°C | |
| DC64d | 6.2 | 7.10 | µA | -40°C | 3.3V, HLVD(3,4) |
| DC64e | | 7.10 | | +25°C | |
| DC64f | | 7.80 | | +60°C | |
| DC64g | | 8.30 | | +85°C | |
| DC63 | 4.5 | 6.60 | µA | -40°C | 3.3V, BOR(3,4) |
| DC63a | | 6.60 | | +25°C | |
| DC63b | | 6.60 | | +60°C | |
| DC63c | | 6.60 | | +85°C | |

**Note 1:** Data in the Typical column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. WDT, etc., are all switched off.

**3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

**4:** Current applies to Sleep only.

**5:** Current applies to Deep Sleep only.

**TABLE 26-8: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD) (CONTINUED)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | |
|---|---|---|---|---|---|
| **Parameter No.** | **Typical(1)** | **Max** | **Units** | **Conditions** | |
| **Power-Down Current (IPD): PMD Bits are Set, PMSLP Bit is '0'(2)** | | | | | |
| DC70 | 0.045 | 0.200 | µA | -40°C | 1.8V, LPBOR(3,4) |
| DC70a | | 0.200 | | +25°C | |
| DC70b | | 0.200 | | +60°C | |
| DC70c | | 0.200 | | +85°C | |
| DC70d | 0.095 | 0.200 | µA | -40°C | 3.3V, LPBOR(3,4) |
| DC70e | | 0.200 | | +25°C | |
| DC70f | | 0.200 | | +60°C | |
| DC70g | | 0.200 | | +85°C | |
| DC71 | 0.35 | 0.55 | µA | -40°C | 1.8V, Deep Sleep Watchdog Timer: DSWDT (SOSC – LP)(5) |
| DC71a | | 0.55 | | +25°C | |
| DC71b | | 0.55 | | +60°C | |
| DC71c | | 0.55 | | +85°C | |
| DC71d | 0.55 | 0.75 | µA | -40°C | 3.3V, Deep Sleep Watchdog Timer: DSWDT (SOSC – LP)(5) |
| DC71e | | 0.75 | | +25°C | |
| DC71f | | 0.75 | | +60°C | |
| DC71g | | 0.75 | | +85°C | |
| DC72 | 0.005 | 0.200 | µA | -40°C | 1.8V, Deep Sleep BOR: DSBOR(3,5) |
| DC72a | | 0.200 | | +25°C | |
| DC72b | | 0.200 | | +60°C | |
| DC72c | | 0.200 | | +85°C | |
| DC72d | 0.010 | 0.200 | µA | -40°C | 3.3V, Deep Sleep BOR: DSBOR(3,5) |
| DC72e | | 0.200 | | +25°C | |
| DC72f | | 0.200 | | +60°C | |
| DC72g | | 0.200 | | +85°C | |

**Note 1:** Data in the Typical column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. WDT, etc., are all switched off.

**3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

**4:** Current applies to Sleep only.

**5:** Current applies to Deep Sleep only.

**TABLE 26-9: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage(4)** | — | — | — | — | |
| DI10 | | I/O Pins | VSS | — | 0.2 VDD | V | |
| DI15 | | $\overline{MCLR}$ | VSS | — | 0.2 VDD | V | |
| DI16 | | OSCI (XT mode) | VSS | — | 0.2 VDD | V | |
| DI17 | | OSCI (HS mode) | VSS | — | 0.2 VDD | V | |
| DI18 | | I/O Pins with I2C™ Buffer | VSS | — | 0.3 VDD | V | SMBus disabled |
| DI19 | | I/O Pins with SMBus Buffer | VSS | — | 0.8 | V | SMBus enabled |
| | VIH | **Input High Voltage(4)** | — | — | — | — | |
| DI20 | | I/O Pins: | | | | | |
| | | with Analog Functions | 0.8 VDD | — | VDD | V | |
| | | Digital Only | 0.8 VDD | — | VDD | V | |
| DI25 | | $\overline{MCLR}$ | 0.8 VDD | — | VDD | V | |
| DI26 | | OSCI (XT mode) | 0.7 VDD | — | VDD | V | |
| DI27 | | OSCI (HS mode) | 0.7 VDD | — | VDD | V | |
| DI28 | | I/O Pins with I2C Buffer: | | | | | |
| | | with Analog Functions | 0.7 VDD | — | VDD | V | |
| | | Digital Only | 0.7 VDD | — | VDD | V | |
| DI29 | | I/O Pins with SMBus | 2.1 | — | VDD | V | 2.5V ≤ VPIN ≤ VDD |
| DI30 | ICNPU | **CNx Pull-up Current** | 50 | 250 | 500 | µA | VDD = 3.3V, VPIN = VSS |
| | IIL | **Input Leakage Current(2,3)** | | | | | |
| DI50 | | I/O Ports | — | 0.050 | ±0.100 | µA | VSS ≤ VPIN ≤ VDD,<br>Pin at high-impedance |
| DI51 | | VREF+, VREF-, AN0, AN1 | — | 0.300 | ±0.500 | µA | VSS ≤ VPIN ≤ VDD,<br>Pin at high-impedance |
| DI55 | | $\overline{MCLR}$ | — | — | ±5.0 | µA | VSS ≤ VPIN ≤ VDD |
| DI56 | | OSCI | — | — | ±5.0 | µA | VSS ≤ VPIN ≤ VDD,<br>XT and HS modes |

**Note 1:** Data in "Typ" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**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.

**4:** Refer to Table 1-2 for I/O pin buffer types.

**TABLE 26-10: DC CHARACTERISTICS: I/O PIN OUTPUT SPECIFICATIONS**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Sym | Characteristic | Min | Typ(1) | Max | Units | Conditions |
| | VOL | **Output Low Voltage** | | | | | |
| DO10 | | All I/O Pins | — | — | 0.4 | V | IOL = 6.5 mA, VDD = 3.6V |
| | | | — | — | 0.4 | V | IOL = 3.5 mA, VDD = 2.0V |
| DO16 | | OSC2/CLKO | — | — | 0.4 | V | IOL = 8.0 mA, VDD = 3.6V |
| | | | — | — | 0.4 | V | IOL = 4.5 mA, VDD = 1.8V |
| | VOH | **Output High Voltage** | — | — | — | — | — |
| DO20 | | All I/O Pins | 3 | — | — | V | IOH = -3.0 mA, VDD = 3.6V |
| | | | 1.8 | — | — | V | IOH = -1.0 mA, VDD = 2.0V |
| DO26 | | OSC2/CLKO | 3 | — | — | V | IOH = -2.5 mA, VDD = 3.6V |
| | | | 1.8 | — | — | V | IOH = -1.0 mA, VDD = 2.0V |

**Note 1:** Data in "Typ" column is at +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**TABLE 26-11: DC CHARACTERISTICS: PROGRAM MEMORY**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Sym | Characteristic | Min | Typ(1) | Max | Units | Conditions |
| | | **Program Flash Memory** | | | | | |
| D130 | EP | Cell Endurance | 10,000(2) | — | — | E/W | |
| D131 | VPR | VDD for Read | VMIN | — | 3.6 | V | VMIN = Minimum operating voltage |
| D132 | VPEW | Supply Voltage for Self-Timed Writes | 2.2 | — | 3.6 | V | |
| D133A | TIW | Self-Timed Write Cycle Time | — | 2 | — | ms | |
| D134 | TRETD | Characteristic Retention | 40 | — | — | Year | Provided no other specifications are violated |
| D135 | IDDP | Supply Current During Programming | — | 10 | — | mA | |

**Note 1:** Data in "Typ" column is at 3.3V, +25°C unless otherwise stated.

**2:** Self-write and block erase.

**TABLE 26-12: COMPARATOR DC SPECIFICATIONS**

| **Operating Conditions:** 2.0V < VDD < 3.6V, -40°C < TA < +85°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| D300 | VIOFF | Input Offset Voltage* | — | 20 | 40 | mV | |
| D301 | VICM | Input Common-Mode Voltage* | 0 | — | VDD | V | |
| D302 | CMRR | Common-Mode Rejection Ratio* | 55 | — | — | dB | |

* Parameters are characterized but not tested.

**TABLE 26-13: COMPARATOR VOLTAGE REFERENCE DC SPECIFICATIONS**

| **Operating Conditions:** 2.0V < VDD < 3.6V, -40°C < TA < +85°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| VRD310 | CVRES | Resolution | VDD/24 | — | VDD/32 | LSb | |
| VRD311 | CVRAA | Absolute Accuracy | — | — | AVDD – 1.5 | LSb | |
| VRD312 | CVRUR | Unit Resistor Value (R) | — | 2k | — | Ω | |

**TABLE 26-14: CTMU CURRENT SOURCE SPECIFICATIONS**

| **DC CHARACTERISTICS** | | | **Standard Operating Conditions: 2.0V to 3.6V (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| | IOUT1 | CTMU Current Source, Base Range | — | 550 | — | nA | CTMUICON<1:0> = 01 |
| | IOUT2 | CTMU Current Source, 10x Range | — | 5.5 | — | μA | CTMUICON<1:0> = 10 |
| | IOUT3 | CTMU Current Source, 100x Range | — | 55 | — | μA | CTMUICON<1:0> = 11 |

**Note 1:** Nominal value at center point of current trim range (CTMUICON<7:2> = 000000).

**TABLE 26-15: COMPARATOR TIMINGS**

| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
|---|---|---|---|---|---|---|---|
| 300 | TRESP | Response Time*(1) | — | 150 | 400 | ns | |
| 301 | TMC2OV | Comparator Mode Change to Output Valid* | — | — | 10 | μs | |

* Parameters are characterized but not tested.

**Note 1:** Response time is measured with one comparator input at (VDD – 1.5)/2, while the other input transitions from VSS to VDD.

**TABLE 26-16: COMPARATOR VOLTAGE REFERENCE SETTLING TIME SPECIFICATIONS**

| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
|---|---|---|---|---|---|---|---|
| VR310 | TSET | Settling Time(1) | — | — | 10 | μs | |

**Note 1:** Settling time measured while CVRR = 1 and CVR<3:0> bits transition from '0000' to '1111'.

## 26.2 AC Characteristics and Timing Parameters

The information contained in this section defines the PIC24F04KA201 family AC characteristics and timing parameters.

**TABLE 26-17: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC**

| AC CHARACTERISTICS | **Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial<br>Operating voltage VDD range as described in **Section 26.1 "DC Characteristics"**. |
|---|---|

**FIGURE 26-2: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**

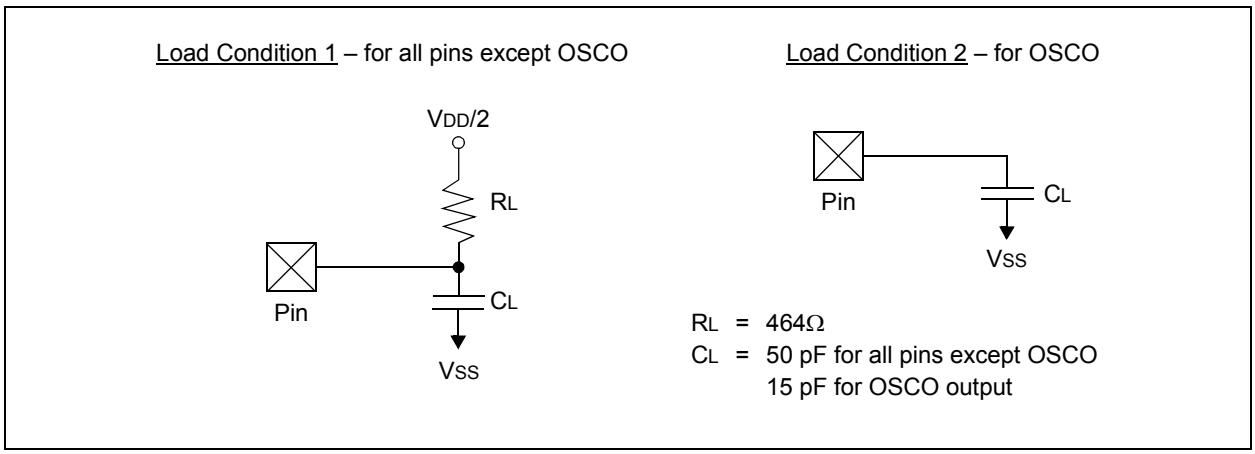



**TABLE 26-18: CAPACITIVE LOADING REQUIREMENTS ON OUTPUT PINS**

| Param No. | Symbol | Characteristic | Min | Typ(1) | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| DO50 | COSC2 | OSCO/CLKO Pin | — | — | 15 | pF | In XT and HS modes when external clock is used to drive OSCI |
| DO56 | CIO | All I/O Pins and OSCO | — | — | 50 | pF | EC mode |
| DO58 | CB | SCLx, SDAx | — | — | 400 | pF | In $I^2C$™ mode |

**Note 1:** Data in "Typ" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**FIGURE 26-3: EXTERNAL CLOCK TIMING**

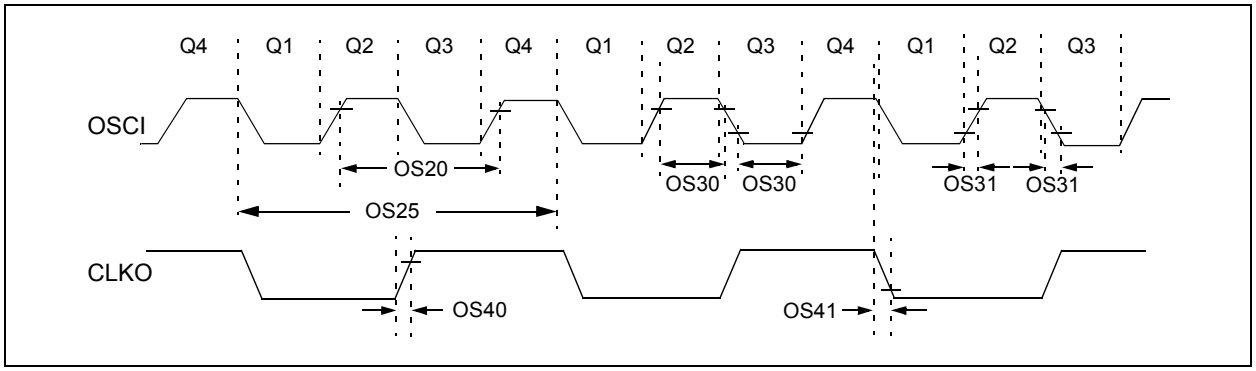



**TABLE 26-19: EXTERNAL CLOCK TIMING REQUIREMENTS**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8 to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| OS10 | FOSC | External CLKI Frequency (External clocks allowed only in EC mode) | DC<br>4 | —<br>— | 32<br>8 | MHz<br>MHz | EC<br>ECPLL |
| | | Oscillator Frequency | 0.2<br>4<br>4<br>31 | —<br>—<br>—<br>— | 4<br>25<br>8<br>33 | MHz<br>MHz<br>MHz<br>kHz | XT<br>HS<br>HSPLL<br>SOSC |
| OS20 | TOSC | TOSC = 1/FOSC | — | — | — | — | See Parameter OS10 for FOSC value |
| OS25 | TCY | Instruction Cycle Time(2) | 62.5 | — | DC | ns | |
| OS30 | TosL, TosH | External Clock in (OSCI) High or Low Time | 0.45 x TOSC | — | — | ns | EC |
| OS31 | TosR, TosF | External Clock in (OSCI) Rise or Fall Time | — | — | 20 | ns | EC |
| OS40 | TckR | CLKO Rise Time(3) | — | 6 | 10 | ns | |
| OS41 | TckF | CLKO Fall Time(3) | — | 6 | 10 | ns | |

**Note 1:** Data in “Typ” column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Instruction cycle period (TCY) equals two 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 OSCI/CLKI pin. When an external clock input is used, the “Max.” cycle time limit is “DC” (no clock) for all devices.

**3:** Measurements are taken in EC mode. The CLKO signal is measured on the OSCO pin. CLKO is low for the Q1-Q2 period (1/2 TCY) and high for the Q3-Q4 period (1/2 TCY).

**TABLE 26-20: PLL CLOCK TIMING SPECIFICATIONS ($V_{DD}$ = 1.8V TO 3.6V)**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic(1)** | **Min** | **Typ(2)** | **Max** | **Units** | **Conditions** |
| OS50 | FPLLI | PLL Input Frequency Range | 4 | — | 8 | MHz | ECPLL, HSPLL modes |
| OS51 | FSYS | PLL Output Frequency Range | 16 | — | 32 | MHz | |
| OS52 | TLOCK | PLL Start-up Time (Lock Time) | — | — | 2 | ms | |
| OS53 | DCLK | CLKO Stability (Jitter) | -2 | 1 | 2 | % | Measured over 100 ms period |

**Note 1:** These parameters are characterized but not tested in manufacturing.

**2:** Data in "Typ" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**TABLE 26-21: AC CHARACTERISTICS: INTERNAL RC ACCURACY**

| AC CHARACTERISTICS | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Conditions** | |
| F20 | **Internal FRC Accuracy @ 8 MHz(1)** | | | | | | |
| | **FRC** | -2 | — | 2 | % | +25°C | 3.0V ≤ VDD ≤ 3.6V |
| | | -5 | — | 5 | % | -40°C ≤ TA ≤ +85°C | |

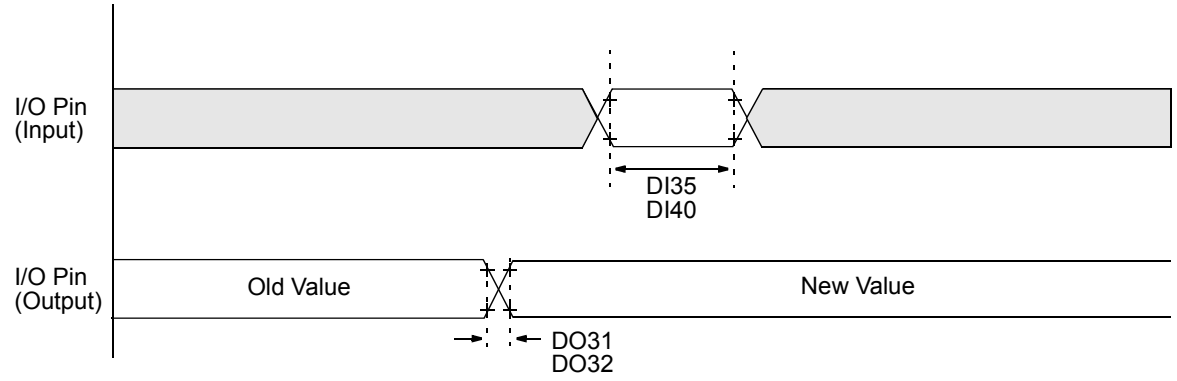
**Note 1:** Frequency calibrated at +25°C and 3.3V. OSCTUN bits can be used to compensate for temperature drift.

**TABLE 26-22: AC CHARACTERISTICS: INTERNAL RC ACCURACY**

| AC CHARACTERISTICS | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Conditions** | |
| F21 | **LPRC @ 31 kHz(1)** | | | | | | |
| | | -15 | — | 15 | % | +25°C | 3.0V ≤ VDD ≤ 3.6V |
| | | -15 | — | 15 | % | -40°C ≤ TA ≤ +85°C | |

**Note 1:** Change of LPRC frequency as VDD changes.

**FIGURE 26-4: CLKO AND I/O TIMING CHARACTERISTICS**




**Note:** Refer to Figure 26-2 for load conditions.

**TABLE 26-23: CLKO AND I/O TIMING REQUIREMENTS**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| DO31 | TIOR | Port Output Rise Time | — | 10 | 25 | ns | |
| DO32 | TIOF | Port Output Fall Time | — | 10 | 25 | ns | |
| DI35 | TINP | INTx pin High or Low Time (output) | 20 | — | — | ns | |
| DI40 | TRBP | CNx High or Low Time (input) | 2 | — | — | TCY | |

**Note 1:** Data in "Typ" column is at 3.3V, +25°C unless otherwise stated.

**TABLE 26-24: ADC MODULE SPECIFICATIONS**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min.** | **Typ** | **Max.** | **Units** | **Conditions** |
| **Device Supply** | | | | | | | |
| AD01 | AVDD | Module VDD Supply | Greater of: VDD – 0.3 or 1.8 | — | Lesser of: VDD + 0.3 or 3.6 | V | |
| AD02 | AVSS | Module VSS Supply | VSS – 0.3 | — | VSS + 0.3 | V | |
| **Reference Inputs** | | | | | | | |
| AD05 | VREFH | Reference Voltage High | AVSS + 1.7 | — | AVDD | V | |
| AD06 | VREFL | Reference Voltage Low | AVSS | — | AVDD – 1.7 | V | |
| AD07 | VREF | Absolute Reference Voltage | AVSS – 0.3 | — | AVDD + 0.3 | V | |
| **Analog Input** | | | | | | | |
| AD10 | VINH-VINL | Full-Scale Input Span | VREFL | — | VREFH | V | **(Note 2)** |
| AD11 | VIN | Absolute Input Voltage | AVSS – 0.3 | — | AVDD + 0.3 | V | |
| AD12 | VINL | Absolute VINL Input Voltage | AVSS – 0.3 | — | AVDD/2 | V | |
| AD17 | RIN | Recommended Impedance of Analog Voltage Source | — | — | 2.5K | Ω | 10-bit |
| **ADC Accuracy** | | | | | | | |
| AD20b | NR | Resolution | — | 10 | — | bits | |
| AD21b | INL | Integral Nonlinearity | — | ±1 | ±2 | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD22b | DNL | Differential Nonlinearity | — | ±1 | ±1.5 | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD23b | GERR | Gain Error | — | ±1 | ±3 | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD24b | EOFF | Offset Error | — | ±1 | ±2 | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD25b | | Monotonicity(1) | — | — | — | — | Guaranteed |

**Note 1:** The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

**2:** Measurements taken with external VREF+ and VREF- used as the ADC voltage reference.

**TABLE 26-25: ADC CONVERSION TIMING REQUIREMENTS(1)**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min.** | **Typ** | **Max.** | **Units** | **Conditions** |
| **Clock Parameters** | | | | | | | |
| AD50 | TAD | ADC Clock Period | 75 | — | — | ns | TCY = 75 ns, AD1CON3 in default state |
| AD51 | TRC | ADC Internal RC Oscillator Period | — | 250 | — | ns | |
| **Conversion Rate** | | | | | | | |
| AD55 | TCONV | Conversion Time | — | 12 | — | TAD | |
| AD56 | FCNV | Throughput Rate | — | — | 500 | ksps | AVDD ≥ 2.7V |
| AD57 | TSAMP | Sample Time | — | 1 | — | TAD | |
| AD58 | TACQ | Acquisition Time | 750 | — | — | ns | **(Note 2)** |
| AD59 | TSWC | Switching Time from Convert to Sample | — | — | **(Note 3)** | | |
| AD60 | TDIS | Discharge Time | 0.5 | — | — | TAD | |
| **Clock Parameters** | | | | | | | |
| AD61 | TPSS | Sample Start Delay from Setting Sample bit (SAMP) | 2 | — | 3 | TAD | |

**Note 1:** Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

**2:** The time for the holding capacitor to acquire the “New” input voltage when the voltage changes full scale after the conversion (VDD to VSS or VSS to VDD).

**3:** On the following cycle of the device clock.

**TABLE 26-26: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET TIMING REQUIREMENTS**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min.** | **Typ(1)** | **Max.** | **Units** | **Conditions** |
| SY10 | TmcL | $\overline{\text{MCLR}}$ Pulse Width (low) | 2 | — | — | µs | |
| SY11 | TPWRT | Power-up Timer Period | 50 | 64 | 90 | ms | |
| SY12 | TPOR | Power-on Reset Delay | 1 | 5 | 10 | µs | |
| SY13 | TIOZ | I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset | — | — | 100 | ns | |
| SY20 | TWDT | Watchdog Timer Time-out Period | 0.85 | 1.0 | 1.15 | ms | 1.32 prescaler |
| | | | 3.4 | 4.0 | 4.6 | ms | 1:128 prescaler |
| SY25 | TBOR | Brown-out Reset Pulse Width | 1 | — | — | µs | |
| SY35 | TFSCM | Fail-Safe Clock Monitor Delay | — | 2 | 2.3 | µs | |
| SY45 | TRST | Configuration Update Time | — | 20 | — | µs | |
| | TVREG | On-Chip Voltage Regulator Output Delay | — | 10 | — | µs | |
| SY55 | TLOCK | PLL Start-up Time | — | 1 | — | ms | |
| SY65 | TOST | Oscillator Start-up Time | — | 1024 | — | TOSC | |
| SY75 | TFRC | Fast RC Oscillator Start-up Time | — | 1 | 1.5 | µs | |
| SY85 | TLPRC | Low-Power Oscillator Start-up Time | — | — | 100 | µs | |

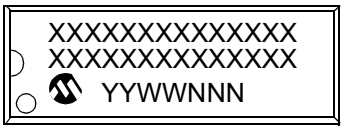
**Note 1:** Data in “Typ” column is at 3.3V, +25°C unless otherwise stated.

**NOTES:**

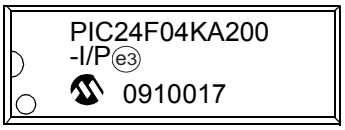
## 27.0 PACKAGING INFORMATION

### 27.1 Package Marking Information

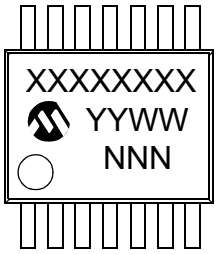
14-Lead PDIP



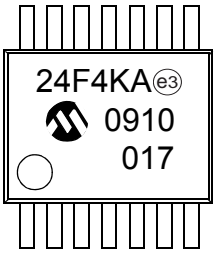

Example



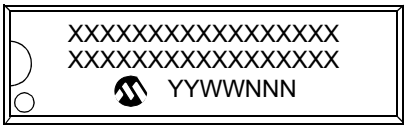

14-Lead TSSOP



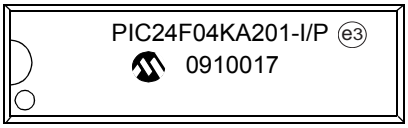

Example




20-Lead PDIP



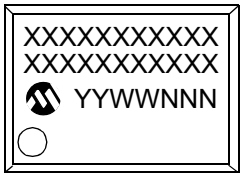

Example



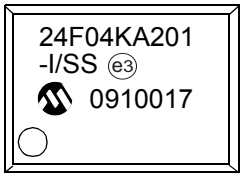

| | | |
|---|---|---|
| **Legend:** | XX...X | Product-specific information |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC designator for Matte Tin (Sn) |
| | * | This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package. |

**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.

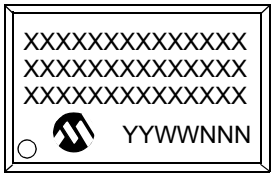
20-Lead SSOP



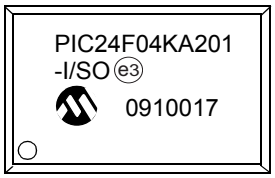

Example



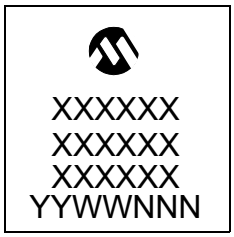

20-Lead SOIC (.300”)




Example




20-Lead QFN



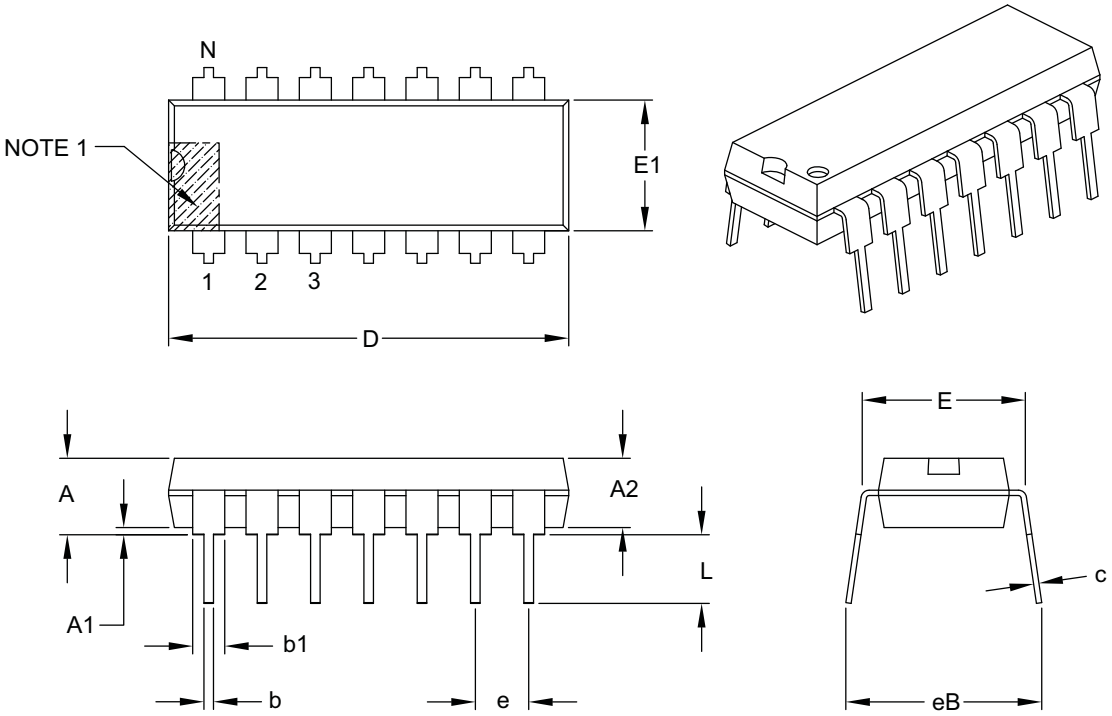

Example

## 27.2 Package Details

The following sections give the technical details of the packages.

### 14-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

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




| Units | | INCHES | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 14 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .210 |
| Molded Package Thickness | A2 | .115 | .130 | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .290 | .310 | .325 |
| Molded Package Width | E1 | .240 | .250 | .280 |
| Overall Length | D | .735 | .750 | .775 |
| Tip to Seating Plane | L | .115 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .045 | .060 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

**Notes:**

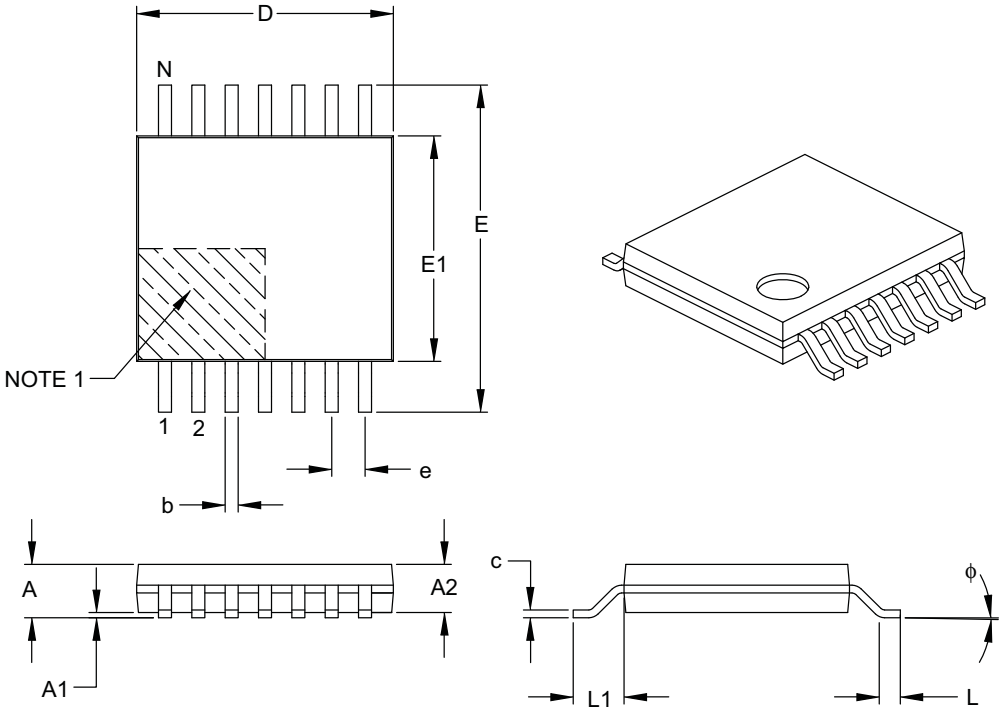
1. Pin 1 visual index feature may vary, but must be located with the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-005B

### 14-Lead Plastic Thin Shrink Small Outline (ST) – 4.4 mm Body [TSSOP]

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




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 14 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | – | – | 1.20 |
| Molded Package Thickness | A2 | 0.80 | 1.00 | 1.05 |
| Standoff | A1 | 0.05 | – | 0.15 |
| Overall Width | E | | 6.40 BSC | |
| Molded Package Width | E1 | 4.30 | 4.40 | 4.50 |
| Molded Package Length | D | 4.90 | 5.00 | 5.10 |
| Foot Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Foot Angle | ϕ | 0° | – | 8° |
| Lead Thickness | c | 0.09 | – | 0.20 |
| Lead Width | b | 0.19 | – | 0.30 |

**Notes:**

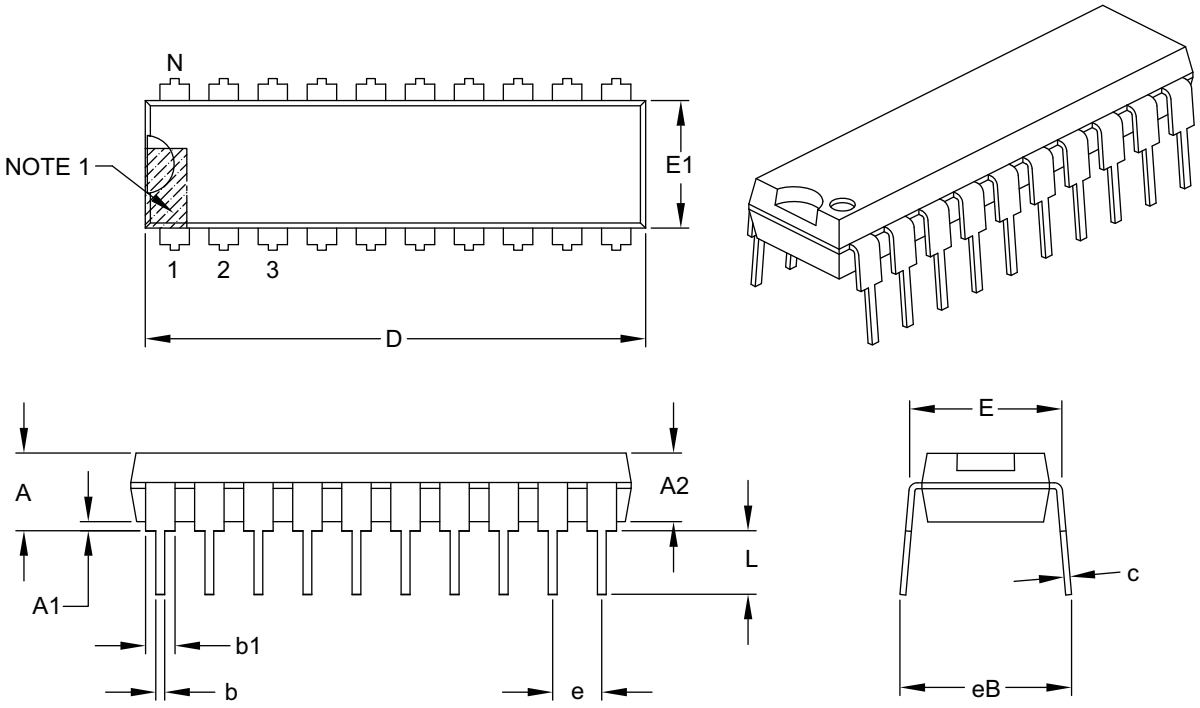
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-087B

## 20-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

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




| Units | | INCHES | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 20 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .210 |
| Molded Package Thickness | A2 | .115 | .130 | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .300 | .310 | .325 |
| Molded Package Width | E1 | .240 | .250 | .280 |
| Overall Length | D | .980 | 1.030 | 1.060 |
| Tip to Seating Plane | L | .115 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .045 | .060 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

**Notes:**

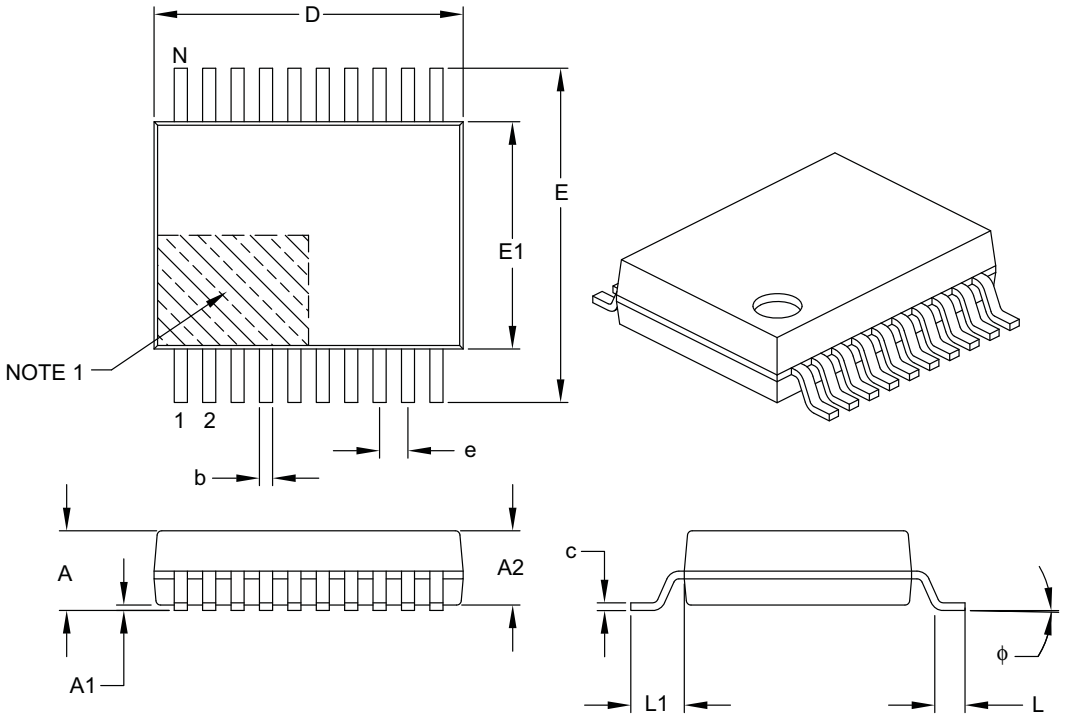
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-019B

## 20-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

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




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 20 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | – | – | 2.00 |
| Molded Package Thickness | A2 | 1.65 | 1.75 | 1.85 |
| Standoff | A1 | 0.05 | – | – |
| Overall Width | E | 7.40 | 7.80 | 8.20 |
| Molded Package Width | E1 | 5.00 | 5.30 | 5.60 |
| Overall Length | D | 6.90 | 7.20 | 7.50 |
| Foot Length | L | 0.55 | 0.75 | 0.95 |
| Footprint | L1 | | 1.25 REF | |
| Lead Thickness | c | 0.09 | – | 0.25 |
| Foot Angle | ϕ | 0° | 4° | 8° |
| Lead Width | b | 0.22 | – | 0.38 |

**Notes:**

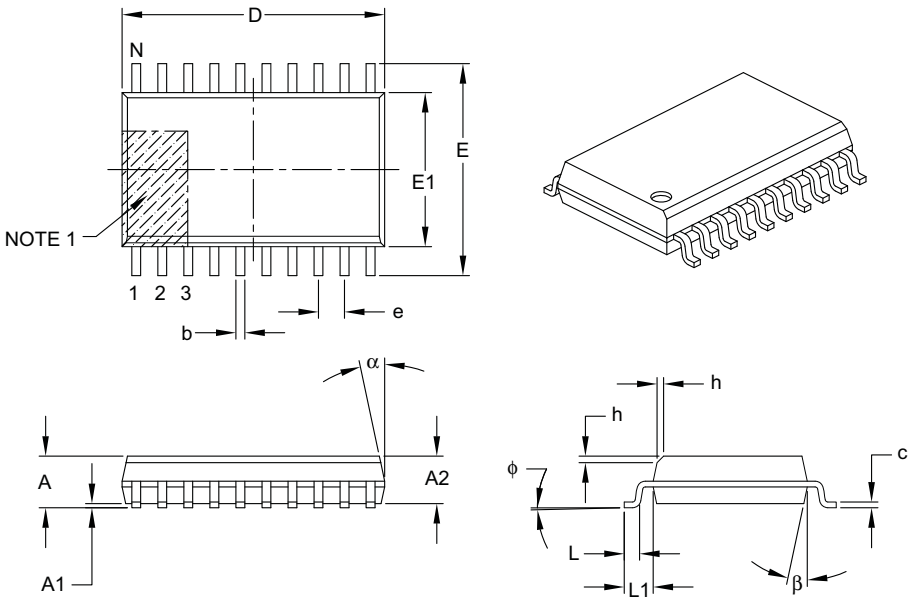
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-072B

## 20-Lead Plastic Small Outline (SO) – Wide, 7.50 mm Body [SOIC]

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




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 20 | |
| Pitch | e | | 1.27 BSC | |
| Overall Height | A | – | – | 2.65 |
| Molded Package Thickness | A2 | 2.05 | – | – |
| Standoff § | A1 | 0.10 | – | 0.30 |
| Overall Width | E | | 10.30 BSC | |
| Molded Package Width | E1 | | 7.50 BSC | |
| Overall Length | D | | 12.80 BSC | |
| Chamfer (optional) | h | 0.25 | – | 0.75 |
| Foot Length | L | 0.40 | – | 1.27 |
| Footprint | L1 | | 1.40 REF | |
| Foot Angle | ϕ | 0° | – | 8° |
| Lead Thickness | c | 0.20 | – | 0.33 |
| Lead Width | b | 0.31 | – | 0.51 |
| Mold Draft Angle Top | α | 5° | – | 15° |
| Mold Draft Angle Bottom | β | 5° | – | 15° |

**Notes:**

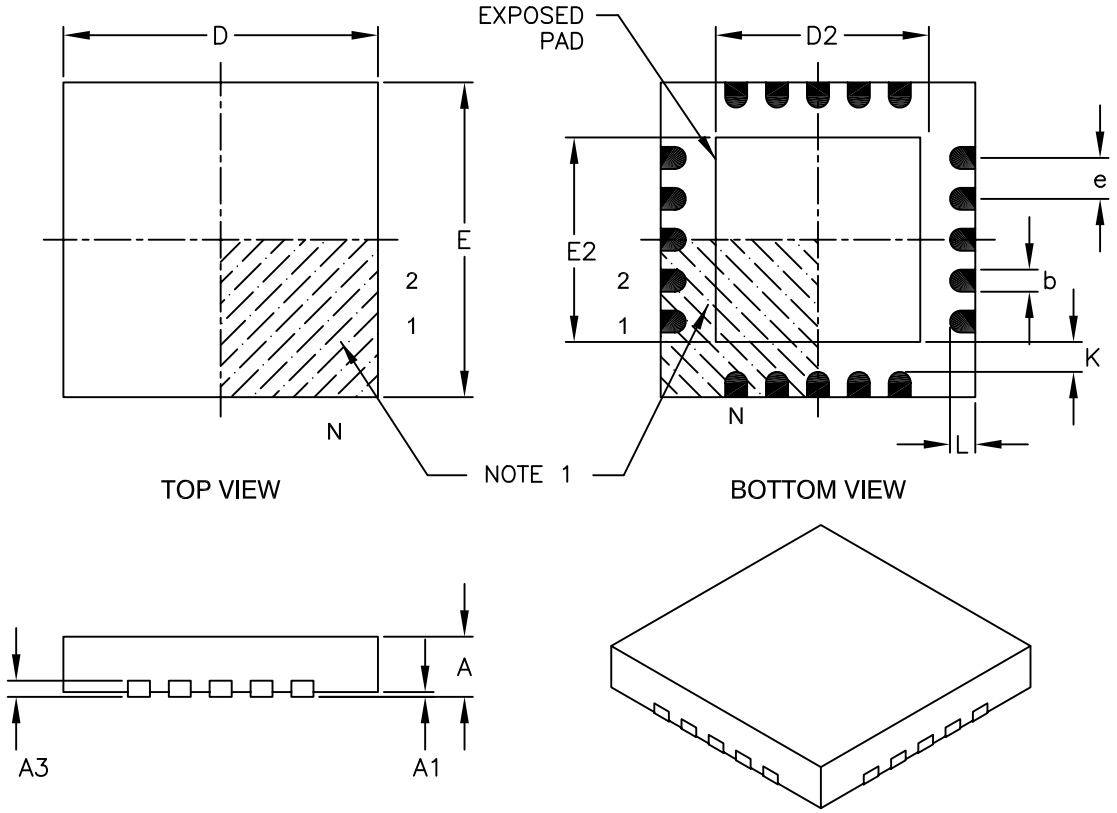
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-094B

## 20-Lead Plastic Quad Flat, No Lead Package (MQ) – 5x5x0.9 mm Body [QFN]

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




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 20 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | 0.00 | 0.02 | 0.05 |
| Contact Thickness | A3 | | 0.20 REF | |
| Overall Width | E | | 5.00 BSC | |
| Exposed Pad Width | E2 | 3.15 | 3.25 | 3.35 |
| Overall Length | D | | 5.00 BSC | |
| Exposed Pad Length | D2 | 3.15 | 3.25 | 3.35 |
| Contact Width | b | 0.25 | 0.30 | 0.35 |
| Contact Length | L | 0.35 | 0.40 | 0.45 |
| Contact-to-Exposed Pad | K | 0.20 | - | - |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-139B

# APPENDIX A: REVISION HISTORY

## Revision A (February 2009)

Original data sheet for the PIC24F04KA201 family of devices.

## Revision B (May 2009)

The title was changed. **Section 2.0 "Guidelines for Getting Started with 16-Bit Microcontrollers"** was added. Extensive changes to **Section 26.0 "Electrical Characteristics"**. Minor text edits throughout document.

## Revision C (March 2014)

Removed the 'Preliminary' status from the data sheet.

**NOTES:**

# INDEX

## R



## S



## T

## U



## W

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

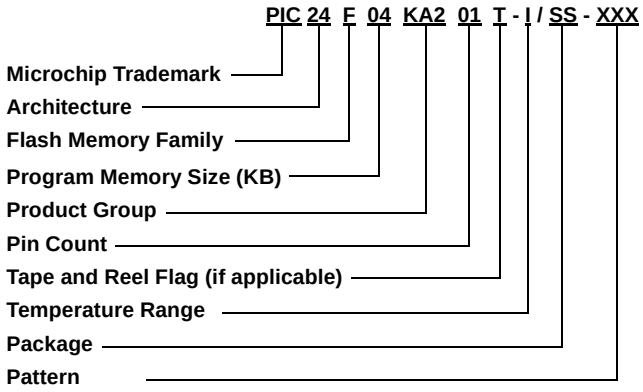
Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the web site at: http://microchip.com/support**

**NOTES:**

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.




| | | |
|---|---|---|
| Architecture | 24 | = 16-bit modified Harvard without DSP |
| Flash Memory Family | F | = Flash program memory |
| Product Group | KA2 | = General purpose microcontrollers |
| Pin Count | 00<br>01 | = 14-pin<br>= 20-pin |
| Temperature Range | I | = -40°C to +85°C (Industrial) |
| Package | P<br>SL<br>SO<br>SS<br>MQ<br>ST | = PDIP<br>= SOIC, Narrow<br>= SOIC<br>= SSOP<br>= QFN<br>= TSSOP |
| Pattern | Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)<br>ES | = Engineering Sample |

**Examples:**

a) PIC24F04KA201-I/MQ: General purpose, 16-Kbyte program memory, 20-pin, Industrial temp., QFN package.

**NOTES:**

**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. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.



**Trademarks**



Printed on recycled paper.

ISBN: 978-1-63276-041-8

**QUALITY MANAGEMENT SYSTEM**
**CERTIFIED BY DNV**
**═ ISO/TS 16949 ═**

*Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*

# 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:
http://www.microchip.com/support
Web Address:
www.microchip.com

**Atlanta**
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

**Austin, TX**
Tel: 512-257-3370

**Boston**
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

**Chicago**
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

**Cleveland**
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

**Dallas**
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

**Detroit**
Novi, MI
Tel: 248-848-4000

**Houston, TX**
Tel: 281-894-5983

**Indianapolis**
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

**Los Angeles**
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

**New York, NY**
Tel: 631-435-6000

**San Jose, CA**
Tel: 408-735-9110

**Canada - Toronto**
Tel: 905-673-0699
Fax: 905-673-6509

## ASIA/PACIFIC

**Asia Pacific Office**
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

**Australia - Sydney**
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

**China - Beijing**
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

**China - Chengdu**
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

**China - Chongqing**
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

**China - Hangzhou**
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

**China - Hong Kong SAR**
Tel: 852-2943-5100
Fax: 852-2401-3431

**China - Nanjing**
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

**China - Qingdao**
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

**China - Shanghai**
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

**China - Shenyang**
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

**China - Shenzhen**
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

**China - Wuhan**
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

**China - Xian**
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

**China - Xiamen**
Tel: 86-592-2388138
Fax: 86-592-2388130

**China - Zhuhai**
Tel: 86-756-3210040
Fax: 86-756-3210049

## ASIA/PACIFIC

**India - Bangalore**
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

**India - New Delhi**
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

**India - Pune**
Tel: 91-20-3019-1500

**Japan - Osaka**
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

**Japan - Tokyo**
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

**Korea - Daegu**
Tel: 82-53-744-4301
Fax: 82-53-744-4302

**Korea - Seoul**
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

**Malaysia - Kuala Lumpur**
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

**Malaysia - Penang**
Tel: 60-4-227-8870
Fax: 60-4-227-4068

**Philippines - Manila**
Tel: 63-2-634-9065
Fax: 63-2-634-9069

**Singapore**
Tel: 65-6334-8870
Fax: 65-6334-8850

**Taiwan - Hsin Chu**
Tel: 886-3-5778-366
Fax: 886-3-5770-955

**Taiwan - Kaohsiung**
Tel: 886-7-213-7830

**Taiwan - Taipei**
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

**Thailand - Bangkok**
Tel: 66-2-694-1351
Fax: 66-2-694-1350

## EUROPE

**Austria - Wels**
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

**Denmark - Copenhagen**
Tel: 45-4450-2828
Fax: 45-4485-2829

**France - Paris**
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

**Germany - Dusseldorf**
Tel: 49-2129-3766400

**Germany - Munich**
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

**Germany - Pforzheim**
Tel: 49-7231-424750

**Italy - Milan**
Tel: 39-0331-742611
Fax: 39-0331-466781

**Italy - Venice**
Tel: 39-049-7625286

**Netherlands - Drunen**
Tel: 31-416-690399
Fax: 31-416-690340

**Poland - Warsaw**
Tel: 48-22-3325737

**Spain - Madrid**
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

**Sweden - Stockholm**
Tel: 46-8-5090-4654

**UK - Wokingham**
Tel: 44-118-921-5800
Fax: 44-118-921-5820

03/25/14