**EN:** This Datasheet/Document is presented by the manufacturer.

Please visit our website for pricing and availability at www.hestore.hu.

# PIC18F2220/2320/4220/4320
# Data Sheet

## 28/40/44-Pin High-Performance, Enhanced Flash Microcontrollers with 10-Bit A/D and nanoWatt Technology

© 2003 Microchip Technology Inc.

**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.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

**Trademarks**

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Application Maestro, dsPICDEM, dsPICDEM.net, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerMate, PowerTool, rfLAB, rfPIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2003, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

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

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


© 2003 Microchip Technology Inc.

# MICROCHIP PIC18F2220/2320/4220/4320

## 28/40/44-Pin High-Performance, Enhanced Flash MCUs with 10-bit A/D and nanoWatt Technology

### Low-Power Features:

- Power Managed modes:
  - Run: CPU on, peripherals on
  - Idle: CPU off, peripherals on
  - Sleep: CPU off, peripherals off
- Power Consumption modes:
  - PRI_RUN: 150 μA, 1 MHz, 2V
  - PRI_IDLE: 37 μA, 1 MHz, 2V
  - SEC_RUN: 14 μA, 32 kHz, 2V
  - SEC_IDLE: 5.8 μA, 32 kHz, 2V
  - RC_RUN: 110 μA, 1 MHz, 2V
  - RC_IDLE: 52 μA, 1 MHz, 2V
  - Sleep: 0.1 μA, 1 MHz, 2V
- Timer1 Oscillator: 1.1 μA, 32 kHz, 2V
- Watchdog Timer: 2.1 μA
- Two-Speed Oscillator Start-up

### Oscillators:

- Four Crystal modes:
  - LP, XT, HS: up to 25 MHz
  - HSPLL: 4-10 MHz (16-40 MHz internal)
- Two External RC modes, up to 4 MHz
- Two External Clock modes, up to 40 MHz
- Internal oscillator block:
  - 8 user selectable frequencies: 31 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz
  - 125 kHz-8 MHz calibrated to 1%
  - Two modes select one or two I/O pins
  - OSCTUNE – Allows user to shift frequency
- Secondary oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor
  - Allows for safe shutdown if peripheral clock stops

### Peripheral Highlights:

- High current sink/source 25 mA/25 mA
- Three external interrupts
- Up to 2 Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution is 6.25 ns (TCY/16)
  - Compare is 16-bit, max. resolution is 100 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead-time
  - Auto-Shutdown and Auto-Restart
- Compatible 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with programmable acquisition time
- Dual analog comparators
- Addressable USART module:
  - RS-232 operation using internal oscillator block (no external crystal required)

### Special Microcontroller Features:

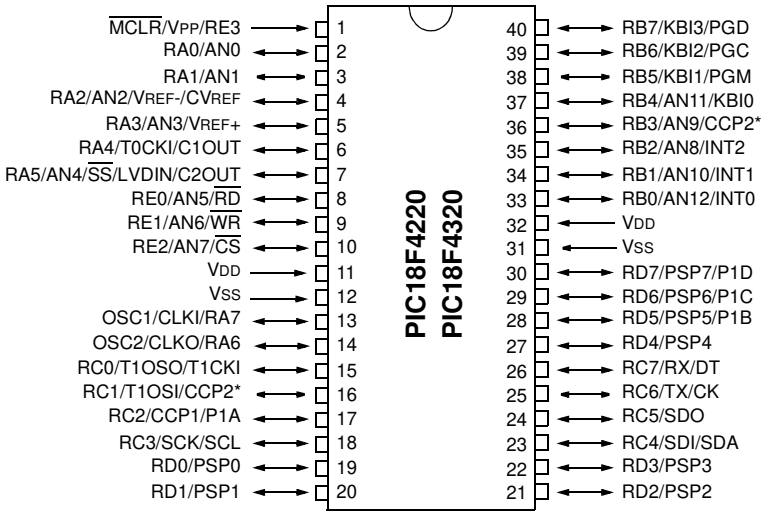
- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Flash/Data EEPROM Retention: > 40 years
- Self-programmable under software control
- Priority levels for interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 41 ms to 131s
  - 2% stability over VDD and Temperature
- Single-supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Wide operating voltage range: 2.0V to 5.5V

| Device | Program Memory | | Data Memory | | I/O | 10-bit A/D (ch) | CCP/ ECCP (PWM) | MSSP | | USART | Comparators | Timers 8/16-bit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Flash (bytes) | # Single Word Instructions | SRAM (bytes) | EEPROM (bytes) | | | | SPI™ | Master I²C™ | | | |
| PIC18F2220 | 4096 | 2048 | 512 | 256 | 25 | 10 | 2/0 | Y | Y | Y | 2 | 2/3 |
| PIC18F2320 | 8192 | 4096 | 512 | 256 | 25 | 10 | 2/0 | Y | Y | Y | 2 | 2/3 |
| PIC18F4220 | 4096 | 2048 | 512 | 256 | 36 | 13 | 1/1 | Y | Y | Y | 2 | 2/3 |
| PIC18F4320 | 8192 | 4096 | 512 | 256 | 36 | 13 | 1/1 | Y | Y | Y | 2 | 2/3 |

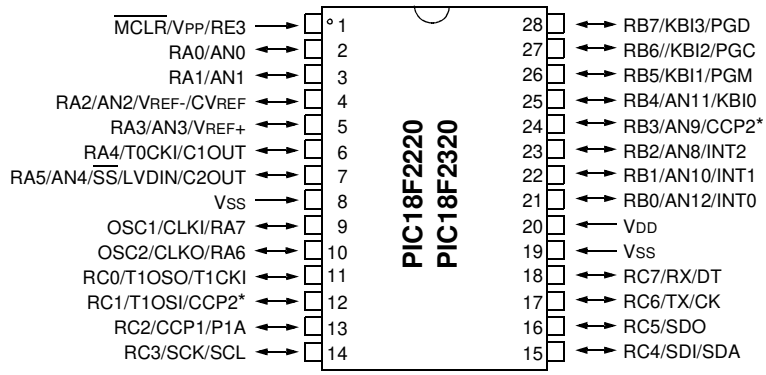
© 2003 Microchip Technology Inc.

## Pin Diagrams

### PDIP




### SPDIP, SOIC



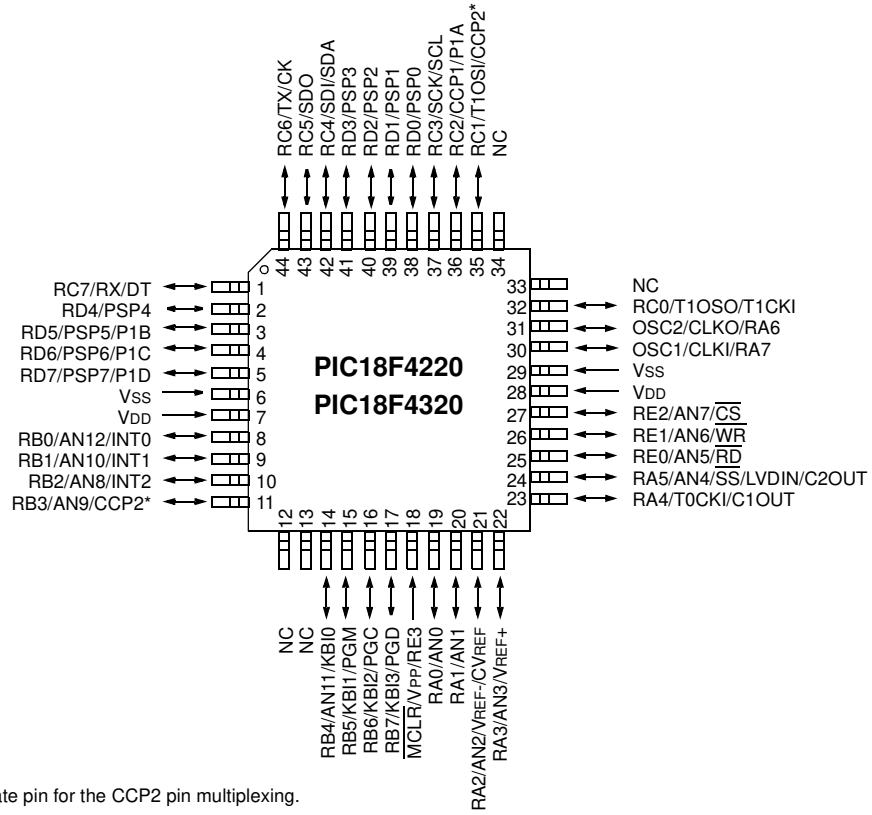

* RB3 is the alternate pin for the CCP2 pin multiplexing.

**Note:** Pin compatible with 40-pin PIC16C7X devices.


© 2003 Microchip Technology Inc.

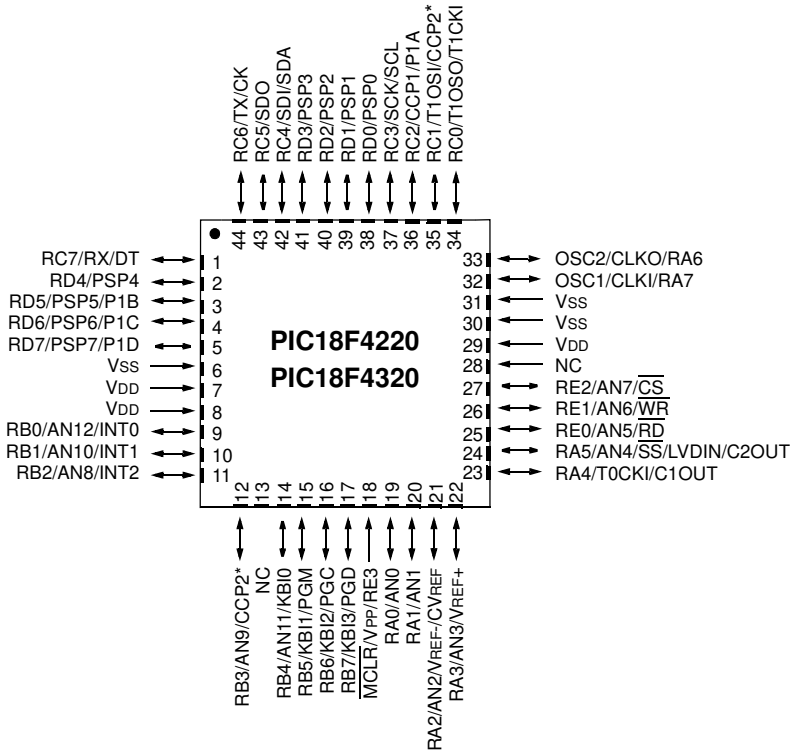
## Pin Diagrams (Cont.'d)

### TQFP




* RB3 is the alternate pin for the CCP2 pin multiplexing.

### QFN




* RB3 is the alternate pin for the CCP2 pin multiplexing.

## Table of Contents

1.0 Device Overview ..... 7
2.0 Oscillator Configurations ..... 19
3.0 Power Managed Modes ..... 29
4.0 Reset ..... 43
5.0 Memory Organization ..... 53
6.0 Flash Program Memory ..... 71
7.0 Data EEPROM Memory ..... 81
8.0 8 X 8 Hardware Multiplier ..... 85
9.0 Interrupts ..... 87
10.0 I/O Ports ..... 101
11.0 Timer0 Module ..... 117
12.0 Timer1 Module ..... 121
13.0 Timer2 Module ..... 127
14.0 Timer3 Module ..... 129
15.0 Capture/Compare/PWM (CCP) Modules ..... 133
16.0 Enhanced Capture/Compare/PWM (ECCP) Module ..... 141
17.0 Master Synchronous Serial Port (MSSP) Module ..... 155
18.0 Addressable Universal Synchronous Asynchronous Receiver Transmitter (USART) ..... 195
19.0 10-bit Analog-to-Digital Converter (A/D) Module ..... 211
20.0 Comparator Module ..... 221
21.0 Comparator Voltage Reference Module ..... 227
22.0 Low-Voltage Detect ..... 231
23.0 Special Features of the CPU ..... 237
24.0 Instruction Set Summary ..... 255
25.0 Development Support ..... 299
26.0 Electrical Characteristics ..... 305
27.0 DC and AC Characteristics Graphs and Tables ..... 343
28.0 Packaging Information ..... 361
Appendix A: Revision History ..... 369
Appendix B: Device Differences ..... 369
Appendix C: Conversion Considerations ..... 370
Appendix D: Migration from Baseline to Enhanced Devices ..... 370
Appendix E: Migration from Mid-Range to Enhanced Devices ..... 371
Appendix F: Migration from High-End to Enhanced Devices ..... 371
Index ..... 373
On-Line Support ..... 383
Systems Information and Upgrade Hot Line ..... 383
Reader Response ..... 384
PIC18F2220/2320/4220/4320 Product Identification System ..... 385


© 2003 Microchip Technology Inc.

## TO OUR VALUED CUSTOMERS

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

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

### Most Current Data Sheet

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

http://www.microchip.com

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

### Errata

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

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

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

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

### Customer Notification System

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

**NOTES:**


© 2003 Microchip Technology Inc.

# 1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC18F2220
- PIC18F2320
- PIC18F4220
- PIC18F4320

This family offers the advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price with the addition of high-endurance Enhanced Flash program memory. On top of these features, the PIC18F2220/2320/4220/4320 family introduces design enhancements that make these microcontrollers a logical choice for many high-performance, power sensitive applications.

## 1.1 New Core Features

### 1.1.1 nanoWatt TECHNOLOGY

All of the devices in the PIC18F2220/2320/4220/4320 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal oscillator block, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled, but the peripherals are still active. In these states, power consumption can be reduced even further, to as little as 4% of normal operation requirements.
- **On-the-fly Mode Switching:** The power managed modes are invoked by user code during operation, allowing the user to incorporate power saving ideas into their application's software design.
- **Lower Consumption in Key Modules:** The power requirements for both Timer1 and the Watchdog Timer have been reduced by up to 80%, with typical values of 1.8 and 2.2 μA, respectively.

### 1.1.2 MULTIPLE OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F2220/2320/4220/4320 family offer nine different oscillator options, allowing users a wide range of choices in developing application hardware. These include:

- Four Crystal modes using crystals or ceramic resonators.
- Two External Clock modes offering the option of using two pins (oscillator input and a divide-by-4 clock output) or one pin (oscillator input with the second pin reassigned as general I/O).
- Two External RC Oscillator modes with the same pin options as the External Clock modes.
- An internal oscillator block, which provides a 31 kHz INTRC clock and an 8 MHz clock with 6 program selectable divider ratios (4 MHz to 125 kHz) for a total of 8 clock frequencies.

Besides its availability as a clock source, the internal oscillator block provides a stable reference source that gives the family additional features for robust operation:

- **Fail-Safe Clock Monitor:** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator block, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up:** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available. This allows for code execution during what would otherwise be the clock start-up interval and can even allow an application to perform routine background activities and return to Sleep without returning to full power operation.

## 1.2 Other Special Features

- **Memory Endurance:** The Enhanced Flash cells for both program memory and data EEPROM are rated to last for many thousands of erase/write cycles – up to 100,000 for program memory and 1,000,000 for EEPROM. Data retention without refresh is conservatively estimated to be greater than 40 years.
- **Self-programmability:** These devices can write to their own program memory spaces under internal software control. By using a bootloader routine located in the protected Boot Block at the top of program memory, it becomes possible to create an application that can update itself in the field.
- **Enhanced CCP Module:** In PWM mode, this module provides 1, 2 or 4 modulated outputs for controlling half-bridge and full-bridge drivers. Other features include Auto-Shutdown for disabling PWM outputs on interrupt or other select conditions and Auto-Restart to reactivate outputs once the condition has cleared.
- **Addressable USART:** This serial communication module is capable of standard RS-232 operation using the internal oscillator block, removing the need for an external crystal (and its accompanying power requirement) in applications that talk to the outside world.
- **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 thus, reduce code overhead.
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing a time-out range from 4 ms to over 2 minutes, that is stable across operating voltage and temperature.

© 2003 Microchip Technology Inc.

## 1.3 Details on Individual Family Members

Devices in the PIC18F2220/2320/4220/4320 family are available in 28-pin (PIC18F2X20) and 40/44-pin (PIC18F4X20) packages. Block diagrams for the two groups are shown in Figure 1-1 and Figure 1-2.

The devices are differentiated from each other in five ways:

1. Flash program memory (4 Kbytes for PIC18FX220 devices, 8 Kbytes for PIC18FX320)
2. A/D channels (10 for PIC18F2X20 devices, 13 for PIC18F4X20 devices)
3. I/O ports (3 bidirectional ports and 1 input only port on PIC18F2X20 devices, 5 bidirectional ports on PIC18F4X20 devices)
4. CCP and Enhanced CCP implementation (PIC18F2X20 devices have 2 standard CCP modules, PIC18F4X20 devices have one standard CCP module and one ECCP module)
5. Parallel Slave Port (present only on PIC18F4X20 devices)

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

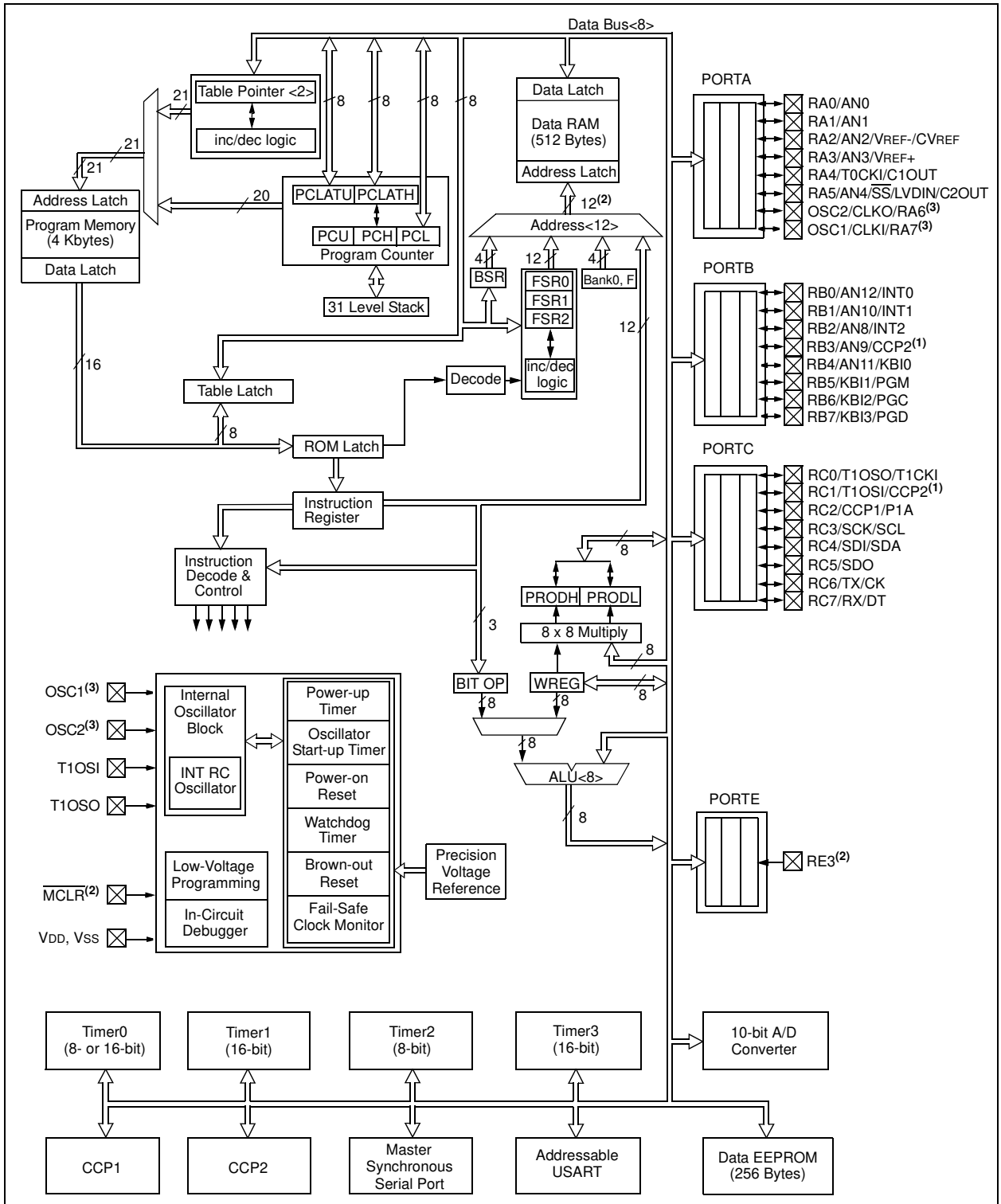
The pinouts for all devices are listed in Table 1-2 and Table 1-3.

**TABLE 1-1: DEVICE FEATURES**

| Features | PIC18F2220 | PIC18F2320 | PIC18F4220 | PIC18F4320 |
|---|---|---|---|---|
| Operating Frequency | DC – 40 MHz | DC – 40 MHz | DC – 40 MHz | DC – 40 MHz |
| Program Memory (Bytes) | 4096 | 8192 | 4096 | 8192 |
| Program Memory (Instructions) | 2048 | 4096 | 2048 | 4096 |
| Data Memory (Bytes) | 512 | 512 | 512 | 512 |
| Data EEPROM Memory (Bytes) | 256 | 256 | 256 | 256 |
| Interrupt Sources | 19 | 19 | 20 | 20 |
| I/O Ports | Ports A, B, C (E) | Ports A, B, C (E) | Ports A, B, C, D, E | Ports A, B, C, D, E |
| Timers | 4 | 4 | 4 | 4 |
| Capture/Compare/PWM Modules | 2 | 2 | 1 | 1 |
| Enhanced Capture/ Compare/PWM Modules | 0 | 0 | 1 | 1 |
| Serial Communications | MSSP, Addressable USART | MSSP, Addressable USART | MSSP, Addressable USART | MSSP, Addressable USART |
| Parallel Communications (PSP) | No | No | Yes | Yes |
| 10-bit Analog-to-Digital Module | 10 Input Channels | 10 Input Channels | 13 Input Channels | 13 Input Channels |
| Resets (and Delays) | POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT | POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT | POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT | POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT |
| Programmable Low-Voltage Detect | Yes | Yes | Yes | Yes |
| Programmable Brown-out Reset | Yes | Yes | Yes | Yes |
| Instruction Set | 75 Instructions | 75 Instructions | 75 Instructions | 75 Instructions |
| Packages | 28-pin SPDIP 28-pin SOIC | 28-pin SPDIP 28-pin SOIC | 40-pin PDIP 44-pin TQFP 44-pin QFN | 40-pin PDIP 44-pin TQFP 44-pin QFN |


© 2003 Microchip Technology Inc.

**FIGURE 1-1: PIC18F2220/2320 BLOCK DIAGRAM**

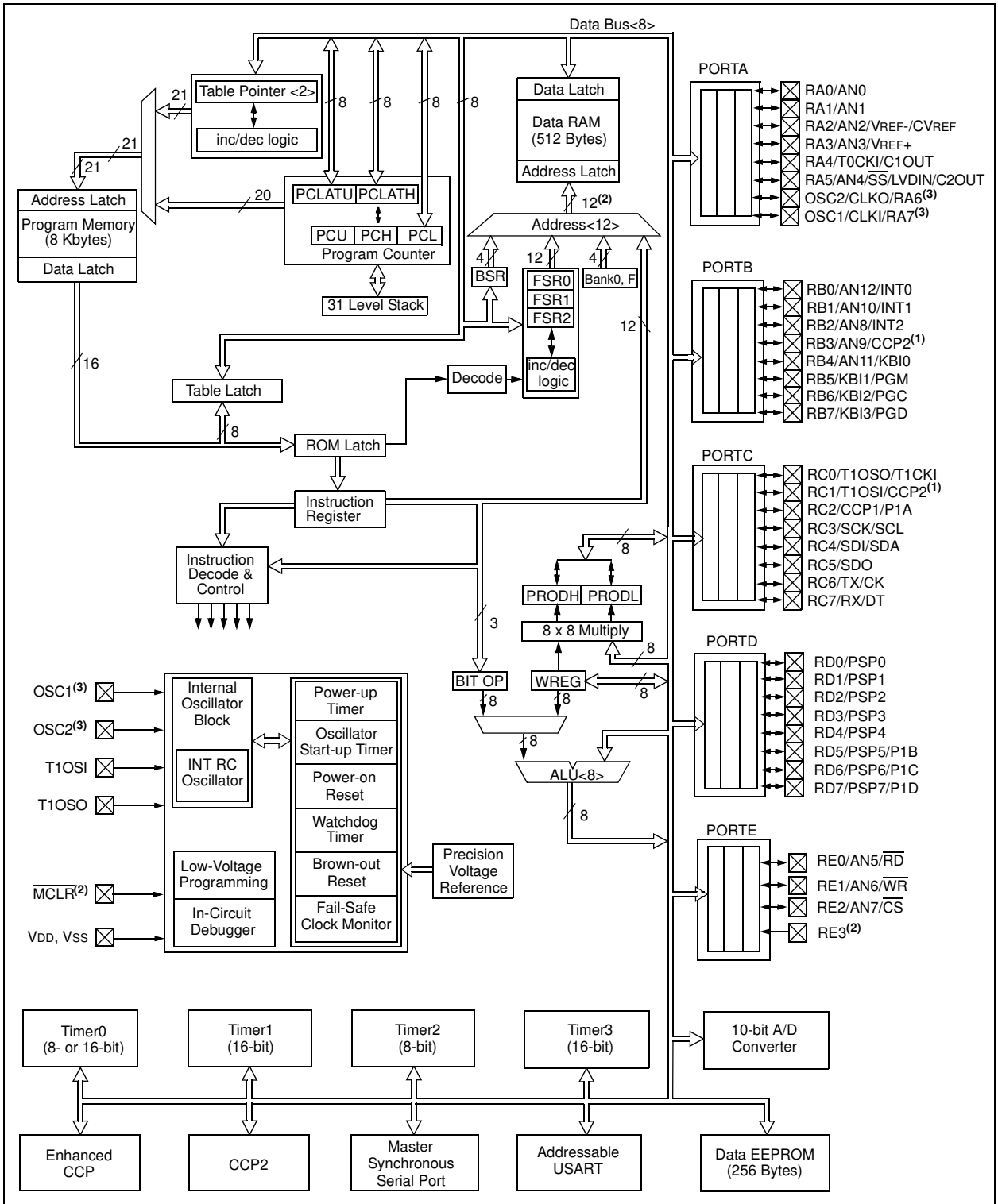



**Note 1:** Optional multiplexing of CCP2 input/output with RB3 is enabled by selection of the CCPMX2 configuration bit.

**2:** RE3 is available only when the $\overline{\text{MCLR}}$ Resets are disabled.

**3:** OSC1, OSC2, CLKI and CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

**FIGURE 1-2: PIC18F4220/4320 BLOCK DIAGRAM**




**Note 1:** Optional multiplexing of CCP2 input/output with RB3 is enabled by selection of the CCP2MX configuration bit.

**2:** RE3 is available only when the $\overline{MCLR}$ Resets are disabled.

**3:** OSC1, OSC2, CLKI and CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.


© 2003 Microchip Technology Inc.

**TABLE 1-2: PIC18F2220/2320 PINOUT I/O DESCRIPTIONS**

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|
| | PDIP | SOIC | | | |
| MCLR/VPP/RE3 | 1 | 1 | | | Master Clear (input) or programming voltage (input). |
| MCLR | | | I | ST | Master Clear (Reset) input. This pin is an active-low Reset to the device. |
| VPP | | | P | | Programming voltage input. |
| RE3 | | | I | ST | Digital input. |
| OSC1/CLKI/RA7 | 9 | 9 | | | Oscillator crystal or external clock input. |
| OSC1 | | | I | ST | Oscillator crystal input or external clock source input. ST buffer when configured in RC mode, CMOS otherwise. |
| CLKI | | | I | CMOS | External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.) |
| RA7 | | | I/O | TTL | General purpose I/O pin. |
| OSC2/CLKO/RA6 | 10 | 10 | | | Oscillator crystal or clock output. |
| OSC2 | | | O | — | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. |
| CLKO | | | O | — | In RC mode, OSC2 pin outputs CLKO which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. |
| RA6 | | | I/O | TTL | General purpose I/O pin. |
| | | | | | PORTA is a bidirectional I/O port. |
| RA0/AN0 | 2 | 2 | | | |
| RA0 | | | I/O | TTL | Digital I/O. |
| AN0 | | | I | Analog | Analog input 0. |
| RA1/AN1 | 3 | 3 | | | |
| RA1 | | | I/O | TTL | Digital I/O. |
| AN1 | | | I | Analog | Analog input 1. |
| RA2/AN2/VREF-/CVREF | 4 | 4 | | | |
| RA2 | | | I/O | TTL | Digital I/O. |
| AN2 | | | I | Analog | Analog input 2. |
| VREF- | | | I | Analog | A/D Reference Voltage (Low) input. |
| CVREF | | | O | Analog | Comparator Reference Voltage output. |
| RA3/AN3/VREF+ | 5 | 5 | | | |
| RA3 | | | I/O | TTL | Digital I/O. |
| AN3 | | | I | Analog | Analog input 3. |
| VREF+ | | | I | Analog | A/D Reference Voltage (High) input. |
| RA4/T0CKI/C1OUT | 6 | 6 | | | |
| RA4 | | | I/O | ST/OD | Digital I/O. Open-drain when configured as output. |
| T0CKI | | | I | ST | Timer0 external clock input. |
| C1OUT | | | O | — | Comparator 1 output. |
| RA5/AN4/SS/LVDIN/C2OUT | 7 | 7 | | | |
| RA5 | | | I/O | TTL | Digital I/O. |
| AN4 | | | I | Analog | Analog input 4. |
| SS | | | I | TTL | SPI Slave Select input. |
| LVDIN | | | I | Analog | Low-Voltage Detect input. |
| C2OUT | | | O | — | Comparator 2 output. |
| RA6 | | | | | See the OSC2/CLKO/RA6 pin. |
| RA7 | | | | | See the OSC1/CLKI/RA7 pin. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to VDD)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.

**2:** Alternate assignment for CCP2 when CCP2MX is cleared.

**TABLE 1-2: PIC18F2220/2320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|
| | PDIP | SOIC | | | |
| | | | | | PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs. |
| RB0/AN12/INT0 | 21 | 21 | | | |
| RB0 | | | I/O | TTL | Digital I/O. |
| AN12 | | | I | Analog | Analog input 12. |
| INT0 | | | I | ST | External interrupt 0. |
| RB1/AN10/INT1 | 22 | 22 | | | |
| RB1 | | | I/O | TTL | Digital I/O. |
| AN10 | | | I | Analog | Analog input 10. |
| INT1 | | | I | ST | External interrupt 1. |
| RB2/AN8/INT2 | 23 | 23 | | | |
| RB2 | | | I/O | TTL | Digital I/O. |
| AN8 | | | I | Analog | Analog input 8. |
| INT2 | | | I | ST | External interrupt 2. |
| RB3/AN9/CCP2 | 24 | 24 | | | |
| RB3 | | | I/O | TTL | Digital I/O. |
| AN9 | | | I | Analog | Analog input 9. |
| CCP2(1) | | | I/O | ST | Capture2 input, Compare2 output, PWM2 output. |
| RB4/AN11/KBI0 | 25 | 25 | | | |
| RB4 | | | I/O | TTL | Digital I/O. |
| AN11 | | | I | Analog | Analog input 11. |
| KBI0 | | | I | TTL | Interrupt-on-change pin. |
| RB5/KBI1/PGM | 26 | 26 | | | |
| RB5 | | | I/O | TTL | Digital I/O. |
| KBI1 | | | I | TTL | Interrupt-on-change pin. |
| PGM | | | I/O | ST | Low-voltage ICSP programming enable pin. |
| RB6/KBI2/PGC | 27 | 27 | | | |
| RB6 | | | I/O | TTL | Digital I/O. |
| KBI2 | | | I | TTL | Interrupt-on-change pin. |
| PGC | | | I/O | ST | In-Circuit Debugger and ICSP programming clock pin. |
| RB7/KBI3/PGD | 28 | 28 | | | |
| RB7 | | | I/O | TTL | Digital I/O. |
| KBI3 | | | I | TTL | Interrupt-on-change pin. |
| PGD | | | I/O | ST | In-Circuit Debugger and ICSP programming data pin. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to VDD)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.

**2:** Alternate assignment for CCP2 when CCP2MX is cleared.


© 2003 Microchip Technology Inc.

**TABLE 1-2: PIC18F2220/2320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|
| | PDIP | SOIC | | | |
| | | | | | PORTC is a bidirectional I/O port. |
| RC0/T1OSO/T1CKI | 11 | 11 | | | |
| RC0 | | | I/O | ST | Digital I/O. |
| T1OSO | | | O | — | Timer1 oscillator output. |
| T1CKI | | | I | ST | Timer1/Timer3 external clock input. |
| RC1/T1OSI/CCP2 | 12 | 12 | | | |
| RC1 | | | I/O | ST | Digital I/O. |
| T1OSI | | | I | CMOS | Timer1 oscillator input. |
| CCP2(2) | | | I/O | ST | Capture2 input, Compare2 output, PWM2 output. |
| RC2/CCP1/P1A | 13 | 13 | | | |
| RC2 | | | I/O | ST | Digital I/O. |
| CCP1 | | | I/O | ST | Capture1 input/Compare1 output/PWM1 output. |
| P1A | | | O | — | Enhanced CCP1 output. |
| RC3/SCK/SCL | 14 | 14 | | | |
| RC3 | | | I/O | ST | Digital I/O. |
| SCK | | | I/O | ST | Synchronous serial clock input/output for SPI mode. |
| SCL | | | I/O | ST | Synchronous serial clock input/output for $I^2C$ mode. |
| RC4/SDI/SDA | 15 | 15 | | | |
| RC4 | | | I/O | ST | Digital I/O. |
| SDI | | | I | ST | SPI data in. |
| SDA | | | I/O | ST | $I^2C$ data I/O. |
| RC5/SDO | 16 | 16 | | | |
| RC5 | | | I/O | ST | Digital I/O. |
| SDO | | | O | — | SPI data out. |
| RC6/TX/CK | 17 | 17 | | | |
| RC6 | | | I/O | ST | Digital I/O. |
| TX | | | O | — | USART asynchronous transmit. |
| CK | | | I/O | ST | USART synchronous clock (see related RX/DT). |
| RC7/RX/DT | 18 | 18 | | | |
| RC7 | | | I/O | ST | Digital I/O. |
| RX | | | I | ST | USART asynchronous receive. |
| DT | | | I/O | ST | USART synchronous data (see related TX/CK). |
| RE3 | — | — | — | — | See $\overline{MCLR}$/VPP/RE3 pin. |
| VSS | 8, 19 | 8, 19 | P | — | Ground reference for logic and I/O pins. |
| VDD | 20 | 20 | P | — | Positive supply for logic and I/O pins. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to VDD)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.

**2:** Alternate assignment for CCP2 when CCP2MX is cleared.

**TABLE 1-3: PIC18F4220/4320 PINOUT I/O DESCRIPTIONS**

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | PDIP | TQFP | QFN | | | |
| MCLR/VPP/RE3 | 1 | 18 | 18 | | | Master Clear (input) or programming voltage (input). |
| MCLR | | | | I | ST | Master Clear (Reset) input. This pin is an active-low Reset to the device. |
| VPP | | | | P | | Programming voltage input. |
| RE3 | | | | I | ST | Digital input. |
| OSC1/CLKI/RA7 | 13 | 30 | 32 | | | Oscillator crystal or external clock input. |
| OSC1 | | | | I | ST | Oscillator crystal input or external clock source input. ST buffer when configured in RC mode, CMOS otherwise. |
| CLKI | | | | I | CMOS | External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.) |
| RA7 | | | | I/O | TTL | General purpose I/O pin. |
| OSC2/CLKO/RA6 | 14 | 31 | 33 | | | Oscillator crystal or clock output. |
| OSC2 | | | | O | — | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. |
| CLKO | | | | O | — | In RC mode, OSC2 pin outputs CLKO which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. |
| RA6 | | | | I/O | TTL | General purpose I/O pin. |
| | | | | | | PORTA is a bidirectional I/O port. |
| RA0/AN0 | 2 | 19 | 19 | | | |
| RA0 | | | | I/O | TTL | Digital I/O. |
| AN0 | | | | I | Analog | Analog input 0. |
| RA1/AN1 | 3 | 20 | 20 | | | |
| RA1 | | | | I/O | TTL | Digital I/O. |
| AN1 | | | | I | Analog | Analog input 1. |
| RA2/AN2/VREF-/CVREF | 4 | 21 | 21 | | | |
| RA2 | | | | I/O | TTL | Digital I/O. |
| AN2 | | | | I | Analog | Analog input 2. |
| VREF- | | | | I | Analog | A/D reference voltage (Low) input. |
| CVREF | | | | O | Analog | Comparator reference voltage output. |
| RA3/AN3/VREF+ | 5 | 22 | 22 | | | |
| RA3 | | | | I/O | TTL | Digital I/O. |
| AN3 | | | | I | Analog | Analog input 3. |
| VREF+ | | | | I | Analog | A/D reference voltage (High) input. |
| RA4/T0CKI/C1OUT | 6 | 23 | 23 | | | |
| RA4 | | | | I/O | ST/OD | Digital I/O. Open-drain when configured as output. |
| T0CKI | | | | I | ST | Timer0 external clock input. |
| C1OUT | | | | O | — | Comparator 1 output. |
| RA5/AN4/SS/LVDIN/C2OUT | 7 | 24 | 24 | | | |
| RA5 | | | | I/O | TTL | Digital I/O. |
| AN4 | | | | I | Analog | Analog input 4. |
| SS | | | | I | TTL | SPI slave select input. |
| LVDIN | | | | I | Analog | Low-Voltage Detect input. |
| C2OUT | | | | O | — | Comparator 2 output. |
| RA6 | | | | | | See the OSC2/CLKO/RA6 pin. |
| RA7 | | | | | | See the OSC1/CLKI/RA7 pin. |

**Legend:** TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power; OD = Open-drain (no diode to VDD)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.
**2:** Alternate assignment for CCP2 when CCP2MX is cleared.


© 2003 Microchip Technology Inc.

**TABLE 1-3: PIC18F4220/4320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | PDIP | TQFP | QFN | | | |
| | | | | | | PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs. |
| RB0/AN12/INT0 | 33 | 8 | 9 | | | |
| RB0 | | | | I/O | TTL | Digital I/O. |
| AN12 | | | | I | Analog | Analog input 12. |
| INT0 | | | | I | ST | External interrupt 0. |
| RB1/AN10/INT1 | 34 | 9 | 10 | | | |
| RB1 | | | | I/O | TTL | Digital I/O. |
| AN10 | | | | I | Analog | Analog input 10. |
| INT1 | | | | I | ST | External interrupt 1. |
| RB2/AN8/INT2 | 35 | 10 | 11 | | | |
| RB2 | | | | I/O | TTL | Digital I/O. |
| AN8 | | | | I | Analog | Analog input 8. |
| INT2 | | | | I | ST | External interrupt 2. |
| RB3/AN9/CCP2 | 36 | 11 | 12 | | | |
| RB3 | | | | I/O | TTL | Digital I/O. |
| AN9 | | | | I | Analog | Analog input 9. |
| CCP2(1) | | | | I/O | ST | Capture2 input, Compare2 output, PWM2 output. |
| RB4/AN11/KBI0 | 37 | 14 | 14 | | | |
| RB4 | | | | I/O | TTL | Digital I/O. |
| AN11 | | | | I | Analog | Analog input 11. |
| KBI0 | | | | I | TTL | Interrupt-on-change pin. |
| RB5/KBI1/PGM | 38 | 15 | 15 | | | |
| RB5 | | | | I/O | TTL | Digital I/O. |
| KBI1 | | | | I | TTL | Interrupt-on-change pin. |
| PGM | | | | I/O | ST | Low-voltage ICSP programming enable pin. |
| RB6/KBI2/PGC | 39 | 16 | 16 | | | |
| RB6 | | | | I/O | TTL | Digital I/O. |
| KBI2 | | | | I | TTL | Interrupt-on-change pin. |
| PGC | | | | I/O | ST | In-Circuit Debugger and ICSP programming clock pin. |
| RB7/KBI3/PGD | 40 | 17 | 17 | | | |
| RB7 | | | | I/O | TTL | Digital I/O. |
| KBI3 | | | | I | TTL | Interrupt-on-change pin. |
| PGD | | | | I/O | ST | In-Circuit Debugger and ICSP programming data pin. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to $V_{DD}$)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.
**2:** Alternate assignment for CCP2 when CCP2MX is cleared.

**TABLE 1-3: PIC18F4220/4320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | PDIP | TQFP | QFN | | | |
| | | | | | | PORTC is a bidirectional I/O port. |
| RC0/T1OSO/T1CKI | 15 | 32 | 34 | | | |
| RC0 | | | | I/O | ST | Digital I/O. |
| T1OSO | | | | O | — | Timer1 oscillator output. |
| T1CKI | | | | I | ST | Timer1/Timer3 external clock input. |
| RC1/T1OSI/CCP2 | 16 | 35 | 35 | | | |
| RC1 | | | | I/O | ST | Digital I/O. |
| T1OSI | | | | I | CMOS | Timer1 oscillator input. |
| CCP2(2) | | | | I/O | ST | Capture2 input, Compare2 output, PWM2 output. |
| RC2/CCP1/P1A | 17 | 36 | 36 | | | |
| RC2 | | | | I/O | ST | Digital I/O. |
| CCP1 | | | | I/O | ST | Capture1 input/Compare1 output/PWM1 output. |
| P1A | | | | O | — | Enhanced CCP1 output. |
| RC3/SCK/SCL | 18 | 37 | 37 | | | |
| RC3 | | | | I/O | ST | Digital I/O. |
| SCK | | | | I/O | ST | Synchronous serial clock input/output for SPI mode. |
| SCL | | | | I/O | ST | Synchronous serial clock input/output for $I^2C$ mode. |
| RC4/SDI/SDA | 23 | 42 | 42 | | | |
| RC4 | | | | I/O | ST | Digital I/O. |
| SDI | | | | I | ST | SPI data in. |
| SDA | | | | I/O | ST | $I^2C$ data I/O. |
| RC5/SDO | 24 | 43 | 43 | | | |
| RC5 | | | | I/O | ST | Digital I/O. |
| SDO | | | | O | — | SPI data out. |
| RC6/TX/CK | 25 | 44 | 44 | | | |
| RC6 | | | | I/O | ST | Digital I/O. |
| TX | | | | O | — | USART asynchronous transmit. |
| CK | | | | I/O | ST | USART synchronous clock (see related RX/DT). |
| RC7/RX/DT | 26 | 1 | 1 | | | |
| RC7 | | | | I/O | ST | Digital I/O. |
| RX | | | | I | ST | USART asynchronous receive. |
| DT | | | | I/O | ST | USART synchronous data (see related TX/CK). |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to $V_{DD}$)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.
**2:** Alternate assignment for CCP2 when CCP2MX is cleared.


© 2003 Microchip Technology Inc.

**TABLE 1-3: PIC18F4220/4320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | PDIP | TQFP | QFN | | | |
| | | | | | | PORTD is a bidirectional I/O port or a Parallel Slave Port (PSP) for interfacing to a microprocessor port. These pins have TTL input buffers when PSP module is enabled. |
| RD0/PSP0 | 19 | 38 | 38 | | | |
| RD0 | | | | I/O | ST | Digital I/O. |
| PSP0 | | | | I/O | TTL | Parallel Slave Port data. |
| RD1/PSP1 | 20 | 39 | 39 | | | |
| RD1 | | | | I/O | ST | Digital I/O. |
| PSP1 | | | | I/O | TTL | Parallel Slave Port data. |
| RD2/PSP2 | 21 | 40 | 40 | | | |
| RD2 | | | | I/O | ST | Digital I/O. |
| PSP2 | | | | I/O | TTL | Parallel Slave Port data. |
| RD3/PSP3 | 22 | 41 | 41 | | | |
| RD3 | | | | I/O | ST | Digital I/O. |
| PSP3 | | | | I/O | TTL | Parallel Slave Port data. |
| RD4/PSP4 | 27 | 2 | 2 | | | |
| RD4 | | | | I/O | ST | Digital I/O. |
| PSP4 | | | | I/O | TTL | Parallel Slave Port data. |
| RD5/PSP5/P1B | 28 | 3 | 3 | | | |
| RD5 | | | | I/O | ST | Digital I/O. |
| PSP5 | | | | I/O | TTL | Parallel Slave Port data. |
| P1B | | | | O | — | Enhanced CCP1 output. |
| RD6/PSP6/P1C | 29 | 4 | 4 | | | |
| RD6 | | | | I/O | ST | Digital I/O. |
| PSP6 | | | | I/O | TTL | Parallel Slave Port data. |
| P1C | | | | O | — | Enhanced CCP1 output. |
| RD7/PSP7/P1D | 30 | 5 | 5 | | | |
| RD7 | | | | I/O | ST | Digital I/O. |
| PSP7 | | | | I/O | TTL | Parallel Slave Port data. |
| P1D | | | | O | — | Enhanced CCP1 output. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to $V_{DD}$)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.
**2:** Alternate assignment for CCP2 when CCP2MX is cleared.

**TABLE 1-3: PIC18F4220/4320 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | PDIP | TQFP | QFN | | | |
| | | | | | | PORTE is a bidirectional I/O port. |
| RE0/AN5/$\overline{RD}$ | 8 | 25 | 25 | | | |
| RE0 | | | | I/O | ST | Digital I/O. |
| AN5 | | | | I | Analog | Analog input 5. |
| $\overline{RD}$ | | | | I | TTL | Read control for Parallel Slave Port (see also $\overline{WR}$ and $\overline{CS}$ pins). |
| RE1/AN6/$\overline{WR}$ | 9 | 26 | 26 | | | |
| RE1 | | | | I/O | ST | Digital I/O. |
| AN6 | | | | I | Analog | Analog input 6. |
| $\overline{WR}$ | | | | I | TTL | Write control for Parallel Slave Port (see $\overline{CS}$ and $\overline{RD}$ pins). |
| RE2/AN7/$\overline{CS}$ | 10 | 27 | 27 | | | |
| RE2 | | | | I/O | ST | Digital I/O. |
| AN7 | | | | I | Analog | Analog input 7. |
| $\overline{CS}$ | | | | I | TTL | Chip select control for Parallel Slave Port (see related $\overline{RD}$ and $\overline{WR}$). |
| RE3 | 1 | 18 | 18 | — | — | See $\overline{MCLR}$/VPP/RE3 pin. |
| VSS | 12, 31 | 6, 29 | 6, 30, 31 | P | — | Ground reference for logic and I/O pins. |
| VDD | 11, 32 | 7, 28 | 7, 8, 28, 29 | P | — | Positive supply for logic and I/O pins. |
| NC | — | — | 13 | NC | NC | No connect. |

**Legend:** TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
I = Input
O = Output
P = Power
OD = Open-drain (no diode to VDD)

**Note 1:** Default assignment for CCP2 when CCP2MX (CONFIG3H<0>) is set.
**2:** Alternate assignment for CCP2 when CCP2MX is cleared.


© 2003 Microchip Technology Inc.

## 2.0 OSCILLATOR CONFIGURATIONS

### 2.1 Oscillator Types

The PIC18F2X20 and PIC18F4X20 devices can be operated in ten different oscillator modes. The user can program the configuration bits, FOSC3:FOSC0, in Configuration Register 1H to select one of these ten modes:

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

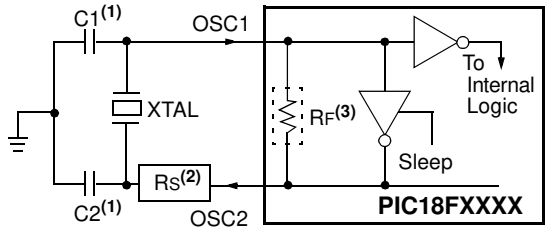
### 2.2 Crystal Oscillator/Ceramic Resonators

In XT, LP, HS or HSPLL Oscillator modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. Figure 2-1 shows the pin connections.

The oscillator design requires the use of a parallel cut crystal.

| **Note:** | Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. |
|---|---|

**FIGURE 2-1: CRYSTAL/CERAMIC RESONATOR OPERATION (XT, LP, HS OR HSPLL CONFIGURATION)**




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

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

**3:** RF varies with the oscillator mode chosen.

**TABLE 2-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS**

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

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

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

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

See the notes on page 20 for additional information.

| Resonators Used: | |
|---|---|
| 455 kHz | 4.0 MHz |
| 2.0 MHz | 8.0 MHz |
| 16.0 MHz | |

**TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR**

| Osc Type | Crystal Freq | Typical Capacitor Values Tested: | |
|---|---|---|---|
| | | C1 | C2 |
| LP | 32 kHz | 33 pF | 33 pF |
| | 200 kHz | 15 pF | 15 pF |
| XT | 1 MHz | 33 pF | 33 pF |
| | 4 MHz | 27 pF | 27 pF |
| HS | 4 MHz | 27 pF | 27 pF |
| | 8 MHz | 22 pF | 22 pF |
| | 20 MHz | 15 pF | 15 pF |

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

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

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

See the notes following this table for additional information.

| Crystals Used: | |
|---|---|
| 32 kHz | 4 MHz |
| 200 kHz | 8 MHz |
| 1 MHz | 20 MHz |

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

**2:** When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.

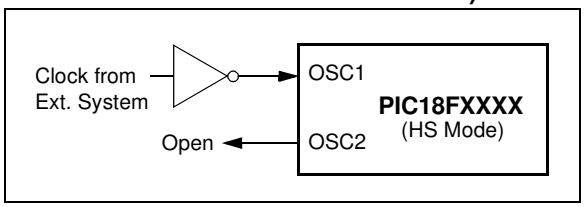
**3:** Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

**4:** RS may be required to avoid overdriving crystals with low drive level specification.

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

An external clock source may also be connected to the OSC1 pin in the HS mode, as shown in Figure 2-2.

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



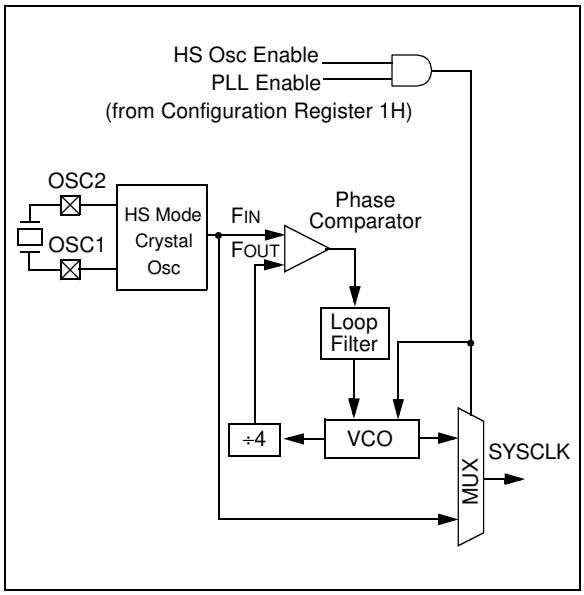

## 2.3 HSPLL

A Phase Locked Loop (PLL) circuit is provided as an option for users who wish to use a lower frequency crystal oscillator circuit, or to clock the device up to its highest rated frequency from a crystal oscillator. This may be useful for customers who are concerned with EMI due to high-frequency crystals.

The HSPLL mode makes use of the HS mode oscillator for frequencies up to 10 MHz. A PLL then multiplies the oscillator output frequency by 4 to produce an internal clock frequency up to 40 MHz.

The PLL is enabled only when the oscillator configuration bits are programmed for HSPLL mode. If programmed for any other mode, the PLL is not enabled.

**FIGURE 2-3: PLL BLOCK DIAGRAM**





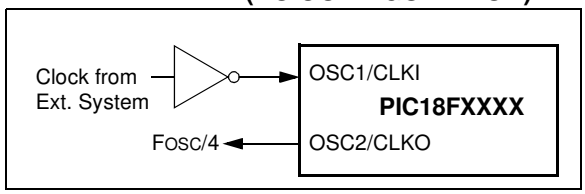
© 2003 Microchip Technology Inc.

## 2.4 External Clock Input

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

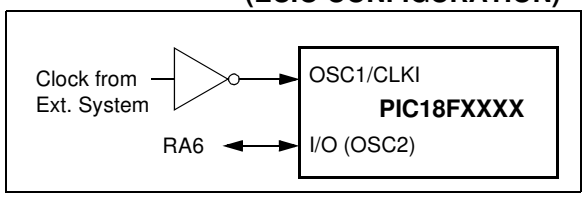
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-4 shows the pin connections for the EC Oscillator mode.

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




The ECIO Oscillator mode functions like the EC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 2-5 shows the pin connections for the ECIO Oscillator mode.

**FIGURE 2-5: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)**

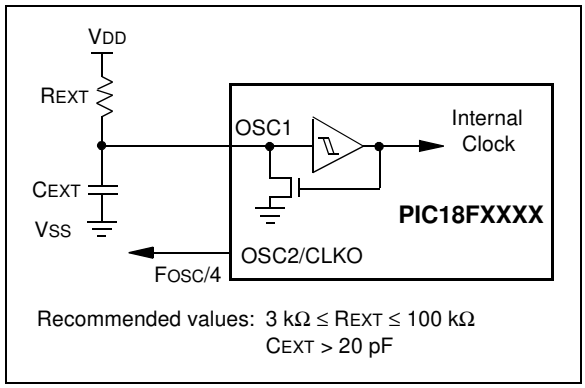



## 2.5 RC Oscillator

For timing insensitive applications, the "RC" and "RCIO" device options offer additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal manufacturing variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 2-6 shows how the R/C combination is connected.

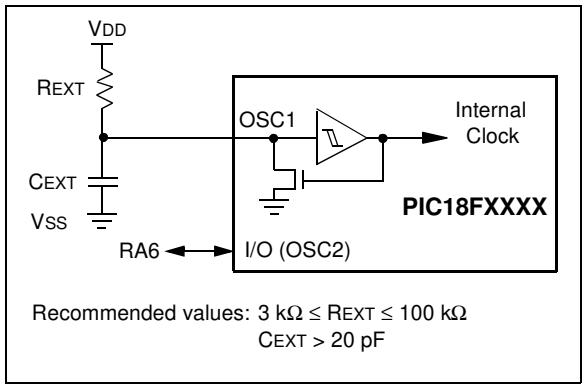
In the RC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic.

**FIGURE 2-6: RC OSCILLATOR MODE**




The RCIO Oscillator mode (Figure 2-7) functions like the RC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6).

**FIGURE 2-7: RCIO OSCILLATOR MODE**



## 2.6 Internal Oscillator Block

The PIC18F2X20/4X20 devices include an internal oscillator block which generates two different clock signals. Either can be used as the system's clock source. This can eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

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

The other clock source is the internal RC oscillator (INTRC) which provides a 31 kHz output. The INTRC oscillator is enabled by selecting the internal oscillator block as the system clock source or when any of the following are enabled:

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

These features are discussed in greater detail in **Section 23.0 "Special Features of the CPU"**.

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (page 26).

### 2.6.1 INTIO MODES

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

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

### 2.6.2 INTRC OUTPUT FREQUENCY

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8.0 MHz. This changes the frequency of the INTRC source from its nominal 31.25 kHz. Peripherals and features that depend on the INTRC source will be affected by this shift in frequency.

Once set during factory calibration, the INTRC frequency will remain within ±1% as temperature and VDD change across their full specified operating ranges.

### 2.6.3 OSCTUNE REGISTER

The internal oscillator's output has been calibrated at the factory but can be adjusted in the user's application. This is done by writing to the OSCTUNE register (Register 2-1). The tuning sensitivity is constant throughout the tuning range.

When the OSCTUNE register is modified, the INTOSC and INTRC frequencies will begin shifting to the new frequency. The INTRC clock will reach the new frequency within 8 clock cycles (approximately 8 * 32 µs = 256 µs). The INTOSC clock will stabilize within 1 ms. Code execution continues during this shift. There is no indication that the shift has occurred. Operation of features that depend on the INTRC clock source frequency, such as the WDT, Fail-Safe Clock Monitor and peripherals, will also be affected by the change in frequency.


© 2003 Microchip Technology Inc.

**REGISTER 2-1: OSCTUNE: OSCILLATOR TUNING REGISTER**

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

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

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

011111 = Maximum frequency (+12.5%, approximately)
•
•
000001
000000 = Center frequency. Oscillator module is running at the calibrated frequency.
111111
•
•
100000 = Minimum frequency (-12.5%, approximately)

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

## 2.7 Clock Sources and Oscillator Switching

Like previous PIC18 devices, the PIC18F2X20 and PIC18F4X20 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low-frequency clock source. PIC18F2X20/4X20 devices offer two alternate clock sources. When enabled, these give additional options for switching to the various power managed operating modes.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator block

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

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

PIC18F2X20/4X20 devices offer only the Timer1 oscillator as a secondary oscillator. This oscillator, in all power managed modes, is often the time base for functions such as a real-time clock.

Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T1CKI and RC1/T1OSI pins. Like the LP mode oscillator circuit, loading capacitors are also connected from each pin to ground.

The Timer1 oscillator is discussed in greater detail in **Section 12.2 "Timer1 Oscillator"**.

In addition to being a primary clock source, the **internal oscillator block** is available as a power managed mode clock source. The INTRC source is also used as the clock source for several special features, such as the WDT and Fail-Safe Clock Monitor.

The clock sources for the PIC18F2X20/4X20 devices are shown in Figure 2-8. See **Section 12.0 "Timer1 Module"** for further details of the Timer1 oscillator. See **Section 23.1 "Configuration Bits"** for Configuration register details.

### 2.7.1 OSCILLATOR CONTROL REGISTER

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

The System Clock Select bits, SCS1:SCS0, select the clock source that is used when the device is operating in power managed modes. The available clock sources are the primary clock (defined in Configuration Register 1H), the secondary clock (Timer1 oscillator) and the internal oscillator block. The clock selection has no effect until a `SLEEP` instruction is executed and the device enters a power managed mode of operation. The SCS bits are cleared on all forms of Reset.

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

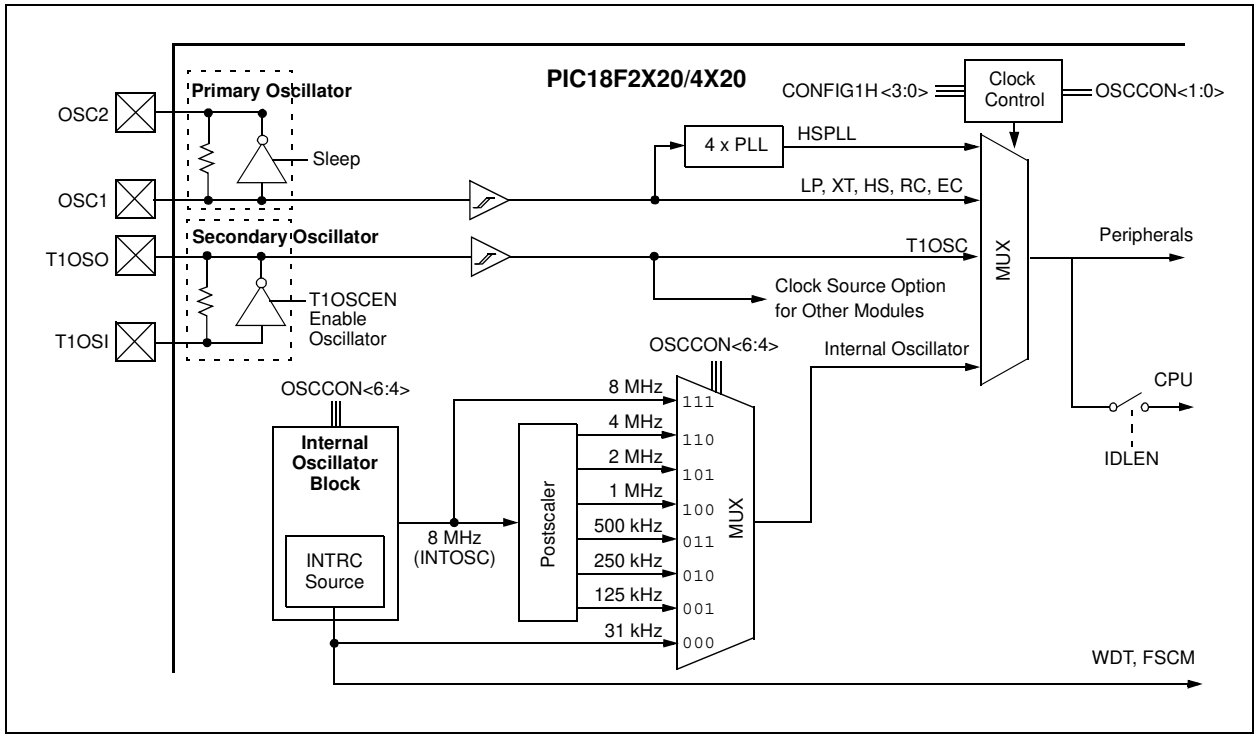
The OSTS, IOFS and T1RUN bits indicate which clock source is currently providing the system clock. The OSTS indicates that the Oscillator Start-up Timer has timed out and the primary clock is providing the system clock in primary clock modes. The IOFS bit indicates when the internal oscillator block has stabilized and is providing the system clock in RC Clock modes. The T1RUN bit (T1CON<6>) indicates when the Timer1 oscillator is providing the system clock in secondary clock modes. If none of these bits are set, the INTRC is providing the system clock, or the internal oscillator block has just started and is not yet stable.

The IDLEN bit controls the selective shutdown of the controller's CPU in power managed modes. The use of these bits is discussed in more detail in **Section 3.0 "Power Managed Modes"**.

| | |
|---|---|
| **Note 1:** | The Timer1 oscillator must be enabled to select the secondary clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON<3>). If the Timer1 oscillator is not enabled, then any attempt to set the SCS0 bit will be ignored. |
| **2:** | It is recommended that the Timer1 oscillator be operating and stable before executing the `SLEEP` instruction or a very long delay may occur while the Timer1 oscillator starts. |


© 2003 Microchip Technology Inc.

**FIGURE 2-8: PIC18F2X20/4X20 CLOCK DIAGRAM**



**REGISTER 2-2: OSCCON REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R[(1)] | R-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| IDLEN | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **IDLEN:** Idle Enable bit
1 = Idle mode enabled; CPU core is not clocked in power managed modes
0 = Run mode enabled; CPU core is clocked in power managed modes

bit 6-4 **IRCF2:IRCF0:** Internal Oscillator Frequency Select bits
111 = 8 MHz (8 MHz source drives clock directly)
110 = 4 MHz
101 = 2 MHz
100 = 1 MHz
011 = 500 kHz
010 = 250 kHz
001 = 125 kHz
000 = 31 kHz (INTRC source drives clock directly)

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit[(1)]
1 = Oscillator start-up time-out timer has expired; primary oscillator is running
0 = Oscillator start-up time-out timer is running; primary oscillator is not ready

bit 2 **IOFS:** INTOSC Frequency Stable bit
1 = INTOSC frequency is stable
0 = INTOSC frequency is not stable

bit 1-0 **SCS1:SCS0:** System Clock Select bits
1x = Internal oscillator block (RC modes)
01 = Timer1 oscillator (Secondary modes)[(2)]
00 = Primary oscillator (Sleep and PRI_IDLE modes)

**Note 1:** Depends on state of IESO bit in Configuration Register 1H.
**2:** SCS0 may not be set while T1OSCEN (T1CON<3>) is clear.

| **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 |


© 2003 Microchip Technology Inc.

### 2.7.2 OSCILLATOR TRANSITIONS

The PIC18F2X20/4X20 devices contain circuitry to prevent clocking "glitches" when switching between clock sources. A short pause in the system clock occurs during the clock switch. The length of this pause is between 8 and 9 clock periods of the new clock source. This ensures that the new clock source is stable and that its pulse width will not be less than the shortest pulse width of the two clock sources.

Clock transitions are discussed in greater detail in **Section 3.1.2 "Entering Power Managed Modes"**.

## 2.8 Effects of Power Managed Modes on the Various Clock Sources

When the device executes a SLEEP instruction, the system is switched to one of the power managed modes, depending on the state of the IDLEN and SCS1:SCS0 bits of the OSCCON register. See **Section 3.0 "Power Managed Modes"** for details.

When PRI_IDLE mode is selected, the designated primary oscillator continues to run without interruption. For all other power managed modes, the oscillator using the OSC1 pin is disabled. The OSC1 pin (and OSC2 pin, if used by the oscillator) will stop oscillating.

In secondary clock modes (SEC_RUN and SEC_IDLE), the Timer1 oscillator is operating and providing the system clock. The Timer1 oscillator may also run in all power managed modes if required to clock Timer1 or Timer3.

In internal oscillator modes (RC_RUN and RC_IDLE), the internal oscillator block provides the system clock source. The INTRC output can be used directly to provide the system clock and may be enabled to support various special features, regardless of the power managed mode (see **Section 23.2 "Watchdog Timer (WDT)"** through **Section 23.4 "Fail-Safe Clock Monitor"**). The INTOSC output at 8 MHz may be used directly to clock the system or may be divided down first. The INTOSC output is disabled if the system clock is provided directly from the INTRC output.

If the Sleep mode is selected, all clock sources are stopped. Since all the transistor switching currents have been stopped, Sleep mode achieves the lowest current consumption of the device (only leakage currents).

Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The INTRC is required to support WDT operation. The Timer1 oscillator may be operating to support a real-time clock. Other features may be operating that do not require a system clock source (i.e., SSP slave, PSP, INTn pins, A/D conversions and others).

## 2.9 Power-up Delays

Power-up delays are controlled by two timers so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply is stable under normal circumstances and the primary clock is operating and stable. For additional information on power-up delays, see **Section 4.1 "Power-on Reset (POR)"** through **Section 4.5 "Brown-out Reset (BOR)"**.

The first timer is the Power-up Timer (PWRT) which provides a fixed delay on power-up (parameter 33, Table 26-10), if enabled, in Configuration Register 2L. The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable (LP, XT and HS modes). The OST does this by counting 1024 oscillator cycles before allowing the oscillator to clock the device.

When the HSPLL Oscillator mode is selected, the device is kept in Reset for an additional 2 ms, following the HS mode OST delay, so the PLL can lock to the incoming clock frequency.

There is a delay of 5 to 10 μs, following POR, while the controller becomes ready to execute instructions. This delay runs concurrently with any other delays. This may be the only delay that occurs when any of the EC, RC or INTIO modes are used as the primary clock source.

**TABLE 2-3: OSC1 AND OSC2 PIN STATES IN SLEEP MODE**

| OSC Mode | OSC1 Pin | OSC2 Pin |
|---|---|---|
| RC, INTIO1 | Floating, external resistor should pull high | At logic low (clock/4 output) |
| RCIO, INTIO2 | Floating, external resistor should pull high | Configured as PORTA, bit 6 |
| ECIO | Floating, pulled by external clock | Configured as PORTA, bit 6 |
| EC | Floating, pulled by external clock | At logic low (clock/4 output) |
| LP, XT, and HS | Feedback inverter disabled at quiescent voltage level | Feedback inverter disabled at quiescent voltage level |

**Note:** See Table 4-1 in **Section 4.0 "Reset"** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

**NOTES:**


© 2003 Microchip Technology Inc.

## 3.0 POWER MANAGED MODES

The PIC18F2X20 and PIC18F4X20 devices offer a total of six operating modes for more efficient power management (see Table 3-1). These operating modes provide a variety of options for selective power conservation in applications where resources may be limited (i.e., battery-powered devices).

There are three categories of power managed modes:

- Sleep mode
- Idle modes
- Run modes

These categories define which portions of the device are clocked and sometimes, what speed. The Run and Idle modes may use any of the three available clock sources (primary, secondary or INTOSC multiplexer); the Sleep mode does not use a clock source.

The clock switching feature offered in other PIC18 devices (i.e., using the Timer1 oscillator in place of the primary oscillator) and the Sleep mode offered by all PICmicro® devices (where all system clocks are stopped) are both offered in the PIC18F2X20/4X20 devices (SEC_RUN and Sleep modes, respectively). However, additional power managed modes are available that allow the user greater flexibility in determining what portions of the device are operating. The power managed modes are event driven; that is, some specific event must occur for the device to enter or (more particularly) exit these operating modes.

For PIC18F2X20/4X20 devices, the power managed modes are invoked by using the existing `SLEEP` instruction. All modes exit to PRI_RUN mode when triggered by an interrupt, a Reset, or a WDT time-out (PRI_RUN mode is the normal full power execution mode; the CPU and peripherals are clocked by the primary oscillator source). In addition, power managed Run modes may also exit to Sleep mode or their corresponding Idle mode.

### 3.1 Selecting Power Managed Modes

Selecting a power managed mode requires deciding if the CPU is to be clocked or not and selecting a clock source. The IDLEN bit controls CPU clocking while the SC1:SCS0 bits select a clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 3-1.

#### 3.1.1 CLOCK SOURCES

The clock source is selected by setting the SCS bits of the OSCCON register. Three clock sources are available for use in power managed Idle modes: the primary clock (as configured in Configuration Register 1H), the secondary clock (Timer1 oscillator) and the internal oscillator block. The secondary and internal oscillator block sources are available for the power managed modes (PRI_RUN mode is the normal full power execution mode; the CPU and peripherals are clocked by the primary oscillator source).

**TABLE 3-1: POWER MANAGED MODES**

| Mode | OSCCON Bits | | Module Clocking | | Available Clock and Oscillator Source |
|---|---|---|---|---|---|
| | IDLEN <7> | SCS1:SCS0 <1:0> | CPU | Peripherals | |
| Sleep | 0 | 00 | Off | Off | None – All clocks are disabled |
| PRI_RUN | 0 | 00 | Clocked | Clocked | Primary – LP, XT, HS, HSPLL, RC, EC, INTRC(1). This is the normal full power execution mode. |
| SEC_RUN | 0 | 01 | Clocked | Clocked | Secondary – Timer1 Oscillator |
| RC_RUN | 0 | 1x | Clocked | Clocked | Internal Oscillator Block(1) |
| PRI_IDLE | 1 | 00 | Off | Clocked | Primary – LP, XT, HS, HSPLL, RC, EC |
| SEC_IDLE | 1 | 01 | Off | Clocked | Secondary – Timer1 Oscillator |
| RC_IDLE | 1 | 1x | Off | Clocked | Internal Oscillator Block(1) |

**Note 1:** Includes INTOSC and INTOSC postscaler, as well as the INTRC source.

### 3.1.2 ENTERING POWER MANAGED MODES

In general, entry, exit and switching between power managed clock sources requires clock source switching. In each case, the sequence of events is the same.

Any change in the power managed mode begins with loading the OSCCON register and executing a SLEEP instruction. The SCS1:SCS0 bits select one of three power managed clock sources; the primary clock (as defined in Configuration Register 1H), the secondary clock (the Timer1 oscillator) and the internal oscillator block (used in RC modes). Modifying the SCS bits will have no effect until a SLEEP instruction is executed. Entry to the power managed mode is triggered by the execution of a SLEEP instruction.

Figure 3-5 shows how the system is clocked while switching from the primary clock to the Timer1 oscillator. When the SLEEP instruction is executed, clocks to the device are stopped at the beginning of the next instruction cycle. Eight clock cycles from the new clock source are counted to synchronize with the new clock source. After eight clock pulses from the new clock source are counted, clocks from the new clock source resume clocking the system. The actual length of the pause is between eight and nine clock periods from the new clock source. This ensures that the new clock source is stable and that its pulse width will not be less than the shortest pulse width of the two clock sources.

Three bits indicate the current clock source: OSTS and IOFS in the OSCCON register and T1RUN in the T1CON register. Only one of these bits will be set while in a power managed mode other than PRI_RUN. When the OSTS bit is set, the primary clock is providing the system clock. When the IOFS bit is set, the INTOSC output is providing a stable 8 MHz clock source and is providing the system clock. When the T1RUN bit is set, the Timer1 oscillator is providing the system clock. If none of these bits are set, then either the INTRC clock source is clocking the system or the INTOSC source is not yet stable.

If the internal oscillator block is configured as the primary clock source in Configuration Register 1H, then both the OSTS and IOFS bits may be set when in PRI_RUN or PRI_IDLE modes. This indicates that the primary clock (INTOSC output) is generating a stable 8 MHz output. Entering a power managed RC mode (same frequency) would clear the OSTS bit.

**Note 1:** Caution should be used when modifying a single IRCF bit. If $V_{DD}$ is less than 3V, it is possible to select a higher clock speed than is supported by the low $V_{DD}$. Improper device operation may result if the $V_{DD}$/FOSC specifications are violated.

**2:** Executing a SLEEP instruction does not necessarily place the device into Sleep mode; executing a SLEEP instruction is simply a trigger to place the controller into a power managed mode selected by the OSCCON register, one of which is Sleep mode.

### 3.1.3 MULTIPLE SLEEP COMMANDS

The power managed mode that is invoked with the SLEEP instruction is determined by the settings of the IDLEN and SCS bits at the time the instruction is executed. If another SLEEP instruction is executed, the device will enter the power managed mode specified by these same bits at that time. If the bits have changed, the device will enter the new power managed mode specified by the new bit settings.

### 3.1.4 COMPARISONS BETWEEN RUN AND IDLE MODES

Clock source selection for the Run modes is identical to the corresponding Idle modes. When a SLEEP instruction is executed, the SCS bits in the OSCCON register are used to switch to a different clock source. As a result, if there is a change of clock source at the time a SLEEP instruction is executed, a clock switch will occur.

In Idle modes, the CPU is not clocked and is not running. In Run modes, the CPU is clocked and executing code. This difference modifies the operation of the WDT when it times out. In Idle modes, a WDT time-out results in a wake from power managed modes. In Run modes, a WDT time-out results in a WDT Reset (see Table 3-2).

During a wake-up from an Idle mode, the CPU starts executing code by entering the corresponding Run mode until the primary clock becomes ready. When the primary clock becomes ready, the clock source is automatically switched to the primary clock. The IDLEN and SCS bits are unchanged during and after the wake-up.

Figure 3-2 shows how the system is clocked during the clock source switch. The example assumes the device was in SEC_IDLE or SEC_RUN mode when a wake is triggered (the primary clock was configured in HSPLL mode).


© 2003 Microchip Technology Inc.

**TABLE 3-2: COMPARISON BETWEEN POWER MANAGED MODES**

| Power Managed Mode | CPU is clocked by ... | WDT time-out causes a ... | Peripherals are clocked by ... | Clock during wake-up (while primary becomes ready) |
|---|---|---|---|---|
| Sleep | Not clocked (not running) | Wake-up | Not clocked | None or INTOSC multiplexer if Two-Speed Start-up or Fail-Safe Clock Monitor are enabled. |
| Any Idle mode | Not clocked (not running) | Wake-up | Primary, Secondary or INTOSC multiplexer | Unchanged from Idle mode (CPU operates as in corresponding Run mode). |
| Any Run mode | Secondary or INTOSC multiplexer | Reset | Secondary or INTOSC multiplexer | Unchanged from Run mode. |

## 3.2 Sleep Mode

The power managed Sleep mode in the PIC18F2X20/4X20 devices is identical to that offered in all other PICmicro controllers. It is entered by clearing the IDLEN and SCS1:SCS0 bits (this is the Reset state) and executing the SLEEP instruction. This shuts down the primary oscillator and the OSTS bit is cleared (see Figure 3-1).

When a wake event occurs in Sleep mode (by interrupt, Reset or WDT time-out), the system will not be clocked until the primary clock source becomes ready (see Figure 3-2), or it will be clocked from the internal oscillator block if either the Two-Speed Start-up or the Fail-Safe Clock Monitor are enabled (see **Section 23.0 "Special Features of the CPU"**). In either case, the OSTS bit is set when the primary clock is providing the system clocks. The IDLEN and SCS bits are not affected by the wake-up.

## 3.3 Idle Modes

The IDLEN bit allows the controller's CPU to be selectively shut down while the peripherals continue to operate. Clearing IDLEN allows the CPU to be clocked. Setting IDLEN disables clocks to the CPU, effectively stopping program execution (see Register 2-2). The peripherals continue to be clocked regardless of the setting of the IDLEN bit.

There is one exception to how the IDLEN bit functions. When all the low-power OSCCON bits are cleared (IDLEN:SCS1:SCS0 = 000), the device enters Sleep mode upon the execution of the SLEEP instruction. This is both the Reset state of the OSCCON register and the setting that selects Sleep mode. This maintains compatibility with other PICmicro devices that do not offer power managed modes.

If the Idle Enable bit, IDLEN (OSCCON<7>), is set to a '1' when a SLEEP instruction is executed, the peripherals will be clocked from the clock source selected using the SCS1:SCS0 bits; however, the CPU will not be clocked. Since the CPU is not executing instructions, the only exits from any of the Idle modes are by interrupt, WDT time-out or a Reset.

When a wake-up event occurs, CPU execution is delayed approximately 10 µs while it becomes ready to execute code. When the CPU begins executing code, it is clocked by the same clock source as was selected in the power managed mode (i.e., when waking from RC_IDLE mode, the internal oscillator block will clock the CPU and peripherals until the primary clock source becomes ready – this is essentially RC_RUN mode). This continues until the primary clock source becomes ready. When the primary clock becomes ready, the OSTS bit is set and the system clock source is switched to the primary clock (see Figure 3-4). The IDLEN and SCS bits are not affected by the wake-up.

While in any Idle mode or the Sleep mode, a WDT time-out will result in a WDT wake-up to full power operation.

**FIGURE 3-1: TIMING TRANSITION FOR ENTRY TO SLEEP MODE**

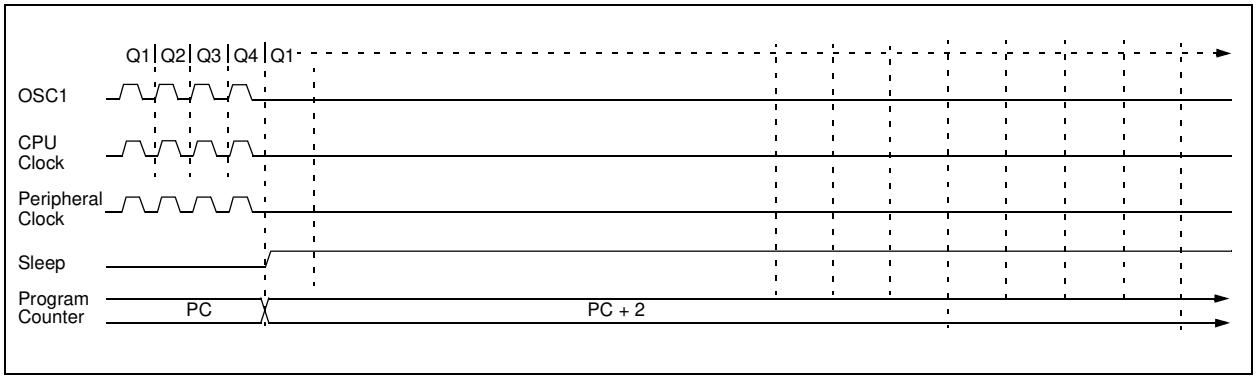
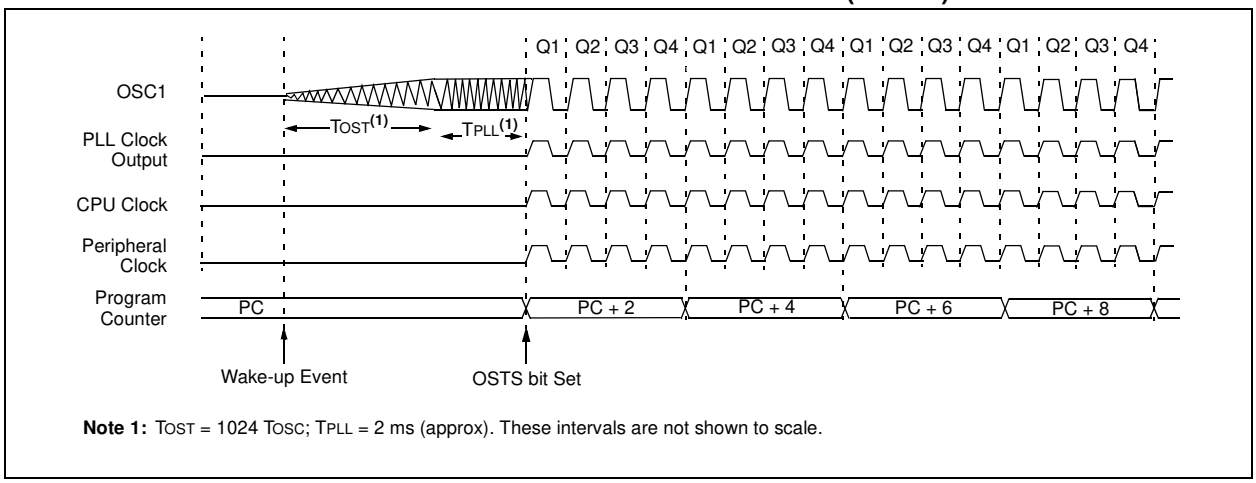



**FIGURE 3-2: TRANSITION TIMING FOR WAKE FROM SLEEP (HSPLL)**




**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.


© 2003 Microchip Technology Inc.

### 3.3.1 PRI_IDLE MODE

This mode is unique among the three Low-Power Idle modes in that it does not disable the primary system clock. For timing sensitive applications, this allows for the fastest resumption of device operation, with its more accurate primary clock source, since the clock source does not have to “warm up” or transition from another oscillator.

PRI_IDLE mode is entered by setting the IDLEN bit, clearing the SCS bits and executing a `SLEEP` instruction. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified in Configuration Register 1H. The OSTS bit remains set in PRI_IDLE mode (see Figure 3-3).

When a wake-up event occurs, the CPU is clocked from the primary clock source. A delay of approximately 10 µs is required between the wake-up event and when code execution starts. This is required to allow the CPU to become ready to execute instructions. After the wake-up, the OSTS bit remains set. The IDLEN and SCS bits are not affected by the wake-up (see Figure 3-4).

**FIGURE 3-3: TRANSITION TIMING TO PRI_IDLE MODE**

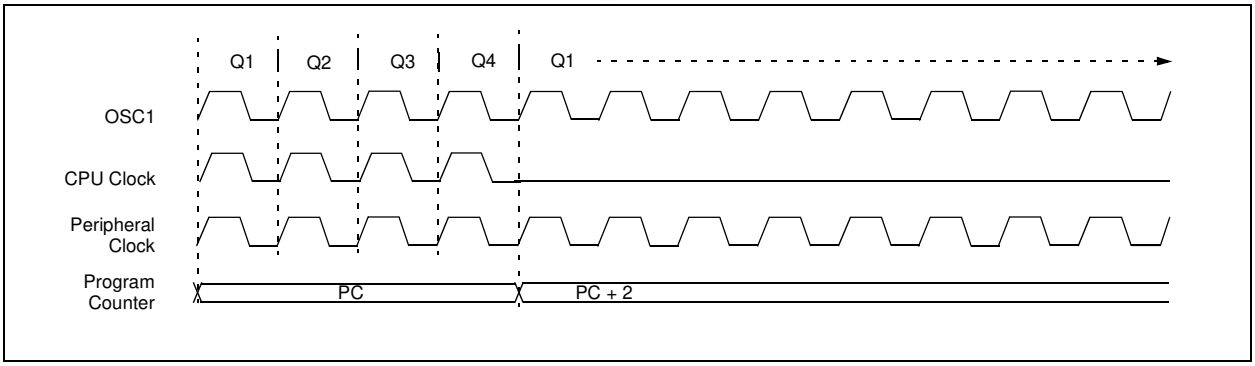
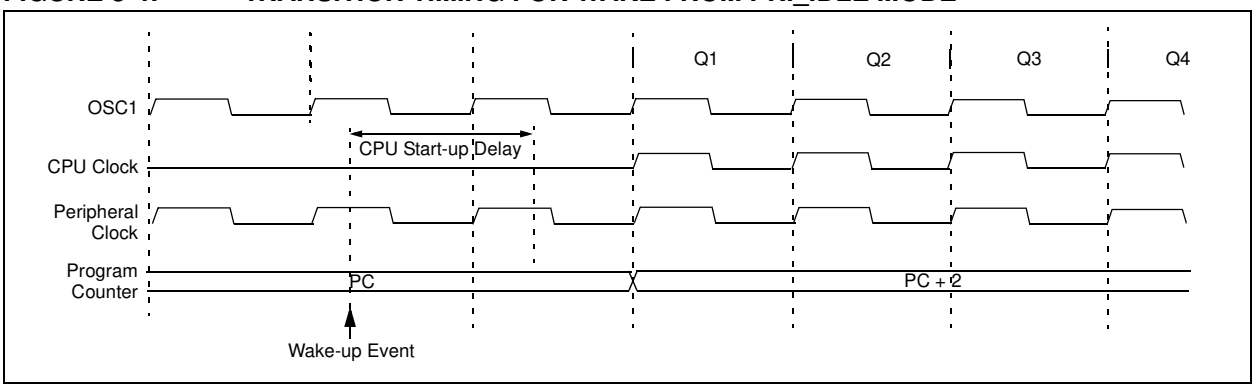



**FIGURE 3-4: TRANSITION TIMING FOR WAKE FROM PRI_IDLE MODE**



### 3.3.2 SEC_IDLE MODE

In SEC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered by setting the IDLEN bit, modifying to SCS1:SCS0 = 01 and executing a SLEEP instruction. When the clock source is switched to the Timer1 oscillator (see Figure 3-5), the primary oscillator is shut down, the OSTS bit is cleared and the T1RUN bit is set.

**Note:** The Timer1 oscillator should already be running prior to entering SEC_IDLE mode. **If the T1OSCEN bit is not set when trying to set the SCS0 bit (OSCCON<0>), the write to SCS0 will not occur.** If the Timer1 oscillator is enabled but not yet running, peripheral clocks will be delayed until the oscillator has started; in such situations, initial oscillator operation is far from stable and unpredictable operation may result.

When a wake-up event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After a 10 µs delay following the wake-up event, the CPU begins executing code, being clocked by the Timer1 oscillator. The microcontroller operates in SEC_RUN mode until the primary clock becomes ready. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 3-6). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

**FIGURE 3-5: TIMING TRANSITION FOR ENTRY TO SEC_IDLE MODE**

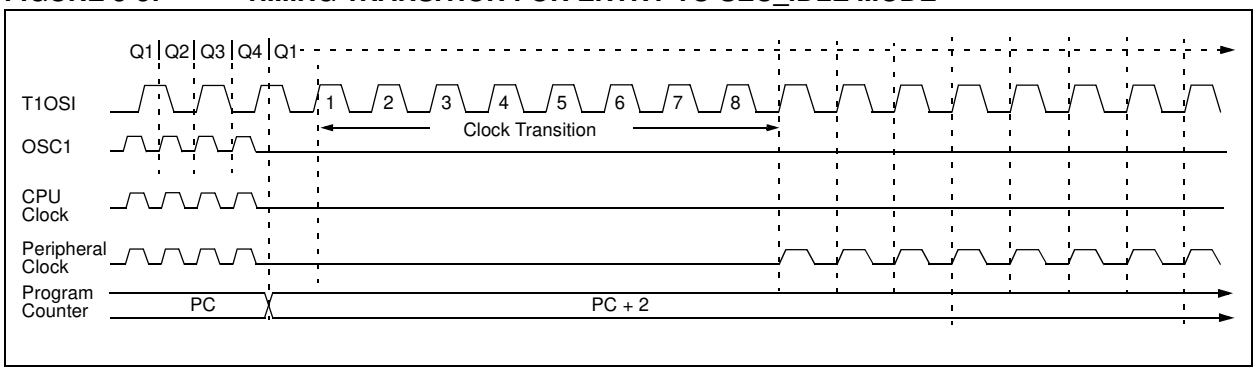
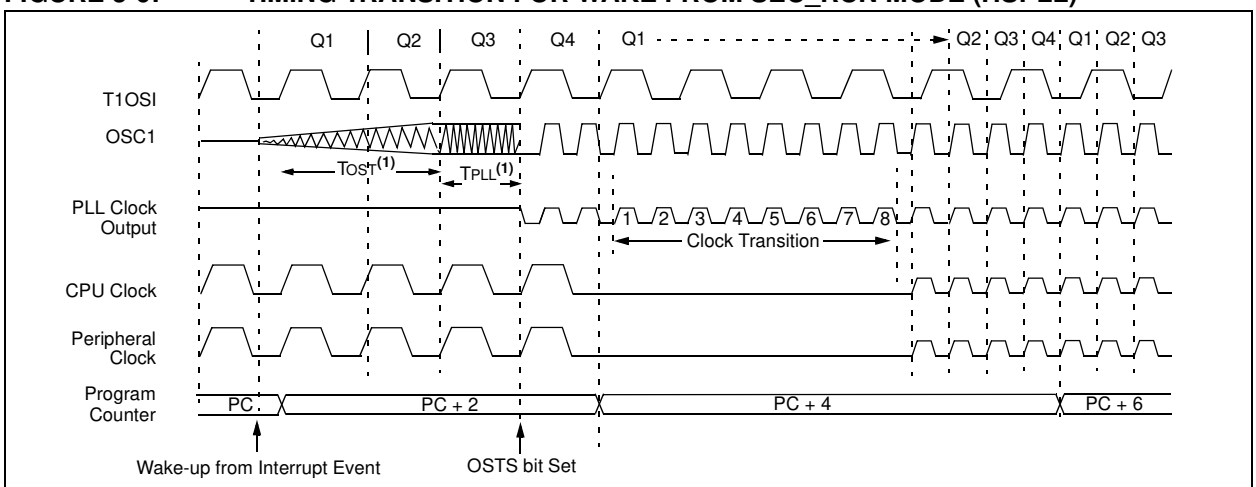



**FIGURE 3-6: TIMING TRANSITION FOR WAKE FROM SEC_RUN MODE (HSPLL)**




**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.


© 2003 Microchip Technology Inc.

### 3.3.3 RC_IDLE MODE

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

This mode is entered by setting the IDLEN bit, setting SCS1 (SCS0 is ignored) and executing a `SLEEP` instruction. The INTOSC multiplexer may be used to select a higher clock frequency by modifying the IRCF bits before executing the `SLEEP` instruction. When the clock source is switched to the INTOSC multiplexer (see Figure 3-7), the primary oscillator is shut down and the OSTS bit is cleared.

If the IRCF bits are set to a non-zero value (thus enabling the INTOSC output), the IOFS bit becomes set after the INTOSC output becomes stable, in about 1 ms. Clocks to the peripherals continue while the INTOSC source stabilizes. If the IRCF bits were previously at a non-zero value before the `SLEEP` instruction was executed and the INTOSC source was already stable, the IOFS bit will remain set. If the IRCF bits are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source.

When a wake-up event occurs, the peripherals continue to be clocked from the INTOSC multiplexer. After a 10 µs delay following the wake-up event, the CPU begins executing code, being clocked by the INTOSC multiplexer. The microcontroller operates in RC_RUN mode until the primary clock becomes ready. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 3-8). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

**FIGURE 3-7: TIMING TRANSITION TO RC_IDLE MODE**

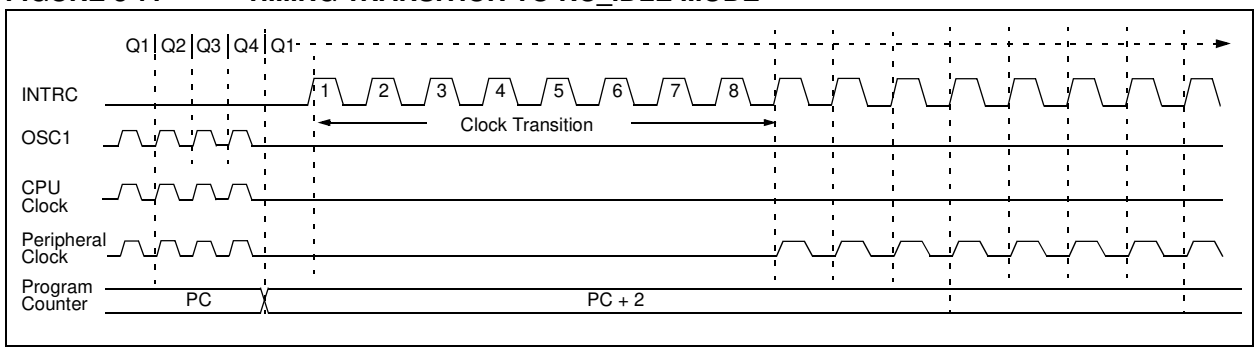
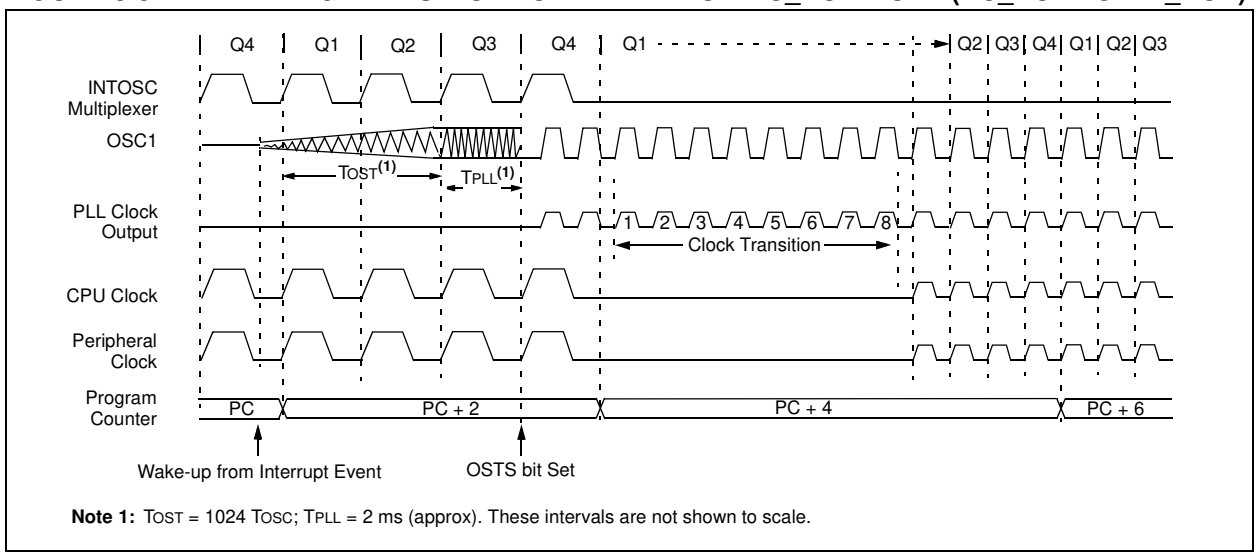



**FIGURE 3-8: TIMING TRANSITION FOR WAKE FROM RC_RUN MODE (RC_RUN TO PRI_RUN)**




**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

## 3.4 Run Modes

If the IDLEN bit is clear when a SLEEP instruction is executed, the CPU and peripherals are both clocked from the source selected using the SCS1:SCS0 bits. While these operating modes may not afford the power conservation of Idle or Sleep modes, they do allow the device to continue executing instructions by using a lower frequency clock source. RC_RUN mode also offers the possibility of executing code at a frequency greater than the primary clock.

Wake-up from a power managed Run mode can be triggered by an interrupt, or any Reset, to return to full power operation. As the CPU is executing code in Run modes, several additional exits from Run modes are possible. They include exit to Sleep mode, exit to a corresponding Idle mode, and exit by executing a RESET instruction. While the device is in any of the power managed Run modes, a WDT time-out will result in a WDT Reset.

### 3.4.1 PRI_RUN MODE

The PRI_RUN mode is the normal full power execution mode. If the SLEEP instruction is never executed, the microcontroller operates in this mode (a SLEEP instruction is executed to enter all other power managed modes). All other power managed modes exit to PRI_RUN mode when an interrupt or WDT time-out occur.

There is no entry to PRI_RUN mode. The OSTS bit is set. The IOFS bit may be set if the internal oscillator block is the primary clock source (see **Section 2.7.1 "Oscillator Control Register"**).

### 3.4.2 SEC_RUN MODE

The SEC_RUN mode is the compatible mode to the "clock switching" feature offered in other PIC18 devices. In this mode, the CPU and peripherals are clocked from the Timer1 oscillator. This gives users the option of lower power consumption while still using a high accuracy clock source.

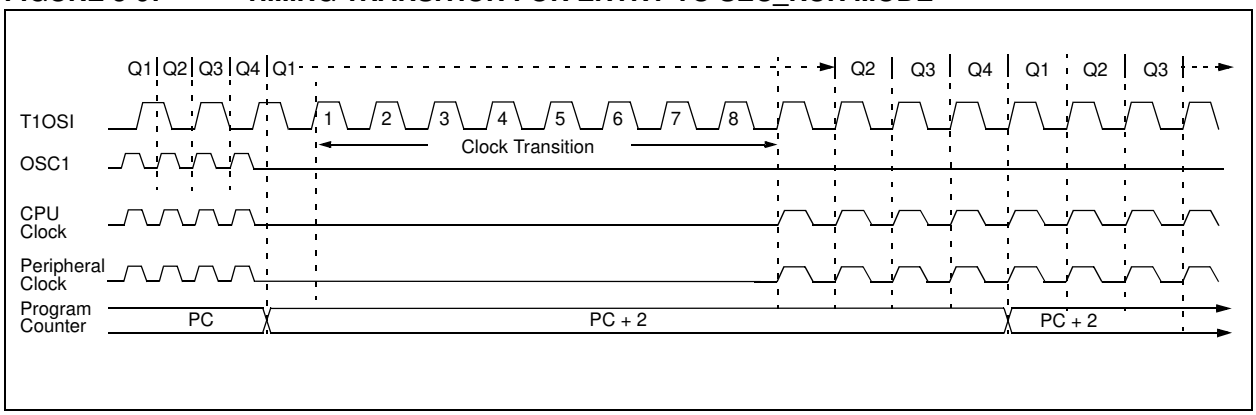
SEC_RUN mode is entered by clearing the IDLEN bit, setting SCS1:SCS0 = 01 and executing a SLEEP instruction. The system clock source is switched to the Timer1 oscillator (see Figure 3-9), the primary oscillator is shut down, the T1RUN bit (T1CON<6>) is set and the OSTS bit is cleared.

| | |
|---|---|
| **Note:** | The Timer1 oscillator should already be running prior to entering SEC_RUN mode. **If the T1OSCEN bit is not set when trying to set the SCS0 bit, the write to SCS0 will not occur.** If the Timer1 oscillator is enabled, but not yet running, system clocks will be delayed until the oscillator has started; in such situations, initial oscillator operation is far from stable and unpredictable operation may result. |

When a wake-up event occurs, the peripherals and CPU continue to be clocked from the Timer1 oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 3-6). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

Firmware can force an exit from SEC_RUN mode. By clearing the T1OSCEN bit (T1CON<3>), an exit from SEC_RUN back to normal full power operation is triggered. The Timer1 oscillator will continue to run and provide the system clock even though the T1OSCEN bit is cleared. The primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 3-6). When the clock switch is complete, the Timer1 oscillator is disabled, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up.

**FIGURE 3-9: TIMING TRANSITION FOR ENTRY TO SEC_RUN MODE**





© 2003 Microchip Technology Inc.

### 3.4.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator block using the INTOSC multiplexer and the primary clock is shut down. When using the INTRC source, this mode provides the best power conservation of all the Run modes while still executing code. It works well for user applications which are not highly timing sensitive or do not require high-speed clocks at all times.

If the primary clock source is the internal oscillator block (either of the INTIO1 or INTIO2 oscillators), there are no distinguishable differences between PRI_RUN and RC_RUN modes during execution. However, a clock switch delay will occur during entry to, and exit from, RC_RUN mode. Therefore, if the primary clock source is the internal oscillator block, the use of RC_RUN mode is not recommended.

This mode is entered by clearing the IDLEN bit, setting SCS1 (SCS0 is ignored) and executing a `SLEEP` instruction. The IRCF bits may select the clock frequency before the `SLEEP` instruction is executed. When the clock source is switched to the INTOSC multiplexer (see Figure 3-10), the primary oscillator is shut down and the OSTS bit is cleared.

The IRCF bits may be modified at any time to immediately change the system clock speed. Executing a `SLEEP` instruction is not required to select a new clock frequency from the INTOSC multiplexer.

| **Note:** | Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/FOSC specifications are violated. |
|---|---|

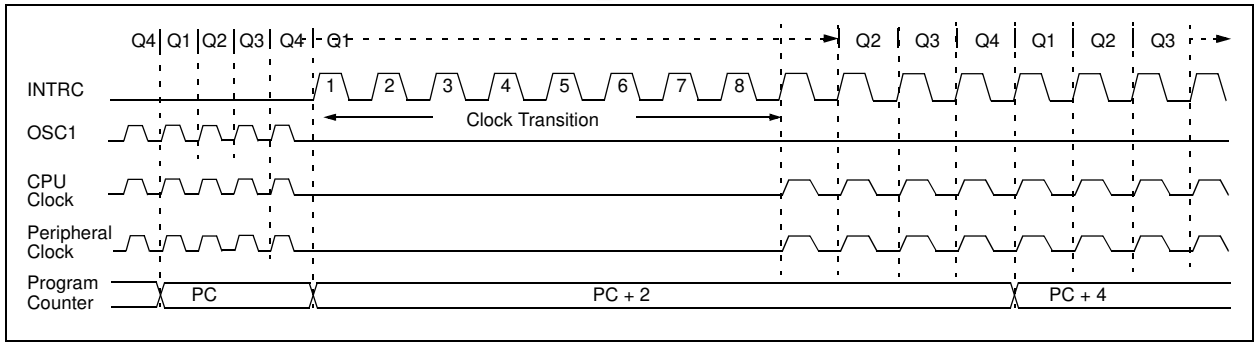
If the IRCF bits are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the system clocks.

If the IRCF bits are changed from all clear (thus enabling the INTOSC output), the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the system continue while the INTOSC source stabilizes in approximately 1 ms.

If the IRCF bits were previously at a non-zero value before the `SLEEP` instruction was executed and the INTOSC source was already stable, the IOFS bit will remain set.

When a wake-up event occurs, the system continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-8). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

**FIGURE 3-10: TIMING TRANSITION TO RC_RUN MODE**



### 3.4.4 EXIT TO IDLE MODE

An exit from a power managed Run mode to its corresponding Idle mode is executed by setting the IDLEN bit and executing a SLEEP instruction. The CPU is halted at the beginning of the instruction following the SLEEP instruction. There are no changes to any of the clock source status bits (OSTS, IOFS or T1RUN). While the CPU is halted, the peripherals continue to be clocked from the previously selected clock source.

### 3.4.5 EXIT TO SLEEP MODE

An exit from a power managed Run mode to Sleep mode is executed by clearing the IDLEN and SCS1:SCS0 bits and executing a SLEEP instruction. The code is no different than the method used to invoke Sleep mode from the normal operating (full power) mode.

The primary clock and internal oscillator block are disabled. The INTRC will continue to operate if the WDT is enabled. The Timer1 oscillator will continue to run, if enabled, in the T1CON register. All clock source status bits are cleared (OSTS, IOFS and T1RUN).

## 3.5 Wake-up From Power Managed Modes

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

| | |
|---|---|
| **Note:** | If application code is timing sensitive, it should wait for the OSTS bit to become set before continuing. Use the interval during the low-power exit sequence (before OSTS is set) to perform timing insensitive “housekeeping” tasks. |

Device behavior during Low-Power mode exits is summarized in Table 3-3.

### 3.5.1 EXIT BY INTERRUPT

Any of the available interrupt sources can cause the device to exit a power managed mode and resume full power operation. To enable this functionality, an interrupt source must be enabled by setting its enable bit in one of the INTCON or PIE registers. The exit sequence is initiated when the corresponding interrupt flag bit is set. On all exits from Lower Power mode by interrupt, code execution branches to the interrupt vector if the GIE/GIEH bit (INTCON<7>) is set. Otherwise, code execution continues or resumes without branching (see **Section 9.0 “Interrupts”**).


© 2003 Microchip Technology Inc.

**TABLE 3-3: ACTIVITY AND EXIT DELAY ON WAKE-UP FROM SLEEP MODE OR ANY IDLE MODE (BY CLOCK SOURCES)**

| Clock in Power Managed Mode | Primary System Clock | Power Managed Mode Exit Delay | Clock Ready Status Bit (OSCCON) | Activity During Wake-up from Power Managed Mode | |
|---|---|---|---|---|---|
| | | | | Exit by Interrupt | Exit by Reset |
| Primary System Clock (PRI_IDLE mode) | LP, XT, HS | 5-10 µs(5) | OSTS | CPU and peripherals clocked by primary clock and executing instructions. | Not clocked or Two-Speed Start-up (if enabled)(3). |
| | HSPLL | | | | |
| | EC, RC, INTRC(1) | | — | | |
| | INTOSC(2) | | IOFS | | |
| T1OSC or INTRC(1) | LP, XT, HS | OST | OSTS | CPU and peripherals clocked by selected power managed mode clock and executing instructions until primary clock source becomes ready. | |
| | HSPLL | OST + 2 ms | | | |
| | EC, RC, INTRC(1) | 5-10 µs(5) | — | | |
| | INTOSC(2) | 1 ms(4) | IOFS | | |
| INTOSC(2) | LP, XT, HS | OST | OSTS | | |
| | HSPLL | OST + 2 ms | | | |
| | EC, RC, INTRC(1) | 5-10 µs(5) | — | | |
| | INTOSC(2) | None | IOFS | | |
| Sleep mode | LP, XT, HS | OST | OSTS | Not clocked or Two-Speed Start-up (if enabled) until primary clock source becomes ready(3). | |
| | HSPLL | OST + 2 ms | | | |
| | EC, RC, INTRC(1) | 5-10 µs(5) | — | | |
| | INTOSC(2) | 1 ms(4) | IOFS | | |

**Note 1:** In this instance, refers specifically to the INTRC clock source.

**2:** Includes both the INTOSC 8 MHz source and postscaler derived frequencies.

**3:** Two-Speed Start-up is covered in greater detail in **Section 23.3 “Two-Speed Start-up”**.

**4:** Execution continues during the INTOSC stabilization period.

**5:** Required delay when waking from Sleep and all Idle modes. This delay runs concurrently with any other required delays (see **Section 3.3 “Idle Modes”**).

### 3.5.2 EXIT BY RESET

Normally, the device is held in Reset by the Oscillator Start-up Timer (OST) until the primary clock (defined in Configuration Register 1H) becomes ready. At that time, the OSTS bit is set and the device begins executing code.

Code execution can begin before the primary clock becomes ready. If either the Two-Speed Start-up (see **Section 23.3 "Two-Speed Start-up"**) or Fail-Safe Clock Monitor (see **Section 23.4 "Fail-Safe Clock Monitor"**) are enabled in Configuration Register 1H, the device may begin execution as soon as the Reset source has cleared. Execution is clocked by the INTOSC multiplexer driven by the internal oscillator block. Since the OSCCON register is cleared following all Resets, the INTRC clock source is selected. A higher speed clock may be selected by modifying the IRCF bits in the OSCCON register. Execution is clocked by the internal oscillator block until either the primary clock becomes ready, or a power managed mode is entered before the primary clock becomes ready; the primary clock is then shut down.

### 3.5.3 EXIT BY WDT TIME-OUT

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

If the device is not executing code (all Idle modes and Sleep mode), the time-out will result in a wake-up from the power managed mode (see **Section 3.2 "Sleep Mode"** through **Section 3.4 "Run Modes"**).

If the device is executing code (all Run modes), the time-out will result in a WDT Reset (see **Section 23.2 "Watchdog Timer (WDT)"**).

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

### 3.5.4 EXIT WITHOUT AN OSCILLATOR START-UP DELAY

Certain exits from power managed modes do not invoke the OST at all. These are:

- PRI_IDLE mode, where the primary clock source is not stopped; and
- the primary clock source is not any of the LP, XT, HS or HSPLL modes.

In these cases, the primary clock source either does not require an oscillator start-up delay, since it is already running (PRI_IDLE), or normally does not require an oscillator start-up delay (RC, EC and INTIO Oscillator modes).

However, a fixed delay (approximately 10 µs) following the wake-up event is required when leaving Sleep and Idle modes. This delay is required for the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

## 3.6 INTOSC Frequency Drift

The factory calibrates the internal oscillator block output (INTOSC) for 8 MHz. However, this frequency may drift as $V_{DD}$ or temperature changes, which can affect the controller operation in a variety of ways.

It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This has the side effect that the INTRC clock source frequency is also affected. However, the features that use the INTRC source often do not require an exact frequency. These features include the Fail-Safe Clock Monitor, the Watchdog Timer and the RC_RUN/RC_IDLE modes when the INTRC clock source is selected.

Being able to adjust the INTOSC requires knowing when an adjustment is required, in which direction it should be made and in some cases, how large a change is needed. Three examples are shown but other techniques may be used.


© 2003 Microchip Technology Inc.

### 3.6.1 EXAMPLE – USART

An adjustment may be indicated when the USART begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the system clock frequency is too high – try decrementing the value in the OSCTUNE register to reduce the system clock frequency. Errors in data may suggest that the system clock speed is too low – increment OSCTUNE.

### 3.6.2 EXAMPLE – TIMERS

This technique compares system clock speed to some reference clock. Two timers may be used; one timer is clocked by the peripheral clock, while the other is clocked by a fixed reference source, such as the Timer1 oscillator.

Both timers are cleared but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is greater than expected, then the internal oscillator block is running too fast – decrement OSCTUNE.

### 3.6.3 EXAMPLE – CCP IN CAPTURE MODE

A CCP module can use free running Timer1 (or Timer3), clocked by the internal oscillator block and an external event with a known period (i.e., AC power frequency). The time of the first event is captured in the CCPRxH:CCPRxL registers and is recorded for use later. When the second event causes a capture, the time of the first event is subtracted from the time of the second event. Since the period of the external event is known, the time difference between events can be calculated.

If the measured time is much greater than the calculated time, the internal oscillator block is running too fast – decrement OSCTUNE. If the measured time is much less than the calculated time, the internal oscillator block is running too slow – increment OSCTUNE.

**NOTES:**


© 2003 Microchip Technology Inc.

## 4.0 RESET

The PIC18F2X20/4X20 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset while executing instructions
c) $\overline{\text{MCLR}}$ Reset when not executing instructions
d) Watchdog Timer (WDT) Reset (during execution)
e) Programmable Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

Most registers are unaffected by a Reset. Their status is unknown on POR and unchanged by all other Resets. The other registers are forced to a "Reset state" depending on the type of Reset that occurred.

Most registers are not affected by a WDT wake-up since this is viewed as the resumption of normal operation. Status bits from the RCON register, $\overline{\text{RI}}$, $\overline{\text{TO}}$, $\overline{\text{PD}}$, $\overline{\text{POR}}$ and $\overline{\text{BOR}}$, are set or cleared differently in different Reset situations as indicated in Table 4-2. These bits are used in software to determine the nature of the Reset. See Table 4-3 for a full description of the Reset states of all registers.

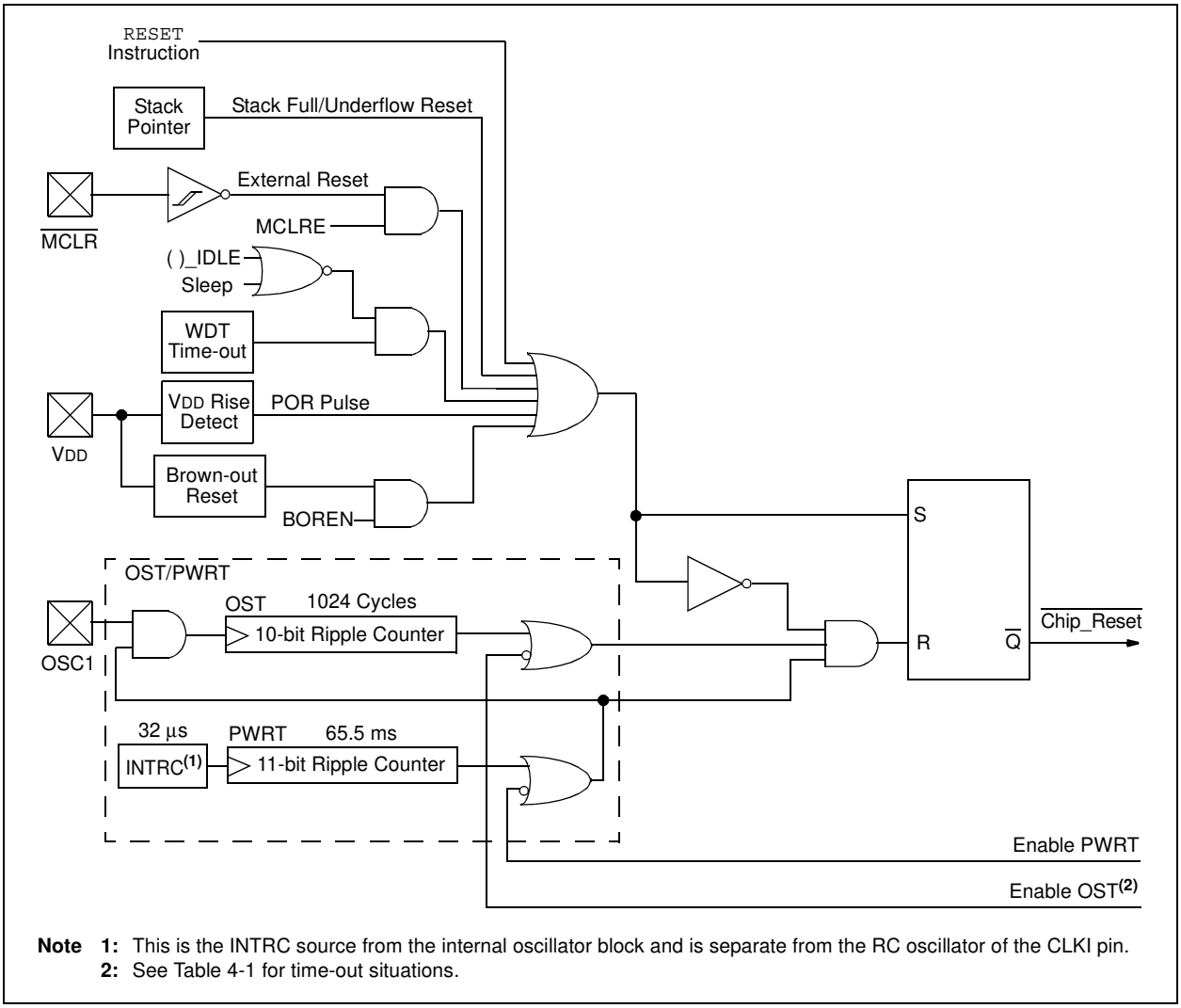
A simplified block diagram of the on-chip Reset circuit is shown in Figure 4-1.

The enhanced MCU devices have a $\overline{\text{MCLR}}$ noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

The $\overline{\text{MCLR}}$ pin is not driven low by any internal Resets, including the WDT.

The $\overline{\text{MCLR}}$ input provided by the $\overline{\text{MCLR}}$ pin can be disabled with the MCLRE bit in Configuration Register 3H (CONFIG3H<7>). See **Section 23.1 "Configuration Bits"** for more information.

**FIGURE 4-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



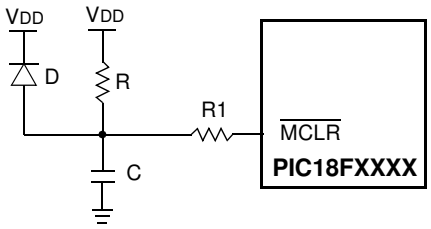

**Note 1:** This is the INTRC source from the internal oscillator block and is separate from the RC oscillator of the CLKI pin.
**2:** See Table 4-1 for time-out situations.

## 4.1 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected. To take advantage of the POR circuitry, just tie the MCLR pin through a resistor (1k to 10 kΩ) to VDD. This will eliminate external RC components usually needed to create a Power-on Reset delay. A minimum rise rate for VDD is specified (parameter D004). For a slow rise time, see Figure 4-2.

When the device starts normal operation (i.e., exits the Reset condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not met, the device must be held in Reset until the operating conditions are met.

**FIGURE 4-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)**




**Note 1:** External Power-on Reset circuit is required only if the VDD power-up slope is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

**2:** R < 40 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

**3:** R1 ≥ 1 kΩ will limit any current flowing into MCLR from external capacitor C, in the event of MCLR/VPP pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

## 4.2 Power-up Timer (PWRT)

The Power-up Timer (PWRT) of the PIC18F2X20/4X20 devices is an 11-bit counter, which uses the INTRC source as the clock input. This yields a count of 2048 x 32 µs = 65.6 ms. While the PWRT is counting, the device is held in Reset.

The power-up time delay depends on the INTRC clock and will vary from chip-to-chip due to temperature and process variation. See DC parameter #33 for details.

The PWRT is enabled by clearing configuration bit, PWRTEN.

## 4.3 Oscillator Start-up Timer (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (parameter #33). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP, HS and HSPLL modes and only on Power-on Reset, or on exit from most power managed modes.

## 4.4 PLL Lock Time-out

With the PLL enabled in its PLL mode, the time-out sequence following a Power-on Reset is slightly different from other oscillator modes. A portion of the Power-up Timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL lock time-out (TPLL) is typically 2 ms and follows the oscillator start-up time-out.

## 4.5 Brown-out Reset (BOR)

A configuration bit, BOREN, can disable (if clear/programmed) or enable (if set) the Brown-out Reset circuitry. If VDD falls below VBOR (parameter D005) for greater than TBOR (parameter #35), the brown-out situation will reset the chip. A Reset may not occur if VDD falls below VBOR for less than TBOR. The chip will remain in Brown-out Reset until VDD rises above VBOR. If the Power-up Timer is enabled, it will be invoked after VDD rises above VBOR; it then will keep the chip in Reset for an additional time delay TPWRT (parameter #33). If VDD drops below VBOR while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be initialized. Once VDD rises above VBOR, the Power-up Timer will execute the additional time delay. Enabling BOR Reset does not automatically enable the PWRT.

## 4.6 Time-out Sequence

On power-up, the time-out sequence is as follows: First, after the POR pulse has cleared, PWRT time-out is invoked (if enabled). Then, the OST is activated. The total time-out will vary based on oscillator configuration and the status of the PWRT. For example, in RC mode with the PWRT disabled, there will be no time-out at all. Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7 depict time-out sequences on power-up.

Since the time-outs occur from the POR pulse, if MCLR is kept low long enough, all time-outs will expire. Bringing MCLR high will begin execution immediately (Figure 4-5). This is useful for testing purposes or to synchronize more than one PIC18FXXXX device operating in parallel.

Table 4-2 shows the Reset conditions for some Special Function Registers, while Table 4-3 shows the Reset conditions for all the registers.


© 2003 Microchip Technology Inc.

**TABLE 4-1: TIME-OUT IN VARIOUS SITUATIONS**

| Oscillator Configuration | Power-up[2] and Brown-out | | Exit from Power Managed Mode |
|---|---|---|---|
| | $\overline{\text{PWRTEN}}$ = 0 | $\overline{\text{PWRTEN}}$ = 1 | |
| HSPLL | 66 ms[1] + 1024 TOSC + 2 ms[2] | 1024 TOSC + 2 ms[2] | 1024 TOSC + 2 ms[2] |
| HS, XT, LP | 66 ms[1] + 1024 TOSC | 1024 TOSC | 1024 TOSC |
| EC, ECIO | 66 ms[1] | — | — |
| RC, RCIO | 66 ms[1] | — | — |
| INTIO1, INTIO2 | 66 ms[1] | — | — |

**Note 1:** 66 ms (65.5 ms) is the nominal Power-up Timer (PWRT) delay.

**2:** 2 ms is the nominal time required for the 4x PLL to lock.

**REGISTER 4-1: RCON REGISTER BITS AND POSITIONS**

| R/W-0 | U-0 | U-0 | R/W-1 | R-1 | R-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| IPEN | — | — | $\overline{\text{RI}}$ | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ |
| bit 7 | | | | | | | bit 0 |

**Note:** Refer to **Section 5.14 "RCON Register"** for bit definitions.

**TABLE 4-2: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER**

| Condition | Program Counter | RCON Register | $\overline{\text{RI}}$ | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ | STKFUL | STKUNF |
|---|---|---|---|---|---|---|---|---|---|
| Power-on Reset | 0000h | 0--1 1100 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| RESET Instruction | 0000h | 0--0 uuuu | 0 | u | u | u | u | u | u |
| Brown-out | 0000h | 0--1 11u- | 1 | 1 | 1 | u | 0 | u | u |
| $\overline{\text{MCLR}}$ during power managed Run modes | 0000h | 0--u 1uuu | u | 1 | u | u | u | u | u |
| $\overline{\text{MCLR}}$ during power managed Idle modes and Sleep mode | 0000h | 0--u 10uu | u | 1 | 0 | u | u | u | u |
| WDT Time-out during full power or power managed Run mode | 0000h | 0--u 0uuu | u | 0 | u | u | u | u | u |
| $\overline{\text{MCLR}}$ during full power execution | 0000h | 0--u uuuu | u | u | u | u | u | u | u |
| Stack Full Reset (STVREN = 1) | | | | | | | | 1 | u |
| Stack Underflow Reset (STVREN = 1) | | | | | | | | u | 1 |
| Stack Underflow Error (not an actual Reset, STVREN = 0) | 0000h | u--u uuuu | u | u | u | u | u | u | 1 |
| WDT Time-out during power managed Idle or Sleep modes | PC + 2 | u--u 00uu | u | 0 | 0 | u | u | u | u |
| Interrupt exit from power managed modes | PC + 2 | u--u u0uu | u | u | 0 | u | u | u | u |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0'

**Note 1:** When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (0x000008h or 0x000018h).

**TABLE 4-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| TOSU | 2220 | 2320 | 4220 | 4320 | ---0 0000 | ---0 0000 | ---0 uuuu(3) |
| TOSH | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu(3) |
| TOSL | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu(3) |
| STKPTR | 2220 | 2320 | 4220 | 4320 | uu-0 0000 | 00-0 0000 | uu-u uuuu(3) |
| PCLATU | 2220 | 2320 | 4220 | 4320 | ---0 0000 | ---0 0000 | ---u uuuu |
| PCLATH | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PCL | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | PC + 2(2) |
| TBLPTRU | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| TBLPTRH | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TBLPTRL | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TABLAT | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PRODH | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PRODL | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INTCON | 2220 | 2320 | 4220 | 4320 | 0000 000x | 0000 000u | uuuu uuuu(1) |
| INTCON2 | 2220 | 2320 | 4220 | 4320 | 1111 -1-1 | 1111 -1-1 | uuuu -u-u(1) |
| INTCON3 | 2220 | 2320 | 4220 | 4320 | 11-0 0-00 | 11-0 0-00 | uu-u u-uu(1) |
| INDF0 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTINC0 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTDEC0 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PREINC0 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PLUSW0 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| FSR0H | 2220 | 2320 | 4220 | 4320 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR0L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| WREG | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF1 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTINC1 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTDEC1 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PREINC1 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PLUSW1 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-2 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.


© 2003 Microchip Technology Inc.

**TABLE 4-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| FSR1H | 2220 | 2320 | 4220 | 4320 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR1L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| BSR | 2220 | 2320 | 4220 | 4320 | ---- 0000 | ---- 0000 | ---- uuuu |
| INDF2 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTINC2 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| POSTDEC2 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PREINC2 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| PLUSW2 | 2220 | 2320 | 4220 | 4320 | N/A | N/A | N/A |
| FSR2H | 2220 | 2320 | 4220 | 4320 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR2L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| STATUS | 2220 | 2320 | 4220 | 4320 | ---x xxxx | ---u uuuu | ---u uuuu |
| TMR0H | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TMR0L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T0CON | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | uuuu uuuu |
| OSCCON | 2220 | 2320 | 4220 | 4320 | 0000 q000 | 0000 q000 | uuuu qquu |
| LVDCON | 2220 | 2320 | 4220 | 4320 | --00 0101 | --00 0101 | --uu uuuu |
| WDTCON | 2220 | 2320 | 4220 | 4320 | ---- ---0 | ---- ---0 | ---- ---u |
| RCON(4) | 2220 | 2320 | 4220 | 4320 | 0--1 11q0 | 0--q qquu | u--u qquu |
| TMR1H | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR1L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T1CON | 2220 | 2320 | 4220 | 4320 | 0000 0000 | u0uu uuuu | uuuu uuuu |
| TMR2 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PR2 | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | 1111 1111 |
| T2CON | 2220 | 2320 | 4220 | 4320 | -000 0000 | -000 0000 | -uuu uuuu |
| SSPBUF | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| SSPADD | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSPSTAT | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSPCON1 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSPCON2 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-2 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

**TABLE 4-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| ADRESH | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADRESL | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADCON0 | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| ADCON1 | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| ADCON2 | 2220 | 2320 | 4220 | 4320 | 0-00 0000 | 0-00 0000 | u-uu uuuu |
| CCPR1H | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR1L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP1CON | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| CCPR2H | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR2L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP2CON | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| PWM1CON | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| ECCPAS | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CVRCON | 2220 | 2320 | 4220 | 4320 | 000- 0000 | 000- 0000 | uuu- uuuu |
| CMCON | 2220 | 2320 | 4220 | 4320 | 0000 0111 | 0000 0111 | uuuu uuuu |
| TMR3H | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR3L | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T3CON | 2220 | 2320 | 4220 | 4320 | 0000 0000 | uuuu uuuu | uuuu uuuu |
| SPBRG | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| RCREG | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TXREG | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TXSTA | 2220 | 2320 | 4220 | 4320 | 0000 -010 | 0000 -010 | uuuu -uuu |
| RCSTA | 2220 | 2320 | 4220 | 4320 | 0000 000x | 0000 000x | uuuu uuuu |
| EEADR | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| EEDATA | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| EECON1 | 2220 | 2320 | 4220 | 4320 | xx-0 x000 | uu-0 u000 | uu-0 u000 |
| EECON2 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | 0000 0000 |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-2 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.


© 2003 Microchip Technology Inc.

**TABLE 4-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| IPR2 | 2220 | 2320 | 4220 | 4320 | 11-1 1111 | 11-1 1111 | uu-u uuuu |
| PIR2 | 2220 | 2320 | 4220 | 4320 | 00-0 0000 | 00-0 0000 | uu-u uuuu(1) |
| PIE2 | 2220 | 2320 | 4220 | 4320 | 00-0 0000 | 00-0 0000 | uu-u uuuu |
| IPR1 | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | uuuu uuuu |
| | 2220 | 2320 | 4220 | 4320 | -111 1111 | -111 1111 | -uuu uuuu |
| PIR1 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu(1) |
| | 2220 | 2320 | 4220 | 4320 | -000 0000 | -000 0000 | -uuu uuuu(1) |
| PIE1 | 2220 | 2320 | 4220 | 4320 | 0000 0000 | 0000 0000 | uuuu uuuu |
| | 2220 | 2320 | 4220 | 4320 | -000 0000 | -000 0000 | -uuu uuuu |
| OSCTUNE | 2220 | 2320 | 4220 | 4320 | --00 0000 | --00 0000 | --uu uuuu |
| TRISE | 2220 | 2320 | 4220 | 4320 | 0000 -111 | 0000 -111 | uuuu -uuu |
| TRISD | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISC | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISB | 2220 | 2320 | 4220 | 4320 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA(5) | 2220 | 2320 | 4220 | 4320 | 1111 1111(5) | 1111 1111(5) | uuuu uuuu(5) |
| LATE | 2220 | 2320 | 4220 | 4320 | ---- -xxx | ---- -uuu | ---- -uuu |
| LATD | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATC | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATB | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATA(5) | 2220 | 2320 | 4220 | 4320 | xxxx xxxx(5) | uuuu uuuu(5) | uuuu uuuu(5) |
| PORTE | 2220 | 2320 | 4220 | 4320 | ---- xxxx | ---- xxxx | ---- uuuu |
| PORTD | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTC | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTB | 2220 | 2320 | 4220 | 4320 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTA(5) | 2220 | 2320 | 4220 | 4320 | xx0x 0000(5) | uu0u 0000(5) | uuuu uuuu(5) |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-2 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

**FIGURE 4-3: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD, VDD RISE < TPWRT)**

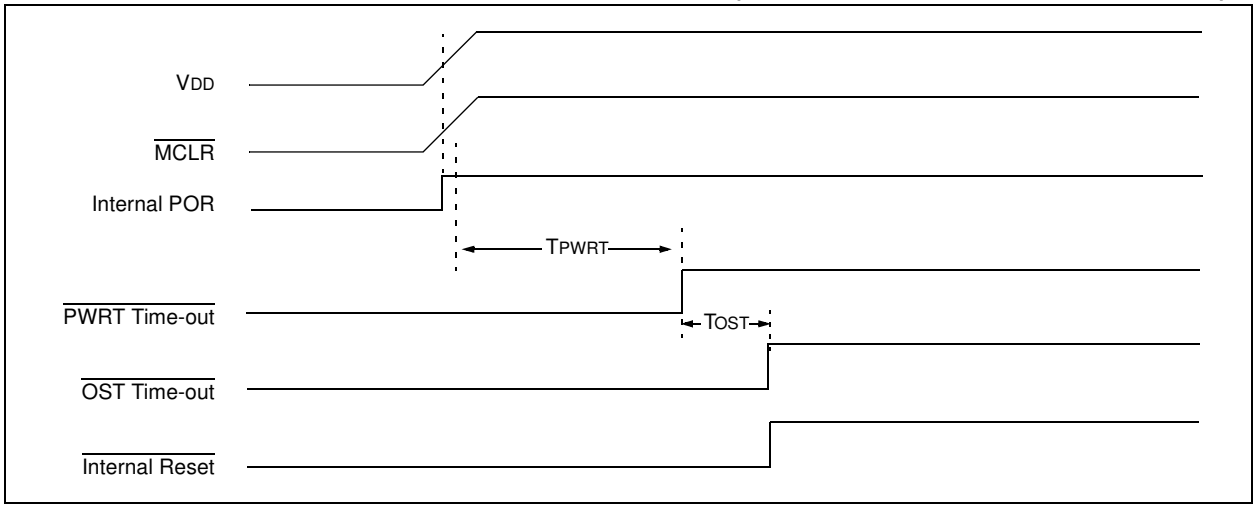



**FIGURE 4-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1**

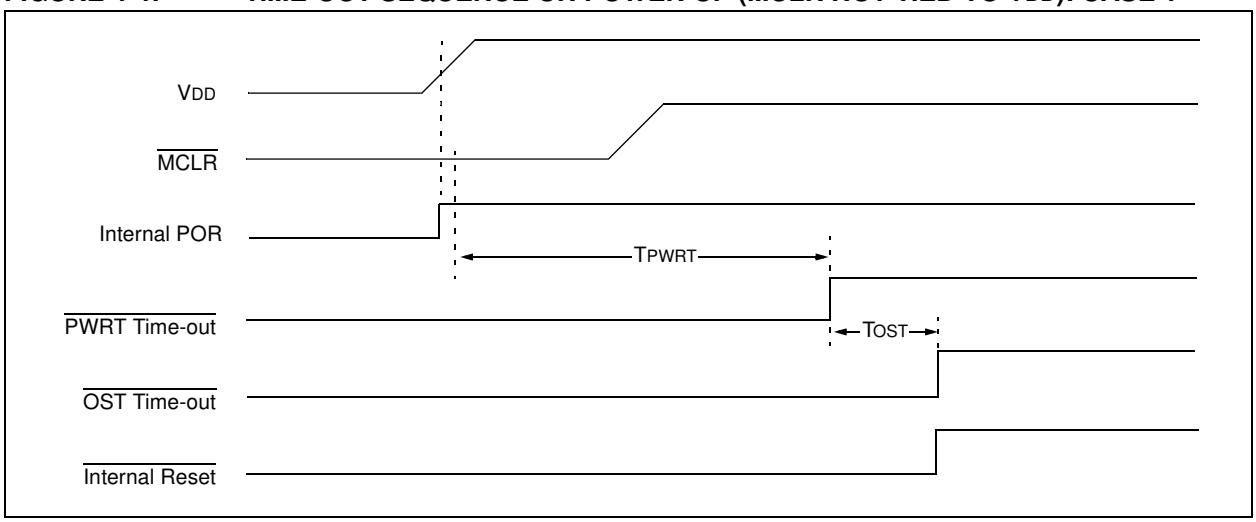
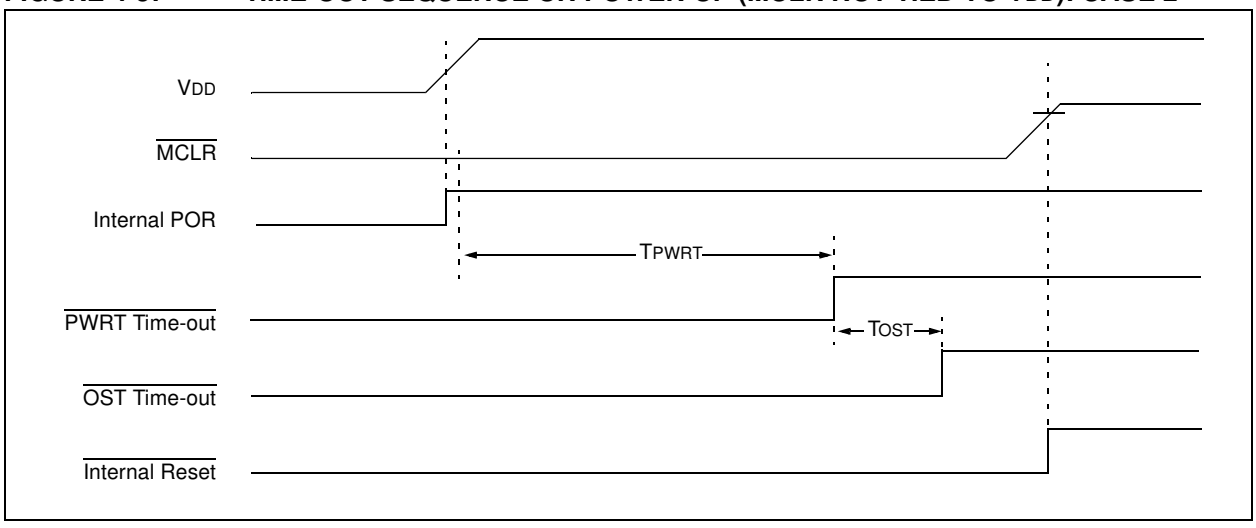



**FIGURE 4-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2**





© 2003 Microchip Technology Inc.

**FIGURE 4-6: SLOW RISE TIME ($\overline{MCLR}$ TIED TO $V_{DD}$, $V_{DD}$ RISE > $T_{PWRT}$)**

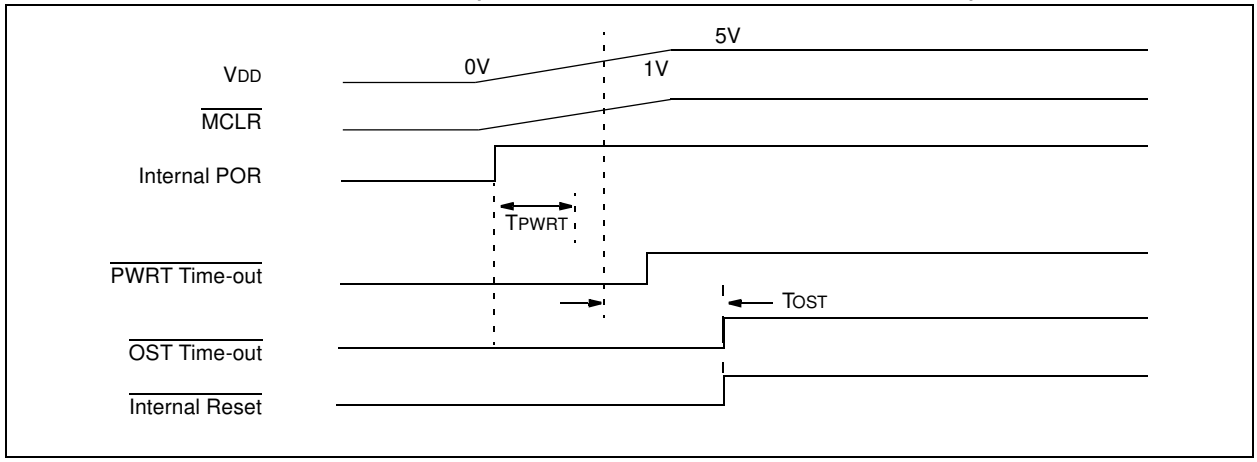
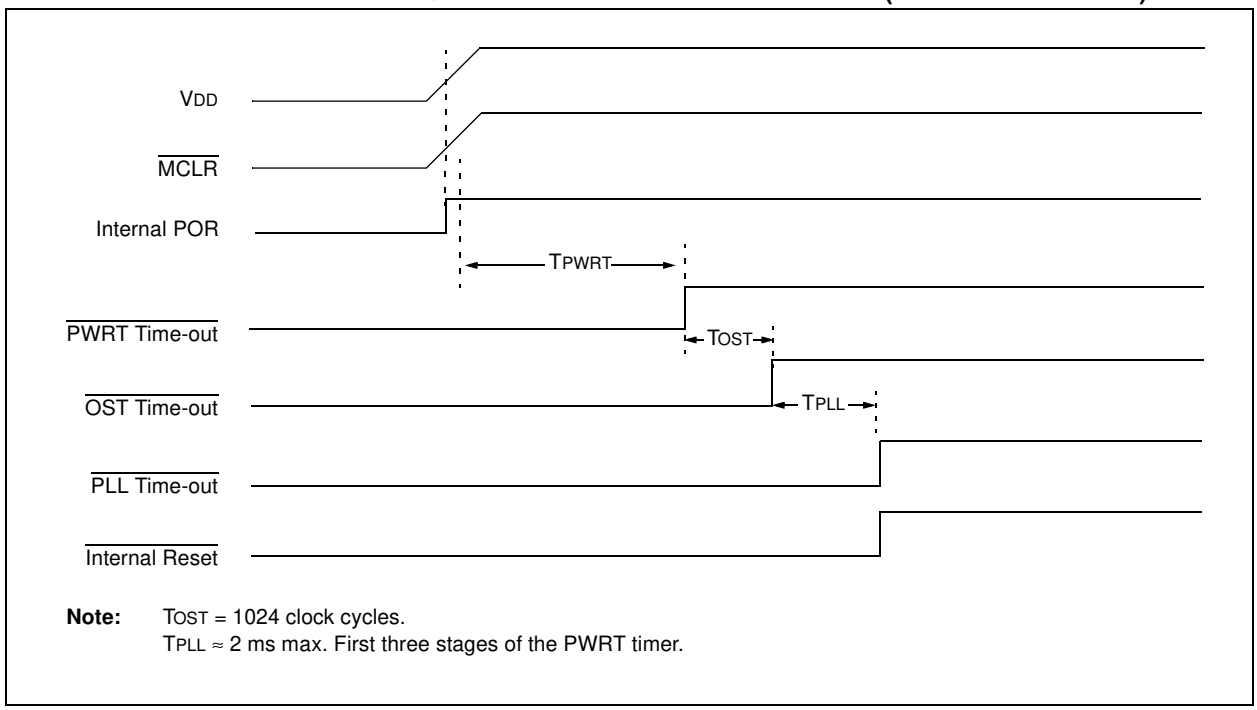



**FIGURE 4-7: TIME-OUT SEQUENCE ON POR W/ PLL ENABLED ($\overline{MCLR}$ TIED TO $V_{DD}$)**




**Note:** $T_{OST}$ = 1024 clock cycles.
$T_{PLL} \approx$ 2 ms max. First three stages of the PWRT timer.

**NOTES:**


© 2003 Microchip Technology Inc.

## 5.0 MEMORY ORGANIZATION

There are three memory types in Enhanced MCU devices. These memory types are:

- Program Memory
- Data RAM
- Data EEPROM

Data and program memory use separate busses which allow for concurrent access of these types.

Additional detailed information for Flash program memory and data EEPROM is provided in **Section 6.0 "Flash Program Memory"** and **Section 7.0 "Data EEPROM Memory"**, respectively.

### 5.1 Program Memory Organization

A 21-bit program counter is capable of addressing the 2-Mbyte program memory space. Accessing a location between the physically implemented memory and the 2-Mbyte address will cause a read of all '0's (a NOP instruction).

The PIC18F2220 and PIC18F4220 each have 4 Kbytes of Flash memory and can store up to 2,048 single-word instructions.

The PIC18F2320 and PIC18F4320 each have 8 Kbytes of Flash memory and can store up to 4,096 single-word instructions.

The Reset vector address is at 0000h and the interrupt vector addresses are at 0008h and 0018h.

The Program Memory Maps for PIC18F2220/4220 and PIC18F2320/4320 devices are shown in Figure 5-1 and Figure 5-2, respectively.

**FIGURE 5-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F2220/4220**

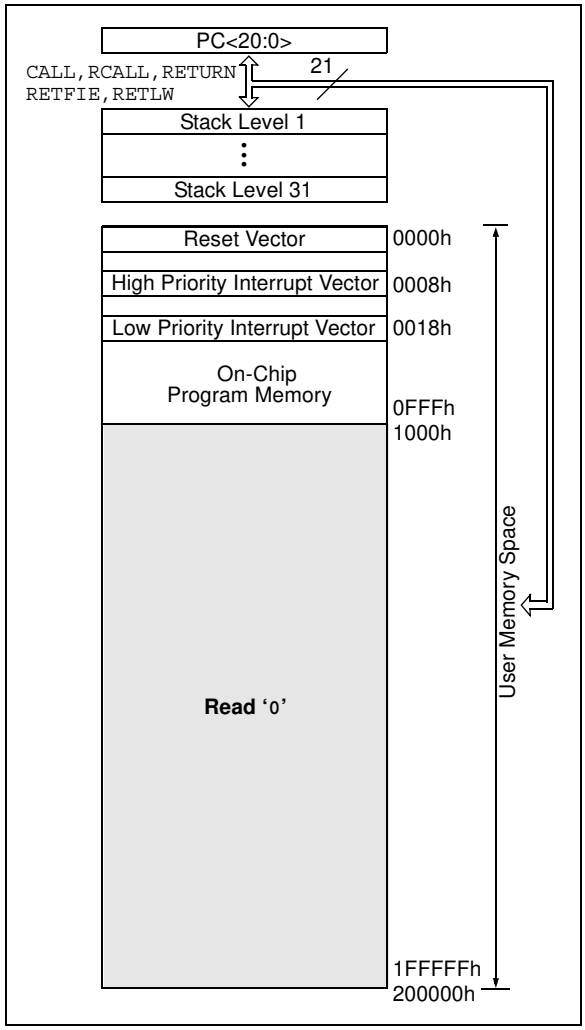
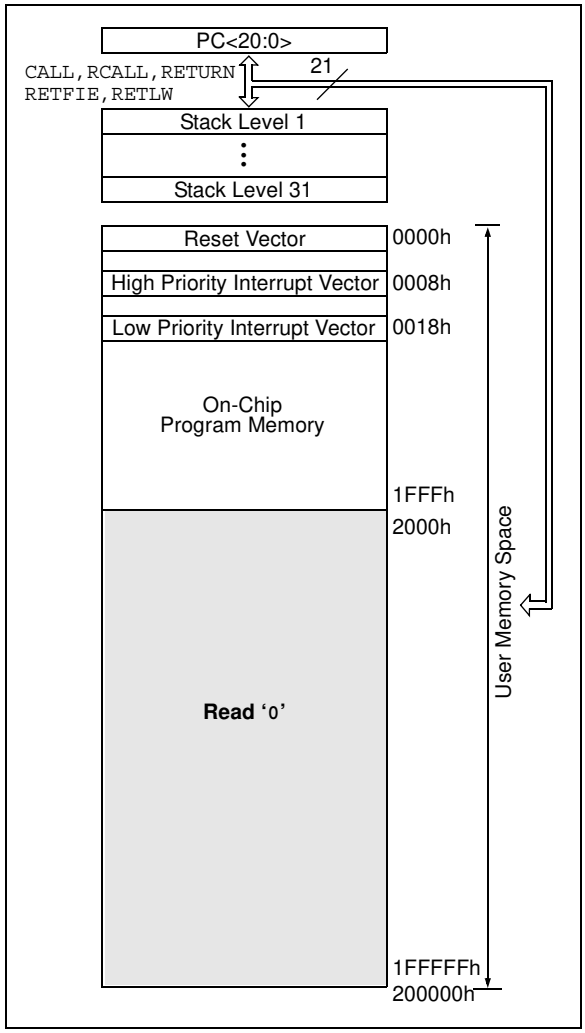



**FIGURE 5-2: PROGRAM MEMORY MAP AND STACK FOR PIC18F2320/4320**



## 5.2 Return Address Stack

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC (Program Counter) is pushed onto the stack when a CALL or RCALL instruction is executed or an interrupt is Acknowledged. The PC value is pulled off the stack on a RETURN, RETLW or a RETFIE instruction. PCLATU and PCLATH are not affected by any of the RETURN or CALL instructions.

The stack operates as a 31-word by 21-bit RAM and a 5-bit stack pointer, with the stack pointer initialized to 00000b after all Resets. There is no RAM associated with stack pointer 00000b. This is only a Reset value. During a CALL type instruction, causing a push onto the stack, the stack pointer is first incremented and the RAM location pointed to by the stack pointer is written with the contents of the PC (already pointing to the instruction following the CALL). During a RETURN type instruction, causing a pop from the stack, the contents of the RAM location pointed to by the STKPTR are transferred to the PC and then the stack pointer is decremented.

The stack space is not part of either program or data space. The stack pointer is readable and writable and the address on the top of the stack is readable and writable through the top-of-stack Special File Registers. Data can also be pushed to, or popped from, the stack using the top-of-stack SFRs. Status bits indicate if the stack is full, has overflowed or underflowed.

### 5.2.1 TOP-OF-STACK ACCESS

The top of the stack is readable and writable. Three register locations, TOSU, TOSH and TOSL, hold the contents of the stack location pointed to by the STKPTR register (Figure 5-3). This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt, the software can read the pushed value by reading the TOSU, TOSH and TOSL registers. These values can be placed on a user defined software stack. At return time, the software can replace the TOSU, TOSH and TOSL and do a return.

The user must disable the global interrupt enable bits while accessing the stack to prevent inadvertent stack corruption.

### 5.2.2 RETURN STACK POINTER (STKPTR)

The STKPTR register (Register 5-1) contains the stack pointer value, the STKFUL (Stack Full) status bit and the STKUNF (Stack Underflow) status bits. The value of the stack pointer can be 0 through 31. The stack pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. At Reset, the stack pointer value will be zero. The user may read and write the stack pointer value. This feature can be used by a Real-Time Operating System for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a POR.

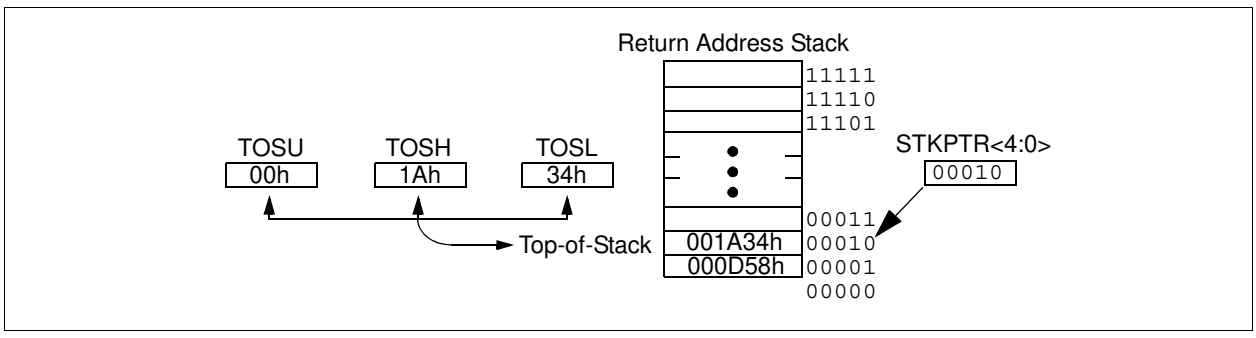
The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) configuration bit. (Refer to **Section 23.1 "Configuration Bits"** for a description of the device configuration bits.) If STVREN is set (default), the 31st push will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the stack pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the stack pointer will increment to 31. Any additional pushes will not overwrite the 31st push, and STKPTR will remain at 31.

When the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC and sets the STKUNF bit, while the stack pointer remains at zero. The STKUNF bit will remain set until cleared by software or a POR occurs.

| **Note:** | Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset, as the contents of the SFRs are not affected. |
|---|---|

**FIGURE 5-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS**





© 2003 Microchip Technology Inc.

**REGISTER 5-1: STKPTR REGISTER**

| R/C-0 | R/C-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| STKFUL | STKUNF | — | SP4 | SP3 | SP2 | SP1 | SP0 |
| bit 7 | | | | | | | bit 0 |

bit 7[(1)] **STKFUL:** Stack Full Flag bit
1 = Stack became full or overflowed
0 = Stack has not become full or overflowed

bit 6[(1)] **STKUNF:** Stack Underflow Flag bit
1 = Stack underflow occurred
0 = Stack underflow did not occur

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

bit 4-0 **SP4:SP0:** Stack Pointer Location bits

**Note 1:** Bit 7 and bit 6 are cleared by user software or by a POR.

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

### 5.2.3 PUSH AND POP INSTRUCTIONS

Since the Top-of-Stack (TOS) is readable and writable, the ability to push values onto the stack and pull values off the stack, without disturbing normal program execution, is a desirable option. To push the current PC value onto the stack, a PUSH instruction can be executed. This will increment the stack pointer and load the current PC value onto the stack. TOSU, TOSH and TOSL can then be modified to place data or a return address on the stack.

The ability to pull the TOS value off of the stack and replace it with the value that was previously pushed onto the stack, without disturbing normal execution, is achieved by using the POP instruction. The POP instruction discards the current TOS by decrementing the stack pointer. The previous value pushed onto the stack then becomes the TOS value.

### 5.2.4 STACK FULL/UNDERFLOW RESETS

These Resets are enabled by programming the STVREN bit in Configuration Register 4L. When the STVREN bit is cleared, a full or underflow condition will set the appropriate STKFUL or STKUNF bit but not cause a device Reset. When the STVREN bit is set, a full or underflow condition will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. The STKFUL or STKUNF bits are cleared by the user software or a POR Reset.

## 5.3 Fast Register Stack

A “fast return” option is available for interrupts. A Fast Register Stack is provided for the Status, WREG and BSR registers and are only one in depth. The stack is not readable or writable and is loaded with the current value of the corresponding register when the processor vectors for an interrupt. The values in the registers are then loaded back into the working registers if the RETFIE, FAST instruction is used to return from the interrupt.

All interrupt sources will push values into the stack registers. If both low and high priority interrupts are enabled, the stack registers cannot be used reliably to return from low priority interrupts. If a high priority interrupt occurs while servicing a low priority interrupt, the stack register values stored by the low priority interrupt will be overwritten. Users must save the key registers in software during a low priority interrupt.

If interrupt priority is not used, all interrupts may use the Fast Register Stack for returns from interrupt.

If no interrupts are used, the Fast Register Stack can be used to restore the Status, WREG and BSR registers at the end of a subroutine call. To use the Fast Register Stack for a subroutine call, a CALL label, FAST instruction must be executed to save the Status, WREG and BSR registers to the Fast Register Stack. A RETURN, FAST instruction is then executed to restore these registers from the Fast Register Stack.

Example 5-1 shows a source code example that uses the Fast Register Stack during a subroutine call and return.

**EXAMPLE 5-1: FAST REGISTER STACK CODE EXAMPLE**

```
CALL SUB1, FAST    ;STATUS, WREG, BSR
                   ;SAVED IN FAST REGISTER
                   ;STACK
       •
       •

SUB1   •
       •
   RETURN FAST     ;RESTORE VALUES SAVED
                   ;IN FAST REGISTER STACK
```

## 5.4 PCL, PCLATH and PCLATU

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21-bits wide. The low byte, known as the PCL register, is both readable and writable. The high byte, or PCH register, contains the PC<15:8> bits and is not directly readable or writable. Updates to the PCH register may be performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits and is not directly readable or writable. Updates to the PCU register may be performed through the PCLATU register.

The contents of PCLATH and PCLATU will be transferred to the program counter by any operation that writes PCL. Similarly, the upper two bytes of the program counter will be transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see **Section 5.8.1 “Computed GOTO”**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the LSB of PCL is fixed to a value of ‘0’. The PC increments by 2 to address sequential instructions in the program memory.

The CALL, RCALL, GOTO and program branch instructions write to the program counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the program counter.


© 2003 Microchip Technology Inc.

## 5.5 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the Program Counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 5-4.

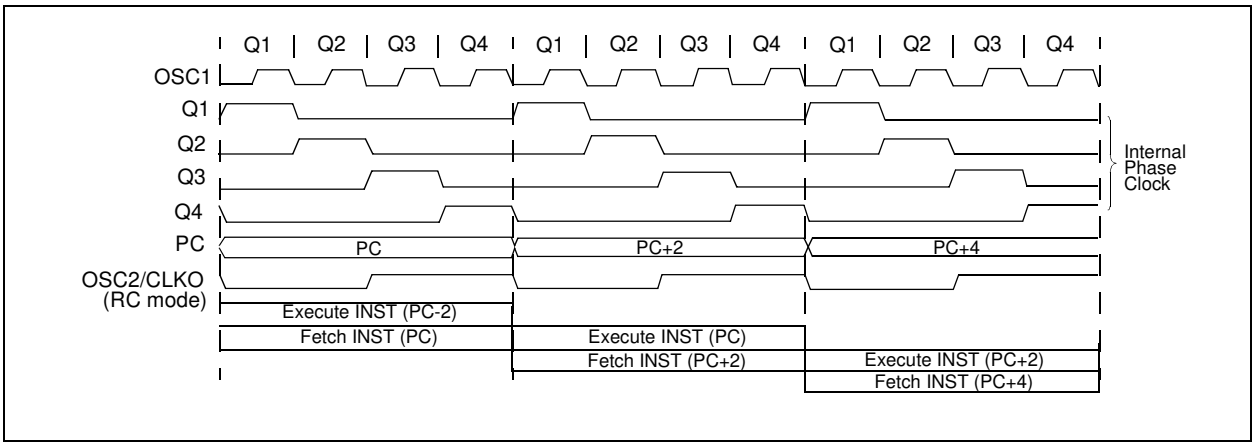
## 5.6 Instruction Flow/Pipelining

An “Instruction Cycle” consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle, while decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., `GOTO`), then two cycles are required to complete the instruction (Example 5-2).

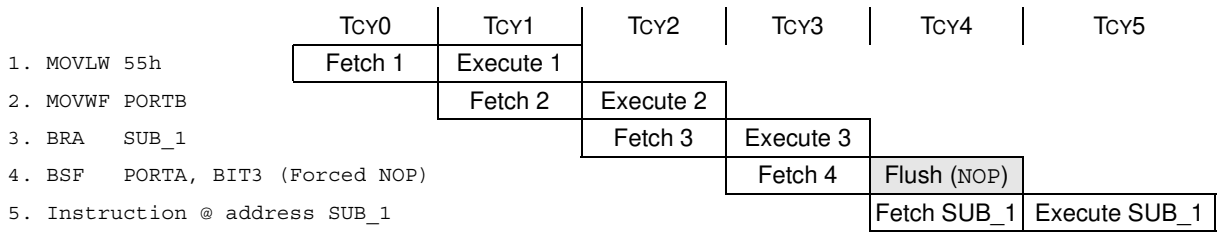
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the “Instruction Register” (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3 and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

**FIGURE 5-4: CLOCK/INSTRUCTION CYCLE**




**EXAMPLE 5-2: INSTRUCTION PIPELINE FLOW**



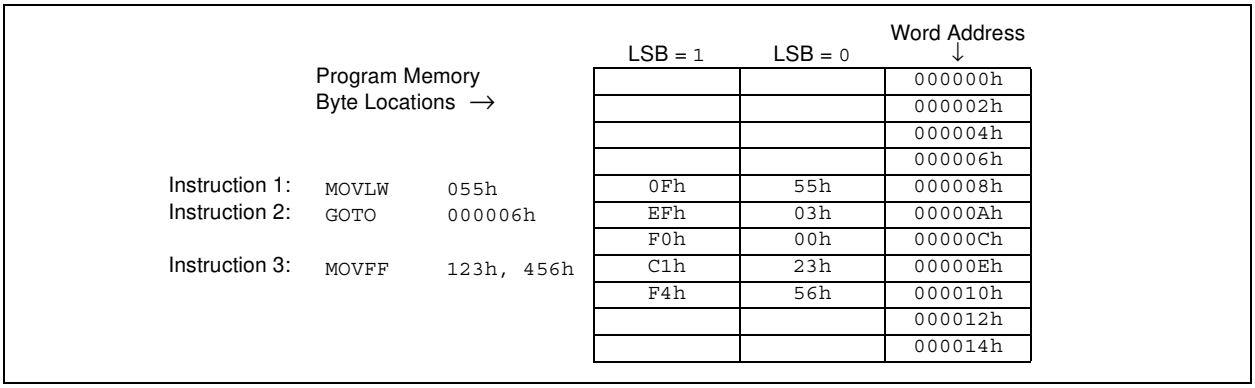

All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is “flushed” from the pipeline while the new instruction is being fetched and then executed.

## 5.7 Instructions in Program Memory

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). Figure 5-5 shows an example of how instruction words are stored in the program memory. To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '0' (see **Section 5.4 "PCL, PCLATH and PCLATU"**).

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction #2 in Figure 5-5 shows how the instruction 'GOTO 000006h' is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 24.0 "Instruction Set Summary"** provides further details of the instruction set.

**FIGURE 5-5: INSTRUCTIONS IN PROGRAM MEMORY**




### 5.7.1 TWO-WORD INSTRUCTIONS

PIC18F2X20/4X20 devices have four two-word instructions: MOVFF, CALL, GOTO and LFSR. The second word of these instructions has the 4 MSBs set to '1's and is decoded as a NOP instruction. The lower 12 bits of the second word contain data to be used by the instruction. If the first word of the instruction is executed, the data in the second word is accessed. If the second word of the instruction is executed by itself (first word was skipped), it will execute as a NOP. This action is necessary when the two-word instruction is preceded by a conditional instruction that results in a skip operation. A program example that demonstrates this concept is shown in Example 5-3. Refer to **Section 24.0 "Instruction Set Summary"** for further details of the instruction set.

**EXAMPLE 5-3: TWO-WORD INSTRUCTIONS**

**CASE 1:**

| Object Code | Source Code | |
|---|---|---|
| 0110 0110 0000 0000 | TSTFSZ REG1 | ; is RAM location 0? |
| 1100 0001 0010 0011 | MOVFF REG1, REG2 | ; No, skip this word |
| 1111 0100 0101 0110 | | ; Execute this word as a NOP |
| 0010 0100 0000 0000 | ADDWF REG3 | ; continue code |

**CASE 2:**

| Object Code | Source Code | |
|---|---|---|
| 0110 0110 0000 0000 | TSTFSZ REG1 | ; is RAM location 0? |
| 1100 0001 0010 0011 | MOVFF REG1, REG2 | ; Yes, execute this word |
| 1111 0100 0101 0110 | | ; 2nd word of instruction |
| 0010 0100 0000 0000 | ADDWF REG3 | ; continue code |


© 2003 Microchip Technology Inc.

## 5.8 Look-up Tables

Look-up tables are implemented two ways:

- Computed GOTO
- Table Reads

### 5.8.1 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter. An example is shown in Example 5-4.

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW 0xnn instructions. WREG is loaded with an offset into the table before executing a call to that table. The first instruction of the called routine is the ADDWF PCL instruction. The next instruction executed will be one of the RETLW 0xnn instructions that returns the value 0xnn to the calling function.

The offset value (in WREG) specifies the number of bytes that the program counter should advance and should be multiples of 2 (LSB = 0).

In this method, only one data byte may be stored in each instruction location and room on the return address stack is required.

**EXAMPLE 5-4: COMPUTED GOTO USING AN OFFSET VALUE**

```
        MOVFW   OFFSET
        CALL    TABLE
ORG     0xnn00
TABLE   ADDWF   PCL
        RETLW   0xnn
        RETLW   0xnn
        RETLW   0xnn
        •
        •
        •
```

### 5.8.2 TABLE READS/TABLE WRITES

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location.

Look-up table data may be stored two bytes per program word by using table reads and writes. The table pointer (TBLPTR) specifies the byte address and the table latch (TABLAT) contains the data that is read from, or written to program memory. Data is transferred to/from program memory, one byte at a time.

The Table Read/Table Write operation is discussed further in **Section 6.1 "Table Reads and Table Writes"**.

## 5.9 Data Memory Organization

The data memory is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4096 bytes of data memory. Figure 5-6 shows the data memory organization for the PIC18F2X20/4X20 devices.

The data memory map is divided into as many as 16 banks that contain 256 bytes each. The lower 4 bits of the Bank Select Register (BSR<3:0>) select which bank will be accessed. The upper 4 bits of the BSR are not implemented.

The data memory contains Special Function Registers (SFR) and General Purpose Registers (GPR). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratch pad operations in the user's application. The SFRs start at the last location of Bank 15 (FFFh) and extend towards F80h. Any remaining space beyond the SFRs in the bank may be implemented as GPRs. GPRs start at the first location of Bank 0 and grow upwards. Any read of an unimplemented location will read as '0's.

The entire data memory may be accessed directly or indirectly. Direct addressing may require the use of the BSR register. Indirect addressing requires the use of a File Select Register (FSRn) and a corresponding Indirect File Operand (INDFn). Each FSR holds a 12-bit address value that can be used to access any location in the data memory map without banking. See **Section 5.12 "Indirect Addressing, INDF and FSR Registers"** for indirect addressing details.

The instruction set and architecture allow operations across all banks. This may be accomplished by indirect addressing or by the use of the MOVFF instruction. The MOVFF instruction is a two-word/two-cycle instruction that moves a value from one register to another.

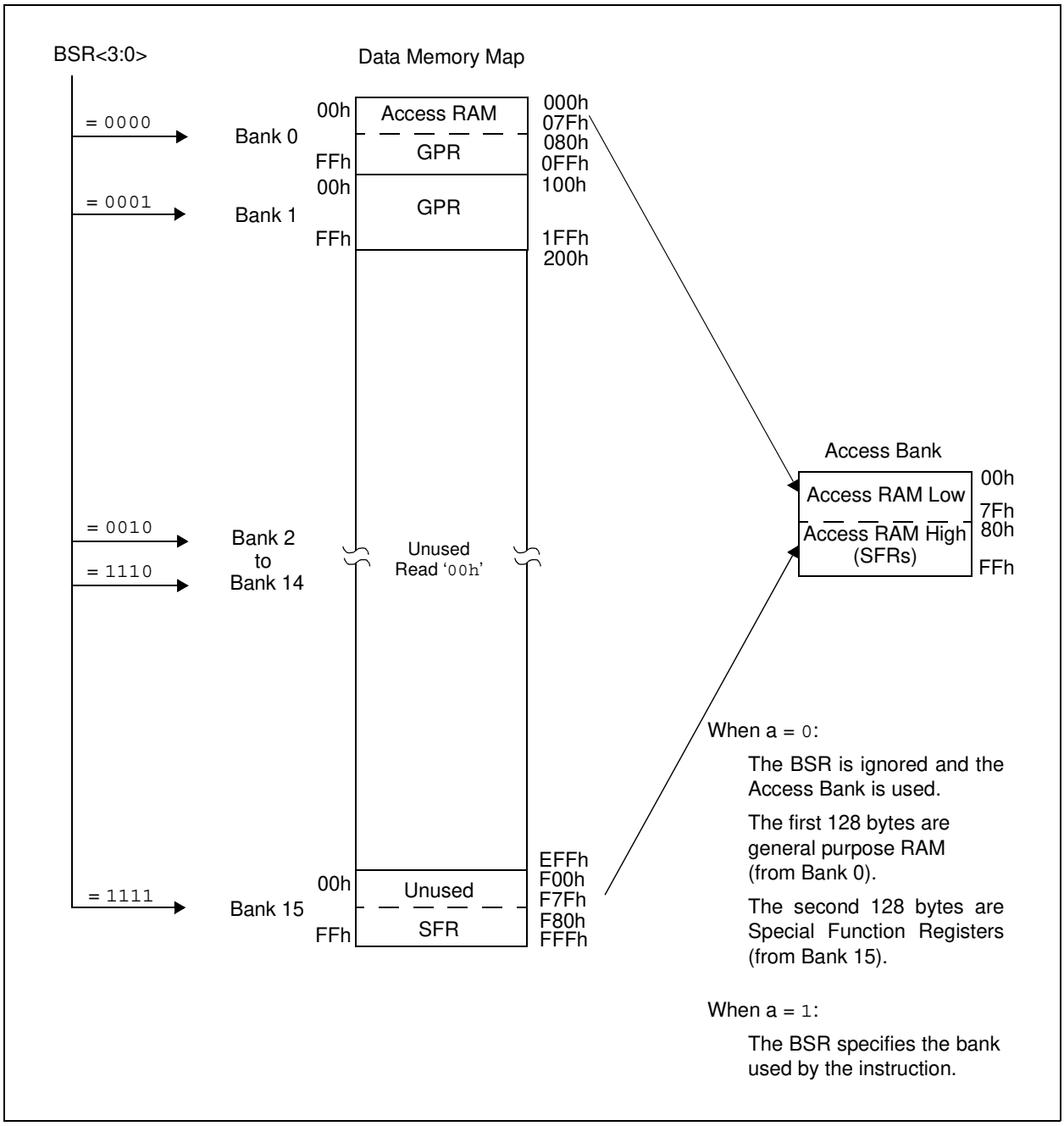
To ensure that commonly used registers (SFRs and select GPRs) can be accessed in a single cycle, regardless of the current BSR values, an Access Bank is implemented. A segment of Bank 0 and a segment of Bank 15 comprise the Access RAM. **Section 5.10 "Access Bank"** provides a detailed description of the Access RAM.

### 5.9.1 GENERAL PURPOSE REGISTER FILE

Enhanced MCU devices may have banked memory in the GPR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

Data RAM is available for use as GPR registers by all instructions. The second half of Bank 15 (F80h to FFFh) contains SFRs. All other banks of data memory contain GPRs, starting with Bank 0.

**FIGURE 5-6: DATA MEMORY MAP FOR PIC18F2X20/4X20 DEVICES**





© 2003 Microchip Technology Inc.

### 5.9.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 5-1 and Table 5-2.

The SFRs can be classified into two sets: those associated with the "core" function and those related to the peripheral functions. Those registers related to the "core" are described in this section, while those related to the operation of the peripheral features are described in the section of that peripheral feature.

The SFRs are typically distributed among the peripherals whose functions they control.

The unused SFR locations will be unimplemented and read as '0's.

**TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F2X20/4X20 DEVICES**

| Address | Name | Address | Name | Address | Name | Address | Name |
|---|---|---|---|---|---|---|---|
| FFFh | TOSU | FDFh | INDF2(2) | FBFh | CCPR1H | F9Fh | IPR1 |
| FFEh | TOSH | FDEh | POSTINC2(2) | FBEh | CCPR1L | F9Eh | PIR1 |
| FFDh | TOSL | FDDh | POSTDEC2(2) | FBDh | CCP1CON | F9Dh | PIE1 |
| FFCh | STKPTR | FDCh | PREINC2(2) | FBCh | CCPR2H | F9Ch | — |
| FFBh | PCLATU | FDBh | PLUSW2(2) | FBBh | CCPR2L | F9Bh | OSCTUNE |
| FFAh | PCLATH | FDAh | FSR2H | FBAh | CCP2CON | F9Ah | — |
| FF9h | PCL | FD9h | FSR2L | FB9h | — | F99h | — |
| FF8h | TBLPTRU | FD8h | STATUS | FB8h | — | F98h | — |
| FF7h | TBLPTRH | FD7h | TMR0H | FB7h | PWM1CON(1) | F97h | — |
| FF6h | TBLPTRL | FD6h | TMR0L | FB6h | ECCPAS(1) | F96h | TRISE(1) |
| FF5h | TABLAT | FD5h | T0CON | FB5h | CVRCON | F95h | TRISD(1) |
| FF4h | PRODH | FD4h | — | FB4h | CMCON | F94h | TRISC |
| FF3h | PRODL | FD3h | OSCCON | FB3h | TMR3H | F93h | TRISB |
| FF2h | INTCON | FD2h | LVDCON | FB2h | TMR3L | F92h | TRISA |
| FF1h | INTCON2 | FD1h | WDTCON | FB1h | T3CON | F91h | — |
| FF0h | INTCON3 | FD0h | RCON | FB0h | — | F90h | — |
| FEFh | INDF0(2) | FCFh | TMR1H | FAFh | SPBRG | F8Fh | — |
| FEEh | POSTINC0(2) | FCEh | TMR1L | FAEh | RCREG | F8Eh | — |
| FEDh | POSTDEC0(2) | FCDh | T1CON | FADh | TXREG | F8Dh | LATE(1) |
| FECh | PREINC0(2) | FCCh | TMR2 | FACh | TXSTA | F8Ch | LATD(1) |
| FEBh | PLUSW0(2) | FCBh | PR2 | FABh | RCSTA | F8Bh | LATC |
| FEAh | FSR0H | FCAh | T2CON | FAAh | — | F8Ah | LATB |
| FE9h | FSR0L | FC9h | SSPBUF | FA9h | EEADR | F89h | LATA |
| FE8h | WREG | FC8h | SSPADD | FA8h | EEDATA | F88h | — |
| FE7h | INDF1(2) | FC7h | SSPSTAT | FA7h | EECON2 | F87h | — |
| FE6h | POSTINC1(2) | FC6h | SSPCON1 | FA6h | EECON1 | F86h | — |
| FE5h | POSTDEC1(2) | FC5h | SSPCON2 | FA5h | — | F85h | — |
| FE4h | PREINC1(2) | FC4h | ADRESH | FA4h | — | F84h | PORTE |
| FE3h | PLUSW1(2) | FC3h | ADRESL | FA3h | — | F83h | PORTD(1) |
| FE2h | FSR1H | FC2h | ADCON0 | FA2h | IPR2 | F82h | PORTC |
| FE1h | FSR1L | FC1h | ADCON1 | FA1h | PIR2 | F81h | PORTB |
| FE0h | BSR | FC0h | ADCON2 | FA0h | PIE2 | F80h | PORTA |

**Legend:** — = Unimplemented registers, read as '0'.

**Note 1:** This register is not available on PIC18F2X20 devices.

**2:** This is not a physical register.

**TABLE 5-2: REGISTER FILE SUMMARY (PIC18F2220/2320/4220/4320)**

| File Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Details on page: |
|---|---|---|---|---|---|---|---|---|---|---|
| TOSU | — | — | — | Top-of-Stack Upper Byte (TOS<20:16>) | | | | | ---0 0000 | 46, 54 |
| TOSH | Top-of-Stack High Byte (TOS<15:8>) | | | | | | | | 0000 0000 | 46, 54 |
| TOSL | Top-of-Stack Low Byte (TOS<7:0>) | | | | | | | | 0000 0000 | 46, 54 |
| STKPTR | STKFUL | STKUNF | — | Return Stack Pointer | | | | | 00-0 0000 | 46, 55 |
| PCLATU | — | — | bit 21(3) | Holding Register for PC<20:16> | | | | | ---0 0000 | 46, 56 |
| PCLATH | Holding Register for PC<15:8> | | | | | | | | 0000 0000 | 46, 56 |
| PCL | PC Low Byte (PC<7:0>) | | | | | | | | 0000 0000 | 46, 56 |
| TBLPTRU | — | — | bit 21 | Program Memory Table Pointer Upper Byte (TBLPTR<20:16>) | | | | | --00 0000 | 46, 74 |
| TBLPTRH | Program Memory Table Pointer High Byte (TBLPTR<15:8>) | | | | | | | | 0000 0000 | 46, 74 |
| TBLPTRL | Program Memory Table Pointer Low Byte (TBLPTR<7:0>) | | | | | | | | 0000 0000 | 46, 74 |
| TABLAT | Program Memory Table Latch | | | | | | | | 0000 0000 | 46, 74 |
| PRODH | Product Register High Byte | | | | | | | | xxxx xxxx | 46, 85 |
| PRODL | Product Register Low Byte | | | | | | | | xxxx xxxx | 46, 85 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 46, 89 |
| INTCON2 | $\overline{\text{RBPU}}$ | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP | 1111 -1-1 | 46, 90 |
| INTCON3 | INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF | 11-0 0-00 | 46, 91 |
| INDF0 | Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register) | | | | | | | | n/a | 46, 66 |
| POSTINC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| POSTDEC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| PREINC0 | Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| PLUSW0 | Uses contents of FSR0 to address data memory – value of FSR0 offset by W (not a physical register) | | | | | | | | n/a | 46, 66 |
| FSR0H | — | — | — | — | Indirect Data Memory Address Pointer 0 High | | | | ---- 0000 | 46, 66 |
| FSR0L | Indirect Data Memory Address Pointer 0 Low Byte | | | | | | | | xxxx xxxx | 46, 66 |
| WREG | Working Register | | | | | | | | xxxx xxxx | 46 |
| INDF1 | Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register) | | | | | | | | n/a | 46, 66 |
| POSTINC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| POSTDEC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| PREINC1 | Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) | | | | | | | | n/a | 46, 66 |
| PLUSW1 | Uses contents of FSR1 to address data memory – value of FSR1 offset by W (not a physical register) | | | | | | | | n/a | 46, 66 |
| FSR1H | — | — | — | — | Indirect Data Memory Address Pointer 1 High | | | | ---- 0000 | 47, 66 |
| FSR1L | Indirect Data Memory Address Pointer 1 Low Byte | | | | | | | | xxxx xxxx | 47, 66 |
| BSR | — | — | — | — | Bank Select Register | | | | ---- 0000 | 47, 65 |
| INDF2 | Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register) | | | | | | | | n/a | 47, 66 |
| POSTINC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register) | | | | | | | | n/a | 47, 66 |
| POSTDEC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register) | | | | | | | | n/a | 47, 66 |
| PREINC2 | Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) | | | | | | | | n/a | 47, 66 |
| PLUSW2 | Uses contents of FSR2 to address data memory – value of FSR2 offset by W (not a physical register) | | | | | | | | n/a | 47, 66 |
| FSR2H | — | — | — | — | Indirect Data Memory Address Pointer 2 High | | | | ---- 0000 | 47, 66 |
| FSR2L | Indirect Data Memory Address Pointer 2 Low Byte | | | | | | | | xxxx xxxx | 47, 66 |
| STATUS | — | — | — | N | OV | Z | DC | C | ---x xxxx | 47, 68 |
| TMR0H | Timer0 Register High Byte | | | | | | | | 0000 0000 | 47, 119 |
| TMR0L | Timer0 Register Low Byte | | | | | | | | xxxx xxxx | 47, 119 |
| T0CON | TMR0ON | T08BIT | T0CS | T0SE | PSA | T0PS2 | T0PS1 | T0PS0 | 1111 1111 | 47, 117 |

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

**Note 1:** RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator mode only and read '0' in all other oscillator modes.

**2:** RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

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

**4:** If PBADEN = 0, PORTB<4:0> are configured as digital input and read unknown and if PBADEN = 1, PORTB<4:0> are configured as analog input and read '0' following a Reset.

**5:** These registers and/or bits are not implemented on the PIC18F2X20 devices and read as '0'.

**6:** The RE3 port bit is only available when MCLRE fuse (CONFIG3H<7>) is programmed to '0'. Otherwise, RE3 reads '0'. This bit is read-only.


© 2003 Microchip Technology Inc.

**TABLE 5-2: REGISTER FILE SUMMARY (PIC18F2220/2320/4220/4320) (CONTINUED)**

| File Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Details on page: |
|---|---|---|---|---|---|---|---|---|---|---|
| OSCCON | IDLEN | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 | 0000 q000 | 26, 47 |
| LVDCON | — | — | IRVST | LVDEN | LVDL3 | LVDL2 | LVDL1 | LVDL0 | --00 0101 | 47, 233 |
| WDTCON | — | — | — | — | — | — | — | SWDTEN | --- ---0 | 47, 246 |
| RCON | IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ | 0--1 11q0 | 45, 69, 98 |
| TMR1H | Timer1 Register High Byte | | | | | | | | xxxx xxxx | 47, 125 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | xxxx xxxx | 47, 125 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | TMR1CS | TMR1ON | 0000 0000 | 47, 121 |
| TMR2 | Timer2 Register | | | | | | | | 0000 0000 | 47, 127 |
| PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 47, 127 |
| T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | 47, 127 |
| SSPBUF | SSP Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | 47, 156, 164 |
| SSPADD | SSP Address Register in $I^2C$ Slave mode. SSP Baud Rate Reload Register in $I^2C$ Master mode. | | | | | | | | 0000 0000 | 47, 164 |
| SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 47, 156, 165 |
| SSPCON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 47, 157, 166 |
| SSPCON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 47, 167 |
| ADRESH | A/D Result Register High Byte | | | | | | | | xxxx xxxx | 48, 220 |
| ADRESL | A/D Result Register Low Byte | | | | | | | | xxxx xxxx | 48, 220 |
| ADCON0 | — | — | CHS3 | CHS2 | CHS1 | CHS0 | GO/$\overline{DONE}$ | ADON | --00 0000 | 48, 211 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0000 | 48, 212 |
| ADCON2 | ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 | 0-00 0000 | 48, 213 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | xxxx xxxx | 48, 134 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | xxxx xxxx | 48, 134 |
| CCP1CON | P1M1(5) | P1M0(5) | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 0000 0000 | 48, 133, 141 |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte | | | | | | | | xxxx xxxx | 48, 134 |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte | | | | | | | | xxxx xxxx | 48, 134 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | 48, 133 |
| PWM1CON(5) | PRSEN | PDC6 | PDC5 | PDC4 | PDC3 | PDC2 | PDC1 | PDC0 | 0000 0000 | 48, 149 |
| ECCPAS(5) | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 | PSSBD0 | 0000 0000 | 48, 150 |
| CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 48, 227 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 48, 221 |
| TMR3H | Timer3 Register High Byte | | | | | | | | xxxx xxxx | 48, 131 |
| TMR3L | Timer3 Register Low Byte | | | | | | | | xxxx xxxx | 48, 131 |
| T3CON | RD16 | T3CCP2 | T3CKPS1 | T3CKPS0 | T3CCP1 | $\overline{T3SYNC}$ | TMR3CS | TMR3ON | 0000 0000 | 48, 129 |
| SPBRG | USART Baud Rate Generator | | | | | | | | 0000 0000 | 48, 198 |
| RCREG | USART Receive Register | | | | | | | | 0000 0000 | 48, 204, 203 |
| TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 48, 202, 203 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 48, 196 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 48, 197 |

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

**Note 1:** RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator mode only and read '0' in all other oscillator modes.

**2:** RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

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

**4:** If PBADEN = 0, PORTB<4:0> are configured as digital input and read unknown and if PBADEN = 1, PORTB<4:0> are configured as analog input and read '0' following a Reset.

**5:** These registers and/or bits are not implemented on the PIC18F2X20 devices and read as '0'.

**6:** The RE3 port bit is only available when MCLRE fuse (CONFIG3H<7>) is programmed to '0'. Otherwise, RE3 reads '0'. This bit is read-only.

**TABLE 5-2: REGISTER FILE SUMMARY (PIC18F2220/2320/4220/4320) (CONTINUED)**

| File Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Details on page: |
|---|---|---|---|---|---|---|---|---|---|---|
| EEADR | EEPROM Address Register | | | | | | | | 0000 0000 | 48, 81 |
| EEDATA | EEPROM Data Register | | | | | | | | 0000 0000 | 48, 84 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 0000 0000 | 48, 72, 81 |
| EECON1 | EEPGD | CFGS | — | FREE | WRERR | WREN | WR | RD | xx-0 x000 | 48, 73, 82 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | 49, 97 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | 49, 93 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | 49, 95 |
| IPR1 | PSPIP(5) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 49, 96 |
| PIR1 | PSPIF(5) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 49, 92 |
| PIE1 | PSPIE(5) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 49, 94 |
| OSCTUNE | — | — | TUN5 | TUN4 | TUN3 | TUN2 | TUN1 | TUN0 | --00 0000 | 23, 49 |
| TRISE(5) | IBF | OBF | IBOV | PSPMODE | — | Data Direction bits for PORTE(5) | | | 0000 -111 | 49, 112 |
| TRISD(5) | Data Direction Control Register for PORTD | | | | | | | | 1111 1111 | 49, 110 |
| TRISC | Data Direction Control Register for PORTC | | | | | | | | 1111 1111 | 49, 108 |
| TRISB | Data Direction Control Register for PORTB | | | | | | | | 1111 1111 | 49, 106 |
| TRISA | TRISA7(2) | TRISA6(1) | Data Direction Control Register for PORTA | | | | | | 1111 1111 | 49, 103 |
| LATE(5) | — | — | — | — | — | Read/Write PORTE Data Latch | | | ---- -xxx | 49, 113 |
| LATD(5) | Read/Write PORTD Data Latch | | | | | | | | xxxx xxxx | 49, 110 |
| LATC | Read/Write PORTC Data Latch | | | | | | | | xxxx xxxx | 49, 108 |
| LATB | Read/Write PORTB Data Latch | | | | | | | | xxxx xxxx | 49, 106 |
| LATA | LATA<7>(2) | LATA<6>(1) | Read/Write PORTA Data Latch | | | | | | xxxx xxxx | 49, 103 |
| PORTE | — | — | — | — | RE3(6) | Read PORTE pins, Write PORTE Data Latch(5) | | | ---- xxxx | 49, 113 |
| PORTD | Read PORTD pins, Write PORTD Data Latch | | | | | | | | xxxx xxxx | 49, 110 |
| PORTC | Read PORTC pins, Write PORTC Data Latch | | | | | | | | xxxx xxxx | 49, 108 |
| PORTB | Read PORTB pins, Write PORTB Data Latch(4) | | | | | | | | xxxx xxxx | 49, 106 |
| PORTA | RA7(2) | RA6(1) | Read PORTA pins, Write PORTA Data Latch | | | | | | xx0x 0000 | 49, 103 |

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

**Note 1:** RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator mode only and read '0' in all other oscillator modes.

**2:** RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

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

**4:** If PBADEN = 0, PORTB<4:0> are configured as digital input and read unknown and if PBADEN = 1, PORTB<4:0> are configured as analog input and read '0' following a Reset.

**5:** These registers and/or bits are not implemented on the PIC18F2X20 devices and read as '0'.

**6:** The RE3 port bit is only available when MCLRE fuse (CONFIG3H<7>) is programmed to '0'. Otherwise, RE3 reads '0'. This bit is read-only.


© 2003 Microchip Technology Inc.

## 5.10 Access Bank

The Access Bank is an architectural enhancement which is very useful for C compiler code optimization. The techniques used by the C compiler may also be useful for programs written in assembly.

This data memory region can be used for:

- Intermediate computational values
- Local variables of subroutines
- Faster context saving/switching of variables
- Common variables
- Faster evaluation/control of SFRs (no banking)

The Access Bank is comprised of the last 128 bytes in Bank 15 (SFRs) and the first 128 bytes in Bank 0. These two sections will be referred to as Access RAM High and Access RAM Low, respectively. Figure 5-6 indicates the Access RAM areas.

A bit in the instruction word specifies if the operation is to occur in the bank specified by the BSR register or in the Access Bank. This bit is denoted as the 'a' bit (for access bit).

When forced in the Access Bank (a = 0), the last address in Access RAM Low is followed by the first address in Access RAM High. Access RAM High maps the Special Function Registers, so these registers can be accessed without any software overhead. This is useful for testing status flags and modifying control bits.

## 5.11 Bank Select Register (BSR)

The need for a large general purpose memory space dictates a RAM banking scheme. The data memory is partitioned into as many as sixteen banks. When using direct addressing, the BSR should be configured for the desired bank.

BSR<3:0> holds the upper 4 bits of the 12-bit RAM address. The BSR<7:4> bits will always read '0's and writes will have no effect (see Figure 5-7).

A MOVLB instruction has been provided in the instruction set to assist in selecting banks.

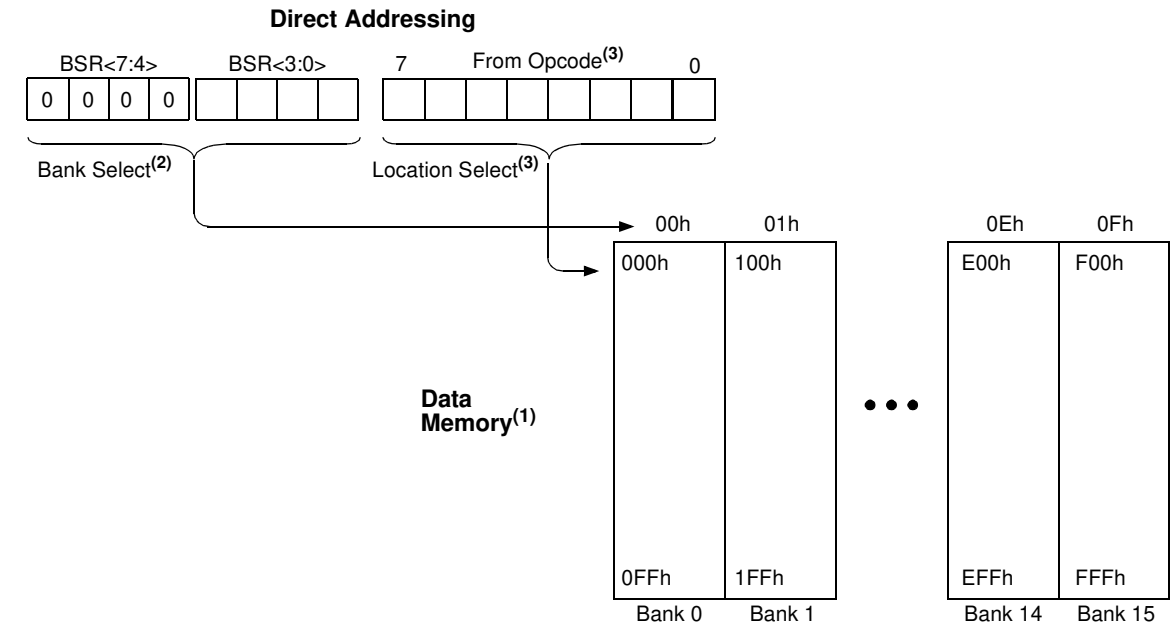
If the currently selected bank is not implemented, any read will return all '0's and all writes are ignored. The Status register bits will be set/cleared as appropriate for the instruction performed.

Each Bank extends up to FFh (256 bytes). All data memory is implemented as static RAM.

A MOVFF instruction ignores the BSR since the 12-bit addresses are embedded into the instruction word.

**Section 5.12 "Indirect Addressing, INDF and FSR Registers"** provides a description of indirect addressing which allows linear addressing of the entire RAM space.

**FIGURE 5-7: DIRECT ADDRESSING**




**Note 1:** For register file map detail, see Table 5-1.
**2:** The access bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.
**3:** The MOVFF instruction embeds the entire 12-bit address in the instruction.

## 5.12 Indirect Addressing, INDF and FSR Registers

Indirect addressing is a mode of addressing data memory, where the data memory address in the instruction is not fixed. An FSR register is used as a pointer to the data memory location that is to be read or written. Since this pointer is in RAM, the contents can be modified by the program. This can be useful for data tables in the data memory and for software stacks. Figure 5-8 shows how the fetched instruction is modified prior to being executed.

Indirect addressing is possible by using one of the INDF registers. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = 0), will read 00h. Writing to the INDF register indirectly, results in a no operation. The FSR register contains a 12-bit address which is shown in Figure 5-9.

The INDFn register is not a physical register. Addressing INDFn actually addresses the register whose address is contained in the FSRn register (FSRn is a pointer); this is indirect addressing.

Example 5-5 shows a simple use of indirect addressing to clear the RAM in Bank 1 (locations 100h-1FFh) in a minimum number of instructions.

**EXAMPLE 5-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING**

```
          LFSR   FSR0,0x100 ;
NEXT      CLRF   POSTINC0   ; Clear INDF
                            ; register then
                            ; inc pointer
          BTFSS  FSR0H, 1   ; All done with
                            ; Bank1?
          GOTO   NEXT       ; NO, clear next
CONTINUE                    ; YES, continue
```

There are three indirect addressing registers. To address the entire data memory space (4096 bytes), these registers are 12 bits wide. To store the 12 bits of addressing information, two 8-bit registers are required:

1. FSR0: composed of FSR0H:FSR0L
2. FSR1: composed of FSR1H:FSR1L
3. FSR2: composed of FSR2H:FSR2L

In addition, there are registers INDF0, INDF1 and INDF2, which are not physically implemented. Reading or writing to these registers activates indirect addressing with the value in the corresponding FSR register being the address of the data. If an instruction writes a value to INDF0, the value will be written to the address pointed to by FSR0H:FSR0L. A read from INDF1 reads the data from the address pointed to by FSR1H:FSR1L. INDFn can be used in code anywhere an operand can be used.

If INDF0, INDF1 or INDF2 are read indirectly via an FSR, all '0's are read (zero bit is set). Similarly, if INDF0, INDF1 or INDF2 are written to indirectly, the operation will be equivalent to a NOP instruction and the status bits are not affected.

### 5.12.1 INDIRECT ADDRESSING OPERATION

Each FSR register has an INDF register associated with it, plus four additional register addresses. Performing an operation using one of these five registers determines how the FSR will be modified during indirect addressing.

When data access is performed using one of the five INDFn locations, the address selected will configure the FSRn register to:

- Do nothing to FSRn after an indirect access (no change) – INDFn
- Auto-decrement FSRn after an indirect access (post-decrement) – POSTDECn
- Auto-increment FSRn after an indirect access (post-increment) – POSTINCn
- Auto-increment FSRn before an indirect access (pre-increment) – PREINCn
- Use the value in the WREG register as an offset to FSRn. Do not modify the value of the WREG or the FSRn register after an indirect access (no change) – PLUSWn

When using the auto-increment or auto-decrement features, the effect on the FSR is not reflected in the Status register. For example, if the indirect address causes the FSR to equal '0', the Z bit will not be set.

Auto-incrementing or auto-decrementing an FSR affects all 12 bits. That is, when FSRnL overflows from an increment, FSRnH will be incremented automatically.

Adding these features allows the FSRn to be used as a stack pointer, in addition to its use for table operations in data memory.

Each FSR has an address associated with it that performs an indexed indirect access. When a data access to this INDFn location (PLUSWn) occurs, the FSRn is configured to add the signed value in the WREG register and the value in FSR to form the address before an indirect access. The FSR value is not changed. The WREG offset range is -128 to +127.

If an FSR register contains a value that points to one of the INDFn, an indirect read will read 00h (zero bit is set) while an indirect write will be equivalent to a NOP (status bits are not affected).

If an indirect addressing write is performed when the target address is an FSRnH or FSRnL register, the data is written to the FSR register but no pre- or post-increment/decrement is performed.


© 2003 Microchip Technology Inc.

**FIGURE 5-8: INDIRECT ADDRESSING OPERATION**

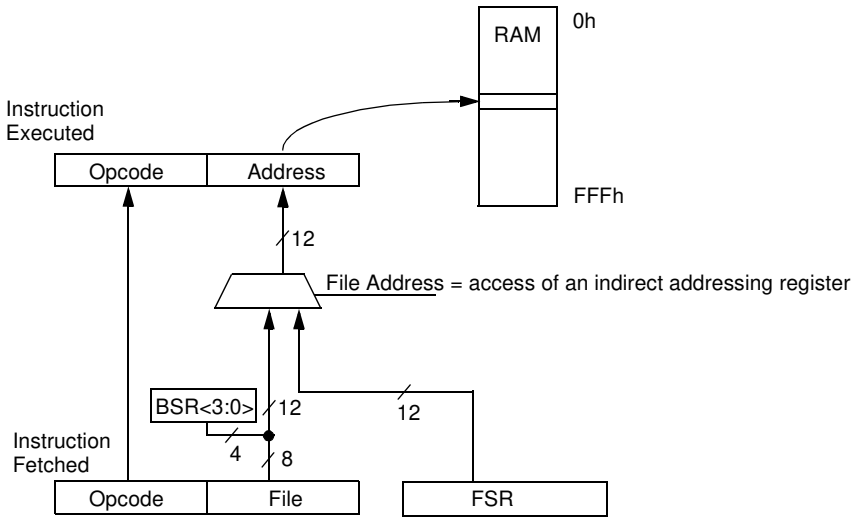
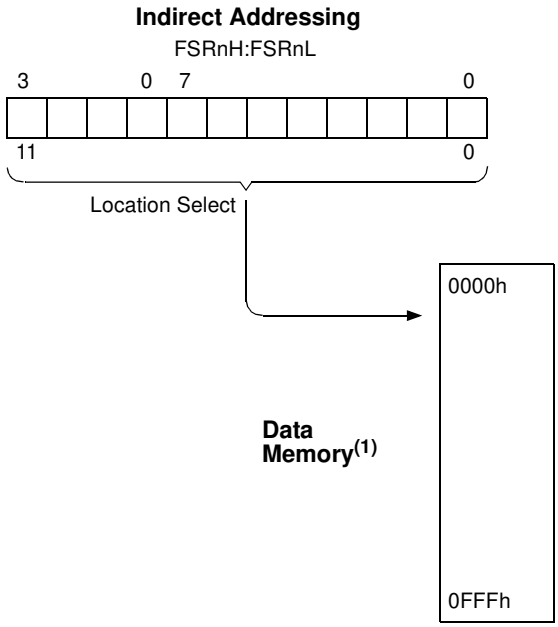



**FIGURE 5-9: INDIRECT ADDRESSING**




**Note 1:** For register file map detail, see Table 5-1.

## 5.13 Status Register

The Status register, shown in Register 5-2, contains the arithmetic status of the ALU. The Status register can be the operand for any instruction as with any other register. If the Status register is the destination for an instruction that affects the Z, DC, C, OV or N bits, then the write to these five bits is disabled. These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the Status register as destination may be different than intended.

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

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF`, `MOVFF` and `MOVWF` instructions are used to alter the Status register, because these instructions do not affect the Z, C, DC, OV or N bits in the Status register. For other instructions not affecting any status bits, see Table 24-2.

| **Note:** | The C and DC bits operate as a $\overline{\text{borrow}}$ and digit $\overline{\text{borrow}}$ bit respectively, in subtraction. |
|---|---|

**REGISTER 5-2: STATUS REGISTER**

| U-0 | U-0 | U-0 | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| — | — | — | N | OV | Z | DC | C |
| bit 7 | | | | | | | bit 0 |

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **N:** Negative bit

This bit is used for signed arithmetic (2's complement). It indicates whether the result was negative (ALU MSB = 1).

1 = Result was negative
0 = Result was positive

bit 3 **OV:** Overflow bit

This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude which causes the sign bit (bit 7) to change state.

1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit

1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit carry/$\overline{\text{borrow}}$ bit

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions.

1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

| **Note:** | For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the bit 4 or bit 3 of the source register. |
|---|---|

bit 0 **C:** Carry/$\overline{\text{borrow}}$ bit

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions.

1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

| **Note:** | For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register. |
|---|---|

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


© 2003 Microchip Technology Inc.

## 5.14 RCON Register

The Reset Control (RCON) register contains flag bits that allow differentiation between the sources of a device Reset. These flags include the $\overline{TO}$, $\overline{PD}$, $\overline{POR}$, $\overline{BOR}$ and $\overline{RI}$ bits. This register is readable and writable.

**Note 1:** If the BOREN configuration bit is set (Brown-out Reset enabled), the $\overline{BOR}$ bit is '1' on a Power-on Reset. After a Brown-out Reset has occurred, the BOR bit will be cleared and must be set by firmware to indicate the occurrence of the next Brown-out Reset.

**2:** It is recommended that the $\overline{POR}$ bit be set after a Power-on Reset has been detected so that subsequent Power-on Resets may be detected.

**REGISTER 5-3: RCON REGISTER**

| R/W-0 | U-0 | U-0 | R/W-1 | R-1 | R-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ |
| bit 7 | | | | | | | bit 0 |

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

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

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Cleared by execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

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

**NOTES:**


© 2003 Microchip Technology Inc.

## 6.0 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range.

A read from program memory is executed on one byte at a time. A write to program memory is executed on blocks of 8 bytes at a time. Program memory is erased in blocks of 64 bytes at a time. A bulk erase operation may not be issued from user code.

While writing or erasing program memory, instruction fetches cease until the operation is complete. The program memory cannot be accessed during the write or erase, therefore, code cannot execute. An internal programming timer terminates program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

### 6.1 Table Reads and Table Writes

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

The program memory space is 16 bits wide while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

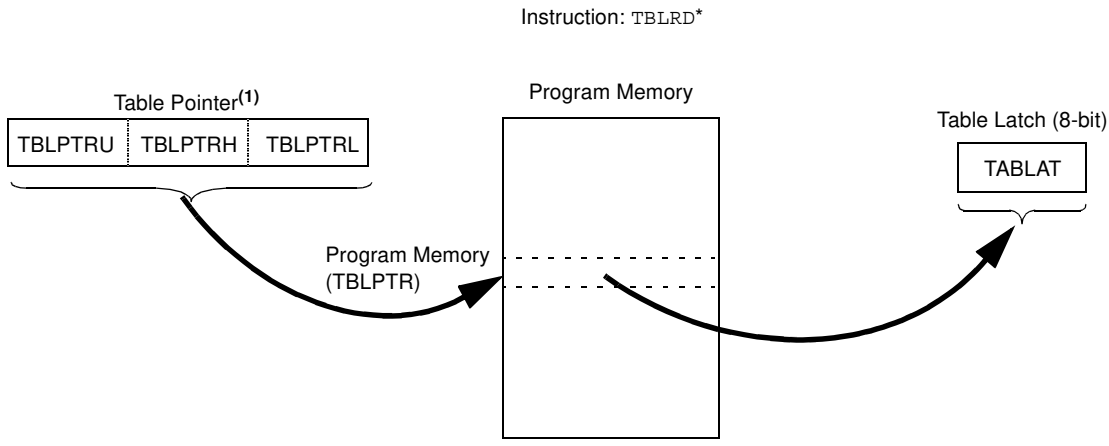
Table read operations retrieve data from program memory and place it into TABLAT in the data RAM space. Figure 6-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from TABLAT in the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 6.5 "Writing to Flash Program Memory"**. Figure 6-2 shows the operation of a table write with program memory and data RAM.

Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word aligned (TBLPTRL<0> = 0).

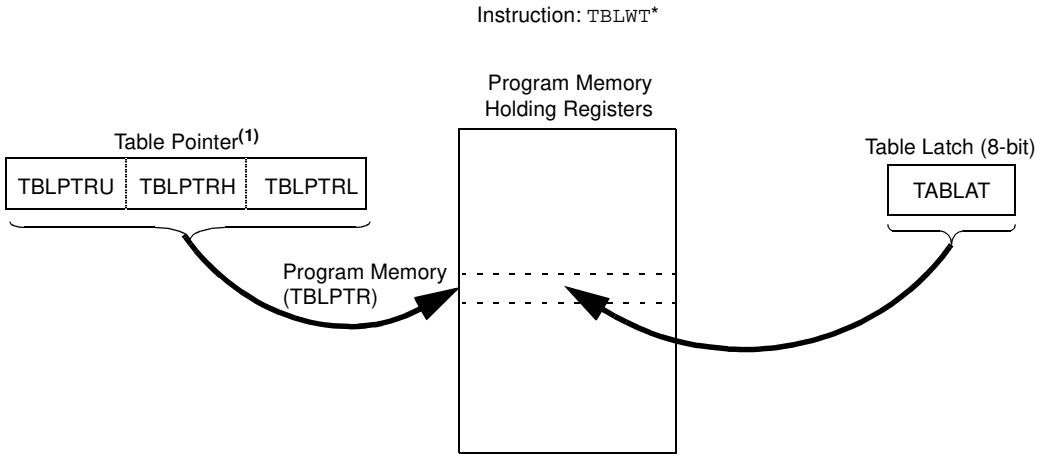
The EEPROM on-chip timer controls the write and erase times. The write and erase voltages are generated by an on-chip charge pump rated to operate over the voltage range of the device for byte or word operations.

**FIGURE 6-1: TABLE READ OPERATION**




**Note 1:** Table Pointer points to a byte in program memory.

**FIGURE 6-2: TABLE WRITE OPERATION**




**Note 1:** Table Pointer actually points to one of eight holding registers, the address of which is determined by TBLPTRL<2:0>. The process for physically writing data to the program memory array is discussed in **Section 6.5 "Writing to Flash Program Memory"**.

## 6.2 Control Registers

Several control registers are used in conjunction with the TBLRD and TBLWT instructions. These include the:

- EECON1 register
- EECON2 register
- TABLAT register
- TBLPTR registers

### 6.2.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit, EEPGD, determines if the access will be to program or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit, CFGS, determines if the access will be to the configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access configuration registers. When CFGS is clear, the EEPGD bit selects either program Flash or data EEPROM memory.

The FREE bit controls program memory erase operations. When the FREE bit is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

The WREN bit enables and disables erase and write operations. When set, erase and write operations are allowed. When clear, erase and write operations are disabled – the WR bit cannot be set while the WREN bit is clear. This process helps to prevent accidental writes to memory due to errant (unexpected) code execution.

Firmware should keep the WREN bit clear at all times except when starting erase or write operations. Once firmware has set the WR bit, the WREN bit may be cleared. Clearing the WREN bit will not affect the operation in progress.

The WRERR bit is set when a write operation is interrupted by a Reset. In these situations, the user can check the WRERR bit and rewrite the location. It will be necessary to reload the data and address registers (EEDATA and EEADR) as these registers have cleared as a result of the Reset.

Control bits, RD and WR, start read and erase/write operations, respectively. These bits are set by firmware and cleared by hardware at the completion of the operation.

The RD bit cannot be set when accessing program memory (EEPGD = 1). Program memory is read using table read instructions. See **Section 6.3 "Reading the Flash Program Memory"** regarding table reads.

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


© 2003 Microchip Technology Inc.

**REGISTER 6-1: EECON1 REGISTER**

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

bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit
1 = Access program Flash memory
0 = Access data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EE or Configuration Select bit
1 = Access configuration registers
0 = Access program Flash or data EEPROM memory

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

bit 4 **FREE:** Flash Row Erase Enable bit
1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation – TBLPTR<5:0> are ignored)
0 = Perform write only

bit 3 **WRERR:** EEPROM Error Flag bit
1 = A write operation was prematurely terminated (any Reset during self-timed programming)
0 = The write operation completed normally

**Note:** When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

bit 2 **WREN:** Write Enable bit
1 = Allows erase or write cycles
0 = Inhibits erase or write cycles

bit 1 **WR:** Write Control bit
1 = Initiates a data EEPROM erase/write cycle or a program memory erase cycle or write cycle. (The operation is self-timed and the bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)
0 = Write cycle completed

bit 0 **RD:** Read Control bit
1 = Initiates a memory read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1.)
0 = Read completed

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

### 6.2.2 TABLAT – TABLE LATCH REGISTER

The Table Latch (TABLAT) is an 8-bit register mapped into the SFR space. The Table Latch register is used to hold 8-bit data during data transfers between program memory and data RAM.

### 6.2.3 TBLPTR – TABLE POINTER REGISTER

The Table Pointer (TBLPTR) register addresses a byte within the program memory. The TBLPTR is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). These three registers join to form a 22-bit wide pointer. The low order 21 bits allow the device to address up to 2 Mbytes of program memory space. Setting the 22nd bit allows access to the device ID, the user ID and the configuration bits.

The table pointer, TBLPTR, is used by the `TBLRD` and `TBLWT` instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations are shown in Table 6-1. These operations on the TBLPTR only affect the low order 21 bits.

### 6.2.4 TABLE POINTER BOUNDARIES

TBLPTR is used in reads, writes and erases of the Flash program memory.

When a `TBLRD` is executed, all 22 bits of the Table Pointer determine which byte is read from program or configuration memory into TABLAT.

When a `TBLWT` is executed, the three LSbs of the Table Pointer (TBLPTR<2:0>) determine which of the eight program memory holding registers is written to. When the timed write to program memory (long write) begins, the 19 MSbs of the TBLPTR (TBLPTR<21:3>) will determine which program memory block of 8 bytes is written to (TBLPTR<2:0> are ignored). For more detail, see **Section 6.5 "Writing to Flash Program Memory"**.

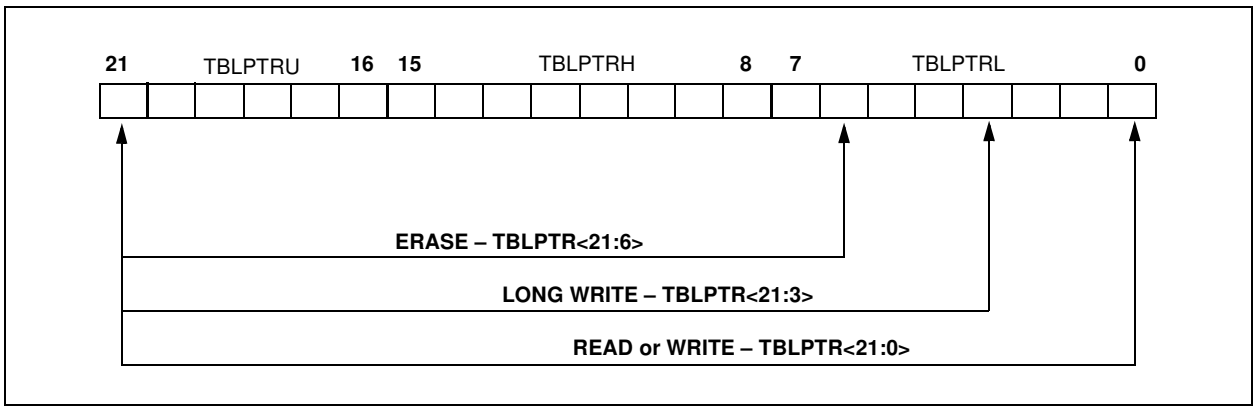
When an erase of program memory is executed, the 16 MSbs of the Table Pointer (TBLPTR<21:6>) point to the 64-byte block that will be erased. The Least Significant bits (TBLPTR<5:0>) are ignored.

Figure 6-3 describes the relevant boundaries of TBLPTR based on Flash program memory operations.

**TABLE 6-1: TABLE POINTER OPERATIONS WITH `TBLRD` AND `TBLWT` INSTRUCTIONS**

| Example | Operation on Table Pointer |
|---|---|
| `TBLRD*`<br>`TBLWT*` | TBLPTR is not modified |
| `TBLRD*+`<br>`TBLWT*+` | TBLPTR is incremented after the read/write |
| `TBLRD*-`<br>`TBLWT*-` | TBLPTR is decremented after the read/write |
| `TBLRD+*`<br>`TBLWT+*` | TBLPTR is incremented before the read/write |

**FIGURE 6-3: TABLE POINTER BOUNDARIES BASED ON OPERATION**





© 2003 Microchip Technology Inc.

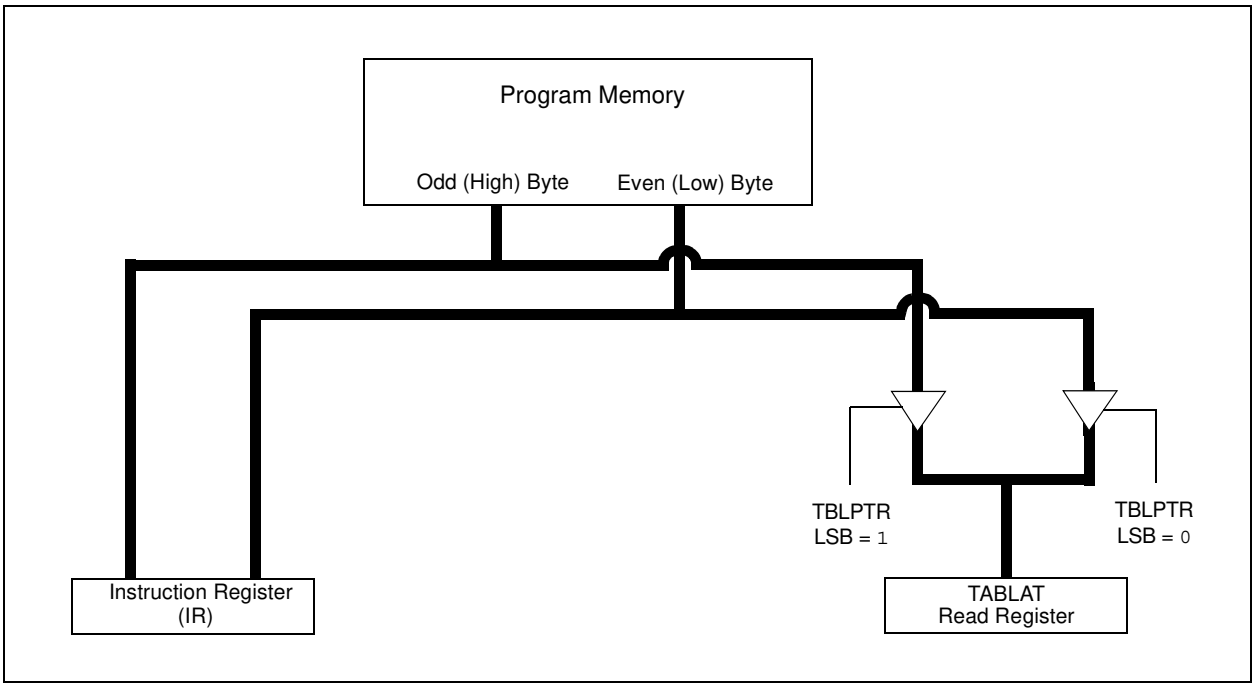
## 6.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and place it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing a TBLRD instruction places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 6-4 shows the interface between the internal program memory and the TABLAT.

**FIGURE 6-4: READS FROM FLASH PROGRAM MEMORY**




**EXAMPLE 6-1: READING A FLASH PROGRAM MEMORY WORD**

```
            MOVLW    CODE_ADDR_UPPER          ; Load TBLPTR with the base
            MOVWF    TBLPTRU                  ; address of the word
            MOVLW    CODE_ADDR_HIGH
            MOVWF    TBLPTRH
            MOVLW    CODE_ADDR_LOW
            MOVWF    TBLPTRL
READ_WORD
            TBLRD*+                           ; read into TABLAT and increment TBLPTR
            MOVFW    TABLAT                   ; get data
            MOVWF    WORD_EVEN
            TBLRD*+                           ; read into TABLAT and increment TBLPTR
            MOVFW    TABLAT                   ; get data
            MOVWF    WORD_ODD
```

## 6.4 Erasing Flash Program Memory

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

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

The EECON1 register commands the erase operation. The EEPGD bit must be set to point to the Flash program memory. The CFGS bit must be clear to access program Flash and data EEPROM memory. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation. The WR bit is set as part of the required instruction sequence (as shown in Example 6-2) and starts the actual erase operation. It is not necessary to load the TABLAT register with any data as it is ignored.

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

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

### 6.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

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

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

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

```
                MOVLW   CODE_ADDR_UPPER     ; load TBLPTR with the base
                MOVWF   TBLPTRU             ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
    ERASE_ROW
                BSF     EECON1,EEPGD        ; point to Flash program memory
                BSF     EECON1,WREN         ; enable write to memory
                BSF     EECON1,FREE         ; enable Row Erase operation
                BCF     INTCON,GIE          ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2              ; write 55H
Required        MOVLW   AAh
Sequence        MOVWF   EECON2              ; write AAH
                BSF     EECON2,WR           ; start erase (CPU stall)
                NOP
                BSF     INTCON,GIE          ; re-enable interrupts
```


© 2003 Microchip Technology Inc.

## 6.5 Writing to Flash Program Memory

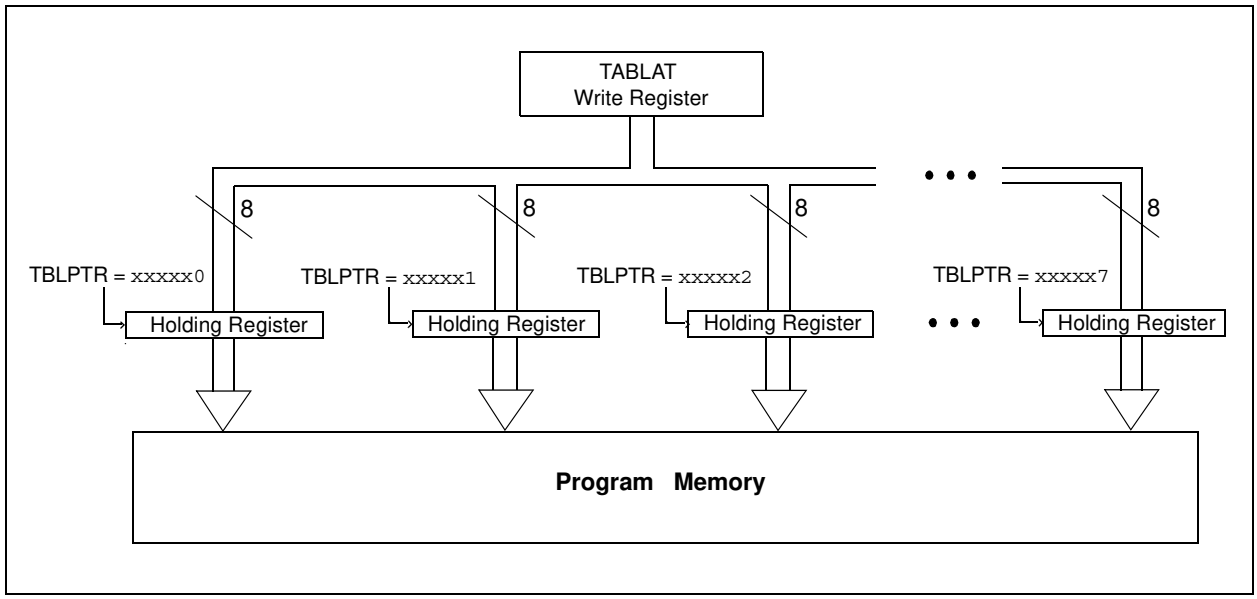
The programming block size is 4 words or 8 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 8 holding registers used by the table writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction has to be executed 8 times for each programming operation. All of the table write operations will essentially be short writes because only the holding registers are written. At the end of updating 8 registers, the EECON1 register must be written to, to start the programming operation with a long write.

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

**FIGURE 6-5: TABLE WRITES TO FLASH PROGRAM MEMORY**




### 6.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. Read 64 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer with address being erased.
4. Do the row erase procedure (see **Section 6.4.1 "Flash Program Memory Erase Sequence"**).
5. Load Table Pointer with address of first byte being written.
6. Write the first 8 bytes into the holding registers with auto-increment.
7. Set the EECON1 register for the write operation:
   - set EEPGD bit to point to program memory;
   - clear the CFGS bit to access program memory;
   - set WREN bit to enable byte writes.
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write AAh to EECON2.
11. Set the WR bit. This will begin the write cycle.
12. The CPU will stall for duration of the write (about 2 ms using internal timer).
13. Execute a NOP.
14. Re-enable interrupts.
15. Repeat steps 6-14 seven times, to write 64 bytes.
16. Verify the memory (table read).

This procedure will require about 18 ms to update one row of 64 bytes of memory. An example of the required code is given in Example 6-3.

**EXAMPLE 6-3: WRITING TO FLASH PROGRAM MEMORY**

```
                MOVLW   D'64                    ; number of bytes in erase block
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW           ; 6 LSB = 0
                MOVWF   TBLPTRL
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVFW   TABLAT                  ; get data
                MOVWF   POSTINC0                ; store data and increment FSR0
                DECFSZ  COUNTER                 ; done?
                GOTO    READ_BLOCK              ; repeat
MODIFY_WORD
                MOVLW   DATA_ADDR_HIGH          ; point to buffer
                MOVWF   FSR0H
                MOVLW   DATA_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   NEW_DATA_LOW            ; update buffer word and increment FSR0
                MOVWF   POSTINC0
                MOVLW   NEW_DATA_HIGH           ; update buffer word
                MOVWF   INDF0
ERASE_BLOCK
                MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW           ; 6 LSB = 0
                MOVWF   TBLPTRL
                BCF     EECON1,CFGS             ; point to PROG/EEPROM memory
                BSF     EECON1,EEPGD            ; point to Flash program memory
                BSF     EECON1,WREN             ; enable write to memory
                BSF     EECON1,FREE             ; enable Row Erase operation
                BCF     INTCON,GIE              ; disable interrupts
                MOVLW   55h                     ; Required sequence
                MOVWF   EECON2                  ; write 55H
                MOVLW   AAh
                MOVWF   EECON2                  ; write AAH
                BSF     EECON1,WR               ; start erase (CPU stall)
                NOP
                BSF     INTCON,GIE              ; re-enable interrupts
WRITE_BUFFER_BACK
                MOVLW   8                       ; number of write buffer groups of 8 bytes
                MOVWF   COUNTER_HI
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
PROGRAM_LOOP
                MOVLW   8                       ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_WORD_TO_HREGS
                MOVFW   POSTINC0                ; get low byte of buffer data and increment FSR0
                MOVWF   TABLAT                  ; present data to table latch
                TBLWT+*                         ; short write
                                                ; to internal TBLWT holding register, increment
                                                  TBLPTR
                DECFSZ  COUNTER                 ; loop until buffers are full
                GOTO    WRITE_WORD_TO_HREGS
```


© 2003 Microchip Technology Inc.

**EXAMPLE 6-3: WRITING TO FLASH PROGRAM MEMORY (CONTINUED)**

```
PROGRAM_MEMORY
                BCF     INTCON,GIE          ; disable interrupts
                MOVLW   55h                 ; required sequence
                MOVWF   EECON2              ; write 55H
                MOVLW   AAh
                MOVWF   EECON2              ; write AAH
                BSF     EECON1,WR           ; start program (CPU stall)
                NOP
                BSF     INTCON,GIE          ; re-enable interrupts
                DECFSZ  COUNTER_HI          ; loop until done
                GOTO    PROGRAM_LOOP
                BCF     EECON1,WREN         ; disable write to memory
```

### 6.5.2 WRITE VERIFY

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

### 6.5.3 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. The WRERR bit is set when a write operation is interrupted by a MCLR Reset, or a WDT Time-out Reset, during normal operation. In these situations, users can check the WRERR bit and rewrite the location.

## 6.6 Flash Program Operation During Code Protection

See **Section 23.0 "Special Features of the CPU"** (**Section 23.5 "Program Verification and Code Protection"**) for details on code protection of Flash program memory.

**TABLE 6-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| TBLPTRU | — | — | bit 21 | Program Memory Table Pointer Upper Byte (TBLPTR<20:16>) | | | | | --00 0000 | --00 0000 |
| TBPLTRH | Program Memory Table Pointer High Byte (TBLPTR<15:8>) | | | | | | | | 0000 0000 | 0000 0000 |
| TBLPTRL | Program Memory Table Pointer High Byte (TBLPTR<7:0>) | | | | | | | | 0000 0000 | 0000 0000 |
| TABLAT | Program Memory Table Latch | | | | | | | | 0000 0000 | 0000 0000 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | — | — |
| EECON1 | EEPGD | CFGS | — | FREE | WRERR | WREN | WR | RD | xx-0 x000 | uu-0 u000 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | ---1 1111 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | ---0 0000 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | ---0 0000 |

**Legend:** x = unknown, u = unchanged, r = reserved, - = unimplemented, read as '0'.
Shaded cells are not used during Flash/EEPROM access.

**NOTES:**


© 2003 Microchip Technology Inc.

## 7.0 DATA EEPROM MEMORY

The data EEPROM is readable and writable during normal operation over the entire VDD range. The data memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers (SFR).

There are four SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADR

The EEPROM data memory allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 256 bytes of data EEPROM with an address range from 00h to FFh.

The EEPROM data memory is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature, as well as from chip to chip. Please refer to parameter D122 (Table 26-1 in **Section 26.0 "Electrical Characteristics"**) for exact limits.

### 7.1 EEADR

The address register can address 256 bytes of data EEPROM.

### 7.2 EECON1 and EECON2 Registers

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit EEPGD determines if the access will be to program or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit CFGS determines if the access will be to the configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access configuration registers. When CFGS is clear, the EEPGD bit selects either program Flash or data EEPROM memory.

The WREN bit enables and disables erase and write operations. When set, erase and write operations are allowed. When clear, erase and write operations are disabled; the WR bit cannot be set while the WREN bit is clear. This mechanism helps to prevent accidental writes to memory due to errant (unexpected) code execution.

Firmware should keep the WREN bit clear at all times except when starting erase or write operations. Once firmware has set the WR bit, the WREN bit may be cleared. Clearing the WREN bit will not affect the operation in progress.

The WRERR bit is set when a write operation is interrupted by a Reset. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR), as these registers have cleared as a result of the Reset.

Control bits, RD and WR, start read and erase/write operations, respectively. These bits are set by firmware and cleared by hardware at the completion of the operation.

The RD bit cannot be set when accessing program memory (EEPGD = 1). Program memory is read using table read instructions. See **Section 6.1 "Table Reads and Table Writes"** regarding table reads.

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

**REGISTER 7-1: EECON1 REGISTER**

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

bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit
1 = Access program Flash memory
0 = Access data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EE or Configuration Select bit
1 = Access configuration or calibration registers
0 = Access program Flash or data EEPROM memory

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

bit 4 **FREE:** Flash Row Erase Enable bit
1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation)
0 = Perform write only

bit 3 **WRERR:** EEPROM Error Flag bit
1 = A write operation was prematurely terminated (MCLR or WDT Reset during self-timed erase or program operation)
0 = The write operation completed normally

**Note:** When a WRERR occurs, the EEPGD or FREE bits are not cleared. This allows tracing of the error condition.

bit 2 **WREN:** Erase/Write Enable bit
1 = Allows erase/write cycles
0 = Inhibits erase/write cycles

bit 1 **WR:** Write Control bit
1 = Initiates a data EEPROM erase/write cycle or a program memory erase cycle or write cycle. (The operation is self-timed and the bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)
0 = Write cycle is completed

bit 0 **RD:** Read Control bit
1 = Initiates a memory read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1.)
0 = Read completed

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


© 2003 Microchip Technology Inc.

## 7.3 Reading the Data EEPROM Memory

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

## 7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register. The sequence in Example 7-2 must be followed to initiate the write cycle.

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

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

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

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

## 7.5 Write Verify

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

## 7.6 Protection Against Spurious Write

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

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

**EXAMPLE 7-1: DATA EEPROM READ**

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

**EXAMPLE 7-2: DATA EEPROM WRITE**

```
            MOVLW   DATA_EE_ADDR        ;
            MOVWF   EEADR               ; Data Memory Address to write
            MOVLW   DATA_EE_DATA        ;
            MOVWF   EEDATA              ; Data Memory Value to write
            BCF     EECON1, EEPGD       ; Point to DATA memory
            BSF     EECON1, WREN        ; Enable writes
            BCF     INTCON, GIE         ; Disable Interrupts
            MOVLW   55h                 ;
Required    MOVWF   EECON2              ; Write 55h
Sequence    MOVLW   AAh                 ;
            MOVWF   EECON2              ; Write AAh
            BSF     EECON1, WR          ; Set WR bit to begin write
            BSF     INTCON, GIE         ; Enable Interrupts

            SLEEP                       ; Wait for interrupt to signal write complete
            BCF     EECON1, WREN        ; Disable writes
```

## 7.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect bits in configuration words. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal Data EEPROM regardless of the state of the code-protect configuration bit. Refer to **Section 23.0 "Special Features of the CPU"** for additional information.

## 7.8 Using the Data EEPROM

The data EEPROM is a high-endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124 or D124A. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 7-3.

| **Note:** | If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124 or D124A. |
|---|---|

**EXAMPLE 7-3: DATA EEPROM REFRESH ROUTINE**

```
        CLRF    EEADR                   ; Start at address 0
        BCF     EECON1, CFGS            ; Set for memory
        BCF     EECON1, EEPGD           ; Set for Data EEPROM
        BCF     INTCON, GIE             ; Disable interrupts
        BSF     EECON1, WREN            ; Enable writes
LOOP                                    ; Loop to refresh array
        BSF     EECON1, RD              ; Read current address
        MOVLW   55h                     ;
        MOVWF   EECON2                  ; Write 55h
        MOVLW   AAh                     ;
        MOVWF   EECON2                  ; Write AAh
        BSF     EECON1, WR              ; Set WR bit to begin write
        BTFSC   EECON1, WR              ; Wait for write to complete
        BRA     $-2
        INCFSZ  EEADR, F                ; Increment address
        BRA     Loop                    ; Not zero, do it again

        BCF     EECON1, WREN            ; Disable writes
        BSF     INTCON, GIE             ; Enable interrupts
```

**TABLE 7-1: REGISTERS ASSOCIATED WITH DATA EEPROM MEMORY**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| EEADR | EEPROM Address Register | | | | | | | | 0000 0000 | 0000 0000 |
| EEDATA | EEPROM Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | — | — |
| EECON1 | EEPGD | CFGS | — | FREE | WRERR | WREN | WR | RD | xx-0 x000 | uu-0 u000 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | ---1 1111 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | ---0 0000 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | ---0 0000 |

**Legend:** x = unknown, u = unchanged, r = reserved, - = unimplemented, read as '0'.
Shaded cells are not used during Flash/EEPROM access.


© 2003 Microchip Technology Inc.

# 8.0 8 X 8 HARDWARE MULTIPLIER

## 8.1 Introduction

An 8 x 8 hardware multiplier is included in the ALU of the PIC18F2X20/4X20 devices. By making the multiply a hardware operation, it completes in a single instruction cycle. This is an unsigned multiply that gives a 16-bit result. The result is stored into the 16-bit product register pair (PRODH:PRODL). The multiplier does not affect any flags in the Status register.

Making the 8 x 8 multiplier execute in a single-cycle gives the following advantages:

- Higher computational throughput
- Reduces code size requirements for multiply algorithms

The performance increase allows the device to be used in applications previously reserved for Digital Signal Processors.

Table 8-1 shows a performance comparison between enhanced devices using the single-cycle hardware multiply and performing the same function without the hardware multiply.

**TABLE 8-1: PERFORMANCE COMPARISON**

| Routine | Multiply Method | Program Memory (Words) | Cycles (Max) | Time | | |
|---|---|---|---|---|---|---|
| | | | | @ 40 MHz | @ 10 MHz | @ 4 MHz |
| 8 x 8 unsigned | Without hardware multiply | 13 | 69 | 6.9 µs | 27.6 µs | 69 µs |
| | Hardware multiply | 1 | 1 | 100 ns | 400 ns | 1 µs |
| 8 x 8 signed | Without hardware multiply | 33 | 91 | 9.1 µs | 36.4 µs | 91 µs |
| | Hardware multiply | 6 | 6 | 600 ns | 2.4 µs | 6 µs |
| 16 x 16 unsigned | Without hardware multiply | 21 | 242 | 24.2 µs | 96.8 µs | 242 µs |
| | Hardware multiply | 28 | 28 | 2.8 µs | 11.2 µs | 28 µs |
| 16 x 16 signed | Without hardware multiply | 52 | 254 | 25.4 µs | 102.6 µs | 254 µs |
| | Hardware multiply | 35 | 40 | 4.0 µs | 16.0 µs | 40 µs |

## 8.2 Operation

Example 8-1 shows the sequence to do an 8 x 8 unsigned multiply. Only one instruction is required when one argument of the multiply is already loaded in the WREG register.

Example 8-2 shows the sequence to do an 8 x 8 signed multiply. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

**EXAMPLE 8-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE**

```
MOVF        ARG1, W     ;
MULWF       ARG2        ; ARG1 * ARG2 ->
                        ;   PRODH:PRODL
```

**EXAMPLE 8-2: 8 x 8 SIGNED MULTIPLY ROUTINE**

```
MOVF        ARG1, W
MULWF       ARG2        ; ARG1 * ARG2 ->
                        ; PRODH:PRODL
BTFSC       ARG2, SB    ; Test Sign Bit
SUBWF       PRODH, F    ; PRODH = PRODH
                        ;         - ARG1
MOVF        ARG2, W
BTFSC       ARG1, SB    ; Test Sign Bit
SUBWF       PRODH, F    ; PRODH = PRODH
                        ;         - ARG2
```

Example 8-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 8-1 shows the algorithm that is used. The 32-bit result is stored in four registers, RES3:RES0.

**EQUATION 8-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM**

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

**EXAMPLE 8-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE**

```
    MOVF    ARG1L, W
    MULWF   ARG2L         ; ARG1L * ARG2L ->
                          ; PRODH:PRODL
    MOVFF   PRODH, RES1   ;
    MOVFF   PRODL, RES0   ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H         ; ARG1H * ARG2H ->
                          ; PRODH:PRODL
    MOVFF   PRODH, RES3   ;
    MOVFF   PRODL, RES2   ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H         ; ARG1L * ARG2H ->
                          ; PRODH:PRODL
    MOVF    PRODL, W      ;
    ADDWF   RES1, F       ; Add cross
    MOVF    PRODH, W      ; products
    ADDWFC  RES2, F       ;
    CLRF    WREG          ;
    ADDWFC  RES3, F       ;
;
    MOVF    ARG1H, W      ;
    MULWF   ARG2L         ; ARG1H * ARG2L ->
                          ; PRODH:PRODL
    MOVF    PRODL, W      ;
    ADDWF   RES1, F       ; Add cross
    MOVF    PRODH, W      ; products
    ADDWFC  RES2, F       ;
    CLRF    WREG          ;
    ADDWFC  RES3, F       ;
```

Example 8-4 shows the sequence to do a 16 x 16 signed multiply. Equation 8-2 shows the algorithm used. The 32-bit result is stored in four registers, RES3:RES0. To account for the sign bits of the arguments, each argument pairs' Most Significant bit (MSb) is tested and the appropriate subtractions are done.

**EQUATION 8-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM**

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

**EXAMPLE 8-4: 16 x 16 SIGNED MULTIPLY ROUTINE**

```
    MOVF    ARG1L, W
    MULWF   ARG2L         ; ARG1L * ARG2L ->
                          ; PRODH:PRODL
    MOVFF   PRODH, RES1   ;
    MOVFF   PRODL, RES0   ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H         ; ARG1H * ARG2H ->
                          ; PRODH:PRODL
    MOVFF   PRODH, RES3   ;
    MOVFF   PRODL, RES2   ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H         ; ARG1L * ARG2H ->
                          ; PRODH:PRODL
    MOVF    PRODL, W      ;
    ADDWF   RES1, F       ; Add cross
    MOVF    PRODH, W      ; products
    ADDWFC  RES2, F       ;
    CLRF    WREG          ;
    ADDWFC  RES3, F       ;
;
    MOVF    ARG1H, W      ;
    MULWF   ARG2L         ; ARG1H * ARG2L ->
                          ; PRODH:PRODL
    MOVF    PRODL, W      ;
    ADDWF   RES1, F       ; Add cross
    MOVF    PRODH, W      ; products
    ADDWFC  RES2, F       ;
    CLRF    WREG          ;
    ADDWFC  RES3, F       ;
;
    BTFSS   ARG2H, 7      ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1     ; no, check ARG1
    MOVF    ARG1L, W      ;
    SUBWF   RES2          ;
    MOVF    ARG1H, W      ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7      ; ARG1H:ARG1L neg?
    BRA     CONT_CODE     ; no, done
    MOVF    ARG2L, W      ;
    SUBWF   RES2          ;
    MOVF    ARG2H, W      ;
    SUBWFB  RES3
;
CONT_CODE
   :
```


© 2003 Microchip Technology Inc.

## 9.0 INTERRUPTS

The PIC18F2320/4320 devices have multiple interrupt sources and an interrupt priority feature that allows each interrupt source to be assigned a high priority level or a low priority level. The high priority interrupt vector is at 000008h and the low priority interrupt vector is at 000018h. High priority interrupt events will interrupt any low priority interrupts that may be in progress.

There are ten registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2
- PIE1, PIE2
- IPR1, IPR2

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/ compiler to automatically take care of the placement of these bits within the specified register.

In general, each interrupt source has three bits to control its operation. The functions of these bits are:

- Flag bit to indicate that an interrupt event occurred
- Enable bit that allows program execution to branch to the interrupt vector address when the flag bit is set
- Priority bit to select high priority or low priority (most interrupt sources have priority bits)

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h, depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PICmicro® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit which enables/disables all interrupt sources. All interrupts branch to address 000008h in Compatibility mode.

When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High priority interrupt sources can interrupt a low priority interrupt. Low priority interrupts are not processed while high priority interrupts are in progress.

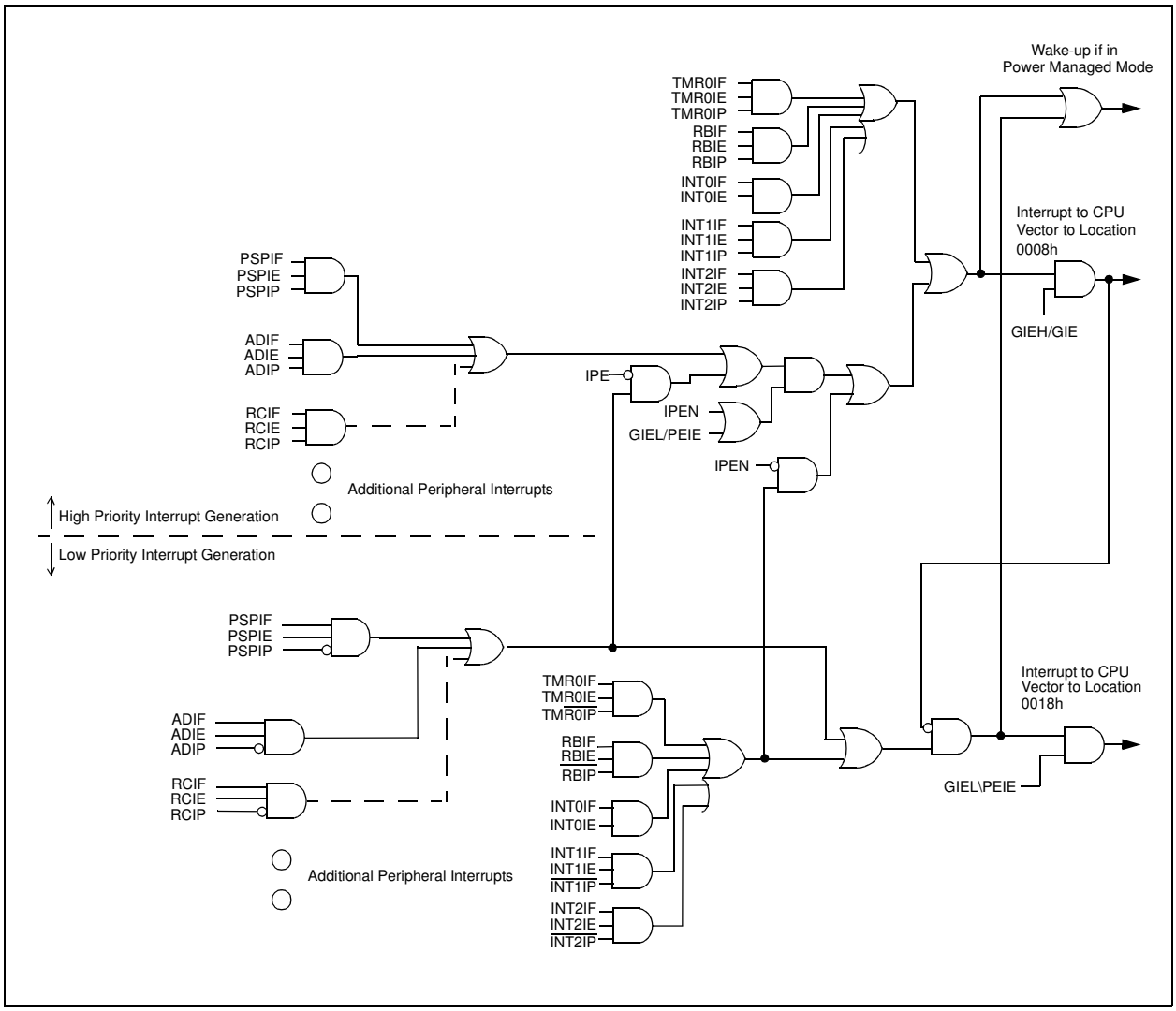
The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used) which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set regardless of the status of their corresponding enable bit or the GIE bit.

| **Note:** | Do not use the `MOVFF` instruction to modify any of the interrupt control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior. |
|---|---|

**FIGURE 9-1: INTERRUPT LOGIC**





© 2003 Microchip Technology Inc.

## 9.1 INTCON Registers

The INTCON registers are readable and writable registers which contain various enable, priority and flag bits.

**Note:** Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

**REGISTER 9-1: INTCON REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-x |
|---|---|---|---|---|---|---|---|
| GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF |
| bit 7 | | | | | | | bit 0 |

bit 7 **GIE/GIEH:** Global Interrupt Enable bit

When IPEN = 0:

1 = Enables all unmasked interrupts

0 = Disables all interrupts

When IPEN = 1:

1 = Enables all high priority interrupts

0 = Disables all high priority interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit

When IPEN = 0:

1 = Enables all unmasked peripheral interrupts

0 = Disables all peripheral interrupts

When IPEN = 1:

1 = Enables all low priority peripheral interrupts

0 = Disables all low priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit

1 = Enables the TMR0 overflow interrupt

0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit

1 = Enables the INT0 external interrupt

0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit

1 = Enables the RB port change interrupt

0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit

1 = TMR0 register has overflowed (must be cleared in software)

0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit

1 = The INT0 external interrupt occurred (must be cleared in software)

0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit

1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)

0 = None of the RB7:RB4 pins have changed state

**Note:** A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

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

**REGISTER 9-2: INTCON2 REGISTER**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | U-0 | R/W-1 | U-0 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{RBPU}}$ | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **INTEDG0:** External Interrupt0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

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

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 0 **RBIP:** RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

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

**Note:** Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.


© 2003 Microchip Technology Inc.

**REGISTER 9-3: INTCON3 REGISTER**

| R/W-1 | R/W-1 | U-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF |
| bit 7 | | | | | | | bit 0 |

bit 7 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

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

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt did not occur

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

**Note:** Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

## 9.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Flag registers (PIR1, PIR2).

**Note 1:** Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>).

**2:** User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

**REGISTER 9-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1**

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

bit 7 **PSPIF(1):** Parallel Slave Port Read/Write Interrupt Flag bit
1 = A read or a write operation has taken place (must be cleared in software)
0 = No read or write has occurred

**Note 1:** This bit is reserved on PIC18F2X20 devices; always maintain this bit clear.

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

bit 5 **RCIF:** USART Receive Interrupt Flag bit
1 = The USART receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The USART receive buffer is empty

bit 4 **TXIF:** USART Transmit Interrupt Flag bit
1 = The USART transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The USART transmit buffer is full

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

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

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

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

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 9-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2**

| R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF |
| bit 7 | | | | | | | bit 0 |

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

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

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

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

bit 3 **BCLIF:** Bus Collision Interrupt Flag bit
1 = A bus collision occurred (must be cleared in software)
0 = No bus collision occurred

bit 2 **LVDIF:** Low-Voltage Detect Interrupt Flag bit
1 = A low-voltage condition occurred (must be cleared in software)
0 = The device voltage is above the Low-Voltage Detect trip point

bit 1 **TMR3IF:** TMR3 Overflow Interrupt Flag bit
1 = TMR3 register overflowed (must be cleared in software)
0 = TMR3 register did not overflow

bit 0 **CCP2IF:** CCPx Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

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

## 9.3 PIE Registers

The PIE registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Enable registers (PIE1, PIE2). When IPEN = 0, the PEIE bit must be set to enable any of these peripheral interrupts.

**REGISTER 9-6: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1**

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

bit 7 **PSPIE(1):** Parallel Slave Port Read/Write Interrupt Enable bit
1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

**Note 1:** This bit is reserved on PIC18F2X20 devices; always maintain this bit clear.

bit 6 **ADIE:** A/D Converter Interrupt Enable bit
1 = Enables the A/D interrupt
0 = Disables the A/D interrupt

bit 5 **RCIE:** USART Receive Interrupt Enable bit
1 = Enables the USART receive interrupt
0 = Disables the USART receive interrupt

bit 4 **TXIE:** USART Transmit Interrupt Enable bit
1 = Enables the USART transmit interrupt
0 = Disables the USART transmit interrupt

bit 3 **SSPIE:** Master Synchronous Serial Port Interrupt Enable bit
1 = Enables the MSSP interrupt
0 = Disables the MSSP interrupt

bit 2 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enables the CCP1 interrupt
0 = Disables the CCP1 interrupt

bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enables the TMR2 to PR2 match interrupt
0 = Disables the TMR2 to PR2 match interrupt

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enables the TMR1 overflow interrupt
0 = Disables the TMR1 overflow interrupt

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 9-7: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2**

| R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE |
| bit 7 | | | | | | | bit 0 |

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

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

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

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

bit 3 **BCLIE:** Bus Collision Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **LVDIE:** Low-Voltage Detect Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR3IE:** TMR3 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **CCP2IE:** CCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

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

## 9.4 IPR Registers

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

**REGISTER 9-8: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP |
| bit 7 | | | | | | | bit 0 |

bit 7 **PSPIP(1):** Parallel Slave Port Read/Write Interrupt Priority bit
1 = High priority
0 = Low priority

**Note 1:** This bit is reserved on PIC18F2X20 devices; always maintain this bit set.

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

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

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

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

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

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

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

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 9-9: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2**

| R/W-1 | R/W-1 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP |
| bit 7 | | | | | | | bit 0 |

bit 7 **OSCFIP:** Oscillator Fail Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **CMIP:** Comparator Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 4 **EEIP:** Data EEPROM/Flash Write Operation Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **BCLIP:** Bus Collision Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **LVDIP:** Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR3IP:** TMR3 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

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

## 9.5 RCON Register

The RCON register contains bits used to determine the cause of the last Reset or wake-up from power managed mode. RCON also contains the bit that enables interrupt priorities (IPEN).

**REGISTER 9-10: RCON REGISTER**

| R/W-0 | U-0 | U-0 | R/W-1 | R-1 | R-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ |
| bit 7 | | | | | | | bit 0 |

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

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

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Cleared by execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

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


© 2003 Microchip Technology Inc.

## 9.6 INTn Pin Interrupts

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge triggered: either rising if the corresponding INTEDGx bit is set in the INTCON2 register, or falling if the INTEDGx bit is clear. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit, INTxF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxE. Flag bit, INTxF, must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt. All external interrupts (INT0, INT1 and INT2) can wake-up the processor from the power managed modes if bit INTxE was set prior to going into power managed modes. If the global interrupt enable bit GIE is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the Interrupt Priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high priority interrupt source.

## 9.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow (FFh → 00h) in the TMR0 register will set flag bit TMR0IF. In 16-bit mode, an overflow (FFFFh → 0000h) in the TMR0H:TMR0L registers will set flag bit TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See **Section 11.0 "Timer0 Module"** for further details on the Timer0 module.

## 9.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

## 9.9 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, Status and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (See **Section 5.3 "Fast Register Stack"**), the user may need to save the WREG, Status and BSR registers on entry to the Interrupt Service Routine. Depending on the user's application, other registers may also need to be saved. Example 9-1 saves and restores the WREG, Status and BSR registers during an Interrupt Service Routine.

**EXAMPLE 9-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM**

```
MOVWF   W_TEMP                  ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP     ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP           ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR           ; Restore BSR
MOVF    W_TEMP, W               ; Restore WREG
MOVFF   STATUS_TEMP, STATUS     ; Restore STATUS
```

**NOTES:**


© 2003 Microchip Technology Inc.

## 10.0 I/O PORTS

Depending on the device selected and features enabled, there are up to five ports available. Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

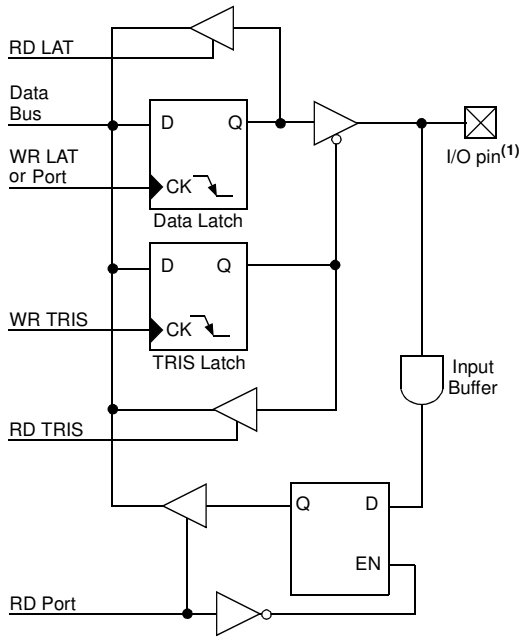
Each port has three registers for its operation. These registers are:

- TRIS register (Data Direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (Data Latch)

The Data Latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A simplified model of a generic I/O port without the interfaces to other peripherals is shown in Figure 10-1.

**FIGURE 10-1: GENERIC I/O PORT OPERATION**




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

## 10.1 PORTA, TRISA and LATA Registers

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

Reading the PORTA register reads the status of the pins, whereas writing to it, will write to the port latch.

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

The RA4 pin is multiplexed with the Timer0 module clock input and one of the comparator outputs to become the RA4/T0CKI/C1OUT pin. Pins RA6 and RA7 are multiplexed with the main oscillator pins; they are enabled as oscillator or I/O pins by the selection of the main oscillator in Configuration Register 1H (see **Section 23.1 "Configuration Bits"** for details). When they are not used as port pins, RA6 and RA7 and their associated TRIS and LAT bits are read as '0'.

The other PORTA pins are multiplexed with analog inputs, the analog VREF+ and VREF- inputs and the comparator voltage reference output. The operation of pins, RA3:RA0 and RA5, as A/D converter inputs is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register 1). Pins RA0 through RA5 may also be used as comparator inputs or outputs by setting the appropriate bits in the CMCON register.

| **Note:** | On a Power-on Reset, RA5 and RA3:RA0 are configured as analog inputs and read as '0'. RA4 is configured as a digital input. |
|---|---|

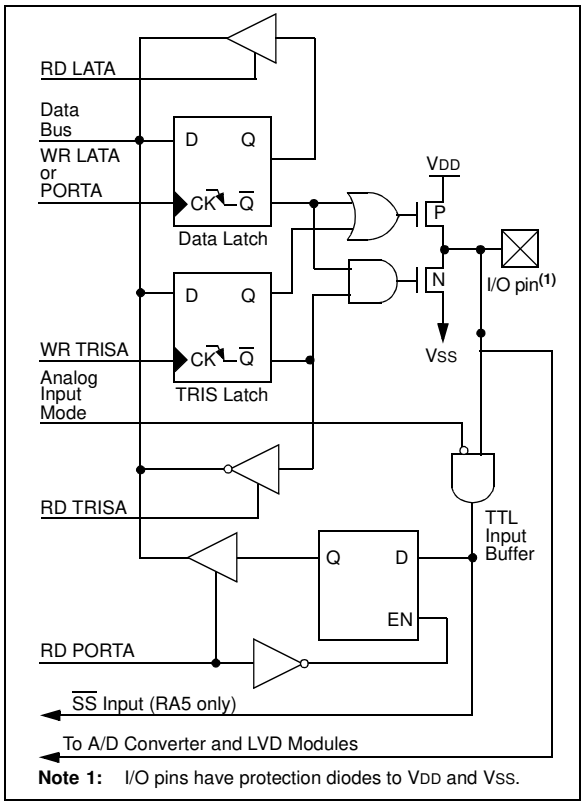
The RA4/T0CKI/C1OUT pin is a Schmitt Trigger input and an open-drain output. All other PORTA pins have TTL input levels and full CMOS output drivers.

The TRISA register controls the direction of the RA pins even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

**EXAMPLE 10-1: INITIALIZING PORTA**

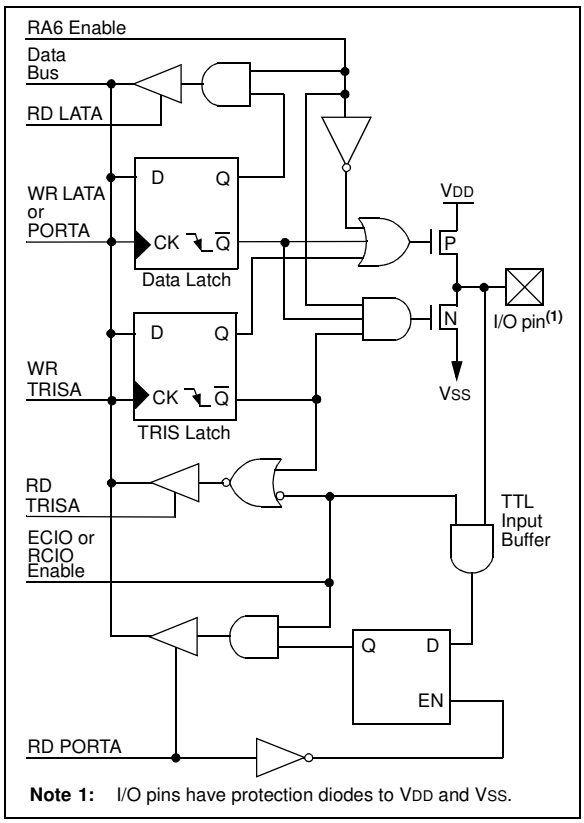
```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x07    ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

**FIGURE 10-2: BLOCK DIAGRAM OF RA3:RA0 AND RA5 PINS**



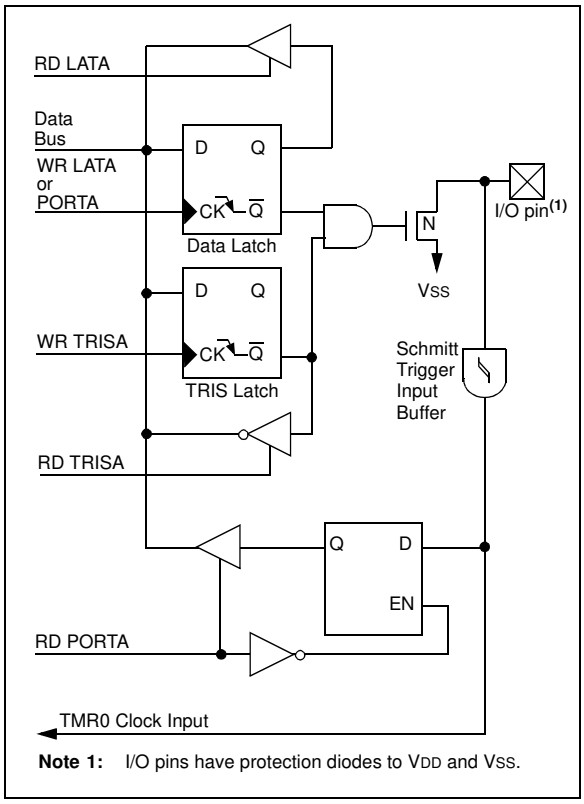

**Note 1:** I/O pins have protection diodes to VDD and VSS.

**FIGURE 10-3: BLOCK DIAGRAM OF RA6 PIN**



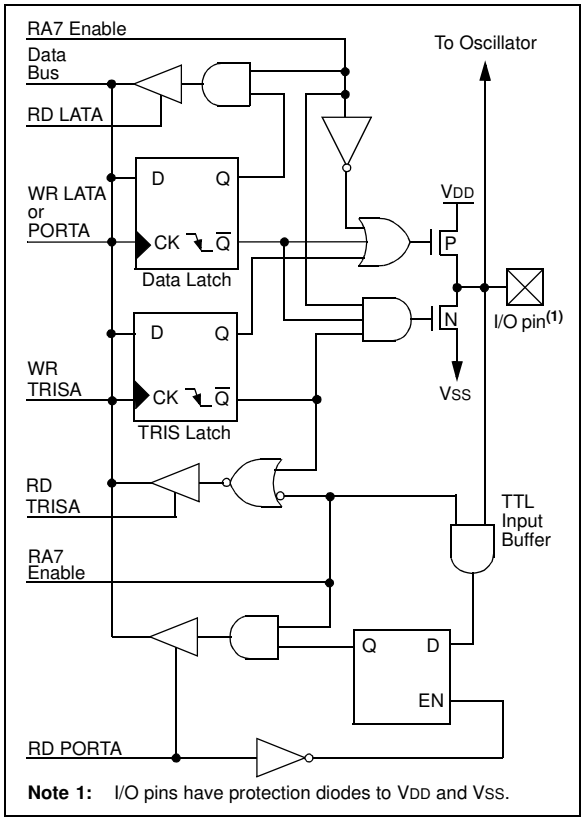

**Note 1:** I/O pins have protection diodes to VDD and VSS.

**FIGURE 10-4: BLOCK DIAGRAM OF RA4/T0CKI PIN**




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

**FIGURE 10-5: BLOCK DIAGRAM OF RA7 PIN**




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


© 2003 Microchip Technology Inc.

**TABLE 10-1: PORTA FUNCTIONS**

| Name | Bit# | Buffer | Function |
|---|---|---|---|
| RA0/AN0 | bit 0 | TTL | Input/output or analog input. |
| RA1/AN1 | bit 1 | TTL | Input/output or analog input. |
| RA2/AN2/VREF-/CVREF | bit 2 | TTL | Input/output, analog input, VREF- or Comparator VREF output. |
| RA3/AN3/VREF+ | bit 3 | TTL | Input/output, analog input or VREF+. |
| RA4/T0CKI/C1OUT | bit 4 | ST | Input/output, external clock input for Timer0 or Comparator 1 output. Output is open-drain type. |
| RA5/AN4/$\overline{SS}$/LVDIN/C2OUT | bit 5 | TTL | Input/output, analog input, Slave Select input for Synchronous Serial Port, Low-Voltage Detect input or Comparator 2 output. |
| OSC2/CLKO/RA6 | bit 6 | TTL | OSC2, clock output or I/O pin. |
| OSC1/CLKI/RA7 | bit 7 | TTL | OSC1, clock input or I/O pin. |

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

**TABLE 10-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTA | RA7(1) | RA6(1) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xx0x 0000 | uu0u 0000 |
| LATA | LATA7(1) | LATA6(1) | LATA Data Latch Register | | | | | | xxxx xxxx | uuuu uuuu |
| TRISA | TRISA7(1) | TRISA6(1) | PORTA Data Direction Register | | | | | | 1111 1111 | 1111 1111 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0000 | --00 0000 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 0000 0111 |
| CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 000- 0000 |

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

**Note 1:** RA7:RA6 and their associated latch and data direction bits are enabled as I/O pins based on oscillator configuration; otherwise, they are read as '0'.

## 10.2 PORTB, TRISB and LATB Registers

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

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

**EXAMPLE 10-2: INITIALIZING PORTB**

```
CLRF    PORTB   ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF    LATB    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x0F    ; Set RB<4:0> as
MOVWF   ADCON1  ; digital I/O pins
                ; (required if config bit
                ; PBADEN is set)
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISB   ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit RBPU (INTCON2<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

**Note:** On a Power-on Reset, RB4:RB0 are configured as analog inputs by default and read as '0'; RB7:RB5 are configured as digital inputs.

By programming the configuration bit, PBADEN (CONFIG3H<1>), RB4:RB0 will alternatively be configured as digital inputs on POR.

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

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

a) Any read or write of PORTB (except with the MOVFF (ANY), PORTB instruction). This will end the mismatch condition.

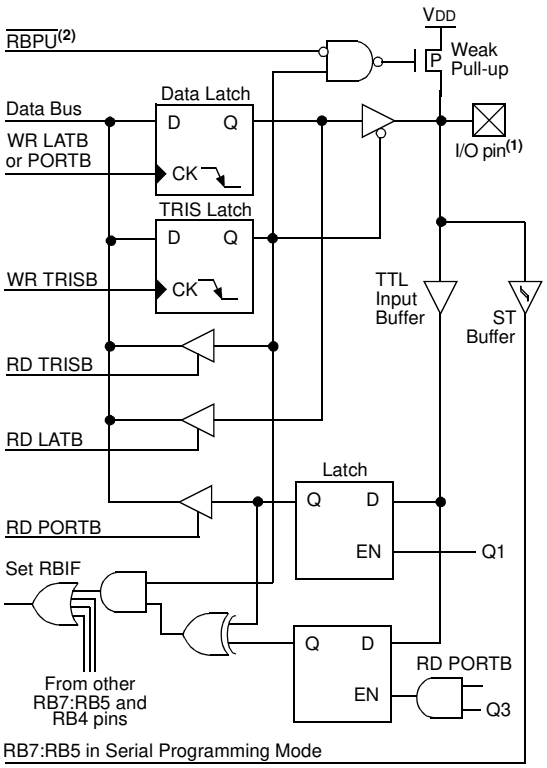
b) Clear flag bit RBIF.

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

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

RB3 can be configured by the configuration bit, CCP2MX, as the alternate peripheral pin for the CCP2 module (CCP2MX = 0).

**FIGURE 10-6: BLOCK DIAGRAM OF RB7:RB5 PINS**

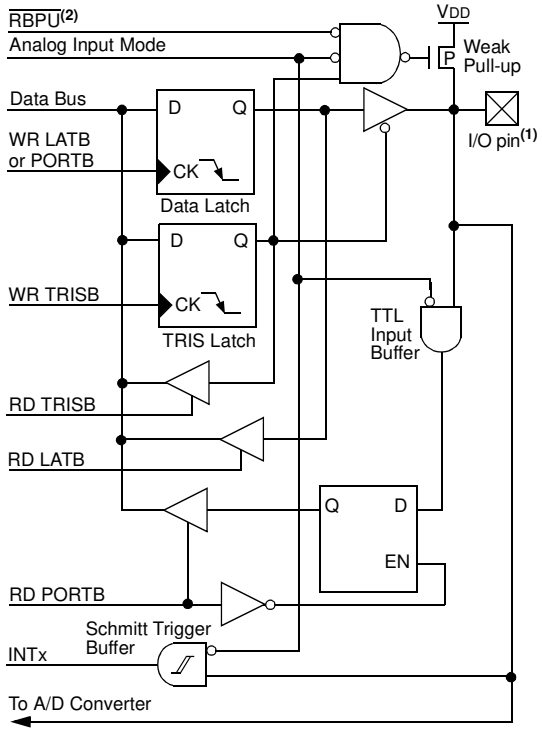



**Note 1:** I/O pins have diode protection to VDD and VSS.

**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit (INTCON2<7>).

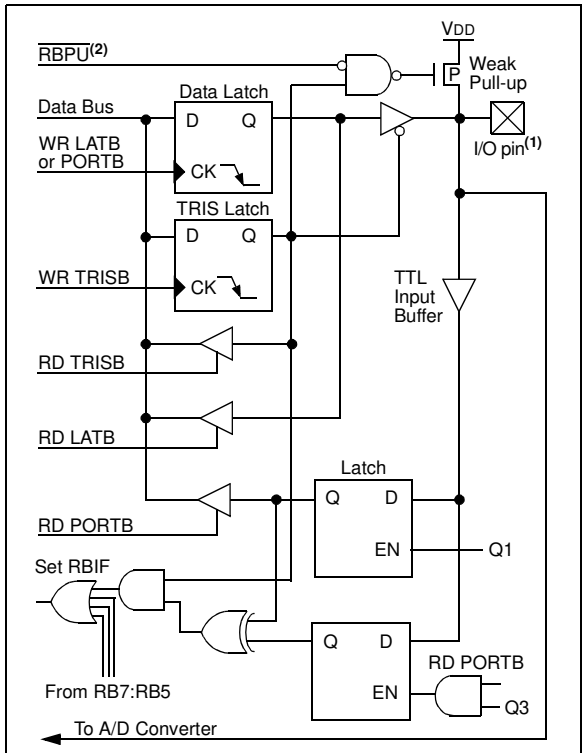

© 2003 Microchip Technology Inc.

**FIGURE 10-7: BLOCK DIAGRAM OF RB2:RB0 PINS**



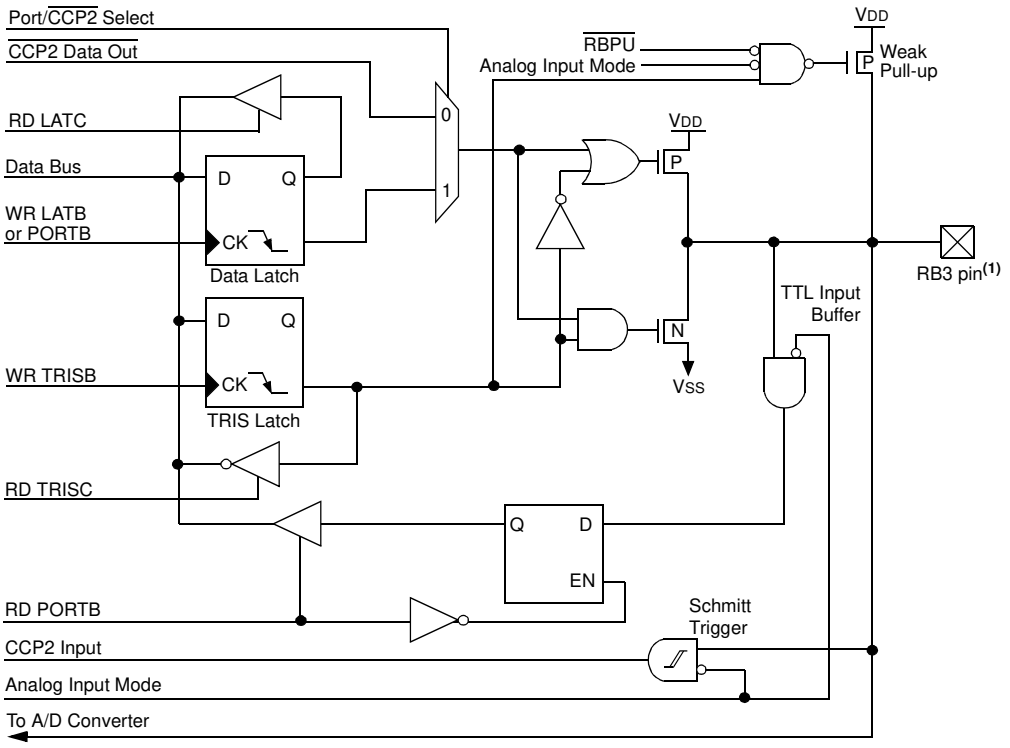

**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit (INTCON2<7>).

**FIGURE 10-8: BLOCK DIAGRAM OF RB4 PIN**




**Note 1:** I/O pins have diode protection to VDD and VSS.
**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the RBPU bit (INTCON2<7>).

**FIGURE 10-9: BLOCK DIAGRAM OF RB3/CCP2 PIN**




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

**TABLE 10-3: PORTB FUNCTIONS**

| Name | Bit# | Buffer | Function |
|---|---|---|---|
| RB0/AN12/INT0 | bit 0 | TTL(1)/ST(2) | Input/output pin, analog input or external interrupt input 0. Internal software programmable weak pull-up. |
| RB1/AN10/INT1 | bit 1 | TTL(1)/ST(2) | Input/output pin, analog input or external interrupt input 1. Internal software programmable weak pull-up. |
| RB2/AN8/INT2 | bit 2 | TTL(1)/ST(2) | Input/output pin, analog input or external interrupt input 2. Internal software programmable weak pull-up. |
| RB3/AN9/CCP2 | bit 3 | TTL(1)/ST(3) | Input/output pin or analog input. Capture2 input/Compare2 output/ PWM output when CCP2MX configuration bit is set(4). Internal software programmable weak pull-up. |
| RB4/AN11/KBI0 | bit 4 | TTL | Input/output pin (with interrupt-on-change) or analog input. Internal software programmable weak pull-up. |
| RB5/KBI1/PGM | bit 5 | TTL/ST(5) | Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Low-voltage ICSP enable pin. |
| RB6/KBI2/PGC | bit 6 | TTL/ST(5) | Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock. |
| RB7/KBI3/PGD | bit 7 | TTL/ST(5) | Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data. |

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

**Note 1:** This buffer is a TTL input when configured as digital I/O.

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

**3:** This buffer is a Schmitt Trigger input when configured as the CCP2 input.

**4:** A device configuration bit selects which I/O pin the CCP2 pin is multiplexed on.

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

**TABLE 10-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxq qqqq | uuuu uuuu |
| LATB | LATB Data Latch Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| INTCON2 | $\overline{\text{RBPU}}$ | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP | 1111 -1-1 | 1111 -1-1 |
| INTCON3 | INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF | 11-0 0-00 | 11-0 0-00 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0000 | --00 0000 |

**Legend:** x = unknown, u = unchanged, q = value depends on condition. Shaded cells are not used by PORTB.


© 2003 Microchip Technology Inc.

## 10.3 PORTC, TRISC and LATC Registers

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

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

PORTC is multiplexed with several peripheral functions (Table 10-5). The pins have Schmitt Trigger input buffers. RC1 is normally configured by configuration bit, CCP2MX (CONFIG3H<0>), as the default peripheral pin of the CCP2 module (default/erased state, CCP2MX = 1).

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTC pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

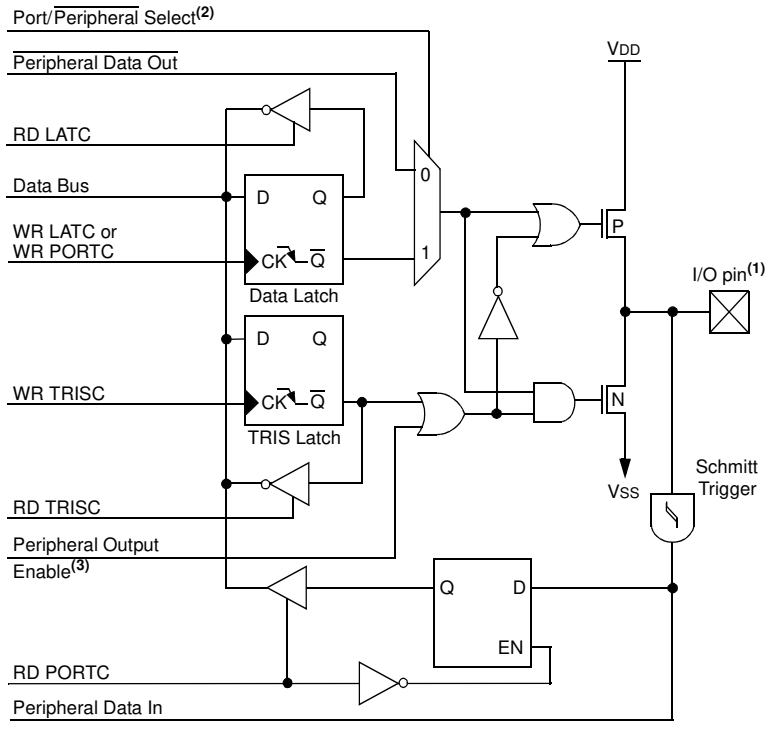
| **Note:** | On a Power-on Reset, these pins are configured as digital inputs. |
|---|---|

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents even though a peripheral device may be overriding one or more of the pins.

**EXAMPLE 10-3: INITIALIZING PORTC**

```
CLRF    PORTC       ; Initialize PORTC by
                    ; clearing output
                    ; data latches
CLRF    LATC        ; Alternate method
                    ; to clear output
                    ; data latches
MOVLW   0xCF        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISC       ; Set RC<3:0> as inputs
                    ; RC<5:4> as outputs
                    ; RC<7:6> as inputs
```

**FIGURE 10-10: PORTC BLOCK DIAGRAM (PERIPHERAL OUTPUT OVERRIDE)**




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

**2:** Port/Peripheral Select signal selects between port data (output) and peripheral output.

**3:** Peripheral Output Enable is only active if Peripheral Select is active.

**TABLE 10-5: PORTC FUNCTIONS**

| Name | Bit# | Buffer Type | Function |
|---|---|---|---|
| RC0/T1OSO/T1CKI | bit 0 | ST | Input/output port pin or Timer1 oscillator output/Timer1 clock input. |
| RC1/T1OSI/CCP2 | bit 1 | ST | Input/output port pin, Timer1 oscillator input or Capture2 input/ Compare2 output/PWM output when CCP2MX configuration bit is disabled. |
| RC2/CCP1/P1A(1) | bit 2 | ST | Input/output port pin, Capture1 input/Compare1 output/PWM1 output or enhanced PWM output A(1). |
| RC3/SCK/SCL | bit 3 | ST | RC3 can also be the synchronous serial clock for both SPI and $I^2C$ modes. |
| RC4/SDI/SDA | bit 4 | ST | RC4 can also be the SPI Data In (SPI mode) or Data I/O ($I^2C$ mode). |
| RC5/SDO | bit 5 | ST | Input/output port pin or Synchronous Serial Port data output. |
| RC6/TX/CK | bit 6 | ST | Input/output port pin, Addressable USART Asynchronous Transmit or Addressable USART Synchronous Clock. |
| RC7/RX/DT | bit 7 | ST | Input/output port pin, Addressable USART Asynchronous Receive or Addressable USART Synchronous Data. |

**Legend:** ST = Schmitt Trigger input

**Note 1:** Enhanced PWM output is available only on PIC18F4X20 devices.

**TABLE 10-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | uuuu uuuu |
| LATC | LATC Data Latch Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |

**Legend:** x = unknown, u = unchanged


© 2003 Microchip Technology Inc.

## 10.4 PORTD, TRISD and LATD Registers

| **Note:** | PORTD is only available on PIC18F4X20 devices. |
|---|---|

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

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

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Three of the PORTD pins are multiplexed with outputs P1B, P1C and P1D of the Enhanced CCP module. The operation of these additional PWM output pins is covered in greater detail in **Section 16.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**.

| **Note:** | On a Power-on Reset, these pins are configured as digital inputs. |
|---|---|

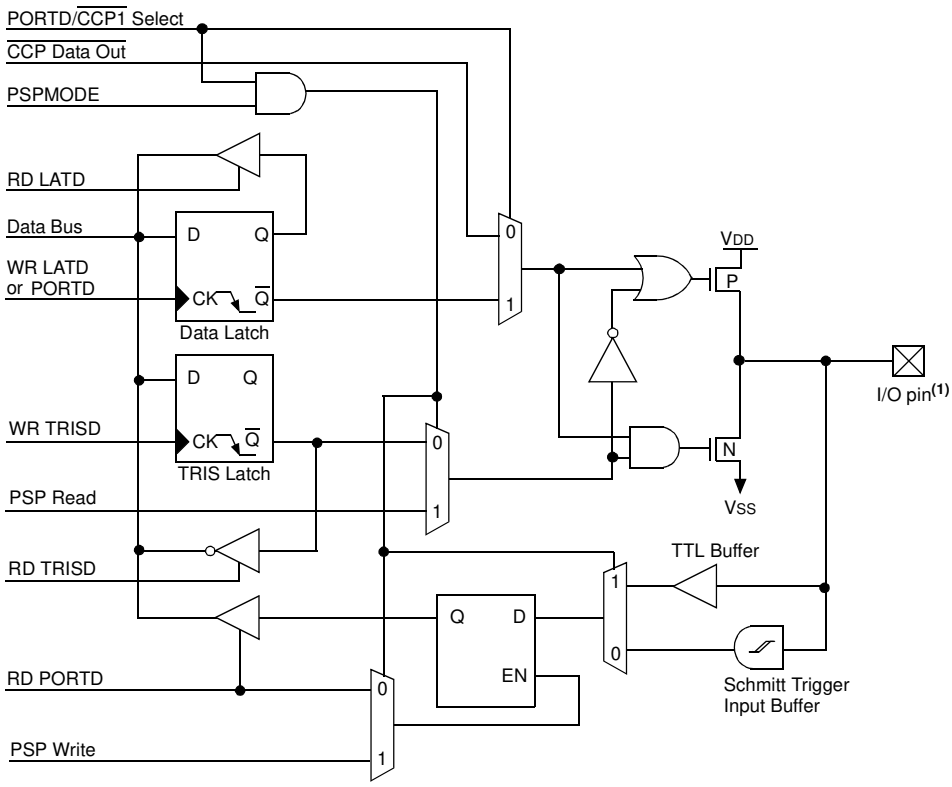
PORTD can also be configured as an 8-bit wide microprocessor port (Parallel Slave Port) by setting control bit, PSPMODE (TRISE<4>). In this mode, the input buffers are TTL. See **Section 10.6 "Parallel Slave Port"** for additional information on the Parallel Slave Port (PSP).

| **Note:** | When the enhanced PWM mode is used with either dual or quad outputs, the PSP functions of PORTD are automatically disabled. |
|---|---|

**EXAMPLE 10-4: INITIALIZING PORTD**

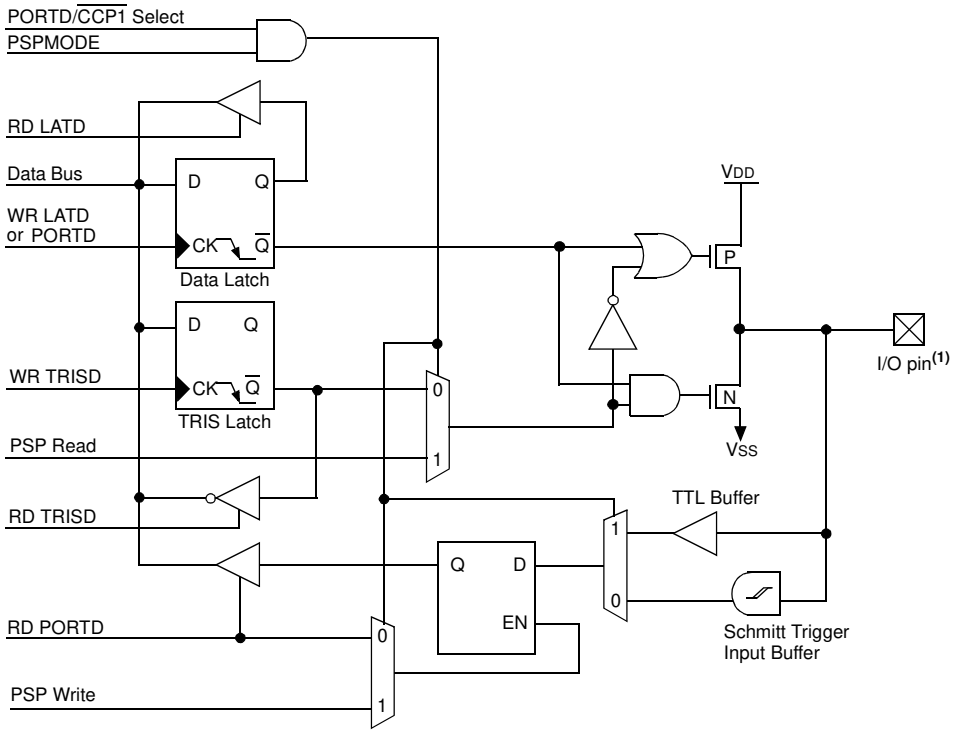
```
CLRF    PORTD    ; Initialize PORTD by
                 ; clearing output
                 ; data latches
CLRF    LATD     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0xCF     ; Value used to
                 ; initialize data
                 ; direction
MOVWF   TRISD    : Set RD<3:0> as inputs
                 ; RD<5:4> as outputs
                 ; RD<7:6> as inputs
```

**FIGURE 10-11: BLOCK DIAGRAM OF RD7:RD5 PINS**




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

**FIGURE 10-12: BLOCK DIAGRAM OF RD4:RD0 PINS**




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

**TABLE 10-7: PORTD FUNCTIONS**

| Name | Bit# | Buffer Type | Function |
|---|---|---|---|
| RD0/PSP0 | bit 0 | ST/TTL(1) | Input/output port pin or Parallel Slave Port bit 0. |
| RD1/PSP1 | bit 1 | ST/TTL(1) | Input/output port pin or Parallel Slave Port bit 1. |
| RD2/PSP2 | bit 2 | ST/TTL(1) | Input/output port pin or Parallel Slave Port bit 2. |
| RD3/PSP3 | bit 3 | ST/TTL(1) | Input/output port pin or Parallel Slave Port bit 3. |
| RD4/PSP4 | bit 4 | ST/TTL(1) | Input/output port pin or Parallel Slave Port bit 4. |
| RD5/PSP5/P1B | bit 5 | ST/TTL(1) | Input/output port pin, Parallel Slave Port bit 5 or enhanced PWM output P1B. |
| RD6/PSP6/P1C | bit 6 | ST/TTL(1) | Input/output port pin, Parallel Slave Port bit 6 or enhanced PWM output P1C. |
| RD7/PSP7/P1D | bit 7 | ST/TTL(1) | Input/output port pin, Parallel Slave Port bit 7 or enhanced PWM output P1D. |

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

**Note 1:** Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in Parallel Slave Port mode.

**TABLE 10-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTD | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | xxxx xxxx | uuuu uuuu |
| LATD | LATD Data Latch Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISD | PORTD Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| TRISE | IBF | OBF | IBOV | PSPMODE | — | PORTE Data Direction bits | | | 0000 -111 | 0000 -111 |
| CCP1CON | P1M1 | P1M0 | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 0000 0000 | 0000 0000 |

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


© 2003 Microchip Technology Inc.

## 10.5 PORTE, TRISE and LATE Registers

Depending on the particular PIC18F2X20/4X20 device selected, PORTE is implemented in two different ways.

For PIC18F4X20 devices, PORTE is a 4-bit wide port. Three pins (RE0/AN5/RD, RE1/AN6/WR and RE2/AN7/CS) are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. When selected as an analog input, these pins will read as '0's.

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

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

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

The upper four bits of the TRISE register also control the operation of the Parallel Slave Port. Their operation is explained in Register 10-1.

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

The fourth pin of PORTE (MCLR/VPP/RE3) is an input only pin. Its operation is controlled by the MCLRE configuration bit in Configuration Register 3H (CONFIG3H<7>). When selected as a port pin (MCLRE = 0), it functions as a digital input only pin; as such, it does not have TRIS or LAT bits associated with its operation. Otherwise, it functions as the device's Master Clear input. In either configuration, RE3 also functions as the programming voltage input during programming.

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

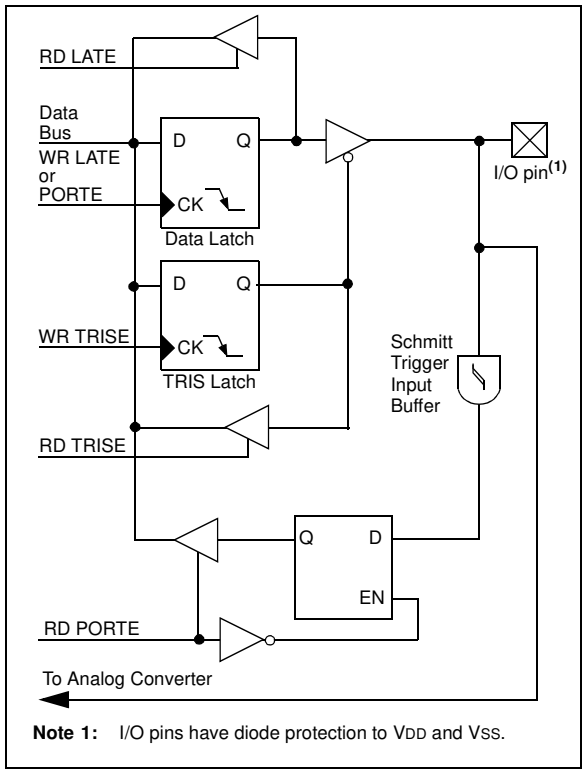
**EXAMPLE 10-5: INITIALIZING PORTE**

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x0A    ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   0x03    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RE<0> as inputs
                ; RE<1> as outputs
                ; RE<2> as inputs
```

### 10.5.1 PORTE IN 28-PIN DEVICES

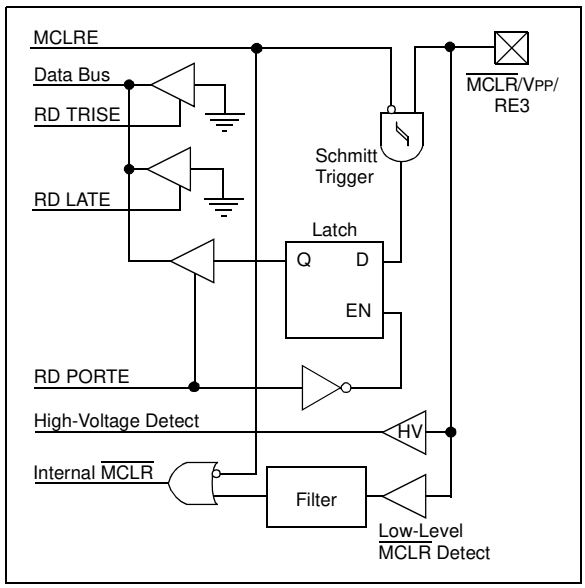
For PIC18F2X20 devices, PORTE is only available when Master Clear functionality is disabled (CONFIG3H<7> = 0). In these cases, PORTE is a single bit, input only port comprised of RE3 only. The pin operates as previously described.

**FIGURE 10-13: BLOCK DIAGRAM OF RE2:RE0 PINS**




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

**FIGURE 10-14: BLOCK DIAGRAM OF MCLR/VPP/RE3 PIN**



**REGISTER 10-1: TRISE REGISTER**

| R-0 | R-0 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **IBF:** Input Buffer Full Status bit
1 = A word has been received and waiting to be read by the CPU
0 = No word has been received

bit 6 **OBF**: Output Buffer Full Status bit
1 = The output buffer still holds a previously written word
0 = The output buffer has been read

bit 5 **IBOV**: Input Buffer Overflow Detect bit (in Microprocessor mode)
1 = A write occurred when a previously input word has not been read (must be cleared in software)
0 = No overflow occurred

bit 4 **PSPMODE**: Parallel Slave Port Mode Select bit
1 = Parallel Slave Port mode
0 = General Purpose I/O mode

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

bit 2 **TRISE2**: RE2 Direction Control bit
1 = Input
0 = Output

bit 1 **TRISE1**: RE1 Direction Control bit
1 = Input
0 = Output

bit 0 **TRISE0**: RE0 Direction Control bit
1 = Input
0 = Output

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


© 2003 Microchip Technology Inc.

**TABLE 10-9: PORTE FUNCTIONS**

| Name | Bit# | Buffer Type | Function |
|---|---|---|---|
| RE0/AN5/$\overline{RD}$ | bit 0 | ST/TTL(1) | Input/output port pin, analog input or read control input in Parallel Slave Port mode.<br>For $\overline{RD}$ (PSP Control mode):<br>1 = PSP is Idle<br>0 = Read operation. Reads PORTD register (if chip selected). |
| RE1/AN6/$\overline{WR}$ | bit 1 | ST/TTL(1) | Input/output port pin, analog input or write control input in Parallel Slave Port mode.<br>For $\overline{WR}$ (PSP Control mode):<br>1 = PSP is Idle<br>0 = Write operation. Writes PORTD register (if chip selected). |
| RE2/AN7/$\overline{CS}$ | bit 2 | ST/TTL(1) | Input/output port pin, analog input or chip select control input in Parallel Slave Port mode.<br>For $\overline{CS}$ (PSP Control mode):<br>1 = PSP is Idle<br>0 = External device is selected |
| $\overline{MCLR}$/VPP/RE3 | bit 3 | ST | Input only port pin or programming voltage input (if $\overline{MCLR}$ is disabled);<br>Master Clear input or programming voltage input (if $\overline{MCLR}$ is enabled). |

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

**Note 1:** Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in Parallel Slave Port mode.

**TABLE 10-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTE | — | — | — | — | RE3(1) | RE2 | RE1 | RE0 | ---- q000 | ---- q000 |
| LATE | — | — | — | — | — | LATE Data Latch Register | | | ---- -xxx | ---- -uuu |
| TRISE | IBF | OBF | IBOV | PSPMODE | — | PORTE Data Direction bits | | | 0000 -111 | 0000 -111 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0000 | --00 0000 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends on condition.
Shaded cells are not used by PORTE.

**Note 1:** Implemented only when Master Clear functionality is disabled (CONFIG3H<7> = 0).

## 10.6 Parallel Slave Port

**Note:** The Parallel Slave Port is only available on PIC18F4X20 devices.

In addition to its function as a general I/O port, PORTD can also operate as an 8-bit wide Parallel Slave Port (PSP) or microprocessor port. PSP operation is controlled by the 4 upper bits of the TRISE register (Register 10-1). Setting control bit, PSPMODE (TRISE<4>), enables PSP operation, as long as the Enhanced CCP module is not operating in dual output or quad output PWM mode. In Slave mode, the port is asynchronously readable and writable by the external world.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an 8-bit latch. Setting the control bit, PSPMODE, enables the PORTE I/O pins to become control inputs for the microprocessor port. When set, port pin RE0 is the $\overline{RD}$ input, RE1 is the $\overline{WR}$ input and RE2 is the $\overline{CS}$ (Chip Select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (set). The A/D port configuration bits PFCG3:PFCG0 (ADCON1<3:0>) must also be set to '1010'.

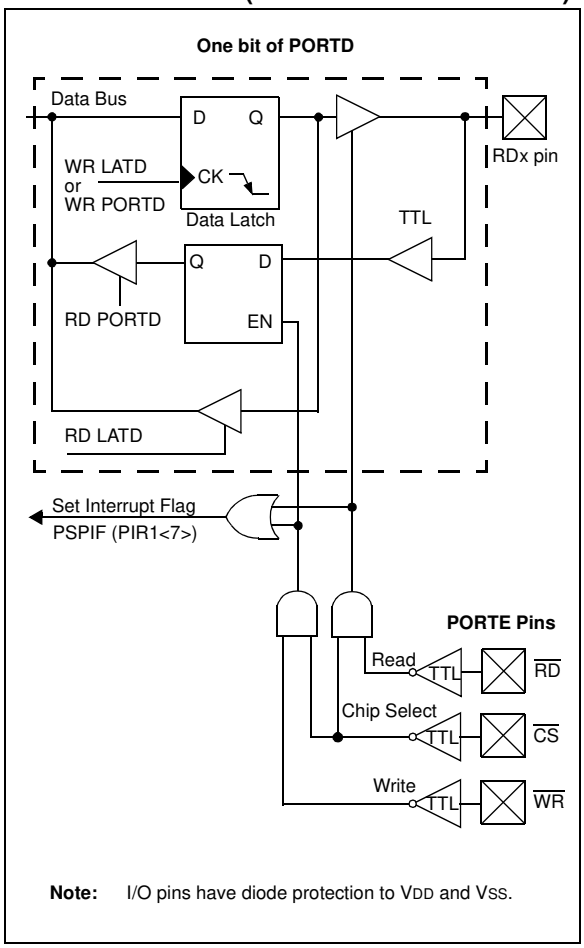
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits are both set when the write ends.

A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low. The data in PORTD is read out and the OBF bit is set. If the user writes new data to PORTD to set OBF, the data is immediately read out; however, the OBF bit is not set.

When either the $\overline{CS}$ or $\overline{RD}$ lines are detected high, the PORTD pins return to the input state and the PSPIF bit is set. User applications should wait for PSPIF to be set before servicing the PSP; when this happens, the IBF and OBF bits can be polled and the appropriate action taken.

The timing for the control signals in Write and Read modes is shown in Figure 10-16 and Figure 10-17, respectively.

**FIGURE 10-15: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)**




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


© 2003 Microchip Technology Inc.

**FIGURE 10-16: PARALLEL SLAVE PORT WRITE WAVEFORMS**

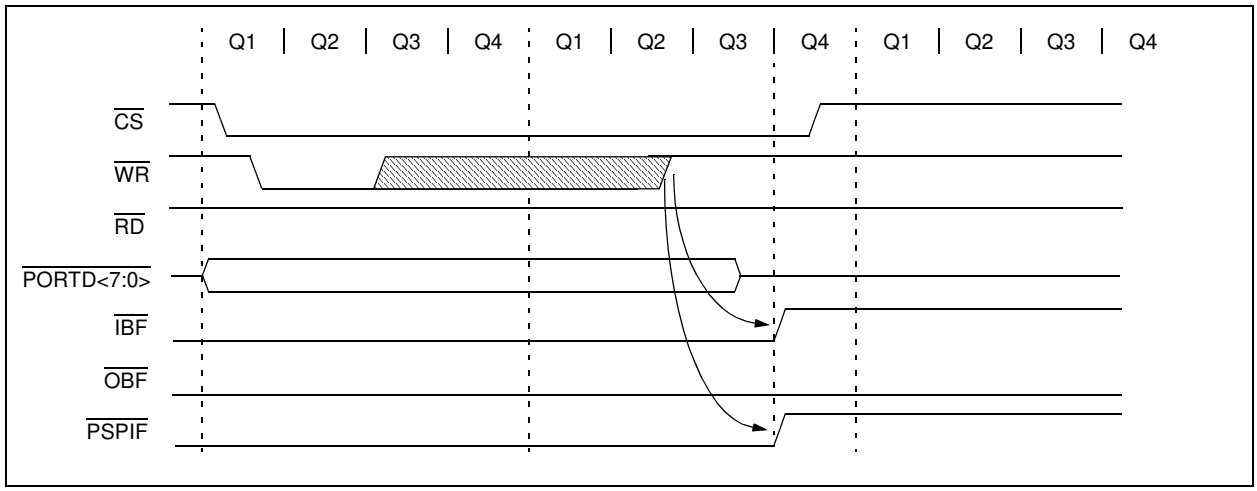



**FIGURE 10-17: PARALLEL SLAVE PORT READ WAVEFORMS**

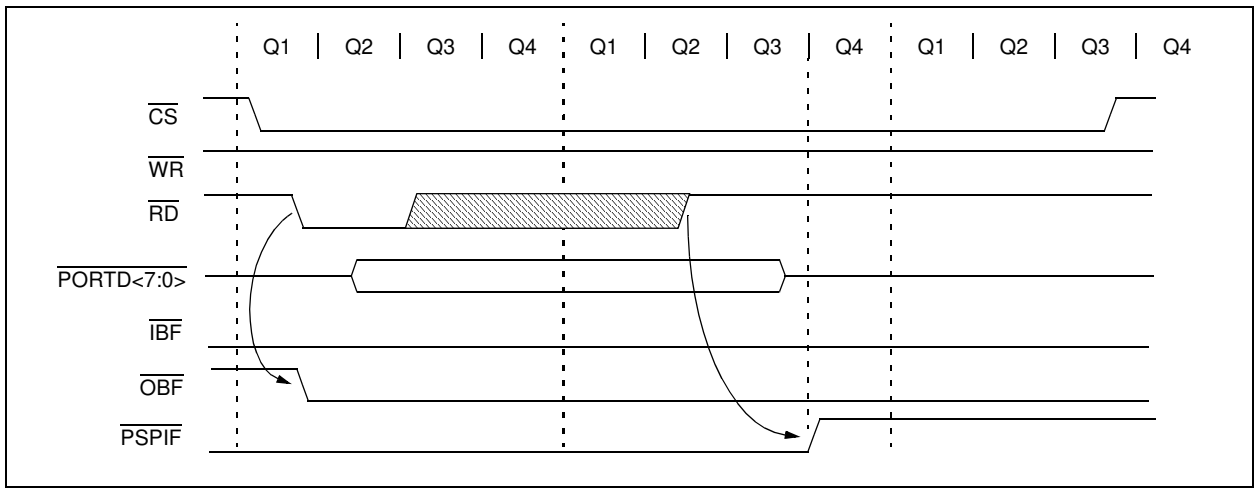



**TABLE 10-11: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| PORTD | Port Data Latch when written; Port pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| LATD | LATD Data Latch bits | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISD | PORTD Data Direction bits | | | | | | | | 1111 1111 | 1111 1111 |
| PORTE | — | — | — | — | RE3 | RE2 | RE1 | RE0 | ---- 0000 | ---- 0000 |
| LATE | — | — | — | — | — | LATE Data Latch bits | | | ---- -xxx | ---- -uuu |
| TRISE | IBF | OBF | IBOV | PSPMODE | — | PORTE Data Direction bits | | | 0000 -111 | 0000 -111 |
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IF | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0000 | --00 0000 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Parallel Slave Port.

**NOTES:**


© 2003 Microchip Technology Inc.

## 11.0 TIMER0 MODULE

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler
- Clock source selectable to be external or internal
- Interrupt-on-overflow from FFh to 00h in 8-bit mode and FFFFh to 0000h in 16-bit mode
- Edge select for external clock

Figure 11-1 shows a simplified block diagram of the Timer0 module in 8-bit mode and Figure 11-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

The T0CON register (Register 11-1) is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

**REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| TMR0ON | T08BIT | T0CS | T0SE | PSA | T0PS2 | T0PS1 | T0PS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT:** Timer0 8-bit/16-bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS:** Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

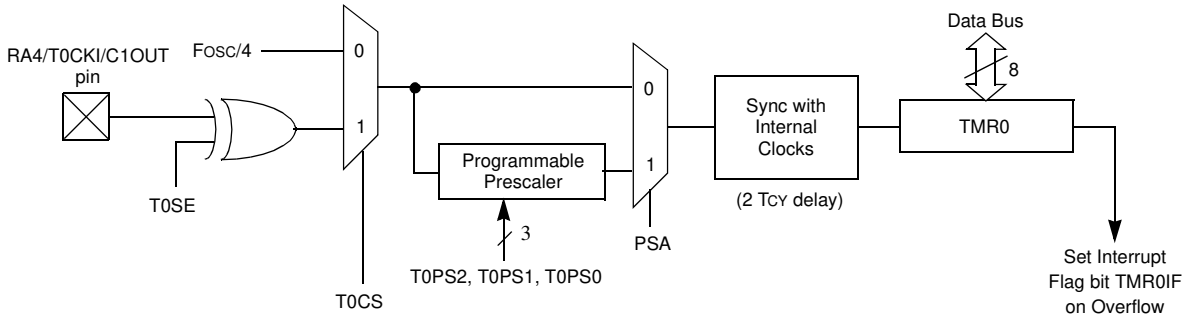
bit 4 **T0SE:** Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA:** Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is not assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0:** Timer0 Prescaler Select bits
111 = 1:256 prescale value
110 = 1:128 prescale value
101 = 1:64 prescale value
100 = 1:32 prescale value
011 = 1:16 prescale value
010 = 1:8 prescale value
001 = 1:4 prescale value
000 = 1:2 prescale value

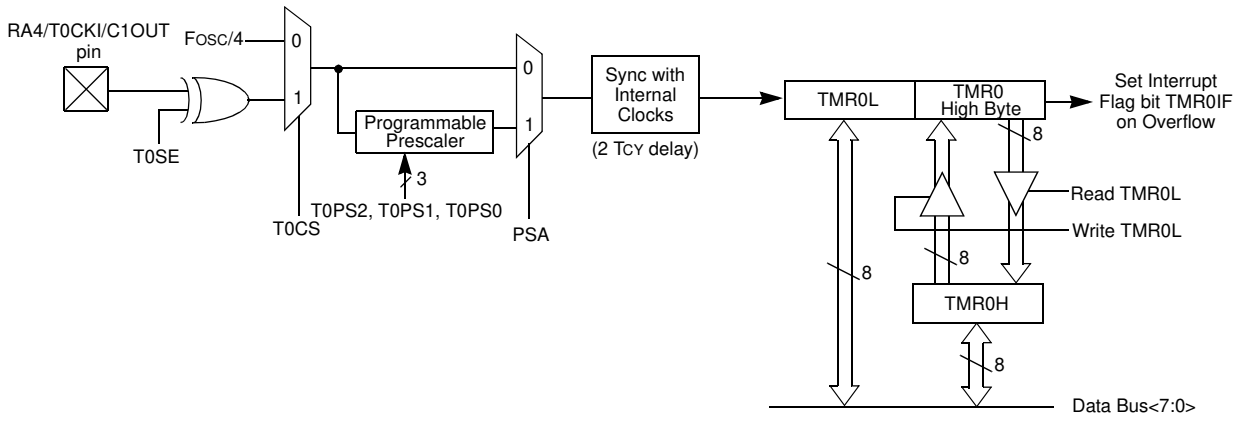
| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as ‘0’ | |
| - n = Value at POR | ‘1’ = Bit is set | ‘0’ = Bit is cleared | x = Bit is unknown |

**FIGURE 11-1: TIMER0 BLOCK DIAGRAM IN 8-BIT MODE**




**Note:** Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.

**FIGURE 11-2: TIMER0 BLOCK DIAGRAM IN 16-BIT MODE**




**Note:** Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI maximum prescale.


© 2003 Microchip Technology Inc.

## 11.1 Timer0 Operation

Timer0 can operate as a timer or as a counter.

Timer mode is selected by clearing the T0CS bit. In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

Counter mode is selected by setting the T0CS bit. In Counter mode, Timer0 will increment, either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit (T0SE). Clearing the T0SE bit selects the rising edge.

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

## 11.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not readable or writable.

The PSA and T0PS2:T0PS0 bits determine the prescaler assignment and prescale ratio.

Clearing bit PSA will assign the prescaler to the Timer0 module. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4,..., 1:256 are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0, MOVWF TMR0, BSF TMR0, x`....etc.) will clear the prescaler count.

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

### 11.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed “on-the-fly” during program execution).

## 11.3 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF bit. The interrupt can be masked by clearing the TMR0IE bit. The TMR0IF bit must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Sleep mode, since the timer requires clock cycles, even when T0CS is set.

## 11.4 16-Bit Mode Timer Reads and Writes

TMR0H is not the high byte of the timer/counter in 16-bit mode but is actually a buffered version of the high byte of Timer0 (refer to Figure 11-2). The high byte of the Timer0 counter/timer is not directly readable nor writable. TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0, without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

A write to the high byte of Timer0 must also take place through the TMR0H Buffer register. Timer0 high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

**TABLE 11-1: REGISTERS ASSOCIATED WITH TIMER0**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| TMR0L | Timer0 Module Low Byte Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR0H | Timer0 Module High Byte Register | | | | | | | | 0000 0000 | 0000 0000 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| T0CON | TMR0ON | T08BIT | T0CS | T0SE | PSA | T0PS2 | T0PS1 | T0PS0 | 1111 1111 | 1111 1111 |
| TRISA | RA7(1) | RA6(1) | PORTA Data Direction Register | | | | | | 1111 1111 | 1111 1111 |

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

**Note 1:** RA6 and RA7 are enabled as I/O pins depending on the oscillator mode selected in Configuration Word 1H.

**NOTES:**


© 2003 Microchip Technology Inc.

## 12.0 TIMER1 MODULE

The Timer1 module timer/counter has the following features:

- 16-bit timer/counter (two 8-bit registers: TMR1H and TMR1L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module special event trigger
- Status of system clock operation

Figure 12-1 is a simplified block diagram of the Timer1 module.

Register 12-1 details the Timer1 Control register. This register controls the operating mode of the Timer1 module and contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

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

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

**REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER**

| R/W-0 | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON |
| bit 7 | | | | | | | bit 0 |

bit 7 **RD16:** 16-bit Read/Write Mode Enable bit
1 = Enables register read/write of TImer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

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

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:8 prescale value
10 = 1:4 prescale value
01 = 1:2 prescale value
00 = 1:1 prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut-off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

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

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T13CKI (on the rising edge)
0 = Internal clock (FOSC/4)

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

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

## 12.1 Timer1 Operation

Timer1 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

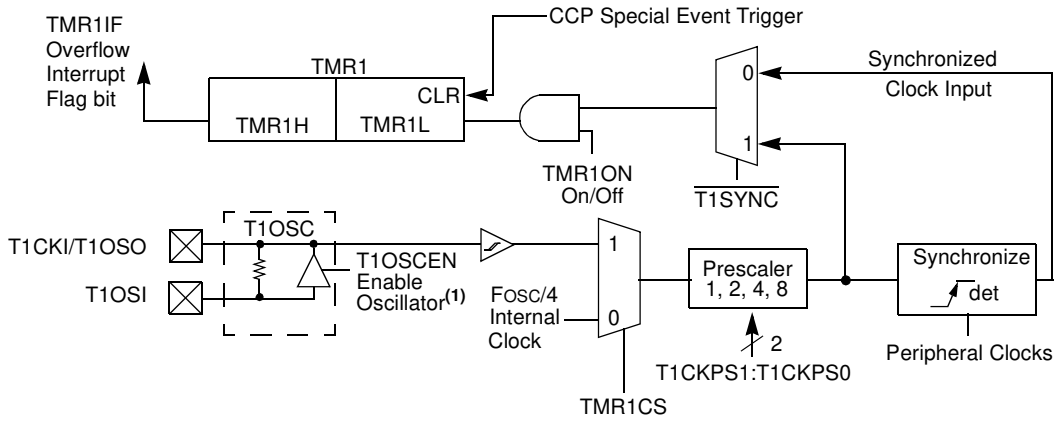
The operating mode is determined by the Clock Select bit, TMR1CS (T1CON<1>).

When TMR1CS = 0, Timer1 increments every instruction cycle. When TMR1CS = 1, Timer1 increments on every rising edge of the external clock input, or the Timer1 oscillator, if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI/CCP2 and RC0/T1OSO/T1CKI pins become inputs. The TRISC1:TRISC0 values are ignored and the pins read as '0'.

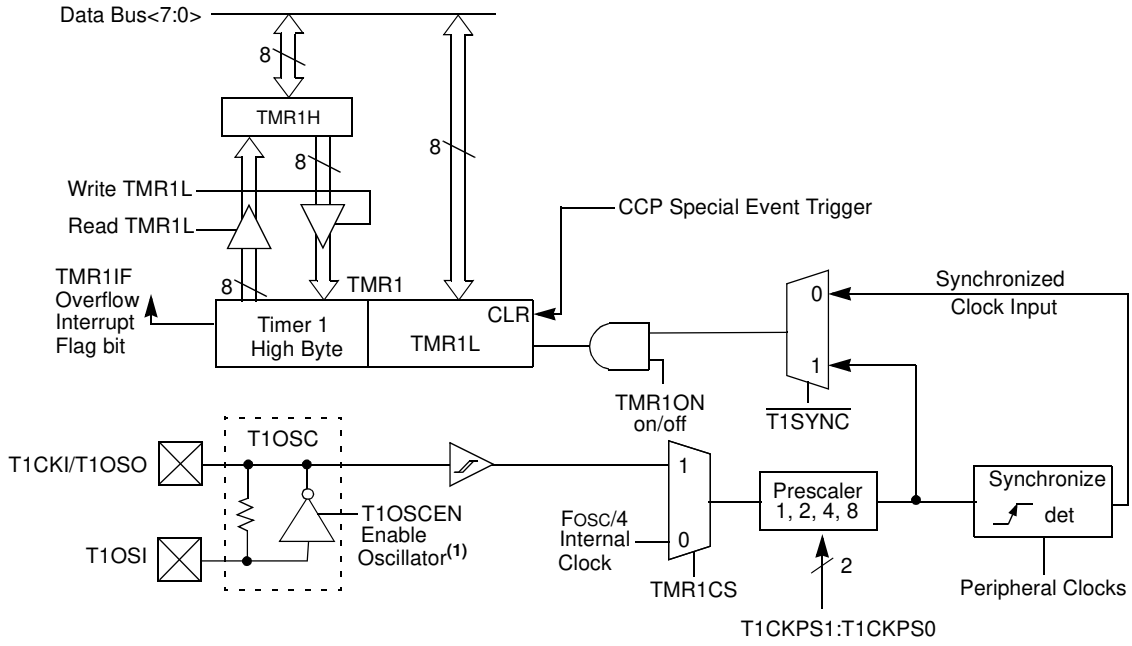
Timer1 also has an internal "Reset input". This Reset can be generated by the CCP module (see **Section 15.4.4 "Special Event Trigger"**).

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




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

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




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

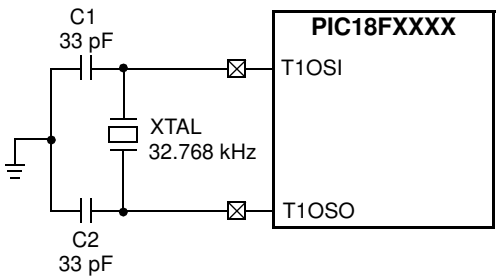

© 2003 Microchip Technology Inc.

## 12.2 Timer1 Oscillator

A crystal oscillator circuit is built-in between pins, T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator rated for 32 kHz crystals. It will continue to run during all power managed modes. The circuit for a typical LP oscillator is shown in Figure 12-3. Table 12-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

**FIGURE 12-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR**




**Note:** See the Notes with Table 12-1 for additional information about capacitor selection.

**TABLE 12-1: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR(2,3,4)**

| Osc Type | Freq | C1 | C2 |
|---|---|---|---|
| LP | 32 kHz | 27 pF(1) | 27 pF(1) |

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

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

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

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

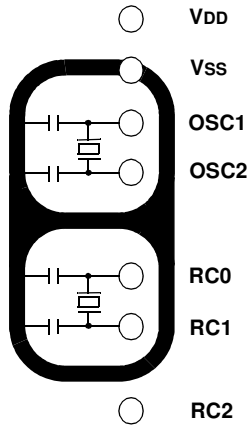
## 12.3 Timer1 Oscillator Layout Considerations

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

The oscillator circuit, shown in Figure 12-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high-speed circuit must be located near the oscillator (such as the CCP1 pin in output compare or PWM mode, or the primary oscillator using the OSC2 pin), a grounded guard ring around the oscillator circuit, as shown in Figure 12-4, may be helpful when used on a single-sided PCB or in addition to a ground plane.

**FIGURE 12-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING**




**Note:** Not drawn to scale.

## 12.4 Timer1 Interrupt

The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The Timer1 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing Timer1 interrupt enable bit, TMR1IE (PIE1<0>).

## 12.5 Resetting Timer1 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a "special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1 and start an A/D conversion if the A/D module is enabled (see **Section 15.4.4 "Special Event Trigger"** for more information).

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

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

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

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

## 12.6 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 12-2). When the RD16 control bit (T1CON<7>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L will load the contents of the high byte of Timer1 into the Timer1 high byte buffer. This provides the user with the ability to accurately read all 16 bits of Timer1 without having to determine whether a read of the high byte, followed by a read of the low byte, is valid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. Timer1 high byte is updated with the contents of TMR1H when a write occurs to TMR1L. This allows a user to write all 16 bits to both the high and low bytes of Timer1 at once.

The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

## 12.7 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in **Section 12.2 "Timer1 Oscillator"** above), gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time base and several lines of application code to calculate the time. When operating in Sleep mode and using a battery or supercapacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCisr`, shown in Example 12-1, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow, triggers the interrupt and calls the routine, which increments the seconds counter by one; additional counters for minutes and hours are incremented as the previous counter overflow.

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

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


© 2003 Microchip Technology Inc.

**EXAMPLE 12-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE**

```
RTCinit
            MOVLW   0x80            ; Preload TMR1 register pair
            MOVWF   TMR1H           ; for 1 second overflow
            CLRF    TMR1L
            MOVLW   b'00001111'     ; Configure for external clock,
            MOVWF   T1OSC           ; Asynchronous operation, external oscillator
            CLRF    secs            ; Initialize timekeeping registers
            CLRF    mins            ;
            MOVLW   .12
            MOVWF   hours
            BSF     PIE1, TMR1IE    ; Enable Timer1 interrupt
            RETURN
RTCisr
            BSF     TMR1H,7         ; Preload for 1 sec overflow
            BCF     PIR1,TMR1IF     ; Clear interrupt flag
            INCF    secs,F          ; Increment seconds
            MOVLW   .59             ; 60 seconds elapsed?
            CPFSGT  secs
            RETURN                  ; No, done
            CLRF    secs            ; Clear seconds
            INCF    mins,F          ; Increment minutes
            MOVLW   .59             ; 60 minutes elapsed?
            CPFSGT  mins
            RETURN                  ; No, done
            CLRF    mins            ; clear minutes
            INCF    hours,F         ; Increment hours
            MOVLW   .23             ; 24 hours elapsed?
            CPFSGT  hours
            RETURN                  ; No, done
            MOVLW   .01             ; Reset hours to 1
            MOVWF   hours
            RETURN                  ; Done
```

**TABLE 12-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYNC | TMR1CS | TMR1ON | 0000 0000 | u0uu uuuu |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.
**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

**NOTES:**


© 2003 Microchip Technology Inc.

## 13.0 TIMER2 MODULE

The Timer2 module timer has the following features:

- 8-bit timer (TMR2 register)
- 8-bit period register (PR2)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match with PR2
- SSP module optional use of TMR2 output to generate clock shift

Timer2 has a control register shown in Register 13-1. TMR2 can be shut-off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption. Figure 13-1 is a simplified block diagram of the Timer2 module. Register 13-1 shows the Timer2 Control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

### 13.1 Timer2 Operation

Timer2 can be used as the PWM time base for the PWM mode of the CCP module. The TMR2 register is readable and writable and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T2CKPS1:T2CKPS0 (T2CON<1:0>). The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit, TMR2IF (PIR1<1>)).

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

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

TMR2 is not cleared when T2CON is written.

**REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER**

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

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

bit 6-3 **TOUTPS3:TOUTPS0:** Timer2 Output Postscale Select bits

0000 = 1:1 postscale
0001 = 1:2 postscale
•
•
•
1111 = 1:16 postscale

bit 2 **TMR2ON:** Timer2 On bit

1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0:** Timer2 Clock Prescale Select bits

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

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

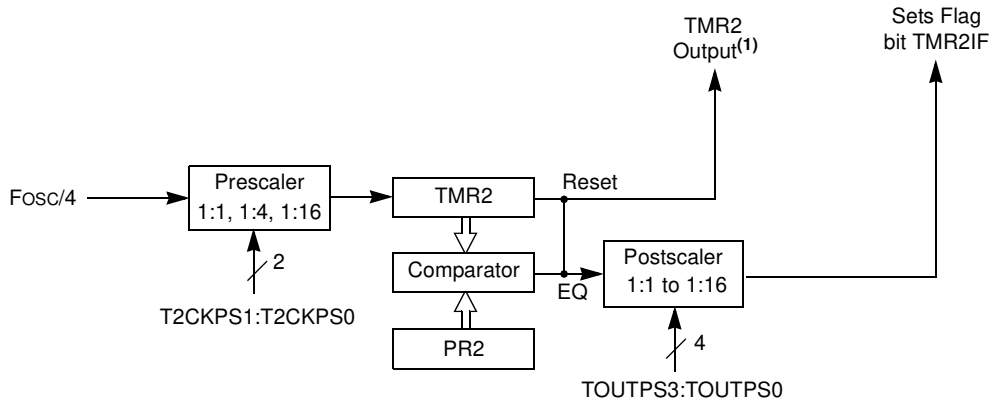
## 13.2 Timer2 Interrupt

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

## 13.3 Output of TMR2

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

**FIGURE 13-1: TIMER2 BLOCK DIAGRAM**




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

**TABLE 13-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | -000 0000 |
| PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| OSCCON | IDLEN | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 | 0000 qq00 | 0000 qq00 |

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

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X2 devices; always maintain these bits clear.


© 2003 Microchip Technology Inc.

## 14.0 TIMER3 MODULE

The Timer3 module timer/counter has the following features:

- 16-bit timer/counter (two 8-bit registers: TMR3H and TMR3L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module trigger

Figure 14-1 is a simplified block diagram of the Timer3 module.

Register 14-1 shows the Timer3 Control register. This register controls the operating mode of the Timer3 module and sets the CCP clock source.

Register 12-1 shows the Timer1 Control register. This register controls the operating mode of the Timer1 module, as well as contains the Timer1 Oscillator Enable bit (T1OSCEN) which can be a clock source for Timer3.

**REGISTER 14-1: T3CON: TIMER3 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 |
|---|---|---|---|---|---|---|---|
| RD16 | T3CCP2 | T3CKPS1 | T3CKPS0 | T3CCP1 | $\overline{\text{T3SYNC}}$ | TMR3CS | TMR3ON |
| bit 7 | | | | | | | bit 0 |

bit 7 **RD16:** 16-bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer3 in one 16-bit operation
0 = Enables register read/write of Timer3 in two 8-bit operations

bit 6, 3 **T3CCP2:T3CCP1:** Timer3 and Timer1 to CCPx Enable bits
1x = Timer3 is the clock source for compare/capture CCP modules
01 = Timer3 is the clock source for compare/capture of CCP2, Timer1 is the clock source for compare/capture of CCP1
00 = Timer1 is the clock source for compare/capture CCP modules

bit 5-4 **T3CKPS1:T3CKPS0:** Timer3 Input Clock Prescale Select bits
11 = 1:8 prescale value
10 = 1:4 prescale value
01 = 1:2 prescale value
00 = 1:1 prescale value

bit 2 **$\overline{\text{T3SYNC}}$:** Timer3 External Clock Input Synchronization Control bit
(Not usable if the system clock comes from Timer1/Timer3.)
When TMR3CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR3CS = 0:
This bit is ignored. Timer3 uses the internal clock when TMR3CS = 0.

bit 1 **TMR3CS:** Timer3 Clock Source Select bit
1 = External clock input from Timer1 oscillator or T1CKI (on the rising edge after the first falling edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR3ON:** Timer3 On bit
1 = Enables Timer3
0 = Stops Timer3

| **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 |

## 14.1 Timer3 Operation

Timer3 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

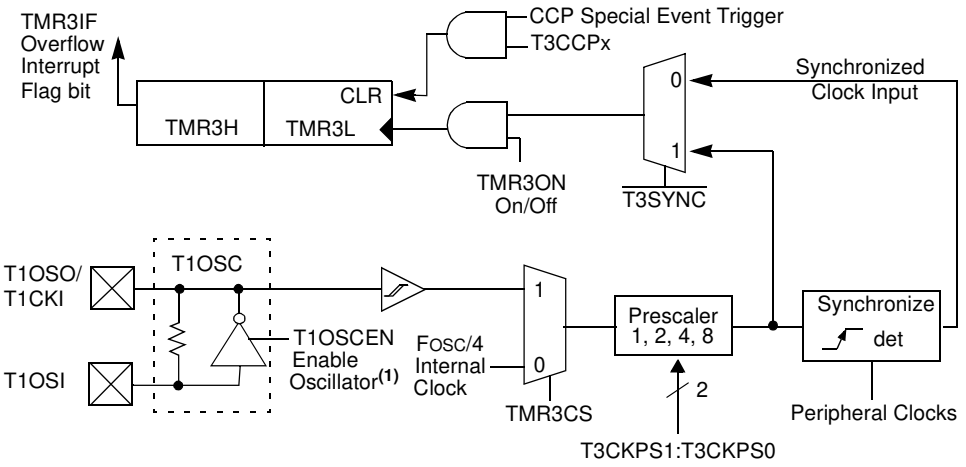
The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>).

When TMR3CS = 0, Timer3 increments every instruction cycle. When TMR3CS = 1, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI/CCP2 and RC0/T1OSO/T1CKI pins become inputs. That is, the TRISC1:TRISC0 value is ignored and the pins are read as '0'.

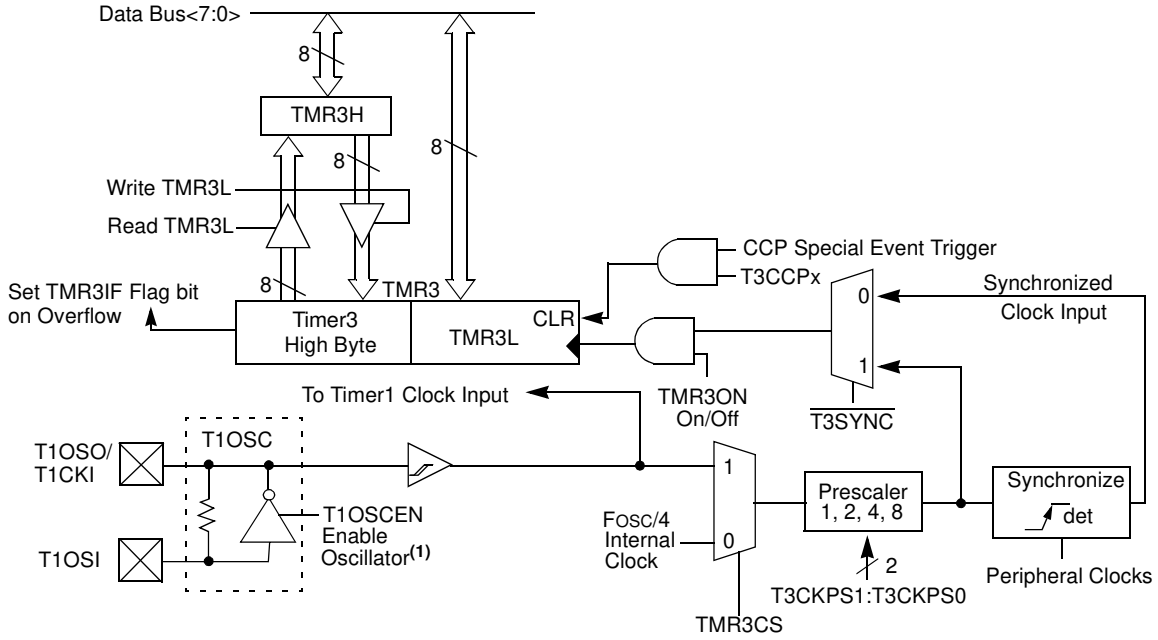
Timer3 also has an internal "Reset input". This Reset can be generated by the CCP module (see **Section 15.4.4 "Special Event Trigger"**).

**FIGURE 14-1: TIMER3 BLOCK DIAGRAM**




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

**FIGURE 14-2: TIMER3 BLOCK DIAGRAM CONFIGURED IN 16-BIT READ/WRITE MODE**




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


© 2003 Microchip Technology Inc.

## 14.2 Timer1 Oscillator

The Timer1 oscillator may be used as the clock source for Timer3. The Timer1 oscillator is enabled by setting the T1OSCEN (T1CON<3>) bit. The oscillator is a low-power oscillator rated for 32 kHz crystals. See **Section 12.2 "Timer1 Oscillator"** for further details.

## 14.3 Timer3 Interrupt

The TMR3 register pair (TMR3H:TMR3L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR3 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit, TMR3IF (PIR2<1>). This interrupt can be enabled/disabled by setting/clearing TMR3 Interrupt Enable bit, TMR3IE (PIE2<1>).

## 14.4 Resetting Timer3 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a "special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer3. See **Section 15.4.4 "Special Event Trigger"** for more information.

| **Note:** | The special event triggers from the CCP module will not set interrupt flag bit, TMR3IF (PIR1<0>). |
|---|---|

Timer3 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer3 is running in Asynchronous Counter mode, this Reset operation may not work. In the event that a write to Timer3 coincides with a special event trigger from CCP1, the write will take precedence. In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer3.

**TABLE 14-1: REGISTERS ASSOCIATED WITH TIMER3 AS A TIMER/COUNTER**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR2 | OSCIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | 00-0 0000 |
| PIE2 | OSCIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | 00-0 0000 |
| IPR2 | OSCIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | 11-1 1111 |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON | 0000 0000 | u0uu uuuu |
| T3CON | RD16 | T3CCP2 | T3CKPS1 | T3CKPS0 | T3CCP1 | $\overline{\text{T3SYNC}}$ | TMR3CS | TMR3ON | 0000 0000 | uuuu uuuu |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer3 module.

**NOTES:**


© 2003 Microchip Technology Inc.

## 15.0 CAPTURE/COMPARE/PWM (CCP) MODULES

The standard CCP (Capture/Compare/PWM) module contains a 16-bit register that can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. Table 15-1 shows the timer resources required for each of the CCP module modes.

The operation of CCP1 is identical to that of CCP2, with the exception of the special event trigger. Therefore, operation of a CCP module is described with respect to CCP1 except where noted. Table 15-2 shows the interaction of the CCP modules.

**Note:** In 28-pin devices, both CCP1 and CCP2 function as standard CCP modules. In 40-pin devices, CCP1 is implemented as an Enhanced CCP module, offering additional capabilities in PWM mode. Capture and Compare modes are identical in all modules regardless of the device.

Please see **Section 16.0 "Enhanced Capture/Compare/PWM (ECCP) Module"** for a discussion of the enhanced PWM capabilities of the CCP1 module.

**REGISTER 15-1: CCPxCON: CCP MODULE CONTROL REGISTER**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | DCxB1 | DCxB0 | CCPxM3 | CCPxM2 | CCPxM1 | CCPxM0 |
| bit 7 | | | | | | | bit 0 |

bit 7-6 **Reserved:** Read as '0'.
See **Section 16.0 "Enhanced Capture/Compare/PWM (ECCP) Module"**.

bit 5-4 **DCxB1:DCxB0:** PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs (bit 1 and bit 0) of the 10-bit PWM duty cycle. The upper eight bits (DCx9:DCx2) of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM3:CCPxM0**: CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0001 = Reserved
0010 = Compare mode, toggle output on match (CCPxIF bit is set)
0011 = Reserved
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, initialize CCP pin Low; on compare match, force CCP pin High (CCPxIF bit is set)
1001 = Compare mode, initialize CCP pin High; on compare match, force CCP pin Low (CCPxIF bit is set)
1010 = Compare mode, generate software interrupt on compare match (CCPxIF bit is set, CCP pin operates as a port pin for input and output)
1011 = Compare mode, trigger special event (CCP2IF bit is set)
11xx = PWM mode

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

## 15.1 CCP1 Module

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. All are readable and writable.

**TABLE 15-1: CCP MODE - TIMER RESOURCE**

| CCP Mode | Timer Resource |
|---|---|
| Capture | Timer1 or Timer3 |
| Compare | Timer1 or Timer3 |
| PWM | Timer2 |

## 15.2 CCP2 Module

Capture/Compare/PWM Register 2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. All are readable and writable.

CCP2 functions identically to CCP1 except for the enhanced PWM modes offered by CCP2

**TABLE 15-2: INTERACTION OF TWO CCP MODULES**

| CCPx Mode | CCPy Mode | Interaction |
|---|---|---|
| Capture | Capture | TMR1 or TMR3 time base. Time base can be different for each CCP. |
| Capture | Compare | The compare could be configured for the special event trigger which clears either TMR1 or TMR3 depending upon which time base is used. |
| Compare | Compare | The compare(s) could be configured for the special event trigger which clears TMR1 or TMR3 depending upon which time base is used. |
| PWM | PWM | The PWMs will have the same frequency and update rate (TMR2 interrupt). |
| PWM | Capture | None. |
| PWM | Compare | None. |


© 2003 Microchip Technology Inc.

## 15.3 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on pin RC2/CCP1/P1A. An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

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

### 15.3.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1/P1A pin should be configured as an input by setting the TRISC<2> bit.

| **Note:** | If the RC2/CCP1/P1A is configured as an output, a write to the port can cause a capture condition. |
|---|---|

### 15.3.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (either Timer1 and/or Timer3) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation may not work. The timer to be used with each CCP module is selected in the T3CON register.

### 15.3.3 SOFTWARE INTERRUPT

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

### 15.3.4 CCP PRESCALER

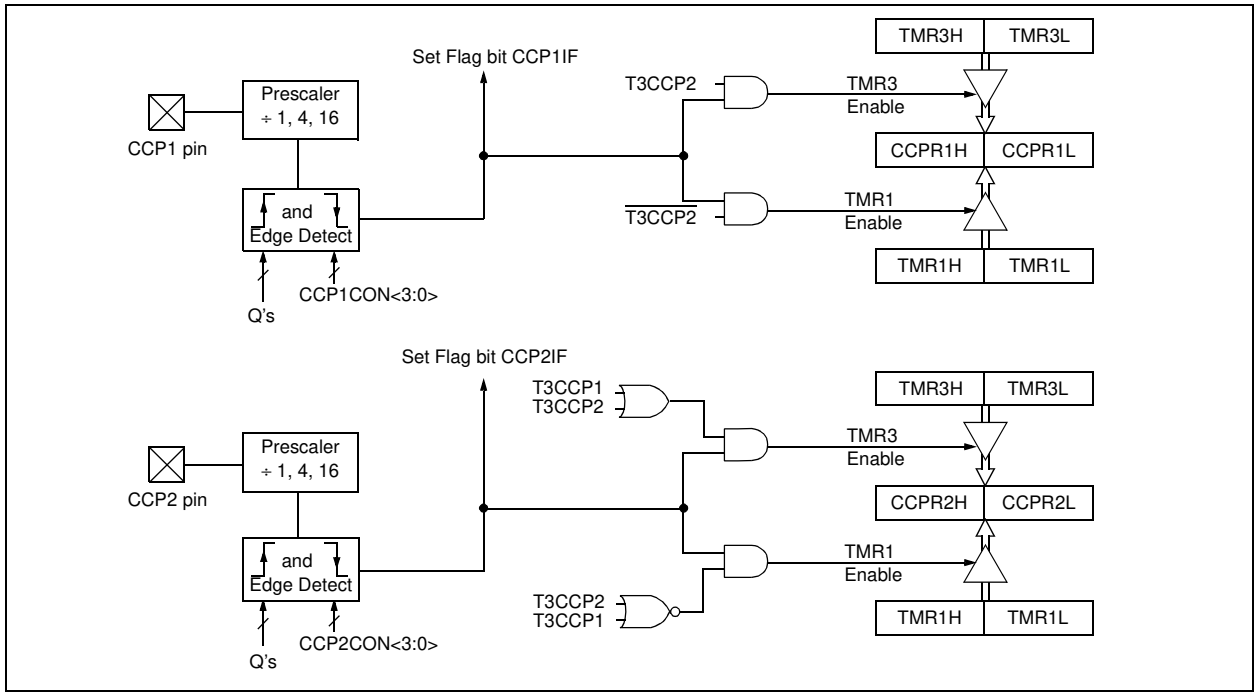
There are four prescaler settings specified by bits CCP1M3:CCP1M0. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

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

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

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

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



## 15.4 Compare Mode

In Compare mode, the 16-bit CCPR1 (CCPR2) register value is constantly compared against either the TMR1 register pair value, or the TMR3 register pair value. When a match occurs, the RC2/CCP1/P1A (RC1/T1OSI/CCP2) pin:

- Is driven High
- Is driven Low
- Toggles output (High to Low or Low to High)
- Remains unchanged (interrupt only)

The action on the pin is based on the value of control bits, CCP1M3:CCP1M0 (CCP2M3:CCP2M0). At the same time, interrupt flag bit, CCP1IF (CCP2IF), is set.

### 15.4.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRISC bit.

| **Note:** | Clearing the CCP1CON register will force the RC2/CCP1/P1A compare output latch to the default low level. This is not the PORTC I/O data latch. |
|---|---|

### 15.4.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode, or Synchronized Counter mode, if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

### 15.4.3 SOFTWARE INTERRUPT MODE

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

### 15.4.4 SPECIAL EVENT TRIGGER

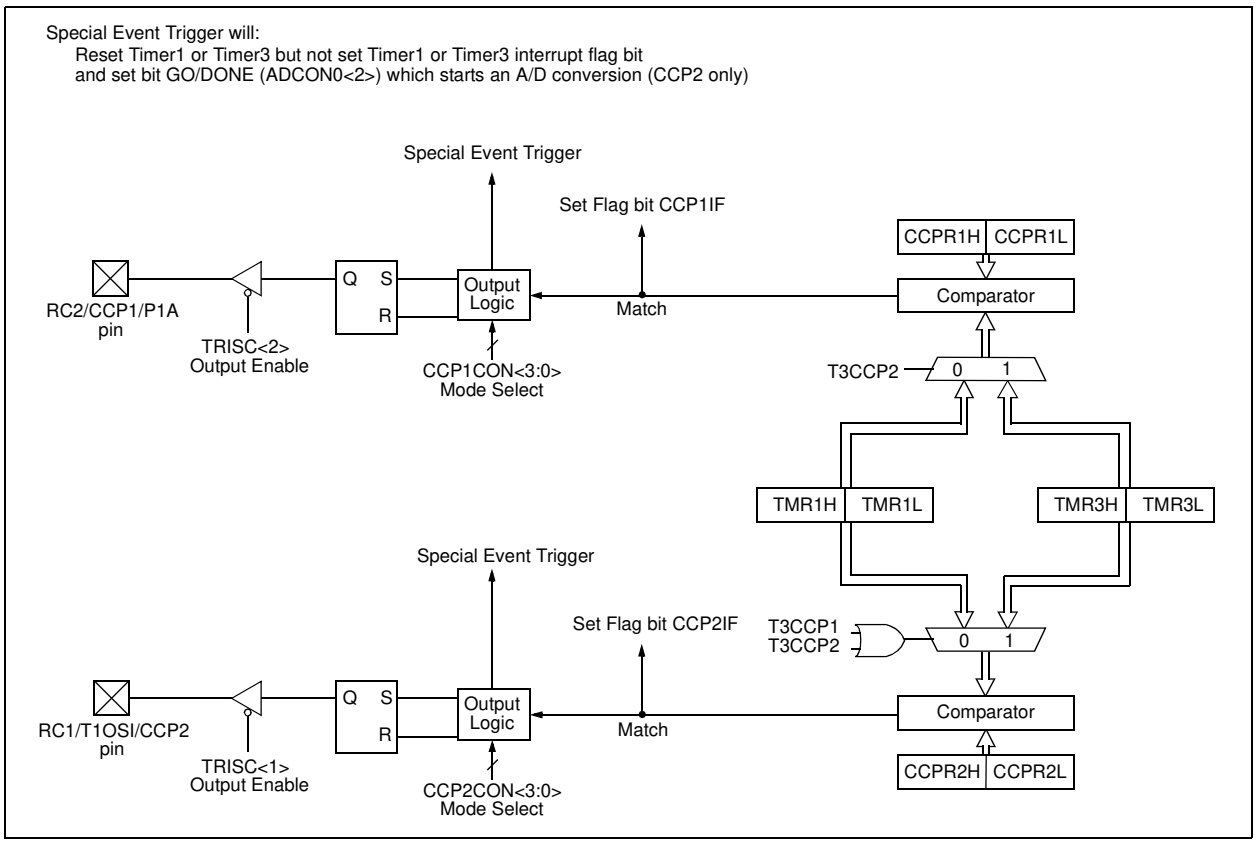
In this mode, an internal hardware trigger is generated which may be used to initiate an action.

The special event trigger output of CCP1 resets the TMR1 register pair. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

The special trigger output of CCP2 resets either the TMR1 or TMR3 register pair. Additionally, the CCP2 special event trigger will start an A/D conversion if the A/D module is enabled.

| **Note:** | The special event trigger from the CCP2 module will not set the Timer1 or Timer3 interrupt flag bits. |
|---|---|

**FIGURE 15-2: COMPARE MODE OPERATION BLOCK DIAGRAM**





© 2003 Microchip Technology Inc.

**TABLE 15-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON | 0000 0000 | uuuu uuuu |
| CCPR1L | Capture/Compare/PWM Register 1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPR1H | Capture/Compare/PWM Register 1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCP1CON | — | — | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |
| CCPR2L | Capture/Compare/PWM Register 2 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPR2H | Capture/Compare/PWM Register 2 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | --00 0000 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | 00-0 0000 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | 00-0 0000 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | 11-1 1111 |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| T3CON | RD16 | T3CCP2 | T3CKPS1 | T3CKPS0 | T3CCP1 | $\overline{\text{T3SYNC}}$ | TMR3CS | TMR3ON | 0000 0000 | uuuu uuuu |

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

**Note 1:** These bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

## 15.5 PWM Mode

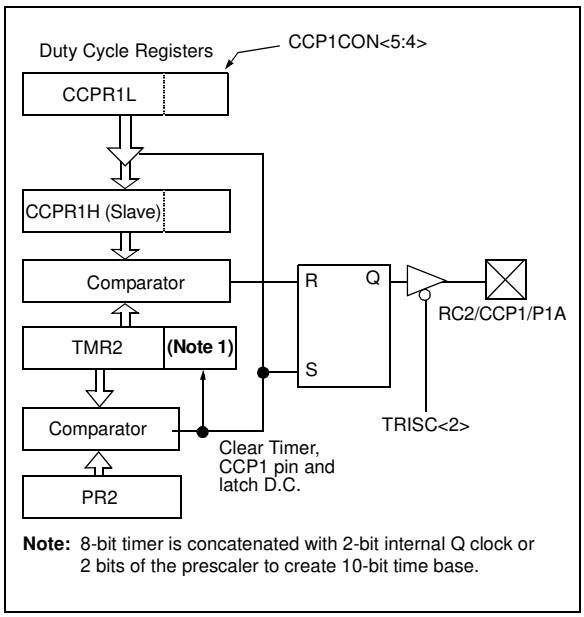
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

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

Figure 15-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 15.5.3 "Setup for PWM Operation"**.

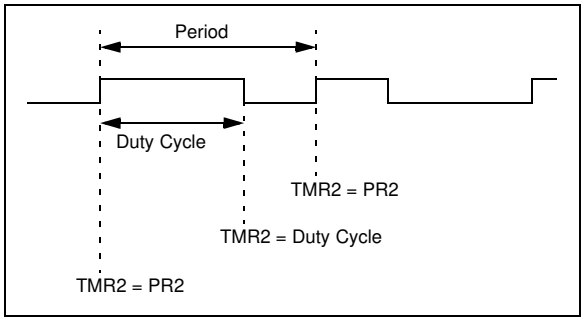
**FIGURE 15-3: SIMPLIFIED PWM BLOCK DIAGRAM**




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

A PWM output (Figure 15-4) has a time base (*period*) and a time that the output is high (*duty cycle*). The frequency of the PWM is the inverse of the period (1/period).

**FIGURE 15-4: PWM OUTPUT**




### 15.5.1 PWM PERIOD

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

**EQUATION 15-1:**

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period]. When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

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

| **Note:** | The Timer2 postscaler (see **Section 13.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output. |
|---|---|

### 15.5.2 PWM DUTY CYCLE

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

**EQUATION 15-2:**

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

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


© 2003 Microchip Technology Inc.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation. When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or two bits of the TMR2 prescaler, the CCP1 pin is cleared. The maximum PWM resolution (bits) for a given PWM frequency is given by the following equation.

**EQUATION 15-3:**

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

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

### 15.5.3 SETUP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and the CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

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

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

**TABLE 15-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| PR2 | Timer2 Module Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | -000 0000 |
| CCPR1L | Capture/Compare/PWM Register 1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPR1H | Capture/Compare/PWM Register 1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCP1CON | — | — | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | --00 0000 | --00 0000 |
| CCPR2L | Capture/Compare/PWM Register 2 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPR2H | Capture/Compare/PWM Register 2 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | --00 0000 |
| OSCCON | IDLEN | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 | 0000 qq00 | 0000 qq00 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.
**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

**NOTES:**


© 2003 Microchip Technology Inc.

## 16.0 ENHANCED CAPTURE/ COMPARE/PWM (ECCP) MODULE

| **Note:** | The ECCP (Enhanced Capture/ Compare/ PWM) module is only available on PIC18F4X20 devices. |
|---|---|

In 40 and 44-pin devices, the CCP1 module is implemented as a standard CCP module with enhanced PWM capabilities. Operation of the Capture, Compare and standard single output PWM modes is described in **Section 15.0 "Capture/Compare/PWM (CCP) Modules"**. Discussion in that section relating to PWM frequency and duty cycle also apply to the enhanced PWM mode.

The ECCP module differs from the CCP with the addition of an enhanced PWM mode which allows for 2 or 4 output channels, user-selectable polarity, dead band control and automatic shutdown and restart. These features are discussed in detail in **Section 16.4 "Enhanced PWM Mode"**.

The control register for CCP1 is shown in Register 16-1. It differs from the CCP1CON register of PIC18F2X20 devices in that the two Most Significant bits are implemented to control enhanced PWM functionality.

**REGISTER 16-1: CCP1CON REGISTER FOR ENHANCED CCP OPERATION (PIC18F4X20 ONLY)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| P1M1 | P1M0 | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 |
| bit 7 | | | | | | | bit 0 |

bit 7-6 **P1M1:P1M0:** PWM Output Configuration bits

If CCP1M<3:2> = 00, 01, 10 (Capture, Compare, or disabled):

xx = P1A assigned as Capture/Compare input; P1B, P1C, P1D assigned as port pins

If CCP1M<3:2> = 11 (PWM modes):

00 = Single output; P1A modulated; P1B, P1C, P1D assigned as port pins
01 = Full-bridge output forward; P1D modulated; P1A active; P1B, P1C inactive
10 = Half-bridge output; P1A, P1B modulated with dead band control; P1C, P1D assigned as port pins
11 = Full-bridge output reverse; P1B modulated; P1C active; P1A, P1D inactive

bit 5-4 **DC1B1:DC1B0:** PWM Duty Cycle Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPR1L.

bit 3-0 **CCP1M3:CCP1M0:** ECCP1 Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCP module)
0001 = Unused (reserved)
0010 = Compare mode, toggle output on match (ECCP1IF bit is set)
0011 = Unused (reserved)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (ECCP1IF bit is set)
1001 = Compare mode, clear output on match (ECCP1IF bit is set)
1010 = Compare mode, generate software interrupt on match (ECCP1IF bit is set, ECCP1 pin operates as a port pin for input and output)
1011 = Compare mode, trigger special event (ECCP1IF bit is set, ECCP resets TMR1or TMR2 and starts an A/D conversion if the A/D module is enabled)
1100 = PWM mode, P1A, P1C active-high, P1B, P1D active-high
1101 = PWM mode, P1A, P1C active-high, P1B, P1D active-low
1110 = PWM mode, P1A, P1C active-low, P1B, P1D active-high
1111 = PWM mode, P1A, P1C active-low, P1B, P1D active-low

| **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 |

In addition to the expanded functions of the CCP1CON register, the ECCP module has two additional registers associated with enhanced PWM operation and Auto-Shutdown features:

- PWM1CON
- ECCPAS

All other registers associated with the ECCP module are identical to those used for the CCP1 module in PIC18F2X20 devices, including register and individual bit names. Likewise, the timer assignments and interactions between the two CCP modules are identical, regardless of whether CCP1 is a standard or enhanced module.

## 16.1 ECCP Outputs

The Enhanced CCP module may have up to four outputs depending on the selected operating mode. These outputs, designated P1A through P1D, are multiplexed with I/O pins on PORTC and PORTD. The pin assignments are summarized in Table 16-1.

To configure I/O pins as PWM outputs, the proper PWM mode must be selected by setting the P1Mn and CCP1Mn bits (CCP1CON<7:6> and <3:0>, respectively). The appropriate TRISC and TRISD direction bits for the port pins must also be set as outputs.

## 16.2 Capture and Compare Modes

The Capture and Compare modes of the ECCP module are identical in operation to that of CCP1, as discussed in **Section 15.3 "Capture Mode"** and **Section 15.4 "Compare Mode"**. No changes are required when moving between these modules on PIC18F2X20 and PIC18F4X20 devices.

## 16.3 Standard PWM Mode

When configured in Single Output mode, the ECCP module functions identically to the standard CCP module in PWM mode, as described in **Section 15.4 "Compare Mode"**.

| **Note:** | When setting up single output PWM operations, users are free to use either of the processes described in **Section 15.5.3 "Setup for PWM Operation"** or **Section 16.4.7 "Setup for PWM Operation"**. The latter is more generic but will work for either single or multi output PWM. |
|---|---|

**TABLE 16-1: PIN ASSIGNMENTS FOR VARIOUS ECCP MODES**

| ECCP Mode | CCP1CON Configuration | RC2 | RD5 | RD6 | RD7 |
|---|---|---|---|---|---|
| Compatible CCP | 00xx11xx | CCP1 | RD5/PSP5 | RD6/PSP6 | RD7/PSP7 |
| Dual PWM | 10xx11xx | P1A | P1B | RD6/PSP6 | RD6/PSP6 |
| Quad PWM | x1xx11xx | P1A | P1B | P1C | P1D |

**Legend:** x = Don't care. Shaded cells indicate pin assignments not used by ECCP in a given mode.

**Note 1:** TRIS register values must be configured appropriately.

**2:** With ECCP in Dual or Quad PWM mode, the PSP input/output control of PORTD is overridden by P1B, P1C and P1D.


© 2003 Microchip Technology Inc.

## 16.4 Enhanced PWM Mode

The Enhanced PWM mode provides additional PWM output options for a broader range of control applications. The module is an upwardly compatible version of the standard CCP module and offers up to four outputs, designated P1A through P1D. Users are also able to select the polarity of the signal (either active-high or active-low). The module's output mode and polarity are configured by setting the P1M1:P1M0 and CCP1M3:CCP1M0 bits of the CCP1CON register (CCP1CON<7:6> and CCP1CON<3:0>, respectively).

Figure 16-1 shows a simplified block diagram of PWM operation. All control registers are double-buffered and are loaded at the beginning of a new PWM cycle (the period boundary when Timer2 resets) in order to prevent glitches on any of the outputs. The exception is the PWM Delay register, ECCP1DEL, which is loaded at either the duty cycle boundary or the boundary period (whichever comes first). Because of the buffering, the module waits until the assigned timer resets instead of starting immediately. This means that enhanced PWM waveforms do not exactly match the standard PWM waveforms but are instead offset by one full instruction cycle (4 TOSC).

As before, the user must manually configure the appropriate TRISD bits for output.

### 16.4.1 PWM OUTPUT CONFIGURATIONS

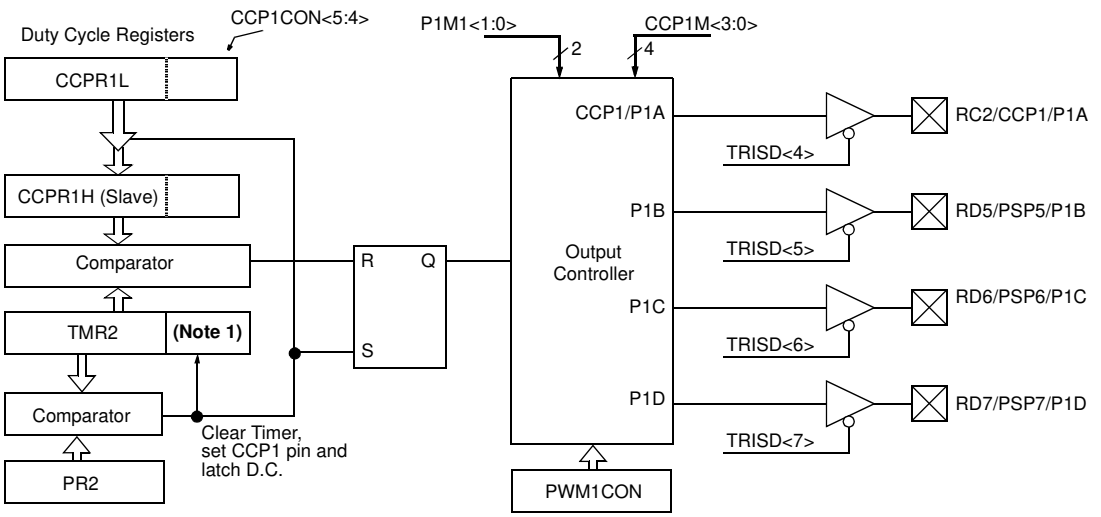
The P1M1:P1M0 bits in the CCP1CON register allow one of four configurations:

- Single Output
- Half-Bridge Output
- Full-Bridge Output, Forward mode
- Full-Bridge Output, Reverse mode

The Single Output mode is the Standard PWM mode discussed in **Section 15.5 "PWM Mode"**. The Half-Bridge and Full-Bridge Output modes are covered in detail in the sections that follow.

The general relationship of the outputs in all configurations is summarized in Figure 16-2.

**FIGURE 16-1: SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODULE**




**Note:** The 8-bit timer TMR2 register is concatenated with the 2-bit internal Q clock or 2 bits of the prescaler to create the 10-bit time base.

**FIGURE 16-2: PWM OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)**

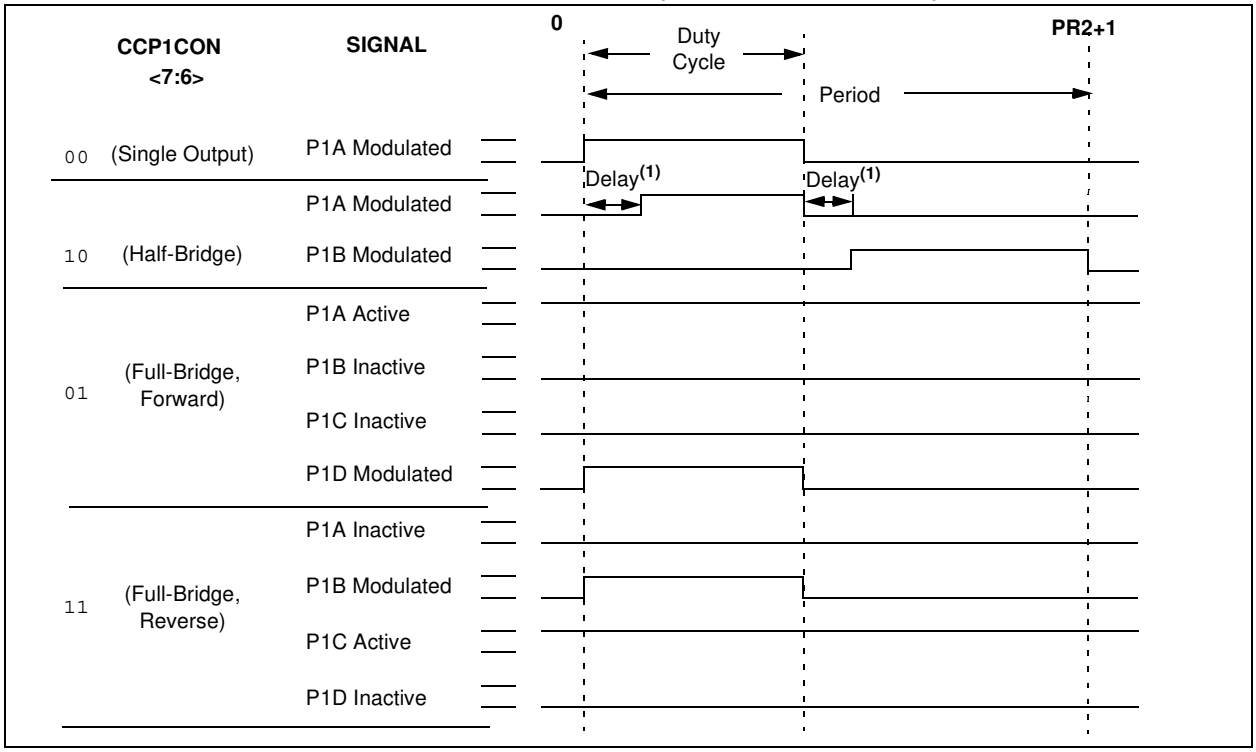
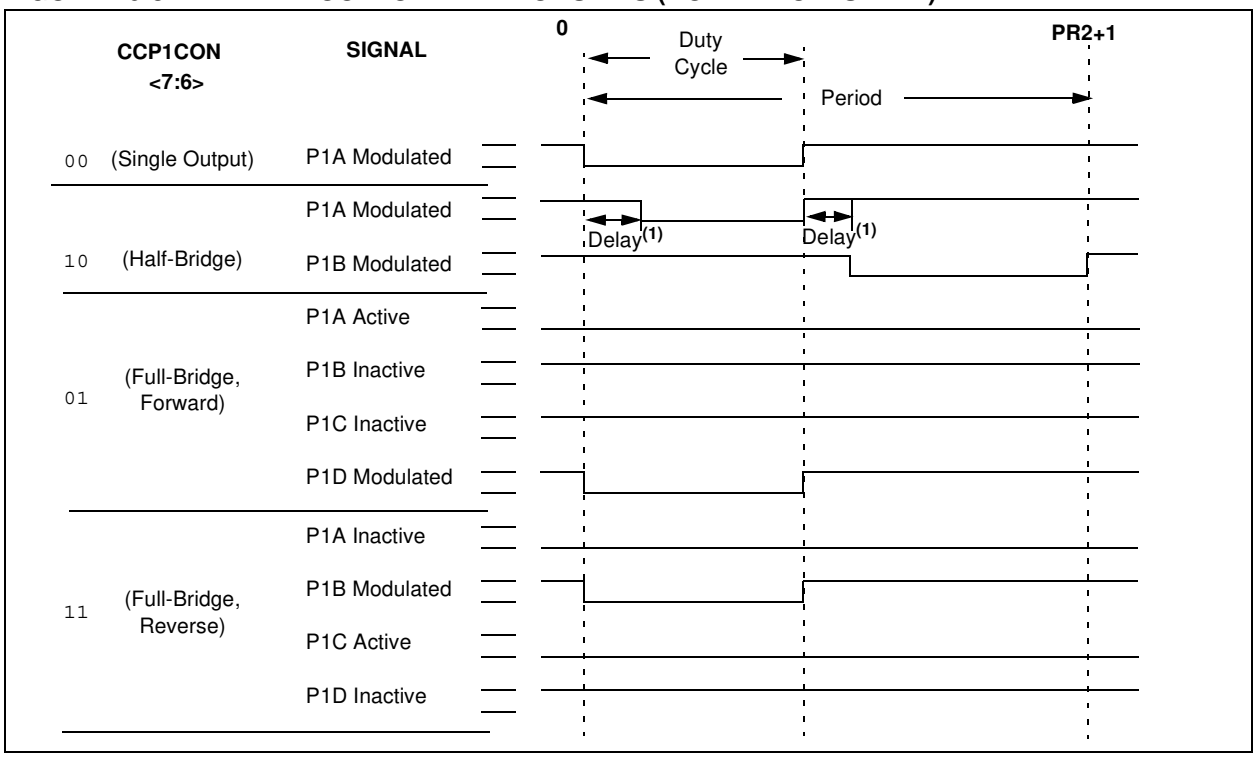



**FIGURE 16-3: PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)**




**Relationships:**

- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Duty Cycle = TOSC * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (PWM1CON<6:0>)

**Note 1:** Dead band delay is programmed using the PWM1CON register (see **Section 16.4.4 "Programmable Dead Band Delay"**).


© 2003 Microchip Technology Inc.

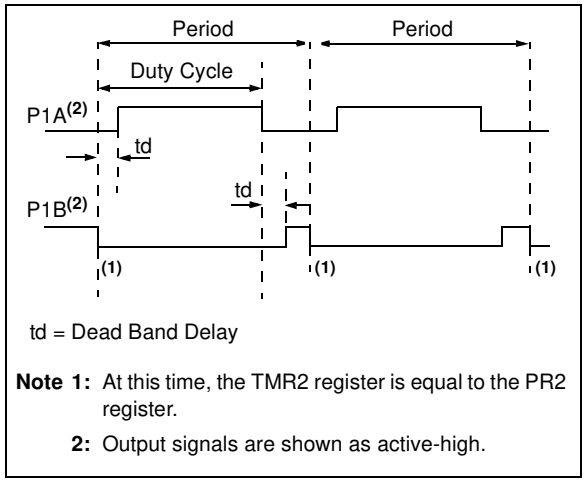
### 16.4.2 HALF-BRIDGE MODE

In the Half-Bridge Output mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the RC2/CCP1/P1A pin, while the complementary PWM output signal is output on the RD5/PSP5/P1B pin (Figure 16-4). This mode can be used for half-bridge applications, as shown in Figure 16-5, or for full-bridge applications where four power switches are being modulated with two PWM signals.

In Half-Bridge Output mode, the programmable dead band delay can be used to prevent shoot-through current in half-bridge power devices. The value of bits PDC6:PDC0 sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. See **Section 16.4.4 "Programmable Dead Band Delay"** for more details of the dead band delay operations.

Since the P1A and P1B outputs are multiplexed with the PORTC<2> and PORTD<5> data latches, the TRISC<2> and TRISD<5> bits must be cleared to configure P1A and P1B as outputs.

**FIGURE 16-4: HALF-BRIDGE PWM OUTPUT**

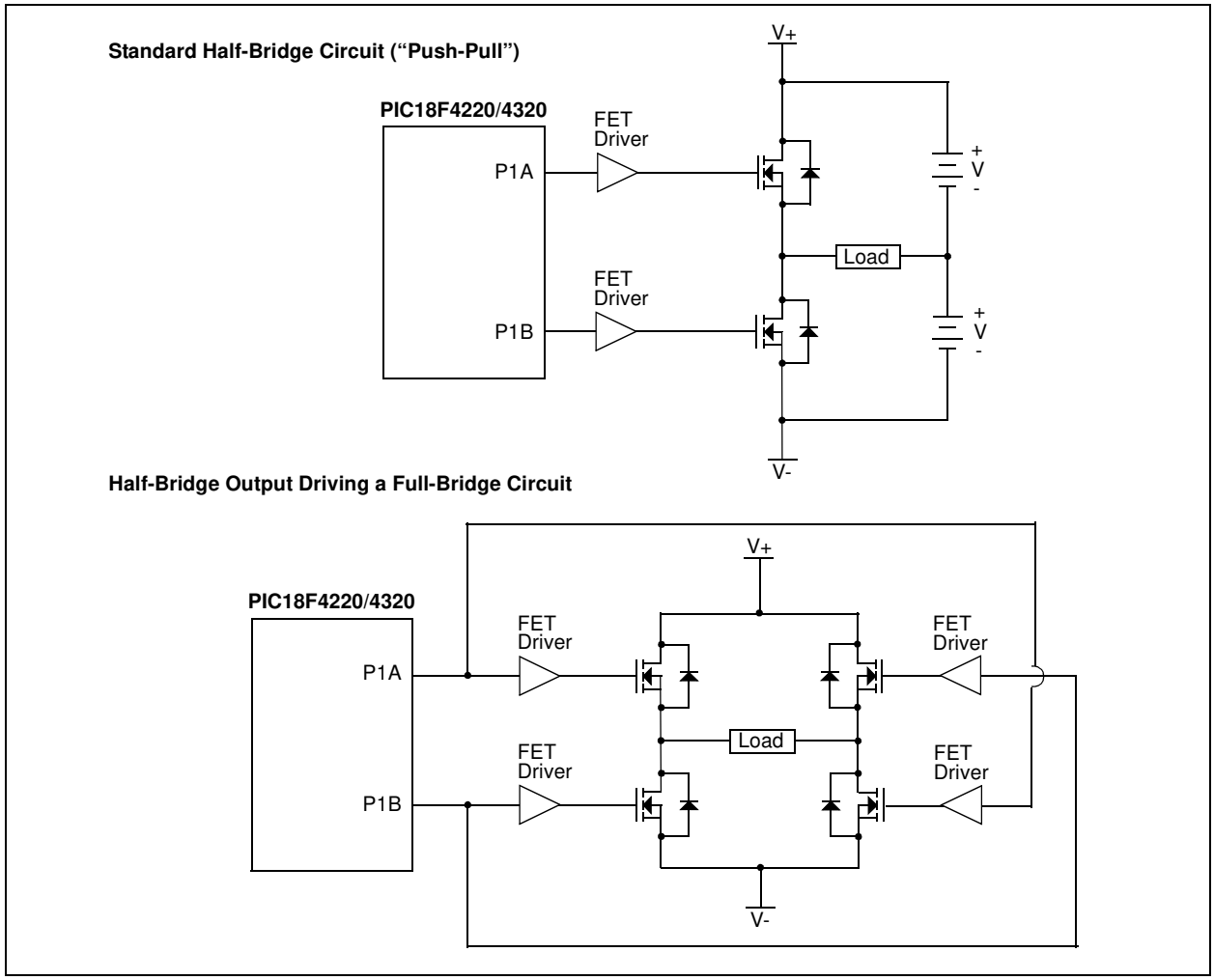



td = Dead Band Delay

**Note 1:** At this time, the TMR2 register is equal to the PR2 register.

**2:** Output signals are shown as active-high.

**FIGURE 16-5: EXAMPLES OF HALF-BRIDGE OUTPUT MODE APPLICATIONS**

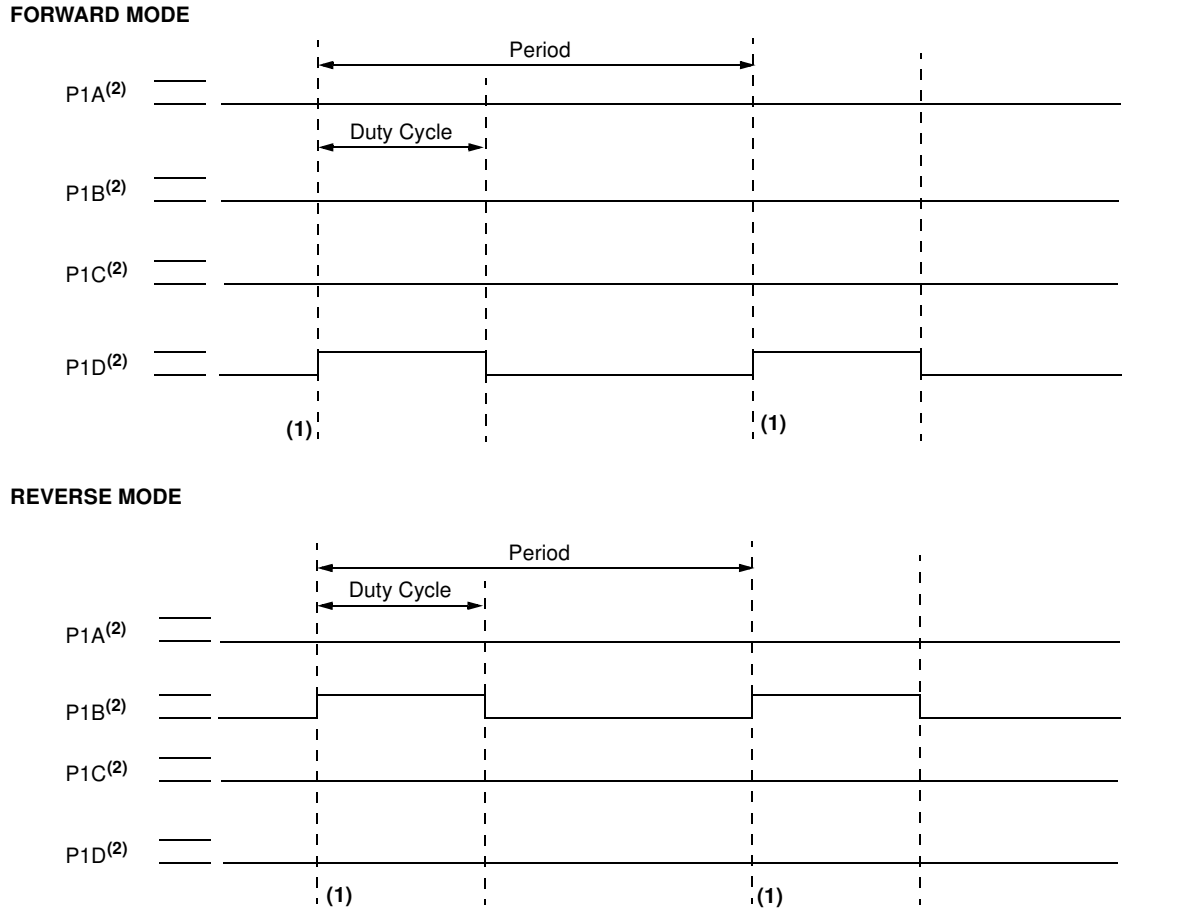


### 16.4.3 FULL-BRIDGE MODE

In Full-Bridge Output mode, four pins are used as outputs; however, only two outputs are active at a time. In the Forward mode, pin RC2/CCP1/P1A is continuously active and pin RD7/PSP7/P1D is modulated. In the Reverse mode, RD6/PSP6/P1C pin is continuously active and RD5/PSP5/P1B pin is modulated. These are illustrated in Figure 16-6.

P1A, P1B, P1C and P1D outputs are multiplexed with the PORTC<2> and PORTD<5:7> data latches. The TRISC<2> and TRISD<5:7> bits must be cleared to make the P1A, P1B, P1C and P1D pins output.

**FIGURE 16-6: FULL-BRIDGE PWM OUTPUT**

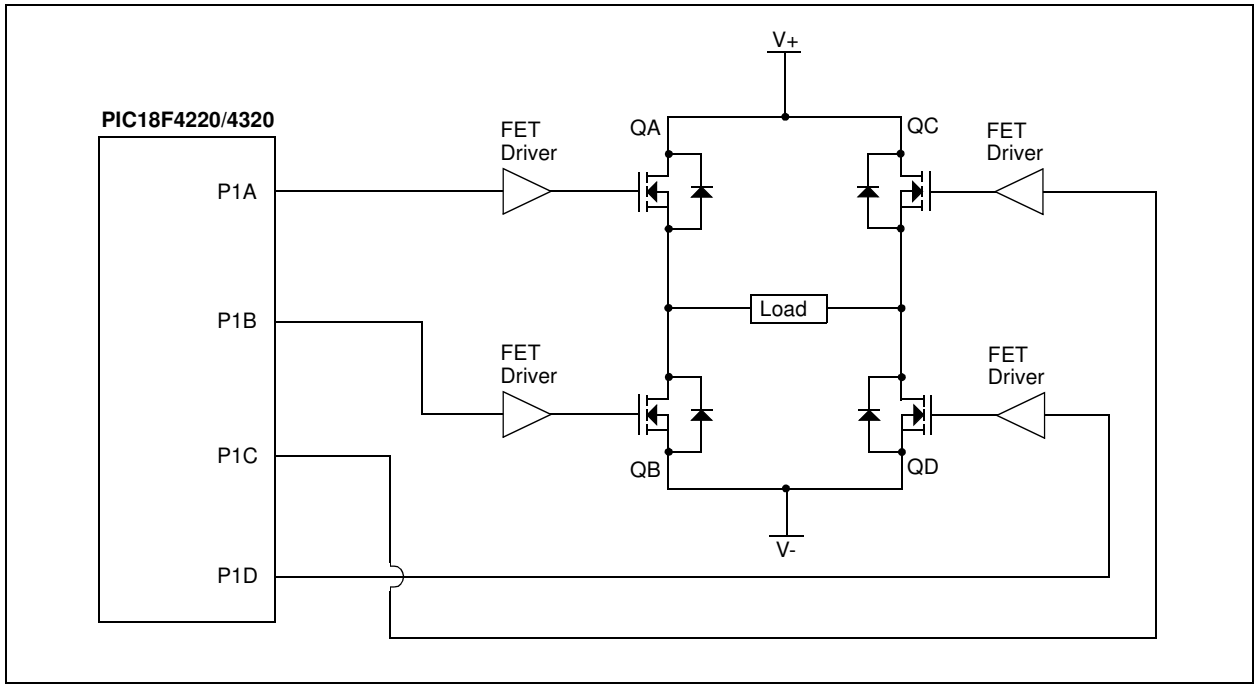



**Note 1:** At this time, the TMR2 register is equal to the PR2 register.

**2:** Output signal is shown as active-high.


© 2003 Microchip Technology Inc.

**FIGURE 16-7: EXAMPLE OF FULL-BRIDGE APPLICATION**




### 16.4.3.1 Direction Change in Full-Bridge Mode

In the Full-Bridge Output mode, the P1M1 bit in the CCP1CON register allows users to control the forward/ reverse direction. When the application firmware changes this direction control bit, the module will assume the new direction on the next PWM cycle.

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

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

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

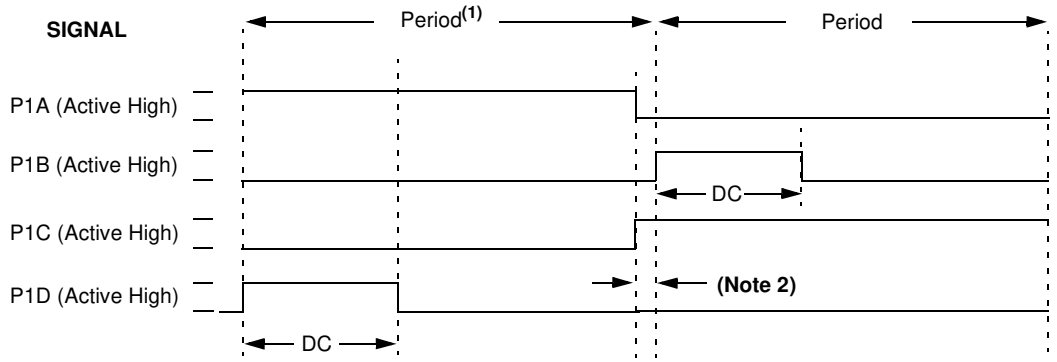
Figure 16-9 shows an example where the PWM direction changes from forward to reverse at a near 100% duty cycle. At time t1, the outputs P1A and P1D become inactive, while output P1C becomes active. In this example, since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current may flow through power devices QC and QD (see Figure 16-7) for the duration of 't'. The same phenomenon will occur to power devices QA and QB for PWM direction change from reverse to forward.

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

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

Other options to prevent shoot-through current may exist.

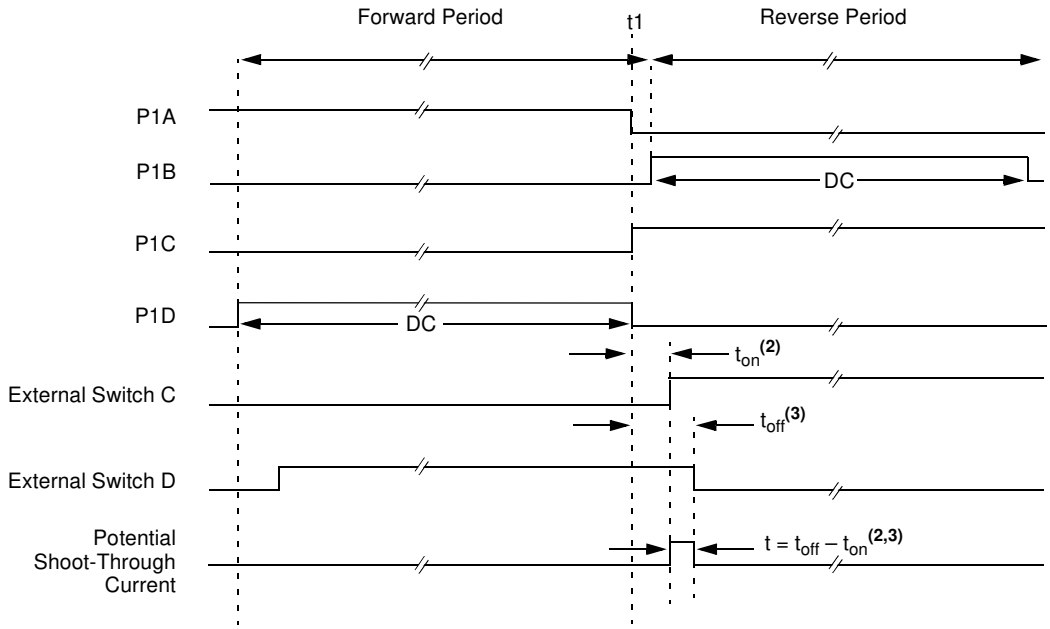
**FIGURE 16-8: PWM DIRECTION CHANGE**




**Note 1:** The direction bit in the CCP1 Control register (CCP1CON<7>) is written any time during the PWM cycle.

**2:** When changing directions, the P1A and P1C signals switch before the end of the current PWM cycle at intervals of 4 TOSC, 16 TOSC or 64 TOSC, depending on the Timer2 prescaler value. The modulated P1B and P1D signals are inactive at this time.

**FIGURE 16-9: PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE(1)**




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

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

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


© 2003 Microchip Technology Inc.

### 16.4.4 PROGRAMMABLE DEAD BAND DELAY

In half-bridge applications, where all power switches are modulated at the PWM frequency at all times, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (*shoot-through current*) may flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In the Half-Bridge Output mode, a digitally programmable dead band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 16-4 for illustration. The lower seven bits of the PWM1CON register (Register 16-2) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

### 16.4.5 ENHANCED PWM AUTO-SHUTDOWN

When the ECCP is programmed for any of the enhanced PWM modes, the active output pins may be configured for auto-shutdown. Auto-shutdown immediately places the enhanced PWM output pins into a defined shutdown state when a shutdown event occurs.

A shutdown event can be caused by either of the two comparator modules or the INT0 pin (or any combination of these three sources). The comparators may be used to monitor a voltage input proportional to a current being monitored in the bridge circuit. If the voltage exceeds a threshold, the comparator switches state and triggers a shutdown. Alternatively, a digital signal on the INT0 pin can also trigger a shutdown. The auto-shutdown feature can be disabled by not selecting any auto-shutdown sources. The auto-shutdown sources to be used are selected using the ECCPAS2:ECCPAS0 bits (ECCPAS<6:4>).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits (ECCPAS<3:0>). Each pin pair (P1A/P1C and P1B/P1D) may be set to drive high, drive low or be tri-stated (not driving). The ECCPASE bit (ECCPAS<7>) is also set to hold the enhanced PWM outputs in their shutdown states.

The ECCPASE bit is set by hardware when a shutdown event occurs. If automatic restarts are not enabled, the ECCPASE bit is cleared by firmware when the cause of the shutdown clears. If automatic restarts are enabled, the ECCPASE bit is automatically cleared when the cause of the auto-shutdown has cleared.

If the ECCPASE bit is set when a PWM period begins, the PWM outputs remain in their shutdown state for that entire PWM period. When the ECCPASE bit is cleared, the PWM outputs will return to normal operation at the beginning of the next PWM period.

| **Note:** | Writing to the ECCPASE bit is disabled while a shutdown condition is active. |
|---|---|

**REGISTER 16-2: PWM1CON: PWM CONFIGURATION REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| PRSEN | PDC6 | PDC5 | PDC4 | PDC3 | PDC2 | PDC1 | PDC0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **PRSEN:** PWM Restart Enable bit
- 1 = Upon auto-shutdown, the ECCPASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically
- 0 = Upon auto-shutdown, ECCPASE must be cleared in software to restart the PWM

bit 6-0 **PDC<6:0>:** PWM Delay Count bits

Number of FOSC/4 (4 * TOSC) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active.

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

**REGISTER 16-3: ECCPAS: ENHANCED CAPTURE/COMPARE/PWM AUTO-SHUTDOWN 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 |
|---|---|---|---|---|---|---|---|
| ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 | PSSBD0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **ECCPASE:** ECCP Auto-Shutdown Event Status bit
0 = ECCP outputs are operating
1 = A shutdown event has occurred; ECCP outputs are in shutdown state

bit 6-4 **ECCPAS<2:0>:** ECCP Auto-Shutdown Source Select bits
000 = Auto-shutdown is disabled
001 = Comparator 1 output
010 = Comparator 2 output
011 = Either Comparator 1 or 2
100 = INT0
101 = INT0 or Comparator 1
110 = INT0 or Comparator 2
111 = INT0 or Comparator 1 or Comparator 2

bit 3-2 **PSSAC<1:0>:** Pin A and C Shutdown State Control bits
00 = Drive Pins A and C to '0'
01 = Drive Pins A and C to '1'
1x = Pins A and C tri-state

bit 1-0 **PSSBD<1:0>:** Pin B and D Shutdown State Control bits
00 = Drive Pins B and D to '0'
01 = Drive Pins B and D to '1'
1x = Pins B and D tri-state

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


© 2003 Microchip Technology Inc.

#### 16.4.5.1 Auto-Shutdown and Automatic Restart

The auto-shutdown feature can be configured to allow automatic restarts of the module following a shutdown event. This is enabled by setting the PRSEN bit of the PWM1CON register (PWM1CON<7>).

In Shutdown mode with PRSEN = 1 (Figure 16-10), the ECCPASE bit will remain set for as long as the cause of the shutdown continues. When the shutdown condition clears, the ECCPASE bit is cleared. If PRSEN = 0 (Figure 16-11), once a shutdown condition occurs, the ECCPASE bit will remain set until it is cleared by firmware. Once ECCPASE is cleared, the enhanced PWM will resume at the beginning of the next PWM period.

| **Note:** | Writing to the ECCPASE bit is disabled while a shutdown condition is active. |
|---|---|

Independent of the PRSEN bit setting, if the auto-shutdown source is one of the comparators, the shutdown condition is a level. The ECCPASE bit cannot be cleared as long as the cause of the shutdown persists.

The Auto-Shutdown mode can be forced by writing a '1' to the ECCPASE bit.

### 16.4.6 START-UP CONSIDERATIONS

When the ECCP module is used in the PWM mode, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins. When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the off state until the microcontroller drives the I/O pins with the proper signal levels or activates the PWM output(s).

The CCP1M1:CCP1M0 bits (CCP1CON<1:0>) allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (P1A/P1C and P1B/P1D). The PWM output polarities must be selected before the PWM pins are configured as outputs. Changing the polarity configuration while the PWM pins are configured as outputs is not recommended since it may result in damage to the application circuits.

The P1A, P1B, P1C and P1D output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pins for output at the same time as the ECCP module may cause damage to the application circuit. The ECCP module must be enabled in the proper output mode and complete a full PWM cycle before configuring the PWM pins as outputs. The completion of a full PWM cycle is indicated by the TMR2IF bit being set as the second PWM period begins.

**FIGURE 16-10: PWM AUTO-SHUTDOWN (PRSEN = 1, AUTO-RESTART ENABLED)**

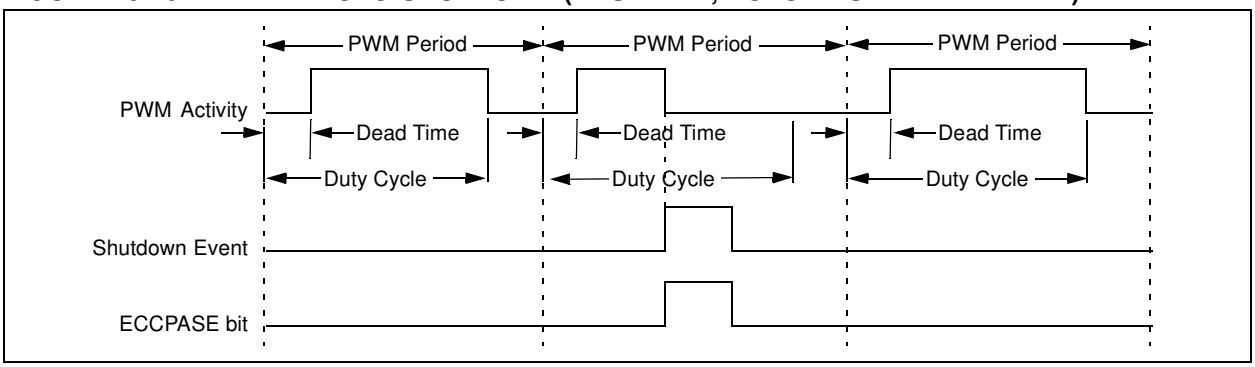
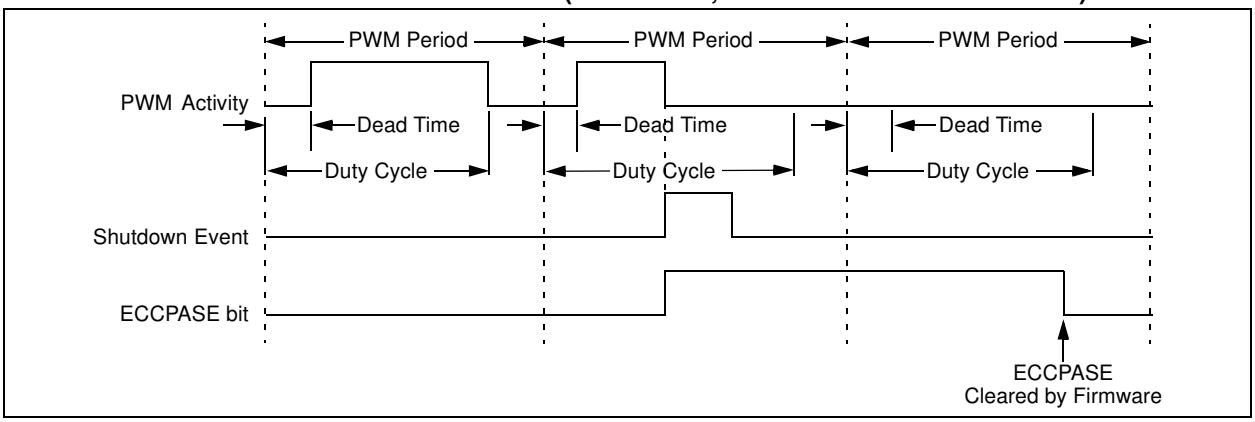



**FIGURE 16-11: PWM AUTO-SHUTDOWN (PRSEN = 0, AUTO-RESTART DISABLED)**



### 16.4.7 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the ECCP1 module for PWM operation:

1. Configure the PWM pins P1A and P1B (and P1C and P1D, if used) as inputs by setting the corresponding TRISC and TRISD bits.
2. Set the PWM period by loading the PR2 register.
3. Configure the ECCP module for the desired PWM mode and configuration by loading the CCP1CON register with the appropriate values:
   - Select one of the available output configurations and direction with the P1M1:P1M0 bits.
   - Select the polarities of the PWM output signals with the CCP1M3:CCP1M0 bits.
4. Set the PWM duty cycle by loading the CCPR1L register and CCP1CON<5:4> bits.
5. For Half-Bridge Output mode, set the dead band delay by loading PWM1CON<6:0> with the appropriate value.
6. If auto-shutdown operation is required, load the ECCPAS register:
   - Select the auto-shutdown sources using the ECCPAS<2:0> bits.
   - Select the shutdown states of the PWM output pins using PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits.
   - Set the ECCPASE bit (ECCPAS<7>).
   - Configure the comparators using the CMCON register.
   - Configure the comparator inputs as analog inputs.
7. If auto-restart operation is required, set the PRSEN bit (PWM1CON<7>).
8. Configure and start TMR2:
   - Clear the TMR2 interrupt flag bit by clearing the TMR2IF bit (PIR1<1>).
   - Set the TMR2 prescale value by loading the T2CKPS bits (T2CON<1:0>).
   - Enable Timer2 by setting the TMR2ON bit (T2CON<2>).
9. Enable PWM outputs after a new PWM cycle has started:
   - Wait until TMR2 overflows (TMR2IF bit is set).
   - Enable the CCP1/P1A, P1B, P1C and/or P1D pin outputs by clearing the respective TRISC and TRISD bits.
   - Clear the ECCPASE bit (ECCPAS<7>).

### 16.4.8 OPERATION IN POWER MANAGED MODES

In Sleep mode, all clock sources are disabled. Timer2 will not increment and the state of the module will not change. If the ECCP pin is driving a value, it will continue to drive that value. When the device wakes up, it will continue from this state. If Two-Speed Start-ups are enabled, the initial start-up frequency from INTOSC and the postscaler may not be stable immediately.

In PRI_IDLE mode, the primary clock will continue to clock the ECCP module without change.

In all other power managed modes, the selected power managed mode clock will clock Timer2. Other power managed mode clocks will most likely be different than the primary clock frequency.

#### 16.4.8.1 OPERATION WITH FAIL-SAFE CLOCK MONITOR

If the Fail-Safe Clock Monitor is enabled (CONFIG1H<6> is programmed), a clock failure will force the device into the RC_RUN Power Managed mode and the OSCFIF bit (PIR2<7>) will be set. The ECCP will then be clocked from the internal oscillator clock source which may have a different clock frequency than the primary clock. By loading the IRCF2:IRCF0 bits on Resets, the user can obtain a frequency higher than the default INTRC clock source in the event of a clock failure.

See the previous section for additional details.

### 16.4.9 EFFECTS OF A RESET

Both Power-on and subsequent Resets will force all ports to Input mode and the CCP registers to their Reset states.

This forces the Enhanced CCP module to reset to a state compatible with the standard CCP module.


© 2003 Microchip Technology Inc.

**TABLE 16-2: REGISTERS ASSOCIATED WITH ENHANCED PWM AND TIMER2**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| RCON | IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ | 0--1 11qq | 0--q qquu |
| PIR1 | PSPIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| PR2 | Timer2 Module Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| T2CON | — | TOUTPS3 | TOUTPS2 | TOUTPS1 | TOUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -000 0000 | -000 0000 |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| TRISD | PORTD Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| CCPR1H | Enhanced Capture/Compare/PWM Register 1 High Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCPR1L | Enhanced Capture/Compare/PWM Register 1 Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| CCP1CON | P1M1 | P1M0 | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 0000 0000 | 0000 0000 |
| ECCPAS | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 | PSSBD0 | 0000 0000 | 0000 0000 |
| PWM1CON | PRSEN | PDC6 | PDC5 | PDC4 | PDC3 | PDC2 | PDC1 | PDC0 | 0000 0000 | 0000 0000 |
| OSCCON | IDLEN | IRCF2 | IRCF1 | IRCF0 | OSTS | IOFS | SCS1 | SCS0 | 0000 q000 | 0000 q000 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are not used by the ECCP module in enhanced PWM mode.

**NOTES:**


© 2003 Microchip Technology Inc.

# 17.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

## 17.1 Master SSP (MSSP) Module Overview

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

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit ($I^2C$)
  - Full Master mode
  - Slave mode (with general address call)

The $I^2C$ interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

## 17.2 Control Registers

The MSSP module has three associated registers. These include a status register (SSPSTAT) and two control registers (SSPCON1 and SSPCON2). The use of these registers and their individual configuration bits differ significantly, depending on whether the MSSP module is operated in SPI or $I^2C$ mode.

Additional details are provided under the individual sections.

## 17.3 SPI Mode

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

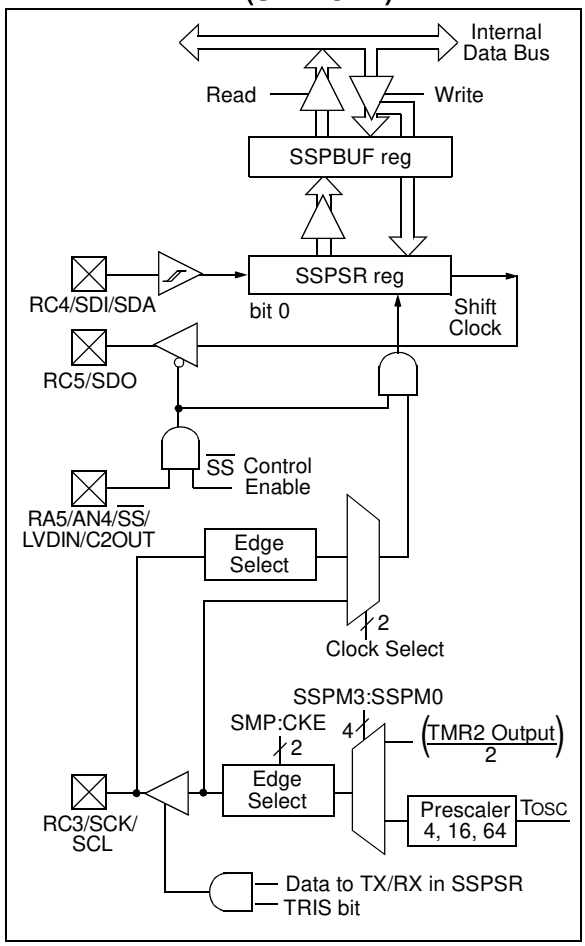
- Serial Data Out (SDO) – RC5/SDO
- Serial Data In (SDI) – RC4/SDI/SDA
- Serial Clock (SCK) – RC3/SCK/SCL

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

- Slave Select ($\overline{SS}$) – RA5/AN4/$\overline{SS}$/LVDIN/C2OUT

Register 17-1 shows the block diagram of the MSSP module when operating in SPI mode.

**FIGURE 17-1: MSSP BLOCK DIAGRAM (SPI MODE)**



### 17.3.1 REGISTERS

The MSSP module has four registers for SPI mode operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower six bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPSR and SSPBUF together create a double-buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double-buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

**REGISTER 17-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)**

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

bit 7 **SMP:** Sample bit
SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Edge Select bit
When CKP = 0:
1 = Data transmitted on rising edge of SCK
0 = Data transmitted on falling edge of SCK
When CKP = 1:
1 = Data transmitted on falling edge of SCK
0 = Data transmitted on rising edge of SCK

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit
Used in $I^2C$ mode only.

bit 4 **P:** Stop bit
Used in $I^2C$ mode only.

bit 3 **S:** Start bit
Used in $I^2C$ mode only.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ bit information
Used in $I^2C$ mode only.

bit 1 **UA:** Update Address bit
Used in $I^2C$ mode only.

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 17-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)**

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

bit 7 **WCOL:** Write Collision Detect bit (Transmit mode only)

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit

SPI Slave mode:

1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode.The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).

0 = No overflow

**Note:** In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.

bit 5 **SSPEN:** Synchronous Serial Port Enable bit

1 = Enables serial port and configures SCK, SDO, SDI and $\overline{SS}$ as serial port pins

0 = Disables serial port and configures these pins as I/O port pins

**Note:** When the MSSP is enabled in SPI mode, these pins must be properly configured as input or output.

bit 4 **CKP:** Clock Polarity Select bit

1 = Idle state for clock is a high level

0 = Idle state for clock is a low level

bit 3-0 **SSPM3:SSPM0:** Synchronous Serial Port Mode Select bits

0101 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control disabled, $\overline{SS}$ can be used as I/O pin

0100 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control enabled

0011 = SPI Master mode, clock = TMR2 output/2

0010 = SPI Master mode, clock = FOSC/64

0001 = SPI Master mode, clock = FOSC/16

0000 = SPI Master mode, clock = FOSC/4

**Note:** Bit combinations not specifically listed here are either reserved or implemented in $I^2C$ mode only.

| **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 |

### 17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

The MSSP consists of a Transmit/Receive Shift register (SSPSR) and a Buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then the Buffer Full Detect bit, BF (SSPSTAT<0>), and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored and the Write Collision Detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

**EXAMPLE 17-1: LOADING THE SSPBUF (SSPSR) REGISTER**

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received(transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF

        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful

        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```


© 2003 Microchip Technology Inc.

### 17.3.3 ENABLING SPI I/O

To enable the serial port, SSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPCON registers and then set the SSPEN bit. This configures the SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed. That is:

- SDI must have TRISC<4> bit set
- SDO must have TRISC<5> bit cleared
- SCK (Master mode) must have TRISC<3> bit cleared
- SCK (Slave mode) must have TRISC<3> bit set
- $\overline{SS}$ must have TRISC<5> bit set

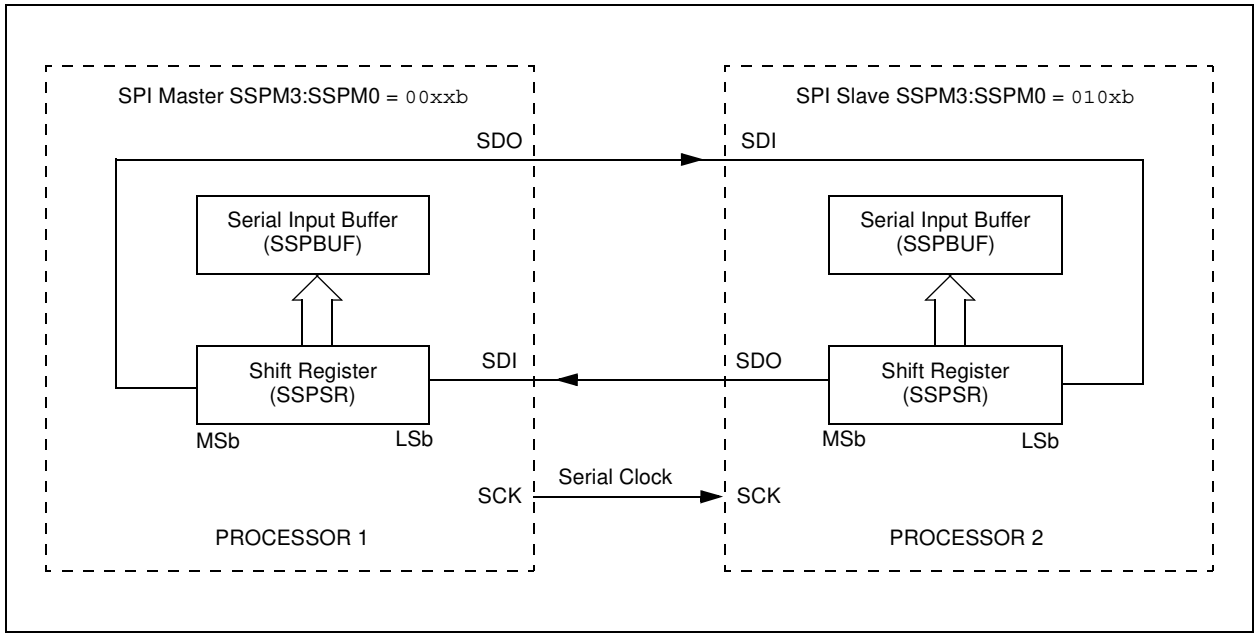
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

### 17.3.4 TYPICAL CONNECTION

Register 17-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

**FIGURE 17-2: SPI MASTER/SLAVE CONNECTION**



### 17.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 17-2) is to broadcast data by the software protocol.

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

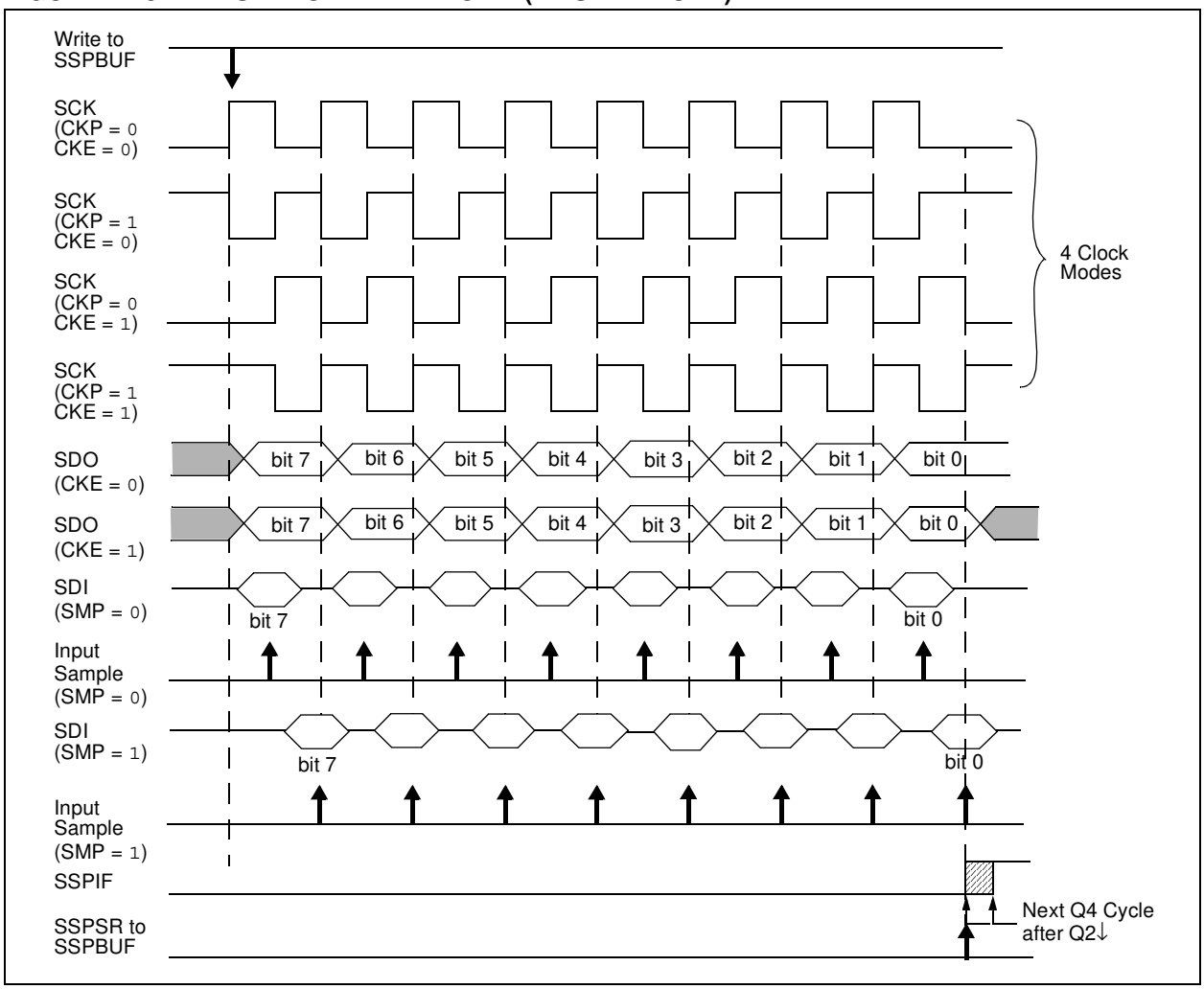
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 17-3, Figure 17-5 and Figure 17-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

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

The maximum data rate is approximately 3.0 Mbps, limited by timing requirements (see Table 26-14 through Table 26-17).

Figure 17-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

**FIGURE 17-3: SPI MODE WAVEFORM (MASTER MODE)**





© 2003 Microchip Technology Inc.

### 17.3.6 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the SSPIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in power managed modes, the slave can transmit/receive data. When a byte is received, the device will wake-up from power managed modes.

### 17.3.7 SLAVE SELECT CONTROL

The $\overline{SS}$ pin allows a master controller to select one of several slave controllers for communications in systems with more than one slave. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 04h). The $\overline{SS}$ pin is configured for input by setting TRISA<5>. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is tri-stated, even if in the middle of a transmitted byte. External pull-up/pull-down resistors may be desirable, depending on the application.

| | |
|---|---|
| **Note 1:** | When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 0100), the SPI module will reset when the $\overline{SS}$ pin is set high. |
| **2:** | If the SPI is used in Slave mode with CKE set, then the $\overline{SS}$ pin control must be enabled. |

When the SPI module resets, SSPSR is cleared. This can be done by either driving the $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.

**FIGURE 17-4: SLAVE SYNCHRONIZATION WAVEFORM**

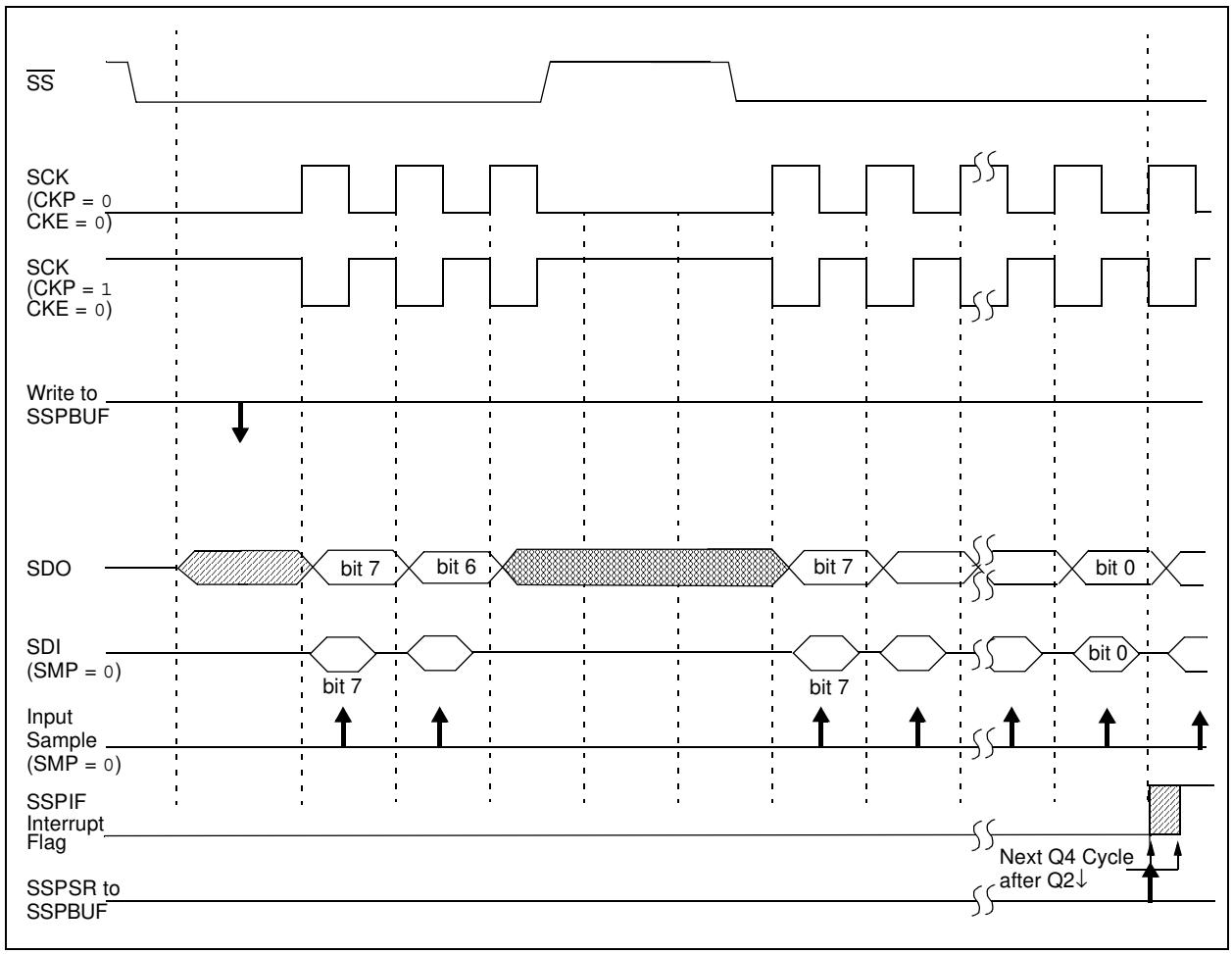


**FIGURE 17-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)**

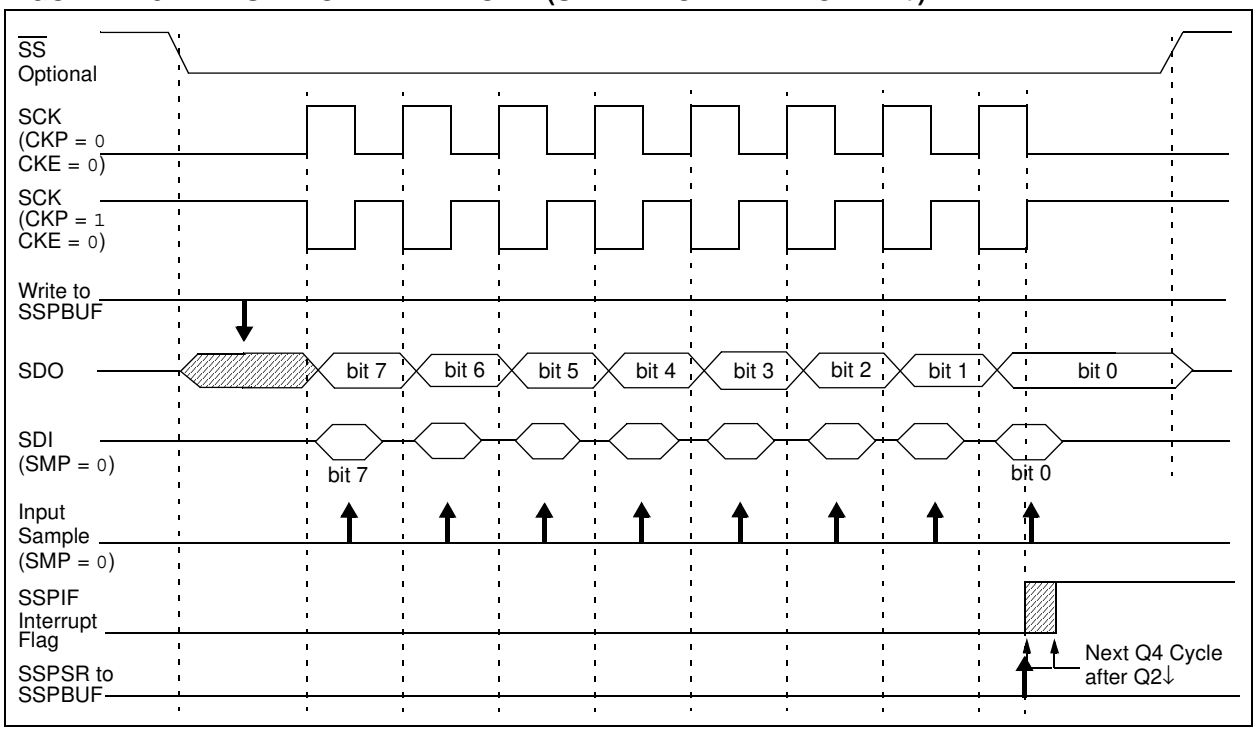
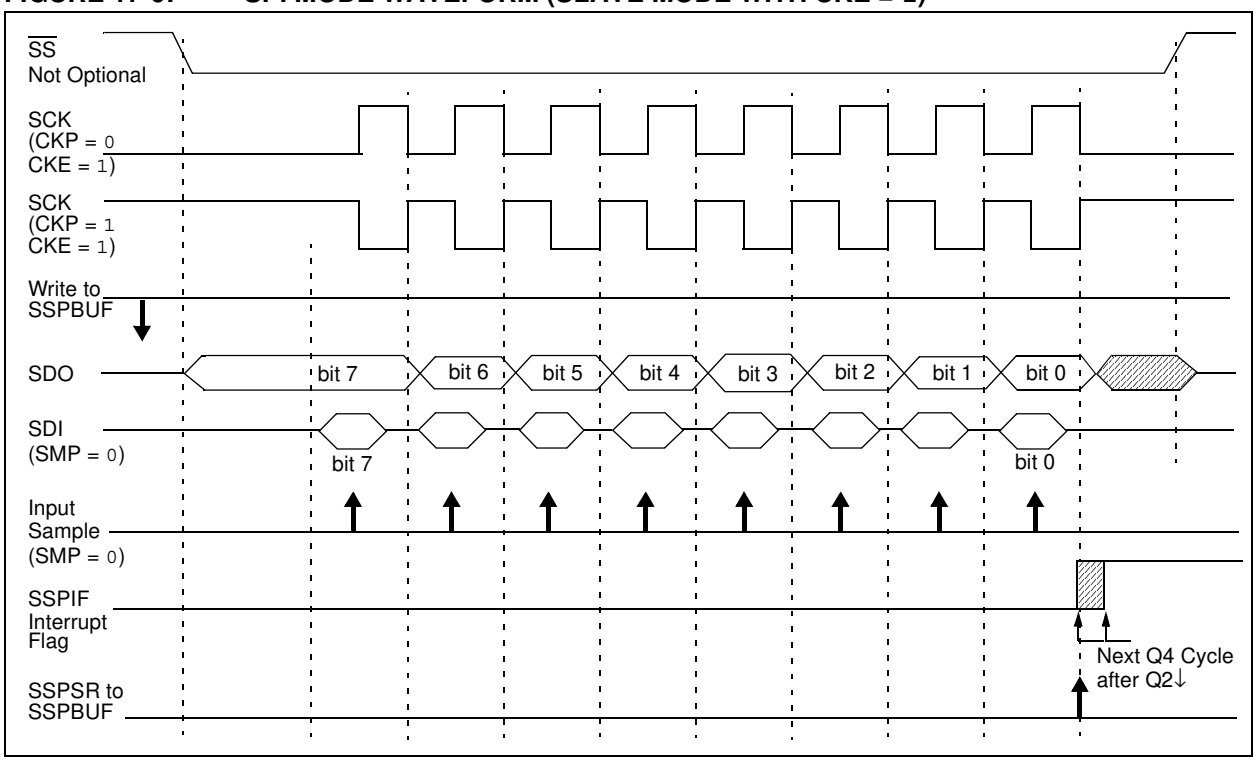



**FIGURE 17-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)**





© 2003 Microchip Technology Inc.

### 17.3.8 MASTER IN POWER MANAGED MODES

In Master mode, module clocks may be operating at a different speed than when in full power mode, or in the case of the Sleep Power Managed mode, all clocks are halted.

In most power managed modes, a clock is provided to the peripherals and is derived from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the internal oscillator block (one of eight frequencies between 31 kHz and 8 MHz). See **Section 2.7 "Clock Sources and Oscillator Switching"** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from a power managed mode when the master completes sending data. If an exit from a power managed mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will pause until the device wakes from the power managed mode. After the device returns to full power mode, the module will resume transmitting and receiving data.

#### 17.3.8.1 Slave in Power Managed Modes

In Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in any power managed mode and data to be shifted into the SPI Transmit/Receive Shift register. When all 8 bits have been received, the MSSP interrupt flag bit will be set and if MSSP interrupts are enabled, will wake the device from a power managed mode.

### 17.3.9 EFFECTS OF A RESET

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

### 17.3.10 BUS MODE COMPATIBILITY

Table 17-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

**TABLE 17-1: SPI BUS MODES**

| Standard SPI Mode Terminology | Control Bits State | |
|---|---|---|
| | CKP | CKE |
| 0, 0 | 0 | 1 |
| 0, 1 | 0 | 0 |
| 1, 0 | 1 | 1 |
| 1, 1 | 1 | 0 |

There is also an SMP bit which controls when the data is sampled.

**TABLE 17-2: REGISTERS ASSOCIATED WITH SPI OPERATION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| SSPBUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| SSPCON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 0000 0000 |
| TRISA | TRISA7(1) | TRISA6(1) | PORTA Data Direction Register | | | | | | --11 1111 | --11 1111 |
| SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the MSSP in SPI mode.
**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

## 17.4 I²C Mode

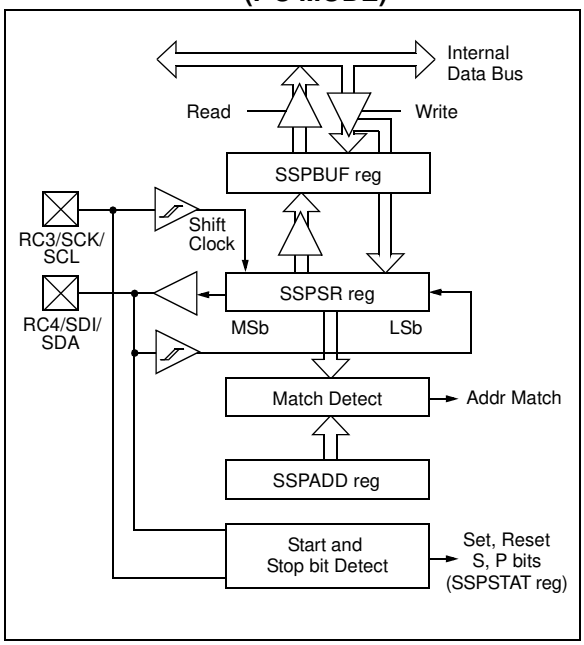
The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial Clock (SCL) – RC3/SCK/SCL
- Serial Data (SDA) – RC4/SDI/SDA

The user must configure these pins as inputs using the TRISC<4:3> bits.

**FIGURE 17-7: MSSP BLOCK DIAGRAM (I²C MODE)**




### 17.4.1 REGISTERS

The MSSP module has six registers for I²C operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON1, SSPCON2 and SSPSTAT are the control and status registers in I²C mode operation. The SSPCON1 and SSPCON2 registers are readable and writable. The lower six bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

SSPADD register holds the slave device address when the SSP is configured in I²C Slave mode. When the SSP is configured in Master mode, the lower seven bits of SSPADD act as the Baud Rate Generator reload value.

In receive operations, SSPSR and SSPBUF together create a double-buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double-buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.


© 2003 Microchip Technology Inc.

**REGISTER 17-3: SSPSTAT: MSSP STATUS REGISTER (I²C MODE)**

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

bit 7 **SMP:** Slew Rate Control bit

In Master or Slave mode:

1 = Slew rate control disabled

0 = Slew rate control enabled

bit 6 **CKE:** SMBus Select bit

In Master or Slave mode:

1 = Enable SMBus specific inputs

0 = Disable SMBus specific inputs

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit

In Master mode:

Reserved.

In Slave mode:

1 = Indicates that the last byte received or transmitted was data

0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit

1 = Indicates that a Stop bit has been detected last

0 = Stop bit was not detected last

**Note:** This bit is cleared on Reset when SSPEN is cleared or a Start bit has been detected.

bit 3 **S:** Start bit

1 = Indicates that a Start bit has been detected last

0 = Start bit was not detected last

**Note:** This bit is cleared on Reset when SSPEN is cleared or a Stop bit has been detected.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ bit Information (I²C mode only)

In Slave mode:

1 = Read

0 = Write

**Note:** This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not $\overline{\text{ACK}}$ bit.

In Master mode:

1 = Transmit is in progress

0 = Transmit is not in progress

**Note:** OR'ing this bit with the SSPCON2 bits, SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Idle mode.

bit 1 **UA:** Update Address (10-bit Slave mode only)

1 = Indicates that the user needs to update the address in the SSPADD register

0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit

In Transmit mode:

1 = Data transmit in progress (does not include the $\overline{\text{ACK}}$ and Stop bits), SSPBUF is full

0 = Data transmit complete (does not include the $\overline{\text{ACK}}$ and Stop bits), SSPBUF is empty

In Receive mode:

1 = Receive complete, SSPBUF is full

0 = Receive not complete, SSPBUF is empty

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

**REGISTER 17-4: SSPCON1: MSSP CONTROL REGISTER 1 ($I^2C$ MODE)**

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

bit 7 **WCOL:** Write Collision Detect bit

In Master Transmit mode:

1 = A write to the SSPBUF register was attempted while the $I^2C$ conditions were not valid for a transmission to be started (must be cleared in software)

0 = No collision

In Slave Transmit mode:

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

In Receive mode (Master or Slave modes):

This is a “don’t care” bit.

bit 6 **SSPOV:** Receive Overflow Indicator bit

In Receive mode:

1 = A byte is received while the SSPBUF register is still holding the previous byte (must be cleared in software)

0 = No overflow

In Transmit mode:

This is a “don’t care” bit in Transmit mode.

bit 5 **SSPEN:** Synchronous Serial Port Enable bit

1 = Enables the serial port and configures the SDA and SCL pins as the serial port pins

0 = Disables serial port and configures these pins as I/O port pins

**Note:** When enabled, the SDA and SCL pins must be configured as input pins.

bit 4 **CKP:** SCK Release Control bit

In Slave mode:

1 = Release clock

0 = Holds clock low (clock stretch), used to ensure data setup time

In Master mode:

Unused in this mode.

bit 3-0 **SSPM3:SSPM0:** Synchronous Serial Port Mode Select bits

1111 = $I^2C$ Slave mode, 10-bit address with Start and Stop bit interrupts enabled

1110 = $I^2C$ Slave mode, 7-bit address with Start and Stop bit interrupts enabled

1011 = $I^2C$ Firmware Controlled Master mode (Slave Idle)

1000 = $I^2C$ Master mode, clock = $F_{OSC}/(4 * (SSPADD + 1))$

0111 = $I^2C$ Slave mode, 10-bit address

0110 = $I^2C$ Slave mode, 7-bit address

**Note:** Bit combinations not specifically listed here are either reserved, or implemented in SPI mode only.

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 17-5: SSPCON2: MSSP CONTROL REGISTER 2 ($I^2C$ MODE)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| bit 7 | | | | | | | bit 0 |

bit 7 **GCEN:** General Call Enable bit (Slave mode only)
1 = Enable interrupt when a general call address (0000h) is received in the SSPSR
0 = General call address disabled

bit 6 **ACKSTAT:** Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave

bit 5 **ACKDT:** Acknowledge Data bit (Master Receive mode only)
1 = Not Acknowledge
0 = Acknowledge

**Note:** Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.

bit 4 **ACKEN:** Acknowledge Sequence Enable bit (Master Receive mode only)
1 = Initiate Acknowledge sequence on SDA and SCL pins and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence Idle

bit 3 **RCEN:** Receive Enable bit (Master mode only)
1 = Enables Receive mode for $I^2C$
0 = Receive Idle

bit 2 **PEN:** Stop Condition Enable bit (Master mode only)
1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enabled bit (Master mode only)
1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enabled/Stretch Enabled bit
In Master mode:
1 = Initiate Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Start condition Idle
In Slave mode:
1 = Clock stretching is enabled for both Slave Transmit and Slave Receive (stretch enabled)
0 = Clock stretching is disabled

**Note:** For bits ACKEN, RCEN, PEN, RSEN, SEN: If the $I^2C$ module is not in the Idle mode, this bit may not be set (no spooling) and the SSPBUF may not be written (or writes to the SSPBUF are disabled).

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

### 17.4.2 OPERATION

The MSSP module functions are enabled by setting MSSP Enable bit, SSPEN (SSPCON1<5>).

The SSPCON1 register allows control of the $I^2C$ operation. Four mode selection bits (SSPCON1<3:0>) allow one of the following $I^2C$ modes to be selected:

- $I^2C$ Master mode, clock = FOSC/(4 * (SSPADD + 1))
- $I^2C$ Slave mode (7-bit address)
- $I^2C$ Slave mode (10-bit address)
- $I^2C$ Slave mode (7-bit address), with Start and Stop bit interrupts enabled
- $I^2C$ Slave mode (10-bit address), with Start and Stop bit interrupts enabled
- $I^2C$ Firmware Controlled Master mode, slave is Idle

Selection of any $I^2C$ mode, with the SSPEN bit set, forces the SCL and SDA pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCL and SDA pins.

### 17.4.3 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISC<4:3> set). The MSSP module will override the input state with the output data when required (slave-transmitter).

The $I^2C$ Slave mode hardware will always generate an interrupt on an address match. Through the mode select bits, the user can also choose to interrupt on Start and Stop bits.

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

Any combination of the following conditions will cause the MSSP module not to give this $\overline{ACK}$ pulse:

- The Buffer Full bit, BF (SSPSTAT<0>), was set before the transfer was received.
- The overflow bit, SSPOV (SSPCON1<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF but bit SSPIF (PIR1<3>) is set. The BF bit is cleared by reading the SSPBUF register, while bit SSPOV is cleared by software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the $I^2C$ specification, as well as the requirement of the MSSP module, are shown in timing parameter #100 and parameter #101.

#### 17.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPSR register value is loaded into the SSPBUF register.
2. The Buffer Full bit, BF, is set.
3. An $\overline{ACK}$ pulse is generated.
4. MSSP Interrupt Flag bit, SSPIF (PIR1<3>), is set (interrupt is generated if enabled) on the falling edge of the ninth SCL pulse.

In 10-bit Address mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{W}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '11110 A9 A8 0', where 'A9' and 'A8' are the two MSbs of the address. The sequence of events for 10-bit address is as follows, with steps 7 through 9 for the slave-transmitter:

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


© 2003 Microchip Technology Inc.

#### 17.4.3.2 Reception

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

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit, BF (SSPSTAT<0>), is set or bit, SSPOV (SSPCON1<6>), is set.

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

If SEN is enabled (SSPCON2<0> = 1), RC3/SCK/SCL will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPCON1<4>). See **Section 17.4.4 "Clock Stretching"** for more detail.

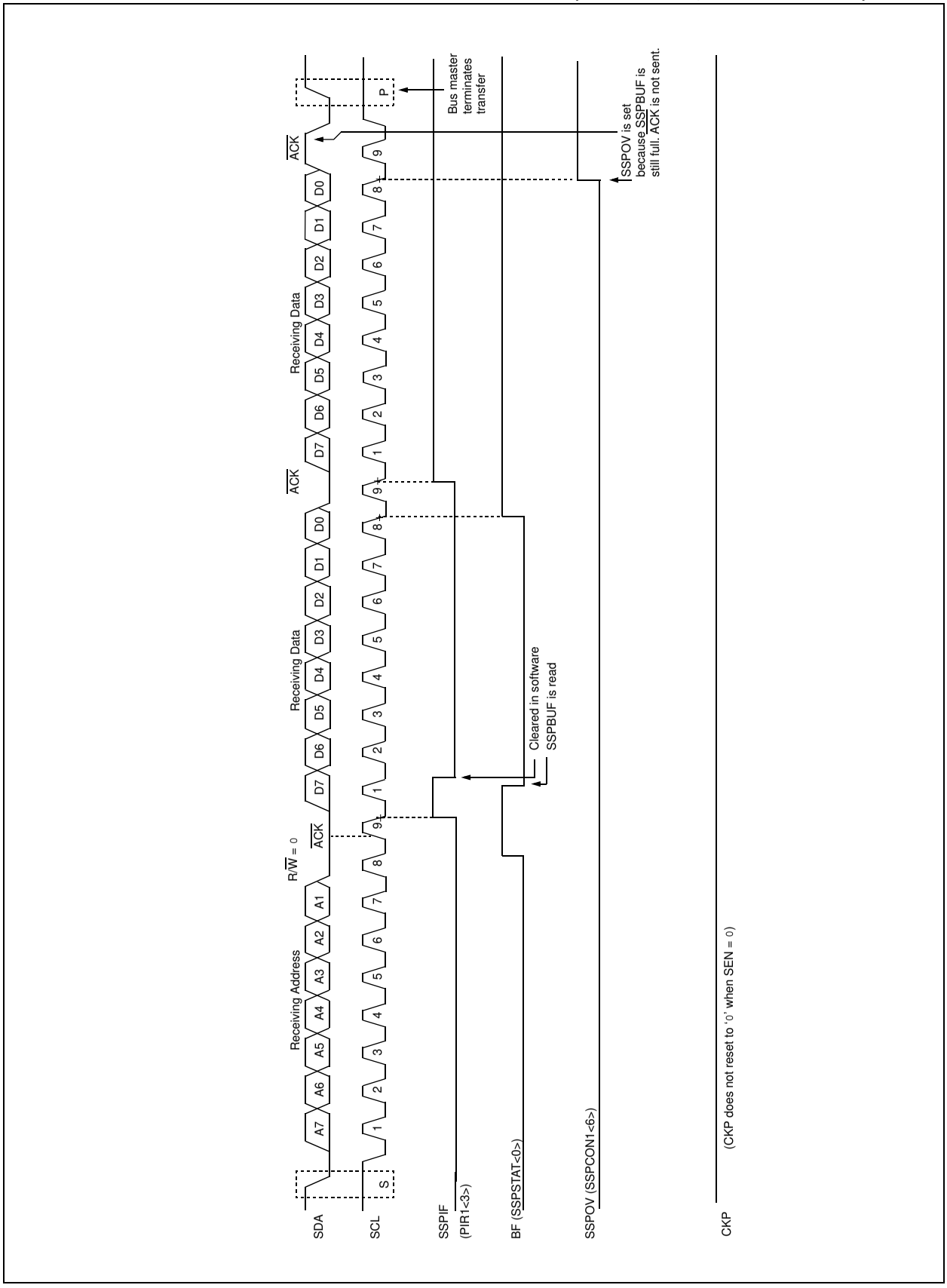
#### 17.4.3.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RC3/SCK/SCL is held low regardless of SEN (see **Section 17.4.4 "Clock Stretching"** for more detail). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then pin RC3/SCK/SCL should be enabled by setting bit, CKP (SSPCON1<4>). The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 17-9).

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not $\overline{ACK}$), then the data transfer is complete. In this case, when the $\overline{ACK}$ is latched by the slave, the slave logic is reset (resets SSPSTAT register) and the slave monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPBUF register. Again, pin RC3/SCK/SCL must be enabled by setting bit CKP.

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

**FIGURE 17-8: $I^2C$ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)**





© 2003 Microchip Technology Inc.

**FIGURE 17-9: I²C SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)**

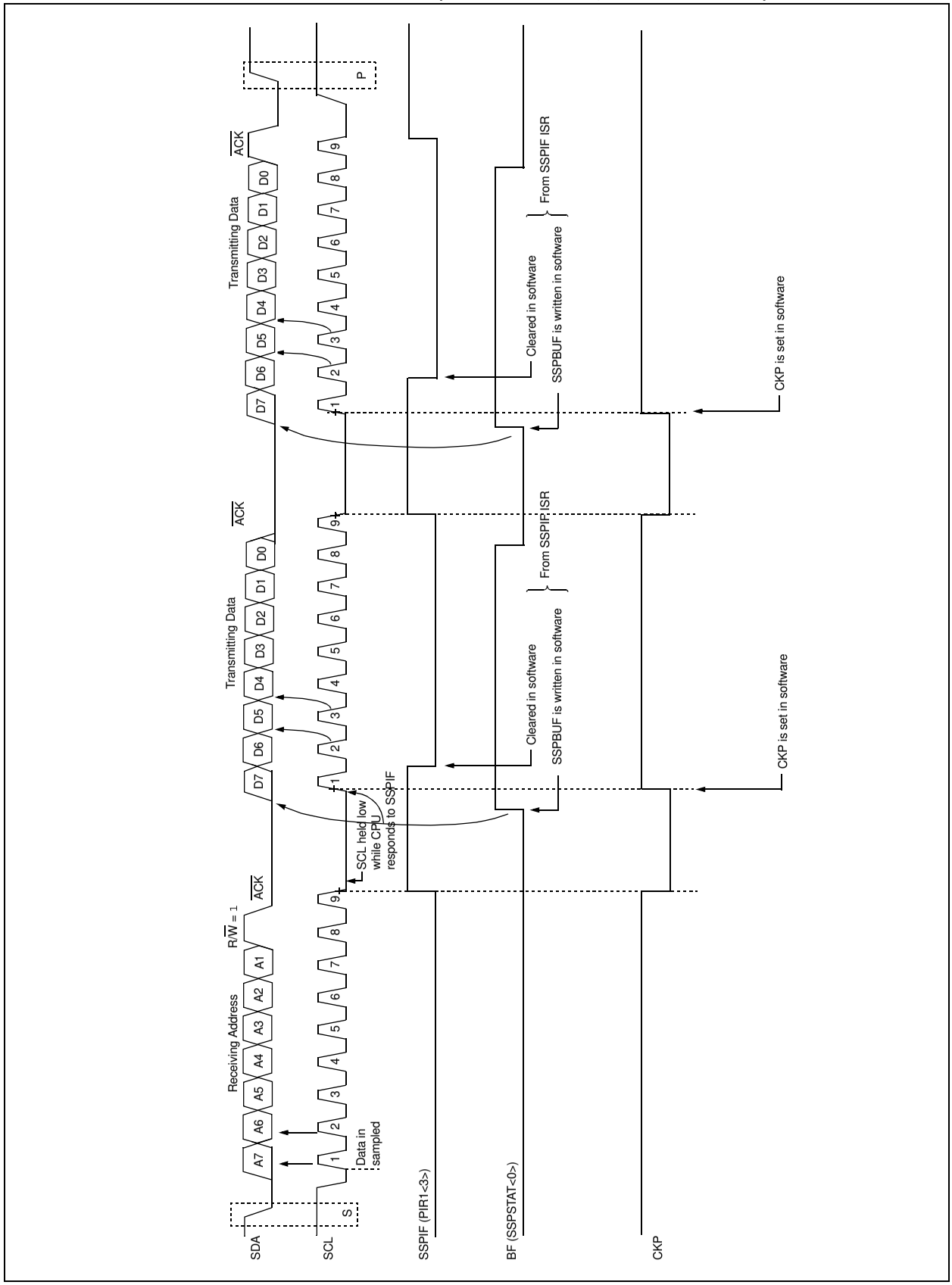
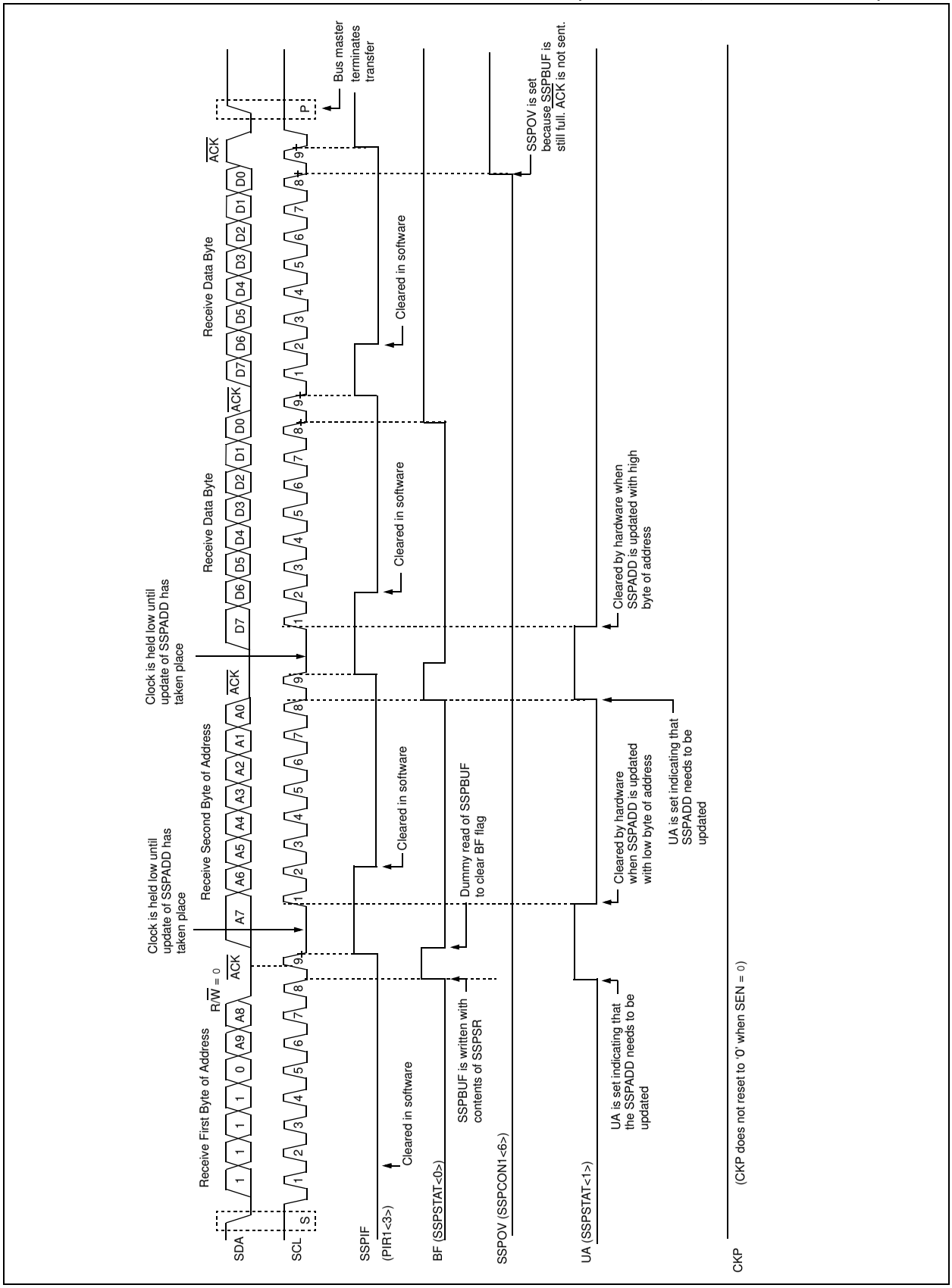


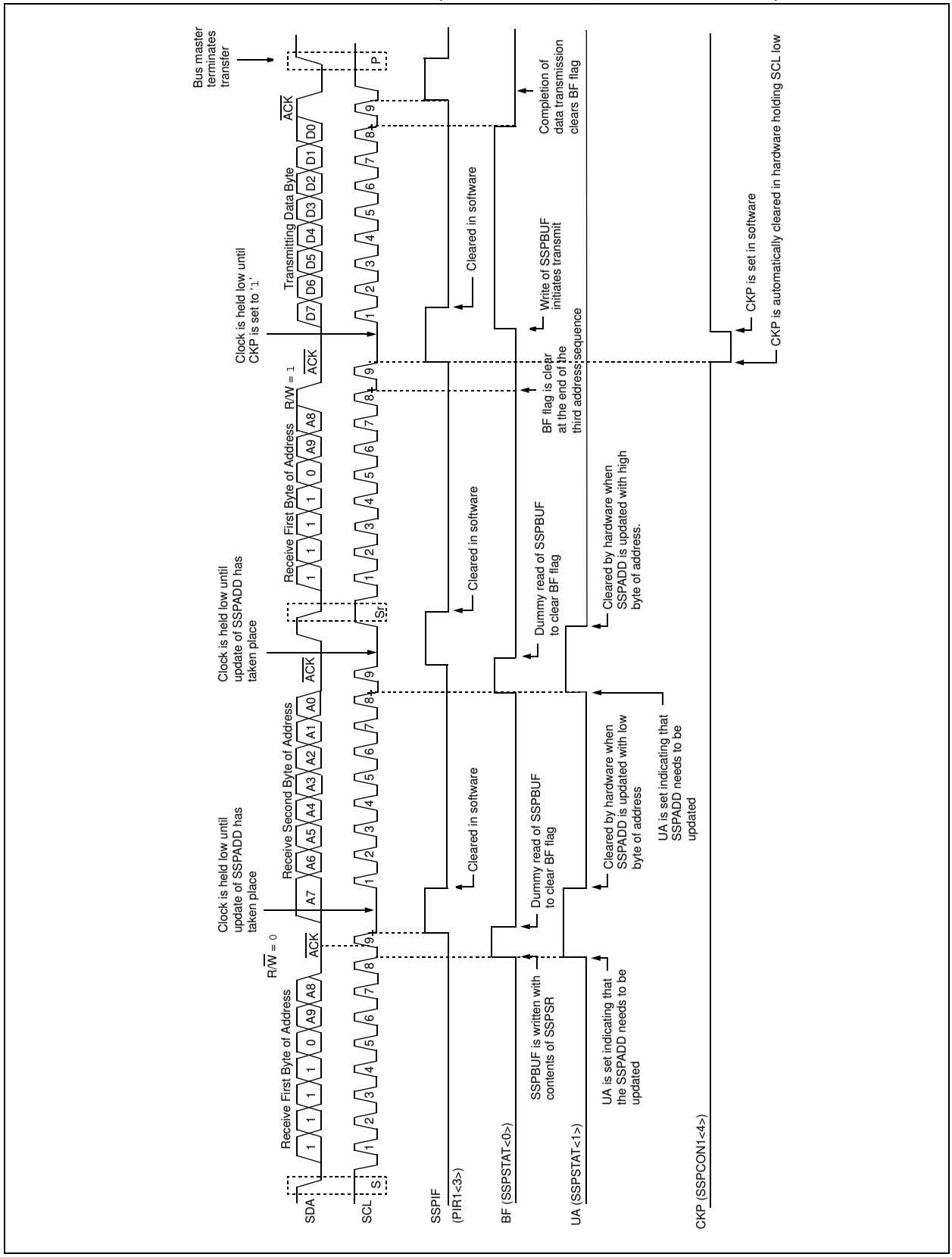
**FIGURE 17-10: $I^2C$ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)**





© 2003 Microchip Technology Inc.

**FIGURE 17-11: I²C SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)**



### 17.4.4 CLOCK STRETCHING

Both 7 and 10-bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

#### 17.4.4.1 Clock Stretching for 7-bit Slave Receive Mode (SEN = 1)

In 7-bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-13).

| | |
|---|---|
| **Note 1:** | If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur. |
| **2:** | The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition. |

#### 17.4.4.2 Clock Stretching for 10-bit Slave Receive Mode (SEN = 1)

In 10-bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but the CKP bit is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

| | |
|---|---|
| **Note:** | If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence. |

#### 17.4.4.3 Clock Stretching for 7-bit Slave Transmit Mode

7-bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-9).

| | |
|---|---|
| **Note 1:** | If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur. |
| **2:** | The CKP bit can be set in software regardless of the state of the BF bit. |

#### 17.4.4.4 Clock Stretching for 10-bit Slave Transmit Mode

In 10-bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-bit Slave Transmit mode (see Figure 17-11).


© 2003 Microchip Technology Inc.

#### 17.4.4.5 Clock Synchronization and the CKP bit (SEN = 1)

The SEN bit is also used to synchronize writes to the CKP bit. If a user clears the CKP bit, the SCL output is forced to '0'. When the SEN bit is set to '1', setting the CKP bit will not assert the SCL output low until the SCL output is already sampled low. If the user attempts to drive SCL low, the CKP bit will not assert the SCL line until an external $I^2C$ master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the $I^2C$ bus have deasserted SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 17-12).

| **Note:** | If the SEN bit is '0', clearing the CKP bit will result in immediately driving the SCL output to '0' regardless of the current state. |
|---|---|

**FIGURE 17-12: CLOCK SYNCHRONIZATION TIMING**

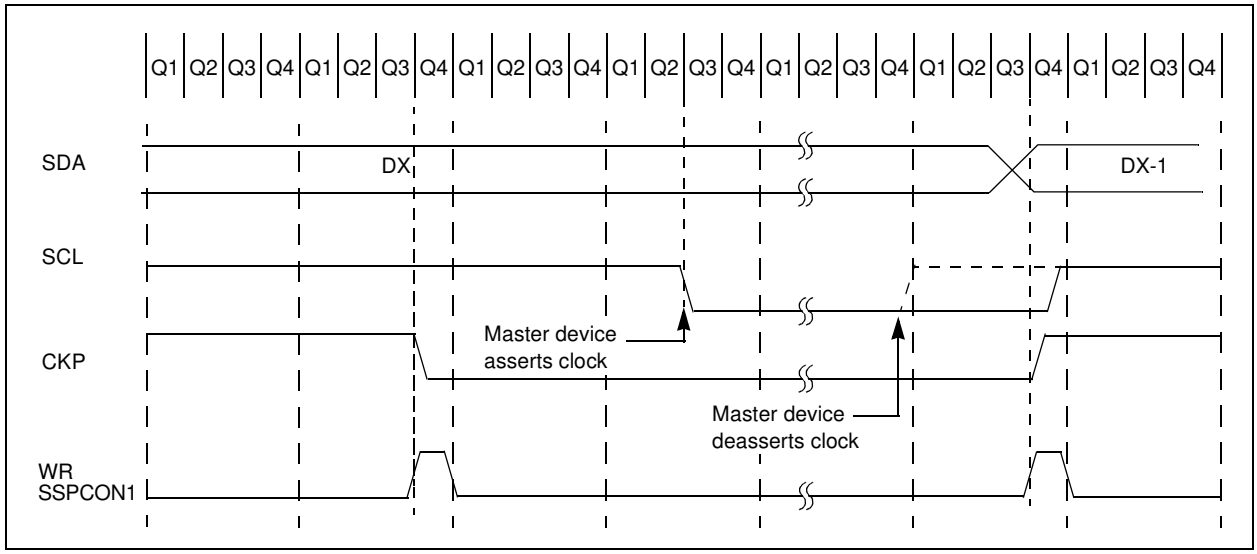
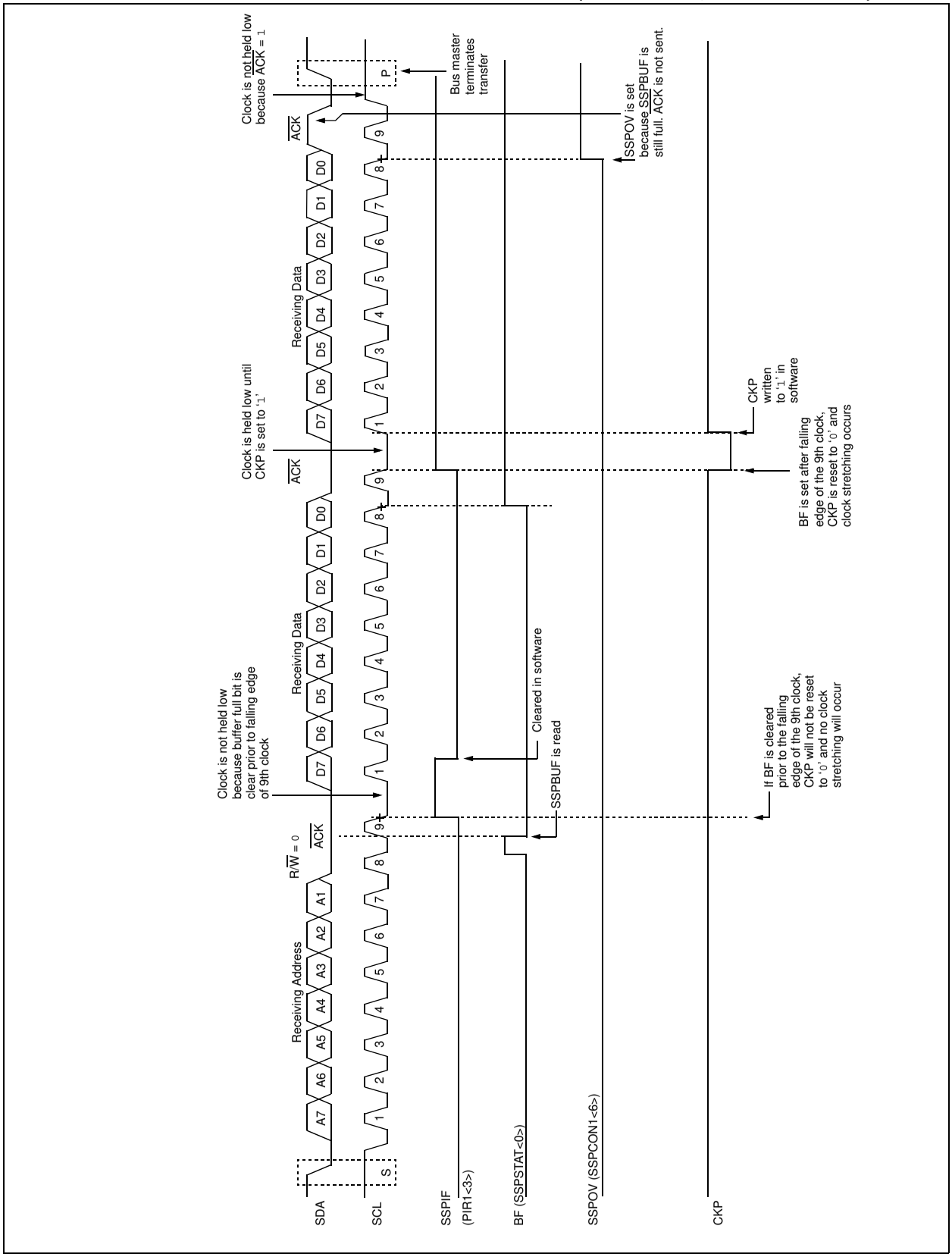


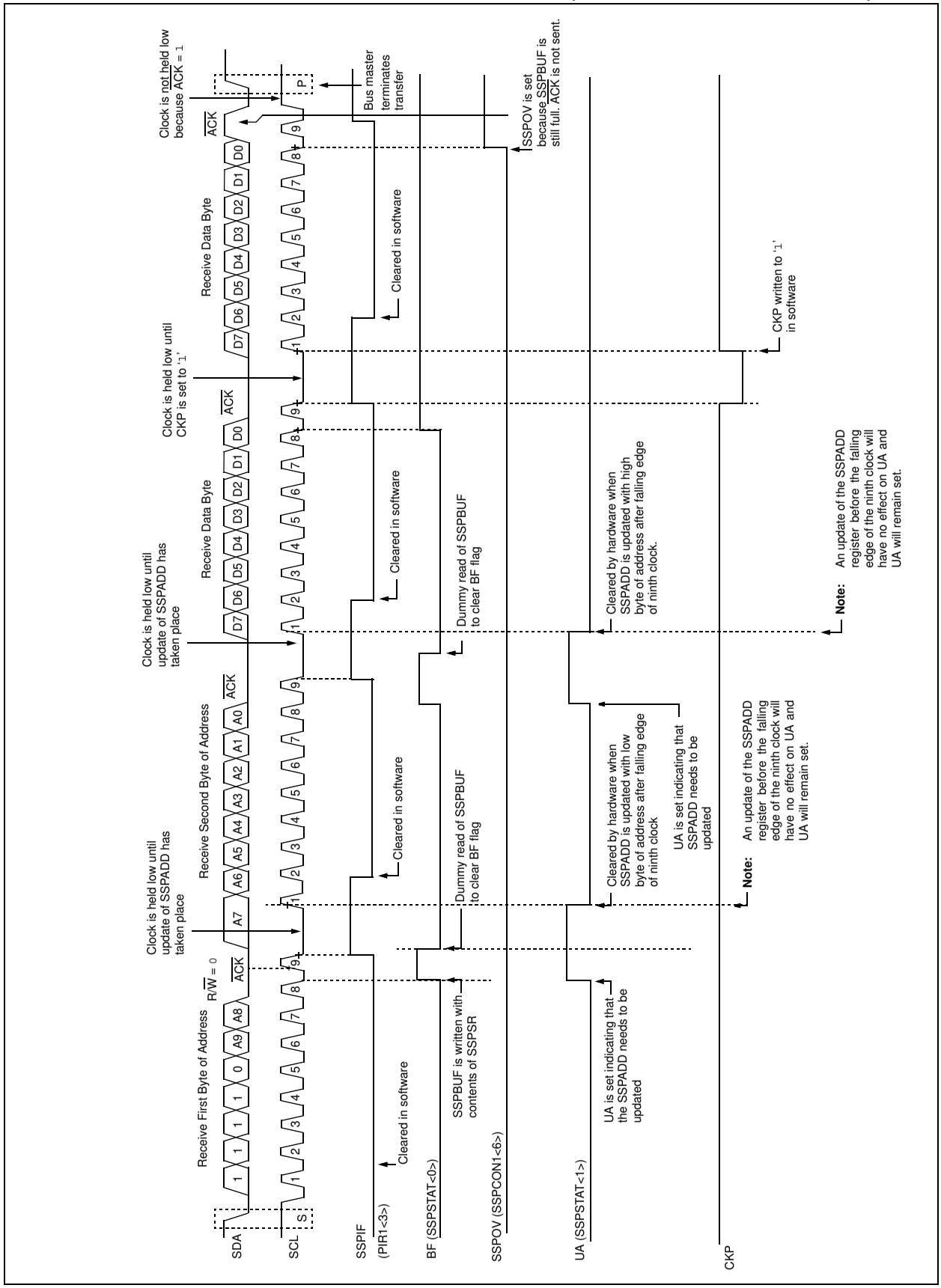
**FIGURE 17-13: $I^2C$ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)**





© 2003 Microchip Technology Inc.

**FIGURE 17-14: I²C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)**



### 17.4.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the $I^2C$ bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address, which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the $I^2C$ protocol. It consists of all '0's with R/$\overline{W}$ = 0.

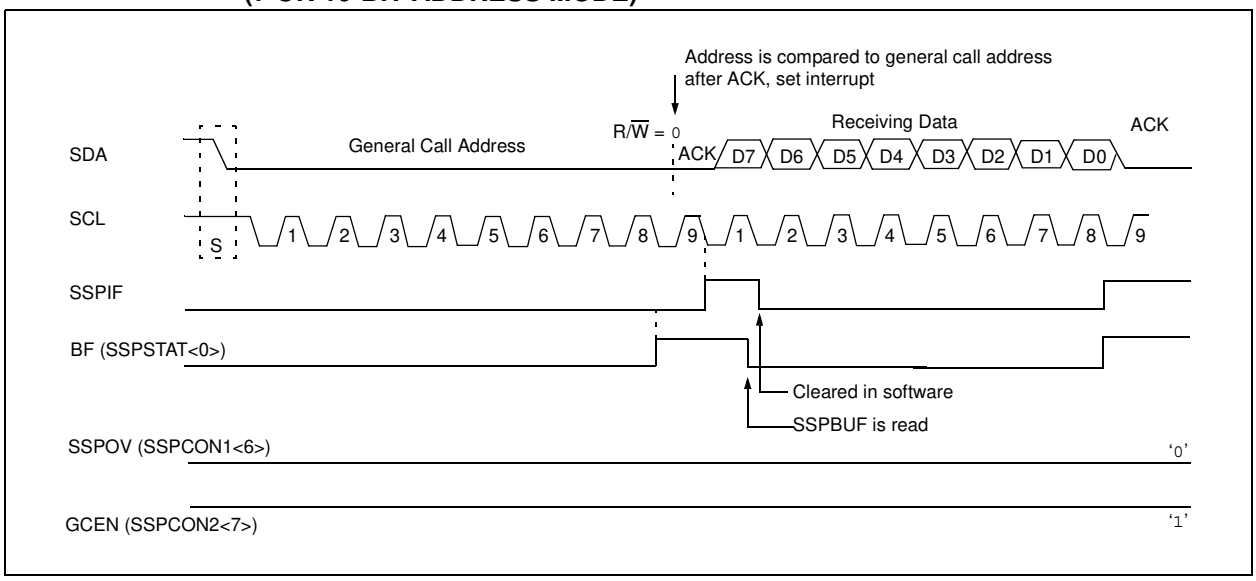
The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPBUF. The value can be used to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPADD is required to be updated for the second half of the address to match and the UA bit is set (SSPSTAT<1>). If the general call address is sampled when the GCEN bit is set while the slave is configured in 10-bit Address mode, then the second half of the address is not necessary, the UA bit will not be set and the slave will begin receiving data after the Acknowledge (Figure 17-15).

**FIGURE 17-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESS MODE)**





© 2003 Microchip Technology Inc.

### 17.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the $I^2C$ bus may be taken when the P bit is set or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all $I^2C$ bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

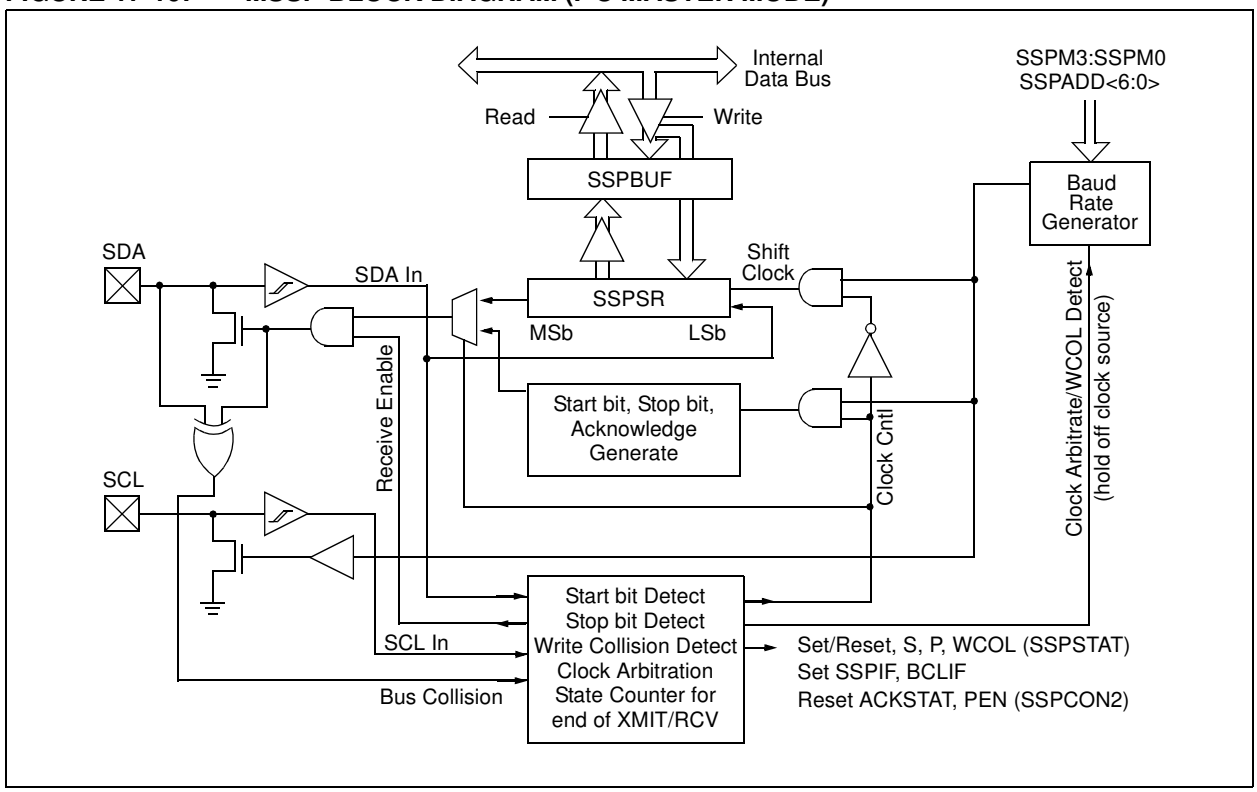
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the $I^2C$ port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

**Note:** The MSSP module, when configured in $I^2C$ Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

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

- Start Condition
- Stop Condition
- Data Transfer Byte Transmitted/Received
- Acknowledge Transmit
- Repeated Start

**FIGURE 17-16: MSSP BLOCK DIAGRAM ($I^2C$ MASTER MODE)**



#### 17.4.6.1 I²C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator used for the SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 17.4.7 "Baud Rate"** for more detail.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. Address is shifted out the SDA pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with eight bits of data.
8. Data is shifted out the SDA pin until all 8 bits are transmitted.
9. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
11. The user generates a Stop condition by setting the Stop Enable bit, PEN (SSPCON2<2>).
12. Interrupt is generated once the Stop condition is complete.


© 2003 Microchip Technology Inc.

### 17.4.7 BAUD RATE

In I²C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPADD register (Figure 17-17). When a write occurs to SSPBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to '0' and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (TCY) on the Q2 and Q4 clocks. In I²C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by $\overline{ACK}$), the internal clock will automatically stop counting and the SCL pin will remain in its last state.

Table 17-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD.

#### 17.4.7.1 Baud Rate Generation in Power Managed Modes

When the device is operating in a power managed mode, the clock source to the Baud Rate Generator may change frequency or stop, depending on the power managed mode and clock source selected.

In most power modes, the Baud Rate Generator continues to be clocked but may be clocked from the primary clock (selected in a configuration word), the secondary clock (Timer1 oscillator at 32.768 kHz) or the internal oscillator block (one of eight frequencies between 31 kHz and 8 MHz). If the Sleep mode is selected, all clocks are stopped and the Baud Rate Generator will not be clocked.

**FIGURE 17-17: BAUD RATE GENERATOR BLOCK DIAGRAM**

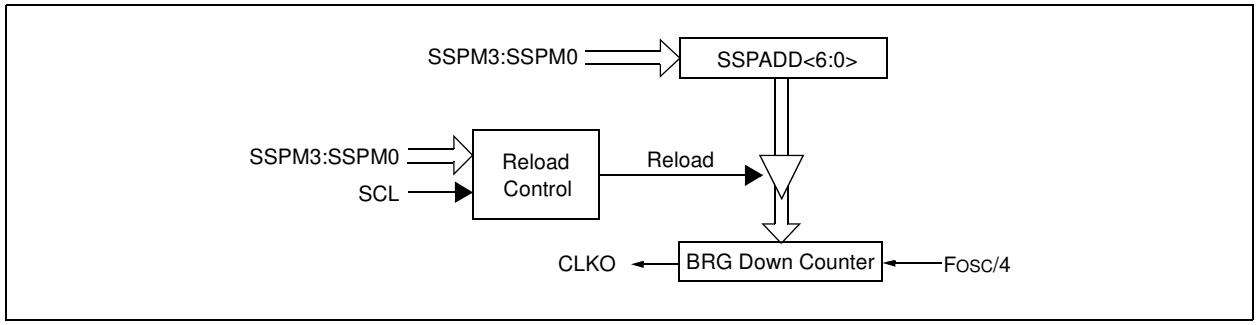



**TABLE 17-3: I²C CLOCK RATE W/BRG**

| FOSC | FCY | FCY*2 | SSPADD VALUE (See Register 17-4, Mode 1000) | FSCL(2) (2 Rollovers of BRG) |
|---|---|---|---|---|
| 40 MHz | 10 MHz | 20 MHz | 18h | 400 kHz(1) |
| 40 MHz | 10 MHz | 20 MHz | 1Fh | 312.5 kHz |
| 40 MHz | 10 MHz | 20 MHz | 63h | 100 kHz |
| 16 MHz | 4 MHz | 8 MHz | 09h | 400 kHz(1) |
| 16 MHz | 4 MHz | 8 MHz | 0Bh | 308 kHz |
| 16 MHz | 4 MHz | 8 MHz | 27h | 100 kHz |
| 4 MHz | 1 MHz | 2 MHz | 02h | 333 kHz(1) |
| 4 MHz | 1 MHz | 2 MHz | 09h | 100kHz |
| 4 MHz | 1 MHz | 2 MHz | 00h | 1 MHz(1) |

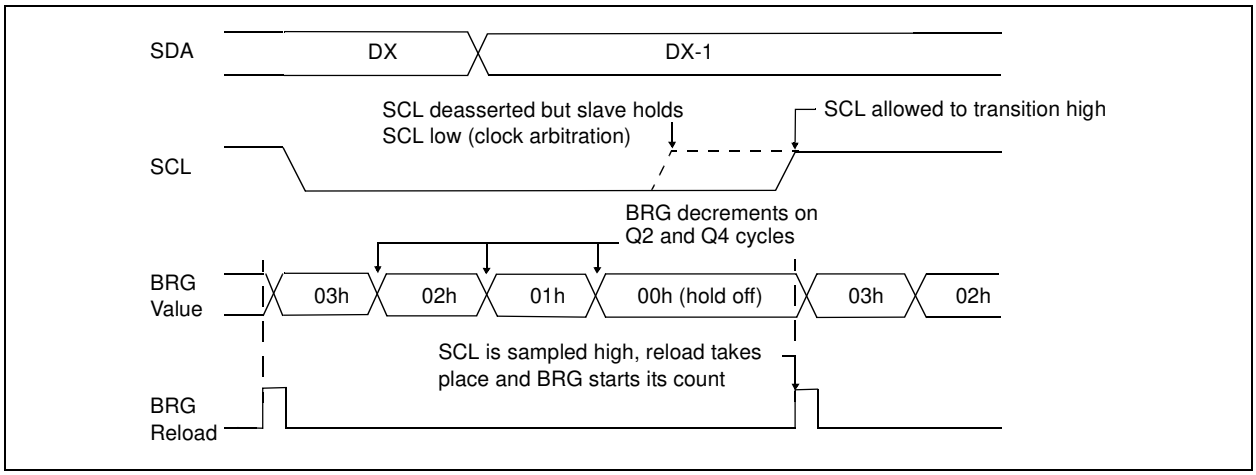
**Note 1:** The I²C interface does not conform to the 400 kHz I²C specification (which applies to rates greater than 100 kHz) in all details, but may be used with care where higher rates are required by the application.

**2:** Actual clock rate will depend on bus conditions. Bus capacitance can increase rise time and extend the low time of the clock period, reducing the effective clock frequency (see **Section 17.4.7.2 "Clock Arbitration"**).

#### 17.4.7.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 17-18).

**FIGURE 17-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**





© 2003 Microchip Technology Inc.

### 17.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Condition Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware, the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

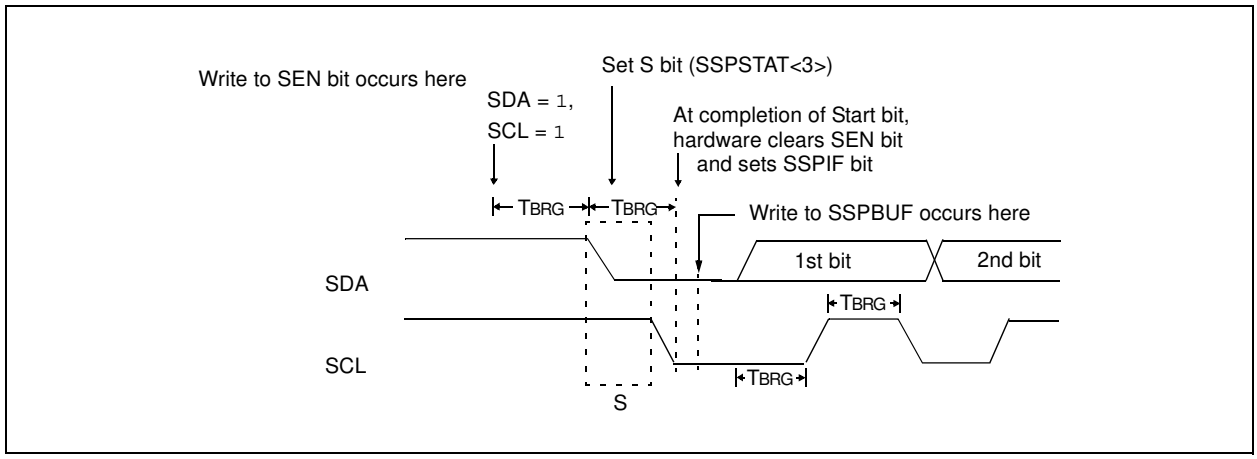
| **Note:** | If at the beginning of the Start condition, the SDA and SCL pins are already sampled low or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state. |
|---|---|

#### 17.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

| **Note:** | Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete. |
|---|---|

**FIGURE 17-19: FIRST START BIT TIMING**



### 17.4.9 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPADD<5:0> and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = 0) for one TBRG while SCL is high. Following this, the RSEN bit (SSPCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the Baud Rate Generator has timed out.

**Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.

**2:** A bus collision during the Repeated Start condition occurs if:

- SDA is sampled low when SCL goes from low to high.
- SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data '1'.

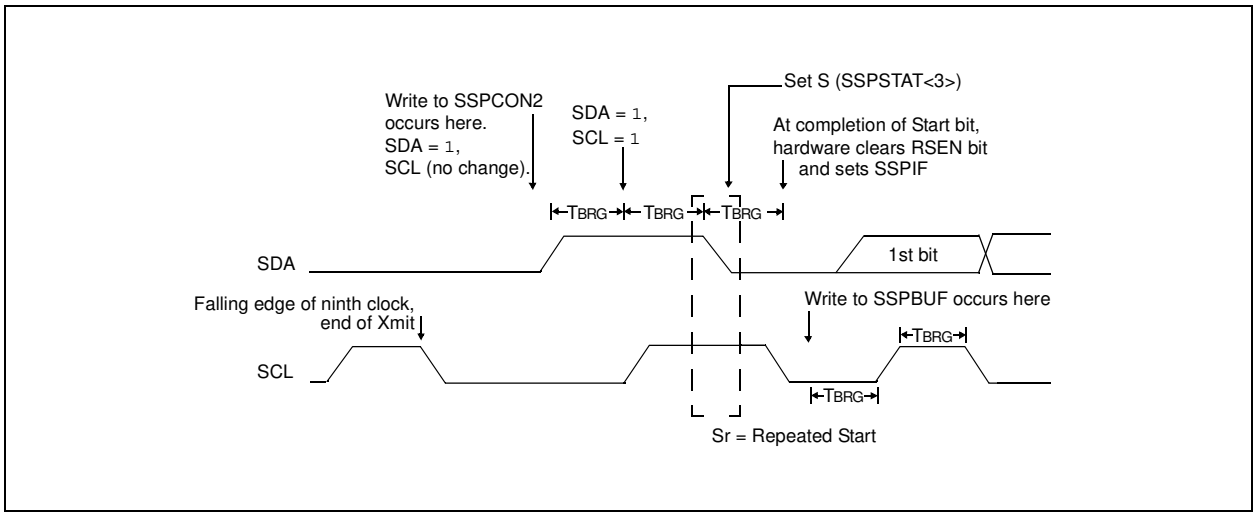
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

#### 17.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

**Note:** Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

**FIGURE 17-20: REPEAT START CONDITION WAVEFORM**





© 2003 Microchip Technology Inc.

### 17.4.10 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPBUF register. This action will set the Buffer Full Flag bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted (see data hold time specification parameter #106). SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high (see data setup time specification parameter #107). When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. This allows the slave device being addressed to respond with an $\overline{ACK}$ bit, during the ninth bit time, if an address match occurred or if data was received properly. The status of $\overline{ACK}$ is written into the ACKDT bit on the falling edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared; if not, the bit is set. After the ninth clock, the SSPIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPBUF, leaving SCL low and SDA unchanged (Figure 17-21).

After the write to the SSPBUF, each bit of address will be shifted out on the falling edge of SCL until all seven address bits and the R/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the master will deassert the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the ACK bit is loaded into the ACKSTAT status bit (SSPCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

#### 17.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPSTAT<0>) is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

#### 17.4.10.2 WCOL Status Flag

If the user writes the SSPBUF when a transmit is already in progress (i.e., SSPSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

WCOL must be cleared in software.

#### 17.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPCON2<6>) is cleared when the slave has sent an Acknowledge ($\overline{ACK}$ = 0) and is set when the slave does not Acknowledge ($\overline{ACK}$ = 1). A slave sends an Acknowledge when it has recognized its address (including a general call) or when the slave has properly received its data.

### 17.4.11 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPCON2<3>).

| **Note:** | The MSSP module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded. |
|---|---|

The Baud Rate Generator begins counting and on each rollover, the state of the SCL pin changes (high to low/low to high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR are loaded into the SSPBUF, the BF flag bit is set, the SSPIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCL low. The MSSP is now in Idle state, awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>).

#### 17.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPBUF from SSPSR. It is cleared when the SSPBUF register is read.

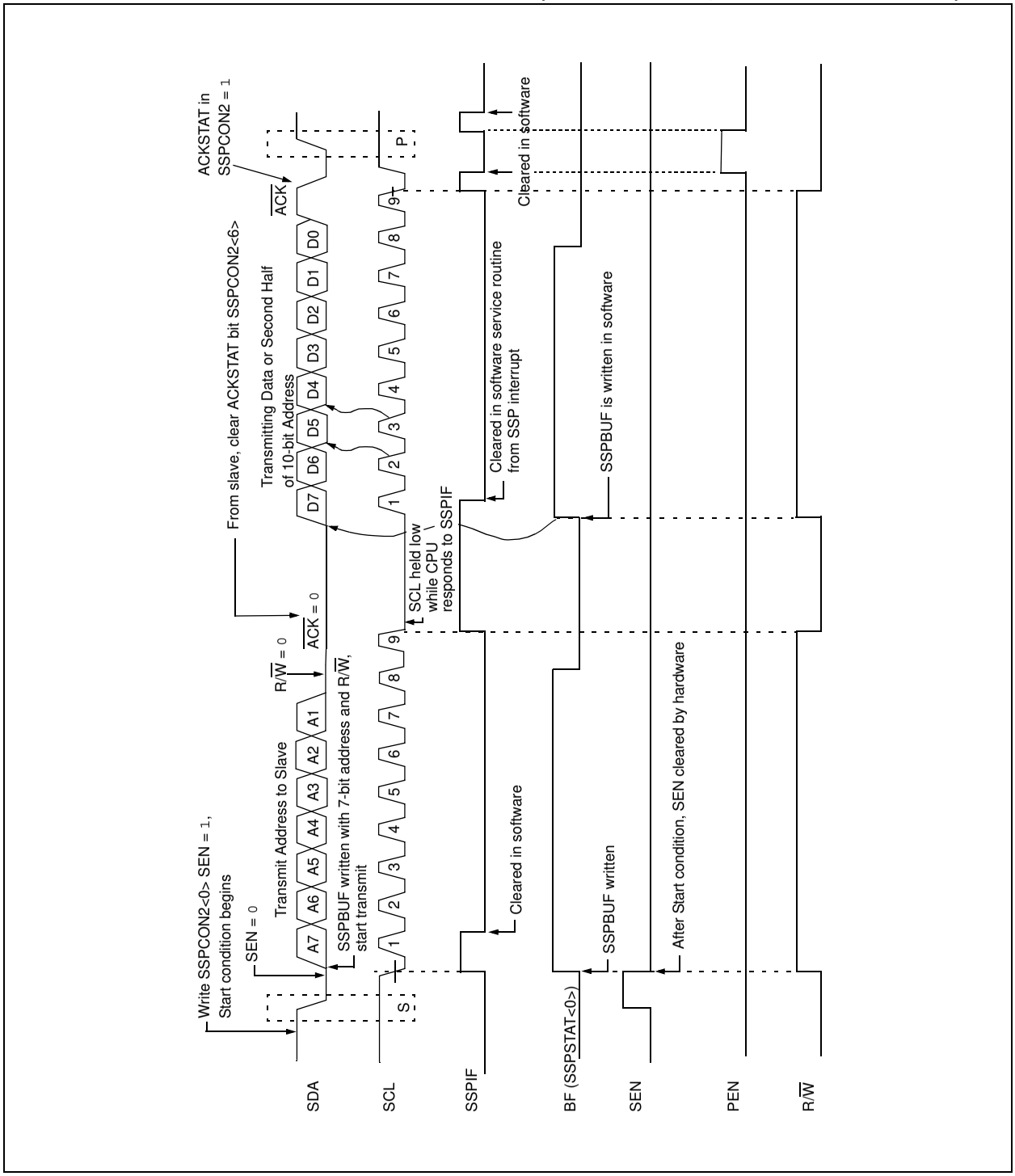
#### 17.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

#### 17.4.11.3 WCOL Status Flag

If the user writes the SSPBUF when a receive is already in progress (i.e., SSPSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

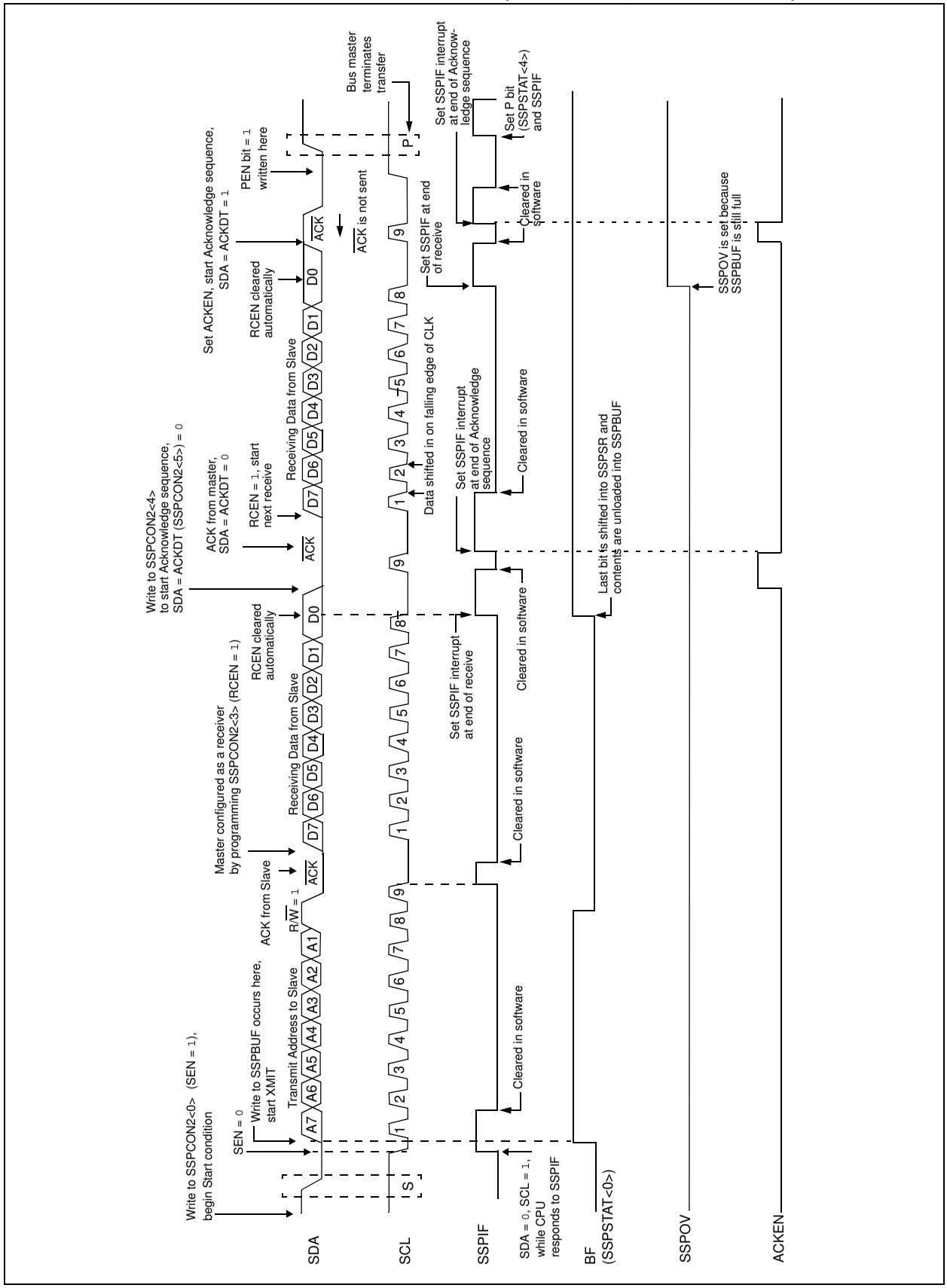
**FIGURE 17-21: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)**





© 2003 Microchip Technology Inc.

**FIGURE 17-22: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)**



### 17.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 17-23).

#### 17.4.12.1 WCOL Status Flag

If the user writes the SSPBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

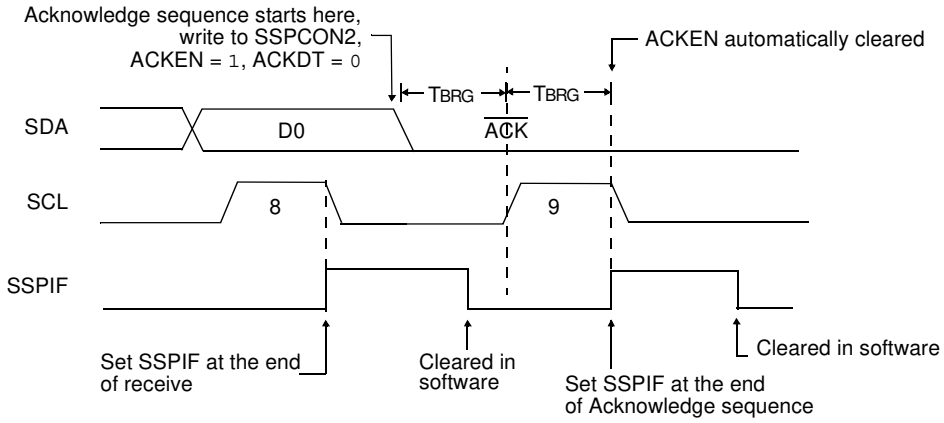
### 17.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to 0. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 17-24).

#### 17.4.13.1 WCOL Status Flag

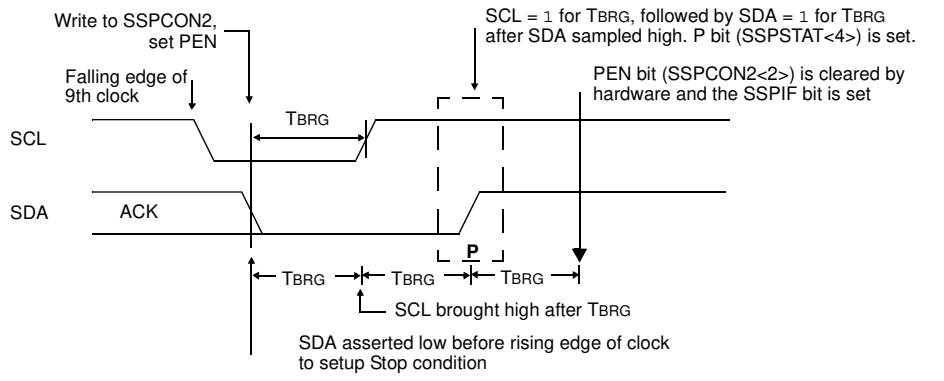
If the user writes the SSPBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

**FIGURE 17-23: ACKNOWLEDGE SEQUENCE WAVEFORM**




**Note:** TBRG = one Baud Rate Generator period.

**FIGURE 17-24: STOP CONDITION RECEIVE OR TRANSMIT MODE**




**Note:** TBRG = one Baud Rate Generator period.


© 2003 Microchip Technology Inc.

### 17.4.14 POWER MANAGED MODE OPERATION

While in any power managed mode, the I²C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

### 17.4.15 EFFECT OF A RESET

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

### 17.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I²C bus may be taken when the P bit (SSPSTAT<4>) is set or the bus is idle with both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

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

The states where arbitration can be lost are:

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

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

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF, and reset the I²C port to its Idle state (Figure 17-25).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I²C bus is free, the user can resume communication by asserting a Start condition.

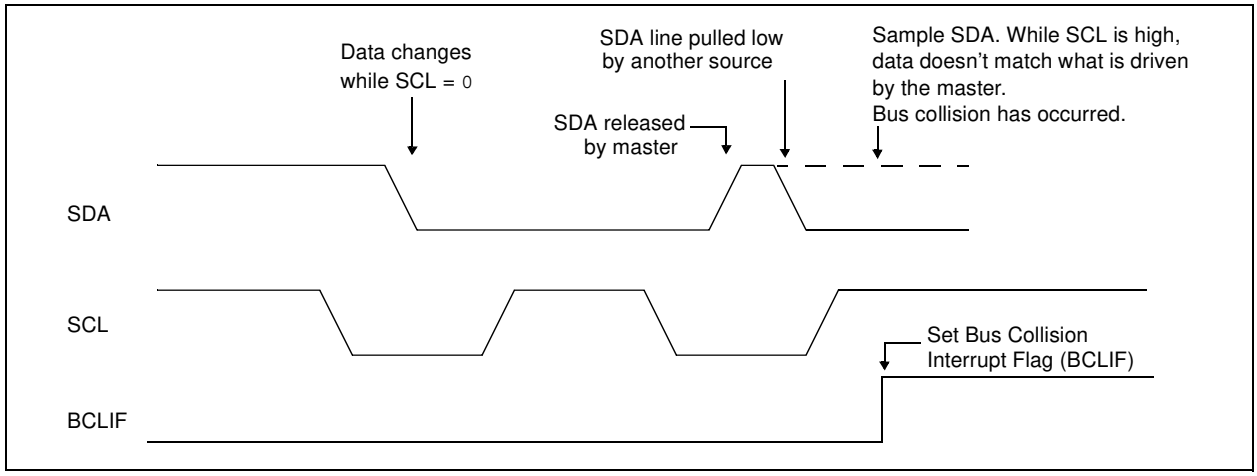
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted, and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I²C bus is free, the user can resume communication by asserting a Start condition.

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

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

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I²C bus can be taken when the P bit is set in the SSPSTAT register or the bus is Idle and the S and P bits are cleared.

**FIGURE 17-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**



#### 17.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDA or SCL is sampled low at the beginning of the Start condition (Figure 17-26).

b) SCL is sampled low before SDA is asserted low (Figure 17-27).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low or the SCL pin is already low, then all of the following occur:

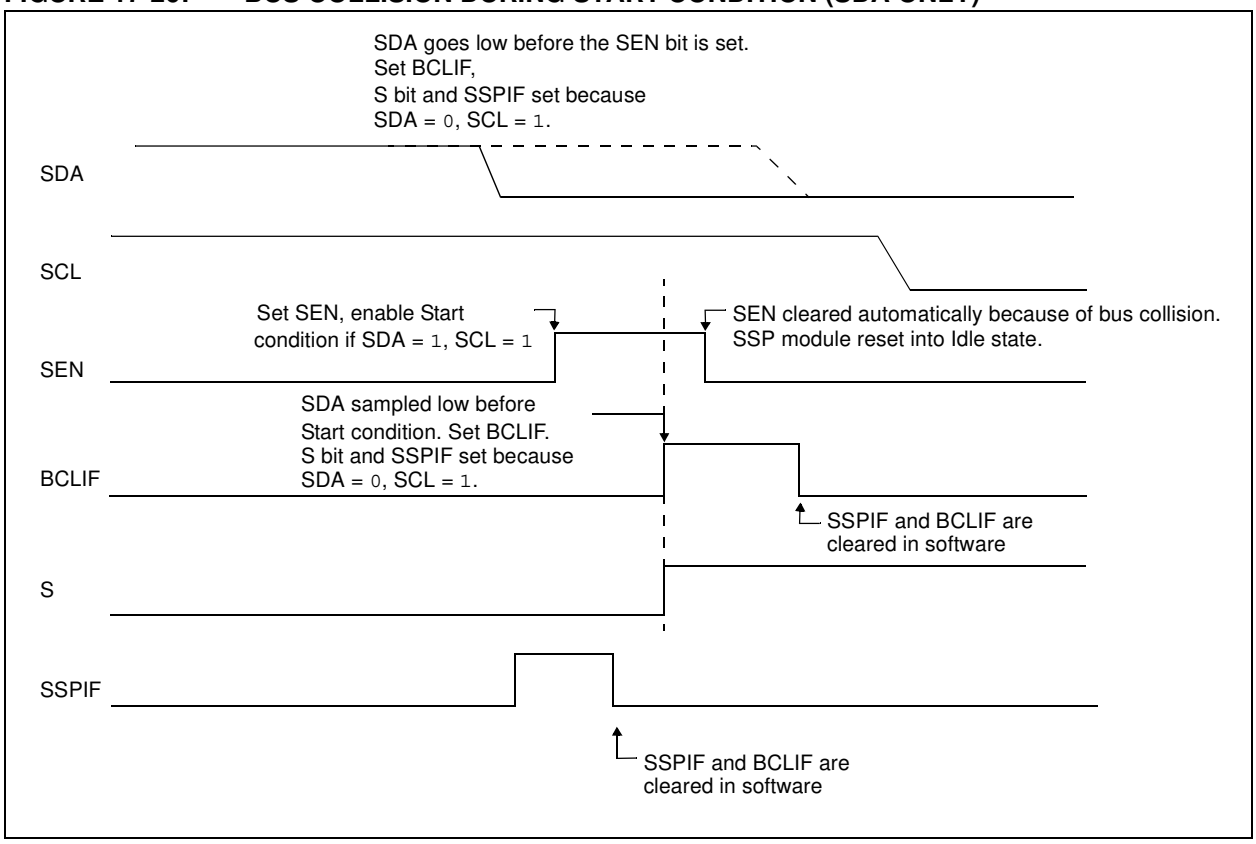
- The Start condition is aborted
- The BCLIF flag is set
- The MSSP module is reset to its Idle state (Figure 17-26)

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded from SSPADD<6:0> and counts down to 0. If the SCL pin is sampled low while SDA is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 17-28). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to 0 and during this time, if the SCL pins are sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

| **Note:** | The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions. |
|---|---|

**FIGURE 17-26: BUS COLLISION DURING START CONDITION (SDA ONLY)**





© 2003 Microchip Technology Inc.

**FIGURE 17-27: BUS COLLISION DURING START CONDITION (SCL = 0)**

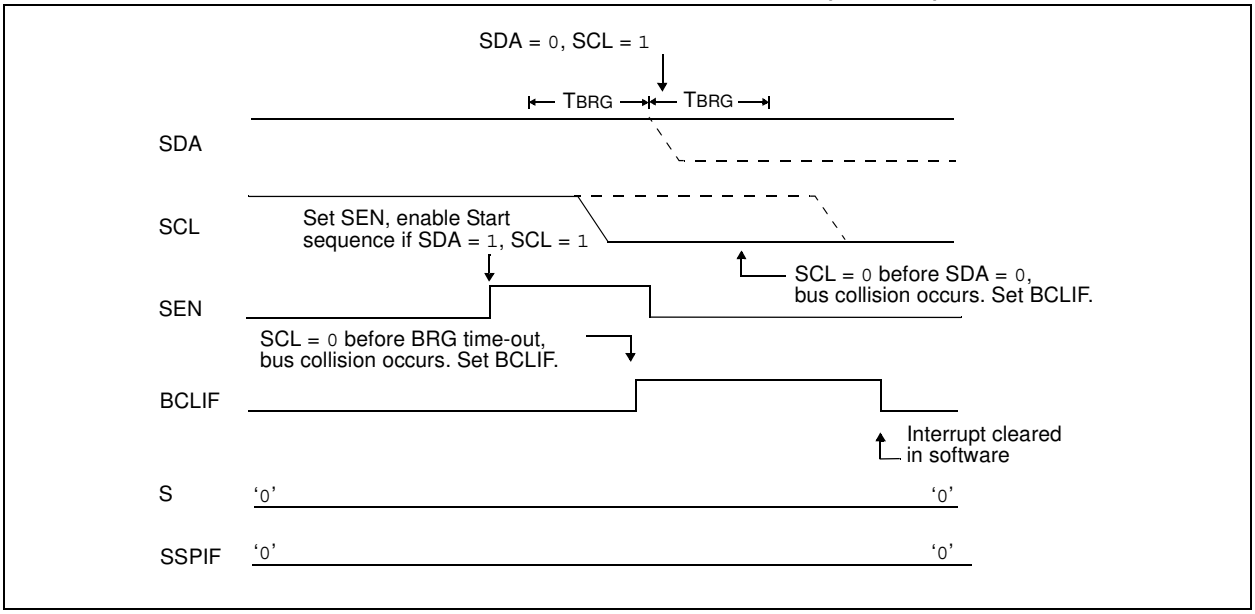
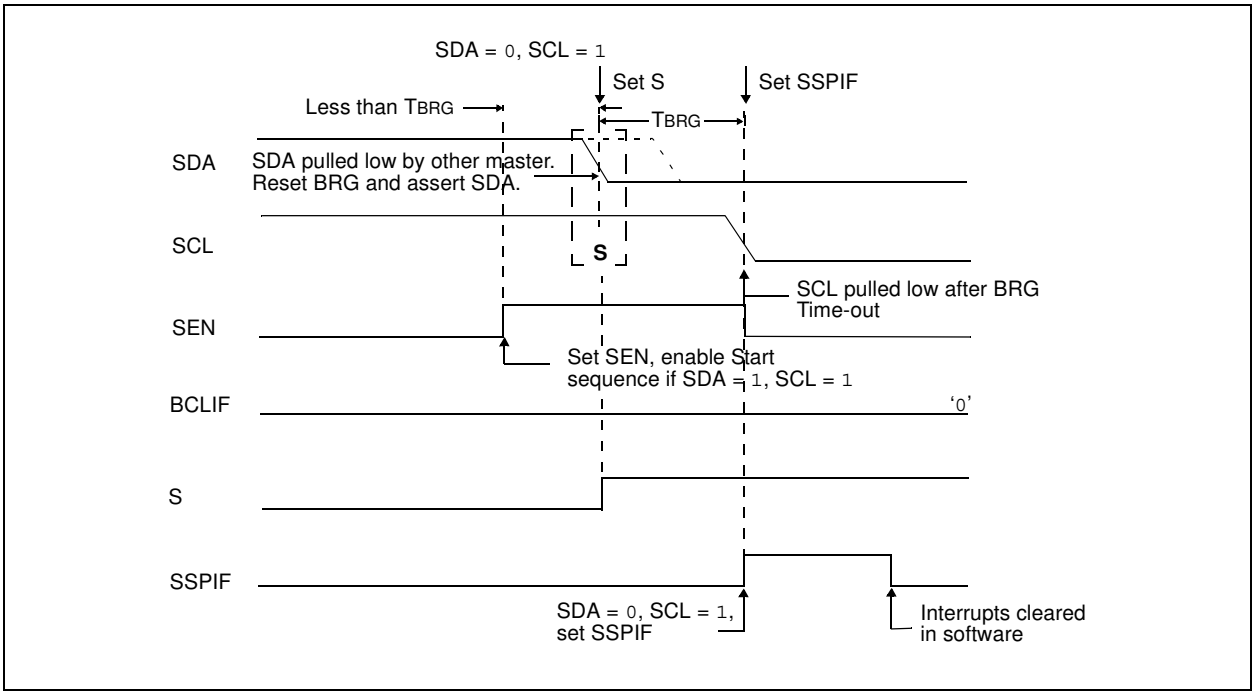



**FIGURE 17-28: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION**



#### 17.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level.

b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<6:0> and counts down to 0. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 17-29). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high to low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high to low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 17-30).

If at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

**FIGURE 17-29: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)**

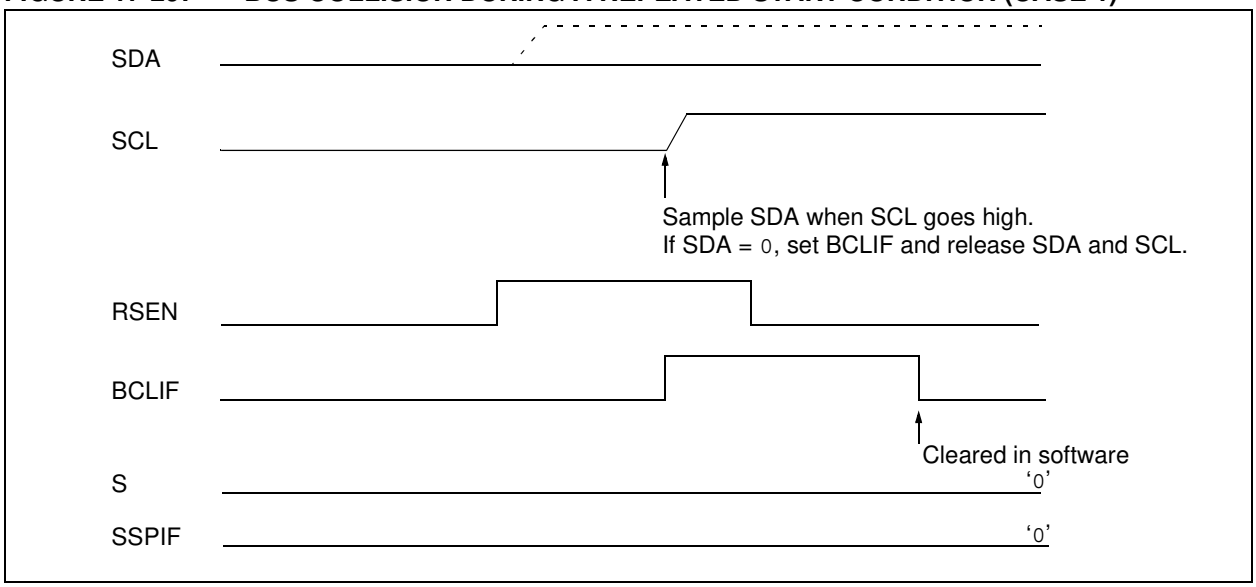
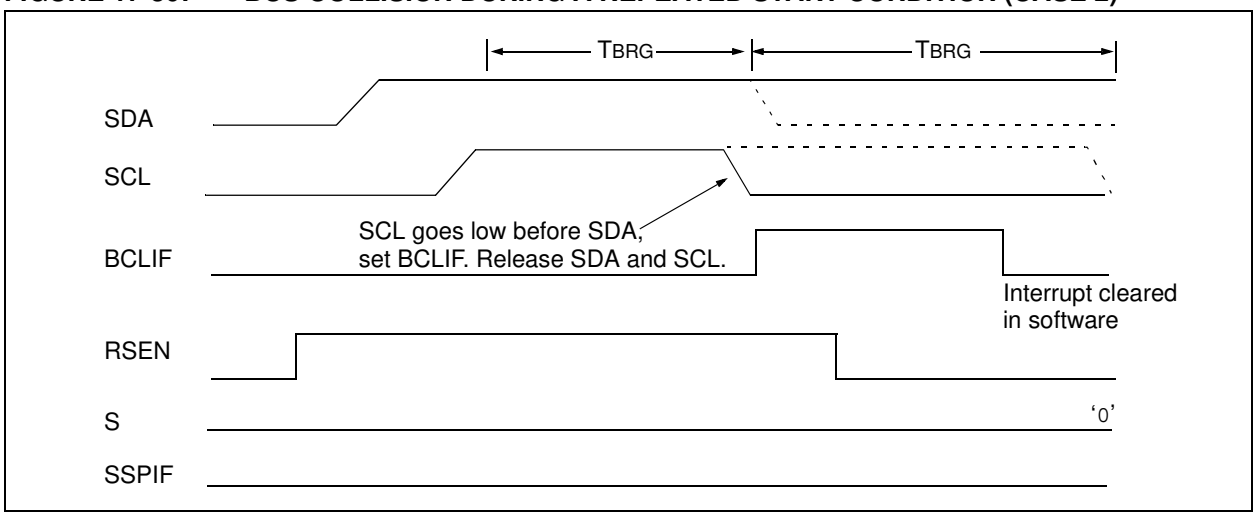



**FIGURE 17-30: BUS COLLISION DURING A REPEATED START CONDITION (CASE 2)**





© 2003 Microchip Technology Inc.

#### 17.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out.

b) After the SCL pin is deasserted, SCL is sampled low before SDA goes high.

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD<6:0> and counts down to 0. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 17-31). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 17-32).

**FIGURE 17-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)**

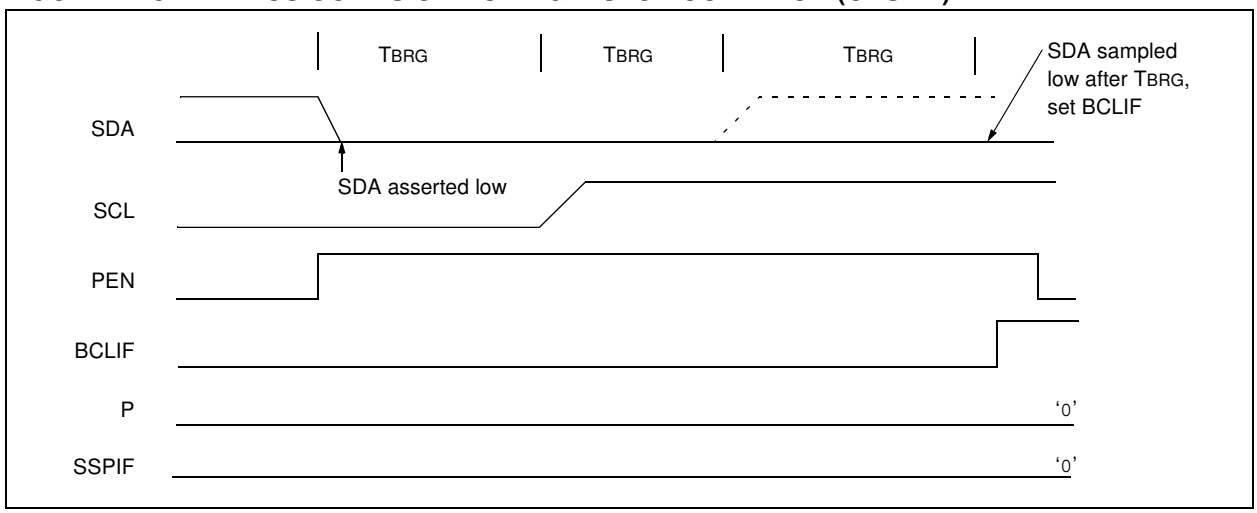
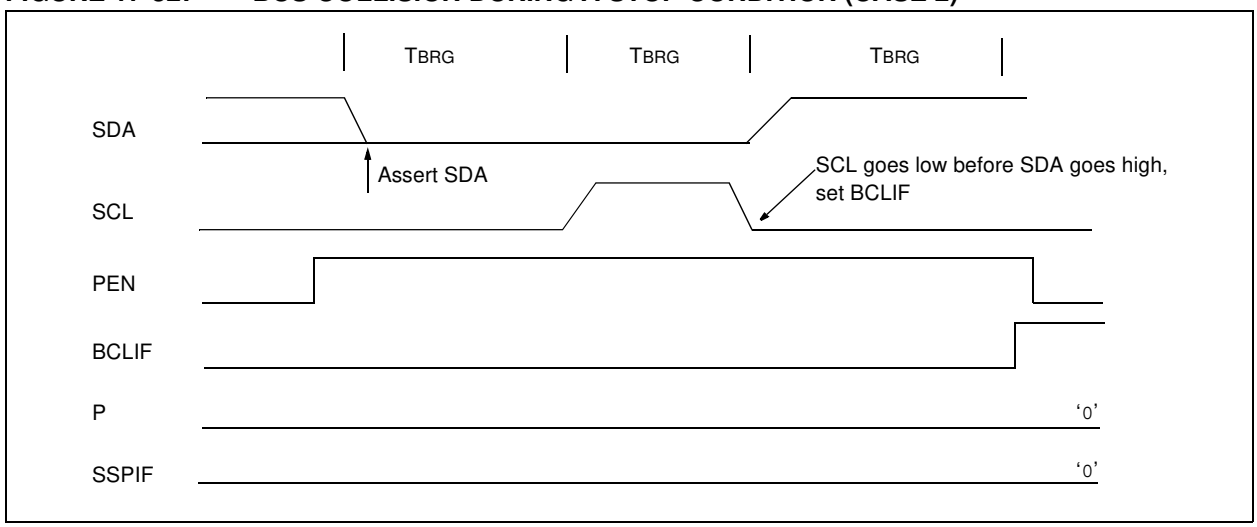



**FIGURE 17-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)**



**NOTES:**


© 2003 Microchip Technology Inc.

# 18.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules available in the PIC18F2X20/4X20 family of microcontrollers. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The USART can be configured in the following modes:

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

The RC6/TX/CK and RC7/RX/DT pins must be configured as shown for use with the Universal Synchronous Asynchronous Receiver Transmitter:

- SPEN (RCSTA<7>) bit must be set (= 1)
- TRISC<7> bit must be set (= 1)
- TRISC<6> bit must be cleared (= 0)

Register 18-1 shows the Transmit Status and Control register (TXSTA) and Register 18-2 shows the Receive Status and Control register (RCSTA).

## 18.1 Asynchronous Operation in Power Managed Modes

The USART may operate in Asynchronous mode while the peripheral clocks are being provided by the internal oscillator block. This mode makes it possible to remove the crystal or resonator that is commonly connected as the primary clock on the OSC1 and OSC2 pins.

The factory calibrates the internal oscillator block output (INTOSC) for 8 MHz. However, this frequency may drift as $V_{DD}$ or temperature changes and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the INTOSC output back to 8 MHz. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source (see **Section 3.6 "INTOSC Frequency Drift"** for more information).

The other method adjusts the value in the Baud Rate Generator since there may be not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.

**REGISTER 18-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER**

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

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

Asynchronous mode:
Don't care.

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

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

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

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

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

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

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

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode.

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

bit 0 **TX9D:** 9th bit of Transmit Data
Can be address/data bit or a parity bit.

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


© 2003 Microchip Technology Inc.

**REGISTER 18-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER**

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

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port enabled (configures RX/DT and TX/CK pins as serial port pins)
0 = Serial port disabled

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

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

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

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

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

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

bit 0 **RX9D:** 9th bit of Received Data
This can be address/data bit or a parity bit and must be calculated by user firmware.

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

## 18.2 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit Baud Rate Generator. The SPBRG register controls the period of a free-running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 18-1 shows the formula for computation of the baud rate for different USART modes which only apply in Master mode (internal clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRG register can be calculated using the formula in Table 18-1. From this, the error in baud rate can be determined.

Example 18-1 shows the calculation of the baud rate error for the following conditions:

- FOSC = 16 MHz
- Desired Baud Rate = 9600
- BRGH = 0
- SYNC = 0

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

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

### 18.2.1 POWER MANAGED MODE OPERATION

The system clock is used to generate the desired baud rate; however, when a power managed mode is entered, the clock source may be operating at a different frequency than in PRI_RUN mode. In Sleep mode, no clocks are present and in PRI_IDLE, the primary clock source continues to provide clocks to the baud rate generator; however, in other power managed modes, the clock frequency will probably change. This may require the value in SPBRG to be adjusted.

### 18.2.2 SAMPLING

The data on the RC7/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

**EXAMPLE 18-1: CALCULATING BAUD RATE ERROR**

Desired Baud Rate = FOSC/(64 (X + 1))

Solving for X:

X = ((FOSC/Desired Baud Rate)/64) – 1
X = ((16000000/9600)/64) – 1
X = [25.042] = 25

Calculated Baud Rate = 16000000/(64 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate)

Desired Baud Rate

= (9615 – 9600)/9600
= 0.16%

**TABLE 18-1: BAUD RATE FORMULA**

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

**Legend:** X = value in SPBRG (0 to 255)

**TABLE 18-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

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


© 2003 Microchip Technology Inc.

**TABLE 18-3: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0, LOW SPEED)**

| BAUD RATE (K) | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 16.000 MHz | | | Fosc = 10.000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | 0.98 | 225.52 | 255 | 0.61 | 103.45 | 255 |
| 1.2 | — | — | — | 1.22 | 1.73 | 255 | 1.20 | 0.16 | 207 | 1.20 | 0.16 | 129 |
| 2.4 | 2.44 | 1.73 | 255 | 2.40 | 0.16 | 129 | 2.40 | 0.16 | 103 | 2.40 | 0.16 | 64 |
| 9.6 | 9.62 | 0.16 | 64 | 9.47 | -1.36 | 32 | 9.62 | 0.16 | 25 | 9.77 | 1.73 | 15 |
| 19.2 | 18.94 | -1.36 | 32 | 19.53 | 1.73 | 15 | 19.23 | 0.16 | 12 | 19.53 | 1.73 | 7 |
| 38.4 | 39.06 | 1.73 | 15 | 39.06 | 1.73 | 7 | 35.71 | -6.99 | 6 | 39.06 | 1.73 | 3 |
| 57.6 | 56.82 | -1.36 | 10 | 62.50 | 8.51 | 4 | 62.50 | 8.51 | 3 | 52.08 | -9.58 | 2 |
| 76.8 | 78.13 | 1.73 | 7 | 78.13 | 1.73 | 3 | 83.33 | 8.51 | 2 | 78.13 | 1.73 | 1 |
| 96.0 | 89.29 | -6.99 | 6 | 104.17 | 8.51 | 2 | — | — | — | — | — | — |
| 115.2 | 125.00 | 8.51 | 4 | — | — | — | 125.00 | 8.51 | 1 | 78.13 | -32.18 | 1 |
| 250.0 | 208.33 | -16.67 | 2 | — | — | | 250.00 | 0.00 | 0 | — | — | — |
| 300.0 | 312.50 | 4.17 | 1 | 312.50 | 4.17 | 0 | — | — | — | — | — | — |
| 625.0 | 625.00 | 0.00 | 0 | — | — | — | — | — | — | — | — | — |

| BAUD RATE (K) | Fosc = 8.000000 MHz | | | Fosc = 7.159090 MHz | | | Fosc = 5.068800 MHz | | | Fosc = 4.000000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.49 | 62.76 | 255 | 0.44 | 45.65 | 255 | 0.31 | 3.13 | 255 | 0.30 | 0.16 | 207 |
| 1.2 | 1.20 | 0.16 | 103 | 1.20 | 0.23 | 92 | 1.20 | 0.00 | 65 | 1.20 | 0.16 | 51 |
| 2.4 | 2.40 | 0.16 | 51 | 2.38 | -0.83 | 46 | 2.40 | 0.00 | 32 | 2.40 | 0.16 | 25 |
| 9.6 | 9.62 | 0.16 | 12 | 9.32 | -2.90 | 11 | 9.90 | 3.13 | 7 | 8.93 | -6.99 | 6 |
| 19.2 | 17.86 | -6.99 | 6 | 18.64 | -2.90 | 5 | 19.80 | 3.13 | 3 | 20.83 | 8.51 | 2 |
| 38.4 | 41.67 | 8.51 | 2 | 37.29 | -2.90 | 2 | 39.60 | 3.13 | 1 | 31.25 | -18.62 | 1 |
| 57.6 | 62.50 | 8.51 | 1 | 55.93 | -2.90 | 1 | — | — | — | 62.50 | 8.51 | 0 |
| — | — | — | — | — | — | — | 79.20 | 3.13 | 0 | — | — | — |
| 115.2 | 125.00 | 8.51 | 0 | 111.86 | -2.90 | 0 | — | — | — | — | — | — |

| BAUD RATE (K) | Fosc = 3.579545 MHz | | | Fosc = 2.000000 MHz | | | Fosc = 1.000000 MHz | | | Fosc = 0.032768 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.30 | 0.23 | 185 | 0.30 | 0.16 | 103 | 0.30 | 0.16 | 51 | 0.26 | -14.67 | 1 |
| 1.2 | 1.19 | -0.83 | 46 | 1.20 | 0.16 | 25 | 1.20 | 0.16 | 12 | — | — | — |
| 2.4 | 2.43 | 1.32 | 22 | 2.40 | 0.16 | 12 | 2.23 | -6.99 | 6 | — | — | — |
| 9.6 | 9.32 | -2.90 | 5 | 10.42 | 8.51 | 2 | 7.81 | -18.62 | 1 | — | — | — |
| 19.2 | 18.64 | -2.90 | 2 | 15.63 | -18.62 | 1 | 15.63 | -18.62 | 0 | — | — | — |
| 38.4 | — | — | — | 31.25 | -18.62 | 0 | — | — | — | — | — | — |
| 57.6 | 55.93 | -2.90 | 0 | — | — | — | — | — | — | — | — | — |

© 2003 Microchip Technology Inc.

**TABLE 18-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1, HIGH SPEED)**

| BAUD RATE (K) | FOSC = 40.000 MHz | | | FOSC = 20.000 MHz | | | FOSC = 16.000 MHz | | | FOSC = 10.000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 2.4 | — | — | — | 4.88 | 103.45 | 255 | 3.91 | 62.76 | 255 | 2.44 | 1.73 | 255 |
| 9.6 | 9.77 | 1.73 | 255 | 9.62 | 0.16 | 129 | 9.62 | 0.16 | 103 | 9.63 | 0.16 | 64 |
| 19.2 | 19.23 | 0.16 | 129 | 19.23 | 0.16 | 64 | 19.23 | 0.16 | 51 | 18.94 | -1.36 | 32 |
| 38.4 | 38.46 | 0.16 | 64 | 37.88 | -1.36 | 32 | 38.46 | 0.16 | 25 | 39.06 | 1.73 | 15 |
| 57.6 | 58.14 | 0.94 | 42 | 56.82 | -1.36 | 21 | 58.82 | 2.12 | 16 | 56.82 | -1.36 | 10 |
| 76.8 | 75.76 | -1.36 | 32 | 78.13 | 1.73 | 15 | 76.92 | 0.16 | 12 | 78.13 | 1.73 | 7 |
| 96.0 | 96.15 | 0.16 | 25 | 96.15 | 0.16 | 12 | 100.00 | 4.17 | 9 | 89.29 | -6.99 | 6 |
| 115.2 | 113.64 | -1.36 | 21 | 113.64 | -1.36 | 10 | 111.11 | -3.55 | 8 | 125.00 | 8.51 | 4 |
| 250.0 | 250.00 | 0.00 | 9 | 250.00 | 0.00 | 4 | 250.00 | 0.00 | 3 | 208.33 | -16.67 | 2 |
| 300.0 | 312.50 | 4.17 | 7 | 312.50 | 4.17 | 3 | 333.33 | 11.11 | 2 | 312.50 | 4.17 | 1 |
| 500.0 | 500.00 | 0.00 | 4 | 416.67 | -16.67 | 2 | 500.00 | 0.00 | 1 | — | — | — |
| 625.0 | 625.00 | 0.00 | 3 | 625.00 | 0.00 | 1 | — | — | — | 625.00 | 0.00 | 0 |
| 1000.0 | 833.33 | -16.67 | 2 | — | — | — | 1000.00 | 0.00 | 0 | — | — | — |
| 1250.0 | 1250.00 | 0.00 | 1 | 1250.00 | 0.00 | 0 | — | — | — | — | — | — |

| BAUD RATE (K) | FOSC = 8.000000 MHz | | | FOSC = 7.159090 MHz | | | FOSC = 5.068800 MHz | | | FOSC = 4.000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | — | — | — | 0.98 | 225.52 | 255 |
| 1.2 | 1.95 | 62.76 | 255 | 1.75 | 45.65 | 255 | 1.24 | 3.13 | 255 | 1.20 | 0.16 | 207 |
| 2.4 | 2.40 | 0.16 | 207 | 2.41 | 0.23 | 185 | 2.40 | 0.00 | 131 | 2.40 | 0.16 | 103 |
| 9.6 | 9.62 | 0.16 | 51 | 9.52 | -0.83 | 46 | 9.60 | 0.00 | 32 | 9.62 | 0.16 | 25 |
| 19.2 | 19.23 | 0.16 | 25 | 19.45 | 1.32 | 22 | 18.64 | -2.94 | 16 | 19.23 | 0.16 | 12 |
| 38.4 | 38.46 | 0.16 | 12 | 37.29 | -2.90 | 11 | 39.60 | 3.13 | 7 | 35.71 | -6.99 | 6 |
| 57.6 | 55.56 | -3.55 | 8 | 55.93 | -2.90 | 7 | 52.80 | -8.33 | 5 | 62.50 | 8.51 | 3 |
| 76.8 | 71.43 | -6.99 | 6 | 74.57 | -2.90 | 5 | 79.20 | 3.13 | 3 | 83.33 | 8.51 | 2 |
| 96.0 | 100.00 | 4.17 | 4 | 89.49 | -6.78 | 4 | — | — | — | — | — | — |
| 115.2 | 125.00 | 8.51 | 3 | 111.86 | -2.90 | 3 | 105.60 | -8.33 | 2 | 125.00 | 8.51 | 1 |
| 250.0 | 250.00 | 0.00 | 1 | 223.72 | -10.51 | 1 | — | — | — | 250.00 | 0.00 | 0 |
| 300.0 | — | — | — | — | — | — | 316.80 | 5.60 | 0 | — | — | — |
| 500.0 | 500.00 | 0.00 | 0 | 447.44 | -10.51 | 0 | — | — | — | — | — | — |

| BAUD RATE (K) | FOSC = 3.579545 MHz | | | FOSC = 2.000000 MHz | | | FOSC = 1.000000 MHz | | | FOSC = 0.032768 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.87 | 191.30 | 255 | 0.49 | 62.76 | 255 | 0.30 | 0.16 | 207 | 0.29 | -2.48 | 6 |
| 1.2 | 1.20 | 0.23 | 185 | 1.20 | 0.16 | 103 | 1.20 | 0.16 | 51 | 1.02 | -14.67 | 1 |
| 2.4 | 2.41 | 0.23 | 92 | 2.40 | 0.16 | 51 | 2.40 | 0.16 | 25 | 2.05 | -14.67 | 0 |
| 9.6 | 9.73 | 1.32 | 22 | 9.62 | 0.16 | 12 | 8.93 | -6.99 | 6 | — | — | — |
| 19.2 | 18.64 | -2.90 | 11 | 17.86 | -6.99 | 6 | 20.83 | 8.51 | 2 | — | — | — |
| 38.4 | 37.29 | -2.90 | 5 | 41.67 | 8.51 | 2 | 31.25 | -18.62 | 1 | — | — | — |
| 57.6 | 55.93 | -2.90 | 3 | 62.50 | 8.51 | 1 | 62.50 | 8.51 | 0 | — | — | — |
| 76.8 | 74.57 | -2.90 | 2 | — | — | — | — | — | — | — | — | — |
| 115.2 | 111.86 | -2.90 | 1 | 125.00 | 8.51 | 0 | — | — | — | — | — | — |
| 250.0 | 223.72 | -10.51 | 0 | — | — | — | — | — | — | — | — | — |


© 2003 Microchip Technology Inc.

**TABLE 18-5: BAUD RATES FOR SYNCHRONOUS MODE (SYNC = 1)**

| BAUD RATE (K) | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 16.000 MHz | | | Fosc = 10.000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 9.6 | — | — | — | — | — | — | 15.63 | 62.76 | 255 | 9.77 | 1.73 | 255 |
| 19.2 | — | — | — | 19.53 | 1.73 | 255 | 19.23 | 0.16 | 207 | 19.23 | 0.16 | 129 |
| 38.4 | 39.06 | 1.73 | 255 | 38.46 | 0.16 | 129 | 38.46 | 0.16 | 103 | 38.46 | 0.16 | 64 |
| 57.6 | 57.47 | -0.22 | 173 | 57.47 | -0.22 | 86 | 57.97 | 0.64 | 68 | 58.14 | 0.94 | 42 |
| 76.8 | 76.92 | 0.16 | 129 | 76.92 | 0.16 | 64 | 76.92 | 0.16 | 51 | 75.76 | -1.36 | 32 |
| 96.0 | 96.15 | 0.16 | 103 | 96.15 | 0.16 | 51 | 95.24 | -0.79 | 41 | 96.15 | 0.16 | 25 |
| 250.0 | 250.00 | 0.00 | 39 | 250.00 | 0.00 | 19 | 250.00 | 0.00 | 15 | 250.00 | 0.00 | 9 |
| 300.0 | 303.03 | 1.01 | 32 | 294.12 | -1.96 | 16 | 307.69 | 2.56 | 12 | 312.50 | 4.17 | 7 |
| 500.0 | 500.00 | 0.00 | 19 | 500.00 | 0.00 | 9 | 500.00 | 0.00 | 7 | 500.00 | 0.00 | 4 |
| 625.0 | 625.00 | 0.00 | 15 | 625.00 | 0.00 | 7 | 666.67 | 6.67 | 5 | 625.00 | 0.00 | 3 |
| 1000.0 | 1000.00 | 0.00 | 9 | 1000.00 | 0.00 | 4 | 1000.00 | 0.00 | 3 | 833.33 | -16.67 | 2 |
| 1250.0 | 1250.00 | 0.00 | 7 | 1250.00 | 0.00 | 3 | 1333.33 | 6.67 | 2 | 1250.00 | 0.00 | 1 |

| BAUD RATE (K) | Fosc = 8.000000 MHz | | | Fosc = 7.159090 MHz | | | Fosc = 5.068800 MHz | | | Fosc = 4.000 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 2.4 | 7.81 | 225.52 | 255 | 6.99 | 191.30 | 255 | 4.95 | 106.25 | 255 | 3.91 | 62.76 | 255 |
| 9.6 | 9.62 | 0.16 | 207 | 9.62 | 0.23 | 185 | 9.60 | 0.00 | 131 | 9.62 | 0.16 | 103 |
| 19.2 | 19.23 | 0.16 | 103 | 19.24 | 0.23 | 92 | 19.20 | 0.00 | 65 | 19.23 | 0.16 | 51 |
| 38.4 | 38.46 | 0.16 | 51 | 38.08 | -0.83 | 46 | 38.40 | 0.00 | 32 | 38.46 | 0.16 | 25 |
| 57.6 | 57.14 | -0.79 | 34 | 57.73 | 0.23 | 30 | 57.60 | 0.00 | 21 | 58.82 | 2.12 | 16 |
| 76.8 | 76.92 | 0.16 | 25 | 77.82 | 1.32 | 22 | 74.54 | -2.94 | 16 | 76.92 | 0.16 | 12 |
| 96.0 | 95.24 | -0.79 | 20 | 94.20 | -1.88 | 18 | 97.48 | 1.54 | 12 | 100.00 | 4.17 | 9 |
| 250.0 | 250.00 | 0.00 | 7 | 255.68 | 2.27 | 6 | 253.44 | 1.38 | 4 | 250.00 | 0.00 | 3 |
| 300.0 | 285.71 | -4.76 | 6 | 298.30 | -0.57 | 5 | 316.80 | 5.60 | 3 | 333.33 | 11.11 | 2 |
| 500.0 | 500.00 | 0.00 | 3 | 447.44 | -10.51 | 3 | 422.40 | -15.52 | 2 | 500.00 | 0.00 | 1 |
| 625.0 | 666.67 | 6.67 | 2 | 596.59 | -4.55 | 2 | 633.60 | 1.38 | 1 | — | — | — |
| 1000.0 | 1000.00 | 0.00 | 1 | 894.89 | -10.51 | 1 | — | — | — | 1000.00 | 0.00 | 0 |
| 1250.0 | — | — | — | 1789.77 | 43.18 | 0 | 1267.20 | 1.38 | 0 | — | — | — |

| BAUD RATE (K) | Fosc = 3.579545 MHz | | | Fosc = 2.000000 MHz | | | Fosc = 1.000000 MHz | | | Fosc = 0.032768 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | 0.98 | 225.52 | 255 | 0.30 | 1.14 | 26 |
| 1.2 | — | — | — | 1.95 | 62.76 | 255 | 1.20 | 0.16 | 207 | 1.17 | -2.48 | 6 |
| 2.4 | 3.50 | 45.65 | 255 | 2.40 | 0.16 | 207 | 2.40 | 0.16 | 103 | 2.73 | 13.78 | 2 |
| 9.6 | 9.62 | 0.23 | 92 | 9.62 | 0.16 | 51 | 9.62 | 0.16 | 25 | 8.19 | -14.67 | 0 |
| 19.2 | 19.04 | -0.83 | 46 | 19.23 | 0.16 | 25 | 19,.23 | 0.16 | 12 | — | — | — |
| 38.4 | 38.91 | 1.32 | 22 | 38.46 | 0.16 | 12 | 35.71 | -6.99 | 6 | — | — | — |
| 57.6 | 55.93 | -2.90 | 15 | 55.56 | -3.55 | 8 | 62.50 | 8.51 | 3 | — | — | — |
| 76.8 | 74.57 | -2.90 | 11 | 71.43 | -6.99 | 6 | 83.33 | 8.51 | 2 | — | — | — |
| 96.0 | 99.43 | 3.57 | 8 | 100.00 | 4.17 | 4 | — | — | — | — | — | — |
| 250.0 | 223.72 | -10.51 | 3 | 250.00 | 0.00 | 1 | 250.00 | 0.00 | 0 | — | — | — |
| 500.0 | 447.44 | -10.51 | 1 | 500.00 | 0.00 | 0 | — | — | — | — | — | — |

© 2003 Microchip Technology Inc.

## 18.3 USART Asynchronous Mode

In this mode, the USART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip dedicated 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The USART's transmitter and receiver are functionally independent but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware but can be implemented in software (and stored as the ninth data bit). Asynchronous mode functions in all power managed modes except Sleep mode when call clock sources are stopped. When in PRI_IDLE mode, no changes to the Baud Rate Generator values are required; however, other power managed mode clocks may operate at another frequency than the primary clock. Therefore, the Baud Rate generator values may need adjusting.

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

The USART Asynchronous module consists of the following important elements:

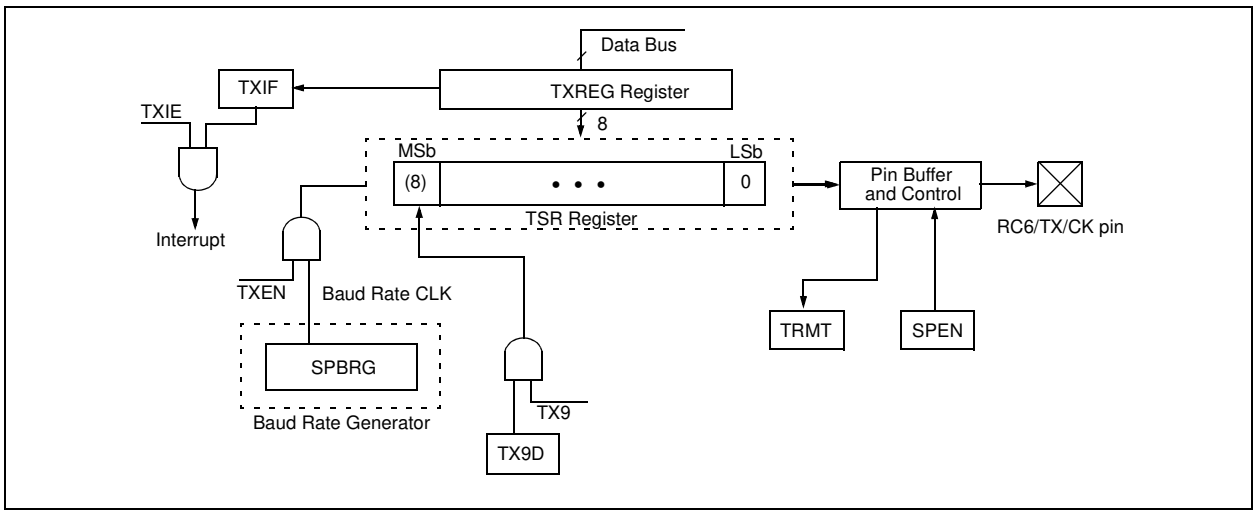
- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

### 18.3.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 18-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit, TXIF (PIR1<4>), is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. Flag bit TXIF is not cleared immediately upon loading the Transmit Buffer register, TXREG. TXIF becomes valid in the second instruction cycle following the load instruction. Polling TXIF immediately following a load of TXREG will return invalid results. While flag bit TXIF indicated the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit, therefore, the user must poll this bit in order to determine whether the TSR register is empty.

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

**FIGURE 18-1: USART TRANSMIT BLOCK DIAGRAM**





© 2003 Microchip Technology Inc.

**FIGURE 18-2: ASYNCHRONOUS TRANSMISSION**

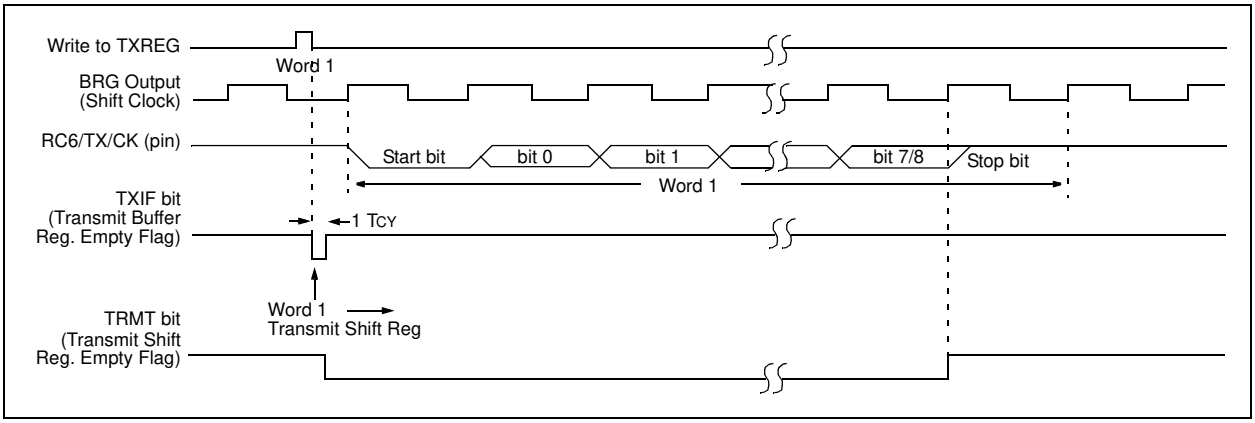
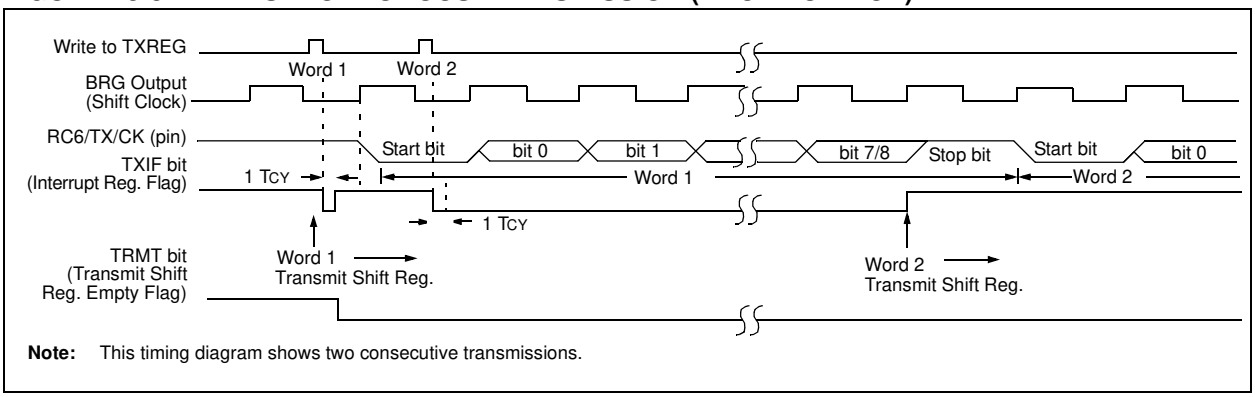



**FIGURE 18-3: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)**




**Note:** This timing diagram shows two consecutive transmissions.

**TABLE 18-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

### 18.3.2 USART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 18-4. The data is received on the RC7/RX/DT pin and drives the data recovery block. The data recovery block is actually a high-speed shifter, operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

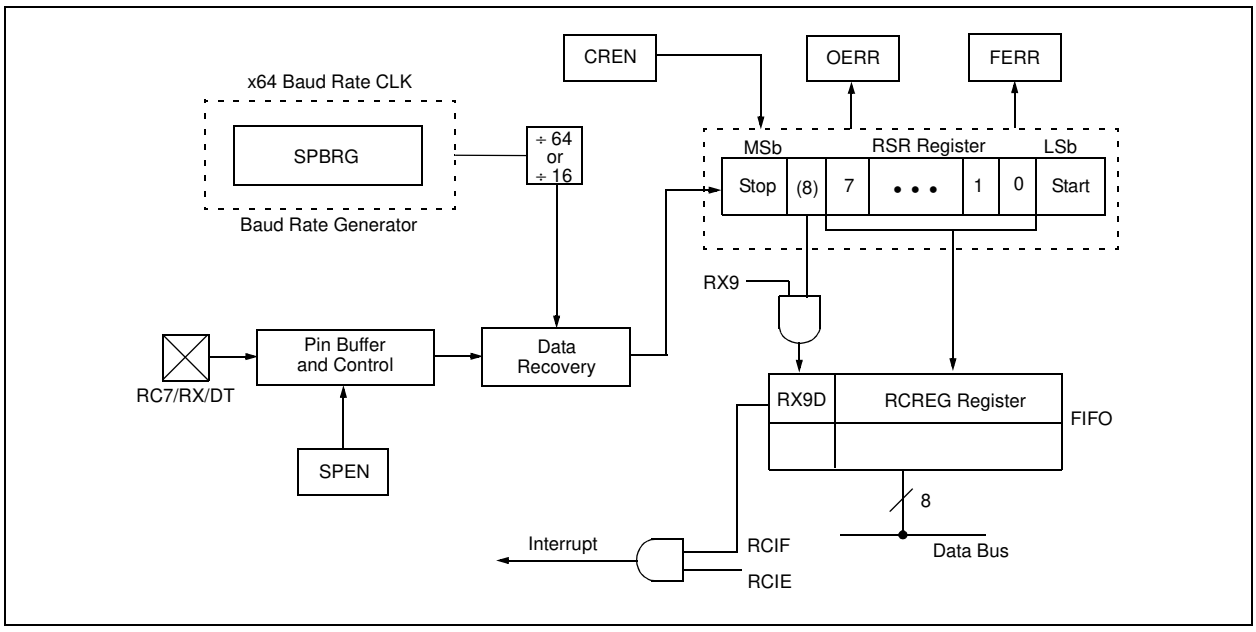
1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 18.2 "USART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit RCIE.
4. If 9-bit reception is desired, set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

### 18.3.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with address detect enable:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is required, set the BRGH bit.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCIP bit.
4. Set the RX9 bit to enable 9-bit reception.
5. Set the ADDEN bit to enable address detect.
6. Enable reception by setting the CREN bit.
7. The RCIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCIE and GIE bits are set.
8. Read the RCSTA register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREG to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

**FIGURE 18-4: USART RECEIVE BLOCK DIAGRAM**



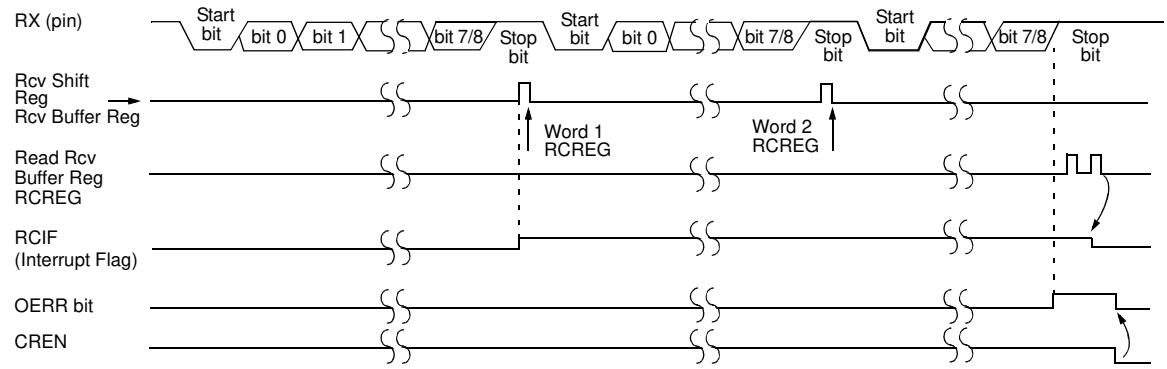


© 2003 Microchip Technology Inc.

To set up an Asynchronous Transmission:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 18.2 "USART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set Transmit bit, TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit TXEN which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 18-5: ASYNCHRONOUS RECEPTION**




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

**TABLE 18-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

## 18.4 USART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit, SYNC (TXSTA<4>). In addition, enable bit, SPEN (RCSTA<7>), is set in order to configure the RC6/TX/CK and RC7/RX/DT I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit, CSRC (TXSTA<7>).

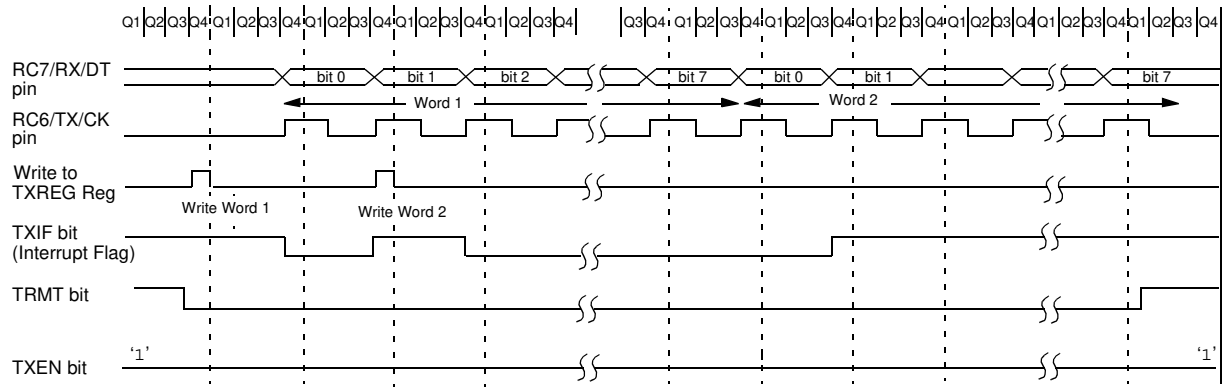
### 18.4.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 18-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and interrupt bit, TXIF (PIR1<4>), is set. The interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (**Section 18.2 "USART Baud Rate Generator (BRG)"**).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 18-6: SYNCHRONOUS TRANSMISSION**




**Note:** Sync Master mode, SPBRG = 0; continuous transmission of two 8-bit words.


© 2003 Microchip Technology Inc.

**FIGURE 18-7: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**

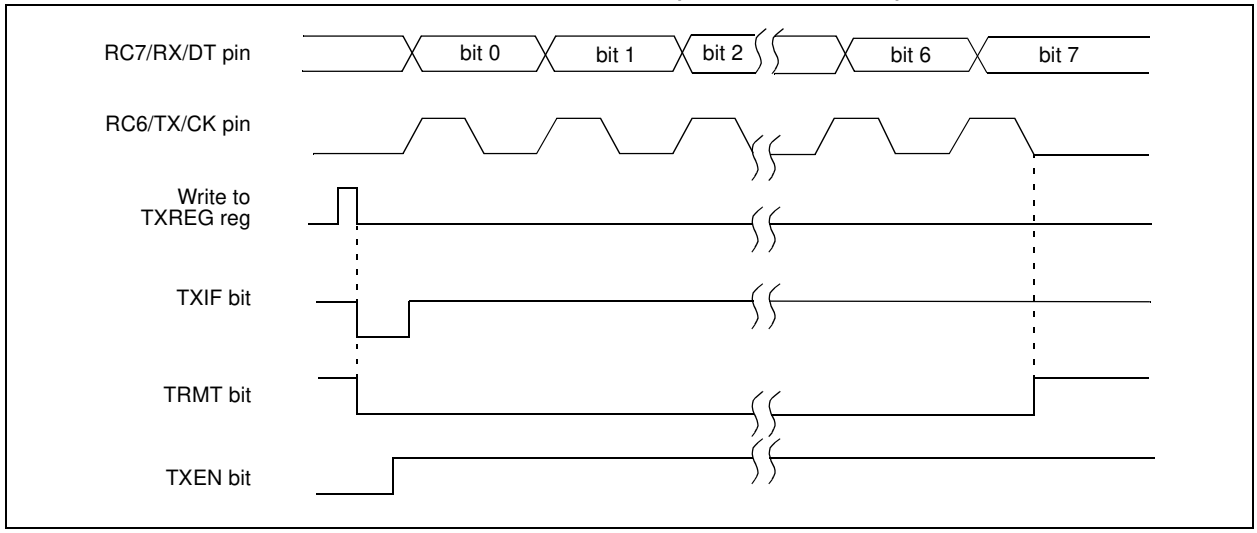



**TABLE 18-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

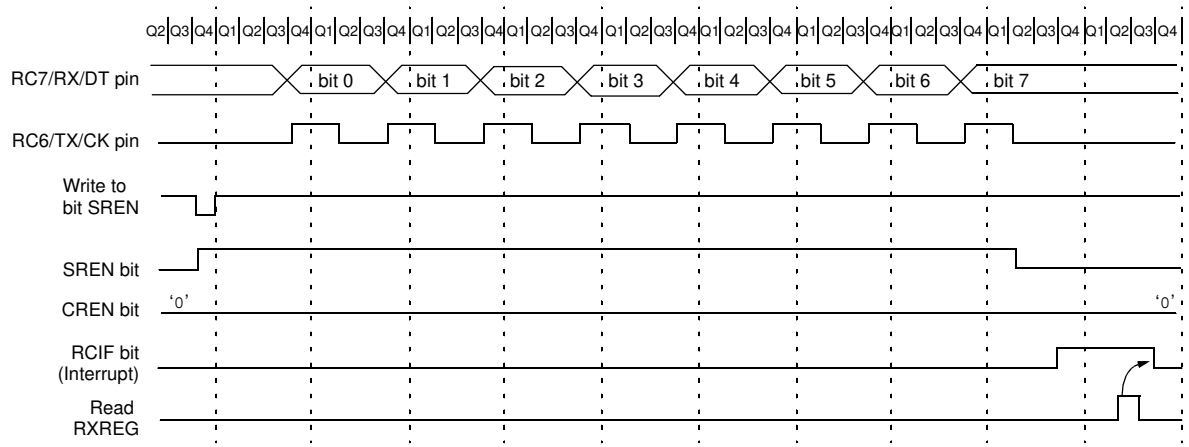
### 18.4.2 USART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either enable bit, SREN (RCSTA<5>), or enable bit, CREN (RCSTA<4>). Data is sampled on the RC7/RX/DT pin on the falling edge of the clock. If enable bit SREN is set, only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set, then CREN takes precedence.

To set up a Synchronous Master Reception:

1. Initialize the SPBRG register for the appropriate baud rate (**Section 18.2 "USART Baud Rate Generator (BRG)"**).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, set enable bit RCIE.
5. If 9-bit reception is desired, set bit RX9.
6. If a single reception is required, set bit SREN. For continuous reception, set bit CREN.
7. Interrupt flag bit RCIF will be set when reception is complete and an interrupt will be generated if the enable bit RCIE was set.
8. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 18-8: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**




**Note:** Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

**TABLE 18-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.


© 2003 Microchip Technology Inc.

## 18.5 USART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the RC6/TX/CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any power managed mode. Slave mode is entered by clearing bit, CSRC (TXSTA<7>).

### 18.5.1 USART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREG register.
c) Flag bit TXIF will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will now be set.
e) If enable bit TXIE is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**TABLE 18-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| TXREG | USART Transmit Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.

### 18.5.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep or any Idle mode and bit SREN, which is a “don't care” in Slave mode.

If receive is enabled by setting bit CREN prior to entering Sleep or any Idle mode, then a word may be received while in this power managed mode. Once the word is received, the RSR register will transfer the data to the RCREG register and if enable bit RCIE bit is set, the interrupt generated will wake the chip from the power managed mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete. An interrupt will be generated if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**TABLE 18-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 0000 000u |
| PIR1 | PSPIF(1) | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE(1) | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP(1) | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 -00x | 0000 -00x |
| RCREG | USART Receive Register | | | | | | | | 0000 0000 | 0000 0000 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | — | BRGH | TRMT | TX9D | 0000 -010 | 0000 -010 |
| SPBRG | Baud Rate Generator Register | | | | | | | | 0000 0000 | 0000 0000 |

**Legend:** x = unknown, - = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

**Note 1:** The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18F2X20 devices; always maintain these bits clear.


© 2003 Microchip Technology Inc.

## 19.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has 10 inputs for the PIC18F2X20 devices and 13 for the PIC18F4X20 devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

A new feature for the A/D converter is the addition of programmable acquisition time. This feature allows the user to select a new channel for conversion and setting the GO/DONE bit immediately. When the GO/DONE bit is set, the selected channel is sampled for the programmed acquisition time before a conversion is actually started. This removes the firmware overhead that may have been required to allow for an acquisition (sampling) period (see Register 19-3 and **Section 19.3 "Selecting and Configuring Automatic Acquisition Time"**).

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 19-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 19-2, configures the functions of the port pins. The ADCON2 register, shown in Register 19-3, configures the A/D clock source, programmed acquisition time and justification.

**REGISTER 19-1: ADCON0 REGISTER**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | CHS3 | CHS2 | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | ADON |
| bit 7 | | | | | | | bit 0 |

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

bit 5-3 **CHS3:CHS0:** Analog Channel Select bits
0000 = Channel 0 (AN0)
0001 = Channel 1 (AN1)
0010 = Channel 2 (AN2)
0011 = Channel 3 (AN3)
0100 = Channel 4 (AN4)
0101 = Channel 5 (AN5)**(1,2)**
0110 = Channel 6 (AN6)**(1,2)**
0111 = Channel 7 (AN7)**(1,2)**
1000 = Channel 8 (AN8)
1001 = Channel 9 (AN9)
1010 = Channel 10 (AN10)
1011 = Channel 11 (AN11)
1100 = Channel 12 (AN12)
1101 = Unimplemented**(2)**
1110 = Unimplemented**(2)**
1111 = Unimplemented**(2)**

**Note 1:** These channels are not implemented on the PIC18F2X20 (28-pin) devices.
**2:** Performing a conversion on unimplemented channels returns full-scale results.

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress
0 = A/D Idle

bit 0 **ADON:** A/D On bit
1 = A/D converter module is enabled
0 = A/D converter module is disabled

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

**REGISTER 19-2: ADCON1 REGISTER**

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-q(1) | R/W-q(1) | R/W-q(1) | R/W-q(1) |
|---|---|---|---|---|---|---|---|
| — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 |
| bit 7 | | | | | | | bit 0 |

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

bit 5 **VCFG1:** Voltage Reference Configuration bit, $V_{REFL}$ Source
1 = $V_{REF-}$ (AN2)
0 = $AV_{SS}$

bit 4 **VCFG0:** Voltage Reference Configuration bit, $V_{REFH}$ Source
1 = $V_{REF+}$ (AN3)
0 = $AV_{DD}$

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

| PCFG3: PCFG0 | AN12 | AN11 | AN10 | AN9 | AN8 | AN7(2) | AN6(2) | AN5(2) | AN4 | AN3 | AN2 | AN1 | AN0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000(1) | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0001 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0010 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0011 | D | A | A | A | A | A | A | A | A | A | A | A | A |
| 0100 | D | D | A | A | A | A | A | A | A | A | A | A | A |
| 0101 | D | D | D | A | A | A | A | A | A | A | A | A | A |
| 0110 | D | D | D | D | A | A | A | A | A | A | A | A | A |
| 0111(1) | D | D | D | D | D | A | A | A | A | A | A | A | A |
| 1000 | D | D | D | D | D | D | A | A | A | A | A | A | A |
| 1001 | D | D | D | D | D | D | D | A | A | A | A | A | A |
| 1010 | D | D | D | D | D | D | D | D | A | A | A | A | A |
| 1011 | D | D | D | D | D | D | D | D | D | A | A | A | A |
| 1100 | D | D | D | D | D | D | D | D | D | D | A | A | A |
| 1101 | D | D | D | D | D | D | D | D | D | D | D | A | A |
| 1110 | D | D | D | D | D | D | D | D | D | D | D | D | A |
| 1111 | D | D | D | D | D | D | D | D | D | D | D | D | D |

A = Analog input D = Digital I/O

**Note 1:** The POR value of the PCFG bits depends on the value of the PBAD bit in Configuration Register 3H. When PBAD = 1, PCFG<3:0> = 0000; when PBAD = 0, PCFG<3:0> = 0111.

**2:** AN5 through AN7 are available only in PIC18F4X20 devices.

| **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 |


© 2003 Microchip Technology Inc.

**REGISTER 19-3: ADCON2 REGISTER**

| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified
0 = Left justified

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

bit 5-3 **ACQT2:ACQT0:** A/D Acquisition Time Select bits
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS1:ADCS0:** A/D Conversion Clock Select bits
111 = FRC (clock derived from A/D RC oscillator)(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock derived from A/D RC oscillator)(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

**Note 1:** If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

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

The analog reference voltage is software selectable to either the device's positive and negative supply voltage ($AV_{DD}$ and $AV_{SS}$), or the voltage level on the RA3/AN3/$V_{REF+}$ and RA2/AN2/$V_{REF-}$/$CV_{REF}$ pins.

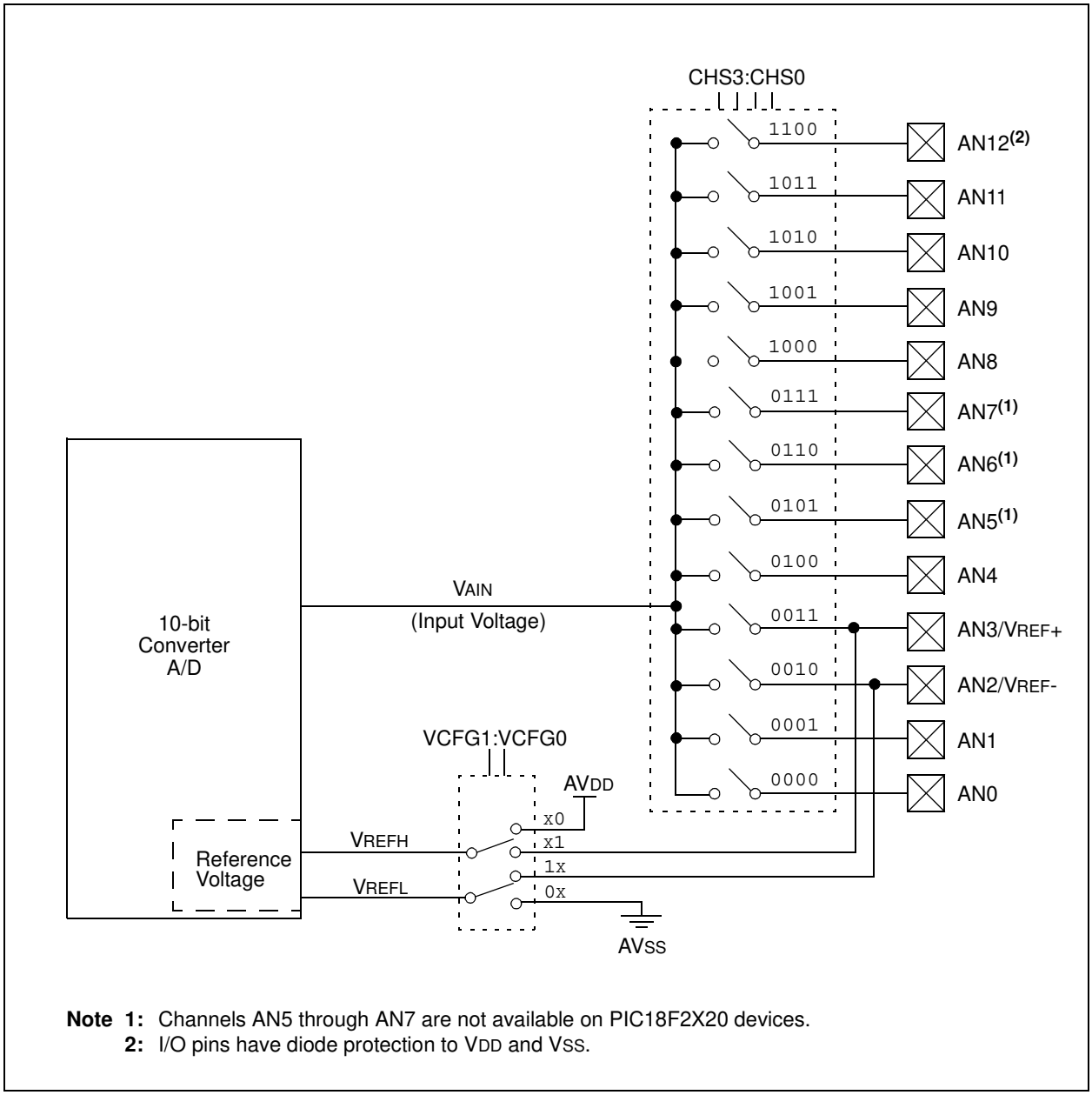
The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in SLEEP, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

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

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

Each port pin associated with the A/D converter can be configured as an analog input or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH/ADRESL registers, the GO/$\overline{DONE}$ bit (ADCON0 register) is cleared and A/D Interrupt Flag bit, ADIF, is set. The block diagram of the A/D module is shown in Figure 19-1.

**FIGURE 19-1: A/D BLOCK DIAGRAM**




**Note 1:** Channels AN5 through AN7 are not available on PIC18F2X20 devices.
**2:** I/O pins have diode protection to $V_{DD}$ and $V_{SS}$.


© 2003 Microchip Technology Inc.

The value in the ADRESH/ADRESL registers is not modified for a Power-on Reset. The ADRESH/ADRESL registers will contain unknown data after a Power-on Reset.

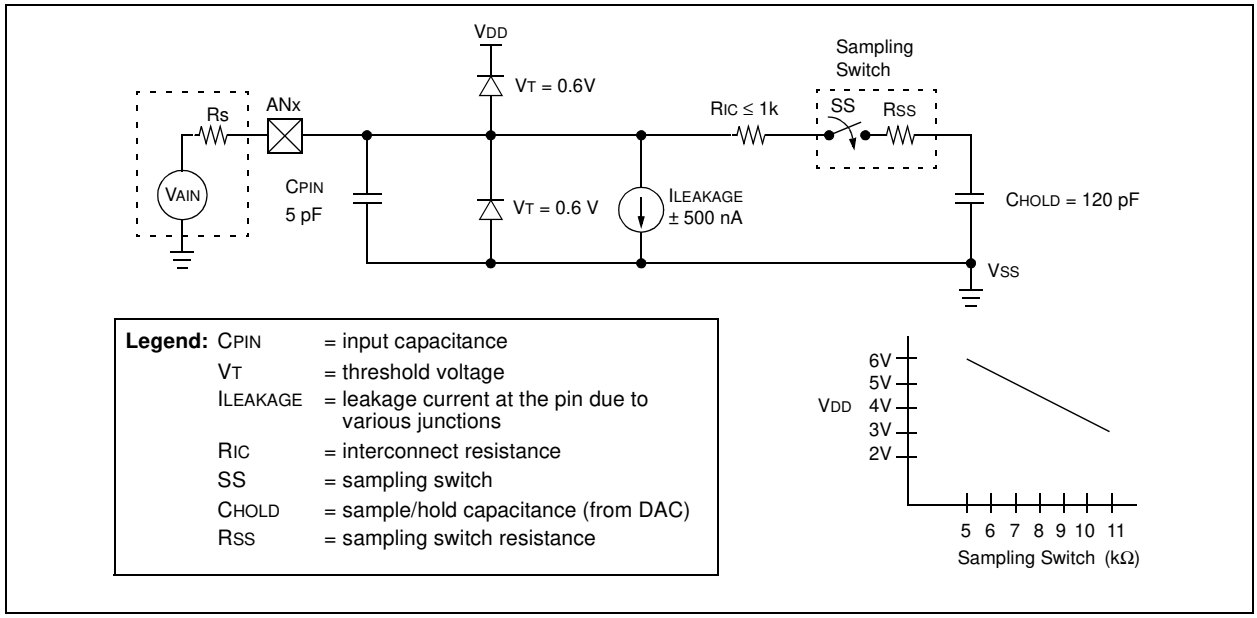
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as an input. To determine acquisition time, see **Section 19.1 "A/D Acquisition Requirements"**. After this acquisition time has elapsed, the A/D conversion can be started. An acquisition time can be programmed to occur between setting the GO/$\overline{\text{DONE}}$ bit and the actual start of the conversion.

The following steps should be followed to do an A/D conversion:

1. Configure the A/D module:
   - Configure analog pins, voltage reference and digital I/O (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D acquisition time (ADCON2)
   - Select A/D conversion clock (ADCON2)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set GIE bit
3. Wait the required acquisition time (if required).
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0 register)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR
   - Waiting for the A/D interrupt
6. Read A/D Result registers (ADRESH:ADRESL); clear bit ADIF if required.
7. For next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2 TAD is required before next acquisition starts.

**FIGURE 19-2: ANALOG INPUT MODEL**



## 19.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the Charge Holding Capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 19-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ.** After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

| **Note:** | When the conversion is started, the holding capacitor is disconnected from the input pin. |
|---|---|

To calculate the minimum acquisition time, Equation 19-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 19-1 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following application system assumptions:

| | | |
|---|---|---|
| CHOLD | = | 120 pF |
| RS | = | 2.5 kΩ |
| Conversion Error | ≤ | 1/2 LSb |
| VDD | = | 5V → Rss = 7 kΩ |
| Temperature | = | 50°C (system max.) |
| VHOLD | = | 0V @ time = 0 |

## 19.2 A/D VREF+ and VREF- References

If external voltage references are used instead of the internal AVDD and AVSS sources, the source impedance of the VREF+ and VREF- voltage sources must be considered. During acquisition, currents supplied by these sources are insignificant. However, during conversion, the A/D module sinks and sources current through the reference sources.

In order to maintain the A/D accuracy, the voltage reference source impedances should be kept low to reduce voltage changes. These voltage changes occur as reference currents flow through the reference source impedance. **The maximum recommended impedance of the VREF+ and VREF- external reference voltage sources is 75Ω.**

| **Note:** | When using external references, the source impedance of the external voltage references must be less than 75Ω in order to achieve the specified ADC resolution. A higher reference source impedance will increase the ADC offset and gain errors. Resistive voltage dividers will not provide a low enough source impedance. To ensure the best possible ADC performance, external VREF inputs should be buffered with an op amp or other low-impedance circuit. |
|---|---|

**EQUATION 19-1: ACQUISITION TIME**

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

**EQUATION 19-2: MINIMUM A/D HOLDING CAPACITOR**

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

or

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ \ln(1/2048)$$

**EXAMPLE 19-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME**

| | | |
|---|---|---|
| TACQ | = | TAMP + TC + TCOFF |
| TAMP | = | 5 µs |
| TCOFF | = | (Temp – 25°C)(0.05 µs/°C)<br>(50°C – 25°C)(0.05 µs/°C)<br>1.25 µs |
| Temperature coefficient is only required for temperatures > 25°C. Below 25°C, TCOFF = 0 µs. | | |
| TC | – | -(CHOLD)(RIC + RSS + RS) ln(1/2047) µs<br>-(120 pF) (1 kΩ + 7 kΩ + 2.5 kΩ) ln(0.0004883) µs<br>9.61 µs |
| TACQ | = | 5 µs + 1.25 µs + 9.61 µs<br>12.86 µs |


© 2003 Microchip Technology Inc.

## 19.3 Selecting and Configuring Automatic Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time the GO/$\overline{\text{DONE}}$ bit is set.

When the GO/$\overline{\text{DONE}}$ bit is set, sampling is stopped and a conversion begins. The user is responsible for ensuring the required acquisition time has passed between selecting the desired input channel and setting the GO/$\overline{\text{DONE}}$ bit. This occurs when the ACQT2:ACQT0 bits (ADCON2<5:3>) remain in their Reset state ('000') and is compatible with devices that do not offer programmable acquisition times.

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When the GO/$\overline{\text{DONE}}$ bit is set, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and setting the GO/$\overline{\text{DONE}}$ bit.

In either case, when the conversion is completed, the GO/$\overline{\text{DONE}}$ bit is cleared, the ADIF flag is set and the A/D begins sampling the currently selected channel again. If an acquisition time is programmed, there is nothing to indicate if the acquisition time has ended or if the conversion has begun.

## 19.4 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 11 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are seven possible options for TAD:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal RC Oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be as short as possible, but greater than the minimum TAD (approximately 2 µs, see parameter #130 for more information).

Table 19-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

**TABLE 19-1: TAD vs. DEVICE OPERATING FREQUENCIES**

| AD Clock Source (TAD) | | Maximum Device Frequency | |
|---|---|---|---|
| **Operation** | **ADCS2:ADCS0** | **PIC18FXX20** | **PIC18LFXX20(4)** |
| 2 TOSC | 000 | 1.25 MHz | 666 kHz |
| 4 TOSC | 100 | 2.50 MHz | 1.33 MHz |
| 8 TOSC | 001 | 5.00 MHz | 2.66 MHz |
| 16 TOSC | 101 | 10.0 MHz | 5.33 MHz |
| 32 TOSC | 010 | 20.0 MHz | 10.65 MHz |
| 64 TOSC | 110 | 40.0 MHz | 21.33 MHz |
| RC(3) | x11 | 1.00 MHz(1) | 1.00 MHz(2) |

**Note 1:** The RC source has a typical TAD time of 4 µs.

**2:** The RC source has a typical TAD time of 6 µs.

**3:** For device frequencies above 1 MHz, the device must be in Sleep for the entire conversion or the A/D accuracy may be out of specification.

**4:** Low-power devices only.

## 19.5 Operation in Power Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined in part by the clock source and frequency while in a power managed mode.

If the A/D is expected to operate while the device is in a power managed mode, the ACQT2:ACQT0 and ADCS2:ADCS0 bits in ADCON2 should be updated in accordance with the power managed mode clock that will be used. After the power managed mode is entered (either of the power managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power managed Idle mode during the conversion.

If the power managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in Sleep mode requires the A/D RC clock to be selected. If bits ACQT2:ACQT0 are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS bits in the OSCCON register must have already been cleared prior to starting the conversion.

## 19.6 Configuring Analog Port Pins

The ADCON1, TRISA, TRISB and TRISE registers all configure the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level ($V_{OH}$ or $V_{OL}$) will be converted.

The A/D operation is independent of the state of the CHS3:CHS0 bits and the TRIS bits.

**Note 1:** When reading the port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will be accurately converted.

**2:** Analog levels on any pin defined as a digital input may cause the digital input buffer to consume current out of the device's specification limits.

**3:** The PBADEN bit in the Configuration register configures PORTB pins to reset as analog or digital pins by controlling how the PCFG0 bits in ADCON1 are reset.


© 2003 Microchip Technology Inc.

## 19.7 A/D Conversions

Figure 19-3 shows the operation of the A/D converter after the GO bit has been set and the ACQT2:ACQT0 bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins.

Figure 19-4 shows the operation of the A/D converter after the GO bit has been set and the ACQT2:ACQT0 bits are set to '010' and selecting a 4 TAD acquisition time before the conversion starts.

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D Result register pair will NOT be updated with the partially completed A/D conversion sample. This means the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers).

After the A/D conversion is completed or aborted, a 2 TAD wait is required before the next acquisition can be started. After this wait, acquisition on the selected channel is automatically started.

| **Note:** | The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D. |
|---|---|

**FIGURE 19-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)**

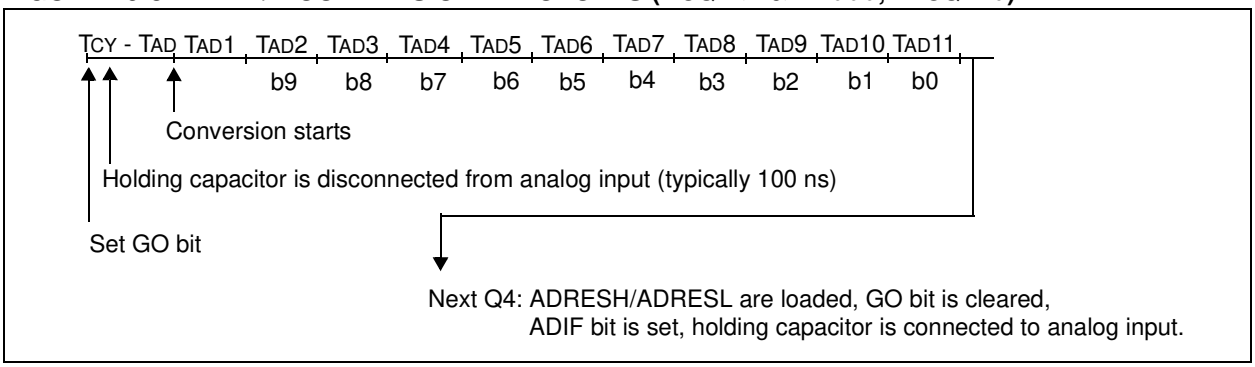
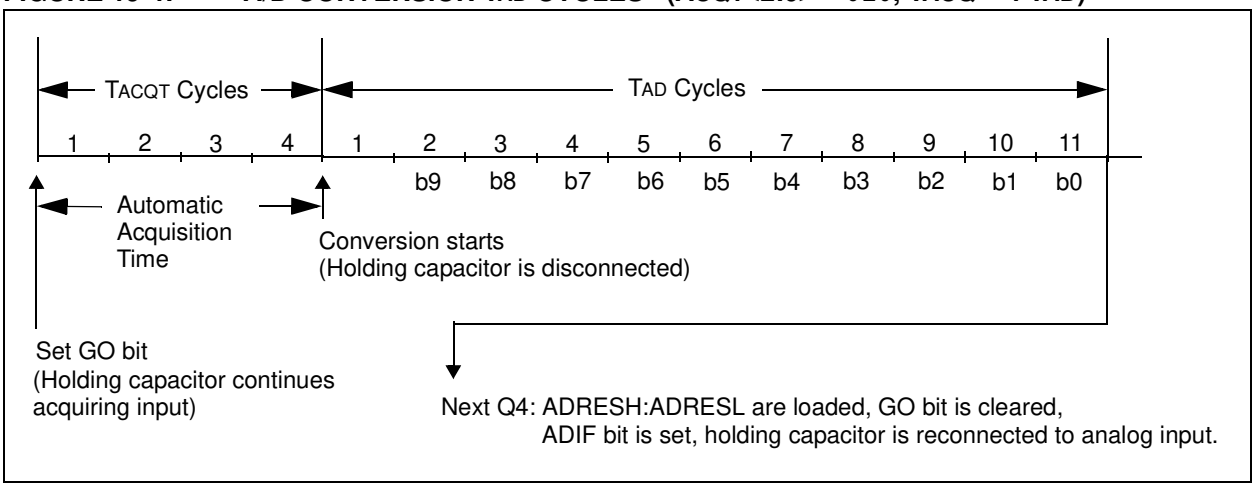



**FIGURE 19-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)**



## 19.8 Use of the CCP2 Trigger

An A/D conversion can be started by the "special event trigger" of the CCP2 module. This requires that the CCP2M3:CCP2M0 bits (CCP2CON<3:0>) be programmed as '1011' and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit will be set, starting the A/D acquisition and conversion and the Timer1 (or Timer3) counter will be reset to zero. Timer1 (or Timer3) is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving ADRESH/ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition period is either timed by the user or an appropriate TACQ time, selected before the "special event trigger", sets the GO/$\overline{\text{DONE}}$ bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), the "special event trigger" will be ignored by the A/D module but will still reset the Timer1 (or Timer3) counter.

**TABLE 19-2: SUMMARY OF A/D REGISTERS**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 0000 | 0000 0000 |
| PIR1 | PSPIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| PIE1 | PSPIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| IPR1 | PSPIP | ADIP | RCIP | TXIP | SSPIP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 1111 1111 |
| PIR2 | OSCFIF | CMIF | — | EEIF | BCLIF | LVDIF | TMR3IF | CCP2IF | 00-0 0000 | 00-0 0000 |
| PIE2 | OSCFIE | CMIE | — | EEIE | BCLIE | LVDIE | TMR3IE | CCP2IE | 00-0 0000 | 00-0 0000 |
| IPR2 | OSCFIP | CMIP | — | EEIP | BCLIP | LVDIP | TMR3IP | CCP2IP | 11-1 1111 | 11-1 1111 |
| ADRESH | A/D Result Register High Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| ADRESL | A/D Result Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| ADCON0 | — | — | CHS3 | CHS3 | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | ADON | --00 0000 | --00 0000 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 qqqq | --00 qqqq |
| ADCON2 | ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 | 0-00 0000 | 0-00 0000 |
| PORTA | RA7(4) | RA6(4) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | --0x 0000 | --0u 0000 |
| TRISA | TRISA7(4) | TRISA6(4) | | | | | | | --11 1111 | --11 1111 |
| PORTB | Read PORTB pins, Write LATB Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| LATB | PORTB Output Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| PORTE | — | — | — | — | RE3(2) | Read PORTE pins, Write LATE(4) | | | ---- xxxx | ---- uuuu |
| TRISE(3) | IBF | OBE | IBOV | PSPMODE | — | PORTE Data Direction | | | 0000 -111 | 0000 -111 |
| LATE(3) | — | — | — | — | PORTE Output Data Latch | | | | ---- -xxx | ---- -uuu |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends on condition.
Shaded cells are not used for A/D conversion.

**Note 1:** RE3 port bit is available only as an input pin when MCLRE bit in configuration register is '0'.
**2:** This register is not implemented on PIC18F2X20 devices.
**3:** These bits are not implemented on PIC18F2X20 devices.
**4:** These pins may be configured as port pins depending on the oscillator mode selected.


© 2003 Microchip Technology Inc.

## 20.0 COMPARATOR MODULE

The comparator module contains two analog comparators. The inputs and outputs for the comparators are multiplexed with the RA0 through RA5 pins. The on-chip voltage reference (**Section 21.0 "Comparator Voltage Reference Module"**) can also be an input to the comparators.

The CMCON register, shown as Register 20-1, controls the comparator module's input and output multiplexers. A block diagram of the various comparator configurations is shown in Figure 20-1.

### 20.1 Comparator Configuration

There are eight modes of operation for the comparators. The CM bits (CMCON<2:0>) are used to select these modes. Figure 20-1 shows the eight possible modes. The TRISA register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in the Electrical Specifications (see **Section 26.0 "Electrical Characteristics"**).

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

**REGISTER 20-1: CMCON REGISTER**

| R-0 | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **C2OUT**: Comparator 2 Output bit
When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
When C2INV = 1:
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT**: Comparator 1 Output bit
When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
When C1INV = 1:
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5 **C2INV**: Comparator 2 Output Inversion bit
1 = C2 output inverted
0 = C2 output not inverted

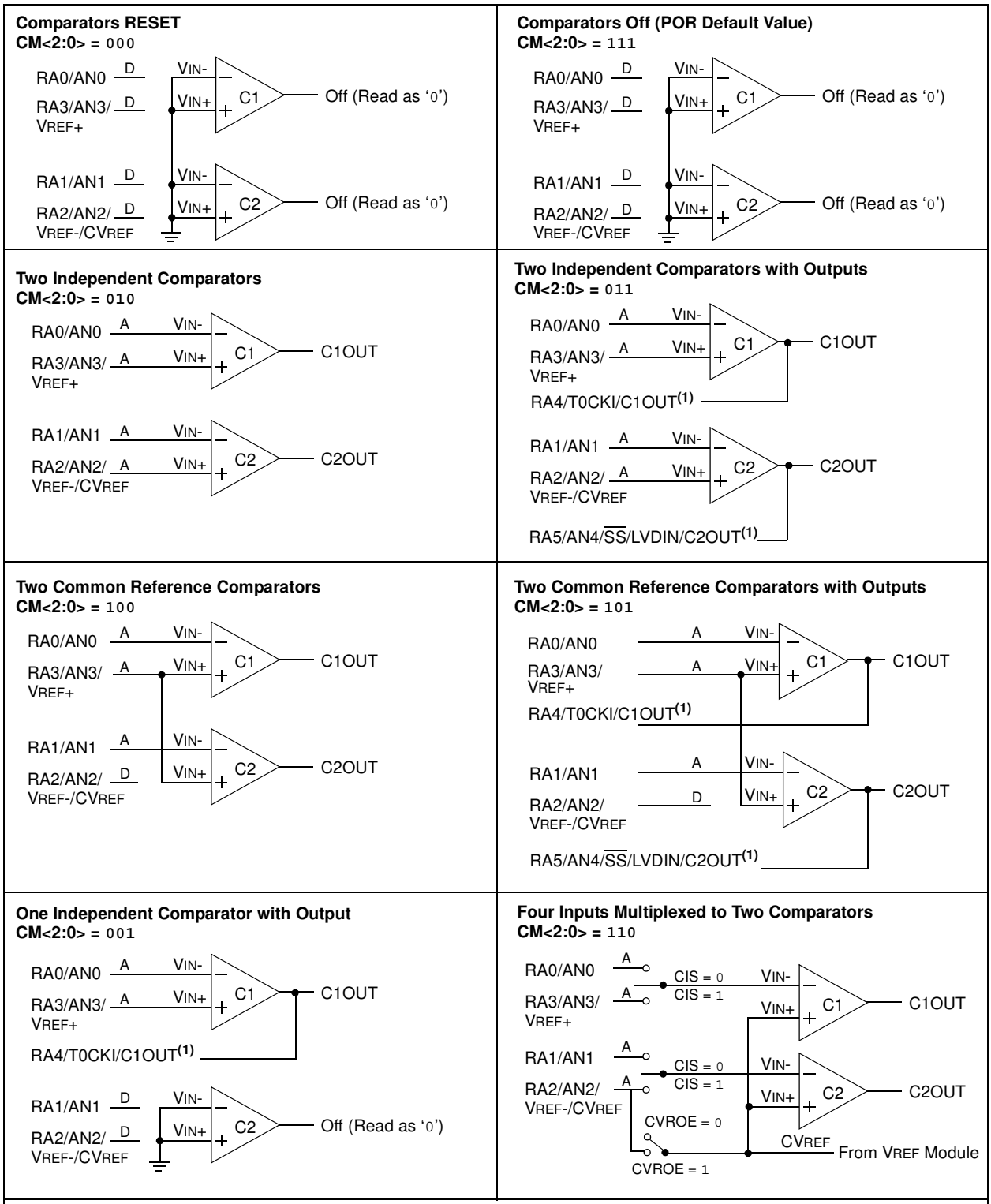
bit 4 **C1INV**: Comparator 1 Output Inversion bit
1 = C1 output inverted
0 = C1 output not inverted

bit 3 **CIS**: Comparator Input Switch bit
When CM2:CM0 = 110:
1 = C1 VIN- connects to RA3/AN3
C2 VIN- connects to RA2/AN2
0 = C1 VIN- connects to RA0/AN0
C2 VIN- connects to RA1/AN1

bit 2-0 **CM2:CM0**: Comparator Mode bits
Figure 20-1 shows the Comparator modes and CM2:CM0 bit settings.

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

**FIGURE 20-1: COMPARATOR I/O OPERATING MODES**




A = Analog Input, port reads zeros always, overrides TRISA bit(2).
D = Digital Input.
CIS (CMCON<3>) is the Comparator Input Switch; CVROE (CVRCON<6>) is the Voltage Reference Output Switch.

**Note 1:** RA4 must be configured as an output pin in TRISA<4> when used to output C1OUT. RA5 ignores TRISA<5> when used as an output for C2OUT.

**2:** Mode 110 is exception. Comparator input pins obey TRISA bits.


© 2003 Microchip Technology Inc.

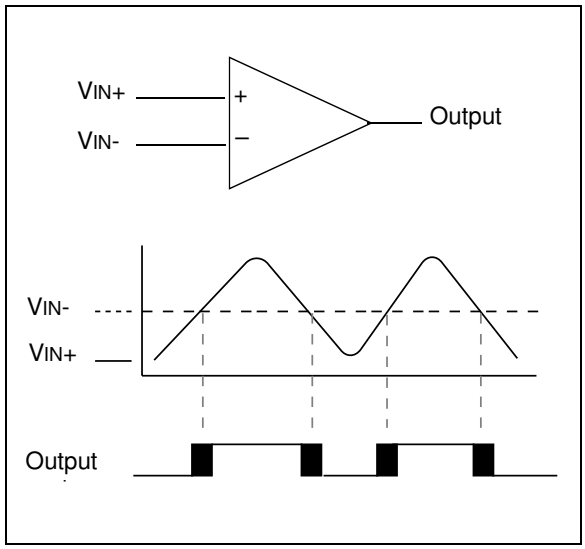
## 20.2 Comparator Operation

A single comparator is shown in Figure 20-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 20-2 represent the uncertainty due to input offsets and response time.

## 20.3 Comparator Reference

An external or internal reference signal may be used depending on the comparator operating mode. The analog signal present at VIN- is compared to the signal at VIN+ and the digital output of the comparator is adjusted accordingly (Figure 20-2).

**FIGURE 20-2: SINGLE COMPARATOR**




### 20.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD and can be applied to either pin of the comparator(s).

### 20.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference for the comparators. **Section 21.0 "Comparator Voltage Reference Module"** contains a detailed description of the comparator voltage reference module that provides this signal. The internal reference signal is used when comparators are in mode, CM2:CM0 = 110 (Figure 20-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

Depending on the setting of the CVROE bit (CVRCON<6>), the voltage reference may also be available on pin RA2.

## 20.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise, the maximum delay of the comparators should be used (see Table 26-2 in **Section 26.0 "Electrical Characteristics"**).

## 20.5 Comparator Outputs

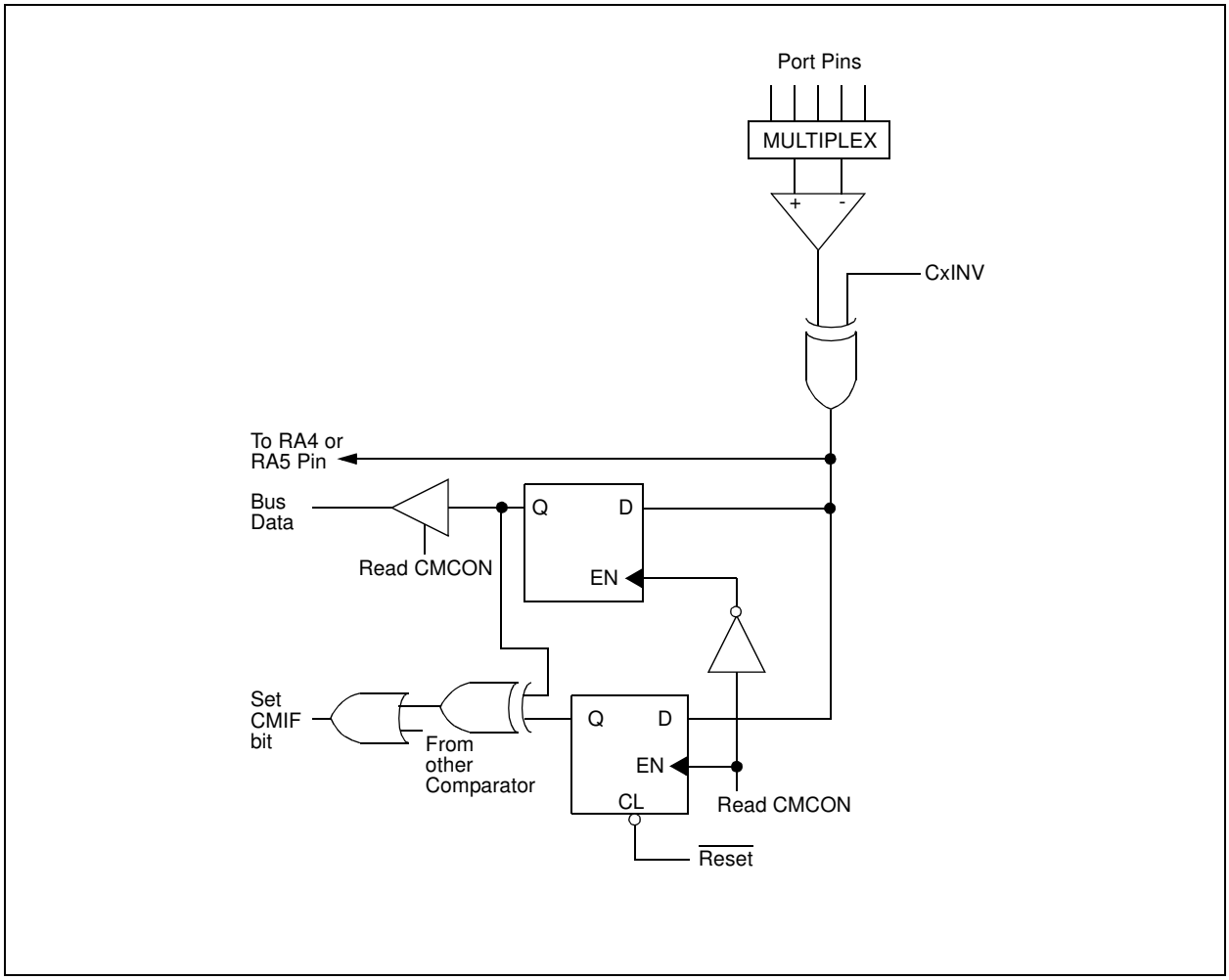
The comparator outputs are read through the CMCON register. These bits are read-only. The comparator outputs may also be directly output to the RA4 and RA5 I/O pins. When enabled, multiplexers in the output path of the RA4 and RA5 pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 20-3 shows the comparator output block diagram.

The TRISA bits will still function as an output enable/ disable for the RA4 and RA5 pins while in this mode.

The polarity of the comparator outputs can be changed using the C2INV and C1INV bits (CMCON<4:5>).

| | |
|---|---|
| **Note 1:** | When reading the Port register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert an analog input according to the Schmitt Trigger input specification. |
| **2:** | Analog levels on any pin defined as a digital input may cause the input buffer to consume more current than is specified. |

**FIGURE 20-3: COMPARATOR OUTPUT BLOCK DIAGRAM**




## 20.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR registers) is the Comparator Interrupt Flag. The CMIF bit is cleared by firmware. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

The CMIE bit (PIE registers) and the PEIE bit (INTCON register) must be set to enable the interrupt. In addition, the GIE bit must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

| **Note:** | If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR registers) interrupt flag may not get set. |
|---|---|

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit CMIF.

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


© 2003 Microchip Technology Inc.

## 20.7 Comparator Operation in Power Managed Modes

When a comparator is active and the device is placed in a power managed mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from a power managed mode when enabled. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in a power managed mode, turn off the comparators (CM<2:0> = 111) before entering the power managed modes. If the device wakes up from a power managed mode, the contents of the CMCON register are not affected.

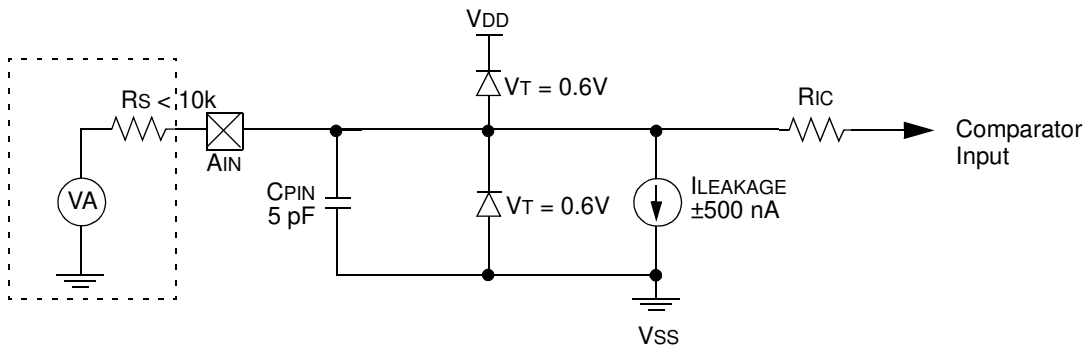
## 20.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator module to be in the Comparator Reset mode (CM<2:0> = 111). This ensures that all potential inputs are analog inputs. Device current is minimized when digital inputs are present at Reset time. The comparators will be powered down during the Reset interval.

## 20.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 20-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. Therefore, the analog input must be between VSS and VDD. If the input voltage exceeds this range by more than 0.6V, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources.

**FIGURE 20-4: COMPARATOR ANALOG INPUT MODEL**




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

**TABLE 20-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 0000 0111 |
| CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 000- 0000 |
| INTCON | GIE/ GIEH | PEIE/ GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 0000 | 0000 0000 |
| PIR2 | — | CMIF | — | — | BCLIF | LVDIF | TMR3IF | CCP2IF | -0-- 0000 | -0-- 0000 |
| PIE2 | — | CMIE | — | — | BCLIE | LVDIE | TMR3IE | CCP2IE | -0-- 0000 | -0-- 0000 |
| IPR2 | — | CMIP | — | — | BCLIP | LVDIP | TMR3IP | CCP2IP | -1-- 1111 | -1-- 1111 |
| PORTA | RA7(1) | RA6(1) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xx0x 0000 | xx0x 0000 |
| LATA | — | — | LATA | Data Output Register | | | | | xxxx xxxx | xxxx xxxx |
| TRISA | — | — | PORTA | Data Direction Register | | | | | 1111 1111 | 1111 1111 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are unused by the comparator module.

**Note 1:** These pins are enabled based on oscillator configuration (see Configuration Register 1H).


© 2003 Microchip Technology Inc.

## 21.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable voltage reference. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The CVRCON register controls the operation of the reference as shown in Register 21-1. The block diagram is given in Figure 21-1.

The comparator reference supply voltage comes from VDD and VSS.

### 21.1 Configuring the Comparator Voltage Reference

The comparator voltage reference can output 16 distinct voltage levels for each range. The equations used to calculate the output of the comparator voltage reference are as follows:

**EQUATION 21-1:**

$$\text{If CVRR} = 1: \quad CV_{REF} = (CVR\langle 3:0\rangle) \bullet \frac{V_{DD}}{24}$$

$$\text{If CVRR} = 0: \quad CV_{REF} = (CVR\langle 3:0\rangle + 8) \bullet \frac{V_{DD}}{32}$$

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 26-2 in **Section 26.0 "Electrical Characteristics"**).

**REGISTER 21-1: CVRCON REGISTER**

| R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **CVREN**: Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

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

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

bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0.00 VDD to 0.75 VDD, with VDD/24 step size
0 = 0.25 VDD to 0.75 VDD, with VDD/32 step size

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

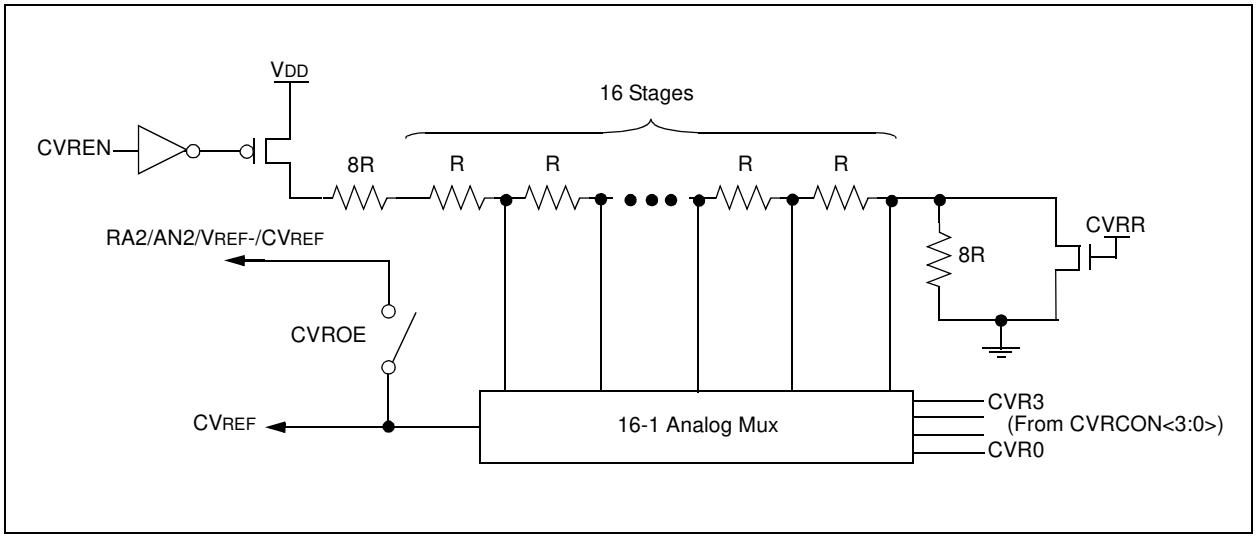
bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection 0 ≤ VR3:VR0 ≤ 15 bits

When CVRR = 1:
$CV_{REF} = (CVR\langle 3:0\rangle) \bullet \frac{V_{DD}}{24}$

When CVRR = 0:
$CV_{REF} = 1/4 \bullet (CV_{RSRC}) + (CVR\langle 3:0\rangle + 8) \bullet \frac{V_{DD}}{32}$

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

**FIGURE 21-1: VOLTAGE REFERENCE BLOCK DIAGRAM**




## 21.2 Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 21-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from VDD; therefore, the CVREF output changes with fluctuations in VDD. The tested absolute accuracy of the voltage reference can be found in **Section 26.0 "Electrical Characteristics"**.

## 21.3 Operation in Power Managed Modes

The contents of the CVRCON register are not affected by entry to or exit from power managed modes. To minimize current consumption in power managed modes, the voltage reference module should be disabled; however, this can cause an interrupt from the comparators so the comparator interrupt should also be disabled while the CVRCON register is being modified.

## 21.4 Effects of a Reset

A device Reset disables the voltage reference by clearing the CVRCON register. This also disconnects the reference from the RA2 pin, selects the high-voltage range and selects the lowest voltage tap from the resistor divider.

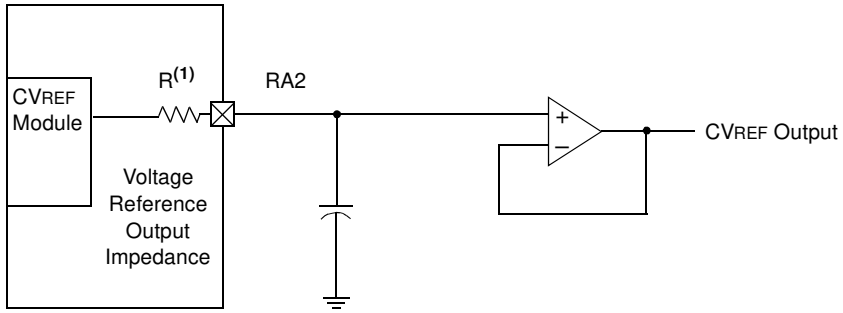
## 21.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be output using the RA2 pin if the CVROE bit is set. Enabling the voltage reference output onto the RA2 pin, with an input signal present, will increase current consumption.

The RA2 pin can be used as a simple D/A output with limited drive capability. Due to the limited current drive capability, an external buffer must be used on the voltage reference output for external connections to VREF. Figure 21-2 shows an example buffering technique.


© 2003 Microchip Technology Inc.

**FIGURE 21-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**




**Note 1:** R is dependent upon the voltage reference configuration bits (CVRCON<3:0> and CVRCON<5>).

**TABLE 21-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|
| CVRCON | CVREN | CVROE | CVRR | — | CVR3 | CVR2 | CVR1 | CVR0 | 000- 0000 | 000- 0000 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 0000 0111 |
| TRISA | RA7(1) | RA6(1) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 1111 1111 | 1111 1111 |

**Legend:** x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are not used with the comparator voltage reference.

**Note 1:** These pins are enabled based on oscillator configuration (see Configuration Register 1H).

**NOTES:**


© 2003 Microchip Technology Inc.

## 22.0 LOW-VOLTAGE DETECT

In many applications, the ability to determine if the device voltage ($V_{DD}$) is below a specified voltage level is a desirable feature. A window of operation for the application can be created, where the application software can do “housekeeping tasks” before the device voltage exits the valid operating range. This can be done using the Low-Voltage Detect (LVD) module.

This module is a software programmable circuitry, where a device voltage trip point can be specified. When the voltage of the device becomes lower then the specified point, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to that interrupt source.

The Low-Voltage Detect circuitry is completely under software control. This allows the circuitry to be turned off by the software which minimizes the current consumption for the device.

Figure 22-1 shows a possible application voltage curve (typically for batteries). Over time, the device voltage decreases. When the device voltage equals voltage $V_A$, the LVD logic generates an interrupt. This occurs at time $T_A$. The application software then has the time, until the device voltage is no longer in valid operating range, to shut down the system. Voltage point $V_B$ is the minimum valid operating voltage specification. This occurs at time $T_B$. The difference, $T_B - T_A$, is the total time for shutdown.

The block diagram for the LVD module is shown in Figure 22-2. A comparator uses an internally generated reference voltage as the set point. When the selected tap output of the device voltage crosses the set point (is lower than), the LVDIF bit is set.

Each node in the resistor divider represents a “trip point” voltage. The “trip point” voltage is the minimum supply voltage level at which the device can operate before the LVD module asserts an interrupt. When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the 1.2V internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal setting the LVDIF bit. This voltage is software programmable to any one of 16 values (see Figure 22-2). The trip point is selected by programming the LVDL3:LVDL0 bits (LVDCON<3:0>).

**FIGURE 22-1: TYPICAL LOW-VOLTAGE DETECT APPLICATION**

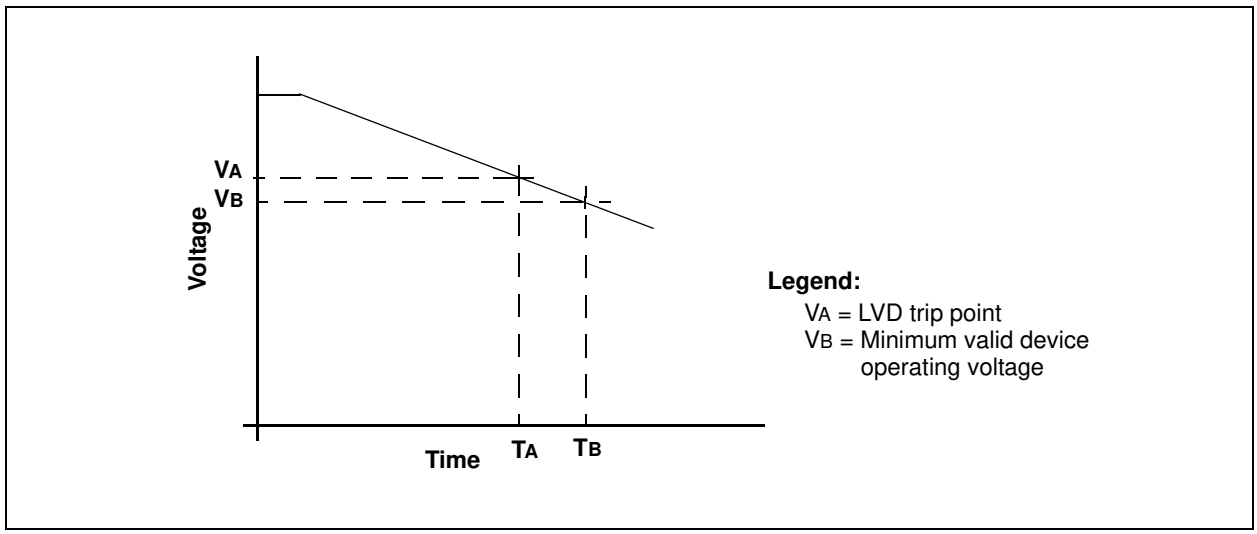
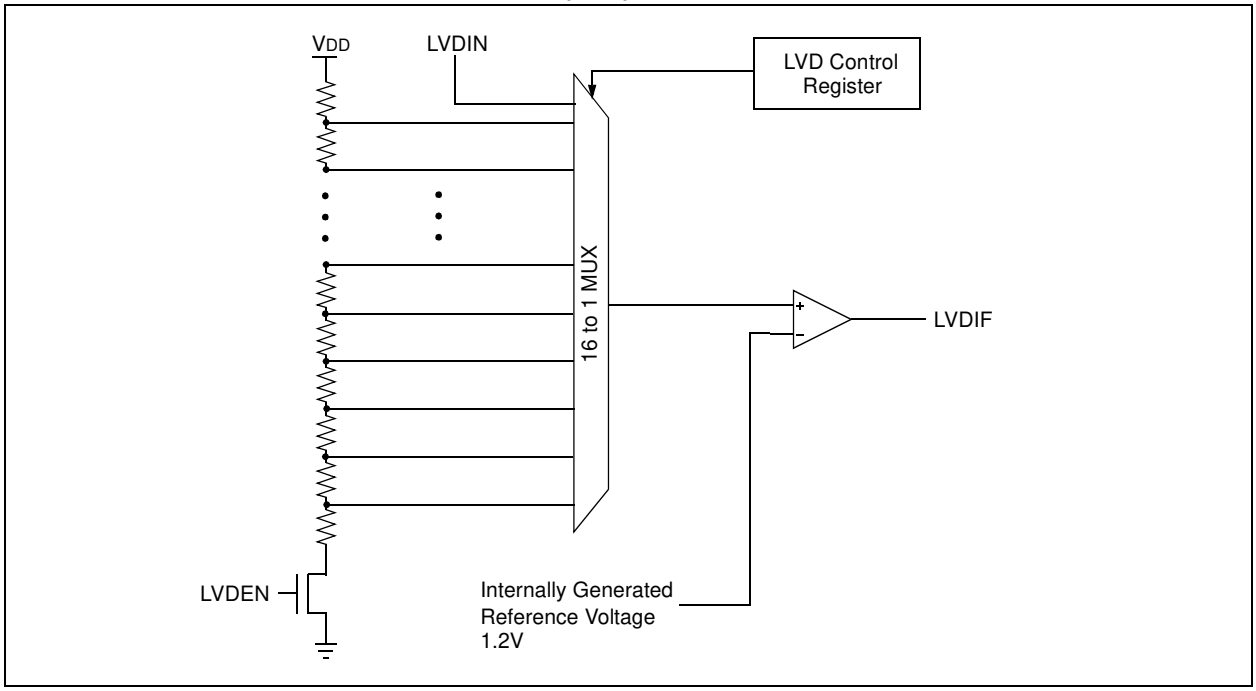
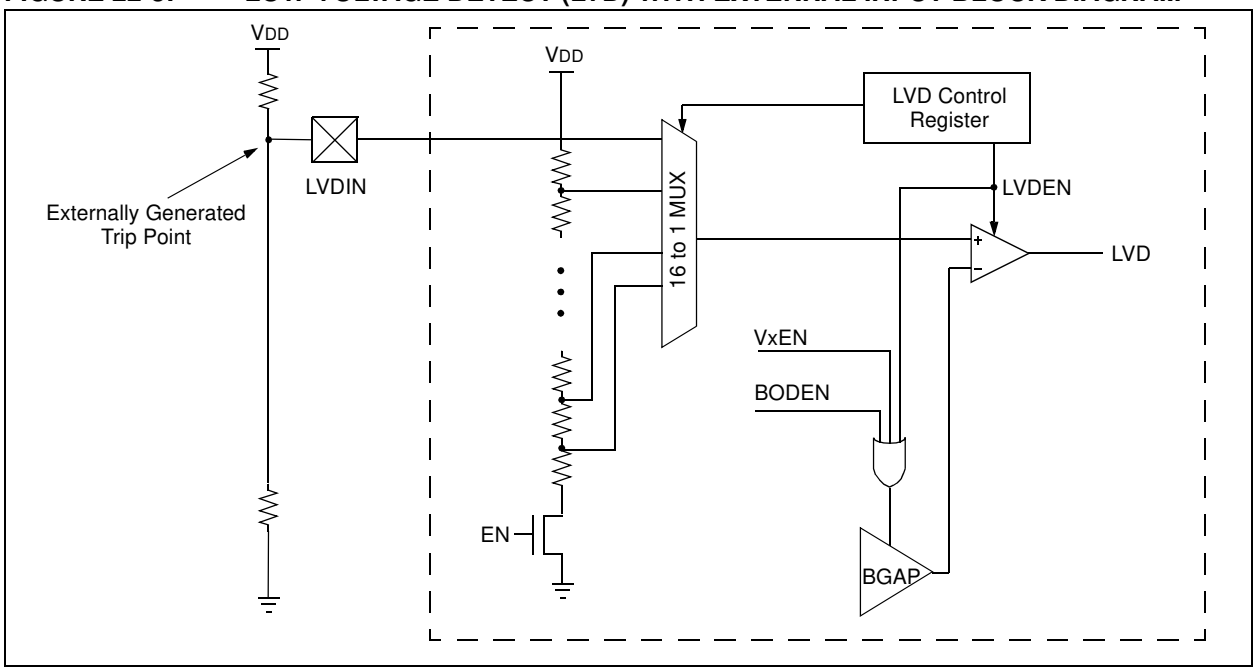


**FIGURE 22-2: LOW-VOLTAGE DETECT (LVD) BLOCK DIAGRAM**




The LVD module has an additional feature that allows the user to supply the sense voltage to the module from an external source. This mode is enabled when bits LVDL3:LVDL0 are set to '1111'. In this state, the comparator input is multiplexed from the external input pin, LVDIN (Figure 22-3). This gives users flexibility because it allows them to configure the Low-Voltage Detect interrupt to occur at any voltage in the valid operating range.

**FIGURE 22-3: LOW-VOLTAGE DETECT (LVD) WITH EXTERNAL INPUT BLOCK DIAGRAM**





© 2003 Microchip Technology Inc.

## 22.1 Control Register

The Low-Voltage Detect Control register controls the operation of the Low-Voltage Detect circuitry.

**REGISTER 22-1: LVDCON REGISTER**

| U-0 | U-0 | R-0 | R/W-0 | R/W-0 | R/W-1 | R/W-0 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | IRVST | LVDEN | LVDL3 | LVDL2 | LVDL1 | LVDL0 |
| bit 7 | | | | | | | bit 0 |

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

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit

1 = Indicates that the Low-Voltage Detect logic will generate the interrupt flag at the specified voltage range

0 = Indicates that the Low-Voltage Detect logic will not generate the interrupt flag at the specified voltage range and the LVD interrupt should not be enabled

bit 4 **LVDEN:** Low-Voltage Detect Power Enable bit

1 = Enables LVD, powers up LVD circuit

0 = Disables LVD, powers down LVD circuit

bit 3-0 **LVDL3:LVDL0:** Low-Voltage Detection Limit bits

1111 = External analog input is used (input comes from the LVDIN pin)
1110 = 4.50V-4.78V
1101 = 4.20V-4.46V
1100 = 4.00V-4.26V
1011 = 3.80V-4.04V
1010 = 3.60V-3.84V
1001 = 3.50V-3.72V
1000 = 3.30V-3.52V
0111 = 3.00V-3.20V
0110 = 2.80V-2.98V
0101 = 2.70V-2.86V
0100 = 2.50V-2.66V
0011 = 2.40V-2.55V
0010 = 2.20V-2.34V
0001 = 2.00V-2.12V
0000 = Reserved

**Note:** LVDL3:LVDL0 modes which result in a trip point below the valid operating voltage of the device are not tested.

| **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 |

## 22.2 Operation

Depending on the power source for the device voltage, the voltage normally decreases relatively slowly. This means that the LVD module does not need to be constantly operating. To decrease the current requirements, the LVD circuitry only needs to be enabled for short periods where the voltage is checked. After doing the check, the LVD module may be disabled.

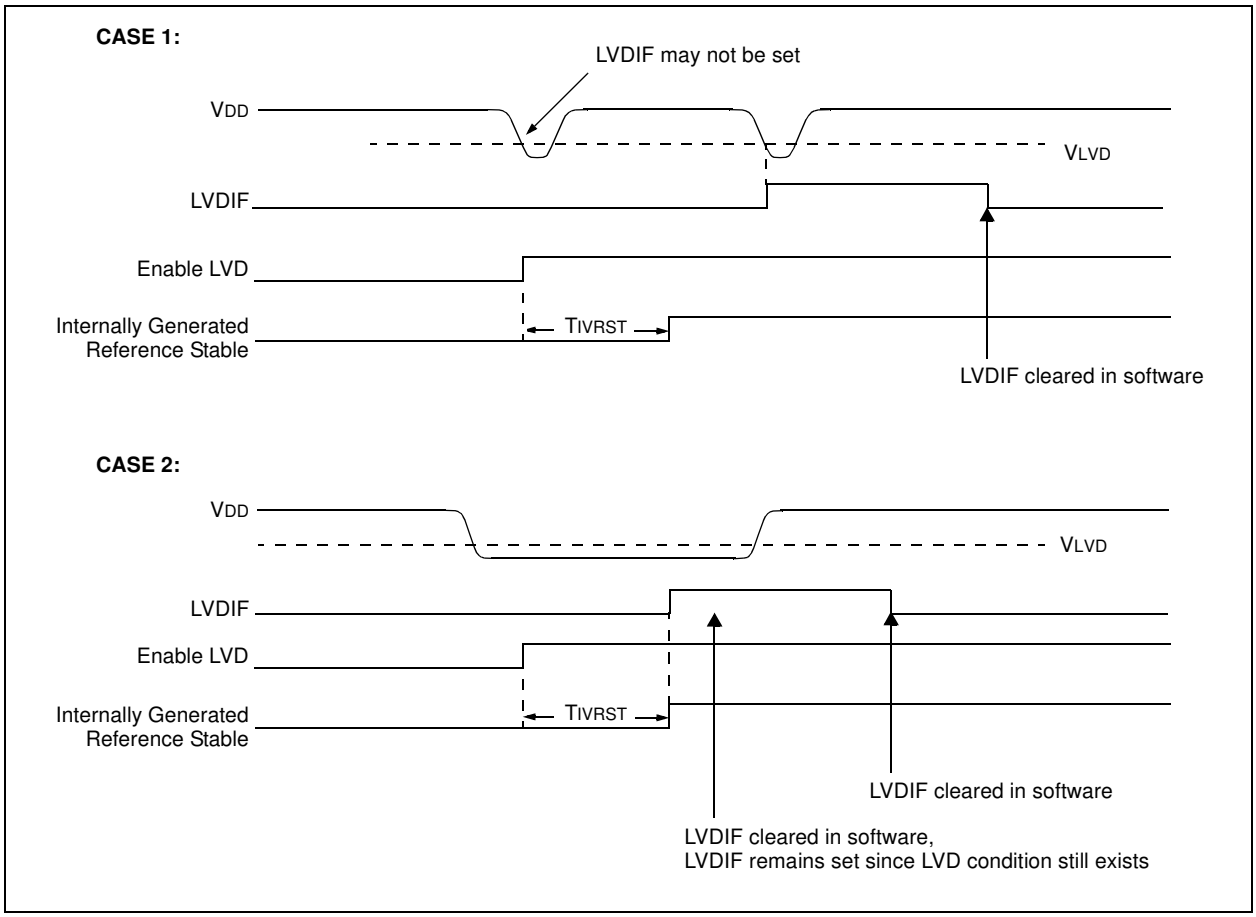
Each time that the LVD module is enabled, the circuitry requires some time to stabilize. After the circuitry has stabilized, all status flags may be cleared. The module will then indicate the proper state of the system.

The following steps are needed to set up the LVD module:

1. Write the value to the LVDL3:LVDL0 bits (LVDCON register) which selects the desired LVD trip point.
2. Ensure that LVD interrupts are disabled (the LVDIE bit is cleared or the GIE bit is cleared).
3. Enable the LVD module (set the LVDEN bit in the LVDCON register).
4. Wait for the LVD module to stabilize (the IRVST bit to become set).
5. Clear the LVD interrupt flag, which may have falsely become set, until the LVD module has stabilized (clear the LVDIF bit).
6. Enable the LVD interrupt (set the LVDIE and the GIE bits).

Figure 22-4 shows typical waveforms that the LVD module may be used to detect.

**FIGURE 22-4: LOW-VOLTAGE DETECT WAVEFORMS**





© 2003 Microchip Technology Inc.

#### 22.2.1 REFERENCE VOLTAGE SET POINT

The internal reference voltage of the LVD module may be used by other internal circuitry (the Programmable Brown-out Reset). If these circuits are disabled (lower current consumption), the reference voltage circuit requires a time to become stable before a low-voltage condition can be reliably detected. This time is invariant of system clock speed. This start-up time is specified in electrical specification parameter #36. The low-voltage interrupt flag will not be enabled until a stable reference voltage is reached. Refer to the waveform in Figure 22-4.

#### 22.2.2 CURRENT CONSUMPTION

When the module is enabled, the LVD comparator and voltage divider are enabled and will consume static current. The voltage divider can be tapped from multiple places in the resistor array. Total current consumption, when enabled, is specified in electrical specification parameter #D022B.

### 22.3 Operation During Sleep

When enabled, the LVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the LVDIF bit will be set and the device will wake-up from Sleep. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

### 22.4 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the LVD module to be turned off.

**NOTES:**


© 2003 Microchip Technology Inc.

## 23.0 SPECIAL FEATURES OF THE CPU

PIC18F2X20/4X20 devices include several features intended to maximize system reliability and minimize cost through elimination of external components. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- In-Circuit Serial Programming

The oscillator can be configured for the application depending on frequency, power, accuracy and cost. All of the options are discussed in detail in **Section 2.0 "Oscillator Configurations"**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, PIC18F2X20/4X20 devices have a Watchdog Timer which is either permanently enabled via the configuration bits or software controlled (if configured as disabled).

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate configuration register bits.

### 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 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFFh) which can only be accessed using table reads and table writes.

Programming the configuration registers is done in a manner similar to programming the Flash memory. The EECON1 register WR bit starts a self-timed write to the configuration register. In normal operation mode, a TBLWT instruction with the TBLPTR pointing to the configuration register sets up the address and the data for the configuration register write. Setting the WR bit starts a long write to the configuration register. The configuration registers are written a byte at a time. To write or erase a configuration cell, a TBLWT instruction can write a '1' or a '0' into the cell. For additional details on Flash programming, refer to **Section 6.5 "Writing to Flash Program Memory"**.

**TABLE 23-1: CONFIGURATION BITS AND DEVICE IDS**

| File Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Default/ Unprogrammed Value |
|---|---|---|---|---|---|---|---|---|---|---|
| 300001h | CONFIG1H | IESO | FSCM | — | — | FOSC3 | FOSC2 | FOSC1 | FOSC0 | 11-- 1111 |
| 300002h | CONFIG2L | — | — | — | — | BORV1 | BORV0 | BOR | $\overline{\text{PWRT}}$ | ---- 1111 |
| 300003h | CONFIG2H | — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | WDT | ---1 1111 |
| 300005h | CONFIG3H | MCLRE | — | — | — | — | — | PBAD | CCP2MX | 1--- --11 |
| 300006h | CONFIG4L | $\overline{\text{DEBUG}}$ | — | — | — | — | LVP | — | STVR | 1--- -1-1 |
| 300008h | CONFIG5L | — | — | — | — | CP3 | CP2 | CP1 | CP0 | ---- 1111 |
| 300009h | CONFIG5H | CPD | CPB | — | — | — | — | — | — | 11-- ---- |
| 30000Ah | CONFIG6L | — | — | — | — | WRT3 | WRT2 | WRT1 | WRT0 | ---- 1111 |
| 30000Bh | CONFIG6H | WRTD | WRTB | WRTC | — | — | — | — | — | 111- ---- |
| 30000Ch | CONFIG7L | — | — | — | — | EBTR3 | EBTR2 | EBTR1 | EBTR0 | ---- 1111 |
| 30000Dh | CONFIG7H | — | EBTRB | — | — | — | — | — | — | -1-- ---- |
| 3FFFFEh | DEVID1(1) | DEV2 | DEV1 | DEV0 | REV4 | REV3 | REV2 | REV1 | REV0 | xxxx xxxx(1) |
| 3FFFFFh | DEVID2(1) | DEV10 | DEV9 | DEV8 | DEV7 | DEV6 | DEV5 | DEV4 | DEV3 | 0000 0101 |

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

**Note 1:** See Register 23-14 for DEVID1 values. DEVID registers are read-only and cannot be programmed by the user.

**REGISTER 23-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)**

| R/P-1 | R/P-1 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| IESO | FSCM | — | — | FOSC3 | FOSC2 | FOSC1 | FOSC0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **IESO:** Internal External Switch Over bit
1 = Internal External Switch Over mode enabled
0 = Internal External Switch Over mode disabled

bit 6 FSCM: Fail-Safe Clock Monitor enable bit
1 = Fail-Safe Clock Monitor enabled
0 = Fail-Safe Clock Monitor disabled

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **FOSC<3:0>:** Oscillator Selection bits
11xx = External RC oscillator, CLKO function on RA6
1001 = Internal oscillator block, CLKO function on RA6 and port function on RA7
1000 = Internal oscillator block, port function on RA6 and port function on RA7
0111 = External RC oscillator, port function on RA6
0110 = HS oscillator, PLL enabled (clock frequency = 4 x FOSC1)
0101 = EC oscillator, port function on RA6
0100 = EC oscillator, CLKO function on RA6
0010 = HS oscillator
0001 = XT oscillator
0000 = LP oscillator

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |


© 2003 Microchip Technology Inc.

**REGISTER 23-2: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)**

| U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | BORV1 | BORV0 | BOR | $\overline{\text{PWRT}}$ |
| bit 7 | | | | | | | bit 0 |

bit 7-4 **Unimplemented:** Read as '0'

bit 3-2 **BORV1:BORV0:** Brown-out Reset Voltage bits
11 = $V_{BOR}$ set to 2.0V
10 = $V_{BOR}$ set to 2.7V
01 = $V_{BOR}$ set to 4.2V
00 = $V_{BOR}$ set to 4.5V

bit 1 **BOR:** Brown-out Reset enable bit(1)
1 = Brown-out Reset enabled
0 = Brown-out Reset disabled

bit 0 **$\overline{\text{PWRT}}$:** Power-up Timer enable bit(1)
1 = PWRT disabled
0 = PWRT enabled

**Note 1:** The Power-up Timer is decoupled from Brown-out Reset, allowing these features to be independently controlled.

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-3: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)**

| U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | WDT |
| bit 7 | | | | | | | bit 0 |

bit 7-5 **Unimplemented:** Read as '0'

bit 4-1 **WDPS<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

bit 0 **WDT:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled (control is placed on the SWDTEN bit)

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-4: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)**

| R/P-1 | U-0 | U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| MCLRE | — | — | — | — | — | PBAD | CCP2MX |
| bit 7 | | | | | | | bit 0 |

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Pin Enable bit
1 = $\overline{\text{MCLR}}$ pin enabled; RE3 input pin disabled
0 = RE3 input pin enabled; $\overline{\text{MCLR}}$ disabled

bit 6-2 **Unimplemented:** Read as '0'

bit 1 **PBAD:** PORTB A/D Enable bit (Affects ADCON1 Reset state. ADCON1 controls PORTB<4:0> pin configuration.)
1 = PORTB<4:0> pins are configured as analog input channels on Reset
0 = PORTB<4:0> pins are configured as digital I/O on Reset

bit 0 **CCP2MX:** CCP2 Mux bit
1 = CCP2 input/output is multiplexed with RC1
0 = CCP2 input/output is multiplexed with RB3

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-5: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)**

| R/P-1 | U-0 | U-0 | U-0 | U-0 | R/P-1 | U-0 | R/P-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{DEBUG}}$ | — | — | — | — | LVP | — | STVR |
| bit 7 | | | | | | | bit 0 |

bit 7 $\overline{\textbf{DEBUG}}$**:** Background Debugger Enable bit
1 = Background debugger disabled, RB6 and RB7 configured as general purpose I/O pins
0 = Background debugger enabled, RB6 and RB7 are dedicated to In-Circuit Debug

bit 6-3 **Unimplemented:** Read as '0'

bit 2 **LVP:** Low-Voltage ICSP Enable bit
1 = Low-voltage ICSP enabled
0 = Low-voltage ICSP disabled

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

bit 0 **STVR:** Stack Full/Underflow Reset Enable bit
1 = Stack full/underflow will cause Reset
0 = Stack full/underflow will not cause Reset

| **Legend:** | | |
|---|---|---|
| R = Readable bit | C = Clearable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |


© 2003 Microchip Technology Inc.

**REGISTER 23-6: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)**

| U-0 | U-0 | U-0 | U-0 | R/C-1 | R/C-1 | R/C-1 | R/C-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | CP3(1) | CP2(1) | CP1 | CP0 |
| bit 7 | | | | | | | bit 0 |

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **CP3:** Code Protection bit(1)
1 = Block 3 (001800-001FFFh) not code-protected
0 = Block 3 (001800-001FFFh) code-protected

bit 2 **CP2:** Code Protection bit(1)
1 = Block 2 (001000-0017FFh) not code-protected
0 = Block 2 (001000-0017FFh) code-protected

bit 1 **CP1:** Code Protection bit
1 = Block 1 (000800-000FFFh) not code-protected
0 = Block 1 (000800-000FFFh) code-protected

bit 0 **CP0:** Code Protection bit
1 = Block 0 (000200-0007FFh) not code-protected
0 = Block 0 (000200-0007FFh) code-protected

**Note 1:** Unimplemented in PIC18FX220 devices; maintain this bit set.

| **Legend:** | | |
|---|---|---|
| R = Readable bit | C = Clearable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-7: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)**

| R/C-1 | R/C-1 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| CPD | CPB | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

bit 7 **CPD:** Data EEPROM Code Protection bit
1 = Data EEPROM not code-protected
0 = Data EEPROM code-protected

bit 6 **CPB:** Boot Block Code Protection bit
1 = Boot block (000000-0001FFh) not code-protected
0 = Boot block (000000-0001FFh) code-protected

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

| **Legend:** | | |
|---|---|---|
| R = Readable bit | C = Clearable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-8: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)**

| U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | WRT3(1) | WRT2(1) | WRT1 | WRT0 |
| bit 7 | | | | | | | bit 0 |

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **WRT3:** Write Protection bit(1)
1 = Block 3 (001800-001FFFh) not write-protected
0 = Block 3 (001800-001FFFh) write-protected

bit 2 **WRT2:** Write Protection bit(1)
1 = Block 2 (001000-0017FFh) not write-protected
0 = Block 2 (001000-0017FFh) write-protected

bit 1 **WRT1:** Write Protection bit
1 = Block 1 (000800-000FFFh) not write-protected
0 = Block 1 (000800-000FFFh) write-protected

bit 0 **WRT0:** Write Protection bit
1 = Block 0 (000200-0007FFh) not write-protected
0 = Block 0 (000200-0007FFh) write-protected

**Note 1:** Unimplemented in PIC18FX220 devices; maintain this bit set.

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-9: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)**

| R/P-1 | R/P-1 | R-1 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| WRTD | WRTB | WRTC | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

bit 7 **WRTD:** Data EEPROM Write Protection bit
1 = Data EEPROM not write-protected
0 = Data EEPROM write-protected

bit 6 **WRTB:** Boot Block Write Protection bit
1 = Boot block (000000-0001FFh) not write-protected
0 = Boot block (000000-0001FFh) write-protected

bit 5 **WRTC:** Configuration Register Write Protection bit
1 = Configuration registers (300000-3000FFh) not write-protected
0 = Configuration registers (300000-3000FFh) write-protected

**Note:** This bit is read-only in normal execution mode; it can be written only in Program mode.

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

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |


© 2003 Microchip Technology Inc.

**REGISTER 23-10: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)**

| U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | EBTR3(1) | EBTR2(1) | EBTR1 | EBTR0 |
| bit 7 | | | | | | | bit 0 |

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **EBTR3:** Table Read Protection bit(1)
1 = Block 3 (001800-001FFFh) not protected from table reads executed in other blocks
0 = Block 3 (001800-001FFFh) protected from table reads executed in other blocks

bit 2 **EBTR2:** Table Read Protection bit(1)
1 = Block 2 (001000-0017FFh) not protected from table reads executed in other blocks
0 = Block 2 (001000-0017FFh) protected from table reads executed in other blocks

bit 1 **EBTR1:** Table Read Protection bit
1 = Block 1 (000800-000FFFh) not protected from table reads executed in other blocks
0 = Block 1 (000800-000FFFh) protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit
1 = Block 0 (000200-0007FFh) not protected from table reads executed in other blocks
0 = Block 0 (000200-0007FFh) protected from table reads executed in other blocks

**Note 1:** Unimplemented in PIC18FX220 devices; maintain this bit set.

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-11: CONFIG7H: CONFIGURATION REGISTER 7 HIGH (BYTE ADDRESS 30000Dh)**

| U-0 | R/P-1 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | EBTRB | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **EBTRB:** Boot Block Table Read Protection bit
1 = Boot block (000000-0001FFh) not protected from table reads executed in other blocks
0 = Boot block (000000-0001FFh) protected from table reads executed in other blocks

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

| **Legend:** | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-12: DEVICE ID REGISTER 1 FOR PIC18F2220/2320/4220/4320 DEVICES**

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| DEV2 | DEV1 | DEV0 | REV4 | REV3 | REV2 | REV1 | REV0 |
| bit 7 | | | | | | | bit 0 |

bit 7-5 **DEV2:DEV0:** Device ID bits

000 = PIC18F4220
001 = PIC18F4320
100 = PIC18F2220
101 = PIC18F2320

bit 4-0 **REV4:REV0:** Revision ID bits

These bits are used to indicate the device revision.

| **Legend:** | | |
|---|---|---|
| R = Read-only bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |

**REGISTER 23-13: DEVICE ID REGISTER 2 FOR PIC18F2220/2320/4220/4320 DEVICES**

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| DEV10 | DEV9 | DEV8 | DEV7 | DEV6 | DEV5 | DEV4 | DEV3 |
| bit 7 | | | | | | | bit 0 |

bit 7-0 **DEV10:DEV3:** Device ID bits

These bits are used with the DEV2:DEV0 bits in the Device ID Register 1 to identify the part number.

0000 0101 = PIC18F2220/2320/4220/4320 devices

**Note:** These values for DEV10:DEV3 may be shared with other devices. The specific device is always identified by using the entire DEV10:DEV0 bit sequence.

| **Legend:** | | |
|---|---|---|
| R = Read-only bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| - n = Value when device is unprogrammed | | u = Unchanged from programmed state |


© 2003 Microchip Technology Inc.

## 23.2 Watchdog Timer (WDT)

For PIC18F2X20/4X20 devices, the WDT is driven by the INTRC source. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the INTRC oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexer, controlled by bits in Configuration Register 2H. Available periods range from 4 ms to 131.072 seconds (2.18 minutes). The WDT and postscaler are cleared when any of the following events occur: execute a SLEEP or CLRWDT instruction, the IRCF bits (OSCCON<6:4>) are changed or a clock failure has occurred.

Adjustments to the internal oscillator clock period using the OSCTUNE register also affect the period of the WDT by the same factor. For example, if the INTRC period is increased by 3%, then the WDT period is increased by 3%.

**Note 1:** The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.

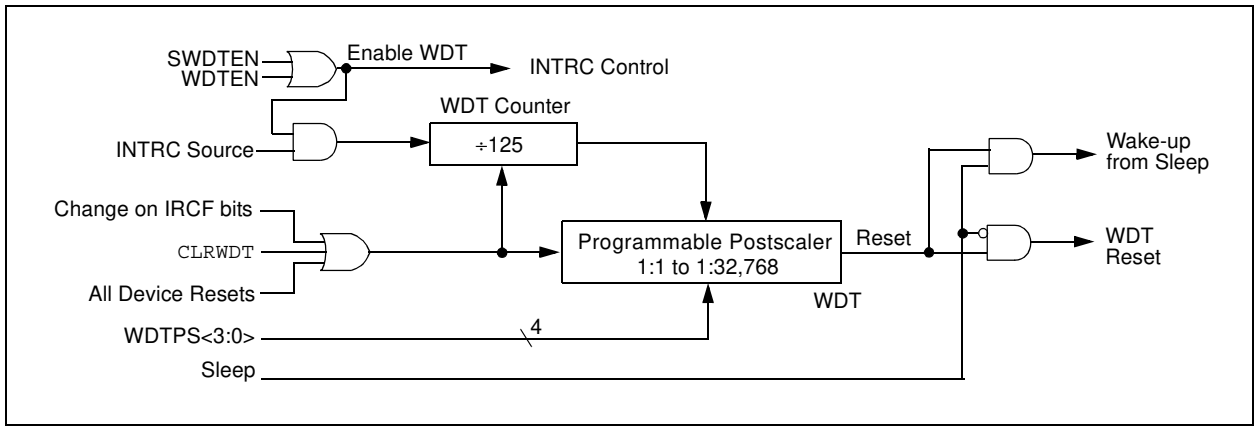
**2:** Changing the setting of the IRCF bits (OSCCON<6:4> clears the WDT and postscaler counts.

**3:** When a CLRWDT instruction is executed, the postscaler count will be cleared.

### 23.2.1 CONTROL REGISTER

Register 23-14 shows the WDTCON register. This is a readable and writable register which contains a control bit that allows software to override the WDT enable configuration bit, but only if the configuration bit has disabled the WDT.

**FIGURE 23-1: WDT BLOCK DIAGRAM**



**REGISTER 23-14: WDTCON REGISTER**

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

bit 7-1 **Unimplemented**: Read as '0'

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit

1 = Watchdog Timer is on
0 = Watchdog Timer is off

**Note 1:** This bit has no effect if the configuration bit, WDTEN (CONFIG2H<0>), is enabled.

| **Legend:** | |
|---|---|
| R = Readable bit | W = Writable bit |
| U = Unimplemented bit, read as '0' | - n = Value at POR |

**TABLE 23-2: SUMMARY OF WATCHDOG TIMER REGISTERS**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| CONFIG2H | — | — | — | WDTPS3 | WDTPS2 | WDTPS2 | WDTPS0 | WDTEN |
| RCON | IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ |
| WDTCON | — | — | — | — | — | — | — | SWDTEN |

**Legend:** Shaded cells are not used by the Watchdog Timer.


© 2003 Microchip Technology Inc.

## 23.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO bit in Configuration Register 1H (CONFIG1H<7>).

Two-Speed Start-up is available only if the primary oscillator mode is LP, XT, HS or HSPLL (Crystal-based modes). Other sources do not require a OST start-up delay; for these, Two-Speed Start-up is disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer after a POR Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

Because the OSCCON register is cleared on Reset events, the INTOSC (or postscaler) clock source is not initially available after a Reset event; the INTRC clock is used directly at its base frequency. To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits IFRC2:IFRC0 immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting IFRC2:IFRC0 prior to entering Sleep mode.

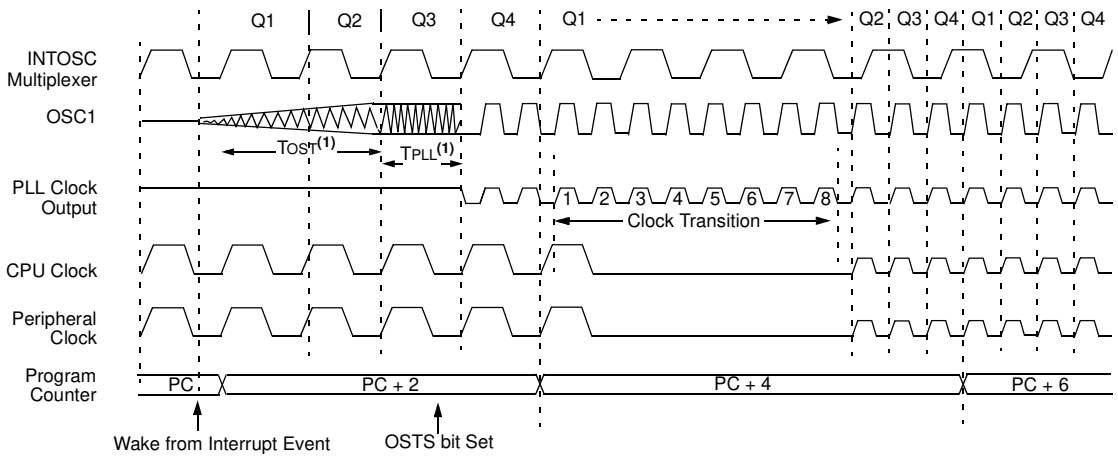
In all other power managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

### 23.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power managed modes, including serial SLEEP instructions (refer to **Section 3.1.3 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS1:SCS0 bit settings and issue SLEEP commands before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the system clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the system clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

**FIGURE 23-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)**



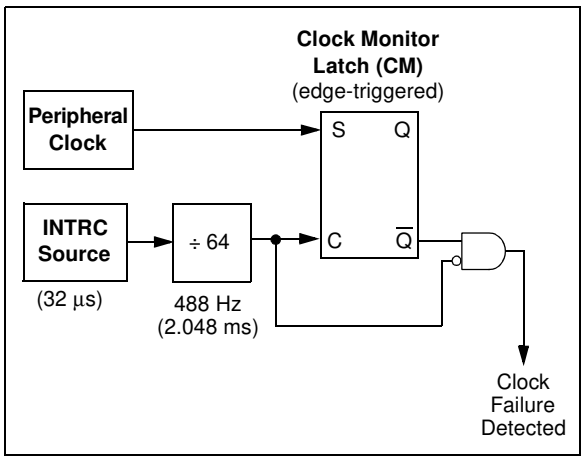

**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

## 23.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation, in the event of an external oscillator failure, by automatically switching the system clock to the internal oscillator block. The FSCM function is enabled by setting the Fail-Safe Clock Monitor Enable bit, FCMEN (CONFIG1H<6>).

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide an instant backup clock in the event of a clock failure. Clock monitoring (shown in Figure 23-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral system clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the system clock source but cleared on the rising edge of the sample clock.

**FIGURE 23-3: FSCM BLOCK DIAGRAM**




Clock failure is tested on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 23-4). This causes the following:

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

Since the postscaler frequency from the internal oscillator block may not be sufficiently stable, it may be desirable to select another clock configuration and enter an alternate power managed mode (see **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"** and **Section 3.1.3 "Multiple Sleep Commands"** for more details). This can be done to attempt a partial recovery or execute a controlled shutdown.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits IFRC2:IFRC0 immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting IFRC2:IFRC0 prior to entering Sleep mode.

Adjustments to the internal oscillator block using the OSCTUNE register also affect the period of the FSCM by the same factor. This can usually be neglected, as the clock frequency being monitored is generally much higher than the sample clock frequency.

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

### 23.4.1 FSCM AND THE WATCHDOG TIMER

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

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF2:IRCF0 bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, fail-safe clock events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.


© 2003 Microchip Technology Inc.

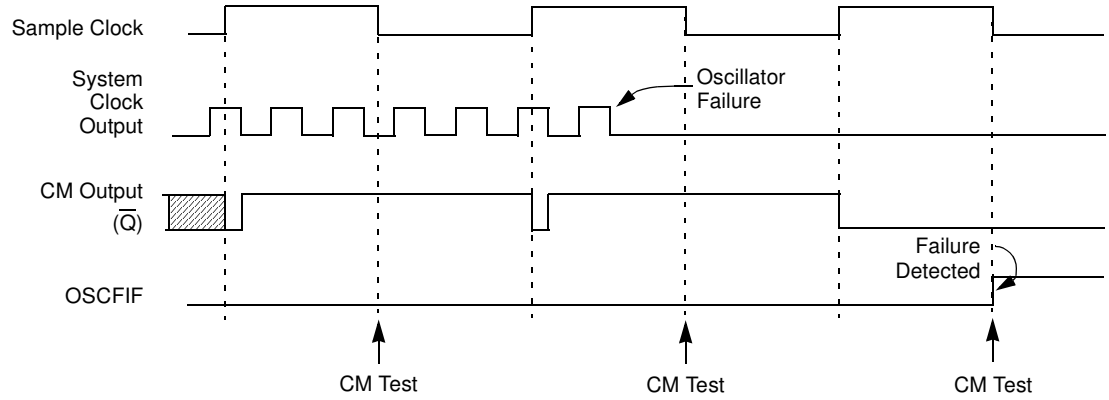
### 23.4.2 EXITING FAIL-SAFE OPERATION

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

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

Entering a power managed mode by loading the OSCCON register and executing a `SLEEP` instruction will clear the fail-safe condition. When the fail-safe condition is cleared, the clock monitor will resume monitoring the peripheral clock.

**FIGURE 23-4: FSCM TIMING DIAGRAM**




**Note:** The system clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

### 23.4.3 FSCM INTERRUPTS IN POWER MANAGED MODES

As previously mentioned, entering a power managed mode clears the fail-safe condition. By entering a power managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. Fail-safe monitoring of the power managed clock source resumes in the power managed mode.

If an oscillator failure occurs during power managed operation, the subsequent events depend on whether or not the oscillator failure interrupt is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTOSC multiplexer. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, the device will not exit the power managed mode on oscillator failure. Instead, the device will continue to operate as before but clocked by the INTOSC multiplexer. While in Idle mode, subsequent interrupts will cause the CPU to begin executing instructions while being clocked by the INTOSC multiplexer. The device will not transition to a different clock source until the fail-safe condition is cleared.

### 23.4.4 POR OR WAKE FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or Low-Power Sleep mode. When the primary system clock is EC, RC or INTRC modes, monitoring can begin immediately following these events.

For oscillator modes involving a crystal or resonator (HS, HSPLL, LP or XT), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FCSM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the system clock and functions until the primary clock is stable (the OST and PLL timers have timed out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

| | |
|---|---|
| **Note:** | The same logic that prevents false oscillator failure interrupts on POR or wake from Sleep will also prevent the detection of the oscillator's failure to start at all following these events. This can be avoided by monitoring the OSTS bit and using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged. |

As noted in **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration and enter an alternate power managed mode while waiting for the primary system clock to become stable. When the new powered managed mode is selected, the primary clock is disabled.


© 2003 Microchip Technology Inc.

## 23.5 Program Verification and Code Protection

The overall structure of the code protection on the PIC18 Flash devices differs significantly from other PICmicro® devices.

The user program memory is divided into five blocks. One of these is a boot block of 512 bytes. The remainder of the memory is divided into four blocks on binary boundaries.

Each of the five blocks has three code protection bits associated with them. They are:

- Code-Protect bit (CPn)
- Write-Protect bit (WRTn)
- External Block Table Read bit (EBTRn)

Figure 23-5 shows the program memory organization for 4 and 8-Kbyte devices and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 23-3.

**FIGURE 23-5: CODE-PROTECTED PROGRAM MEMORY FOR PIC18F2X20/4X20**

| MEMORY SIZE/DEVICE | | | |
|---|---|---|---|
| **4 Kbytes (PIC18F2220/4220)** | **8 Kbytes (PIC18F2320/4320)** | **Address Range** | **Block Code Protection Controlled By:** |
| Boot Block | Boot Block | 000000h<br>0001FFh | CPB, WRTB, EBTRB |
| Block 0 | Block 0 | 000200h<br>0007FFh | CP0, WRT0, EBTR0 |
| Block 1 | Block 1 | 000800h<br>000FFFh | CP1, WRT1, EBTR1 |
| Unimplemented Read '0's | Block 2 | 001000h<br>0017FFh | CP2, WRT2, EBTR2 |
| Unimplemented Read '0's | Block 3 | 001800h<br>001FFFh | CP3, WRT3, EBTR3 |
| Unimplemented Read '0's | Unimplemented Read '0's | 002000h<br>1FFFFFh | (Unimplemented Memory Space) |

**TABLE 23-3: SUMMARY OF CODE PROTECTION REGISTERS**

| File Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| 300008h | CONFIG5L | — | — | — | — | CP3 | CP2 | CP1 | CP0 |
| 300009h | CONFIG5H | CPD | CPB | — | — | — | — | — | — |
| 30000Ah | CONFIG6L | — | — | — | — | WRT3 | WRT2 | WRT1 | WRT0 |
| 30000Bh | CONFIG6H | WRTD | WRTB | WRTC | — | — | — | — | — |
| 30000Ch | CONFIG7L | — | — | — | — | EBTR3 | EBTR2 | EBTR1 | EBTR0 |
| 30000Dh | CONFIG7H | — | EBTRB | — | — | — | — | — | — |

**Legend:** Shaded cells are unimplemented.

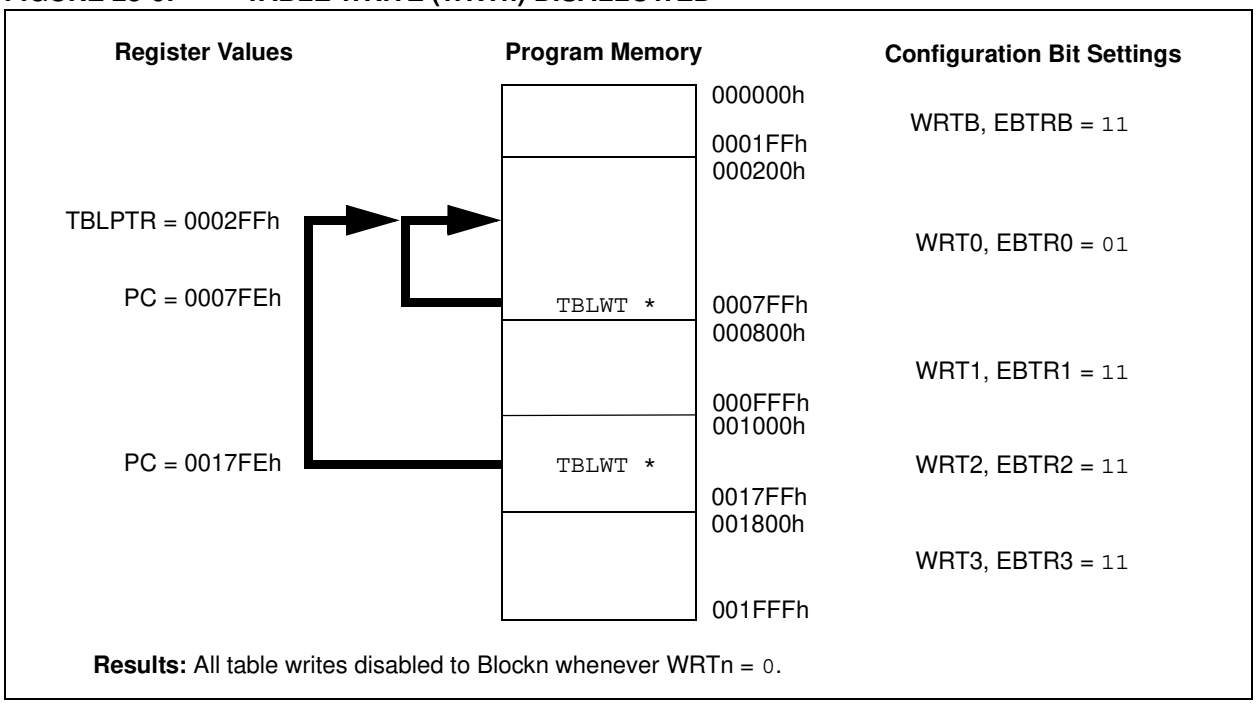
### 23.5.1 PROGRAM MEMORY CODE PROTECTION

The program memory may be read to or written from any location using the table read and table write instructions. The device ID may be read with table reads. The configuration registers may be read and written with the table read and table write instructions.

In normal execution mode, the CPn bits have no direct effect. CPn bits inhibit external reads and writes. A block of user memory may be protected from table writes if the WRTn configuration bit is '0'. The EBTRn bits control table reads. For a block of user memory with the EBTRn bit set to '0', a table read instruction that executes from within that block is allowed to read. A table read instruction that executes from a location outside of that block is not allowed to read and will result in reading '0's. Figures 23-6 through 23-8 illustrate table write and table read protection.

| **Note:** | Code protection bits may only be written to a '0' from a '1' state. It is not possible to write a '1' to a bit in the '0' state. Code protection bits are only set to '1' by a full chip erase or block erase function. The full chip erase and block erase functions can only be initiated via ICSP or an external programmer. |
|---|---|

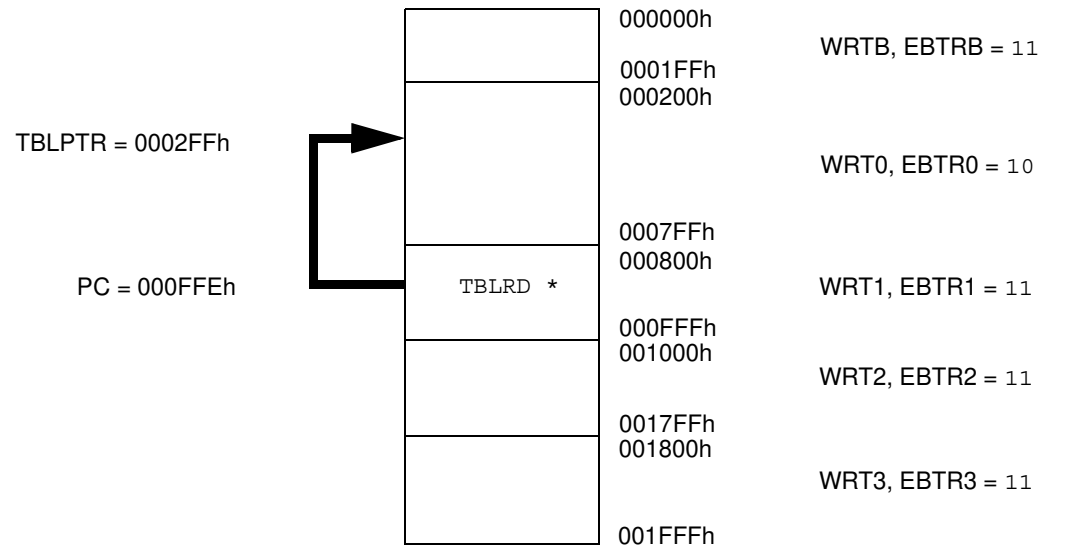
**FIGURE 23-6: TABLE WRITE (WRTn) DISALLOWED**




**Results:** All table writes disabled to Blockn whenever WRTn = 0.

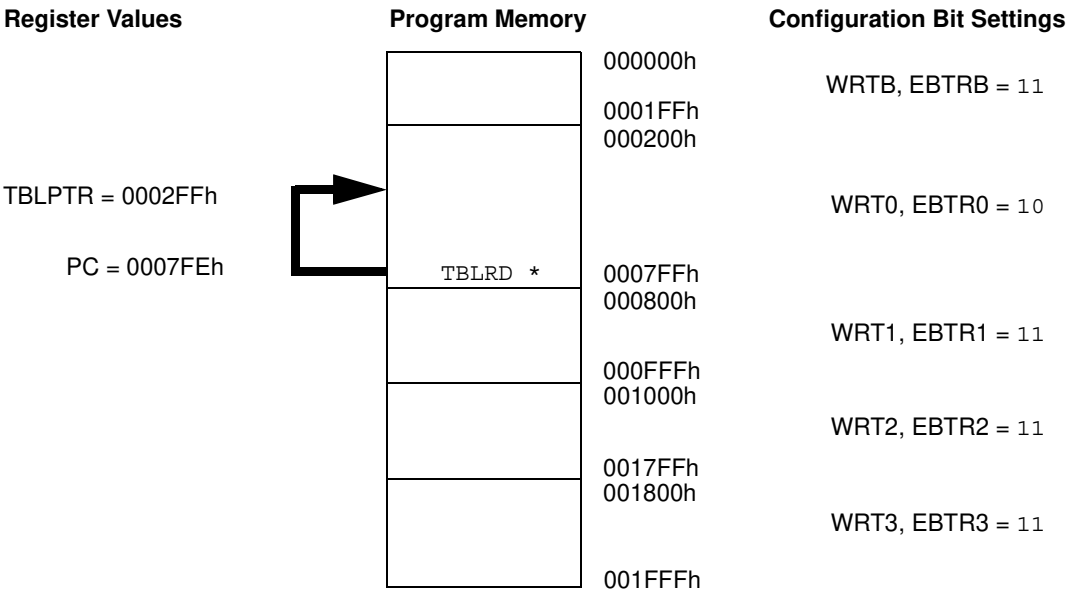

© 2003 Microchip Technology Inc.

**FIGURE 23-7: EXTERNAL BLOCK TABLE READ (EBTRn) DISALLOWED**




**Results:** All table reads from external blocks to Blockn are disabled whenever EBTRn = 0.
TABLAT register returns a value of '0'.

**FIGURE 23-8: EXTERNAL BLOCK TABLE READ (EBTRn) ALLOWED**




**Results:** Table reads permitted within Blockn, even when EBTRBn = 0.
TABLAT register returns the value of the data at the location TBLPTR.

### 23.5.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

### 23.5.3 CONFIGURATION REGISTER PROTECTION

The configuration registers can be write-protected. The WRTC bit controls protection of the configuration registers. In normal execution mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

## 23.6 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are both readable and writable during normal execution through the TBLRD and TBLWT instructions, or during program/verify. The ID locations can be read when the device is code-protected.

## 23.7 In-Circuit Serial Programming

PIC18F2X20/4X20 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. 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 (see Table 23-5).

## 23.8 In-Circuit Debugger

When the DEBUG bit in configuration register, CONFIG4L, is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 23-4 shows which resources are required by the background debugger.

**TABLE 23-4: DEBUGGER RESOURCES**

| | |
|---|---|
| I/O pins: | RB6, RB7 |
| Stack: | 2 levels |
| Program Memory: | 512 bytes |
| Data Memory: | 10 bytes |

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{MCLR}$/VPP, VDD, VSS, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip or one of the third party development tool companies.

## 23.9 Low-Voltage ICSP Programming

The LVP bit in Configuration Register 4L (CONFIG4L<2>) enables Low-Voltage ICSP Programming (LVP). When LVP is enabled, the microcontroller can be programmed without requiring high voltage being applied to the $\overline{MCLR}$/VPP pin, but the RB5/PGM pin is then dedicated to controlling Program mode entry and is not available as a general purpose I/O pin.

LVP is enabled in erased devices.

While programming using LVP, VDD is applied to the $\overline{MCLR}$/VPP pin as in normal execution mode. To enter Programming mode, VDD is applied to the PGM pin.

**Note 1:** High-voltage programming is always available, regardless of the state of the LVP bit or the PGM pin, by applying VIHH to the $\overline{MCLR}$ pin.

**2:** When Low-Voltage Programming is enabled, the RB5 pin can no longer be used as a general purpose I/O pin.

**3:** When LVP is enabled, externally pull the PGM pin to VSS to allow normal program execution.

If Low-Voltage ICSP Programming mode will not be used, the LVP bit can be cleared and RB5/PGM becomes available as the digital I/O pin, RB5. The LVP bit may be set or cleared only when using standard high-voltage programming (VIHH applied to the $\overline{MCLR}$/VPP pin). Once LVP has been disabled, only the standard high-voltage programming is available and must be used to program the device.

Memory that is not code-protected can be erased using either a block erase, or erased row by row, then written at any specified VDD. If code-protected memory is to be erased, a block erase is required. If a block erase is to be performed when using Low-Voltage Programming, the device must be supplied with VDD of 4.5V to 5.5V.

**TABLE 23-5: ICSP/ICD CONNECTIONS**

| Signal | Pin | Notes |
|---|---|---|
| PGD | RB7 | May require isolation from application circuits |
| PGC | RB6 | |
| $\overline{MCLR}$ | $\overline{MCLR}$ | |
| VDD | VDD | |
| VSS | VSS | |
| PGM | RB5 | Pull RB5 low if LVP is enabled |


© 2003 Microchip Technology Inc.

## 24.0 INSTRUCTION SET SUMMARY

The PIC18 instruction set adds many enhancements to the previous PICmicro instruction sets, while maintaining an easy migration from these PICmicro instruction sets.

Most instructions are a single program memory word (16 bits) but there are three instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 24-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 24-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The destination of the result (specified by 'd')
3. The accessed memory (specified by 'a')

The file register designator 'f' specifies which file register is to be used by the instruction.

The destination designator 'd' specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the WREG register. If 'd' is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The bit in the file register (specified by 'b')
3. The accessed memory (specified by 'a')

The bit field designator 'b' selects the number of the bit affected by the operation, while the file register designator 'f' represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by 'k')
- The desired FSR register to load the literal value into (specified by 'f')
- No operand required (specified by '—')

The **control** instructions may use some of the following operands:

- A program memory address (specified by 'n')
- The mode of the `CALL` or `RETURN` instructions (specified by 's')
- The mode of the table read and table write instructions (specified by 'm')
- No operand required (specified by '—')

All instructions are a single word except for three double word instructions. These three instructions were made double word instructions so that all the required information is available in these 32 bits. In the second word, the 4 MSbs are '1's. If this second word is executed as an instruction (by itself), it will execute as a `NOP`.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter 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`.

The double word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 µs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 µs. Two-word branch instructions (if true) would take 3 µs.

Figure 24-1 shows the general formats that the instructions can have.

All examples use the format 'nnh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

The Instruction Set Summary, shown in Table 24-2, lists the instructions recognized by the Microchip Assembler (MPASM™). **Section 24.2 "Instruction Set"** provides a description of each instruction.

### 24.1 READ-MODIFY-WRITE OPERATIONS

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified and the result is stored according to either the instruction or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a "`BCF PORTB,1`" instruction will read PORTB, clear bit 1 of the data, then write the result back to PORTB. The read operation would have the unintended result that any condition that sets the RBIF flag would be cleared. The R-M-W operation may also copy the level of an input pin to its corresponding output latch.

© 2003 Microchip Technology Inc.

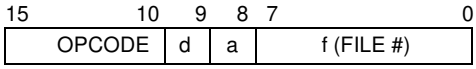
**TABLE 24-1: OPCODE FIELD DESCRIPTIONS**

| Field | Description |
|---|---|
| a | RAM access bit:<br>a = 0: RAM location in Access RAM (BSR register is ignored)<br>a = 1: RAM bank is specified by BSR register |
| bbb | Bit address within an 8-bit file register (0 to 7). |
| BSR | Bank Select Register. Used to select the current RAM bank. |
| d | Destination select bit:<br>d = 0: store result in WREG<br>d = 1: store result in file register f |
| dest | Destination either the WREG register or the specified register file location. |
| f | 8-bit register file address (0x00 to 0xFF). |
| fs | 12-bit register file address (0x000 to 0xFFF). This is the source address. |
| fd | 12-bit register file address (0x000 to 0xFFF). This is the destination address. |
| k | Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value). |
| label | Label name. |
| mm<br><br>*<br>*+<br>*-<br>+* | The mode of the TBLPTR register for the table read and table write instructions.<br>Only used with table read and table write instructions:<br>No Change to register (such as TBLPTR with table reads and writes).<br>Post-Increment register (such as TBLPTR with table reads and writes).<br>Post-Decrement register (such as TBLPTR with table reads and writes).<br>Pre-Increment register (such as TBLPTR with table reads and writes). |
| n | The relative address (2's complement number) for relative branch instructions, or the direct address for Call/Branch and Return instructions. |
| PRODH | Product of Multiply High Byte. |
| PRODL | Product of Multiply Low Byte. |
| s | Fast Call/Return mode select bit:<br>s = 0: do not update into/from shadow registers<br>s = 1: certain registers loaded into/from shadow registers (Fast mode) |
| u | Unused or Unchanged. |
| WREG | Working register (accumulator). |
| x | Don't care ('0' or '1') .<br>The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools. |
| TBLPTR | 21-bit Table Pointer (points to a Program Memory location). |
| TABLAT | 8-bit Table Latch. |
| TOS | Top-of-Stack. |
| PC | Program Counter. |
| PCL | Program Counter Low Byte. |
| PCH | Program Counter High Byte. |
| PCLATH | Program Counter High Byte Latch. |
| PCLATU | Program Counter Upper Byte Latch. |
| GIE | Global Interrupt Enable bit. |
| WDT | Watchdog Timer. |
| $\overline{TO}$ | Time-out bit. |
| $\overline{PD}$ | Power-down bit. |
| C, DC, Z, OV, N | ALU status bits Carry, Digit Carry, Zero, Overflow, Negative. |
| [ ] | Optional. |
| ( ) | Contents. |
| → | Assigned to. |
| < > | Register bit field. |
| ∈ | In the set of. |
| italics | User defined term (font is courier). |


© 2003 Microchip Technology Inc.

**FIGURE 24-1: GENERAL FORMAT FOR INSTRUCTIONS**

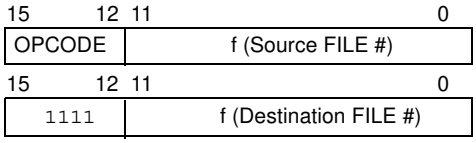
**Byte-oriented** file register operations — **Example Instruction**




ADDWF MYREG, W, B

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

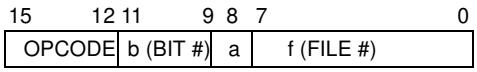
**Byte to Byte** move operations (2-word)




MOVFF MYREG1, MYREG2

f = 12-bit file register address

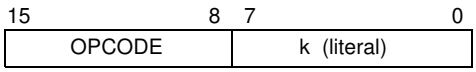
**Bit-oriented** file register operations




BSF MYREG, bit, B

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

**Literal** operations

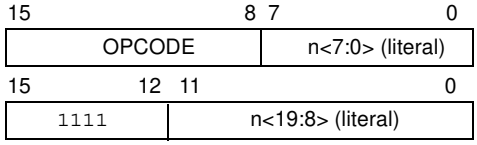



MOVLW 0x7F

k = 8-bit immediate value

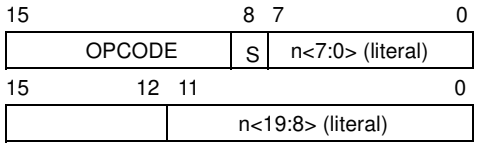
**Control** operations

**CALL, GOTO and Branch** operations



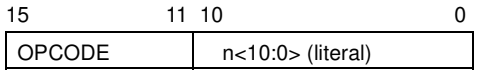

GOTO Label

n = 20-bit immediate value

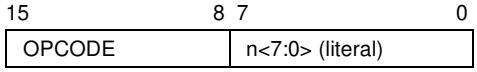



CALL MYFUNC

S = Fast bit




BRA MYFUNC




BC MYFUNC

**TABLE 24-2: PIC18FXXX INSTRUCTION SET**

| Mnemonic, Operands | | Description | Cycles | 16-Bit Instruction Word | | | | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | MSb | | | LSb | | |
| **BYTE-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| ADDWF | f, d, a | Add WREG and f | 1 | 0010 | 01da | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| ADDWFC | f, d, a | Add WREG and Carry bit to f | 1 | 0010 | 00da | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| ANDWF | f, d, a | AND WREG with f | 1 | 0001 | 01da | ffff | ffff | Z, N | 1,2 |
| CLRF | f, a | Clear f | 1 | 0110 | 101a | ffff | ffff | Z | 2 |
| COMF | f, d, a | Complement f | 1 | 0001 | 11da | ffff | ffff | Z, N | 1, 2 |
| CPFSEQ | f, a | Compare f with WREG, skip = | 1 (2 or 3) | 0110 | 001a | ffff | ffff | None | 4 |
| CPFSGT | f, a | Compare f with WREG, skip > | 1 (2 or 3) | 0110 | 010a | ffff | ffff | None | 4 |
| CPFSLT | f, a | Compare f with WREG, skip < | 1 (2 or 3) | 0110 | 000a | ffff | ffff | None | 1, 2 |
| DECF | f, d, a | Decrement f | 1 | 0000 | 01da | ffff | ffff | C, DC, Z, OV, N | 1, 2, 3, 4 |
| DECFSZ | f, d, a | Decrement f, Skip if 0 | 1 (2 or 3) | 0010 | 11da | ffff | ffff | None | 1, 2, 3, 4 |
| DCFSNZ | f, d, a | Decrement f, Skip if Not 0 | 1 (2 or 3) | 0100 | 11da | ffff | ffff | None | 1, 2 |
| INCF | f, d, a | Increment f | 1 | 0010 | 10da | ffff | ffff | C, DC, Z, OV, N | 1, 2, 3, 4 |
| INCFSZ | f, d, a | Increment f, Skip if 0 | 1 (2 or 3) | 0011 | 11da | ffff | ffff | None | 4 |
| INFSNZ | f, d, a | Increment f, Skip if Not 0 | 1 (2 or 3) | 0100 | 10da | ffff | ffff | None | 1, 2 |
| IORWF | f, d, a | Inclusive OR WREG with f | 1 | 0001 | 00da | ffff | ffff | Z, N | 1, 2 |
| MOVF | f, d, a | Move f | 1 | 0101 | 00da | ffff | ffff | Z, N | 1 |
| MOVFF | $f_s$, $f_d$ | Move $f_s$ (source) to 1st word<br>$f_d$ (destination) 2nd word | 2 | 1100<br>1111 | ffff<br>ffff | ffff<br>ffff | ffff<br>ffff | None | |
| MOVWF | f, a | Move WREG to f | 1 | 0110 | 111a | ffff | ffff | None | |
| MULWF | f, a | Multiply WREG with f | 1 | 0000 | 001a | ffff | ffff | None | |
| NEGF | f, a | Negate f | 1 | 0110 | 110a | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| RLCF | f, d, a | Rotate Left f through Carry | 1 | 0011 | 01da | ffff | ffff | C, Z, N | |
| RLNCF | f, d, a | Rotate Left f (No Carry) | 1 | 0100 | 01da | ffff | ffff | Z, N | 1, 2 |
| RRCF | f, d, a | Rotate Right f through Carry | 1 | 0011 | 00da | ffff | ffff | C, Z, N | |
| RRNCF | f, d, a | Rotate Right f (No Carry) | 1 | 0100 | 00da | ffff | ffff | Z, N | |
| SETF | f, a | Set f | 1 | 0110 | 100a | ffff | ffff | None | |
| SUBFWB | f, d, a | Subtract f from WREG with borrow | 1 | 0101 | 01da | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| SUBWF | f, d, a | Subtract WREG from f | 1 | 0101 | 11da | ffff | ffff | C, DC, Z, OV, N | |
| SUBWFB | f, d, a | Subtract WREG from f with borrow | 1 | 0101 | 10da | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| SWAPF | f, d, a | Swap nibbles in f | 1 | 0011 | 10da | ffff | ffff | None | 4 |
| TSTFSZ | f, a | Test f, skip if 0 | 1 (2 or 3) | 0110 | 011a | ffff | ffff | None | 1, 2 |
| XORWF | f, d, a | Exclusive OR WREG with f | 1 | 0001 | 10da | ffff | ffff | Z, N | |
| **BIT-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| BCF | f, b, a | Bit Clear f | 1 | 1001 | bbba | ffff | ffff | None | 1, 2 |
| BSF | f, b, a | Bit Set f | 1 | 1000 | bbba | ffff | ffff | None | 1, 2 |
| BTFSC | f, b, a | Bit Test f, Skip if Clear | 1 (2 or 3) | 1011 | bbba | ffff | ffff | None | 3, 4 |
| BTFSS | f, b, a | Bit Test f, Skip if Set | 1 (2 or 3) | 1010 | bbba | ffff | ffff | None | 3, 4 |
| BTG | f, d, a | Bit Toggle f | 1 | 0111 | bbba | ffff | ffff | None | 1, 2 |

**Note 1:** When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

**4:** Some instructions are 2-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

**5:** If the table write starts the write cycle to internal memory, the write will continue until terminated.


© 2003 Microchip Technology Inc.

**TABLE 24-2: PIC18FXXX INSTRUCTION SET (CONTINUED)**

| Mnemonic, Operands | | Description | Cycles | 16-Bit Instruction Word MSb LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|
| **CONTROL OPERATIONS** | | | | | | |
| BC | n | Branch if Carry | 1 (2) | 1110 0010 nnnn nnnn | None | |
| BN | n | Branch if Negative | 1 (2) | 1110 0110 nnnn nnnn | None | |
| BNC | n | Branch if Not Carry | 1 (2) | 1110 0011 nnnn nnnn | None | |
| BNN | n | Branch if Not Negative | 1 (2) | 1110 0111 nnnn nnnn | None | |
| BNOV | n | Branch if Not Overflow | 1 (2) | 1110 0101 nnnn nnnn | None | |
| BNZ | n | Branch if Not Zero | 1 (2) | 1110 0001 nnnn nnnn | None | |
| BOV | n | Branch if Overflow | 1 (2) | 1110 0100 nnnn nnnn | None | |
| BRA | n | Branch Unconditionally | 2 | 1101 0nnn nnnn nnnn | None | |
| BZ | n | Branch if Zero | 1 (2) | 1110 0000 nnnn nnnn | None | |
| CALL | n, s | Call subroutine 1st word | 2 | 1110 110s kkkk kkkk | None | |
| | | 2nd word | | 1111 kkkk kkkk kkkk | | |
| CLRWDT | — | Clear Watchdog Timer | 1 | 0000 0000 0000 0100 | $\overline{TO}$, $\overline{PD}$ | |
| DAW | — | Decimal Adjust WREG | 1 | 0000 0000 0000 0111 | C, DC | |
| GOTO | n | Go to address 1st word | 2 | 1110 1111 kkkk kkkk | None | |
| | | 2nd word | | 1111 kkkk kkkk kkkk | | |
| NOP | — | No Operation | 1 | 0000 0000 0000 0000 | None | |
| NOP | — | No Operation **(Note 4)** | 1 | 1111 xxxx xxxx xxxx | None | |
| POP | — | Pop top of return stack (TOS) | 1 | 0000 0000 0000 0110 | None | |
| PUSH | — | Push top of return stack (TOS) | 1 | 0000 0000 0000 0101 | None | |
| RCALL | n | Relative Call | 2 | 1101 1nnn nnnn nnnn | None | |
| RESET | | Software device Reset | 1 | 0000 0000 1111 1111 | All | |
| RETFIE | s | Return from interrupt enable | 2 | 0000 0000 0001 000s | GIE/GIEH, PEIE/GIEL | |
| RETLW | k | Return with literal in WREG | 2 | 0000 1100 kkkk kkkk | None | |
| RETURN | s | Return from Subroutine | 2 | 0000 0000 0001 001s | None | |
| SLEEP | — | Go into Standby mode | 1 | 0000 0000 0000 0011 | $\overline{TO}$, $\overline{PD}$ | |

**Note 1:** When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

**4:** Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

**5:** If the table write starts the write cycle to internal memory, the write will continue until terminated.

**TABLE 24-2: PIC18FXXX INSTRUCTION SET (CONTINUED)**

| Mnemonic, Operands | | Description | Cycles | 16-Bit Instruction Word MSb LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|
| **LITERAL OPERATIONS** | | | | | | |
| ADDLW | k | Add literal and WREG | 1 | 0000 1111 kkkk kkkk | C, DC, Z, OV, N | |
| ANDLW | k | AND literal with WREG | 1 | 0000 1011 kkkk kkkk | Z, N | |
| IORLW | k | Inclusive OR literal with WREG | 1 | 0000 1001 kkkk kkkk | Z, N | |
| LFSR | f, k | Move literal (12-bit) 2nd word to FSRx 1st word | 2 | 1110 1110 00ff kkkk<br>1111 0000 kkkk kkkk | None | |
| MOVLB | k | Move literal to BSR<3:0> | 1 | 0000 0001 0000 kkkk | None | |
| MOVLW | k | Move literal to WREG | 1 | 0000 1110 kkkk kkkk | None | |
| MULLW | k | Multiply literal with WREG | 1 | 0000 1101 kkkk kkkk | None | |
| RETLW | k | Return with literal in WREG | 2 | 0000 1100 kkkk kkkk | None | |
| SUBLW | k | Subtract WREG from literal | 1 | 0000 1000 kkkk kkkk | C, DC, Z, OV, N | |
| XORLW | k | Exclusive OR literal with WREG | 1 | 0000 1010 kkkk kkkk | Z, N | |
| **DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS** | | | | | | |
| TBLRD* | | Table Read | 2 | 0000 0000 0000 1000 | None | |
| TBLRD*+ | | Table Read with post-increment | | 0000 0000 0000 1001 | None | |
| TBLRD*- | | Table Read with post-decrement | | 0000 0000 0000 1010 | None | |
| TBLRD+* | | Table Read with pre-increment | | 0000 0000 0000 1011 | None | |
| TBLWT* | | Table Write | 2 (5) | 0000 0000 0000 1100 | None | |
| TBLWT*+ | | Table Write with post-increment | | 0000 0000 0000 1101 | None | |
| TBLWT*- | | Table Write with post-decrement | | 0000 0000 0000 1110 | None | |
| TBLWT+* | | Table Write with pre-increment | | 0000 0000 0000 1111 | None | |

**Note 1:** When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

**4:** Some instructions are 2-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

**5:** If the table write starts the write cycle to internal memory, the write will continue until terminated.


© 2003 Microchip Technology Inc.

## 24.2 Instruction Set

**ADDLW** **ADD literal to W**

Syntax: [ *label* ] ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

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

Encoding:

| 0000 | 1111 | kkkk | kkkk |
|---|---|---|---|

Description: The contents of W are added to the 8-bit literal 'k' and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ADDLW 0x15`

Before Instruction
W = 0x10
After Instruction
W = 0x25

**ADDWF** **ADD W to f**

Syntax: [ *label* ] ADDWF f [,d [,a]]

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

Operation: $(W) + (f) \rightarrow$ dest

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

Encoding:

| 0010 | 01da | ffff | ffff |
|---|---|---|---|

Description: Add W to register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR is used.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ADDWF REG, W`

Before Instruction
W = 0x17
REG = 0xC2
After Instruction
W = 0xD9
REG = 0xC2

**ADDWFC** **ADD W and Carry bit to f**

Syntax: [ *label* ] ADDWFC f [,d [,a]]

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

Operation: (W) + (f) + (C) → dest

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

Encoding:

| | | | |
|---|---|---|---|
| 0010 | 00da | ffff | ffff |

Description: Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ADDWFC REG, W`

Before Instruction
Carry bit = 1
REG = 0x02
W = 0x4D

After Instruction
Carry bit = 0
REG = 0x02
W = 0x50

**ANDLW** **AND literal with W**

Syntax: [ *label* ] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 1011 | kkkk | kkkk |

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ANDLW 0x5F`

Before Instruction
W = 0xA3

After Instruction
W = 0x03


© 2003 Microchip Technology Inc.

**ANDWF** **AND W with f**

Syntax: [ *label* ] ANDWF f [,d [,a]]

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

Operation: (W) .AND. (f) → dest

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 01da | ffff | ffff |

Description: The contents of W are AND'ed with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `ANDWF REG, W`

Before Instruction
W = 0x17
REG = 0xC2

After Instruction
W = 0x02
REG = 0xC2

**BC** **Branch if Carry**

Syntax: [ *label* ] BC n

Operands: $-128 \le n \le 127$

Operation: if carry bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0010 | nnnn | nnnn |

Description: If the Carry bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BC JUMP`

Before Instruction
PC = address (HERE)

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

**BCF** **Bit Clear f**

Syntax: [ *label* ] BCF f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

| 1001 | bbba | ffff | ffff |
|---|---|---|---|

Description: Bit 'b' in register 'f' is cleared. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `BCF FLAG_REG, 7`

Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

**BN** **Branch if Negative**

Syntax: [ *label* ] BN n

Operands: $-128 \le n \le 127$

Operation: if negative bit is '1'
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

| 1110 | 0110 | nnnn | nnnn |
|---|---|---|---|

Description: If the Negative bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example:
```
HERE    BN   Jump
```

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


© 2003 Microchip Technology Inc.

**BNC** **Branch if Not Carry**

Syntax: [ *label* ] BNC n

Operands: -128 ≤ n ≤ 127

Operation: if carry bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0011 | nnnn | nnnn |

Description: If the Carry bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNC Jump`

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

**BNN** **Branch if Not Negative**

Syntax: [ *label* ] BNN n

Operands: -128 ≤ n ≤ 127

Operation: if negative bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0111 | nnnn | nnnn |

Description: If the Negative bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNN Jump`

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

**BNOV** **Branch if Not Overflow**

Syntax: [ *label* ] BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if overflow bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0101 | nnnn | nnnn |

Description: If the Overflow bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNOV Jump`

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

**BNZ** **Branch if Not Zero**

Syntax: [ *label* ] BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if zero bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0001 | nnnn | nnnn |

Description: If the Zero bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BNZ Jump`

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


© 2003 Microchip Technology Inc.

**BRA** **Unconditional Branch**

Syntax: [ *label* ] BRA n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1101 | 0nnn | nnnn | nnnn |

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

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

Example: `HERE BRA Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
PC = address `(Jump)`

**BSF** **Bit Set f**

Syntax: [ *label* ] BSF f,b[,a]

Operands: 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

Operation: 1 → f<b>

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1000 | bbba | ffff | ffff |

Description: Bit 'b' in register 'f' is set. If 'a' is '0', Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `BSF FLAG_REG, 7`

Before Instruction
FLAG_REG = 0x0A
After Instruction
FLAG_REG = 0x8A

© 2003 Microchip Technology Inc.

**BTFSC** **Bit Test File, Skip if Clear**

Syntax: [ *label* ] BTFSC f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

| 1011 | bbba | ffff | ffff |
|---|---|---|---|

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped.
If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSC   FLAG, 1
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

**BTFSS** **Bit Test File, Skip if Set**

Syntax: [ *label* ] BTFSS f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

| 1010 | bbba | ffff | ffff |
|---|---|---|---|

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped.
If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSS   FLAG, 1
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)


© 2003 Microchip Technology Inc.

**BTG** **Bit Toggle f**

Syntax: [ *label* ] BTG f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0111 | bbba | ffff | ffff |

Description: Bit 'b' in data memory location 'f' is inverted. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `BTG PORTC, 4`

Before Instruction:
PORTC = 0111 0101 [0x75]
After Instruction:
PORTC = 0110 0101 [0x65]

**BOV** **Branch if Overflow**

Syntax: [ *label* ] BOV n

Operands: $-128 \le n \le 127$

Operation: if overflow bit is '1'
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0100 | nnnn | nnnn |

Description: If the Overflow bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example:
```
HERE    BOV  JUMP
```
Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 1;
PC = address (JUMP)
If Overflow = 0;
PC = address (HERE+2)

**BZ** **Branch if Zero**

Syntax: [ *label* ] BZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 | 0000 | nnnn | nnnn |

Description: If the Zero bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

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

If No Jump:

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

Example: `HERE BZ Jump`

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

**CALL** **Subroutine Call**

Syntax: [ *label* ] CALL k [,s]

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>,
if s = 1
(W) → WS,
(STATUS) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

| | | | | |
|---|---|---|---|---|
| 1st word (k<7:0>) | 1110 | 110s | $k_7$kkk | kkkk$_0$ |
| 2nd word(k<19:8>) | 1111 | $k_{19}$kkk | kkkk | kkkk$_8$ |

Description: Subroutine call of entire 2 Mbyte memory range. First, return address (PC+ 4) is pushed onto the return stack. If 's' = 1, the W, Status and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

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

Example: `HERE CALL THERE,FAST`

Before Instruction
PC = address (HERE)
After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS= STATUS


© 2003 Microchip Technology Inc.

**CLRF** **Clear f**

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

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

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

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

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

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: CLRF FLAG_REG

Before Instruction
FLAG_REG = 0x5A
After Instruction
FLAG_REG = 0x00

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

Syntax: [ *label* ] CLRWDT

Operands: None

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

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

Encoding:

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

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

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: CLRWDT

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 0x00
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

**COMF** **Complement f**

Syntax: [ *label* ] COMF f [,d [,a]]

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

Operation: $(\bar{f}) \rightarrow$ dest

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 11da | ffff | ffff |

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: COMF REG, W

Before Instruction
REG = 0x13
After Instruction
REG = 0x13
W = 0xEC

**CPFSEQ** **Compare f with W, skip if f = W**

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

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

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 001a | ffff | ffff |

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    CPFSEQ REG
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)


© 2003 Microchip Technology Inc.

**CPFSGT** **Compare f with W, skip if f > W**

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

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

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

| 0110 | 010a | ffff | ffff |
|---|---|---|---|

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.
If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE      CPFSGT REG
NGREATER  :
GREATER   :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG > W;
PC = Address (GREATER)
If REG ≤ W;
PC = Address (NGREATER)

**CPFSLT** **Compare f with W, skip if f < W**

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

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

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

| 0110 | 000a | ffff | ffff |
|---|---|---|---|

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    CPFSLT REG
NLESS   :
LESS    :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

**DAW** **Decimal Adjust W Register**

Syntax: [ *label* ] DAW

Operands: None

Operation: If [W<3:0> >9] or [DC = 1] then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>;

If [W<7:4> >9] or [C = 1] then
(W<7:4>) + 6 → W<7:4>;
else
(W<7:4>) → W<7:4>;

Status Affected: C, DC

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0111 |

Description: DAW adjusts the eight-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result. The carry bit may be set by DAW regardless of its setting prior to the DAW execution.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register W | Process Data | Write W |

Example1: DAW

Before Instruction
W = 0xA5
C = 0
DC = 0

After Instruction
W = 0x05
C = 1
DC = 0

Example 2:

Before Instruction
W = 0xCE
C = 0
DC = 0

After Instruction
W = 0x34
C = 1
DC = 0

**DECF** **Decrement f**

Syntax: [ *label* ] DECF f [,d [,a]]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – 1 → dest

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

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 01da | ffff | ffff |

Description: Decrement register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: DECF CNT,

Before Instruction
CNT = 0x01
Z = 0

After Instruction
CNT = 0x00
Z = 1


© 2003 Microchip Technology Inc.

**DECFSZ** **Decrement f, skip if 0**

Syntax: [ *label* ] DECFSZ f [,d [,a]]

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

Operation: (f) – 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0010 | 11da | ffff | ffff |

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    DECFSZ  CNT
        GOTO    LOOP
CONTINUE
```

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT - 1
If CNT = 0;
PC = Address (CONTINUE)
If CNT ≠ 0;
PC = Address (HERE+2)

**DCFSNZ** **Decrement f, skip if not 0**

Syntax: [ *label* ] DCFSNZ f [,d [,a]]

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

Operation: (f) – 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0100 | 11da | ffff | ffff |

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    DCFSNZ  TEMP
ZERO    :
NZERO   :
```

Before Instruction
TEMP = ?

After Instruction
TEMP = TEMP - 1,
If TEMP = 0;
PC = Address (ZERO)
If TEMP ≠ 0;
PC = Address (NZERO)

**GOTO** **Unconditional Branch**

Syntax: [ *label* ] GOTO k

Operands: $0 \le k \le 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

| | | | | |
|---|---|---|---|---|
| 1st word (k<7:0>) | 1110 | 1111 | $k_7kkk$ | $kkkk_0$ |
| 2nd word(k<19:8>) | 1111 | $k_{19}kkk$ | kkkk | $kkkk_8$ |

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

Words: 2

Cycles: 2

Q Cycle Activity:

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

Example: GOTO THERE

After Instruction
PC = Address (THERE)

**INCF** **Increment f**

Syntax: [ *label* ] INCF f [,d [,a]]

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

Operation: (f) + 1 → dest

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

Encoding:

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

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: INCF CNT,

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

After Instruction
CNT = 0x00
Z = 1
C = 1
DC = 1


© 2003 Microchip Technology Inc.

**INCFSZ** **Increment f, skip if 0**

Syntax: [ *label* ] INCFSZ f [,d [,a]]

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

Operation: (f) + 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0011 | 11da | ffff | ffff |

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    INCFSZ   CNT
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT ≠ 0;
PC = Address (NZERO)

**INFSNZ** **Increment f, skip if not 0**

Syntax: [ *label* ] INFSNZ f [,d [,a]]

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

Operation: (f) + 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0100 | 10da | ffff | ffff |

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    INFSNZ   REG
ZERO
NZERO
```

Before Instruction
PC = Address (HERE)

After Instruction
REG = REG + 1
If REG ≠ 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

**IORLW** **Inclusive OR literal with W**

Syntax: [ *label* ] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 1001 | kkkk | kkkk |

Description: The contents of W are OR'ed with the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `IORLW 0x35`

Before Instruction
W = 0x9A

After Instruction
W = 0xBF

**IORWF** **Inclusive OR W with f**

Syntax: [ *label* ] IORWF f [,d [,a]]

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

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 00da | ffff | ffff |

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `IORWF RESULT, W`

Before Instruction
RESULT = 0x13
W = 0x91

After Instruction
RESULT = 0x13
W = 0x93


© 2003 Microchip Technology Inc.

**LFSR** **Load FSR**

Syntax: [ *label* ] LFSR f,k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1110 1111 | 1110 0000 | 00ff $k_7$kkk | $k_{11}$kkk kkkk |

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' MSB | Process Data | Write literal 'k' MSB to FSRfH |
| Decode | Read literal 'k' LSB | Process Data | Write literal 'k' to FSRfL |

Example: `LFSR 2, 0x3AB`

After Instruction
FSR2H = 0x03
FSR2L = 0xAB

**MOVF** **Move f**

Syntax: [ *label* ] MOVF f [,d [,a]]

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

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0101 | 00da | ffff | ffff |

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256-byte bank. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVF REG, W`

Before Instruction
REG = 0x22
W = 0xFF

After Instruction
REG = 0x22
W = 0x22

**MOVFF** **Move f to f**

Syntax: [ *label* ] MOVFF $f_s$,$f_d$

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

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

Description: The contents of source register '$f_s$' are moved to destination register '$f_d$'. Location of source '$f_s$' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination '$f_d$' can also be anywhere from 000h to FFFh. Either source or destination can be W (a useful special situation). MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

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

The MOVFF instruction should not be used to modify interrupt settings while any interrupt is enabled (see Page 87).

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

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

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 0x33
REG2 = 0x11

After Instruction
REG1 = 0x33,
REG2 = 0x33

**MOVLB** **Move literal to low nibble in BSR**

Syntax: [ *label* ] MOVLB k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0001 | kkkk | kkkk |

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

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: MOVLB 5

Before Instruction
BSR register = 0x02

After Instruction
BSR register = 0x05


© 2003 Microchip Technology Inc.

**MOVLW** **Move literal to W**

Syntax: [ *label* ] MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 1110 | kkkk | kkkk |

Description: The eight-bit literal 'k' is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVLW 0x5A`

After Instruction

W = 0x5A

**MOVWF** **Move W to f**

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

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

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 111a | ffff | ffff |

Description: Move data from W to register 'f'. Location 'f' can be anywhere in the 256-byte bank. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVWF REG`

Before Instruction

W = 0x4F
REG = 0xFF

After Instruction

W = 0x4F
REG = 0x4F

© 2003 Microchip Technology Inc.

**MULLW** **Multiply Literal with W**

Syntax: [ *label* ] MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

| 0000 | 1101 | kkkk | kkkk |
|---|---|---|---|

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged. None of the status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process Data | Write registers PRODH: PRODL |

Example: `MULLW 0xC4`

Before Instruction

W = 0xE2
PRODH = ?
PRODL = ?

After Instruction

W = 0xE2
PRODH = 0xAD
PRODL = 0x08

**MULWF** **Multiply W with f**

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

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

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

| 0000 | 001a | ffff | ffff |
|---|---|---|---|

Description: An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged. None of the status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a'= 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | Write registers PRODH: PRODL |

Example: `MULWF REG`

Before Instruction

W = 0xC4
REG = 0xB5
PRODH = ?
PRODL = ?

After Instruction

W = 0xC4
REG = 0xB5
PRODH = 0x8A
PRODL = 0x94


© 2003 Microchip Technology Inc.

**NEGF** **Negate f**

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

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

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

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

Encoding:

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

Description: Location 'f' is negated using two's complement. The result is placed in the data memory location 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: NEGF REG, 1

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

**NOP** **No Operation**

Syntax: [ *label* ] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 1111 | 0000 xxxx | 0000 xxxx | 0000 xxxx |

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example:

None.

**POP** **Pop Top of Return Stack**

Syntax: [ *label* ] POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0110 |

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack.
This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | POP TOS value | No operation |

Example:
```
POP
GOTO    NEW
```
Before Instruction
TOS = 0x0031A2
Stack (1 level down) = 0x014332

After Instruction
TOS = 0x014332
PC = NEW

**PUSH** **Push Top of Return Stack**

Syntax: [ *label* ] PUSH

Operands: None

Operation: (PC+2) → TOS

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0101 |

Description: The PC+2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows to implement a software stack by modifying TOS, and then push it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | PUSH PC+2 onto return stack | No operation | No operation |

Example:
```
PUSH
```
Before Instruction
TOS = 0x00345A
PC = 0x000124

After Instruction
PC = 0x000126
TOS = 0x000126
Stack (1 level down) = 0x00345A


© 2003 Microchip Technology Inc.

**RCALL** **Relative Call**

Syntax: [ *label* ] RCALL n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1101 | 1nnn | nnnn | nnnn |

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC+2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n'<br>Push PC to stack | Process Data | Write to PC |
| No operation | No operation | No operation | No operation |

Example: `HERE RCALL Jump`

Before Instruction
PC = Address `(HERE)`

After Instruction
PC = Address `(Jump)`
TOS = Address `(HERE+2)`

**RESET** **Reset**

Syntax: [ *label* ] RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 1111 | 1111 |

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `RESET`

After Instruction
Registers = Reset Value
Flags* = Reset Value

**RETFIE** **Return from Interrupt**

Syntax: [ *label* ] RETFIE [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL,
if s = 1
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged.

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0001 | 000s |

Description: Return from Interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers, W, Status and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | No operation | pop PC from stack<br>Set GIEH or GIEL |
| No operation | No operation | No operation | No operation |

Example: `RETFIE 1`

After Interrupt
PC = TOS
W = WS
BSR = BSRS
STATUS = STATUSS
GIE/GIEH, PEIE/GIEL = 1

**RETLW** **Return Literal to W**

Syntax: [ *label* ] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 1100 | kkkk | kkkk |

Description: W is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process Data | pop PC from stack, Write to W |
| No operation | No operation | No operation | No operation |

Example:

```
    CALL TABLE  ; W contains table
                ; offset value
                ; W now has
                ; table value
    :
TABLE
    ADDWF PCL   ; W = offset
    RETLW k0    ; Begin table
    RETLW k1    ;
    :
    :
    RETLW kn    ; End of table
```

Before Instruction
W = 0x07
After Instruction
W = value of kn


© 2003 Microchip Technology Inc.

**RETURN** **Return from Subroutine**

Syntax: [ *label* ] RETURN [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
if s = 1
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0001 | 001s |

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If 's'= 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers, W, Status and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | Process Data | pop PC from stack |
| No operation | No operation | No operation | No operation |

Example: `RETURN`

After Interrupt
PC = TOS

**RLCF** **Rotate Left f through Carry**

Syntax: [ *label* ] RLCF f [,d [,a]]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

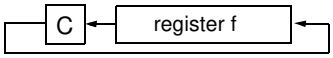
Operation: (f<n>) → dest<n+1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0011 | 01da | ffff | ffff |

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).




Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `RLCF REG, W`

Before Instruction
REG = 1110 0110
C = 0

After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

**RLNCF** **Rotate Left f (no carry)**

Syntax: [ *label* ] RLNCF f [,d [,a]]

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

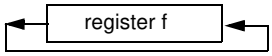
Operation: (f<n>) → dest<n+1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0100 | 01da | ffff | ffff |

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).




Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RLNCF REG

Before Instruction
REG = 1010 1011

After Instruction
REG = 0101 0111

**RRCF** **Rotate Right f through Carry**

Syntax: [ *label* ] RRCF f [,d [,a]]

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

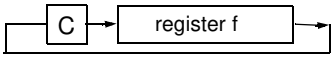
Operation: (f<n>) → dest<n-1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0011 | 00da | ffff | ffff |

Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).




Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RRCF REG, W

Before Instruction
REG = 1110 0110
C = 0

After Instruction
REG = 1110 0110
W = 0111 0011
C = 0


© 2003 Microchip Technology Inc.

**RRNCF** **Rotate Right f (no carry)**

Syntax: [ *label* ] RRNCF f [,d [,a]]

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

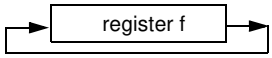
Operation: (f<n>) → dest<n-1>,
(f<0>) → dest<7>

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0100 | 00da | ffff | ffff |

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).




Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `RRNCF REG, 1, 0`

Before Instruction
REG = 1101 0111
After Instruction
REG = 1110 1011

Example 2: `RRNCF REG, W`

Before Instruction
W = ?
REG = 1101 0111
After Instruction
W = 1110 1011
REG = 1101 0111

**SETF** **Set f**

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

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

Operation: FFh → f

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 100a | ffff | ffff |

Description: The contents of the specified register are set to FFh. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `SETF REG`

Before Instruction
REG = 0x5A
After Instruction
REG = 0xFF

**SLEEP** **Enter SLEEP mode**

Syntax: [ *label* ] SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

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

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0011 |

Description: The power-down status bit ($\overline{PD}$) is cleared. The time-out status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared.
The processor is put into Sleep mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | Process Data | Go to Sleep |

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
PD = ?

After Instruction
$\overline{TO}$ = 1 †
PD = 0

† If WDT causes wake-up, this bit is cleared.

**SUBFWB** **Subtract f from W with borrow**

Syntax: [ *label* ] SUBFWB f [,d [,a]]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (W) – (f) – ($\overline{C}$) → dest

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

Encoding:

| | | | |
|---|---|---|---|
| 0101 | 01da | ffff | ffff |

Description: Subtract register 'f' and carry flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored in register 'd' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: SUBFWB REG

Before Instruction
REG = 0x03
W = 0x02
C = 0x01

After Instruction
REG = 0xFF
W = 0x02
C = 0x00
Z = 0x00
N = 0x01 ; result is negative

Example 2: SUBFWB REG, 0, 0

Before Instruction
REG = 2
W = 5
C = 1

After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3: SUBFWB REG, 1, 0

Before Instruction
REG = 1
W = 2
C = 0

After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0


© 2003 Microchip Technology Inc.

**SUBLW** **Subtract W from literal**

Syntax: [ *label* ] SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

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

Encoding:

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

Description: W is subtracted from the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `SUBLW 0x02`

Before Instruction
W = 1
C = ?

After Instruction
W = 1
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBLW 0x02`

Before Instruction
W = 2
C = ?

After Instruction
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBLW 0x02`

Before Instruction
W = 3
C = ?

After Instruction
W = FF ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

**SUBWF** **Subtract W from f**

Syntax: [ *label* ] SUBWF f [,d [,a]]

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

Operation: $(f) - (W) \rightarrow$ dest

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

Encoding:

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

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If = 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `SUBWF REG`

Before Instruction
REG = 3
W = 2
C = ?

After Instruction
REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBWF REG, W`

Before Instruction
REG = 2
W = 2
C = ?

After Instruction
REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBWF REG`

Before Instruction
REG = 0x01
W = 0x02
C = ?

After Instruction
REG = 0xFFh ;(2's complement)
W = 0x02
C = 0x00 ; result is negative
Z = 0x00
N = 0x01

© 2003 Microchip Technology Inc.

**SUBWFB** **Subtract W from f with Borrow**

Syntax: [ *label* ] SUBWFB f [,d [,a]]

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

Operation: (f) – (W) – ($\overline{C}$) → dest

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

Encoding:

| | | | |
|---|---|---|---|
| 0101 | 10da | ffff | ffff |

Description: Subtract W and the carry flag (borrow) from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1:
```
SUBWFB  REG, 1, 0
```
Before Instruction
```
REG  =  0x19    (0001 1001)
W    =  0x0D    (0000 1101)
C    =  0x01
```
After Instruction
```
REG  =  0x0C    (0000 1011)
W    =  0x0D    (0000 1101)
C    =  0x01
Z    =  0x00
N    =  0x00    ; result is positive
```

Example 2:
```
SUBWFB  REG, 0, 0
```
Before Instruction
```
REG  =  0x1B    (0001 1011)
W    =  0x1A    (0001 1010)
C    =  0x00
```
After Instruction
```
REG  =  0x1B    (0001 1011)
W    =  0x00
C    =  0x01
Z    =  0x01    ; result is zero
N    =  0x00
```

Example 3:
```
SUBWFB  REG, 1, 0
```
Before Instruction
```
REG  =  0x03    (0000 0011)
W    =  0x0E    (0000 1101)
C    =  0x01
```
After Instruction
```
REG  =  0xF5    (1111 0100)
                ; [2's comp]
W    =  0x0E    (0000 1101)
C    =  0x00
Z    =  0x00
N    =  0x01    ; result is negative
```


© 2003 Microchip Technology Inc.

**SWAPF** **Swap f**

Syntax: [ *label* ] SWAPF f [,d [,a]]

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

Operation: (f<3:0>) → dest<7:4>,
(f<7:4>) → dest<3:0>

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0011 | 10da | ffff | ffff |

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: SWAPF REG

Before Instruction
REG = 0x53
After Instruction
REG = 0x35

**TBLRD** **Table Read**

Syntax: [ *label* ] TBLRD ( *; *+; *-; +*)

Operands: None

Operation: if TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT;
TBLPTR - No Change;
if TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) +1 → TBLPTR;
if TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) -1 → TBLPTR;
if TBLRD +*,
(TBLPTR) +1 → TBLPTR;
(Prog Mem (TBLPTR)) → TABLAT;

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 10nn<br>nn=0 *<br>=1 *+<br>=2 *-<br>=3 +* |

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used. The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2 Mbyte address range.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | No operation | No operation |
| No operation | No operation (Read Program Memory) | No operation | No operation (Write TABLAT) |

**TBLRD** **Table Read (cont'd)**

Example1:

```
TBLRD  *+ ;
```

Before Instruction
TABLAT = 0x55
TBLPTR = 0x00A356
MEMORY(0x00A356) = 0x34

After Instruction
TABLAT = 0x34
TBLPTR = 0x00A357

Example2:

```
TBLRD  +* ;
```

Before Instruction
TABLAT = 0xAA
TBLPTR = 0x01A357
MEMORY(0x01A357) = 0x12
MEMORY(0x01A358) = 0x34

After Instruction
TABLAT = 0x34
TBLPTR = 0x01A358


© 2003 Microchip Technology Inc.

**TBLWT** **Table Write**

Syntax: [ *label* ] TBLWT ( *; *+; *-; +*)

Operands: None

Operation: if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR - No Change;
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) +1 → TBLPTR;
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) -1 → TBLPTR;
if TBLWT+*,
(TBLPTR) +1 → TBLPTR;
(TABLAT) → Holding Register;

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 11nn<br>nn=0 *<br>=1 *+<br>=2 *-<br>=3 +* |

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 6.0 "Flash Program Memory"** for additional details on programming Flash memory.)
The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2 MBtye address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

**TBLWT Table Write (Continued)**

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | No operation | No operation | No operation |
| No operation | No operation (Read TABLAT) | No operation | No operation (Write to Holding Register ) |

Example1: TBLWT *+;

Before Instruction
TABLAT = 0x55
TBLPTR = 0x00A356
HOLDING REGISTER (0x00A356) = 0xFF

After Instructions (table write completion)
TABLAT = 0x55
TBLPTR = 0x00A357
HOLDING REGISTER (0x00A356) = 0x55

Example 2: TBLWT +*;

Before Instruction
TABLAT = 0x34
TBLPTR = 0x01389A
HOLDING REGISTER (0x01389A) = 0xFF
HOLDING REGISTER (0x01389B) = 0xFF

After Instruction (table write completion)
TABLAT = 0x34
TBLPTR = 0x01389B
HOLDING REGISTER (0x01389A) = 0xFF
HOLDING REGISTER (0x01389B) = 0x34

**TSTFSZ** **Test f, skip if 0**

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

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

Operation: skip if f = 0

Status Affected: None

Encoding:

| 0110 | 011a | ffff | ffff |
|---|---|---|---|

Description: If 'f' = 0, the next instruction, fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    TSTFSZ  CNT
NZERO     :
ZERO    :
```

Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 0x00,
PC = Address (ZERO)
If CNT ≠ 0x00,
PC = Address (NZERO)

**XORLW** **Exclusive OR literal with W**

Syntax: [ *label* ] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

| 0000 | 1010 | kkkk | kkkk |
|---|---|---|---|

Description: The contents of W are XOR'ed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: XORLW 0xAF

Before Instruction
W = 0xB5
After Instruction
W = 0x1A


© 2003 Microchip Technology Inc.

**XORWF** **Exclusive OR W with f**

Syntax: [ *label* ] XORWF f [,d [,a]]

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

Operation: (W) .XOR. (f) → dest

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 10da | ffff | ffff |

Description: Exclusive OR the contents of W with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: XORWF REG

Before Instruction
REG = 0xAF
W = 0xB5

After Instruction
REG = 0x1A
W = 0xB5

**NOTES:**


© 2003 Microchip Technology Inc.

## 25.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C17 and MPLAB C18 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB C30 C Compiler
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
  - MPLAB dsPIC30 Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PRO MATE® II Universal Device Programmer
  - PICSTART® Plus Development Programmer
- Low-Cost Demonstration Boards
  - PICDEM™ 1 Demonstration Board
  - PICDEM.net™ Demonstration Board
  - PICDEM 2 Plus Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 4 Demonstration Board
  - PICDEM 17 Demonstration Board
  - PICDEM 18R Demonstration Board
  - PICDEM LIN Demonstration Board
  - PICDEM USB Demonstration Board
- Evaluation Kits
  - KEELOQ®
  - PICDEM MSC
  - microID®
  - CAN
  - PowerSmart®
  - Analog

### 25.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® based application that contains:

- An interface to debugging tools
  - simulator
  - programmer (sold separately)
  - emulator (sold separately)
  - in-circuit debugger (sold separately)
- A full-featured editor with color coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Extensive on-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
  - source files (assembly or C)
  - absolute listing file (mixed assembly and C)
  - machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increasing flexibility and power.

### 25.2 MPASM Assembler

The MPASM assembler is a full-featured, universal macro assembler for all PICmicro 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 IDE projects
- User defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

## 25.3 MPLAB C17 and MPLAB C18 C Compilers

The MPLAB C17 and MPLAB C18 Code Development Systems are complete ANSI C compilers for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

## 25.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can 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

## 25.5 MPLAB C30 C Compiler

The MPLAB C30 C compiler is a full-featured, ANSI compliant, optimizing compiler that translates standard ANSI C programs into dsPIC30F assembly language source. The compiler also supports many command-line options and language extensions to take full advantage of the dsPIC30F device hardware capabilities and afford fine control of the compiler code generator.

MPLAB C30 is distributed with a complete ANSI C standard library. All library functions have been validated and conform to the ANSI C library standard. The library includes functions for string manipulation, dynamic memory allocation, data conversion, time-keeping and math functions (trigonometric, exponential and hyperbolic). The compiler provides symbolic information for high-level source debugging with the MPLAB IDE.

## 25.6 MPLAB ASM30 Assembler, Linker and Librarian

MPLAB ASM30 assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 compiler uses the assembler to produce it's 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 dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

## 25.7 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC hosted environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any pin. The execution can be performed in Single-Step, Execute Until Break or Trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and MPLAB C18 C Compilers, as well as the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent, economical software development tool.

## 25.8 MPLAB SIM30 Software Simulator

The MPLAB SIM30 software simulator allows code development in a PC hosted environment by simulating the dsPIC30F series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any of the pins.

The MPLAB SIM30 simulator fully supports symbolic debugging using the MPLAB C30 C Compiler and MPLAB ASM30 assembler. The simulator runs in either a Command Line mode for automated tasks, or from MPLAB IDE. This high-speed simulator is designed to debug, analyze and optimize time intensive DSP routines.


© 2003 Microchip Technology Inc.

## 25.9 MPLAB ICE 2000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 2000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers. Software control of the MPLAB ICE 2000 in-circuit emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE 2000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

## 25.10 MPLAB ICE 4000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 4000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PICmicro microcontrollers. Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICD 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high-speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, up to 2 Mb of emulation memory and the ability to view variables in real-time.

The MPLAB ICE 4000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

## 25.11 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low-cost, run-time development tool, connecting to the host PC via an RS-232 or high-speed USB interface. This tool is based on the Flash PICmicro MCUs and can be used to develop for these and other PICmicro microcontrollers. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the Flash devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost effective in-circuit Flash debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single-stepping and watching variables, CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real-time. MPLAB ICD 2 also serves as a development programmer for selected PICmicro devices.

## 25.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at $V_{DDMIN}$ and $V_{DDMAX}$ for maximum reliability. It features an LCD display for instructions and error messages and a modular detachable socket assembly to support various package types. In Stand-Alone mode, the PRO MATE II device programmer can read, verify and program PICmicro devices without a PC connection. It can also set code protection in this mode.

## 25.13 PICSTART Plus Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low-cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus development programmer supports most PICmicro devices up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

© 2003 Microchip Technology Inc.

### 25.14 PICDEM 1 PICmicro Demonstration Board

The PICDEM 1 demonstration board demonstrates the capabilities of the PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The sample microcontrollers provided with the PICDEM 1 demonstration board can be programmed with a PRO MATE II device programmer or a PICSTART Plus development programmer. The PICDEM 1 demonstration board can be connected to the MPLAB ICE in-circuit emulator for testing. A prototype area extends the circuitry for additional application components. Features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs.

### 25.15 PICDEM.net Internet/Ethernet Demonstration Board

The PICDEM.net demonstration board is an Internet/ Ethernet demonstration board using the PIC18F452 microcontroller and TCP/IP firmware. The board supports any 40-pin DIP device that conforms to the standard pinout used by the PIC16F877 or PIC18C452. This kit features a user friendly TCP/IP stack, web server with HTML, a 24L256 Serial EEPROM for Xmodem download to web pages into Serial EEPROM, ICSP/MPLAB ICD 2 interface connector, an Ethernet interface, RS-232 interface and a 16 x 2 LCD display. Also included is the book and CD-ROM *"TCP/IP Lean, Web Servers for Embedded Systems,"* by Jeremy Bentham

### 25.16 PICDEM 2 Plus Demonstration Board

The PICDEM 2 Plus demonstration board supports many 18, 28 and 40-pin microcontrollers, including PIC16F87X and PIC18FXX2 devices. All the necessary hardware and software is included to run the demonstration programs. The sample microcontrollers provided with the PICDEM 2 demonstration board can be programmed with a PRO MATE II device programmer, PICSTART Plus development programmer, or MPLAB ICD 2 with a Universal Programmer Adapter. The MPLAB ICD 2 and MPLAB ICE in-circuit emulators may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area extends the circuitry for additional application components. Some of the features include an RS-232 interface, a 2 x 16 LCD display, a piezo speaker, an on-board temperature sensor, four LEDs and sample PIC18F452 and PIC16F877 Flash microcontrollers.

### 25.17 PICDEM 3 PIC16C92X Demonstration Board

The PICDEM 3 demonstration board supports the PIC16C923 and PIC16C924 in the PLCC package. All the necessary hardware and software is included to run the demonstration programs.

### 25.18 PICDEM 4 8/14/18-Pin Demonstration Board

The PICDEM 4 can be used to demonstrate the capabilities of the 8, 14 and 18-pin PIC16XXXX and PIC18XXXX MCUs, including the PIC16F818/819, PIC16F87/88, PIC16F62XA and the PIC18F1320 family of microcontrollers. PICDEM 4 is intended to showcase the many features of these low pin count parts, including LIN and Motor Control using ECCP. Special provisions are made for low-power operation with the supercapacitor circuit and jumpers allow on-board hardware to be disabled to eliminate current draw in this mode. Included on the demo board are provisions for Crystal, RC or Canned Oscillator modes, a five volt regulator for use with a nine volt wall adapter or battery, DB-9 RS-232 interface, ICD connector for programming via ICSP and development with MPLAB ICD 2, 2x16 liquid crystal display, PCB footprints for H-Bridge motor driver, LIN transceiver and EEPROM. Also included are: header for expansion, eight LEDs, four potentiometers, three push buttons and a prototyping area. Included with the kit is a PIC16F627A and a PIC18F1320. Tutorial firmware is included along with the User's Guide.

### 25.19 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. A programmed sample is included. The PRO MATE II device programmer, or the PICSTART Plus development programmer, can be used to reprogram the device for user tailored application development. The PICDEM 17 demonstration board supports program download and execution from external on-board Flash memory. A generous prototype area is available for user hardware expansion.


© 2003 Microchip Technology Inc.

## 25.20 PICDEM 18R PIC18C601/801 Demonstration Board

The PICDEM 18R demonstration board serves to assist development of the PIC18C601/801 family of Microchip microcontrollers. It provides hardware implementation of both 8-bit Multiplexed/Demultiplexed and 16-bit Memory modes. The board includes 2 Mb external Flash memory and 128 Kb SRAM memory, as well as serial EEPROM, allowing access to the wide range of memory types supported by the PIC18C601/801.

## 25.21 PICDEM LIN PIC16C43X Demonstration Board

The powerful LIN hardware and software kit includes a series of boards and three PICmicro microcontrollers. The small footprint PIC16C432 and PIC16C433 are used as slaves in the LIN communication and feature on-board LIN transceivers. A PIC16F874 Flash microcontroller serves as the master. All three microcontrollers are programmed with firmware to provide LIN bus communication.

## 25.22 PICkit™ 1 Flash Starter Kit

A complete “development system in a box”, the PICkit Flash Starter Kit includes a convenient multi-section board for programming, evaluation and development of 8/14-pin Flash PIC® microcontrollers. Powered via USB, the board operates under a simple Windows GUI. The PICkit 1 Starter Kit includes the user's guide (on CD ROM), PICkit 1 tutorial software and code for various applications. Also included are MPLAB® IDE (Integrated Development Environment) software, software and hardware “Tips 'n Tricks for 8-pin Flash PIC® Microcontrollers” Handbook and a USB Interface Cable. Supports all current 8/14-pin Flash PIC microcontrollers, as well as many future planned devices.

## 25.23 PICDEM USB PIC16C7X5 Demonstration Board

The PICDEM USB Demonstration Board shows off the capabilities of the PIC16C745 and PIC16C765 USB microcontrollers. This board provides the basis for future USB products.

## 25.24 Evaluation and Programming Tools

In addition to the PICDEM series of circuits, Microchip has a line of evaluation kits and demonstration software for these products.

- KEELOQ evaluation and programming tools for Microchip’s HCS Secure Data Products
- CAN developers kit for automotive network applications
- Analog design boards and filter design software
- PowerSmart battery charging evaluation/ calibration kits
- IrDA® development kit
- microID development and rfLab™ development software
- SEEVAL® designer kit for memory evaluation and endurance calculations
- PICDEM MSC demo boards for Switching mode power supply, high-power IR driver, delta sigma ADC and flow rate sensor

Check the Microchip web page and the latest Product Line Card for the complete list of demonstration and evaluation kits.

**NOTES:**


© 2003 Microchip Technology Inc.

## 26.0 ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings (†)

Ambient temperature under bias ..... -55°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on any pin with respect to $V_{SS}$ (except $V_{DD}$, $\overline{MCLR}$ and RA4) ..... -0.3V to ($V_{DD}$ + 0.3V)
Voltage on $V_{DD}$ with respect to $V_{SS}$ ..... -0.3V to +7.5V
Voltage on $\overline{MCLR}$ with respect to $V_{SS}$ **(Note 2)** ..... 0V to +13.25V
Voltage on RA4 with respect to $V_{SS}$ ..... 0V to +8.5V
Total power dissipation **(Note 1)** ..... 1.0W
Maximum current out of $V_{SS}$ pin ..... 300 mA
Maximum current into $V_{DD}$ pin ..... 250 mA
Input clamp current, $I_{IK}$ ($V_I$ < 0 or $V_I$ > $V_{DD}$) ..... ±20 mA
Output clamp current, $I_{OK}$ ($V_O$ < 0 or $V_O$ > $V_{DD}$) ..... ±20 mA
Maximum output current sunk by any I/O pin ..... 25 mA
Maximum output current sourced by any I/O pin ..... 25 mA
Maximum current sunk by all ports ..... 200 mA
Maximum current sourced by all ports ..... 200 mA

**Note 1:** Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD}-V_{OH}) \times I_{OH}\} + \sum(V_{ol} \times I_{OL})$

**2:** Voltage spikes below $V_{SS}$ at the $\overline{MCLR}$/$V_{PP}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$/$V_{PP}$ pin, rather than pulling this pin directly to $V_{SS}$.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

**FIGURE 26-1: PIC18F2220/2320/4220/4320 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)**

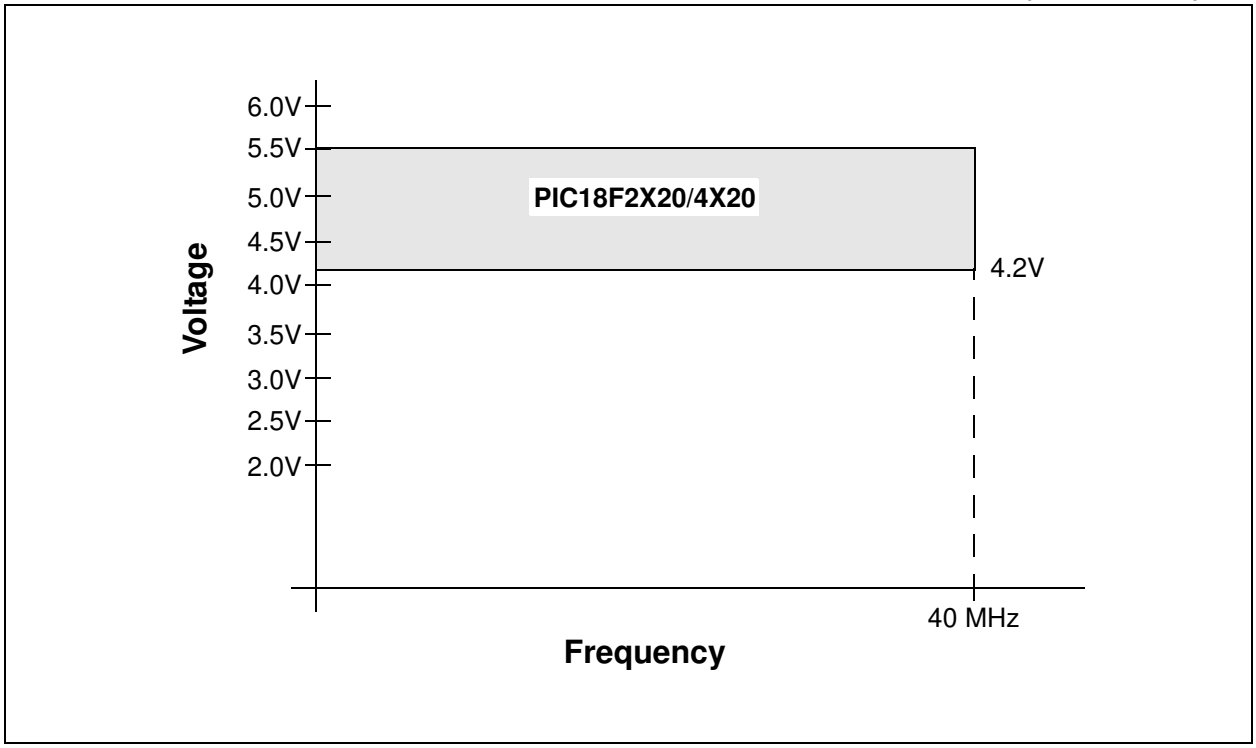
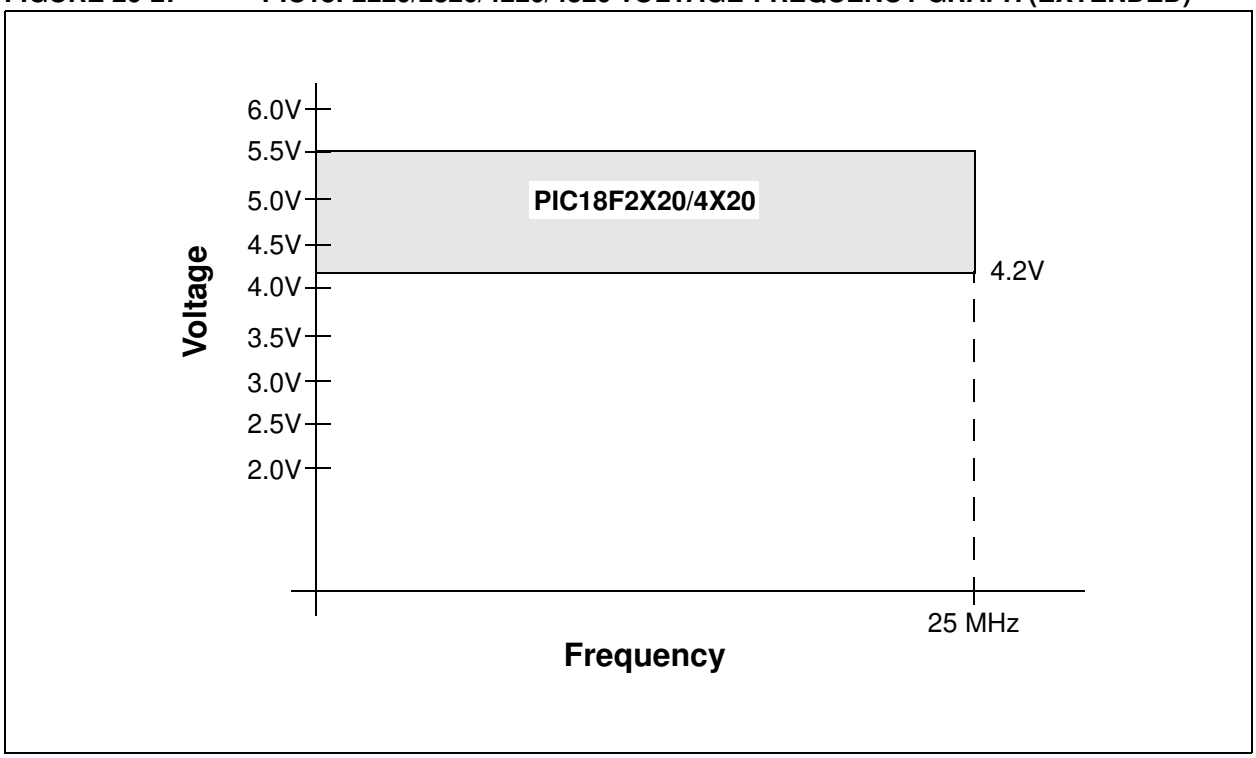


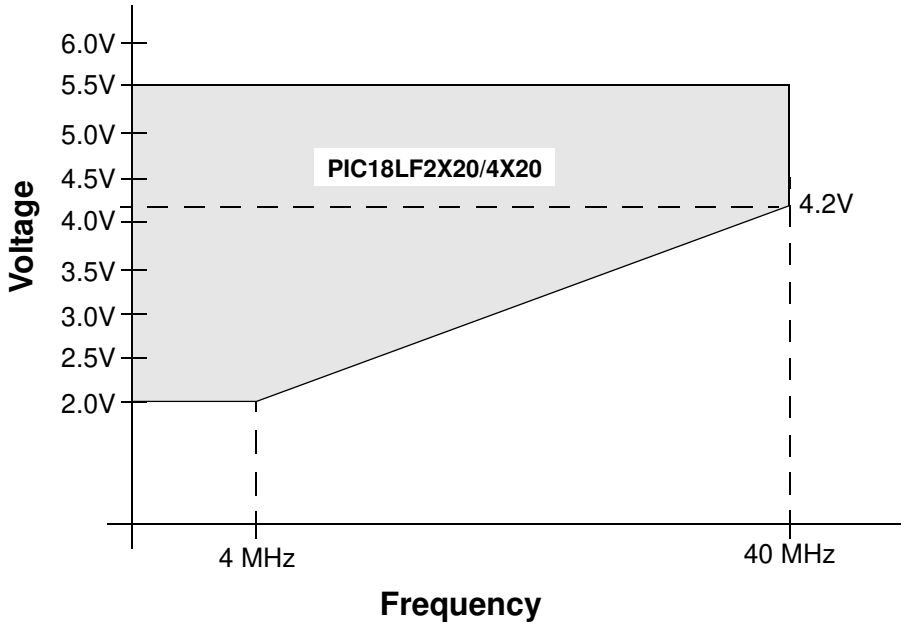

**FIGURE 26-2: PIC18F2220/2320/4220/4320 VOLTAGE-FREQUENCY GRAPH (EXTENDED)**





© 2003 Microchip Technology Inc.

**FIGURE 26-3: PIC18LF2220/2320/4220/4320 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)**




FMAX = (16.36 MHz/V) (VDDAPPMIN – 2.0V) + 4 MHz

**Note:** VDDAPPMIN is the minimum voltage of the PICmicro® device in the application.

## 26.1 DC Characteristics: Supply Voltage PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Typ** | **Max** | **Units** | **Conditions** |
| | VDD | **Supply Voltage** | | | | | |
| D001 | | PIC18LF2X20/4X20 | 2.0 | — | 5.5 | V | HS, XT, RC and LP Osc mode |
| | | PIC18F2X20/4X20 | 4.2 | — | 5.5 | V | |
| D002 | VDR | **RAM Data Retention Voltage(1)** | 1.5 | — | — | V | |
| D003 | VPOR | **VDD Start Voltage** to ensure internal Power-on Reset signal | — | — | 0.7 | V | See section on Power-on Reset for details |
| D004 | SVDD | **VDD Rise Rate** to ensure internal Power-on Reset signal | 0.05 | — | — | V/ms | See section on Power-on Reset for details |
| | VBOR | **Brown-out Reset Voltage** | | | | | |
| | | PIC18LF2X20/4X20 | Industrial Low Voltage | | | | |
| D005 | | BORV1:BORV0 = 11 | NA | — | NA | V | Reserved |
| | | BORV1:BORV0 = 10 | 2.50 | 2.72 | 2.94 | V | |
| | | BORV1:BORV0 = 01 | 3.88 | 4.22 | 4.56 | V | |
| | | BORV1:BORV0 = 00 | 4.18 | 4.54 | 4.90 | V | |
| D005 | | PIC18F2X20/4X20 | Industrial | | | | |
| | | BORV1:BORV0 = 1x | NA | — | NA | V | Not in operating voltage range of device |
| | | BORV1:BORV0 = 01 | 3.88 | 4.22 | 4.56 | V | |
| | | BORV1:BORV0 = 00 | 4.18 | 4.54 | 4.90 | V | |
| D005E | | PIC18F2X20/4X20 | Extended | | | | |
| | | BORV1:BORV0 = 1x | NA | — | NA | V | Not in operating voltage range of device |
| | | BORV1:BORV0 = 01 | 3.71 | 4.22 | 4.73 | V | |
| | | BORV1:BORV0 = 00 | 4.00 | 4.54 | 5.08 | V | |

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

**Note 1:** This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.


© 2003 Microchip Technology Inc.

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Power-down Current (IPD)(1)** | | | | | | |
| | PIC18LF2X20/4X20 | 0.1 | 0.5 | µA | -40°C | | VDD = 2.0V, (**Sleep** mode) |
| | | 0.1 | 0.5 | µA | +25°C | | |
| | | 0.2 | 1.7 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 0.1 | 0.5 | µA | -40°C | | VDD = 3.0V, (**Sleep** mode) |
| | | 0.1 | 0.5 | µA | +25°C | | |
| | | 0.3 | 1.7 | µA | +85°C | | |
| | All devices | 0.1 | 2.0 | µA | -40°C | | VDD = 5.0V, (**Sleep** mode) |
| | | 0.1 | 2.0 | µA | +25°C | | |
| | | 0.4 | 6.5 | µA | +85°C | | |
| | Extended devices | 11.2 | 50 | µA | +125°C | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 11 | 25 | µA | -40°C | VDD = 2.0V | FOSC = 31 kHz (**RC_RUN** mode, internal oscillator source) |
| | | 13 | 25 | µA | +25°C | | |
| | | 14 | 25 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 34 | 40 | µA | -40°C | VDD = 3.0V | |
| | | 28 | 40 | µA | +25°C | | |
| | | 25 | 40 | µA | +85°C | | |
| | All devices | 77 | 80 | µA | -40°C | VDD = 5.0V | |
| | | 62 | 80 | µA | +25°C | | |
| | | 53 | 80 | µA | +85°C | | |
| | Extended devices | 50 | 80 | µA | +125°C | | |

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

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

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

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

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

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 100 | 220 | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**RC_RUN** mode, internal oscillator source) |
| | | 110 | 220 | µA | +25°C | | |
| | | 120 | 220 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 180 | 330 | µA | -40°C | VDD = 3.0V | |
| | | 180 | 330 | µA | +25°C | | |
| | | 170 | 330 | µA | +85°C | | |
| | All devices | 340 | 550 | µA | -40°C | VDD = 5.0V | |
| | | 330 | 550 | µA | +25°C | | |
| | | 310 | 550 | µA | +85°C | | |
| | Extended devices | 410 | 650 | µA | +125°C | | |
| | PIC18LF2X20/4X20 | 350 | 600 | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**RC_RUN** mode, internal oscillator source) |
| | | 360 | 600 | µA | +25°C | | |
| | | 370 | 600 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 580 | 900 | µA | -40°C | VDD = 3.0V | |
| | | 580 | 900 | µA | +25°C | | |
| | | 560 | 900 | µA | +85°C | | |
| | All devices | 1.1 | 1.8 | mA | -40°C | VDD = 5.0V | |
| | | 1.1 | 1.8 | mA | +25°C | | |
| | | 1.0 | 1.8 | mA | +85°C | | |
| | Extended devices | 1.2 | 1.8 | mA | +125°C | | |

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

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

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

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

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


© 2003 Microchip Technology Inc.

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 4.7 | 8 | µA | -40°C | VDD = 2.0V | FOSC = 31 kHz (**RC_IDLE** mode, internal oscillator source) |
| | | 4.6 | 8 | µA | +25°C | | |
| | | 5.1 | 11 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 6.9 | 11 | µA | -40°C | VDD = 3.0V | |
| | | 6.3 | 11 | µA | +25°C | | |
| | | 6.8 | 15 | µA | +85°C | | |
| | All devices | 12 | 16 | µA | -40°C | VDD = 5.0V | |
| | | 10 | 16 | µA | +25°C | | |
| | | 10 | 22 | µA | +85°C | | |
| | Extended devices | 25 | 75 | µA | +125°C | | |
| | PIC18LF2X20/4X20 | 49 | 150 | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**RC_IDLE** mode, internal oscillator source) |
| | | 52 | 150 | µA | +25°C | | |
| | | 56 | 150 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 73 | 180 | µA | -40°C | VDD = 3.0V | |
| | | 77 | 180 | µA | +25°C | | |
| | | 77 | 180 | µA | +85°C | | |
| | All devices | 130 | 300 | µA | -40°C | VDD = 5.0V | |
| | | 130 | 300 | µA | +25°C | | |
| | | 130 | 300 | µA | +85°C | | |
| | Extended devices | 350 | 435 | µA | +125°C | | |

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

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

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

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

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

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 140 | 275 | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**RC_IDLE** mode, internal oscillator source) |
| | | 140 | 275 | µA | +25°C | | |
| | | 150 | 275 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 220 | 375 | µA | -40°C | VDD = 3.0V | |
| | | 220 | 375 | µA | +25°C | | |
| | | 210 | 375 | µA | +85°C | | |
| | All devices | 390 | 800 | µA | -40°C | VDD = 5.0V | |
| | | 400 | 800 | µA | +25°C | | |
| | | 380 | 800 | µA | +85°C | | |
| | Extended devices | 410 | 800 | µA | +125°C | | |
| | PIC18LF2X20/4X20 | 150 | 250 | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**PRI_RUN**, EC oscillator) |
| | | 150 | 250 | µA | +25°C | | |
| | | 160 | 250 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 340 | 350 | µA | -40°C | VDD = 3.0V | |
| | | 300 | 350 | µA | +25°C | | |
| | | 280 | 350 | µA | +85°C | | |
| | All devices | 0.72 | 1.0 | mA | -40°C | VDD = 5.0V | |
| | | 0.63 | 1.0 | mA | +25°C | | |
| | | 0.57 | 1.0 | mA | +85°C | | |
| | Extended devices | 0.53 | 1.0 | mA | +125°C | | |

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

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

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

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

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


© 2003 Microchip Technology Inc.

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 440 | 600 | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**PRI_RUN**, EC oscillator) |
| | | 450 | 600 | µA | +25°C | | |
| | | 460 | 600 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 0.80 | 1.0 | mA | -40°C | VDD = 3.0V | |
| | | 0.78 | 1.0 | mA | +25°C | | |
| | | 0.77 | 1.0 | mA | +85°C | | |
| | All devices | 1.6 | 2.0 | mA | -40°C | VDD = 5.0V | |
| | | 1.5 | 2.0 | mA | +25°C | | |
| | | 1.5 | 2.0 | mA | +85°C | | |
| | Extended devices | 1.5 | 2.0 | mA | +125°C | | |
| | Extended devices | 6.3 | 9.0 | mA | +125°C | VDD = 4.2V | FOSC = 25 MHz (**PRI_RUN**, EC oscillator) |
| | | 7.9 | 10.0 | mA | +125°C | VDD = 5.0V | |
| | All devices | 9.5 | 12 | mA | -40°C | VDD = 4.2V | FOSC = 40 MHz (**PRI_RUN**, EC oscillator) |
| | | 9.7 | 12 | mA | +25°C | | |
| | | 9.9 | 12 | mA | +85°C | | |
| | All devices | 11.9 | 15 | mA | -40°C | VDD = 5.0V | |
| | | 12.1 | 15 | mA | +25°C | | |
| | | 12.3 | 15 | mA | +85°C | | |

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

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

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

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

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

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| PIC18LF2220/2320/4220/4320 (Industrial) | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial |
|---|---|
| PIC18F2220/2320/4220/4320 (Industrial, Extended) | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended |

| Param No. | Device | Typ | Max | Units | Conditions | | |
|---|---|---|---|---|---|---|---|
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | PIC18LF2X20/4X20 | 37 | 50 | µA | -40°C | VDD = 2.0V | FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 37 | 50 | µA | +25°C | | |
| | | 38 | 60 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 58 | 80 | µA | -40°C | VDD = 3.0V | |
| | | 59 | 80 | µA | +25°C | | |
| | | 60 | 100 | µA | +85°C | | |
| | All devices | 110 | 180 | µA | -40°C | VDD = 5.0V | |
| | | 110 | 180 | µA | +25°C | | |
| | | 110 | 180 | µA | +85°C | | |
| | Extended devices | 125 | 300 | µA | +125°C | | |
| | PIC18LF2X20/4X20 | 140 | 180 | µA | -40°C | VDD = 2.0V | FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 140 | 180 | µA | +25°C | | |
| | | 140 | 180 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 220 | 280 | µA | -40°C | VDD = 3.0V | |
| | | 230 | 280 | µA | +25°C | | |
| | | 230 | 280 | µA | +85°C | | |
| | All devices | 410 | 525 | µA | -40°C | VDD = 5.0V | |
| | | 420 | 525 | µA | +25°C | | |
| | | 430 | 525 | µA | +85°C | | |
| | Extended devices | 450 | 800 | µA | +125°C | | |
| | Extended devices | 2.2 | 3.0 | mA | +125°C | VDD = 4.2V | FOSC = 25 MHz (**PRI_IDLE**, EC oscillator) |
| | | 2.7 | 3.5 | mA | +125°C | VDD = 5.0V | |

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

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

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

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

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


© 2003 Microchip Technology Inc.

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Supply Current (IDD)(2,3)** | | | | | | |
| | All devices | 3.1 | 4.1 | mA | -40°C | VDD = 4.2 V | FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator) |
| | | 3.2 | 4.1 | mA | +25°C | | |
| | | 3.3 | 4.1 | mA | +85°C | | |
| | All devices | 4.4 | 5.1 | mA | -40°C | VDD = 5.0V | |
| | | 4.6 | 5.1 | mA | +25°C | | |
| | | 4.6 | 5.1 | mA | +85°C | | |
| | PIC18LF2X20/4X20 | 9 | 15 | µA | -40°C | VDD = 2.0V | FOSC = 32 kHz(4) (**SEC_RUN** mode, Timer1 as clock) |
| | | 10 | 15 | µA | +25°C | | |
| | | 13 | 18 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 22 | 30 | µA | -40°C | VDD = 3.0V | |
| | | 21 | 30 | µA | +25°C | | |
| | | 20 | 35 | µA | +85°C | | |
| | All devices | 50 | 80 | µA | -40°C | VDD = 5.0V | |
| | | 50 | 80 | µA | +25°C | | |
| | | 45 | 85 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 5.1 | 9 | µA | -40°C | VDD = 2.0V | FOSC = 32 kHz(4) (**SEC_IDLE** mode, Timer1 as clock) |
| | | 5.8 | 9 | µA | +25°C | | |
| | | 7.9 | 11 | µA | +85°C | | |
| | PIC18LF2X20/4X20 | 7.9 | 12 | µA | -40°C | VDD = 3.0V | |
| | | 8.9 | 12 | µA | +25°C | | |
| | | 10.5 | 14 | µA | +85°C | | |
| | All devices | 13 | 20 | µA | -40°C | VDD = 5.0V | |
| | | 16 | 20 | µA | +25°C | | |
| | | 18 | 25 | µA | +85°C | | |

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

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

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

$\overline{OSC1}$ = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

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

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

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)** | | | | | | |
| D022 (ΔIWDT) | **Watchdog Timer** | 1.5 | 3.8 | µA | -40°C | VDD = 2.0V | |
| | | 2.2 | 3.8 | µA | +25°C | | |
| | | 2.7 | 4.0 | µA | +85°C | | |
| | | 2.3 | 4.6 | µA | -40°C | VDD = 3.0V | |
| | | 2.7 | 4.6 | µA | +25°C | | |
| | | 3.1 | 4.8 | µA | +85°C | | |
| | | 3.0 | 10.0 | µA | -40°C | VDD = 5.0V | |
| | | 3.3 | 10.0 | µA | +25°C | | |
| | | 3.9 | 10.0 | µA | +85°C | | |
| | Extended devices only | 4.0 | 13.0 | µA | +125°C | | |
| D022A (ΔIBOR) | **Brown-out Reset** | 17 | 35.0 | µA | -40°C to +85°C | VDD = 3.0V | |
| | | 47 | 45.0 | µA | -40°C to +85°C | VDD = 5.0V | |
| | Extended devices only | 48 | 50.0 | µA | -40°C to +125°C | | |
| D022B (ΔILVD) | **Low-Voltage Detect** | 14 | 25.0 | µA | -40°C to +85°C | VDD = 2.0V | |
| | | 18 | 35.0 | µA | -40°C to +85°C | VDD = 3.0V | |
| | | 21 | 45.0 | µA | -40°C to +85°C | VDD = 5.0V | |
| | Extended devices only | 24 | 50.0 | µA | -40°C to +125°C | | |

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

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

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

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

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


© 2003 Microchip Technology Inc.

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC18LF2220/2320/4220/4320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | |
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device** | **Typ** | **Max** | **Units** | **Conditions** | | |
| D025 (ΔIOSCB) | **Timer1 Oscillator** | 2.1 | 2.2 | µA | -40°C | VDD = 2.0V | 32 kHz on Timer1 **(4)** |
| | | 1.8 | 2.2 | µA | +25°C | | |
| | | 2.1 | 2.2 | µA | +85°C | | |
| | | 2.2 | 3.8 | µA | -40°C | VDD = 3.0V | 32 kHz on Timer1 **(4)** |
| | | 2.6 | 3.8 | µA | +25°C | | |
| | | 2.9 | 3.8 | µA | +85°C | | |
| | | 3.0 | 6.0 | µA | -40°C | VDD = 5.0V | 32 kHz on Timer1 **(4)** |
| | | 3.2 | 6.0 | µA | +25°C | | |
| | | 3.4 | 7.0 | µA | +85°C | | |
| D026 (ΔIAD) | **A/D Converter** | 1.0 | 2.0 | µA | -40°C to +85°C | VDD = 2.0V | A/D on, not converting |
| | | 1.0 | 2.0 | µA | -40°C to +85°C | VDD = 3.0V | |
| | | 1.0 | 2.0 | µA | -40°C to +85°C | VDD = 5.0V | |
| | Extended devices only | 1.0 | 8.0 | µA | -40°C to +125°C | VDD = 5.0V | |

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

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

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

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

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

## 26.3 DC Characteristics: PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | |
|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Max** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage** | | | | |
| | | I/O ports: | | | | |
| D030 | | with TTL buffer | VSS | 0.15 VDD | V | VDD < 4.5V |
| D030A | | | — | 0.8 | V | 4.5V ≤ VDD ≤ 5.5V |
| D031 | | with Schmitt Trigger buffer | VSS | 0.2 VDD | V | |
| | | RC3 and RC4 | VSS | 0.3 VDD | V | |
| D032 | | $\overline{MCLR}$ | VSS | 0.2 VDD | V | |
| D032A | | OSC1 and T1OSI | VSS | 0.2 VDD | V | LP, XT, HS, HSPLL modes(1) |
| D033 | | OSC1 | VSS | 0.2 VDD | V | EC mode(1) |
| | VIH | **Input High Voltage** | | | | |
| | | I/O ports: | | | | |
| D040 | | with TTL buffer | 0.25 VDD + 0.8V | VDD | V | VDD < 4.5V |
| D040A | | | 2.0 | VDD | V | 4.5V ≤ VDD ≤ 5.5V |
| D041 | | with Schmitt Trigger buffer | 0.8 VDD | VDD | V | |
| | | RC3 and RC4 | 0.7 VDD | VDD | V | |
| D042 | | $\overline{MCLR}$ | 0.8 VDD | VDD | V | |
| D042A | | OSC1 and T1OSI | 1.6 | VDD | V | LP, XT, HS, HSPLL modes(1) |
| D043 | | OSC1 | 0.8 VDD | VDD | V | EC mode(1) |
| | IIL | **Input Leakage Current(2,3)** | | | | |
| D060 | | I/O ports | — | ±0.2 | µA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance |
| D061 | | $\overline{MCLR}$, RA4 | — | ±1.0 | µA | Vss ≤ VPIN ≤ VDD |
| D063 | | OSC1 | — | ±1.0 | µA | Vss ≤ VPIN ≤ VDD |
| | IPU | **Weak Pull-up Current** | | | | |
| D070 | IPURB | PORTB weak pull-up current | 50 | 400 | µA | VDD = 5V, VPIN = VSS |

**Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PICmicro device be driven with an external clock while in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**3:** Negative current is defined as current sourced by the pin.

**4:** Parameter is characterized but not tested.


© 2003 Microchip Technology Inc.

## 26.3 DC Characteristics: PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial) (Continued)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | |
|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Max** | **Units** | **Conditions** |
| | VOL | **Output Low Voltage** | | | | |
| D080 | | I/O ports | — | 0.6 | V | IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C |
| D080A | | | — | 0.6 | V | IOL = 7.0 mA, VDD = 4.5V, -40°C to +125°C |
| D083 | | OSC2/CLKO (RC mode) | — | 0.6 | V | IOL = 1.6 mA, VDD = 4.5V, -40°C to +85°C |
| D083A | | | — | 0.6 | V | IOL = 1.2 mA, VDD = 4.5V, -40°C to +125°C |
| | VOH | **Output High Voltage(3)** | | | | |
| D090 | | I/O ports | VDD – 0.7 | — | V | IOH = -3.0 mA, VDD = 4.5V, -40°C to +85°C |
| D090A | | | VDD – 0.7 | — | V | IOH = -2.5 mA, VDD = 4.5V, -40°C to +125°C |
| D092 | | OSC2/CLKO (RC mode) | VDD – 0.7 | — | V | IOH = -1.3 mA, VDD = 4.5V, -40°C to +85°C |
| D092A | | | VDD – 0.7 | — | V | IOH = -1.0 mA, VDD = 4.5V, -40°C to +125°C |
| D150 | VOD | **Open-Drain High Voltage** | — | 8.5 | V | RA4 pin |
| | | **Capacitive Loading Specs on Output Pins** | | | | |
| D100(4) | COSC2 | OSC2 pin | — | 15 | pF | In XT, HS and LP modes when external clock is used to drive OSC1 |
| D101 | CIO | All I/O pins and OSC2 (in RC mode) | — | 50 | pF | To meet the AC Timing Specifications |
| D102 | CB | SCL, SDA | — | 400 | pF | In $I^2C$ mode |

**Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PICmicro device be driven with an external clock while in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**3:** Negative current is defined as current sourced by the pin.

**4:** Parameter is characterized but not tested.

**TABLE 26-1: MEMORY PROGRAMMING REQUIREMENTS**

| DC Characteristics | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | | **Internal Program Memory Programming Specifications** | | | | | |
| D110 | VPP | Voltage on $\overline{MCLR}$/VPP pin | 9.00 | — | 13.25 | V | **(Note 2)** |
| D112 | IPP | Current into $\overline{MCLR}$/VPP pin | — | — | 300 | µA | |
| D113 | IDDP | Supply Current during Programming | — | — | 1.0 | mA | |
| | | **Data EEPROM Memory** | | | | | |
| D120 | ED | Byte Endurance | 100K<br>10K | 1M<br>100K | —<br>— | E/W<br>E/W | -40°C to +85°C<br>-40°C to +125°C |
| D121 | VDRW | VDD for Read/Write | VMIN | — | 5.5 | V | Using EECON to read/write<br>VMIN = Minimum operating voltage |
| D122 | TDEW | Erase/Write Cycle Time | — | 4 | — | ms | |
| D123 | TRETD | Characteristic Retention | 40 | — | — | Year | Provided no other specifications are violated |
| D124 | TREF | Number of Total Erase/Write Cycles before Refresh(1) | 1M<br>100K | 10M<br>1M | —<br>— | E/W<br>E/W | -40°C to +85°C<br>-40°C to +125°C |
| | | **Program Flash Memory** | | | | | |
| D130 | EP | Cell Endurance | 10K<br>1K | 100K<br>10K | —<br>— | E/W<br>E/W | -40°C to +85°C<br>-40°C to +125°C |
| D131 | VPR | VDD for Read | VMIN | — | 5.5 | V | VMIN = Minimum operating voltage |
| D132 | VIE | VDD for Block Erase | 4.5 | — | 5.5 | V | Using ICSP port |
| D132A | VIW | VDD for Externally Timed Erase or Write | 4.5 | — | 5.5 | V | Using ICSP port |
| D132B | VPEW | VDD for Self-timed Write | VMIN | — | 5.5 | V | VMIN = Minimum operating voltage |
| D133 | TIE | ICSP Block Erase Cycle Time | — | 4 | — | ms | VDD > 4.5V |
| D133A | TIW | ICSP Erase or Write Cycle Time (externally timed) | 1 | — | — | ms | VDD > 4.5V |
| D133A | TIW | Self-timed Write Cycle Time | — | 2 | — | ms | |
| D134 | TRETD | Characteristic Retention | 40 | — | — | Year | Provided no other specifications are violated |

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Refer to **Section 7.8 "Using the Data EEPROM"** for a more detailed discussion on data EEPROM endurance.

**2:** Required only if Low-Voltage Programming is disabled.


© 2003 Microchip Technology Inc.

**TABLE 26-2: COMPARATOR SPECIFICATIONS**

| **Operating Conditions:** 3.0V < VDD < 5.5V, -40°C < TA < +125°C, unless otherwise stated. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristics** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| D300 | VIOFF | Input Offset Voltage | — | ± 5.0 | ± 10 | mV | |
| D301 | VICM | Input Common Mode Voltage* | 0 | — | VDD – 1.5 | V | |
| D302 | CMRR | Common Mode Rejection Ratio* | 55 | — | — | dB | |
| 300<br>300A | TRESP | Response Time(1)* | — | 150 | 400<br>600 | ns<br>ns | PIC18FXX20<br>PIC18LFXX20 |
| 301 | TMC2OV | Comparator Mode Change to Output Valid* | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at (VDD – 1.5)/2, while the other input transitions from VSS to VDD.

**TABLE 26-3: VOLTAGE REFERENCE SPECIFICATIONS**

| **Operating Conditions:** 3.0V < VDD < 5.5V, -40°C < TA < +125°C, unless otherwise stated. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristics** | **Min** | **Typ** | **Max** | **Units** | **Comments** |
| D310 | VRES | Resolution | VDD/24 | — | VDD/32 | LSb | |
| D311 | VRAA | Absolute Accuracy | —<br>— | —<br>— | 1/2<br>1/2 | LSb<br>LSb | Low Range (VRR = 1)<br>High Range (VRR = 0) |
| D312 | VRUR | Unit Resistor Value (R)* | — | 2k | — | Ω | |
| 310 | TSET | Settling Time(1)* | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while VRR = 1 and VR<3:0> transitions from '0000' to '1111'.

**FIGURE 26-4: LOW-VOLTAGE DETECT CHARACTERISTICS**

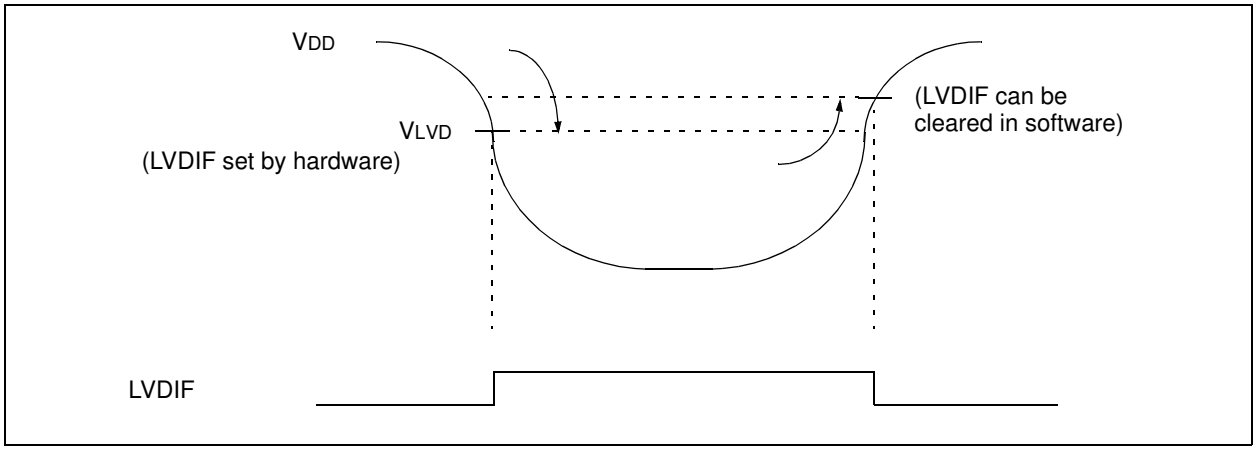



**TABLE 26-4: LOW-VOLTAGE DETECT CHARACTERISTICS**

| **PIC18LF2220/2320/4220/4320** (Industrial) | | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | |
|---|---|---|---|---|---|---|---|---|
| **PIC18F2220/2320/4220/4320** (Industrial, Extended) | | | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param No.** | **Symbol** | **Characteristic** | | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| D420 | | LVD Voltage on VDD Transition High to Low | | Industrial | | | | |
| | | PIC18LF2X20/4X20 | LVDL<3:0> = 0000 | N/A | N/A | N/A | V | Reserved |
| | | | LVDL<3:0> = 0001 | N/A | N/A | N/A | V | Reserved |
| | | | LVDL<3:0> = 0010 | 2.15 | 2.26 | 2.37 | V | |
| | | | LVDL<3:0> = 0011 | 2.33 | 2.45 | 2.58 | V | |
| | | | LVDL<3:0> = 0100 | 2.43 | 2.55 | 2.68 | V | |
| | | | LVDL<3:0> = 0101 | 2.63 | 2.77 | 2.91 | V | |
| | | | LVDL<3:0> = 0110 | 2.73 | 2.87 | 3.01 | V | |
| | | | LVDL<3:0> = 0111 | 2.91 | 3.07 | 3.22 | V | |
| | | | LVDL<3:0> = 1000 | 3.20 | 3.36 | 3.53 | V | |
| | | | LVDL<3:0> = 1001 | 3.39 | 3.57 | 3.75 | V | |
| | | | LVDL<3:0> = 1010 | 3.49 | 3.67 | 3.85 | V | |
| | | | LVDL<3:0> = 1011 | 3.68 | 3.87 | 4.07 | V | |
| | | | LVDL<3:0> = 1100 | 3.87 | 4.07 | 4.28 | V | |
| | | | LVDL<3:0> = 1101 | 4.06 | 4.28 | 4.49 | V | |
| | | | LVDL<3:0> = 1110 | 4.37 | 4.60 | 4.82 | V | |
| D420 | | LVD Voltage on VDD Transition High to Low | | Industrial | | | | |
| | | PIC18F2X20/4X20 | LVDL<3:0> = 1011 | 3.68 | 3.87 | 4.07 | V | |
| | | | LVDL<3:0> = 1100 | 3.87 | 4.07 | 4.28 | V | |
| | | | LVDL<3:0> = 1101 | 4.06 | 4.28 | 4.49 | V | |
| | | | LVDL<3:0> = 1110 | 4.37 | 4.60 | 4.82 | V | |
| D420E | | LVD Voltage on VDD Transition High to Low | | Extended | | | | |
| | | PIC18F2X20/4X20 | LVDL<3:0> = 1011 | 3.48 | 3.87 | 4.25 | V | |
| | | | LVDL<3:0> = 1100 | 3.66 | 4.07 | 4.48 | V | |
| | | | LVDL<3:0> = 1101 | 3.85 | 4.28 | 4.70 | V | |
| | | | LVDL<3:0> = 1110 | 4.14 | 4.60 | 5.05 | V | |

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

† Production tested at TAMB = 25°C. Specifications over temperature limits ensured by characterization.


© 2003 Microchip Technology Inc.

## 26.4 AC (Timing) Characteristics

### 26.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

| | | | |
|---|---|---|---|
| 1. TppS2ppS | | 3. TCC:ST | ($I^2C$ specifications only) |
| 2. TppS | | 4. Ts | ($I^2C$ specifications only) |

| | | | |
|---|---|---|---|
| T | | | |
| F | Frequency | T | Time |

Lowercase letters (pp) and their meanings:

| | | | |
|---|---|---|---|
| pp | | | |
| cc | CCP1 | osc | OSC1 |
| ck | CLKO | rd | $\overline{RD}$ |
| cs | $\overline{CS}$ | rw | $\overline{RD}$ or $\overline{WR}$ |
| di | SDI | sc | SCK |
| do | SDO | ss | $\overline{SS}$ |
| dt | Data in | t0 | T0CKI |
| io | I/O port | t1 | T1CKI |
| mc | $\overline{MCLR}$ | wr | $\overline{WR}$ |

Uppercase letters and their meanings:

| | | | |
|---|---|---|---|
| S | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (High-impedance) | V | Valid |
| L | Low | Z | High-impedance |
| $I^2C$ only | | | |
| AA | output access | High | High |
| BUF | Bus free | Low | Low |

TCC:ST ($I^2C$ specifications only)

| | | | |
|---|---|---|---|
| CC | | | |
| HD | Hold | SU | Setup |
| ST | | | |
| DAT | DATA input hold | STO | Stop condition |
| STA | Start condition | | |

### 26.4.2 TIMING CONDITIONS

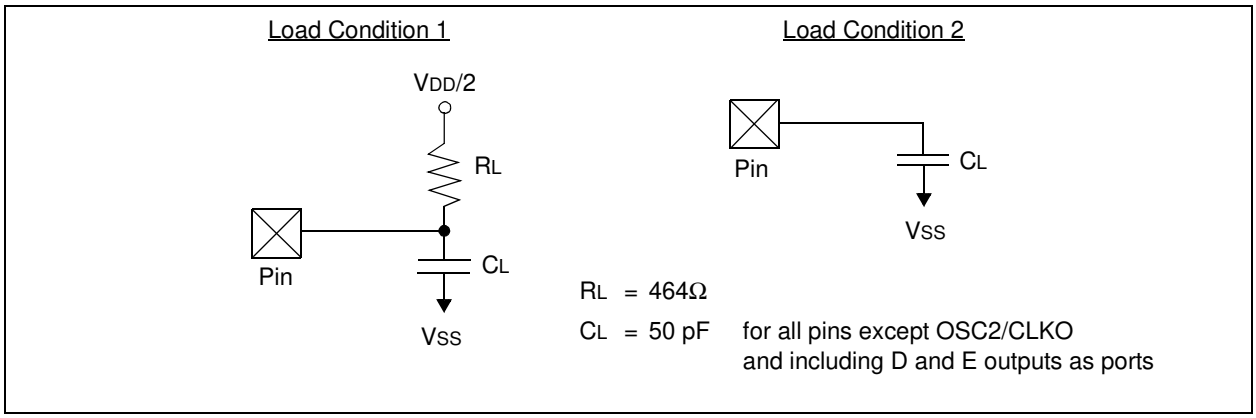
The temperature and voltages specified in Table 26-5 apply to all timing specifications unless otherwise noted. Figure 26-5 specifies the load conditions for the timing specifications.

| | |
|---|---|
| **Note:** | Because of space limitations, the generic terms “PIC18FXX20” and “PIC18LFXX20” are used throughout this section to refer to the PIC18F2220/2320/4220/4320 and PIC18LF2220/2320/4220/4320 families of devices specifically and only those devices. |

**TABLE 26-5: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC**

| | |
|---|---|
| **AC CHARACTERISTICS** | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended<br>Operating voltage VDD range as described in DC spec **Section 26.1** and **Section 26.3**.<br>LF parts operate up to industrial temperatures only. |

**FIGURE 26-5: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**





© 2003 Microchip Technology Inc.

### 26.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

**FIGURE 26-6: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)**

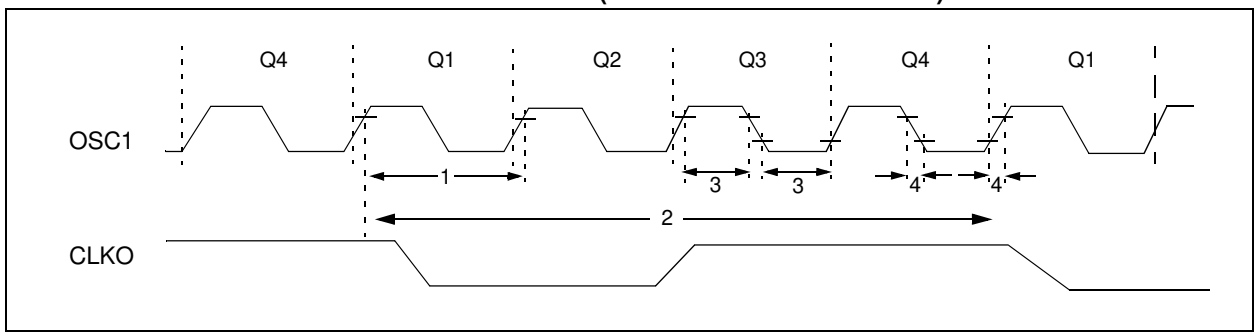



**TABLE 26-6: EXTERNAL CLOCK TIMING REQUIREMENTS**

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|
| 1A | FOSC | External CLKI Frequency(1) | DC | 40 | MHz | EC, ECIO (industrial) |
| | | | DC | 25 | MHz | EC, ECIO (extended) |
| | | Oscillator Frequency(1) | DC | 4 | MHz | RC osc |
| | | | 0.1 | 1 | MHz | XT osc |
| | | | 4 | 25 | MHz | HS osc |
| | | | 4 | 10 | MHz | HS + PLL osc (industrial) |
| | | | 4 | 6.25 | MHz | HS + PLL osc (extended) |
| | | | 5 | 33 | kHz | LP Osc mode |
| 1 | TOSC | External CLKI Period(1) | 25 | — | ns | EC, ECIO (industrial) |
| | | | 40 | — | ns | EC, ECIO (extended) |
| | | Oscillator Period(1) | 250 | — | ns | RC osc |
| | | | 1 | — | µs | XT osc |
| | | | 40 | 250 | ns | HS osc |
| | | | 100 | 250 | ns | HS + PLL osc (industrial) |
| | | | 160 | 250 | ns | HS + PLL osc (extended) |
| | | | 30 | — | µs | LP osc |
| 2 | TCY | Instruction Cycle Time(1) | 100 | — | ns | TCY = 4/FOSC (industrial) |
| | | | 160 | — | ns | TCY = 4/FOSC (extended) |
| 3 | TOSL, TOSH | External Clock in (OSC1) High or Low Time | 30 | — | ns | XT osc |
| | | | 2.5 | — | µs | LP osc |
| | | | 10 | — | ns | HS osc |
| 4 | TOSR, TOSF | External Clock in (OSC1) Rise or Fall Time | — | 20 | ns | XT osc |
| | | | — | 50 | ns | LP osc |
| | | | — | 7.5 | ns | HS osc |

**Note 1:** Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

**TABLE 26-7: PLL CLOCK TIMING SPECIFICATIONS ($V_{DD}$ = 4.2V TO 5.5V)**

| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| F10 | $F_{OSC}$ | Oscillator Frequency Range | 4 | — | 10 | MHz | HS mode only |
| F11 | $F_{SYS}$ | On-Chip VCO System Frequency | 16 | — | 40 | MHz | HS mode only |
| F12 | $t_{PLL}$ | PLL Start-up Time (Lock Time) | — | — | 2 | ms | |
| F13 | $\Delta_{CLK}$ | CLKO Stability (Jitter) | -2 | — | +2 | % | |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**TABLE 26-8: INTERNAL RC ACCURACY: PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial, Extended)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **PIC18LF1220/1320** (Industrial) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ $T_A$ ≤ +85°C for industrial | | | | | | |
| **PIC18F1220/1320** (Industrial, Extended) | | **Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ $T_A$ ≤ +85°C for industrial -40°C ≤ $T_A$ ≤ +125°C for extended | | | | | | |
| **Param No.** | **Device** | **Min** | **Typ** | **Max** | **Units** | **Conditions** | | |
| | **INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz(1)** | | | | | | | |
| F14 | PIC18LF2220/2320/4220/4320 | -2 | +/-1 | 2 | % | +25°C | $V_{DD}$ = 2.7-3.3V | |
| F15 | | -5 | — | 5 | % | -10°C to +85°C | $V_{DD}$ = 2.7-3.3V | |
| F16 | | -10 | — | 10 | % | -40°C to +85°C | $V_{DD}$ = 2.7-3.3V | |
| F17 | PIC18F2220/2320/4220/4320 | -2 | +/-1 | 2 | % | +25°C | $V_{DD}$ = 4.5-5.5V | |
| F18 | | -5 | — | 5 | % | -10°C to +85°C | $V_{DD}$ = 4.5-5.5V | |
| F19 | | -10 | — | 10 | % | -40°C to +85°C | $V_{DD}$ = 4.5-5.5V | |
| | **INTRC Accuracy @ Freq = 31 kHz(2)** | | | | | | | |
| F20 | PIC18LF2220/2320/4220/4320 | 26.562 | — | 35.938 | kHz | -40°C to +85°C | $V_{DD}$ = 2.7-3.3V | |
| F21 | PIC18F2220/2320/4220/4320 | 26.562 | — | 35.938 | kHz | -40°C to +85°C | $V_{DD}$ = 4.5-5.5V | |

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

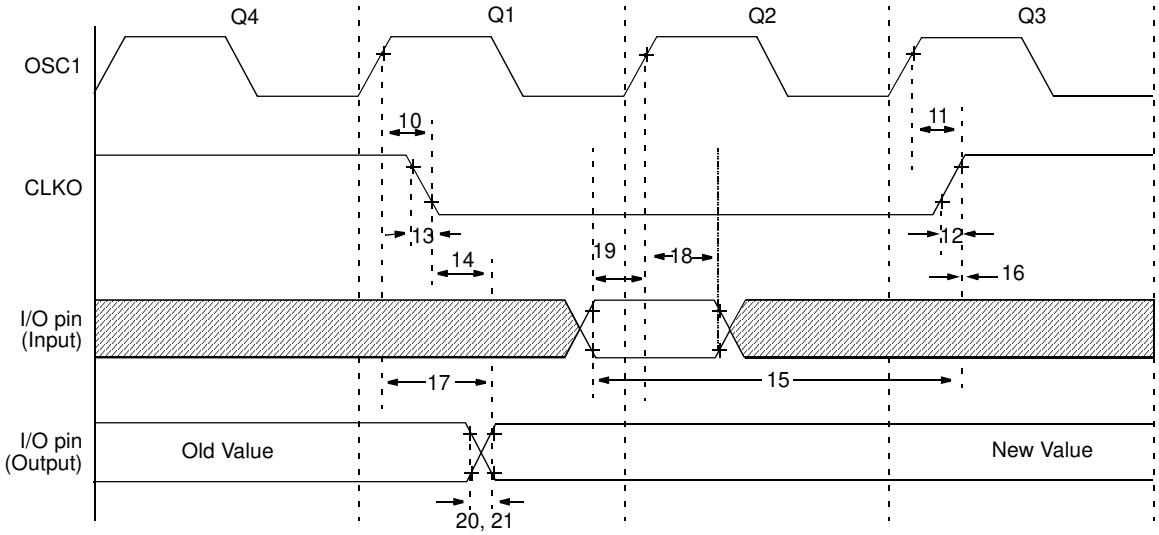
**Note 1:** Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.

**2:** INTRC frequency after calibration.

**3:** Change of INTRC frequency as $V_{DD}$ changes.


© 2003 Microchip Technology Inc.

**FIGURE 26-7: CLKO AND I/O TIMING**



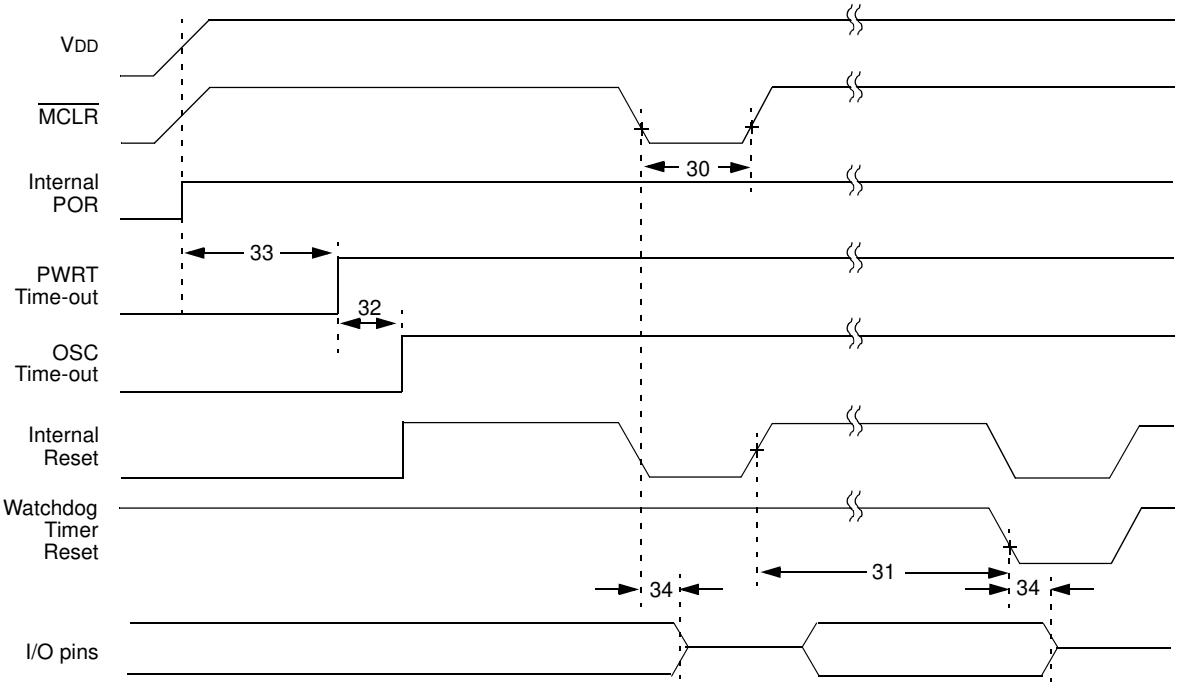

**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-9: CLKO AND I/O TIMING REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 10 | TOSH2CKL | OSC1 ↑ to CLKO ↓ | | — | 75 | 200 | ns | **(1)** |
| 11 | TOSH2CKH | OSC1 ↑ to CLKO ↑ | | — | 75 | 200 | ns | **(1)** |
| 12 | TCKR | CLKO Rise Time | | — | 35 | 100 | ns | **(1)** |
| 13 | TCKF | CLKO Fall Time | | — | 35 | 100 | ns | **(1)** |
| 14 | TCKL2IOV | CLKO ↓ to Port Out Valid | | — | — | 0.5 TCY + 20 | ns | **(1)** |
| 15 | TIOV2CKH | Port In Valid before CLKO ↑ | | 0.25 TCY + 25 | — | — | ns | **(1)** |
| 16 | TCKH2IOI | Port In Hold after CLKO ↑ | | 0 | — | — | ns | **(1)** |
| 17 | TOSH2IOV | OSC1↑ (Q1 cycle) to Port Out Valid | | — | 50 | 150 | ns | |
| 18 | TOSH2IOI | OSC1↑ (Q2 cycle) to Port Input Invalid (I/O in hold time) | PIC18**F**XX20 | 100 | — | — | ns | |
| 18A | | | PIC18**LF**XX20 | 200 | — | — | ns | |
| 19 | TIOV2OSH | Port Input Valid to OSC1↑ (I/O in setup time) | | 0 | — | — | ns | |
| 20 | TIOR | Port Output Rise Time | PIC18**F**XX20 | — | 10 | 25 | ns | |
| 20A | | | PIC18**LF**XX20 | — | — | 60 | ns | |
| 21 | TIOF | Port Output Fall Time | PIC18**F**XX20 | — | 10 | 25 | ns | |
| 21A | | | PIC18**LF**XX20 | — | — | 60 | ns | |

**Note 1:** Measurements are taken in RC mode, where CLKO output is 4 x TOSC.

**FIGURE 26-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**




**Note:** Refer to Figure 26-5 for load conditions.

**FIGURE 26-9: BROWN-OUT RESET TIMING**

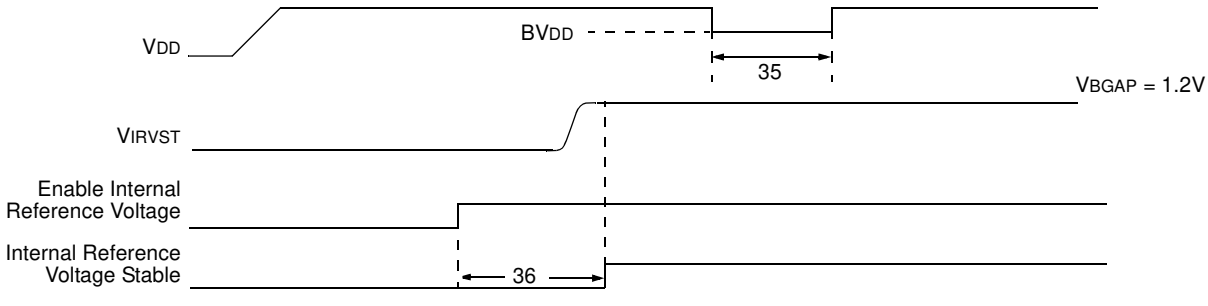
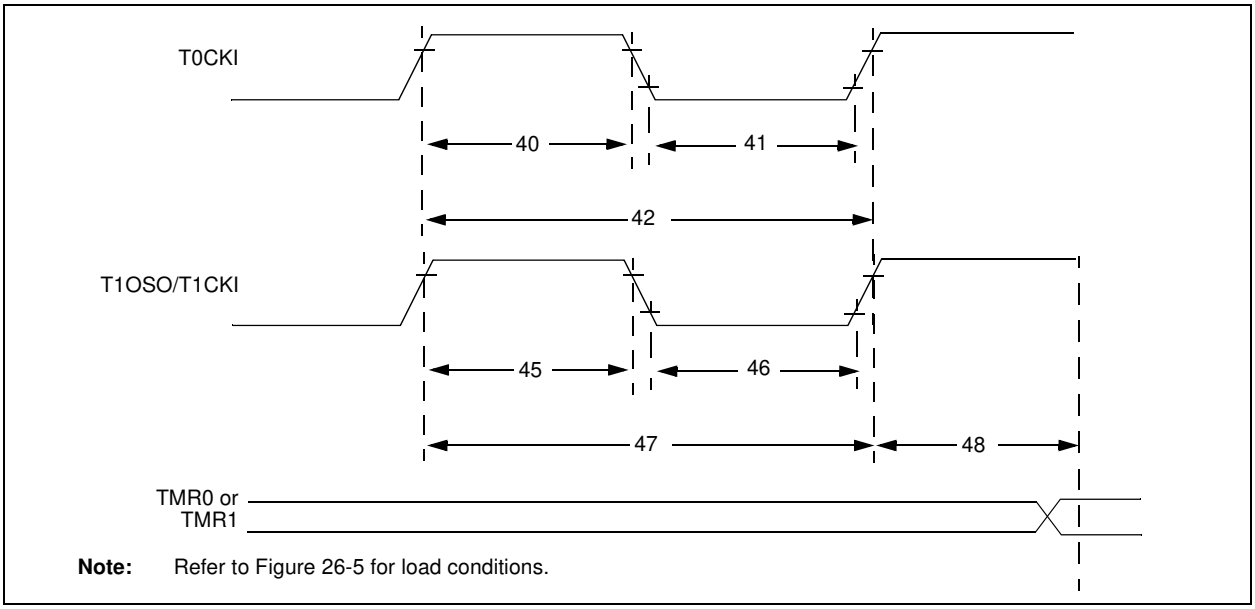



**TABLE 26-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS**

| Param. No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 30 | TMCL | $\overline{\text{MCLR}}$ Pulse Width (low) | 2 | — | — | µs | |
| 31 | TWDT | Watchdog Timer Time-out Period (no postscaler) | 3.48 | 4.00 | 4.71 | ms | |
| 32 | TOST | Oscillation Start-up Timer Period | 1024 TOSC | — | 1024 TOSC | — | TOSC = OSC1 period |
| 33 | TPWRT | Power-up Timer Period | 57.0 | 65.5 | 77.2 | ms | |
| 34 | TIOZ | I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset | — | 2 | — | µs | |
| 35 | TBOR | Brown-out Reset Pulse Width | 200 | — | — | µs | VDD ≤ BVDD (see D005) |
| 36 | TIVRST | Time for Internal Reference Voltage to become stable | — | 20 | 50 | µs | |
| 37 | TLVD | Low-Voltage Detect Pulse Width | 200 | — | — | µs | VDD ≤ VLVD |


© 2003 Microchip Technology Inc.

**FIGURE 26-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**

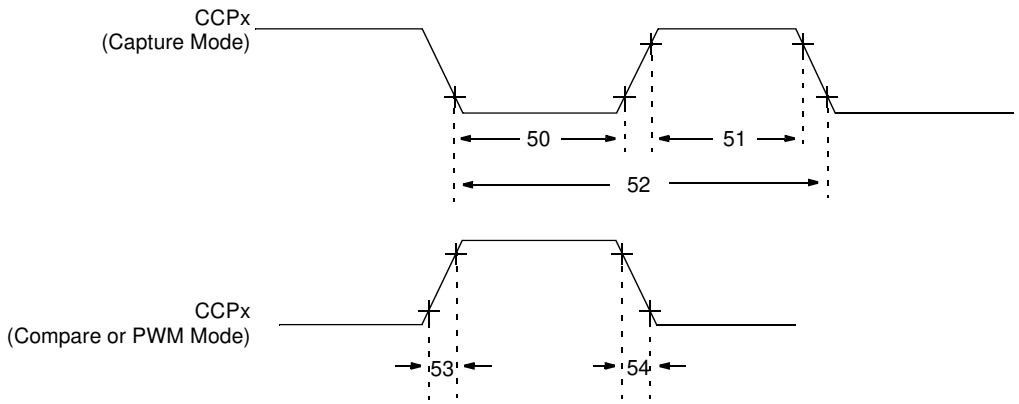



**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-11: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

| Param No. | Symbol | Characteristic | | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 40 | TT0H | T0CKI High Pulse Width | | No prescaler | 0.5 TCY + 20 | — | ns | |
| | | | | With prescaler | 10 | — | ns | |
| 41 | TT0L | T0CKI Low Pulse Width | | No prescaler | 0.5 TCY + 20 | — | ns | |
| | | | | With prescaler | 10 | — | ns | |
| 42 | TT0P | T0CKI Period | | No prescaler | TCY + 10 | — | ns | |
| | | | | With prescaler | Greater of: 20 ns or $\frac{T_{CY} + 40}{N}$ | — | ns | N = prescale value (1, 2, 4,..., 256) |
| 45 | TT1H | T1CKI High Time | Synchronous, no prescaler | | 0.5 TCY + 20 | — | ns | |
| | | | Synchronous, with prescaler | PIC18**F**XX20 | 10 | — | ns | |
| | | | | PIC18**LF**XX20 | 25 | — | ns | |
| | | | Asynchronous | PIC18**F**XX20 | 30 | — | ns | |
| | | | | PIC18**LF**XX20 | 50 | — | ns | |
| 46 | TT1L | T1CKI Low Time | Synchronous, no prescaler | | 0.5 TCY + 5 | — | ns | |
| | | | Synchronous, with prescaler | PIC18**F**XX20 | 10 | — | ns | |
| | | | | PIC18**LF**XX20 | 25 | — | ns | |
| | | | Asynchronous | PIC18**F**XX20 | 30 | — | ns | |
| | | | | PIC18**LF**XX20 | 50 | — | ns | |
| 47 | TT1P | T1CKI Input Period | Synchronous | | Greater of: 20 ns or $\frac{T_{CY} + 40}{N}$ | — | ns | N = prescale value (1, 2, 4, 8) |
| | | | Asynchronous | | 60 | — | ns | |
| | FT1 | T1CKI Oscillator Input Frequency Range | | | DC | 50 | kHz | |
| 48 | TCKE2TMRI | Delay from External T1CKI Clock Edge to Timer Increment | | | 2 TOSC | 7 TOSC | — | |

**FIGURE 26-11: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)**



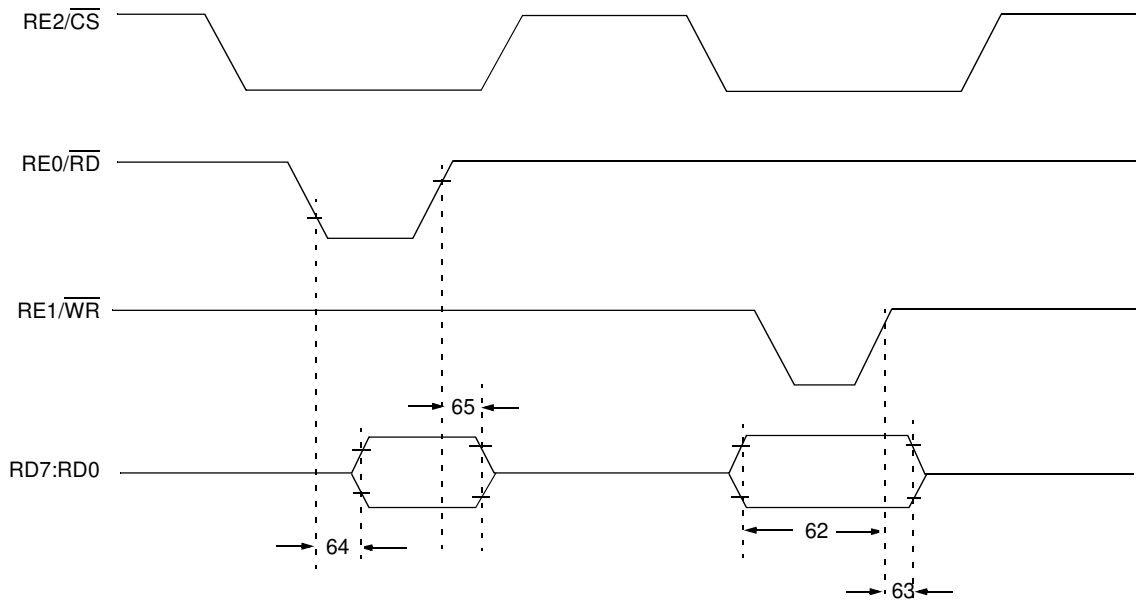

**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-12: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)**

| Param No. | Symbol | Characteristic | | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 50 | TccL | CCPx Input Low Time | No prescaler | | 0.5 TCY + 20 | — | ns | |
| | | | With prescaler | PIC18**F**XX20 | 10 | — | ns | |
| | | | | PIC18**LF**XX20 | 20 | — | ns | |
| 51 | TccH | CCPx Input High Time | No prescaler | | 0.5 TCY + 20 | — | ns | |
| | | | With prescaler | PIC18**F**XX20 | 10 | — | ns | |
| | | | | PIC18**LF**XX20 | 20 | — | ns | |
| 52 | TccP | CCPx Input Period | | | $\frac{3\ T_{CY} + 40}{N}$ | — | ns | N = prescale value (1,4 or 16) |
| 53 | TccR | CCPx Output Fall Time | | PIC18**F**XX20 | — | 25 | ns | |
| | | | | PIC18**LF**XX20 | — | 45 | ns | |
| 54 | TccF | CCPx Output Fall Time | | PIC18**F**XX20 | — | 25 | ns | |
| | | | | PIC18**LF**XX20 | — | 45 | ns | |


© 2003 Microchip Technology Inc.

**FIGURE 26-12: PARALLEL SLAVE PORT TIMING (PIC18F4X20)**

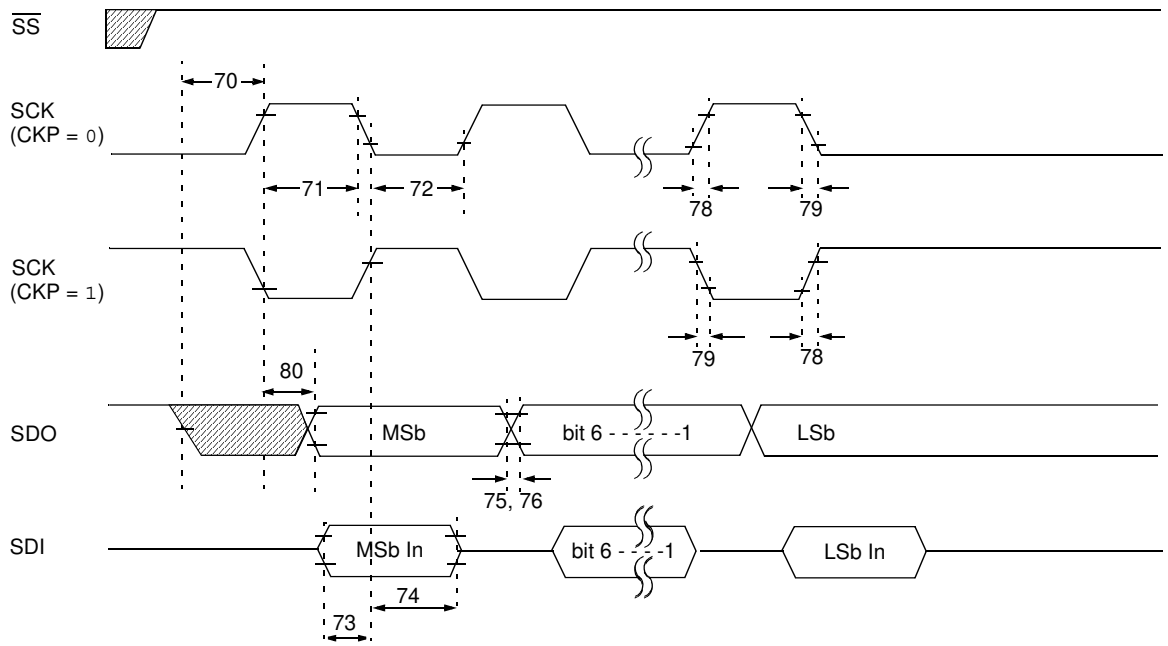



**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-13: PARALLEL SLAVE PORT REQUIREMENTS (PIC18F4X20)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 62 | TDTV2WRH | Data in valid before $\overline{WR}$ ↑ or $\overline{CS}$ ↑ (setup time) | | 20 | — | ns | |
| 63 | TWRH2DTI | $\overline{WR}$ ↑ or $\overline{CS}$ ↑ to data–in invalid (hold time) | PIC18**F**XX20 | 20 | — | ns | |
| | | | PIC18**LF**XX20 | 35 | — | ns | |
| 64 | TRDL2DTV | $\overline{RD}$ ↓ and $\overline{CS}$ ↓ to data–out valid | | — | 80 | ns | |
| 65 | TRDH2DTI | $\overline{RD}$ ↑ or $\overline{CS}$ ↓ to data–out invalid | | 10 | 30 | ns | |
| 66 | TIBFINH | Inhibit of the IBF flag bit being cleared from $\overline{WR}$ ↑ or $\overline{CS}$ ↑ | | — | 3 TCY | | |

**FIGURE 26-13: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)**




**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-14: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)**

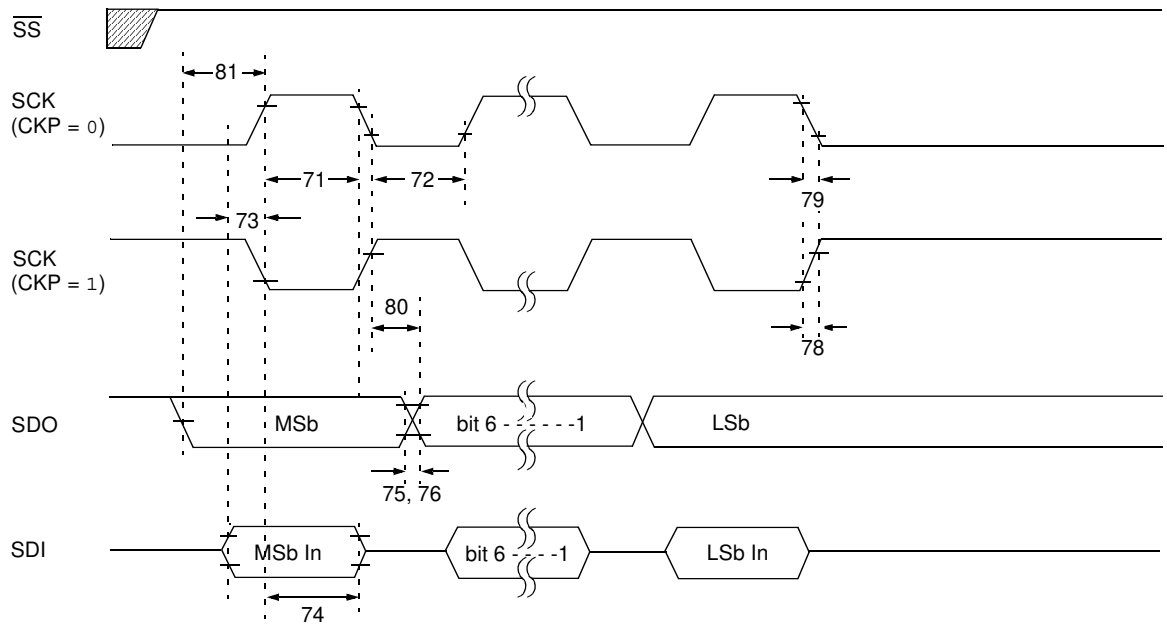
| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 70 | TssL2scH, TssL2scL | $\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input | | TCY | — | ns | |
| 71 | TscH | SCK Input High Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 71A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 72 | TscL | SCK Input Low Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 72A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 73 | TdiV2scH, TdiV2scL | Setup Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 73A | Tb2b | Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2 | | 1.5 TCY + 40 | — | ns | **(Note 2)** |
| 74 | TscH2diL, TscL2diL | Hold Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 75 | TdoR | SDO Data Output Rise Time | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | — | 45 | ns | |
| 76 | TdoF | SDO Data Output Fall Time | | — | 25 | ns | |
| 78 | TscR | SCK Output Rise Time (Master mode) | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | — | 45 | ns | |
| 79 | TscF | SCK Output Fall Time (Master mode) | | — | 25 | ns | |
| 80 | TscH2doV, TscL2doV | SDO Data Output Valid after SCK Edge | PIC18**F**XX20 | — | 50 | ns | |
| | | | PIC18**LF**XX20 | — | 100 | ns | |

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

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


© 2003 Microchip Technology Inc.

**FIGURE 26-14: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)**




**Note:** Refer to Figure 26-5 for load conditions.

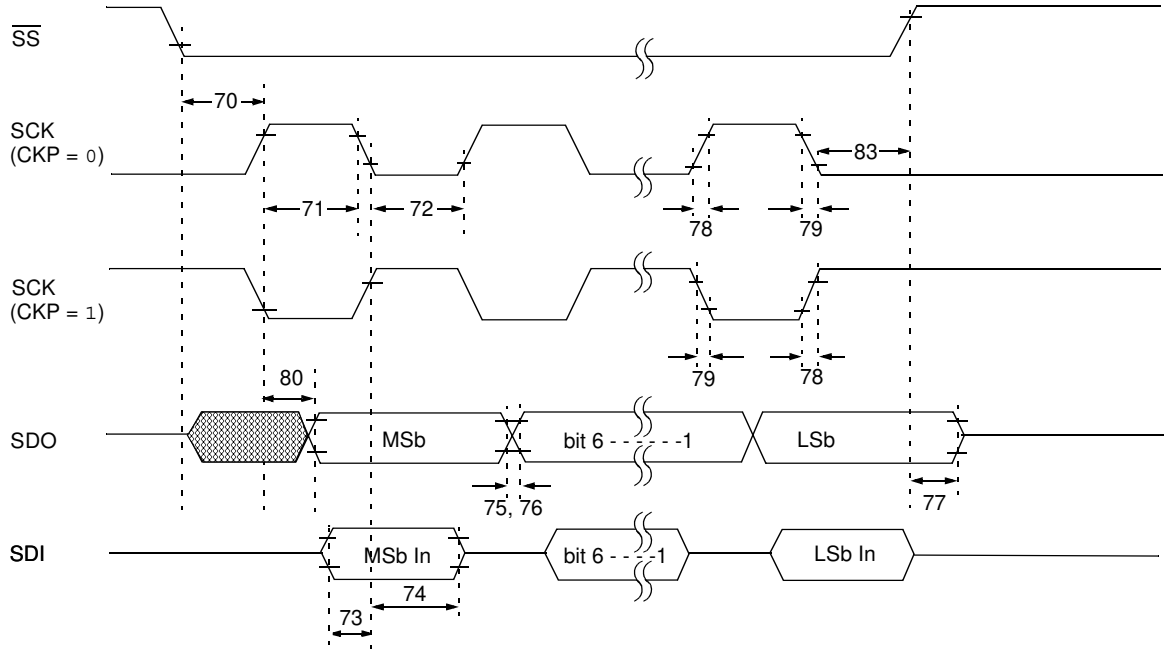
**TABLE 26-15: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 71 | TscH | SCK Input High Time (Slave mode) | Continuous | 1.25 Tcy + 30 | — | ns | |
| 71A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 72 | TscL | SCK Input Low Time (Slave mode) | Continuous | 1.25 Tcy + 30 | — | ns | |
| 72A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 73 | TdiV2scH, TdiV2scL | Setup Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 73A | Tb2b | Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2 | | 1.5 Tcy + 40 | — | ns | **(Note 2)** |
| 74 | TscH2diL, TscL2diL | Hold Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 75 | TdoR | SDO Data Output Rise Time | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | | 45 | ns | |
| 76 | TdoF | SDO Data Output Fall Time | | — | 25 | ns | |
| 78 | TscR | SCK Output Rise Time (Master mode) | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | | 45 | ns | |
| 79 | TscF | SCK Output Fall Time (Master mode) | | — | 25 | ns | |
| 80 | TscH2doV, TscL2doV | SDO Data Output Valid after SCK Edge | PIC18**F**XX20 | — | 50 | ns | |
| | | | PIC18**LF**XX20 | | 100 | ns | |
| 81 | TdoV2scH, TdoV2scL | SDO Data Output Setup to SCK Edge | | Tcy | — | ns | |

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

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

**FIGURE 26-15: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)**




**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-16: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)**

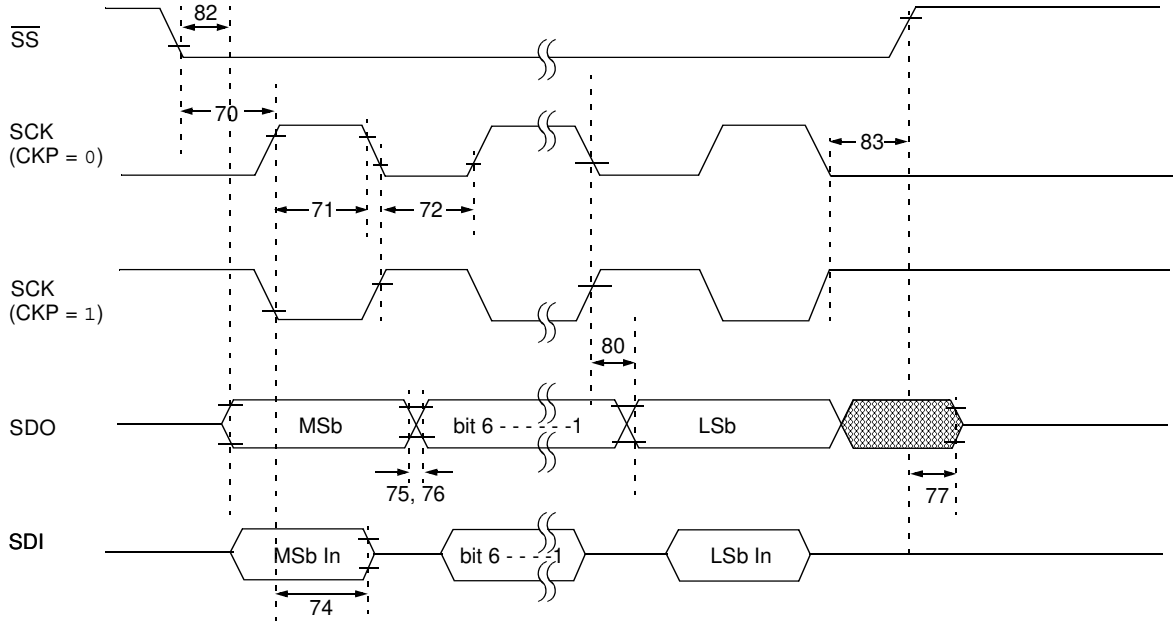
| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 70 | TssL2scH, TssL2scL | $\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input | | TCY | — | ns | |
| 71 | TscH | SCK Input High Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 71A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 72 | TscL | SCK Input Low Time (Slave mode) | Continuous | 1.25 TCY + 30 | — | ns | |
| 72A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 73 | TdiV2scH, TdiV2scL | Setup Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 73A | Tb2b | Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2 | | 1.5 TCY + 40 | — | ns | **(Note 2)** |
| 74 | TscH2diL, TscL2diL | Hold Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 75 | TdoR | SDO Data Output Rise Time | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | | 45 | ns | |
| 76 | TdoF | SDO Data Output Fall Time | | — | 25 | ns | |
| 77 | TssH2doZ | $\overline{SS}$ ↑ to SDO Output High-Impedance | | 10 | 50 | ns | |
| 78 | TscR | SCK Output Rise Time (Master mode) | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | | 45 | ns | |
| 79 | TscF | SCK Output Fall Time (Master mode) | | — | 25 | ns | |
| 80 | TscH2doV, TscL2doV | SDO Data Output Valid after SCK Edge | PIC18**F**XX20 | — | 50 | ns | |
| | | | PIC18**LF**XX20 | | 100 | ns | |
| 83 | TscH2ssH, TscL2ssH | $\overline{SS}$ ↑ after SCK Edge | | 1.5 TCY + 40 | — | ns | |

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

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


© 2003 Microchip Technology Inc.

**FIGURE 26-16: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)**




**Note:** Refer to Figure 26-5 for load conditions.

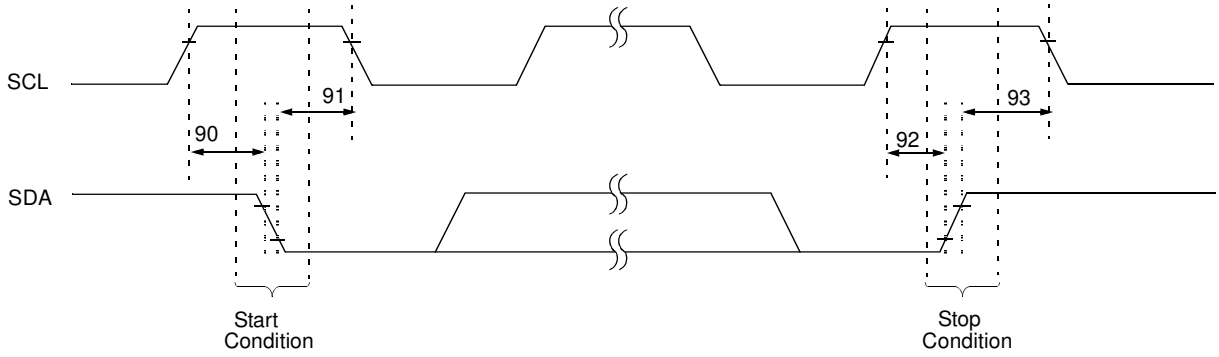
**TABLE 26-17: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)**

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 70 | TssL2scH, TssL2scL | $\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input | | Tcy | — | ns | |
| 71 | TscH | SCK Input High Time (Slave mode) | Continuous | 1.25 Tcy + 30 | — | ns | |
| 71A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 72 | TscL | SCK Input Low Time (Slave mode) | Continuous | 1.25 Tcy + 30 | — | ns | |
| 72A | | | Single Byte | 40 | — | ns | **(Note 1)** |
| 73A | Tb2b | Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2 | | 1.5 Tcy + 40 | — | ns | **(Note 2)** |
| 74 | TscH2diL, TscL2diL | Hold Time of SDI Data Input to SCK Edge | | 100 | — | ns | |
| 75 | TdoR | SDO Data Output Rise Time | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | | 45 | ns | |
| 76 | TdoF | SDO Data Output Fall Time | | — | 25 | ns | |
| 77 | TssH2doZ | $\overline{SS}$↑ to SDO Output High-Impedance | | 10 | 50 | ns | |
| 78 | TscR | SCK Output Rise Time (Master mode) | PIC18**F**XX20 | — | 25 | ns | |
| | | | PIC18**LF**XX20 | — | 45 | ns | |
| 79 | TscF | SCK Output Fall Time (Master mode) | | — | 25 | ns | |
| 80 | TscH2doV, TscL2doV | SDO Data Output Valid after SCK Edge | PIC18**F**XX20 | — | 50 | ns | |
| | | | PIC18**LF**XX20 | — | 100 | ns | |
| 82 | TssL2doV | SDO Data Output Valid after $\overline{SS}$ ↓ Edge | PIC18**F**XX20 | — | 50 | ns | |
| | | | PIC18**LF**XX20 | — | 100 | ns | |
| 83 | TscH2ssH, TscL2ssH | $\overline{SS}$ ↑ after SCK edge | | 1.5 Tcy + 40 | — | ns | |

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

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

**FIGURE 26-17: I²C BUS START/STOP BITS TIMING**

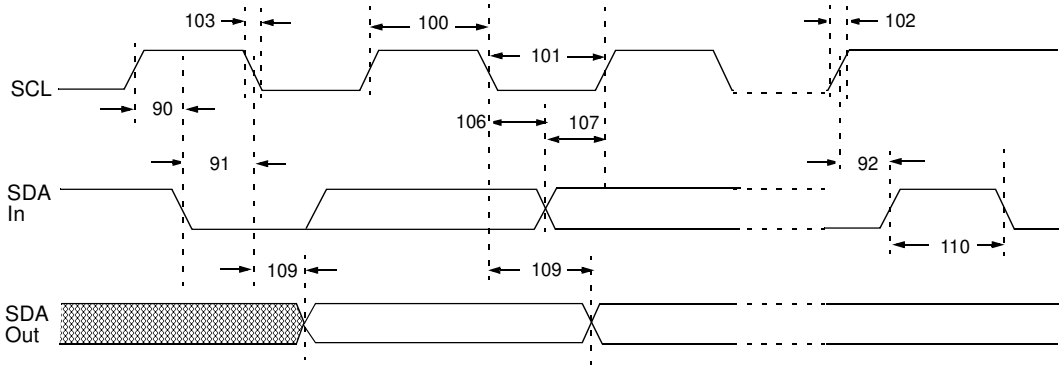



**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-18: I²C BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 90 | TSU:STA | Start condition | 100 kHz mode | 4700 | — | ns | Only relevant for Repeated Start condition |
| | | Setup time | 400 kHz mode | 600 | — | | |
| 91 | THD:STA | Start condition | 100 kHz mode | 4000 | — | ns | After this period, the first clock pulse is generated |
| | | Hold time | 400 kHz mode | 600 | — | | |
| 92 | TSU:STO | Stop condition | 100 kHz mode | 4700 | — | ns | |
| | | Setup time | 400 kHz mode | 600 | — | | |
| 93 | THD:STO | Stop condition | 100 kHz mode | 4000 | — | ns | |
| | | Hold time | 400 kHz mode | 600 | — | | |

**FIGURE 26-18: I²C BUS DATA TIMING**




**Note:** Refer to Figure 26-5 for load conditions.


© 2003 Microchip Technology Inc.

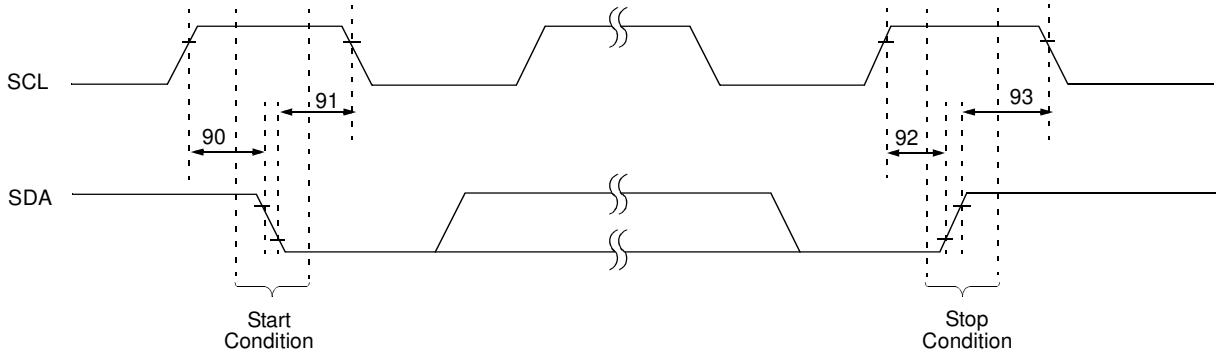
**TABLE 26-19: $I^2C$ BUS DATA REQUIREMENTS (SLAVE MODE)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 100 | THIGH | Clock High Time | 100 kHz mode | 4.0 | — | µs | PIC18FXX20 must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 0.6 | — | µs | PIC18FXX20 must operate at a minimum of 10 MHz |
| | | | SSP module | 1.5 TCY | — | | |
| 101 | TLOW | Clock Low Time | 100 kHz mode | 4.7 | — | µs | PIC18FXX20 must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 1.3 | — | µs | PIC18FXX20 must operate at a minimum of 10 MHz |
| | | | SSP module | 1.5 TCY | — | | |
| 102 | TR | SDA and SCL Rise Time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | CB is specified to be from 10 to 400 pF |
| 103 | TF | SDA and SCL Fall Time | 100 kHz mode | — | 300 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | CB is specified to be from 10 to 400 pF |
| 90 | TSU:STA | Start Condition Setup Time | 100 kHz mode | 4.7 | — | µs | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 91 | THD:STA | Start Condition Hold Time | 100 kHz mode | 4.0 | — | µs | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 106 | THD:DAT | Data Input Hold Time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | 0.9 | µs | |
| 107 | TSU:DAT | Data Input Setup Time | 100 kHz mode | 250 | — | ns | **(Note 2)** |
| | | | 400 kHz mode | 100 | — | ns | |
| 92 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | 4.7 | — | µs | |
| | | | 400 kHz mode | 0.6 | — | µs | |
| 109 | TAA | Output Valid from Clock | 100 kHz mode | — | 3500 | ns | **(Note 1)** |
| | | | 400 kHz mode | — | — | ns | |
| 110 | TBUF | Bus Free Time | 100 kHz mode | 4.7 | — | µs | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | µs | |
| D102 | CB | Bus Capacitive Loading | | — | 400 | pF | |

**Note 1:** As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

**2:** A fast mode $I^2C$ bus device can be used in a standard mode $I^2C$ bus system but the requirement, $T_{SU:DAT} \geq 250$ ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line, $T_R$ max. + $T_{SU:DAT}$ = 1000 + 250 = 1250 ns (according to the standard mode $I^2C$ bus specification), before the SCL line is released.

**FIGURE 26-19: MASTER SSP I²C BUS START/STOP BITS TIMING WAVEFORMS**



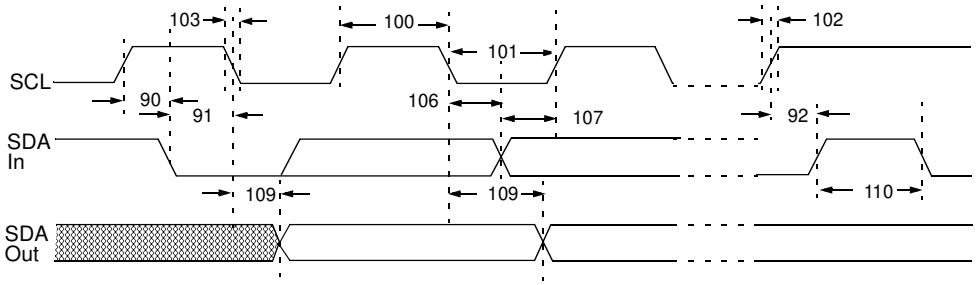

**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-20: MASTER SSP I²C BUS START/STOP BITS REQUIREMENTS**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 90 | TSU:STA | Start condition Setup time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | | |
| 91 | THD:STA | Start condition Hold time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | | |
| 92 | TSU:STO | Stop condition Setup time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ns | |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | | |
| 93 | THD:STO | Stop condition Hold time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ns | |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | | |

**Note 1:** Maximum pin capacitance = 10 pF for all I²C pins.

**FIGURE 26-20: MASTER SSP I²C BUS DATA TIMING**




**Note:** Refer to Figure 26-5 for load conditions.


© 2003 Microchip Technology Inc.

**TABLE 26-21: MASTER SSP $I^2C$ BUS DATA REQUIREMENTS**

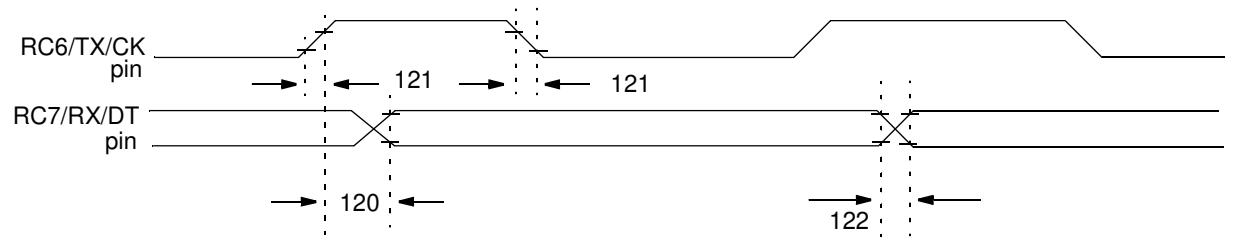
| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 100 | THIGH | Clock High Time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | ms | |
| 101 | TLOW | Clock Low Time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | ms | |
| 102 | TR | SDA and SCL Rise Time | 100 kHz mode | — | 1000 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | |
| | | | 1 MHz mode(1) | — | 300 | ns | |
| 103 | TF | SDA and SCL Fall Time | 100 kHz mode | — | 300 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | |
| | | | 1 MHz mode(1) | — | 100 | ns | |
| 90 | TSU:STA | Start Condition Setup Time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ms | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | ms | |
| 91 | THD:STA | Start Condition Hold Time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ms | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | ms | |
| 106 | THD:DAT | Data Input Hold Time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | 0.9 | ms | |
| | | | 1 MHz mode(1) | TBD | — | ns | |
| 107 | TSU:DAT | Data Input Setup Time | 100 kHz mode | 250 | — | ns | **(Note 2)** |
| | | | 400 kHz mode | 100 | — | ns | |
| | | | 1 MHz mode(1) | TBD | — | ns | |
| 92 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 400 kHz mode | 2(TOSC)(BRG + 1) | — | ms | |
| | | | 1 MHz mode(1) | 2(TOSC)(BRG + 1) | — | ms | |
| 109 | TAA | Output Valid from Clock | 100 kHz mode | — | 3500 | ns | |
| | | | 400 kHz mode | — | 1000 | ns | |
| | | | 1 MHz mode(1) | — | — | ns | |
| 110 | TBUF | Bus Free Time | 100 kHz mode | 4.7 | — | ms | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | ms | |
| | | | 1 MHz mode(1) | TBD | — | ms | |
| D102 | CB | Bus Capacitive Loading | | — | 400 | pF | |

**Note 1:** Maximum pin capacitance = 10 pF for all $I^2C$ pins.

**2:** A fast mode $I^2C$ bus device can be used in a standard mode $I^2C$ bus system, but parameter #107 ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line, parameter #102 + parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCL line is released.

© 2003 Microchip Technology Inc.

**FIGURE 26-21: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**

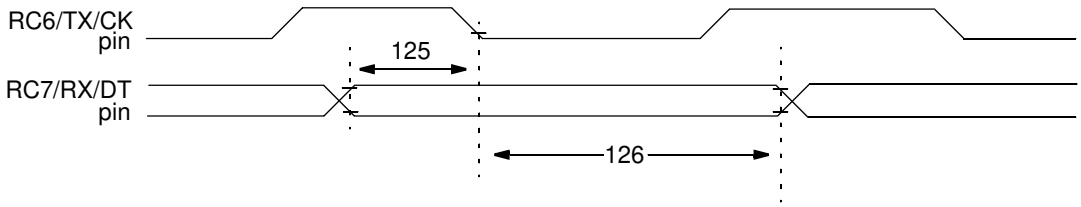



**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-22: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 120 | TCKH2DTV | SYNC XMIT (MASTER & SLAVE) Clock High to Data Out Valid | PIC18**F**XX20 | — | 40 | ns | |
| | | | PIC18**LF**XX20 | — | 100 | ns | |
| 121 | TCKRF | Clock Out Rise Time and Fall Time (Master mode) | PIC18**F**XX20 | — | 20 | ns | |
| | | | PIC18**LF**XX20 | — | 50 | ns | |
| 122 | TDTRF | Data Out Rise Time and Fall Time | PIC18**F**XX20 | — | 20 | ns | |
| | | | PIC18**LF**XX20 | — | 50 | ns | |

**FIGURE 26-22: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**




**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-23: USART SYNCHRONOUS RECEIVE REQUIREMENTS**

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|
| 125 | TDTV2CKL | SYNC RCV (MASTER & SLAVE) Data Hold before CK ↓ (DT hold time) | 10 | — | ns | |
| 126 | TCKL2DTL | Data Hold after CK ↓ (DT hold time) | 15 | — | ns | |


© 2003 Microchip Technology Inc.

**TABLE 26-24: A/D CONVERTER CHARACTERISTICS: PIC18F2220/2320/4220/4320 (INDUSTRIAL)**
**PIC18F2220/2320/4220/4320 (EXTENDED)**
**PIC18LF2220/2320/4220/4320 (INDUSTRIAL)**

| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| A01 | NR | Resolution | | — | — | 10 | bit | ΔVREF ≥ 3.0V |
| A03 | EIL | Integral Linearity Error | | — | — | <±1 | LSb | ΔVREF ≥ 3.0V |
| A04 | EDL | Differential Linearity Error | | — | — | <±1 | LSb | ΔVREF ≥ 3.0V |
| A06 | EOFF | Offset Error | | — | — | <±1 | LSb | ΔVREF ≥ 3.0V |
| A07 | EGN | Gain Error | | — | — | <±1 | LSb | ΔVREF ≥ 3.0V |
| A10 | — | Monotonicity | | guaranteed(2) | | | — | |
| A20 | ΔVREF | Reference Voltage Range (VREFH – VREFL) | | 3 | — | AVDD – AVSS | V | For 10-bit resolution |
| A21 | VREFH | Reference Voltage High | | AVSS + 3.0V | — | AVDD + 0.3V | V | For 10-bit resolution |
| A22 | VREFL | Reference Voltage Low | | AVSS – 0.3V | — | AVDD – 3.0V | V | For 10-bit resolution |
| A25 | VAIN | Analog Input Voltage | | VREFL | — | VREFH | V | |
| A28 | AVDD | Analog Supply Voltage | | VDD – 0.3 | — | VDD + 0.3 | V | Tie to VDD |
| A29 | AVSS | Analog Supply Voltage | | VSS – 0.3 | — | VSS + 0.3 | V | Tie to VSS |
| A30 | ZAIN | Recommended Impedance of Analog Voltage Source | | — | — | 2.5(4) | kΩ | |
| A40 | IAD | A/D Current from VDD | PIC18**F**XX20 | — | — | 180(5) | μA | Average current during conversion(1) |
| | | | PIC18**LF**XX20 | — | — | 90(5) | μA | |
| A50 | IREF | VREF Input Current (3) | | —<br>— | —<br>— | ±5(5)<br>±150(5) | μA<br>μA | During VAIN acquisition.<br>During A/D conversion cycle. |

**Note 1:** When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

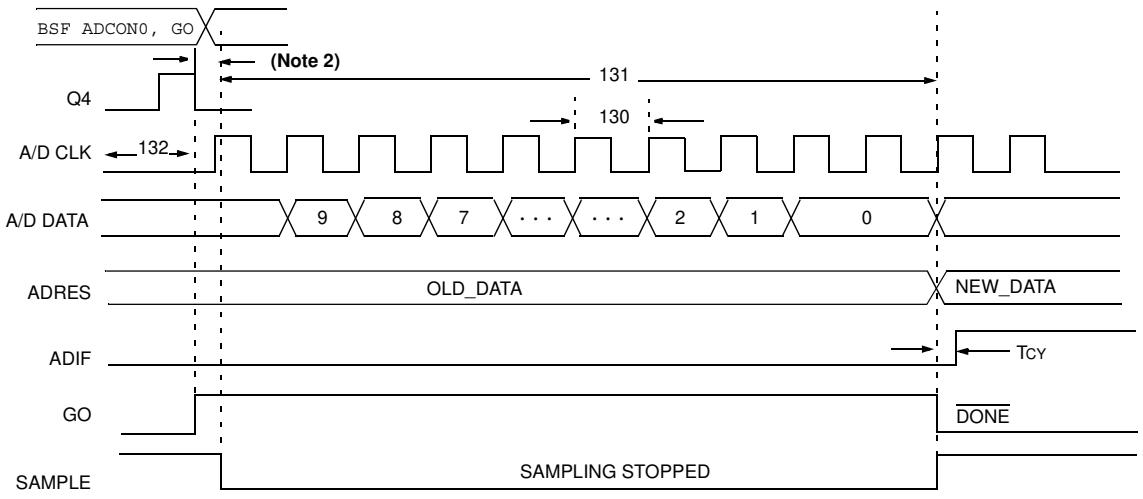
**2:** The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

**3:** VREFH current is from RA3/AN3/VREF+ pin or AVDD, whichever is selected as the VREFH source. VREFL current is from RA2/AN2/VREF- pin or AVSS, whichever is selected as the VREFL source.

**4:** Assume quiet environment. If adjacent pins have high-frequency signals (analog or digital), ZAIN may need to be reduced to as low as 1 kΩ to fight crosstalk effects.

**5:** For guidance only.

**FIGURE 26-23: A/D CONVERSION TIMING**




**Note 1:** If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

**2:** This is a minimal RC delay (typically 100 ns), which also disconnects the holding capacitor from the analog input.

**TABLE 26-25: A/D CONVERSION REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 130 | TAD | A/D Clock Period | PIC18**F**XX20 | 1.6 | 20(2) | µs | TOSC based, VREF ≥ 3.0V |
| | | | PIC18**LF**XX20 | 3.0 | 20(2) | µs | TOSC based, VREF full range |
| | | | PIC18**F**XX20 | 2.0 | 6.0 | µs | A/D RC mode |
| | | | PIC18**LF**XX20 | 3.0 | 9.0 | µs | A/D RC mode |
| 131 | TCNV | Conversion Time (not including acquisition time)(1) | | 11 | 12 | TAD | |

**Note 1:** ADRES register may be read on the following TCY cycle.

**2:** The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.


© 2003 Microchip Technology Inc.

## 27.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

**Note:** The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.

"Typical" represents the mean of the distribution at 25°C. "Maximum" or "minimum" represents (mean + 3σ) or (mean – 3σ) respectively, where σ is a standard deviation, over the whole temperature range.

**FIGURE 27-1: TYPICAL IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, +25°C**

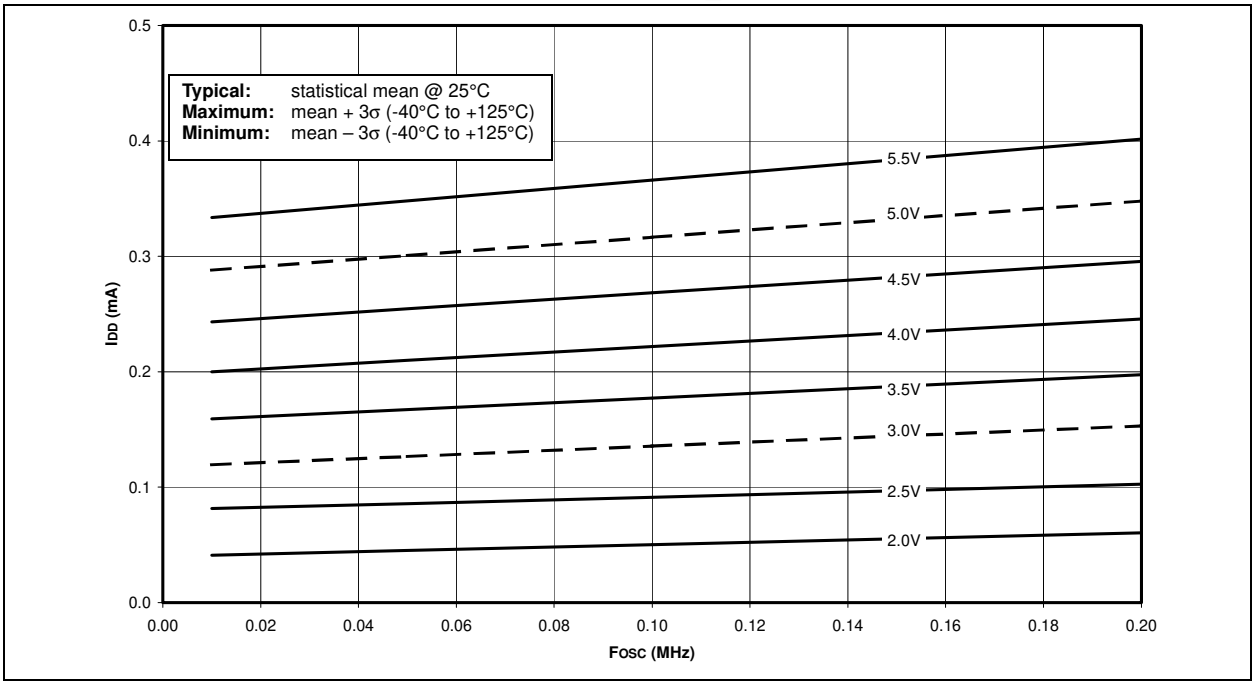



**FIGURE 27-2: MAXIMUM IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, -40°C TO +85°C**

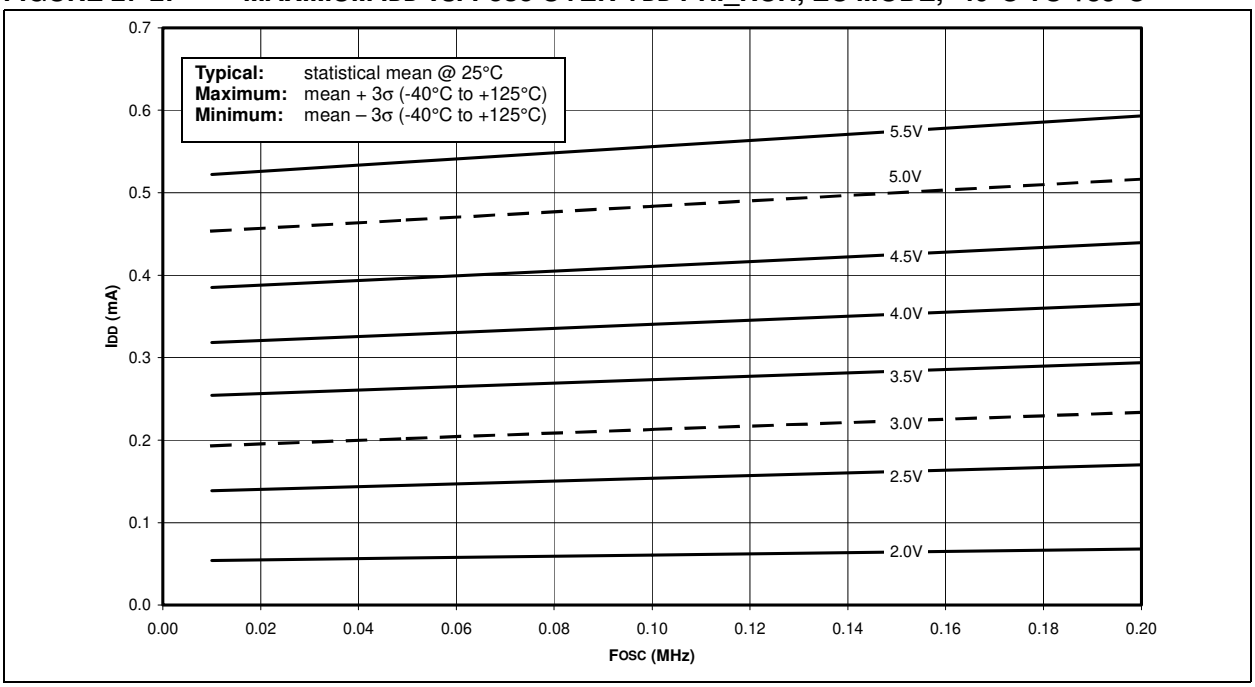


**FIGURE 27-3: MAXIMUM IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, -40°C TO +125°C**

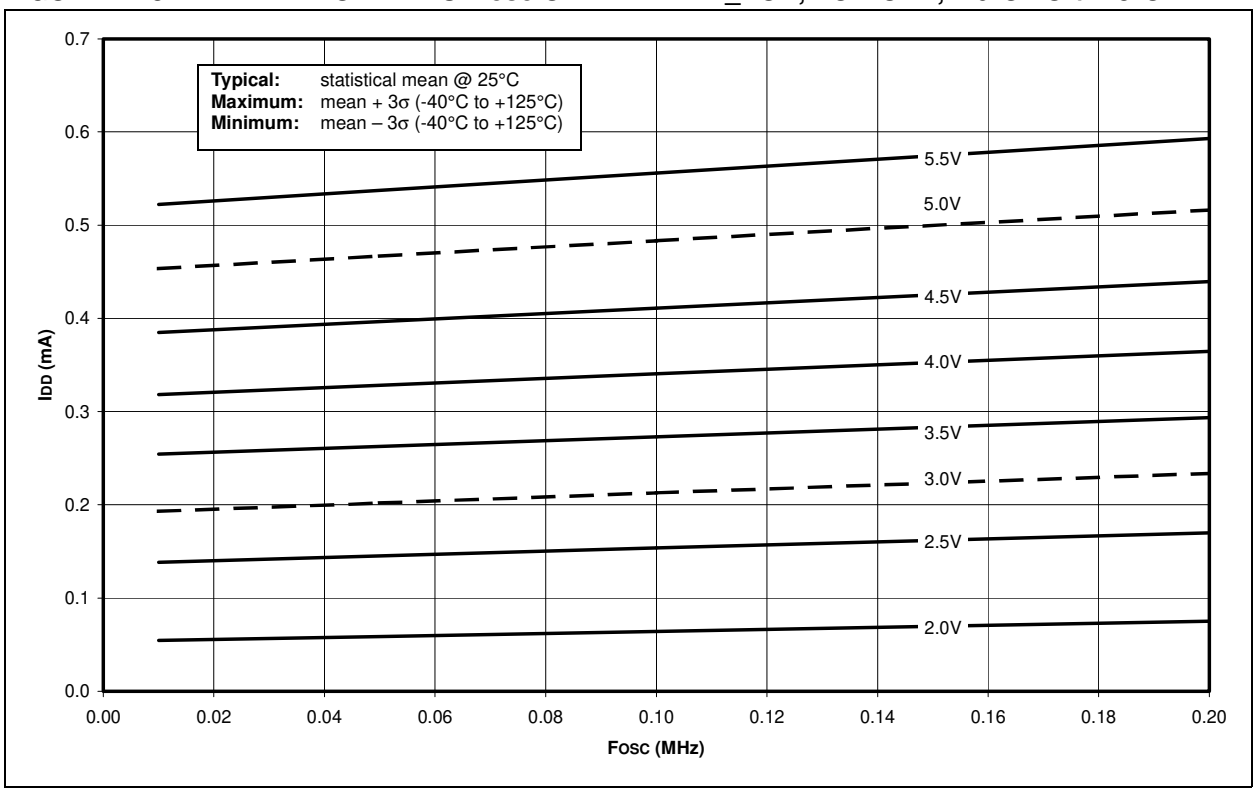
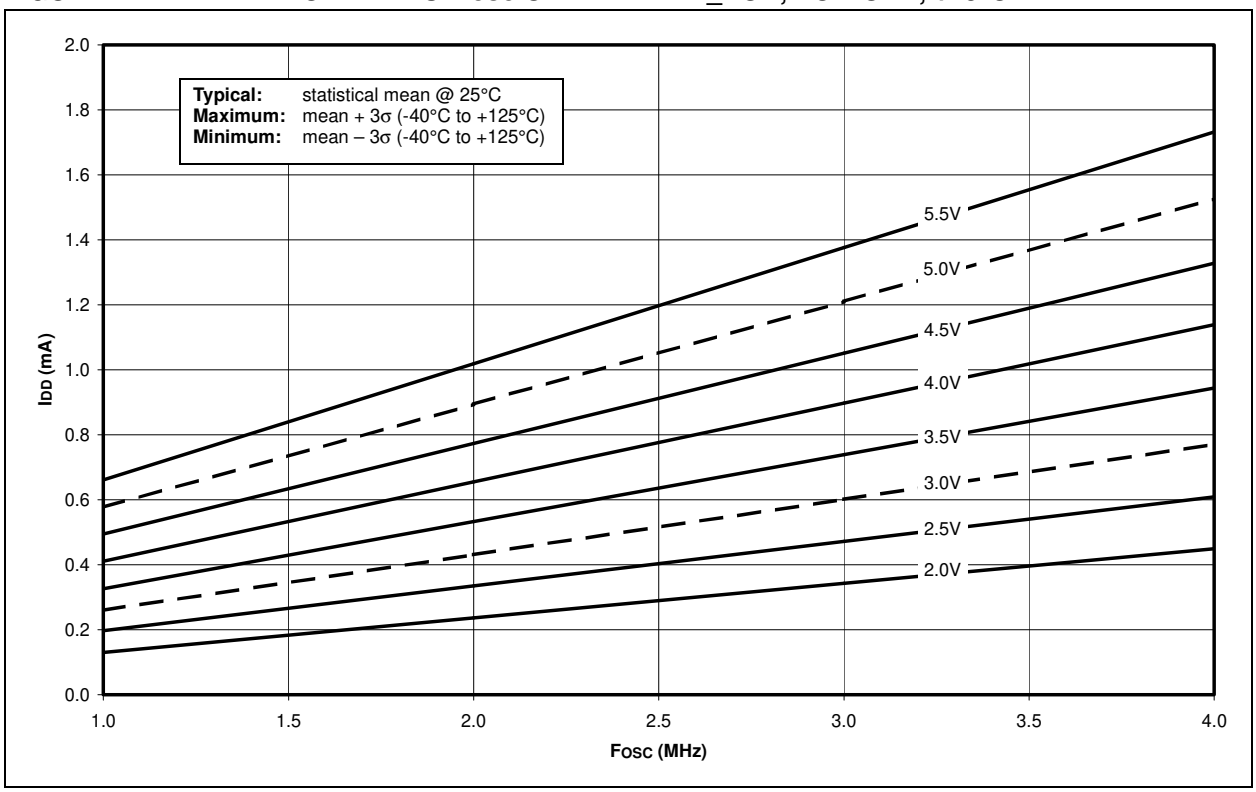



**FIGURE 27-4: TYPICAL IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, +25°C**





© 2003 Microchip Technology Inc.

**FIGURE 27-5: MAXIMUM IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, -40°C TO +125°C**

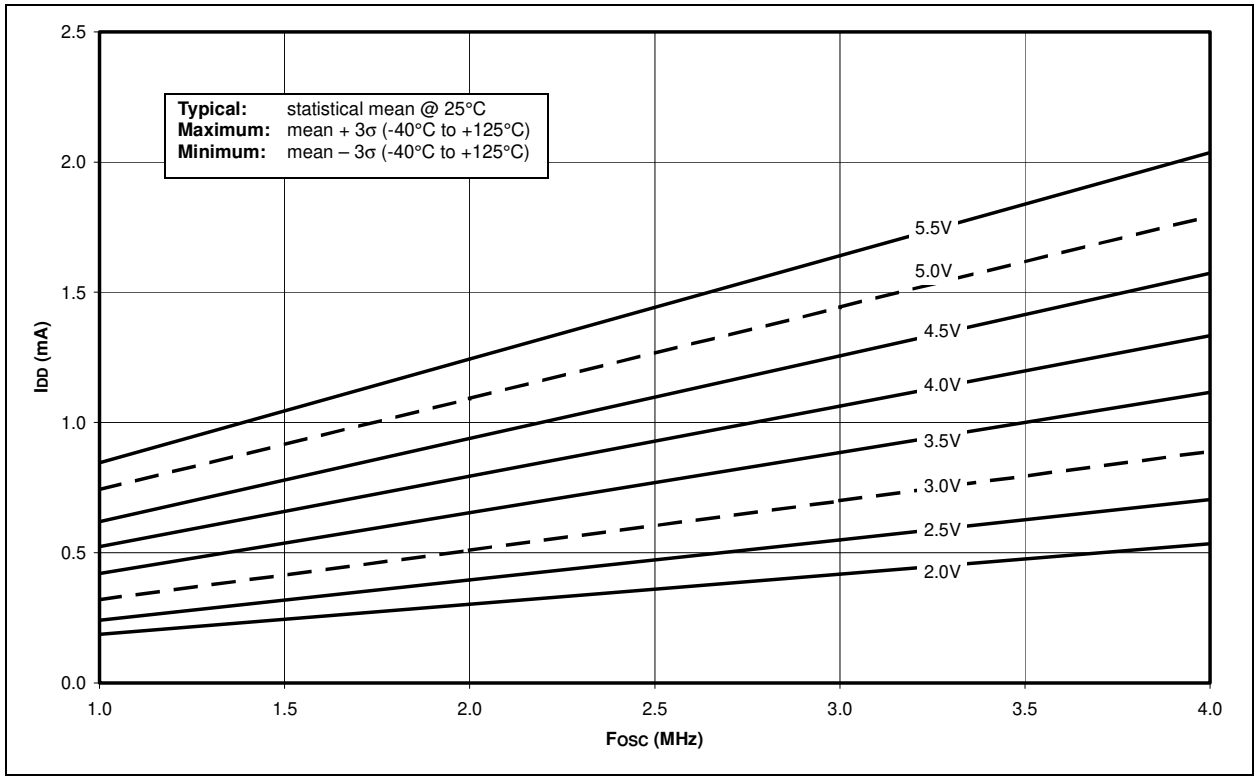



**FIGURE 27-6: TYPICAL IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, +25°C**

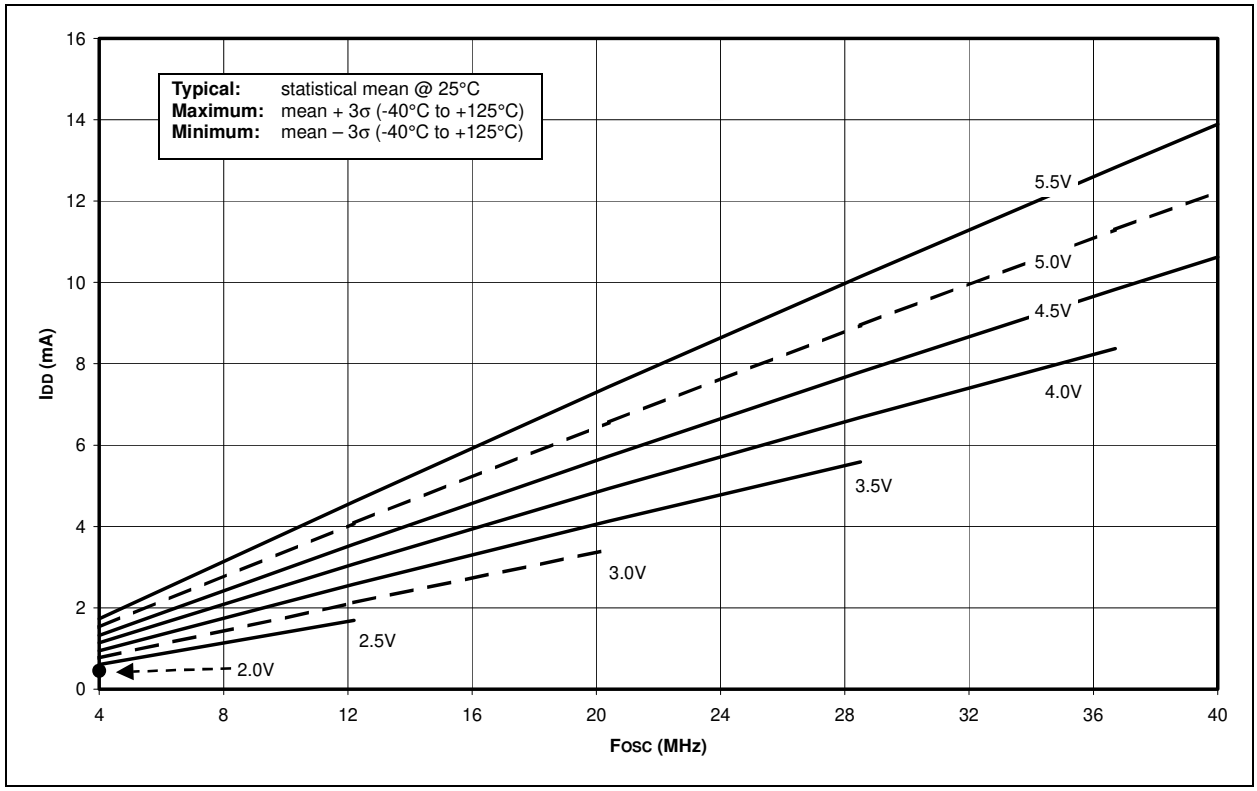


**FIGURE 27-7: MAXIMUM IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, -40°C TO +125°C**

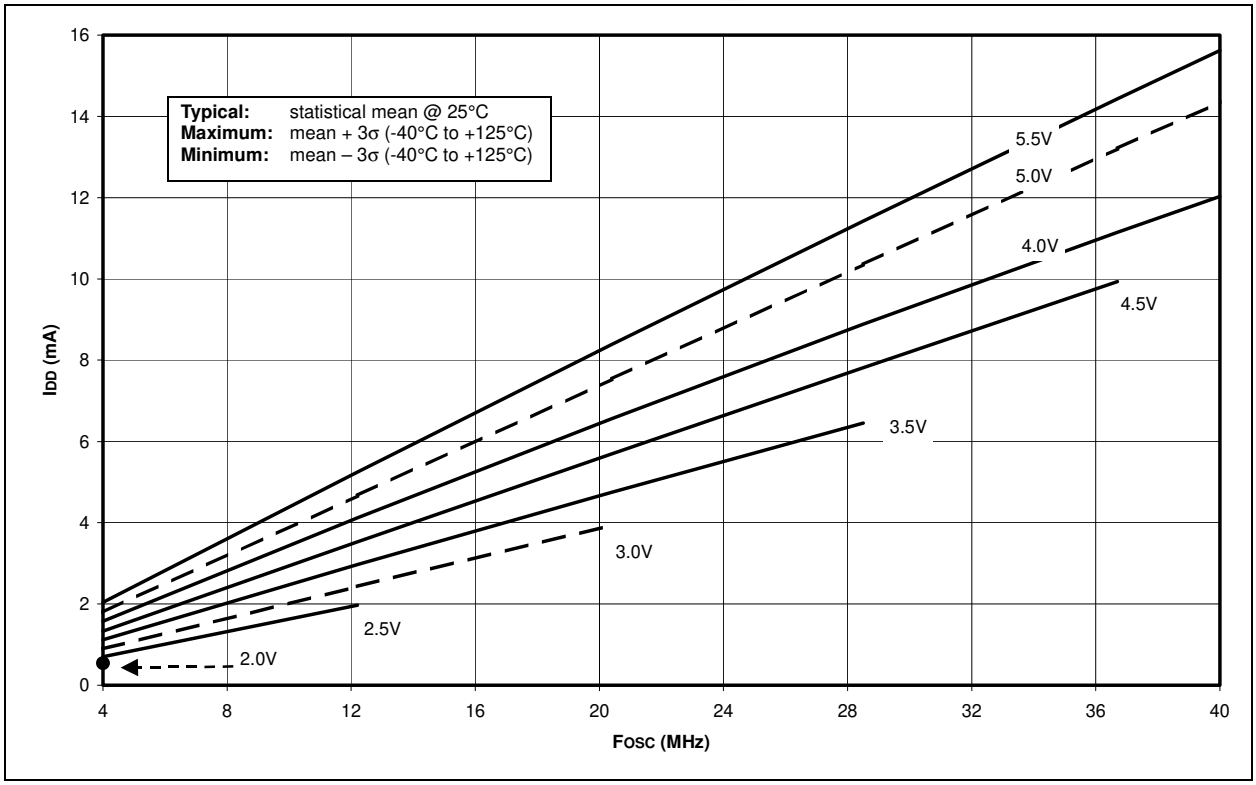
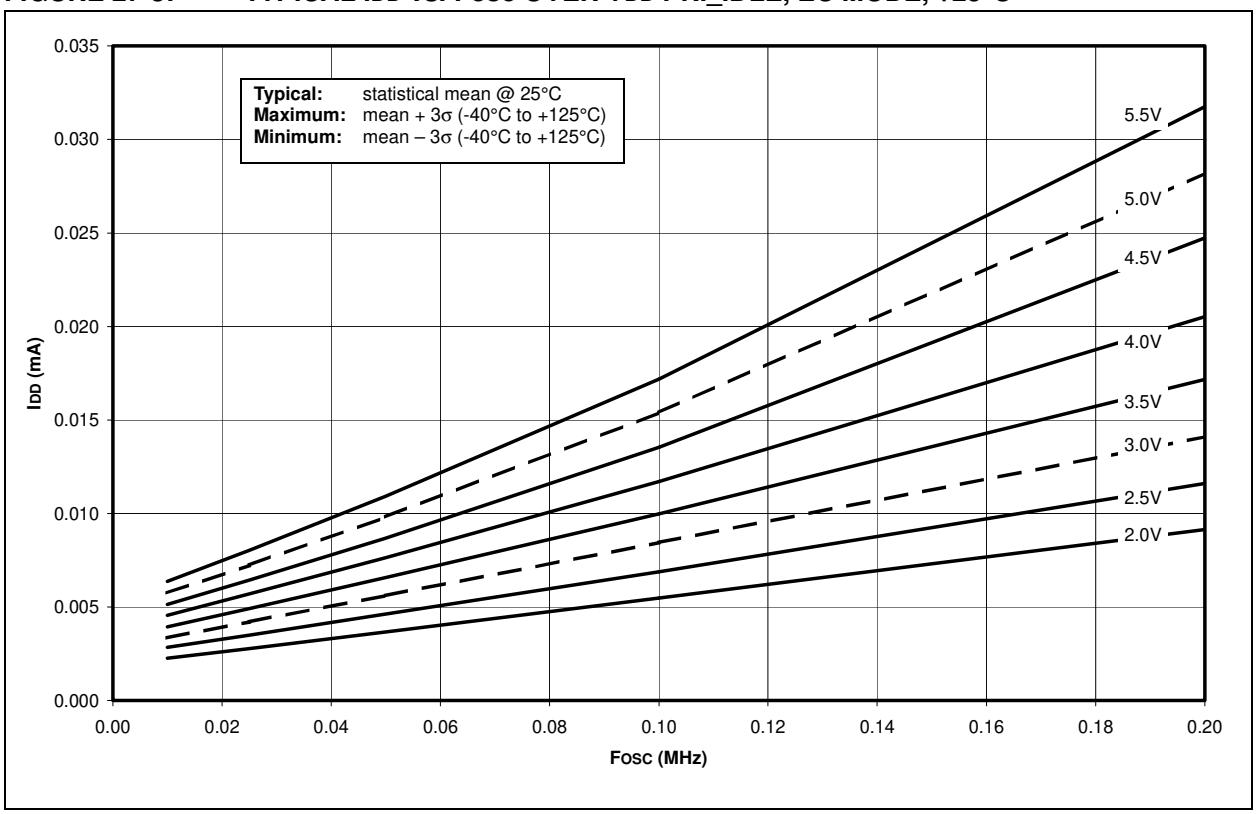



**FIGURE 27-8: TYPICAL IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, +25°C**





© 2003 Microchip Technology Inc.

**FIGURE 27-9: MAXIMUM IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, -40°C TO +85°C**

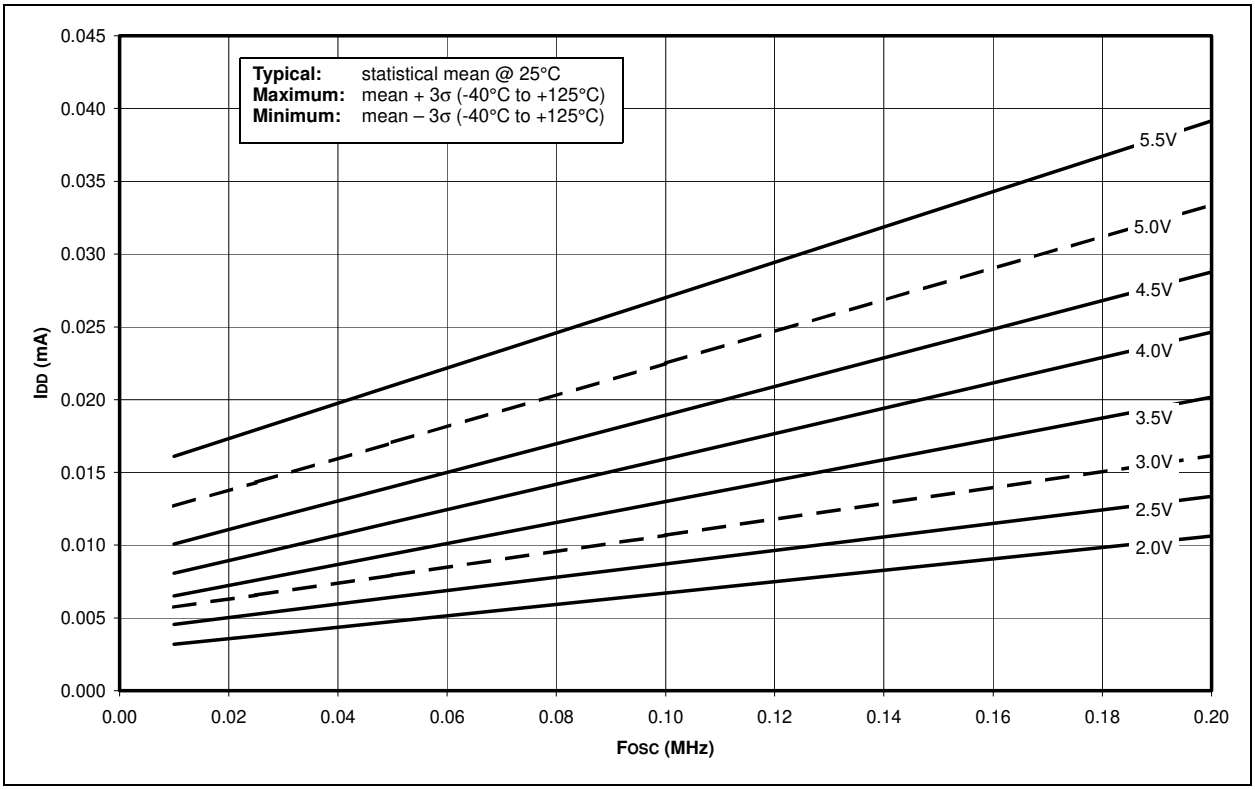



**FIGURE 27-10: MAXIMUM IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, -40°C TO +125°C**

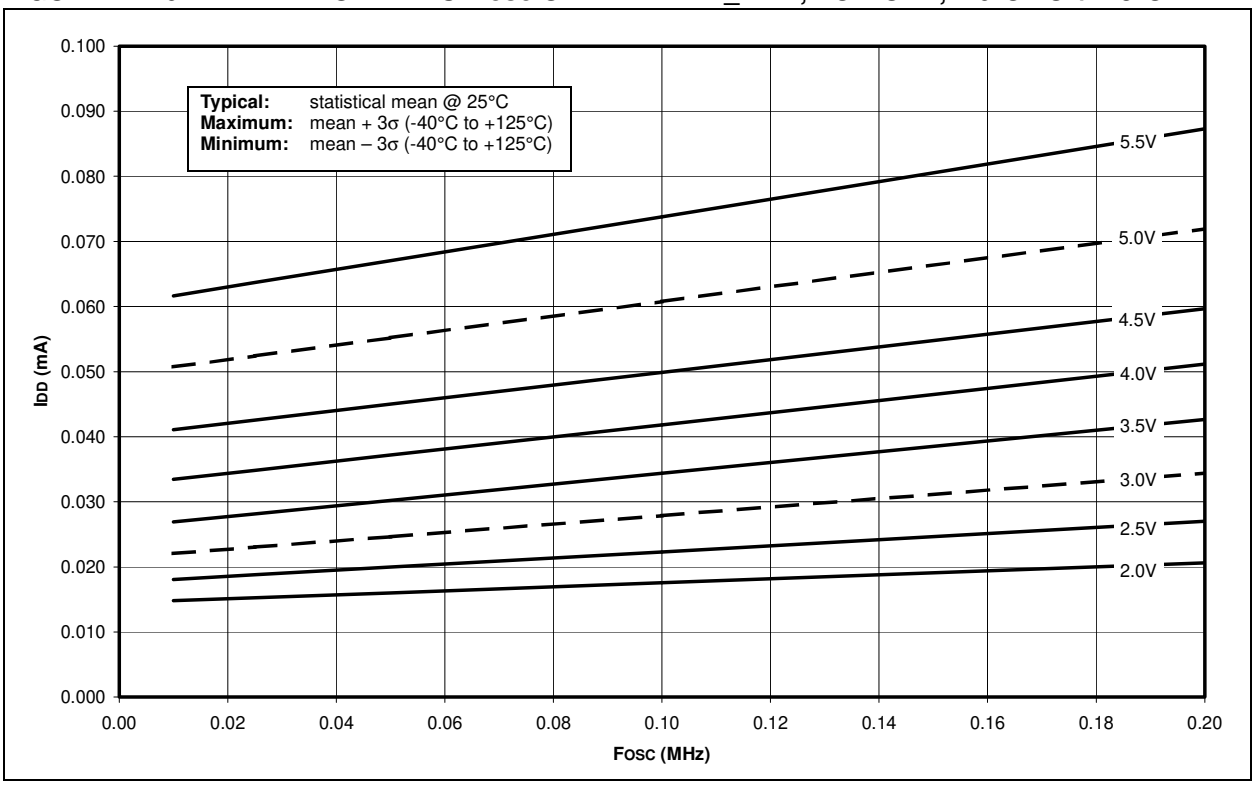


**FIGURE 27-11: TYPICAL IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, +25°C**

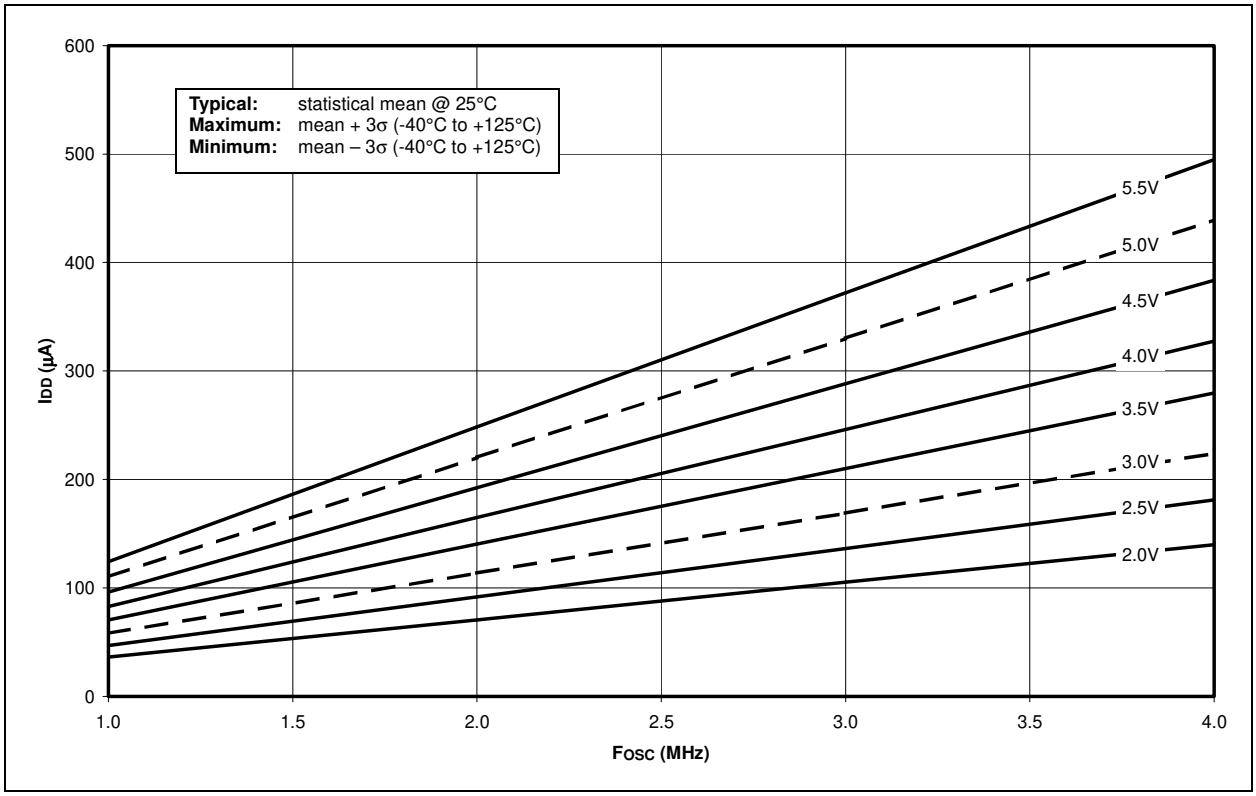
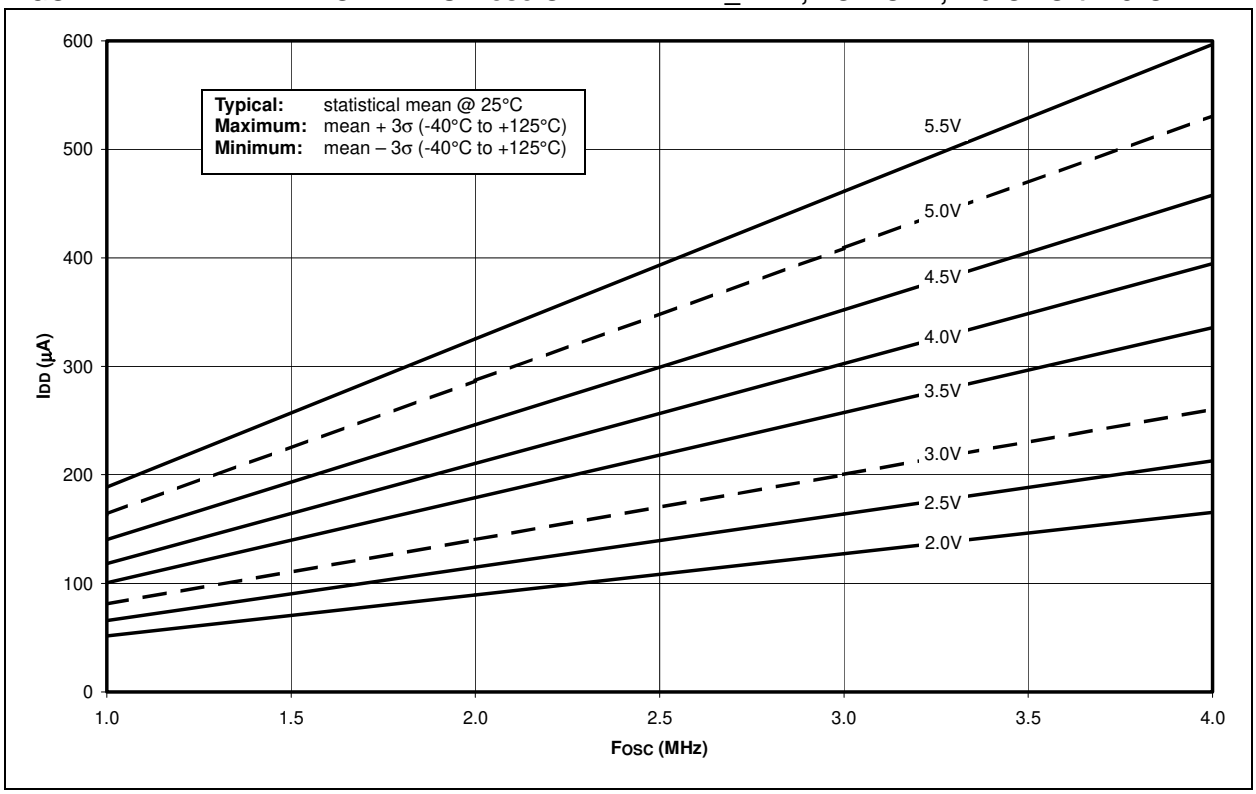



**FIGURE 27-12: MAXIMUM IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, -40°C TO +125°C**





© 2003 Microchip Technology Inc.

**FIGURE 27-13: TYPICAL IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, +25°C**

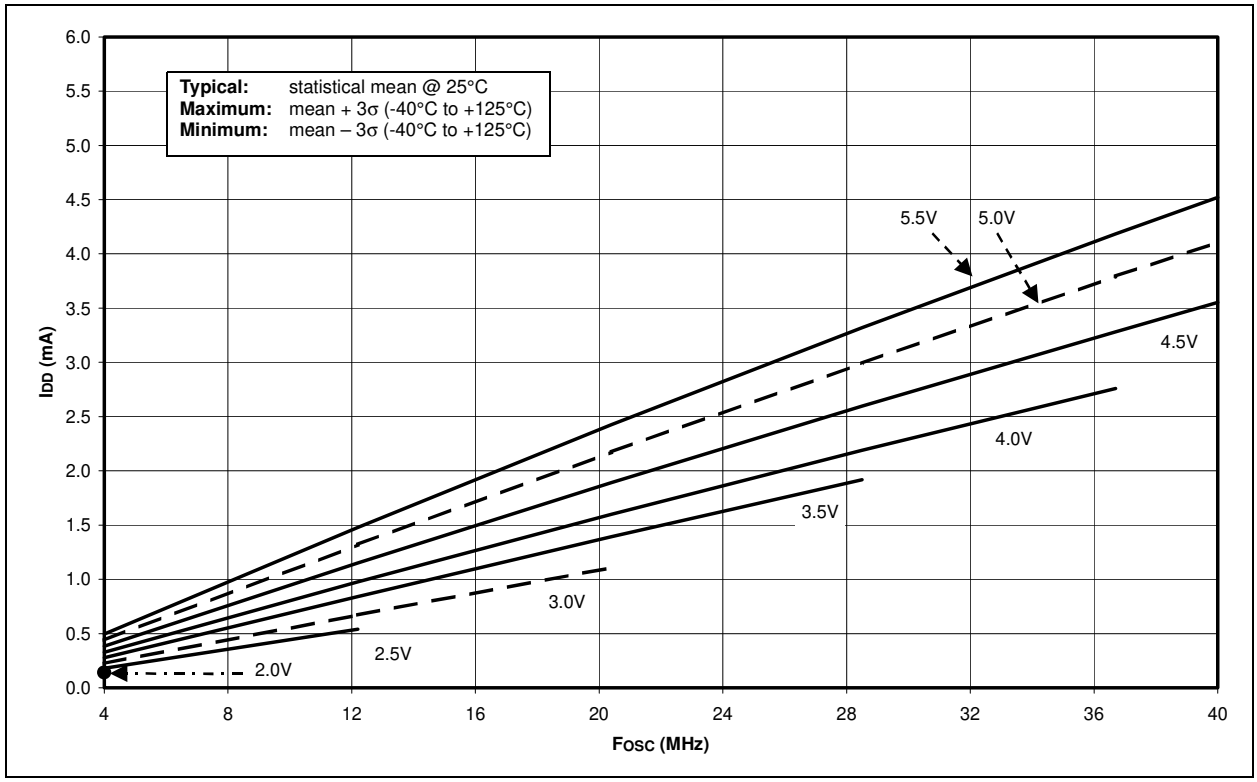



**FIGURE 27-14: MAXIMUM IDD vs. FOSC OVER VDD PRI_IDLE, EC MODE, -40°C TO +125°C**

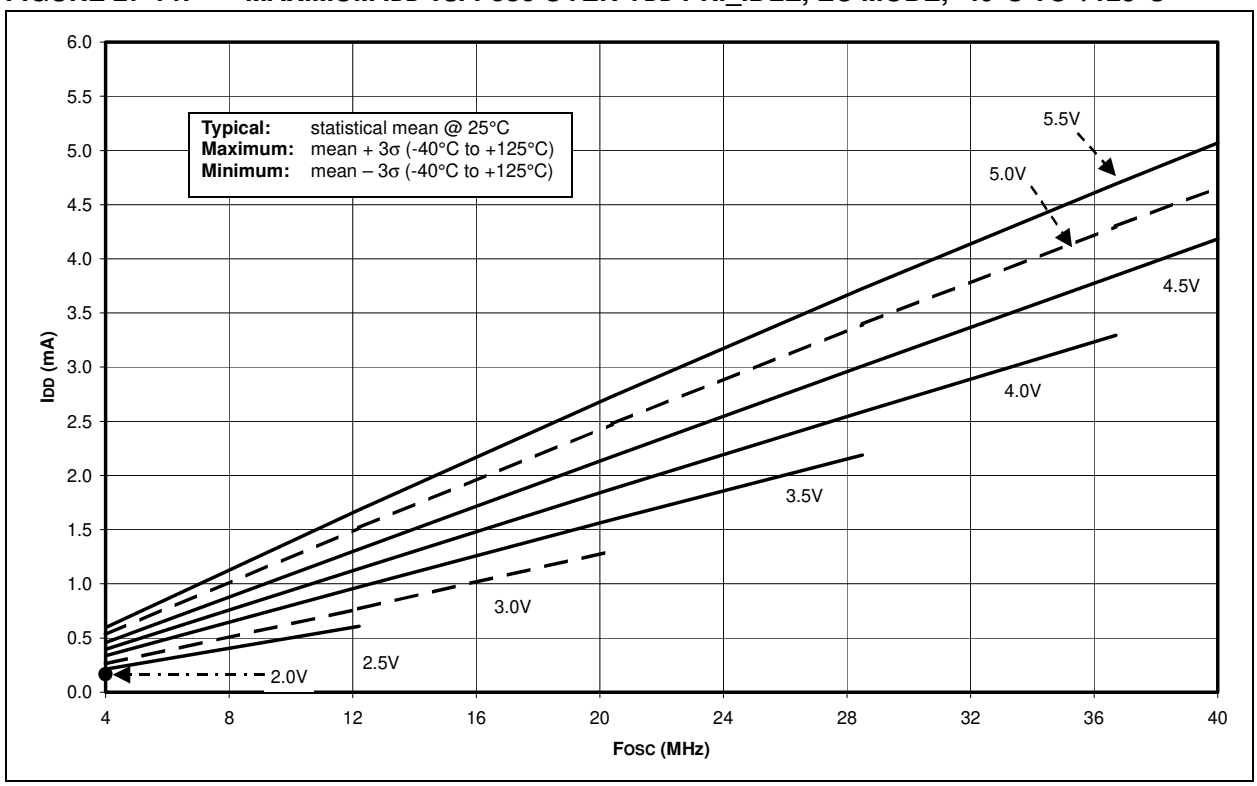


**FIGURE 27-15: TYPICAL $I_{PD}$ vs. $V_{DD}$ (+25°C), 125 kHz TO 8 MHz RC_RUN MODE, ALL PERIPHERALS DISABLED**

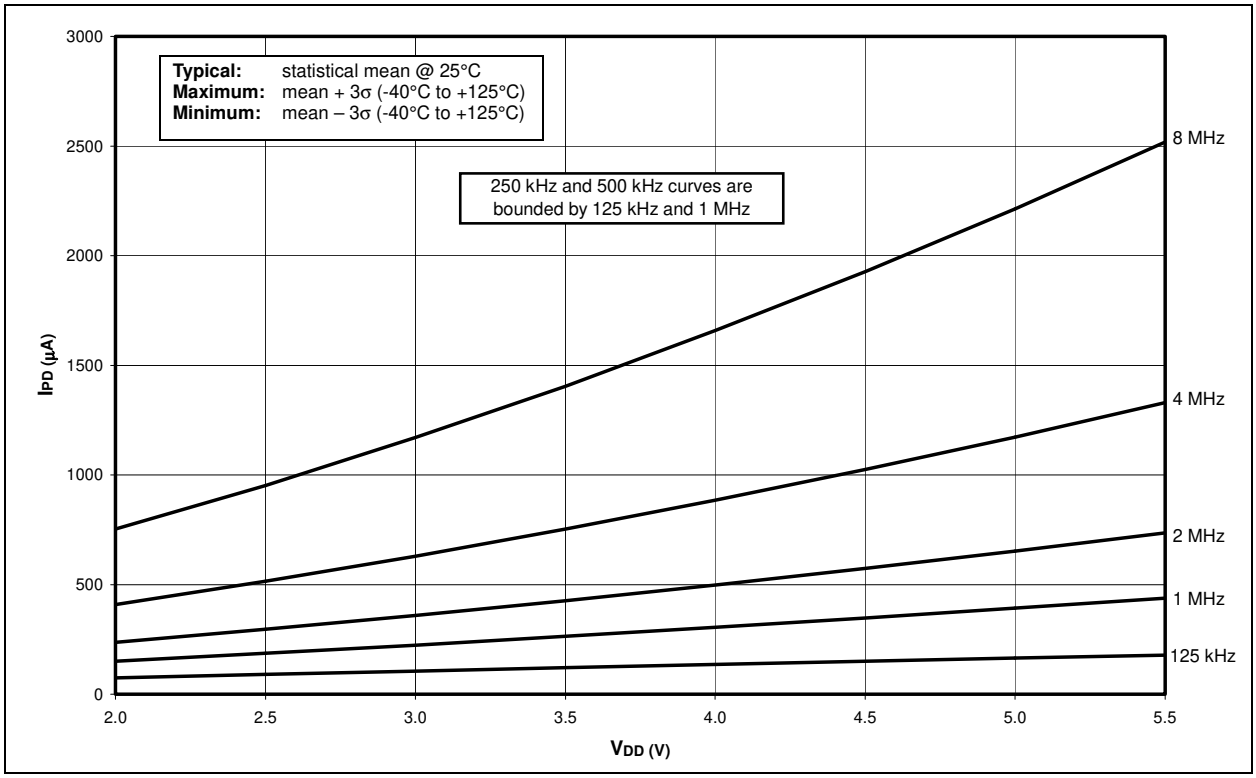
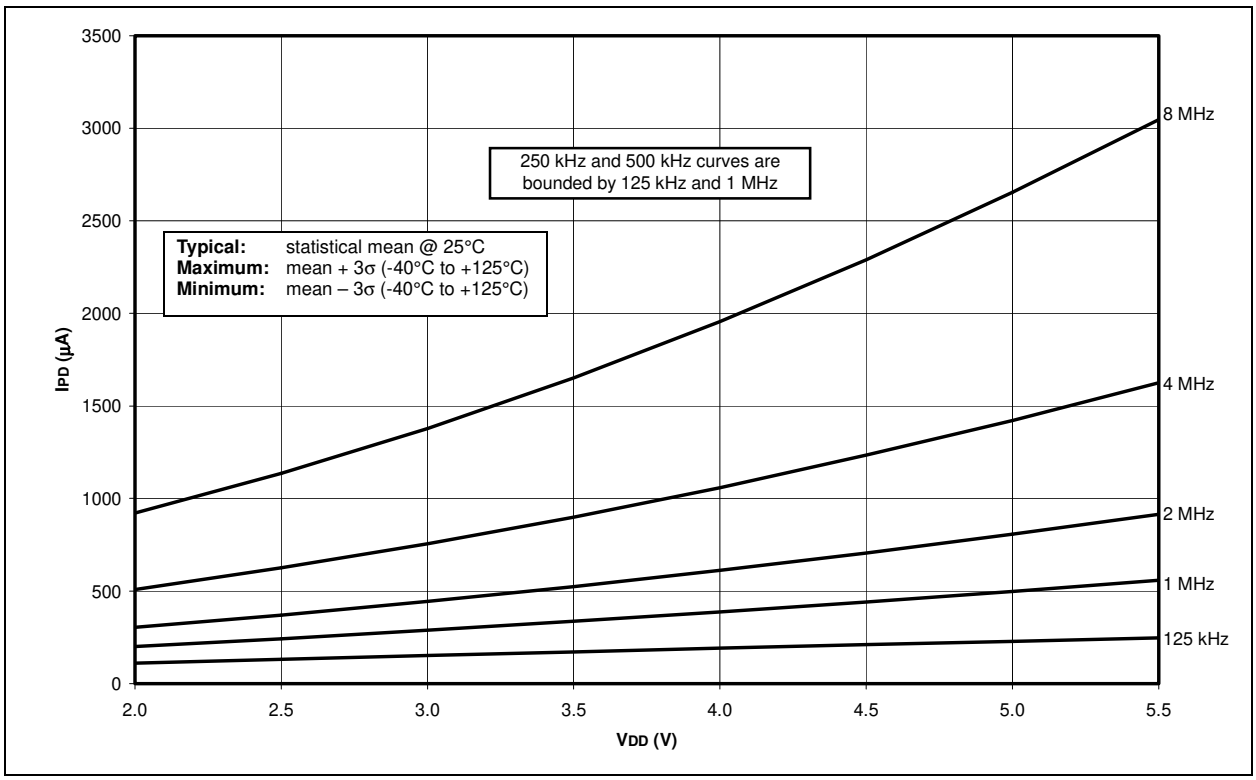



**FIGURE 27-16: MAXIMUM $I_{PD}$ vs. $V_{DD}$ (-40°C TO +125°C), 125 kHz TO 8 MHz RC_RUN, ALL PERIPHERALS DISABLED**





© 2003 Microchip Technology Inc.

**FIGURE 27-17: TYPICAL AND MAXIMUM $I_{PD}$ vs. $V_{DD}$ (-40°C TO +125°C), 31.25 kHz RC_RUN, ALL PERIPHERALS DISABLED**

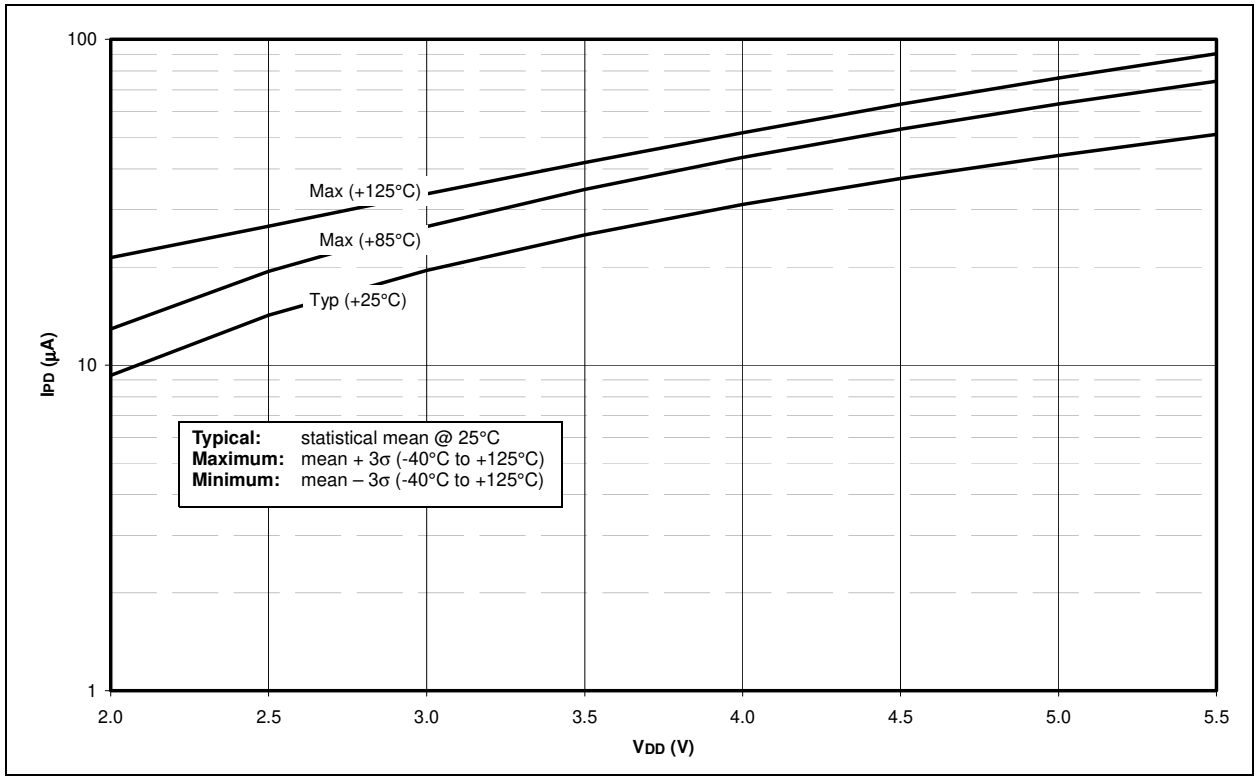



**FIGURE 27-18: TYPICAL $I_{PD}$ vs. $V_{DD}$ (+25°C), 125 kHz TO 8 MHz RC_IDLE MODE, ALL PERIPHERALS DISABLED**

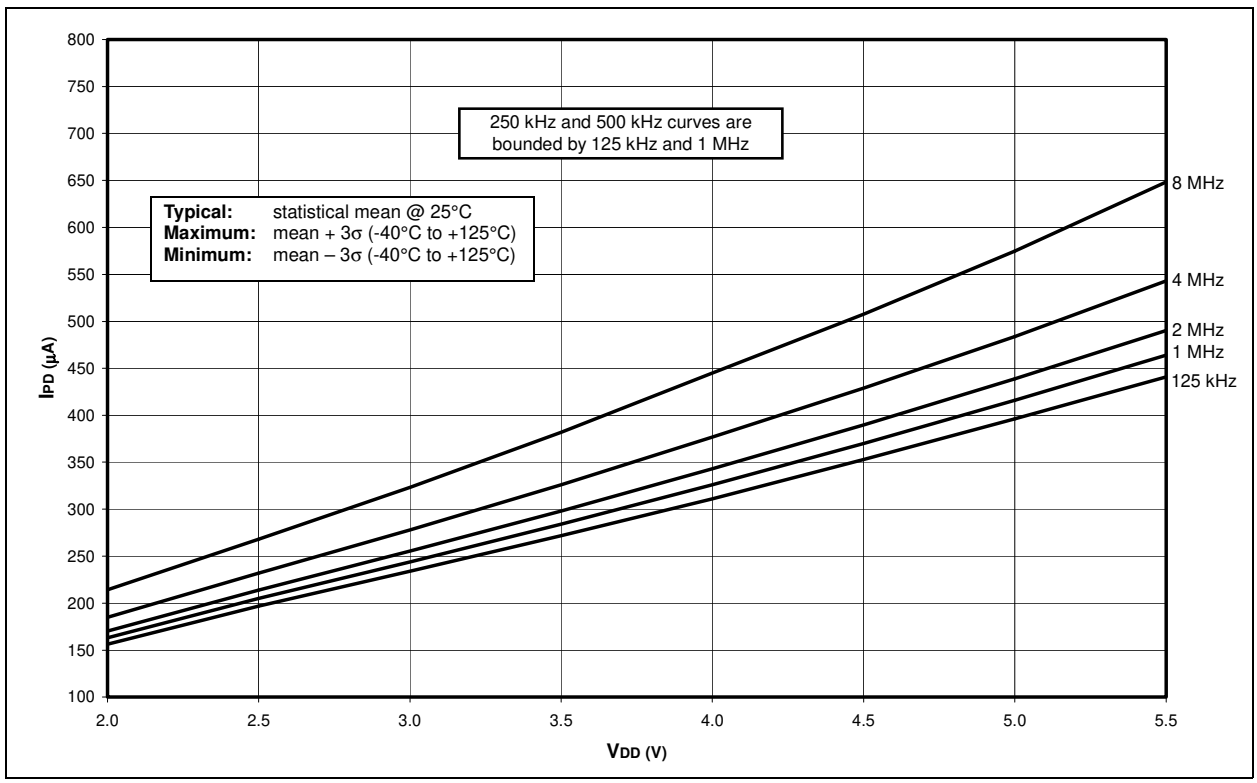


**FIGURE 27-19: MAXIMUM $I_{PD}$ vs. $V_{DD}$ (-40°C TO +125°C), 125 kHz TO 8 MHz RC_IDLE, ALL PERIPHERALS DISABLED**

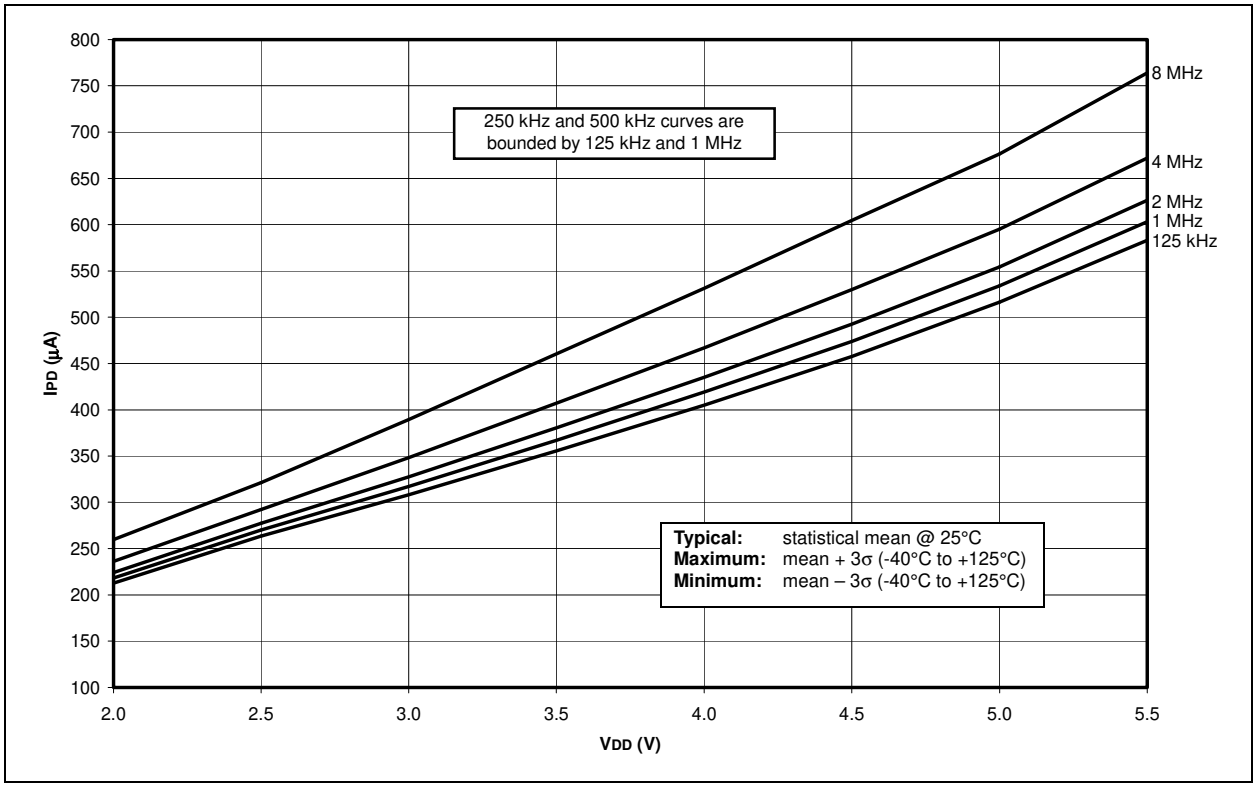
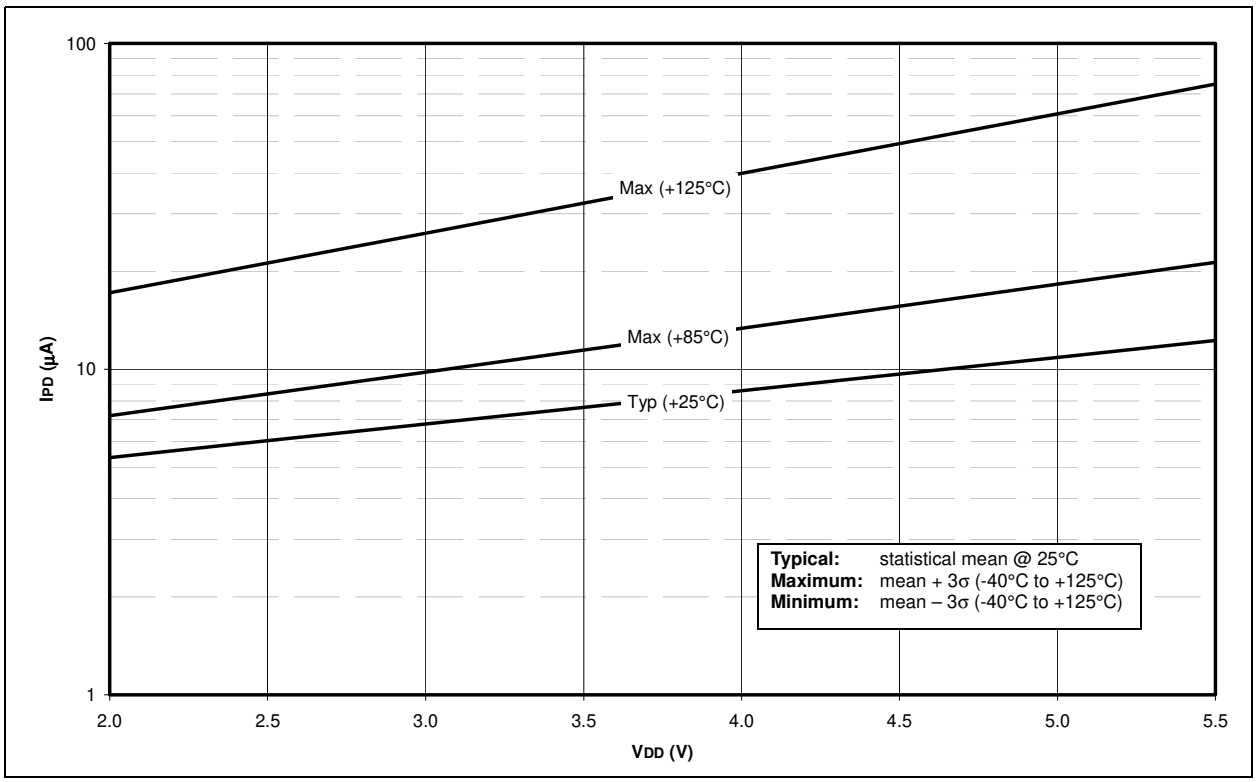



**FIGURE 27-20: TYPICAL AND MAXIMUM $I_{PD}$ vs. $V_{DD}$ (-40°C TO +125°C), 31.25 kHz RC_IDLE, ALL PERIPHERALS DISABLED**





© 2003 Microchip Technology Inc.

**FIGURE 27-21: IPD SEC_RUN MODE, -10°C TO +70°C 32.768 kHz XTAL 2 X 22 pF, ALL PERIPHERALS DISABLED**

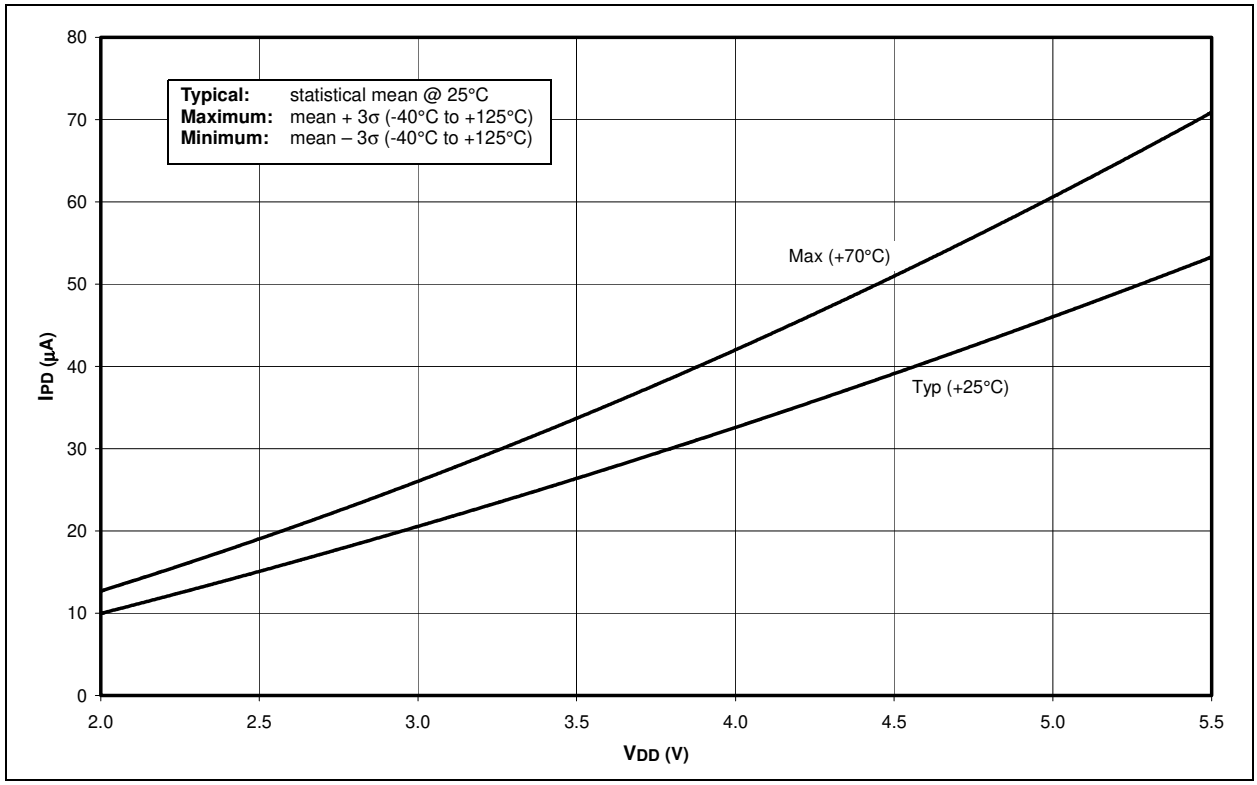



**FIGURE 27-22: IPD SEC_IDLE, -10°C TO +70°C 32.768 kHz 2 X 22 pF, ALL PERIPHERALS DISABLED**

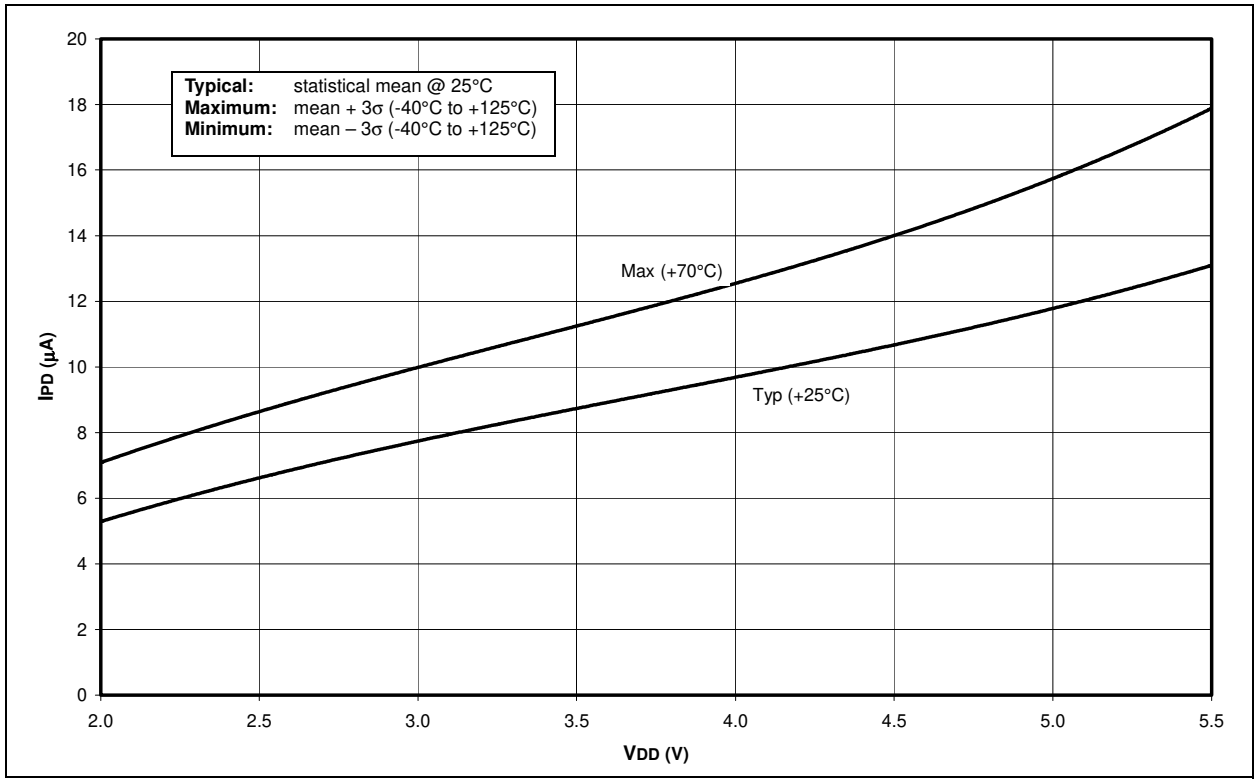


**FIGURE 27-23: TOTAL $I_{PD}$, -40°C TO +125°C SLEEP MODE, ALL PERIPHERALS DISABLED**

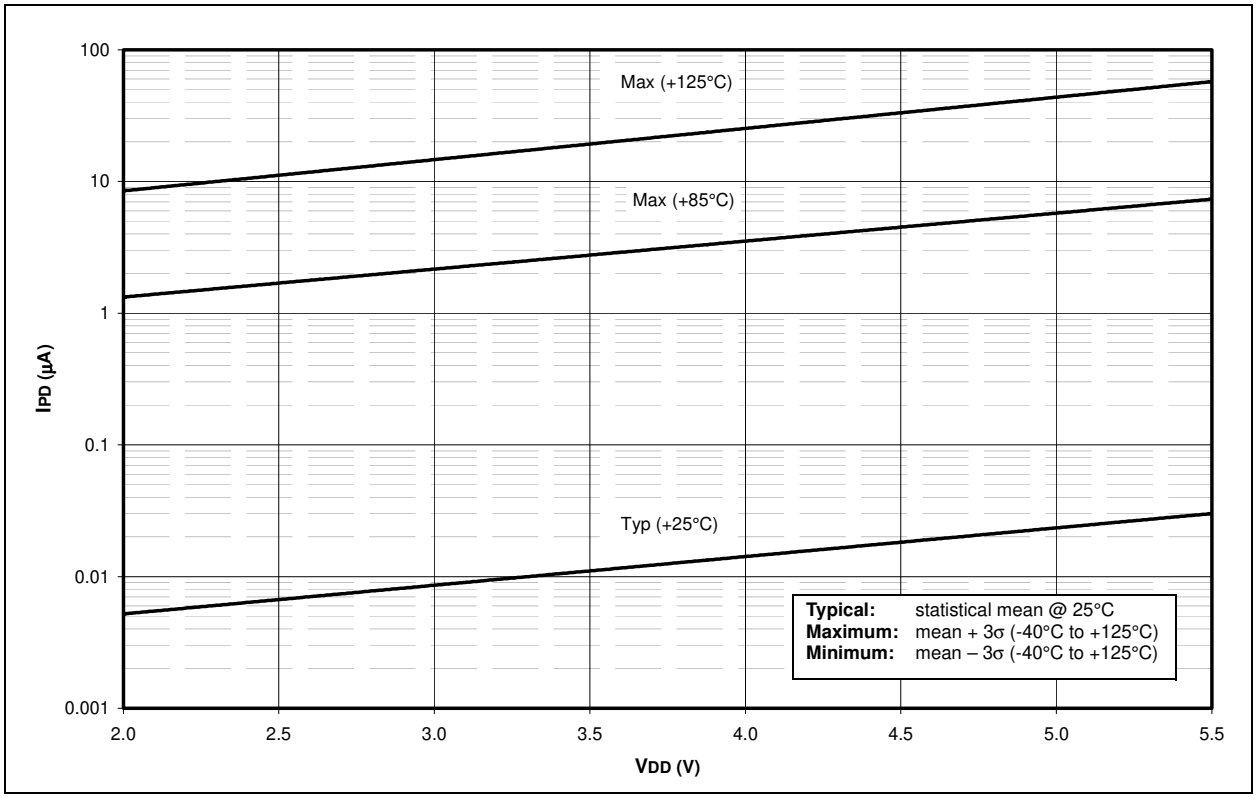
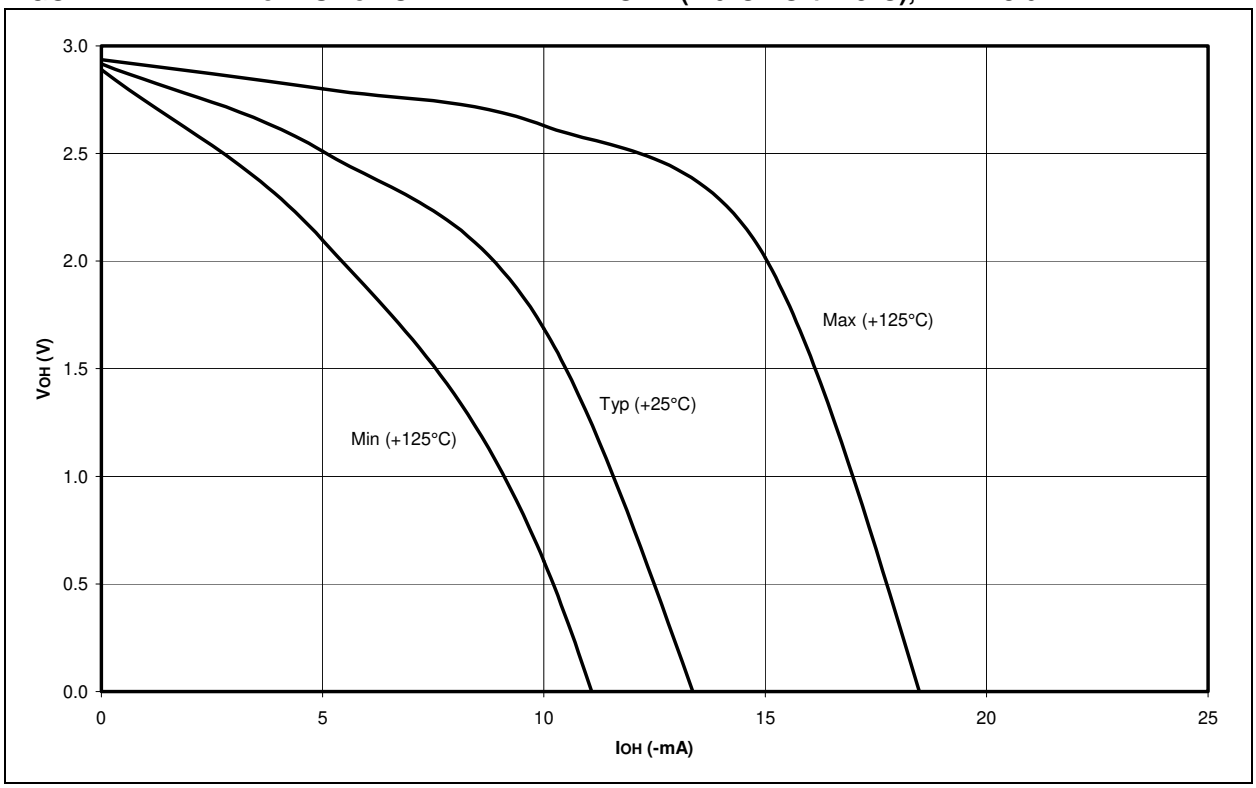



**FIGURE 27-24: $V_{OH}$ vs. $I_{OH}$ OVER TEMPERATURE (-40°C TO +125°C), $V_{DD}$ = 3.0V**





© 2003 Microchip Technology Inc.

**FIGURE 27-25: $V_{OH}$ vs. $I_{OH}$ OVER TEMPERATURE (-40°C TO +125°C), $V_{DD}$ = 5.0V**

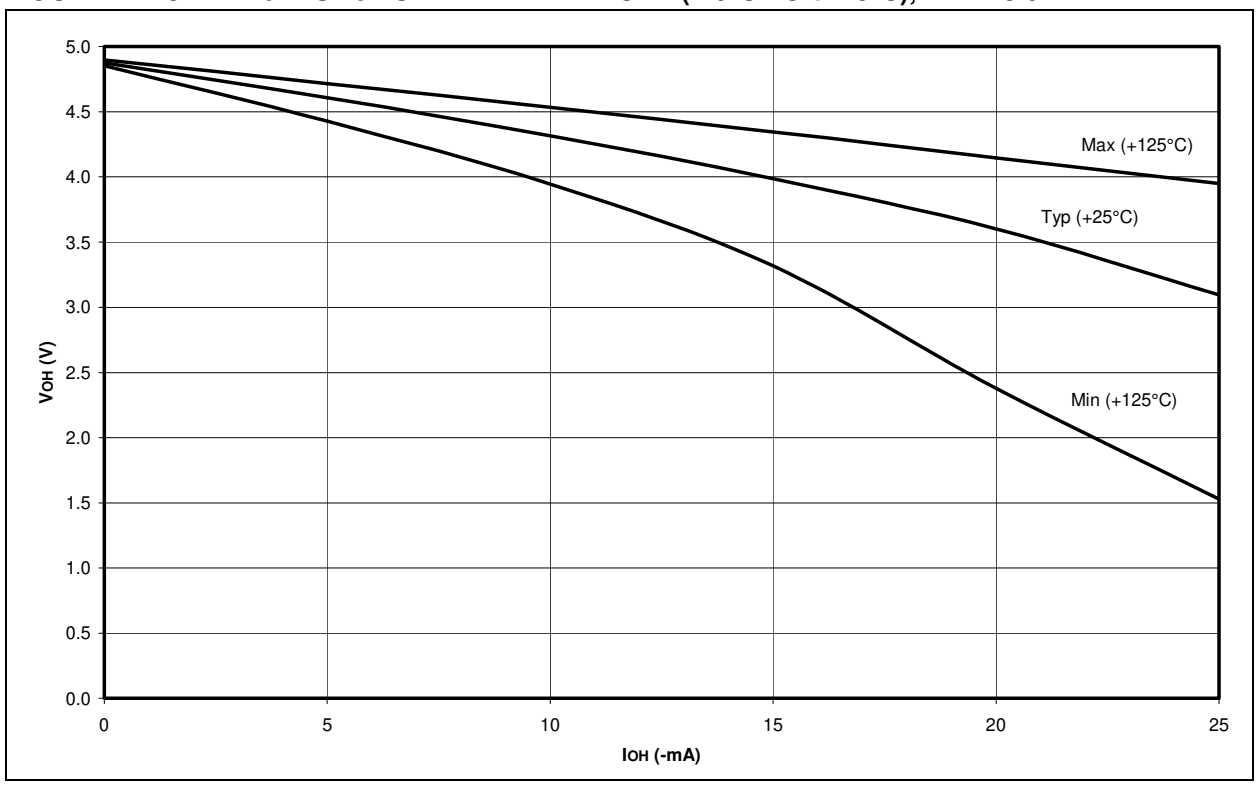



**FIGURE 27-26: $V_{OL}$ vs. $I_{OL}$ OVER TEMPERATURE (-40°C TO +125°C), $V_{DD}$ = 3.0V**

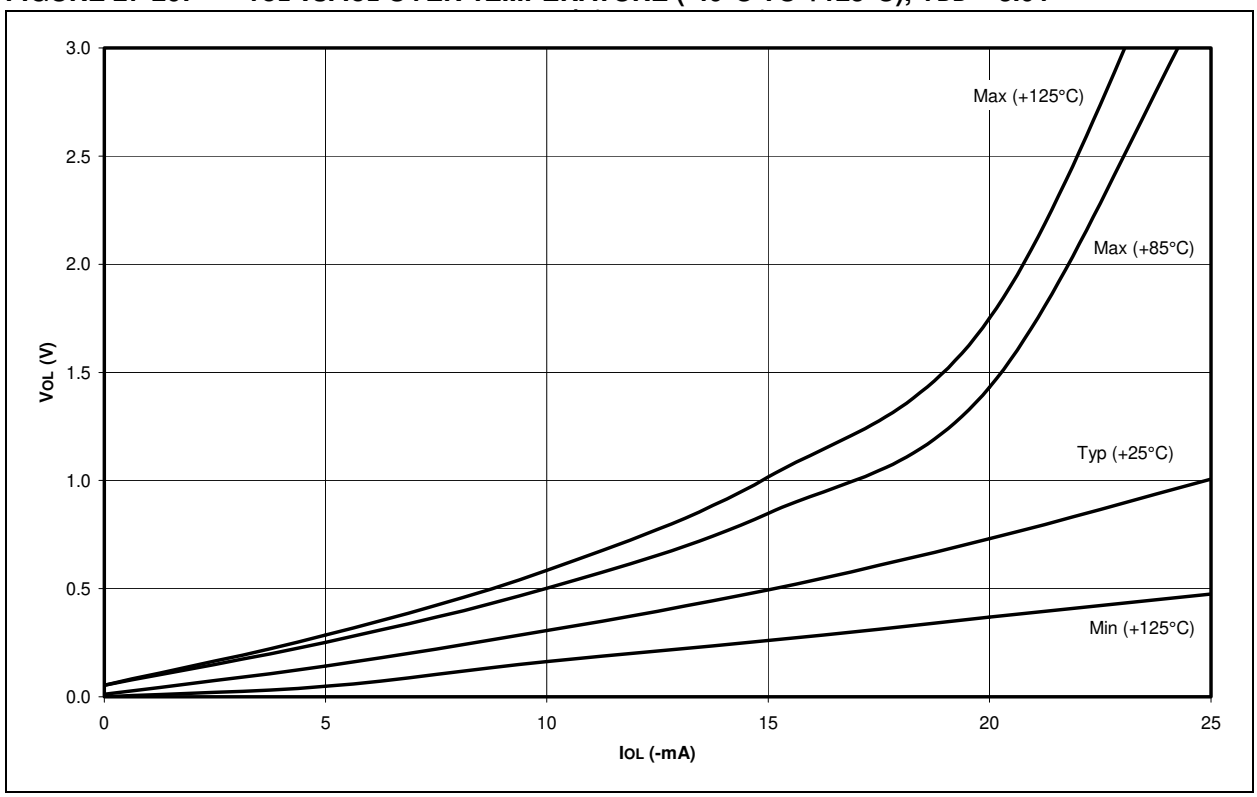


**FIGURE 27-27: $V_{OL}$ vs. $I_{OL}$ OVER TEMPERATURE (-40°C TO +125°C), $V_{DD}$ = 5.0V**

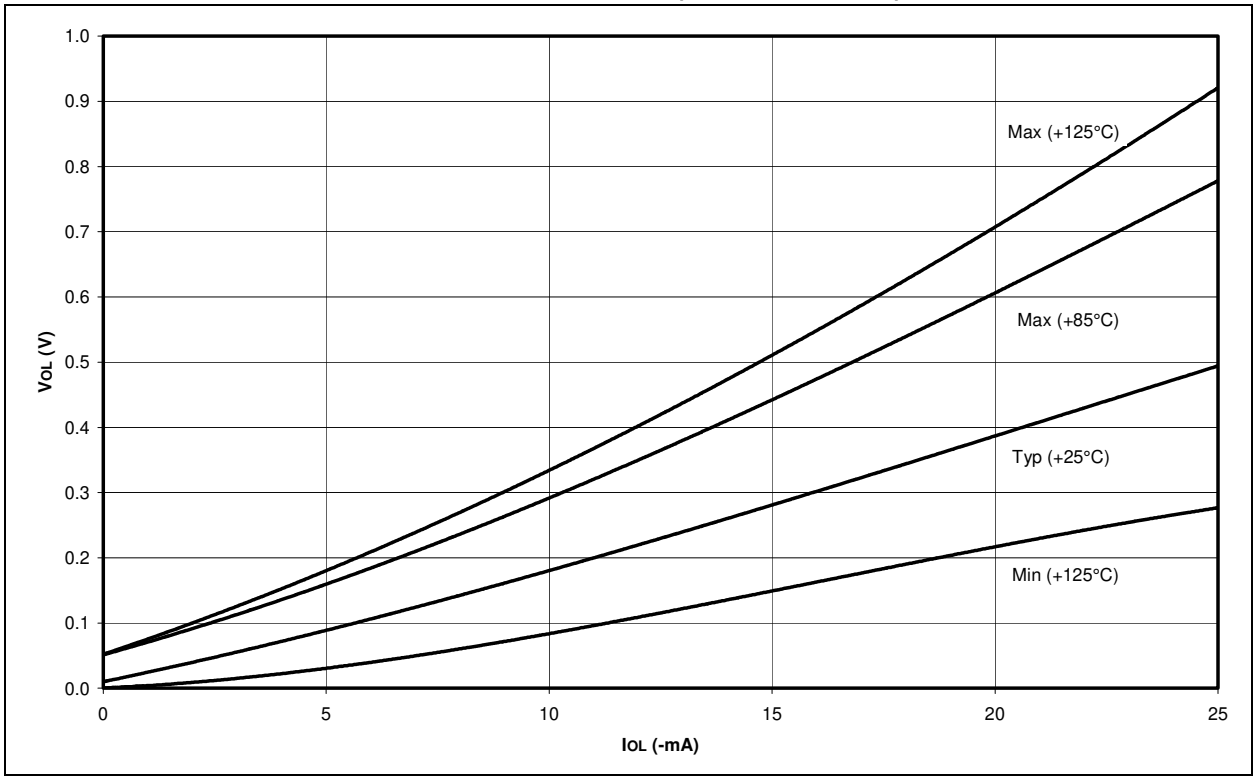
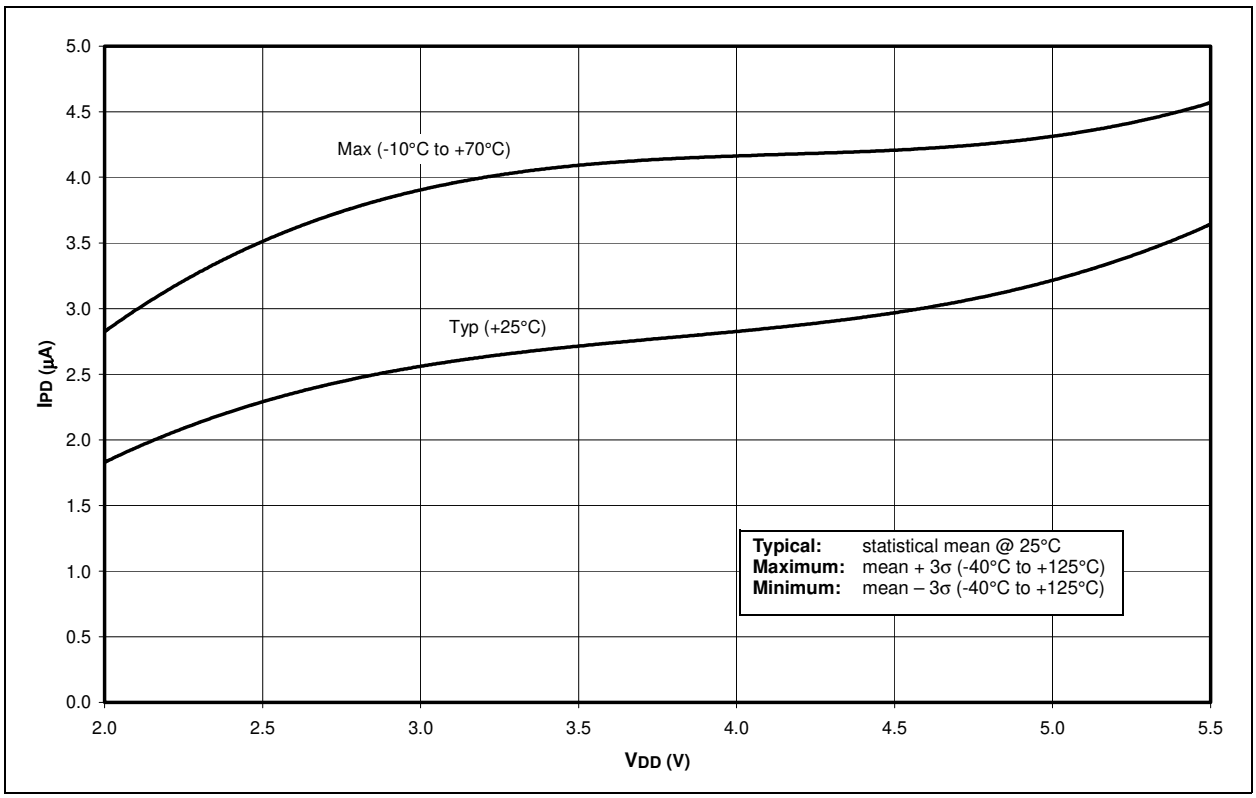



**FIGURE 27-28: Δ $I_{PD}$ TIMER1 OSCILLATOR, -10°C TO +70°C SLEEP MODE, TMR1 COUNTER DISABLED**





© 2003 Microchip Technology Inc.

**FIGURE 27-29: ΔIPD FSCM vs. VDD OVER TEMPERATURE PRI_IDLE, EC OSCILLATOR AT 32 kHz, -40°C TO +125°C**

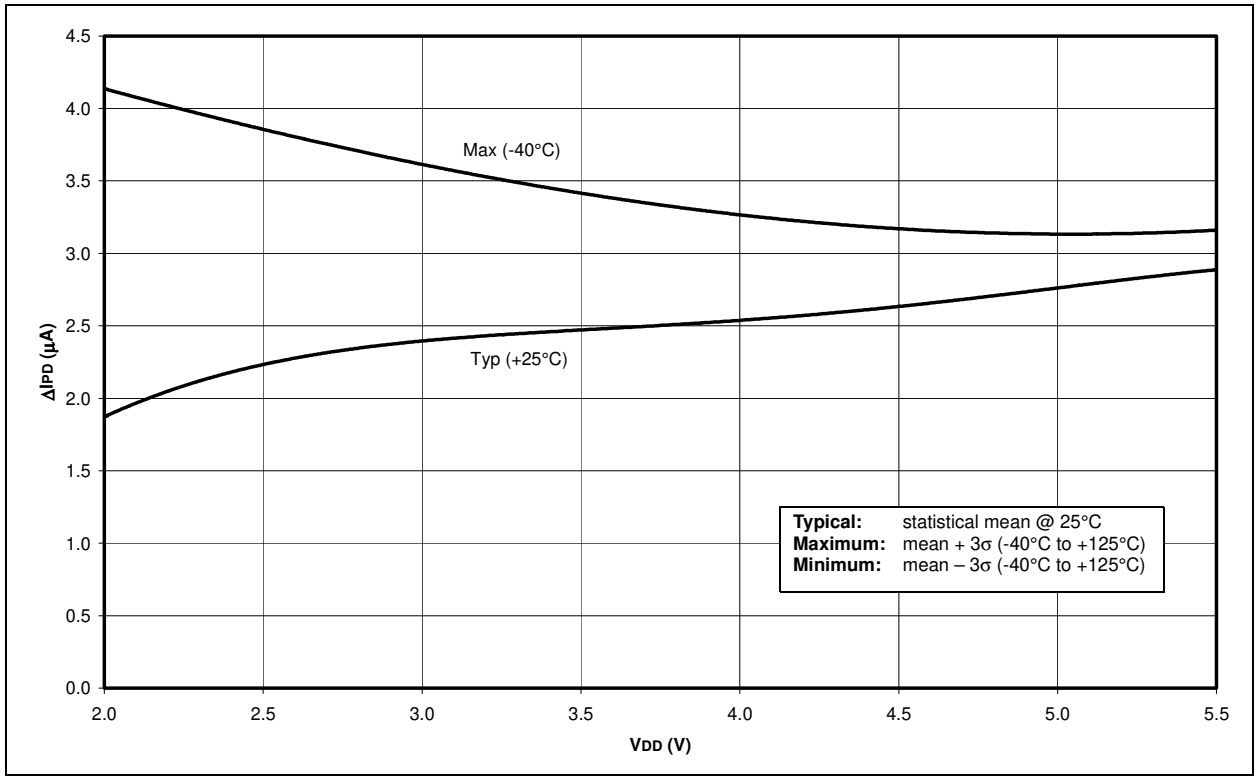



**FIGURE 27-30: ΔIPD WDT, -40°C TO +125°C SLEEP MODE, ALL PERIPHERALS DISABLED**

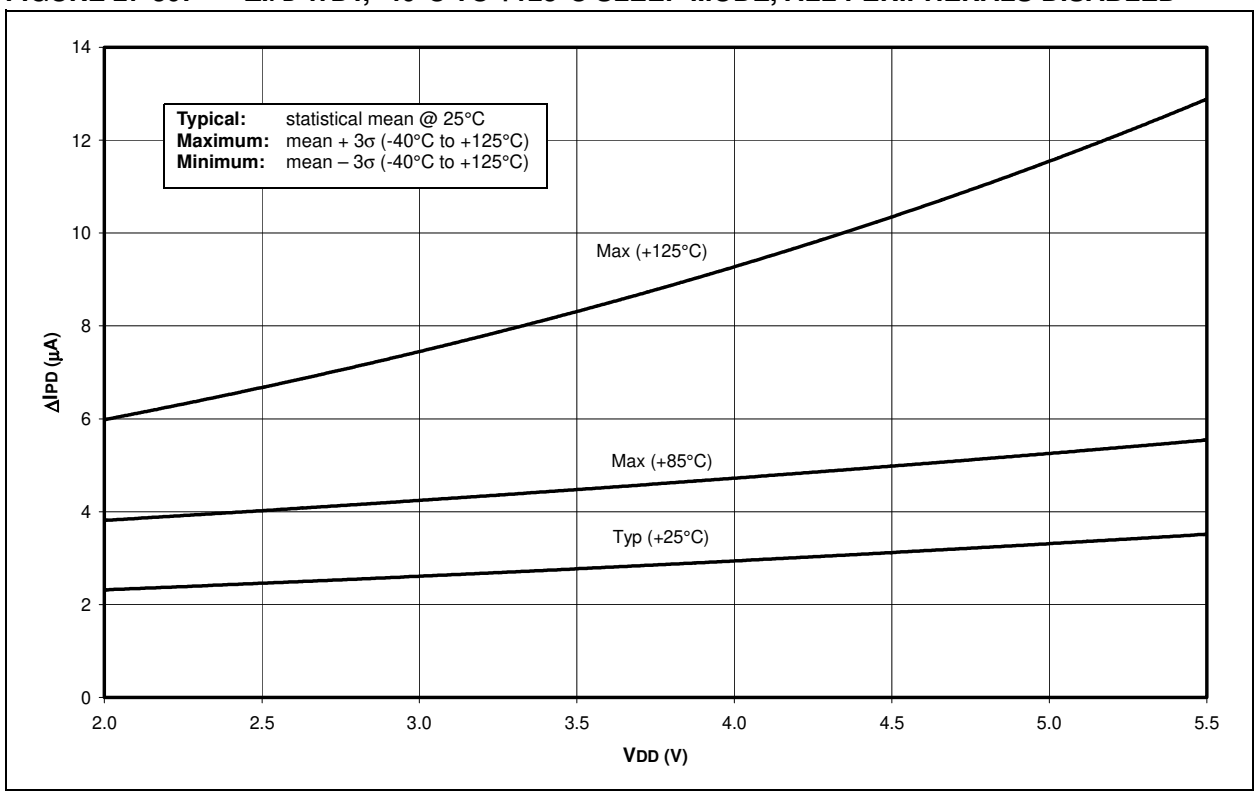


**FIGURE 27-31: ΔIPD LVD vs. VDD SLEEP MODE, LVD = 2.00V-2.12V**

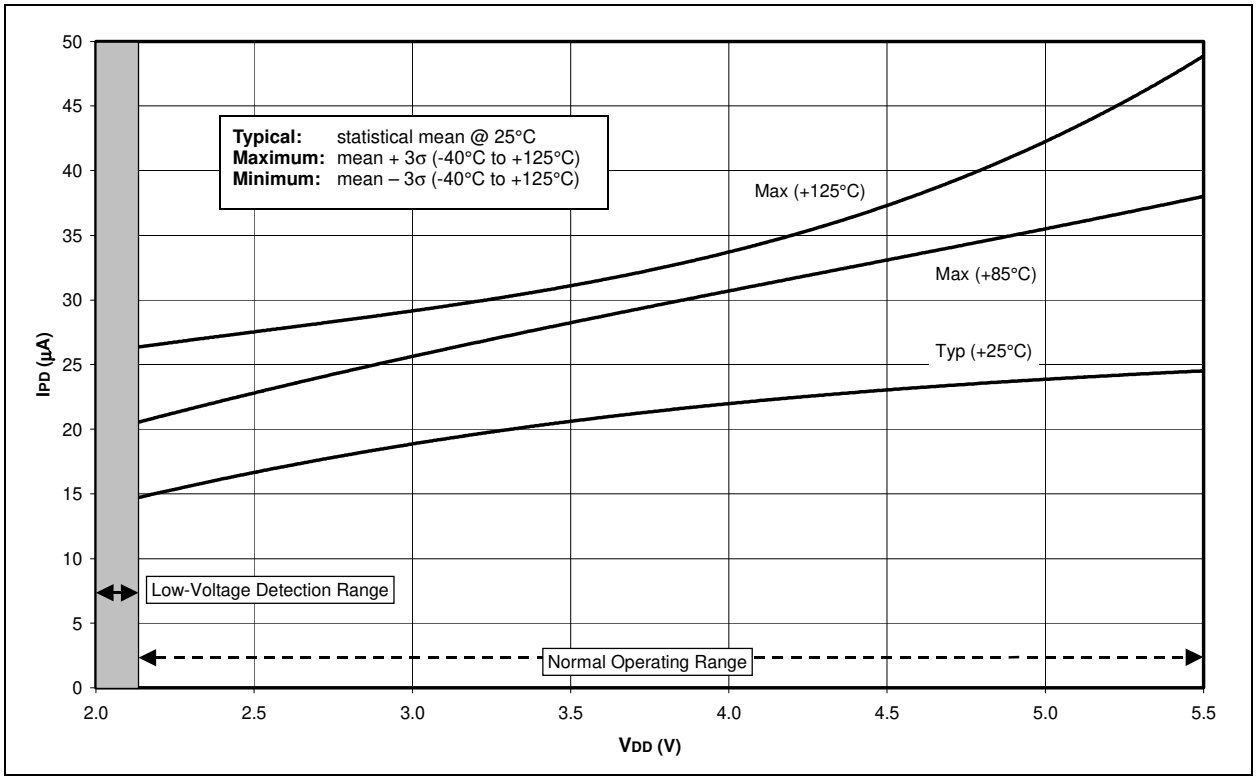
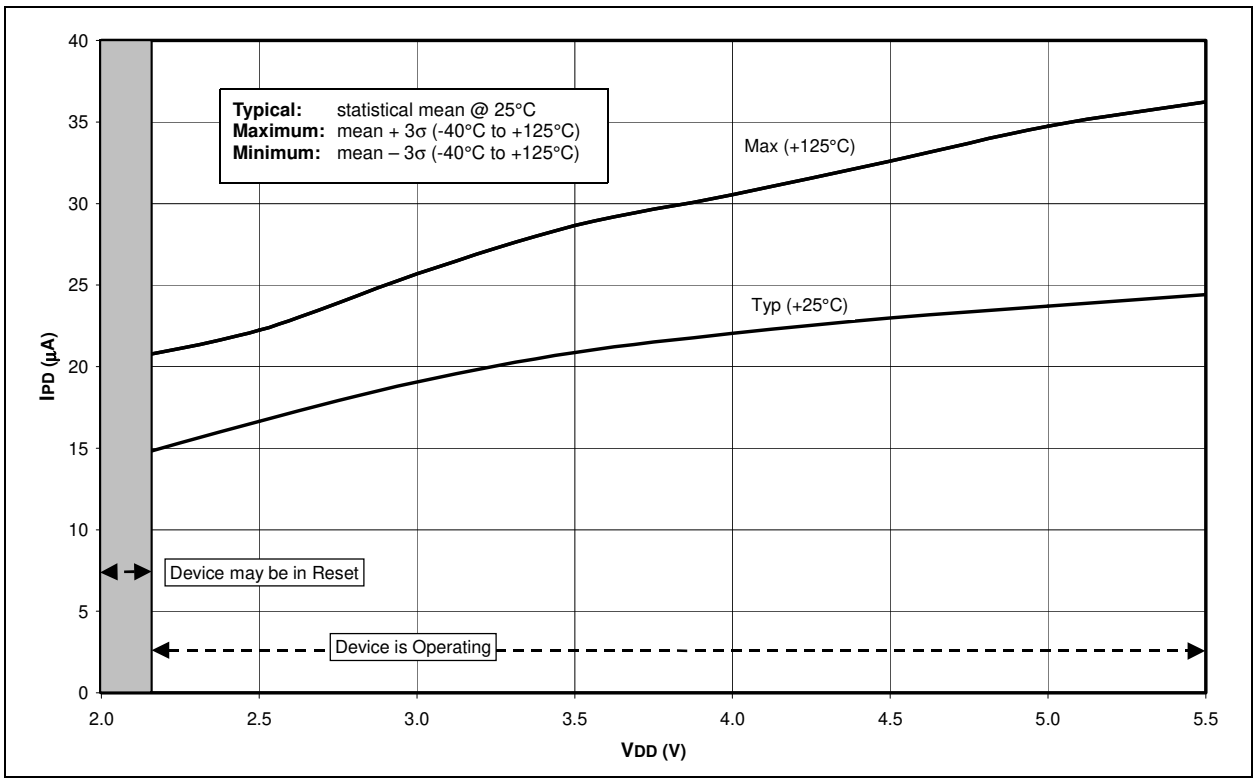



**FIGURE 27-32: ΔIPD BOR vs. VDD, -40°C TO +125°C SLEEP MODE, BOR ENABLED AT 2.00V-2.16V**





© 2003 Microchip Technology Inc.

**FIGURE 27-33: ΔIPD A/D, -40°C TO +125°C SLEEP MODE, A/D ENABLED (NOT CONVERTING)**

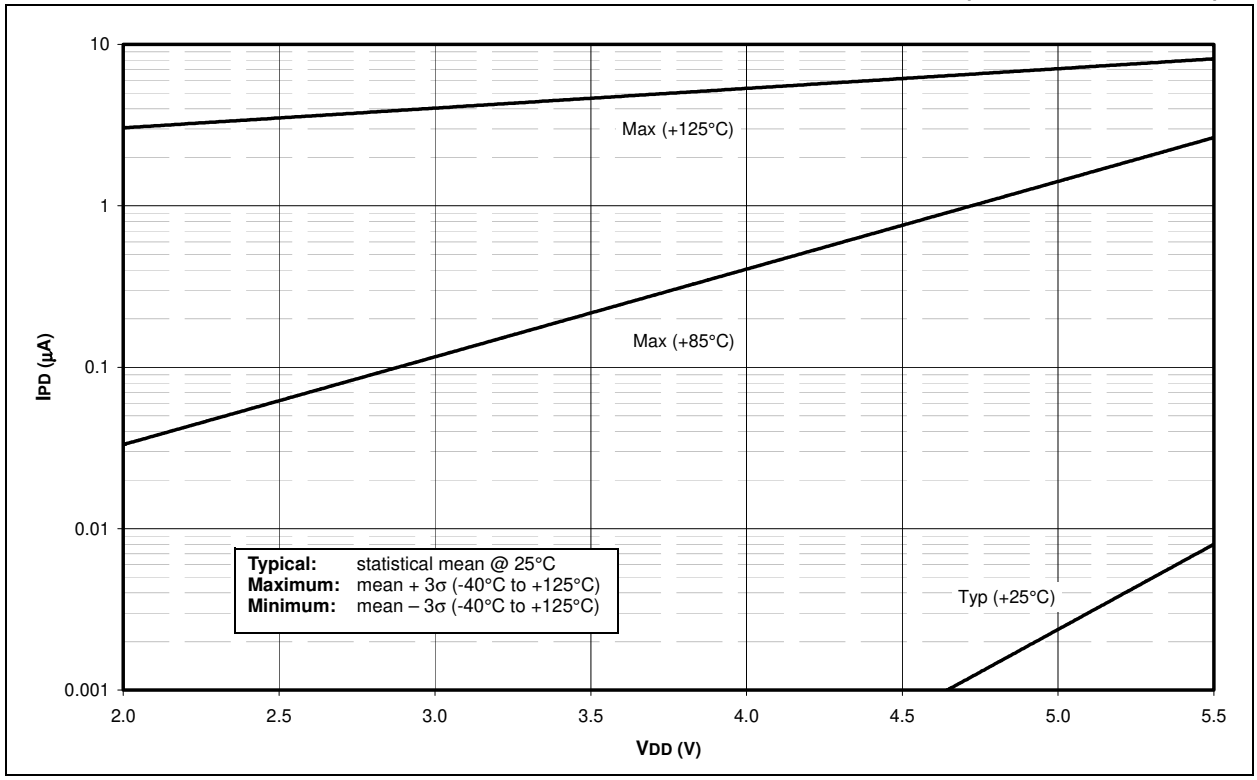



**FIGURE 27-34: AVERAGE FOSC vs. VDD FOR VARIOUS R'S EXTERNAL RC MODE, C = 20 pF, TEMPERATURE = +25°C**

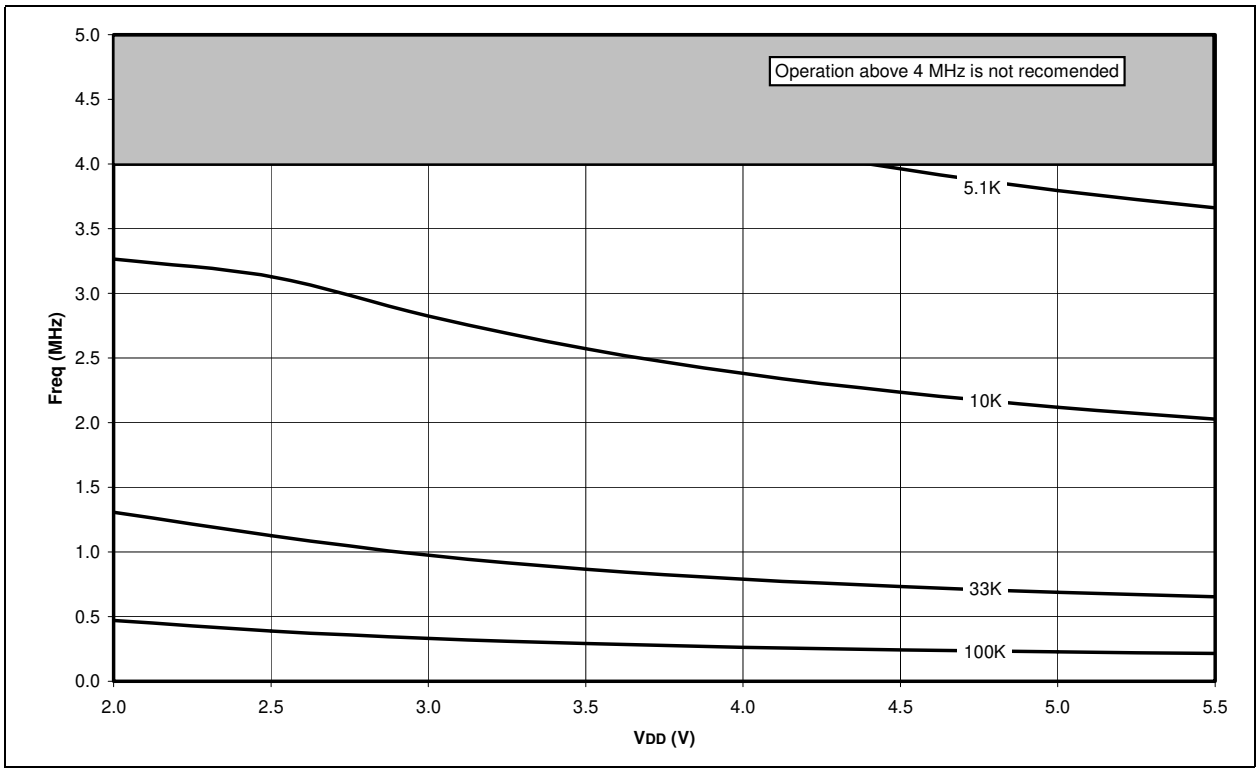


**FIGURE 27-35: AVERAGE FOSC vs. VDD FOR VARIOUS R'S EXTERNAL RC MODE, C = 100 pF, TEMPERATURE = +25°C**

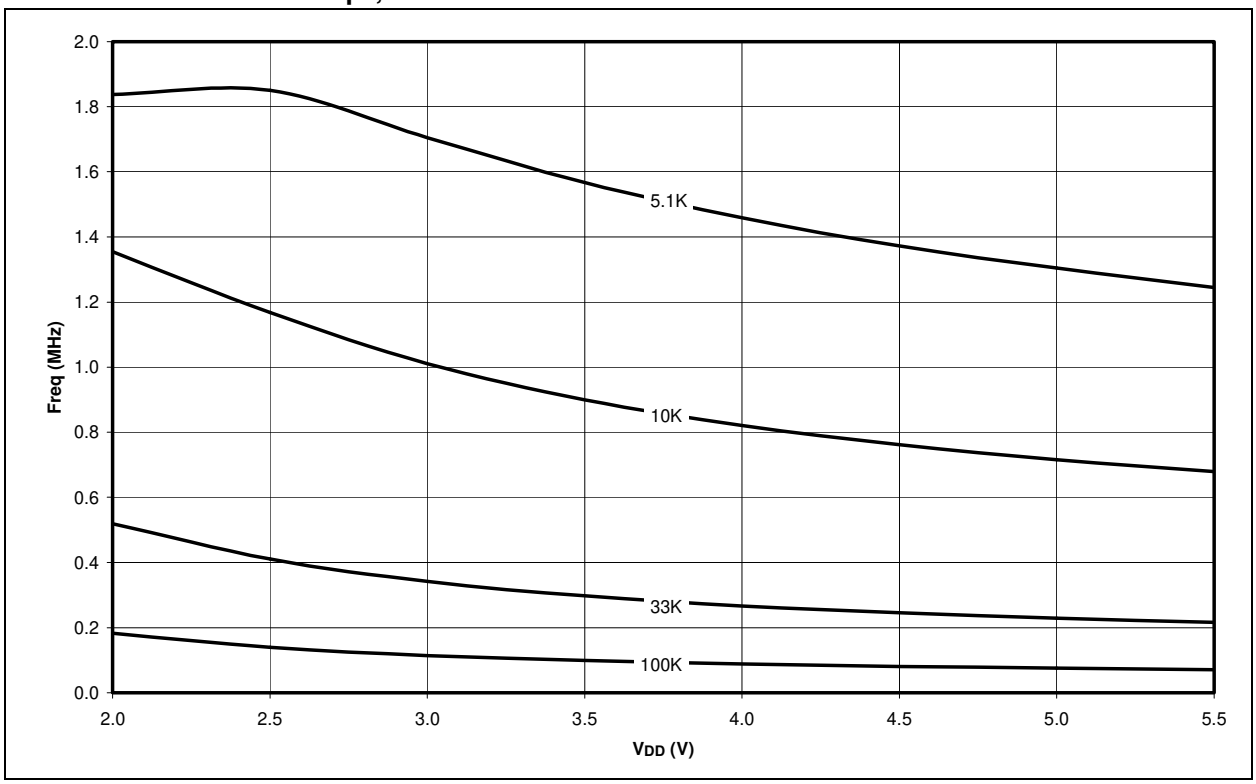
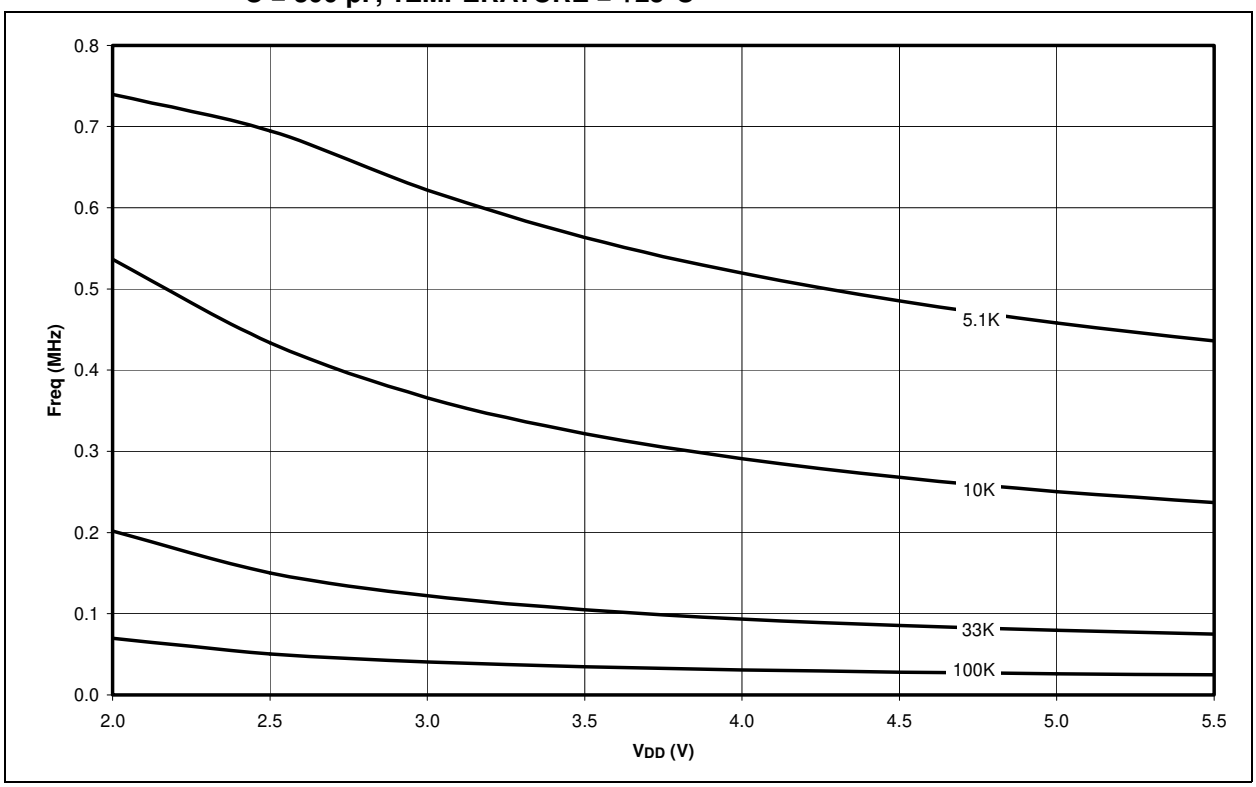



**FIGURE 27-36: AVERAGE FOSC vs. VDD FOR VARIOUS R'S EXTERNAL RC MODE, C = 300 pF, TEMPERATURE = +25°C**



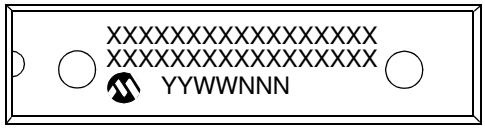


© 2003 Microchip Technology Inc.

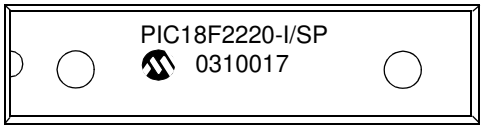
## 28.0 PACKAGING INFORMATION

### 28.1 Package Marking Information

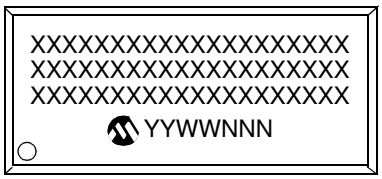
28-Lead SPDIP



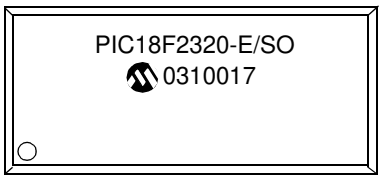

Example



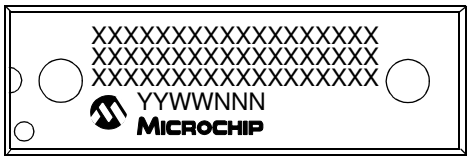

28-Lead SOIC



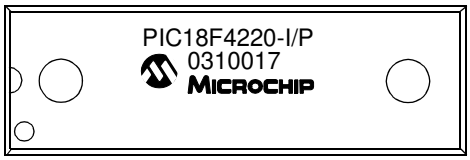

Example




40-Lead PDIP




Example



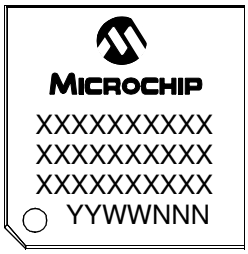

| | | |
|---|---|---|
| **Legend:** | XX...X | Customer specific information* |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week ‘01’) |
| | NNN | Alphanumeric traceability code |

**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

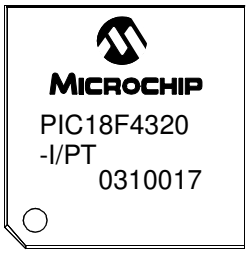
* Standard PICmicro device marking consists of Microchip part number, year code, week code, and traceability code. For PICmicro device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

**Package Marking Information (Continued)**

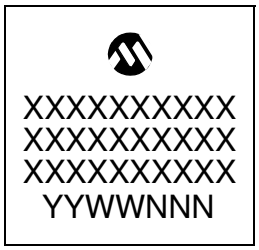
44-Lead TQFP




Example




44-Lead QFN



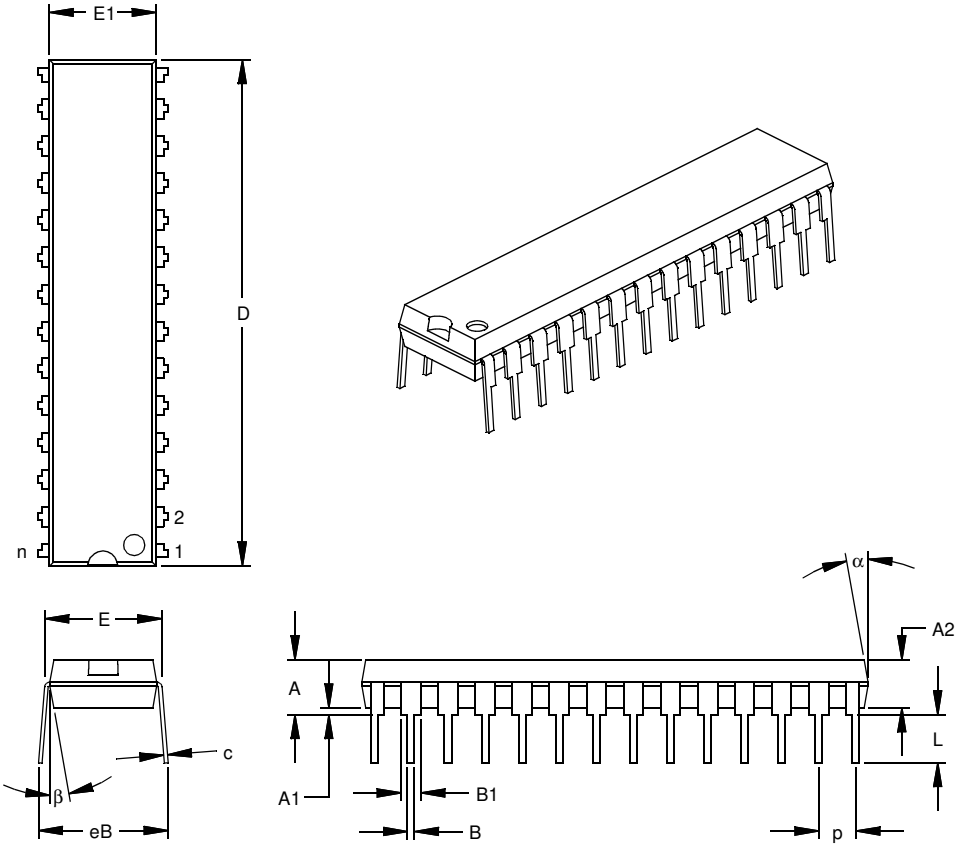

Example

© 2003 Microchip Technology Inc.

## 28.2 Package Details

The following sections give the technical details of the packages.

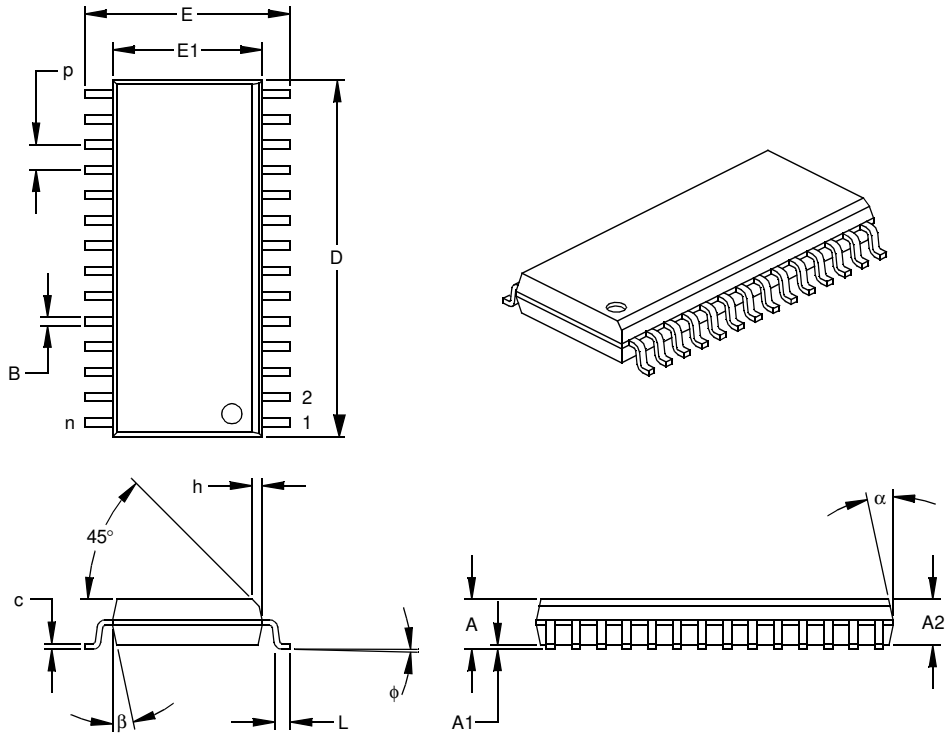
### 28-Lead Skinny Plastic Dual In-line (SP) – 300 mil (PDIP)




| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .150 | .160 | 3.56 | 3.81 | 4.06 |
| Molded Package Thickness | A2 | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .310 | .325 | 7.62 | 7.87 | 8.26 |
| Molded Package Width | E1 | .275 | .285 | .295 | 6.99 | 7.24 | 7.49 |
| Overall Length | D | 1.345 | 1.365 | 1.385 | 34.16 | 34.67 | 35.18 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .040 | .053 | .065 | 1.02 | 1.33 | 1.65 |
| Lower Lead Width | B | .016 | .019 | .022 | 0.41 | 0.48 | 0.56 |
| Overall Row Spacing § | eB | .320 | .350 | .430 | 8.13 | 8.89 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
§ Significant Characteristic
Notes:
Dimension D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MO-095
Drawing No. C04-070

### 28-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)



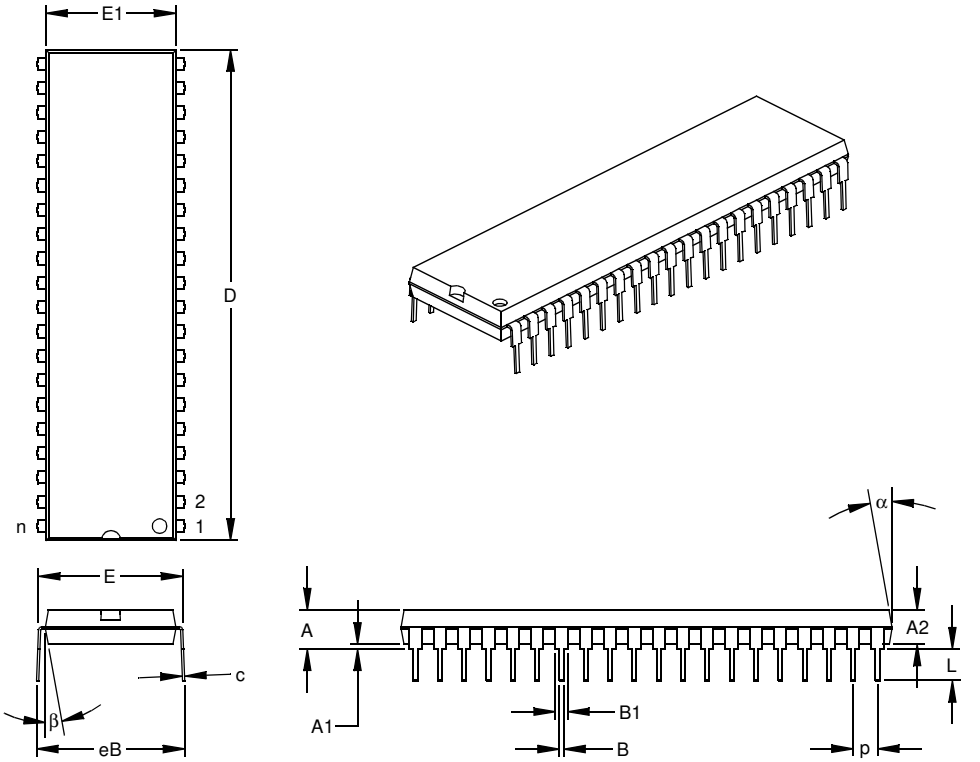

| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .288 | .295 | .299 | 7.32 | 7.49 | 7.59 |
| Overall Length | D | .695 | .704 | .712 | 17.65 | 17.87 | 18.08 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle Top | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .013 | 0.23 | 0.28 | 0.33 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-013
Drawing No. C04-052


© 2003 Microchip Technology Inc.

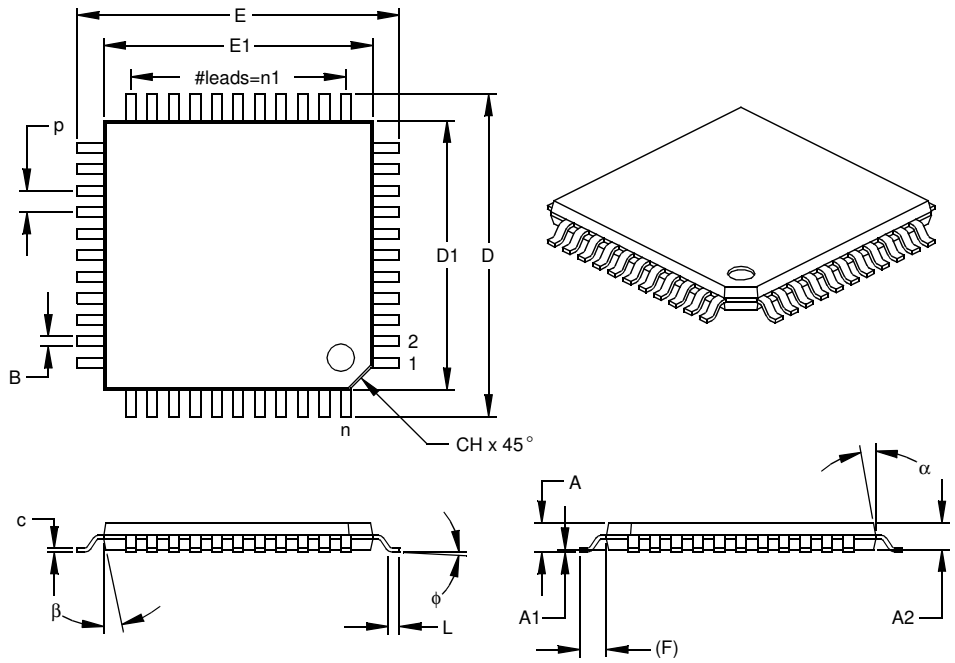
## 40-Lead Plastic Dual In-line (P) – 600 mil (PDIP)




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

* Controlling Parameter
§ Significant Characteristic
Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MO-011
Drawing No. C04-016

### 44-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



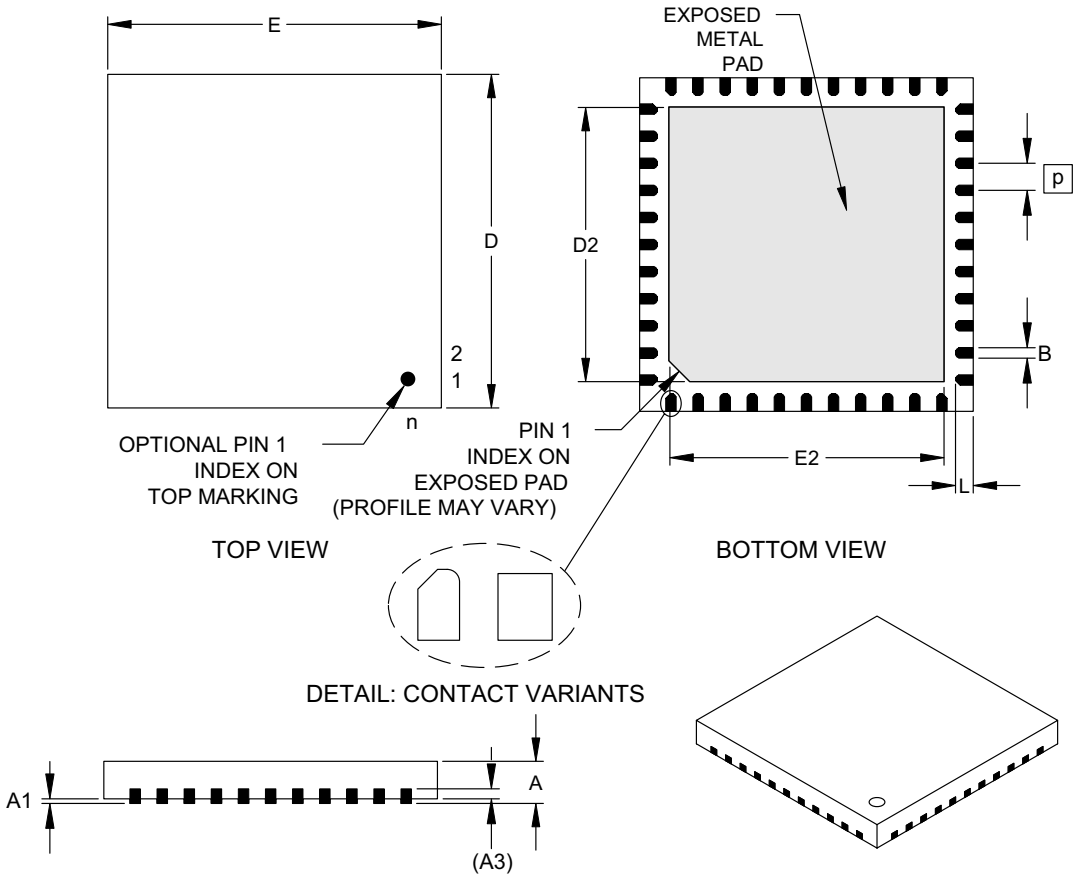

| Units | | INCHES | | | MILLIMETERS* | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 44 | | | 44 | |
| Pitch | p | | .031 | | | 0.80 | |
| Pins per Side | n1 | | 11 | | | 11 | |
| Overall Height | A | .039 | .043 | .047 | 1.00 | 1.10 | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | .004 | .006 | 0.05 | 0.10 | 0.15 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | 1.00 | | |
| Foot Angle | φ | 0 | 3.5 | 7 | 0 | 3.5 | 7 |
| Overall Width | E | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Overall Length | D | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Molded Package Width | E1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Molded Package Length | D1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Lead Thickness | c | .004 | .006 | .008 | 0.09 | 0.15 | 0.20 |
| Lead Width | B | .012 | .015 | .017 | 0.30 | 0.38 | 0.44 |
| Pin 1 Corner Chamfer | CH | .025 | .035 | .045 | 0.64 | 0.89 | 1.14 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MS-026
Drawing No. C04-076


© 2003 Microchip Technology Inc.

## 44-Lead Plastic Quad Flat No Lead Package (ML) 8x8 mm Body (QFN)




| Units | | INCHES | | | MILLIMETERS* | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Contacts | n | | 44 | | | 44 | |
| Pitch | p | | .026 BSC [1] | | | 0.65 BSC [1] | |
| Overall Height | A | .031 | .035 | .039 | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | .000 | .001 | .002 | 0 | 0.02 | 0.05 |
| Base Thickness | (A3) | | .010 REF [2] | | | 0.25 REF [2] | |
| Overall Width | E | .309 | .315 | .321 | 7.85 | 8.00 | 8.15 |
| Exposed Pad Width | E2 | .246 | .268 | .274 | 6.25 | 6.80 | 6.95 |
| Overall Length | D | .309 | .315 | .321 | 7.85 | 8.00 | 8.15 |
| Exposed Pad Length | D2 | .246 | .268 | .274 | 6.25 | 6.80 | 6.95 |
| Contact Width | B | .008 | .013 | .013 | 0.20 | 0.33 | 0.35 |
| Contact Length | L | .014 | .016 | .019 | 0.35 | 0.40 | 0.48 |

*Controlling Parameter

Notes:

1. BSC: Basic Dimension. Theoretically exact value shown without tolerances. See ASME Y14.5M
2. REF: Reference Dimension, usually without tolerance, for information purposes only. See ASME Y14.5M
3. Contact profiles may vary.
4. JEDEC equivalent: M0-220

Drawing No. C04-103

**NOTES:**


© 2003 Microchip Technology Inc.

## APPENDIX A: REVISION HISTORY

### Revision A (June 2002)

Original data sheet for PIC18F2X20/4X20 devices.

### Revision B (October 2002)

This revision includes major changes to **Section 2.0 "Oscillator Configurations"** and **Section 3.0 "Power Managed Modes"**, updates to the Electrical Specifications in **Section 26.0 "Electrical Characteristics"** and minor corrections to the data sheet text.

### Revision C (October 2003)

This revision includes updates to the Electrical Specifications in **Section 26.0 "Electrical Characteristics"** and to the DC Characteristics Graphs and Charts in **Section 27.0 "DC and AC Characteristics Graphs and Tables"** and minor corrections to the data sheet text.

## APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

**TABLE B-1: DEVICE DIFFERENCES**

| Features | PIC18F2220 | PIC18F2320 | PIC18F4220 | PIC18F4320 |
|---|---|---|---|---|
| Program Memory (Bytes) | 4096 | 8192 | 4096 | 8192 |
| Program Memory (Instructions) | 2048 | 4096 | 2048 | 4096 |
| Interrupt Sources | 19 | 19 | 20 | 20 |
| I/O Ports | Ports A, B, C, (E) | Ports A, B, C, (E) | Ports A, B, C, D, E | Ports A, B, C, D, E |
| Capture/Compare/PWM Modules | 2 | 2 | 1 | 1 |
| Enhanced Capture/Compare/ PWM Modules | 0 | 0 | 1 | 1 |
| Parallel Communications (PSP) | No | No | Yes | Yes |
| 10-bit Analog-to-Digital Module | 10 input channels | 10 input channels | 13 input channels | 13 input channels |
| Packages | 28-pin SPDIP<br>28-pin SOIC | 28-pin SPDIP<br>28-pin SOIC | 40-pin PDIP<br>44-pin TQFP<br>44-pin QFN | 40-pin PDIP<br>44-pin TQFP<br>44-pin QFN |

## APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

**Not Applicable**

## APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a Baseline device (i.e., PIC16C5X) to an Enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

**Not Currently Available**


© 2003 Microchip Technology Inc.

## APPENDIX E: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in *AN716, "Migrating Designs from PIC16C74A/74B to PIC18C442."* The changes discussed, while device specific, are generally applicable to all mid-range to enhanced device migrations.

This Application Note is available as Literature Number DS00716.

## APPENDIX F: MIGRATION FROM HIGH-END TO ENHANCED DEVICES

A detailed discussion of the migration pathway and differences between the high-end MCU devices (i.e., PIC17CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in *AN726, "PIC17CXXX to PIC18CXXX Migration."* This Application Note is available as Literature Number DS00726.

**NOTES:**


© 2003 Microchip Technology Inc.

# INDEX

## A


A/D ..... 211
A/D Converter Interrupt, Configuring ..... 215
Acquisition Requirements ..... 216
ADCON0 Register ..... 211
ADCON1 Register ..... 211
ADCON2 Register ..... 211
ADRESH Register ..... 211, 214
ADRESL Register ..... 211
Analog Port Pins, Configuring ..... 218
Associated Registers ..... 220
Automatic Acquisition Time ..... 217
Calculating the Minimum Required Acquisition Time ..... 216
Configuring the Module ..... 215
Conversion Clock (TAD) ..... 217
Conversion Status (GO/DONE Bit) ..... 214
Conversions ..... 219
Converter Characteristics ..... 341
Operation in Power Managed Modes ..... 218
Special Event Trigger (CCP) ..... 136, 220
Use of the CCP2 Trigger ..... 220
VREF+ and VREF- References ..... 216
Absolute Maximum Ratings ..... 305
AC (Timing) Characteristics ..... 323
Load Conditions for Device Timing Specifications ..... 324
Parameter Symbology ..... 323
Temperature and Voltage Specifications ..... 324
Timing Conditions ..... 324
Access Bank ..... 65
ACKSTAT Status Flag ..... 185
ADCON0 Register ..... 211
GO/DONE Bit ..... 214
ADCON1 Register ..... 211
ADCON2 Register ..... 211
ADDLW ..... 261
Addressable Universal Synchronous Asynchronous Receiver Transmitter. *See* USART.
ADDWF ..... 261
ADDWFC ..... 262
ADRESH Register ..... 211
ADRESL Register ..... 211, 214
Analog-to-Digital Converter. *See* A/D.
ANDLW ..... 262
ANDWF ..... 263
Assembler
MPASM Assembler ..... 299


## B


Bank Select Register (BSR) ..... 65
Baud Rate Generator ..... 181
BC ..... 263
BCF ..... 264
BF Status Flag ..... 185
Block Diagrams
A/D ..... 214
Analog Input Model ..... 215
Baud Rate Generator ..... 181
Capture Mode Operation ..... 135
Comparator I/O Operating Modes ..... 222
Comparator Output ..... 224
Comparator Voltage Reference ..... 228
Compare Mode Operation ..... 136
External Power-on Reset Circuit (Slow VDD Power-up) ..... 44
Fail-Safe Clock Monitor ..... 248
Generic I/O Port Operation ..... 101
Interrupt Logic ..... 88
Low-Voltage Detect (LVD) ..... 232
Low-Voltage Detect (LVD) with External Input ..... 232
MCLR/VPP/RE3 Pin ..... 111
MSSP (I2C Master Mode) ..... 179
MSSP (I2C Mode) ..... 164
MSSP (SPI Mode) ..... 155
On-Chip Reset Circuit ..... 43
PIC18F2220/2320 ..... 9
PIC18F4220/4320 ..... 10
PLL ..... 20
PORTC (Peripheral Output Override) ..... 107
PORTD and PORTE (Parallel Slave Port) ..... 114
PWM (Enhanced) ..... 143
PWM (Standard) ..... 138
RA3:RA0 and RA5 Pins ..... 102
RA4/T0CKI Pin ..... 102
RA6 Pin ..... 102
RA7 Pin ..... 102
RB2:RB0 Pins ..... 105
RB3/CCP2 Pin ..... 105
RB4 Pin ..... 105
RB7:RB5 Pins ..... 104
RD4:RD0 Pins ..... 110
RD7:RD5 Pins ..... 109
RE2:RE0 Pins ..... 111
Reads from Flash Program Memory ..... 75
System Clock ..... 25
Table Read Operation ..... 71
Table Write Operation ..... 72
Table Writes to Flash Program Memory ..... 77
Timer0 in 16-bit Mode ..... 118
Timer0 in 8-bit Mode ..... 118
Timer1 ..... 122
Timer1 (16-bit Read/Write Mode) ..... 122
Timer2 ..... 128
Timer3 ..... 130
Timer3 (16-bit Read/Write Mode) ..... 130
USART Receive ..... 204
USART Transmit ..... 202
Watchdog Timer ..... 245
BN ..... 264
BNC ..... 265
BNN ..... 265
BNOV ..... 266
BNZ ..... 266
BOR. *See* Brown-out Reset.
BOV ..... 269
BRA ..... 267
BRG. *See* Baud Rate Generator.
Brown-out Reset (BOR) ..... 44, 237
BSF ..... 267
BTFSC ..... 268
BTFSS ..... 268
BTG ..... 269
BZ ..... 270


© 2003 Microchip Technology Inc.

## C


C Compilers
- MPLAB C17 ... 300
- MPLAB C18 ... 300
- MPLAB C30 ... 300

CALL ... 270
Capture (CCP Module) ... 135
- Associated Registers ... 137
- CCP Pin Configuration ... 135
- CCPR1H:CCPR1L Registers ... 135
- Software Interrupt ... 135
- Timer1/Timer3 Mode Selection ... 135

Capture (ECCP Module) ... 142
Capture/Compare/PWM (CCP) ... 133
- Capture Mode. *See* Capture.
- CCP1 ... 134
  - CCPR1H Register ... 134
  - CCPR1L Register ... 134
- CCP2 ... 134
  - CCPR2H Register ... 134
  - CCPR2L Register ... 134
- Compare Mode. *See* Compare.
- Interaction of Two CCP Modules ... 134
- PWM Mode. *See* PWM.
- Timer Resources ... 134

Clock Sources ... 24
- Selection Using OSCCON Register ... 24

Clocking Scheme/Instruction Cycle ... 57
CLRF ... 271
CLRWDT ... 271
Code Examples
- 16 x 16 Signed Multiply Routine ... 86
- 16 x 16 Unsigned Multiply Routine ... 86
- 8 x 8 Signed Multiply Routine ... 85
- 8 x 8 Unsigned Multiply Routine ... 85
- Changing Between Capture Prescalers ... 135
- Computed GOTO Using an Offset Value ... 59
- Data EEPROM Read ... 83
- Data EEPROM Refresh Routine ... 84
- Data EEPROM Write ... 83
- Erasing a Flash Program Memory Row ... 76
- Fast Register Stack ... 56
- How to Clear RAM (Bank 1) Using Indirect Addressing ... 66
- Implementing a Real-Time Clock Using a Timer1 Interrupt Service ... 125
- Initializing PORTA ... 101
- Initializing PORTB ... 104
- Initializing PORTC ... 107
- Initializing PORTD ... 109
- Initializing PORTE ... 111
- Loading the SSPBUF (SSPSR) Register ... 158
- Reading a Flash Program Memory Word ... 75
- Saving Status, WREG and BSR Registers in RAM ... 99
- Writing to Flash Program Memory ... 78–79

Code Protection ... 237, 251
COMF ... 272
Comparator ... 221
- Analog Input Connection Considerations ... 225
- Associated Registers ... 226
- Configuration ... 221
- Effects of a Reset ... 225
- Interrupts ... 224
- Operation ... 223
- Operation in Power Managed Modes ... 225
- Outputs ... 223
- Reference ... 223
- Response Time ... 223

Comparator Specifications ... 321
Comparator Voltage Reference ... 227
- Accuracy and Error ... 228
- Associated Registers ... 229
- Configuring ... 227
- Connection Considerations ... 228
- Effects of a Reset ... 228
- Operation in Power Managed Modes ... 228

Compare (CCP Module) ... 136
- Associated Registers ... 137
- CCP Pin Configuration ... 136
- CCPR1 Register ... 136
- Software Interrupt ... 136
- Special Event Trigger ... 136, 220
- Timer1/Timer3 Mode Selection ... 136

Compare (ECCP Mode) ... 142
Computed GOTO ... 59
Configuration Bits ... 237
Configuration Register Protection ... 254
Context Saving During Interrupts ... 99
Control Registers
- EECON1 and EECON2 ... 72

Conversion Considerations ... 370
CPFSEQ ... 272
CPFSGT ... 273
CPFSLT ... 273
Crystal Oscillator/Ceramic Resonator ... 19


## D


Data EEPROM Code Protection ... 254
Data EEPROM Memory ... 81
- Associated Registers ... 84
- EEADR Register ... 81
- EECON1 and EECON2 Registers ... 81
- Operation During Code-Protect ... 84
- Protection Against Spurious Write ... 83
- Reading ... 83
- Using ... 84
- Write Verify ... 83
- Writing ... 83

Data Memory ... 59
- General Purpose Registers ... 59
- Map for PIC18F2X20/4X20 ... 60
- Special Function Registers ... 61

DAW ... 274
DC and AC Characteristics
- Graphs and Tables ... 343

DC Characteristics ... 318
- Power-Down and Supply Current ... 309
- Supply Voltage ... 308

DCFSNZ ... 275
DECF ... 274
DECFSZ ... 275



© 2003 Microchip Technology Inc.

Demonstration Boards
PICDEM 1 ... 302
PICDEM 17 ... 302
PICDEM 18R PIC18C601/801 ... 303
PICDEM 2 Plus ... 302
PICDEM 3 PIC16C92X ... 302
PICDEM 4 ... 302
PICDEM LIN PIC16C43X ... 303
PICDEM USB PIC16C7X5 ... 303
PICDEM.net Internet/Ethernet ... 302
Development Support ... 299
Device Differences ... 369
Device Overview ... 7
Features (table) ... 8
New Core Features ... 7
Other Special Features ... 7
Direct Addressing ... 67

## E

ECCP ... 141
Auto-Shutdown ... 149
and Automatic Restart ... 151
Capture and Compare Modes ... 142
Outputs ... 142
Standard PWM Mode ... 142
Start-up Considerations ... 151
Effects of Power Managed Modes on
Various Clock Sources ... 27
Electrical Characteristics ... 305
Enhanced Capture/Compare/PWM (ECCP) ... 141
Capture Mode. *See* Capture (ECCP Module).
PWM Mode. *See* PWM (ECCP Module).
Enhanced CCP Auto-Shutdown ... 149
Enhanced PWM Mode. *See* PWM (ECCP Module).
Equations
16 x 16 Signed Multiplication Algorithm ... 86
16 x 16 Unsigned Multiplication Algorithm ... 86
A/D Acquisition Time ... 216
A/D Minimum Holding Capacitor ... 216
Errata ... 5
Evaluation and Programming Tools ... 303
External Clock Input ... 21

## F

Fail-Safe Clock Monitor ... 237, 248
Interrupts in Power Managed Modes ... 250
POR or Wake-up from Sleep ... 250
WDT During Oscillator Failure ... 248
Fast Register Stack ... 56
Firmware Instructions ... 255
Flash Program Memory ... 71
Associated Registers ... 79
Control Registers ... 72
Erase Sequence ... 76
Erasing ... 76
Operation During Code-Protect ... 79
Reading ... 75
TABLAT Register ... 74
Table Pointer ... 74
Boundaries Based on Operation ... 74
Table Pointer Boundaries ... 74
Table Reads and Table Writes ... 71
Unexpected Termination of Write Operation ... 79
Write Verify ... 79
Writing to ... 77
FSCM. *See* Fail-Safe Clock Monitor.

## G

GOTO ... 276

## H

Hardware Multiplier ... 85
Introduction ... 85
Operation ... 85
Performance Comparison ... 85
HSPLL ... 20

## I

I/O Ports ... 101
$I^2C$ Mode
$\overline{ACK}$ Pulse ... 168, 169
Acknowledge Sequence Timing ... 188
Baud Rate Generator ... 181
Bus Collision During a Repeated
Start Condition ... 192
Bus Collision During a Start Condition ... 190
Bus Collision During a Stop Condition ... 193
Clock Arbitration ... 182
Clock Stretching ... 174
Effect of a Reset ... 189
General Call Address Support ... 178
Master Mode ... 179
Master Mode (Reception, 7-bit Address) ... 187
Master Mode Operation ... 180
Master Mode Reception ... 185
Master Mode Repeated Start
Condition Timing ... 184
Master Mode Start Condition Timing ... 183
Master Mode Transmission ... 185
Multi-Master Communication, Bus Collision
and Bus Arbitration ... 189
Multi-Master Mode ... 189
Operation ... 168
Operation in Power Managed Mode ... 189
Read/Write Bit Information (R/$\overline{W}$ Bit) ... 168, 169
Registers ... 164
Serial Clock (RC3/SCK/SCL) ... 169
Slave Mode ... 168
Addressing ... 168
Reception ... 169
Transmission ... 169
Stop Condition Timing ... 188
ID Locations ... 237, 254
INCF ... 276
INCFSZ ... 277
In-Circuit Debugger ... 254
In-Circuit Serial Programming (ICSP) ... 237, 254
Indirect Addressing
INDF and FSR Registers ... 66
Operation ... 66
Indirect Addressing Operation ... 67
Indirect File Operand ... 59
INFSNZ ... 277
Initialization Conditions for all Registers ... 46–49
Instruction Cycle ... 57
Instruction Flow/Pipelining ... 57
Instruction Format ... 257

Instruction Set ..... 255
ADDLW ..... 261
ADDWF ..... 261
ADDWFC ..... 262
ANDLW ..... 262
ANDWF ..... 263
BC ..... 263
BCF ..... 264
BN ..... 264
BNC ..... 265
BNN ..... 265
BNOV ..... 266
BNZ ..... 266
BOV ..... 269
BRA ..... 267
BSF ..... 267
BTFSC ..... 268
BTFSS ..... 268
BTG ..... 269
BZ ..... 270
CALL ..... 270
CLRF ..... 271
CLRWDT ..... 271
COMF ..... 272
CPFSEQ ..... 272
CPFSGT ..... 273
CPFSLT ..... 273
DAW ..... 274
DCFSNZ ..... 275
DECF ..... 274
DECFSZ ..... 275
GOTO ..... 276
INCF ..... 276
INCFSZ ..... 277
INFSNZ ..... 277
IORLW ..... 278
IORWF ..... 278
LFSR ..... 279
MOVF ..... 279
MOVFF ..... 280
MOVLB ..... 280
MOVLW ..... 281
MOVWF ..... 281
MULLW ..... 282
MULWF ..... 282
NEGF ..... 283
NOP ..... 283
POP ..... 284
PUSH ..... 284
RCALL ..... 285
Reset ..... 285
RETFIE ..... 286
RETLW ..... 286
RETURN ..... 287
RLCF ..... 287
RLNCF ..... 288
RRCF ..... 288
RRNCF ..... 289
SETF ..... 289
SLEEP ..... 290
SUBFWB ..... 290
SUBLW ..... 291
SUBWF ..... 291
SUBWFB ..... 292
SWAPF ..... 293
TBLRD ..... 294
TBLWT ..... 295
TSTFSZ ..... 296
XORLW ..... 296
XORWF ..... 297
Summary Table ..... 258
INTCON Register
RBIF Bit ..... 104
INTCON Registers ..... 89
Inter-Integrated Circuit. *See* $I^2C$.
Internal Oscillator Block ..... 22
Adjustment ..... 22
INTIO Modes ..... 22
INTRC Output Frequency ..... 22
OSCTUNE Register ..... 22
Internal RC Oscillator
Use with WDT ..... 245
Interrupt Sources ..... 237
A/D Conversion Complete ..... 215
Capture Complete (CCP) ..... 135
Compare Complete (CCP) ..... 136
Interrupt-on-Change (RB7:RB4) ..... 104
INTn Pin ..... 99
PORTB, Interrupt-on-Change ..... 99
TMR0 ..... 99
TMR1 Overflow ..... 121
TMR2 to PR2 Match ..... 128
TMR2 to PR2 Match (PWM) ..... 127, 138
TMR3 Overflow ..... 129, 131
USART Receive/Transmit Complete ..... 195
Interrupts ..... 87
Interrupts, Enable Bits
CCP1 Enable (CCP1IE Bit) ..... 135
Interrupts, Flag Bits
CCP1 Flag (CCP1IF Bit) ..... 135
CCP1IF Flag (CCP1IF Bit) ..... 136
Interrupt-on-Change (RB7:RB4) Flag (RBIF Bit) ..... 104
INTOSC Frequency Drift ..... 40
INTOSC, INTRC. *See* Internal Oscillator Block.
IORLW ..... 278
IORWF ..... 278
IPR Registers ..... 96

## L

LFSR ..... 279
Look-up Tables ..... 59
Low-Voltage Detect ..... 231
Characteristics ..... 322
Effects of a Reset ..... 235
Operation ..... 234
Current Consumption ..... 235
Reference Voltage Set Point ..... 235
Operation During Sleep ..... 235
Low-Voltage ICSP Programming ..... 254
LVD. *See* Low-Voltage Detect.


© 2003 Microchip Technology Inc.

## M


Master Synchronous Serial Port (MSSP). *See* MSSP.
Memory Organization ..... 53
    Data Memory ..... 59
    Program Memory ..... 53
Memory Programming Requirements ..... 320
Migration from Baseline to Enhanced Devices ..... 370
Migration from High-End to Enhanced Devices ..... 371
Migration from Mid-Range to Enhanced Devices ..... 371
MOVF ..... 279
MOVFF ..... 280
MOVLB ..... 280
MOVLW ..... 281
MOVWF ..... 281
MPLAB ASM30 Assembler, Linker, Librarian ..... 300
MPLAB ICD 2 In-Circuit Debugger ..... 301
MPLAB ICE 2000 High Performance
    Universal In-Circuit Emulator ..... 301
MPLAB ICE 4000 High Performance
    Universal In-Circuit Emulator ..... 301
MPLAB Integrated Development
    Environment Software ..... 299
MPLINK Object Linker/MPLIB Object Librarian ..... 300
MSSP ..... 155
    Control Registers (General) ..... 155
    Enabling SPI I/O ..... 159
    I2C Master Mode ..... 179
    I2C Mode
    I2C Slave Mode ..... 168
    Operation ..... 158
    Overview ..... 155
    Slave Select Control ..... 161
    SPI Master Mode ..... 160
    SPI Master/Slave Connection ..... 159
    SPI Mode ..... 155
    SPI Slave Mode ..... 161
    Typical Connection ..... 159
MULLW ..... 282
MULWF ..... 282


## N


NEGF ..... 283
NOP ..... 283


## O


Opcode Field Descriptions ..... 256
OPTION_REG Register
    PSA Bit ..... 119
    T0CS Bit ..... 119
    T0PS2:T0PS0 Bits ..... 119
    T0SE Bit ..... 119
Oscillator Configuration ..... 19
    EC ..... 19
    ECIO ..... 19
    HS ..... 19
    HSPLL ..... 19
    Internal Oscillator Block ..... 22
    INTIO1 ..... 19
    INTIO2 ..... 19
    LP ..... 19
    RC ..... 19
    RCIO ..... 19
    XT ..... 19
Oscillator Selection ..... 237
Oscillator Start-up Timer (OST) ..... 27, 44, 237
Oscillator Switching ..... 24
Oscillator Transitions ..... 27
Oscillator, Timer1 ..... 121, 131
Oscillator, Timer3 ..... 129


## P


Packaging Information ..... 361
    Marking ..... 361, 362
Parallel Slave Port (PSP) ..... 109, 114
    Associated Registers ..... 115
    CS (Chip Select) ..... 113, 114
    PORTD ..... 114
    RD (Read Input) ..... 113, 114
    RE0/AN5/RD Pin ..... 113
    RE1/AN6/WR Pin ..... 113
    RE2/AN7/CS Pin ..... 113
    Select (PSPMODE Bit) ..... 109, 114
    WR (Write Input) ..... 113, 114
PICkit 1 Flash Starter Kit ..... 303
PICSTART Plus Development Programmer ..... 301
PIE Registers ..... 94
Pin Functions
    MCLR/VPP/RE3 ..... 11, 14
    OSC1/CLKI/RA7 ..... 11, 14
    OSC2/CLKO/RA6 ..... 11, 14
    RA0/AN0 ..... 11, 14
    RA1/AN1 ..... 11, 14
    RA2/AN2/VREF-/CVREF ..... 11, 14
    RA3/AN3/VREF+ ..... 11, 14
    RA4/T0CKI/C1OUT ..... 11, 14
    RA5/AN4/SS/LVDIN/C2OUT ..... 11, 14
    RB0/AN12/INT0 ..... 12, 15
    RB1/AN10/INT1 ..... 12, 15
    RB2/AN8/INT2 ..... 12, 15
    RB3/AN9/CCP2 ..... 12, 15
    RB4/AN11/KBI0 ..... 12, 15
    RB5/KBI1/PGM ..... 12, 15
    RB6/KBI2/PGC ..... 12, 15
    RB7/KBI3/PGD ..... 12
    RB7/PGD ..... 15
    RC0/T1OSO/T1CKI ..... 13, 16
    RC1/T1OSI/CCP2 ..... 13, 16
    RC2/CCP1/P1A ..... 13, 16
    RC3/SCK/SCL ..... 13, 16
    RC4/SDI/SDA ..... 13, 16
    RC5/SDO ..... 13, 16
    RC6/TX/CK ..... 13, 16
    RC7/RX/DT ..... 13, 16
    RD0/PSP0 ..... 17
    RD1/PSP1 ..... 17
    RD2/PSP2 ..... 17
    RD3/PSP3 ..... 17
    RD4/PSP4 ..... 17
    RD5/PSP5/P1B ..... 17
    RD6/PSP6/P1C ..... 17
    RD7/PSP7/P1D ..... 17
    RE0/AN5/RD ..... 18
    RE1/AN6/WR ..... 18
    RE2/AN7/CS ..... 18
    RE3 ..... 18
    VDD ..... 13, 18
    VSS ..... 13, 18

Pinout I/O Descriptions
PIC18F2220/2320 ..... 11
PIC18F4220/4320 ..... 14
PIR Registers ..... 92
PLL Lock Time-out ..... 44
Pointer, FSRn ..... 66
POP ..... 284
POR. *See* Power-on Reset.
PORTA
Associated Registers ..... 103
LATA Register ..... 101
PORTA Register ..... 101
TRISA Register ..... 101
PORTB
Associated Registers ..... 106
LATB Register ..... 104
PORTB Register ..... 104
RB7:RB4 Interrupt-on-Change Flag (RBIF Bit) ..... 104
TRISB Register ..... 104
PORTC
Associated Registers ..... 108
LATC Register ..... 107
PORTC Register ..... 107
TRISC Register ..... 107
PORTD
Associated Registers ..... 110
LATD Register ..... 109
Parallel Slave Port (PSP) Function ..... 109
PORTD Register ..... 109
TRISD Register ..... 109
PORTE
Analog Port Pins ..... 113
Associated Registers ..... 113
LATE Register ..... 111
PORTE Register ..... 111
PSP Mode Select (PSPMODE Bit) ..... 109
RE0/AN5/$\overline{RD}$ Pin ..... 113
RE1/AN6/$\overline{WR}$ Pin ..... 113
RE2/AN7/$\overline{CS}$ Pin ..... 113
TRISE Register ..... 111
Postscaler, WDT
Assignment (PSA Bit) ..... 119
Rate Select (T0PS2:T0PS0 Bits) ..... 119
Power Managed Modes ..... 29
Entering ..... 30
Idle Modes ..... 31
Run Modes ..... 36
Selecting ..... 29
Sleep Mode ..... 31
Summary (table) ..... 29
Wake-up from ..... 38
Power-on Reset (POR) ..... 44, 237
Power-up Delays ..... 27
Power-up Timer (PWRT) ..... 27, 44, 237
Prescaler, Capture ..... 135
Prescaler, Timer0 ..... 119
Assignment (PSA Bit) ..... 119
Rate Select (T0PS2:T0PS0 Bits) ..... 119
Prescaler, Timer2 ..... 139
PRO MATE II Universal Device Programmer ..... 301
Product Identification System ..... 385
Program Counter
PCL Register ..... 56
PCLATH Register ..... 56
PCLATU Register ..... 56
Program Memory
Instructions ..... 58
Two-Word ..... 58
Interrupt Vector ..... 53
Map and Stack for PIC18F2220/4220 ..... 53
Map and Stack for PIC18F2320/4320 ..... 53
Reset Vector ..... 53
Program Memory Code Protection ..... 252
Program Verification ..... 251
Program Verification and Code Protection
Associated Registers ..... 251
Programming, Device Instructions ..... 255
PSP. *See* Parallel Slave Port.
Pulse Width Modulation. *See* PWM (CCP Module) and PWM (ECCP Module).
PUSH ..... 284
PUSH and POP Instructions ..... 55
PWM (CCP Module) ..... 138
Associated Registers ..... 139
CCPR1H:CCPR1L Registers ..... 138
Duty Cycle ..... 138
Example Frequencies/Resolutions ..... 139
Period ..... 138
Setup for PWM Operation ..... 139
TMR2 to PR2 Match ..... 127, 138
PWM (ECCP Module) ..... 143
Associated Registers ..... 153
Direction Change in Full-Bridge Output Mode ..... 147
Effects of a Reset ..... 152
Full-Bridge Application Example ..... 147
Full-Bridge Mode ..... 146
Half-Bridge Mode ..... 145
Half-Bridge Output Mode Applications Example ..... 145
Operation in Power Managed Modes ..... 152
Operation with Fail-Safe Clock Monitor ..... 152
Output Configurations ..... 143
Output Relationships (Active-High State) ..... 144
Output Relationships (Active-Low State) ..... 144
Programmable Dead Band Delay ..... 149
Setup for Operation ..... 152
Shoot-Through Current ..... 149
Start-up Considerations ..... 151


## Q


Q Clock ..... 139


## R


RAM. *See* Data Memory.
RC Oscillator ..... 21
RCIO Oscillator Mode ..... 21
RCALL ..... 285
RCON Register
Bit Status During Initialization ..... 45
Bits and Positions ..... 45
RCSTA Register
SPEN Bit ..... 195
Register File ..... 59
Registers
ADCON0 (A/D Control 0) ..... 211
ADCON1 (A/D Control 1) ..... 212
ADCON2 (A/D Control 2) ..... 213
CCP1CON (Enhanced CCP Operation Control 1) ..... 141
CCPxCON (Capture/Compare/PWM Control) ..... 133
CMCON (Comparator Control) ..... 221
CONFIG1H (Configuration 1 High) ..... 238



© 2003 Microchip Technology Inc.

CONFIG2H (Configuration 2 High) ..... 239
CONFIG2L (Configuration 2 Low) ..... 239
CONFIG3H (Configuration 3 High) ..... 240
CONFIG4L (Configuration 4 Low) ..... 240
CONFIG5H (Configuration 5 High) ..... 241
CONFIG5L (Configuration 5 Low) ..... 241
CONFIG6H (Configuration 6 High) ..... 242
CONFIG6L (Configuration 6 Low) ..... 242
CONFIG7H (Configuration 7 High) ..... 243
CONFIG7L (Configuration 7 Low) ..... 243
CVRCON (Comparator Voltage Reference Control) ..... 227
Device ID Register 1 ..... 244
Device ID Register 2 ..... 244
ECCPAS (Enhanced CCP Auto-Shutdown Control) ..... 150
EECON1 (Data EEPROM Control 1) ..... 73, 82
INTCON (Interrupt Control) ..... 89
INTCON2 (Interrupt Control 2) ..... 90
INTCON3 (Interrupt Control 3) ..... 91
IPR1 (Peripheral Interrupt Priority 1) ..... 96
IPR2 (Peripheral Interrupt Priority 2) ..... 97
LVDCON (LVD Control) ..... 233
OSCCON (Oscillator Control) ..... 26
OSCTUNE (Oscillator Tuning) ..... 23
PIE1 (Peripheral Interrupt Enable 1) ..... 94
PIE2 (Peripheral Interrupt Enable 2) ..... 95
PIR1 (Peripheral Interrupt Request (Flag) 1) ..... 92
PIR2 (Peripheral Interrupt Request (Flag) 2) ..... 93
PWM1CON (Enhanced PWM Configuration) ..... 149
RCON (Reset Control) ..... 69, 98
RCSTA (Receive Status and Control) ..... 197
SSPCON1 (MSSP Control 1, $I^2C$ Mode) ..... 166
SSPCON1 (MSSP Control 1, SPI Mode) ..... 157
SSPCON2 (MSSP Control 2, $I^2C$ Mode) ..... 167
SSPSTAT (MSSP Status, $I^2C$ Mode) ..... 165
SSPSTAT (MSSP Status, SPI Mode) ..... 156
Status ..... 68
STKPTR (Stack Pointer) ..... 55
Summary ..... 62–64
T0CON (Timer0 Control) ..... 117
T1CON (Timer 1 Control) ..... 121
T2CON (Timer 2 Control) ..... 127
T3CON (Timer3 Control) ..... 129
TRISE ..... 112
TXSTA (Transmit Status and Control) ..... 196
WDTCON (Watchdog Timer Control) ..... 246
Reset ..... 43, 285
Resets ..... 237
RETFIE ..... 286
RETLW ..... 286
RETURN ..... 287
Return Address Stack ..... 54
Return Stack Pointer (STKPTR) ..... 54
Revision History ..... 369
RLCF ..... 287
RLNCF ..... 288
RRCF ..... 288
RRNCF ..... 289


## S


SCI. *See* USART.
SCK ..... 155
SDI ..... 155
SDO ..... 155
Serial Clock (SCK) Pin ..... 155
Serial Communication Interface. *See* USART.
Serial Data In (SDI) Pin ..... 155
Serial Data Out (SDO) Pin ..... 155
Serial Peripheral Interface. *See* SPI Mode.
SETF ..... 289
Shoot-Through Current ..... 149
Slave Select ($\overline{SS}$) Pin ..... 155
SLEEP ..... 290
Sleep
OSC1 and OSC2 Pin States ..... 27
Software Simulator (MPLAB SIM) ..... 300
Software Simulator (MPLAB SIM30) ..... 300
Special Event Trigger. *See* Compare (CCP Module)
Special Features of the CPU ..... 237
Special Function Registers ..... 61
Map ..... 61
SPI Mode
Associated Registers ..... 163
Bus Mode Compatibility ..... 163
Effects of a Reset ..... 163
Master in Power Managed Modes ..... 163
Master Mode ..... 160
Master/Slave Connection ..... 159
Registers ..... 156
Serial Clock ..... 155
Serial Data In ..... 155
Serial Data Out ..... 155
Slave in Power Managed Modes ..... 163
Slave Mode ..... 161
Slave Select ..... 155
SPI Clock ..... 160
$\overline{SS}$ ..... 155
SSP
$I^2C$ Mode. *See* $I^2C$.
SSPBUF Register ..... 160
SSPSR Register ..... 160
TMR2 Output for Clock Shift ..... 127, 128
SSPOV Status Flag ..... 185
SSPSTAT Register
R/$\overline{W}$ Bit ..... 168, 169
Stack Full/Underflow Resets ..... 55
SUBFWB ..... 290
SUBLW ..... 291
SUBWF ..... 291
SUBWFB ..... 292
SWAPF ..... 293


## T


TABLAT Register ..... 74
Table Pointer Operations (table) ..... 74
Table Reads/Table Writes ..... 59
TBLPTR Register ..... 74
TBLRD ..... 294
TBLWT ..... 295
Time-out in Various Situations (table) ..... 45
Time-out Sequence ..... 44

Timer0 ..... 117
16-bit Mode Timer Reads and Writes ..... 119
Associated Registers ..... 119
Clock Source Edge Select (T0SE Bit) ..... 119
Clock Source Select (T0CS Bit) ..... 119
Interrupt ..... 119
Operation ..... 119
Prescaler. *See* Prescaler, Timer0.
Switching Prescaler Assignment ..... 119
Timer1 ..... 121
16-bit Read/Write Mode ..... 124
Associated Registers ..... 125
Interrupt ..... 124
Operation ..... 122
Oscillator ..... 121, 123
Oscillator Layout Considerations ..... 123
Overflow Interrupt ..... 121
Resetting, Using a Special Event Trigger Output (CCP) ..... 124
Special Event Trigger (CCP) ..... 136
TMR1H Register ..... 121
TMR1L Register ..... 121
Use as a Real-Time Clock ..... 124
Timer2 ..... 127
Associated Registers ..... 128
Operation ..... 127
Postscaler. *See* Postscaler, Timer2.
PR2 Register ..... 127, 138
Prescaler. *See* Prescaler, Timer2.
SSP Clock Shift ..... 127, 128
TMR2 Register ..... 127
TMR2 to PR2 Match Interrupt ..... 127, 128, 138
Timer3 ..... 129
Associated Registers ..... 131
Operation ..... 130
Oscillator ..... 129, 131
Overflow Interrupt ..... 129, 131
Resetting, Using a Special Event Trigger Output (CCP) ..... 131
TMR3H Register ..... 129
TMR3L Register ..... 129
Timing Diagrams
A/D Conversion ..... 342
Acknowledge Sequence ..... 188
Asynchronous Reception ..... 205
Asynchronous Transmission ..... 203
Asynchronous Transmission (Back to Back) ..... 203
Baud Rate Generator with Clock Arbitration ..... 182
BRG Reset Due to SDA Arbitration During Start Condition ..... 191
Brown-out Reset (BOR) ..... 328
Bus Collision During a Repeated Start Condition (Case 1) ..... 192
Bus Collision During a Repeated Start Condition (Case 2) ..... 192
Bus Collision During a Stop Condition (Case 1) ..... 193
Bus Collision During a Stop Condition (Case 2) ..... 193
Bus Collision During Start Condition (SCL = 0) ..... 191
Bus Collision During Start Condition (SDA Only) ..... 190
Bus Collision for Transmit and Acknowledge ..... 189
Capture/Compare/PWM (CCP) ..... 330
CLKO and I/O ..... 327
Clock Synchronization ..... 175
Clock, Instruction Cycle ..... 57
Example SPI Master Mode (CKE = 0) ..... 332
Example SPI Master Mode (CKE = 1) ..... 333
Example SPI Slave Mode (CKE = 0) ..... 334
Example SPI Slave Mode (CKE = 1) ..... 335
External Clock (All Modes except PLL) ..... 325
Fail-Safe Clock Monitor (FSCM) ..... 249
First Start Bit ..... 183
Full-Bridge PWM Output ..... 146
Half-Bridge PWM Output ..... 145
$I^2C$ Bus Data ..... 336
$I^2C$ Bus Start/Stop Bits ..... 336
$I^2C$ Master Mode (Transmission, 7 or 10-bit Address) ..... 186
$I^2C$ Slave Mode (Transmission, 10-bit Address) ..... 173
$I^2C$ Slave Mode (Transmission, 7-bit Address) ..... 171
$I^2C$ Slave Mode with SEN = 0 (Reception, 10-bit Address) ..... 172
$I^2C$ Slave Mode with SEN = 0 (Reception, 7-bit Address) ..... 170
$I^2C$ Slave Mode with SEN = 1 (Reception, 10-bit Address) ..... 177
$I^2C$ Slave Mode with SEN = 1 (Reception, 7-bit Address) ..... 176
Low-Voltage Detect ..... 234
Low-Voltage Detect Characteristics ..... 322
Master SSP $I^2C$ Bus Data ..... 338
Master SSP $I^2C$ Bus Start/Stop Bits ..... 338
Parallel Slave Port (PIC18F4X20) ..... 331
Parallel Slave Port (PSP) Read ..... 115
Parallel Slave Port (PSP) Write ..... 115
PWM Auto-Shutdown (PRSEN = 0, Auto-Restart Disabled) ..... 151
PWM Auto-Shutdown (PRSEN = 1, Auto-Restart Enabled) ..... 151
PWM Direction Change ..... 148
PWM Direction Change at Near 100% Duty Cycle ..... 148
PWM Output ..... 138
Repeat Start Condition ..... 184
Reset, Watchdog Timer (WDT), Oscillator Start-up Timer (OST), Power-up Timer (PWRT) ..... 328
Slave Mode General Call Address Sequence (7 or 10-bit Address Mode) ..... 178
Slave Synchronization ..... 161
Slow Rise Time ($\overline{MCLR}$ Tied to $V_{DD}$, $V_{DD}$ Rise > $T_{PWRT}$) ..... 51
SPI Mode (Master Mode) ..... 160
SPI Mode (Slave Mode with CKE = 0) ..... 162
SPI Mode (Slave Mode with CKE = 1) ..... 162
Stop Condition Receive or Transmit Mode ..... 188
Synchronous Transmission ..... 206
Synchronous Transmission (Through TXEN) ..... 207
Time-out Sequence on POR w/ PLL Enabled ($\overline{MCLR}$ Tied to $V_{DD}$) ..... 51
Time-out Sequence on Power-up ($\overline{MCLR}$ Not Tied to $V_{DD}$): Case 1 ..... 50
Time-out Sequence on Power-up ($\overline{MCLR}$ Not Tied to $V_{DD}$): Case 2 ..... 50
Time-out Sequence on Power-up ($\overline{MCLR}$ Tied to $V_{DD}$, $V_{DD}$ Rise $T_{PWRT}$) ..... 50



© 2003 Microchip Technology Inc.

Timer0 and Timer1 External Clock ..... 329
Transition for Entry to SEC_IDLE Mode ..... 34
Transition for Entry to SEC_RUN Mode ..... 36
Transition for Entry to Sleep Mode ..... 32
Transition for Two-Speed Start-up (INTOSC to HSPLL) ..... 247
Transition for Wake from PRI_IDLE Mode ..... 33
Transition for Wake from RC_RUN Mode (RC_RUN to PRI_RUN) ..... 35
Transition for Wake from SEC_RUN Mode (HSPLL) ..... 34
Transition for Wake from Sleep (HSPLL) ..... 32
Transition to PRI_IDLE Mode ..... 33
Transition to RC_IDLE Mode ..... 35
Transition to RC_RUN Mode ..... 37
USART Synchronous Receive (Master/Slave) ..... 340
USART Synchronous Reception (Master Mode, SREN) ..... 208
USART SynchronousTransmission (Master/Slave) ..... 340
Timing Diagrams and Specifications ..... 325
A/D Conversion Requirements ..... 342
Capture/Compare/PWM Requirements ..... 330
CLKO and I/O Requirements ..... 327
DC Characteristics - Internal RC Accuracy ..... 326
Example SPI Mode Requirements (Master Mode, CKE = 0) ..... 332
Example SPI Mode Requirements (Master Mode, CKE = 1) ..... 333
Example SPI Mode Requirements (Slave Mode, CKE = 0) ..... 334
Example SPI Slave Mode Requirements (CKE = 1) ..... 335
External Clock Requirements ..... 325
$I^2C$ Bus Data Requirements (Slave Mode) ..... 337
Master SSP $I^2C$ Bus Data Requirements ..... 339
Master SSP $I^2C$ Bus Start/Stop Bits Requirements ..... 338
Parallel Slave Port Requirements (PIC18F4X20) ..... 331
PLL Clock ..... 326
Reset, Watchdog Timer, Oscillator Start-up Timer, Power-up Timer and Brown-out Reset Requirements ..... 328
Timer0 and Timer1 External Clock Requirements ..... 329
USART Synchronous Receive Requirements ..... 340
USART Synchronous Transmission Requirements ..... 340
Top-of-Stack Access ..... 54
TRISE Register
PSPMODE Bit ..... 109
TSTFSZ ..... 296
Two-Speed Start-up ..... 237, 247
Two-Word Instructions
Example Cases ..... 58
TXSTA Register
BRGH Bit ..... 198

**U**

USART ..... 195
Asynchronous Mode ..... 202
Associated Registers, Receive ..... 205
Associated Registers, Transmit ..... 203
Receiver ..... 204
Transmitter ..... 202
Baud Rate Generator (BRG) ..... 198
Associated Registers ..... 198
Baud Rate Formula ..... 198
Baud Rates, Asynchronous Mode (BRGH = 0, Low Speed) ..... 199
Baud Rates, Asynchronous Mode (BRGH = 1, High Speed) ..... 200
Baud Rates, Synchronous Mode (SYNC = 1) ..... 201
High Baud Rate Select (BRGH Bit) ..... 198
Operation in Power Managed Mode ..... 198
Sampling ..... 198
Serial Port Enable (SPEN Bit) ..... 195
Setting Up 9-bit Mode with Address Detect ..... 204
Synchronous Master Mode ..... 206
Associated Registers, Reception ..... 208
Associated Registers, Transmit ..... 207
Reception ..... 208
Transmission ..... 206
Synchronous Slave Mode ..... 209
Associated Registers, Receive ..... 210
Associated Registers, Transmit ..... 209
Reception ..... 210
Transmission ..... 209

**V**

Voltage Reference Specifications ..... 321

**W**

Watchdog Timer (WDT) ..... 237, 245
Associated Registers ..... 246
Control Register ..... 245
During Oscillator Failure ..... 248
Programming Considerations ..... 245
WCOL ..... 183
WCOL Status Flag ..... 183, 185, 188
WWW, On-Line Support ..... 5

**X**

XORLW ..... 296
XORWF ..... 297

**NOTES:**


© 2003 Microchip Technology Inc.

## ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® or Microsoft® Internet Explorer. Files are also available for FTP download from our FTP site.

### Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

**www.microchip.com**

The file transfer site is available by using an FTP service to connect to:

**ftp://ftp.microchip.com**

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

## SYSTEMS INFORMATION AND UPGRADE HOT LINE

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits. The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

042003

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC18F2220/2320/4220/4320 Literature Number: DS39599C

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?


© 2003 Microchip Technology Inc.

## PIC18F2220/2320/4220/4320 PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

| PART NO. | – | X | /XX | XXX |
|---|---|---|---|---|
| Device | | Temperature Range | Package | Pattern |

| | |
|---|---|
| Device | PIC18F2220/2320/4220/4320(1), PIC18F2220/2320/4220/4320T(1,2); VDD range 4.2V to 5.5V<br><br>PIC18LF2220/2320/4220/4320(1), PIC18LF2220/2320/4220/4320T(1,2); VDD range 2.0V to 5.5V |
| Temperature Range | I = -40°C to +85°C (Industrial) |
| Package | PT = TQFP (Thin Quad Flatpack)<br>SO = SOIC<br>SP = Skinny Plastic DIP<br>P = PDIP<br>ML = QFN |
| Pattern | QTP, SQTP, Code or Special Requirements (blank otherwise) |

**Examples:**

a) PIC18LF4320-I/P 301 = Industrial temp., PDIP package, Extended VDD limits, QTP pattern #301.

b) PIC18LF2220-I/SO = Industrial temp., SOIC package, Extended VDD limits.

c) PIC18F4220-I/P = Industrial temp., PDIP package, normal VDD limits.

**Note 1:** F = Standard Voltage Range
LF = Wide Voltage Range

**2:** T = in tape and reel – SOIC and TQFP packages only.

# WORLDWIDE SALES AND SERVICE

## AMERICAS

**Corporate Office**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

**Atlanta**
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034
Fax: 770-640-0307

**Boston**
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848
Fax: 978-692-3821

**Chicago**
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071
Fax: 630-285-0075

**Dallas**
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423
Fax: 972-818-2924

**Detroit**
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250
Fax: 248-538-2260

**Kokomo**
2767 S. Albright Road
Kokomo, IN 46902
Tel: 765-864-8360
Fax: 765-864-8387

**Los Angeles**
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888
Fax: 949-263-1338

**Phoenix**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966
Fax: 480-792-4338

**San Jose**
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950
Fax: 408-436-7955

**Toronto**
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699
Fax: 905-673-6509

## ASIA/PACIFIC

**Australia**
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

**China - Beijing**
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100
Fax: 86-10-85282104

**China - Chengdu**
Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200
Fax: 86-28-86766599

**China - Fuzhou**
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506
Fax: 86-591-7503521

**China - Hong Kong SAR**
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

**China - Shanghai**
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

**China - Shenzhen**
Rm. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380
Fax: 86-755-8295-1393

**China - Shunde**
Room 401, Hongjian Building
No. 2 Fengxiangnan Road, Ronggui Town
Shunde City, Guangdong 528303, China
Tel: 86-765-8395507 Fax: 86-765-8395571

**China - Qingdao**
Rm. B505A, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

**India**
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

**Japan**
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

**Korea**
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5932 or
82-2-558-5934

**Singapore**
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

**Taiwan**
Kaohsiung Branch
30F - 1 No. 8
Min Chuan 2nd Road
Kaohsiung 806, Taiwan
Tel: 886-7-536-4818
Fax: 886-7-536-4803

**Taiwan**
Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

## EUROPE

**Austria**
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

**Denmark**
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45-4420-9895 Fax: 45-4420-9910

**France**
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

**Germany**
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

**Italy**
Via Quasimodo, 12
20025 Legnano (MI)
Milan, Italy
Tel: 39-0331-742611
Fax: 39-0331-466781

**Netherlands**
P. A. De Biesbosch 14
NL-5152 SC Drunen, Netherlands
Tel: 31-416-690399
Fax: 31-416-690340

**United Kingdom**
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-118-921-5869
Fax: 44-118-921-5820

07/28/03


© 2003 Microchip Technology Inc.