



PIC18F45J10 Family Data Sheet

28/40/44-Pin High-Performance,
RISC Microcontrollers
with nanoWatt Technology

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

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

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. 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in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. 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 Microchip intellectual property rights.

Trademarks

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


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

Analog-for-the-Digital Age, Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, rfLAB, rfPICDEM, Select Mode, Smart Serial, SmartTel, Total Endurance and WiperLock are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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.

© 2005, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949:2002 ==

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.



PIC18F45J10 FAMILY

28/40/44-Pin High-Performance, RISC Microcontrollers with nanoWatt Technology

Special Microcontroller Features:

- Operating voltage range: 2.0V to 3.6V
- 5.5V tolerant input (digital pins only)
- On-chip 2.5V regulator
- Low-power, high-speed CMOS Flash technology
- C compiler optimized architecture:
 - Optional extended instruction set designed to optimize re-entrant code
- Priority levels for interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
 - Programmable period from 4 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) with three Break points via two pins
- Power-Managed modes:
 - Run: CPU on, peripherals on
 - Idle: CPU off, peripherals on
 - Sleep: CPU off, peripherals off

Flexible Oscillator Structure:

- Two Crystal modes, up to 40 MHz
- Two External Clock modes, up to 40 MHz
- Internal 31 kHz oscillator
- Secondary oscillator using Timer1 @ 32 kHz
- Two-Speed Oscillator Start-up
- Fail-Safe Clock Monitor:
 - Allows for safe shutdown if peripheral clock stops

Peripheral Highlights:

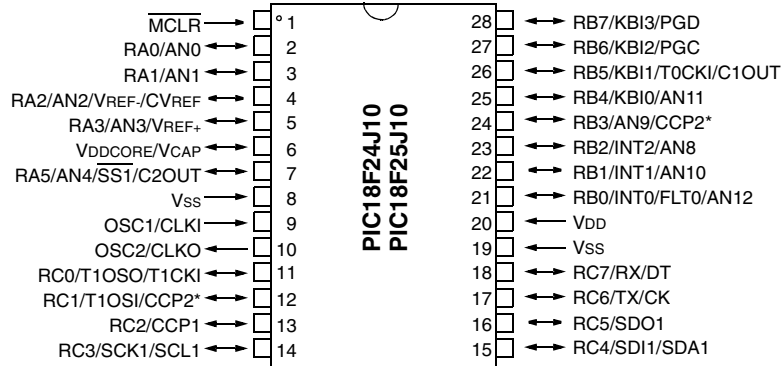
- High-current sink/source 25 mA/25 mA (PORTB and PORTC)
- Three programmable external interrupts
- Four input change interrupts
- One Capture/Compare/PWM (CCP) module
- One Enhanced Capture/Compare/PWM (ECCP) module:
 - One, two or four PWM outputs
 - Selectable polarity
 - Programmable dead time
 - Auto-Shutdown and Auto-Restart
- Two Master Synchronous Serial Port (MSSP) modules supporting 3-wire SPI™ (all 4 modes) and I²C™ Master and Slave modes
- One Enhanced Addressable USART module:
 - Supports RS-485, RS-232 and LIN 1.2
 - Auto-Wake-up on Start bit
 - Auto-Baud Detect
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D):
 - Auto-acquisition capability
 - Conversion available during Sleep
 - Self-calibration feature
- Dual analog comparators with input multiplexing

| Device | Program Memory | | SRAM Data Memory (bytes) | I/O | 10-bit A/D (ch) | CCP/ ECCP (PWM) | MSSP | | EUSART | Comparators | Timers 8/16-bit | |
|-------------|----------------|----------------------------|--------------------------|-----|-----------------|-----------------|------|--------------------------|--------|-------------|-----------------|-----|
| | Flash (bytes) | # Single-Word Instructions | | | | | SPI™ | Master I ² C™ | | | | |
| PIC18F24J10 | 16K | 8192 | 1024 | 21 | 10 | 2/0 | 1 | Y | Y | 1 | 2 | 1/2 |
| PIC18F25J10 | 32K | 16384 | 1024 | 21 | 10 | 2/0 | 1 | Y | Y | 1 | 2 | 1/2 |
| PIC18F44J10 | 16K | 8192 | 1024 | 32 | 13 | 1/1 | 2 | Y | Y | 1 | 2 | 1/2 |
| PIC18F45J10 | 32K | 16384 | 1024 | 32 | 13 | 1/1 | 2 | Y | Y | 1 | 2 | 1/2 |

PIC18F45J10 FAMILY

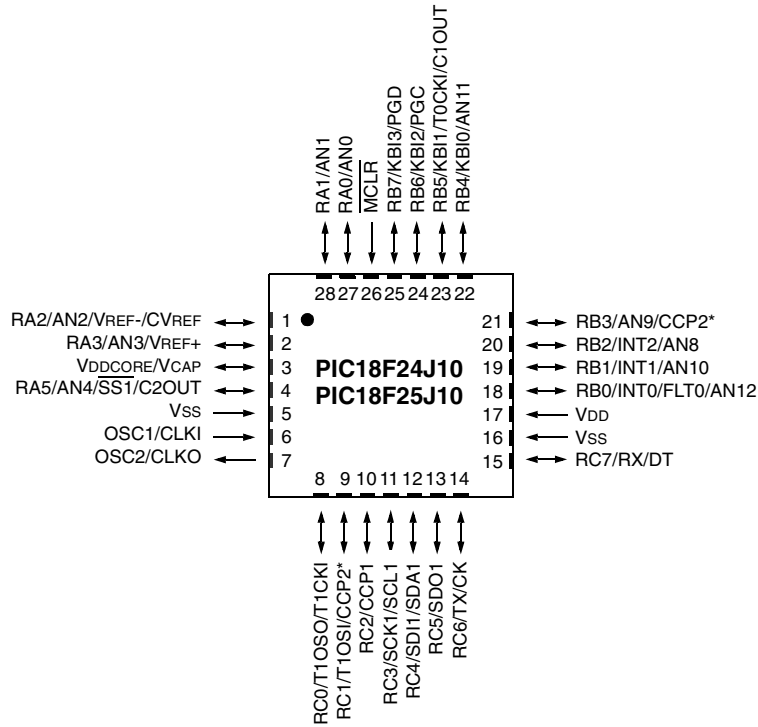
Pin Diagrams

28-Pin SPDIP, SOIC, SSOP (300 MIL)



* Pin feature is dependent on device configuration.

28-Pin QFN

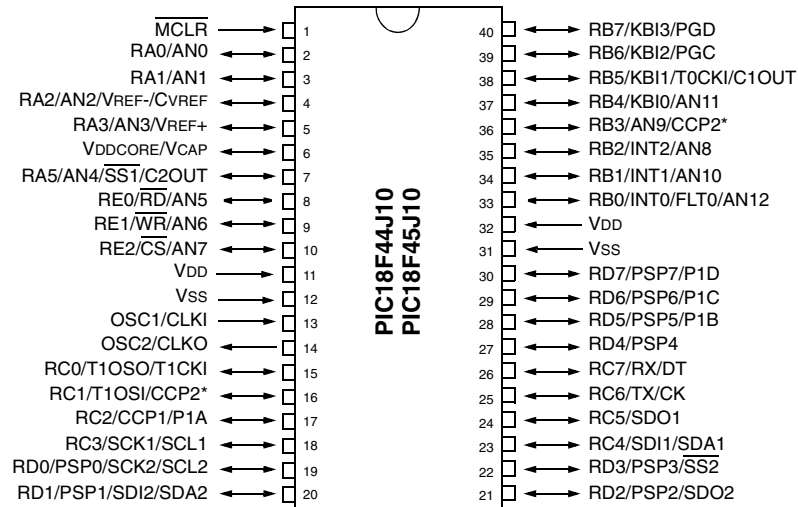


* Pin feature is dependent on device configuration.

PIC18F45J10 FAMILY

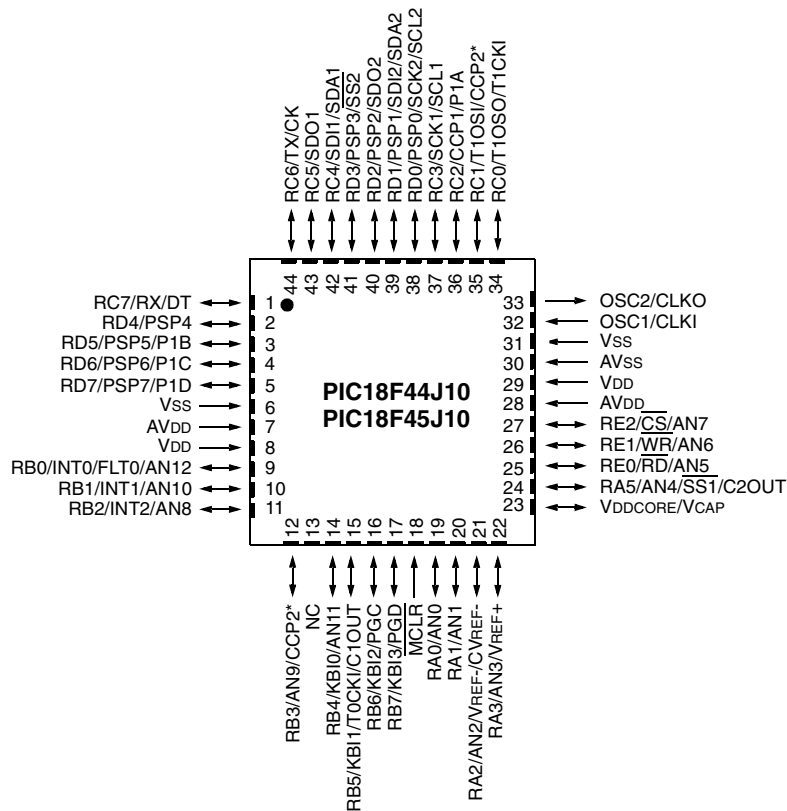
Pin Diagrams (Continued)

40-Pin PDIP (600 MIL)



* Pin feature is dependent on device configuration.

44-Pin QFN

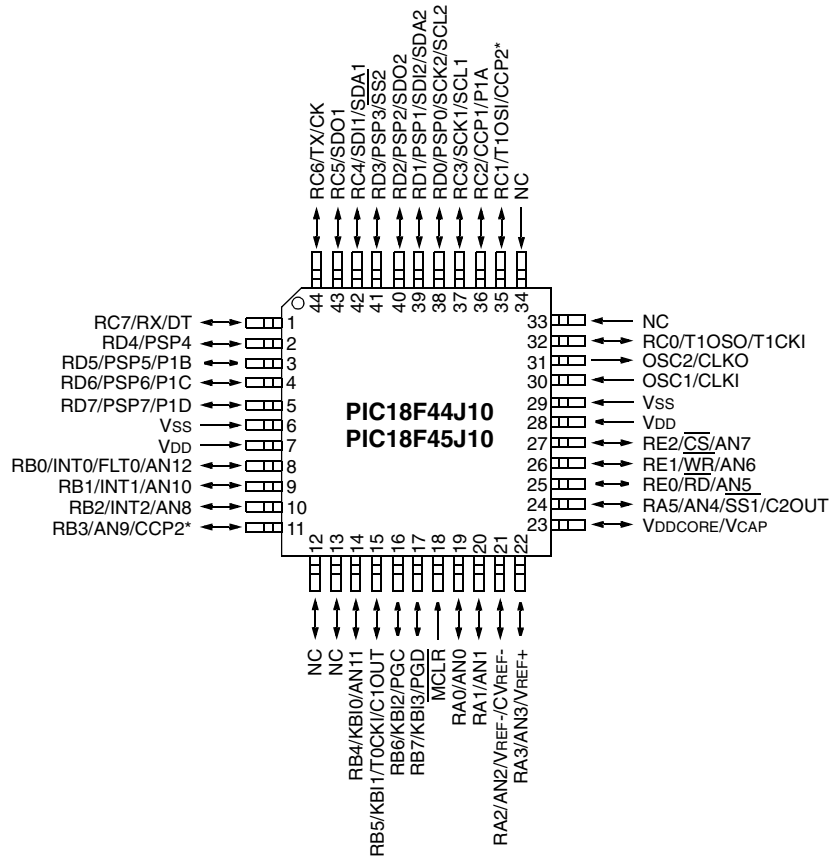


* Pin feature is dependent on device configuration.

PIC18F45J10 FAMILY

Pin Diagrams (Continued)

44-Pin TQFP



* Pin feature is dependent on device configuration.

PIC18F45J10 FAMILY

Table of Contents

| | | |
|------|---|-----|
| 1.0 | Device Overview | 7 |
| 2.0 | Oscillator Configurations | 23 |
| 3.0 | Power-Managed Modes | 31 |
| 4.0 | Reset | 37 |
| 5.0 | Memory Organization | 47 |
| 6.0 | Flash Program Memory | 67 |
| 7.0 | 8 x 8 Hardware Multiplier | 77 |
| 8.0 | Interrupts | 79 |
| 9.0 | I/O Ports | 93 |
| 10.0 | Timer0 Module | 111 |
| 11.0 | Timer1 Module | 115 |
| 12.0 | Timer2 Module | 121 |
| 13.0 | Capture/Compare/PWM (CCP) Modules | 123 |
| 14.0 | Enhanced Capture/Compare/PWM (ECCP) Module | 131 |
| 15.0 | Master Synchronous Serial Port (MSSP) Module | 145 |
| 16.0 | Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) | 187 |
| 17.0 | 10-Bit Analog-to-Digital Converter (A/D) Module | 209 |
| 18.0 | Comparator Module | 219 |
| 19.0 | Comparator Voltage Reference Module | 225 |
| 20.0 | Special Features of the CPU | 229 |
| 21.0 | Instruction Set Summary | 241 |
| 22.0 | Development Support | 291 |
| 23.0 | Electrical Characteristics | 297 |
| 24.0 | DC and AC Characteristics Graphs and Tables | 331 |
| 25.0 | Packaging Information | 333 |
| | Appendix A: Revision History | 343 |
| | Appendix B: Migration Between High-End Device Families | 343 |
| | Index | 345 |
| | The Microchip Web Site | 355 |
| | Customer Change Notification Service | 355 |
| | Customer Support | 355 |
| | Reader Response | 356 |
| | PIC18F45J10 Family Product Identification System | 357 |

PIC18F45J10 FAMILY

TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at docerrors@microchip.com 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)

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

Customer Notification System

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

1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC18F24J10
- PIC18F25J10
- PIC18F44J10
- PIC18F45J10
- PIC18LF24J10
- PIC18LF25J10
- PIC18LF44J10
- PIC18LF45J10

This family offers the advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price. The PIC18F45J10 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 PIC18F45J10 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 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.
- **Low Consumption in Key Modules:** The power requirements for both Timer1 and the Watchdog Timer are minimized. See **Section 23.0 “Electrical Characteristics”** for values.

1.1.2 MULTIPLE OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F45J10 family offer three different oscillator options. These include:

- One Crystal mode, using crystals or ceramic resonators
- One External Clock mode
- INTRC source (approximately 31 kHz)

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.

PIC18F45J10 FAMILY

1.2 Other Special Features

- **Communications:** The PIC18F45J10 family incorporates a range of serial communication peripherals, including 1 independent Enhanced USART and 2 Master SSP modules capable of both SPI and I²C (Master and Slave) modes of operation. Also, one of the general purpose I/O ports can be reconfigured as an 8-bit Parallel Slave Port for direct processor-to-processor communications.
- **Self-programmability:** These devices can write to their own program memory spaces under internal software control. By using a bootloader routine, it becomes possible to create an application that can update itself in the field.
- **Extended Instruction Set:** The PIC18F45J10 family introduces an optional extension to the PIC18 instruction set, which adds 8 new instructions and an Indexed Addressing mode. This extension, enabled as a device configuration option, has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as C.
- **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.
- **Enhanced Addressable USART:** This serial communication module is capable of standard RS-232 operation and provides support for the LIN bus protocol. Other enhancements include automatic baud rate detection and a 16-bit Baud Rate Generator for improved resolution.
- **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 an extended time-out range that is stable across operating voltage and temperature. See **Section 23.0 “Electrical Characteristics”** for time-out periods.

1.3 Details on Individual Family Members

Devices in the PIC18F45J10 family are available in 28-pin and 40/44-pin 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 (16 Kbytes for PIC18F24J10/44J10 devices and 32 Kbytes for PIC18F25J10/45J10).
2. A/D channels (10 for 28-pin devices, 13 for 40/44-pin devices).
3. I/O ports (3 bidirectional ports on 28-pin devices, 5 bidirectional ports on 40/44-pin devices).
4. CCP and Enhanced CCP implementation (28-pin devices have 2 standard CCP modules, 40/44-pin devices have one standard CCP module and one ECCP module).
5. Parallel Slave Port (present only on 40/44-pin devices).
6. One MSSP module for PIC18F24J10/25J10 devices and 2 MSSP modules for PIC18F44J10/45J10 devices
7. Parts designated with an “F” part number (i.e., PIC18F25J10) have a minimum V_{DD} of 2.8 volts, whereas parts designated with an “LF” part number (i.e., PIC18LF25J10) can operate between 2.0-3.6 volts on V_{DD}; however, V_{DDCORE} should never exceed V_{DD}.

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.

The PIC18F45J10 family of devices provides an on-chip voltage regulator to supply the correct voltage levels to the core. Parts designated with an “F” part number (such as PIC18F25J10) have the voltage regulator enabled. These parts can run from 2.7-3.6 volts on V_{DD} but should have the V_{DDCORE} pin connected to V_{SS} through a low-ESR capacitor. Parts designated with an “LF” part number (such as PIC18LF24J10) do not enable the voltage regulator. An external supply of 2.0-2.7 Volts has to be supplied to the V_{DDCORE} pin while 2.0-3.6 Volts can be supplied to V_{DD} (V_{DDCORE} should never exceed V_{DD}). See **Section 20.3 “On-Chip Voltage Regulator”** for more details about the internal voltage regulator.

PIC18F45J10 FAMILY

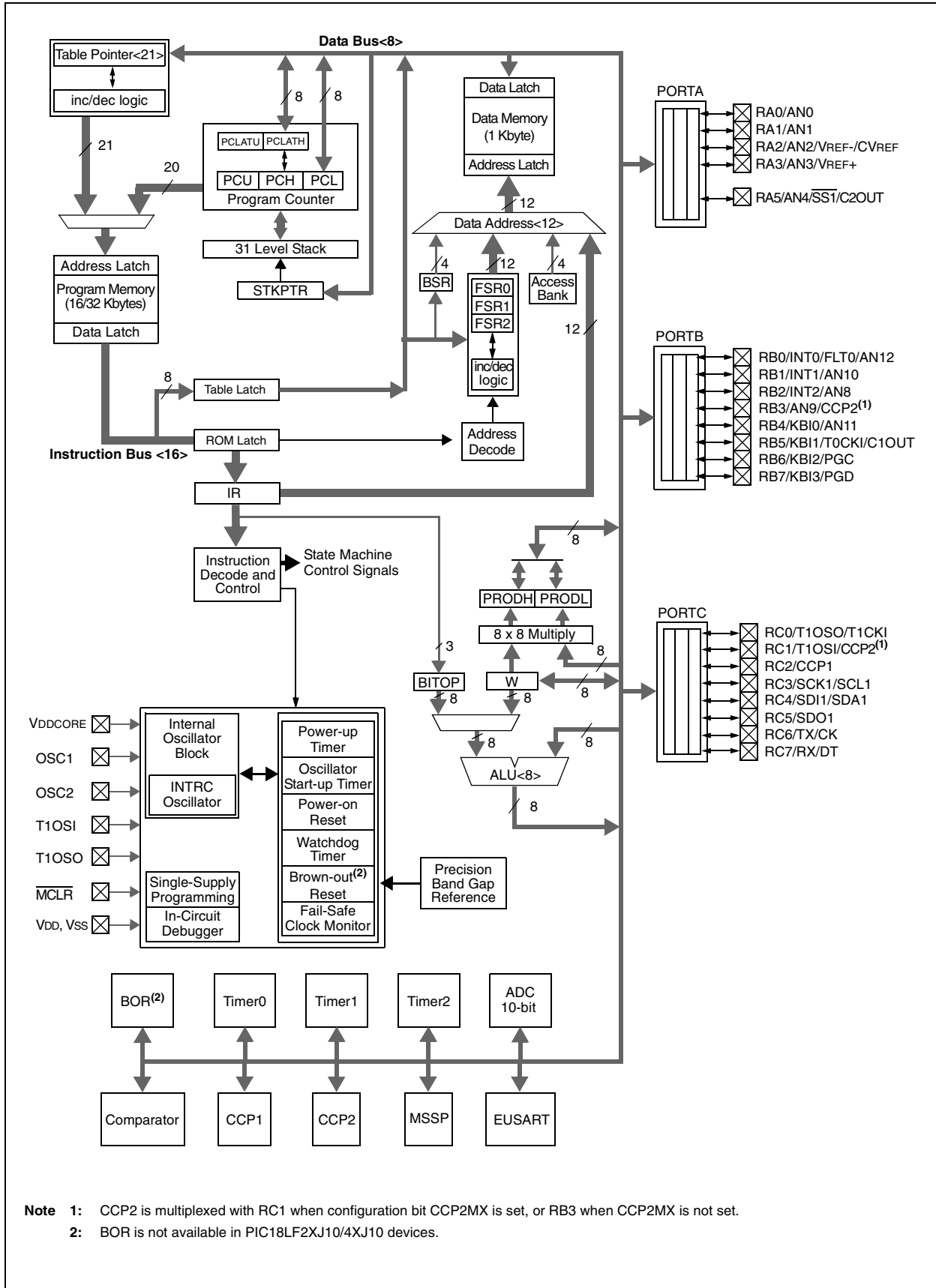
TABLE 1-1: DEVICE FEATURES

| Features | PIC18F24J10 | PIC18F25J10 | PIC18F44J10 | PIC18F45J10 |
|--------------------------------------|---|---|---|---|
| Operating Frequency | DC – 40 MHz | DC – 40 MHz | DC – 40 MHz | DC – 40 MHz |
| Program Memory (Bytes) | 16384 | 32768 | 16384 | 32768 |
| Program Memory (Instructions) | 8192 | 16384 | 8192 | 16384 |
| Data Memory (Bytes) | 768 | 1536 | 768 | 1536 |
| Interrupt Sources | 19 | 19 | 20 | 20 |
| I/O Ports | Ports A, B, C | Ports A, B, C | Ports A, B, C, D, E | Ports A, B, C, D, E |
| Timers | 3 | 3 | 3 | 3 |
| Capture/Compare/PWM Modules | 2 | 2 | 1 | 1 |
| Enhanced Capture/Compare/PWM Modules | 0 | 0 | 1 | 1 |
| Serial Communications | MSSP, Enhanced USART | MSSP, Enhanced USART | MSSP, Enhanced USART | MSSP, Enhanced 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 ⁽¹⁾ , RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR, WDT | POR, BOR ⁽¹⁾ , RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR, WDT | POR, BOR ⁽¹⁾ , RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR, WDT | POR, BOR ⁽¹⁾ , RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR, WDT |
| Programmable Brown-out Reset | Yes | Yes | Yes | Yes |
| Instruction Set | 75 Instructions; 83 with Extended Instruction Set enabled | 75 Instructions; 83 with Extended Instruction Set enabled | 75 Instructions; 83 with Extended Instruction Set enabled | 75 Instructions; 83 with Extended Instruction Set enabled |
| Packages | 28-pin SPDIP 28-pin SOIC 28-pin SSOP 28-pin QFN | 28-pin SPDIP 28-pin SOIC 28-pin SSOP 28-pin QFN | 40-pin PDIP 44-pin QFN 44-pin TQFP | 40-pin PDIP 44-pin QFN 44-pin TQFP |

Note 1: BOR is not available in PIC18LF2XJ10/4XJ10 devices.

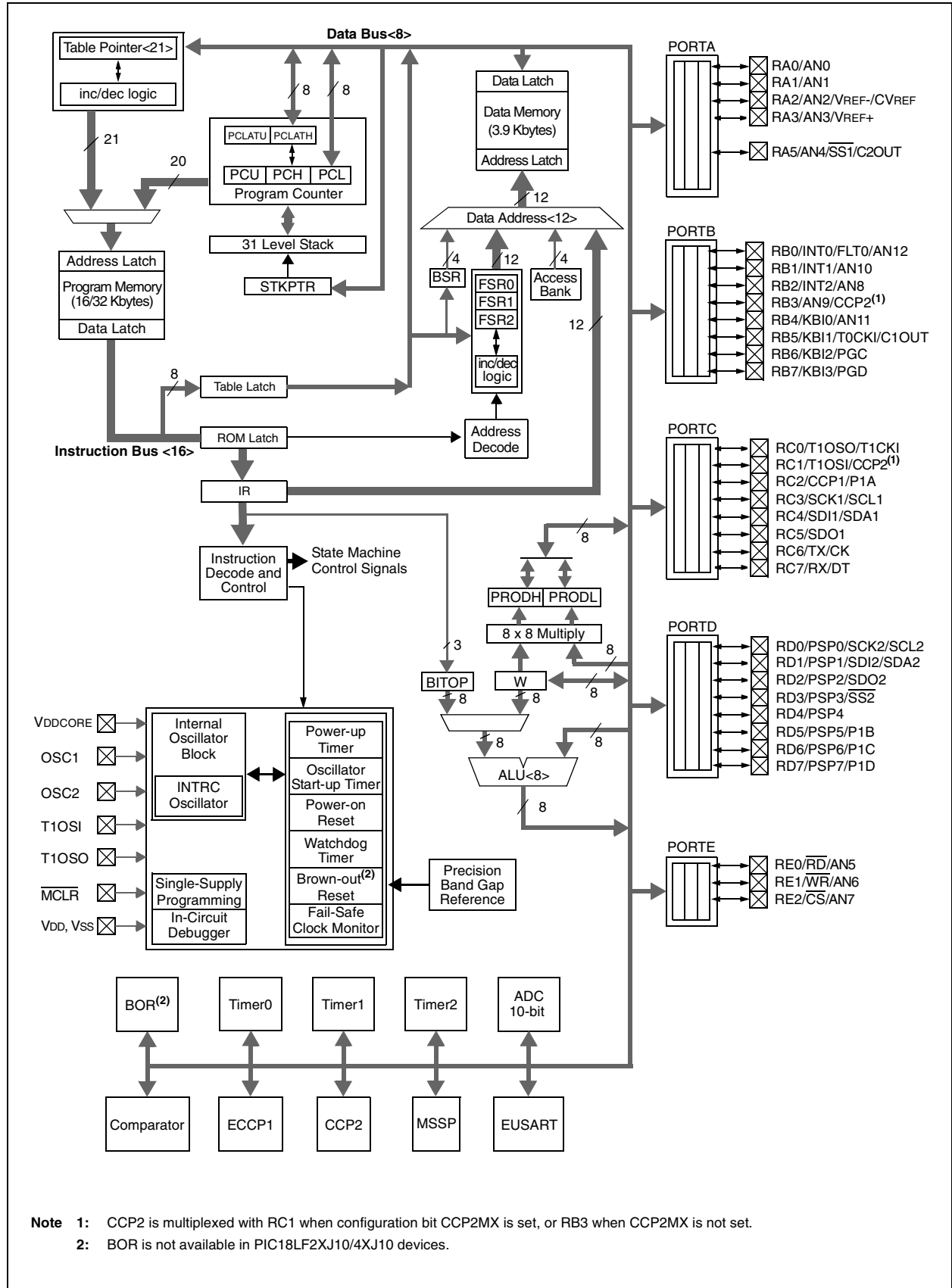
PIC18F45J10 FAMILY

FIGURE 1-1: PIC18F24J10/25J10 (28-PIN) BLOCK DIAGRAM



PIC18F45J10 FAMILY

FIGURE 1-2: PIC18F44J10/45J10 (40/44-PIN) BLOCK DIAGRAM



PIC18F45J10 FAMILY

TABLE 1-2: PIC18F24J10/25J10 PINOUT I/O DESCRIPTIONS

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|----------------------------------|-------------------------|-----|------------|---------------|---|
| | SPDIP, SOIC, SSOP | QFN | | | |
| $\overline{\text{MCLR}}$ MCLR | 1 | 26 | I | ST | Master Clear (input) or programming voltage (input). Master Clear (Reset) input. This pin is an active-low Reset to the device. |
| OSC1/CLKI OSC1 CLKI | 9 | 6 | I I | — CMOS | Oscillator crystal or external clock input. Oscillator crystal input or external clock source input. External clock source input. Always associated with pin function OSC1. See related OSC2/CLKO pins. |
| OSC2/CLKO OSC2 CLKO | 10 | 7 | O O | — — | Oscillator crystal or clock output. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. In EC mode, OSC2 pin outputs CLKO which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. |

Legend: TTL = TTL compatible input CMOS = CMOS compatible input or output
 ST = Schmitt Trigger input with CMOS levels I = Input
 O = Output P = Power

Note 1: Default assignment for CCP2 when configuration bit CCP2MX is set.
Note 2: Alternate assignment for CCP2 when configuration bit CCP2MX is cleared.

PIC18F45J10 FAMILY

TABLE 1-2: PIC18F24J10/25J10 PINOUT I/O DESCRIPTIONS (CONTINUED)

| Pin Name | Pin Number | | Pin Type | Buffer Type | Description |
|---|-------------------------|-----|--------------------|-----------------------------------|---|
| | SPDIP, SOIC, SSOP | QFN | | | |
| RA0/AN0 RA0 AN0 | 2 | 27 | I/O I | TTL Analog | PORTA is a bidirectional I/O port. Digital I/O. Analog input 0. |
| RA1/AN1 RA1 AN1 | 3 | 28 | I/O I | TTL Analog | Digital I/O. Analog input 1. |
| RA2/AN2/VREF-/CVREF RA2 AN2 VREF- CVREF | 4 | 1 | I/O I I O | TTL Analog Analog Analog | Digital I/O. Analog input 2. A/D reference voltage (low) input. Comparator reference voltage output. |
| RA3/AN3/VREF+ RA3 AN3 VREF+ | 5 | 2 | I/O I I | TTL Analog Analog | Digital I/O. Analog input 3. A/D reference voltage (high) input. |
| RA5/AN4/ $\overline{SS1}$ /C2OUT RA5 AN4 $\overline{SS1}$ C2OUT | 7 | 4 | I/O I I O | TTL Analog TTL — | Digital I/O. Analog input 4. SPI™ slave select input. Comparator 2 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

Note 1: Default assignment for CCP2 when configuration bit CCP2MX is set.
Note 2: Alternate assignment for CCP2 when configuration bit CCP2MX is cleared.

PIC18F45J10 FAMILY

TABLE 1-3: PIC18F44J10/45J10 PINOUT I/O DESCRIPTIONS (CONTINUED)

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|------------------------------|------------|-----|------|----------|-------------|---|
| | PDIP | QFN | TQFP | | | |
| RC0/T1OSO/T1CKI | 15 | 34 | 32 | I/O | ST | PORTC is a bidirectional I/O port. Digital I/O. Timer1 oscillator output. Timer1 external clock input. |
| RC0 | | | | O | — | |
| T1OSO | | | | I | ST | |
| T1CKI | | | | | | |
| RC1/T1OSI/CCP2 | 16 | 35 | 35 | I/O | ST | Digital I/O. Timer1 oscillator input. Capture 2 input/Compare 2 output/PWM 2 output. |
| RC1 | | | | I | CMOS | |
| T1OSI CCP2 ⁽²⁾ | | | | I/O | ST | |
| RC2/CCP1/P1A | 17 | 36 | 36 | I/O | ST | Digital I/O. Capture 1 input/Compare 1 output/PWM 1 output. Enhanced CCP1 output. |
| RC2 | | | | I/O | ST | |
| CCP1 P1A | | | | O | — | |
| RC3/SCK1/SCL1 | 18 | 37 | 37 | I/O | ST | Digital I/O. Synchronous serial clock input/output for SPI™ mode. Synchronous serial clock input/output for I ² C™ mode. |
| RC3 | | | | I/O | ST | |
| SCK1 | | | | | | |
| SCL1 | | | | I/O | ST | |
| RC4/SDI1/SDA1 | 23 | 42 | 42 | I/O | ST | Digital I/O. SPI data in. I ² C data I/O. |
| RC4 | | | | I | ST | |
| SDI1 SDA1 | | | | I/O | ST | |
| RC5/SDO1 | 24 | 43 | 43 | I/O | ST | Digital I/O. SPI data out. |
| RC5 | | | | O | — | |
| RC6/TX/CK | 25 | 44 | 44 | I/O | ST | Digital I/O. EUSART asynchronous transmit. EUSART synchronous clock (see related RX/DT). |
| RC6 | | | | O | — | |
| TX CK | | | | I/O | ST | |
| RC7/RX/DT | 26 | 1 | 1 | I/O | ST | Digital I/O. EUSART asynchronous receive. EUSART synchronous data (see related TX/CK). |
| RC7 | | | | I | ST | |
| RX | | | | | | |
| DT | | | | I/O | ST | |

Legend: TTL = TTL compatible input

ST = Schmitt Trigger input with CMOS levels

O = Output

CMOS = CMOS compatible input or output

I = Input

P = Power

Note 1: Default assignment for CCP2 when configuration bit CCP2MX is set.

2: Alternate assignment for CCP2 when configuration bit CCP2MX is cleared.

PIC18F45J10 FAMILY

TABLE 1-3: PIC18F44J10/45J10 PINOUT I/O DESCRIPTIONS (CONTINUED)

| Pin Name | Pin Number | | | Pin Type | Buffer Type | Description |
|------------------------|------------|-----|------|----------|-------------|--|
| | PDIP | QFN | TQFP | | | |
| RD0/PSP0/SCK2/ SCL2 | 19 | 38 | 38 | | | 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 | | | | I/O | ST | Digital I/O. |
| PSP0 | | | | I/O | TTL | Parallel Slave Port data. |
| SCK2 | | | | I/O | ST | Synchronous serial clock input/output for SPI™ mode. |
| SCL2 | | | | I/O | ST | Synchronous serial clock input/output for I ² C™ mode. |
| RD1/PSP1/SDI2/SDA2 | 20 | 39 | 39 | | | |
| RD1 | | | | I/O | ST | Digital I/O. |
| PSP1 | | | | I/O | TTL | Parallel Slave Port data. |
| SDI2 | | | | I | ST | SPI data in. |
| SDA2 | | | | I/O | ST | I ² C data I/O. |
| RD2/PSP2/SDO2 | 21 | 40 | 40 | | | |
| RD2 | | | | I/O | ST | Digital I/O. |
| PSP2 | | | | I/O | TTL | Parallel Slave Port data. |
| SDO2 | | | | O | — | SPI data out. |
| RD3/PSP3/SS2 | 22 | 41 | 41 | | | |
| RD3 | | | | I/O | ST | Digital I/O. |
| PSP3 | | | | I/O | TTL | Parallel Slave Port data. |
| SS2 | | | | I | TTL | SPI slave select input. |
| 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

ST = Schmitt Trigger input with CMOS levels

O = Output

CMOS = CMOS compatible input or output

I = Input

P = Power

Note 1: Default assignment for CCP2 when configuration bit CCP2MX is set.

Note 2: Alternate assignment for CCP2 when configuration bit CCP2MX is cleared.

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

2.0 OSCILLATOR CONFIGURATIONS

2.1 Oscillator Types

The PIC18F45J10 family of devices can be operated in five different oscillator modes:

1. HS High-Speed Crystal/Resonator
2. HSPLL High-Speed Crystal/Resonator with Software PLL Control
3. EC External Clock with FOSC/4 Output
4. ECPLL External Clock with Software PLL Control
5. INTRC Internal 31 kHz Oscillator

Four of these are selected by the user by programming the FOSC2:FOSC0 configuration bits. The fifth mode (INTRC) may be invoked under software control; it can also be configured as the default mode on device Resets.

2.2 Crystal Oscillator/Ceramic Resonators (HS Modes)

In 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 manufacturer's specifications.

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

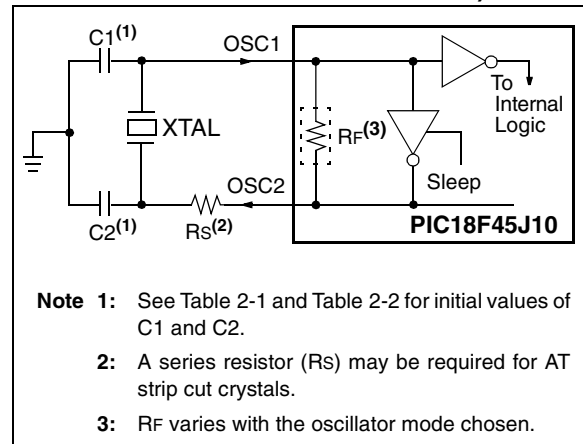


TABLE 2-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

| Typical Capacitor Values Used: | | | |
|--------------------------------|----------|-------|-------|
| Mode | Freq. | OSC1 | OSC2 |
| HS | 8.0 MHz | 27 pF | 27 pF |
| | 16.0 MHz | 22 pF | 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 following Table 2-2 for additional information.

| Resonators Used: |
|------------------|
| 4.0 MHz |
| 8.0 MHz |
| 16.0 MHz |

PIC18F45J10 FAMILY

TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

| Osc Type | Crystal Freq. | Typical Capacitor Values Tested: | |
|----------|---------------|----------------------------------|-------|
| | | C1 | C2 |
| 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: |
|----------------|
| 4 MHz |
| 8 MHz |
| 20 MHz |

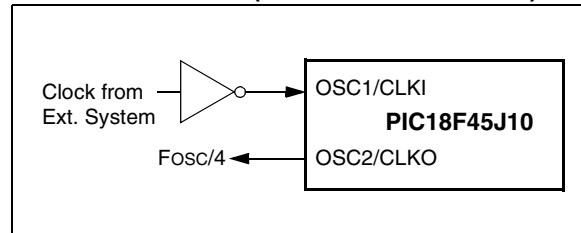
- Note 1:** Higher capacitance increases the stability of oscillator but also increases the start-up time.
- 2:** Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.
- 3:** Rs may be required to avoid overdriving crystals with low drive level specification.
- 4:** Always verify oscillator performance over the VDD and temperature range that is expected for the application.

2.3 External Clock Input (EC Modes)

The EC and ECPLL 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.

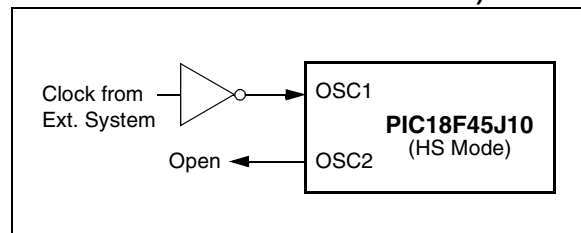
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-2 shows the pin connections for the EC Oscillator mode.

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



An external clock source may also be connected to the OSC1 pin in the HS mode, as shown in Figure 2-3. In this configuration, the divide-by-4 output on OSC2 is not available.

FIGURE 2-3: EXTERNAL CLOCK INPUT OPERATION (HS OSC CONFIGURATION)



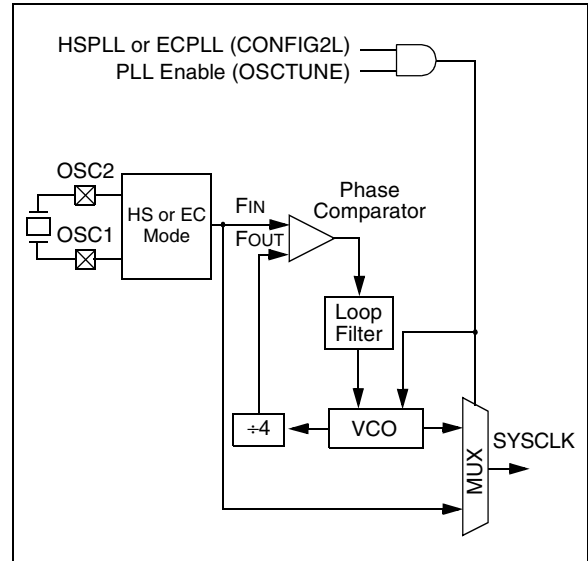
PIC18F45J10 FAMILY

2.4 PLL Frequency Multiplier

A Phase Locked Loop (PLL) circuit is provided as an option for users who want to use a lower frequency 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, or users who require higher clock speeds from an internal oscillator. For these reasons, the HSPLL and ECPLL modes are available.

The HSPLL and ECPLL modes provide the ability to selectively run the device at 4 times the external oscillating source to produce frequencies up to 40 MHz. The PLL is enabled by setting the PLEN bit in the OSCTUNE register (Register 2-1).

FIGURE 2-4: PLL BLOCK DIAGRAM



REGISTER 2-1: OSCTUNE: PLL CONTROL REGISTER

| | | | | | | | |
|-------|----------------------|-----|-----|-----|-----|-----|-------|
| U-0 | R/W-0 ⁽¹⁾ | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
| — | PLEN ⁽¹⁾ | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 6 **PLEN:** Frequency Multiplier PLL Enable bit⁽¹⁾

1 = PLL enabled

0 = PLL disabled

Note 1: Available only for ECPLL and HSPLL oscillator configurations; otherwise, this bit is unavailable and read as '0'.

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

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

PIC18F45J10 FAMILY

2.5 Internal Oscillator Block

The PIC18F45J10 family of devices includes an internal oscillator source (INTRC) which provides a nominal 31 kHz output. The INTRC is enabled on device power-up and clocks the device during its configuration cycle until it enters operating mode. INTRC is also enabled if it is selected as the device clock source or if any of the following are enabled:

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

These features are discussed in greater detail in **Section 20.0 “Special Features of the CPU”**.

The INTRC can also be optionally configured as the default clock source on device start-up by setting the FOSC2 configuration bit. This is discussed in **Section 2.6.1 “Oscillator Control Register”**.

2.6 Clock Sources and Oscillator Switching

The PIC18F45J10 family includes a feature that allows the device clock source to be switched from the main oscillator to an alternate clock source. PIC18F45J10 family devices offer two alternate clock sources. When an alternate clock source is enabled, the various power-managed operating modes are available.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator

The **primary oscillators** include the External Crystal and Resonator modes and the External Clock modes. The particular mode is defined by the FOSC2:FOSC0 configuration bits. 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.

PIC18F45J10 family devices offer 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.

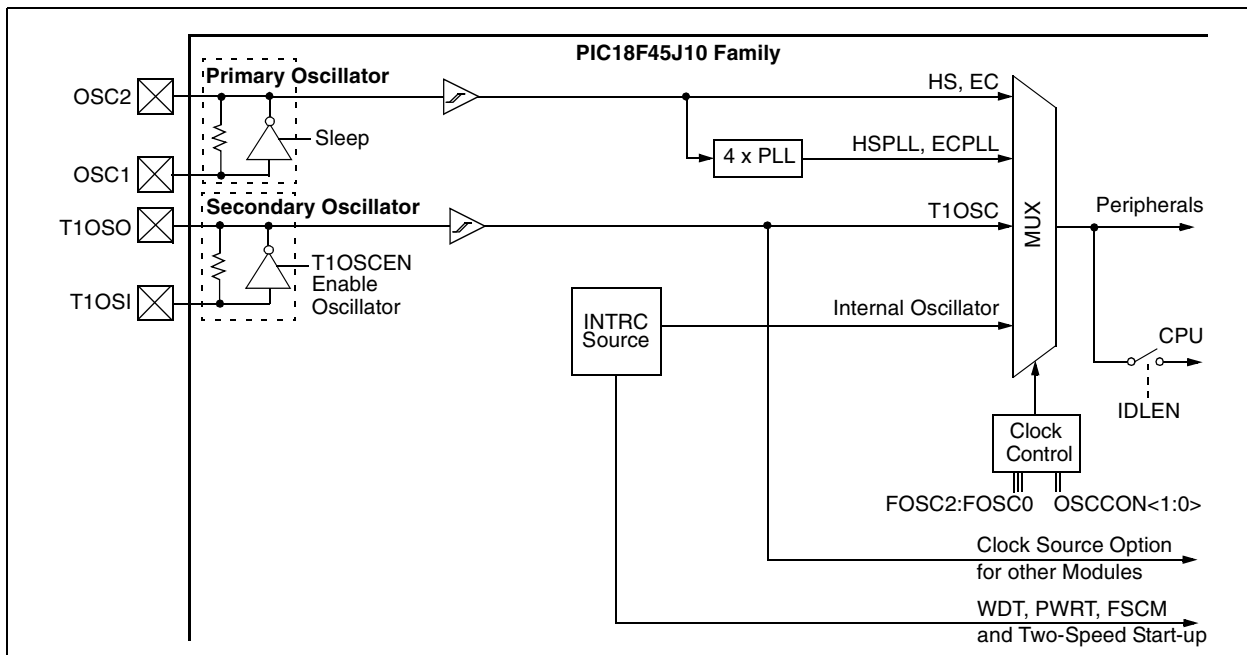
Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T13CKI and RC1/T1OSI pins. Loading capacitors are also connected from each pin to ground.

The Timer1 oscillator is discussed in greater detail in **Section 11.3 “Timer1 Oscillator”**.

In addition to being a primary clock source, the **internal oscillator** 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 PIC18F45J10 family devices are shown in Figure 2-5. See **Section 20.0 “Special Features of the CPU”** for Configuration register details.

FIGURE 2-5: PIC18F45J10 FAMILY CLOCK DIAGRAM



2.6.1 OSCILLATOR CONTROL REGISTER

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

The System Clock Select bits, SCS1:SCS0, select the clock source. The available clock sources are the primary clock (defined by the FOSC2:FOSC0 configuration bits), the secondary clock (Timer1 oscillator) and the internal oscillator. The clock source changes after one or more of the bits are written to, following a brief clock transition interval.

The OSTS (OSCCON<3>) and T1RUN (T1CON<6>) bits indicate which clock source is currently providing the device clock. The OSTS bit indicates that the Oscillator Start-up Timer (OST) has timed out and the primary clock is providing the device clock in primary clock modes. The T1RUN bit indicates when the Timer1 oscillator is providing the device clock in secondary clock modes. In power-managed modes, only one of these bits will be set at any time. If neither of these bits are set, the INTRC is providing the clock, or the internal oscillator has just started and is not yet stable.

The IDLEN bit determines if the device goes into Sleep mode or one of the Idle modes when the SLEEP instruction is executed.

The use of the flag and control bits in the OSCCON register 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 select a secondary clock source when executing a SLEEP instruction 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.

2.6.1.1 System Clock Selection and the FOSC2 Configuration Bit

The SCS bits are cleared on all forms of Reset. In the device's default configuration, this means the primary oscillator defined by FOSC1:FOSC0 (that is, one of the HC or EC modes) is used as the primary clock source on device Resets.

The default clock configuration on Reset can be changed with the FOSC2 configuration bit. The effect of this bit is to set the clock source selected when SCS1:SCS0 = 00. When FOSC2 = 1 (default), the oscillator source defined by FOSC1:FOSC0 is selected whenever SCS1:SCS0 = 00. When FOSC2 = 0, the INTRC oscillator is selected whenever SCS1:SCS2 = 00. Because the SCS bits are cleared on Reset, the FOSC2 setting also changes the default oscillator mode on Reset.

Regardless of the setting of FOSC2, INTRC will always be enabled on device power-up. It will serve as the clock source until the device has loaded its configuration values from memory. It is at this point that the FOSC configuration bits are read and the oscillator selection of operational mode is made.

Note that either the primary clock or the internal oscillator will have two bit setting options, at any given time, depending on the setting of FOSC2.

2.6.2 OSCILLATOR TRANSITIONS

PIC18F45J10 family devices contain circuitry to prevent clock "glitches" when switching between clock sources. A short pause in the device clock occurs during the clock switch. The length of this pause is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

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

PIC18F45J10 FAMILY

REGISTER 2-2: OSCCON: OSCILLATOR CONTROL REGISTER

| | | | | | | | |
|-------|-----|-----|-----|--------------------|-----|-------|-------|
| R/W-0 | U-0 | U-0 | U-0 | R-q ⁽¹⁾ | U-0 | R/W-0 | R/W-0 |
| IDLEN | — | — | — | OSTS | — | SCS1 | SCS0 |
| bit 7 | | | | bit 0 | | | |

bit 7 **IDLEN:** Idle Enable bit

- 1 = Device enters Idle mode on SLEEP instruction
- 0 = Device enters Sleep mode on SLEEP instruction

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

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit⁽¹⁾

- 1 = Oscillator Start-up Timer time-out has expired; primary oscillator is running
- 0 = Oscillator Start-up Timer time-out is running; primary oscillator is not ready

Note 1: The Reset value is '0' when HS mode and Two-Speed Start-up are both enabled; otherwise, it is '1'.

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

bit 1-0 **SCS1:SCS0:** System Clock Select bits

- 11 = Internal oscillator
- 10 = Primary oscillator
- 01 = Timer1 oscillator

When FOSC2 = 1:

- 00 = Primary oscillator

When FOSC2 = 0:

- 00 = Internal oscillator

Legend:

U = Unimplemented, read as '0'

'q' = Value determined by configuration

-n = Value at POR

R = Readable bit

'0' = Bit is cleared

W = Writable bit

2.7 Effects of Power-Managed Modes on the Various Clock Sources

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 device clock. The Timer1 oscillator may also run in all power-managed modes if required to clock Timer1 or Timer3.

In RC_RUN and RC_IDLE modes, the internal oscillator provides the device clock source. The 31 kHz INTRC output can be used directly to provide the clock and may be enabled to support various special features, regardless of the power-managed mode (see **Section 20.2 "Watchdog Timer (WDT)"** through **Section 20.5 "Fail-Safe Clock Monitor"** for more information on WDT, Fail-Safe Clock Monitor and Two-Speed Start-up).

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 device clock source (i.e., MSSP slave, PSP, INTn pins and others). Peripherals that may add significant current consumption are listed in **Section 23.2 "DC Characteristics: Power-Down and Supply Current"**.

2.8 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 4.5 “Power-up Timer (PWRT)”**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on power-up (parameter 33, Table 23-10). It is always enabled.

The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable (HS modes). The OST does this by counting 1024 oscillator cycles before allowing the oscillator to clock the device.

There is a delay of interval T_{CSD} (parameter 38, Table 23-10), following POR, while the controller becomes ready to execute instructions.

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

| Oscillator Mode | OSC1 Pin | OSC2 Pin |
|-----------------|---|---|
| EC, ECPLL | Floating, pulled by external clock | At logic low (clock/4 output) |
| HS, HSPLL | Feedback inverter disabled at quiescent voltage level | Feedback inverter disabled at quiescent voltage level |

Note: See Table 4-2 in **Section 4.0 “Reset”** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

PIC18F45J10 FAMILY

NOTES:

3.0 POWER-MANAGED MODES

The PIC18F45J10 family devices provide the ability to manage power consumption by simply managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. For the sake of managing power in an application, there are three primary modes of operation:

- Run mode
- Idle mode
- Sleep mode

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

The power-managed modes include several power-saving features offered on previous PICmicro® devices. One is the clock switching feature, offered in other PIC18 devices, allowing the controller to use the Timer1 oscillator in place of the primary oscillator. Also included is the Sleep mode, offered by all PICmicro devices, where all device clocks are stopped.

3.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires two decisions: if the CPU is to be clocked or not and which clock source is to be used. The IDLEN bit (OSCCON<7>) controls CPU clocking, while the SCS1:SCS0 bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 3-1.

TABLE 3-1: POWER-MANAGED MODES

| Mode | OSCCON bits | | Module Clocking | | Available Clock and Oscillator Source |
|----------|-------------------------|----------------|-----------------|-------------|---|
| | IDLEN<7> ⁽¹⁾ | SCS1:SCS0<1:0> | CPU | Peripherals | |
| Sleep | 0 | N/A | Off | Off | None – All clocks are disabled |
| PRI_RUN | N/A | 10 | Clocked | Clocked | Primary – HS, EC; this is the normal full power execution mode |
| SEC_RUN | N/A | 01 | Clocked | Clocked | Secondary – Timer1 Oscillator |
| RC_RUN | N/A | 11 | Clocked | Clocked | Internal Oscillator |
| PRI_IDLE | 1 | 10 | Off | Clocked | Primary – HS, EC |
| SEC_IDLE | 1 | 01 | Off | Clocked | Secondary – Timer1 Oscillator |
| RC_IDLE | 1 | 11 | Off | Clocked | Internal Oscillator |

Note 1: IDLEN reflects its value when the SLEEP instruction is executed.

3.1.1 CLOCK SOURCES

The SCS1:SCS0 bits allow the selection of one of three clock sources for power-managed modes. They are:

- the primary clock, as defined by the FOSC1:FOSC0 configuration bits
- the secondary clock (Timer1 oscillator)
- the internal oscillator

3.1.2 ENTERING POWER-MANAGED MODES

Switching from one power-managed mode to another begins by loading the OSCCON register. The SCS1:SCS0 bits select the clock source and determine which Run or Idle mode is to be used. Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch may also be subject to clock transition delays. These are discussed in **Section 3.1.3 “Clock Transitions and Status Indicators”** and subsequent sections.

Entry to the power-managed Idle or Sleep modes is triggered by the execution of a SLEEP instruction. The actual mode that results depends on the status of the IDLEN bit.

Depending on the current mode and the mode being switched to, a change to a power-managed mode does not always require setting all of these bits. Many transitions may be done by changing the oscillator select bits, or changing the IDLEN bit, prior to issuing a SLEEP instruction. If the IDLEN bit is already configured correctly, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

PIC18F45J10 FAMILY

3.1.3 CLOCK TRANSITIONS AND STATUS INDICATORS

The length of the transition between clock sources is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

Two bits indicate the current clock source and its status: OSTS (OSCCON<3>) and T1RUN (T1CON<6>). In general, only one of these bits will be set while in a given power-managed mode. When the OSTS bit is set, the primary clock is providing the device clock. When the T1RUN bit is set, the Timer1 oscillator is providing the clock. If neither of these bits is set, INTRC is clocking the device.

Note: Executing a SLEEP instruction does not necessarily place the device into Sleep mode. It acts as the trigger to place the controller into either the Sleep mode or one of the Idle modes, depending on the setting of the IDLEN bit.

3.1.4 MULTIPLE SLEEP COMMANDS

The power-managed mode that is invoked with the SLEEP instruction is determined by the setting of the IDLEN bit at the time the instruction is executed. If another SLEEP instruction is executed, the device will enter the power-managed mode specified by IDLEN at that time. If IDLEN has changed, the device will enter the new power-managed mode specified by the new setting.

3.2 Run Modes

In the Run modes, clocks to both the core and peripherals are active. The difference between these modes is the clock source.

3.2.1 PRI_RUN MODE

The PRI_RUN mode is the normal, full power execution mode of the microcontroller. This is also the default mode upon a device Reset unless Two-Speed Start-up is enabled (see Section 20.4 “Two-Speed Start-up” for details). In this mode, the OSTS bit is set. (see Section 2.6.1 “Oscillator Control Register”).

3.2.2 SEC_RUN MODE

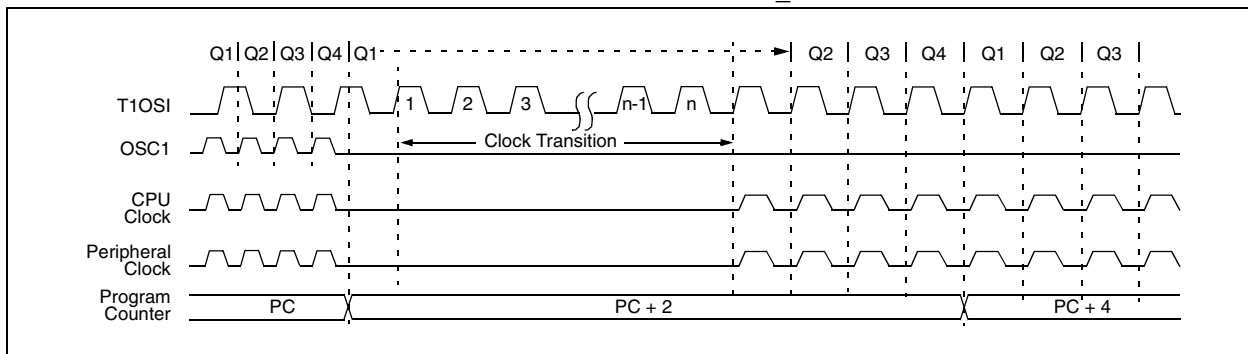
The SEC_RUN mode is the compatible mode to the “clock switching” feature offered in other PIC18 devices. In this mode, the CPU and peripherals are clocked from the Timer1 oscillator. This gives users the option of lower power consumption while still using a high-accuracy clock source.

SEC_RUN mode is entered by setting the SCS1:SCS0 bits to ‘01’. The device clock source is switched to the Timer1 oscillator (see Figure 3-1), 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 the SCS1:SCS0 bits are set to ‘01’, entry to SEC_RUN mode will not occur. If the Timer1 oscillator is enabled, but not yet running, device clocks will be delayed until the oscillator has started. In such situations, initial oscillator operation is far from stable and unpredictable operation may result.

On transitions from SEC_RUN mode to PRI_RUN mode, 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 3-2). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

FIGURE 3-1: TRANSITION TIMING FOR ENTRY TO SEC_RUN MODE



PIC18F45J10 FAMILY

3.2.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator; the primary clock is shut down. 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.

This mode is entered by setting SCS to '11'. When the clock source is switched to the INTRC (see Figure 3-2), the primary oscillator is shut down and the OSTS bit is cleared.

On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTRC while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-3). When the clock switch is complete, the OSTS bit is set and the primary clock is providing the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

FIGURE 3-2: TRANSITION TIMING TO RC_RUN MODE

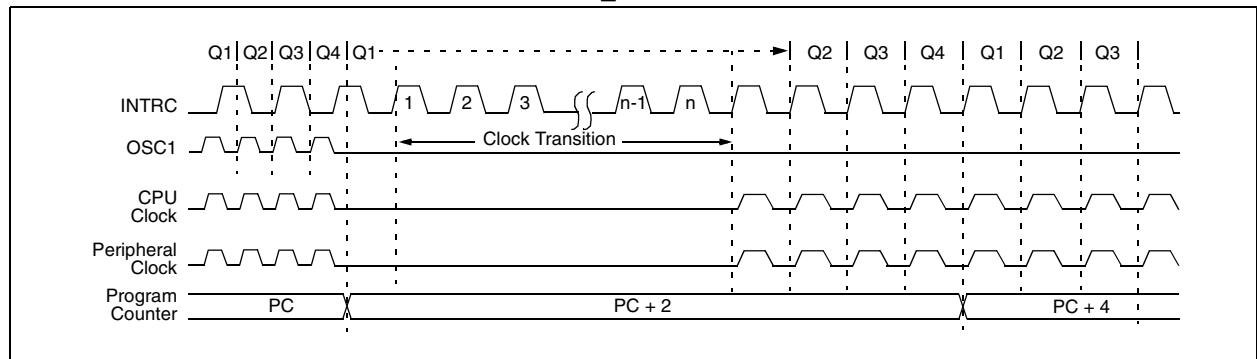
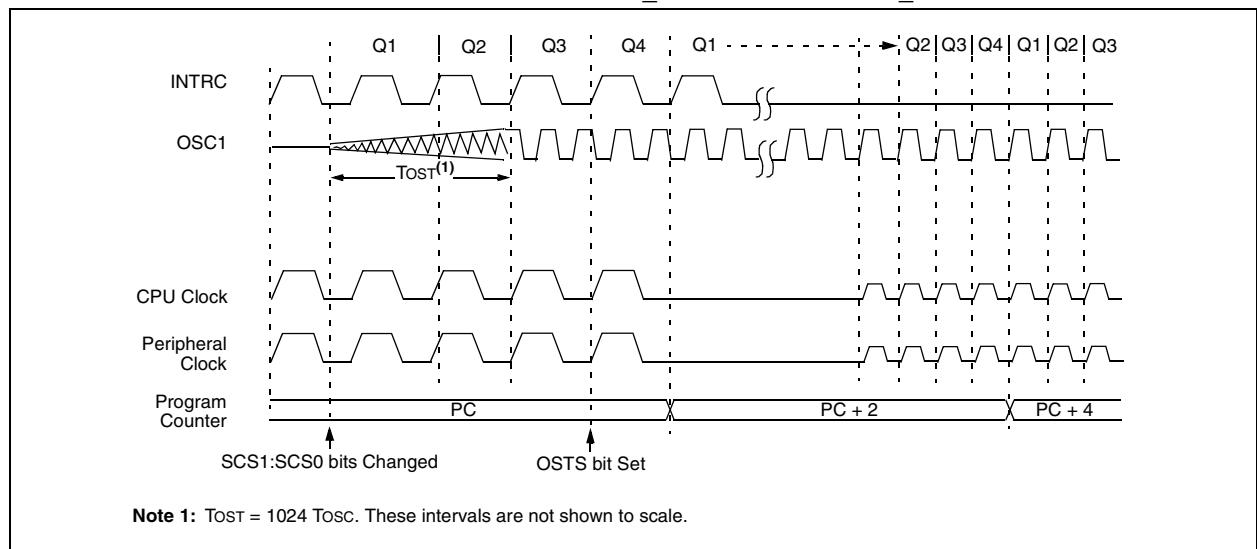


FIGURE 3-3: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



PIC18F45J10 FAMILY

3.3 Sleep Mode

The power-managed Sleep mode is identical to the legacy Sleep mode offered in all other PICmicro devices. It is entered by clearing the IDLEN bit (the default state on device Reset) and executing the SLEEP instruction. This shuts down the selected oscillator (Figure 3-4). All clock source status bits are cleared.

Entering the Sleep mode from any other mode does not require a clock switch. This is because no clocks are needed once the controller has entered Sleep. If the WDT is selected, the INTRC source will continue to operate. If the Timer1 oscillator is enabled, it will also continue to run.

When a wake event occurs in Sleep mode (by interrupt, Reset or WDT time-out), the device will not be clocked until the clock source selected by the SCS1:SCS0 bits becomes ready (see Figure 3-5), or it will be clocked from the internal oscillator if either the Two-Speed Start-up or the Fail-Safe Clock Monitor are enabled (see Section 20.0 “Special Features of the CPU”). In either case, the OSTS bit is set when the primary clock is providing the device clocks. The IDLEN and SCS bits are not affected by the wake-up.

3.4 Idle Modes

The Idle modes allow the controller’s CPU to be selectively shut down while the peripherals continue to operate. Selecting a particular Idle mode allows users to further manage power consumption.

If the IDLEN bit 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. The clock source status bits are not affected. Setting IDLEN and executing a SLEEP instruction provides a quick method of switching from a given Run mode to its corresponding Idle mode.

If the WDT is selected, the INTRC source will continue to operate. If the Timer1 oscillator is enabled, it will also continue to run.

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 event occurs, CPU execution is delayed by an interval of T_{CSD} (parameter 38, Table 23-10) while it becomes ready to execute code. When the CPU begins executing code, it resumes with the same clock source for the current Idle mode. For example, when waking from RC_IDLE mode, the internal oscillator block will clock the CPU and peripherals (in other words, RC_RUN mode). 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 the Run mode currently specified by the SCS1:SCS0 bits.

FIGURE 3-4: TRANSITION TIMING FOR ENTRY TO SLEEP MODE

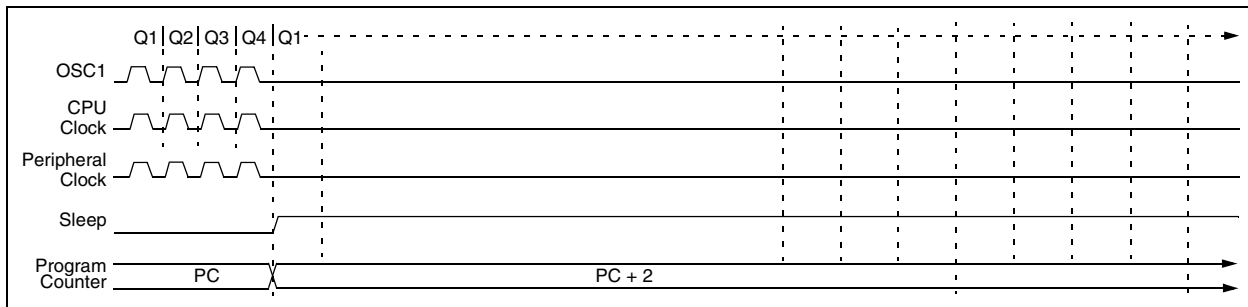
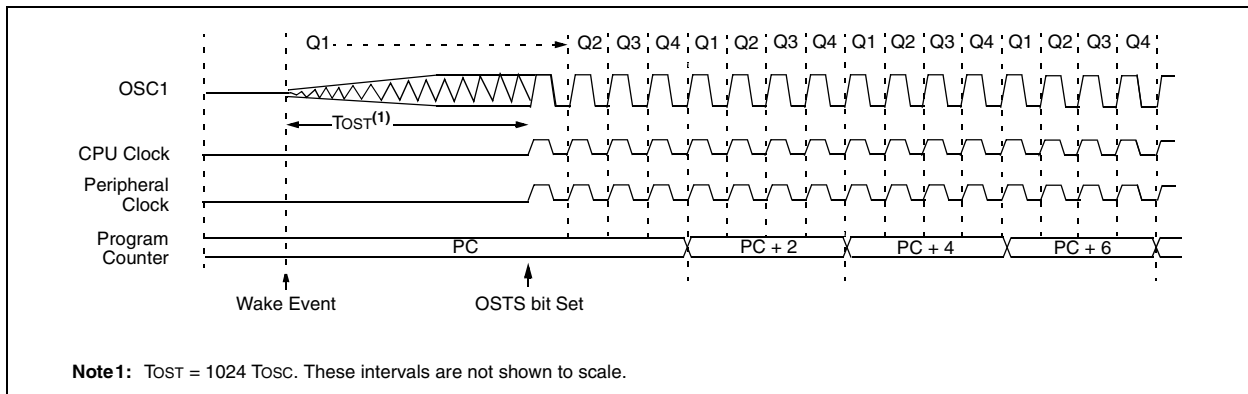


FIGURE 3-5: TRANSITION TIMING FOR WAKE FROM SLEEP



Note 1: TOST = 1024 TOSC. These intervals are not shown to scale.

PIC18F45J10 FAMILY

3.4.1 PRI_IDLE MODE

This mode is unique among the three low-power Idle modes, in that it does not disable the primary device 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 from PRI_RUN mode by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then set the SCS bits to ‘10’ and execute `SLEEP`. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the FOSC0 configuration bit. The OSTS bit remains set (see Figure 3-6).

When a wake event occurs, the CPU is clocked from the primary clock source. A delay of interval TcSD is required between the wake 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 3-7).

3.4.2 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 from SEC_RUN by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then set SCS1:SCS0 to ‘01’ and execute `SLEEP`. When the clock source is switched to the Timer1 oscillator, the primary oscillator is shut-down, the OSTS bit is cleared and the T1RUN bit is set.

When a wake event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After an interval of TcSD following the wake event, the CPU begins executing code being clocked by the Timer1 oscillator. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run (see Figure 3-7).

Note: The Timer1 oscillator should already be running prior to entering SEC_IDLE mode. If the T1OSCEN bit is not set when the `SLEEP` instruction is executed, the `SLEEP` instruction will be ignored and entry to SEC_IDLE mode 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.

FIGURE 3-6: TRANSITION TIMING FOR ENTRY TO IDLE MODE

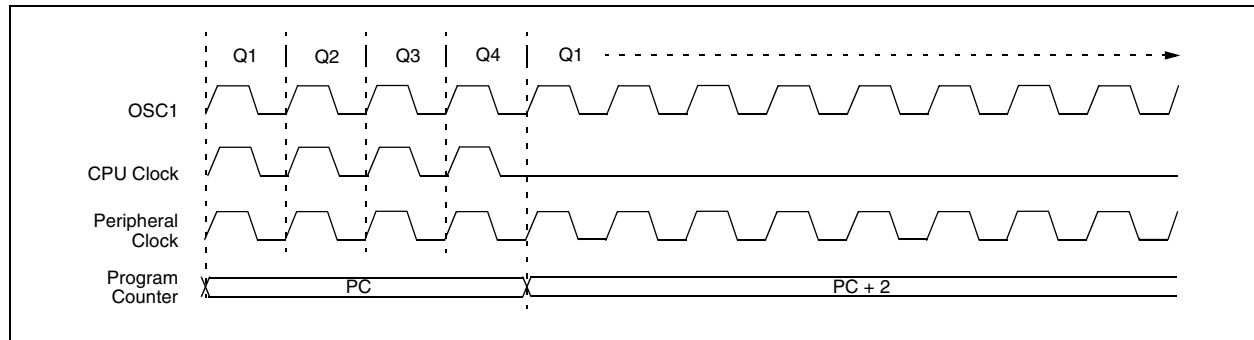
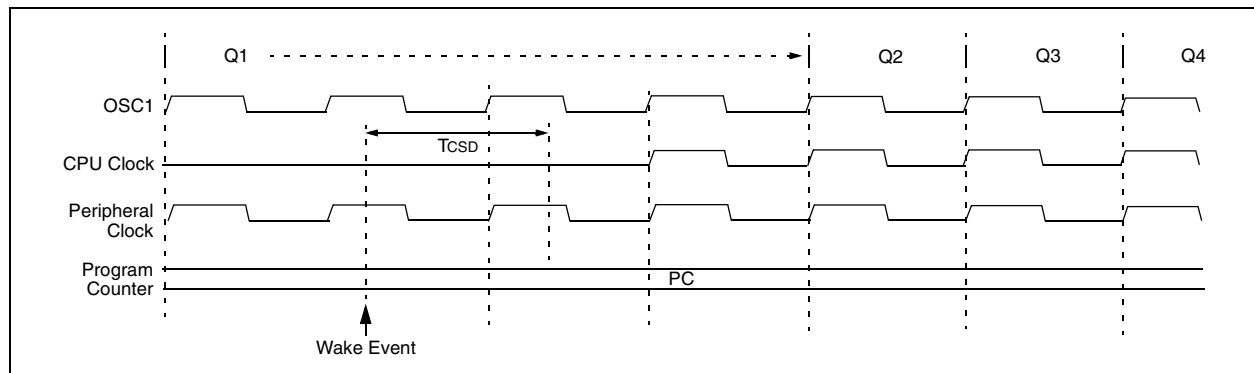


FIGURE 3-7: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE



PIC18F45J10 FAMILY

3.4.3 RC_IDLE MODE

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

From RC_RUN, this mode is entered by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, first set IDLEN, then clear the SCS bits and execute SLEEP. When the clock source is switched to the INTRC, the primary oscillator is shut down and the OSTC bit is cleared.

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

3.5 Exiting Idle and Sleep Modes

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

3.5.1 EXIT BY INTERRUPT

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

On all exits from Idle or Sleep modes by interrupt, code execution branches to the interrupt vector if the GIE/GIEH bit (INTCON<7>) is set. Otherwise, code execution continues or resumes without branching (see **Section 8.0 “Interrupts”**).

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

3.5.2 EXIT BY WDT TIME-OUT

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

If the device is not executing code (all Idle modes and Sleep mode), the time-out will result in an exit from the power-managed mode (see **Section 3.2 “Run Modes”** and **Section 3.3 “Sleep Mode”**). If the device is executing code (all Run modes), the time-out will result in a WDT Reset (see **Section 20.2 “Watchdog Timer (WDT)”**).

The WDT timer and postscaler are cleared by one of the following events:

- executing a SLEEP or CLRWDT instruction
- the loss of a currently selected clock source (if the Fail-Safe Clock Monitor is enabled)

3.5.3 EXIT BY RESET

Exiting an Idle or Sleep mode by Reset automatically forces the device to run from the INTRC.

3.5.4 EXIT WITHOUT AN OSCILLATOR START-UP DELAY

Certain exits from power-managed modes do not invoke the OST at all. There are two cases:

- PRI_IDLE mode where the primary clock source is not stopped; and
- the primary clock source is the EC mode.

In these instances, 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 (EC). However, a fixed delay of interval T_{CSD} following the wake event is still required when leaving Sleep and Idle modes to allow the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

PIC18F45J10 FAMILY

4.0 RESET

The PIC18F45J10 family of devices differentiate between various kinds of Reset:

- Power-on Reset (POR)
- $\overline{\text{MCLR}}$ Reset during normal operation
- $\overline{\text{MCLR}}$ Reset during power-managed modes
- Watchdog Timer (WDT) Reset (during execution)
- Brown-out Reset (BOR)
- RESET Instruction
- Stack Full Reset
- Stack Underflow Reset

This section discusses Resets generated by $\overline{\text{MCLR}}$, POR and BOR and covers the operation of the various start-up timers. Stack Reset events are covered in **Section 5.1.2.4 “Stack Full and Underflow Resets”**. WDT Resets are covered in **Section 20.2 “Watchdog Timer (WDT)”**.

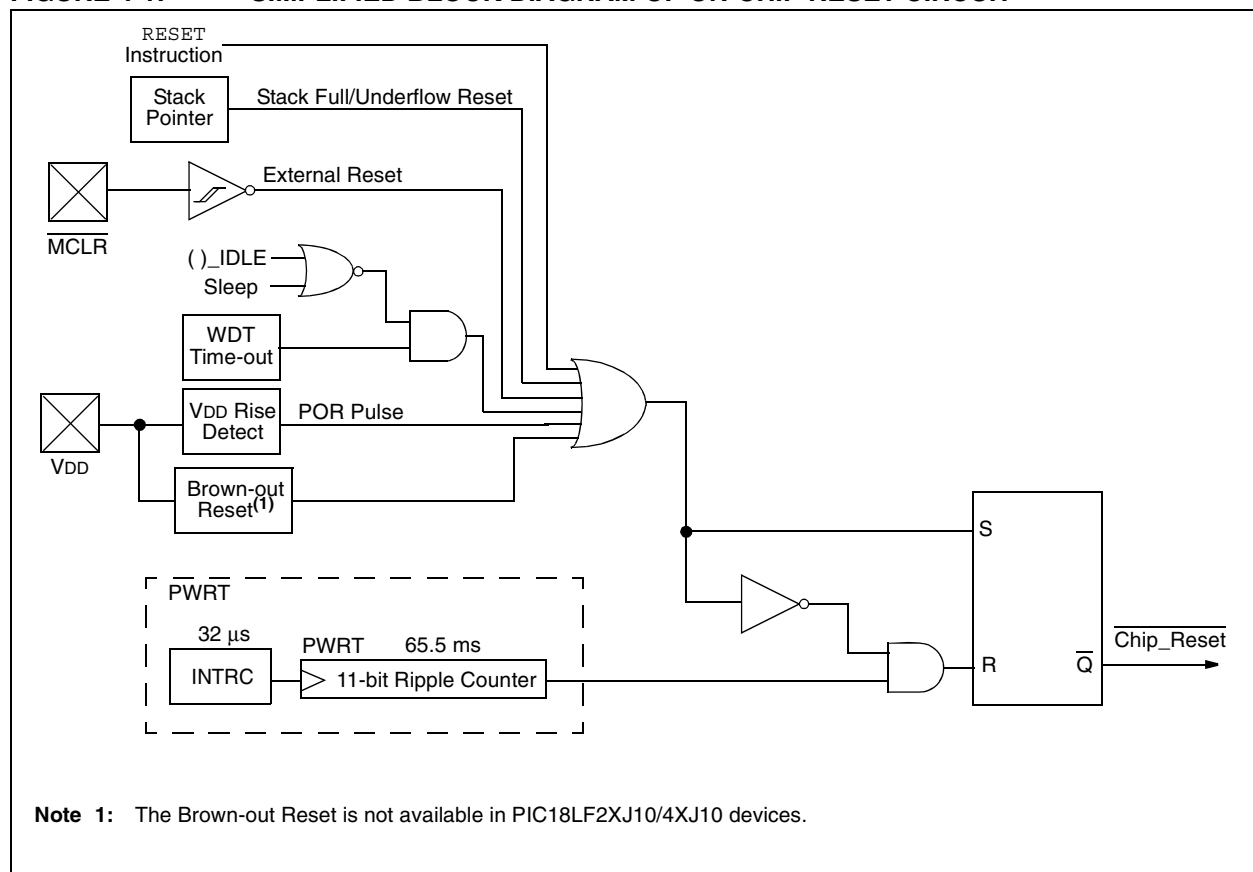
A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 4-1.

4.1 RCON Register

Device Reset events are tracked through the RCON register (Register 4-1). The lower five bits of the register indicate that a specific Reset event has occurred. In most cases, these bits can only be set by the event and must be cleared by the application after the event. The state of these flag bits, taken together, can be read to indicate the type of Reset that just occurred. This is described in more detail in **Section 4.6 “Reset State of Registers”**.

The RCON register also has a control bit for setting interrupt priority (IPEN). Interrupt priority is discussed in **Section 8.0 “Interrupts”**.

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



PIC18F45J10 FAMILY

REGISTER 4-1: RCON: RESET CONTROL 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 = Set 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)
- Note:** BOR is not available in PIC18LF2XJ10/4XJ10 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

- Note 1:** 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.
- 2:** If the on-chip voltage regulator is disabled, \overline{BOR} remains '0' at all times. See **Section 4.4.1 "Detecting BOR"** for more information.
- 3:** Brown-out Reset is said to have occurred when \overline{BOR} is '0' and \overline{POR} is '1' (assuming that \overline{POR} was set to '1' by software immediately after POR).

4.2 Master Clear ($\overline{\text{MCLR}}$)

The $\overline{\text{MCLR}}$ pin provides a method for triggering a hard external Reset of the device. A Reset is generated by holding the pin low. PIC18 extended microcontroller devices have a noise filter in the $\overline{\text{MCLR}}$ Reset path which detects and ignores small pulses.

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

4.3 Power-on Reset (POR)

A Power-on Reset condition is generated on-chip whenever V_{DD} rises above a certain threshold. This allows the device to start in the initialized state when V_{DD} is adequate for operation.

To take advantage of the POR circuitry, tie the $\overline{\text{MCLR}}$ pin through a resistor (1 k Ω to 10 k Ω) to V_{DD} . This will eliminate external RC components usually needed to create a Power-on Reset delay. A minimum rise rate for V_{DD} is specified (parameter D004). For a slow rise time, see Figure 4-2.

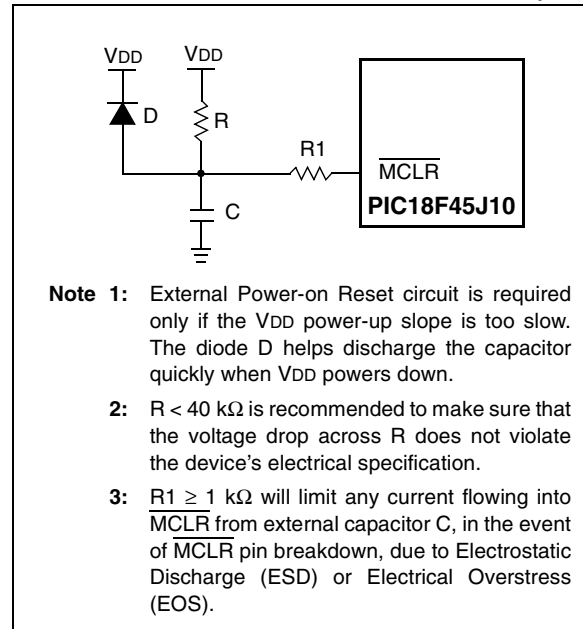
When the device starts normal operation (i.e., exits the Reset condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not met, the device must be held in Reset until the operating conditions are met.

POR events are captured by the $\overline{\text{POR}}$ bit (RCON<1>). The state of the bit is set to '0' whenever a POR occurs; it does not change for any other Reset event. POR is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to '1' in software following any POR.

4.4 Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)

Once a BOR has occurred, the Power-up Timer will keep the chip in Reset for TPWRT (parameter 33). If V_{DD} drops below V_{BOR} 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 V_{DD} rises above V_{BOR} , the Power-up Timer will execute the additional time delay.

FIGURE 4-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW V_{DD} POWER-UP)



4.4.1 DETECTING BOR

The $\overline{\text{BOR}}$ bit always resets to '0' on any BOR or POR event. This makes it difficult to determine if a BOR event has occurred just by reading the state of $\overline{\text{BOR}}$ alone. A more reliable method is to simultaneously check the state of both $\overline{\text{POR}}$ and $\overline{\text{BOR}}$. This assumes that the $\overline{\text{POR}}$ bit is reset to '1' in software immediately after any POR event. If $\overline{\text{BOR}}$ is '0' while $\overline{\text{POR}}$ is '1', it can be reliably assumed that a BOR event has occurred.

In devices designated with an "LF" part number (such as PIC18LF25J10), Brown-out Reset functionality is disabled. In this case, the $\overline{\text{BOR}}$ bit cannot be used to determine a BOR event. The $\overline{\text{BOR}}$ bit is still cleared by a POR event.

PIC18F45J10 FAMILY

4.5 Power-up Timer (PWRT)

PIC18F45J10 family devices incorporate an on-chip Power-up Timer (PWRT) to help regulate the Power-on Reset process. The PWRT is always enabled. The main function is to ensure that the device voltage is stable before code is executed.

The Power-up Timer (PWRT) of the PIC18F45J10 family devices is an 11-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of $2048 \times 32 \mu\text{s} = 65.6 \text{ ms}$. 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.

4.5.1 TIME-OUT SEQUENCE

If enabled, the PWRT time-out is invoked after the POR pulse has cleared. The total time-out will vary based on the status of the PWRT. Figure 4-3, Figure 4-4, Figure 4-5 and Figure 4-6 all depict time-out sequences on power-up with the Power-up Timer enabled.

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, the PWRT will expire. Bringing $\overline{\text{MCLR}}$ high will begin execution immediately (Figure 4-5). This is useful for testing purposes, or to synchronize more than one PIC18F device operating in parallel.

FIGURE 4-3: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO V_{DD} , V_{DD} RISE $<$ T_{PWRT})

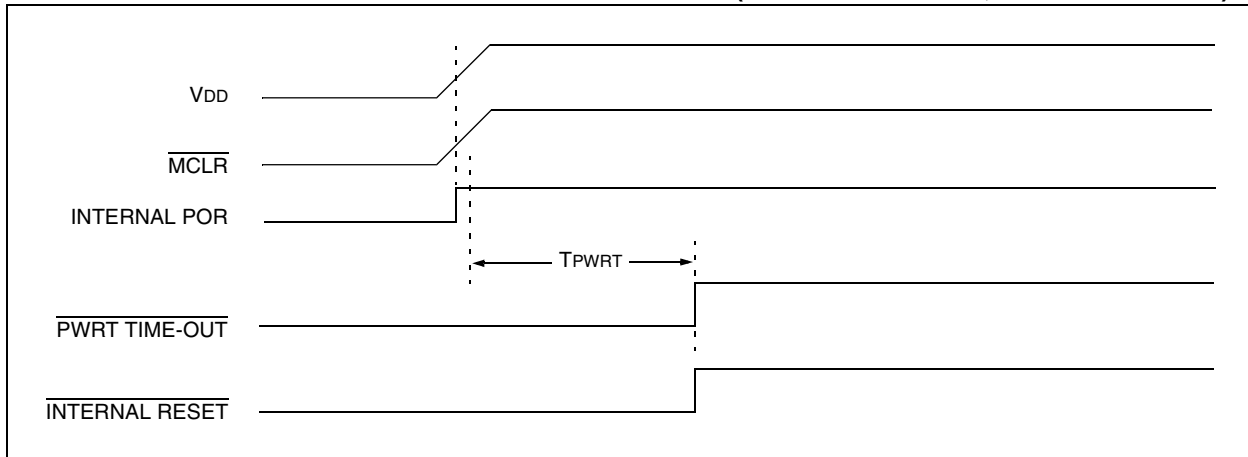
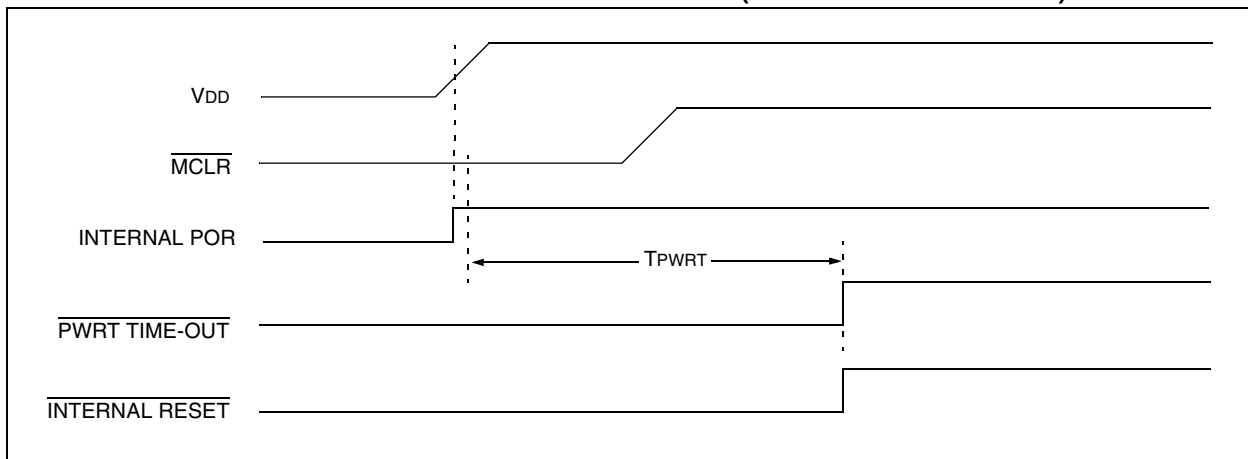


FIGURE 4-4: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 1



PIC18F45J10 FAMILY

FIGURE 4-5: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 2

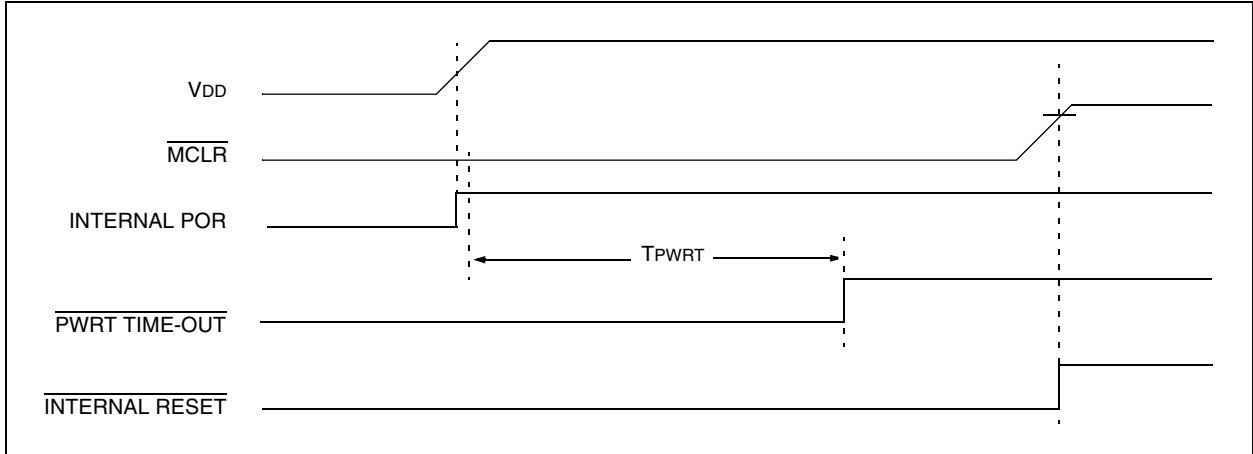
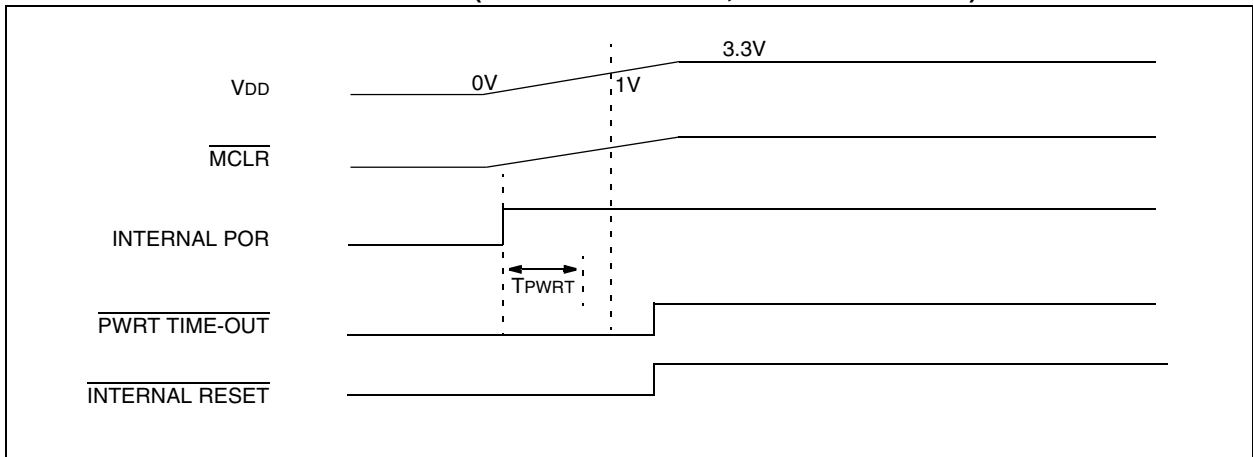


FIGURE 4-6: SLOW RISE TIME ($\overline{\text{MCLR}}$ TIED TO V_{DD} , V_{DD} RISE $>$ T_{PWRT})



PIC18F45J10 FAMILY

4.6 Reset State of Registers

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{RI} , \overline{TO} , \overline{PD} , \overline{POR} and \overline{BOR} , are set or cleared differently in different Reset situations, as indicated in Table 4-1. These bits are used in software to determine the nature of the Reset.

Table 4-2 describes the Reset states for all of the Special Function Registers. These are categorized by Power-on and Brown-out Resets, Master Clear and WDT Resets and WDT wake-ups.

TABLE 4-1: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER

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

Legend: u = unchanged

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 (0008h or 0018h).

2: BOR is not available in PIC18LF2XJ10/4XJ10 devices.

PIC18F45J10 FAMILY

TABLE 4-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS

| Register | Applicable Devices | | Power-on Reset, Brown-out Reset | MCLR Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|----------|--------------------|-------------|------------------------------------|---|---------------------------------|
| TOSU | PIC18F2XJ10 | PIC18F4XJ10 | ---0 0000 | ---0 0000 | ---0 uuuu ⁽¹⁾ |
| TOSH | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu ⁽¹⁾ |
| TOSL | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu ⁽¹⁾ |
| STKPTR | PIC18F2XJ10 | PIC18F4XJ10 | 00-0 0000 | uu-0 0000 | uu-u uuuu ⁽¹⁾ |
| PCLATU | PIC18F2XJ10 | PIC18F4XJ10 | ---0 0000 | ---0 0000 | ---u uuuu |
| PCLATH | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PCL | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | PC + 2 ⁽²⁾ |
| TBLPTRU | PIC18F2XJ10 | PIC18F4XJ10 | --00 0000 | --00 0000 | --uu uuuu |
| TBLPTRH | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TBLPTRL | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TABLAT | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PRODH | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PRODL | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INTCON | PIC18F2XJ10 | PIC18F4XJ10 | 0000 000x | 0000 000u | uuuu uuuu ⁽³⁾ |
| INTCON2 | PIC18F2XJ10 | PIC18F4XJ10 | 1111 -1-1 | 1111 -1-1 | uuuu -u-u ⁽³⁾ |
| INTCON3 | PIC18F2XJ10 | PIC18F4XJ10 | 11-0 0-00 | 11-0 0-00 | uu-u u-uu ⁽³⁾ |
| INDF0 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTINC0 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTDEC0 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PREINC0 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PLUSW0 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| FSR0H | PIC18F2XJ10 | PIC18F4XJ10 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR0L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| WREG | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF1 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTINC1 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTDEC1 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PREINC1 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PLUSW1 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| FSR1H | PIC18F2XJ10 | PIC18F4XJ10 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR1L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| BSR | PIC18F2XJ10 | PIC18F4XJ10 | ---- 0000 | ---- 0000 | ---- 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: 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.

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

4: See Table 4-1 for Reset value for specific condition.

PIC18F45J10 FAMILY

TABLE 4-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

| Register | Applicable Devices | | Power-on Reset, Brown-out Reset | MCLR Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|---------------------|--------------------|-------------|------------------------------------|---|---------------------------------|
| INDF2 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTINC2 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| POSTDEC2 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PREINC2 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| PLUSW2 | PIC18F2XJ10 | PIC18F4XJ10 | N/A | N/A | N/A |
| FSR2H | PIC18F2XJ10 | PIC18F4XJ10 | ---- xxxx | ---- uuuu | ---- uuuu |
| FSR2L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| STATUS | PIC18F2XJ10 | PIC18F4XJ10 | ---x xxxx | ---u uuuu | ---u uuuu |
| TMR0H | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TMR0L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T0CON | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | uuuu uuuu |
| OSCCON | PIC18F2XJ10 | PIC18F4XJ10 | 0--- q-00 | 0--- q-00 | u--- q-uu |
| WDTCON | PIC18F2XJ10 | PIC18F4XJ10 | ---- ---0 | ---- ---0 | ---- ---u |
| RCON ⁽⁴⁾ | PIC18F2XJ10 | PIC18F4XJ10 | 0--1 11q0 | 0--q qquu | u--u qquu |
| TMR1H | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR1L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T1CON | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | u0uu uuuu | uuuu uuuu |
| TMR2 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PR2 | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | 1111 1111 |
| T2CON | PIC18F2XJ10 | PIC18F4XJ10 | -00 0000 | -00 0000 | -uuu uuuu |
| SSP1BUF | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| SSP1ADD | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1STAT | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1CON1 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1CON2 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| ADRESH | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADRESL | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADCON0 | PIC18F2XJ10 | PIC18F4XJ10 | 0-00 0000 | 0-00 0000 | u-uu uuuu |
| ADCON1 | PIC18F2XJ10 | PIC18F4XJ10 | --00 0qqq | --00 0qqq | --uu uqqq |
| ADCON2 | PIC18F2XJ10 | PIC18F4XJ10 | 0-00 0000 | 0-00 0000 | u-uu 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:** 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.
- 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:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 4:** See Table 4-1 for Reset value for specific condition.

PIC18F45J10 FAMILY

TABLE 4-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

| Register | Applicable Devices | | Power-on Reset, Brown-out Reset | MCLR Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|----------|--------------------|-------------|------------------------------------|---|---------------------------------|
| CCPR1H | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR1L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP1CON | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CCPR2H | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR2L | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP2CON | PIC18F2XJ10 | PIC18F4XJ10 | --00 0000 | --00 0000 | --uu uuuu |
| BAUDCON | PIC18F2XJ10 | PIC18F4XJ10 | 01-0 0-00 | 01-0 0-00 | uu-u u-uu |
| ECCP1DEL | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| ECCP1AS | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CVRCON | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CMCON | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0111 | 0000 0111 | uuuu uuuu |
| SPBRGH | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SPBRG | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| RCREG | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TXREG | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TXSTA | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0010 | 0000 0010 | uuuu uuuu |
| RCSTA | PIC18F2XJ10 | PIC18F4XJ10 | 0000 000x | 0000 000x | uuuu uuuu |
| EECON2 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| EECON1 | PIC18F2XJ10 | PIC18F4XJ10 | ---0 x00- | ---0 x00- | ---u uu- |
| IPR3 | PIC18F2XJ10 | PIC18F4XJ10 | 11-- ---- | 11-- ---- | uu-- ---- |
| PIR3 | PIC18F2XJ10 | PIC18F4XJ10 | 00-- ---- | 00-- ---- | uu-- ---- ⁽³⁾ |
| PIE3 | PIC18F2XJ10 | PIC18F4XJ10 | 00-- ---- | 00-- ---- | uu-- ---- |
| IPR2 | PIC18F2XJ10 | PIC18F4XJ10 | 11-- 1--1 | 11-- 1--1 | uu-- u--u |
| PIR2 | PIC18F2XJ10 | PIC18F4XJ10 | 00-- 0--0 | 00-- 0--0 | uu-- u--u ⁽³⁾ |
| PIE2 | PIC18F2XJ10 | PIC18F4XJ10 | 00-- 0--0 | 00-- 0--0 | uu-- u--u |
| IPR1 | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | uuuu uuuu |
| PIR1 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu ⁽³⁾ |
| PIE1 | PIC18F2XJ10 | PIC18F4XJ10 | 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: 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.

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

4: See Table 4-1 for Reset value for specific condition.

PIC18F45J10 FAMILY

TABLE 4-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

| Register | Applicable Devices | | Power-on Reset, Brown-out Reset | MCLR Resets WDT Reset RESET Instruction Stack Resets | Wake-up via WDT or Interrupt |
|----------|--------------------|-------------|------------------------------------|---|---------------------------------|
| TRISE | PIC18F2XJ10 | PIC18F4XJ10 | 0000 -111 | 1111 -111 | uuuu -uuu |
| TRISD | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISC | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISB | PIC18F2XJ10 | PIC18F4XJ10 | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA | PIC18F2XJ10 | PIC18F4XJ10 | --1- 1111 | --1- 1111 | --u- uuuu |
| SSP2BUF | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATE | PIC18F2XJ10 | PIC18F4XJ10 | ---- -xxx | ---- -uuu | ---- -uuu |
| LATD | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATC | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATB | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| LATA | PIC18F2XJ10 | PIC18F4XJ10 | --x- xxxx | --u- uuuu | --u- uuuu |
| SSP2ADD | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP2STAT | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP2CON1 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP2CON2 | PIC18F2XJ10 | PIC18F4XJ10 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PORTE | PIC18F2XJ10 | PIC18F4XJ10 | ---- -xxx | ---- -uuu | ---- -uuu |
| PORTD | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTC | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTB | PIC18F2XJ10 | PIC18F4XJ10 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTA | PIC18F2XJ10 | PIC18F4XJ10 | --0- 0000 | --0- 0000 | --u- 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:** 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.
- 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:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 4:** See Table 4-1 for Reset value for specific condition.

PIC18F45J10 FAMILY

5.0 MEMORY ORGANIZATION

There are two types of memory in PIC18 Enhanced microcontroller devices:

- Program Memory
- Data RAM

As Harvard architecture devices, the data and program memories use separate busses; this allows for concurrent access of the two memory spaces.

Additional detailed information on the operation of the Flash program memory is provided in **Section 6.0 “Flash Program Memory”**.

5.1 Program Memory Organization

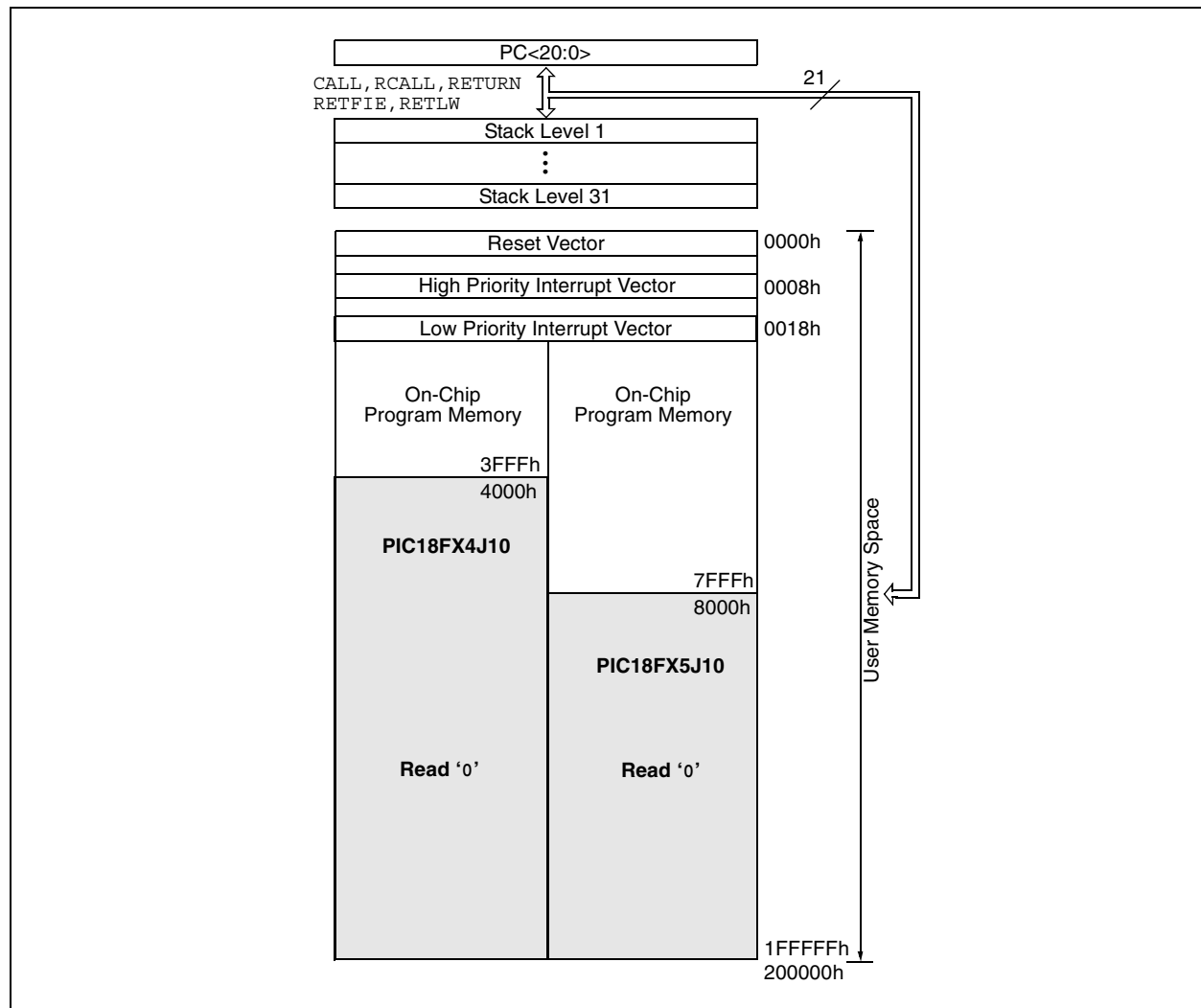
PIC18 microcontrollers implement a 21-bit program counter, which is capable of addressing a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all ‘0’s (a NOP instruction).

The PIC18F24J10 and PIC18F44J10 each have 16 Kbytes of Flash memory and can store up to 8,192 single-word instructions. The PIC18F25J10 and PIC18F45J10 each have 32 Kbytes of Flash memory and can store up to 16,384 single-word instructions.

PIC18 devices have two interrupt vectors. The Reset vector address is at 0000h and the interrupt vector addresses are at 0008h and 0018h.

The program memory map for the PIC18F45J10 family devices is shown in Figure 5-1.

FIGURE 5-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F45J10 FAMILY DEVICES



PIC18F45J10 FAMILY

5.1.1 PROGRAM COUNTER

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide and is contained in three separate 8-bit registers. 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; it is not directly readable or writable. Updates to the PCH register are performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits; it is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.

The contents of PCLATH and PCLATU are transferred to the program counter by any operation that writes PCL. Similarly, the upper two bytes of the program counter are transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see **Section 5.1.4.1 “Computed GOTO”**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit 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.

5.1.2 RETURN ADDRESS STACK

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC 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 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, STKPTR. 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 these registers.

A CALL type instruction causes a push onto the stack; the Stack Pointer is first incremented and the location pointed to by the Stack Pointer is written with the contents of the PC (already pointing to the instruction following the CALL). A RETURN type instruction causes a pop from the stack; the contents of the location pointed to by the STKPTR are transferred to the PC and then the Stack Pointer is decremented.

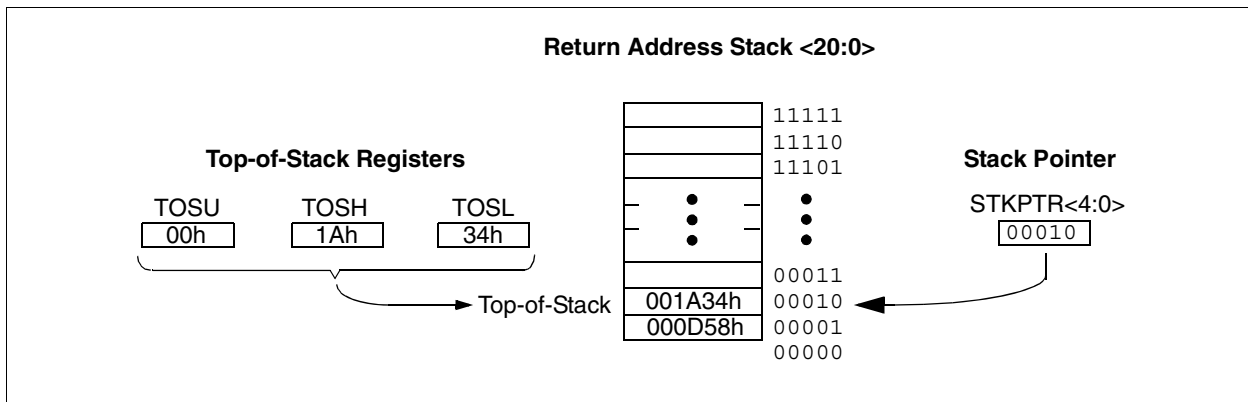
The Stack Pointer is initialized to ‘00000’ after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of ‘00000’; this is only a Reset value. Status bits indicate if the stack is full or has overflowed or has underflowed.

5.1.2.1 Top-of-Stack Access

Only the top of the return address stack (TOS) is readable and writable. A set of three registers, TOSU:TOSH:TOSL, hold the contents of the stack location pointed to by the STKPTR register (Figure 5-2). 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:TOSL registers. These values can be placed on a user-defined software stack. At return time, the software can return these values to TOSU:TOSH:TOSL and do a return.

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

FIGURE 5-2: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



PIC18F45J10 FAMILY

5.1.2.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 5-1) contains the Stack Pointer value, the STKFUL (Stack Overflow) 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. On 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 (RTOS) 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 20.1 “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 the 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 until 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.

5.1.2.3 PUSH and POP Instructions

Since the Top-of-Stack 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 feature. The PIC18 instruction set includes two instructions, `PUSH` and `POP`, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

The `PUSH` instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack.

The `POP` instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

REGISTER 5-1: STKPTR: STACK POINTER 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 ⁽¹⁾ | STKUNF ⁽¹⁾ | — | SP4 | SP3 | SP2 | SP1 | SP0 | |
| bit 7 | | | | | | | | bit 0 |

- bit 7 **STKFUL:** Stack Overflow Flag bit⁽¹⁾
 1 = Stack became full or overflowed
 0 = Stack has not become full or overflowed
- bit 6 **STKUNF:** Stack Underflow Flag bit⁽¹⁾
 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 |

PIC18F45J10 FAMILY

5.1.2.4 Stack Full and Underflow Resets

Device Resets on stack overflow and stack underflow conditions are enabled by setting the STVREN bit in Configuration Register 4L. When STVREN is set, a full or underflow will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. When STVREN is cleared, a full or underflow condition will set the appropriate STKFUL or STKUNF bit but not cause a device Reset. The STKFUL or STKUNF bits are cleared by the user software or a Power-on Reset.

5.1.3 FAST REGISTER STACK

A fast register stack is provided for the STATUS, WREG and BSR registers, to provide a “fast return” option for interrupts. The stack for each register is only one level deep and is neither readable nor writable. It is loaded with the current value of the corresponding register when the processor vectors for an interrupt. All interrupt sources will push values into the stack registers. The values in the registers are then loaded back into their associated registers if the RETFIE, FAST instruction is used to return from the interrupt.

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. In these cases, 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
```

5.1.4 LOOK-UP TABLES IN PROGRAM MEMORY

There may be programming situations that require the creation of data structures, or look-up tables, in program memory. For PIC18 devices, look-up tables can be implemented in two ways:

- Computed GOTO
- Table Reads

5.1.4.1 Computed GOTO

A computed GOTO is accomplished by adding an offset to the program counter. An example is shown in Example 5-2.

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW nn instructions. The W register 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 nn instructions that returns the value ‘nn’ 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 5-2: COMPUTED GOTO USING AN OFFSET VALUE

```
MOVWF OFFSET, W
CALL TABLE
ORG nn00h
TABLE ADDWF PCL
      RETLW nnh
      RETLW nnh
      RETLW nnh
      .
      .
      .
```

5.1.4.2 Table Reads and 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) register specifies the byte address and the Table Latch (TABLAT) register contains the data that is read from or written to program memory. Data is transferred to or from program memory one byte at a time.

Table read and table write operations are discussed further in Section 6.1 “Table Reads and Table Writes”.

PIC18F45J10 FAMILY

5.2 PIC18 Instruction Cycle

5.2.1 CLOCKING SCHEME

The microcontroller clock input, whether from an internal or external source, is internally divided by four to generate four non-overlapping quadrature clocks (Q1, Q2, Q3 and Q4). Internally, the program counter is incremented on every Q1; the instruction is fetched from the program memory and latched into the instruction register during Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 5-3.

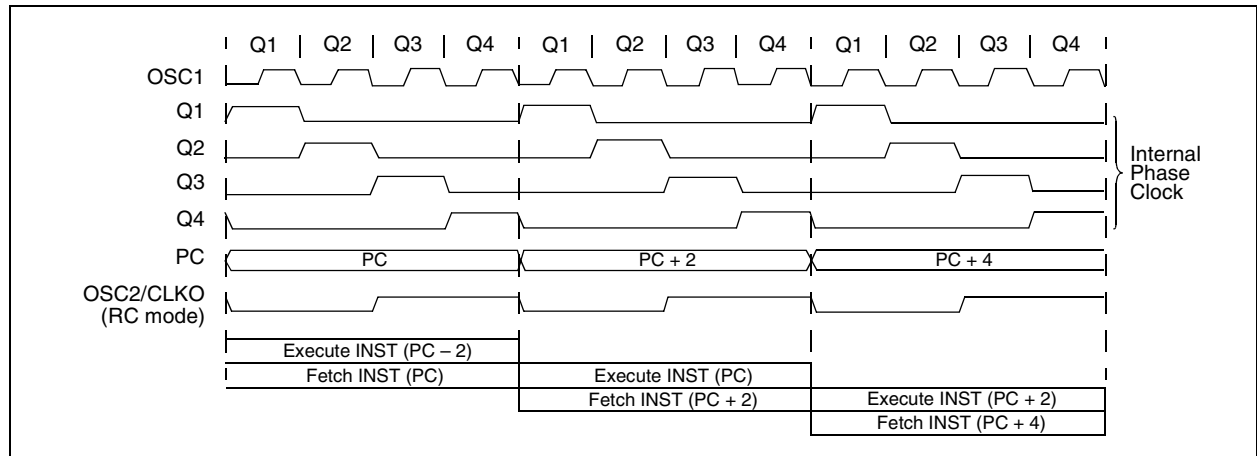
5.2.2 INSTRUCTION FLOW/PIPELINING

An "Instruction Cycle" consists of four Q cycles: Q1 through Q4. The instruction fetch and execute are pipelined in such a manner that a fetch takes one instruction cycle, while the 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 5-3).

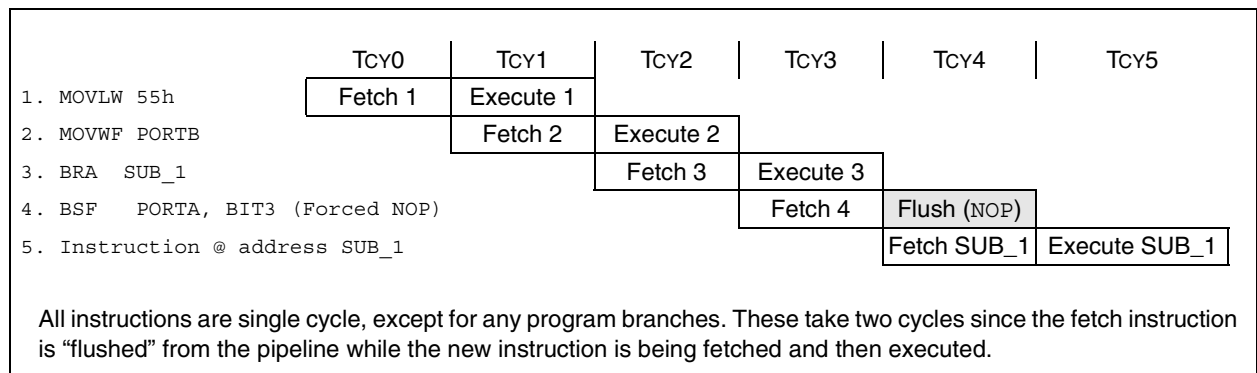
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

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 5-3: CLOCK/INSTRUCTION CYCLE



EXAMPLE 5-3: INSTRUCTION PIPELINE FLOW



PIC18F45J10 FAMILY

5.2.3 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). To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSb will always read '0' (see **Section 5.1.1 "Program Counter"**).

Figure 5-4 shows an example of how instruction words are stored in the program memory.

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 5-4 shows how the instruction, **GOTO 0006h**, 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 21.0 "Instruction Set Summary"** provides further details of the instruction set.

FIGURE 5-4: INSTRUCTIONS IN PROGRAM MEMORY

| Program Memory Byte Locations → | | | Word Address | | |
|------------------------------------|-------|------------|--------------|---------|---------|
| | | | LSB = 1 | LSB = 0 | |
| | | | | 000000h | |
| | | | | 000002h | |
| | | | | 000004h | |
| | | | | 000006h | |
| Instruction 1: | MOVLW | 055h | 0Fh | 55h | 000008h |
| Instruction 2: | GOTO | 0006h | EFh | 03h | 00000Ah |
| | | | F0h | 00h | 00000Ch |
| Instruction 3: | MOVFF | 123h, 456h | C1h | 23h | 00000Eh |
| | | | F4h | 56h | 000010h |
| | | | | | 000012h |
| | | | | | 000014h |

5.2.4 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: **CALL**, **MOVFF**, **GOTO** and **LSFR**. In all cases, the second word of the instructions always has '1111' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSBs of an instruction specifies a special form of **NOP**. If the instruction is executed in proper sequence – immediately after the first word – the data in the second word is accessed and used by

the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a **NOP** is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 5-4 shows how this works.

Note: See **Section 5.6 "PIC18 Instruction Execution and the Extended Instruction Set"** for information on two-word instructions in the extended instruction set.

EXAMPLE 5-4: 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 |

5.3 Data Memory Organization

Note: The operation of some aspects of data memory are changed when the PIC18 extended instruction set is enabled. See **Section 5.5 “Data Memory and the Extended Instruction Set”** for more information.

The data memory in PIC18 devices is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4096 bytes of data memory. The memory space is divided into as many as 16 banks that contain 256 bytes each; PIC18F45J10 family devices implement all 16 banks. Figure 5-5 shows the data memory organization for the PIC18F45J10 family devices.

The data memory contains Special Function Registers (SFRs) and General Purpose Registers (GPRs). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratchpad operations in the user's application. Any read of an unimplemented location will read as '0's.

The instruction set and architecture allow operations across all banks. The entire data memory may be accessed by Direct, Indirect or Indexed Addressing modes. Addressing modes are discussed later in this subsection.

To ensure that commonly used registers (SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to SFRs and the lower portion of GPR Bank 0 without using the BSR. **Section 5.3.2 “Access Bank”** provides a detailed description of the Access RAM.

5.3.1 BANK SELECT REGISTER (BSR)

Large areas of data memory require an efficient addressing scheme to make rapid access to any address possible. Ideally, this means that an entire address does not need to be provided for each read or write operation. For PIC18 devices, this is accomplished with a RAM banking scheme. This divides the memory space into 16 contiguous banks of 256 bytes. Depending on the instruction, each location can be addressed directly by its full 12-bit address, or an 8-bit low-order address and a 4-bit bank pointer.

Most instructions in the PIC18 instruction set make use of the bank pointer, known as the Bank Select Register (BSR). This SFR holds the 4 Most Significant bits of a location's address; the instruction itself includes the 8 Least Significant bits. Only the four lower bits of the BSR are implemented (BSR3:BSR0). The upper four bits are unused; they will always read '0' and cannot be written to. The BSR can be loaded directly by using the `MOVLB` instruction.

The value of the BSR indicates the bank in data memory. The 8 bits in the instruction show the location in the bank and can be thought of as an offset from the bank's lower boundary. The relationship between the BSR's value and the bank division in data memory is shown in Figure 5-6.

Since up to 16 registers may share the same low-order address, the user must always be careful to ensure that the proper bank is selected before performing a data read or write. For example, writing what should be program data to an 8-bit address of F9h while the BSR is 0Fh will end up resetting the program counter.

While any bank can be selected, only those banks that are actually implemented can be read or written to. Writes to unimplemented banks are ignored, while reads from unimplemented banks will return '0's. Even so, the STATUS register will still be affected as if the operation was successful. The data memory map in Figure 5-5 indicates which banks are implemented.

In the core PIC18 instruction set, only the `MOVFF` instruction fully specifies the 12-bit address of the source and target registers. This instruction ignores the BSR completely when it executes. All other instructions include only the low-order address as an operand and must use either the BSR or the Access Bank to locate their target registers.

PIC18F45J10 FAMILY

FIGURE 5-5: DATA MEMORY MAP FOR PIC18F24J10/44J10 DEVICES

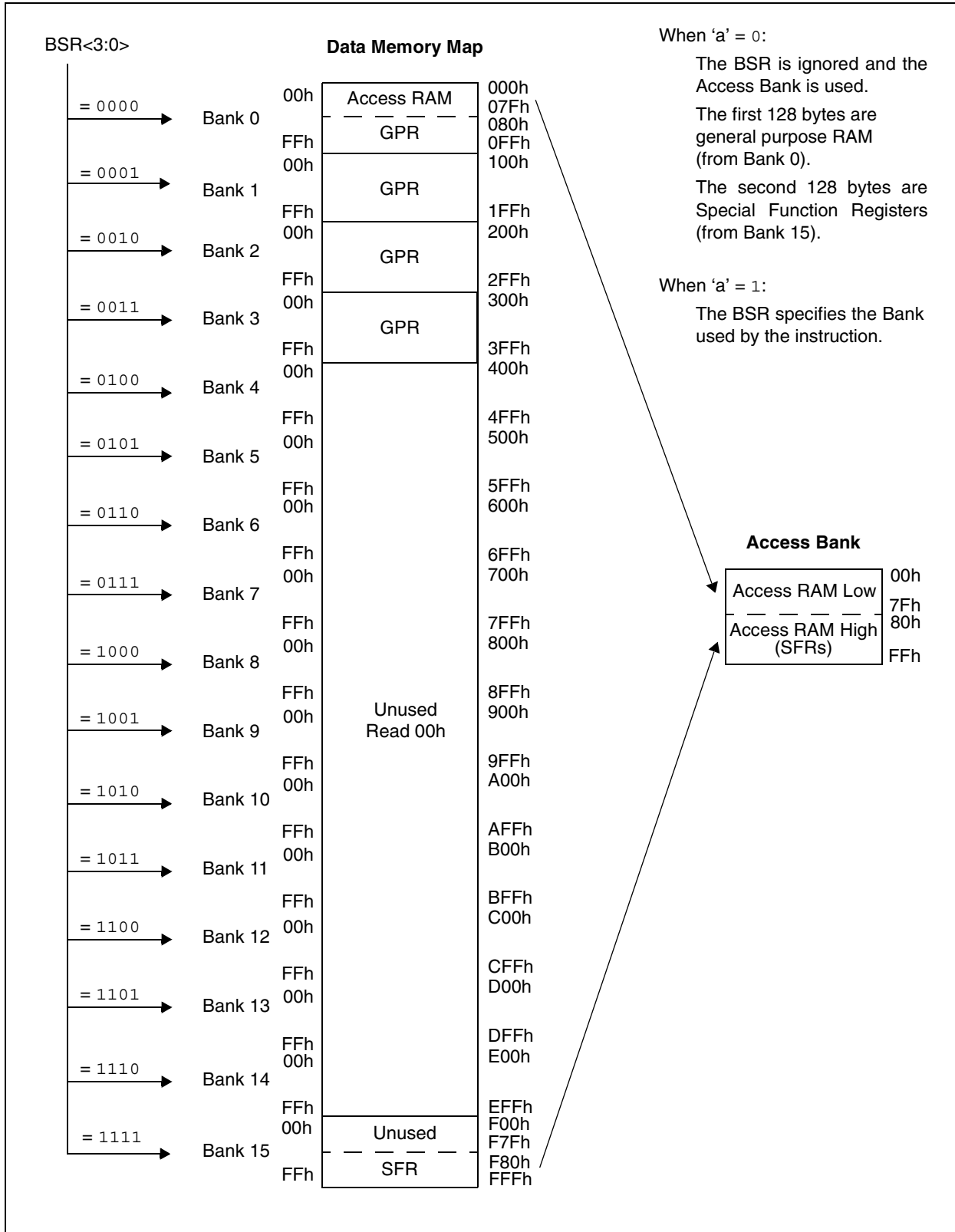
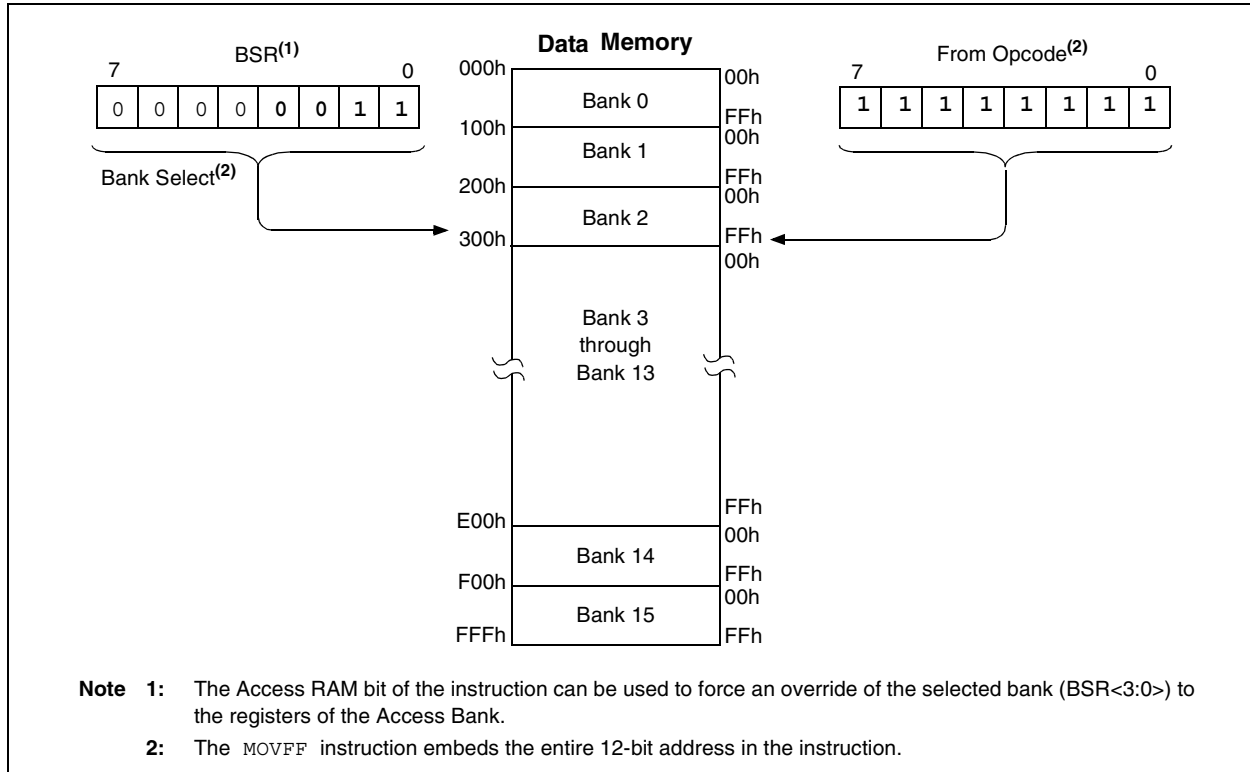


FIGURE 5-6: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



5.3.2 ACCESS BANK

While the use of the BSR with an embedded 8-bit address allows users to address the entire range of data memory, it also means that the user must always ensure that the correct bank is selected. Otherwise, data may be read from or written to the wrong location. This can be disastrous if a GPR is the intended target of an operation but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 128 bytes of memory (00h-7Fh) in Bank 0 and the last 128 bytes of memory (80h-FFh) in Block 15. The lower half is known as the "Access RAM" and is composed of GPRs. This upper half is also where the device's SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an 8-bit address (Figure 5-5).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the 'a' parameter in the instruction). When 'a' is equal to '1', the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When 'a' is '0',

however, the instruction is forced to use the Access Bank address map; the current value of the BSR is ignored entirely.

Using this "forced" addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 80h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 80h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables. Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST configuration bit = 1). This is discussed in more detail in **Section 5.5.3 "Mapping the Access Bank in Indexed Literal Offset Mode"**.

5.3.3 GENERAL PURPOSE REGISTER FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

PIC18F45J10 FAMILY

5.3.4 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. SFRs start at the top of data memory (FFFh) and extend downward to occupy the top half of Bank 15 (F80h to FFFh). 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” device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The reset and interrupt registers are described in their respective chapters, while the ALU’s STATUS register is described later in this section. Registers related to the operation of a peripheral feature are described in the chapter for that peripheral.

The SFRs are typically distributed among the peripherals whose functions they control. Unused SFR locations are unimplemented and read as ‘0’s.

TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F45J10 FAMILY DEVICES

| Address | Name | Address | Name | Address | Name | Address | Name |
|---------|-------------------------|---------|-------------------------|---------|-------------------------|---------|-------------------------|
| FFFh | TOSU | FDfH | INDF2 ⁽¹⁾ | FBFh | CCPR1H | F9Fh | IPR1 |
| FFEh | TOSH | FDEh | POSTINC2 ⁽¹⁾ | FBEh | CCPR1L | F9Eh | PIR1 |
| FFDh | TOSL | FDDh | POSTDEC2 ⁽¹⁾ | FBDh | CCP1CON | F9Dh | PIE1 |
| FFCh | STKPTR | FDCh | PREINC2 ⁽¹⁾ | FBCh | CCPR2H | F9Ch | — ⁽²⁾ |
| FFBh | PCLATU | FDBh | PLUSW2 ⁽¹⁾ | FBBh | CCPR2L | F9Bh | — ⁽²⁾ |
| FFAh | PCLATH | FDAh | FSR2H | FBAh | CCP2CON | F9Ah | — ⁽²⁾ |
| FF9h | PCL | FD9h | FSR2L | FB9h | — ⁽²⁾ | F99h | — ⁽²⁾ |
| FF8h | TBLPTRU | FD8h | STATUS | FB8h | BAUDCON | F98h | — ⁽²⁾ |
| FF7h | TBLPTRH | FD7h | TMR0H | FB7h | ECCP1DEL ⁽³⁾ | F97h | — ⁽²⁾ |
| FF6h | TBLPTRL | FD6h | TMR0L | FB6h | ECCP1AS ⁽³⁾ | F96h | TRISE ⁽³⁾ |
| FF5h | TABLAT | FD5h | T0CON | FB5h | CVRCON | F95h | TRISD ⁽³⁾ |
| FF4h | PRODH | FD4h | — ⁽²⁾ | FB4h | CMCON | F94h | TRISC |
| FF3h | PRODL | FD3h | OSCCON | FB3h | — ⁽²⁾ | F93h | TRISB |
| FF2h | INTCON | FD2h | — ⁽²⁾ | FB2h | — ⁽²⁾ | F92h | TRISA |
| FF1h | INTCON2 | FD1h | WDTCON | FB1h | — ⁽²⁾ | F91h | — ⁽²⁾ |
| FF0h | INTCON3 | FD0h | RCON | FB0h | SPBRGH | F90h | — ⁽²⁾ |
| FEFh | INDF0 ⁽¹⁾ | FCFh | TMR1H | FAFh | SPBRG | F8Fh | — ⁽²⁾ |
| FEeh | POSTINC0 ⁽¹⁾ | FCEh | TMR1L | FAEh | RCREG | F8Eh | SSP2BUF |
| FEDh | POSTDEC0 ⁽¹⁾ | FCDh | T1CON | FADh | TXREG | F8Dh | LATE ⁽³⁾ |
| FECh | PREINC0 ⁽¹⁾ | FCCh | TMR2 | FACh | TXSTA | F8Ch | LATD ⁽³⁾ |
| FEbh | PLUSW0 ⁽¹⁾ | FCBh | PR2 | FABh | RCSTA | F8Bh | LATC |
| FEAh | FSR0H | CAh | T2CON | FAAh | — ⁽²⁾ | F8Ah | LATB |
| FE9h | FSR0L | FC9h | SSP1BUF | FA9h | — ⁽²⁾ | F89h | LATA |
| FE8h | WREG | FC8h | SSP1ADD | FA8h | — ⁽²⁾ | F88h | SSP2ADD ⁽³⁾ |
| FE7h | INDF1 ⁽¹⁾ | FC7h | SSP1STAT | FA7h | EECON2 ⁽¹⁾ | F87h | SSP2STAT ⁽³⁾ |
| FE6h | POSTINC1 ⁽¹⁾ | FC6h | SSP1CON1 | FA6h | EECON1 | F86h | SSP2CON1 ⁽³⁾ |
| FE5h | POSTDEC1 ⁽¹⁾ | FC5h | SSP1CON2 | FA5h | IPR3 | F85h | SSP2CON2 ⁽³⁾ |
| FE4h | PREINC1 ⁽¹⁾ | FC4h | ADRESH | FA4h | PIR3 | F84h | PORTE ⁽³⁾ |
| FE3h | PLUSW1 ⁽¹⁾ | FC3h | ADRESL | FA3h | PIE3 | F83h | PORTD ⁽³⁾ |
| FE2h | FSR1H | FC2h | ADCON0 | FA2h | IPR2 | F82h | PORTC |
| FE1h | FSR1L | FC1h | ADCON1 | FA1h | PIR2 | F81h | PORTB |
| FE0h | BSR | FC0h | ADCON2 | FA0h | PIE2 | F80h | PORTA |

- Note 1:** This is not a physical register.
2: Unimplemented registers are read as ‘0’.
3: This register is not available in 28-pin devices.

PIC18F45J10 FAMILY

TABLE 5-2: REGISTER FILE SUMMARY (PIC18F24J10/25J10/44J10/45J10)

| 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 | 43, 48 |
| TOSH | Top-of-Stack High Byte (TOS<15:8>) | | | | | | | | 0000 0000 | 43, 48 |
| TOSL | Top-of-Stack Low Byte (TOS<7:0>) | | | | | | | | 0000 0000 | 43, 48 |
| STKPTR | STKFUL | STKUNF | — | Return Stack Pointer | | | | | 00-0 0000 | 43, 49 |
| PCLATU | — | — | — | Holding Register for PC<20:16> | | | | | ---0 0000 | 43, 48 |
| PCLATH | Holding Register for PC<15:8> | | | | | | | | 0000 0000 | 43, 48 |
| PCL | PC Low Byte (PC<7:0>) | | | | | | | | 0000 0000 | 43, 48 |
| TBLPTRU | — | — | bit 21 | Program Memory Table Pointer Upper Byte (TBLPTR<20:16>) | | | | | --00 0000 | 43, 70 |
| TBLPTRH | Program Memory Table Pointer High Byte (TBLPTR<15:8>) | | | | | | | | 0000 0000 | 43, 70 |
| TBLPTRL | Program Memory Table Pointer Low Byte (TBLPTR<7:0>) | | | | | | | | 0000 0000 | 43, 70 |
| TABLAT | Program Memory Table Latch | | | | | | | | 0000 0000 | 43, 70 |
| PRODH | Product Register High Byte | | | | | | | | xxxx xxxx | 43, 77 |
| PRODL | Product Register Low Byte | | | | | | | | xxxx xxxx | 43, 77 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 0000 000x | 43, 81 |
| INTCON2 | RBP \bar{U} | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP | 1111 -1-1 | 43, 82 |
| INTCON3 | INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF | 11-0 0-00 | 43, 83 |
| INDF0 | Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register) | | | | | | | | N/A | 43, 62 |
| POSTINC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| POSTDEC0 | Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| PREINC0 | Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| PLUSW0 | Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by W | | | | | | | | N/A | 43, 62 |
| FSR0H | — | — | — | — | Indirect Data Memory Address Pointer 0 High Byte | | | | ---- xxxxx | 43, 62 |
| FSR0L | Indirect Data Memory Address Pointer 0 Low Byte | | | | | | | | xxxxx xxxxx | 43, 62 |
| WREG | Working Register | | | | | | | | xxxxx xxxxx | 43 |
| INDF1 | Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register) | | | | | | | | N/A | 43, 62 |
| POSTINC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| POSTDEC1 | Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| PREINC1 | Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) | | | | | | | | N/A | 43, 62 |
| PLUSW1 | Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by W | | | | | | | | N/A | 43, 62 |
| FSR1H | — | — | — | — | Indirect Data Memory Address Pointer 1 High Byte | | | | ---- xxxxx | 43, 62 |
| FSR1L | Indirect Data Memory Address Pointer 1 Low Byte | | | | | | | | xxxxx xxxxx | 43, 62 |
| BSR | — | — | — | — | Bank Select Register | | | | ---- 0000 | 43, 53 |
| INDF2 | Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register) | | | | | | | | N/A | 44, 62 |
| POSTINC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register) | | | | | | | | N/A | 44, 62 |
| POSTDEC2 | Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register) | | | | | | | | N/A | 44, 62 |
| PREINC2 | Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) | | | | | | | | N/A | 44, 62 |
| PLUSW2 | Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by W | | | | | | | | N/A | 44, 62 |
| FSR2H | — | — | — | — | Indirect Data Memory Address Pointer 2 High Byte | | | | ---- xxxxx | 44, 62 |
| FSR2L | Indirect Data Memory Address Pointer 2 Low Byte | | | | | | | | xxxxx xxxxx | 44, 62 |
| STATUS | — | — | — | N | OV | Z | DC | C | ---x xxxxx | 44, 60 |

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

Note 1: See Section 4.4 “Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)”.

Note 2: These registers and/or bits are not implemented on 28-pin devices and are read as ‘0’. Reset values are shown for 40/44-pin devices; individual unimplemented bits should be interpreted as ‘-’.

PIC18F45J10 FAMILY

TABLE 5-2: REGISTER FILE SUMMARY (PIC18F24J10/25J10/44J10/45J10) (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: |
|-----------|--|---------------------|---------------------|---------------------|---------------------|---------------------|-----------------------|-----------------------|-------------------|------------------|
| TMR0H | Timer0 Register High Byte | | | | | | | | 0000 0000 | 44, 113 |
| TMR0L | Timer0 Register Low Byte | | | | | | | | xxxx xxxx | 44, 113 |
| T0CON | TMR0ON | T08BIT | T0CS | T0SE | PSA | T0PS2 | T0PS1 | T0PS0 | 1111 1111 | 44, 111 |
| OSCCON | IDLEN | — | — | — | OSTS | — | SCS1 | SCS0 | 0--- q-00 | 28, 44 |
| WDTCON | — | — | — | — | — | — | — | SWDTEN | --- -0 | 44, 235 |
| RCON | IPEN | — | — | RI | TO | PD | POR | BOR ⁽¹⁾ | 0--1 11q0 | 38, 42, 90 |
| TMR1H | Timer1 Register High Byte | | | | | | | | xxxx xxxx | 44, 119 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | xxxx xxxx | 44, 119 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYNC | TMR1CS | TMR1ON | 0000 0000 | 44, 115 |
| TMR2 | Timer2 Register | | | | | | | | 0000 0000 | 44, 122 |
| PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 44, 122 |
| T2CON | — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | -00 0000 | 44, 121 |
| SSP1BUF | MSSP1 Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | 44, 154 |
| SSP1ADD | MSSP1 Address Register in I ² C™ Slave mode. MSSP1 Baud Rate Reload Register in I ² C Master mode. | | | | | | | | 0000 0000 | 44, 155 |
| SSP1STAT | SMP | CKE | D/A | P | S | R/W | UA | BF | 0000 0000 | 44, 146, 156 |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 44, 147, 157 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 44, 158 |
| ADRESH | A/D Result Register High Byte | | | | | | | | xxxx xxxx | 44, 218 |
| ADRESL | A/D Result Register Low Byte | | | | | | | | xxxx xxxx | 44, 218 |
| ADCON0 | ADCAL | — | CHS3 | CHS2 | CHS1 | CHS0 | GO/DONE | ADON | 0-00 0000 | 44, 209 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | --00 0qqq | 44, 210 |
| ADCON2 | ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 | 0-00 0000 | 44, 211 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | xxxx xxxx | 45, 124 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | xxxx xxxx | 45, 124 |
| CCP1CON | P1M1 ⁽²⁾ | P1M0 ⁽²⁾ | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 0000 0000 | 45, 123 |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte | | | | | | | | xxxx xxxx | 45, 124 |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte | | | | | | | | xxxx xxxx | 45, 124 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | --00 0000 | 45, 123 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 01-0 0-00 | 45, 190 |
| ECCP1DEL | PRSEN | PDC6 ⁽²⁾ | PDC5 ⁽²⁾ | PDC4 ⁽²⁾ | PDC3 ⁽²⁾ | PDC2 ⁽²⁾ | PDC1 ⁽²⁾ | PDC0 ⁽²⁾ | 0000 0000 | 45, 140 |
| ECCP1AS | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 ⁽²⁾ | PSSBD0 ⁽²⁾ | 0000 0000 | 45, 141 |
| CVRCON | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 0000 0000 | 45, 225 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 0000 0111 | 45, 219 |

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

Note 1: See Section 4.4 “Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)”.

Note 2: These registers and/or bits are not implemented on 28-pin devices and are read as ‘0’. Reset values are shown for 40/44-pin devices; individual unimplemented bits should be interpreted as ‘-’.

PIC18F45J10 FAMILY

TABLE 5-2: REGISTER FILE SUMMARY (PIC18F24J10/25J10/44J10/45J10) (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: |
|----------------------|--|---------|--|---------|--------|--|--------------------|--------------------|-------------------|------------------|
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 0000 0000 | 45, 192 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 0000 0000 | 45, 192 |
| RCREG | EUSART Receive Register | | | | | | | | 0000 0000 | 45, 199 |
| TXREG | EUSART Transmit Register | | | | | | | | xxxx xxxx | 45, 197 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 45, 188 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 45, 189 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 0000 0000 | 45, 68 |
| EECON1 | — | — | — | FREE | WRERR | WREN | WR | — | ---0 x00- | 45, 69 |
| IPR3 | SSP2IP | BCL2IP | — | — | — | — | — | — | 11-- ---- | 45, 89 |
| PIR3 | SSP2IF | BCL2IF | — | — | — | — | — | — | 00-- ---- | 45, 85 |
| PIE3 | SSP2IE | BCL2IE | — | — | — | — | — | — | 00-- ---- | 45, 87 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 11-- 1--1 | 45, 89 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 00-- 0--0 | 45, 85 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 00-- 0--0 | 45, 87 |
| IPR1 | PSPIP ⁽²⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 1111 1111 | 45, 88 |
| PIR1 | PSPIF ⁽²⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 45, 84 |
| PIE1 | PSPIE ⁽²⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 45, 86 |
| TRISE ⁽²⁾ | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 1111 -111 | 46, 107 |
| TRISD ⁽²⁾ | PORTD Data Direction Control Register | | | | | | | | 1111 1111 | 46, 103 |
| TRISC | PORTC Data Direction Control Register | | | | | | | | 1111 1111 | 46, 100 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 1111 1111 | 46, 97 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | --1- 1111 | 46, 94 |
| SSP2BUF | MSSP2 Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | 46, 154 |
| LATE ⁽²⁾ | — | — | — | — | — | PORTE Data Latch Register (Read and Write to Data Latch) | | — | ---- -xxx | 46, 106 |
| LATD ⁽²⁾ | PORTD Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 46, 103 |
| LATC | PORTC Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 46, 100 |
| LATB | PORTB Data Latch Register (Read and Write to Data Latch) | | | | | | | | xxxx xxxx | 46, 97 |
| LATA | — | — | PORTA Data Latch Register (Read and Write to Data Latch) | | | | | — | --xx xxxx | 46, 94 |
| SSP2ADD | MSSP2 Address Register in I ² C™ Slave mode. MSSP2 Baud Rate Reload Register in I ² C Master mode. | | | | | | | | 0000 0000 | 46, 154 |
| SSP2STAT | SMP | CKE | D/Ā | P | S | R/W | UA | BF | 0000 0000 | 46, 146, 156 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 0000 0000 | 46, 147, 157 |
| SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 46, 158 |
| PORTE ⁽²⁾ | — | — | — | — | — | RE2 ⁽²⁾ | RE1 ⁽²⁾ | RE0 ⁽²⁾ | ---- -xxx | 46, 106 |
| PORTD ⁽²⁾ | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | xxxx xxxx | 46, 103 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | 46, 100 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | 46, 97 |
| PORTA | — | — | RA5 | — | RA3 | RA2 | RA1 | RA0 | --0- 0000 | 46, 94 |

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

Note 1: See Section 4.4 “Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)”.

Note 2: These registers and/or bits are not implemented on 28-pin devices and are read as ‘0’. Reset values are shown for 40/44-pin devices; individual unimplemented bits should be interpreted as ‘-’.

PIC18F45J10 FAMILY

5.3.5 STATUS REGISTER

The STATUS register, shown in Register 5-2, contains the arithmetic status of the ALU. As with any other SFR, it can be the operand for any instruction.

If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, the results of the instruction are not written; instead, the STATUS register is updated according to the instruction performed. Therefore, the result of an instruction with the STATUS register as its destination may be different than intended. As an example, `CLRF STATUS` will set the Z bit and leave the remaining status bits unchanged ('000u u1uu').

It is recommended 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 that do not affect Status bits, see the instruction set summaries in Table 21-2 and Table 21-3.

Note: The C and DC bits operate as the borrow and digit borrow bits, 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 of the result) 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/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 borrow, the polarity is reversed. A subtraction is executed by adding the 2's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either bit 4 or bit 3 of the source register.
- bit 0 **C:** Carry/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 borrow, the polarity is reversed. A subtraction is executed by adding the 2'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 |

5.4 Data Addressing Modes

Note: The execution of some instructions in the core PIC18 instruction set are changed when the PIC18 extended instruction set is enabled. See **Section 5.5 “Data Memory and the Extended Instruction Set”** for more information.

While the program memory can be addressed in only one way – through the program counter – information in the data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST configuration bit = 1). Its operation is discussed in greater detail in **Section 5.5.1 “Indexed Addressing with Literal Offset”**.

5.4.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all; they either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples include `SLEEP`, `RESET` and `DAW`.

Other instructions work in a similar way but require an additional explicit argument in the opcode. This is known as Literal Addressing mode because they require some literal value as an argument. Examples include `ADDLW` and `MOVLW`, which respectively, add or move a literal value to the W register. Other examples include `CALL` and `GOTO`, which include a 20-bit program memory address.

5.4.2 DIRECT ADDRESSING

Direct addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of direct addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 5.3.3 “General Purpose Register File”**) or a location in the Access Bank (**Section 5.3.2 “Access Bank”**) as the data source for the instruction.

The Access RAM bit ‘a’ determines how the address is interpreted. When ‘a’ is ‘1’, the contents of the BSR (**Section 5.3.1 “Bank Select Register (BSR)”**) are used with the address to determine the complete 12-bit address of the register. When ‘a’ is ‘0’, the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as `MOVFF`, include the entire 12-bit address (either source or destination) in their opcodes. In these cases, the BSR is ignored entirely.

The destination of the operation’s results is determined by the destination bit ‘d’. When ‘d’ is ‘1’, the results are stored back in the source register, overwriting its original contents. When ‘d’ is ‘0’, the results are stored in the W register. Instructions without the ‘d’ argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

5.4.3 INDIRECT ADDRESSING

Indirect addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations to be read or written to. Since the FSRs are themselves located in RAM as Special File Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for indirect addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 5-5.

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

```

NEXT    LFSR    FSR0, 100h ;
        CLRF   POSTINC0 ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1  ; All done with
                          ; Bank1?
        BRA    NEXT      ; NO, clear next
CONTINUE                ; YES, continue
    
```

PIC18F45J10 FAMILY

5.4.3.1 FSR Registers and the INDF Operand

At the core of indirect addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. The four upper bits of the FSRnH register are not used, so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as “virtual” registers; they are mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction’s target. The INDF operand is just a convenient way of using the pointer.

Because indirect addressing uses a full 12-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

5.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

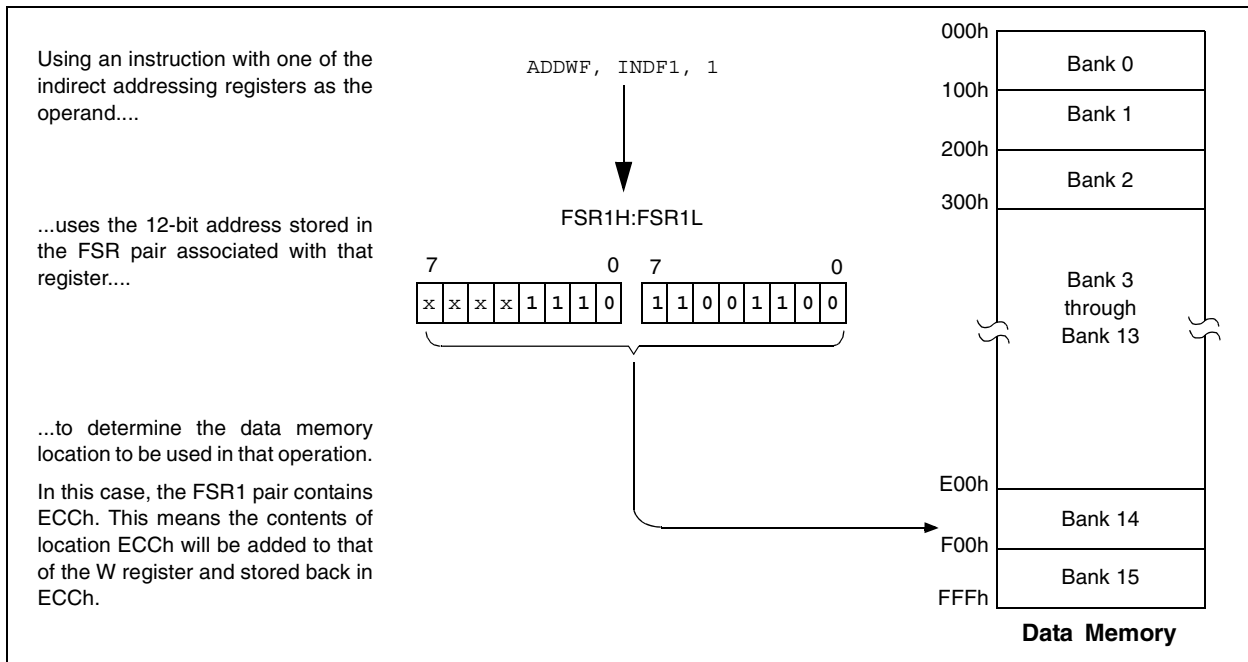
In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on its stored value. They are:

- **POSTDEC:** accesses the FSR value, then automatically decrements it by 1 afterwards
- **POSTINC:** accesses the FSR value, then automatically increments it by 1 afterwards
- **PREINC:** increments the FSR value by 1, then uses it in the operation
- **PLUSW:** adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the new value in the operation.

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value offset by that in the W register; neither value is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair; that is, roll-overs of the FSRnL register, from FFh to 00h, carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (e.g., Z, N, OV, etc.).

FIGURE 5-7: INDIRECT ADDRESSING



The PLUSW register can be used to implement a form of indexed addressing in the data memory space. By manipulating the value in the W register, users can reach addresses that are fixed offsets from pointer addresses. In some applications, this can be used to implement some powerful program control structure, such as software stacks, inside of data memory.

5.4.3.3 Operations by FSRs on FSRs

Indirect addressing operations that target other FSRs or virtual registers represent special cases. For example, using an FSR to point to one of the virtual registers will not result in successful operations. As a specific case, assume that FSR0H:FSR0L contains FE7h, the address of INDF1. Attempts to read the value of the INDF1 using INDF0 as an operand will return 00h. Attempts to write to INDF1 using INDF0 as the operand will result in a NOP.

On the other hand, using the virtual registers to write to an FSR pair may not occur as planned. In these cases, the value will be written to the FSR pair but without any incrementing or decrementing. Thus, writing to INDF2 or POSTDEC2 will write the same value to the FSR2H:FSR2L.

Since the FSRs are physical registers mapped in the SFR space, they can be manipulated through all direct operations. Users should proceed cautiously when working on these registers, particularly if their code uses indirect addressing.

Similarly, operations by indirect addressing are generally permitted on all other SFRs. Users should exercise the appropriate caution that they do not inadvertently change settings that might affect the operation of the device.

5.5 Data Memory and the Extended Instruction Set

Enabling the PIC18 extended instruction set (XINST configuration bit = 1) significantly changes certain aspects of data memory and its addressing. Specifically, the use of the Access Bank for many of the core PIC18 instructions is different; this is due to the introduction of a new addressing mode for the data memory space.

What does not change is just as important. The size of the data memory space is unchanged, as well as its linear addressing. The SFR map remains the same. Core PIC18 instructions can still operate in both Direct and Indirect Addressing mode; inherent and literal instructions do not change at all. Indirect addressing with FSR0 and FSR1 also remain unchanged.

5.5.1 INDEXED ADDRESSING WITH LITERAL OFFSET

Enabling the PIC18 extended instruction set changes the behavior of indirect addressing using the FSR2 register pair within Access RAM. Under the proper conditions, instructions that use the Access Bank – that is, most bit-oriented and byte-oriented instructions – can invoke a form of indexed addressing using an offset specified in the instruction. This special addressing mode is known as Indexed Addressing with Literal Offset, or Indexed Literal Offset mode.

When using the extended instruction set, this addressing mode requires the following:

- The use of the Access Bank is forced ('a' = 0) and
- The file address argument is less than or equal to 5Fh.

Under these conditions, the file address of the instruction is not interpreted as the lower byte of an address (used with the BSR in direct addressing), or as an 8-bit address in the Access Bank. Instead, the value is interpreted as an offset value to an address pointer, specified by FSR2. The offset and the contents of FSR2 are added to obtain the target address of the operation.

5.5.2 INSTRUCTIONS AFFECTED BY INDEXED LITERAL OFFSET MODE

Any of the core PIC18 instructions that can use direct addressing are potentially affected by the Indexed Literal Offset Addressing mode. This includes all byte-oriented and bit-oriented instructions, or almost one-half of the standard PIC18 instruction set. Instructions that only use Inherent or Literal Addressing modes are unaffected.

Additionally, byte-oriented and bit-oriented instructions are not affected if they do not use the Access Bank (Access RAM bit is '1'), or include a file address of 60h or above. Instructions meeting these criteria will continue to execute as before. A comparison of the different possible addressing modes when the extended instruction set is enabled is shown in Figure 5-8.

Those who desire to use byte-oriented or bit-oriented instructions in the Indexed Literal Offset mode should note the changes to assembler syntax for this mode. This is described in more detail in **Section 21.2.1 “Extended Instruction Syntax”**.

PIC18F45J10 FAMILY

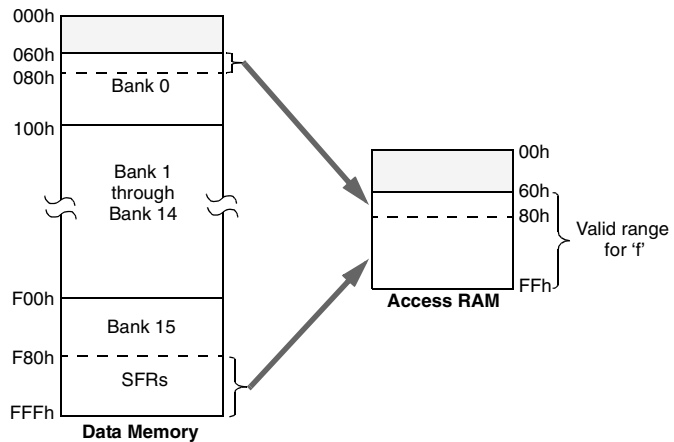
FIGURE 5-8: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)

Example Instruction: `ADDWF, f, d, a` (Opcode: `0010 01da ffff ffff`)

When 'a' = 0 and f ≥ 60h:

The instruction executes in Direct Forced mode. 'f' is interpreted as a location in the Access RAM between 060h and 0FFh. This is the same as locations 060h to 07Fh (Bank 0) and F80h to FFFh (Bank 15) of data memory.

Locations below 60h are not available in this addressing mode.



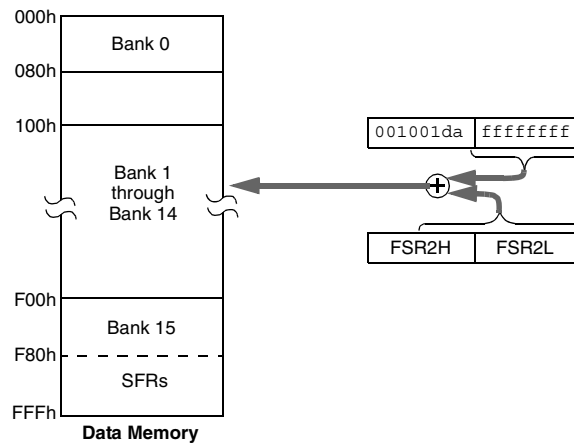
When 'a' = 0 and f ≤ 5Fh:

The instruction executes in Indexed Literal Offset mode. 'f' is interpreted as an offset to the address value in FSR2. The two are added together to obtain the address of the target register for the instruction. The address can be anywhere in the data memory space.

Note that in this mode, the correct syntax is now:

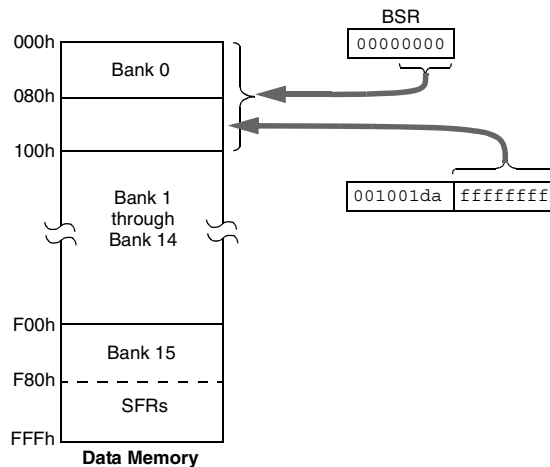
`ADDWF [k], d`

where 'k' is the same as 'f'.



When 'a' = 1 (all values of f):

The instruction executes in Direct mode (also known as Direct Long mode). 'f' is interpreted as a location in one of the 16 banks of the data memory space. The bank is designated by the Bank Select Register (BSR). The address can be in any implemented bank in the data memory space.



5.5.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

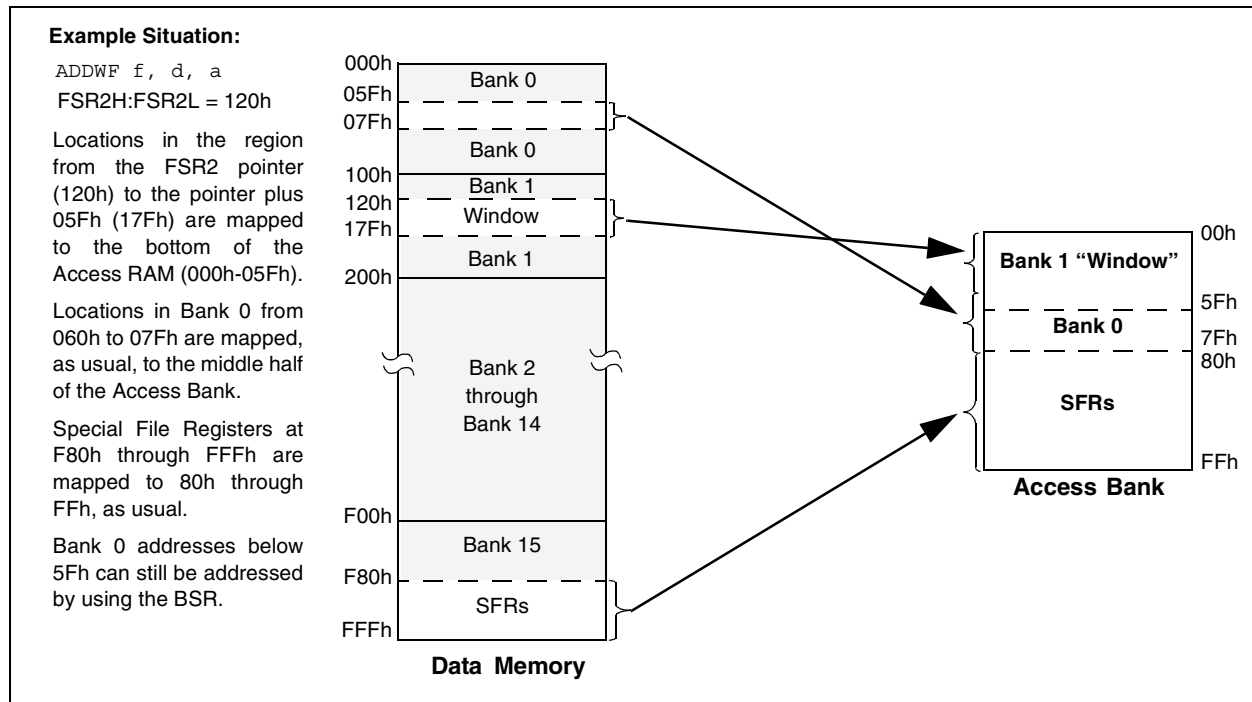
The use of Indexed Literal Offset Addressing mode effectively changes how the first 96 locations of Access RAM (00h to 5Fh) are mapped. Rather than containing just the contents of the bottom half of Bank 0, this mode maps the contents from Bank 0 and a user-defined “window” that can be located anywhere in the data memory space. The value of FSR2 establishes the lower boundary of the addresses mapped into the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described (see **Section 5.3.2 “Access Bank”**). An example of Access Bank remapping in this addressing mode is shown in Figure 5-9.

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is ‘1’) will continue to use direct addressing as before.

5.6 PIC18 Instruction Execution and the Extended Instruction Set

Enabling the extended instruction set adds eight additional commands to the existing PIC18 instruction set. These instructions are executed as described in **Section 21.2 “Extended Instruction Set”**.

FIGURE 5-9: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING



PIC18F45J10 FAMILY

NOTES:

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 64 bytes at a time. Program memory is erased in blocks of 1024 bytes at a time. A bulk erase operation may not be issued from user code.

Writing or erasing program memory will cease instruction fetches 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)

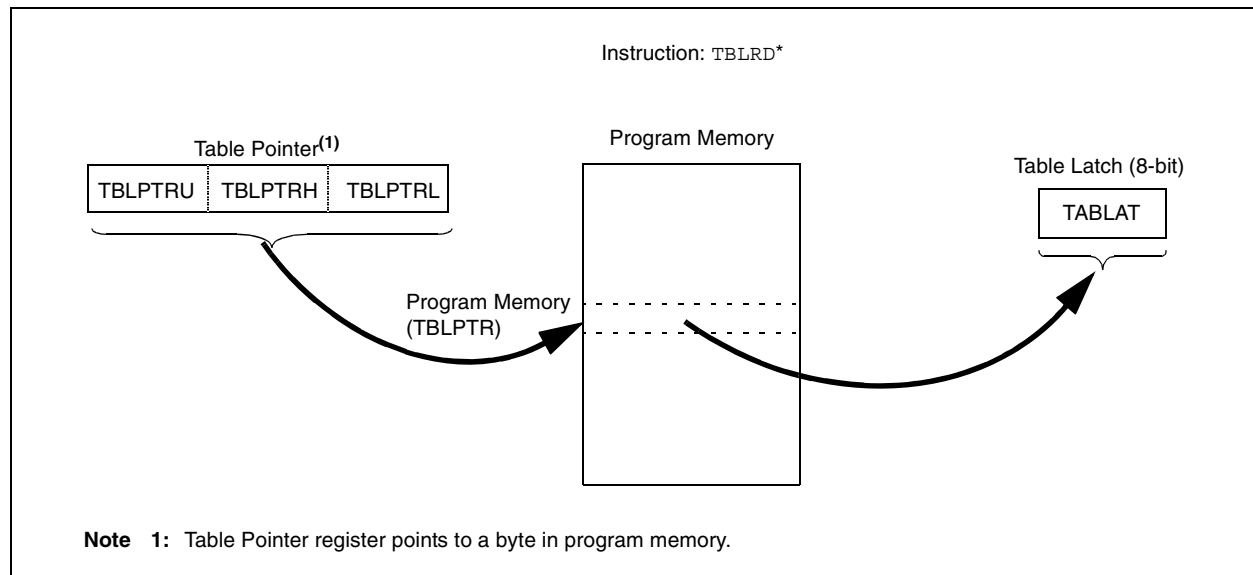
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into 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 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.

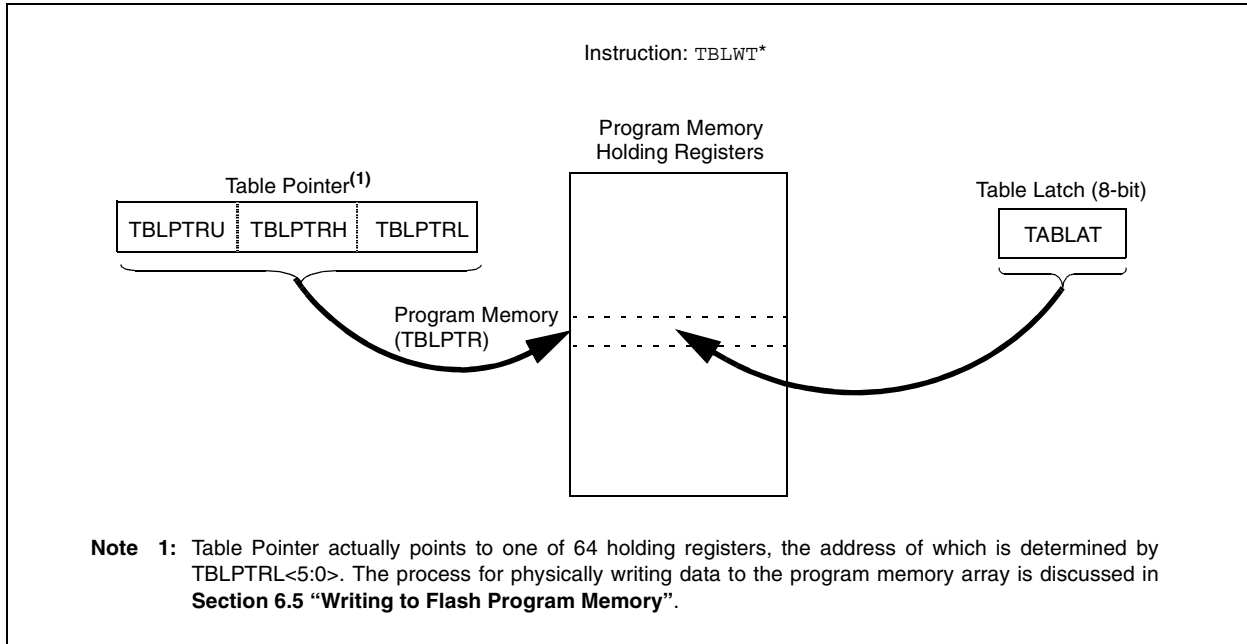
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions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

FIGURE 6-1: TABLE READ OPERATION



PIC18F45J10 FAMILY

FIGURE 6-2: TABLE WRITE OPERATION



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

The EECON1 register (Register 6-1) is the control register for memory accesses. The EECON2 register is not a physical register; it is used exclusively in the memory write and erase sequences. Reading EECON2 will read all '0's.

The FREE bit, when set, will allow a program memory erase operation. When FREE is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set in hardware when the WR bit is set and cleared when the internal programming timer expires and the write operation is complete.

Note: During normal operation, the WRERR is read as '1'. This can indicate that a write operation was prematurely terminated by a Reset, or a write operation was attempted improperly.

The WR control bit initiates write operations. The bit cannot be cleared, only set, in software; it is cleared in hardware at the completion of the write operation.

PIC18F45J10 FAMILY

REGISTER 6-1: EECON1: DATA EEPROM CONTROL REGISTER 1

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

- bit 7-5 **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:** Flash Program/Data EEPROM Error Flag bit
 1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation, or an improper write attempt)
 0 = The write operation completed
- bit 2 **WREN:** Flash Program/Data EEPROM Write Enable bit
 1 = Allows write cycles to Flash program/data EEPROM
 0 = Inhibits write cycles to Flash program/data EEPROM
- 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 to the EEPROM is complete
- bit 0 **Unimplemented:** Read as '0'

Legend:

| | | |
|---|------------------------------------|--|
| R = Readable bit | W = Writable bit | |
| S = Bit can be set by software, but not cleared | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared x = Bit is unknown |

PIC18F45J10 FAMILY

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. The 22nd bit allows access to the device ID, the user ID and the configuration bits.

The Table Pointer register, 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 TBLPTR determine which byte is read from program memory into TABLAT.

When a TBLWT is executed, the six LSBs of the Table Pointer register (TBLPTR<5:0>) determine which of the 64 program memory holding registers is written to. When the timed write to program memory begins (via the WR bit), the 16 MSBs of the TBLPTR (TBLPTR<21:6>) determine which program memory block of 64 bytes is written to. For more detail, see **Section 6.5 “Writing to Flash Program Memory”**.

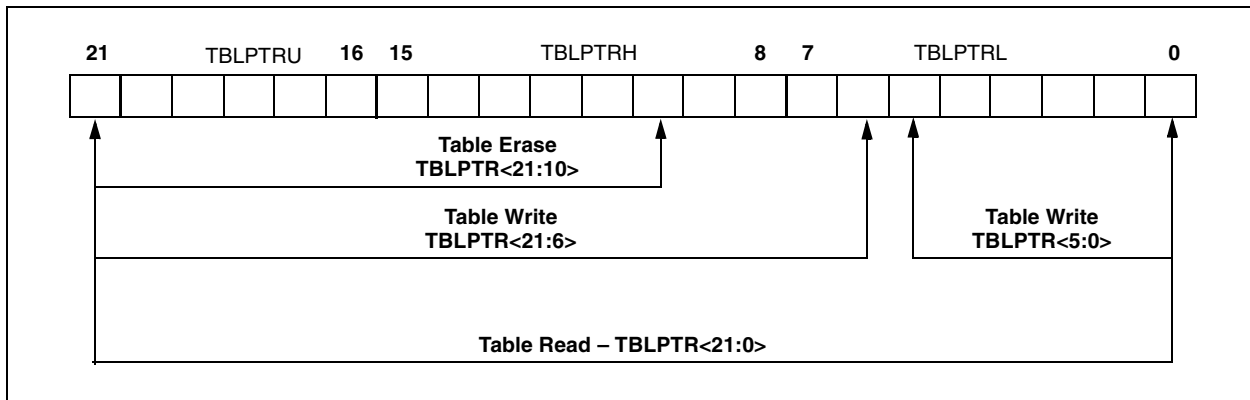
When an erase of program memory is executed, the 7 MSBs of the Table Pointer register (TBLPTR<21:10>) point to the 1024-byte block that will be erased. The Least Significant bits (TBLPTR<9:0>) are ignored.

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* TBLWT* | TBLPTR is not modified |
| TBLRD*+ TBLWT*+ | TBLPTR is incremented after the read/write |
| TBLRD*- TBLWT*- | TBLPTR is decremented after the read/write |
| TBLRD+* TBLWT+* | TBLPTR is incremented before the read/write |

FIGURE 6-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



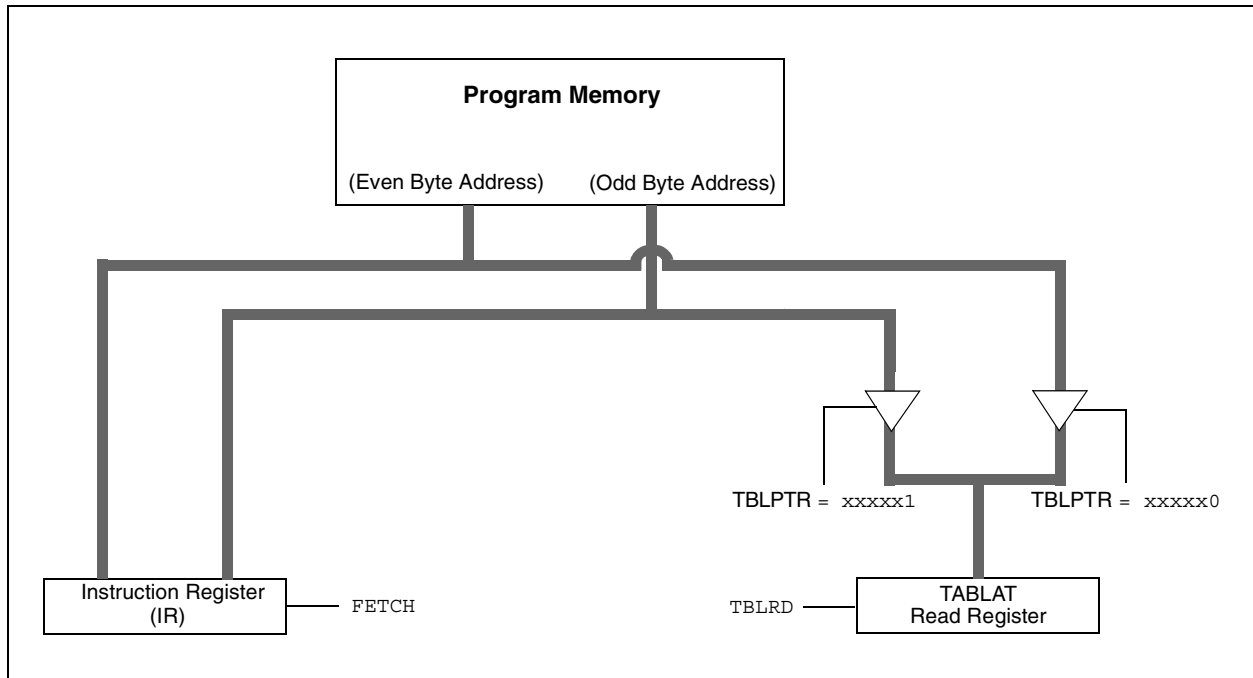
6.3 Reading the Flash Program Memory

The `TBLRD` instruction is used to retrieve data from program memory and places 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 `TBLRD` 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
    MOVF    TABLAT, W                ; get data
    MOVWF   WORD_EVEN
    TBLRD*+                          ; read into TABLAT and increment
    MOVFW   TABLAT, W                ; get data
    MOVF    WORD_ODD

```

PIC18F45J10 FAMILY

6.4 Erasing Flash Program Memory

The minimum erase block is 1024 bytes. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the microcontroller itself, a block of 1024 bytes of program memory is erased. The Most Significant 7 bits of the TBLPTR<21:10> point to the block being erased. TBLPTR<9:0> are ignored.

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

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

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 register with address of row being erased.
2. Set the EECON1 register for the erase operation:
 - set WREN bit to enable writes;
 - set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh 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. 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, WREN | | ; enable write to memory |
| | BSF | EECON1, FREE | | ; enable Row Erase operation |
| | BCF | INTCON, GIE | | ; disable interrupts |
| Required Sequence | MOVLW | 55h | | |
| | MOVWF | EECON2 | | ; write 55h |
| | MOVLW | 0AAh | | |
| | MOVWF | EECON2 | | ; write 0AAh |
| | BSF | EECON1, WR | | ; start erase (CPU stall) |
| | BSF | INTCON, GIE | | ; re-enable interrupts |

6.5 Writing to Flash Program Memory

The minimum programming block is 32 words or 64 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 64 holding registers used by the table writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction may need to be executed 64 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 the 64 holding registers, the EECON1 register must be written to in order to start the programming operation with a long write.

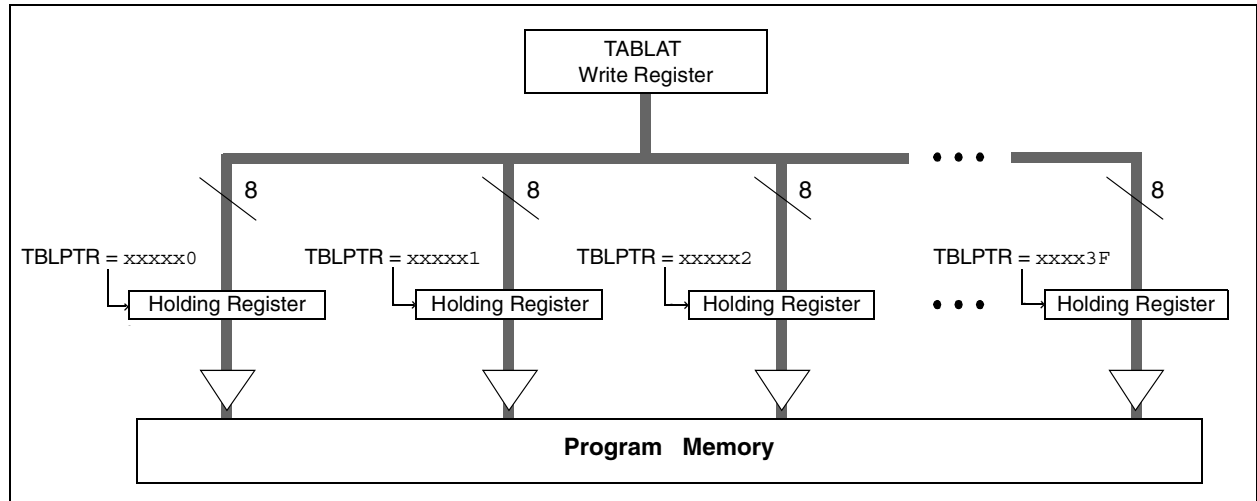
The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The EEPROM on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device.

Note: Unlike previous devices, the PIC18F45J10 family of devices does not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence.

In order to maintain the endurance of the cells, each Flash byte should not be programmed more than twice between erase operations. Either a bulk or row erase of the target row is required before attempting to modify the contents a third time.

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. If the section of program memory to be written to has been programmed previously, then the memory will need to be erased before the write occurs (see 6.4.1 “Flash Program Memory Erase Sequence”).
2. Write the 64 bytes into the holding registers with auto-increment.
3. Set the EECON1 register for the write operation:
 - set WREN to enable byte writes.
4. Disable interrupts.

5. Write 55h to EECON2.
6. Write 0AAh to EECON2.
7. Set the WR bit. This will begin the write cycle.
8. The CPU will stall for duration of the write (about 2 ms using internal timer).
9. Re-enable interrupts.
10. Verify the memory (table read).

An example of the required code is shown in Example 6-3.

Note: Before setting the WR bit, the Table Pointer address needs to be within the intended address range of the 64 bytes in the holding register.

PIC18F45J10 FAMILY

EXAMPLE 6-3: WRITING TO FLASH PROGRAM MEMORY

| | | | | |
|--------------------------|--------|---------------------|--|--|
| | 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_BLOCK | | | | |
| | 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 |
| | MOVLW | 0AAh | | |
| | MOVWF | EECON2 | | ; write 0AAh |
| | BSF | EECON1, WR | | ; start erase (CPU stall) |
| | BSF | INTCON, GIE | | ; re-enable interrupts |
| | MOVLW | D'16' | | |
| | MOVWF | WRITE_COUNTER | | ; Need to write 16 blocks of 64 to write ; one erase block of 1024 |
| RESTART_BUFFER | | | | |
| | MOVLW | D'64' | | |
| | MOVWF | COUNTER | | |
| | MOVLW | BUFFER_ADDR_HIGH | | ; point to buffer |
| | MOVWF | FSR0H | | |
| | MOVLW | BUFFER_ADDR_LOW | | |
| | MOVWF | FSR0L | | |
| FILL_BUFFER | | | | |
| | ... | | | ; read the new data from I2C, SPI, ; PSP, USART, etc. |
| WRITE_BUFFER | | | | |
| | MOVLW | D'64 | | ; number of bytes in holding register |
| | MOVWF | COUNTER | | |
| WRITE_BYTE_TO_HREGS | | | | |
| | MOVFF | POSTINC0, WREG | | ; get low byte of buffer data |
| | MOVWF | TABLAT | | ; present data to table latch |
| | TBLWT* | | | ; write data, perform a short write ; to internal TBLWT holding register. |
| | DECFSZ | COUNTER | | ; loop until buffers are full |
| | BRA | WRITE_WORD_TO_HREGS | | |
| PROGRAM_MEMORY | | | | |
| | BSF | EECON1, WREN | | ; enable write to memory |
| | BCF | INTCON, GIE | | ; disable interrupts |
| | MOVLW | 55h | | |
| Required Sequence | MOVWF | EECON2 | | ; write 55h |
| | MOVLW | 0AAh | | |
| | MOVWF | EECON2 | | ; write 0AAh |
| | BSF | EECON1, WR | | ; start program (CPU stall) |
| | BSF | INTCON, GIE | | ; re-enable interrupts |
| | BCF | EECON1, WREN | | ; disable write to memory |
| | DECFSZ | WRITE_COUNTER | | ; done with one write cycle |
| | BRA | RESTART_BUFFER | | ; if not done replacing the erase block |

PIC18F45J10 FAMILY

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. If the write operation is interrupted by a MCLR Reset or a WDT Time-out Reset during normal operation, the user can check the WRERR bit and rewrite the location(s) as needed.

6.5.4 PROTECTION AGAINST SPURIOUS WRITES

To protect against spurious writes to Flash program memory, the write initiate sequence must also be followed. See **Section 20.0 “Special Features of the CPU”** for more detail.

6.6 Flash Program Operation During Code Protection

See **Section 20.6 “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 | Reset Values on page |
|---------|---|-----------|--------|---|--------|--------|--------|--------|----------------------|
| TBLPTRU | — | — | bit 21 | Program Memory Table Pointer Upper Byte (TBLPTR<20:16>) | | | | | 43 |
| TBPLTRH | Program Memory Table Pointer High Byte (TBLPTR<15:8>) | | | | | | | | 43 |
| TBLPTRL | Program Memory Table Pointer Low Byte (TBLPTR<7:0>) | | | | | | | | 43 |
| TABLAT | Program Memory Table Latch | | | | | | | | 43 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 45 |
| EECON1 | — | — | — | FREE | WRERR | WREN | WR | — | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used during Flash/EEPROM access.

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

7.0 8 x 8 HARDWARE MULTIPLIER

7.1 Introduction

All PIC18 devices include an 8 x 8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 7-1.

7.2 Operation

Example 7-1 shows the instruction sequence for an 8 x 8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 7-2 shows the sequence to do an 8 x 8 signed multiplication. 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 7-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

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

EXAMPLE 7-2: 8 x 8 SIGNED MULTIPLY ROUTINE

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

BTFSC ARG2, SB ; Test Sign Bit
SUBWF PRODH, F ; PRODH = PRODH
; - ARG1

MOVWF ARG2, W
BTFSC ARG1, SB ; Test Sign Bit
SUBWF PRODH, F ; PRODH = PRODH
; - ARG2
```

TABLE 7-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

| 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 μ s | 27.6 μ s | 69 μ s |
| | Hardware multiply | 1 | 1 | 100 ns | 400 ns | 1 μ s |
| 8 x 8 signed | Without hardware multiply | 33 | 91 | 9.1 μ s | 36.4 μ s | 91 μ s |
| | Hardware multiply | 6 | 6 | 600 ns | 2.4 μ s | 6 μ s |
| 16 x 16 unsigned | Without hardware multiply | 21 | 242 | 24.2 μ s | 96.8 μ s | 242 μ s |
| | Hardware multiply | 28 | 28 | 2.8 μ s | 11.2 μ s | 28 μ s |
| 16 x 16 signed | Without hardware multiply | 52 | 254 | 25.4 μ s | 102.6 μ s | 254 μ s |
| | Hardware multiply | 35 | 40 | 4.0 μ s | 16.0 μ s | 40 μ s |

PIC18F45J10 FAMILY

Example 7-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 7-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 7-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

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

EXAMPLE 7-3: 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 7-4 shows the sequence to do a 16 x 16 signed multiply. Equation 7-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 7-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$\begin{aligned} \text{RES3:RES0} &= \text{ARG1H:ARG1L} \cdot \text{ARG2H:ARG2L} \\ &= (\text{ARG1H} \cdot \text{ARG2H} \cdot 2^{16}) + \\ &\quad (\text{ARG1H} \cdot \text{ARG2L} \cdot 2^8) + \\ &\quad (\text{ARG1L} \cdot \text{ARG2H} \cdot 2^8) + \\ &\quad (\text{ARG1L} \cdot \text{ARG2L}) + \\ &\quad (-1 \cdot \text{ARG2H} < 7 > \cdot \text{ARG1H:ARG1L} \cdot 2^{16}) + \\ &\quad (-1 \cdot \text{ARG1H} < 7 > \cdot \text{ARG2H:ARG2L} \cdot 2^{16}) \end{aligned}$$

EXAMPLE 7-4: 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      ;
;

BTFS ARG2H, 7        ; ARG2H:ARG2L neg?
BRA SIGN_ARG1        ; no, check ARG1
MOVF ARG1L, W
SUBWF RES2
MOVF ARG1H, W
SUBWFB RES3
;

SIGN_ARG1
BTFS ARG1H, 7        ; ARG1H:ARG1L neg?
BRA CONT_CODE        ; no, done
MOVF ARG2L, W
SUBWF RES2
MOVF ARG2H, W
SUBWFB RES3
;

CONT_CODE
:

```


8.0 INTERRUPTS

Members of the PIC18F45J10 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high priority level or a low priority level. The high priority interrupt vector is at 0008h and the low priority interrupt vector is at 0018h. High priority interrupt events will interrupt any low priority interrupts that may be in progress.

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

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3
- PIE1, PIE2, PIE3
- IPR1, IPR2, IPR3

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, interrupt sources have three bits to control their operation. They 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

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 0008h or 0018h, 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 0008h 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 (0008h or 0018h). 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.

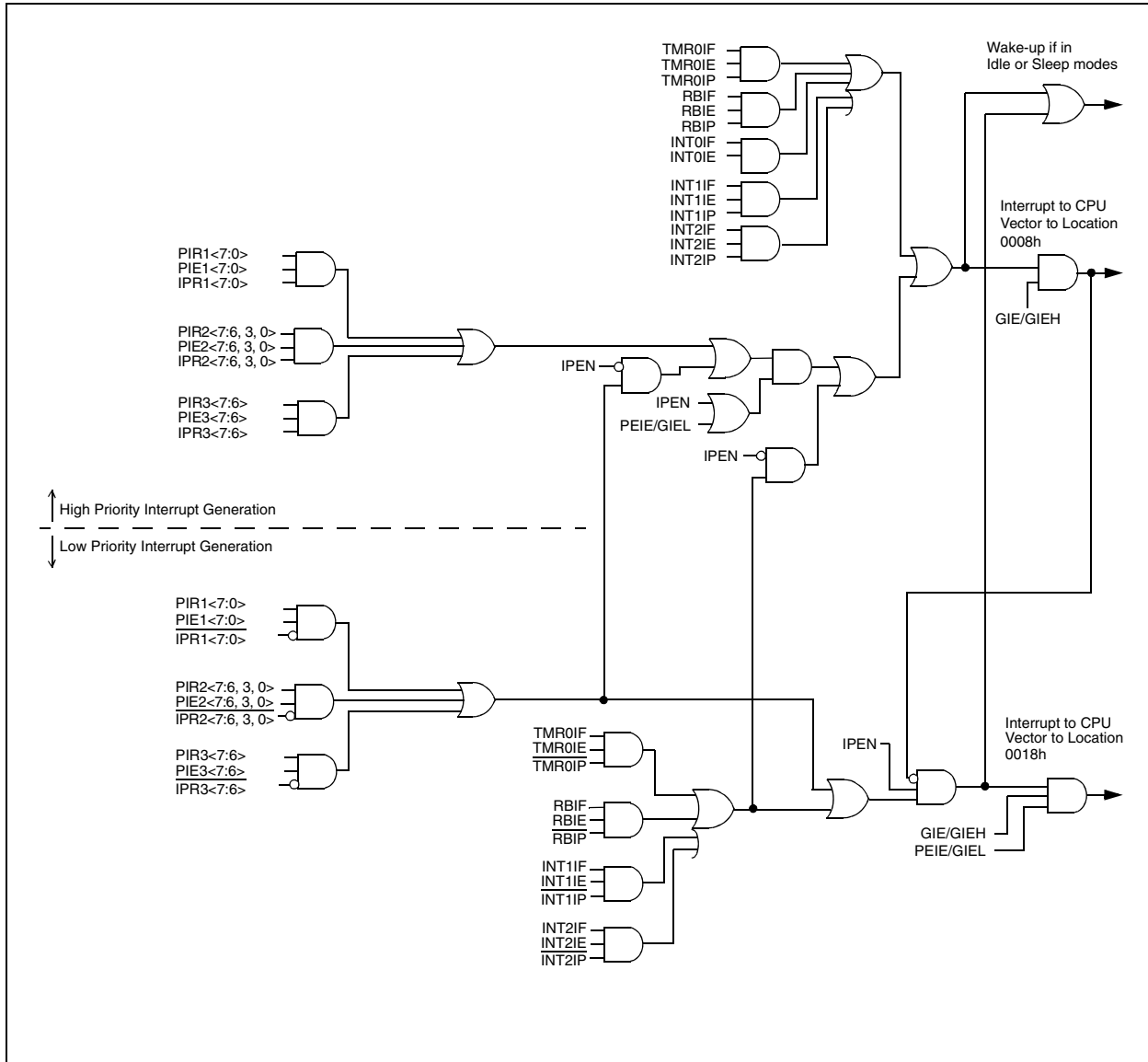
The “return from interrupt” instruction, RETFIE, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used) which re-enables interrupts.

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.

PIC18F45J10 FAMILY

FIGURE 8-1: PIC18F24J10/25J10/44J10/45J10 INTERRUPT LOGIC



PIC18F45J10 FAMILY

8.1 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 interrupt 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 8-1: INTCON: INTERRUPT 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-x |
|----------|-----------|--------|--------|-------|--------|--------|-------|
| GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF |
| | | | | | | | bit 0 |
| bit 7 | | | | | | | |

- 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 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

PIC18F45J10 FAMILY

REGISTER 8-2: INTCON2: INTERRUPT CONTROL REGISTER 2

| | | | | | | | |
|--------------------------|---------|---------|---------|-----|--------|-----|-------|
| 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 Interrupt 0 Edge Select bit
 1 = Interrupt on rising edge
 0 = Interrupt on falling edge
- bit 5 **INTEDG1**: External Interrupt 1 Edge Select bit
 1 = Interrupt on rising edge
 0 = Interrupt on falling edge
- bit 4 **INTEDG2**: External Interrupt 2 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 interrupt enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

PIC18F45J10 FAMILY

REGISTER 8-3: INTCON3: INTERRUPT CONTROL REGISTER 3

| 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 interrupt enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

PIC18F45J10 FAMILY

8.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Request (Flag) registers (PIR1, PIR2, PIR3).

Note 1: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt 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 8-4: 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 | SSP1IF | CCP1IF | TMR2IF | TMR1IF |
| bit 7 | | | | | | | | bit 0 |

- bit 7 **PSPIF:** Parallel Slave Port Read/Write Interrupt Flag bit⁽¹⁾
 1 = A read or a write operation has taken place (must be cleared in software)
 0 = No read or write has occurred
Note: This bit is not implemented on 28-pin devices and should be read as '0'.
- 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:** EUSART Receive Interrupt Flag bit
 1 = The EUSART receive buffer, RCREG, is full (cleared when RCREG is read)
 0 = The EUSART receive buffer is empty
- bit 4 **TXIF:** EUSART Transmit Interrupt Flag bit
 1 = The EUSART transmit buffer, TXREG, is empty (cleared when TXREG is written)
 0 = The EUSART transmit buffer is full
- bit 3 **SSP1IF:** Master Synchronous Serial Port 1 Interrupt Flag bit
 1 = The transmission/reception is complete (must be cleared in software)
 0 = Waiting to transmit/receive
- bit 2 **CCP1IF:** ECCP1/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 |

PIC18F45J10 FAMILY

REGISTER 8-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

| R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 | U-0 | U-0 | R/W-0 |
|--------|-------|-----|-----|--------|-----|-----|--------|
| OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF |
| bit 7 | | | | bit 0 | | | |

- bit 7 **OSCFIF:** Oscillator Fail Interrupt Flag bit
 1 = Device oscillator failed, clock input has changed to INTOSC (must be cleared in software)
 0 = Device 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 5-4 **Unimplemented:** Read as '0'
- bit 3 **BCL1IF:** Bus Collision Interrupt Flag bit (MSSP1 module)
 1 = A bus collision occurred (must be cleared in software)
 0 = No bus collision occurred
- bit 2-1 **Unimplemented:** Read as '0'
- bit 0 **CCP2IF:** CCP2 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

REGISTER 8-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

| R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|--------|--------|-----|-----|-------|-----|-----|-----|
| SSP2IF | BCL2IF | — | — | — | — | — | — |
| bit 7 | | | | bit 0 | | | |

- bit 7 **SSP2IF:** Master Synchronous Serial Port 2 Interrupt Flag bit
 1 = The transmission/reception is complete (must be cleared in software)
 0 = Waiting to transmit/receive
- bit 6 **BCL2IF:** Bus Collision Interrupt Flag bit (MSSP2 module)
 1 = A bus collision occurred (must be cleared in software)
 0 = No bus collision occurred
- bit 5-0 **Unimplemented:** Read as '0'

Legend:

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

PIC18F45J10 FAMILY

8.3 PIE Registers

The PIE registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Enable registers (PIE1, PIE2, PIE3). When IPEN = 0, the PEIE bit must be set to enable any of these peripheral interrupts.

REGISTER 8-7: 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 | SSP1IE | CCP1IE | TMR2IE | TMR1IE |
| bit 7 | | | | | | | bit 0 |

bit 7 **PSPIE:** Parallel Slave Port Read/Write Interrupt Enable bit⁽¹⁾

1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

Note: This bit is not implemented on 28-pin devices and should be read as '0'.

bit 6 **ADIE:** A/D Converter Interrupt Enable bit

1 = Enables the A/D interrupt
0 = Disables the A/D interrupt

bit 5 **RCIE:** EUSART Receive Interrupt Enable bit

1 = Enables the EUSART receive interrupt
0 = Disables the EUSART receive interrupt

bit 4 **TXIE:** EUSART Transmit Interrupt Enable bit

1 = Enables the EUSART transmit interrupt
0 = Disables the EUSART transmit interrupt

bit 3 **SSP1IE:** Master Synchronous Serial Port 1 Interrupt Enable bit

1 = Enables the MSSP1 interrupt
0 = Disables the MSSP1 interrupt

bit 2 **CCP1IE:** ECCP1/CCP1 Interrupt Enable bit

1 = Enables the ECCP1/CCP1 interrupt
0 = Disables the ECCP1/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 |

PIC18F45J10 FAMILY

REGISTER 8-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

| R/W-0 | R/W-0 | U-0 | U-0 | R/W-0 | U-0 | U-0 | R/W-0 |
|--------|-------|-----|-----|--------|-----|-----|--------|
| OSCFIE | CMIE | — | — | BCL1IE | — | — | 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 5-4 **Unimplemented:** Read as '0'

bit 3 **BCL1IE:** Bus Collision Interrupt Enable bit (MSSP1 module)

1 = Enabled
0 = Disabled

bit 2-1 **Unimplemented:** Read as '0'

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

REGISTER 8-9: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

| R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|--------|--------|-----|-----|-------|-----|-----|-----|
| SSP2IE | BCL2IE | — | — | — | — | — | — |
| bit 7 | | | | bit 0 | | | |

bit 7 **SSP2IE:** Master Synchronous Serial Port 2 Interrupt Enable bit

1 = Enabled
0 = Disabled

bit 6 **BCL2IE:** Bus Collision Interrupt Enable bit (MSSP2 module)

1 = Enabled
0 = Disabled

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

Legend:

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

PIC18F45J10 FAMILY

8.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Priority registers (IPR1, IPR2, IPR3). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 8-10: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|----------------------|-------|-------|-------|--------|--------|--------|--------|
| PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP |
| bit 7 | | | | | | | bit 0 |

- bit 7 **PSPIP:** Parallel Slave Port Read/Write Interrupt Priority bit⁽¹⁾
1 = High priority
0 = Low priority
Note: This bit is not implemented on 28-pin devices and should be read as '0'.
- bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority
- bit 5 **RCIP:** EUSART Receive Interrupt Priority bit
1 = High priority
0 = Low priority
- bit 4 **TXIP:** EUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority
- bit 3 **SSP1IP:** Master Synchronous Serial Port 1 Interrupt Priority bit
1 = High priority
0 = Low priority
- bit 2 **CCP1IP:** ECCP1/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 |

PIC18F45J10 FAMILY

REGISTER 8-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

| R/W-1 | R/W1 | U-0 | U-0 | R/W-1 | U-0 | U-0 | R/W-1 |
|--------|------|-----|-----|--------|-----|-----|--------|
| OSCFIP | CMIP | — | — | BCL1IP | — | — | 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 5-4 **Unimplemented:** Read as '0'

bit 3 **BCL1IP:** Bus Collision Interrupt Priority bit (MSSP1 module)

1 = High priority

0 = Low priority

bit 2-1 **Unimplemented:** Read as '0'

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

REGISTER 8-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

| R/W-1 | R/W-1 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|--------|--------|-----|-----|-------|-----|-----|-----|
| SSP2IP | BCL2IP | — | — | — | — | — | — |
| bit 7 | | | | bit 0 | | | |

bit 7 **SSP2IP:** Master Synchronous Serial Port 2 Interrupt Priority bit

1 = High priority

0 = Low priority

bit 6 **BCL2IP:** Bus Collision Interrupt Priority bit (MSSP2 module)

1 = High priority

0 = Low priority

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

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

PIC18F45J10 FAMILY

8.5 RCON Register

The RCON register contains bits used to determine the cause of the last Reset or wake-up from Idle or Sleep modes. RCON also contains the bit that enables interrupt priorities (IPEN).

REGISTER 8-13: RCON: RESET CONTROL 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
For details of bit operation, see Register 4-1.
- bit 3 **\overline{TO} :** Watchdog Timer Time-out Flag bit
For details of bit operation, see Register 4-1.
- bit 2 **\overline{PD} :** Power-Down Detection Flag bit
For details of bit operation, see Register 4-1.
- bit 1 **\overline{POR} :** Power-on Reset Status bit
For details of bit operation, see Register 4-1.
- bit 0 **\overline{BOR} :** Brown-out Reset Status bit
For details of bit operation, see Register 4-1.

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 |

8.6 INTn Pin Interrupts

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge-triggered. If the corresponding INTEDGx bit in the INTCON2 register is set (= 1), the interrupt is triggered by a rising edge; if the bit is clear, the trigger is on the falling edge. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit, INTxIF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxIE. Flag bit, INTxIF, 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 INTxIE was set prior to going into the 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.

8.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow in the TMR0 register (FFh → 00h) will set flag bit, TMR0IF. In 16-bit mode, an overflow in the TMR0H:TMR0L register pair (FFFFh → 0000h) will set 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 10.0 “Timer0 Module”** for further details on the Timer0 module.

8.8 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>).

8.9 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 “Data Memory Organization”**), 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 8-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 8-1: 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_TEMP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR          ; Restore BSR
MOVFF    W_TEMP, W              ; Restore WREG
MOVFF    STATUS_TEMP, STATUS    ; Restore STATUS
    
```

PIC18F45J10 FAMILY

NOTES:

9.0 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.

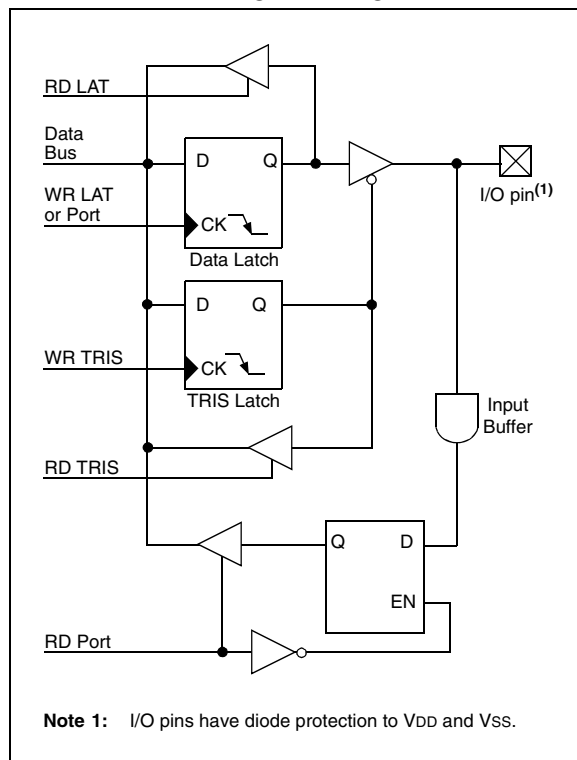
Each port has three registers for its operation. These registers are:

- TRIS register (data direction register)
- Port register (reads the levels on the pins of the device)
- LAT register (output 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 9-1.

FIGURE 9-1: GENERIC I/O PORT OPERATION



9.1 I/O Port Pin Capabilities

When developing an application, the capabilities of the port pins must be considered. Outputs on some pins have higher output drive strength than others. Similarly, some pins can tolerate higher than VDD input levels.

9.1.1 PIN OUTPUT DRIVE

The output pin drive strengths vary for groups of pins intended to meet the needs for a variety of applications. PORTB and PORTC are designed to drive higher loads, such as LEDs. All other ports are designed for small loads, typically indication only. Table 9-1 summarizes the output capabilities. Refer to **Section 23.0 “Electrical Characteristics”** for more details.

TABLE 9-1: OUTPUT DRIVE LEVELS

| Port | Drive | Description |
|-------|---------|---------------------------------------|
| PORTA | Minimum | Intended for indication. |
| PORTD | | |
| PORTE | | |
| PORTB | High | Suitable for direct LED drive levels. |
| PORTC | | |

9.1.2 INPUT PINS AND VOLTAGE CONSIDERATIONS

The voltage tolerance of pins used as device inputs is dependent on the pin’s input function. Pins that are used as digital only inputs are able to handle DC voltages up to 5.5V; a level typical for digital logic circuits. In contrast, pins that also have analog input functions of any kind can only tolerate voltages up to VDD. Voltage excursions beyond VDD on these pins should be avoided. Table 9-2 summarizes the input capabilities. Refer to **Section 23.0 “Electrical Characteristics”** for more details.

TABLE 9-2: INPUT VOLTAGE LEVELS

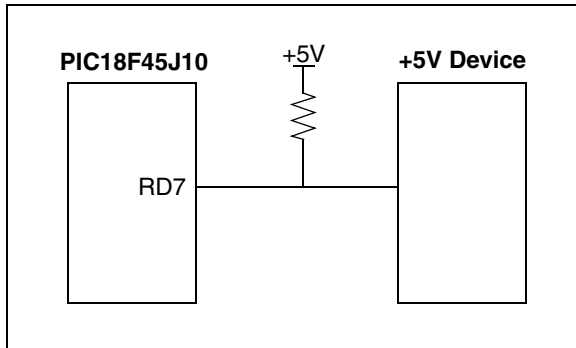
| Port or Pin | Tolerated Input | Description |
|-------------|-----------------|---|
| PORTA<5:0> | VDD | Only VDD input levels tolerated. |
| PORTB<5:0> | | |
| PORTC<1:0> | | |
| PORTE<2:0> | 5.5V | Tolerates input levels above VDD, useful for most standard logic. |
| PORTB<7:6> | | |
| PORTC<7:2> | | |
| PORTD<7:0> | | |

PIC18F45J10 FAMILY

9.1.3 INTERFACING TO A 5V SYSTEM

Though the VDDMAX of the PIC18F45J10 family is 3.6V, these devices are still capable of interfacing with 5V systems, even if the VIH of the target system is above 3.6V. This is accomplished by adding a pull-up resistor to the port pin (Figure 9-2), clearing the LAT bit for that pin and manipulating the corresponding TRIS bit (Figure 9-1) to either allow the line to be pulled high or to drive the pin low. Only port pins that are tolerant of voltages up to 5.5V can be used for this type of interface (refer to **Section 9.1.2 “Input Pins and Voltage Considerations”**).

FIGURE 9-2: +5V SYSTEM HARDWARE INTERFACE



EXAMPLE 9-1: COMMUNICATING WITH THE +5V SYSTEM

```
BCF LATD, 7 ; set up LAT register so
                ; changing TRIS bit will
                ; drive line low
BCF TRISD, 7 ; send a 0 to the 5V system
BCF TRISD, 7 ; send a 1 to the 5V system
```

9.2 PORTA, TRISA and LATA Registers

PORTA is a 5-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 (LATA) register is also memory mapped. Read-modify-write operations on the LATA register read and write the latched output value for PORTA.

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 or setting the control bits in the ADCON1 register (A/D Control Register 1).

Pins RA0 and RA3 may also be used as comparator inputs and RA5 may be used as the C2 comparator output by setting the appropriate bits in the CMCON register. To use RA3:RA0 as digital inputs, it is also necessary to turn off the comparators.

Note: On a Power-on Reset, RA5 and RA3:RA0 are configured as analog inputs and read as '0'.

All PORTA pins have TTL input levels and full CMOS output drivers.

The TRISA register controls the direction of the PORTA 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 9-2: INITIALIZING PORTA

```
CLRF PORTA ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF LATA ; Alternate method
                ; to clear output
                ; data latches
MOVLW 07h ; Configure A/D
MOVWF ADCON1 ; for digital inputs
MOVWF 07h ; Configure comparators
MOVWF CMCON ; for digital input
MOVLW 0CFh ; Value used to
                ; initialize data
                ; direction
MOVWF TRISA ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```


PIC18F45J10 FAMILY

TABLE 9-3: PORTA I/O SUMMARY

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|--------------------------------------|------------------|--------------|-----|----------|---|
| RA0/AN0 | RA0 | 0 | O | DIG | LATA<0> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<0> data input; disabled when analog input enabled. |
| | AN0 | 1 | I | ANA | A/D input channel 0 and Comparator C1- input. Default input configuration on POR; does not affect digital output. |
| RA1/AN1 | RA1 | 0 | O | DIG | LATA<1> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<1> data input; disabled when analog input enabled. |
| | AN1 | 1 | I | ANA | A/D input channel 1 and Comparator C2- input. Default input configuration on POR; does not affect digital output. |
| RA2/AN2/ VREF-/CVREF | RA2 | 0 | O | DIG | LATA<2> data output; not affected by analog input. Disabled when CVREF output enabled. |
| | | 1 | I | TTL | PORTA<2> data input. Disabled when analog functions enabled; disabled when CVREF output enabled. |
| | AN2 | 1 | I | ANA | A/D input channel 2 and Comparator C2+ input. Default input configuration on POR; not affected by analog output. |
| | VREF- | 1 | I | ANA | A/D and comparator voltage reference low input. |
| | CVREF | x | O | ANA | Comparator voltage reference output. Enabling this feature disables digital I/O. |
| RA3/AN3/VREF+ | RA3 | 0 | O | DIG | LATA<3> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<3> data input; disabled when analog input enabled. |
| | AN3 | 1 | I | ANA | A/D input channel 3 and Comparator C1+ input. Default input configuration on POR. |
| | VREF+ | 1 | I | ANA | A/D and comparator voltage reference high input. |
| RA5/AN4/ $\overline{SS1}$ / C2OUT | RA5 | 0 | O | DIG | LATA<5> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<5> data input; disabled when analog input enabled. |
| | AN4 | 1 | I | ANA | A/D input channel 4. Default configuration on POR. |
| | $\overline{SS1}$ | 1 | I | TTL | Slave select input for MSSP1 (MSSP1 module). |
| | C2OUT | 0 | O | DIG | Comparator 2 output; takes priority over port data. |
| OSC2/CLKO | OSC2 | x | O | ANA | Main oscillator feedback output connection (HS mode). |
| | CLKO | x | O | DIG | System cycle clock output (Fosc/4) in RC and EC Oscillator modes. |
| OSC1/CLKI | OSC1 | x | I | ANA | Main oscillator input connection. |
| | CLKI | x | I | ANA | Main clock input connection. |

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

PIC18F45J10 FAMILY

TABLE 9-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|-------|-------|--|-------|--------|--------|--------|--------|----------------------|
| PORTA | — | — | RA5 | — | RA3 | RA2 | RA1 | RA0 | 46 |
| LATA | — | — | PORTA Data Latch Register (Read and Write to Data Latch) | | | | | | 46 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 44 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 45 |
| CVRCON | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 45 |

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

9.3 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 9-3: INITIALIZING PORTB

```

CLRF   PORTB   ; Initialize PORTB by
              ; clearing output
              ; data latches
CLRF   LATB    ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0Fh    ; Set RB<4:0> as
MOVWF  ADCON1  ; digital I/O pins
              ; (required if config bit
              ; PBADEN is set)
MOVLW  0CFh   ; 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, $\overline{\text{RBPU}}$ (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, 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 ORed together to generate the RB Port Change Interrupt with Flag bit, RBIF (INTCON<0>).

This interrupt can wake the device from Sleep mode or any of the Idle modes. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

- Any read or write of PORTB (except with the MOVFF (ANY), PORTB instruction).
- 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).

The RB5 pin is multiplexed with the Timer0 module clock input and one of the comparator outputs to become the RB5/KBI1/T0CKI/C1OUT pin.

PIC18F45J10 FAMILY

TABLE 9-5: PORTB I/O SUMMARY

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|--------------------------|---------------------|--------------|-----|--------------------------------------|--|
| RB0/INT0/FLT0/ AN12 | RB0 | 0 | O | DIG | LATB<0> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTB<0> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. ⁽¹⁾ |
| | INT0 | 1 | I | ST | External interrupt 0 input. |
| | FLT0 | 1 | I | ST | PWM Fault input (ECCP1/CCP1 module); enabled in software. |
| AN12 | 1 | I | ANA | A/D input channel 12. ⁽¹⁾ | |
| RB1/INT1/AN10 | RB1 | 0 | O | DIG | LATB<1> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTB<1> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. ⁽¹⁾ |
| | INT1 | 1 | I | ST | External interrupt 1 input. |
| | AN10 | 1 | I | ANA | A/D input channel 10. ⁽¹⁾ |
| RB2/INT2/AN8 | RB2 | 0 | O | DIG | LATB<2> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTB<2> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. ⁽¹⁾ |
| | INT2 | 1 | I | ST | External interrupt 2 input. |
| | AN8 | 1 | I | ANA | A/D input channel 8. ⁽¹⁾ |
| RB3/AN9/CCP2 | RB3 | 0 | O | DIG | LATB<3> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTB<3> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. ⁽¹⁾ |
| | AN9 | 1 | I | ANA | A/D input channel 9. ⁽¹⁾ |
| | CCP2 ⁽²⁾ | 0 | O | DIG | CCP2 compare and PWM output. |
| | | 1 | I | ST | CCP2 capture input |
| RB4/KBI0/AN11 | RB4 | 0 | O | DIG | LATB<4> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTB<4> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. ⁽¹⁾ |
| | KBI0 | 1 | I | TTL | Interrupt-on-change pin. |
| | AN11 | 1 | I | ANA | A/D input channel 11. ⁽¹⁾ |
| RB5/KBI1/TOCKI/ C1OUT | RB5 | 0 | O | DIG | LATB<5> data output. |
| | | 1 | I | TTL | PORTB<5> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. |
| | KBI1 | 1 | I | TTL | Interrupt-on-change pin. |
| | TOCKI | 1 | I | ST | Timer0 clock input. |
| | C1OUT | 0 | O | DIG | Comparator 1 output; takes priority over port data. |
| RB6/KBI2/PGC | RB6 | 0 | O | DIG | LATB<6> data output. |
| | | 1 | I | TTL | PORTB<6> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. |
| | KBI2 | 1 | I | TTL | Interrupt-on-change pin. |
| | PGC | x | I | ST | Serial execution (ICSP™) clock input for ICSP and ICD operation. ⁽³⁾ |
| RB7/KBI3/PGD | RB7 | 0 | O | DIG | LATB<7> data output. |
| | | 1 | I | TTL | PORTB<7> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. |
| | KBI3 | 1 | I | TTL | Interrupt-on-change pin. |
| | PGD | x | O | DIG | Serial execution data output for ICSP and ICD operation. ⁽³⁾ |
| | | x | I | ST | Serial execution data input for ICSP and ICD operation. ⁽³⁾ |

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: Configuration on POR is determined by the PBADEN configuration bit. Pins are configured as analog inputs by default when PBADEN is set and digital inputs when PBADEN is cleared.

2: Alternate assignment for CCP2 when the CCP2MX configuration bit is '0'. Default assignment is RC1.

3: All other pin functions are disabled when ICSP or ICD are enabled.

PIC18F45J10 FAMILY

TABLE 9-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|--|-----------|---------|---------|--------|--------|--------|--------|----------------------|
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 46 |
| LATB | PORTB Data Latch Register (Read and Write to Data Latch) | | | | | | | | 46 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 46 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| INTCON2 | $\overline{\text{RBP}}\text{U}$ | INTEDG0 | INTEDG1 | INTEDG2 | — | TMR0IP | — | RBIP | 43 |
| INTCON3 | INT2IP | INT1IP | — | INT2IE | INT1IE | — | INT2IF | INT1IF | 43 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 44 |

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

PIC18F45J10 FAMILY

9.4 PORTC, TRISC and LATIC 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 (LATIC) is also memory mapped. Read-modify-write operations on the LATIC register read and write the latched output value for PORTC.

PORTC is multiplexed with several peripheral functions (Table 9-7). The pins have Schmitt Trigger input buffers. RC1 is normally configured by configuration bit, CCP2MX, 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 additional information.

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 9-4: INITIALIZING PORTC

```
CLRF  PORTC    ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF  LATIC     ; Alternate method
                ; to clear output
                ; data latches
MOVLW 0CFh     ; Value used to
                ; initialize data
                ; direction
MOVWF TRISC    ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

PIC18F45J10 FAMILY

TABLE 9-7: PORTC I/O SUMMARY

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|---------------------|---------------------|--------------|----------------------|--|---|
| RC0/T1OSO/ T1CKI | RC0 | 0 | O | DIG | LATC<0> data output. |
| | | 1 | I | ST | PORTC<0> data input. |
| | T1OSO | x | O | ANA | Timer1 oscillator output; enabled when Timer1 oscillator enabled. Disables digital I/O. |
| | T1CKI | 1 | I | ST | Timer1 counter input. |
| RC1/T1OSI/CCP2 | RC1 | 0 | O | DIG | LATC<1> data output. |
| | | 1 | I | ST | PORTC<1> data input. |
| | T1OSI | x | I | ANA | Timer1 oscillator input; enabled when Timer1 oscillator enabled. Disables digital I/O. |
| | CCP2 ⁽¹⁾ | 0 | O | DIG | CCP2 compare and PWM output; takes priority over port data. |
| 1 | | I | ST | CCP2 capture input. | |
| RC2/CCP1/P1A | RC2 | 0 | O | DIG | LATC<2> data output. |
| | | 1 | I | ST | PORTC<2> data input. |
| | CCP1 | 0 | O | DIG | ECCP1/CCP1 compare or PWM output; takes priority over port data. |
| | | 1 | I | ST | ECCP1/CCP1 capture input. |
| | P1A ⁽²⁾ | 0 | O | DIG | ECCP1 Enhanced PWM output, channel A. May be configured for tri-state during Enhanced PWM shutdown events. Takes priority over port data. |
| RC3/SCK1/SCL1 | RC3 | 0 | O | DIG | LATC<3> data output. |
| | | 1 | I | ST | PORTC<3> data input. |
| | SCK1 | 0 | O | DIG | SPI™ clock output (MSSP1 module); takes priority over port data. |
| | | 1 | I | ST | SPI clock input (MSSP1 module). |
| | SCL1 | 0 | O | DIG | I ² C™ clock output (MSSP1 module); takes priority over port data. |
| 1 | | I | I ² C/SMB | I ² C clock input (MSSP1 module); input type depends on module setting. | |
| RC4/SDI1/SDA1 | RC4 | 0 | O | DIG | LATC<4> data output. |
| | | 1 | I | ST | PORTC<4> data input. |
| | SDI1 | 1 | I | ST | SPI data input (MSSP1 module). |
| | SDA1 | 1 | O | DIG | I ² C data output (MSSP1 module); takes priority over port data. |
| 1 | | I | I ² C/SMB | I ² C data input (MSSP1 module); input type depends on module setting. | |
| RC5/SDO1 | RC5 | 0 | O | DIG | LATC<5> data output. |
| | | 1 | I | ST | PORTC<5> data input. |
| | SDO1 | 0 | O | DIG | SPI data output (MSSP1 module); takes priority over port data. |
| RC6/TX/CK | RC6 | 0 | O | DIG | LATC<6> data output. |
| | | 1 | I | ST | PORTC<6> data input. |
| | TX | 1 | O | DIG | Asynchronous serial transmit data output (EUSART module); takes priority over port data. User must configure as output. |
| | CK | 1 | O | DIG | Synchronous serial clock output (EUSART module); takes priority over port data. |
| | | 1 | I | ST | Synchronous serial clock input (EUSART module). |
| RC7/RX/DT | RC7 | 0 | O | DIG | LATC<7> data output. |
| | | 1 | I | ST | PORTC<7> data input. |
| | RX | 1 | I | ST | Asynchronous serial receive data input (EUSART module). |
| | DT | 1 | O | DIG | Synchronous serial data output (EUSART module); takes priority over port data. |
| | | 1 | I | ST | Synchronous serial data input (EUSART module). User must configure as an input. |

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; I²C/SMB = I²C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: Default assignment for CCP2 when the CCP2MX configuration bit is set. Alternate assignment is RB3.

2: Enhanced PWM output is available only on PIC18F44J10/45J10 devices.

PIC18F45J10 FAMILY

TABLE 9-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|-------|--|-------|-------|-------|-------|-------|-------|-------|----------------------|
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 46 |
| LATC | PORTC Data Latch Register (Read and Write to Data Latch) | | | | | | | | 46 |
| TRISC | PORTC Data Direction Control Register | | | | | | | | 46 |

9.5 PORTD, TRISD and LATD Registers

Note: PORTD is only available in 40/44-pin 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 14.0 “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 micro-processor port (Parallel Slave Port) by setting control bit, PSPMODE (TRISE<4>). In this mode, the input buffers are TTL. See **Section 9.7 “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 9-5: INITIALIZING PORTD

```
CLRF   PORTD   ; Initialize PORTD by
              ; clearing output
              ; data latches
CLRF   LATD    ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0CFh   ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISD   ; Set RD<3:0> as inputs
              ; RD<5:4> as outputs
              ; RD<7:6> as inputs
```

PIC18F45J10 FAMILY

TABLE 9-9: PORTD I/O SUMMARY

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|--------------------|--------------|--------------|----------------------|--|---|
| RD0/PSP0/SCK2/SCL2 | RD0 | 0 | O | DIG | LATD<0> data output. |
| | | 1 | I | ST | PORTD<0> data input. |
| | PSP0 | x | O | DIG | PSP read data output (LATD<0>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | SCK2 | 0 | O | DIG | SPI™ clock output (MSSP2 module); takes priority over port data. |
| | | 1 | I | ST | SPI clock input (MSSP2 module). |
| SCL2 | 0 | O | DIG | I ² C™ clock output (MSSP2 module); takes priority over port data. | |
| | 1 | I | I ² C/SMB | I ² C clock input (MSSP2 module); input type depends on module setting. | |
| RD1/PSP1/SDI2/SDA2 | RD1 | 0 | O | DIG | LATD<1> data output. |
| | | 1 | I | ST | PORTD<1> data input. |
| | PSP1 | x | O | DIG | PSP read data output (LATD<1>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | SDI2 | 1 | I | ST | SPI data input (MSSP2 module). |
| | SDA2 | 1 | O | DIG | I ² C data output (MSSP2 module); takes priority over port data. |
| 1 | | I | I ² C/SMB | I ² C data input (MSSP2 module); input type depends on module setting. | |
| RD2/PSP2/SDO2 | RD2 | 0 | O | DIG | LATD<2> data output. |
| | | 1 | I | ST | PORTD<2> data input. |
| | PSP2 | x | O | DIG | PSP read data output (LATD<2>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | SDO2 | 0 | O | DIG | SPI data output (MSSP2 module); takes priority over port data. |
| | RD3/PSP3/SS2 | RD3 | 0 | O | DIG |
| 1 | | | I | ST | PORTD<3> data input. |
| PSP3 | | x | O | DIG | PSP read data output (LATD<3>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| SS2 | | 1 | I | TTL | Slave select input for MSSP2 (MSSP2 module). |
| RD4/PSP4 | | RD4 | 0 | O | DIG |
| | 1 | | I | ST | PORTD<4> data input. |
| | PSP4 | x | O | DIG | PSP read data output (LATD<4>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| RD5/PSP5/P1B | RD5 | 0 | O | DIG | LATD<5> data output. |
| | | 1 | I | ST | PORTD<5> data input. |
| | PSP5 | x | O | DIG | PSP read data output (LATD<5>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1B | 0 | O | DIG | ECCP1 Enhanced PWM output, channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |
| RD6/PSP6/P1C | RD6 | 0 | O | DIG | LATD<6> data output. |
| | | 1 | I | ST | PORTD<6> data input. |
| | PSP6 | x | O | DIG | PSP read data output (LATD<6>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1C | 0 | O | DIG | ECCP1 Enhanced PWM output, channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |
| RD7/PSP7/P1D | RD7 | 0 | O | DIG | LATD<7> data output. |
| | | 1 | I | ST | PORTD<7> data input. |
| | PSP7 | x | O | DIG | PSP read data output (LATD<7>); takes priority over port data. |
| | | x | I | TTL | PSP write data input. |
| | P1D | 0 | O | DIG | ECCP1 Enhanced PWM output, channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events. |

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; I²C/SMB = I²C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

PIC18F45J10 FAMILY

TABLE 9-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|--|-------|-------|---------|--------|--------|--------|--------|----------------------|
| PORTD ⁽¹⁾ | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 46 |
| LATD ⁽¹⁾ | PORTD Data Latch Register (Read and Write to Data Latch) | | | | | | | | 46 |
| TRISD ⁽¹⁾ | PORTD Data Direction Control Register | | | | | | | | 46 |
| TRISE ⁽¹⁾ | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 46 |
| CCP1CON | P1M1 | P1M0 | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 45 |

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

Note 1: These registers are not available in 28-pin devices.

PIC18F45J10 FAMILY

9.6 PORTE, TRISE and LATE Registers

Note: PORTE is only available in 40/44-pin devices.

Depending on the particular PIC18F45J10 family device selected, PORTE is implemented in two different ways.

For 40/44-pin devices, PORTE is a 4-bit wide port. Three pins (RE0/RD/AN5, RE1/WR/AN6 and RE2/CS/AN7) are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. When selected as analog inputs, 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 9-1.

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.

EXAMPLE 9-6: INITIALIZING PORTE

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

PIC18F45J10 FAMILY

REGISTER 9-1: TRISE REGISTER (40/44-PIN DEVICES ONLY)

| 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 |

PIC18F45J10 FAMILY

TABLE 9-11: PORTE I/O SUMMARY

| Pin | Function | TRIS Setting | I/O | I/O Type | Description |
|----------------------------------|------------------------|--------------|-----|----------|--|
| RE0/ $\overline{\text{RD}}$ /AN5 | RE0 | 0 | O | DIG | LATE<0> data output; not affected by analog input. |
| | | 1 | I | ST | PORTE<0> data input; disabled when analog input enabled. |
| | $\overline{\text{RD}}$ | 1 | I | TTL | PSP read enable input (PSP enabled). |
| | AN5 | 1 | I | ANA | A/D input channel 5; default input configuration on POR. |
| RE1/ $\overline{\text{WR}}$ /AN6 | RE1 | 0 | O | DIG | LATE<1> data output; not affected by analog input. |
| | | 1 | I | ST | PORTE<1> data input; disabled when analog input enabled. |
| | $\overline{\text{WR}}$ | 1 | I | TTL | PSP write enable input (PSP enabled). |
| | AN6 | 1 | I | ANA | A/D input channel 6; default input configuration on POR. |
| RE2/ $\overline{\text{CS}}$ /AN7 | RE2 | 0 | O | DIG | LATE<2> data output; not affected by analog input. |
| | | 1 | I | ST | PORTE<2> data input; disabled when analog input enabled. |
| | $\overline{\text{CS}}$ | 1 | I | TTL | PSP write enable input (PSP enabled). |
| | AN7 | 1 | I | ANA | A/D input channel 7; default input configuration on POR. |

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 9-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|-------|-------|-------|---------|-------|---|--------|--------|----------------------|
| PORTE ⁽¹⁾ | — | — | — | — | — | RE2 | RE1 | RE0 | 46 |
| LATE ⁽¹⁾ | — | — | — | — | — | PORTE Data Latch Register (Read and Write to Data Latch) | | | 46 |
| TRISE ⁽¹⁾ | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 46 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 44 |

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

Note 1: These registers are not available in 28-pin devices.

PIC18F45J10 FAMILY

9.7 Parallel Slave Port

Note: The Parallel Slave Port is only available in 40/44-pin 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 9-1). 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 a value in the range of '1010' through '1111'.

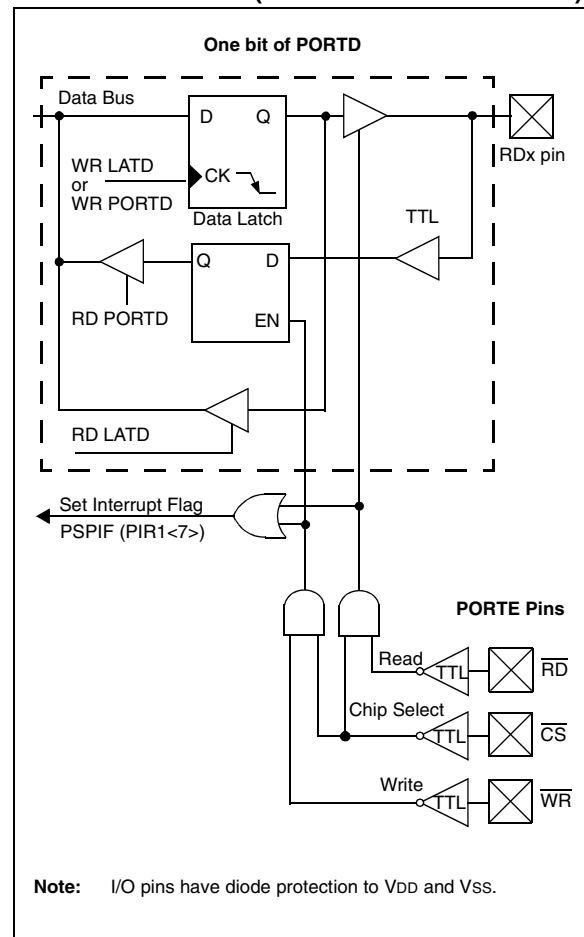
A write to the PSP occurs when both the \overline{CS} and \overline{WR} lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits are both set when the write ends.

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 clear. 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 9-4 and Figure 9-5, respectively.

FIGURE 9-3: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



PIC18F45J10 FAMILY

FIGURE 9-4: PARALLEL SLAVE PORT WRITE WAVEFORMS

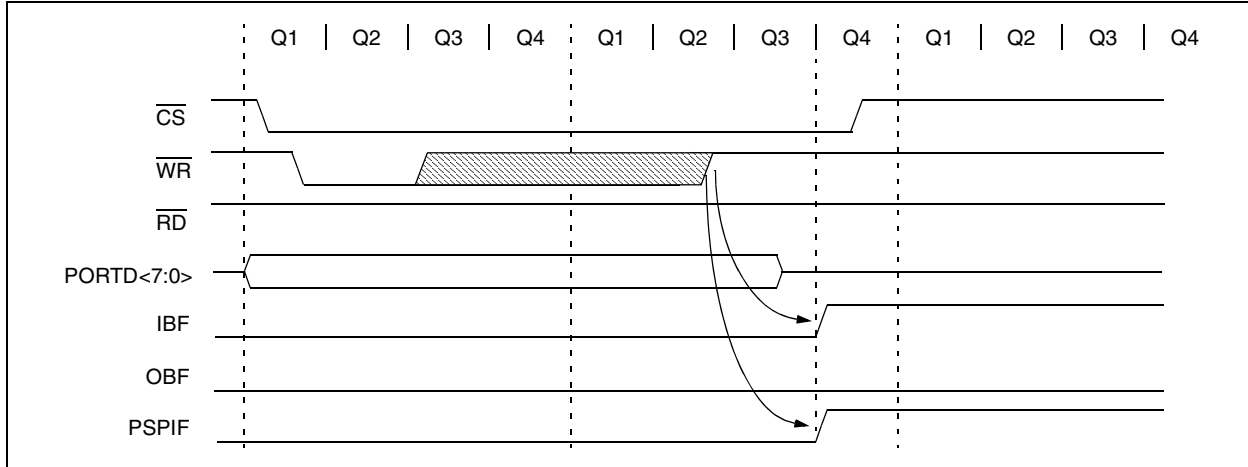


FIGURE 9-5: PARALLEL SLAVE PORT READ WAVEFORMS

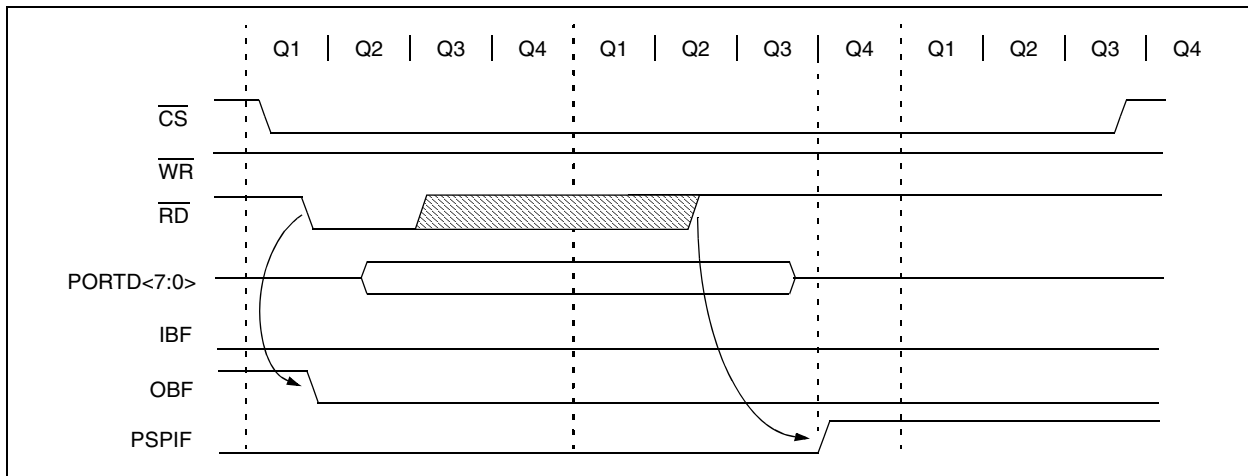


TABLE 9-13: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|--|-----------|--------|---------|--------|--|--------|--------|----------------------|
| PORTD ⁽¹⁾ | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 46 |
| LATD ⁽¹⁾ | PORTD Data Latch Register (Read and Write to Data Latch) | | | | | | | | 46 |
| TRISD ⁽¹⁾ | PORTD Data Direction Control Register | | | | | | | | 46 |
| PORTE ⁽¹⁾ | — | — | — | — | — | RE2 | RE1 | RE0 | 46 |
| LATE ⁽¹⁾ | — | — | — | — | — | PORTE Data Latch Register (Read and Write to Data Latch) | | | 46 |
| TRISE ⁽¹⁾ | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 46 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 44 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Parallel Slave Port.

Note 1: These registers and/or bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

10.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

- Software selectable operation as a timer or counter in both 8-bit or 16-bit modes
- Readable and writable registers
- Dedicated 8-bit, software programmable prescaler
- Selectable clock source (internal or external)
- Edge select for external clock
- Interrupt-on-overflow

The T0CON register (Register 10-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

A simplified block diagram of the Timer0 module in 8-bit mode is shown in Figure 10-1. Figure 10-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 10-1: 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 0 |
| | | | | | | | bit 7 |

- 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 |

PIC18F45J10 FAMILY

10.1 Timer0 Operation

Timer0 can operate as either a timer or a counter; the mode is selected with the T0CS bit (T0CON<5>). In Timer mode (T0CS = 0), the module increments on every clock by default unless a different prescaler value is selected (see **Section 10.3 “Prescaler”**). If the TMR0 register is written to, 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.

The Counter mode is selected by setting the T0CS bit (= 1). In this mode, Timer0 increments either on every rising or falling edge of pin RB5/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (T0CON<4>); clearing this bit selects the rising edge. Restrictions on the external clock input are discussed below.

An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the

internal phase clock (Tosc). There is a delay between synchronization and the onset of incrementing the timer/counter.

10.2 Timer0 Reads and Writes in 16-Bit Mode

TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0 which is not directly readable nor writable (refer to Figure 10-2). 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.

Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The 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.

FIGURE 10-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)

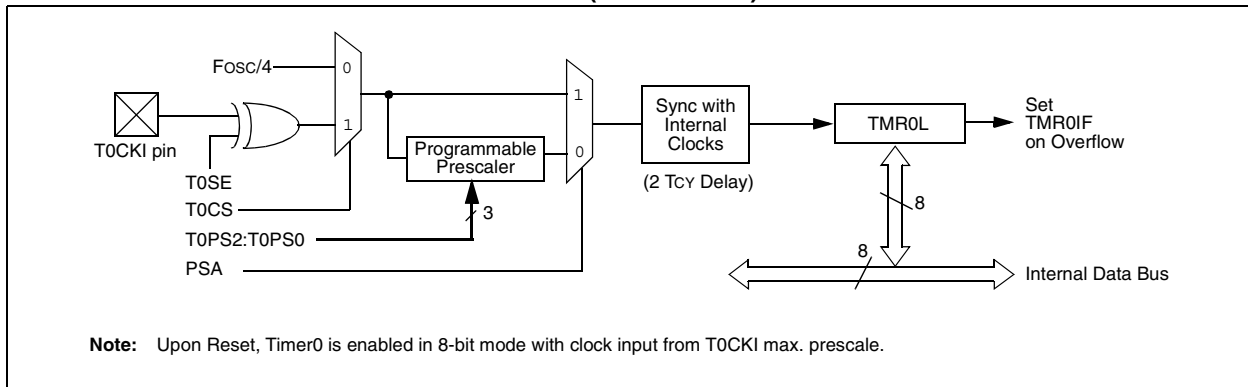
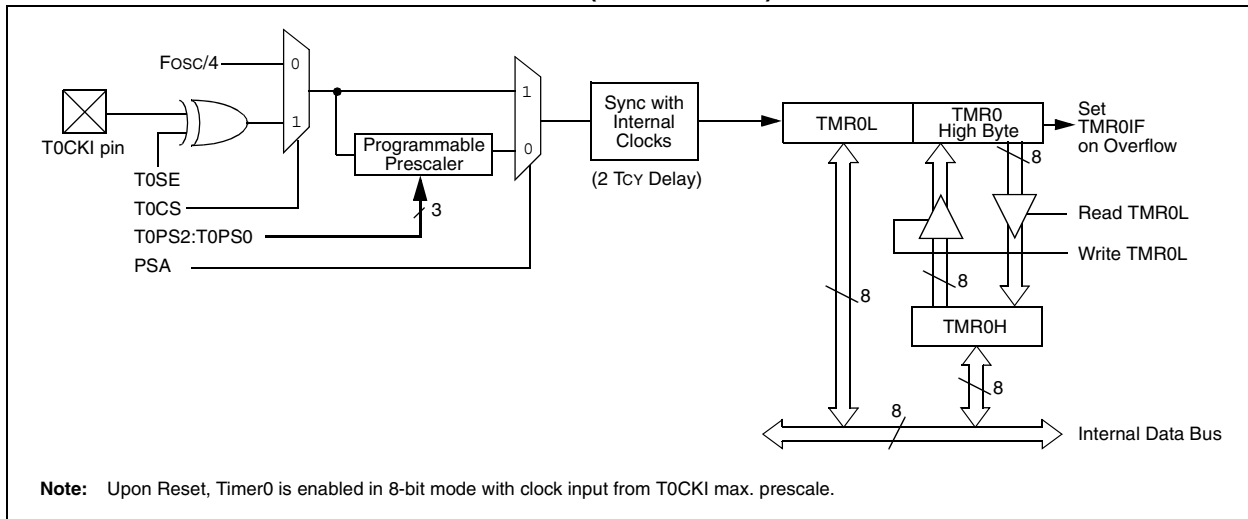


FIGURE 10-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



PIC18F45J10 FAMILY

10.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable. Its value is set by the PSA and T0PS2:T0PS0 bits (T0CON<3:0>) which determine the prescaler assignment and prescale ratio.

Clearing the PSA bit assigns the prescaler to the Timer0 module. When it is assigned, prescale values from 1:2 through 1:256 in power-of-2 increments are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., CLRF TMR0, MOVWF TMR0, BSF TMR0, etc.) 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.

10.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control and can be changed “on-the-fly” during program execution.

10.4 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or from FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF flag bit. The interrupt can be masked by clearing the TMR0IE bit (INTCON<5>). Before re-enabling the interrupt, the TMR0IF bit must be cleared in software by the Interrupt Service Routine.

Since Timer0 is shut down in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

TABLE 10-1: REGISTERS ASSOCIATED WITH TIMER0

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|---------------------------|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| TMR0L | Timer0 Register Low Byte | | | | | | | | 44 |
| TMR0H | Timer0 Register High Byte | | | | | | | | 44 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| T0CON | TMR0ON | T08BIT | T0CS | T0SE | PSA | T0PS2 | T0PS1 | T0PS0 | 44 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used by Timer0.

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

11.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Reset on CCP Special Event Trigger
- Device clock status flag (T1RUN)

A simplified block diagram of the Timer1 module is shown in Figure 11-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 11-2.

The module incorporates its own low-power oscillator to provide an additional clocking option. The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation.

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.

Timer1 is controlled through the T1CON Control register (Register 11-1). It also contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

REGISTER 11-1: 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 | 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 = Device clock is derived from Timer1 oscillator
 0 = Device 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 **T1SYNC:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
 1 = Do not synchronize external clock input
 0 = Synchronize external clock input
When TMR1CS = 0:
 This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.
- bit 1 **TMR1CS:** Timer1 Clock Source Select bit
 1 = External clock from pin RC0/T1OSO/T1CKI (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 |

PIC18F45J10 FAMILY

11.1 Timer1 Operation

Timer1 can operate in one of these modes:

- Timer
- Synchronous Counter
- Asynchronous Counter

The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>). When TMR1CS is cleared (= 0), Timer1 increments on every internal instruction

cycle ($F_{OSC}/4$). When the bit is set, Timer1 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

When Timer1 is enabled, the RC1/T1OSI and RC0/T1OSO/T1CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 11-1: TIMER1 BLOCK DIAGRAM

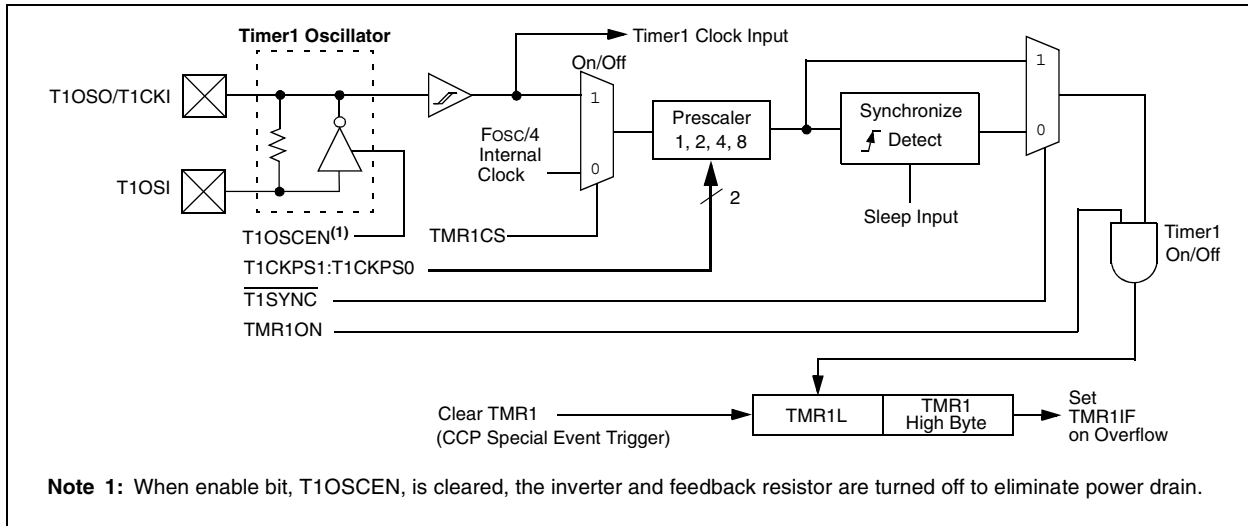
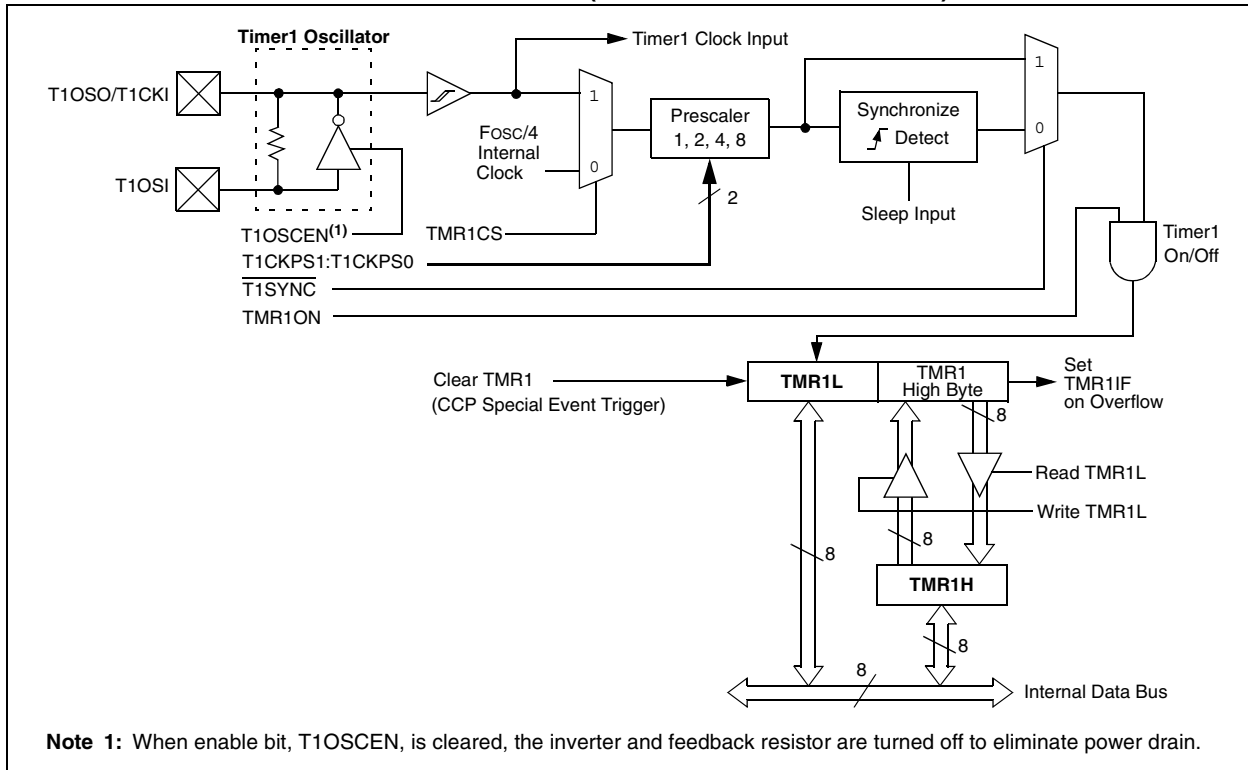


FIGURE 11-2: TIMER1 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



PIC18F45J10 FAMILY

11.2 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 11-2). 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 register. 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, has become invalid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. The 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.

11.3 Timer1 Oscillator

An on-chip crystal oscillator circuit is incorporated between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting the Timer1 Oscillator Enable bit, T1OSCEN (T1CON<3>). The oscillator is a low-power circuit rated for 32 kHz crystals. It will continue to run during all power-managed modes. The circuit for a typical oscillator is shown in Figure 11-3. Table 11-1 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 11-3: EXTERNAL COMPONENTS FOR THE TIMER1 OSCILLATOR

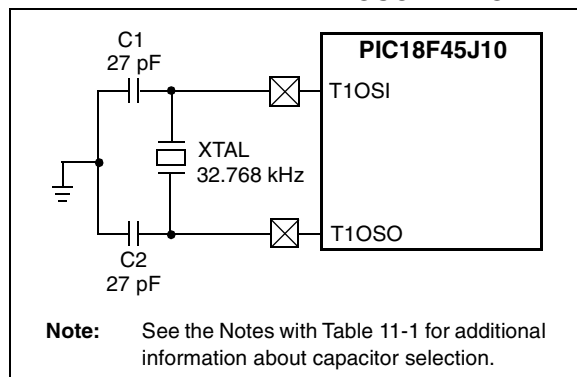


TABLE 11-1: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR^(2,3,4)

| Oscillator Type | Freq. | C1 | C2 |
|-----------------|--------|----------------------|----------------------|
| LP | 32 kHz | 27 pF ⁽¹⁾ | 27 pF ⁽¹⁾ |

Note 1: Microchip suggests these values 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.

11.3.1 USING TIMER1 AS A CLOCK SOURCE

The Timer1 oscillator is also available as a clock source in power-managed modes. By setting the clock select bits, SCS1:SCS0 (OSCCON<1:0>), to '01', the device switches to SEC_RUN mode; both the CPU and peripherals are clocked from the Timer1 oscillator. If the IDLEN bit (OSCCON<7>) is cleared and a SLEEP instruction is executed, the device enters SEC_IDLE mode. Additional details are available in **Section 3.0 "Power-Managed Modes"**.

Whenever the Timer1 oscillator is providing the clock source, the Timer1 system clock status flag, T1RUN (T1CON<6>), is set. This can be used to determine the controller's current clocking mode. It can also indicate the clock source being currently used by the Fail-Safe Clock Monitor. If the Clock Monitor is enabled and the Timer1 oscillator fails while providing the clock, polling the T1RUN bit will indicate whether the clock is being provided by the Timer1 oscillator or another source.

PIC18F45J10 FAMILY

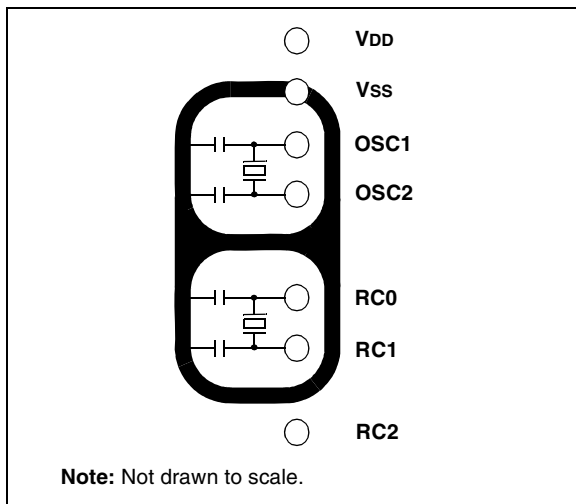
11.3.2 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

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 11-3, 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 11-4, may be helpful when used on a single-sided PCB or in addition to a ground plane.

FIGURE 11-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING



11.4 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 or disabled by setting or clearing the Timer1 Interrupt Enable bit, TMR1IE (PIE1<0>).

11.5 Resetting Timer1 Using the ECCP/CCP Special Event Trigger

If ECCP1/CCP1 or CCP2 are configured to generate a Special Event Trigger in Compare mode (CCPxM3:CCPxM0 = 1011), this signal will reset Timer1. The trigger from CCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 14.2.1 “Special Event Trigger”** for more information).

The module must be configured as either a timer or a synchronous counter to take advantage of this feature. When used this way, the CCPRH:CCPRL register pair effectively becomes a period register for Timer1.

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, the write operation will take precedence.

Note: The Special Event Triggers from the ECCP1/CCPx module will not set the TMR1IF interrupt flag bit (PIR1<0>).

11.6 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in **Section 11.3 “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 11-1, 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 overflows.

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 MSb 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.

PIC18F45J10 FAMILY

EXAMPLE 11-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```

RTCinit
    MOVLW    80h                ; Preload TMR1 register pair
    MOVWF   TMR1H              ; for 1 second overflow
    CLRF    TMR1L
    MOVLW   b'00001111'        ; Configure for external clock,
    MOVWF   T1CON              ; Asynchronous operation, external oscillator
    CLRF    secs               ; Initialize timekeeping registers
    CLRF    mins
    MOVLW   .12
    MOVWF   hours
    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
    CLRF    hours             ; Reset hours
    RETURN                    ; Done
    
```

TABLE 11-2: 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 | Reset Values on page |
|--------|---------------------------|-----------|---------|---------|---------|-----------------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | 44 |
| TMR1H | Timer1 Register High Byte | | | | | | | | 44 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYN \bar{C} | TMR1CS | TMR1ON | 44 |

Legend: Shaded cells are not used by the Timer1 module.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

12.0 TIMER2 MODULE

The Timer2 timer module incorporates the following features:

- 8-bit timer and period registers (TMR2 and PR2, respectively)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4 and 1:16)
- Software programmable postscaler (1:1 through 1:16)
- Interrupt on TMR2-to-PR2 match
- Optional use as the shift clock for the MSSP module

The module is controlled through the T2CON register (Register 12-1) which enables or disables the timer and configures the prescaler and postscaler. Timer2 can be shut off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption.

A simplified block diagram of the module is shown in Figure 12-1.

12.1 Timer2 Operation

In normal operation, TMR2 is incremented from 00h on each clock ($F_{OSC}/4$). A 4-bit counter/prescaler on the clock input gives direct input, divide-by-4 and divide-by-16 prescale options; these are selected by the prescaler control bits, T2CKPS1:T2CKPS0 (T2CON<1:0>). The value of TMR2 is compared to that of the period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/postscaler (see **Section 12.2 “Timer2 Interrupt”**).

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, while the PR2 register initializes at FFh. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset (Power-on Reset, \overline{MCLR} Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 12-1: 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 |
| — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 |
| bit 7 | | | | | | | bit 0 |

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **T2OUTPS3:T2OUTPS0:** 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

PIC18F45J10 FAMILY

12.2 Timer2 Interrupt

Timer2 can also generate an optional device interrupt. The Timer2 output signal (TMR2-to-PR2 match) provides the input for the 4-bit output counter/postscaler. This counter generates the TMR2 match interrupt flag which is latched in TMR2IF (PIR1<1>). The interrupt is enabled by setting the TMR2 Match Interrupt Enable bit, TMR2IE (PIE1<1>).

A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, T2OUTPS3:T2OUTPS0 (T2CON<6:3>).

12.3 Timer2 Output

The unscaled output of TMR2 is available primarily to the CCP modules, where it is used as a time base for operations in PWM mode.

Timer2 can be optionally used as the shift clock source for the MSSP module operating in SPI mode. Additional information is provided in **Section 15.0 “Master Synchronous Serial Port (MSSP) Module”**.

FIGURE 12-1: TIMER2 BLOCK DIAGRAM

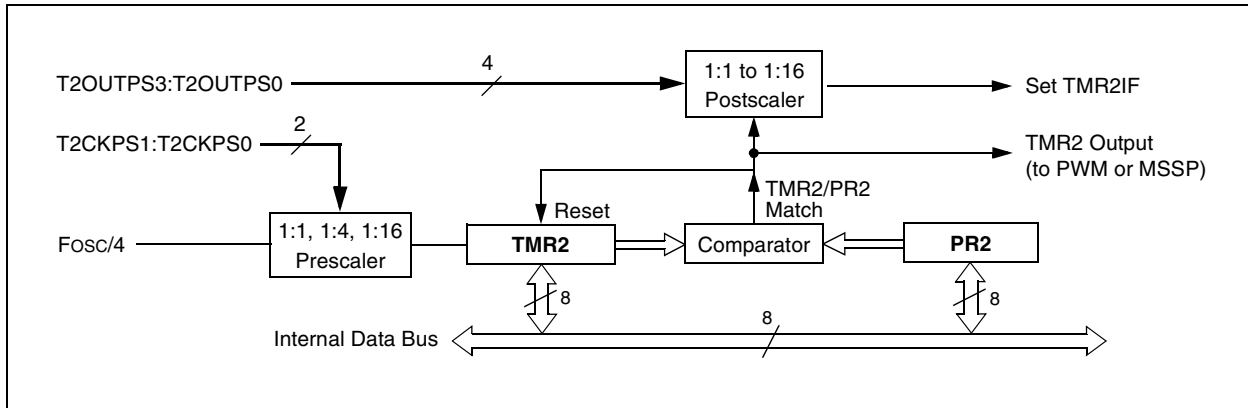


TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|------------------------|-----------|----------|----------|----------|--------|---------|---------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| TMR2 | Timer2 Register | | | | | | | | 44 |
| T2CON | — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | 44 |
| PR2 | Timer2 Period Register | | | | | | | | 44 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer2 module.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

13.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F45J10 family devices all have two CCP (Capture/Compare/PWM) modules. Each module contains a 16-bit register which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register.

In 28-pin devices, the two standard CCP modules (CCP1 and CCP2) operate as described in this chapter. In 40/44-pin devices, CCP1 is implemented as an Enhanced CCP module (ECCP1) with standard Capture and Compare modes and Enhanced PWM modes. The Enhanced CCP implementation is discussed in **Section 14.0 “Enhanced Capture/Compare/PWM (ECCP) Module”**.

The Capture and Compare operations described in this chapter apply to all standard and Enhanced CCP modules.

Note: Throughout this section and **Section 14.0 “Enhanced Capture/Compare/PWM (ECCP) Module”**, references to the register and bit names for CCP modules are referred to generically by the use of ‘x’ or ‘y’ in place of the specific module number. Thus, “CCPxCON” might refer to the control register for CCP1, CCP2 or ECCP1. “CCPxCON” is used throughout these sections to refer to the module control register regardless of whether the CCP module is a standard or Enhanced implementation.

REGISTER 13-1: CCPxCON REGISTER (CCP1, CCP2 MODULES IN 28-PIN DEVICES)

| 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 **Unimplemented:** Read as ‘0’

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 eight MSbs (DCx9:DCx2) of the duty cycle are found in CCPxL.

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 CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)

1001 = Compare mode: initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)

1010 = Compare mode: generate software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)

1011 = Compare mode: trigger special event, reset timer, start A/D conversion on CCPx match (CCPxIF 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

PIC18F45J10 FAMILY

13.1 CCP Module Configuration

Each Capture/Compare/PWM module is associated with a control register (generically, CCPxCON) and a data register (CCPRx). The data register, in turn, is comprised of two 8-bit registers: CCPRxL (low byte) and CCPRxH (high byte). All registers are both readable and writable.

13.1.1 CCP MODULES AND TIMER RESOURCES

The CCP modules utilize Timers 1 or 2, depending on the mode selected. Timer1 is available to modules in Capture or Compare modes, while Timer2 is available for modules in PWM mode.

TABLE 13-1: ECCP/CCP MODE – TIMER RESOURCE

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

Both modules may be active at any given time and may share the same timer resource if they are configured to operate in the same mode (Capture/Compare or PWM) at the same time. The interactions between the two modules are summarized in Figure 13-1 and Figure 13-2. In Timer1 in Asynchronous Counter mode, the capture operation will not work.

13.1.2 CCP2 PIN ASSIGNMENT

The pin assignment for CCP2 (Capture input, Compare and PWM output) can change, based on device configuration. The CCP2MX configuration bit determines which pin CCP2 is multiplexed to. By default, it is assigned to RC1 (CCP2MX = 1). If the configuration bit is cleared, CCP2 is multiplexed with RB3.

Changing the pin assignment of CCP2 does not automatically change any requirements for configuring the port pin. Users must always verify that the appropriate TRIS register is configured correctly for CCP2 operation regardless of where it is located.

TABLE 13-2: INTERACTIONS BETWEEN ECCP1/CCP1 AND CCP2 FOR TIMER RESOURCES

| CCP1 Mode | CCP2 Mode | Interaction |
|--------------------|--------------------|--|
| Capture | Capture | Each module uses TMR1 as the time base. |
| Capture | Compare | CCP2 can be configured for the Special Event Trigger to reset TMR1. Automatic A/D conversions on the trigger event can also be done. Operation of ECCP1/CCP1 will be affected. |
| Compare | Capture | ECCP1/CCP1 can be configured for the Special Event Trigger to reset TMR1. Operation of CCP2 will be affected. |
| Compare | Compare | Either module can be configured for the Special Event Trigger to reset TMR1. Automatic A/D conversions on the CCP2 trigger event can be done. |
| Capture | PWM ⁽¹⁾ | None |
| Compare | PWM ⁽¹⁾ | None |
| PWM ⁽¹⁾ | Capture | None |
| PWM ⁽¹⁾ | Compare | None |
| PWM ⁽¹⁾ | PWM | Both PWMs will have the same frequency and update rate (TMR2 interrupt). |

Note 1: Includes standard and Enhanced PWM operation.

13.2 Capture Mode

In Capture mode, the CCPRxH:CCPRxL register pair captures the 16-bit value of the TMR1 register when an event occurs on the corresponding CCPx pin. 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 the mode select bits, CCPxM3:CCPxM0 (CCPxCON<3:0>). When a capture is made, the interrupt request flag bit, CCPxIF, is set; it must be cleared in software. If another capture occurs before the value in register CCPRx is read, the old captured value is overwritten by the new captured value.

13.2.1 CCP PIN CONFIGURATION

In Capture mode, the appropriate CCPx pin should be configured as an input by setting the corresponding TRIS direction bit.

Note: If RB3/CCP2 or RC1/CCP2 is configured as an output, a write to the port can cause a capture condition.

13.2.2 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit clear to avoid false interrupts. The interrupt flag bit, CCPxIF, should also be cleared following any such change in operating mode.

13.2.3 CCP PRESCALER

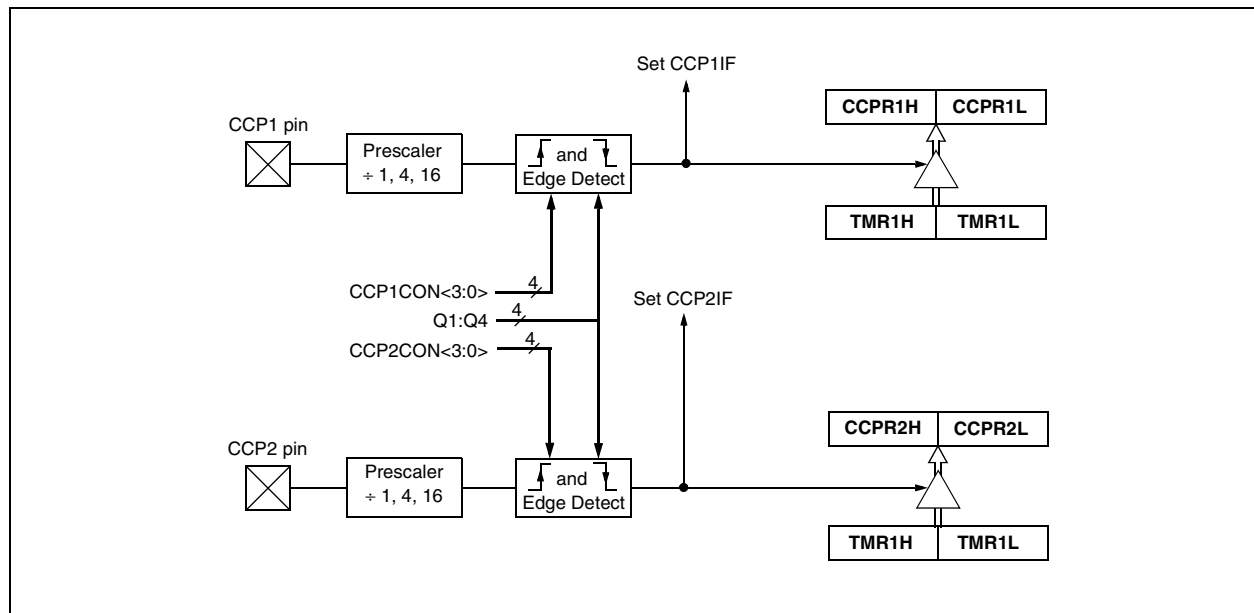
There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCPxM3:CCPxM0). Whenever the CCP module is turned off or Capture mode is disabled, the prescaler counter is cleared. This means that any Reset will clear the prescaler 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 13-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 13-1: CHANGING BETWEEN CAPTURE PRESCALERS (CCP2 SHOWN)

```
CLRF  CCP2CON      ; Turn CCP module off
MOVLW NEW_CAPT_PS ; Load WREG with the
                  ; new prescaler mode
MOVWF  CCP2CON     ; value and CCP ON
                  ; Load CCP2CON with
                  ; this value
```

FIGURE 13-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



PIC18F45J10 FAMILY

13.3 Compare Mode

In Compare mode, the 16-bit CCPRx register value is constantly compared against the TMR1 register value. When a match occurs, the CCPx pin can be:

- driven high
- driven low
- toggled (high-to-low or low-to-high)
- remain unchanged (that is, reflects the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCPxM3:CCPxM0). At the same time, the interrupt flag bit, CCPxIF, is set.

13.3.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRIS bit.

Note: Clearing the CCP2CON register will force the RB3 or RC1 compare output latch (depending on device configuration) to the default low level. This is not the PORTB or PORTC I/O data latch.

13.3.2 TIMER1 MODE SELECTION

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

13.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCPxM3:CCPxM0 = 1010), the corresponding CCPx pin is not affected. Only a CCP interrupt is generated, if enabled and the CCPxIE bit is set.

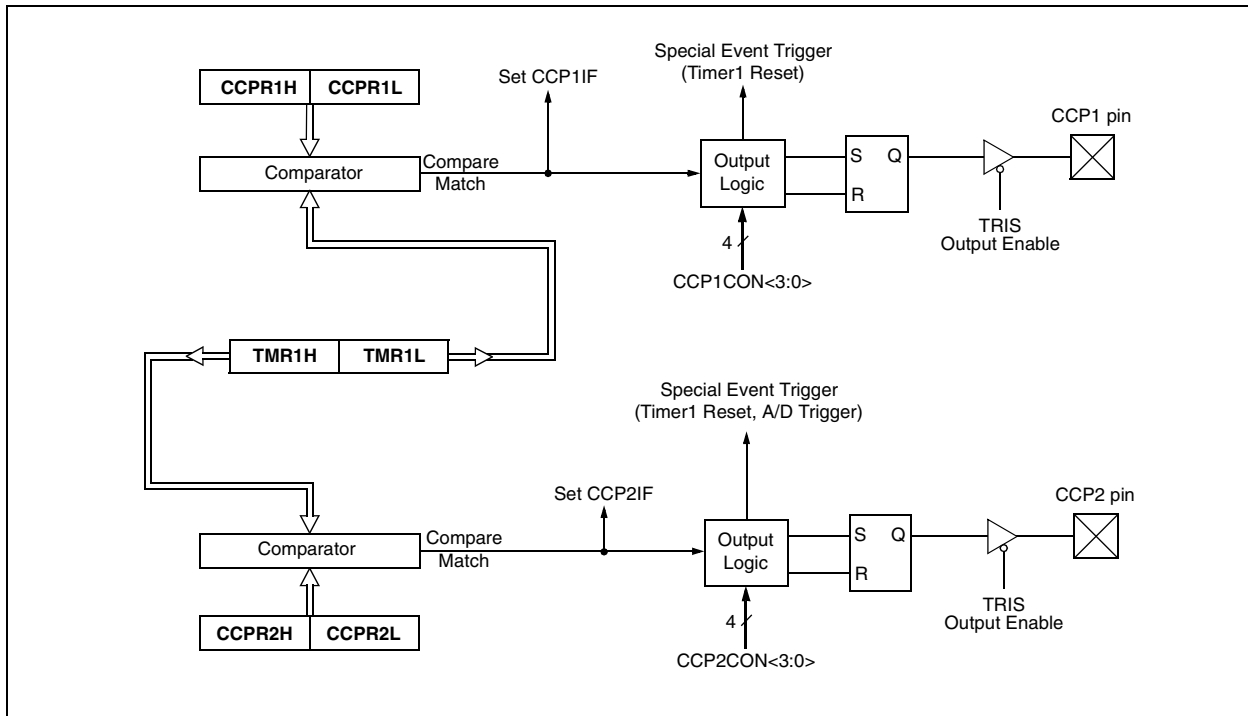
13.3.4 SPECIAL EVENT TRIGGER

Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCPxM3:CCPxM0 = 1011).

For either CCP module, the Special Event Trigger resets the Timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a programmable period register for either timer.

The Special Event Trigger for CCP2 can also start an A/D conversion. In order to do this, the A/D converter must already be enabled.

FIGURE 13-2: COMPARE MODE OPERATION BLOCK DIAGRAM



PIC18F45J10 FAMILY

TABLE 13-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND TIMER1

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|--|---------------------|---------|-----------------|-----------------|---------------------|------------------|------------------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| RCON | IPEN | — | — | \overline{RI} | \overline{TO} | \overline{PD} | \overline{POR} | \overline{BOR} | 42 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 46 |
| TRISC | PORTC Data Direction Control Register | | | | | | | | 46 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | 44 |
| TMR1H | Timer1 Register High Byte | | | | | | | | 44 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYNC}$ | TMR1CS | TMR1ON | 44 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | 45 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | 45 |
| CCP1CON | P1M1 ⁽¹⁾ | P1M0 ⁽¹⁾ | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 45 |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte | | | | | | | | 45 |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte | | | | | | | | 45 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by Capture/Compare or Timer1.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

13.4 PWM Mode

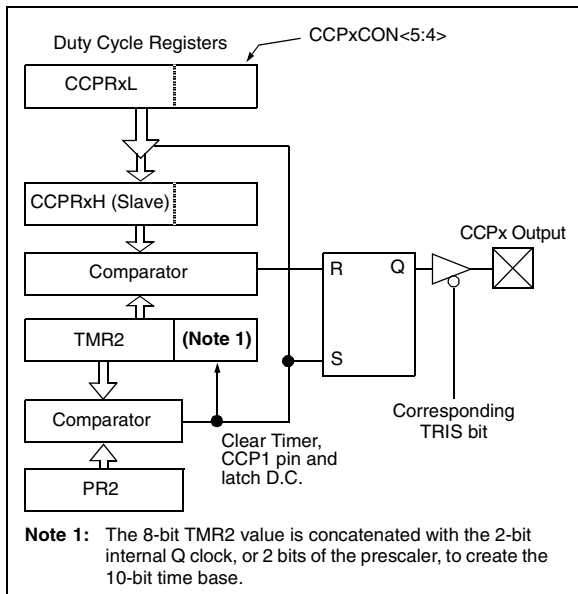
In Pulse-Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP2 pin is multiplexed with a PORTB or PORTC data latch, the appropriate TRIS bit must be cleared to make the CCP2 pin an output.

Note: Clearing the CCP2CON register will force the RB3 or RC1 output latch (depending on device configuration) to the default low level. This is not the PORTB or PORTC I/O data latch.

Figure 13-3 shows a simplified block diagram of the CCP module in PWM mode.

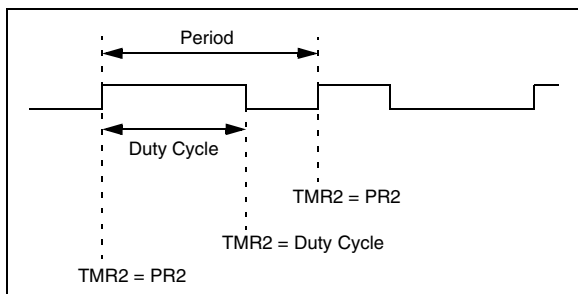
For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 13.4.4 “Setup for PWM Operation”**.

FIGURE 13-3: SIMPLIFIED PWM BLOCK DIAGRAM



A PWM output (Figure 13-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 13-4: PWM OUTPUT



13.4.1 PWM PERIOD

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

EQUATION 13-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot \text{Tosc} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as $1/[\text{PWM period}]$.

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

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

Note: The Timer2 postscalers (see **Section 12.0 “Timer2 Module”**) are 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.

13.4.2 PWM DUTY CYCLE

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

EQUATION 13-2:

$$\text{PWM Duty Cycle} = (\text{CCPRxL:CCPxCON<5:4>}) \cdot \text{Tosc} \cdot (\text{TMR2 Prescale Value})$$

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

PIC18F45J10 FAMILY

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

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

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

EQUATION 13-3:

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

Note: If the PWM duty cycle value is longer than the PWM period, the CCP2 pin will not be cleared.

TABLE 13-4: 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 |

13.4.3 PWM AUTO-SHUTDOWN (CCP1 ONLY)

The PWM auto-shutdown features of the Enhanced CCP module are also available to CCP1 in 28-pin devices. The operation of this feature is discussed in detail in **Section 14.4.7 “Enhanced PWM Auto-Shutdown”**.

Auto-shutdown features are not available for CCP2.

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

PIC18F45J10 FAMILY

TABLE 13-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------|--|---------------------|---------------------|---------------------|---------------------|---------------------|-----------------------|-----------------------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| RCON | IPEN | — | — | \overline{RI} | \overline{TO} | \overline{PD} | \overline{POR} | \overline{BOR} | 42 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 46 |
| TRISC | PORTC Data Direction Control Register | | | | | | | | 46 |
| TMR2 | Timer2 Register | | | | | | | | 44 |
| PR2 | Timer2 Period Register | | | | | | | | 44 |
| T2CON | — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | 44 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | 45 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | 45 |
| CCP1CON | P1M1 ⁽¹⁾ | P1M0 ⁽¹⁾ | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 45 |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte | | | | | | | | 45 |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte | | | | | | | | 45 |
| CCP2CON | — | — | DC2B1 | DC2B0 | CCP2M3 | CCP2M2 | CCP2M1 | CCP2M0 | 45 |
| ECCP1AS | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 ⁽¹⁾ | PSSBD0 ⁽¹⁾ | 45 |
| ECCP1DEL | PRSEN | PDC6 ⁽¹⁾ | PDC5 ⁽¹⁾ | PDC4 ⁽¹⁾ | PDC3 ⁽¹⁾ | PDC2 ⁽¹⁾ | PDC1 ⁽¹⁾ | PDC0 ⁽¹⁾ | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by PWM or Timer2.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

14.0 ENHANCED CAPTURE/ COMPARE/PWM (ECCP) MODULE

Note: The ECCP module is implemented only in 40/44-pin devices.

In PIC18F44J10/45J10 devices, ECCP1 is implemented as a standard CCP module with Enhanced PWM capabilities. These include the provisions for 2 or 4 output channels, user-selectable polarity, dead-band control and automatic shutdown

and restart. The Enhanced features are discussed in detail in **Section 14.4 “Enhanced PWM Mode”**. Capture, Compare and single output PWM functions of the ECCP module are the same as described for the standard CCP module.

The control register for the Enhanced CCP module is shown in Register 14-1. It differs from the CCP1CON register in PIC18F24J10/25J10 devices in that the two Most Significant bits are implemented to control PWM functionality.

REGISTER 14-1: CCP1CON REGISTER (ECCP1 MODULE, 40/44-PIN DEVICES)

| 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:** Enhanced PWM Output Configuration bits

If CCP1M3:CCP1M2 = 00, 01, 10:

xx = P1A assigned as Capture/Compare input/output; P1B, P1C, P1D assigned as port pins

If CCP1M3:CCP1M2 = 11:

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 bit 1 and bit 0

Capture mode:

Unused.

Compare mode:

Unused.

PWM mode:

These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPR1L.

bit 3-0 **CCP1M3:CCP1M0:** Enhanced CCP Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCP module)

0001 = Reserved

0010 = Compare mode, toggle output on match

0011 = Capture mode

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 CCP1 pin low, set output on compare match (set CCP1IF)

1001 = Compare mode, initialize CCP1 pin high, clear output on compare match (set CCP1IF)

1010 = Compare mode, generate software interrupt only, CCP1 pin reverts to I/O state

1011 = Compare mode, trigger special event (ECCP resets TMR1, sets CCP1IF bit)

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

PIC18F45J10 FAMILY

In addition to the expanded range of modes available through the CCP1CON register and ECCP1AS register, the ECCP module has an additional register associated with Enhanced PWM operation and auto-shutdown features. It is:

- ECCP1DEL (Dead-Band Delay)

14.1 ECCP Outputs and Configuration

The Enhanced CCP module may have up to four PWM outputs, depending on the selected operating mode. These outputs, designated P1A through P1D, are multiplexed with I/O pins on PORTC and PORTD. The outputs that are active depend on the ECCP operating mode selected. The pin assignments are summarized in Table 14-1.

To configure the I/O pins as PWM outputs, the proper PWM mode must be selected by setting the P1M1:P1M0 and CCP1M3:CCP1M0 bits. The appropriate TRISC and TRISD direction bits for the port pins must also be set as outputs.

14.1.1 ECCP MODULES AND TIMER RESOURCES

Like the standard CCP modules, the ECCP module can utilize Timers 1 or 2, depending on the mode selected. Timer1 is available for modules in Capture or Compare modes, while Timer2 is available for modules in PWM mode. Interactions between the standard and Enhanced CCP modules are identical to those described for standard CCP modules. Additional details on timer resources are provided in **Section 13.1.1 “CCP Modules and Timer Resources”**.

14.2 Capture and Compare Modes

Except for the operation of the Special Event Trigger discussed below, the Capture and Compare modes of the ECCP module are identical in operation to that of CCP2. These are discussed in detail in **Section 13.2 “Capture Mode”** and **Section 13.3 “Compare Mode”**. No changes are required when moving between 28-pin and 40/44-pin devices.

14.2.1 SPECIAL EVENT TRIGGER

The Special Event Trigger output of ECCP1 resets the TMR1 register pair. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

14.3 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 13.4 “PWM Mode”**. This is also sometimes referred to as “Compatible CCP” mode, as in Table 14-1.

Note: When setting up single output PWM operations, users are free to use either of the processes described in **Section 13.4.4 “Setup for PWM Operation”** or **Section 14.4.9 “Setup for PWM Operation”**. The latter is more generic and will work for either single or multi-output PWM.

TABLE 14-1: PIN ASSIGNMENTS FOR VARIOUS ECCP1 MODES

| ECCP Mode | CCP1CON Configuration | RC2 | RD5 | RD6 | RD7 |
|-------------------------------|-----------------------|------|----------|----------|----------|
| All 40/44-pin Devices: | | | | | |
| Compatible CCP | 00xx 11xx | CCP1 | RD5/PSP5 | RD6/PSP6 | RD7/PSP7 |
| Dual PWM | 10xx 11xx | P1A | P1B | RD6/PSP6 | RD7/PSP7 |
| Quad PWM | x1xx 11xx | P1A | P1B | P1C | P1D |

Legend: x = Don't care. Shaded cells indicate pin assignments not used by ECCP1 in a given mode.

PIC18F45J10 FAMILY

14.4 Enhanced PWM Mode

The Enhanced PWM mode provides additional PWM output options for a broader range of control applications. The module is a backward 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.

Figure 14-1 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 period boundary (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 TRIS bits for output.

14.4.1 PWM PERIOD

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

EQUATION 14-1:

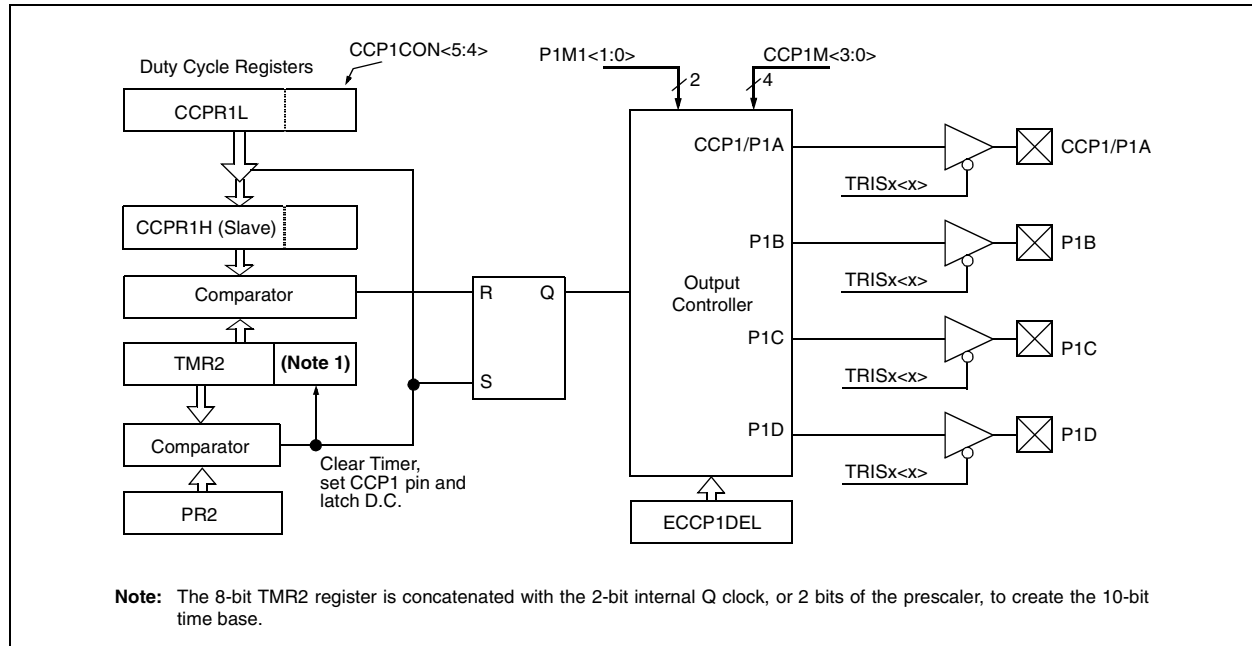
$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

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 12.0 "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.

FIGURE 14-1: SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODULE



PIC18F45J10 FAMILY

14.4.2 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 register 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 14-2:

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot \text{Tosc} \cdot (\text{TMR2 Prescale Value})$$

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.

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 14-3:

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

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

14.4.3 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 14.4 “Enhanced 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 14-2.

TABLE 14-2: 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 |

PIC18F45J10 FAMILY

FIGURE 14-2: PWM OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)

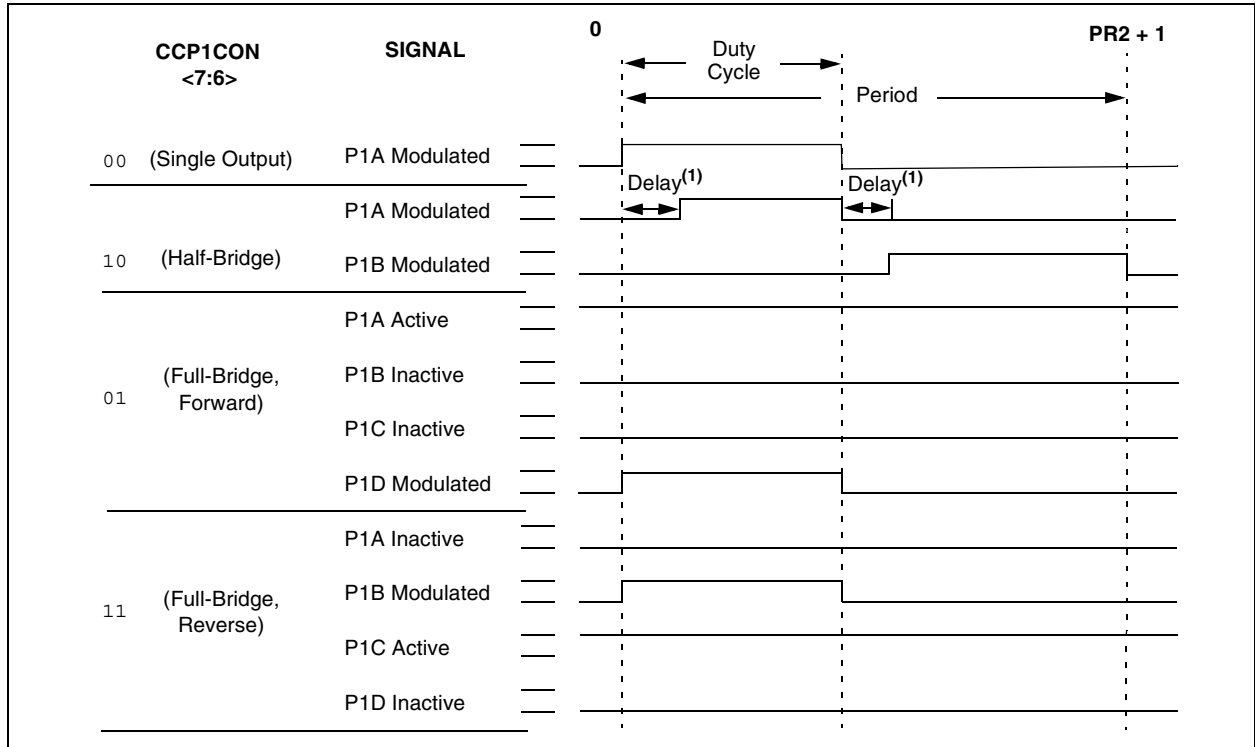
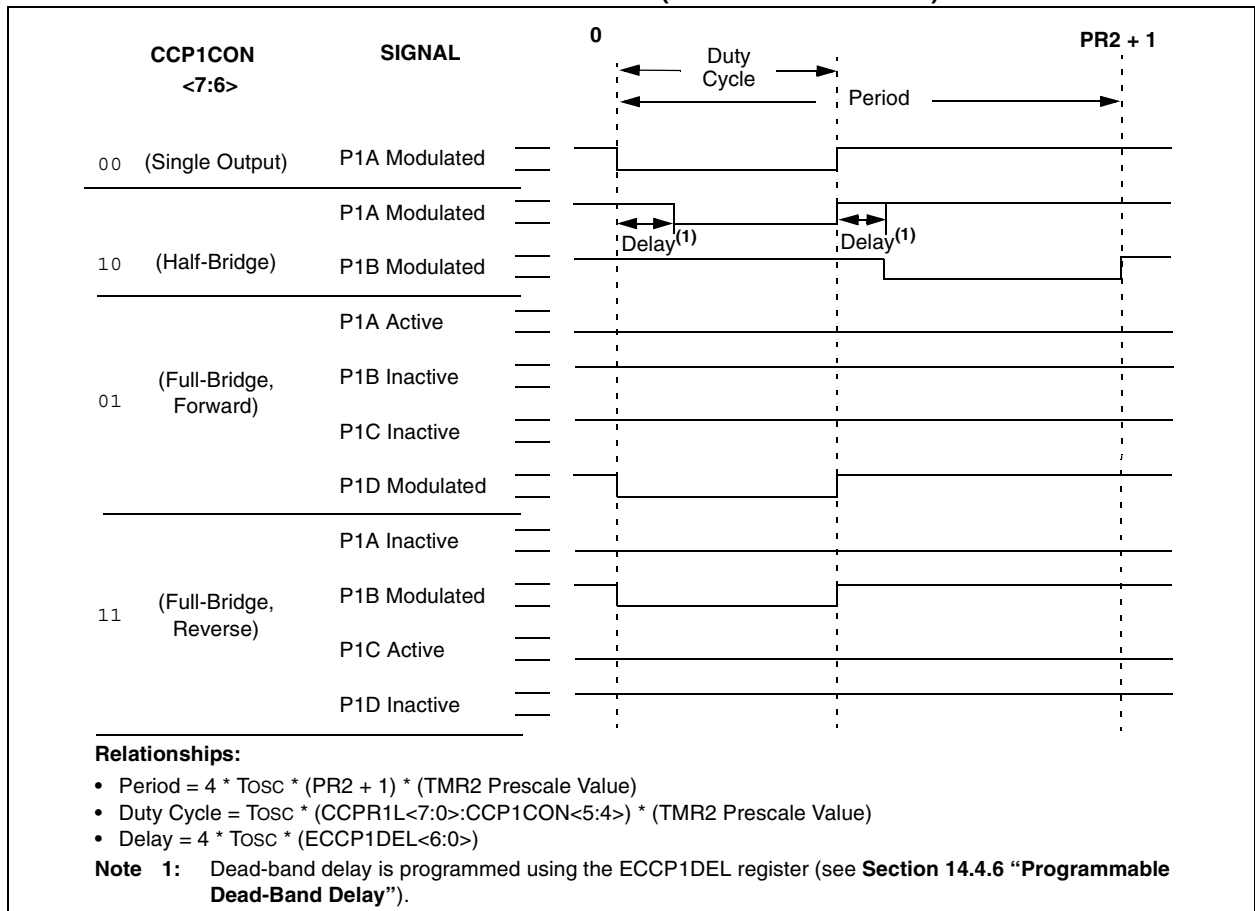


FIGURE 14-3: PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)



PIC18F45J10 FAMILY

14.4.4 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 P1A pin, while the complementary PWM output signal is output on the P1B pin (Figure 14-4). This mode can be used for half-bridge applications, as shown in Figure 14-5, 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 14.4.6 “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 14-4: HALF-BRIDGE PWM OUTPUT

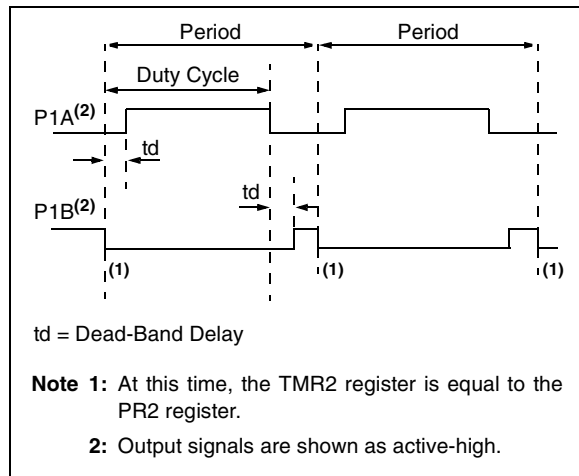
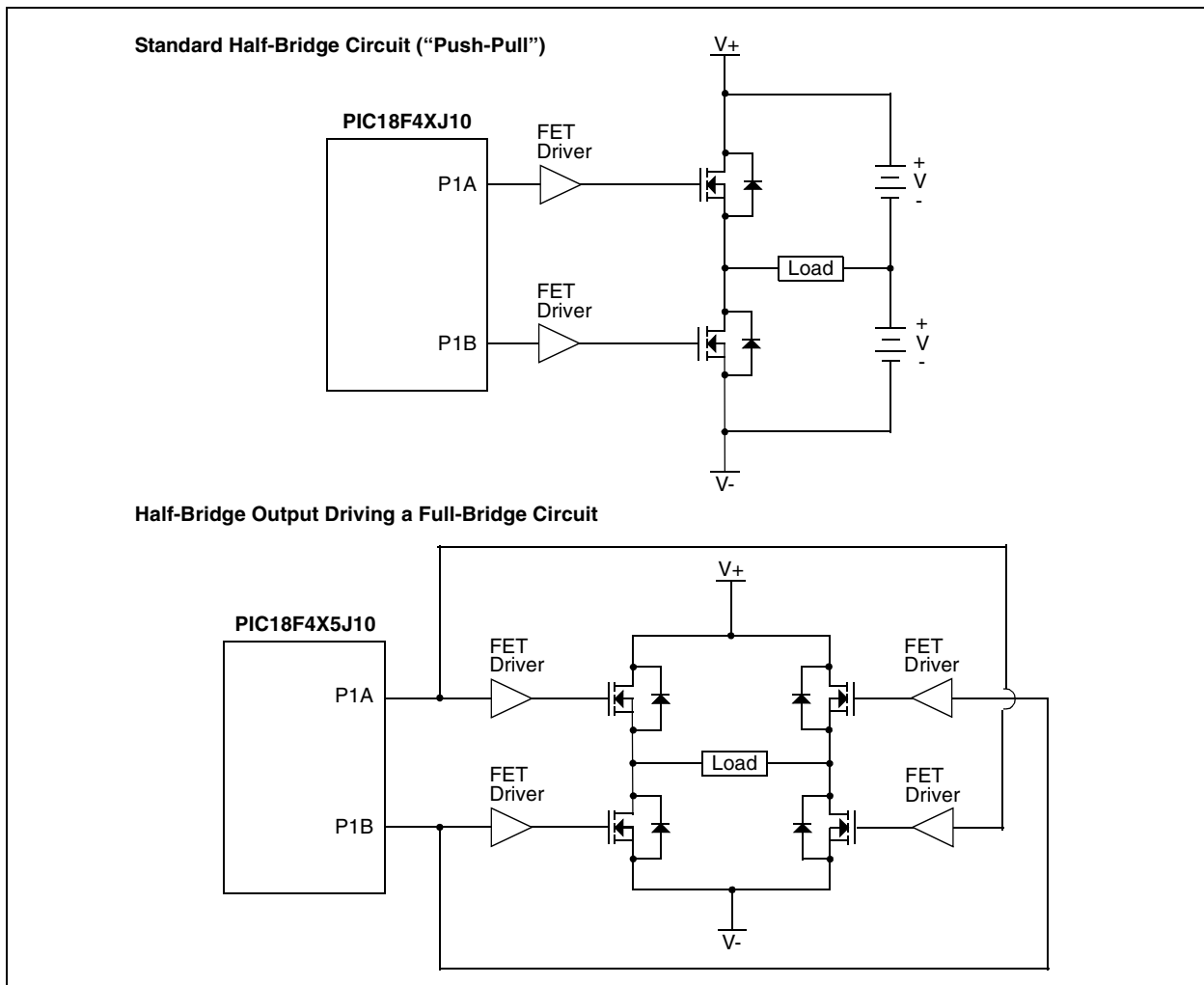


FIGURE 14-5: EXAMPLES OF HALF-BRIDGE OUTPUT MODE APPLICATIONS



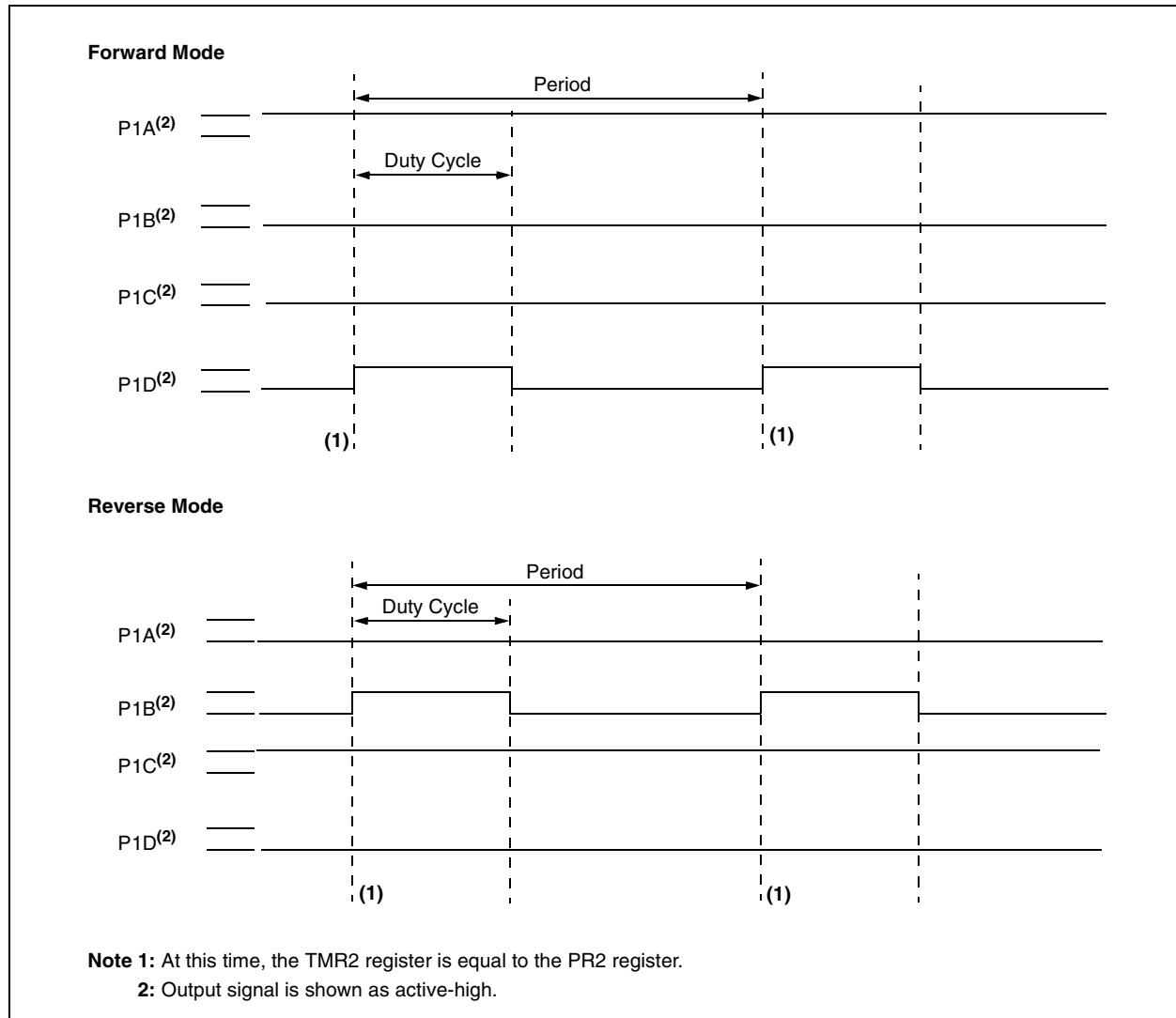
PIC18F45J10 FAMILY

14.4.5 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 P1A is continuously active and pin P1D is modulated. In the Reverse mode, pin P1C is continuously active and pin P1B is modulated. These are illustrated in Figure 14-6.

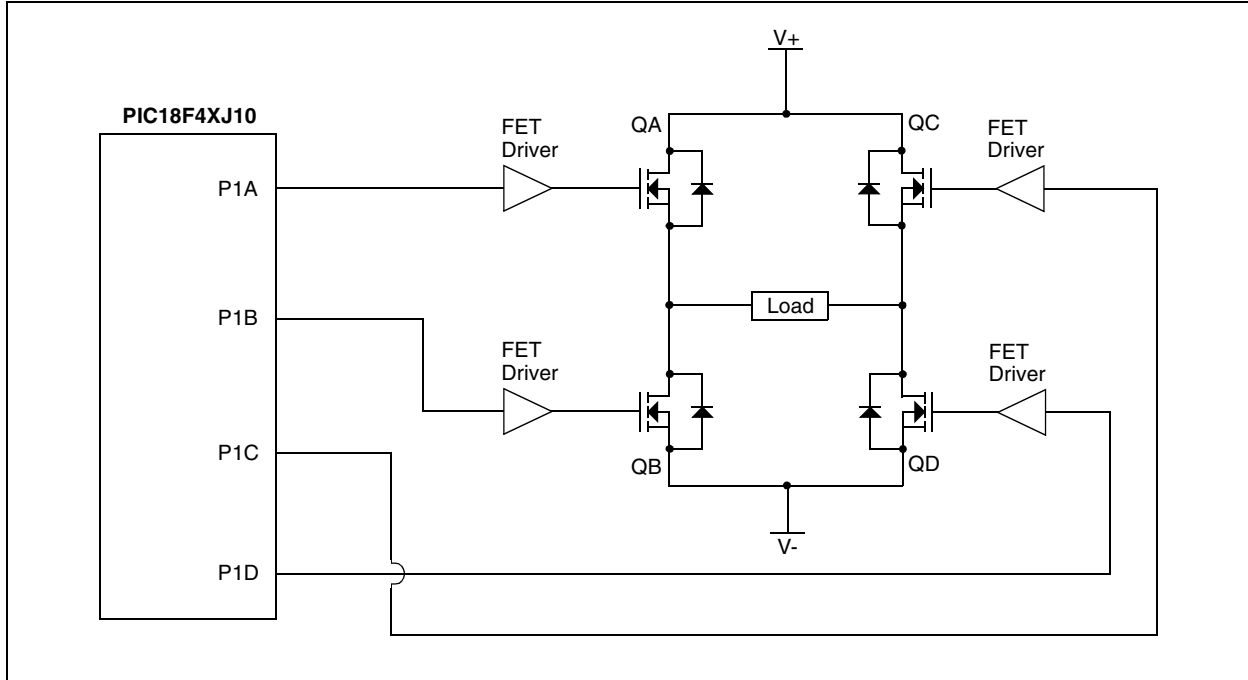
P1A, P1B, P1C and P1D outputs are multiplexed with the PORTC<2> and PORTD<7:5> data latches. The TRISC<2> and TRISD<7:5> bits must be cleared to make the P1A, P1B, P1C and P1D pins outputs.

FIGURE 14-6: FULL-BRIDGE PWM OUTPUT



PIC18F45J10 FAMILY

FIGURE 14-7: EXAMPLE OF FULL-BRIDGE APPLICATION



14.4.5.1 Direction Change in Full-Bridge Mode

In the Full-Bridge Output mode, the P1M1 bit in the CCP1CON register allows the user to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will assume the new direction on the next PWM cycle.

Just before the end of the current PWM period, the modulated outputs (P1B and P1D) are placed in their inactive state, while the unmodulated outputs (P1A and P1C) are switched to drive in the opposite direction. This occurs in the time interval, $4 T_{osc} * (\text{Timer2 Prescale Value})$, before the next PWM period begins. The Timer2 prescaler will be either 1, 4 or 16, depending on the value of the T2CKPS1:T2CKPS0 bits (T2CON<1:0>). During the interval from the switch of the unmodulated outputs to the beginning of the next period, the modulated outputs (P1B and P1D) remain inactive. This relationship is shown in Figure 14-8.

Note that in the Full-Bridge Output mode, the ECCP1 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 14-9 shows an example where the PWM direction changes from forward to reverse at a near 100% duty cycle. At time t_1 , 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 14-7), 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.

PIC18F45J10 FAMILY

FIGURE 14-8: PWM DIRECTION CHANGE

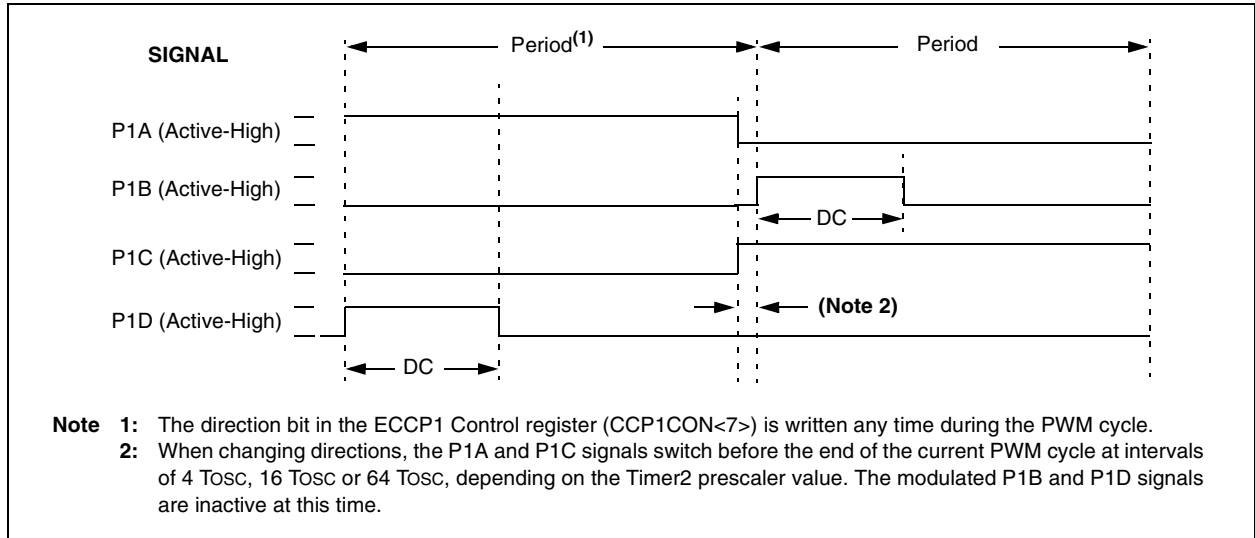
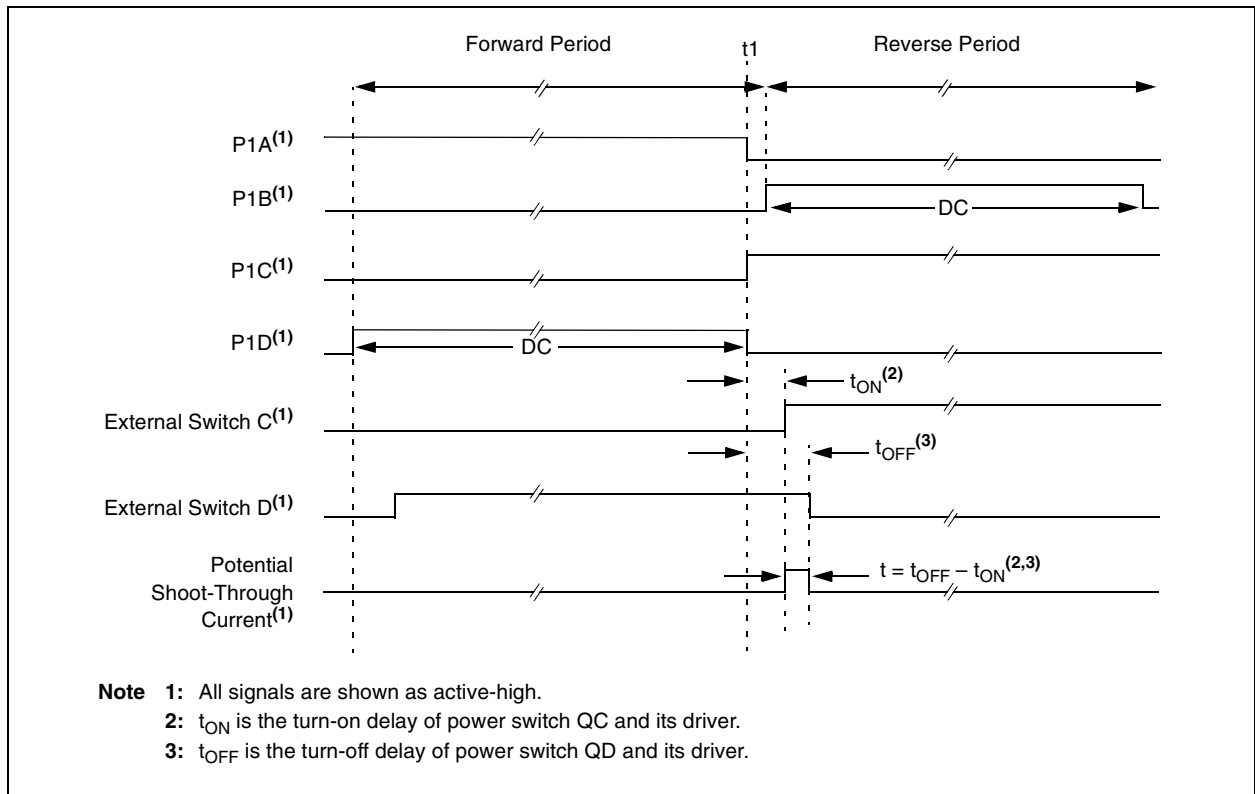


FIGURE 14-9: PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



PIC18F45J10 FAMILY

14.4.6 PROGRAMMABLE DEAD-BAND DELAY

Note: Programmable dead-band delay is not implemented in 28-pin devices with standard CCP modules.

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 nonactive state to the active state. See Figure 14-4 for an illustration. Bits PDC6:PDC0 of the ECCP1DEL register (Register 14-2) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC). These bits are not available in 28-pin devices as the standard CCP module does not support half-bridge operation.

14.4.7 ENHANCED PWM AUTO-SHUTDOWN

When the ECCP1 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 comparator modules, a low level on the Fault input pin (FLT0) 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 low digital signal on FLT0 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 (bits<6:4> of the ECCP1AS register).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits (ECCPAS3:ECCPAS0). 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 (ECCP1AS<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 14-2: ECCP1DEL: PWM DEAD-BAND DELAY 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 ⁽¹⁾ | PDC5 ⁽¹⁾ | PDC4 ⁽¹⁾ | PDC3 ⁽¹⁾ | PDC2 ⁽¹⁾ | PDC1 ⁽¹⁾ | PDC0 ⁽¹⁾ |
| 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 **PDC6:PDC0:** PWM Delay Count bits⁽¹⁾

Delay time, in number of FOSC/4 (4 * TOSC) cycles, between the scheduled and actual time for a PWM signal to transition to active.

Note 1: Reserved on 28-pin devices; maintain these bits 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

PIC18F45J10 FAMILY

REGISTER 14-3: ECCP1AS: 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 ⁽¹⁾ | PSSBD0 ⁽¹⁾ |

bit 7

bit 0

- bit 7 **ECCPASE:** ECCP Auto-Shutdown Event Status bit
 1 = A shutdown event has occurred; ECCP outputs are in shutdown state
 0 = ECCP outputs are operating
- bit 6-4 **ECCPAS2:ECCPAS0:** ECCP Auto-Shutdown Source Select bits
 111 = FLT0, Comparator 1 or Comparator 2
 110 = FLT0 or Comparator 2
 101 = FLT0 or Comparator 1
 100 = FLT0
 011 = Either Comparator 1 or 2
 010 = Comparator 2 output
 001 = Comparator 1 output
 000 = Auto-shutdown is disabled
- bit 3-2 **PSSAC1:PSSAC0:** Pins A and C Shutdown State Control bits
 1x = Pins A and C are tri-state (40/44-pin devices);
 PWM output is tri-state (28-pin devices)
 01 = Drive Pins A and C to '1'
 00 = Drive Pins A and C to '0'
- bit 1-0 **PSSBD1:PSSBD0:** Pins B and D Shutdown State Control bits⁽¹⁾
 1x = Pins B and D tri-state
 01 = Drive Pins B and D to '1'
 00 = Drive Pins B and D to '0'

Note 1: Reserved on 28-pin devices; maintain these bits 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 |

PIC18F45J10 FAMILY

14.4.7.1 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 ECCP1DEL register (ECCP1DEL<7>).

In Shutdown mode with PRSEN = 1 (Figure 14-10), 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 14-11), 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.

14.4.8 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 14-10: PWM AUTO-SHUTDOWN (PRSEN = 1, AUTO-RESTART ENABLED)

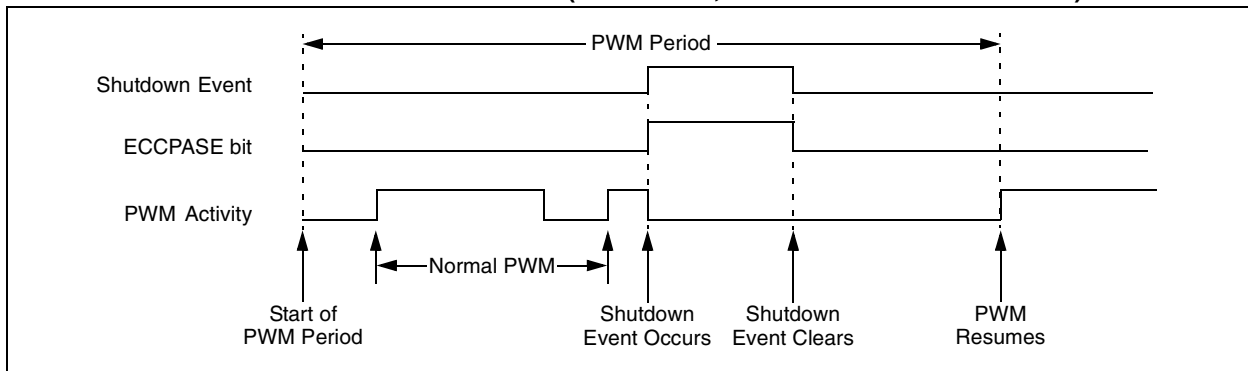
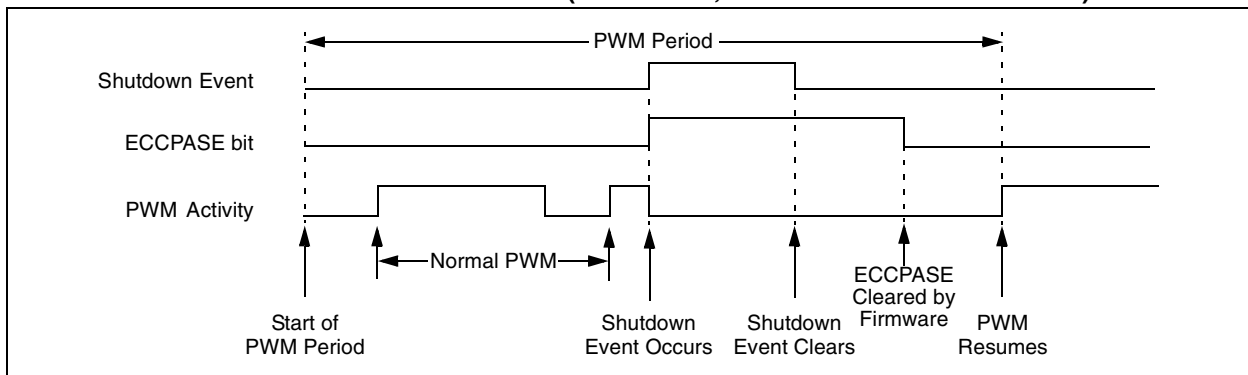


FIGURE 14-11: PWM AUTO-SHUTDOWN (PRSEN = 0, AUTO-RESTART DISABLED)



14.4.9 SETUP FOR PWM OPERATION

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

1. Configure the PWM pins, P1A and P1B (and P1C and P1D, if used), as inputs by setting the corresponding TRIS bits.
2. Set the PWM period by loading the PR2 register.
3. If auto-shutdown is required:
 - Disable auto-shutdown (ECCPASE = 0)
 - Configure source (FLT0, Comparator 1 or Comparator 2)
 - Wait for non-shutdown condition
4. 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.
5. Set the PWM duty cycle by loading the CCPR1L register and CCP1CON<5:4> bits.
6. For Half-Bridge Output mode, set the dead-band delay by loading ECCP1DEL<6:0> with the appropriate value.
7. If auto-shutdown operation is required, load the ECCP1AS register:
 - Select the auto-shutdown sources using the ECCPAS2:ECCPAS0 bits.
 - Select the shutdown states of the PWM output pins using the PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits.
 - Set the ECCPASE bit (ECCP1AS<7>).
 - Configure the comparators using the CMCON register.
 - Configure the comparator inputs as analog inputs.
8. If auto-restart operation is required, set the PRSEN bit (ECCP1DEL<7>).
9. 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>).
10. Enable PWM outputs after a new PWM cycle has started:
 - Wait until TMRn overflows (TMRnIF bit is set).
 - Enable the CCP1/P1A, P1B, P1C and/or P1D pin outputs by clearing the respective TRIS bits.
 - Clear the ECCPASE bit (ECCP1AS<7>).

14.4.10 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 CCP1 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.

14.4.10.1 Operation with Fail-Safe Clock Monitor

If the Fail-Safe Clock Monitor is enabled, a clock failure will force the device into the power-managed RC_RUN 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.

See the previous section for additional details.

14.4.11 EFFECTS OF A RESET

Both Power-on Reset 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.

PIC18F45J10 FAMILY

TABLE 14-3: REGISTERS ASSOCIATED WITH ECCP1 MODULE AND TIMER1

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|--|---------------------|---------------------|---------------------|---------------------|--------------------------------|-----------------------|-----------------------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| RCON | IPEN | — | — | \overline{RI} | \overline{TO} | \overline{PD} | \overline{POR} | \overline{BOR} | 42 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 46 |
| TRISC | PORTC Data Direction Control Register | | | | | | | | 46 |
| TRISD ⁽¹⁾ | PORTD Data Direction Control Register | | | | | | | | 46 |
| TMR1L | Timer1 Register Low Byte | | | | | | | | 44 |
| TMR1H | Timer1 Register High Byte | | | | | | | | 44 |
| T1CON | RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{T1SYN\overline{C}}$ | TMR1CS | TMR1ON | 44 |
| TMR2 | Timer2 Register | | | | | | | | 44 |
| T2CON | — | T2OUTPS3 | T2OUTPS2 | T2OUTPS1 | T2OUTPS0 | TMR2ON | T2CKPS1 | T2CKPS0 | 44 |
| PR2 | Timer2 Period Register | | | | | | | | 44 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte | | | | | | | | 45 |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte | | | | | | | | 45 |
| CCP1CON | P1M1 ⁽¹⁾ | P1M0 ⁽¹⁾ | DC1B1 | DC1B0 | CCP1M3 | CCP1M2 | CCP1M1 | CCP1M0 | 45 |
| ECCP1AS | ECCPASE | ECCPAS2 | ECCPAS1 | ECCPAS0 | PSSAC1 | PSSAC0 | PSSBD1 ⁽¹⁾ | PSSBD0 ⁽¹⁾ | 45 |
| ECCP1DEL | PRSEN | PDC6 ⁽¹⁾ | PDC5 ⁽¹⁾ | PDC4 ⁽¹⁾ | PDC3 ⁽¹⁾ | PDC2 ⁽¹⁾ | PDC1 ⁽¹⁾ | PDC0 ⁽¹⁾ | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used during ECCP operation.

Note 1: These registers and/or bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

15.3.1 REGISTERS

Each MSSP module has four registers for SPI mode operation. These are:

- MSSP Control Register 1 (SSPxCON1)
- MSSP Status Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSP Shift Register (SSPxSR) – Not directly accessible

SSPxCON1 and SSPxSTAT are the control and status registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

SSPxSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPxSR and SSPxBUF together create a double-buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

REGISTER 15-1: SSPxSTAT: MSSPx 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/ \bar{A} | P | S | R/ \bar{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 Select bit
 1 = Transmit occurs on transition from active to Idle clock state
 0 = Transmit occurs on transition from Idle to active clock state
Note: Polarity of clock state is set by the CKP bit (SSPxCON1<4>).
- bit 5 **D/ \bar{A} :** Data/Address bit
 Used in I²C mode only.
- bit 4 **P:** Stop bit
 Used in I²C mode only. This bit is cleared when the MSSP module is disabled, SSPEN is cleared.
- bit 3 **S:** Start bit
 Used in I²C mode only.
- bit 2 **R/ \bar{W} :** Read/Write Information bit
 Used in I²C mode only.
- bit 1 **UA:** Update Address bit
 Used in I²C mode only.
- bit 0 **BF:** Buffer Full Status bit (Receive mode only)
 1 = Receive complete, SSPxBUF is full
 0 = Receive not complete, SSPxBUF 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 |

PIC18F45J10 FAMILY

REGISTER 15-2: SSPxCON1: MSSPx 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 SSPxBUF 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 SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. The user must read the SSPxBUF, 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 SSPxBUF register.

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit

- 1 = Enables serial port and configures SCKx, SDOx, SDIx and \overline{SSx} as serial port pins
- 0 = Disables serial port and configures these pins as I/O port pins

Note: When enabled, 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:** Master Synchronous Serial Port Mode Select bits

- 0101 = SPI Slave mode, clock = SCKx pin, \overline{SSx} pin control disabled, \overline{SSx} can be used as I/O pin
- 0100 = SPI Slave mode, clock = SCKx pin, \overline{SSx} 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 I²C 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 |

PIC18F45J10 FAMILY

15.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPxCON1<5:0> and SSPxSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCKx is the clock output)
- Slave mode (SCKx is the clock input)
- Clock Polarity (Idle state of SCKx)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCKx)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

Each MSSP consists of a transmit/receive shift register (SSPxSR) and a buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full detect bit BF (SSPxSTAT<0>) and the interrupt flag bit SSPxIF are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the

SSPxBUF register during transmission/reception of data will be ignored and the Write Collision detect bit, WCOL (SSPxCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPxBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF (SSPxSTAT<0>), indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF 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 SSPxBUF 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 15-1 shows the loading of the SSP1BUF (SSP1SR) for data transmission.

The SSPxSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register. Additionally, the SSPxSTAT register indicates the various status conditions.

EXAMPLE 15-1: LOADING THE SSP1BUF (SSP1SR) REGISTER

| | | | |
|------|-------|--------------|--|
| LOOP | BTFSS | SSP1STAT, BF | ;Has data been received (transmit complete)? |
| | BRA | LOOP | ;No |
| | MOVF | SSP1BUF, W | ;WREG reg = contents of SSP1BUF |
| | MOVWF | RXDATA | ;Save in user RAM, if data is meaningful |
| | MOVF | TXDATA, W | ;W reg = contents of TXDATA |
| | MOVWF | SSP1BUF | ;New data to xmit |

PIC18F45J10 FAMILY

15.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx. The master determines when the slave (Processor 2, Figure 15-2) will broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDOx output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDIx pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF 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.

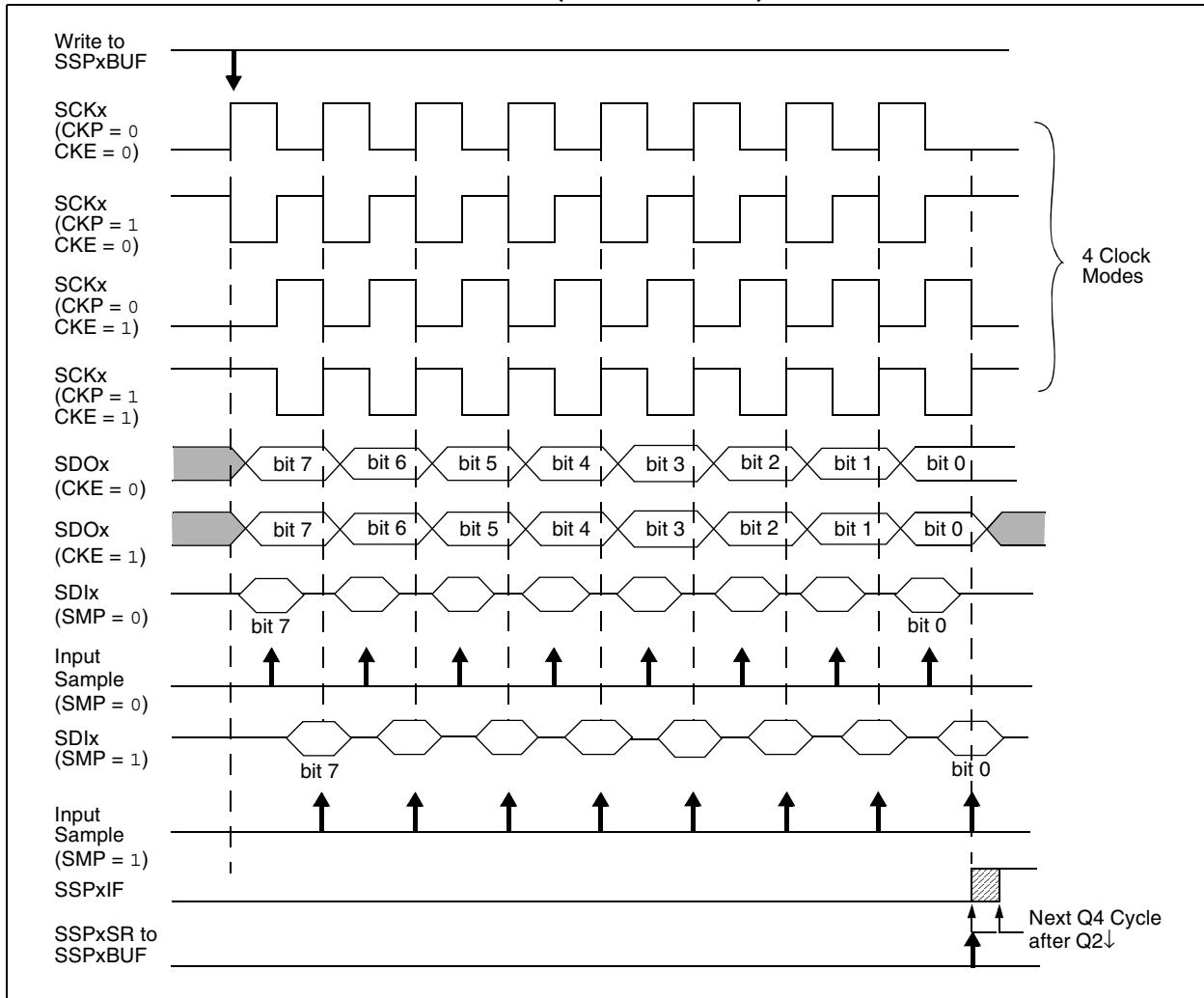
The clock polarity is selected by appropriately programming the CKP bit (SSPxCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 15-3, Figure 15-5 and Figure 15-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- Fosc/4 (or Tcy)
- Fosc/16 (or 4 • Tcy)
- Fosc/64 (or 16 • Tcy)
- Timer2 output/2

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

Figure 15-3 shows the waveforms for Master mode. When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 15-3: SPI™ MODE WAVEFORM (MASTER MODE)



15.3.6 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCKx pin. The Idle state is determined by the CKP bit (SSPxCON1<4>).

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device will wake-up from Sleep.

15.3.7 SLAVE SELECT SYNCHRONIZATION

The \overline{SSx} pin allows a Synchronous Slave mode. The SPI must be in Slave mode with \overline{SSx} pin control enabled (SSPxCON1<3:0> = 04h). When the \overline{SSx} pin is low, transmission and reception are enabled and the

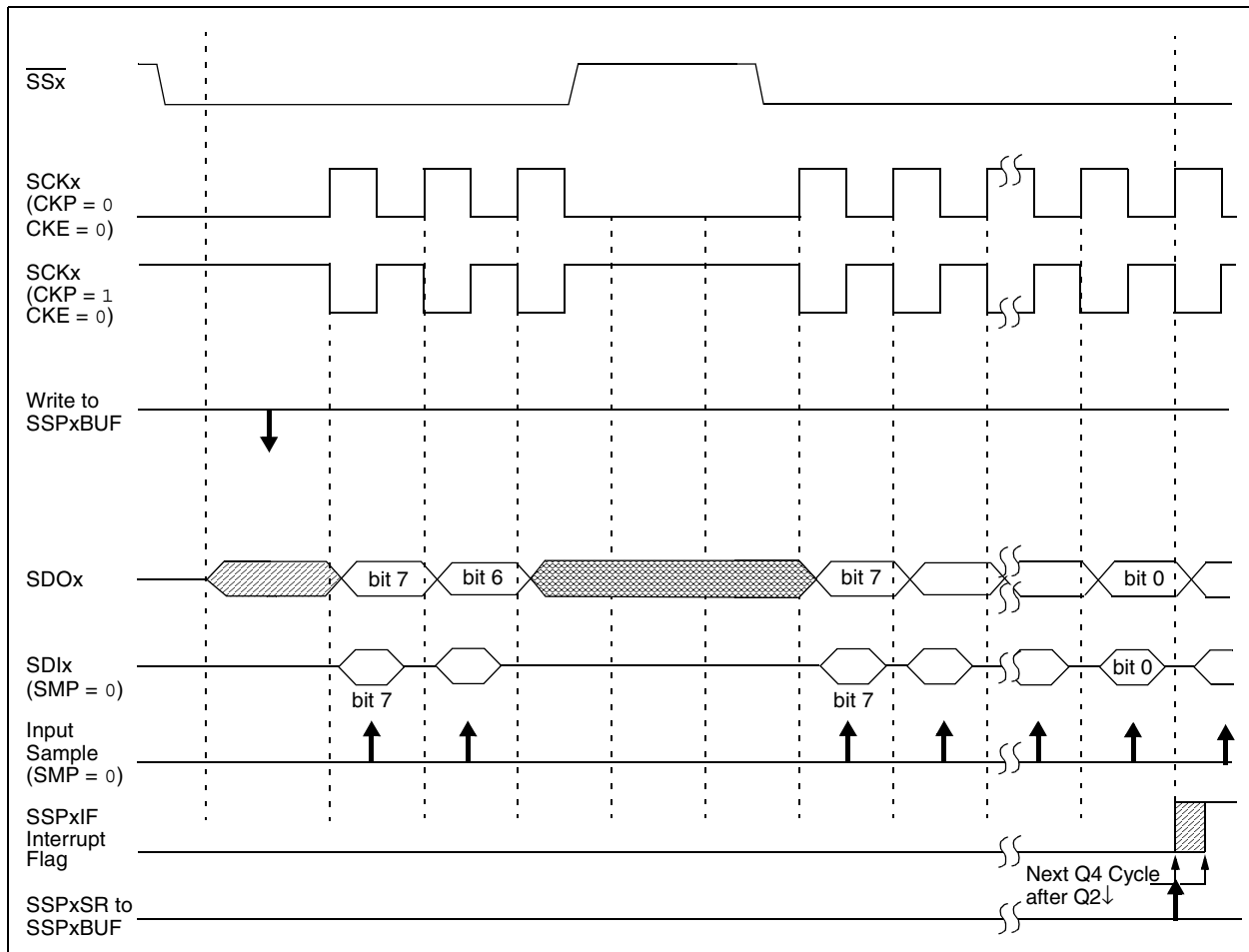
SDOx pin is driven. When the \overline{SSx} pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

- Note 1:** When the SPI is in Slave mode with \overline{SSx} pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the \overline{SSx} pin is set to VDD.
- 2:** If the SPI is used in Slave mode with CKE set, then the \overline{SSx} pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the \overline{SSx} pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDOx pin can be connected to the SDIx pin. When the SPI needs to operate as a receiver, the SDOx pin can be configured as an input. This disables transmissions from the SDOx. The SDIx can always be left as an input (SDIx function) since it cannot create a bus conflict.

FIGURE 15-4: SLAVE SYNCHRONIZATION WAVEFORM



PIC18F45J10 FAMILY

FIGURE 15-5: SPI™ MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

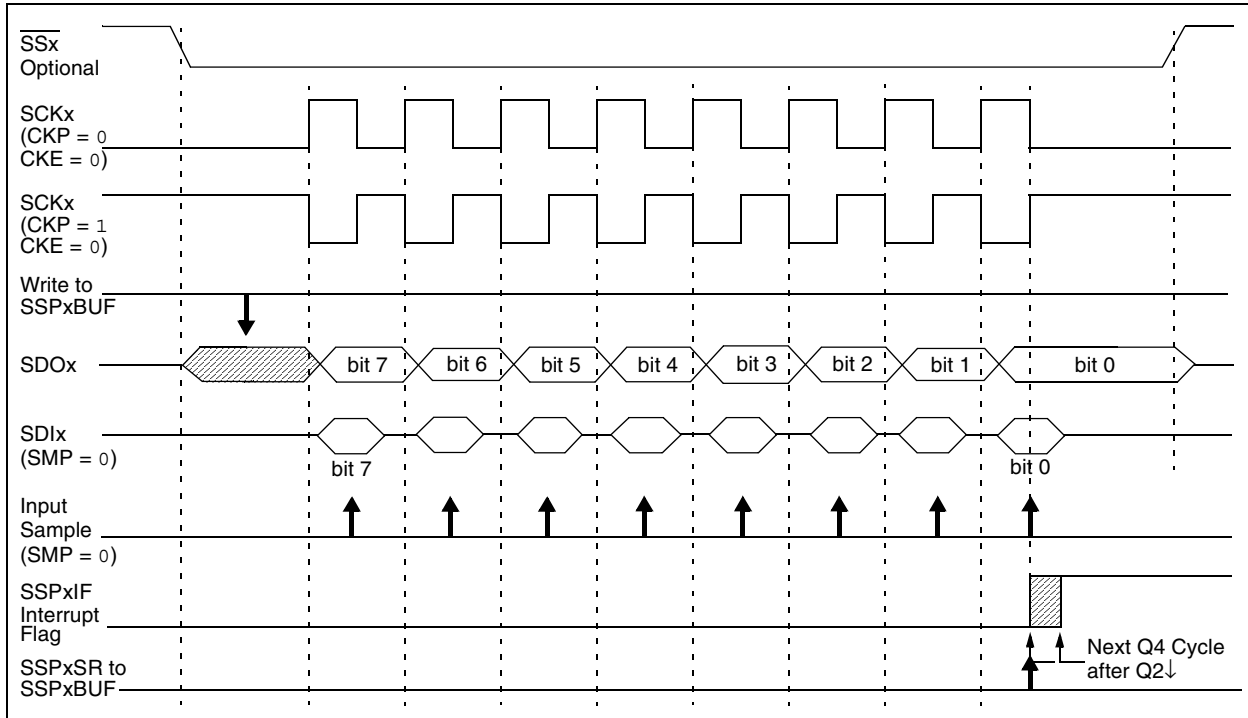
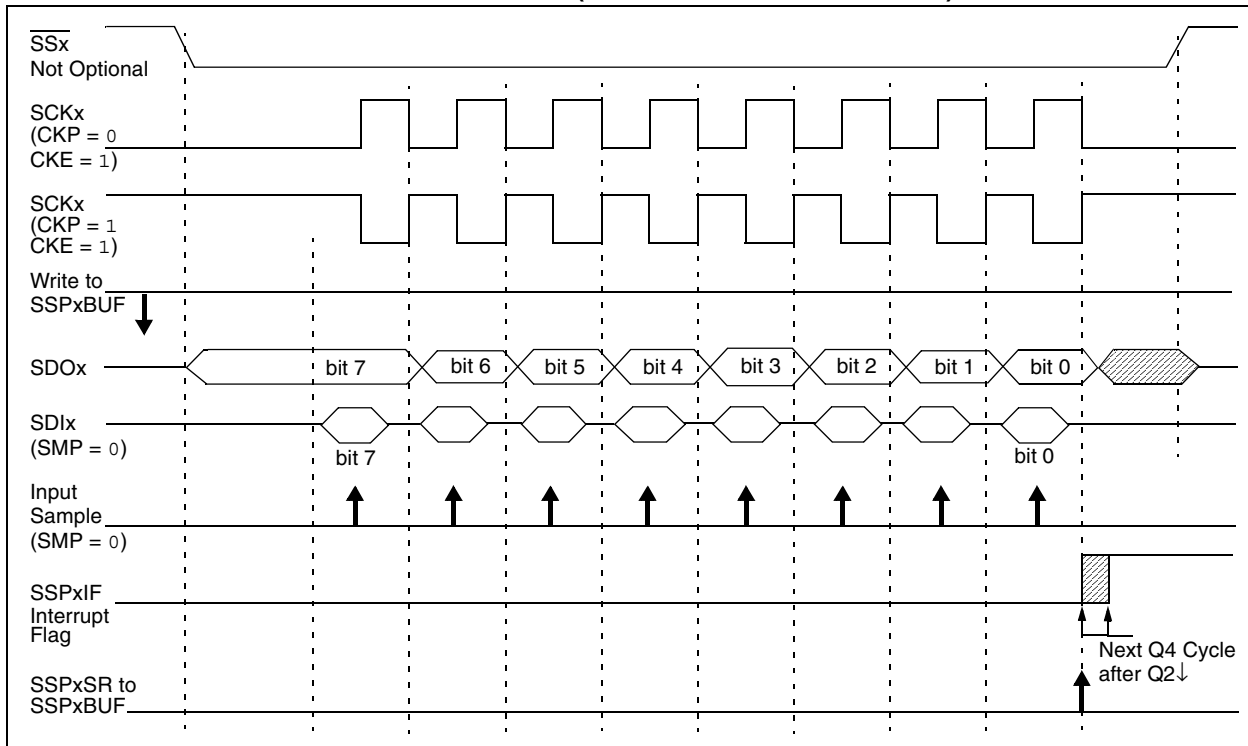


FIGURE 15-6: SPI™ MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



15.3.8 OPERATION IN POWER-MANAGED MODES

In SPI Master mode, module clocks may be operating at a different speed than when in full power mode; in the case of Sleep mode, all clocks are halted.

In Idle modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTOSC source. See **Section 2.6 “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 Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle 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 remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI 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 enabled, will wake the device.

15.3.9 EFFECTS OF A RESET

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

15.3.10 BUS MODE COMPATIBILITY

Table 15-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 15-1: 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.

15.3.11 SPI CLOCK SPEED AND MODULE INTERACTIONS

Because MSSP1 and MSSP2 are independent modules, they can operate simultaneously at different data rates. Setting the SSPM3:SSPM0 bits of the SSPxCON1 register determines the rate for the corresponding module.

An exception is when both modules use Timer2 as a time base in Master mode. In this instance, any changes to the Timer2 operation will affect both MSSP modules equally. If different bit rates are required for each module, the user should select one of the other three time base options for one of the modules.

PIC18F45J10 FAMILY

TABLE 15-2: REGISTERS ASSOCIATED WITH SPI™ OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|--|-----------|--------------|--------|--------|--------------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| PIR3 | SSP2IF | BCL2IF | — | — | — | — | — | — | 45 |
| PIE3 | SSP2IE | BCL2IE | — | — | — | — | — | — | 45 |
| IPR3 | SSP2IP | BCL2IP | — | — | — | — | — | — | 45 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 46 |
| TRISD ⁽¹⁾ | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 46 |
| SSP1BUF | MSSP1 Receive Buffer/Transmit Register | | | | | | | | 44 |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 44 |
| SSP1STAT | SMP | CKE | D/ \bar{A} | P | S | R/ \bar{W} | UA | BF | 44 |
| SSP2BUF | MSSP2 Receive Buffer/Transmit Register | | | | | | | | 46 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 46 |
| SSP2STAT | SMP | CKE | D/ \bar{A} | P | S | R/ \bar{W} | UA | BF | 46 |

Legend: Shaded cells are not used by the MSSP module in SPI™ mode.

Note 1: These registers and/or bits are not implemented on 28-pin devices and should be read as '0'.

15.4 I²C Mode

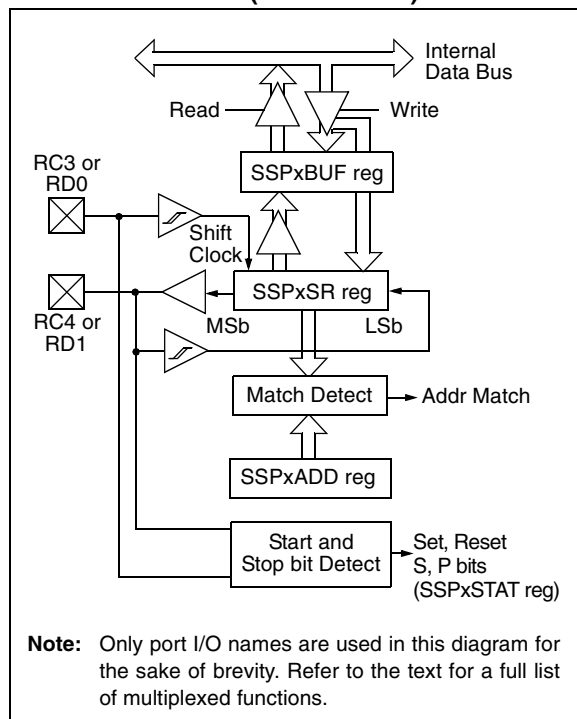
The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial clock (SCLx) – RC3/SCK1/SCL1 or RD0/PSP0/SCK2/SCL2
- Serial data (SDAx) – RC4/SDI1/SDA1 or RD1/PSP1/SDI2/SDA2

The user must configure these pins as inputs by setting the TRISC<4:3> or TRISD<1:0> bits.

FIGURE 15-7: MSSP BLOCK DIAGRAM (I²C™ MODE)



15.4.1 REGISTERS

The MSSP module has six registers for I²C operation. These are:

- MSSP Control Register 1 (SSPxCON1)
- MSSP Control Register 2 (SSPxCON2)
- MSSP Status Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSP Shift Register (SSPxSR) – Not directly accessible
- MSSP Address Register (SSPxADD)

SSPxCON1, SSPxCON2 and SSPxSTAT are the control and status registers in I²C mode operation. The SSPxCON1 and SSPxCON2 registers are readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper 2 bits of the SSPxSTAT are read/write.

SSPxSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from.

SSPxADD register holds the slave device address when the MSSP is configured in I²C Slave mode. When the MSSP is configured in Master mode, the lower seven bits of SSPxADD act as the Baud Rate Generator reload value.

In receive operations, SSPxSR and SSPxBUF together create a double-buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

PIC18F45J10 FAMILY

REGISTER 15-3: SSPxSTAT: MSSPx 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 \bar{A} | P | S | R \bar{W} | UA | BF |
| bit 7 | | | | | | bit 0 | |

- bit 7 **SMP:** Slew Rate Control bit
In Master or Slave mode:
 1 = Slew rate control disabled for Standard Speed mode (100 kHz and 1 MHz)
 0 = Slew rate control enabled for High-Speed mode (400 kHz)
- bit 6 **CKE:** SMBus Select bit
In Master or Slave mode:
 1 = Enable SMBus specific inputs
 0 = Disable SMBus specific inputs
- bit 5 **D \bar{A} :** Data/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 and when SSPEN is cleared.
- 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 and when SSPEN is cleared.
- bit 2 **R \bar{W} :** Read/Write Information bit (I²C mode only)
In Slave mode:
 1 = Read
 0 = Write
Note: This bit holds the R \bar{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 ACK bit.
In Master mode:
 1 = Transmit is in progress
 0 = Transmit is not in progress
Note: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Active mode.
- bit 1 **UA:** Update Address bit (10-bit Slave mode only)
 1 = Indicates that the user needs to update the address in the SSPxADD register
 0 = Address does not need to be updated
- bit 0 **BF:** Buffer Full Status bit
In Transmit mode:
 1 = SSPxBUF is full
 0 = SSPxBUF is empty
In Receive mode:
 1 = SSPxBUF is full (does not include the \overline{ACK} and Stop bits)
 0 = SSPxBUF is empty (does not include the \overline{ACK} and Stop bits)

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

PIC18F45J10 FAMILY

REGISTER 15-4: SSPxCON1: MSSPx CONTROL REGISTER 1 (I²C™ 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 SSPxBUF register was attempted while the I²C conditions were not valid for a transmission to be started (must be cleared in software)
 0 = No collision
In Slave Transmit mode:
 1 = The SSPxBUF 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 SSPxBUF 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:** Master Synchronous Serial Port Enable bit
 1 = Enables the serial port and configures the SDAx and SCLx pins as the serial port pins
 0 = Disables serial port and configures these pins as I/O port pins
Note: When enabled, the SDAx and SCLx pins must be properly configured as input or output.
- bit 4 **CKP:** SCKx 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:** Master Synchronous Serial Port Mode Select bits
 1111 = I²C Slave mode, 10-bit address with Start and Stop bit interrupts enabled
 1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled
 1011 = I²C Firmware Controlled Master mode (Slave Idle)
 1000 = I²C Master mode, clock = FOSC/(4 * (SSPxADD + 1))
 0111 = I²C Slave mode, 10-bit address
 0110 = I²C Slave mode, 7-bit address
 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 |

PIC18F45J10 FAMILY

REGISTER 15-5: SSPxCON2: MSSPx CONTROL REGISTER 2 (I²C™ 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 ⁽¹⁾ | RCEN ⁽¹⁾ | PEN ⁽¹⁾ | RSEN ⁽¹⁾ | SEN ⁽¹⁾ | |
| 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 SSPxSR
 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⁾
 1 = Initiate Acknowledge sequence on SDAx and SCLx pins and transmit ACKDT data bit. Automatically cleared by hardware.
 0 = Acknowledge sequence Idle
- bit 3 **RCEN:** Receive Enable bit (Master mode only)⁽¹⁾
 1 = Enables Receive mode for I²C
 0 = Receive Idle
- bit 2 **PEN:** Stop Condition Enable bit (Master mode only)⁽¹⁾
 1 = Initiate Stop condition on SDAx and SCLx pins. Automatically cleared by hardware.
 0 = Stop condition Idle
- bit 1 **RSEN:** Repeated Start Condition Enable bit (Master mode only)⁽¹⁾
 1 = Initiate Repeated Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
 0 = Repeated Start condition Idle
- bit 0 **SEN:** Start Condition Enable/Stretch Enable bit⁽¹⁾
In Master mode:
 1 = Initiate Start condition on SDAx and SCLx 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 1:** For bits ACKEN, RCEN, PEN, RSEN, SEN: If the I²C module is not in the Idle mode, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF 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 |

15.4.2 OPERATION

The MSSP module functions are enabled by setting the MSSP Enable bit, SSPEN (SSPxCON1<5>).

The SSPxCON1 register allows control of the I²C operation. Four mode selection bits (SSPxCON1<3:0>) allow one of the following I²C modes to be selected:

- I²C Master mode, clock = (Fosc/4) x (SSPxADD + 1)
- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address) with Start and Stop bit interrupts enabled
- I²C Slave mode (10-bit address) with Start and Stop bit interrupts enabled
- I²C Firmware Controlled Master mode, slave is Idle

Selection of any I²C mode, with the SSPEN bit set, forces the SCLx and SDAx pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC or TRISD bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCLx and SDAx pins.

15.4.3 SLAVE MODE

In Slave mode, the SCLx and SDAx pins must be configured as inputs (TRISC<4:3> or TRISD<1:0> set). The MSSP module will override the input state with the output data when required (slave-transmitter).

The I²C 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 (ACK) pulse and load the SSPxBUF register with the received value currently in the SSPxSR register.

Any combination of the following conditions will cause the MSSP module not to give this ACK pulse:

- The Buffer Full bit, BF (SSPxSTAT<0>), was set before the transfer was received.
- The overflow bit, SSPOV (SSPxCON1<6>), was set before the transfer was received.

In this case, the SSPxSR register value is not loaded into the SSPxBUF, but bit SSPxIF is set. The BF bit is cleared by reading the SSPxBUF register, while bit SSPOV is cleared through software.

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

15.4.3.1 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 SSPxSR register. All incoming bits are sampled with the rising edge of the clock (SCLx) line. The value of register SSPxSR<7:1> is compared to the value of the SSPxADD register. The address is compared on the falling edge of the eighth clock (SCLx) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPxSR register value is loaded into the SSPxBUF register.
2. The Buffer Full bit, BF, is set.
3. An ACK pulse is generated.
4. The MSSP Interrupt Flag bit, SSPxIF, is set (and interrupt is generated, if enabled) on the falling edge of the ninth SCLx 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/W (SSPxSTAT<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 SSPxIF, BF and UA (SSPxSTAT<1>) are set).
2. Update the SSPxADD register with second (low) byte of address (clears bit UA and releases the SCLx line).
3. Read the SSPxBUF register (clears bit BF) and clear flag bit SSPxIF.
4. Receive second (low) byte of address (bits SSPxIF, BF and UA are set).
5. Update the SSPxADD register with the first (high) byte of address. If match releases SCLx line, this will clear bit UA.
6. Read the SSPxBUF register (clears bit BF) and clear flag bit SSPxIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits SSPxIF and BF are set).
9. Read the SSPxBUF register (clears bit BF) and clear flag bit SSPxIF.

PIC18F45J10 FAMILY

15.4.3.2 Reception

When the $\overline{R/W}$ bit of the address byte is clear and an address match occurs, the $\overline{R/W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and the SDAx 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 (SSPxSTAT<0>) is set, or bit SSPOV (SSPxCON1<6>) is set.

An MSSP interrupt is generated for each data transfer byte. The interrupt flag bit, SSPxIF, must be cleared in software. The SSPxSTAT register is used to determine the status of the byte.

If SEN is enabled (SSPxCON2<0> = 1), SCKx/SCLx (RC3 or RD0) will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPxCON1<4>). See **Section 15.4.4 “Clock Stretching”** for more details.

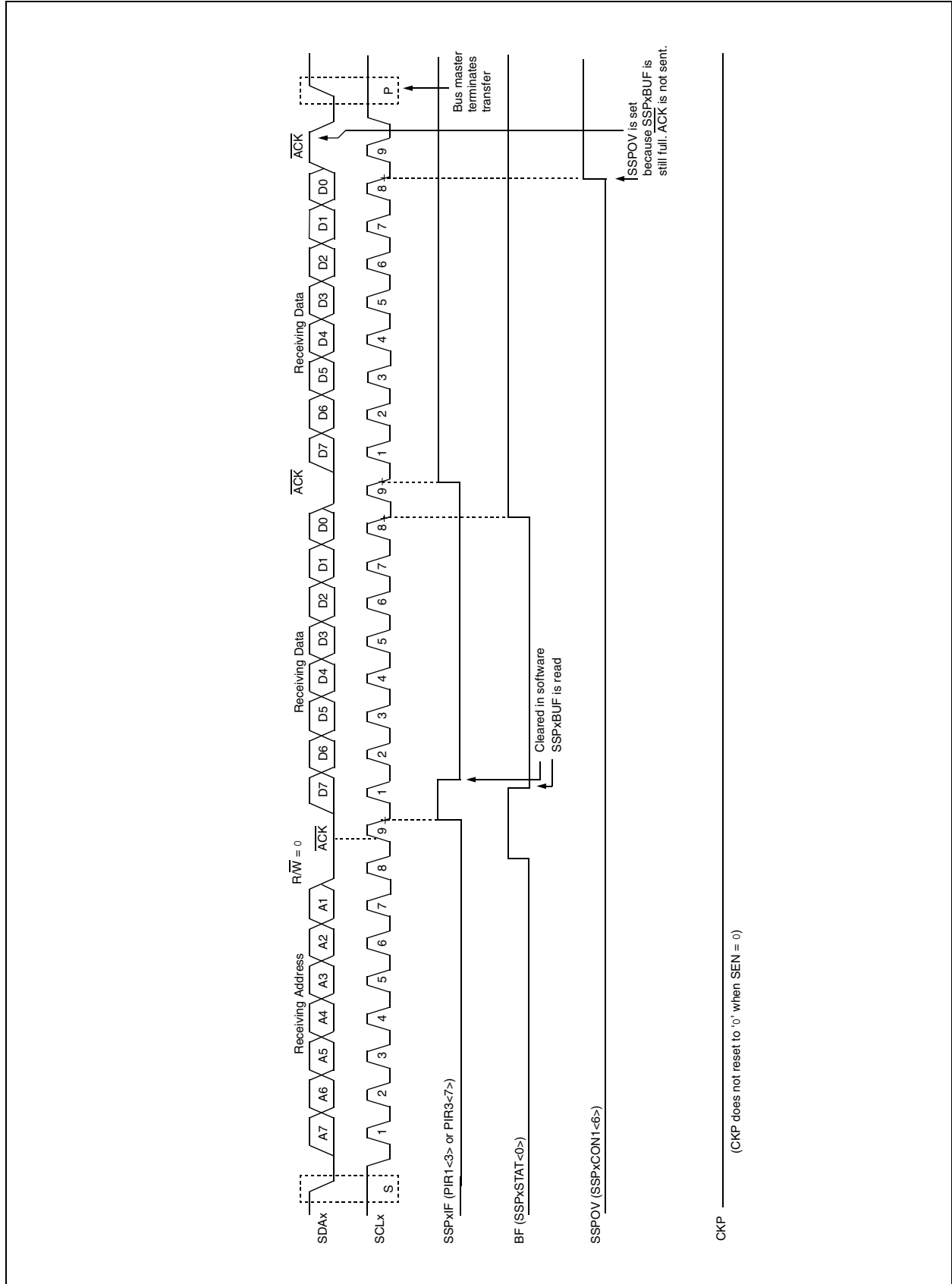
15.4.3.3 Transmission

When the $\overline{R/W}$ bit of the incoming address byte is set and an address match occurs, the $\overline{R/W}$ bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register. The \overline{ACK} pulse will be sent on the ninth bit and pin RC3 or RD6 is held low, regardless of SEN (see **Section 15.4.4 “Clock Stretching”** for more details). 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 SSPxBUF register which also loads the SSPxSR register. Then pin RC3 or RD0 should be enabled by setting bit, CKP (SSPxCON1<4>). The eight data bits are shifted out on the falling edge of the SCLx input. This ensures that the SDAx signal is valid during the SCLx high time (Figure 15-9).

The \overline{ACK} pulse from the master-receiver is latched on the rising edge of the ninth SCLx input pulse. If the SDAx 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 SSPxSTAT register) and the slave monitors for another occurrence of the Start bit. If the SDAx line was low (\overline{ACK}), the next transmit data must be loaded into the SSPxBUF register. Again, pin RC3 or RD0 must be enabled by setting bit CKP.

An MSSP interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared in software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

FIGURE 15-8: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)



PIC18F45J10 FAMILY

FIGURE 15-9: I²C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

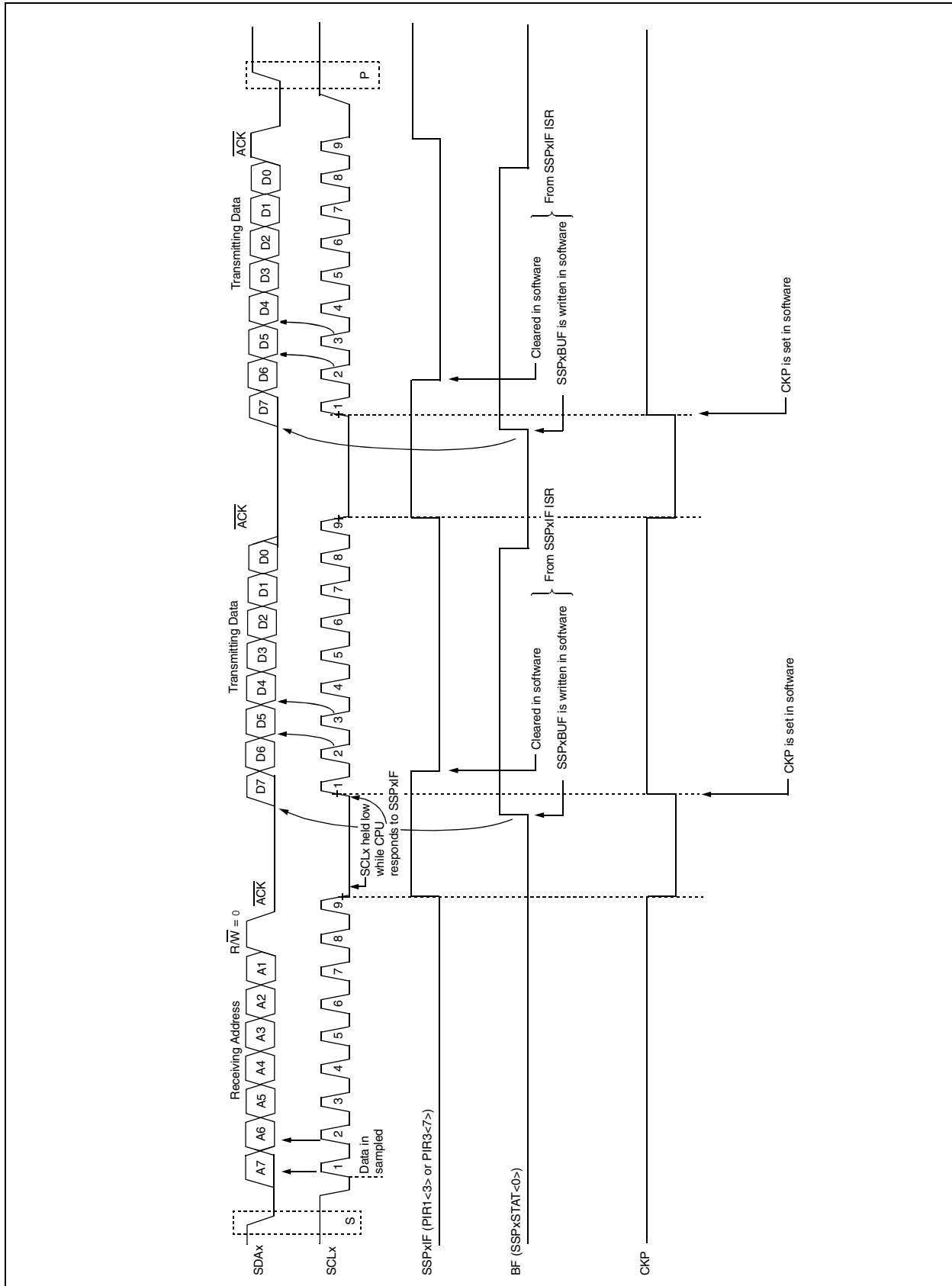
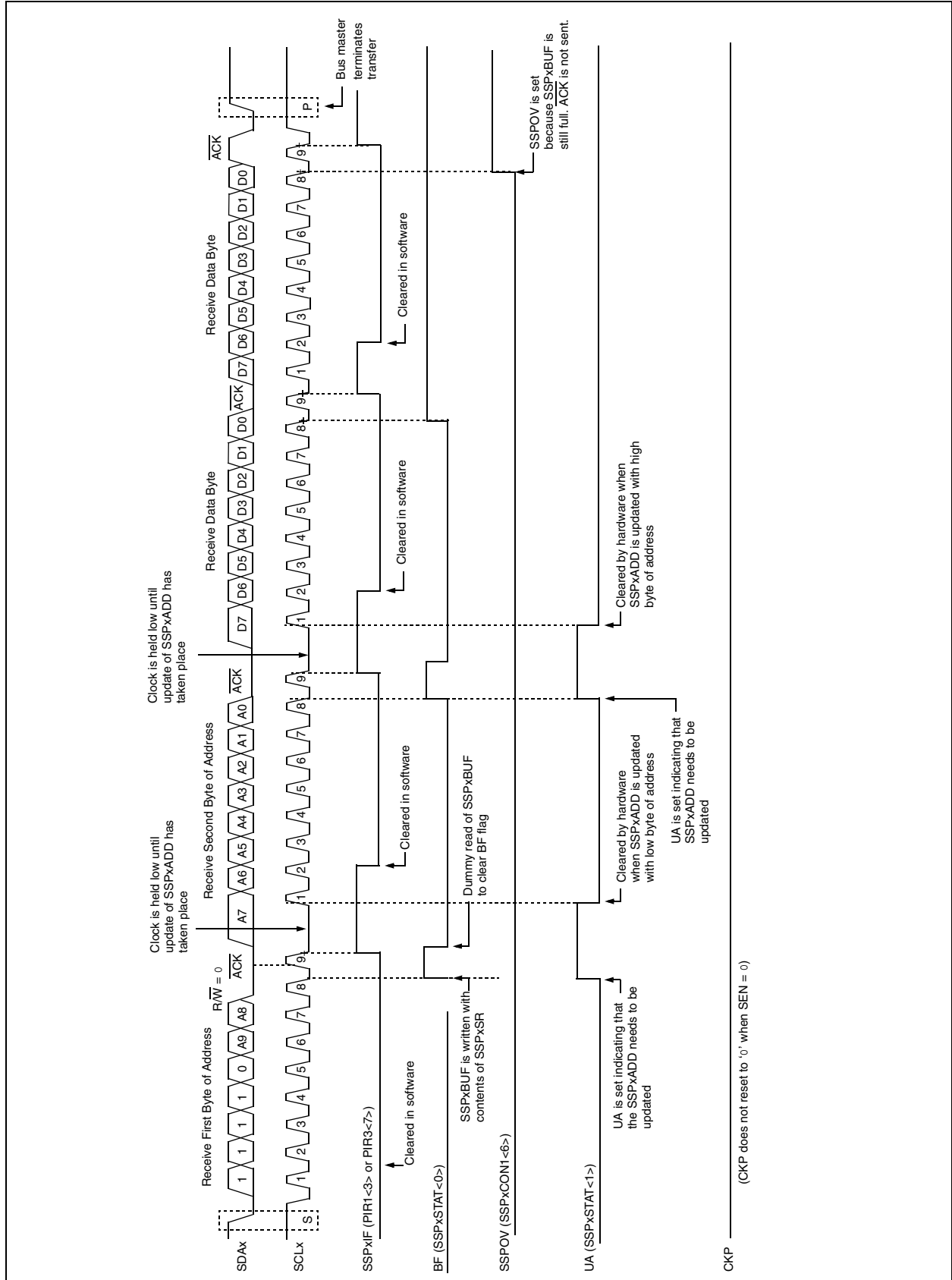
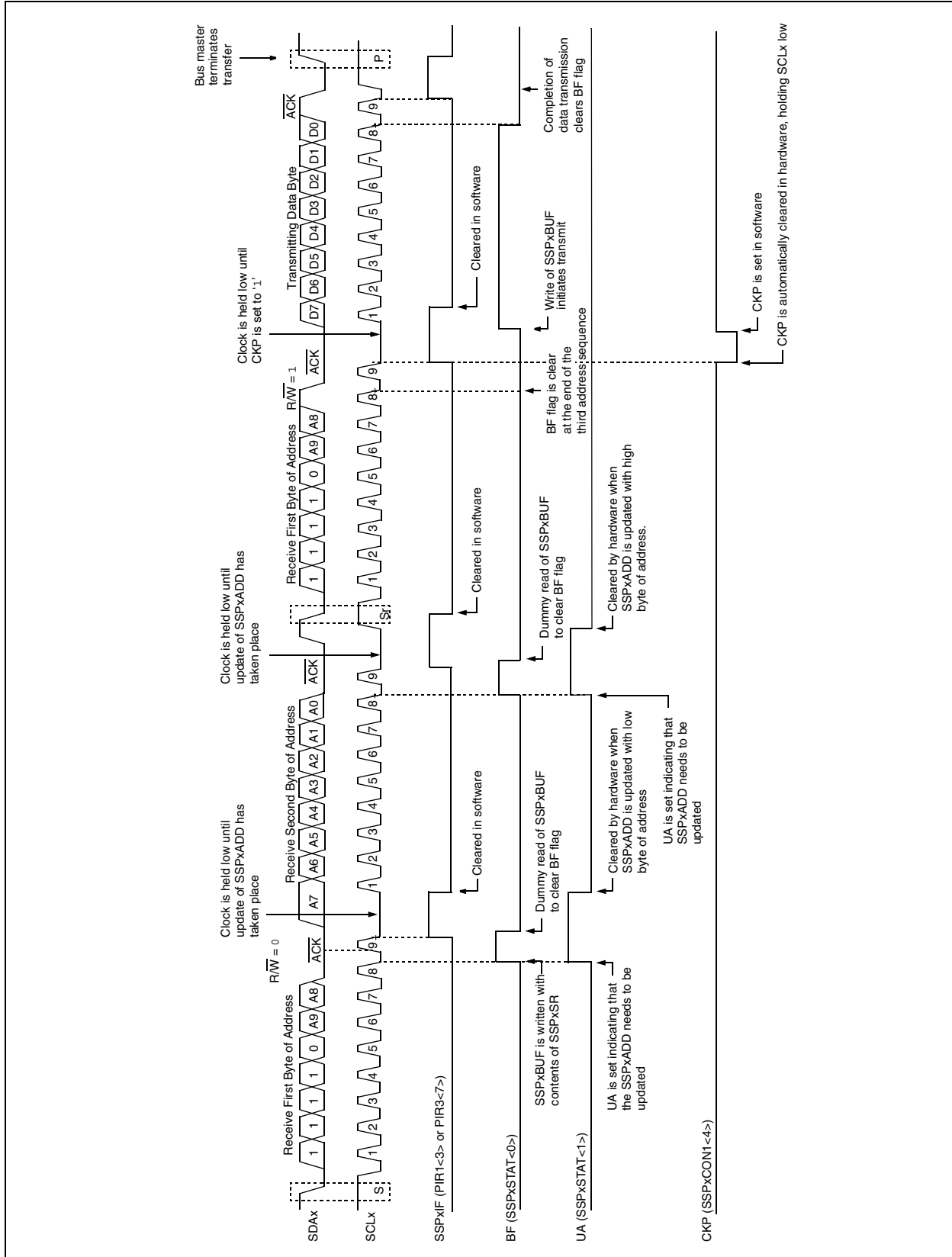


FIGURE 15-10: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)



PIC18F45J10 FAMILY

FIGURE 15-11: I²C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



15.4.4 CLOCK STRETCHING

Both 7-bit and 10-bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPxCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCLx pin to be held low at the end of each data receive sequence.

15.4.4.1 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 SSPxCON1 register is automatically cleared, forcing the SCLx output to be held low. The CKP being cleared to '0' will assert the SCLx line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and read the contents of the SSPxBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 15-13).

Note 1: If the user reads the contents of the SSPxBUF 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.

15.4.4.2 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 CKP 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/W bit cleared to '0'. The release of the clock line occurs upon updating SSPxADD. 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 SSPxADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPxBUF 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.

15.4.4.3 Clock Stretching for 7-bit Slave Transmit Mode

The 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 SCLx line low, the user has time to service the ISR and load the contents of the SSPxBUF before the master device can initiate another transmit sequence (see Figure 15-9).

Note 1: If the user loads the contents of SSPxBUF, 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.

15.4.4.4 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/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 15-11).

PIC18F45J10 FAMILY

15.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCLx output is forced to '0'. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external I²C master device has

already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I²C bus have deasserted SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 15-12).

FIGURE 15-12: CLOCK SYNCHRONIZATION TIMING

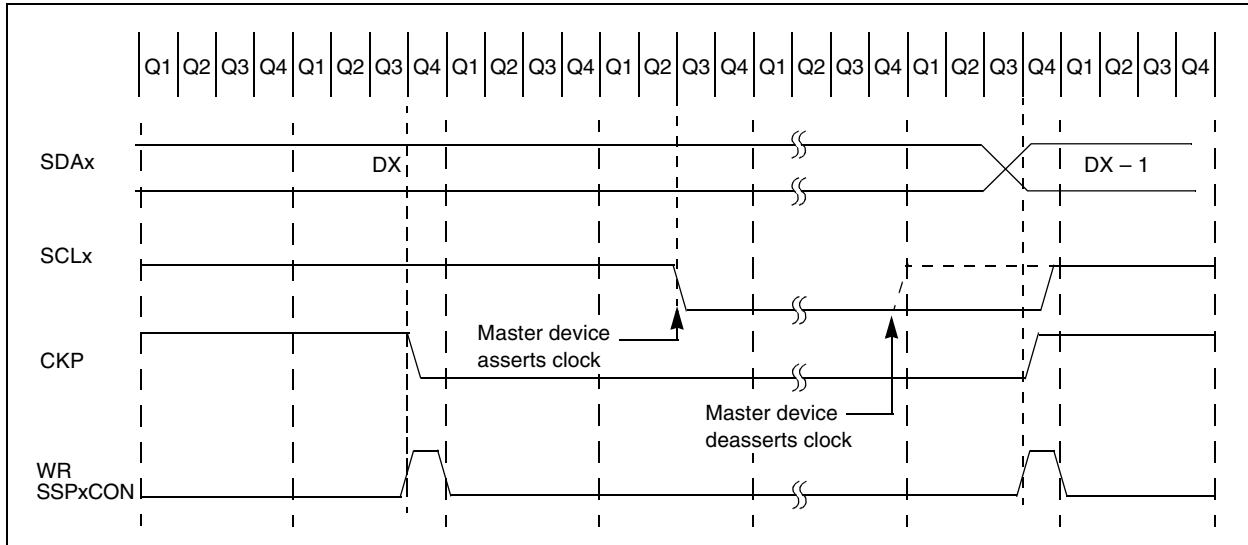
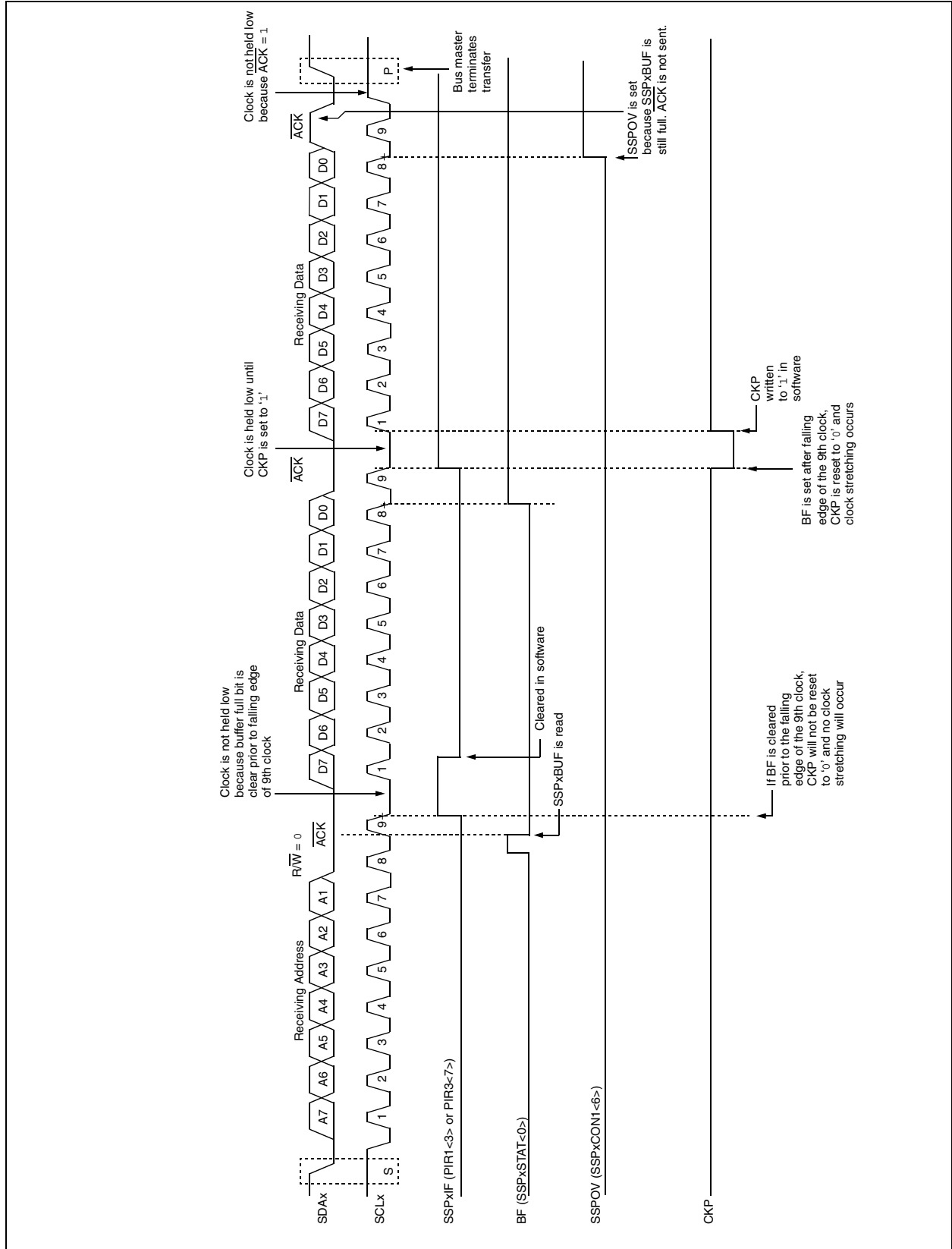
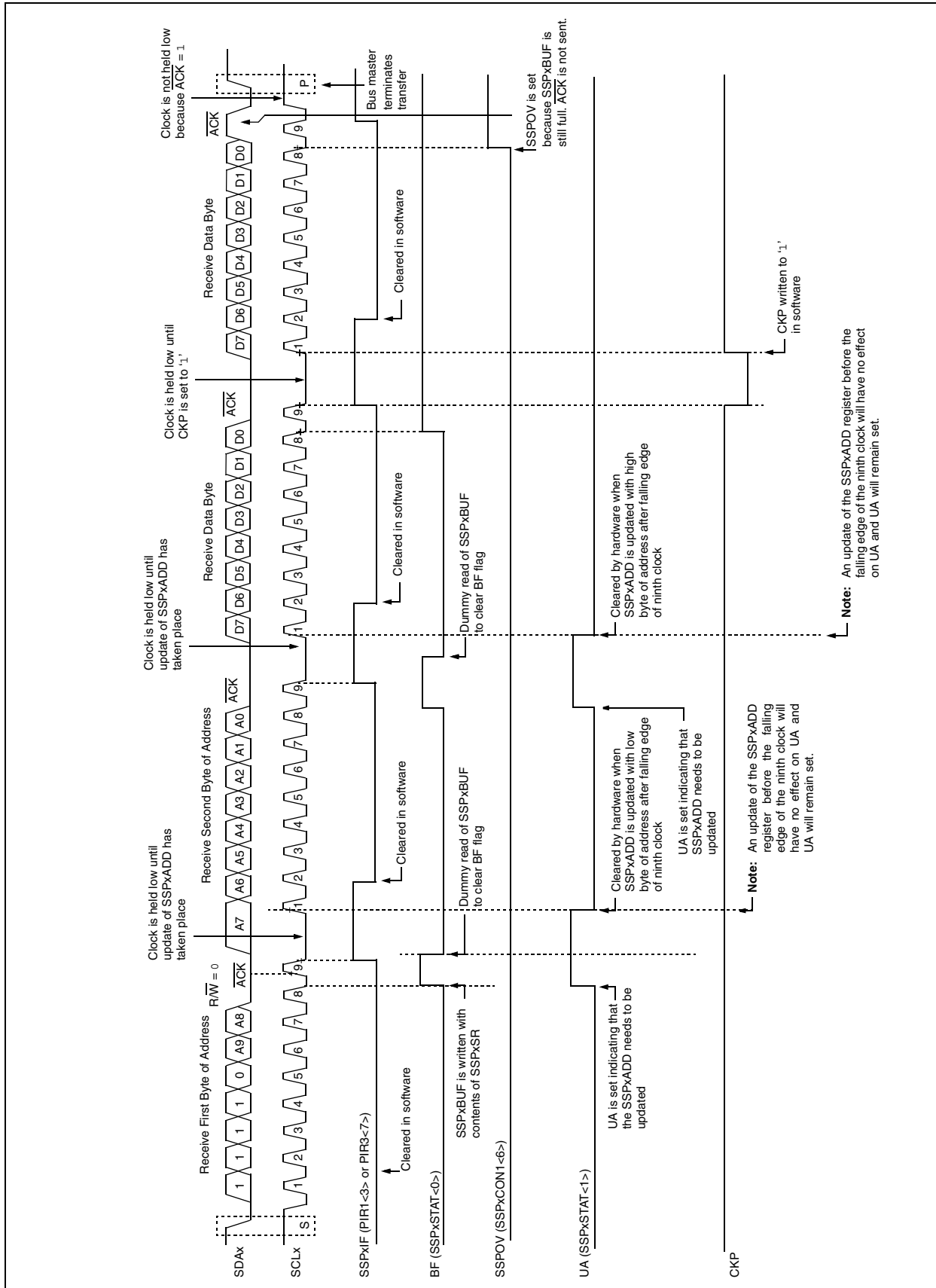


FIGURE 15-13: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)



PIC18F45J10 FAMILY

FIGURE 15-14: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)



15.4.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I²C 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 I²C protocol. It consists of all '0's with $R/\overline{W} = 0$.

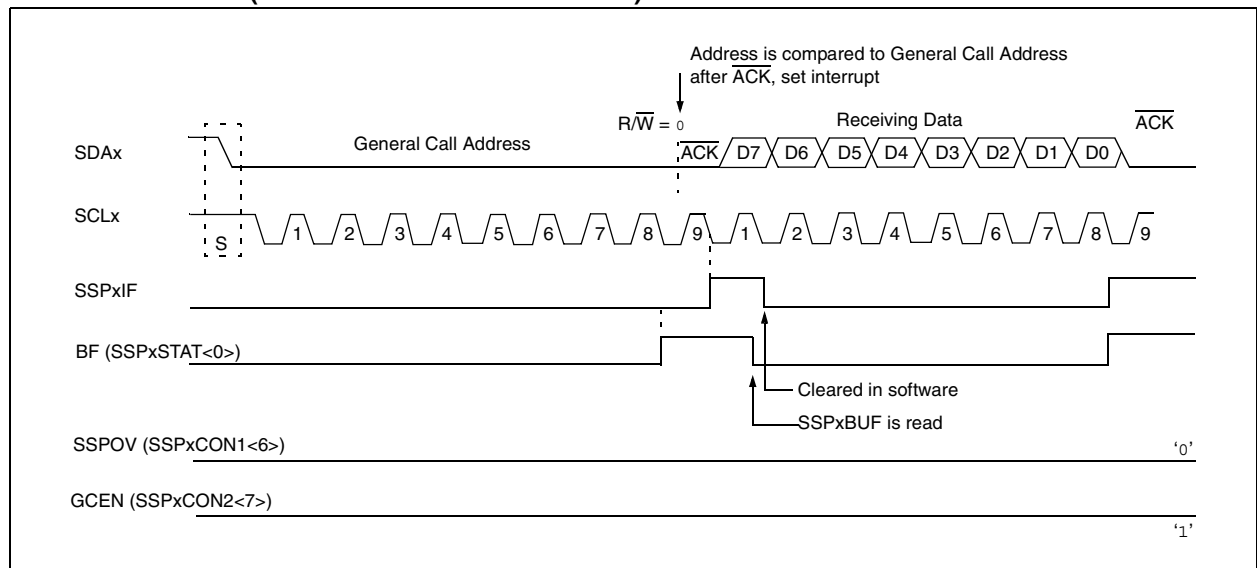
The general call address is recognized when the General Call Enable bit, GCEN, is enabled (SSPxCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPxSR and the address is compared against the SSPxADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPxSR is transferred to the SSPxBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit (\overline{ACK} bit), the SSPxIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPxBUF. The value can be used to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPxADD is required to be updated for the second half of the address to match and the UA bit is set (SSPxSTAT<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 15-15).

FIGURE 15-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESS MODE)



PIC18F45J10 FAMILY

15.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPxCON1 and by setting the SSPEN bit. In Master mode, the SCLx and SDAx 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 I²C 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 I²C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

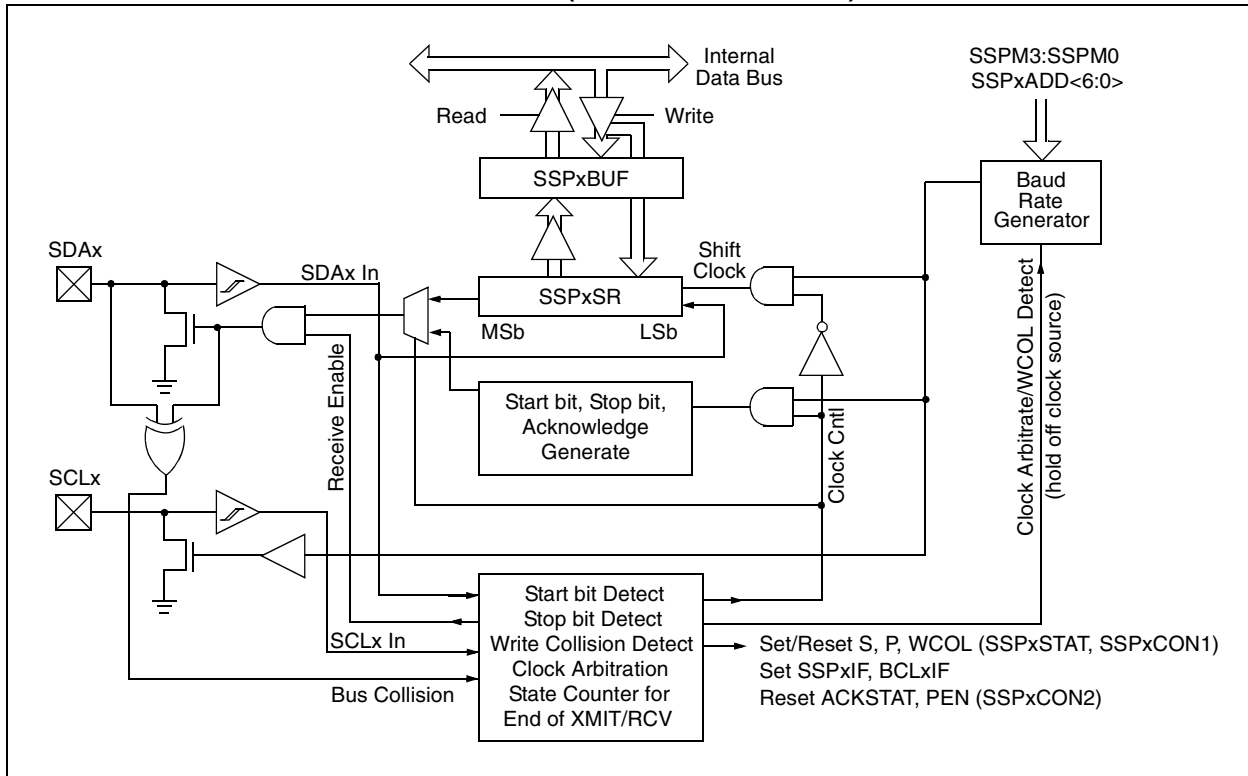
1. Assert a Start condition on SDAx and SCLx.
2. Assert a Repeated Start condition on SDAx and SCLx.
3. Write to the SSPxBUF register initiating transmission of data/address.
4. Configure the I²C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDAx and SCLx.

Note: The MSSP module, when configured in I²C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur.

The following events will cause the MSSP Interrupt Flag bit, SSPxIF, to be set (and MSSP interrupt, if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmit
- Repeated Start

FIGURE 15-16: MSSP BLOCK DIAGRAM (I²C™ MASTER MODE)



15.4.6.1 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_x, while SCL_x outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/W) bit. In this case, the R/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/W bit. In this case, the R/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_x, while SCL_x 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_x clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 15.4.7 "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 (SSPxCON2<0>).
2. SSPxIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPxBUF with the slave address to transmit.
4. Address is shifted out the SDA_x 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 SSPxCON2 register (SSPxCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
7. The user loads the SSPxBUF with eight bits of data.
8. Data is shifted out the SDA_x 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 SSPxCON2 register (SSPxCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
11. The user generates a Stop condition by setting the Stop Enable bit, PEN (SSPxCON2<2>).
12. Interrupt is generated once the Stop condition is complete.

PIC18F45J10 FAMILY

15.4.7 BAUD RATE

In I²C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPxADD register (Figure 15-17). When a write occurs to SSPxBUF, 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 ACK), the internal clock will automatically stop counting and the SCLx pin will remain in its last state.

Table 15-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

15.4.7.1 Baud Rate and Module Interdependence

Because MSSP1 and MSSP2 are independent, they can operate simultaneously in I²C Master mode at different baud rates. This is done by using different BRG reload values for each module.

Because this mode derives its basic clock source from the system clock, any changes to the clock will affect both modules in the same proportion. It may be possible to change one or both baud rates back to a previous value by changing the BRG reload value.

FIGURE 15-17: BAUD RATE GENERATOR BLOCK DIAGRAM

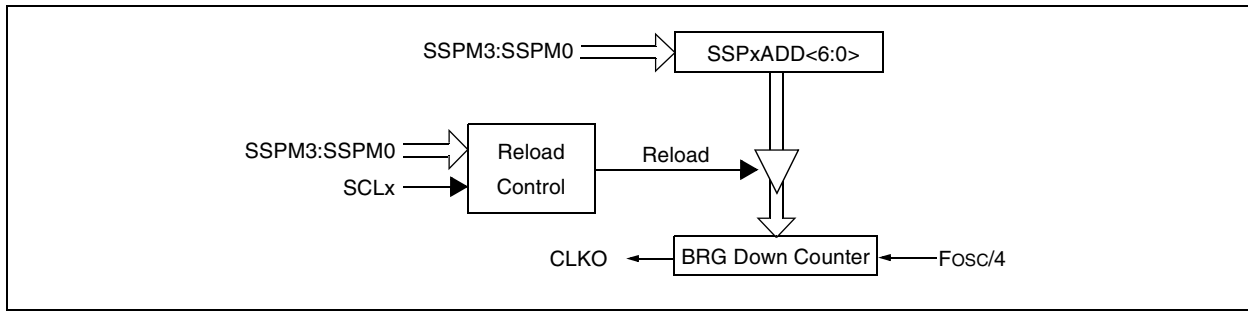


TABLE 15-3: I²C™ CLOCK RATE w/BRG

| Fcy | Fcy * 2 | BRG Value | Fscl (2 Rollovers of BRG) |
|--------|---------|-----------|------------------------------|
| 10 MHz | 20 MHz | 18h | 400 kHz ⁽¹⁾ |
| 10 MHz | 20 MHz | 1Fh | 312.5 kHz |
| 10 MHz | 20 MHz | 63h | 100 kHz |
| 4 MHz | 8 MHz | 09h | 400 kHz ⁽¹⁾ |
| 4 MHz | 8 MHz | 0Ch | 308 kHz |
| 4 MHz | 8 MHz | 27h | 100 kHz |
| 1 MHz | 2 MHz | 02h | 333 kHz ⁽¹⁾ |
| 1 MHz | 2 MHz | 09h | 100 kHz |
| 1 MHz | 2 MHz | 00h | 1 MHz ⁽¹⁾ |

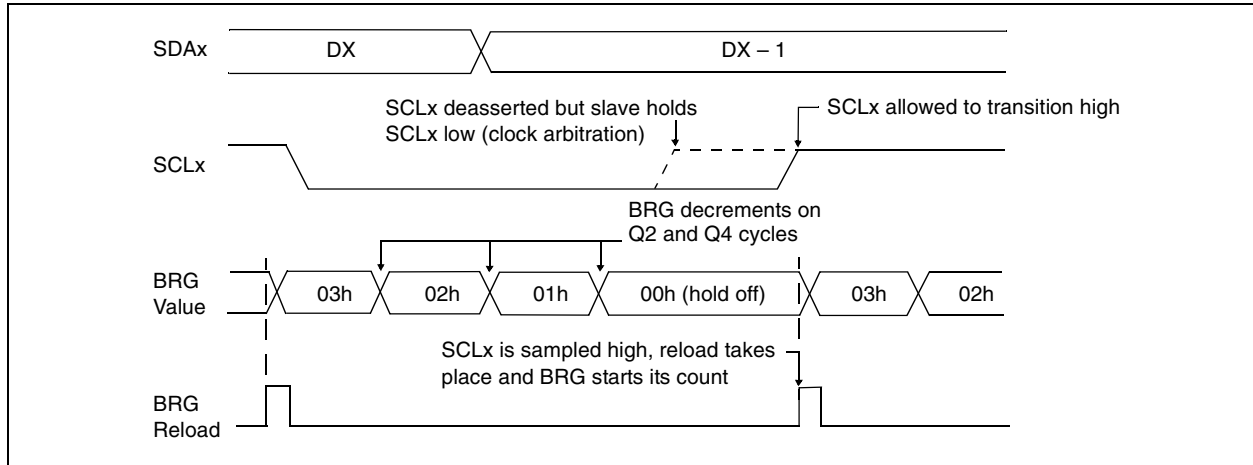
Note 1: The I²C™ interface does not conform to the 400 kHz I²C specification (which applies to rates greater than 100 kHz) in all details, but may be used with care where higher rates are required by the application.

15.4.7.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the

SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. This ensures that the SCLx 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 15-18).

FIGURE 15-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



PIC18F45J10 FAMILY

15.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPxCON2<0>). If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit (SSPxSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPxCON2<0>) will be automatically cleared by hardware. The Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

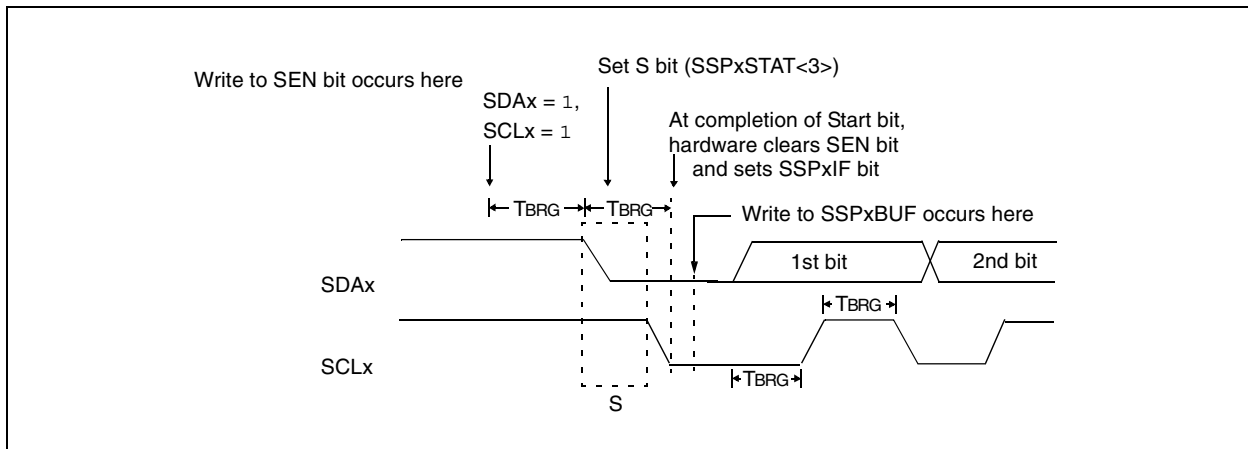
Note: If at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs. The Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

15.4.8.1 WCOL Status Flag

If the user writes the SSPxBUF 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 SSPxCON2 is disabled until the Start condition is complete.

FIGURE 15-19: FIRST START BIT TIMING



15.4.9 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPxCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPxADD<6:0> and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator times out (TBRG). When the Baud Rate Generator times out, if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. Following this, the RSEN bit (SSPxCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit (SSPxSTAT<3>) will be set. The SSPxIF 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:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

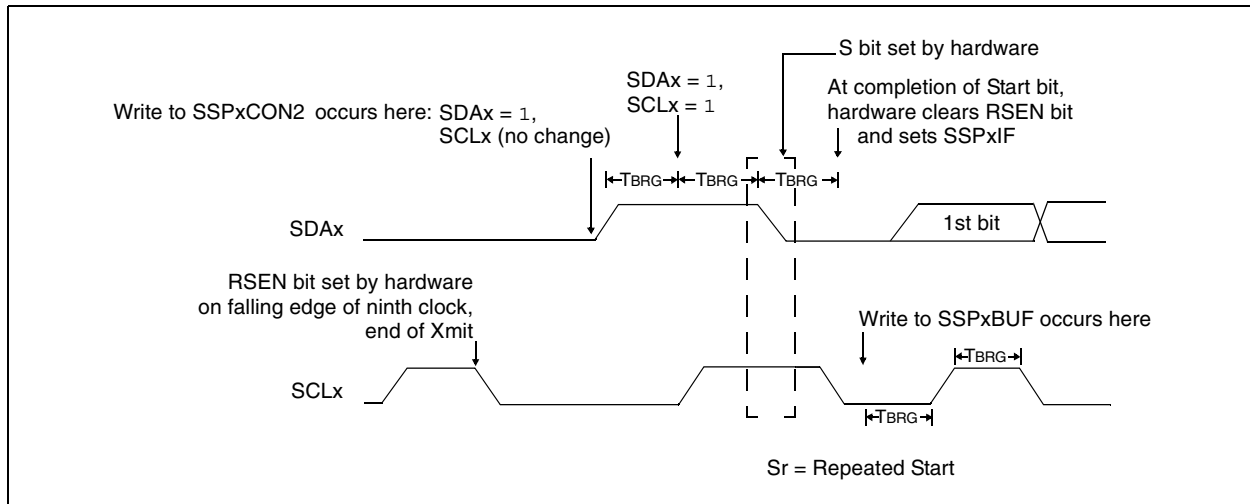
Immediately following the SSPxIF bit getting set, the user may write the SSPxBUF with the 7-bit address in 7-bit mode or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

15.4.9.1 WCOL Status Flag

If the user writes the SSPxBUF 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 SSPxCON2 is disabled until the Repeated Start condition is complete.

FIGURE 15-20: REPEATED START CONDITION WAVEFORM



PIC18F45J10 FAMILY

15.4.10 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 SSPxBUF 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 SDAx pin after the falling edge of SCLx is asserted (see data hold time specification parameter 106). SCLx is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCLx is released high (see data setup time specification parameter 107). When the SCLx pin is released high, it is held that way for TBRG. The data on the SDAx pin must remain stable for that duration and some hold time after the next falling edge of SCLx. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDAx. This allows the slave device being addressed to respond with an ACK bit during the ninth bit time if an address match occurred, or if data was received properly. The status of 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 SSPxIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 15-21).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCLx until all seven address bits and the R/W bit are completed. On the falling edge of the eighth clock, the master will deassert the SDAx pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDAx pin to see if the address was recognized by a slave. The status of the ACK bit is loaded into the ACKSTAT status bit (SSPxCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPxIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

15.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPxSTAT<0>) is set when the CPU writes to SSPxBUF and is cleared when all 8 bits are shifted out.

15.4.10.2 WCOL Status Flag

If the user writes to the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur) after 2 T_{CY} after the SSPxBUF write. If SSPxBUF is rewritten within 2 T_{CY}, the WCOL bit is set and SSPxBUF is updated. This may result in a corrupted transfer.

The user should verify that the WCOL is clear after each write to SSPxBUF to ensure the transfer is correct. In all cases, WCOL must be cleared in software.

15.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPxCON2<6>) is cleared when the slave has sent an Acknowledge ($\overline{\text{ACK}} = 0$) and is set when the slave does not Acknowledge ($\overline{\text{ACK}} = 1$). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

15.4.11 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPxCON2<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 SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx 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 (SSPxCON2<4>).

15.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

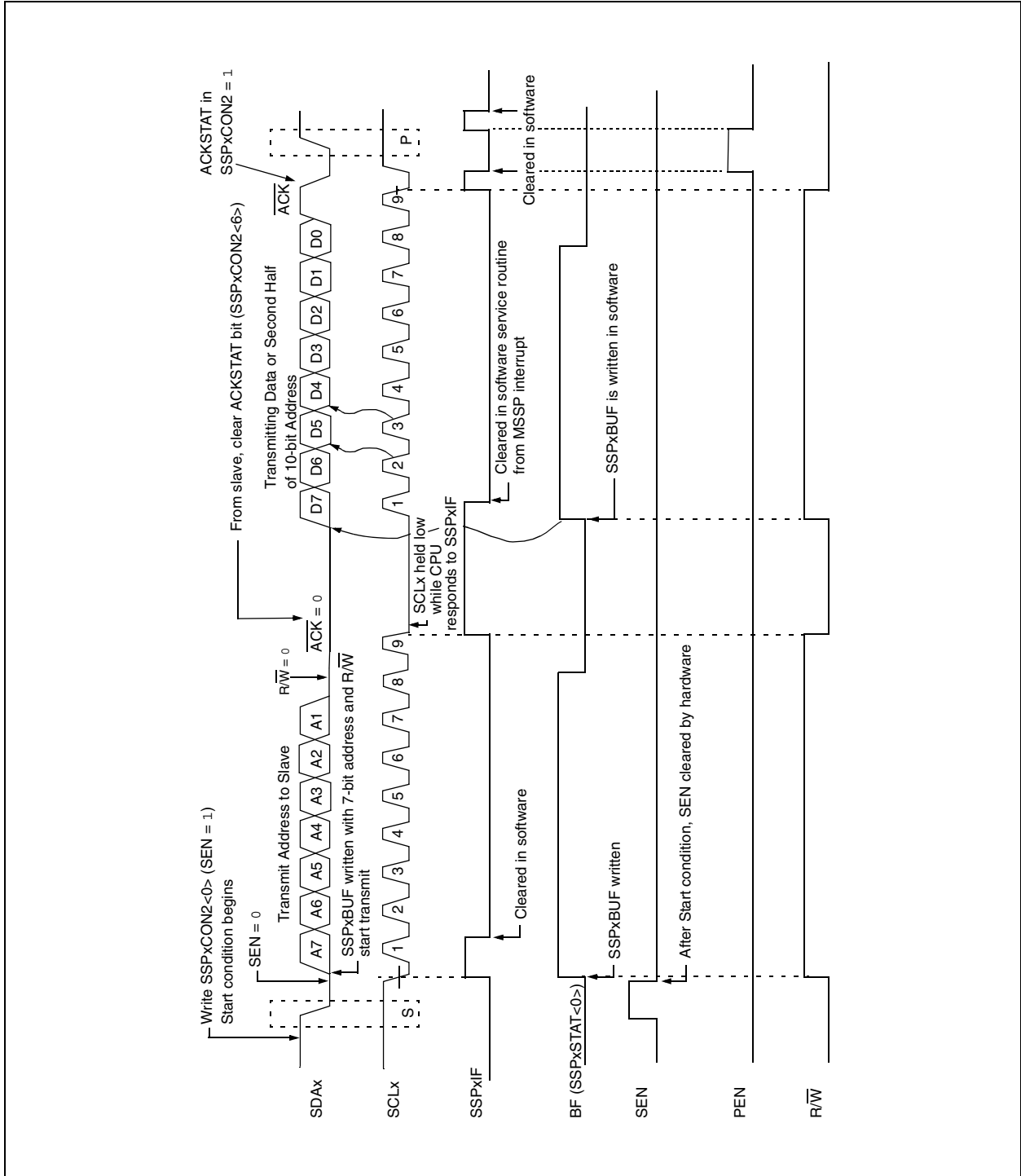
15.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

15.4.11.3 WCOL Status Flag

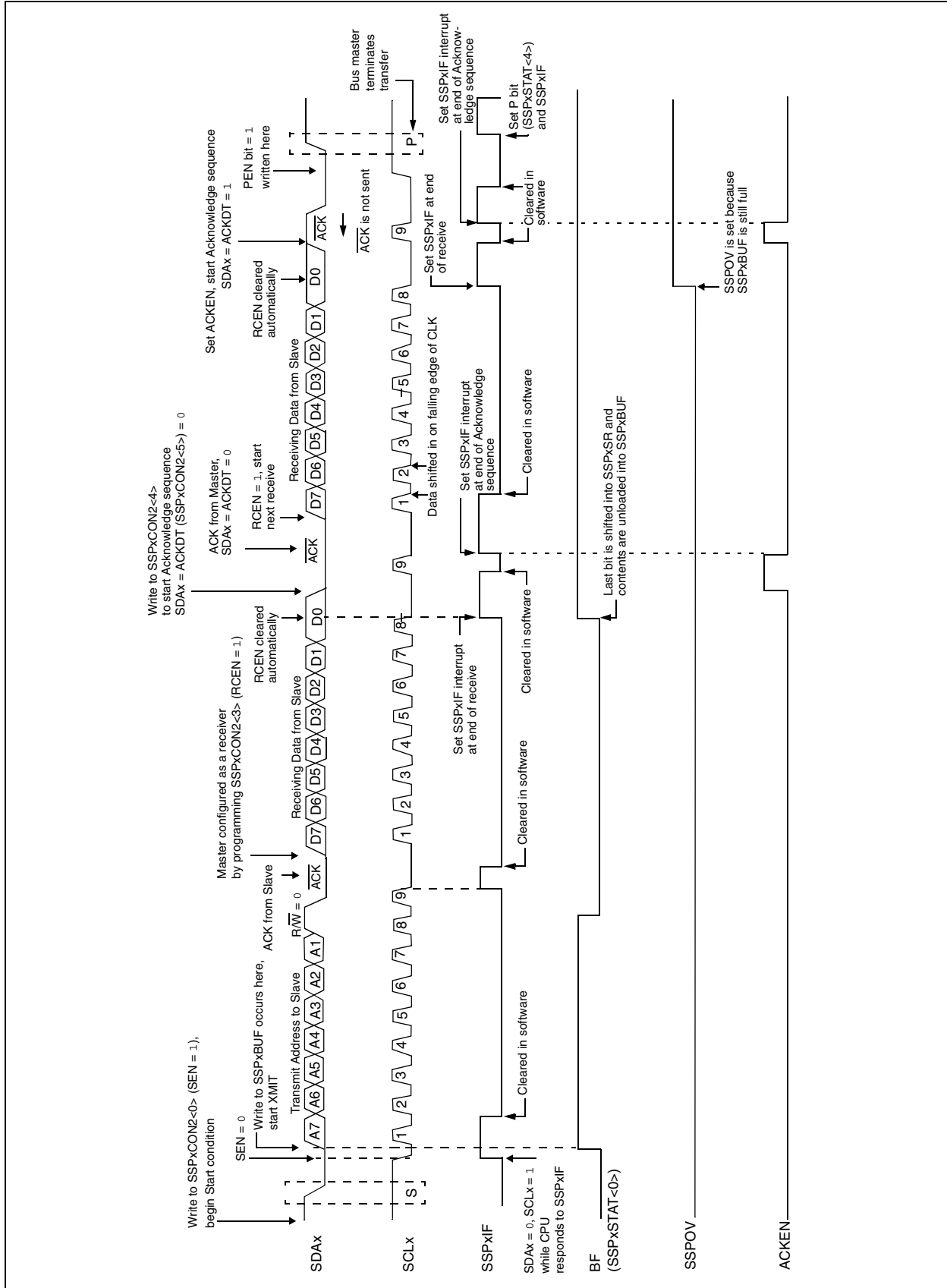
If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 15-21: I²C™ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)



PIC18F45J10 FAMILY

FIGURE 15-22: I²C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



15.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPxCON2<4>). When this bit is set, the SCLx pin is pulled low and the contents of the Acknowledge data bit are presented on the SDAx 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 SCLx pin is deasserted (pulled high). When the SCLx pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCLx 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 15-23).

15.4.12.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

15.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDAx pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPxCON2<2>). At the end of a receive/transmit, the SCLx line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDAx line low. When the SDAx line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCLx pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the P bit (SSPxSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 15-24).

15.4.13.1 WCOL Status Flag

If the user writes the SSPxBUF 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 15-23: ACKNOWLEDGE SEQUENCE WAVEFORM

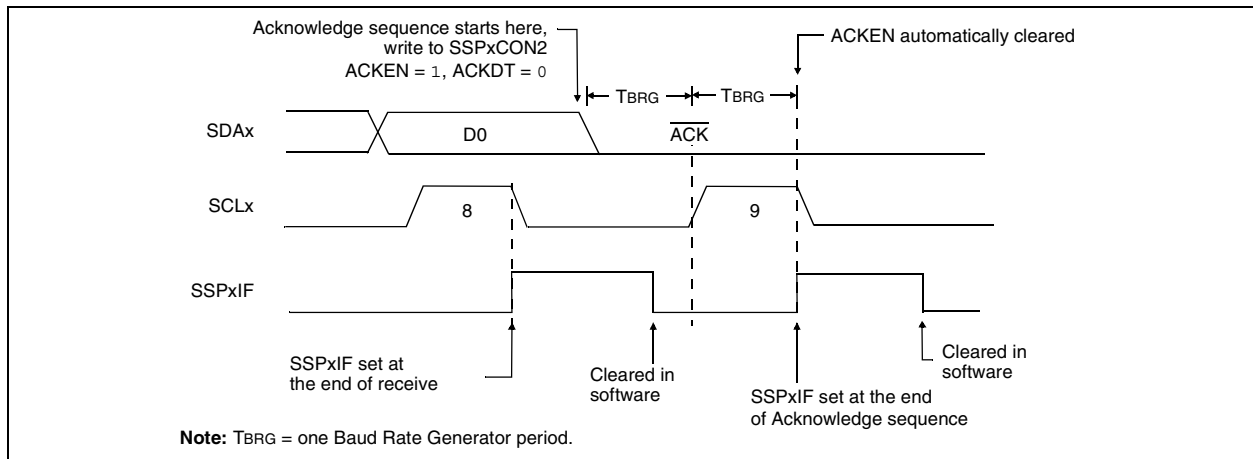
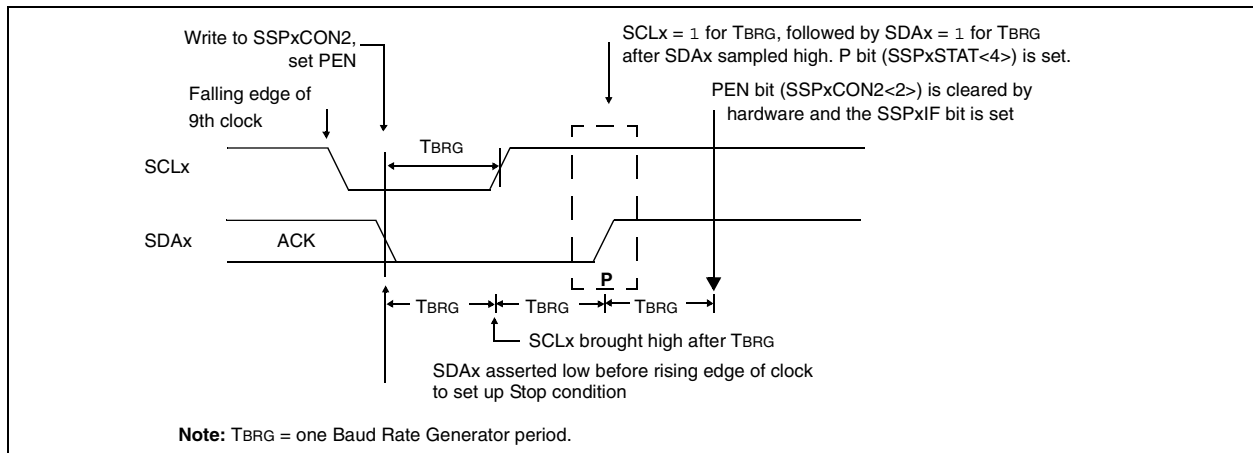


FIGURE 15-24: STOP CONDITION RECEIVE OR TRANSMIT MODE



PIC18F45J10 FAMILY

15.4.14 SLEEP OPERATION

While in Sleep 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).

15.4.15 EFFECTS OF A RESET

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

15.4.16 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 (SSPxSTAT<4>) is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the MSSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDAx 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 BCLxIF bit.

The states where arbitration can be lost are:

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

15.4.17 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 SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF and reset the I²C port to its Idle state (Figure 15-25).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF 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.

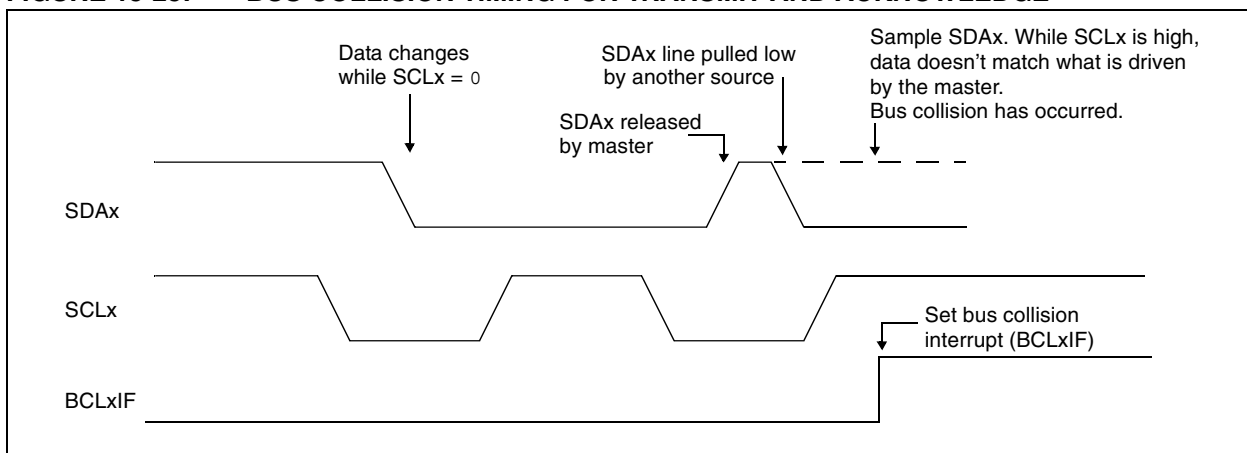
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service 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 SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

A write to the SSPxBUF 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 SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 15-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



15.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

- SDAx or SCLx are sampled low at the beginning of the Start condition (Figure 15-26).
- SCLx is sampled low before SDAx is asserted low (Figure 15-27).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

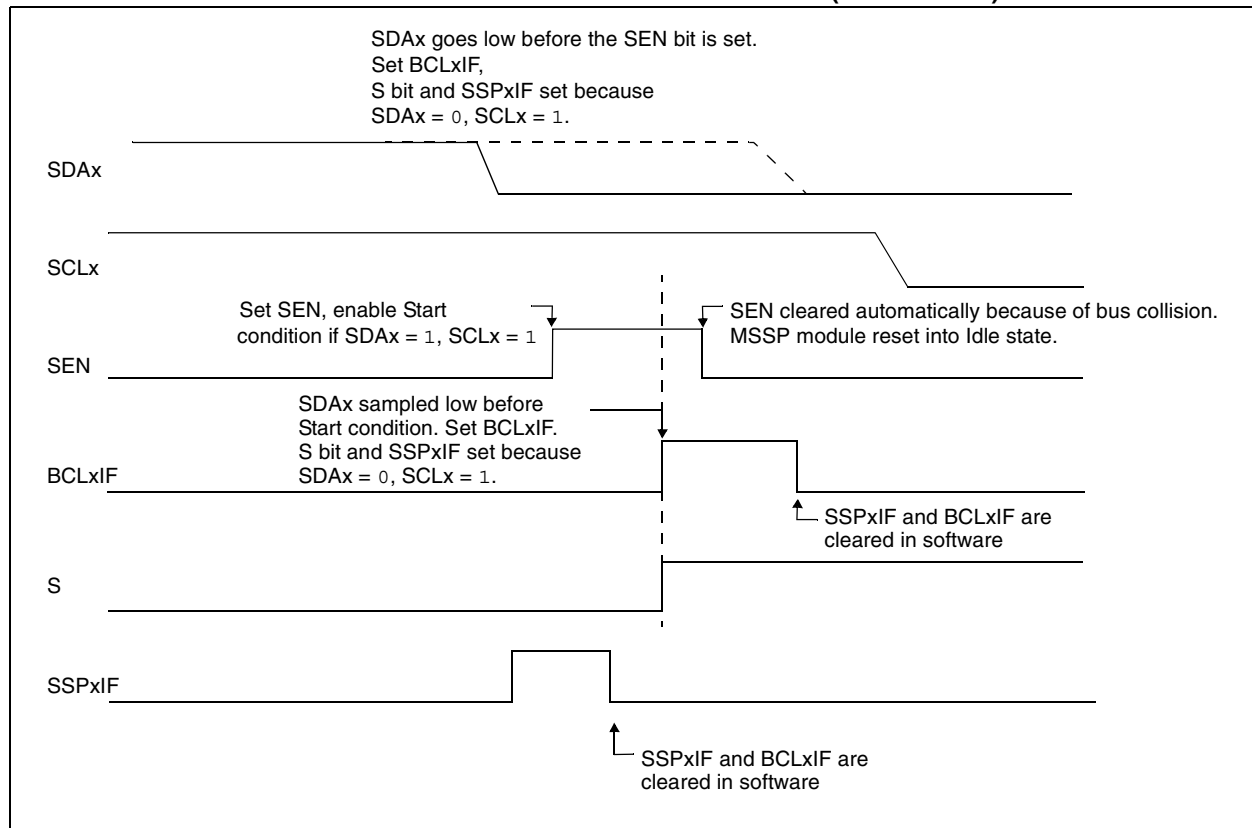
- the Start condition is aborted;
- the BCLxIF flag is set; and
- the MSSP module is reset to its Idle state (Figure 15-26).

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded from SSPxADD<6:0> and counts down to '0'. If the SCLx pin is sampled low while SDAx 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 SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 15-28). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to '0'. If the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx 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 SDAx 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 15-26: BUS COLLISION DURING START CONDITION (SDAx ONLY)



PIC18F45J10 FAMILY

FIGURE 15-27: BUS COLLISION DURING START CONDITION (SCLx = 0)

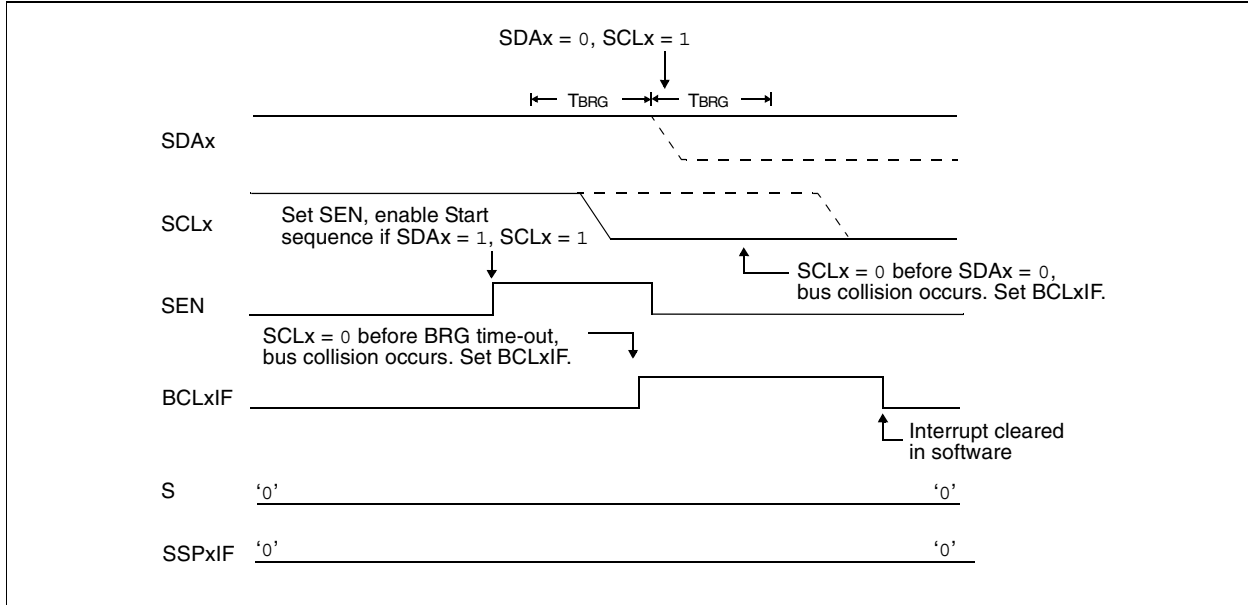
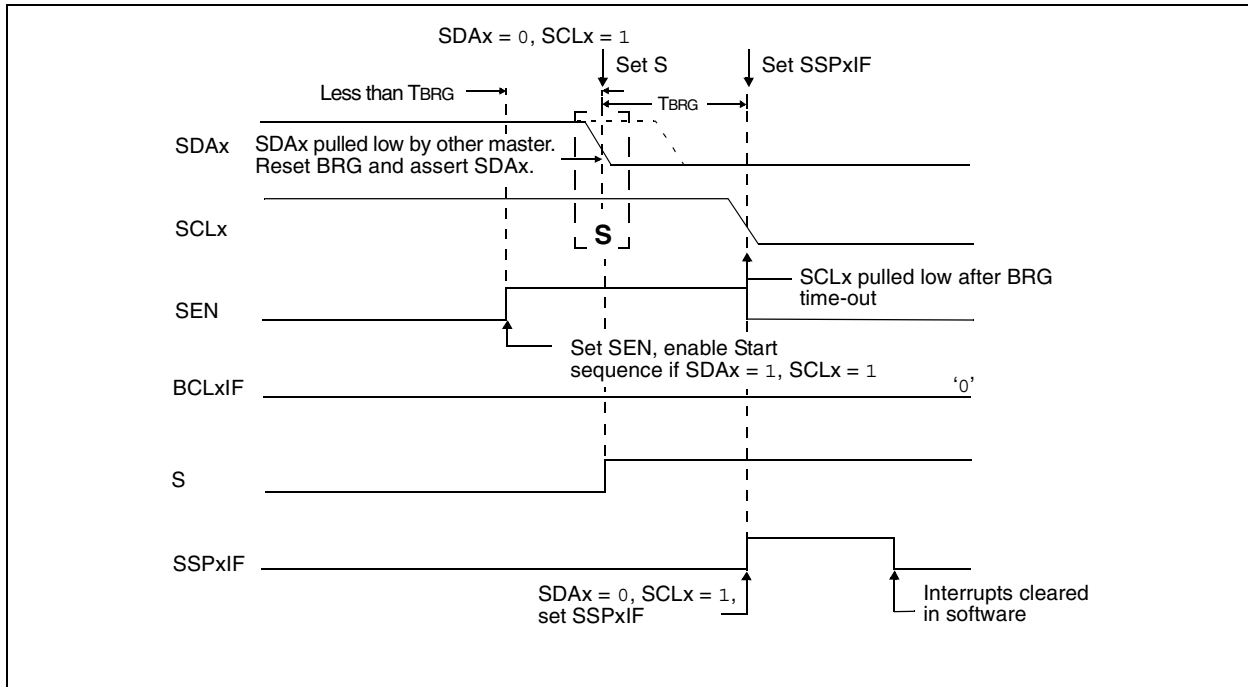


FIGURE 15-28: BRG RESET DUE TO SDAx ARBITRATION DURING START CONDITION



PIC18F45J10 FAMILY

15.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

- A low level is sampled on SDAx when SCLx goes from low level to high level.
- SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD<6:0> and counts down to '0'. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', see Figure 15-29). If SDAx is sampled high, the BRG is reloaded and begins counting. If SDAx goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDAx at exactly the same time.

If SCLx goes from high-to-low before the BRG times out and SDAx 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 15-30).

If, at the end of the BRG time-out, both SCLx and SDAx are still high, the SDAx pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCLx pin, the SCLx pin is driven low and the Repeated Start condition is complete.

FIGURE 15-29: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

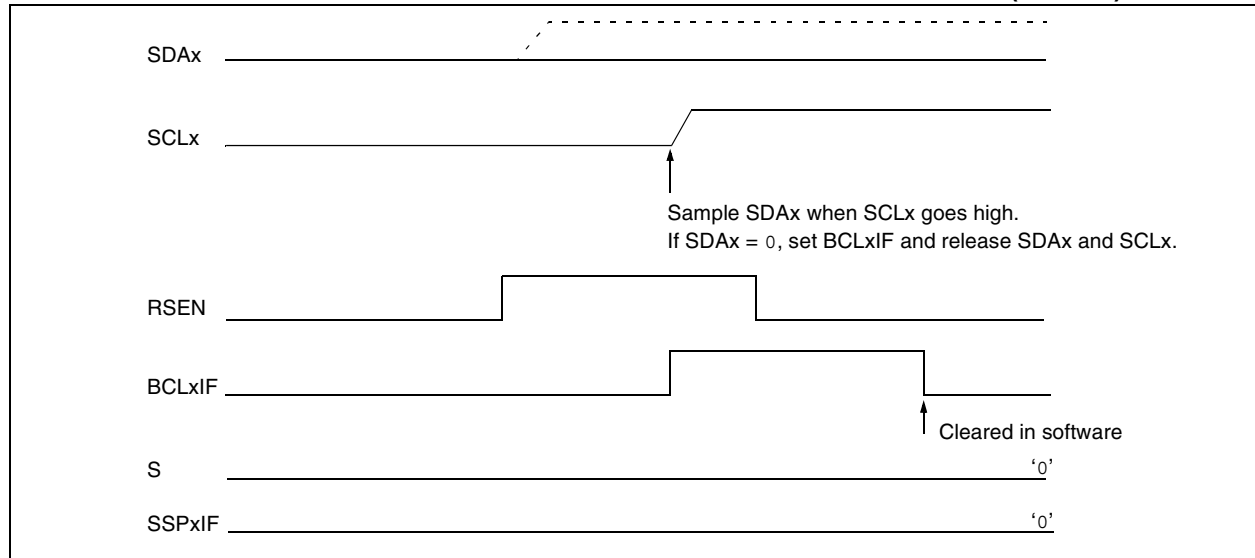
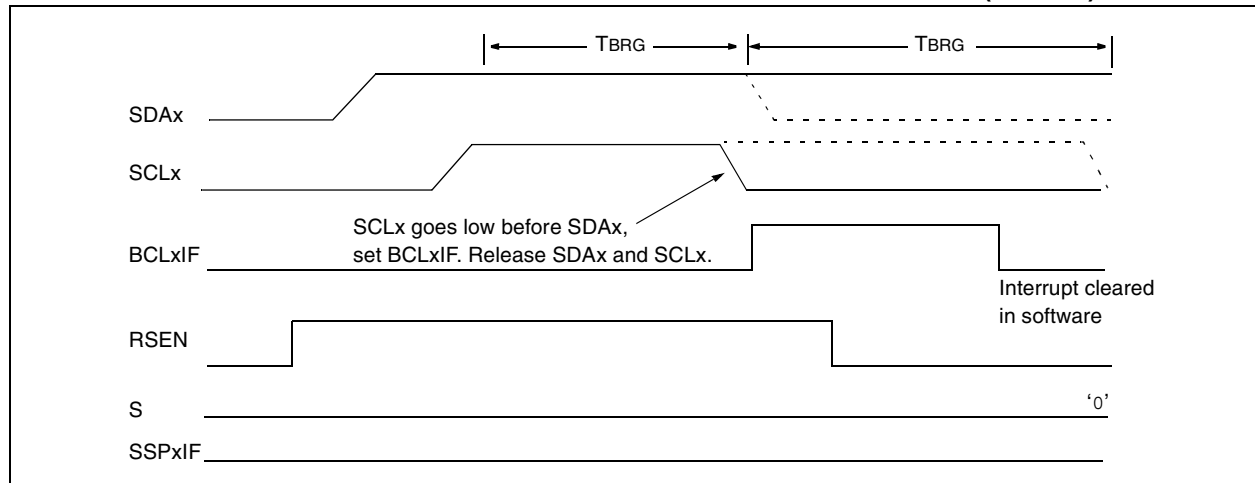


FIGURE 15-30: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



PIC18F45J10 FAMILY

15.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

- After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out.
- After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high.

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD<6:0> and counts down to '0'. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 15-31). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 15-32).

FIGURE 15-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)

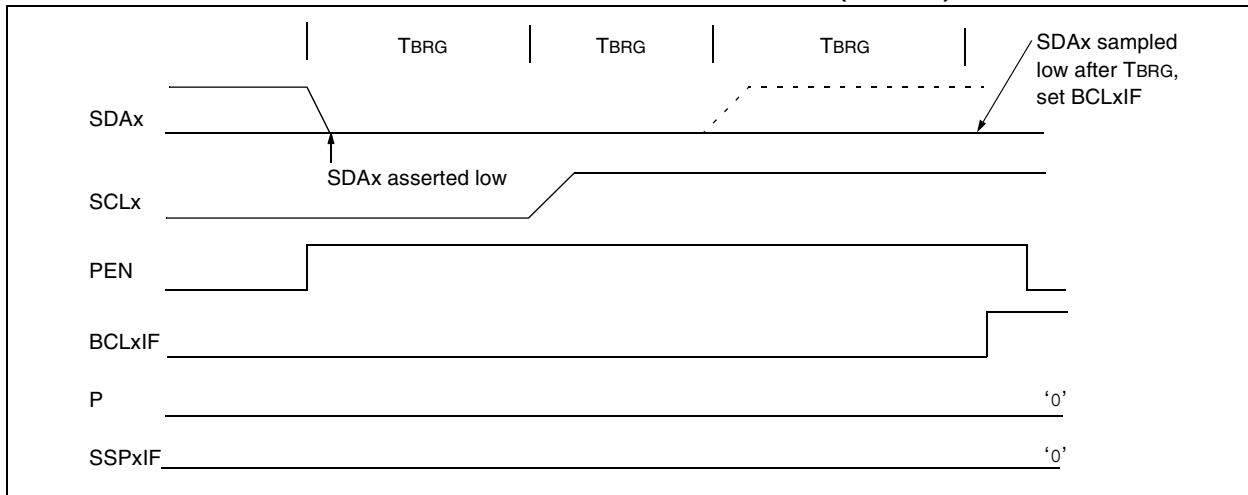
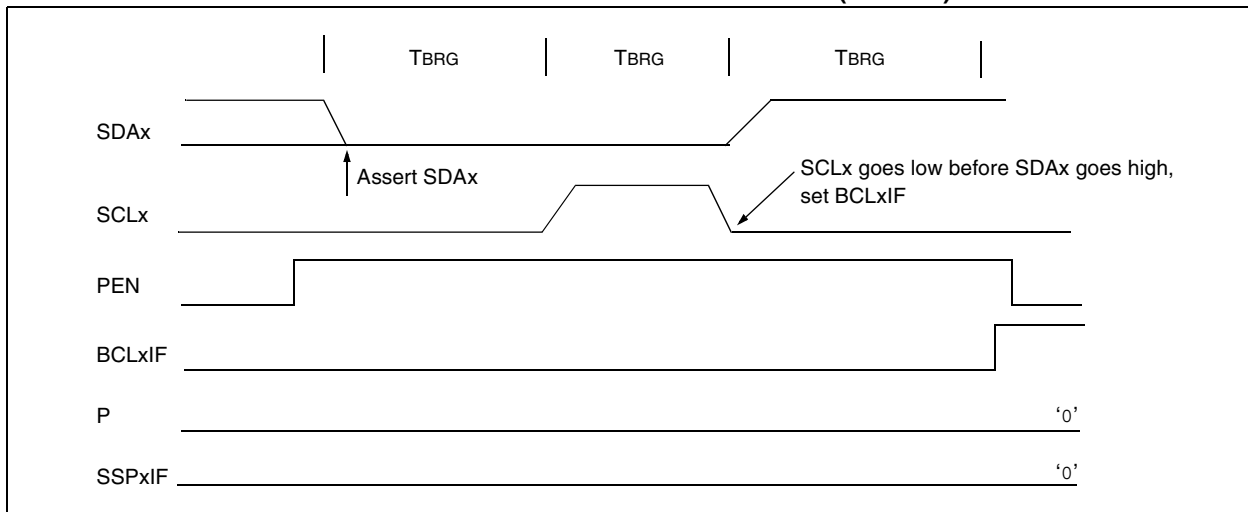


FIGURE 15-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)



PIC18F45J10 FAMILY

TABLE 15-4: REGISTERS ASSOCIATED WITH I²C™ OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page |
|----------------------|---|-----------|--------------|--------|--------|--------------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| PIR3 | SSP2IF | BCL2IF | — | — | — | — | — | — | 45 |
| PIE3 | SSP2IE | BCL2IE | — | — | — | — | — | — | 45 |
| IPR3 | SSP2IP | BCL2IP | — | — | — | — | — | — | 45 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 46 |
| TRISD ⁽¹⁾ | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 46 |
| SSP1BUF | MSSP1 Receive Buffer/Transmit Register | | | | | | | | 44 |
| SSP1ADD | MSSP1 Address Register (I ² C™ Slave mode). MSSP1 Baud Rate Reload Register (I ² C Master mode). | | | | | | | | 44 |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 44 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 44 |
| SSP1STAT | SMP | CKE | D/ \bar{A} | P | S | R/ \bar{W} | UA | BF | 44 |
| SSP2BUF | MSSP2 Receive Buffer/Transmit Register | | | | | | | | 46 |
| SSP2ADD | MSSP2 Address Register (I ² C Slave mode). MSSP2 Baud Rate Reload Register (I ² C Master mode). | | | | | | | | 46 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 | 46 |
| SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 46 |
| SSP2STAT | SMP | CKE | D/ \bar{A} | P | S | R/ \bar{W} | UA | BF | 46 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the MSSP module in I²C™ mode.

Note 1: These registers and/or bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

NOTES:

16.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of the two serial I/O modules. (Generically, the USART is also known as a Serial Communications Interface or SCI.) The EUSART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also 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 Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN bus) systems.

The EUSART can be configured in the following modes:

- Asynchronous (full duplex) with:
 - Auto-wake-up on character reception
 - Auto-baud calibration
 - 12-bit Break character transmission
- Synchronous – Master (half duplex) with selectable clock polarity
- Synchronous – Slave (half duplex) with selectable clock polarity

The pins of the Enhanced USART are multiplexed with PORTC. In order to configure RC6/TX/CK and RC7/RX/DT as an EUSART:

- bit SPEN (RCSTA<7>) must be set (= 1)
- bit TRISC<7> must be set (= 1)
- bit TRISC<6> must be set (= 1)

| |
|--|
| Note: The EUSART control will automatically reconfigure the pin from input to output as needed. |
|--|

The operation of the Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

These are detailed on the following pages in Register 16-1, Register 16-2 and Register 16-3, respectively.

PIC18F45J10 FAMILY

REGISTER 16-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER

| | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|
| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-1 | R/W-0 |
| CSRC | TX9 | TXEN | SYNC | SENDB | 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:** EUSART Mode Select bit

1 = Synchronous mode

0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit

Asynchronous mode:

1 = Send Sync Break on next transmission (cleared by hardware upon completion)

0 = Sync Break transmission completed

Synchronous mode:

Don't care.

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

PIC18F45J10 FAMILY

REGISTER 16-2: 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 (held in Reset)
- 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 = Enables address detection, enables interrupt and loads 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
Asynchronous mode 9-bit (RX9 = 0):
 Don't care.
- 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 |

PIC18F45J10 FAMILY

REGISTER 16-3: BAUDCON: BAUD RATE CONTROL REGISTER

| R/W-0 | R-1 | U-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | |
|--------|-------|-----|-------|-------|-----|-------|-------|-------|
| ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | |
| bit 7 | | | | | | | | bit 0 |

- bit 7 **ABDOVF:** Auto-Baud Acquisition Rollover Status bit
 1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
 0 = No BRG rollover has occurred
- bit 6 **RCIDL:** Receive Operation Idle Status bit
 1 = Receive operation is Idle
 0 = Receive operation is active
- bit 5 **Unimplemented:** Read as '0'
- bit 4 **SCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
 Unused in this mode.
Synchronous mode:
 1 = Idle state for clock (CK) is a high level
 0 = Idle state for clock (CK) is a low level
- bit 3 **BRG16:** 16-bit Baud Rate Register Enable bit
 1 = 16-bit Baud Rate Generator – SPBRGH and SPBRG
 0 = 8-bit Baud Rate Generator – SPBRG only (Compatible mode), SPBRGH value ignored
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
 1 = EUSART will continue to sample the RX pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
 0 = RX pin not monitored or rising edge detected
Synchronous mode:
 Unused in this mode.
- bit 0 **ABDEN:** Auto-Baud Detect Enable bit
Asynchronous mode:
 1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in hardware upon completion
 0 = Baud rate measurement disabled or completed
Synchronous 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 |

16.1 Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit or 16-bit generator that supports both the Asynchronous and Synchronous modes of the EUSART. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCON<3>) selects 16-bit mode.

The SPBRGH:SPBRG register pair controls the period of a free running timer. In Asynchronous mode, bits BRGH (TXSTA<2>) and BRG16 (BAUDCON<3>) also control the baud rate. In Synchronous mode, BRGH is ignored. Table 16-1 shows the formula for computation of the baud rate for different EUSART modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGH:SPBRG registers can be calculated using the formulas in Table 16-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 16-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 16-2. It may be advan-

tageous to use the high baud rate (BRGH = 1) or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGH:SPBRG registers 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.

16.1.1 OPERATION IN POWER-MANAGED MODES

The device clock is used to generate the desired baud rate. When one of the power-managed modes is entered, the new clock source may be operating at a different frequency. This may require an adjustment to the value in the SPBRG register pair.

16.1.2 SAMPLING

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

TABLE 16-1: BAUD RATE FORMULAS

| Configuration Bits | | | BRG/EUSART Mode | Baud Rate Formula |
|--------------------|-------|------|---------------------|------------------------|
| SYNC | BRG16 | BRGH | | |
| 0 | 0 | 0 | 8-bit/Asynchronous | $F_{OSC}/[64 (n + 1)]$ |
| 0 | 0 | 1 | 8-bit/Asynchronous | $F_{OSC}/[16 (n + 1)]$ |
| 0 | 1 | 0 | 16-bit/Asynchronous | |
| 0 | 1 | 1 | 16-bit/Asynchronous | $F_{OSC}/[4 (n + 1)]$ |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

Legend: x = Don't care, n = value of SPBRGH:SPBRG register pair

PIC18F45J10 FAMILY

EXAMPLE 16-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$\text{Desired Baud Rate} = \text{FOSC}/(64 ([\text{SPBRGH}:\text{SPBRG}] + 1))$$

Solving for SPBRGH:SPBRG:

$$X = ((\text{FOSC}/\text{Desired Baud Rate})/64) - 1$$

$$= ((16000000/9600)/64) - 1$$

$$= [25.042] = 25$$

$$\text{Calculated Baud Rate} = 16000000/(64 (25 + 1))$$

$$= 9615$$

$$\text{Error} = (\text{Calculated Baud Rate} - \text{Desired Baud Rate})/\text{Desired Baud Rate}$$

$$= (9615 - 9600)/9600 = 0.16\%$$

TABLE 16-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-------|-------|-------|-------|-------|-------|-------|----------------------|
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

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

PIC18F45J10 FAMILY

TABLE 16-3: BAUD RATES FOR ASYNCHRONOUS MODES

| BAUD RATE (K) | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 10.000 MHz | | | Fosc = 8.000 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 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1.2 | — | — | — | 1.221 | 1.73 | 255 | 1.202 | 0.16 | 129 | 1201 | -0.16 | 103 |
| 2.4 | 2.441 | 1.73 | 255 | 2.404 | 0.16 | 129 | 2.404 | 0.16 | 64 | 2403 | -0.16 | 51 |
| 9.6 | 9.615 | 0.16 | 64 | 9.766 | 1.73 | 31 | 9.766 | 1.73 | 15 | 9615 | -0.16 | 12 |
| 19.2 | 19.531 | 1.73 | 31 | 19.531 | 1.73 | 15 | 19.531 | 1.73 | 7 | — | — | — |
| 57.6 | 56.818 | -1.36 | 10 | 62.500 | 8.51 | 4 | 52.083 | -9.58 | 2 | — | — | — |
| 115.2 | 125.000 | 8.51 | 4 | 104.167 | -9.58 | 2 | 78.125 | -32.18 | 1 | — | — | — |

| BAUD RATE (K) | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 4.000 MHz | | | Fosc = 2.000 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.300 | 0.16 | 207 | 300 | -0.16 | 103 | 300 | -0.16 | 51 |
| 1.2 | 1.202 | 0.16 | 51 | 1201 | -0.16 | 25 | 1201 | -0.16 | 12 |
| 2.4 | 2.404 | 0.16 | 25 | 2403 | -0.16 | 12 | — | — | — |
| 9.6 | 8.929 | -6.99 | 6 | — | — | — | — | — | — |
| 19.2 | 20.833 | 8.51 | 2 | — | — | — | — | — | — |
| 57.6 | 62.500 | 8.51 | 0 | — | — | — | — | — | — |
| 115.2 | 62.500 | -45.75 | 0 | — | — | — | — | — | — |

| BAUD RATE (K) | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 10.000 MHz | | | Fosc = 8.000 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 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1.2 | — | — | — | — | — | — | — | — | — | — | — | — |
| 2.4 | — | — | — | — | — | — | 2.441 | 1.73 | 255 | 2403 | -0.16 | 207 |
| 9.6 | 9.766 | 1.73 | 255 | 9.615 | 0.16 | 129 | 9.615 | 0.16 | 64 | 9615 | -0.16 | 51 |
| 19.2 | 19.231 | 0.16 | 129 | 19.231 | 0.16 | 64 | 19.531 | 1.73 | 31 | 19230 | -0.16 | 25 |
| 57.6 | 58.140 | 0.94 | 42 | 56.818 | -1.36 | 21 | 56.818 | -1.36 | 10 | 55555 | 3.55 | 8 |
| 115.2 | 113.636 | -1.36 | 21 | 113.636 | -1.36 | 10 | 125.000 | 8.51 | 4 | — | — | — |

| BAUD RATE (K) | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 4.000 MHz | | | Fosc = 2.000 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | — | — | — | — | — | — | 300 | -0.16 | 207 |
| 1.2 | 1.202 | 0.16 | 207 | 1201 | -0.16 | 103 | 1201 | -0.16 | 51 |
| 2.4 | 2.404 | 0.16 | 103 | 2403 | -0.16 | 51 | 2403 | -0.16 | 25 |
| 9.6 | 9.615 | 0.16 | 25 | 9615 | -0.16 | 12 | — | — | — |
| 19.2 | 19.231 | 0.16 | 12 | — | — | — | — | — | — |
| 57.6 | 62.500 | 8.51 | 3 | — | — | — | — | — | — |
| 115.2 | 125.000 | 8.51 | 1 | — | — | — | — | — | — |

PIC18F45J10 FAMILY

TABLE 16-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

| BAUD RATE (K) | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 10.000 MHz | | | Fosc = 8.000 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 | 0.300 | 0.00 | 8332 | 0.300 | 0.02 | 4165 | 0.300 | 0.02 | 2082 | 300 | -0.04 | 1665 |
| 1.2 | 1.200 | 0.02 | 2082 | 1.200 | -0.03 | 1041 | 1.200 | -0.03 | 520 | 1201 | -0.16 | 415 |
| 2.4 | 2.402 | 0.06 | 1040 | 2.399 | -0.03 | 520 | 2.404 | 0.16 | 259 | 2403 | -0.16 | 207 |
| 9.6 | 9.615 | 0.16 | 259 | 9.615 | 0.16 | 129 | 9.615 | 0.16 | 64 | 9615 | -0.16 | 51 |
| 19.2 | 19.231 | 0.16 | 129 | 19.231 | 0.16 | 64 | 19.531 | 1.73 | 31 | 19230 | -0.16 | 25 |
| 57.6 | 58.140 | 0.94 | 42 | 56.818 | -1.36 | 21 | 56.818 | -1.36 | 10 | 55555 | 3.55 | 8 |
| 115.2 | 113.636 | -1.36 | 21 | 113.636 | -1.36 | 10 | 125.000 | 8.51 | 4 | — | — | — |

| BAUD RATE (K) | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | |
|---------------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 4.000 MHz | | | Fosc = 2.000 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.300 | 0.04 | 832 | 300 | -0.16 | 415 | 300 | -0.16 | 207 |
| 1.2 | 1.202 | 0.16 | 207 | 1201 | -0.16 | 103 | 1201 | -0.16 | 51 |
| 2.4 | 2.404 | 0.16 | 103 | 2403 | -0.16 | 51 | 2403 | -0.16 | 25 |
| 9.6 | 9.615 | 0.16 | 25 | 9615 | -0.16 | 12 | — | — | — |
| 19.2 | 19.231 | 0.16 | 12 | — | — | — | — | — | — |
| 57.6 | 62.500 | 8.51 | 3 | — | — | — | — | — | — |
| 115.2 | 125.000 | 8.51 | 1 | — | — | — | — | — | — |

| BAUD RATE (K) | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---------------|--|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 40.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 10.000 MHz | | | Fosc = 8.000 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 | 0.300 | 0.00 | 33332 | 0.300 | 0.00 | 16665 | 0.300 | 0.00 | 8332 | 300 | -0.01 | 6665 |
| 1.2 | 1.200 | 0.00 | 8332 | 1.200 | 0.02 | 4165 | 1.200 | 0.02 | 2082 | 1200 | -0.04 | 1665 |
| 2.4 | 2.400 | 0.02 | 4165 | 2.400 | 0.02 | 2082 | 2.402 | 0.06 | 1040 | 2400 | -0.04 | 832 |
| 9.6 | 9.606 | 0.06 | 1040 | 9.596 | -0.03 | 520 | 9.615 | 0.16 | 259 | 9615 | -0.16 | 207 |
| 19.2 | 19.193 | -0.03 | 520 | 19.231 | 0.16 | 259 | 19.231 | 0.16 | 129 | 19230 | -0.16 | 103 |
| 57.6 | 57.803 | 0.35 | 172 | 57.471 | -0.22 | 86 | 58.140 | 0.94 | 42 | 57142 | 0.79 | 34 |
| 115.2 | 114.943 | -0.22 | 86 | 116.279 | 0.94 | 42 | 113.636 | -1.36 | 21 | 117647 | -2.12 | 16 |

| BAUD RATE (K) | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | |
|---------------|--|---------|-----------------------|------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 4.000 MHz | | | Fosc = 2.000 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) | Actual Rate (K) | % Error | SPBRG value (decimal) |
| 0.3 | 0.300 | 0.01 | 3332 | 300 | -0.04 | 1665 | 300 | -0.04 | 832 |
| 1.2 | 1.200 | 0.04 | 832 | 1201 | -0.16 | 415 | 1201 | -0.16 | 207 |
| 2.4 | 2.404 | 0.16 | 415 | 2403 | -0.16 | 207 | 2403 | -0.16 | 103 |
| 9.6 | 9.615 | 0.16 | 103 | 9615 | -0.16 | 51 | 9615 | -0.16 | 25 |
| 19.2 | 19.231 | 0.16 | 51 | 19230 | -0.16 | 25 | 19230 | -0.16 | 12 |
| 57.6 | 58.824 | 2.12 | 16 | 55555 | 3.55 | 8 | — | — | — |
| 115.2 | 111.111 | -3.55 | 8 | — | — | — | — | — | — |

16.1.3 AUTO-BAUD RATE DETECT

The Enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 16-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RX signal, the RX signal is timing the BRG. In ABD mode, the internal Baud Rate Generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a Start bit. The Auto-Baud Rate Detect must receive a byte with the value 55h (ASCII “U”, which is also the LIN bus Sync character) in order to calculate the proper bit rate. The measurement is taken over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRG begins counting up, using the preselected clock source on the first rising edge of RX. After eight bits on the RX pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGH:SPBRG register pair. Once the 5th edge is seen (this should correspond to the Stop bit), the ABDEN bit is automatically cleared.

If a rollover of the BRG occurs (an overflow from FFFFh to 0000h), the event is trapped by the ABDOVF status bit (BAUDCON<7>). It is set in hardware by BRG rollovers and can be set or cleared by the user in software. ABD mode remains active after rollover events and the ABDEN bit remains set (Figure 16-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. Note that the BRG clock will be configured by the BRG16 and BRGH bits. Independent of the BRG16 bit setting, both the SPBRG and SPBRGH will be used as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGH register. Refer to Table 16-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSART state machine is held in Idle. The RCIF interrupt is set once the fifth rising edge on RX is detected. The value in the RCREG needs to be read to clear the RCIF interrupt. The contents of RCREG should be discarded.

Note 1: If the WUE bit is set with the ABDEN bit, Auto-Baud Rate Detection will occur on the byte *following* the Break character.

2: It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible due to bit error rates. Overall system timing and communication baud rates must be taken into consideration when using the Auto-Baud Rate Detection feature.

TABLE 16-4: BRG COUNTER CLOCK RATES

| BRG16 | BRGH | BRG Counter Clock |
|-------|------|-------------------|
| 0 | 0 | Fosc/512 |
| 0 | 1 | Fosc/128 |
| 1 | 0 | Fosc/128 |
| 1 | 1 | Fosc/32 |

Note: During the ABD sequence, SPBRG and SPBRGH are both used as a 16-bit counter, independent of BRG16 setting.

16.1.3.1 ABD and EUSART Transmission

Since the BRG clock is reversed during ABD acquisition, the EUSART transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREG cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSART operation.

PIC18F45J10 FAMILY

FIGURE 16-1: AUTOMATIC BAUD RATE CALCULATION

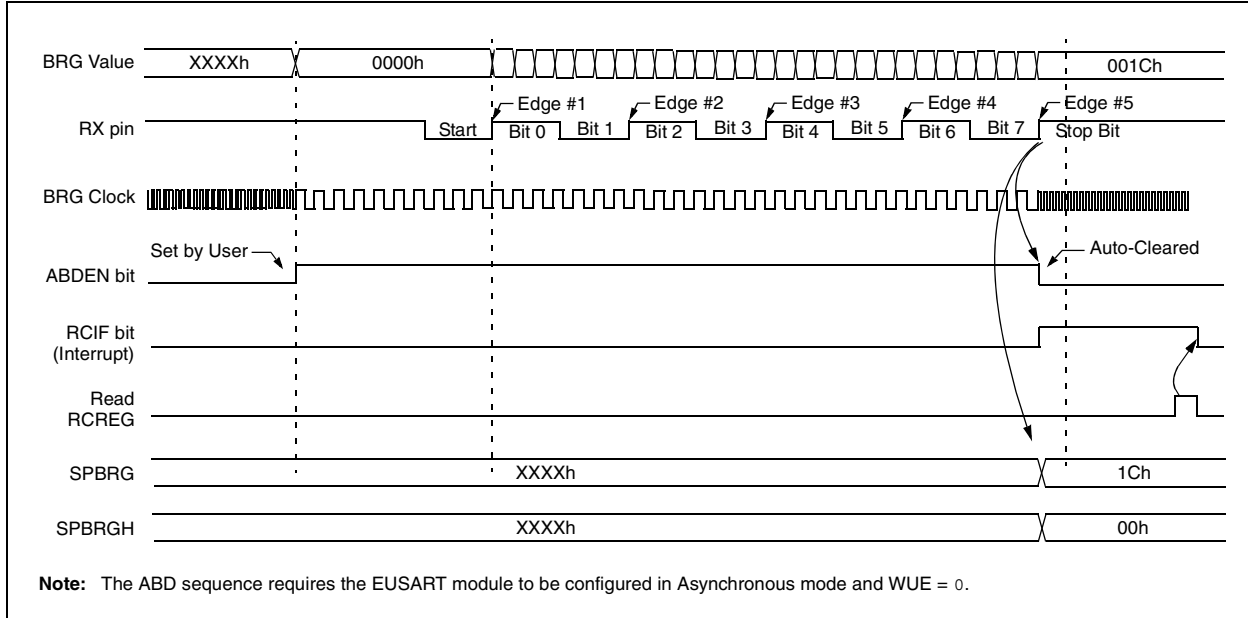
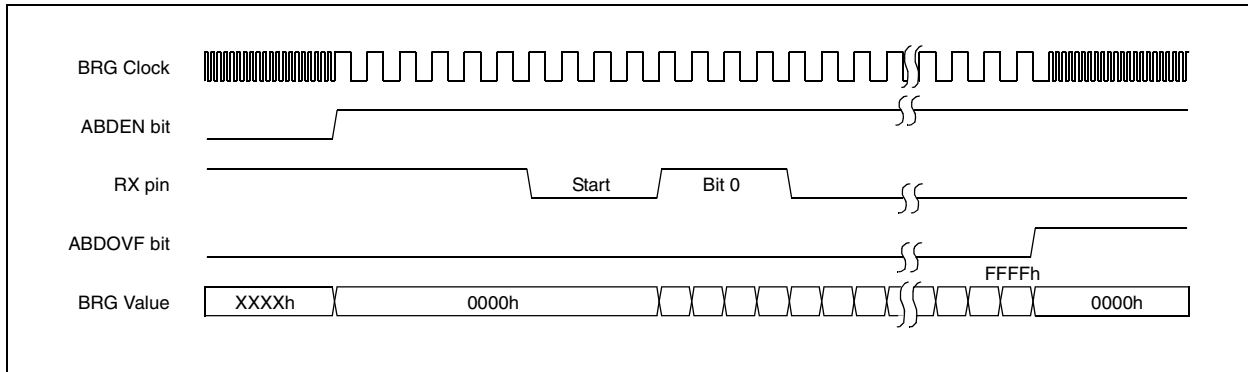


FIGURE 16-2: BRG OVERFLOW SEQUENCE



16.2 EUSART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTA<4>). In this mode, the EUSART 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/16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The EUSART transmits and receives the LSb first. The EUSART'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 the BRGH and BRG16 bits (TXSTA<2> and BAUDCON<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the EUSART module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto-Wake-up on Sync Break Character
- 12-bit Break Character Transmit
- Auto-Baud Rate Detection

16.2.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 16-3. 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 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 the TXIF flag bit (PIR1<4>) is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXIE (PIE1<4>). TXIF will be set regardless of the state of TXIE; it cannot be cleared in software. TXIF is also not cleared immediately upon loading TXREG, but becomes valid in the second instruction cycle following the load instruction. Polling TXIF immediately following a load of TXREG will return invalid results.

While 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 register is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty.

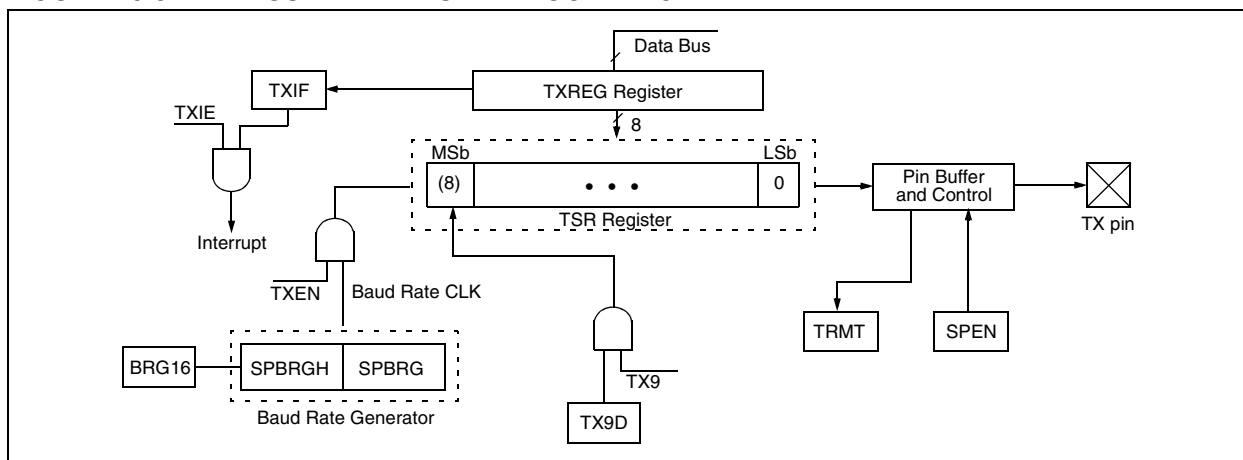
Note 1: The TSR register is not mapped in data memory so it is not available to the user.

Note 2: Flag bit TXIF is set when enable bit TXEN is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
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 16-3: EUSART TRANSMIT BLOCK DIAGRAM



PIC18F45J10 FAMILY

FIGURE 16-4: ASYNCHRONOUS TRANSMISSION

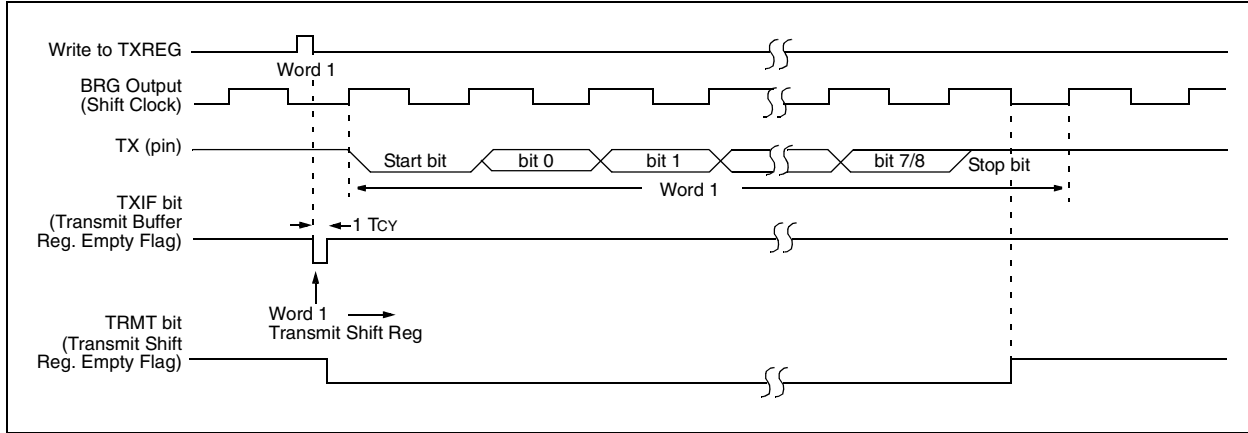
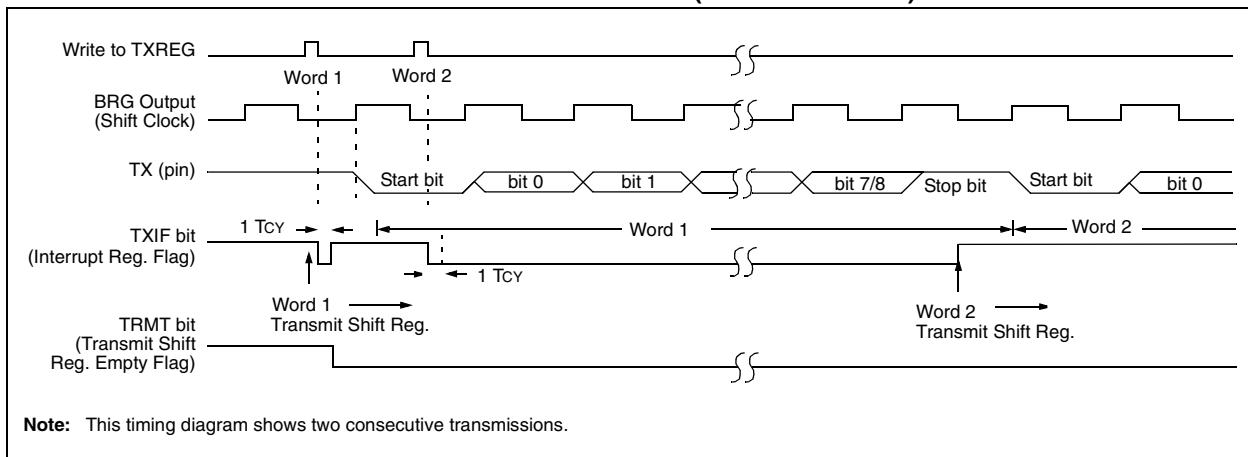


FIGURE 16-5: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 16-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| TXREG | EUSART Transmit Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDER | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

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

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

16.2.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 16-6. The data is received on the RX 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:

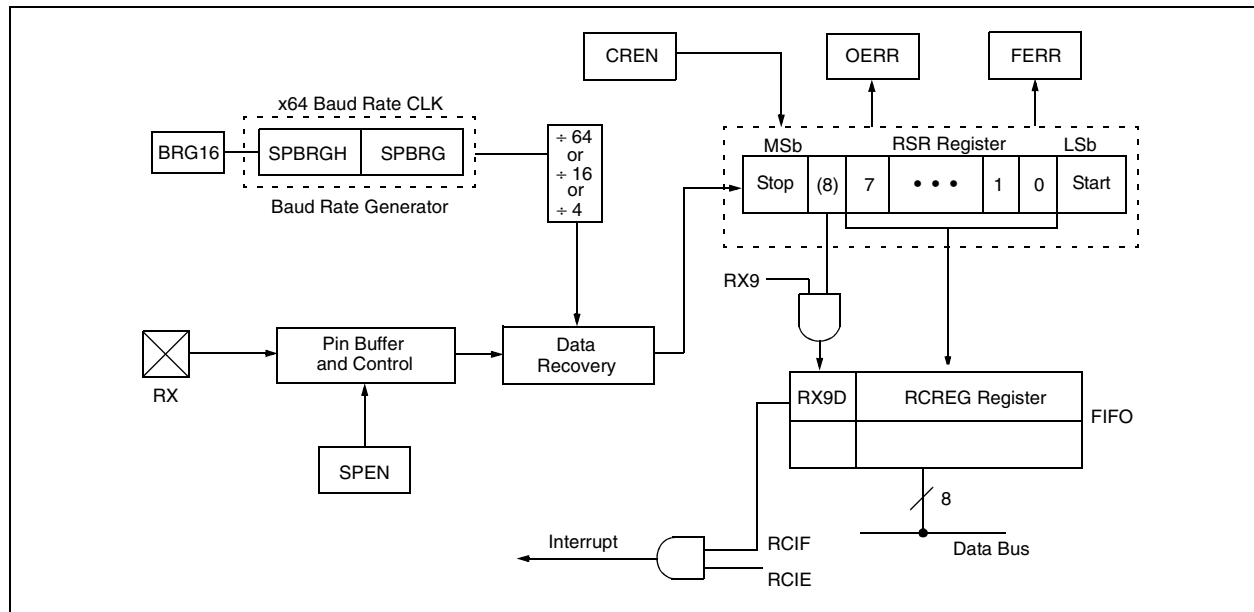
1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit RCIE.
4. If 9-bit 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 9th 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.

16.2.3 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 SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
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 16-6: EUSART RECEIVE BLOCK DIAGRAM



PIC18F45J10 FAMILY

FIGURE 16-7: ASYNCHRONOUS RECEPTION

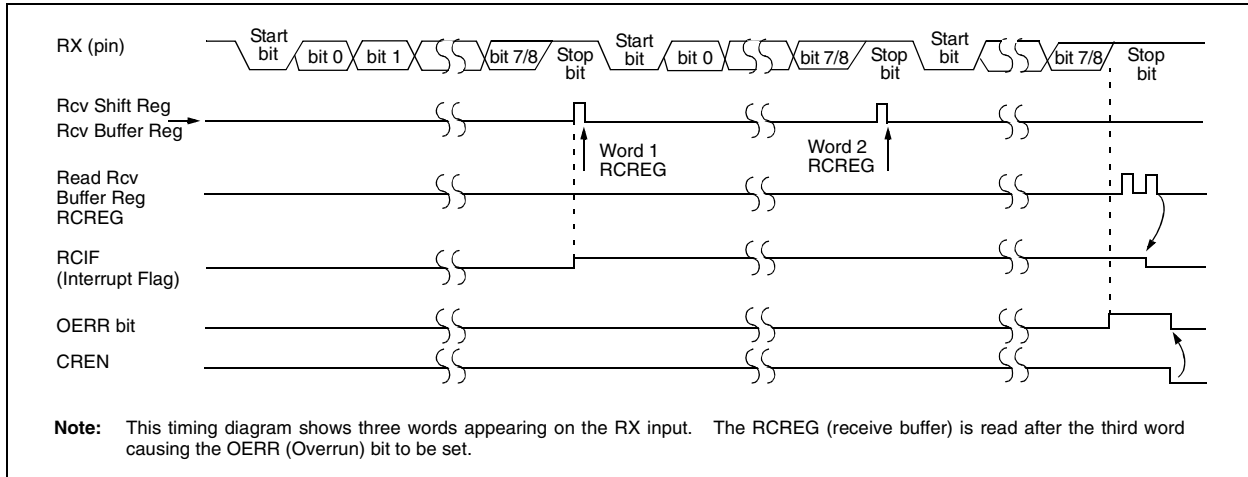


TABLE 16-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| RCREG | EUSART Receive Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

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

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

16.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper byte reception cannot be performed. The auto-wake-up feature allows the controller to wake-up due to activity on the RX/DT line while the EUSART is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCON<1>). Once set, the typical receive sequence on RX/DT is disabled and the EUSART remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RX/DT line. (This coincides with the start of a Sync Break or a Wake-up Signal character for the LIN protocol.)

Following a wake-up event, the module generates an RCIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 16-8) and asynchronously, if the device is in Sleep mode (Figure 16-9). The interrupt condition is cleared by reading the RCREG register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RX line following the wake-up event. At this point, the EUSART module is in Idle mode and returns to normal operation. This signals to the user that the Sync Break event is over.

16.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RX/DT, information with any state changes before the Stop bit may signal a false end-of-character and cause data or framing errors. To work properly, therefore, the initial character in the transmission must be all '0's. This can be 00h (8 bytes) for standard RS-232 devices or 000h (12 bits) for LIN bus.

Oscillator start-up time must also be considered, especially in applications using oscillators with longer start-up intervals (i.e., HS mode). The Sync Break (or Wake-up Signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

16.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the EUSART in an Idle mode. The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared after this when a rising edge is seen on RX/DT. The interrupt condition is then cleared by reading the RCREG register. Ordinarily, the data in RCREG will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set) and the RCIF flag is set should not be used as an indicator of the integrity of the data in RCREG. Users should consider implementing a parallel method in firmware to verify received data integrity.

To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 16-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION

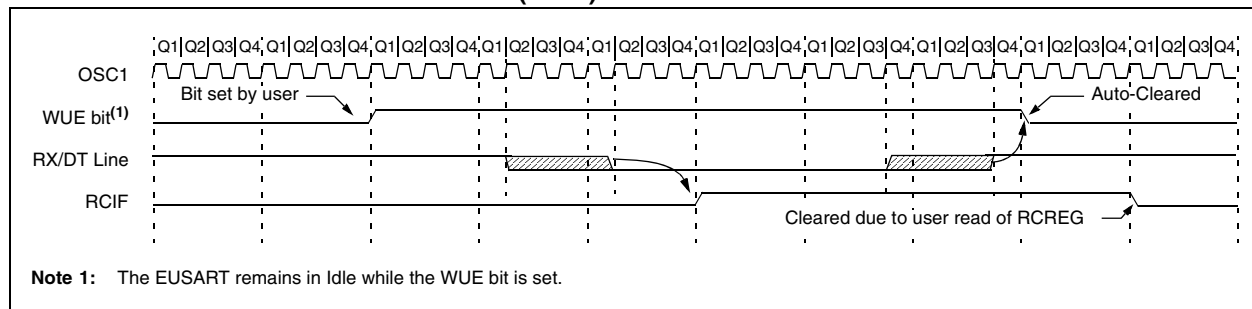
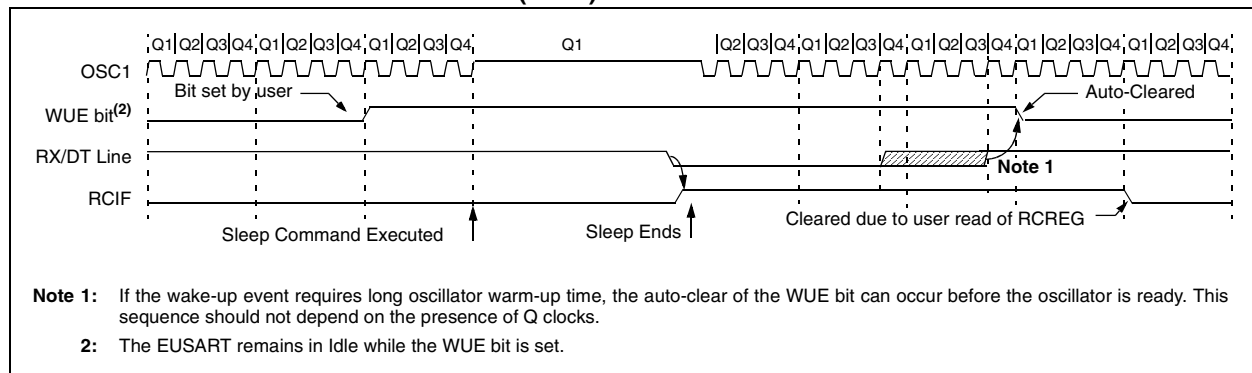


FIGURE 16-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



PIC18F45J10 FAMILY

16.2.5 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The frame Break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift register is loaded with data. Note that the value of data written to TXREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

Note that the data value written to the TXREG for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 16-10 for the timing of the Break character sequence.

16.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREG becomes empty, as indicated by the TXIF, the next data byte can be written to TXREG.

16.2.6 RECEIVING A BREAK CHARACTER

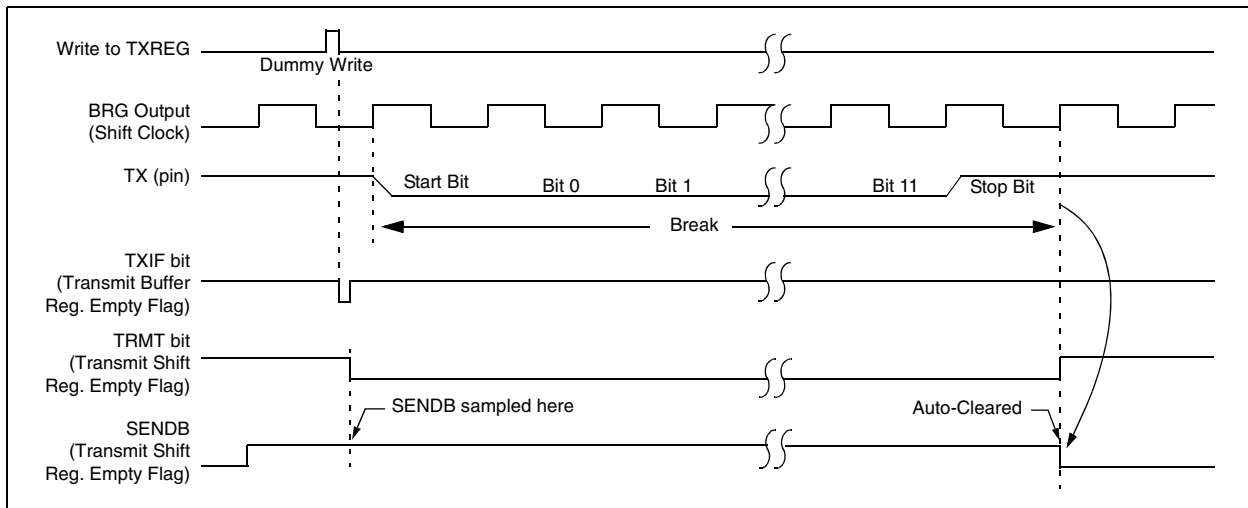
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 16.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABD bit once the TXIF interrupt is observed.

FIGURE 16-10: SEND BREAK CHARACTER SEQUENCE



16.3 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA<7>). In this 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 TX and RX pins to CK (clock) and DT (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK line. Clock polarity is selected with the SCKP bit (BAUDCON<4>). Setting SCKP sets the Idle state on CK as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

16.3.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 16-3. 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).

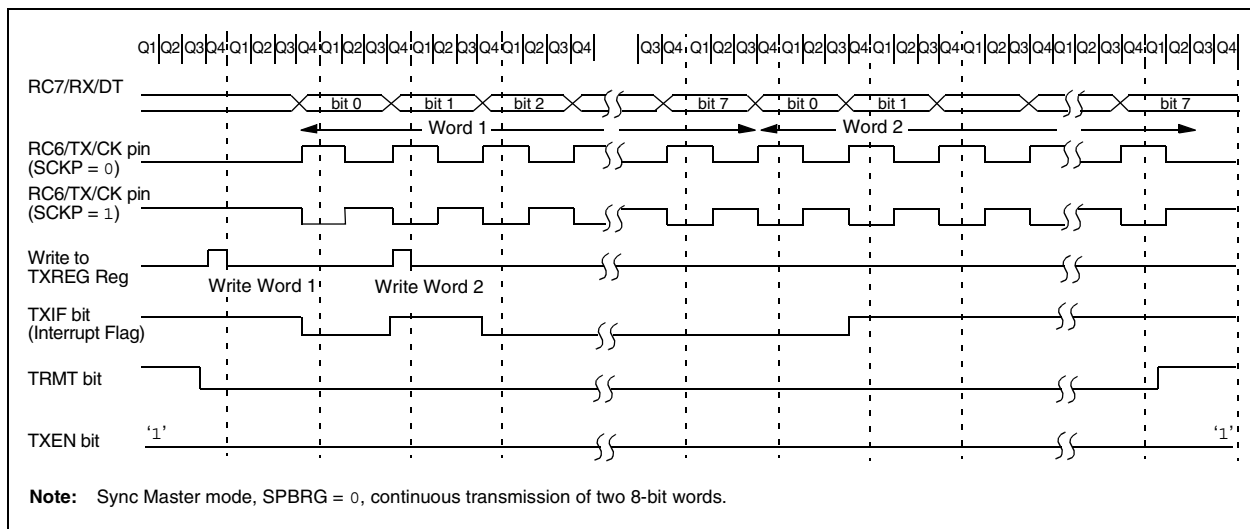
Once the TXREG register transfers the data to the TSR register (occurs in one T_{cy}), the TXREG is empty and the TXIF flag bit (PIR1<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXIE (PIE1<4>). TXIF is set regardless of the state of enable bit TXIE; it 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 SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
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 16-11: SYNCHRONOUS TRANSMISSION



PIC18F45J10 FAMILY

FIGURE 16-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

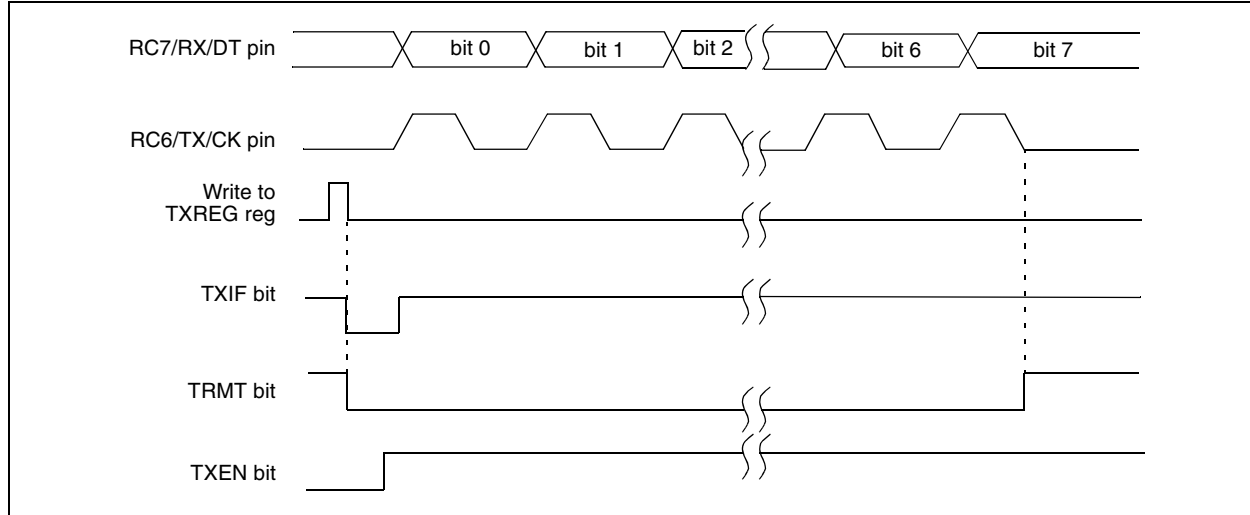


TABLE 16-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| TXREG | EUSART Transmit Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SEnDB | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

16.3.2 EUSART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA<5>), or the Continuous Receive Enable bit, CREN (RCSTA<4>). Data is sampled on the RX 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 SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
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 9th 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 16-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)

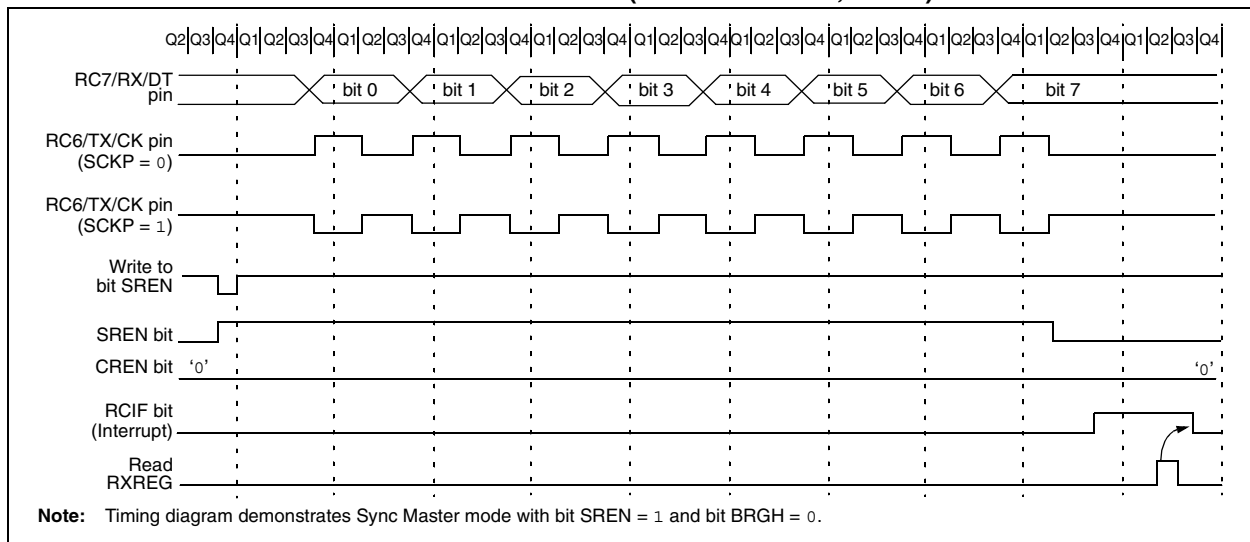


TABLE 16-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBFIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| RCREG | EUSART Receive Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

16.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

16.4.1 EUSART SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

- a) The first word will immediately transfer to the TSR register and transmit.
- b) The second word will remain in the 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 16-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| TXREG | EUSART Transmit Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

Note 1: These bits are not implemented on 28-pin devices and should be read as '0'.

PIC18F45J10 FAMILY

16.4.2 EUSART 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 the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this low-power mode. Once the word is received, the RSR register will transfer the data to the RCREG register; if the RCIE enable bit is set, the interrupt generated will wake the chip from the low-power 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 9th 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 16-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---------|---|-----------|--------|--------|--------|--------|--------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| RCSTA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 45 |
| RCREG | EUSART Receive Register | | | | | | | | 45 |
| TXSTA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 45 |
| BAUDCON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 45 |
| SPBRGH | EUSART Baud Rate Generator Register High Byte | | | | | | | | 45 |
| SPBRG | EUSART Baud Rate Generator Register Low Byte | | | | | | | | 45 |

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

Note 1: These bits are not implemented on 28-pin devices and should be read as ‘0’.

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

17.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has 10 inputs for the 28-pin devices and 13 for the 40/44-pin devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

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 17-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 17-2, configures the functions of the port pins. The ADCON2 register, shown in Register 17-3, configures the A/D clock source, programmed acquisition time and justification.

REGISTER 17-1: ADCON0: A/D CONTROL REGISTER 0

| | | | | | | | |
|-------|-----|-------|-------|-------|-------|---------|-------|
| R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
| ADCAL | — | CHS3 | CHS2 | CHS1 | CHS0 | GO/DONE | ADON |

bit 7

bit 0

bit 7 **ADCAL:** A/D Calibration bit

1 = Calibration is performed on next A/D conversion

0 = Normal A/D converter operation

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

bit 5-2 **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 28-pin devices.

2: Performing a conversion on unimplemented channels will return a floating input measurement.

bit 1 **GO/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

PIC18F45J10 FAMILY

REGISTER 17-2: ADCON1: A/D CONTROL REGISTER 1

| | | | | | | | |
|-------|-----|-------|-------|----------------------|--------------------|--------------------|--------------------|
| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 ⁽¹⁾ | R/W ⁽¹⁾ | R/W ⁽¹⁾ | R/W ⁽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 (VREF- source)
1 = VREF- (AN2)
0 = VSS
- bit 4 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)
1 = VREF+ (AN3)
0 = VDD
- 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 PBADEN configuration bit. When PBADEN = 1, PCFG<3:0> = 0000; when PBADEN = 0, PCFG<3:0> = 0111.

2: AN5 through AN7 are available only on 40/44-pin 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 |

PIC18F45J10 FAMILY

REGISTER 17-3: ADCON2: A/D CONTROL REGISTER 2

| 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 **ADCS2: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 T_{cy} (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

PIC18F45J10 FAMILY

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

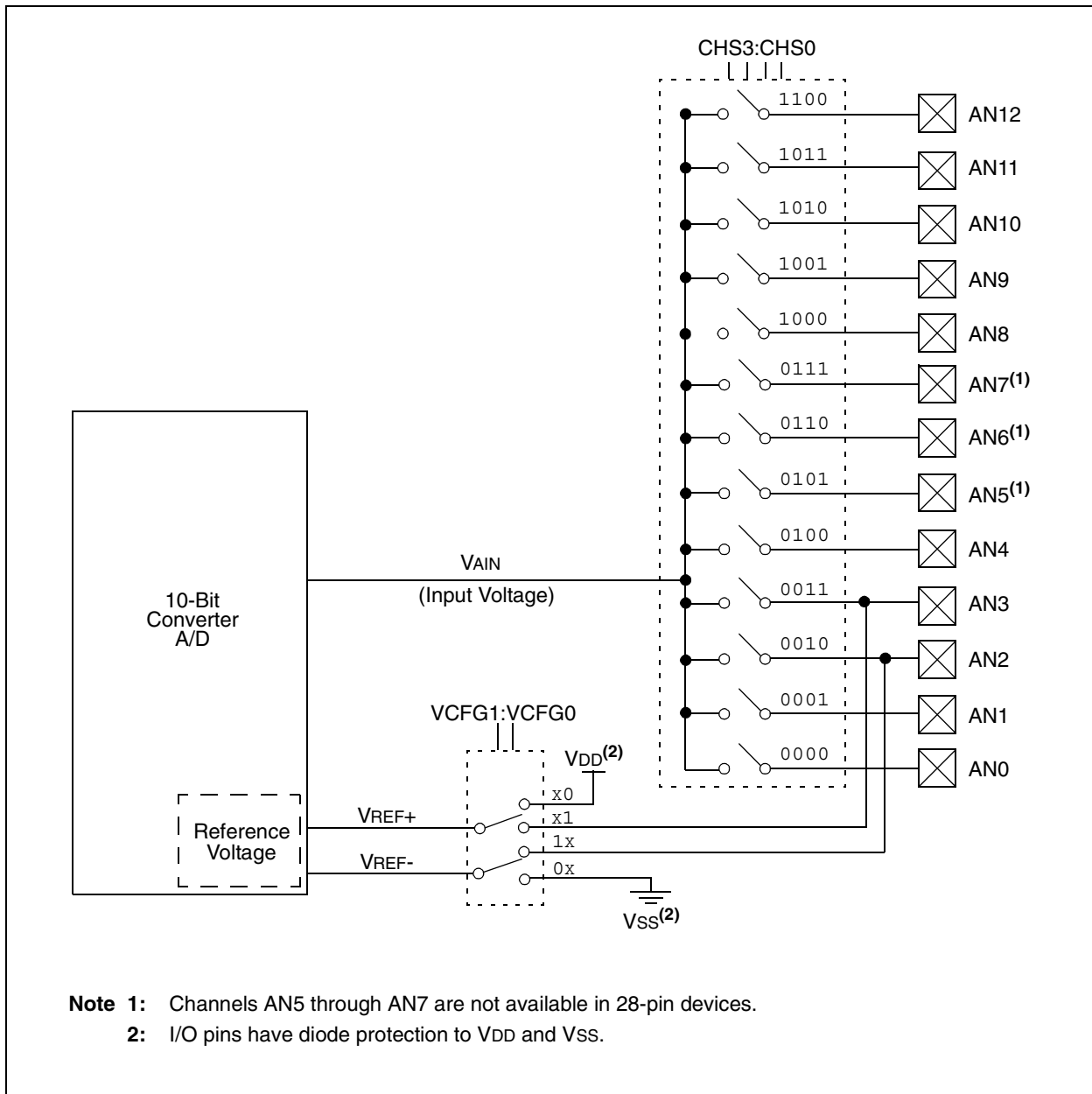
The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

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

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

Each port pin associated with the A/D converter can be configured as an analog input, or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH:ADRESL register pair, the GO/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 17-1.

FIGURE 17-1: A/D BLOCK DIAGRAM



- Note 1:** Channels AN5 through AN7 are not available in 28-pin devices.
Note 2: I/O pins have diode protection to VDD and VSS.

PIC18F45J10 FAMILY

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 17.1 “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/DONE bit and the actual start of the conversion.

The following steps should be followed to perform 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/DONE bit (ADCON0 register)

5. Wait for A/D conversion to complete, by either:
 - Polling for the GO/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 T_{AD}. A minimum wait of 2 T_{AD} is required before the next acquisition starts.

FIGURE 17-2: A/D TRANSFER FUNCTION

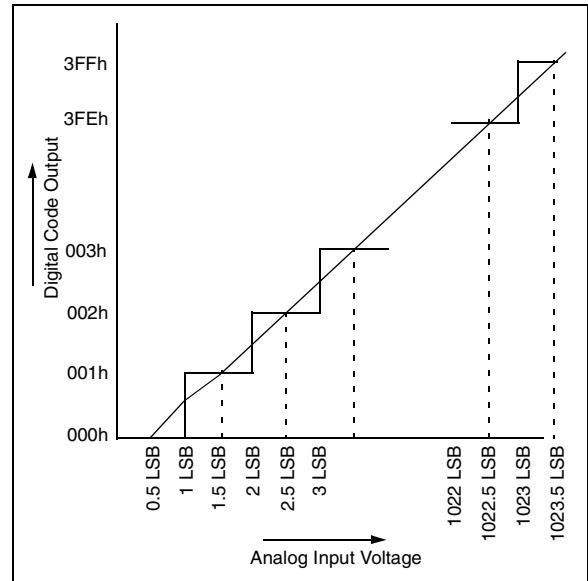
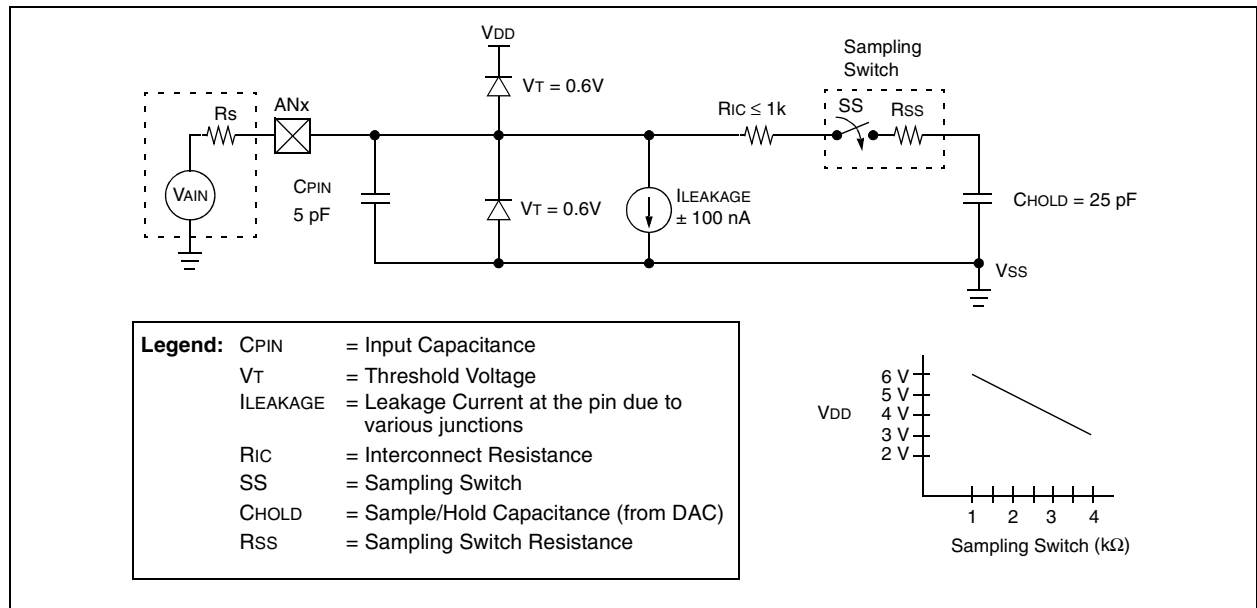


FIGURE 17-3: ANALOG INPUT MODEL



PIC18F45J10 FAMILY

17.1 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 17-3. 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 17-1 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 17-3 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following application system assumptions:

| | | |
|------------------|---|--------------------|
| CHOLD | = | 25 pF |
| Rs | = | 2.5 kΩ |
| Conversion Error | ≤ | 1/2 LSB |
| VDD | = | 5V → Rss = 2 kΩ |
| Temperature | = | 85°C (system max.) |

EQUATION 17-1: ACQUISITION TIME

$$\begin{aligned} \text{TACQ} &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= \text{TAMP} + \text{TC} + \text{TCOFF} \end{aligned}$$

EQUATION 17-2: A/D MINIMUM CHARGING TIME

$$\begin{aligned} \text{V}_{\text{HOLD}} &= (\text{V}_{\text{REF}} - (\text{V}_{\text{REF}}/2048)) \cdot (1 - e^{-(\text{TC}/\text{CHOLD})(\text{RIC} + \text{RSS} + \text{RS})}) \\ \text{or} \\ \text{TC} &= -(\text{CHOLD})(\text{RIC} + \text{RSS} + \text{RS}) \ln(1/2048) \end{aligned}$$

EQUATION 17-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$\begin{aligned} \text{TACQ} &= \text{TAMP} + \text{TC} + \text{TCOFF} \\ \text{TAMP} &= 0.2 \mu\text{s} \\ \text{TCOFF} &= (\text{Temp} - 25^\circ\text{C})(0.02 \mu\text{s}/^\circ\text{C}) \\ &\quad (85^\circ\text{C} - 25^\circ\text{C})(0.02 \mu\text{s}/^\circ\text{C}) \\ &\quad 1.2 \mu\text{s} \\ \text{Temperature coefficient is only required for temperatures} &> 25^\circ\text{C}. \text{ Below } 25^\circ\text{C}, \text{TCOFF} = 0 \text{ ms.} \\ \text{TC} &= -(\text{CHOLD})(\text{RIC} + \text{RSS} + \text{RS}) \ln(1/2047) \mu\text{s} \\ &\quad -(25 \text{ pF})(1 \text{ k}\Omega + 2 \text{ k}\Omega + 2.5 \text{ k}\Omega) \ln(0.0004883) \mu\text{s} \\ &\quad 1.05 \mu\text{s} \\ \text{TACQ} &= 0.2 \mu\text{s} + 1 \mu\text{s} + 1.2 \mu\text{s} \\ &\quad 2.4 \mu\text{s} \end{aligned}$$

17.2 Selecting and Configuring Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time the $\overline{\text{GO/DONE}}$ bit is set. It also gives users the option to use an automatically determined acquisition time.

Acquisition time may be set with the ACQT2:ACQT0 bits (ADCON2<5:3>), which provides a range of 2 to 20 TAD. When the $\overline{\text{GO/DONE}}$ 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 $\overline{\text{GO/DONE}}$ bit.

Manual acquisition is selected when ACQT2:ACQT0 = 000. When the $\overline{\text{GO/DONE}}$ 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 $\overline{\text{GO/DONE}}$ bit. This option is also the default Reset state of the ACQT2:ACQT0 bits and is compatible with devices that do not offer programmable acquisition times.

In either case, when the conversion is completed, the $\overline{\text{GO/DONE}}$ 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.

17.3 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 (see parameter 130 for more information).

Table 17-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 17-1: TAD vs. DEVICE OPERATING FREQUENCIES

| AD Clock Source (TAD) | | Maximum Device Frequency | |
|-----------------------|-------------|--------------------------|-----------------------------------|
| Operation | ADCS2:ADCS0 | PIC18F2XJ10/4XJ10 | PIC18LF2XJ10/4XJ10 ⁽⁴⁾ |
| 2 TOSC | 000 | 2.86 MHz | 1.43 MHz |
| 4 TOSC | 100 | 5.71 MHz | 2.86 MHz |
| 8 TOSC | 001 | 11.43 MHz | 5.72 MHz |
| 16 TOSC | 101 | 22.86 MHz | 11.43 MHz |
| 32 TOSC | 010 | 40.0 MHz | 22.86 MHz |
| 64 TOSC | 110 | 40.0 MHz | 40.0 MHz |
| RC ⁽³⁾ | x11 | 1.00 MHz ⁽¹⁾ | 1.00 MHz ⁽²⁾ |

- Note 1:** The RC source has a typical TAD time of 1.2 μs .
- 2:** The RC source has a typical TAD time of 2.5 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 (PIC18LF2XJ10/4XJ10) devices only.

PIC18F45J10 FAMILY

17.4 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 clock source to be used in that mode. After entering the mode, an A/D acquisition or conversion may be started. Once started, the device should continue to be clocked by the same clock source until the conversion has been completed.

If desired, the device may be placed into the corresponding Idle mode during the conversion. If the device clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in the Sleep mode requires the A/D FRC 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 bit (OSCCON<7>) must have already been cleared prior to starting the conversion.

17.5 A/D Converter Calibration

The A/D converter in the PIC18F45J10 family of devices includes a self-calibration feature which compensates for any offset generated within the module. The calibration process is automated and is initiated by setting the ADCAL bit (ADCON0<7>). The next time the GO/DONE bit is set, the module will perform a "dummy" conversion (that is, with reading none of the input channels) and store the resulting value internally to compensate for offset. Thus, subsequent offsets will be compensated.

The calibration process assumes that the device is in a relatively steady-state operating condition. If A/D calibration is used, it should be performed after each device Reset, or if there are other major changes in operating conditions.

17.6 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 (VOH or VOL) 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 as analog inputs. 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 Configuration Register 3H configures PORTB pins to reset as analog or digital pins by controlling how the PCFG0 bits in ADCON1 are reset.

17.7 A/D Conversions

Figure 17-4 shows the operation of the A/D converter after the $\overline{\text{GO/DONE}}$ 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 17-5 shows the operation of the A/D converter after the $\overline{\text{GO/DONE}}$ 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 $\overline{\text{GO/DONE}}$ 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 $\overline{\text{GO/DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

FIGURE 17-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)

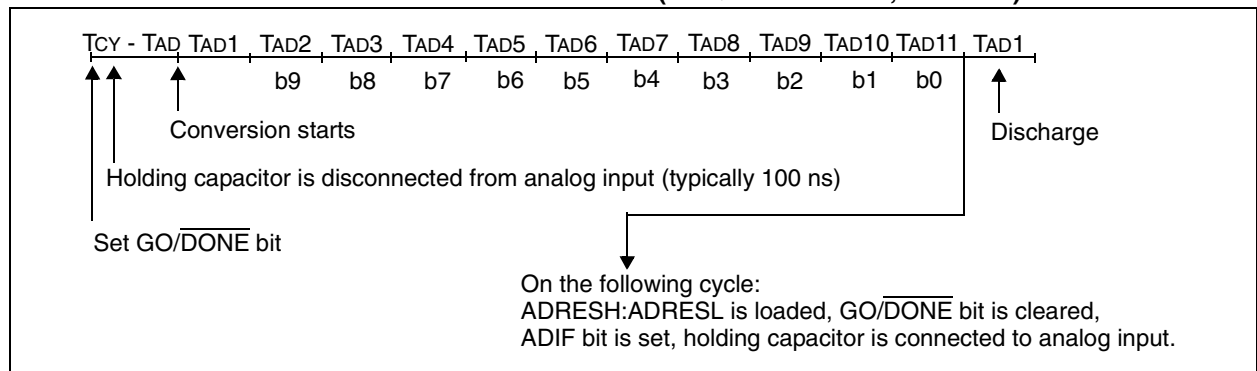
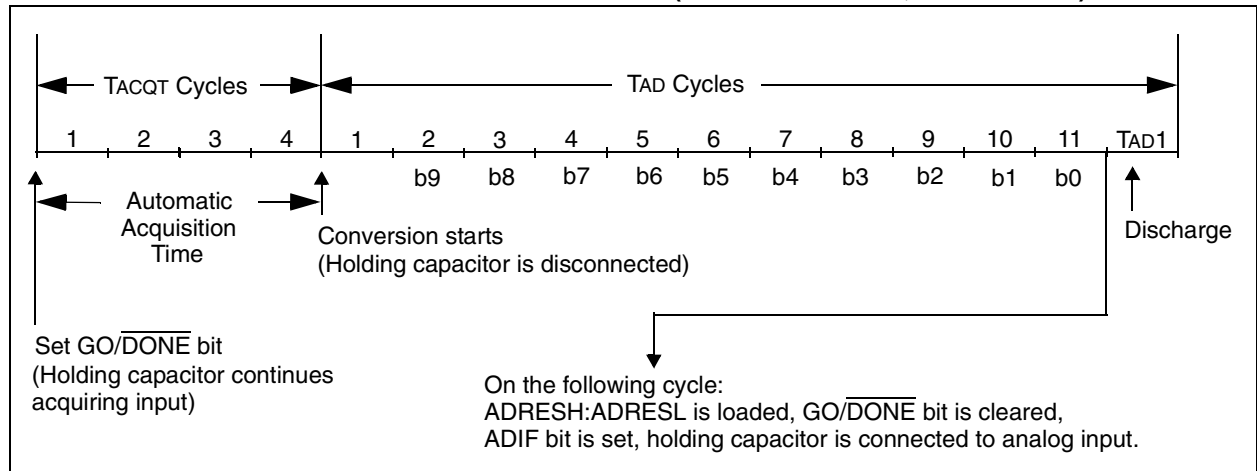


FIGURE 17-5: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)



PIC18F45J10 FAMILY

17.8 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/DONE bit will be set, starting the A/D acquisition and conversion and the Timer1 counter will be reset to zero. Timer1 is reset to automatically repeat the A/D acquisition period with minimal software overhead

(moving 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/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 counter.

TABLE 17-2: REGISTERS ASSOCIATED WITH A/D OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------------------|--|-----------|--------|---------|--------|--|---------|--------|----------------------|
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 43 |
| PIR1 | PSPIF ⁽¹⁾ | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 45 |
| PIE1 | PSPIE ⁽¹⁾ | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 45 |
| IPR1 | PSPIP ⁽¹⁾ | ADIP | RCIP | TXIP | SSP1IP | CCP1IP | TMR2IP | TMR1IP | 45 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| ADRESH | A/D Result Register High Byte | | | | | | | | 44 |
| ADRESL | A/D Result Register Low Byte | | | | | | | | 44 |
| ADCON0 | ADCAL | — | CHS3 | CHS2 | CHS1 | CHS0 | GO/DONE | ADON | 44 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 44 |
| ADCON2 | ADFM | — | ACQT2 | ACQT1 | ACQT0 | ADCS2 | ADCS1 | ADCS0 | 44 |
| PORTA | — | — | RA5 | — | RA3 | RA2 | RA1 | RA0 | 46 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 46 |
| TRISB | PORTB Data Direction Control Register | | | | | | | | 46 |
| LATB | PORTB Data Latch Register (Read and Write to Data Latch) | | | | | | | | 46 |
| PORTE ⁽¹⁾ | — | — | — | — | — | RE2 | RE1 | RE0 | 46 |
| TRISE ⁽¹⁾ | IBF | OBF | IBOV | PSPMODE | — | TRISE2 | TRISE1 | TRISE0 | 46 |
| LATE ⁽¹⁾ | — | — | — | — | — | PORTE Data Latch Register (Read and Write to Data Latch) | | | 46 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: These registers and/or bits are not implemented on 28-pin devices and should be read as '0'.

18.0 COMPARATOR MODULE

The analog comparator module contains two comparators that can be configured in a variety of ways. The inputs can be selected from the analog inputs multiplexed with pins RA0 through RA5, as well as the on-chip voltage reference (see **Section 19.0 “Comparator Voltage Reference Module”**). The digital outputs (normal or inverted) are available at the pin level and can also be read through the control register.

The CMCON register (Register 18-1) selects the comparator input and output configuration. Block diagrams of the various comparator configurations are shown in Figure 18-1.

REGISTER 18-1: CMCON: COMPARATOR CONTROL 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/VREF+
 C2 VIN- connects to RA2/AN2/VREF-/CVREF
 0 = C1 VIN- connects to RA0/AN0
 C2 VIN- connects to RA1/AN1
- bit 2-0 **CM2:CM0**: Comparator Mode bits
 Figure 18-1 shows the Comparator modes and the 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 |

PIC18F45J10 FAMILY

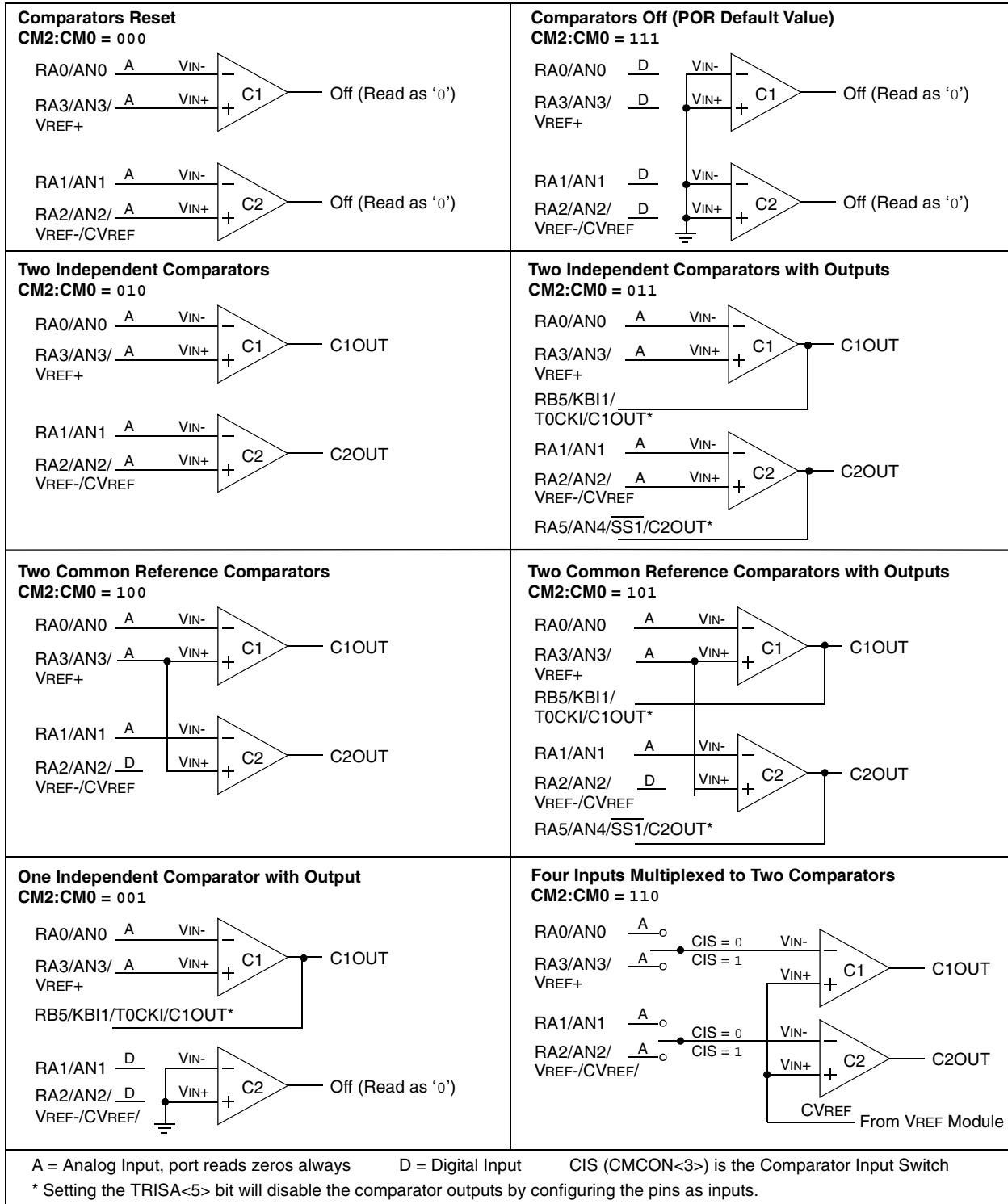
18.1 Comparator Configuration

There are eight modes of operation for the comparators, shown in Figure 18-1. Bits CM2:CM0 of the CMCON register are used to select these 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 Section 23.0 “Electrical Characteristics”.

Note: Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur.

FIGURE 18-1: COMPARATOR I/O OPERATING MODES



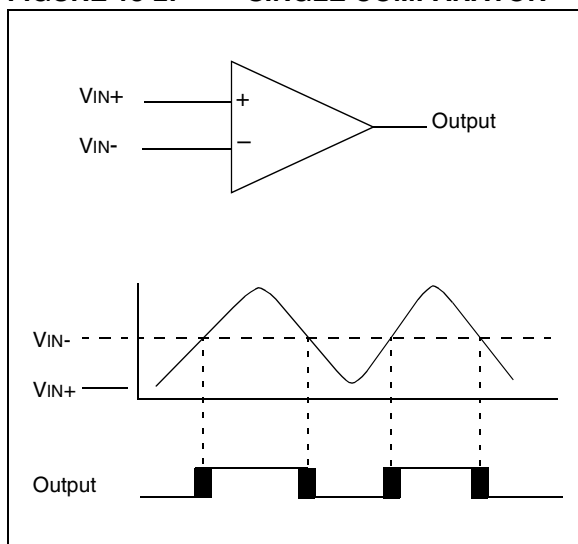
18.2 Comparator Operation

A single comparator is shown in Figure 18-2, along with the relationship between the analog input levels and the digital output. When the analog input at V_{IN+} is less than the analog input V_{IN-} , the output of the comparator is a digital low level. When the analog input at V_{IN+} is greater than the analog input V_{IN-} , the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 18-2 represent the uncertainty, due to input offsets and response time.

18.3 Comparator Reference

Depending on the comparator operating mode, either an external or internal voltage reference may be used. The analog signal present at V_{IN-} is compared to the signal at V_{IN+} and the digital output of the comparator is adjusted accordingly (Figure 18-2).

FIGURE 18-2: SINGLE COMPARATOR



18.3.1 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 V_{SS} and V_{DD} and can be applied to either pin of the comparator(s).

18.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference from the comparator voltage reference module. This module is described in more detail in **Section 19.0 “Comparator Voltage Reference Module”**.

The internal reference is only available in the mode where four inputs are multiplexed to two comparators ($CM2:CM0 = 110$). In this mode, the internal voltage reference is applied to the V_{IN+} pin of both comparators.

18.4 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 **Section 23.0 “Electrical Characteristics”**).

18.5 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 $RB5$ and $RA5$ I/O pins. When enabled, multiplexors in the output path 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 18-3 shows the comparator output block diagram.

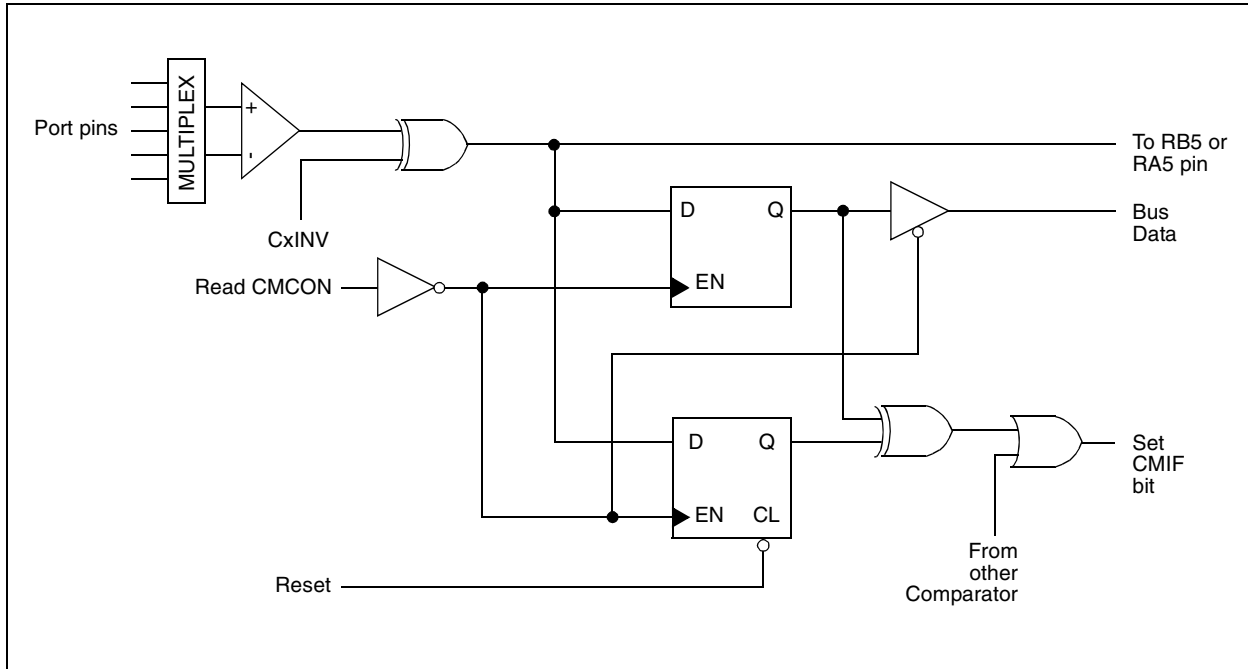
The $TRISA$ bits will still function as an output enable/disable for the $RB5$ and $RA5$ pins while in this mode.

The polarity of the comparator outputs can be changed using the $C2INV$ and $C1INV$ bits ($CMCON<5:4>$).

- | |
|--|
| <p>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.</p> <p>2: Analog levels on any pin defined as a digital input may cause the input buffer to consume more current than is specified.</p> |
|--|

PIC18F45J10 FAMILY

FIGURE 18-3: COMPARATOR OUTPUT BLOCK DIAGRAM



18.6 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 (PIR2<6>) is the Comparator Interrupt Flag. The CMIF bit must be reset by clearing it. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

Both the CMIE bit (PIE2<6>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) 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 (PIR2 register) 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.

18.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from Sleep mode, when enabled. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators (CM2:CM0 = 111) before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

18.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator modules to be turned off (CM2:CM0 = 111). However, the input pins (RA0 through RA3) are configured as analog inputs by default on device Reset. The I/O configuration for these pins is determined by the setting of the PCFG3:PCFG0 bits (ADCON1<3:0>). Therefore, device current is minimized when analog inputs are present at Reset time.

18.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 18-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this

range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 18-4: COMPARATOR ANALOG INPUT MODEL

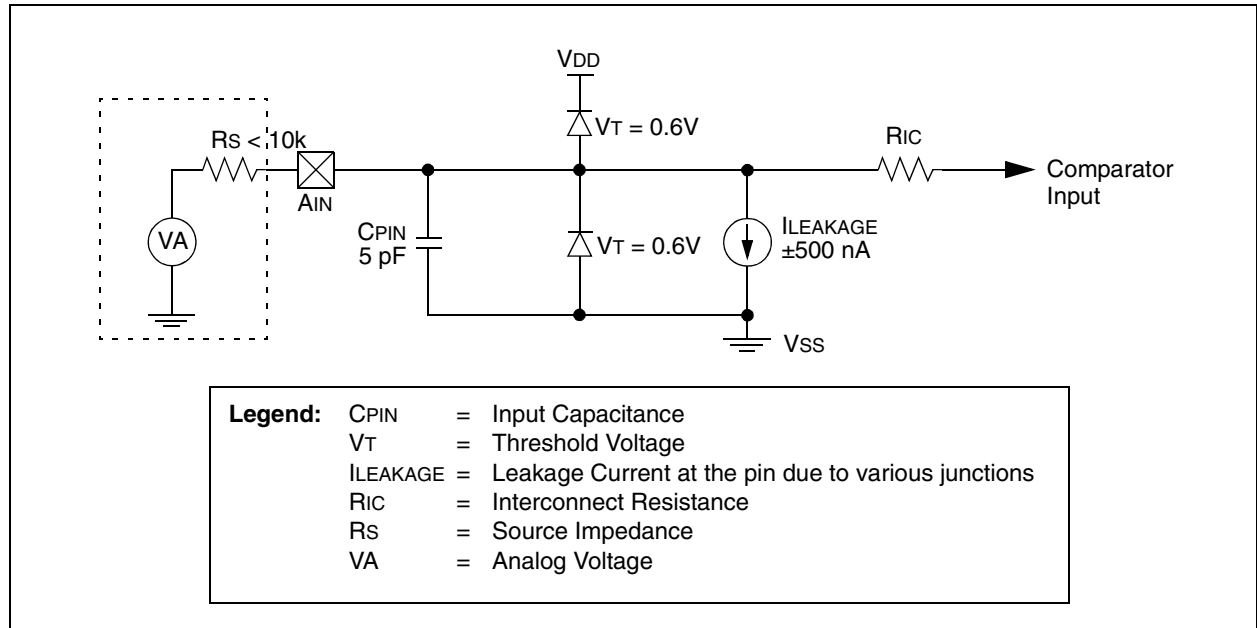


TABLE 18-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|----------|-----------|--|--------|--------|--------|--------|--------|----------------------|
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 45 |
| CVRCON | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 45 |
| INTCON | GIE/GIEH | PEIE/GIEL | TMR0IE | INT0IE | RBIE | TMR0IF | INT0IF | RBIF | 46 |
| PIR2 | OSCFIF | CMIF | — | — | BCL1IF | — | — | CCP2IF | 45 |
| PIE2 | OSCFIE | CMIE | — | — | BCL1IE | — | — | CCP2IE | 45 |
| IPR2 | OSCFIP | CMIP | — | — | BCL1IP | — | — | CCP2IP | 45 |
| PORTA | — | — | RA5 | — | RA3 | RA2 | RA1 | RA0 | 46 |
| LATA | — | — | PORTA Data Latch Register (Read and Write to Data Latch) | | | | | | 46 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |

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

PIC18F45J10 FAMILY

NOTES:

19.0 COMPARATOR VOLTAGE REFERENCE MODULE

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

A block diagram of the module is shown in Figure 19-1. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device VDD/VSS or an external voltage reference.

19.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 19-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be

used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR3:CVR0), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

$$\text{If CVRR} = 1: \\ \text{CVREF} = ((\text{CVR3:CVR0})/24) \times \text{CVRSRC}$$

$$\text{If CVRR} = 0: \\ \text{CVREF} = (\text{CVRSRC} \times 1/4) + (((\text{CVR3:CVR0})/32) \times \text{CVRSRC})$$

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

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 23-3 in **Section 23.0 "Electrical Characteristics"**).

REGISTER 19-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|-------|----------------------|-------|-------|-------|-------|-------|-------|
| CVREN | CVROE ⁽¹⁾ | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 |
| bit 7 | | | | | | | bit 0 |

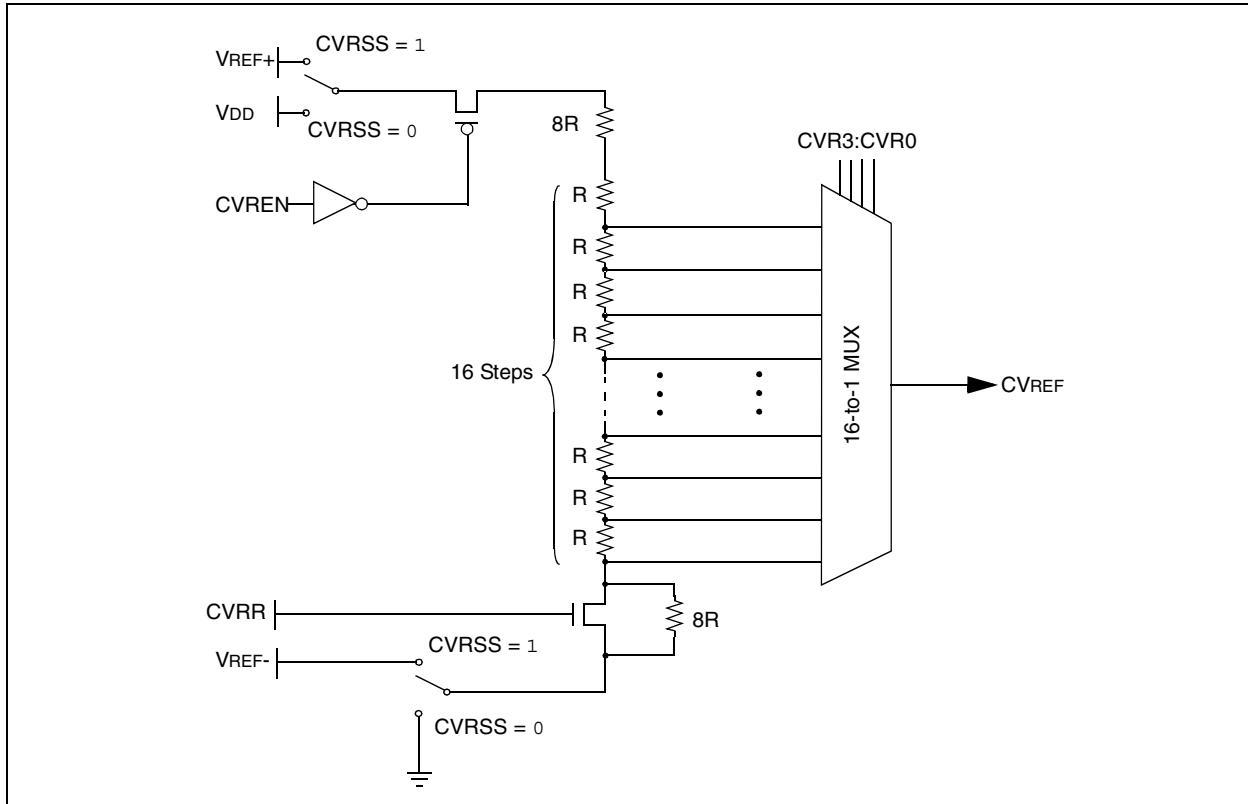
- bit 7 **CVREN:** Comparator Voltage Reference Enable bit
 1 = CVREF circuit powered on
 0 = CVREF circuit powered down
- bit 6 **CVROE:** Comparator VREF Output Enable bit⁽¹⁾
 1 = CVREF voltage level is also output on the RA2/AN2/VREF-/CVREF pin
 0 = CVREF voltage is disconnected from the RA2/AN2/VREF-/CVREF pin
- Note 1:** CVROE overrides the TRISA<2> bit setting.
- bit 5 **CVRR:** Comparator VREF Range Selection bit
 1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)
 0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)
- bit 4 **CVRSS:** Comparator VREF Source Selection bit
 1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)
 0 = Comparator reference source, CVRSRC = VDD – VSS
- bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ (CVR3:CVR0) ≤ 15)
When CVRR = 1:
 CVREF = ((CVR3:CVR0)/24) • (CVRSRC)
When CVRR = 0:
 CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

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 |

PIC18F45J10 FAMILY

FIGURE 19-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



19.2 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 19-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in **Section 23.0 “Electrical Characteristics”**.

19.3 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the CVRCON register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

19.4 Effects of a Reset

A device Reset disables the voltage reference by clearing bit, CVREN (CVRCON<7>). This Reset also disconnects the reference from the RA2 pin by clearing bit, CVROE (CVRCON<6>) and selects the high-voltage range by clearing bit, CVRR (CVRCON<5>). The CVR value select bits are also cleared.

19.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RA2 pin if the CVROE bit is set. Enabling the voltage reference output onto RA2 when it is configured as a digital input will increase current consumption. Connecting RA2 as a digital output with CVRSS enabled will also 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, a buffer must be used on the voltage reference output for external connections to VREF. Figure 19-2 shows an example buffering technique.

PIC18F45J10 FAMILY

FIGURE 19-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE

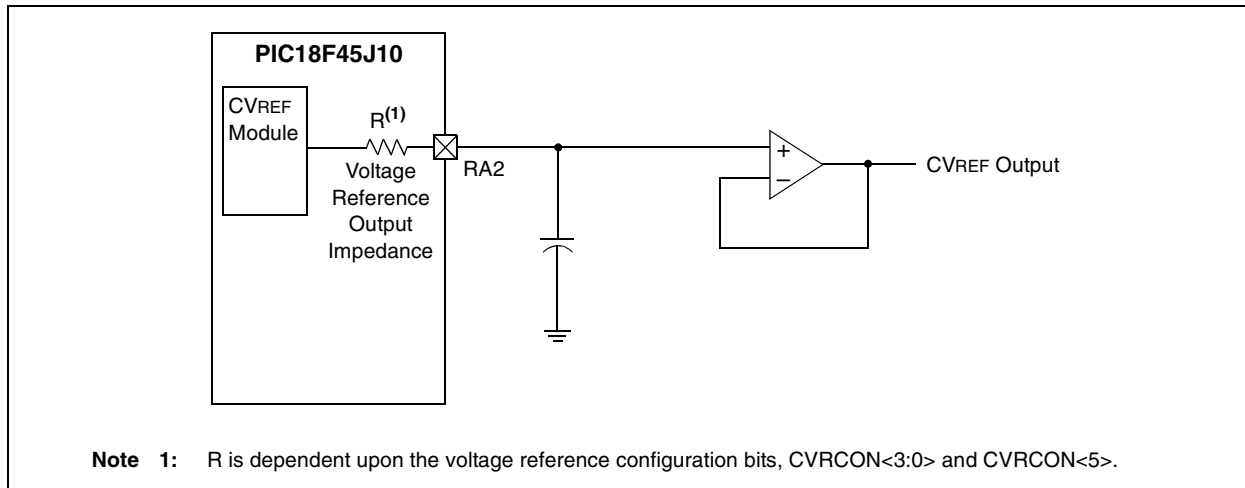


TABLE 19-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|-------|-------|--------|-------|--------|--------|--------|--------|----------------------|
| CVRCON | CVREN | CVROE | CVRR | CVRSS | CVR3 | CVR2 | CVR1 | CVR0 | 45 |
| CMCON | C2OUT | C1OUT | C2INV | C1INV | CIS | CM2 | CM1 | CM0 | 45 |
| TRISA | — | — | TRISA5 | — | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 46 |

Legend: Shaded cells are not used with the comparator voltage reference.

PIC18F45J10 FAMILY

NOTES:

20.0 SPECIAL FEATURES OF THE CPU

PIC18F45J10 family devices include several features intended to maximize 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
- 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, the PIC18F45J10 family of devices have a configurable Watchdog Timer which is controlled in software.

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.

20.1 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. A complete list is shown in Table 20-1. A detailed explanation of the various bit functions is provided in Register 20-1 through Register 20-6.

Note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFFFh) which can only be accessed using table reads and table writes.

20.1.1 CONSIDERATIONS FOR CONFIGURING THE PIC18F45J10 FAMILY DEVICES

Unlike most PIC18 microcontrollers, devices of the PIC18F45J10 family do not use persistent memory registers to store configuration information. The configuration bytes are implemented as volatile memory which means that configuration data must be programmed each time the device is powered up.

Configuration data is stored in the four words at the top of the on-chip program memory space, known as the Flash Configuration Words. It is stored in program memory in the same order shown in Table 20-1, with CONFIG1L at the lowest address and CONFIG3H at the highest. The data is automatically loaded in the proper Configuration registers during device power-up.

When creating applications for these devices, users should always specifically allocate the location of the Flash Configuration Word for configuration data; this is to make certain that program code is not stored in this address when the code is compiled.

The volatile memory cells used for the configuration bits always reset to ‘1’ on Power-on Resets. For all other type of Reset events, the previously programmed values are maintained and used without reloading from program memory.

The four Most Significant bits of CONFIG1H, CONFIG2H and CONFIG3H in program memory should also be ‘1111’. This makes these Configuration Words appear to be NOP instructions in the remote event that their locations are ever executed by accident. Since configuration bits are not implemented in the corresponding locations, writing ‘1’s to these locations has no effect on device operation.

To prevent inadvertent configuration changes during code execution, all programmable configuration bits are write-once. After a bit is initially programmed during a power cycle, it cannot be written to again. Changing a device configuration requires a device Reset.

PIC18F45J10 FAMILY

TABLE 20-1: 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 ⁽¹⁾ |
|-----------|----------|---------------------------|------------------|------------------|------------------|------------------|--------|--------|--------|--|
| 300000h | CONFIG1L | $\overline{\text{DEBUG}}$ | XINST | STVREN | — | — | — | — | WDTEN | 111- ---1 |
| 300001h | CONFIG1H | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | — ⁽³⁾ | CP0 | — | — | ---- x1-- |
| 300002h | CONFIG2L | IESO | FCMEN | — | — | — | FOSC2 | FOSC1 | FOSC0 | 11-- -111 |
| 300003h | CONFIG2H | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | ---- 1111 |
| 300004h | CONFIG3L | — | — | — | — | — | — | — | — | ---- ---- |
| 300005h | CONFIG3H | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | — ⁽²⁾ | — | — | — | CCP2MX | ---- ---1 |
| 3FFFFEh | DEVID1 | DEV2 | DEV1 | DEV0 | REV4 | REV3 | REV2 | REV1 | REV0 | xxxx xxxx ⁽⁴⁾ |
| 3FFFFFh | DEVID2 | DEV10 | DEV9 | DEV8 | DEV7 | DEV6 | DEV5 | DEV4 | DEV3 | 0001 110x ⁽⁴⁾ |

Legend: x = unknown, u = unchanged, - = unimplemented. Shaded cells are unimplemented, read as '0'.

- Note 1:** Values reflect the unprogrammed state as received from the factory and following Power-on Resets. In all other Reset states, the configuration bytes maintain their previously programmed states.
- 2:** The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.
- 3:** This bit should always be maintained as '0'.
- 4:** See Register 20-7 and Register 20-8 for DEVID values. These registers are read-only and cannot be programmed by the user.

PIC18F45J10 FAMILY

REGISTER 20-1: CONFIG1L: CONFIGURATION REGISTER 1 LOW (BYTE ADDRESS 300000h)

| R/WO-1 | R/WO-1 | R/WO-1 | U-0 | U-0 | U-0 | U-0 | R/WO-1 |
|--------|--------|--------|-----|-----|-----|-----|--------|
| DEBUG | XINST | STVREN | — | — | — | — | WDTEN |
| bit 7 | | | | | | | bit 0 |

- bit 7 **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 **XINST:** Extended Instruction Set Enable bit
 1 = Instruction set extension and Indexed Addressing mode enabled
 0 = Instruction set extension and Indexed Addressing mode disabled (Legacy mode)
- bit 5 **STVREN:** Stack Overflow/Underflow Reset Enable bit
 1 = Reset on stack overflow/underflow enabled
 0 = Reset on stack overflow/underflow disabled
- bit 4-1 **Unimplemented:** Read as '0'
- bit 0 **WDTEN:** Watchdog Timer Enable bit
 1 = WDT enabled
 0 = WDT disabled (control is placed on SWDTEN bit)

Legend:

R = Readable bit WO = Write-once bit U = Unimplemented bit, read as '0'
 -n = Value when device is unprogrammed '1' = Bit is set '0' = Bit is cleared

REGISTER 20-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

| U-0 | U-0 | U-0 | U-0 | U-0 | R/WO-1 | U-0 | U-0 |
|-------|-----|-----|-----|------------------|--------|-----|-----|
| — | — | — | — | — ⁽¹⁾ | CP0 | — | — |
| bit 7 | | | | | bit 0 | | |

- bit 7-3 **Unimplemented:** Read as '0'
- bit 2 **CP0:** Code Protection bit
 1 = Program memory is not code-protected
 0 = Program memory is code-protected
- bit 1-0 **Unimplemented:** Read as '0'

Note 1: This bit should always be maintained as '0'.

Legend:

R = Readable bit WO = Write-once bit U = Unimplemented bit, read as '0'
 -n = Value when device is unprogrammed '1' = Bit is set '0' = Bit is cleared

PIC18F45J10 FAMILY

REGISTER 20-3: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)

| R/WO-1 | R/WO-1 | U-0 | U-0 | U-0 | R/WO-1 | R/WO-1 | R/WO-1 |
|--------|--------|-----|-----|-------|--------|--------|--------|
| IESO | FCMEN | — | — | — | FOSC2 | FOSC1 | FOSC0 |
| bit 7 | | | | bit 0 | | | |

- bit 7 **IESO:** Two-Speed Start-up (Internal/External Oscillator Switchover) Control bit
 1 = Two-Speed Start-up enabled
 0 = Two-Speed Start-up disabled
- bit 6 **FCMEN:** Fail-Safe Clock Monitor Enable bit
 1 = Fail-Safe Clock Monitor enabled
 0 = Fail-Safe Clock Monitor disabled
- bit 5-3 **Unimplemented:** Read as '0'
- bit 2 **FOSC2:** Default/Reset System Clock Select bit
 1 = Clock selected by FOSC1:FOSC0 as system clock is enabled when OSCCON<1:0> = 00
 0 = INTRC enabled as system clock when OSCCON<1:0> = 00
- bit 1-0 **FOSC1:FOSC0:** Oscillator Selection bits
 11 = EC oscillator, PLL enabled and under software control, CLKO function on OSC2
 10 = EC oscillator, CLKO function on OSC2
 01 = HS oscillator, PLL enabled and under software control
 00 = HS oscillator

Legend:

R = Readable bit WO = Write-once bit U = Unimplemented bit, read as '0'
 -n = Value when device is unprogrammed '1' = Bit is set '0' = Bit is cleared

REGISTER 20-4: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

| U-0 | U-0 | U-0 | U-0 | R/WO-1 | R/WO-1 | R/WO-1 | R/WO-1 |
|-------|-----|-----|-----|--------|--------|--------|--------|
| — | — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 |
| bit 7 | | | | bit 0 | | | |

- bit 7-4 **Unimplemented:** Read as '0'
- bit 3-0 **WDTPS3:WDTPS0:** 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

Legend:

R = Readable bit WO = Write-once bit U = Unimplemented bit, read as '0'
 -n = Value when device is unprogrammed '1' = Bit is set '0' = Bit is cleared

PIC18F45J10 FAMILY

REGISTER 20-5: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| — | — | — | — | — | — | — | — |

bit 7 bit 0

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

Legend:

R = Readable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

'1' = Bit is set

'0' = Bit is cleared

REGISTER 20-6: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/WO-1 |
|-----|-----|-----|-----|-----|-----|-----|--------|
| — | — | — | — | — | — | — | CCP2MX |

bit 7 bit 0

bit 7-1 **Unimplemented:** Read as '0'

bit 0 **CCP2MX:** CCP2 Mux bit

1 = CCP2 is multiplexed with RC1

0 = CCP2 is multiplexed with RB3

Legend:

R = Readable bit

WO = Write-once bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

'1' = Bit is set

'0' = Bit is cleared

PIC18F45J10 FAMILY

REGISTER 20-7: DEVICE ID REGISTER 1 FOR PIC18F45J10 FAMILY 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

011 = PIC18F4XJ10
 010 = PIC18F2XJ10
 001 = PIC18F4XJ10
 000 = PIC18F2XJ10

Note: Where values for DEV2:DEV0 are shared by more than one device number, the specific device is always identified by using the entire DEV10:DEV0 bit sequence.

bit 4-0 **REV4:REV0:** Revision ID bits

These bits are used to indicate the device revision.

| | |
|--|-------------------------------------|
| Legend: | |
| R = Read-only bit | U = Unimplemented bit, read as '0' |
| -n = Value when device is unprogrammed | u = Unchanged from programmed state |

REGISTER 20-8: DEVICE ID REGISTER 2 FOR PIC18F45J10 FAMILY 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.

0001 1100 = PIC18FX5J10 devices
 0001 1101 = PIC18FX4J10 devices

Note: The values for DEV10:DEV3 may be shared with other device families. The specific device is always identified by using the entire DEV10:DEV0 bit sequence.

| | |
|--|-------------------------------------|
| Legend: | |
| R = Read-only bit | U = Unimplemented bit, read as '0' |
| -n = Value when device is unprogrammed | u = Unchanged from programmed state |

PIC18F45J10 FAMILY

20.2 Watchdog Timer (WDT)

For PIC18F45J10 family devices, the WDT is driven by the INTRC oscillator. 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 multiplexor, controlled by the WDTPS bits in Configuration Register 2H. Available periods range from 4 ms to 131.072 seconds (2.18 minutes). The WDT and postscaler are cleared whenever a SLEEP or CLRWDT instruction is executed, or a clock failure (primary or Timer1 oscillator) has occurred.

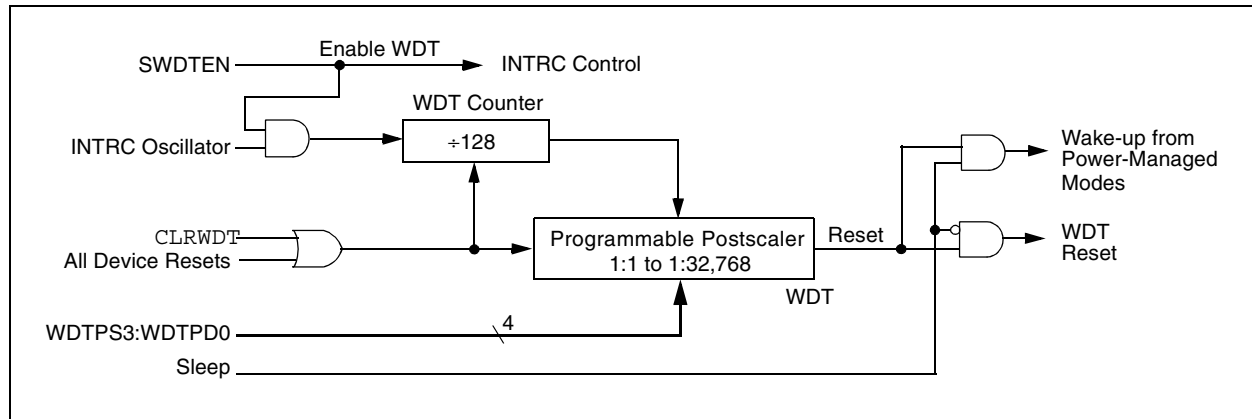
Note 1: The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.

2: When a CLRWDT instruction is executed, the postscaler count will be cleared.

20.2.1 CONTROL REGISTER

The WDTCON register (Register 20-9) is a readable and writable register. The SWDTEN bit enables or disables WDT operation.

FIGURE 20-1: WDT BLOCK DIAGRAM



REGISTER 20-9: WDTCON: WATCHDOG TIMER CONTROL REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|-----------------------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 |
| — | — | — | — | — | — | — | SWDTEN ⁽¹⁾ |
| bit 7 | | | | | | | bit 0 |

bit 7-1 **Unimplemented:** Read as '0'

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit⁽¹⁾

1 = Watchdog Timer is on

0 = Watchdog Timer is off

Note 1: This bit has no effect if the configuration bit, WDTEN, is enabled.

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

TABLE 20-2: SUMMARY OF WATCHDOG TIMER REGISTERS

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|--------|-------|-------|-------|-----------------|-----------------|-----------------|------------------|--------|----------------------|
| RCON | IPEN | — | — | \overline{RI} | \overline{TO} | \overline{PD} | \overline{POR} | BOR | 44 |
| WDTCON | — | — | — | — | — | — | — | SWDTEN | 44 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Watchdog Timer.

PIC18F45J10 FAMILY

20.3 On-Chip Voltage Regulator

Note: The on-chip voltage regulator is only available in parts designated with an “F”, such as PIC18F45J10.

In parts designated “LF”, the microcontroller core can be powered from an external source that is separate from VDD, or it can be powered from an on-chip regulator which derives power from VDD. Both sources use the common VDDCORE/VCAP pin.

In “F” devices, a low ESR capacitor must be connected to the VDDCORE/VCAP pin for proper device operation. In parts designated with an “LF” part number (i.e., PIC18LF45J10), power to the core must be supplied on VDDCORE/VCAP. It is always good design practice to have sufficient capacitance on all supply pins. Examples are shown in Figure 20-2.

Note: In parts designated with an “LF”, such as PIC18LF45J10, VDDCORE must never exceed VDD.

The specifications for core voltage and capacitance are listed in **Section 23.4 “AC (Timing) Characteristics”**.

20.3.1 ON-CHIP REGULATOR AND BOR

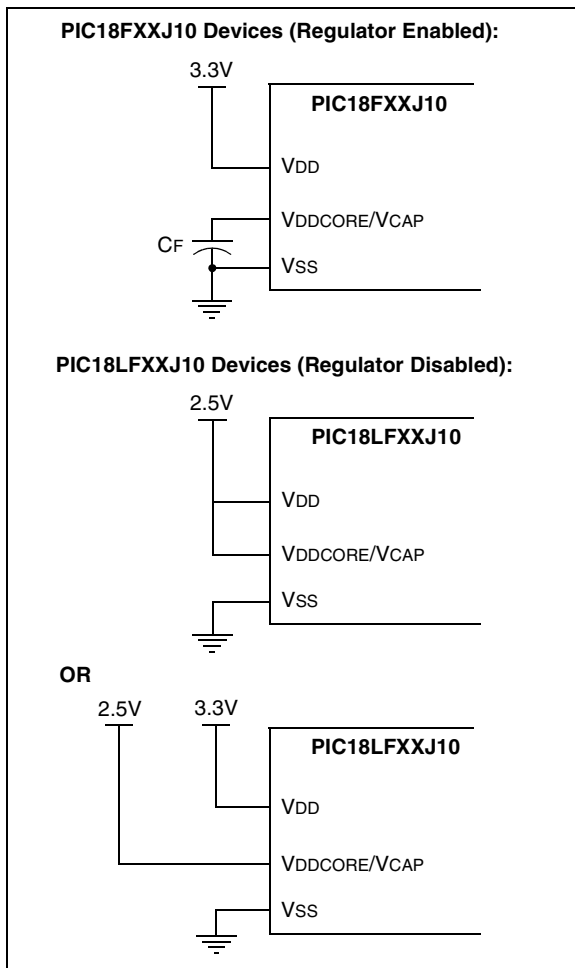
PIC18F45J10 family devices (designated with an “F” part number) also have a simple brown-out capability. If the voltage supplied to the regulator is inadequate to maintain a regulated level, the regulator Reset circuitry will generate a BOR Reset. This event is captured by the $\overline{\text{BOR}}$ flag bit (RCON<0>).

The operation of the BOR is described in more detail in **Section 4.4 “Brown-out Reset (BOR) (PIC18F2XJ10/4XJ10 Devices Only)”** and **Section 4.4.1 “Detecting BOR”**. The brown-out voltage levels are specific in **Section 23.1 “DC Characteristics”**.

20.3.2 POWER-UP REQUIREMENTS

The on-chip regulator is designed to meet the power-up requirements for the device. While powering up, VDDCORE must never exceed VDD by 0.3 volts.

FIGURE 20-2: CONNECTIONS FOR THE ON-CHIP REGULATOR



20.4 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 configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is HS (Crystal-based) modes. Since the EC mode does not require an OST start-up delay, Two-Speed Start-up should be 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.

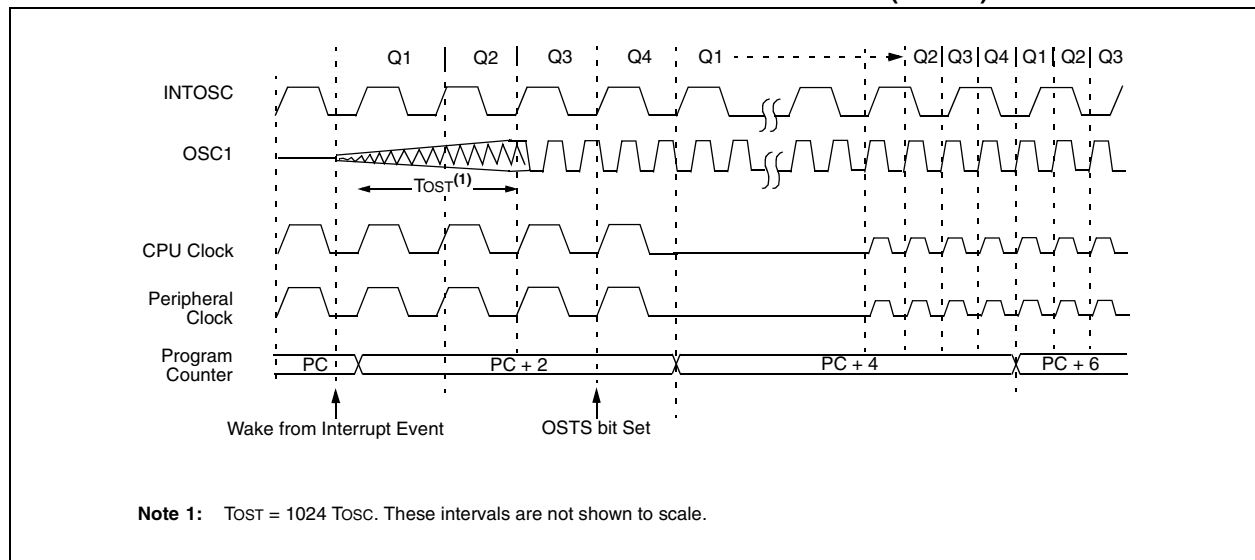
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

20.4.1 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 3.1.4 “Multiple Sleep Commands”**). In practice, this means that user code can change the `SCS1:SCS0` bit settings or issue `SLEEP` instructions 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 device clocking by checking the status of the OSTS bit (`OSCCON<3>`). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

FIGURE 20-3: TIMING TRANSITION FOR TWO-SPEED START-UP (INTRC)



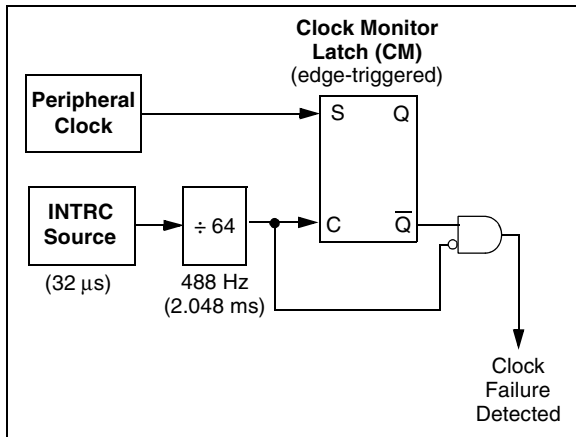
PIC18F45J10 FAMILY

20.5 Fail-Safe Clock Monitor

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

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 20-4) is accomplished by creating a sample clock signal which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the device clock source but cleared on the rising edge of the sample clock.

FIGURE 20-4: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 20-5). This causes the following:

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

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 3.1.4 “Multiple Sleep Commands”** and **Section 20.4.1 “Special Considerations for Using Two-Speed Start-up”** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits IRCF2:IRCF0 immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting IRCF2:IRCF0 prior to entering Sleep mode.

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

20.5.1 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.

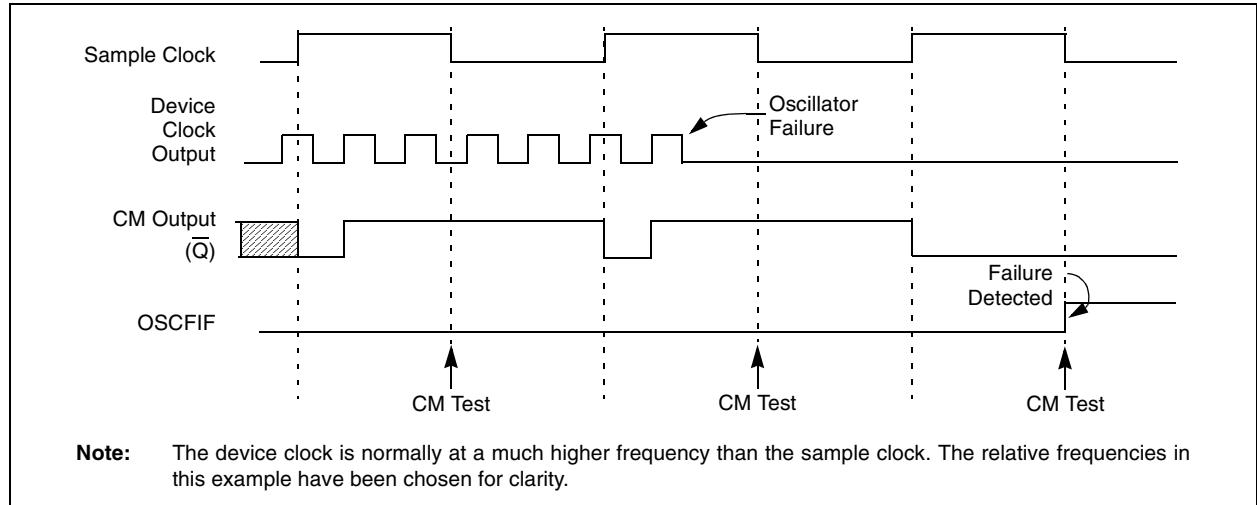
As already noted, the clock source is switched to the INTRC clock when a clock failure is detected; this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

20.5.2 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 2H (with the OST oscillator, start-up delays if running in HS mode). The INTRC oscillator provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock (indicated by the OST 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 INTRC oscillator. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

FIGURE 20-5: FSCM TIMING DIAGRAM



20.5.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexor 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 multiplexor. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, subsequent interrupts while in Idle mode will cause the CPU to begin executing instructions while being clocked by the INTOSC source.

20.5.4 POR OR WAKE-UP 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 device clock is either EC or INTRC modes, monitoring can begin immediately following these events.

For HS mode, the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FSCM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the device clock and functions until the primary clock is stable (the OST timer has 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 20.4.1 "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 clock to become stable. When the new power-managed mode is selected, the primary clock is disabled.

PIC18F45J10 FAMILY

20.6 Program Verification and Code Protection

For all devices in the PIC18F45J10 family of devices, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by one configuration bit, CP0. This bit inhibits external reads and writes to the program memory space. It has no direct effect in normal execution mode.

20.6.1 CONFIGURATION REGISTER PROTECTION

The Configuration registers are protected against untoward changes or reads in two ways. The primary protection is the write-once feature of the configuration bits which prevents reconfiguration once the bit has been programmed during a power cycle. To safeguard against unpredictable events, configuration bit changes resulting from individual cell-level disruptions (such as ESD events) will cause a parity error and trigger a device Reset.

The data for the Configuration registers is derived from the Flash Configuration Words in program memory. When the CP0 bit is set, the source data for device configuration is also protected as a consequence.

20.7 In-Circuit Serial Programming

PIC18F45J10 family 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.

20.8 In-Circuit Debugger

When the DEBUG configuration bit 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 20-3 shows which resources are required by the background debugger.

TABLE 20-3: DEBUGGER RESOURCES

| | |
|-----------------|-----------|
| I/O pins: | RB6, RB7 |
| Stack: | 2 levels |
| Program Memory: | 512 bytes |
| Data Memory: | 32 bytes |

21.0 INSTRUCTION SET SUMMARY

PIC18F45J10 family devices incorporate the standard set of 75 PIC18 core instructions, as well as an extended set of 8 new instructions, for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

21.1 Standard Instruction Set

The standard 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 four 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 21-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 21-1 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 four double-word instructions. These instructions were made double-word to contain the required information in 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 μ s. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μ s. Two-word branch instructions (if true) would take 3 μ s.

Figure 21-1 shows the general formats that the instructions can have. All examples use the convention 'nnh' to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 21-2, lists the standard instructions recognized by the Microchip Assembler (MPASM™).

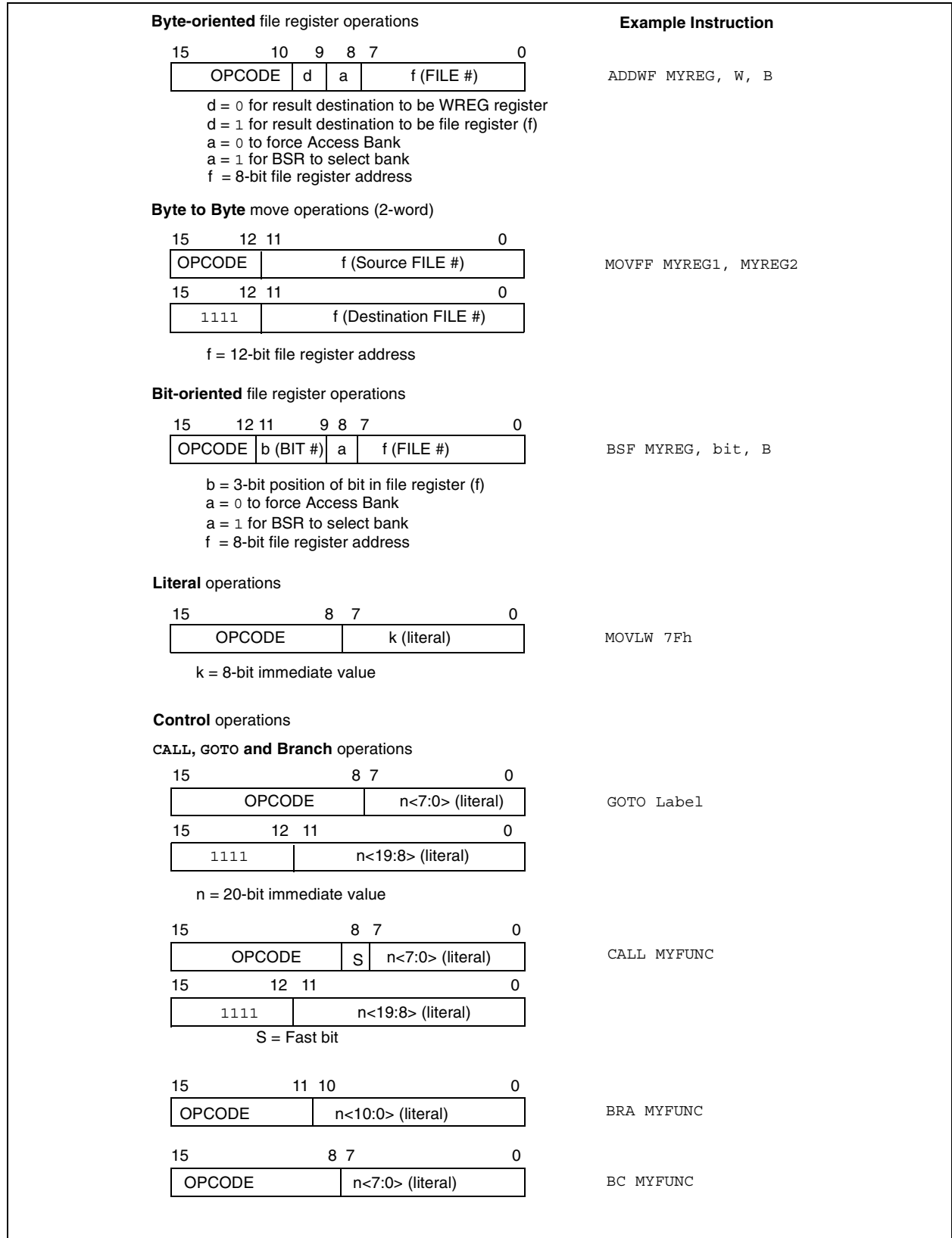
Section 21.1.1 “Standard Instruction Set” provides a description of each instruction.

PIC18F45J10 FAMILY

TABLE 21-1: OPCODE FIELD DESCRIPTIONS

| Field | Description |
|------------------|---|
| a | RAM access bit a = 0: RAM location in Access RAM (BSR register is ignored) 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. |
| C, DC, Z, OV, N | ALU Status bits: C arry, D igit C arry, Z ero, O verflow, N egative. |
| d | Destination select bit d = 0: store result in WREG 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 (00h to FFh) or 2-bit FSR designator (0h to 3h). |
| f _s | 12-bit Register file address (000h to FFFh). This is the source address. |
| f _d | 12-bit Register file address (000h to FFFh). This is the destination address. |
| GIE | Global Interrupt Enable bit. |
| k | Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value). |
| label | Label name. |
| mm | The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: |
| * | No change to register (such as TBLPTR with table reads and writes) |
| *+ | Post-Increment register (such as TBLPTR with table reads and writes) |
| *- | Post-Decrement register (such as TBLPTR with table reads and writes) |
| +* | 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. |
| 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. |
| \overline{PD} | Power-down bit. |
| PRODH | Product of Multiply High Byte. |
| PRODL | Product of Multiply Low Byte. |
| s | Fast Call/Return mode select bit s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode) |
| TBLPTR | 21-bit Table Pointer (points to a Program Memory location). |
| TABLAT | 8-bit Table Latch. |
| T \overline{O} | Time-out bit. |
| TOS | Top-of-Stack. |
| u | Unused or unchanged. |
| WDT | Watchdog Timer. |
| WREG | Working register (accumulator). |
| x | Don't care ('0' or '1'). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools. |
| z _s | 7-bit offset value for indirect addressing of register files (source). |
| z _d | 7-bit offset value for indirect addressing of register files (destination). |
| { } | Optional argument. |
| [text] | Indicates an indexed address. |
| (text) | The contents of text. |
| [expr] <n> | Specifies bit n of the register indicated by the pointer expr. |
| → | Assigned to. |
| < > | Register bit field. |
| ∈ | In the set of. |
| <i>italics</i> | User defined term (font is Courier). |

FIGURE 21-1: GENERAL FORMAT FOR INSTRUCTIONS



PIC18F45J10 FAMILY

TABLE 21-2: PIC18FXXXX INSTRUCTION SET

| Mnemonic, Operands | Description | Cycles | 16-Bit Instruction Word | | | | Status Affected | Notes | |
|---------------------------------|---------------------------------|---|-------------------------|------|------|------|--------------------|-----------------|------------|
| | | | MSb | LSb | | | | | |
| BYTE-ORIENTED 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 f _d (destination) 2nd word | 2 | 1100 | ffff | ffff | 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 | 1, 2 |
| NEGf | f, a | Negate f | 1 | 0110 | 110a | ffff | ffff | C, DC, Z, OV, N | |
| RLCF | f, d, a | Rotate Left f through Carry | 1 | 0011 | 01da | ffff | ffff | C, Z, N | 1, 2 |
| RLNCF | f, d, a | Rotate Left f (No Carry) | 1 | 0100 | 01da | ffff | ffff | Z, N | |
| 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 | 1, 2 |
| SUBFWB | f, d, a | Subtract f from WREG with borrow | 1 | 0101 | 01da | ffff | ffff | C, DC, Z, OV, N | |
| SUBWF | f, d, a | Subtract WREG from f | 1 | 0101 | 11da | ffff | ffff | C, DC, Z, OV, N | 1, 2 |
| SUBWFB | f, d, a | Subtract WREG from f with borrow | 1 | 0101 | 10da | ffff | ffff | C, DC, Z, OV, N | |
| 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 | |

- 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 two-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.

PIC18F45J10 FAMILY

TABLE 21-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

| Mnemonic, Operands | Description | Cycles | 16-Bit Instruction Word | | | | Status Affected | Notes | |
|--------------------------------|-------------|--------------------------------|-------------------------|------|------|------|--------------------|-----------------------------------|------|
| | | | MSb | | | LSb | | | |
| BIT-ORIENTED 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 |
| BTFS | f, b, a | Bit Test f, Skip if Clear | 1 (2 or 3) | 1011 | bbba | ffff | ffff | None | 3, 4 |
| BTFS | 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 |
| 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 | |
| BNV | 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 | 2 | 1110 | 110s | kkkk | kkkk | None | |
| | | 1st word | | | | | | | |
| | | 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 | |
| GOTO | n | Go to address | 2 | 1110 | 1111 | kkkk | kkkk | None | |
| | | 1st word | | | | | | | |
| | | 2nd word | | 1111 | kkkk | kkkk | kkkk | | |
| NOP | — | No Operation | 1 | 0000 | 0000 | 0000 | 0000 | None | |
| NOP | — | No Operation | 1 | 1111 | xxxx | xxxx | xxxx | None | 4 |
| 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 PORTE, 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 two-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.

PIC18F45J10 FAMILY

TABLE 21-2: PIC18FXXX INSTRUCTION SET (CONTINUED)

| Mnemonic, Operands | Description | Cycles | 16-Bit Instruction Word | | | | Status Affected | Notes |
|--|-------------|--|-------------------------|------|------|------|--------------------|-----------------|
| | | | MSb | LSb | | | | |
| 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 FSR(f) 1st word | 2 | 1110 | 1110 | 00ff | 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 | 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 PORTE, 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 two-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.

PIC18F45J10 FAMILY

21.1.1 STANDARD INSTRUCTION SET

ADDLW ADD Literal to W

Syntax: ADDLW k
 Operands: $0 \leq k \leq 255$
 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 15h

Before Instruction
 W = 10h
 After Instruction
 W = 25h

ADDWF ADD W to f

Syntax: ADDWF f {,d {,a}}
 Operands: $0 \leq f \leq 255$
 $d \in [0,1]$
 $a \in [0,1]$
 Operation: $(W) + (f) \rightarrow \text{dest}$
 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1
 Cycles: 1

Q Cycle Activity:

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

Example: ADDWF REG, 0, 0

Before Instruction
 W = 17h
 REG = 0C2h
 After Instruction
 W = 0D9h
 REG = 0C2h

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

PIC18F45J10 FAMILY

ADDWFC **ADD W and Carry bit to f**

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

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

Operation: $(W) + (f) + (C) \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ADDWFC REG, 0, 1

Before Instruction

Carry bit = 1
REG = 02h
W = 4Dh

After Instruction

Carry bit = 0
REG = 02h
W = 50h

ANDLW **AND Literal with W**

Syntax: ANDLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{AND. } k \rightarrow W$

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 05Fh

Before Instruction

W = A3h

After Instruction

W = 03h

PIC18F45J10 FAMILY

ANDWF

AND W with f

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

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

Operation: (W) .AND. (f) → dest

Status Affected: N, Z

Encoding:

| | | | |
|------|------|------|------|
| 0001 | 01da | ffff | ffff |
|------|------|------|------|

Description: The contents of W are ANDed 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ANDWF REG, 0, 0

Before Instruction

W = 17h
 REG = C2h

After Instruction

W = 02h
 REG = C2h

BC

Branch if Carry

Syntax: BC n

Operands: $-128 \leq n \leq 127$

Operation: if Carry bit is '1'
 $(PC) + 2 + 2n \rightarrow PC$

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 5

Before Instruction

PC = address (HERE)

After Instruction

If Carry = 1;
 PC = address (HERE + 12)
 If Carry = 0;
 PC = address (HERE + 2)

PIC18F45J10 FAMILY

BCF

Bit Clear f

Syntax: BCF f, b {,a}

Operands: $0 \leq f \leq 255$
 $0 \leq b \leq 7$
 $a \in [0,1]$

Operation: $0 \rightarrow f \leftarrow b$

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 1001 | bbba | ffff | ffff |
|------|------|------|------|

Description: Bit 'b' in register 'f' is cleared.
 If 'a' is '0', the Access Bank is selected.
 If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: BCF FLAG_REG, 7, 0

Before Instruction
 FLAG_REG = C7h

After Instruction
 FLAG_REG = 47h

BN

Branch if Negative

Syntax: BN n

Operands: $-128 \leq n \leq 127$

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)

PIC18F45J10 FAMILY

BNC Branch if Not Carry

Syntax: BNC n

Operands: $-128 \leq n \leq 127$

Operation: if Carry bit is '0'
 $(PC) + 2 + 2n \rightarrow PC$

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: BNN n

Operands: $-128 \leq n \leq 127$

Operation: if Negative bit is '0'
 $(PC) + 2 + 2n \rightarrow PC$

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)

PIC18F45J10 FAMILY

BNOV Branch if Not Overflow

Syntax: BNOV n

Operands: $-128 \leq n \leq 127$

Operation: if Overflow bit is '0'
 $(PC) + 2 + 2n \rightarrow PC$

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: BNZ n

Operands: $-128 \leq n \leq 127$

Operation: if Zero bit is '0'
 $(PC) + 2 + 2n \rightarrow PC$

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)

PIC18F45J10 FAMILY

BRA Unconditional Branch

Syntax: BRA n

Operands: $-1024 \leq n \leq 1023$

Operation: $(PC) + 2 + 2n \rightarrow PC$

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: BSF f, b {,a}

Operands: $0 \leq f \leq 255$
 $0 \leq b \leq 7$
 $a \in [0,1]$

Operation: $1 \rightarrow f\langle b \rangle$

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 1000 | bbba | ffff | ffff |
|------|------|------|------|

Description: Bit 'b' in register 'f' is set. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example:

```

BSF    FLAG_REG, 7, 1
Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah
    
```

PIC18F45J10 FAMILY

BTFSK Bit Test File, Skip if Clear

Syntax: BTFSK f, b {,a}

Operands: $0 \leq f \leq 255$
 $0 \leq b \leq 7$
 $a \in [0,1]$

Operation: skip if (f) = 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 '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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh).
 See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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 BTFSK FLAG, 1, 0
 FALSE :
 TRUE :

Before Instruction
 PC = address (HERE)

After Instruction
 If FLAG<1> = 0;
 PC = address (TRUE)
 If FLAG<1> = 1;
 PC = address (FALSE)

BTFSK Bit Test File, Skip if Set

Syntax: BTFSK f, b {,a}

Operands: $0 \leq f \leq 255$
 $0 \leq b < 7$
 $a \in [0,1]$

Operation: skip if (f) = 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 '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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh).
 See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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 BTFSK FLAG, 1, 0
 FALSE :
 TRUE :

Before Instruction
 PC = address (HERE)

After Instruction
 If FLAG<1> = 0;
 PC = address (FALSE)
 If FLAG<1> = 1;
 PC = address (TRUE)

PIC18F45J10 FAMILY

BTG **Bit Toggle f**

Syntax: BTG f, b {,a}

Operands: $0 \leq f \leq 255$
 $0 \leq b < 7$
 $a \in [0,1]$

Operation: $(\bar{f} < b >) \rightarrow f < b >$

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 is selected.
If 'a' is '1', the BSR is used to select the GPR bank (default).
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: BTG PORTC, 4, 0

Before Instruction:
PORTC = 0111 0101 [75h]

After Instruction:
PORTC = 0110 0101 [65h]

BOV **Branch if Overflow**

Syntax: BOV n

Operands: $-128 \leq n \leq 127$

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
PC = 0;
PC = address (HERE + 2)

PIC18F45J10 FAMILY

BZ Branch if Zero

Syntax: BZ n

Operands: $-128 \leq n \leq 127$

Operation: if Zero bit is '1'
 $(PC) + 2 + 2n \rightarrow PC$

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: CALL k {,s}

Operands: $0 \leq k \leq 1048575$
 $s \in [0,1]$

Operation: $(PC) + 4 \rightarrow TOS$,
 $k \rightarrow PC<20:1>$,
 if $s = 1$
 $(W) \rightarrow WS$,
 $(STATUS) \rightarrow STATUSS$,
 $(BSR) \rightarrow BSRS$

Status Affected: None

Encoding:

| | | | |
|------|-------------|----------|----------|
| 1110 | 110s | k_7kkk | $kkkk_0$ |
| 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, 1

Before Instruction
 PC = address (HERE)

After Instruction
 PC = address (THERE)
 TOS = address (HERE + 4)
 WS = W
 BSRS = BSR
 STATUSS = STATUS

PIC18F45J10 FAMILY

| | | | | | | | | | |
|-------------------|--|--------------|--------------------|------|------|--------|-------------------|--------------|--------------------|
| CLRF | Clear f | | | | | | | | |
| Syntax: | CLRF f{,a} | | | | | | | | |
| Operands: | 0 ≤ f ≤ 255 a ∈ [0,1] | | | | | | | | |
| Operation: | 000h → f 1 → Z | | | | | | | | |
| Status Affected: | Z | | | | | | | | |
| Encoding: | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px;">0110</td> <td style="padding: 2px;">101a</td> <td style="padding: 2px;">ffff</td> <td style="padding: 2px;">ffff</td> </tr> </table> | 0110 | 101a | ffff | ffff | | | | |
| 0110 | 101a | ffff | ffff | | | | | | |
| Description: | Clears the contents of the specified register. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode" for details. | | | | | | | | |
| Words: | 1 | | | | | | | | |
| Cycles: | 1 | | | | | | | | |
| Q Cycle Activity: | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px;">Q1</td> <td style="padding: 2px;">Q2</td> <td style="padding: 2px;">Q3</td> <td style="padding: 2px;">Q4</td> </tr> <tr> <td style="padding: 2px;">Decode</td> <td style="padding: 2px;">Read register 'f'</td> <td style="padding: 2px;">Process Data</td> <td style="padding: 2px;">Write register 'f'</td> </tr> </table> | Q1 | Q2 | Q3 | Q4 | Decode | Read register 'f' | Process Data | Write register 'f' |
| Q1 | Q2 | Q3 | Q4 | | | | | | |
| Decode | Read register 'f' | Process Data | Write register 'f' | | | | | | |

Example: CLRF FLAG_REG, 1

Before Instruction
FLAG_REG = 5Ah

After Instruction
FLAG_REG = 00h

| | | | | | | | | | |
|-------------------|--|--------------|--------------|------|------|--------|--------------|--------------|--------------|
| CLRWDT | Clear Watchdog Timer | | | | | | | | |
| Syntax: | CLRWDT | | | | | | | | |
| Operands: | None | | | | | | | | |
| Operation: | 000h → WDT, 000h → WDT postscaler, 1 → \overline{TO} , 1 → PD | | | | | | | | |
| Status Affected: | \overline{TO} , PD | | | | | | | | |
| Encoding: | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0100</td> </tr> </table> | 0000 | 0000 | 0000 | 0100 | | | | |
| 0000 | 0000 | 0000 | 0100 | | | | | | |
| Description: | CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits, \overline{TO} and PD, are set. | | | | | | | | |
| Words: | 1 | | | | | | | | |
| Cycles: | 1 | | | | | | | | |
| Q Cycle Activity: | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px;">Q1</td> <td style="padding: 2px;">Q2</td> <td style="padding: 2px;">Q3</td> <td style="padding: 2px;">Q4</td> </tr> <tr> <td style="padding: 2px;">Decode</td> <td style="padding: 2px;">No operation</td> <td style="padding: 2px;">Process Data</td> <td style="padding: 2px;">No operation</td> </tr> </table> | Q1 | Q2 | Q3 | Q4 | Decode | No operation | Process Data | No operation |
| Q1 | Q2 | Q3 | Q4 | | | | | | |
| Decode | No operation | Process Data | No operation | | | | | | |

Example: CLRWDT

Before Instruction
WDT Counter = ?

After Instruction
WDT Counter = 00h
WDT Postscaler = 0
 \overline{TO} = 1
PD = 1

PIC18F45J10 FAMILY

COMF Complement f

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

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

Operation: $(\bar{f}) \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: COMF REG, 0, 0

Before Instruction
 REG = 13h

After Instruction
 REG = 13h
 W = ECh

CPFSEQ Compare f with W, Skip if f = W

Syntax: CPFSEQ f {,a}

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

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)

Note: 3 cycles if skip and followed by a 2-word instruction.

Words: 1
 Cycles: 1(2)

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, 0
 NEQUAL :
 EQUAL :

Before Instruction
 PC Address = HERE
 W = ?
 REG = ?

After Instruction
 If REG = W;
 PC = Address (EQUAL)

If REG \neq W;
 PC = Address (NEQUAL)

PIC18F45J10 FAMILY

CPFSGT Compare f with W, Skip if f > W

Syntax: CPFSGT f {,a}
Operands: $0 \leq f \leq 255$
 $a \in [0,1]$
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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 0
 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: CPFSLT f {,a}
Operands: $0 \leq f \leq 255$
 $a \in [0,1]$
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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (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, 1
 NLESS :
 LESS :

Before Instruction
 PC = Address (HERE)
 W = ?
After Instruction
 If REG < W;
 PC = Address (LESS)
 If REG ≥ W;
 PC = Address (NLESS)

PIC18F45J10 FAMILY

DAW Decimal Adjust W Register

Syntax: DAW

Operands: None

Operation: If $[W<3:0> > 9]$ or $[DC = 1]$ then $(W<3:0>) + 6 \rightarrow W<3:0>;$
 else $(W<3:0>) \rightarrow W<3:0>$

If $[W<7:4> + DC > 9]$ or $[C = 1]$ then $(W<7:4>) + 6 + DC \rightarrow W<7:4>;$
 else $(W<7:4>) + DC \rightarrow W<7:4>$

Status Affected: C

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.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|-----------------|--------------|---------|
| Decode | Read register W | Process Data | Write W |

Example 1:

DAW

Before Instruction

W = A5h
 C = 0
 DC = 0

After Instruction

W = 05h
 C = 1
 DC = 0

Example 2:

Before Instruction

W = CEh
 C = 0
 DC = 0

After Instruction

W = 34h
 C = 1
 DC = 0

DECF Decrement f

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

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

Operation: $(f) - 1 \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: DECF CNT, 1, 0

Before Instruction

CNT = 01h
 Z = 0

After Instruction

CNT = 00h
 Z = 1

PIC18F45J10 FAMILY

DECFSZ **Decrement f, Skip if 0**

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

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

Operation: $(f) - 1 \rightarrow \text{dest}$,
 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 1, 1
            GOTO      LOOP
            CONTINUE
  
```

Before Instruction
 PC = Address (HERE)

After Instruction
 CNT = CNT - 1
 If CNT = 0;
 PC = Address (CONTINUE)
 If CNT \neq 0;
 PC = Address (HERE + 2)

DCFSNZ **Decrement f, Skip if not 0**

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

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

Operation: $(f) - 1 \rightarrow \text{dest}$,
 skip if result \neq 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 1, 0
ZERO      :
NZERO     :
  
```

Before Instruction
 TEMP = ?

After Instruction
 TEMP = TEMP - 1,
 If TEMP = 0;
 PC = Address (ZERO)
 If TEMP \neq 0;
 PC = Address (NZERO)

PIC18F45J10 FAMILY

GOTO Unconditional Branch

Syntax: GOTO k

Operands: $0 \leq k \leq 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word ($k<7:0>$)

2nd word ($k<19:8>$)

| | | | |
|------|-------------|----------|----------|
| 1110 | 1111 | k_7kkk | $kkkk_0$ |
| 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: INCF f {,d {,a}}

Operands: $0 \leq f \leq 255$

$d \in [0,1]$

$a \in [0,1]$

Operation: $(f) + 1 \rightarrow \text{dest}$

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

Encoding:

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

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: INCF CNT, 1, 0

Before Instruction

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

After Instruction

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

PIC18F45J10 FAMILY

INCFSZ Increment f, Skip if 0

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

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

Operation: $(f) + 1 \rightarrow \text{dest}$,
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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 1, 0
 NZERO :
 ZERO :

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT \neq 0;
PC = Address (NZERO)

INFSNZ Increment f, Skip if not 0

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

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

Operation: $(f) + 1 \rightarrow \text{dest}$,
skip if result \neq 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 1, 0
 ZERO :
 NZERO :

Before Instruction
PC = Address (HERE)

After Instruction
REG = REG + 1
If REG \neq 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

PIC18F45J10 FAMILY

IORLW Inclusive OR Literal with W

Syntax: IORLW k
 Operands: $0 \leq k \leq 255$
 Operation: (W) .OR. k \rightarrow W
 Status Affected: N, Z
 Encoding:

| | | | |
|------|------|------|------|
| 0000 | 1001 | kkkk | kkkk |
|------|------|------|------|

 Description: The contents of W are ORed 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 35h

Before Instruction
 W = 9Ah
 After Instruction
 W = BFh

IORWF Inclusive OR W with f

Syntax: IORWF f {,d {,a}}
 Operands: $0 \leq f \leq 255$
 $d \in [0,1]$
 $a \in [0,1]$
 Operation: (W) .OR. (f) \rightarrow 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1
 Cycles: 1

Q Cycle Activity:

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

Example: IORWF RESULT, 0, 1

Before Instruction
 RESULT = 13h
 W = 91h
 After Instruction
 RESULT = 13h
 W = 93h

PIC18F45J10 FAMILY

LFSR Load FSR

Syntax: LFSR f, k

Operands: $0 \leq f \leq 2$
 $0 \leq k \leq 4095$

Operation: $k \rightarrow \text{FSRf}$

Status Affected: None

Encoding:

| | | | |
|------|------|----------|-------------|
| 1110 | 1110 | 00ff | $k_{11}kkk$ |
| 1111 | 0000 | k_7kkk | $kkkk$ |

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, 3ABh

After Instruction

FSR2H = 03h
 FSR2L = ABh

MOVF Move f

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

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

Operation: $f \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: MOVF REG, 0, 0

Before Instruction

REG = 22h
 W = FFh

After Instruction

REG = 22h
 W = 22h

PIC18F45J10 FAMILY

MOVFF Move f to f

Syntax: MOVFF f_s, f_d

Operands: $0 \leq f_s \leq 4095$
 $0 \leq f_d \leq 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

| | | | | |
|--------------------|------|------|------|------------|
| Encoding: | | | | |
| 1st word (source) | 1100 | ffff | ffff | ffff f_s |
| 2nd word (destin.) | 1111 | ffff | ffff | ffff f_d |

Description: The contents of source register ' f_s ' are moved to destination register ' f_d '. Location of source ' f_s ' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination ' f_d ' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|-------------------------------|--------------|-------------------------------|
| Decode | Read register ' f ' (src) | Process Data | No operation |
| Decode | No operation No dummy read | No operation | Write register ' f ' (dest) |

Example: MOVFF REG1, REG2

Before Instruction

REG1 = 33h
 REG2 = 11h

After Instruction

REG1 = 33h
 REG2 = 33h

MOVLB Move Literal to Low Nibble in BSR

Syntax: MOVLW k

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow \text{BSR}$

Status Affected: None

| | | | | |
|-----------|------|------|------|------|
| Encoding: | 0000 | 0001 | kkkk | kkkk |
|-----------|------|------|------|------|

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

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|------------------|--------------|--------------------------|
| Decode | Read literal 'k' | Process Data | Write literal 'k' to BSR |

Example: MOVLB 5

Before Instruction

BSR Register = 02h

After Instruction

BSR Register = 05h

PIC18F45J10 FAMILY

MOVLW Move Literal to W

Syntax: MOVLW k

Operands: $0 \leq k \leq 255$

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 5Ah

After Instruction

W = 5Ah

MOVWF Move W to f

Syntax: MOVWF f{,a}

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

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: MOVWF REG, 0

Before Instruction

W = 4Fh

REG = FFh

After Instruction

W = 4Fh

REG = 4Fh

PIC18F45J10 FAMILY

MULLW Multiply Literal with W

Syntax: MULLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) \times k \rightarrow \text{PRODH:PRODL}$

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 the 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 0C4h

Before Instruction

W = E2h
 PRODH = ?
 PRODL = ?

After Instruction

W = E2h
 PRODH = ADh
 PRODL = 08h

MULWF Multiply W with f

Syntax: MULWF f {,a}

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

Operation: $(W) \times (f) \rightarrow \text{PRODH:PRODL}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|-------------------|--------------|-----------------------------|
| Decode | Read register 'f' | Process Data | Write registers PRODH:PRODL |

Example: MULWF REG, 1

Before Instruction

W = C4h
 REG = B5h
 PRODH = ?
 PRODL = ?

After Instruction

W = C4h
 REG = B5h
 PRODH = 8Ah
 PRODL = 94h

PIC18F45J10 FAMILY

NEGF

Negate f

Syntax: NEGF f{,a}

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

Operation: $(\bar{f}) + 1 \rightarrow f$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example:

NEGF REG, 1

Before Instruction

REG = 0011 1010 [3Ah]

After Instruction

REG = 1100 0110 [C6h]

NOP

No Operation

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 0000 | 0000 | 0000 | 0000 |
| 1111 | xxxx | xxxx | xxxx |

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|--------------|--------------|--------------|
| Decode | No operation | No operation | No operation |

Example:

None.

PIC18F45J10 FAMILY

POP Pop Top of Return Stack

Syntax: 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 | = | 0031A2h |
| Stack (1 level down) | = | 014332h |
| After Instruction | | |
| TOS | = | 014332h |
| PC | = | NEW |

PUSH Push Top of Return Stack

Syntax: 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 implementing a software stack by modifying TOS and then pushing 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 | = | 345Ah |
| PC | = | 0124h |
| After Instruction | | |
| PC | = | 0126h |
| TOS | = | 0126h |
| Stack (1 level down) | = | 345Ah |

PIC18F45J10 FAMILY

RCALL Relative Call

Syntax: RCALL n

Operands: $-1024 \leq n \leq 1023$

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' 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: RESET

Operands: None

Operation: Reset all registers and flags that are affected by a MCLR Reset.

Status Affected: All

Encoding:

| | | | |
|------|------|------|------|
| 0000 | 0000 | 1111 | 1111 |
|------|------|------|------|

Description: This instruction provides a way to execute a 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

PIC18F45J10 FAMILY

RETFIE Return from Interrupt

Syntax: RETFIE {s}

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL,
if s = 1
(WS) → W;
(STATUS) → 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, STATUS 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 Set GIEH or GIEL |
| No operation | No operation | No operation | No operation |

Example: RETFIE 1

After Interrupt

```

PC          = TOS
W           = WS
BSR        = BSRS
STATUS     = STATUS
GIE/GIEH, PEIE/GIEL = 1
    
```

RETLW Return Literal to W

Syntax: 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 = 07h

After Instruction

W = value of kn

PIC18F45J10 FAMILY

RETURN Return from Subroutine

Syntax: 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 Instruction:
PC = TOS

RLCF Rotate Left f through Carry

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

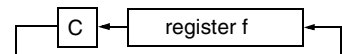
Operation: (f<n>) → dest<n + 1>,
(f<7>) → C,
(C) → dest<0>

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RLCF REG, 0, 0

Before Instruction

REG = 1110 0110
C = 0

After Instruction

REG = 1110 0110
W = 1100 1100
C = 1

PIC18F45J10 FAMILY

RLNCF Rotate Left f (No Carry)

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

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

Operation: $(f\langle n \rangle) \rightarrow \text{dest}\langle n + 1 \rangle$,
 $(f\langle 7 \rangle) \rightarrow \text{dest}\langle 0 \rangle$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RLNCF REG, 1, 0

Before Instruction
REG = 1010 1011
After Instruction
REG = 0101 0111

RRCF Rotate Right f through Carry

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

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

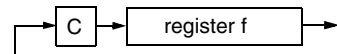
Operation: $(f\langle n \rangle) \rightarrow \text{dest}\langle n - 1 \rangle$,
 $(f\langle 0 \rangle) \rightarrow C$,
 $(C) \rightarrow \text{dest}\langle 7 \rangle$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: RRCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 0111 0011
C = 0

PIC18F45J10 FAMILY

RRNCF Rotate Right f (No Carry)

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

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

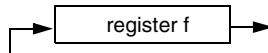
Operation: $(f\langle n \rangle) \rightarrow \text{dest}\langle n - 1 \rangle$,
 $(f\langle 0 \rangle) \rightarrow \text{dest}\langle 7 \rangle$

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). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: RRNCF REG, 1, 0

Before Instruction
 REG = 1101 0111
 After Instruction
 REG = 1110 1011

Example 2: RRNCF REG, 0, 0

Before Instruction
 W = ?
 REG = 1101 0111
 After Instruction
 W = 1110 1011
 REG = 1101 0111

SETF Set f

Syntax: SETF f {,a}

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

Operation: FFh \rightarrow 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: SETF REG, 1

Before Instruction
 REG = 5Ah
 After Instruction
 REG = FFh

PIC18F45J10 FAMILY

SLEEP Enter Sleep mode

Syntax: SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → \overline{TO} ,
0 → 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} = ?
 \overline{PD} = ?

After Instruction
 \overline{TO} = 1 †
 \overline{PD} = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB Subtract f from W with Borrow

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

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

Operation: $(W) - (f) - (\overline{C}) \rightarrow \text{dest}$

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 'f' (default). If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: SUBFWB REG, 1, 0

Before Instruction
REG = 3
W = 2
C = 1

After Instruction
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; 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

PIC18F45J10 FAMILY

SUBLW Subtract W from Literal

Syntax: SUBLW k

Operands: $0 \leq k \leq 255$

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 02h

Before Instruction
W = 01h
C = ?

After Instruction
W = 01h
C = 1 ; result is positive
Z = 0
N = 0

Example 2: SUBLW 02h

Before Instruction
W = 02h
C = ?

After Instruction
W = 00h
C = 1 ; result is zero
Z = 1
N = 0

Example 3: SUBLW 02h

Before Instruction
W = 03h
C = ?

After Instruction
W = FFh ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF Subtract W from f

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

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

Operation: $(f) - (W) \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1
Cycles: 1

Q Cycle Activity:

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

Example 1: SUBWF REG, 1, 0

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, 0, 0

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, 1, 0

Before Instruction
REG = 1
W = 2
C = ?

After Instruction
REG = FFh ; (2's complement)
W = 2
C = 0 ; result is negative
Z = 0
N = 1

PIC18F45J10 FAMILY

SUBWFB Subtract W from f with Borrow

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

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

Operation: $(f) - (W) - (\overline{C}) \rightarrow \text{dest}$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1
Cycles: 1

Q Cycle Activity:

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

Example 1: SUBWFB REG, 1, 0

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

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

Example 2: SUBWFB REG, 0, 0

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

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

Example 3: SUBWFB REG, 1, 0

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

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

SWAPF Swap f

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

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

Operation: $(f<3:0>) \rightarrow \text{dest}<7:4>$,
 $(f<7:4>) \rightarrow \text{dest}<3:0>$

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1
Cycles: 1

Q Cycle Activity:

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

Example: SWAPF REG, 1, 0

Before Instruction
REG = 53h

After Instruction
REG = 35h

PIC18F45J10 FAMILY

TBLRD Table Read

Syntax: 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 nn=0 * =1 *+ =2 *- =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 | No operation (Read Program Memory) | No operation | No operation | No operation (Write TABLAT) |

TBLRD Table Read (Continued)

Example 1: TBLRD *+ ;

Before Instruction

| | | |
|------------------|---|---------|
| TABLAT | = | 55h |
| TBLPTR | = | 00A356h |
| MEMORY (00A356h) | = | 34h |

After Instruction

| | | |
|--------|---|---------|
| TABLAT | = | 34h |
| TBLPTR | = | 00A357h |

Example 2: TBLRD +* ;

Before Instruction

| | | |
|------------------|---|---------|
| TABLAT | = | AAh |
| TBLPTR | = | 01A357h |
| MEMORY (01A357h) | = | 12h |
| MEMORY (01A358h) | = | 34h |

After Instruction

| | | |
|--------|---|---------|
| TABLAT | = | 34h |
| TBLPTR | = | 01A358h |

PIC18F45J10 FAMILY

TBLWT Table Write

Syntax: 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 nn=0 * =1 *+ =2 *- =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-MByte 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

Words: 1

Cycles: 2

Q Cycle Activity:

| | Q1 | Q2 | Q3 | Q4 |
|--------------|----------------------------|--------------|--------------|--|
| Decode | No operation | No operation | No operation | No operation |
| No operation | No operation (Read TABLAT) | No operation | No operation | No operation (Write to Holding Register) |

TBLWT Table Write (Continued)

Example 1: TBLWT *+;

Before Instruction

| | | |
|----------------------------|---|---------|
| TABLAT | = | 55h |
| TBLPTR | = | 00A356h |
| HOLDING REGISTER (00A356h) | = | FFh |

After Instructions (table write completion)

| | | |
|----------------------------|---|---------|
| TABLAT | = | 55h |
| TBLPTR | = | 00A357h |
| HOLDING REGISTER (00A356h) | = | 55h |

Example 2: TBLWT *+;

Before Instruction

| | | |
|----------------------------|---|---------|
| TABLAT | = | 34h |
| TBLPTR | = | 01389Ah |
| HOLDING REGISTER (01389Ah) | = | FFh |
| HOLDING REGISTER (01389Bh) | = | FFh |

After Instruction (table write completion)

| | | |
|----------------------------|---|---------|
| TABLAT | = | 34h |
| TBLPTR | = | 01389Bh |
| HOLDING REGISTER (01389Ah) | = | FFh |
| HOLDING REGISTER (01389Bh) | = | 34h |

PIC18F45J10 FAMILY

TSTFSZ **Test f, Skip if 0**

Syntax: TSTFSZ f {,a}

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

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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
 If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

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, 1
NZERO   :
ZERO    :
```

Before Instruction
 PC = Address (HERE)

After Instruction
 If CNT = 00h,
 PC = Address (ZERO)
 If CNT \neq 00h,
 PC = Address (NZERO)

XORLW **Exclusive OR Literal with W**

Syntax: XORLW k

Operands: $0 \leq k \leq 255$

Operation: (W) .XOR. k \rightarrow W

Status Affected: N, Z

Encoding:

| | | | |
|------|------|------|------|
| 0000 | 1010 | kkkk | kkkk |
|------|------|------|------|

Description: The contents of W are XORed 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 0AFh

Before Instruction
 W = B5h

After Instruction
 W = 1Ah

PIC18F45J10 FAMILY

XORWF Exclusive OR W with f

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

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

Operation: (W) .XOR. (f) \rightarrow 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 is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 21.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: XORWF REG, 1, 0

Before Instruction

REG = AFh

W = B5h

After Instruction

REG = 1Ah

W = B5h

21.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, PIC18F45J10 family devices also provide an optional extension to the core CPU functionality. The added features include eight additional instructions that augment indirect and indexed addressing operations and the implementation of Indexed Literal Offset Addressing mode for many of the standard PIC18 instructions.

The additional features of the extended instruction set are disabled by default. To enable them, users must set the XINST configuration bit.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers, or use them for indexed addressing. Two of the instructions, ADDFSR and SUBFSR, each have an additional special instantiation for using FSR2. These versions (ADDULNK and SUBULNK) allow for automatic return after execution.

The extended instructions are specifically implemented to optimize re-entrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- dynamic allocation and deallocation of software stack space when entering and leaving subroutines
- function pointer invocation
- software stack pointer manipulation
- manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 21-3. Detailed descriptions are provided in **Section 21.2.2 “Extended Instruction Set”**. The opcode field descriptions in Table 21-1 (page 242) apply to both the standard and extended PIC18 instruction sets.

Note: The instruction set extension and the Indexed Literal Offset Addressing mode were designed for optimizing applications written in C; the user may likely never use these instructions directly in assembler. The syntax for these commands is provided as a reference for users who may be reviewing code that has been generated by a compiler.

21.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the File Select Registers and some offset to specify a source or destination register. When an argument for an instruction serves as part of indexed addressing, it is enclosed in square brackets (“[]”). This is done to indicate that the argument is used as an index or offset. MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 21.2.3.1 “Extended Instruction Syntax with Standard PIC18 Commands”**.

Note: In the past, square brackets have been used to denote optional arguments in the PIC18 and earlier instruction sets. In this text and going forward, optional arguments are denoted by braces (“{ }”).

TABLE 21-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

| Mnemonic, Operands | Description | Cycles | 16-Bit Instruction Word | | | | Status Affected |
|---------------------------------------|---|--------|-------------------------|------|------|------|--------------------|
| | | | MSb | | | LSb | |
| ADDFSR f, k | Add literal to FSR | 1 | 1110 | 1000 | ffkk | kkkk | None |
| ADDULNK k | Add literal to FSR2 and return | 2 | 1110 | 1000 | 11kk | kkkk | None |
| CALLW | Call subroutine using WREG | 2 | 0000 | 0000 | 0001 | 0100 | None |
| MOVSF Z _s , f _d | Move z _s (source) to 1st word f _d (destination) 2nd word | 2 | 1110 | 1011 | 0zzz | zzzz | None |
| MOVSS Z _s , Z _d | Move z _s (source) to 1st word Z _d (destination) 2nd word | 2 | 1110 | 1011 | 1zzz | zzzz | None |
| PUSHL k | Store literal at FSR2, decrement FSR2 | 1 | 1110 | 1010 | kkkk | kkkk | None |
| SUBFSR f, k | Subtract literal from FSR | 1 | 1110 | 1001 | ffkk | kkkk | None |
| SUBULNK k | Subtract literal from FSR2 and return | 2 | 1110 | 1001 | 11kk | kkkk | None |

PIC18F45J10 FAMILY

21.2.2 EXTENDED INSTRUCTION SET

ADDFSR **Add Literal to FSR**

Syntax: ADDFSR f, k
 Operands: $0 \leq k \leq 63$
 $f \in [0, 1, 2]$
 Operation: $FSR(f) + k \rightarrow FSR(f)$
 Status Affected: None
 Encoding:

| | | | |
|------|------|------|------|
| 1110 | 1000 | ffkk | kkkk |
|------|------|------|------|

 Description: The 6-bit literal 'k' is added to the contents of the FSR specified by 'f'.
 Words: 1
 Cycles: 1
 Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|---------------------|-----------------|-----------------|
| Decode | Read literal 'k' | Process Data | Write to FSR |

Example: ADDFSR 2, 23h

Before Instruction
 FSR2 = 03FFh
 After Instruction
 FSR2 = 0422h

ADDULNK **Add Literal to FSR2 and Return**

Syntax: ADDULNK k
 Operands: $0 \leq k \leq 63$
 Operation: $FSR2 + k \rightarrow FSR2,$
 $(TOS) \rightarrow PC$
 Status Affected: None
 Encoding:

| | | | |
|------|------|------|------|
| 1110 | 1000 | 11kk | kkkk |
|------|------|------|------|

 Description: The 6-bit literal 'k' is added to the contents of FSR2. A RETURN is then executed by loading the PC with the TOS.

The instruction takes two cycles to execute; a NOP is performed during the second cycle.

This may be thought of as a special case of the ADDFSR instruction, where $f = 3$ (binary '11'); it operates only on FSR2.

Words: 1
 Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|-----------------|---------------------|-----------------|-----------------|
| Decode | Read literal 'k' | Process Data | Write to FSR |
| No Operation | No Operation | No Operation | No Operation |

Example: ADDULNK 23h

Before Instruction
 FSR2 = 03FFh
 PC = 0100h
 After Instruction
 FSR2 = 0422h
 PC = (TOS)

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction syntax then becomes: {label} instruction argument(s).

PIC18F45J10 FAMILY

CALLW Subroutine Call Using WREG

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 0000 | 0000 | 0001 | 0100 |
|------|------|------|------|

Description First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched. Unlike CALL, there is no option to update W, STATUS or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------------|--------------|------------------|--------------|
| Decode | Read WREG | PUSH PC to stack | No operation |
| No operation | No operation | No operation | No operation |

Example: HERE CALLW

Before Instruction

PC = address (HERE)

PCLATH = 10h

PCLATU = 00h

W = 06h

After Instruction

PC = 001006h

TOS = address (HERE + 2)

PCLATH = 10h

PCLATU = 00h

W = 06h

MOVSF Move Indexed to f

Syntax: MOVSF [z_s], f_d

Operands: 0 ≤ z_s ≤ 127
0 ≤ f_d ≤ 4095

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

| | | | |
|------|------|------|-------------------|
| 1110 | 1011 | 0zzz | zzzz _s |
| 1111 | ffff | ffff | ffff _d |

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s' in the first word to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh). The MOVSF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register. If the resultant source address points to an indirect addressing register, the value returned will be 00h.

Words: 2

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|-------------------------------|-----------------------|---------------------------|
| Decode | Determine source addr | Determine source addr | Read source reg |
| Decode | No operation No dummy read | No operation | Write register 'f' (dest) |

Example: MOVSF [05h], REG2

Before Instruction

FSR2 = 80h

Contents of 85h = 33h

REG2 = 11h

After Instruction

FSR2 = 80h

Contents of 85h = 33h

REG2 = 33h

PIC18F45J10 FAMILY

MOVSS Move Indexed to Indexed

Syntax: MOVSS [z_s], [z_d]

Operands: 0 ≤ z_s ≤ 127
0 ≤ z_d ≤ 127

Operation: ((FSR2) + z_s) → ((FSR2) + z_d)

Status Affected: None

Encoding:

| | | | |
|------|------|------|-------------------|
| 1110 | 1011 | 1zzz | zzzz _s |
| 1111 | xxxx | xzzz | zzzz _d |

1st word (source)
2nd word (dest.)

Description

The contents of the source register are moved to the destination register. The addresses of the source and destination registers are determined by adding the 7-bit literal offsets 'z_s' or 'z_d', respectively, to the value of FSR2. Both registers can be located anywhere in the 4096-byte data memory space (000h to FFFh).

The MOVSS instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an indirect addressing register, the value returned will be 00h. If the resultant destination address points to an indirect addressing register, the instruction will execute as a NOP.

Words: 2
Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|-----------------------|-----------------------|-------------------|
| Decode | Determine source addr | Determine source addr | Read source reg |
| Decode | Determine dest addr | Determine dest addr | Write to dest reg |

Example: MOVSS [05h], [06h]

Before Instruction

FSR2 = 80h
 Contents of 85h = 33h
 Contents of 86h = 11h

After Instruction

FSR2 = 80h
 Contents of 85h = 33h
 Contents of 86h = 33h

PUSHL Store Literal at FSR2, Decrement FSR2

Syntax: PUSHL k

Operands: 0 ≤ k ≤ 255

Operation: k → (FSR2),
FSR2 - 1 → FSR2

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 1111 | 1010 | kkkk | kkkk |
|------|------|------|------|

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation. This instruction allows users to push values onto a software stack.

Words: 1
Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|----------|--------------|----------------------|
| Decode | Read 'k' | Process data | Write to destination |

Example: PUSHL 08h

Before Instruction

FSR2H:FSR2L = 01ECh
 Memory (01ECh) = 00h

After Instruction

FSR2H:FSR2L = 01EBh
 Memory (01ECh) = 08h

PIC18F45J10 FAMILY

SUBFSR Subtract Literal from FSR

Syntax: SUBFSR f, k
 Operands: $0 \leq k \leq 63$
 $f \in [0, 1, 2]$
 Operation: $FSR(f) - k \rightarrow FSRf$
 Status Affected: None
 Encoding:

| | | | |
|------|------|------|------|
| 1110 | 1001 | ffkk | kkkk |
|------|------|------|------|

 Description: The 6-bit literal 'k' is subtracted from the contents of the FSR specified by 'f'.
 Words: 1
 Cycles: 1
 Q Cycle Activity:

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

Example: SUBFSR 2, 23h

Before Instruction
 FSR2 = 03FFh
 After Instruction
 FSR2 = 03DCh

SUBULNK Subtract Literal from FSR2 and Return

Syntax: SUBULNK k
 Operands: $0 \leq k \leq 63$
 Operation: $FSR2 - k \rightarrow FSR2$
 (TOS) \rightarrow PC
 Status Affected: None
 Encoding:

| | | | |
|------|------|------|------|
| 1110 | 1001 | 11kk | kkkk |
|------|------|------|------|

 Description: The 6-bit literal 'k' is subtracted from the contents of the FSR2. A RETURN is then executed by loading the PC with the TOS. The instruction takes two cycles to execute; a NOP is performed during the second cycle.
 This may be thought of as a special case of the SUBFSR instruction, where $f = 3$ (binary '11'); it operates only on FSR2.
 Words: 1
 Cycles: 2
 Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------------|-------------------|--------------|----------------------|
| Decode | Read register 'f' | Process Data | Write to destination |
| No Operation | No Operation | No Operation | No Operation |

Example: SUBULNK 23h

Before Instruction
 FSR2 = 03FFh
 PC = 0100h
 After Instruction
 FSR2 = 03DCh
 PC = (TOS)

PIC18F45J10 FAMILY

21.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note: Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing mode (**Section 5.5.1 “Indexed Addressing with Literal Offset”**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank ('a' = 0), or in a GPR bank designated by the BSR ('a' = 1). When the extended instruction set is enabled and 'a' = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the stack pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 21.2.3.1 “Extended Instruction Syntax with Standard PIC18 Commands”**).

Although the Indexed Literal Offset Addressing mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset Addressing mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

21.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument, 'f', in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value, 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets (“[]”). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within brackets, will generate an error in the MPASM Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled) when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument, 'd', functions as before.

Refer to the MPLAB® IDE, MPASM™ or MPLAB C18 documentation for information on enabling Extended Instruction set support

21.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18F45J10 family, it is very important to consider the type of code. A large, re-entrant application that is written in 'C' and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

PIC18F45J10 FAMILY

ADDWF **ADD W to Indexed (Indexed Literal Offset mode)**

Syntax: ADDWF [k] {,d}

Operands: $0 \leq k \leq 95$
 $d \in [0,1]$

Operation: $(W) + ((FSR2) + k) \rightarrow dest$

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

Encoding:

| | | | |
|------|------|------|------|
| 0010 | 01d0 | kkkk | kkkk |
|------|------|------|------|

Description: The contents of W are added to the contents of the register indicated by FSR2, offset by the value 'k'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ADDWF [OFST] , 0

Before Instruction

W = 17h
 OFST = 2Ch
 FSR2 = 0A00h
 Contents of 0A2Ch = 20h

After Instruction

W = 37h
 Contents of 0A2Ch = 20h

BSF **Bit Set Indexed (Indexed Literal Offset mode)**

Syntax: BSF [k], b

Operands: $0 \leq f \leq 95$
 $0 \leq b \leq 7$

Operation: $1 \rightarrow ((FSR2) + k) < b >$

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 1000 | bbb0 | kkkk | kkkk |
|------|------|------|------|

Description: Bit 'b' of the register indicated by FSR2, offset by the value 'k', is set.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: BSF [FLAG_OFST] , 7

Before Instruction

FLAG_OFST = 0Ah
 FSR2 = 0A00h
 Contents of 0A0Ah = 55h

After Instruction

Contents of 0A0Ah = D5h

SETF **Set Indexed (Indexed Literal Offset mode)**

Syntax: SETF [k]

Operands: $0 \leq k \leq 95$

Operation: $FFh \rightarrow ((FSR2) + k)$

Status Affected: None

Encoding:

| | | | |
|------|------|------|------|
| 0110 | 1000 | kkkk | kkkk |
|------|------|------|------|

Description: The contents of the register indicated by FSR2, offset by 'k', are set to FFh.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|--------|----------|--------------|----------------|
| Decode | Read 'k' | Process Data | Write register |

Example: SETF [OFST]

Before Instruction

OFST = 2Ch
 FSR2 = 0A00h
 Contents of 0A2Ch = 00h

After Instruction

Contents of 0A2Ch = FFh

PIC18F45J10 FAMILY

21.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set of the PIC18F45J10 family of devices. This includes the MPLAB C18 C compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default configuration bits for that device. The default setting for the XINST configuration bit is '0', disabling the extended instruction set and Indexed Literal Offset Addressing mode. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option, or dialog box within the environment, that allows the user to configure the language tool and its settings for the project
- A command line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

22.0 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
 - MPLAB PM3 Device 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[®] Evaluation and Programming Tools
 - PICDEM MSC
 - microID[®] Developer Kits
 - CAN
 - PowerSmart[®] Developer Kits
 - Analog

22.1 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)
 - 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.

22.2 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

PIC18F45J10 FAMILY

22.3 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.

22.4 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/librarian 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

22.5 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.

22.6 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 its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

22.7 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.

22.8 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.

22.9 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.

22.10 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 ICE 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.

22.11 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.

22.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX 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.

22.13 MPLAB PM3 Device Programmer

The MPLAB PM3 is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 device programmer can read, verify and program PICmicro devices without a PC connection. It can also set code protection in this mode. MPLAB PM3 connects to the host PC via an RS-232 or USB cable. MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an SD/MMC card for file storage and secure data applications.

PIC18F45J10 FAMILY

22.14 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.

22.15 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.

22.16 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

22.17 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.

22.18 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.

22.19 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, 2 x 16 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.

22.20 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.

22.21 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.

22.22 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.

22.23 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.

22.24 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.

22.25 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 rLab™ 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 Selector Guide for the complete list of demonstration and evaluation kits.

PIC18F45J10 FAMILY

NOTES:

23.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings^(†)

| | |
|---|-----------------------|
| Ambient temperature under bias | -40°C to +125°C |
| Storage temperature | -65°C to +150°C |
| Voltage on any digital-only I/O pin with respect to VSS (except VDD and $\overline{\text{MCLR}}$) | -0.3V to 5.5V |
| Voltage on any combined digital and analog pin with respect to VSS (except VDD and $\overline{\text{MCLR}}$) | -0.3V to (VDD + 0.3V) |
| Voltage on VDDCORE with respect to VSS | -0.3V to 2.75V |
| Voltage on VDD with respect to VSS | -0.3V to 3.6V |
| Voltage on $\overline{\text{MCLR}}$ with respect to VSS | 0V to 5.5V |
| Total power dissipation (Note 1) | 1.0W |
| Maximum current out of VSS pin | 300 mA |
| Maximum current into VDD pin | 250 mA |
| Input clamp current, I _{IK} (V _I < 0 or V _I > VDD) | TBD |
| Output clamp current, I _{OK} (V _O < 0 or V _O > VDD) | TBD |
| Maximum output current sunk by PORTB and PORTC I/O pins | 25 mA |
| Maximum output current sunk by PORTA, PORTD, and PORTE I/O pins | 2 mA |
| Maximum output current sourced by PORTB and PORTC I/O pins | 25 mA |
| Maximum output current sourced by PORTA, PORTD, and PORTE I/O pins | 2 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:

$$P_{dis} = VDD \times \{I_{DD} - \sum I_{OH}\} + \sum \{(VDD - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$$

† **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.

PIC18F45J10 FAMILY

FIGURE 23-1: PIC18LF45J10 FAMILY VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)

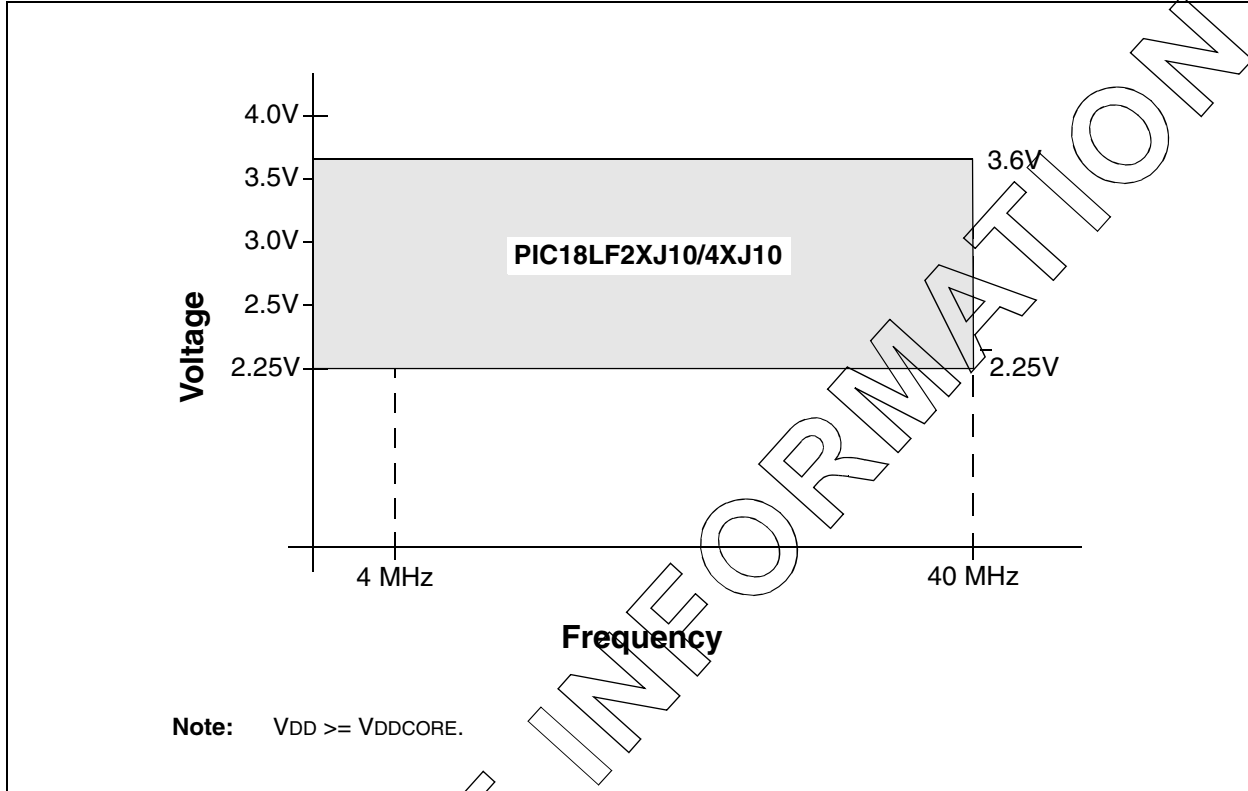
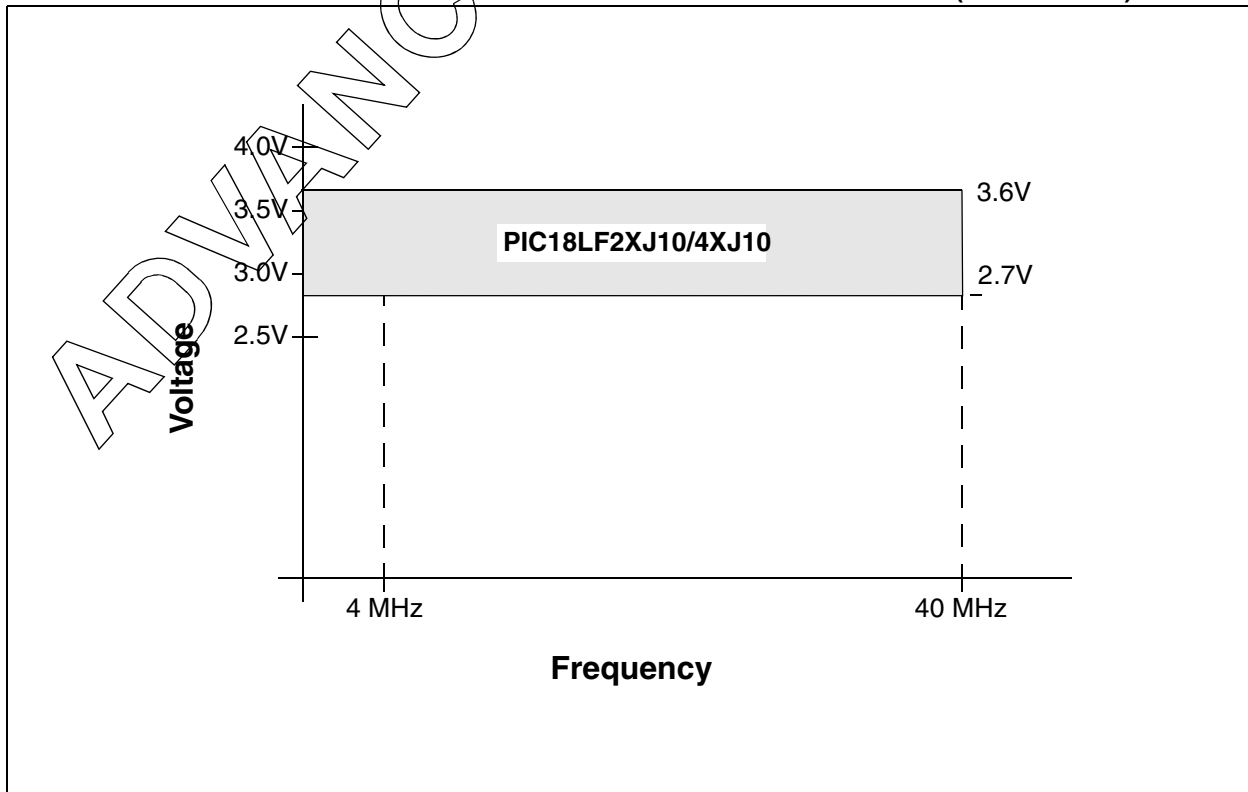


FIGURE 23-2: PIC18LF45J10 FAMILY VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



PIC18F45J10 FAMILY

23.1 DC Characteristics: Supply Voltage PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial)

| PIC18F45J10 Family (Industrial) | | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | |
|------------------------------------|---------|--|---|-----|------|-------|--|
| Param No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
| D001 | VDD | Supply Voltage | VDDCORE ⁽²⁾ | — | 3.6 | V | |
| D001B | VDDCORE | External Supply for Microcontroller Core | 2.0 | — | 2.75 | V | Valid only in parts designated "LF". See Section 20.3 "On-Chip Voltage Regulator" for details. |
| D002 | VDR | RAM Data Retention Voltage⁽¹⁾ | 1.5 | — | — | V | |
| D003 | VPOR | VDD Start Voltage to ensure internal Power-on Reset signal | — | — | TBD | V | See Section 4.3 "Power-on Reset (POR)" for details |
| D004 | SVDD | VDD Rise Rate to ensure internal Power-on Reset signal | 0.05 | — | — | V/ms | See Section 4.3 "Power-on Reset (POR)" for details |
| D005A | VBOR | Brown-out Reset Voltage | — | — | 2.6 | V | On-chip voltage regulator enabled. See Section 4.3 "Power-on Reset (POR)" for details. |

Legend: TBD = To Be Determined

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: In parts designated with an "F", VDD >= 2.7V.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | |
|---|-------------|---|-----|---------------|-----------------------|
| Param No. | Device | Typ | Max | Units | Conditions |
| Power-Down Current (I_{PD})⁽¹⁾ | | | | | |
| | All devices | TBD | TBD | μA | -40°C |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ |
| | All devices | TBD | TBD | μA | -40°C |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ |

Legend: TBD = To Be Determined

- 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 $I_r = V_{DD}/2R_{EXT}$ (mA) with REXT in k Ω .
- 4:** Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to $+70^{\circ}\text{C}$. Extended temperature crystals are available at a much higher cost.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | | | |
|---|-------------|---|---------------|-----------------------|------------------------|------------------------|---|--|
| Param No. | Device | Typ | Max | Units | Conditions | | | |
| Supply Current (I_{DD})^(2,3) | | | | | | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | FOSC = 31 kHz (RC_RUN mode, Internal oscillator source) | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 3.3\text{V}$ | | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | | FOSC = 31 kHz (RC_IDLE mode, Internal oscillator source) |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| All devices | TBD | TBD | μA | -40°C | $V_{DD} = 3.3\text{V}$ | | | |
| | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | | |
| | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | | |

Legend: TBD = To Be Determined

- 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 I_{DD} 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 $I_r = V_{DD}/2R_{EXT}$ (mA) with REXT in k Ω .
- 4:** Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to $+70^{\circ}\text{C}$. Extended temperature crystals are available at a much higher cost.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial | | | | | | |
|---|-------------|--|-----|-------|------------|---|--|---|
| Param No. | Device | Typ | Max | Units | Conditions | | | |
| Supply Current (IDD)^(2,3) | | | | | | | | |
| | All devices | TBD | TBD | μA | -40°C | VDD = 2.5V Fosc = 1 MHz (PRI_RUN mode, EC oscillator) | | |
| | | TBD | TBD | μA | +25°C | | | |
| | | TBD | TBD | μA | +85°C | | | |
| | All devices | TBD | TBD | mA | -40°C | | VDD = 3.3V Fosc = 4 MHz (PRI_RUN mode, EC oscillator) | |
| | | TBD | TBD | mA | +25°C | | | |
| | | TBD | TBD | mA | +85°C | | | |
| | All devices | TBD | TBD | mA | -40°C | | | VDD = 2.5V Fosc = 40 MHz (PRI_RUN mode, EC oscillator) |
| | | TBD | TBD | mA | +25°C | | | |
| | | TBD | TBD | mA | +85°C | | | |
| | All devices | TBD | TBD | mA | -40°C | VDD = 3.3V Fosc = 40 MHz (PRI_RUN mode, EC oscillator) | | |
| | | TBD | TBD | mA | +25°C | | | |
| | | TBD | TBD | mA | +85°C | | | |

Legend: TBD = To Be Determined

- 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, TimerN 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:
QSC1 = 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 $I_R = V_{DD}/2R_{EXT}$ (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.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | |
|--|-------------|---|-----|-------|------------|---|
| Param No. | Device | Typ | Max | Units | Conditions | |
| Supply Current (I_{DD})⁽²⁾ | | | | | | |
| | All devices | TBD | TBD | mA | -40°C | V _{DD} = 2.5V F _{OSC} = 4 MHz, (PRI_RUN HS mode) |
| | | TBD | TBD | mA | +25°C | |
| | | TBD | TBD | mA | +85°C | |
| | All devices | TBD | TBD | mA | -40°C | V _{DD} = 3.3V F _{OSC} = 4 MHz, (PRI_RUN HS mode) |
| | | TBD | TBD | mA | +25°C | |
| | | TBD | TBD | mA | +85°C | |
| | All devices | TBD | TBD | mA | -40°C | V _{DD} = 2.5V F _{OSC} = 10 MHz, (PRI_RUN HS mode) |
| | | TBD | TBD | mA | +25°C | |
| | | TBD | TBD | mA | +85°C | |
| | All devices | TBD | TBD | mA | -40°C | V _{DD} = 3.3V F _{OSC} = 10 MHz, (PRI_RUN HS mode) |
| | | TBD | TBD | mA | +25°C | |
| | | TBD | TBD | mA | +85°C | |

Legend: TBD = To Be Determined

- 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 V_{DD} or V_{SS} 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 I_{DD} measurements in active operation mode are:
 OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD};
 MCLR = V_{DD}; WDT enabled/disabled as specified.
- 3:** For RC oscillator configurations, current through R_{EXT} is not included. The current through the resistor can be estimated by the formula $I_r = V_{DD}/2R_{EXT}$ (mA) with R_{EXT} 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.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | | | |
|---|--------|---|-----|---------------|-----------------------|---|---|--|
| Param No. | Device | Typ | Max | Units | Conditions | | | |
| Supply Current (I_{DD})^(2,3) | | | | | | | | |
| All devices | | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ $F_{OSC} = 1\text{ MHz}$ (PRI_IDLE mode, EC oscillator) | | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| All devices | | TBD | TBD | μA | -40°C | | $V_{DD} = 3.3\text{V}$ $F_{OSC} = 4\text{ MHz}$ (PRI_IDLE mode, EC oscillator) | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| All devices | | TBD | TBD | μA | -40°C | | | $V_{DD} = 2.5\text{V}$ $F_{OSC} = 40\text{ MHz}$ (PRI_IDLE mode, EC oscillator) |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| All devices | | TBD | TBD | mA | -40°C | $V_{DD} = 3.3\text{V}$ | | |
| | | TBD | TBD | mA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | mA | $+85^{\circ}\text{C}$ | | | |
| All devices | | TBD | TBD | mA | -40°C | | $V_{DD} = 2.5\text{V}$ | |
| | | TBD | TBD | mA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | mA | $+85^{\circ}\text{C}$ | | | |
| All devices | | TBD | TBD | mA | -40°C | | | $V_{DD} = 3.3\text{V}$ |
| | | TBD | TBD | mA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | mA | $+85^{\circ}\text{C}$ | | | |

Legend: TBD = To Be Determined

- 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 I_{DD} measurements in active operation mode are:
 $OSC1 =$ external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
 $MCLR = V_{DD}$; 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 $I_r = V_{DD}/2R_{EXT}$ (mA) with REXT in k Ω .
- 4:** Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to $+70^{\circ}\text{C}$. Extended temperature crystals are available at a much higher cost.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | | | |
|---|-------------|---|---------------|-----------------------|------------------------|------------------------|---|--|
| Param No. | Device | Typ | Max | Units | Conditions | | | |
| Supply Current (I_{DD})^(2,3) | | | | | | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | $F_{OSC} = 32\text{ kHz}^{(4)}$ (SEC_RUN mode, Timer1 as clock) | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 3.3\text{V}$ | | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| | All devices | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | | $F_{OSC} = 32\text{ kHz}^{(4)}$ (SEC_IDLE mode, Timer1 as clock) |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | |
| All devices | TBD | TBD | μA | -40°C | $V_{DD} = 3.3\text{V}$ | | | |
| | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | | |
| | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | | |

Legend: TBD = To Be Determined

- 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 V_{DD} or V_{SS} 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 I_{DD} measurements in active operation mode are:
 $OSC1$ = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD} ;
 $MCLR$ = V_{DD} ; 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 $I_r = V_{DD}/2REXT$ (mA) with $REXT$ in $k\Omega$.
- 4:** Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to $+70^{\circ}\text{C}$. Extended temperature crystals are available at a much higher cost.

PIC18F45J10 FAMILY

23.2 DC Characteristics: Power-Down and Supply Current PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| PIC18F45J10 Family (Industrial) | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | | | | |
|------------------------------------|----------------|---|-------------------|---------------|-----------------------|--|------------------------|------------------------|---------------------------------|
| Param No. | Device | Typ | Max | Units | Conditions | | | | |
| D022 (ΔI_{WDT}) | Watchdog Timer | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | | | |
| | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | | | |
| | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | | | |
| | | D025 (ΔI_{OSCB}) | Timer1 Oscillator | TBD | TBD | μA | -40°C | $V_{DD} = 2.5\text{V}$ | 32 kHz on Timer1 ⁽³⁾ |
| | | | | TBD | TBD | μA | $+25^{\circ}\text{C}$ | | |
| | | | | TBD | TBD | μA | $+85^{\circ}\text{C}$ | | |
| D026 (ΔI_{AD}) | A/D Converter | TBD | | TBD | μA | -40°C to $+85^{\circ}\text{C}$ | $V_{DD} = 2.5\text{V}$ | A/D on, not converting | |
| | | TBD | | TBD | μA | -40°C to $+85^{\circ}\text{C}$ | | | $V_{DD} = 3.3\text{V}$ |
| | | | | | | | | | |

Legend: TBD = To Be Determined

- 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 V_{DD} or V_{SS} 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 I_{DD} measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD} ;
MCLR = V_{DD} ; 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 $I_r = V_{DD}/2R_{EXT}$ (mA) with REXT in k Ω .
- 4:** Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to $+70^{\circ}\text{C}$. Extended temperature crystals are available at a much higher cost.

PIC18F45J10 FAMILY

23.3 DC Characteristics: PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | |
|--|--------------------------------------|--|---|---|---------------------------------|---|
| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
| D030 D030A D031 D032 D033 D033A D034 | V _{IL} | Input Low Voltage I/O ports: with TTL buffer with Schmitt Trigger buffer MCLR OSC1 OSC1 T1CKI | V _{SS} — V _{SS} V _{SS} V _{SS} V _{SS} V _{SS} | 0.15 V _{DD} 0.8 0.2 V _{DD} 0.2 V _{DD} 0.3 V _{DD} 0.2 V _{DD} 0.3 | V V V V V V V | V _{DD} < 3.3V 3.3V ≤ V _{DD} ≤ 3.6V HS, HSPLL modes EC mode ⁽¹⁾ |
| D040 D040A D041 D042 D043 D043A D044 | V _{IH} | Input High Voltage I/O ports: with TTL buffer with Schmitt Trigger buffer MCLR OSC1 OSC1 T1CKI | 0.25 V _{DD} + 0.8V 2.0 0.8 V _{DD} 0.8 V _{DD} 0.7 V _{DD} 0.8 V _{DD} TBD | V _{DD} V _{DD} V _{DD} V _{DD} V _{DD} V _{DD} V _{DD} | V V V V V V V | V _{DD} < 3.3V 3.3V ≤ V _{DD} ≤ 3.6V HS, HSPLL modes EC mode |
| D060 D061 D063 | I _{IL} | Input Leakage Current ^(2,3) I/O ports MCLR OSC1 | — — — | TBD TBD TBD | μA μA μA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , Pin at high-impedance V _{SS} ≤ V _{PIN} ≤ V _{DD} V _{SS} ≤ V _{PIN} ≤ V _{DD} |
| D070 | I _{PU} I _{PURB} | Weak Pull-up Current PORTB weak pull-up current | 50 | 400 | μA | V _{DD} = 3.3V, V _{PIN} = V _{SS} |

Legend: TBD = To Be Determined

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 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.

PIC18F45J10 FAMILY

23.3 DC Characteristics: PIC18F24J10/25J10/44J10/45J10 (Industrial) PIC18LF24J10/25J10/44J10/45J10 (Industrial) (Continued)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | |
|--|--------|---|---|-----|-------|--|
| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
| D080A | VOL | Output Low Voltage I/O ports | — | TBD | V | TBD, $V_{DD} = 3.3\text{V}$, -40°C to $+85^{\circ}\text{C}$ |
| D080B | | | — | — | | |
| D080C | | | — | — | | |
| D083 | | OSC2/CLKO (EC mode) | — | 0.6 | V | $I_{OL} = 1.6\text{ mA}$, $V_{DD} = 3.3\text{V}$, -40°C to $+85^{\circ}\text{C}$ |
| D090A | VOH | Output High Voltage⁽³⁾ I/O ports | TBD | — | V | TBD, $V_{DD} = 3.3\text{V}$, -40°C to $+85^{\circ}\text{C}$ |
| D090B | | | — | — | | |
| D090C | | | — | — | | |
| D092 | | OSC2/CLKO (EC mode) | $V_{DD} - 0.7$ | — | V | $I_{OH} = -1.3\text{ mA}$, $V_{DD} = 3.3\text{V}$, -40°C to $+85^{\circ}\text{C}$ |
| Capacitive Loading Specs on Output Pins | | | | | | |
| D100 ⁽⁴⁾ | COSC2 | OSC2 pin | — | 15 | pF | In HS mode when external clock is used to drive OSC1 |
| D101 | CIO | All I/O pins and OSC2 | — | 50 | pF | To meet the AC Timing Specifications |
| D102 | CB | SCLx, SDAx | — | 400 | pF | I ² C™ Specification |

Legend: TBD = To Be Determined

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 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.

PIC18F45J10 FAMILY

TABLE 23-1: MEMORY PROGRAMMING REQUIREMENTS

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial | | | | |
|-----------------------------|--------|-----------------------------------|---|------|-----|-------|--|
| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
| Program Flash Memory | | | | | | | |
| D130 | EP | Cell Endurance | 100 | 1K | — | E/W | -40°C to $+85^{\circ}\text{C}$ |
| D131 | VPR | VDD for Read | V _{MIN} | — | 3.6 | V | V _{MIN} = Minimum operating voltage |
| D132B | VPEW | VDD for Self-Timed Write | V _{MIN} | — | 3.6 | V | V _{MIN} = Minimum operating voltage |
| D133A | TIW | Self-Timed Write Cycle Time | — | 10 | — | ms | |
| D134 | TRET D | Characteristic Retention | 10 | 20 | — | Year | Provided no other specifications are violated |
| D135 | IDDP | Supply Current during Programming | — | 10 | — | mA | |

† Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

ADVANCE INFORMATION

PIC18F45J10 FAMILY

TABLE 23-2: COMPARATOR SPECIFICATIONS

| Operating Conditions: $3.0V < V_{DD} < 3.6V$, $-40^{\circ}C < T_A < +85^{\circ}C$ (unless otherwise stated) | | | | | | | |
|--|--------------------|---|-----|------|----------------|-------|----------|
| Param No. | Sym | Characteristics | Min | Typ | Max | Units | Comments |
| D300 | V _{IOFF} | Input Offset Voltage | — | ±5.0 | ±10 | mV | |
| D301 | V _{ICM} | Input Common Mode Voltage* | 0 | — | $V_{DD} - 1.5$ | V | |
| D302 | CMRR | Common Mode Rejection Ratio* | 55 | — | — | dB | |
| 300 | T _{RESP} | Response Time ^{(1)*} | — | 150 | 400 | ns | |
| 301 | T _{MC2OV} | 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 $(V_{DD} - 1.5)/2$, while the other input transitions from V_{SS} to V_{DD}.

TABLE 23-3: VOLTAGE REFERENCE SPECIFICATIONS

| Operating Conditions: $3.0V < V_{DD} < 3.6V$, $-40^{\circ}C < T_A < +85^{\circ}C$ (unless otherwise stated) | | | | | | | |
|--|------------------|------------------------------|-------------|-----|-------------|-------|----------|
| Param No. | Sym | Characteristics | Min | Typ | Max | Units | Comments |
| D310 | V _{RES} | Resolution | $V_{DD}/24$ | — | $V_{DD}/32$ | LSb | |
| D311 | V _{RAA} | Absolute Accuracy | — | — | 1/2 | LSb | |
| D312 | V _{RUR} | Unit Resistor Value (R) | — | 2k | — | Ω | |
| 310 | T _{SET} | Settling Time ⁽¹⁾ | — | — | 10 | µs | |

Note 1: Settling time measured while CVRR = 1 and CVR3:CVR0 transitions from '0000' to '1111'.

TABLE 23-4: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

| Operating Conditions: $-40^{\circ}C < T_A < +85^{\circ}C$ (unless otherwise stated) | | | | | | | |
|---|--------------------|---------------------------------|-----|-----|-----|-------|---------------------------|
| Param No. | Sym | Characteristics | Min | Typ | Max | Units | Comments |
| | V _{RGOUT} | Regulator Output Voltage | — | 2.5 | — | V | |
| | CEFC | External Filter Capacitor Value | 1 | 10 | — | µF | Capacitor must be low ESR |

* These parameters are characterized but not tested. Parameter numbers not yet assigned for these specifications.

PIC18F45J10 FAMILY

23.4 AC (Timing) Characteristics

23.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

- | | | |
|-------------|-----------|--|
| 1. TppS2ppS | 3. Tcc:ST | (I ² C specifications only) |
| 2. TppS | 4. Ts | (I ² C specifications only) |

| | | | |
|---|-----------|---|------|
| T | | 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 ² C only | | | |
| AA | output access | High | High |
| BUF | Bus free | Low | Low |

Tcc:ST (I²C specifications only)

| | | | |
|-----|-----------------|-----|----------------|
| CC | | | |
| HD | Hold | SU | Setup |
| ST | | | |
| DAT | DATA input hold | STO | Stop condition |
| STA | Start condition | | |

PIC18F45J10 FAMILY

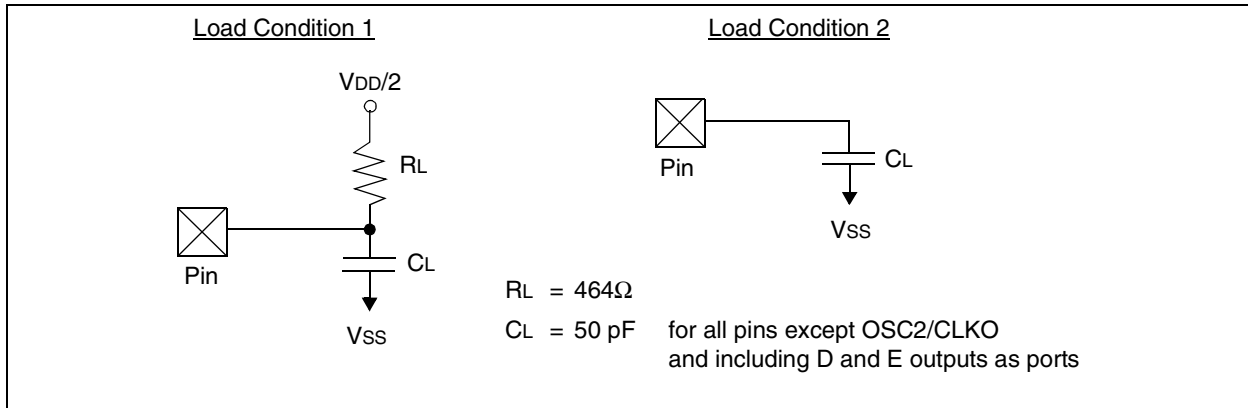
23.4.2 TIMING CONDITIONS

The temperature and voltages specified in Table 23-5 apply to all timing specifications unless otherwise noted. Figure 23-3 specifies the load conditions for the timing specifications.

TABLE 23-5: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

| | |
|---------------------------|--|
| AC CHARACTERISTICS | Standard Operating Conditions (unless otherwise stated) |
| | Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial Operating voltage V_{DD} range as described in DC spec Section 23.1 and Section 23.3 . |

FIGURE 23-3: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



PIC18F45J10 FAMILY

23.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 23-4: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)

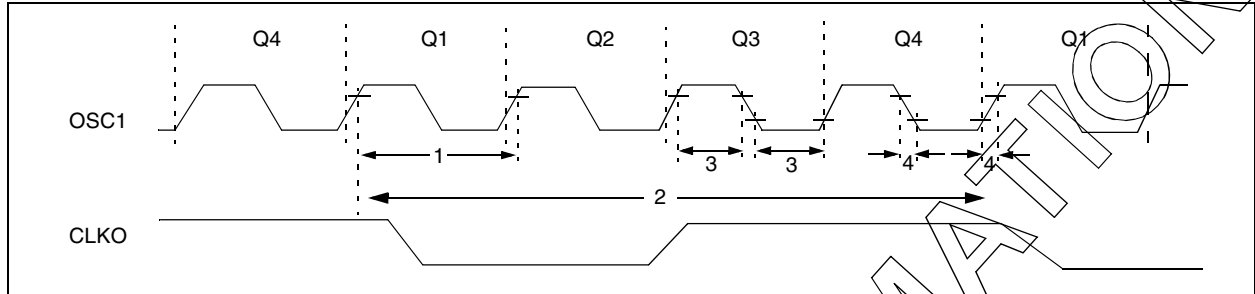


TABLE 23-6: EXTERNAL CLOCK TIMING REQUIREMENTS

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|------------|---------------|---|-----|-----|-------|--------------------------|
| 1A | FOSC | External CLKI Frequency ⁽¹⁾ | DC | 40 | MHz | HS Oscillator mode |
| | | Oscillator Frequency ⁽¹⁾ | DC | 40 | MHz | HS Oscillator mode |
| 1 | TOSC | External CLKI Period ⁽¹⁾ | 25 | — | ns | HS Oscillator mode |
| | | Oscillator Period ⁽¹⁾ | 25 | 250 | ns | HS Oscillator mode |
| 2 | TCY | Instruction Cycle Time ⁽¹⁾ | 100 | — | ns | TCY = 4/FOSC, Industrial |
| 3 | TosL, TosH | External Clock in (OSC1) High or Low Time | 10 | — | ns | HS Oscillator mode |
| 4 | TosR, TosF | External Clock in (OSC1) Rise or Fall Time | — | 7.5 | ns | HS Oscillator mode |

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations. 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.

PIC18F45J10 FAMILY

TABLE 23-7: PLL CLOCK TIMING SPECIFICATIONS ($V_{DD} = 2.5V$ TO $3.6V$)

| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|-----------|--------------|------------------------------|-----|------|-----|-------|--------------|
| F10 | FOSC | Oscillator Frequency Range | 4 | — | 10 | MHz | HS mode only |
| F11 | FSYS | On-Chip VCO System Frequency | 16 | — | 40 | MHz | HS mode only |
| F13 | Δ 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 23-8: AC CHARACTERISTICS: INTERNAL RC ACCURACY
PIC18F24J10/25J10/44J10/45J10 (INDUSTRIAL)**

| Param No. | Characteristic | Min | Typ | Max | Units | Conditions |
|-----------|---|--------|-----|--------|-------|-------------------------------------|
| | INTRC Accuracy @ Freq = 31 kHz ⁽¹⁾ | 26.562 | — | 35.938 | kHz | -40°C to +85°C, $V_{DD} = 2.0-3.3V$ |

Note 1: INTRC frequency after calibration. Change of INTRC frequency as V_{DD} changes.

PIC18F45J10 FAMILY

FIGURE 23-5: CLKO AND I/O TIMING

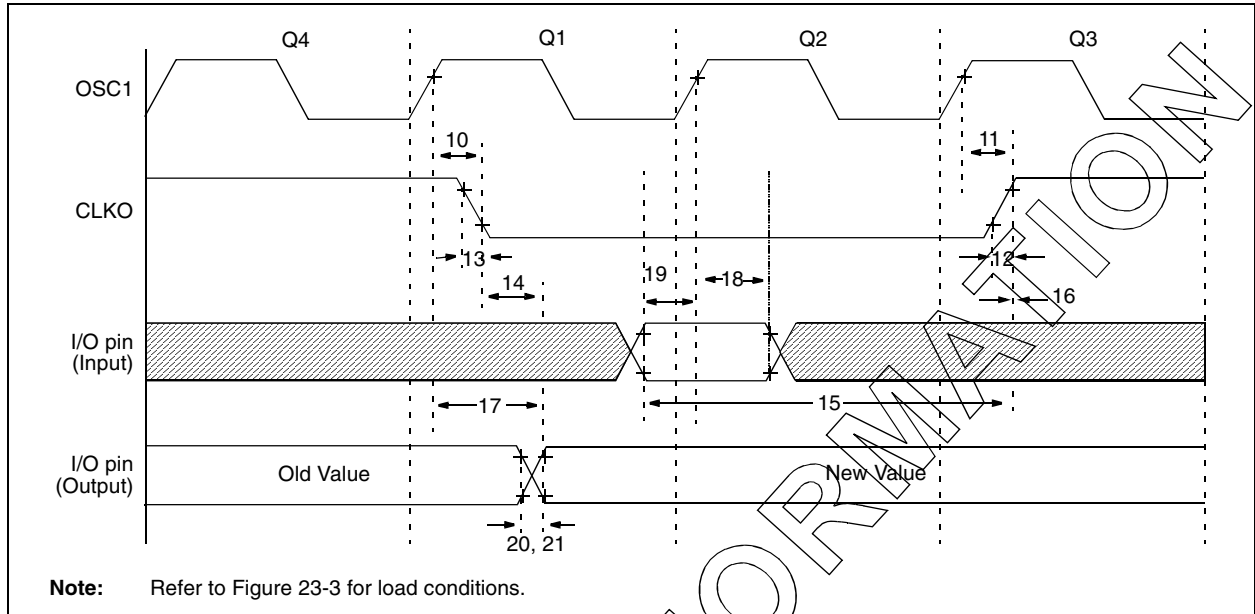


TABLE 23-9: CLKO AND I/O TIMING REQUIREMENTS

| Param No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
|-----------|----------|--|---------------------------|-----|--------------------------|-------|------------|
| 10 | TosH2ckL | OSC1 ↑ to CLKO ↓ | — | TBD | TBD | ns | |
| 11 | TosH2ckH | OSC1 ↑ to CLKO ↑ | — | TBD | TBD | ns | |
| 12 | TckR | CLKO Rise Time | — | TBD | TBD | ns | |
| 13 | TckF | CLKO Fall Time | — | TBD | TBD | ns | |
| 14 | TckL2ioV | CLKO ↓ to Port Out Valid | — | — | 0.5 T _{CY} + 20 | ns | |
| 15 | TioV2ckH | Port In Valid before CLKO ↑ | 0.25 T _{CY} + 25 | — | — | ns | |
| 16 | TckH2ioI | Port In Hold after CLKO ↑ | 0 | — | — | ns | |
| 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) | 100 | — | — | ns | |
| 18A | | | 200 | — | — | ns | |
| 19 | TioV2osH | Port Input Valid to OSC1 ↑ (I/O in setup time) | 0 | — | — | ns | |
| 20 | TioR | Port Output Rise Time | — | TBD | TBD | ns | |
| 20A | | | — | — | TBD | ns | |
| 21 | TioF | Port Output Fall Time | — | TBD | TBD | ns | |
| 21A | | | — | — | TBD | ns | |
| 22† | TINP | INT pin High or Low Time | T _{CY} | — | — | ns | |
| 23† | TRBP | RB7:RB4 Change INT High or Low Time | T _{CY} | — | — | ns | |

Legend: TBD = To Be Determined

† These parameters are asynchronous events not related to any internal clock edges.

PIC18F45J10 FAMILY

FIGURE 23-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

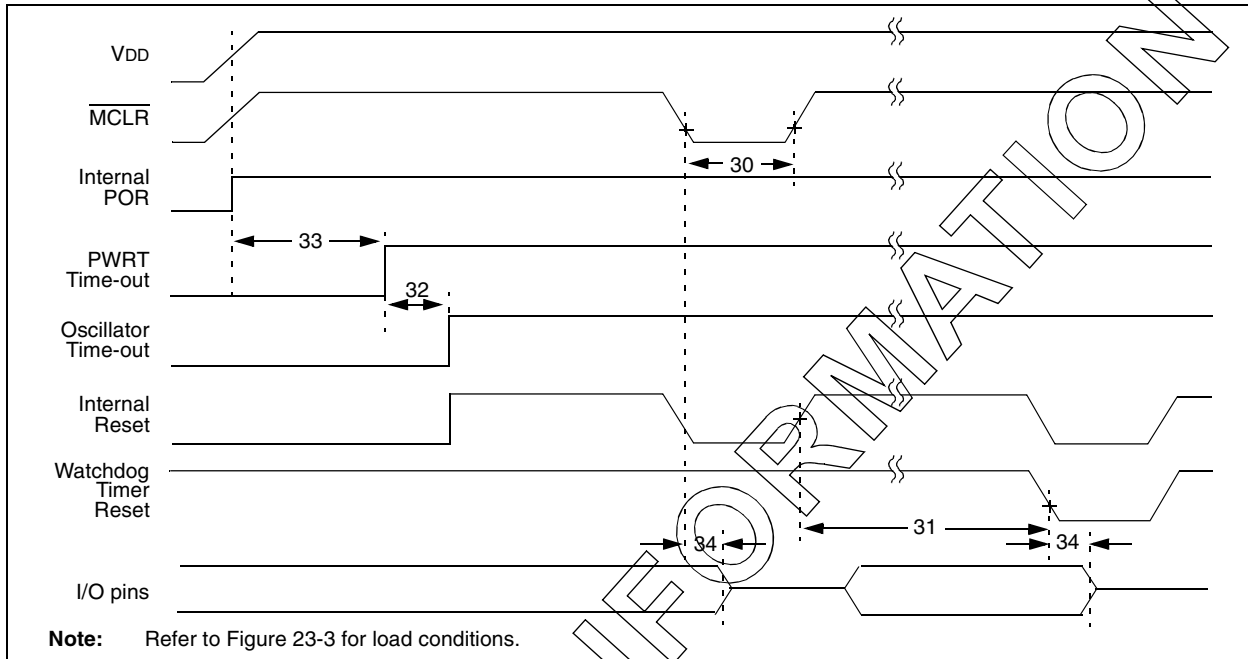


FIGURE 23-7: BROWN-OUT RESET TIMING

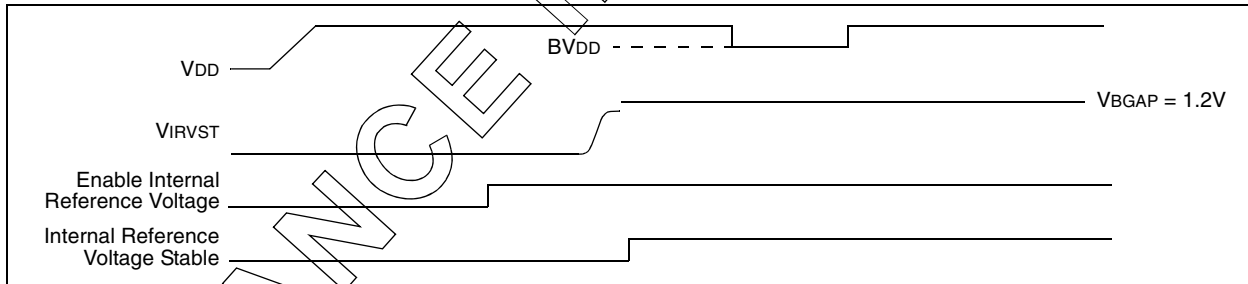


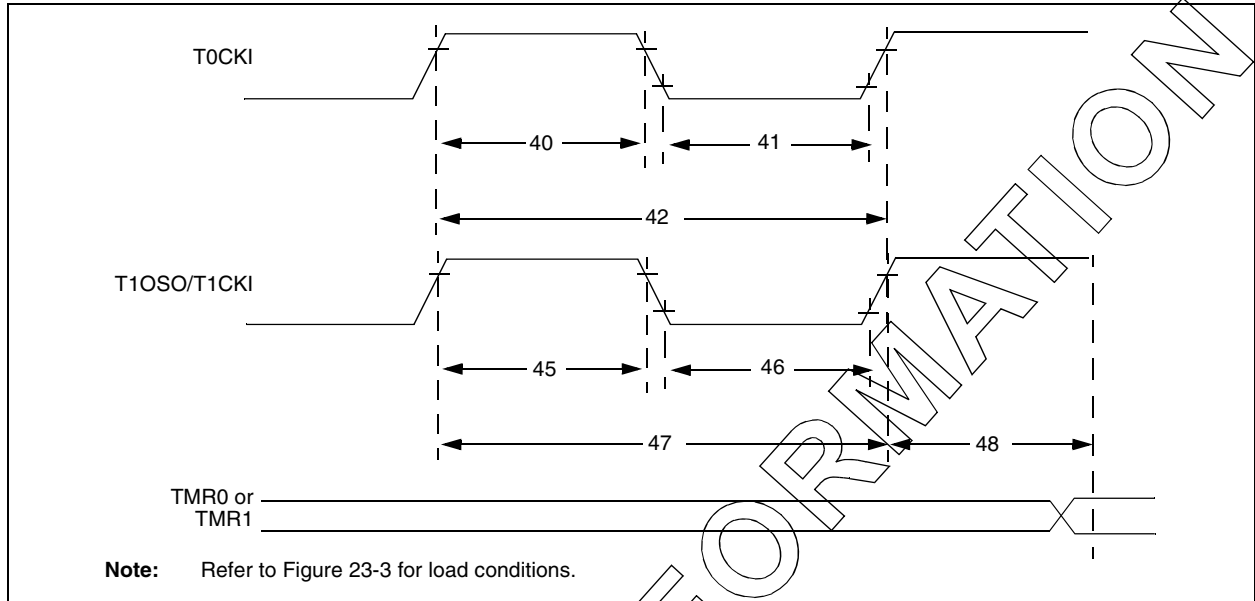
TABLE 23-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

| Param. No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
|------------|--------|--|-----------------------|------|-----------------------|-------|--------------------------------|
| 30 | TMCL | MCLR Pulse Width (low) | 2 | — | — | μs | |
| 31 | TWDT | Watchdog Timer Time-out Period (no postscaler) | TBD | TBD | TBD | ms | |
| 32 | TOST | Oscillation Start-up Timer Period | 1024 T _{osc} | — | 1024 T _{osc} | — | T _{osc} = OSC1 period |
| 33 | TPWRT | Power-up Timer Period | 55.6 | 65.5 | 75 | ms | |
| 34 | TIOZ | I/O High-Impedance from MCLR Low or Watchdog Timer Reset | — | 2 | — | μs | |
| 38 | TcSD | CPU Start-up Time | — | 200 | — | μs | |
| 39 | TIOBST | Time for INTRC to Stabilize | — | TBD | — | μs | |

Legend: TBD = To Be Determined

PIC18F45J10 FAMILY

FIGURE 23-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



Note: Refer to Figure 23-3 for load conditions.

TABLE 23-11: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions | |
|-----------|-----------|---|-----------------------------|--|-------|------------|--|
| 40 | Tt0H | T0CKI High Pulse Width | No prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With prescaler | 10 | — | ns | |
| 41 | Tt0L | T0CKI Low Pulse Width | No prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With prescaler | 10 | — | ns | |
| 42 | Tt0P | T0CKI Period | No prescaler | $T_{CY} + 10$ | — | ns | |
| | | | With prescaler | Greater of: 20 ns or $(T_{CY} + 40)/N$ | — | ns | N = prescale value (1, 2, 4, ..., 256) |
| 45 | Tt1H | T1CKI High Time | Synchronous, no prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | Synchronous, with prescaler | 10 | — | ns | |
| | | | Asynchronous | 30 | — | ns | |
| 46 | Tt1L | T1CKI Low Time | Synchronous, no prescaler | $0.5 T_{CY} + 5$ | — | ns | |
| | | | Synchronous, with prescaler | 10 | — | ns | |
| | | | Asynchronous | 30 | — | ns | |
| 47 | Tt1P | T1CKI Input Period | Synchronous | Greater of: 20 ns or $(T_{CY} + 40)/N$ | — | ns | N = prescale value (1, 2, 4, 8) |
| | | | Asynchronous | 60 | — | ns | |
| | Ft1 | T1CKI Oscillator Input Frequency Range | DC | 50 | kHz | | |
| 48 | Tcke2TMR1 | Delay from External T1CKI Clock Edge to Timer Increment | $2 T_{osc}$ | $7 T_{osc}$ | — | | |

PIC18F45J10 FAMILY

FIGURE 23-9: CAPTURE/COMPARE/PWM TIMINGS (INCLUDING ECCP MODULE)

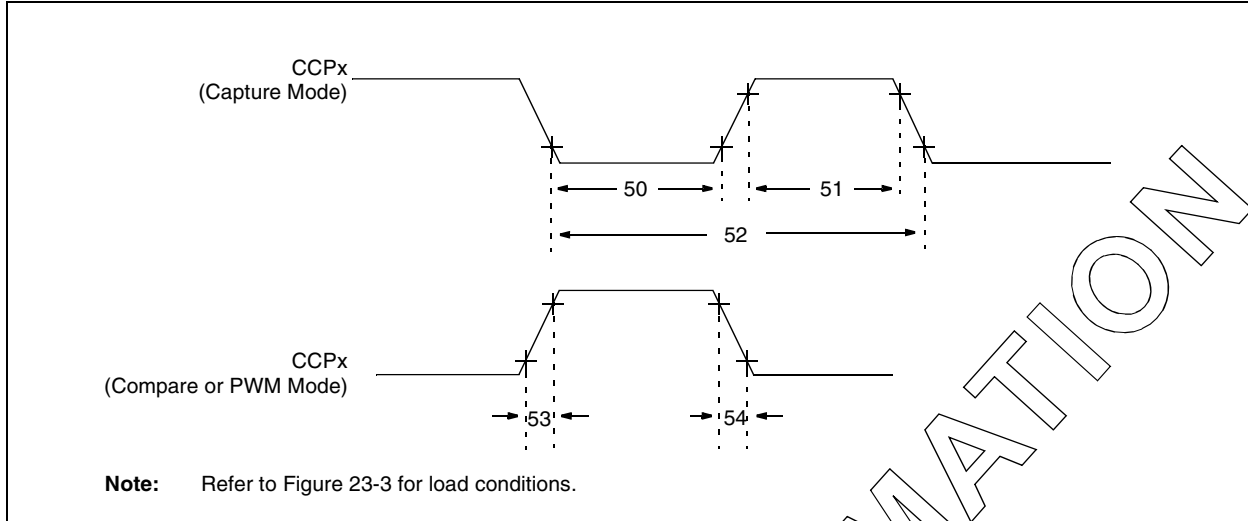


TABLE 23-12: CAPTURE/COMPARE/PWM REQUIREMENTS (INCLUDING ECCP MODULE)

| Param No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|-----------|--------|-----------------------|----------------|---------------------------|-----|-------|---------------------------------|
| 50 | TccL | CCPx Input Low Time | No prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With prescaler | 10 | — | ns | |
| 51 | TccH | CCPx Input High Time | No prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With prescaler | 10 | — | 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 | | — | 25 | ns | |
| 54 | TccF | CCPx Output Rise Time | | — | 25 | ns | |

TABLE 23-13: PARALLEL SLAVE PORT REQUIREMENTS

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|------------|----------|---|-----|------------|-------|------------|
| 62 | TdtV2wrH | Data In Valid before $\overline{WR} \uparrow$ or $\overline{CS} \uparrow$ (setup time) | 20 | — | ns | |
| 63 | TwrH2dtl | $\overline{WR} \uparrow$ or $\overline{CS} \uparrow$ to Data-In Invalid (hold time) | 20 | — | ns | |
| 64 | TrdL2dtV | $\overline{RD} \downarrow$ and $\overline{CS} \downarrow$ to Data-Out Valid | — | 80 | ns | |
| 65 | TrdH2dtl | $\overline{RD} \uparrow$ or $\overline{CS} \downarrow$ to Data-Out Invalid | 10 | 30 | ns | |
| 66 | TtbfINH | Inhibit of the IBF Flag bit being Cleared from $\overline{WR} \uparrow$ or $\overline{CS} \uparrow$ | — | $3 T_{CY}$ | | |

PIC18F45J10 FAMILY

FIGURE 23-10: EXAMPLE SPI™ MASTER MODE TIMING (CKE = 0)

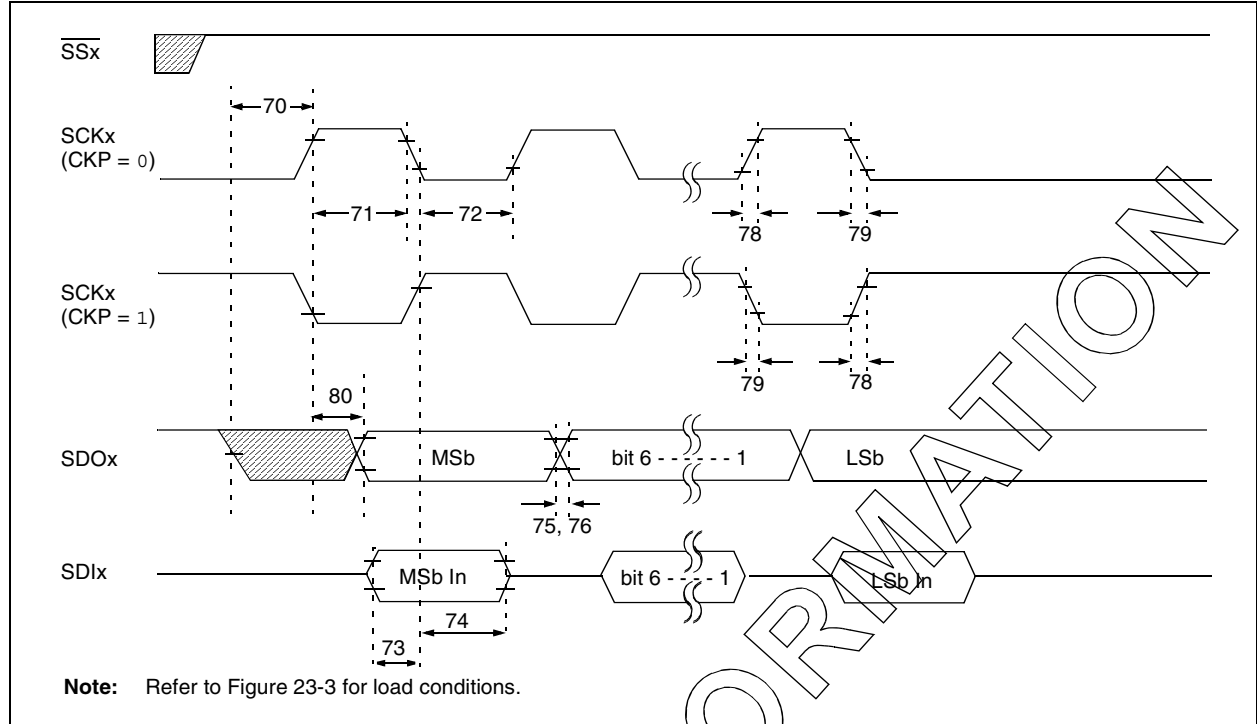


TABLE 23-14: EXAMPLE SPI™ MODE REQUIREMENTS (MASTER MODE, CKE = 0)

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|-----------|--------------------|---|--------------------------|---------------------------|-------|------------|
| 70 | TsL2sCH, TsL2sCL | SSx ↓ to SCKx ↓ or SCKx ↑ Input | T _{CY} | — | ns | |
| 71 | T _{sCH} | SCKx Input High Time (Slave mode) | Continuous | 1.25 T _{CY} + 30 | — | ns |
| 71A | | | Single Byte | 40 | — | ns |
| 72 | T _{sCL} | SCKx Input Low Time (Slave mode) | Continuous | 1.25 T _{CY} + 30 | — | ns |
| 72A | | | Single Byte | 40 | — | ns |
| 73 | TdIV2sCH, TdIV2sCL | Setup Time of SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 73A | T _{B2B} | Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2 | 1.5 T _{CY} + 40 | — | ns | (Note 2) |
| 74 | TsCH2dIL, TsCL2dIL | Hold Time of SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 75 | T _{DOH} | SDOx Data Output Rise Time | — | 25 | ns | |
| 76 | T _{DOF} | SDOx Data Output Fall Time | — | 25 | ns | |
| 78 | T _{sCH} | SCKx Output Rise Time (Master mode) | — | 25 | ns | |
| 79 | T _{sCF} | SCKx Output Fall Time (Master mode) | — | 25 | ns | |
| 80 | TsCH2DoV, TsCL2DoV | SDOx Data Output Valid after SCKx Edge | — | 50 | ns | |

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

PIC18F45J10 FAMILY

FIGURE 23-11: EXAMPLE SPI™ MASTER MODE TIMING (CKE = 1)

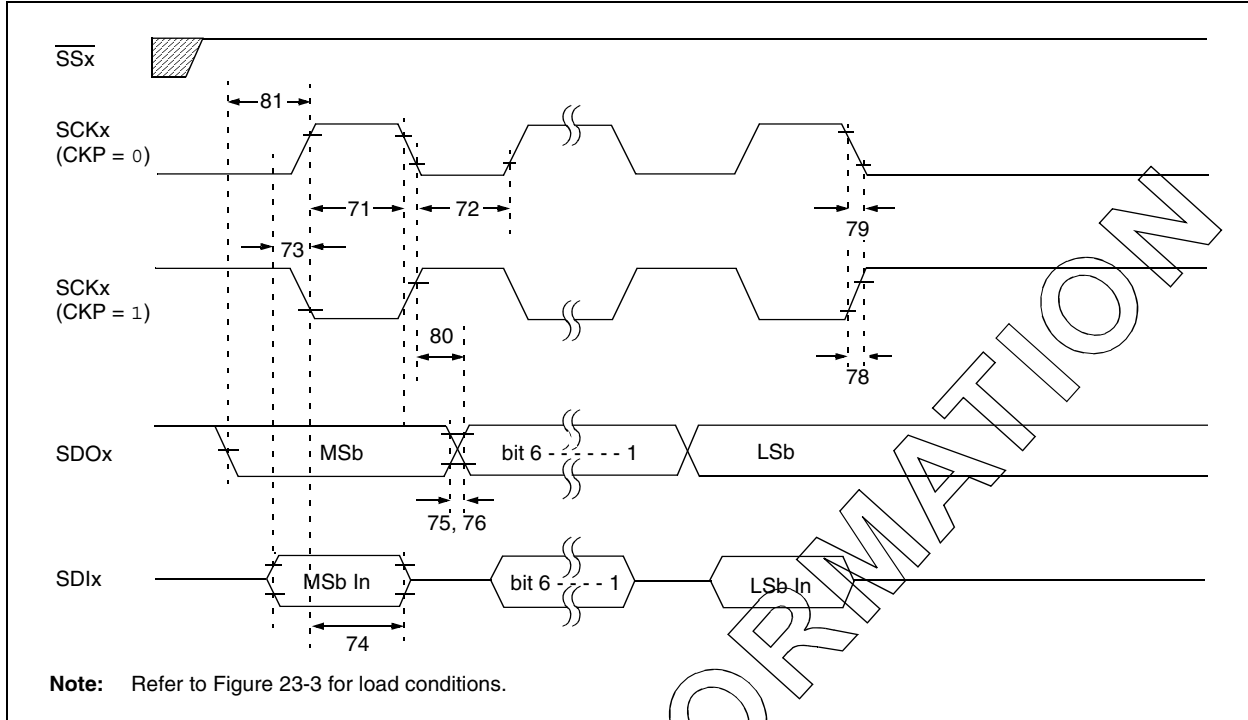


TABLE 23-15: EXAMPLE SPI™ MODE REQUIREMENTS (MASTER MODE, CKE = 1)

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|------------|--------------------|---|---------------|-----|-------|------------|
| 71 | Tsch | SCKx Input High Time (Slave mode) | 1.25 Tcy + 30 | — | ns | |
| 71A | | Single Byte | 40 | — | ns | (Note 1) |
| 72 | Tscl | SCKx Input Low Time (Slave mode) | 1.25 Tcy + 30 | — | ns | |
| 72A | | Single Byte | 40 | — | ns | (Note 1) |
| 73 | TdIV2sch, TdIV2scl | Setup Time of SDIx Data Input to SCKx 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 SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 75 | TdOR | SDOx Data Output Rise Time | — | 25 | ns | |
| 76 | TdOF | SDOx Data Output Fall Time | — | 25 | ns | |
| 78 | TscR | SCKx Output Rise Time (Master mode) | — | 25 | ns | |
| 79 | TscF | SCKx Output Fall Time (Master mode) | — | 25 | ns | |
| 80 | Tsch2doV, TscL2doV | SDOx Data Output Valid after SCKx Edge | — | 50 | ns | |
| 81 | TdoV2sch, TdoV2scl | SDOx Data Output Setup to SCKx Edge | Tcy | — | ns | |

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

PIC18F45J10 FAMILY

FIGURE 23-12: EXAMPLE SPI™ SLAVE MODE TIMING (CKE = 0)

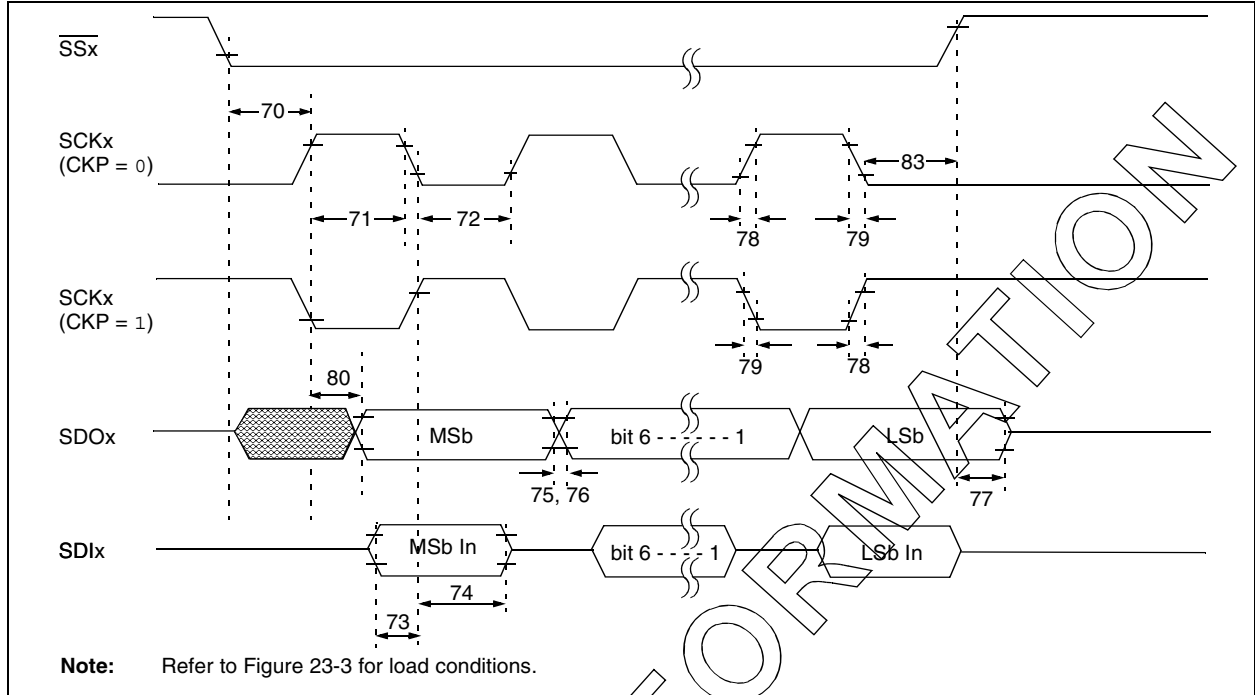


TABLE 23-16: EXAMPLE SPI™ MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|-----------|--------------------|---|--------------------------|---------------------------|-------|-----------------|
| 70 | Tssl2sch, Tssl2scL | $\overline{SSx} \downarrow$ to SCKx \downarrow or SCKx \uparrow Input | T _{CY} | — | ns | |
| 71 | Tsch | SCKx Input High Time (Slave mode) | Continuous | 1.25 T _{CY} + 30 | — | ns |
| 71A | | | Single Byte | 40 | — | ns |
| 72 | TscL | SCKx Input Low Time (Slave mode) | Continuous | 1.25 T _{CY} + 30 | — | ns |
| 72A | | | Single Byte | 40 | — | ns |
| 73 | TdIV2sch, TdIV2scL | Setup Time of SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 73A | Tb2B | Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2 | 1.5 T _{CY} + 40 | — | ns | (Note 2) |
| 74 | Tsch2dIL, TscL2dIL | Hold Time of SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 75 | TdOR | SDOx Data Output Rise Time | — | 25 | ns | |
| 76 | TdOF | SDOx Data Output Fall Time | — | 25 | ns | |
| 77 | Tssh2boZ | $\overline{SSx} \uparrow$ to SDOx Output High-Impedance | 10 | 50 | ns | |
| 78 | TscR | SCKx Output Rise Time (Master mode) | — | 25 | ns | |
| 79 | TscF | SCKx Output Fall Time (Master mode) | — | 25 | ns | |
| 80 | Tsch2doV, TscL2doV | SDOx Data Output Valid after SCKx Edge | — | 50 | ns | |
| 83 | Tsch2ssH, TscL2ssH | $\overline{SSx} \uparrow$ after SCKx Edge | 1.5 T _{CY} + 40 | — | ns | |

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

PIC18F45J10 FAMILY

FIGURE 23-13: EXAMPLE SPI™ SLAVE MODE TIMING (CKE = 1)

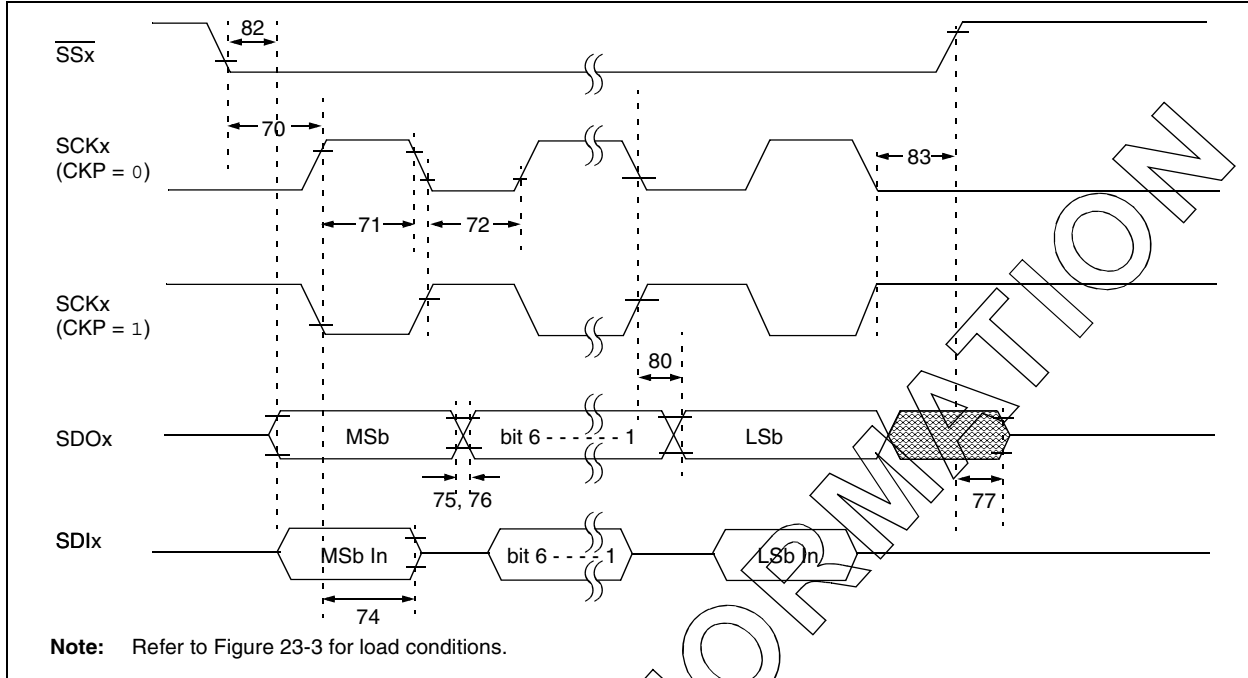


TABLE 23-17: EXAMPLE SPI™ SLAVE MODE REQUIREMENTS (CKE = 1)

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|-----------|--------------------|---|---------------|-----|-------|------------|
| 70 | TssL2sCH, TssL2sCL | SSx ↓ to SCKx ↓ or SCKx ↑ Input | Tcy | — | ns | |
| 71 | Tsch | SCKx Input High Time (Slave mode) | 1.25 Tcy + 30 | — | ns | |
| 71A | | Single Byte | 40 | — | ns | (Note 1) |
| 72 | Tscl | SCKx Input Low Time (Slave mode) | 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 SDIx Data Input to SCKx Edge | 100 | — | ns | |
| 75 | TdoR | SDOx Data Output Rise Time | — | 25 | ns | |
| 76 | TdoF | SDOx Data Output Fall Time | — | 25 | ns | |
| 77 | TssH2doZ | SSx ↑ to SDOx Output High-Impedance | 10 | 50 | ns | |
| 78 | TscR | SCKx Output Rise Time (Master mode) | — | 25 | ns | |
| 79 | TscF | SCKx Output Fall Time (Master mode) | — | 25 | ns | |
| 80 | Tsch2doV, Tscl2doV | SDOx Data Output Valid after SCKx Edge | — | 50 | ns | |
| 82 | TssL2doV | SDOx Data Output Valid after SSx ↓ Edge | — | 50 | ns | |
| 83 | Tsch2ssH, Tscl2ssH | SSx ↑ after SCKx Edge | 1.5 Tcy + 40 | — | ns | |

Note 1: Requires the use of Parameter #73A.

Note 2: Only if Parameter #71A and #72A are used.

PIC18F45J10 FAMILY

FIGURE 23-14: I²C™ BUS START/STOP BITS TIMING

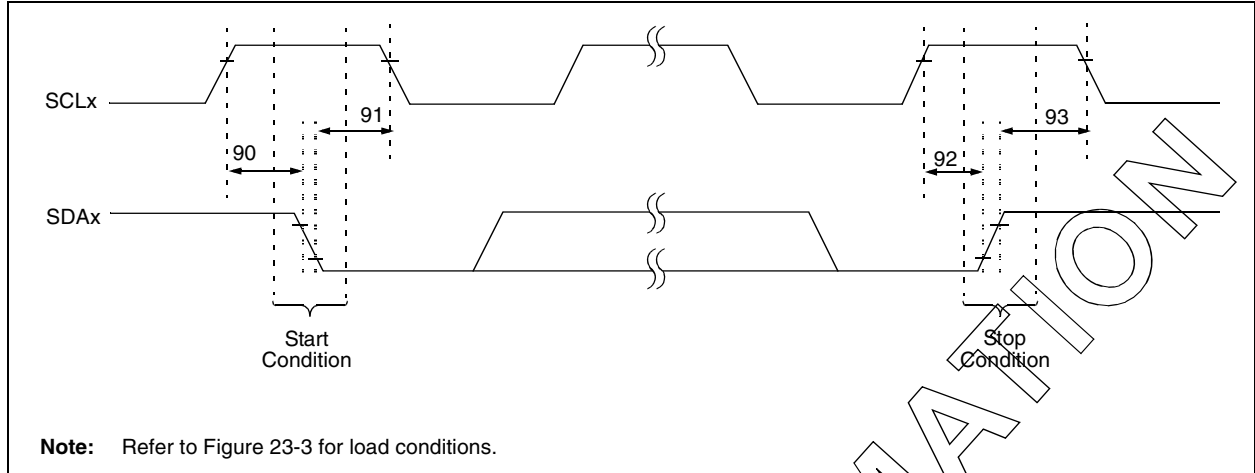
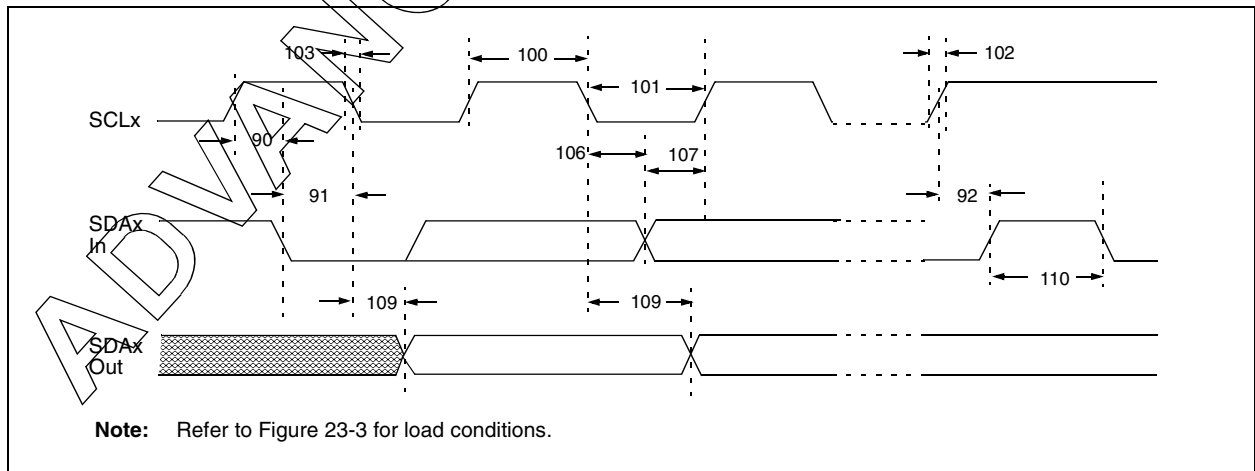


TABLE 23-18: I²C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|------------|---------|----------------------------|--------------|------|-----|-------|---|
| 90 | TSU:STA | Start Condition Setup Time | 100 kHz mode | 4700 | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 600 | — | | |
| 91 | THD:STA | Start Condition Hold Time | 100 kHz mode | 4000 | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 600 | — | | |
| 92 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | 4700 | — | ns | |
| | | | 400 kHz mode | 600 | — | | |
| 93 | THD:STO | Stop Condition Hold Time | 100 kHz mode | 4000 | — | ns | |
| | | | 400 kHz mode | 600 | — | | |

FIGURE 23-15: I²C™ BUS DATA TIMING



PIC18F45J10 FAMILY

TABLE 23-19: I²C™ BUS DATA REQUIREMENTS (SLAVE MODE)

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|------------|---------|----------------------------|--------------|-------------------------|------|-------|---|
| 100 | THIGH | Clock High Time | 100 kHz mode | 4.0 | — | μs | PIC18F45J10 must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 0.6 | — | μs | PIC18F45J10 must operate at a minimum of 10 MHz |
| | | | MSSP Module | 1.5 T _{cy} | — | | |
| 101 | TLOW | Clock Low Time | 100 kHz mode | 4.7 | — | μs | PIC18F45J10 must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 1.3 | — | μs | PIC18F45J10 must operate at a minimum of 10 MHz |
| | | | MSSP Module | 1.5 T _{cy} | — | | |
| 102 | TR | SDAx and SCLx Rise Time | 100 kHz mode | — | 1000 | ns | C _B is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 C _B | 300 | ns | |
| 103 | TF | SDAx and SCLx Fall Time | 100 kHz mode | — | 300 | ns | C _B is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 C _B | 300 | ns | |
| 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 SCLx to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I²C™ bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, $T_{R\ max.} + TSU:DAT = 1000 + 250 = 1250$ ns (according to the Standard mode I²C bus specification), before the SCLx line is released.

PIC18F45J10 FAMILY

FIGURE 23-16: MASTER SSP I²C™ BUS START/STOP BITS TIMING WAVEFORMS

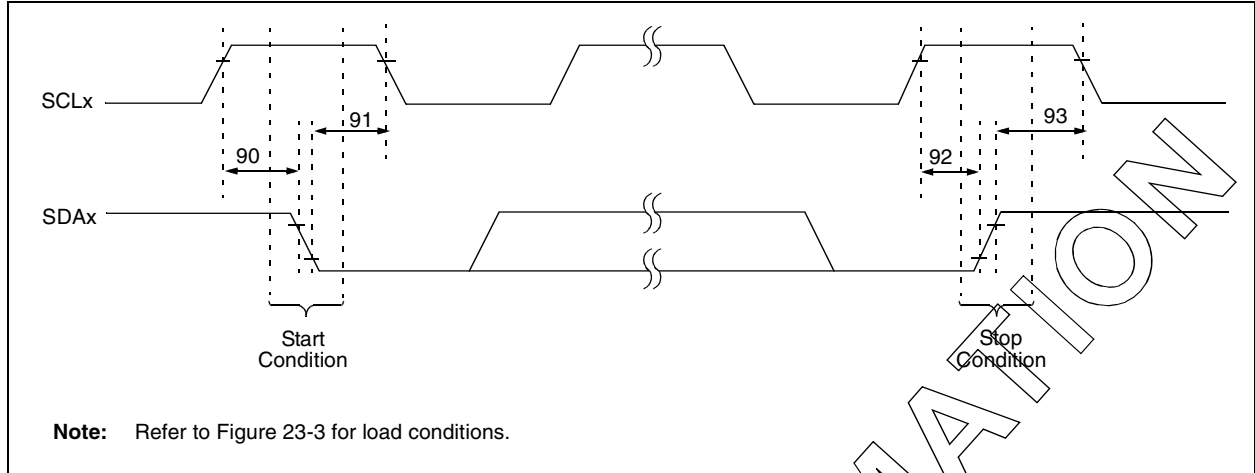
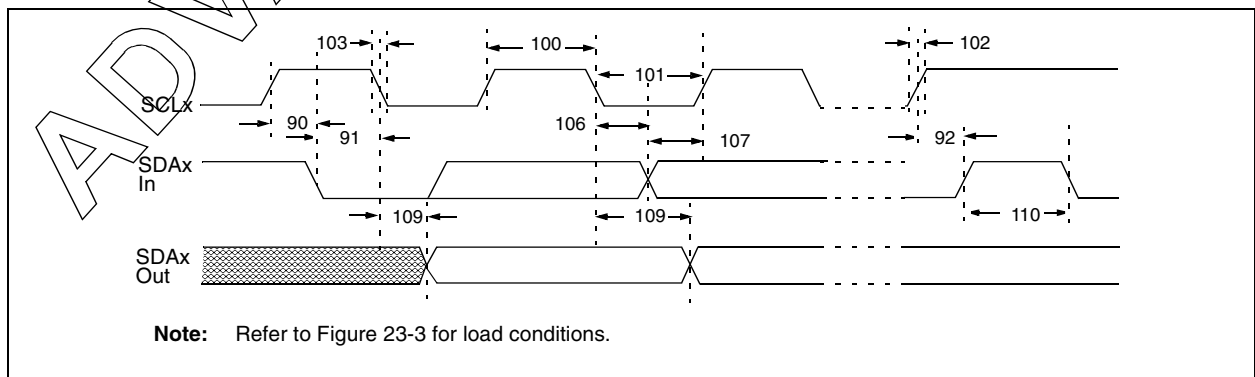


TABLE 23-20: MASTER SSP I²C™ BUS START/STOP BITS REQUIREMENTS

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|------------|---------|----------------------------|---------------------------|-----------------------|-----|-------|---|
| 90 | TSU:STA | Start Condition Setup Time | 100 kHz mode | $2(T_{osc})(BRG + 1)$ | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | $2(T_{osc})(BRG + 1)$ | — | | |
| | | | 1 MHz mode ⁽¹⁾ | $2(T_{osc})(BRG + 1)$ | — | | |
| 91 | THD:STA | Start Condition Hold Time | 100 kHz mode | $2(T_{osc})(BRG + 1)$ | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | $2(T_{osc})(BRG + 1)$ | — | | |
| | | | 1 MHz mode ⁽¹⁾ | $2(T_{osc})(BRG + 1)$ | — | | |
| 92 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | $2(T_{osc})(BRG + 1)$ | — | ns | |
| | | | 400 kHz mode | $2(T_{osc})(BRG + 1)$ | — | | |
| | | | 1 MHz mode ⁽¹⁾ | $2(T_{osc})(BRG + 1)$ | — | | |
| 93 | THD:STO | Stop Condition Hold Time | 100 kHz mode | $2(T_{osc})(BRG + 1)$ | — | ns | |
| | | | 400 kHz mode | $2(T_{osc})(BRG + 1)$ | — | | |
| | | | 1 MHz mode ⁽¹⁾ | $2(T_{osc})(BRG + 1)$ | — | | |

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

FIGURE 23-17: MASTER SSP I²C™ BUS DATA TIMING



PIC18F45J10 FAMILY

TABLE 23-21: MASTER SSP I²C™ 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 | SDAx and SCLx Rise Time | 100 kHz mode | — | 1000 | ns |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns |
| | | | 1 MHz mode ⁽¹⁾ | — | 300 | ns |
| 103 | TF | SDAx and SCLx Fall Time | 100 kHz mode | — | 300 | ns |
| | | | 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 |
| | | | 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 |
| | | | 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 |
| | | | 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 |
| | | | 400 kHz mode | 1.3 | — | ms |
| | | | 1 MHz mode ⁽¹⁾ | TBD | — | ms |
| D102 | CB | Bus Capacitive Loading | — | 400 | pF | |

Legend: TBD = To Be Determined

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

Note 2: A Fast mode I²C bus device can be used in a Standard mode I²C 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 SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, parameter #102 + parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCLx line is released.

PIC18F45J10 FAMILY

FIGURE 23-18: EUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

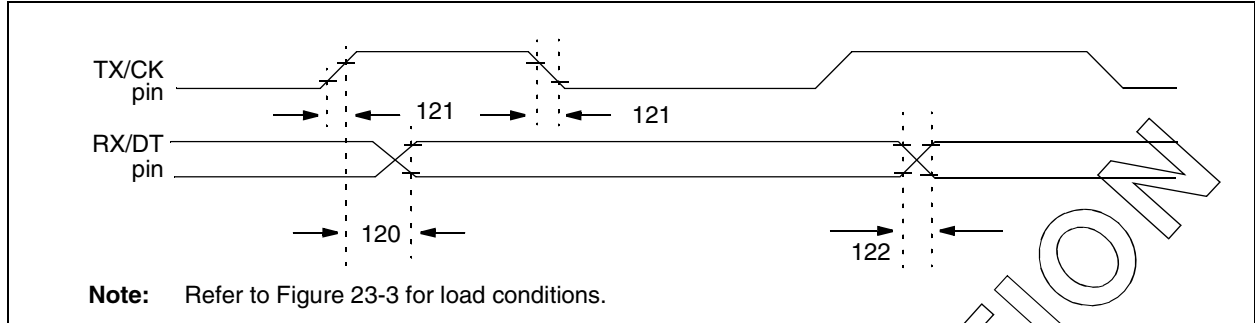


TABLE 23-22: EUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|-----------|----------|--|-----|-----|-------|------------|
| 120 | TckH2DTV | SYNC XMIT (MASTER and SLAVE) Clock High to Data Out Valid | — | 40 | ns | |
| 121 | TckRF | Clock Out Rise Time and Fall Time (Master mode) | — | 20 | ns | |
| 122 | TdTRF | Data Out Rise Time and Fall Time | — | 20 | ns | |

FIGURE 23-19: EUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

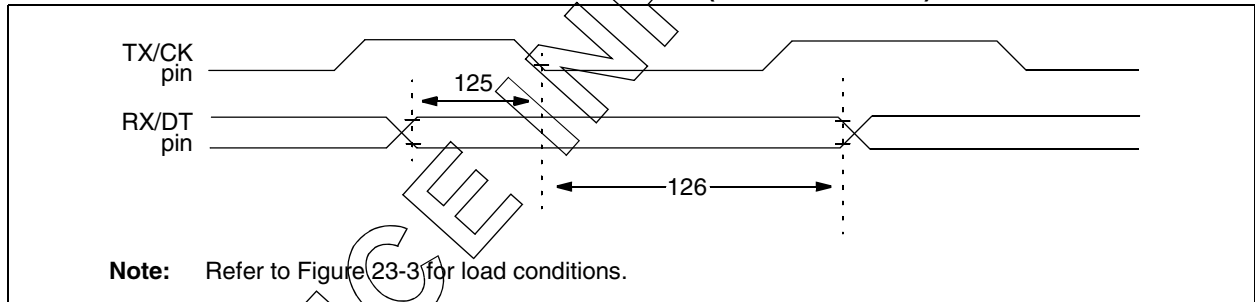


TABLE 23-23: EUSART SYNCHRONOUS RECEIVE REQUIREMENTS

| Param. No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|------------|-----------|---|-----|-----|-------|------------|
| 125 | TDTV2cckL | SYNC RCV (MASTER and SLAVE) Data Hold before CK ↓ (DT hold time) | 10 | — | ns | |
| 126 | TckL2DTL | Data Hold after CK ↓ (DT hold time) | 15 | — | ns | |

PIC18F45J10 FAMILY

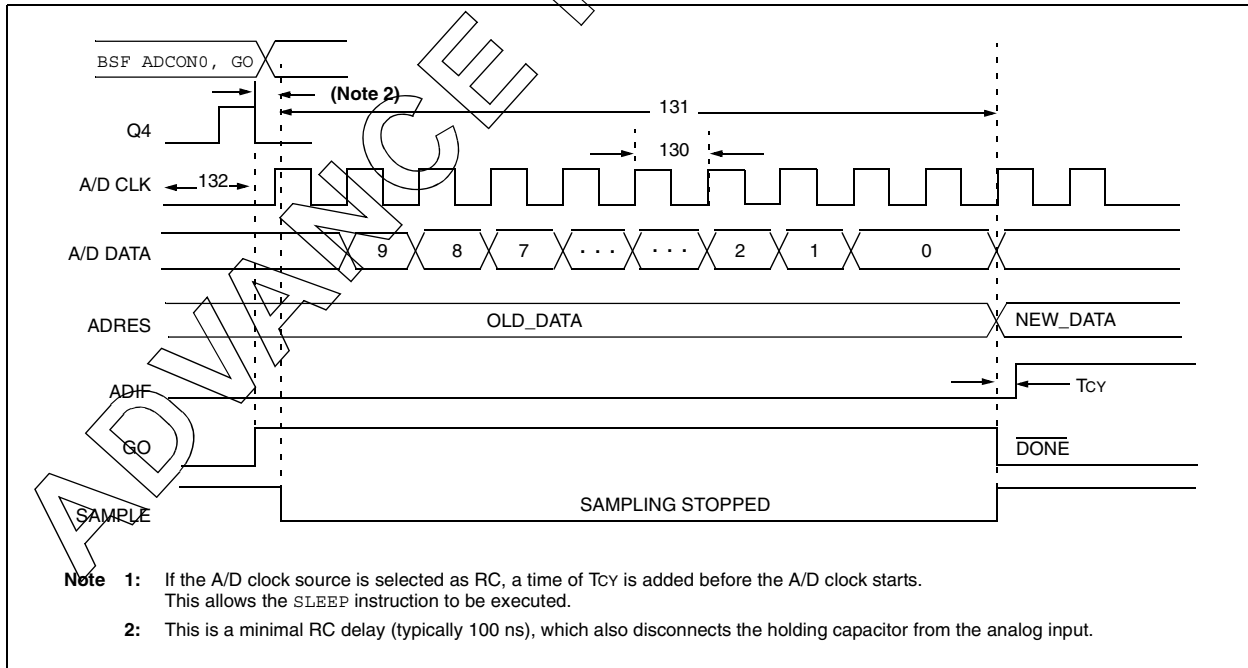
TABLE 23-24: A/D CONVERTER CHARACTERISTICS: PIC18F24J10/25J10/44J10/45J10 (INDUSTRIAL)

| Param No. | Symbol | Characteristic | Min | Typ | Max | Units | Conditions |
|-----------|------------------|--|---------------------------|-----|-----------------|------------|--|
| A01 | NR | Resolution | — | — | 10 | bit | $\Delta V_{REF} \geq 3.0V$ |
| A03 | EIL | Integral Linearity Error | — | — | $<\pm 1$ | LSb | $\Delta V_{REF} \geq 3.0V$ |
| A04 | EDL | Differential Linearity Error | — | — | $<\pm 1$ | LSb | $\Delta V_{REF} \geq 3.0V$ |
| A06 | EOFF | Offset Error | — | — | $<\pm 1.5$ | LSb | $\Delta V_{REF} \geq 3.0V$ |
| A07 | EGN | Gain Error | — | — | $<\pm 1$ | LSb | $\Delta V_{REF} \geq 3.0V$ |
| A10 | — | Monotonicity | Guaranteed ⁽¹⁾ | | | — | $V_{SS} \leq V_{AIN} \leq V_{REF}$ |
| A20 | ΔV_{REF} | Reference Voltage Range ($V_{REFH} - V_{REFL}$) | 1.8 | — | — | V | $V_{DD} < 3.0V$ |
| | | | 3 | — | — | V | $V_{DD} \geq 3.0V$ |
| A21 | V_{REFH} | Reference Voltage High | V_{SS} | — | V_{REFH} | V | |
| A22 | V_{REFL} | Reference Voltage Low | $V_{SS} - 0.3V$ | — | $V_{DD} - 3.0V$ | V | |
| A25 | V_{AIN} | Analog Input Voltage | V_{REFL} | — | V_{REFH} | V | |
| A30 | Z_{AIN} | Recommended Impedance of Analog Voltage Source | — | — | 2.5 | k Ω | |
| A50 | I $_{REF}$ | V_{REF} Input Current ⁽²⁾ | — | — | 5 | μA | During V_{AIN} acquisition. During A/D conversion cycle. |
| | | | — | — | 150 | μA | |

Note 1: The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

- 2:** V_{REFH} current is from RA3/AN3/ V_{REF+} pin or V_{DD} , whichever is selected as the V_{REFH} source.
 V_{REFL} current is from RA2/AN2/ V_{REF-} pin or V_{SS} , whichever is selected as the V_{REFL} source.

FIGURE 23-20: A/D CONVERSION TIMING



PIC18F45J10 FAMILY

TABLE 23-25: A/D CONVERSION REQUIREMENTS

| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
|-----------|--------|---|-----|---------------------|-------|-------------------------|
| 130 | TAD | A/D Clock Period | 0.7 | 25.0 ⁽¹⁾ | μs | TOSC based, VREF ≥ 3.0V |
| | | | TBD | 1 | μs | A/D RC mode |
| 131 | TcNV | Conversion Time (not including acquisition time) (Note 2) | 11 | 12 | TAD | |
| 132 | TACQ | Acquisition Time (Note 3) | 1.4 | — | μs | -40°C to +85°C |
| | | | TBD | — | μs | 0°C ≤ to ≤ +85°C |
| 135 | TswC | Switching Time from Convert → Sample | — | (Note 4) | | |

Legend: TBD = To Be Determined

Note 1: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

2: ADRES registers may be read on the following TcY cycle.

3: The time for the holding capacitor to acquire the “New” input voltage when the voltage changes full scale after the conversion (VDD to VSS or VSS to VDD). The source impedance (Rs) on the input channels is 50Ω.

4: On the following cycle of the device clock.

ADVANCE INFORMATION

PIC18F45J10 FAMILY

NOTES:

24.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Graphs and tables are not available at this time.

PIC18F45J10 FAMILY

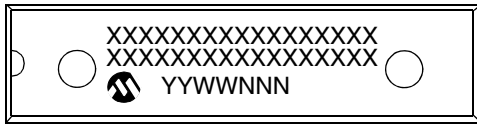
NOTES:

PIC18F45J10 FAMILY

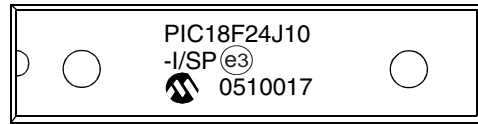
25.0 PACKAGING INFORMATION

25.1 Package Marking Information

28-Lead SPDIP



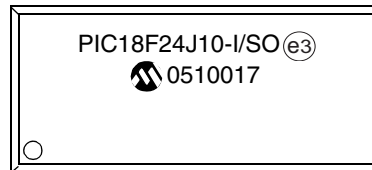
Example



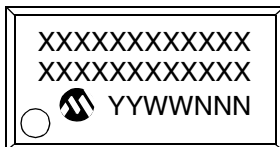
28-Lead SOIC



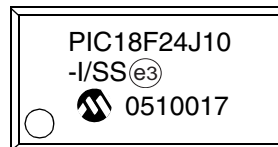
Example



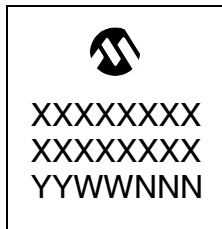
28-Lead SSOP



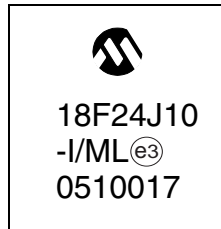
Example



28-Lead QFN



Example



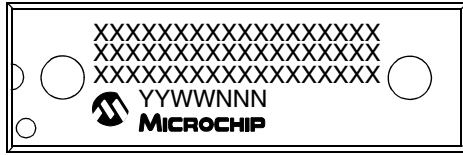
| | | |
|----------------|--------|--|
| Legend: | XX...X | Customer-specific information |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC designator for Matte Tin (Sn) |
| | * | This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package. |

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

PIC18F45J10 FAMILY

Package Marking Information (Continued)

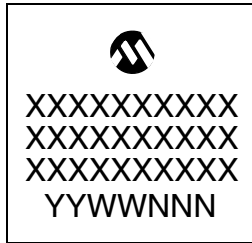
40-Lead PDIP



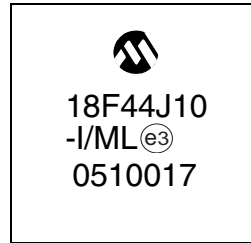
Example



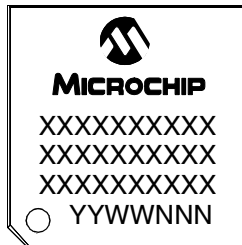
44-Lead QFN



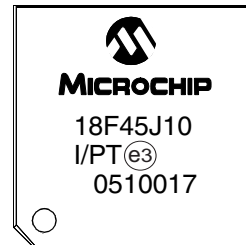
Example



44-Lead TQFP



Example

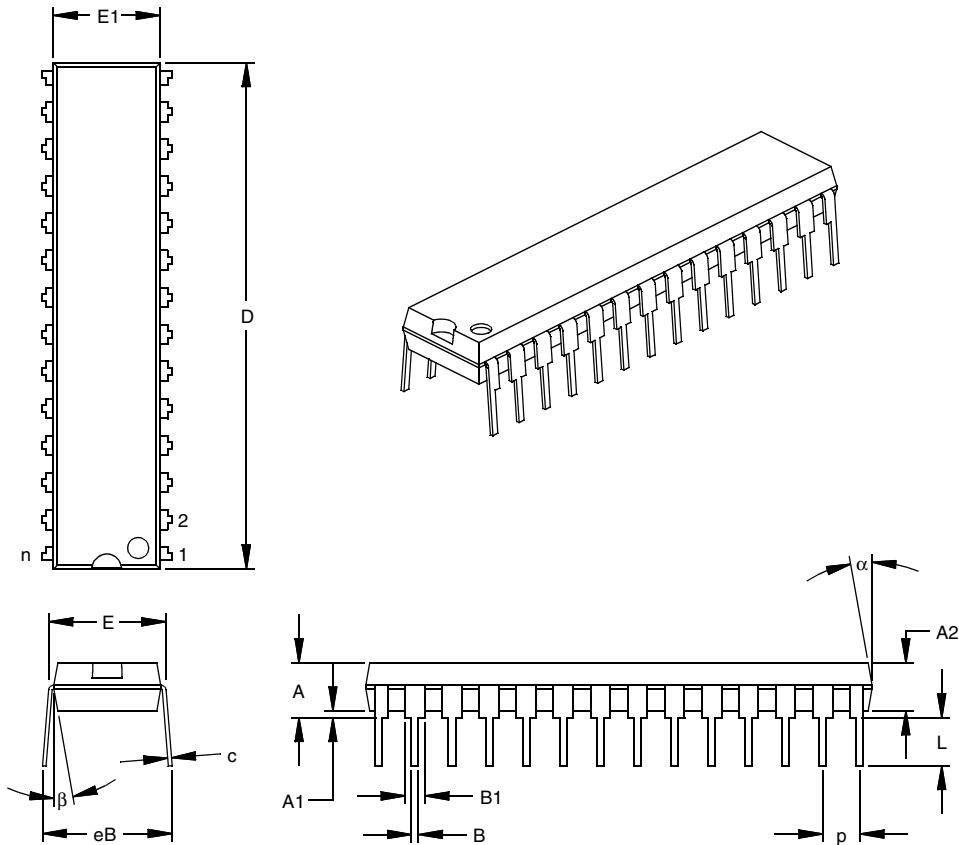


PIC18F45J10 FAMILY

25.2 Package Details

The following sections give the technical details of the packages.

28-Lead Skinny Plastic Dual In-line (SP) – 300 mil Body (PDIP)



| Dimension Limits | Units | INCHES* | | | MILLIMETERS | | |
|----------------------------|-------|---------|-------|-------|-------------|-------|-------|
| | | 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

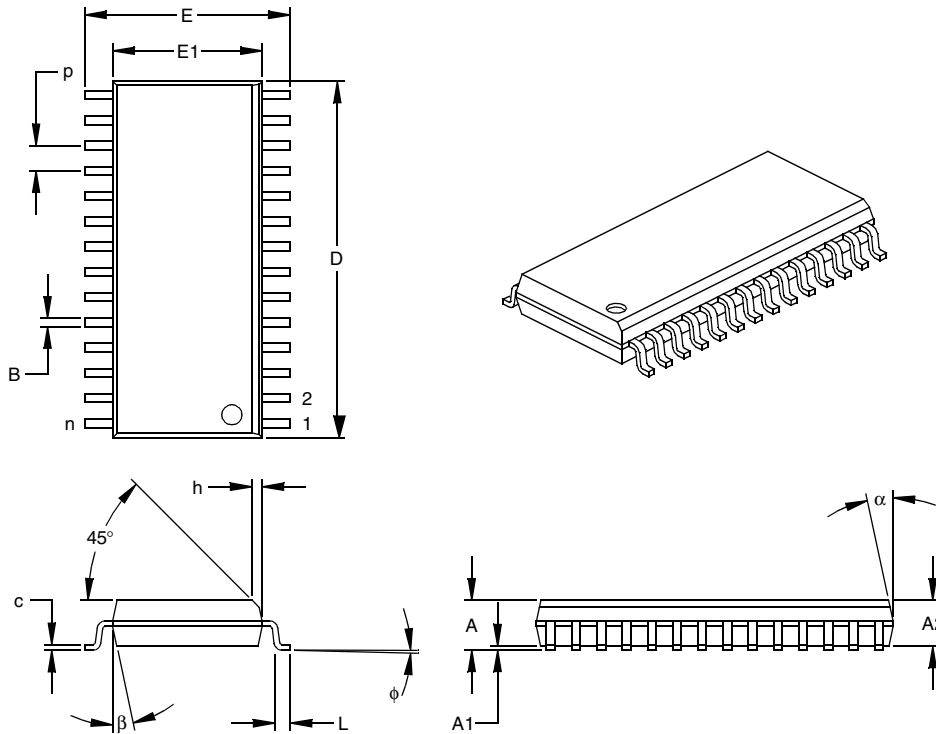
.010" (0.254mm) per side.

JEDEC Equivalent: MO-095

Drawing No. C04-070

PIC18F45J10 FAMILY

28-Lead Plastic Small Outline (SO) – Wide, 300 mil Body (SOIC)



| Dimension Limits | Units | INCHES* | | | MILLIMETERS | | |
|--------------------------|-------|---------|------|------|-------------|-------|-------|
| | | 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:

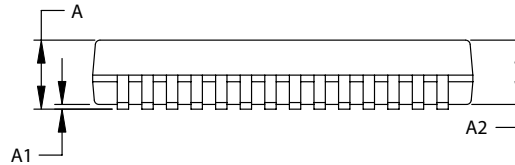
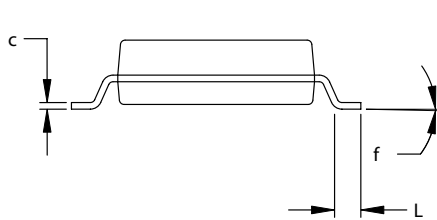
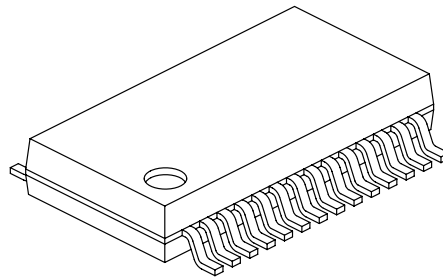
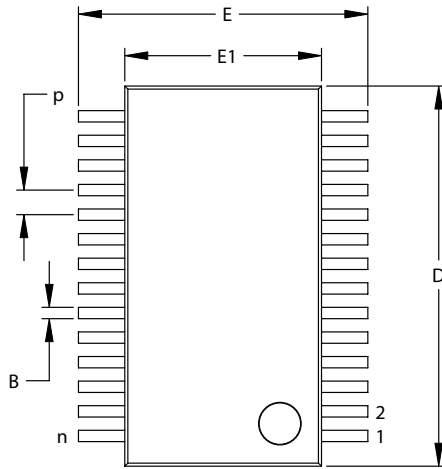
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-013

Drawing No. C04-052

PIC18F45J10 FAMILY

28-Lead Plastic Shrink Small Outline (SS) – 209 mil Body, 5.30 mm (SSOP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|-------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | 28 | | | 28 | | |
| Pitch | P | | .026 | | | 0.65 | |
| Overall Height | A | - | - | .079 | - | - | 2.0 |
| Molded Package Thickness | A2 | .065 | .069 | .073 | 1.65 | 1.75 | 1.85 |
| Standoff | A1 | .002 | - | - | 0.05 | - | - |
| Overall Width | E | .295 | .307 | .323 | 7.49 | 7.80 | 8.20 |
| Molded Package Width | E1 | .009 | .209 | .220 | 5.00 | 5.30 | 5.60 |
| Overall Length | D | .390 | .402 | .413 | 9.90 | 10.20 | 10.50 |
| Foot Length | L | .022 | .030 | .037 | 0.55 | 0.75 | 0.95 |
| Lead Thickness | c | .004 | - | .010 | 0.09 | - | 0.25 |
| Foot Angle | f | 0° | 4° | 8° | 0° | 4° | 8° |
| Lead Width | B | .009 | - | .015 | 0.22 | - | 0.38 |

*Controlling Parameter

Notes:

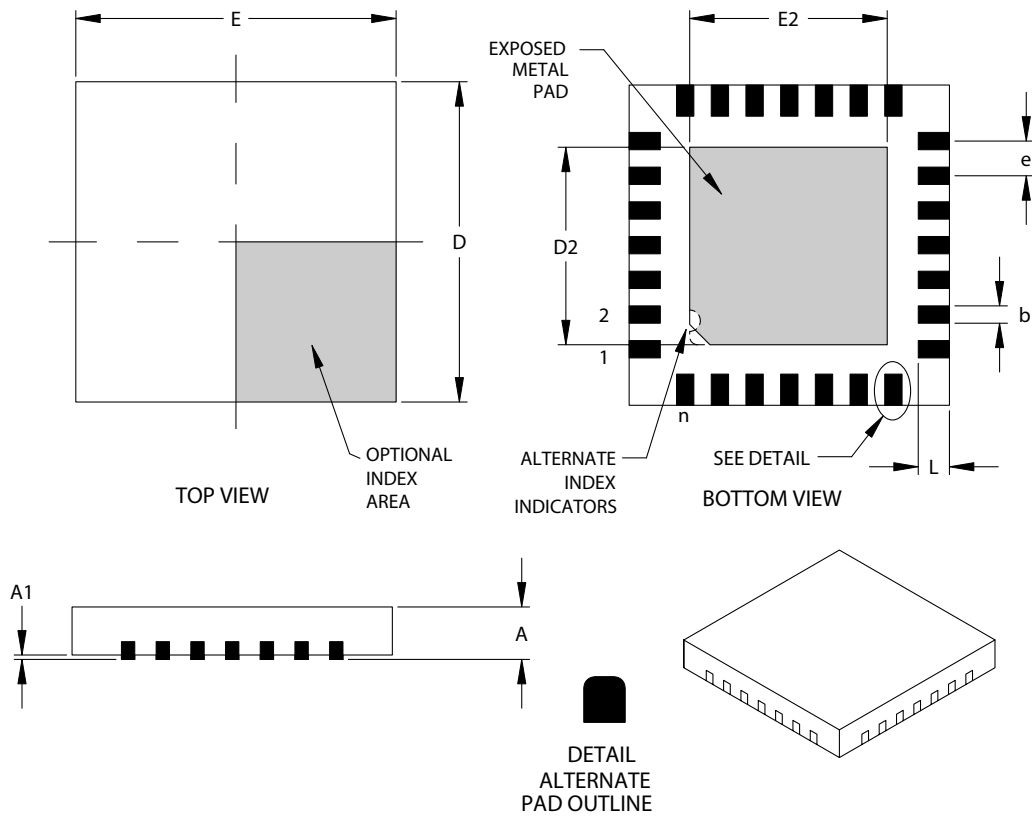
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-150

Drawing No. C04-073

PIC18F45J10 FAMILY

28-Lead Plastic Quad Flat No Lead Package (ML) 6x6 mm Body (QFN) – With 0.55 mm Contact Length (Saw Singulated)



| Units | | INCHES | | | MILLIMETERS* | | |
|--------------------|----|----------|------|------|--------------|------|------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | 28 | | | 28 | | |
| Pitch | e | .026 BSC | | | 0.65 BSC | | |
| Overall Height | A | .031 | .035 | .039 | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | .000 | .001 | .002 | 0.00 | 0.02 | 0.05 |
| Contact Thickness | A3 | .008 REF | | | 0.20 REF | | |
| Overall Width | E | .232 | .236 | .240 | 5.90 | 6.00 | 6.10 |
| Exposed Pad Width | E2 | .140 | .146 | .152 | 3.55 | 3.70 | 3.85 |
| Overall Length | D | .232 | .236 | .240 | 5.90 | 6.00 | 6.10 |
| Exposed Pad Length | D2 | .140 | .146 | .152 | 3.55 | 3.70 | 3.85 |
| Contact Width | b | .009 | .011 | .013 | 0.23 | 0.28 | 0.33 |
| Contact Length | L | .018 | .022 | .024 | 0.45 | 0.55 | 0.65 |

*Controlling Parameter

Notes:

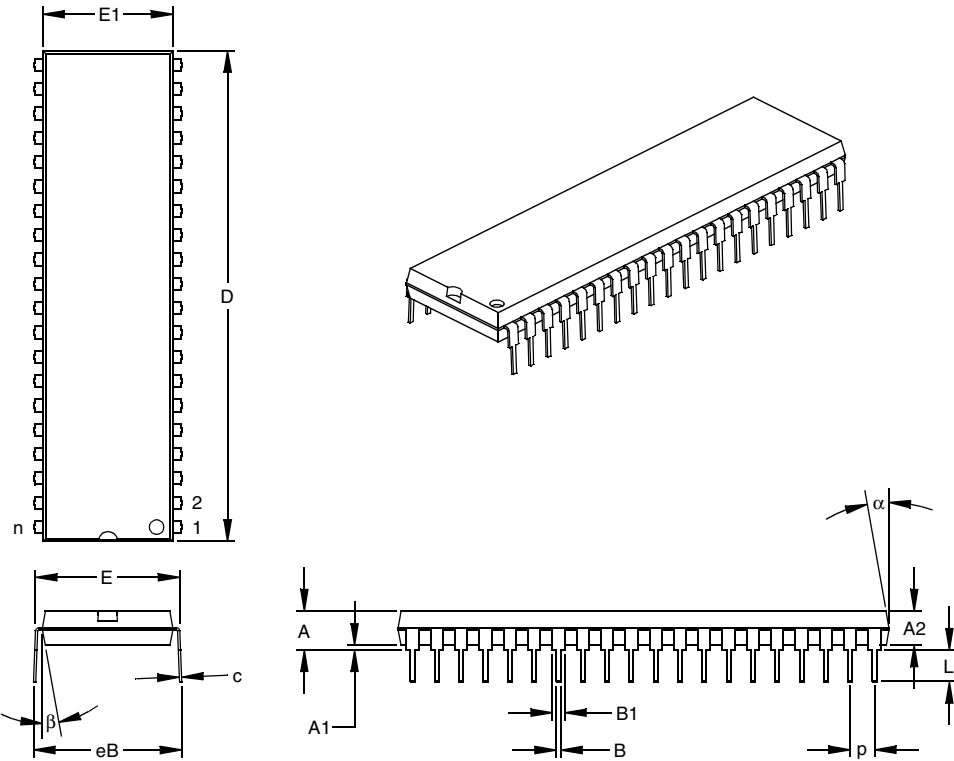
JEDEC equivalent: MO-220

Drawing No. C04-105

Revised 02-18-05

PIC18F45J10 FAMILY

40-Lead Plastic Dual In-line (P) – 600 mil Body (PDIP)



| Dimension Limits | Units | INCHES* | | | MILLIMETERS | | |
|----------------------------|-------|---------|-------|-------|-------------|-------|-------|
| | | 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:

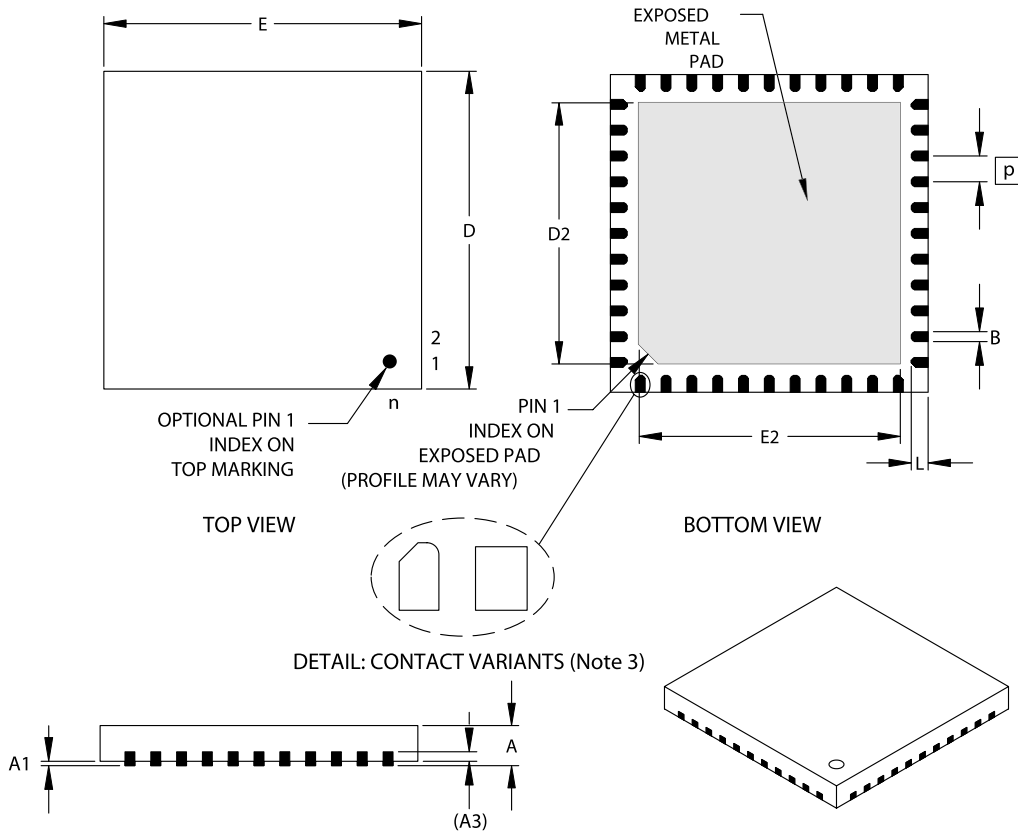
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-011

Drawing No. C04-016

PIC18F45J10 FAMILY

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

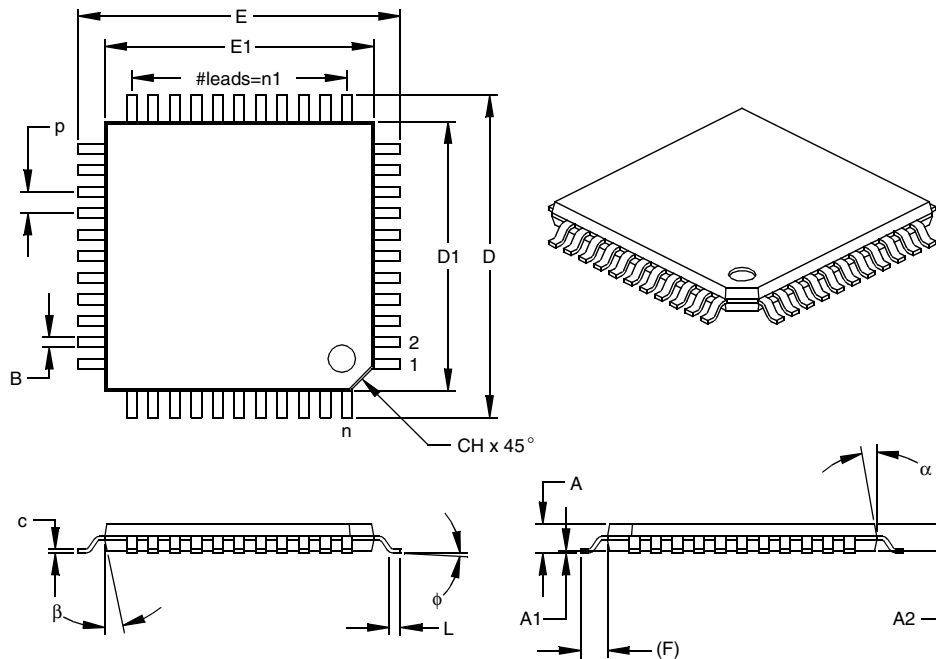
Notes:

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.

JEDEC equivalent: M0-220
Drawing No. C04-103

PIC18F45J10 FAMILY

44-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|-------|
| | | 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

PIC18F45J10 FAMILY

NOTES:

PIC18F45J10 FAMILY

APPENDIX A: REVISION HISTORY

Revision A (March 2005)

Original data sheet for PIC18F45J10 family devices.

APPENDIX B: MIGRATION BETWEEN HIGH-END DEVICE FAMILIES

Devices in the PIC18F45J10 family and PIC18F4520 families are very similar in their functions and feature sets. However, there are some potentially important differences which should be considered when migrating an application across device families to achieve a new design goal. These are summarized in Table B-1. The areas of difference which could be a major impact on migration are discussed in greater detail later in this section.

TABLE B-1: NOTABLE DIFFERENCES BETWEEN PIC18F4520 AND PIC18FXXXX FAMILIES

| Characteristic | PIC18FXXXX Family | PIC18F4520 Family |
|-------------------------------------|---|---|
| Operating Frequency | 40 MHz @ 2.15V | 40 MHz @ 4.2V |
| Supply Voltage | 2.0V-3.6V | 2.0V-5.5V |
| Operating Current | Low | Lower |
| Program Memory Endurance | 1,000 write/erase cycles (typical) | 100,000 write/erase cycles (typical) |
| I/O Sink/Source at 25 mA | PORTB and PORTC only | All ports |
| Input Voltage Tolerance on I/O pins | 5.5V on digital only pins | VDD on all I/O pins |
| I/O | 32 | 36 |
| Pull-ups | PORTB | PORTB |
| Oscillator Options | Limited options (EC, HS, fixed 32 kHz INTRC) | More options (EC, HS, XT, LP, RC, PLL, flexible INTRC) |
| Program Memory Retention | 10 years (minimum) | 40 years (minimum) |
| Programming Time (Normalized) | 156 μ s/byte (10 ms/64-byte block) | 15.6 μ s/byte (1 ms/64-byte block) |
| Programming Entry | Low Voltage, Key Sequence | VPP and LVP |
| Code Protection | Single block, all or nothing | Multiple code protection blocks |
| Configuration Words | Stored in last 4 words of Program Memory space | Stored in Configuration Space, starting at 300000h |
| Start-up Time from Sleep | 200 μ s (typical) | 10 μ s (typical) |
| Power-up Timer | Always on | Configurable |
| Data EEPROM | Not available | Available |
| BOR | Simple BOR ⁽¹⁾ | Programmable BOR |
| LVD | Not available | Available |
| A/D Calibration | Required | Not required |
| In-Circuit Emulation | Not available | Available |
| TMR3 | Not available | Available |
| Second MSSP | Available ⁽²⁾ | Not available |

Note 1: BOR is not available on PIC18LFXXJ10 devices.

2: Available on 40/44-pin devices only.

PIC18F45J10 FAMILY

B.1 Power Requirement Differences

The most significant difference between the PIC18F45J10 family and PIC18F4520 device families is the power requirements. PIC18F45J10 family devices are designed on a smaller process; this results in lower maximum voltage and higher leakage current.

The operating voltage range for PIC18F45J10 family devices is 2.0V to 3.6V. One of the VDD pins is separated for the core logic supply (VDDCORE). This pin has specific voltage and capacitor requirements as described in **Section 23.0 “Electrical Characteristics”**.

The current specifications for PIC18F45J10 family devices are yet to be determined.

B.2 Pin Differences

There are several differences in the pinouts between the PIC18F45J10 family and the PIC18F4520 families:

- Input voltage tolerance
- Output current capabilities
- Available I/O

Pins on the PIC18F45J10 family that have digital only input capability will tolerate voltages up to 5.5V and are thus tolerant to voltages above VDD. Table 9-1 in **Section 9.0 “I/O Ports”** contains the complete list.

In addition to input differences, there are output differences as well. Not all I/O pins can source or sink equal levels of current. Only PORTB and PORTC support the 25 mA source/sink capability that is supported by all output pins on the PIC18F4520. Table 9-2 in **Section 9.0 “I/O Ports”** contains the complete list of output capabilities.

There are additional differences in how some pin functions are implemented on PIC18F45J10 family devices. First, the OSC1/OSC2 oscillator pins are strictly dedicated to the external oscillator function; there is no option to re-allocate these pins to I/O (RA6 or RA7) as on PIC18F4520 devices. Second, the MCLR pin is dedicated only to MCLR and cannot be configured as an input (RE3). Finally, RA4 does not exist on PIC18F45J10 family devices.

All of these pin differences (including power pin differences) should be accounted for when making a conversion between PIC18F4520 and PIC18F45J10 family devices.

B.3 Oscillator Differences

PIC18F4520 family devices have a greater range of oscillator options than PIC18F45J10 family devices. The latter family is limited primarily to operating modes that support HS and EC oscillators.

In addition, the PIC18F45J10 family has an internal RC oscillator with only a fixed 32 kHz output. The higher frequency RC modes of the PIC18F4520 family are not available.

B.4 Peripherals

The PIC18F45J10 family is able to operate at 40 MHz down to 2.15 volts unlike the PIC18F4520 family where 40 MHz operation is limited to 4.2 +V applications.

Peripherals must also be considered when making a conversion between the PIC18F45J10 family and the PIC18F4520 families:

- **Data EEPROM:** PIC18F45J10 family devices do not have this module.
- **BOR:** PIC18F45J10 family devices do not have a programmable BOR. Simple brown-out capability is provided through the use of the internal voltage regulator (not available in PIC18LFXXJ10 devices).
- **LVD:** PIC18F45J10 family devices do not have this module.
- Timer3 (TMR3) has been removed from the PIC18F45J10 family.
- The T0CKI/C1OUT pins have been moved from RA4 to RB5.
- The 40/44-pin devices in the PIC18F45J10 family have a second MSSP module available on pins RD0:RD3

PIC18F45J10 FAMILY

INDEX

A

| | | | |
|--|----------|---|-----|
| A/D | 209 | BF Status Flag | 176 |
| A/D Converter Interrupt, Configuring | 213 | Block Diagrams | |
| Acquisition Requirements | 214 | A/D | 212 |
| ADCAL Bit | 216 | Analog Input Model | 213 |
| ADCON0 Register | 209 | Baud Rate Generator | 172 |
| ADCON1 Register | 209 | Capture Mode Operation | 125 |
| ADCON2 Register | 209 | Comparator Analog Input Model | 223 |
| ADRESH Register | 209, 212 | Comparator I/O Operating Modes | 220 |
| ADRESL Register | 209 | Comparator Output | 222 |
| Analog Port Pins, Configuring | 216 | Comparator Voltage Reference | 226 |
| Associated Registers | 218 | Comparator Voltage Reference Output | |
| Calculating the Minimum Required | | Buffer Example | 227 |
| Acquisition Time | 214 | Compare Mode Operation | 126 |
| Calibration | 216 | Device Clock | 26 |
| Configuring the Module | 213 | Enhanced PWM | 133 |
| Conversion Clock (TAD) | 215 | EUSART Receive | 199 |
| Conversion Status (GO/DONE Bit) | 212 | EUSART Transmit | 197 |
| Conversions | 217 | External Power-on Reset Circuit | |
| Converter Characteristics | 328 | (Slow VDD Power-up) | 39 |
| Operation in Power-Managed Modes | 216 | Fail-Safe Clock Monitor | 238 |
| Selecting and Configuring Acquisition Time | 215 | Generic I/O Port Operation | 93 |
| Special Event Trigger (CCP) | 218 | Interrupt Logic | 80 |
| Special Event Trigger (ECCP) | 132 | MSSP (I ² C Master Mode) | 170 |
| Use of the CCP2 Trigger | 218 | MSSP (I ² C Mode) | 155 |
| Absolute Maximum Ratings | 297 | MSSP (SPI Mode) | 145 |
| AC (Timing) Characteristics | 311 | On-Chip Reset Circuit | 37 |
| Load Conditions for Device | | PIC18F24J10/25J10 | 10 |
| Timing Specifications | 312 | PIC18F44J10/45J10 | 11 |
| Parameter Symbolology | 311 | PLL | 25 |
| Temperature and Voltage Specifications | 312 | PORTD and PORTE (Parallel Slave Port) | 109 |
| Timing Conditions | 312 | PWM Operation (Simplified) | 128 |
| Access Bank | | Reads from Flash Program Memory | 71 |
| Mapping with Indexed Literal Offset Mode | 65 | Single Comparator | 221 |
| ACKSTAT | 176 | Table Read Operation | 67 |
| ACKSTAT Status Flag | 176 | Table Write Operation | 68 |
| ADCAL Bit | 216 | Table Writes to Flash Program Memory | 73 |
| ADCON0 Register | 209 | Timer0 in 16-Bit Mode | 112 |
| GO/DONE Bit | 212 | Timer0 in 8-Bit Mode | 112 |
| ADCON1 Register | 209 | Timer1 | 116 |
| ADCON2 Register | 209 | Timer1 (16-Bit Read/Write Mode) | 116 |
| ADDFSR | 284 | Timer2 | 122 |
| ADDLW | 247 | Watchdog Timer | 235 |
| ADDLWLNK | 284 | BN | 250 |
| ADDWF | 247 | BNC | 251 |
| ADDWFC | 248 | BNN | 251 |
| ADRESH Register | 209 | BNOV | 252 |
| ADRESL Register | 209, 212 | BNZ | 252 |
| Analog-to-Digital Converter. <i>See</i> A/D. | | BOR. <i>See</i> Brown-out Reset. | |
| ANDLW | 248 | BOV | 255 |
| ANDWF | 249 | BRA | 253 |
| Assembler | | Break Character (12-Bit) Transmit and Receive | 202 |
| MPASM Assembler | 291 | BRG. <i>See</i> Baud Rate Generator. | |
| Auto-Wake-up on Sync Break Character | 200 | Brown-out Reset (BOR) | 39 |
| | | and On-Chip Voltage Regulator | 236 |
| | | Disabling in Sleep Mode | 39 |
| | | BSF | 253 |
| | | BTFSC | 254 |
| | | BTFSS | 254 |
| | | BTG | 255 |
| | | BZ | 256 |

PIC18F45J10 FAMILY

C

| | |
|--|-----|
| C Compilers | |
| MPLAB C17 | 292 |
| MPLAB C18 | 292 |
| MPLAB C30 | 292 |
| Calibration (A/D Converter) | 216 |
| CALL | 256 |
| CALLW | 285 |
| Capture (CCP Module) | 125 |
| Associated Registers | 127 |
| CCP Pin Configuration | 125 |
| CCPRxH:CCPRxL Registers | 125 |
| Prescaler | 125 |
| Software Interrupt | 125 |
| Capture (ECCP Module) | 132 |
| Capture/Compare/PWM (CCP) | 123 |
| Capture Mode. <i>See</i> Capture. | |
| CCP Modules and Timer Resources | 124 |
| CCPRxH Register | 124 |
| CCPRxL Register | 124 |
| Compare Mode. <i>See</i> Compare. | |
| Interactions Between ECCP1/CCP1 and CCP2 for Timer Resources | 124 |
| Module Configuration | 124 |
| Clock Sources | 26 |
| Default System Clock on Reset | 27 |
| Selection Using OSCCON Register | 27 |
| CLRF | 257 |
| CLRWDT | 257 |
| Code Examples | |
| 16 x 16 Signed Multiply Routine | 78 |
| 16 x 16 Unsigned Multiply Routine | 78 |
| 8 x 8 Signed Multiply Routine | 77 |
| 8 x 8 Unsigned Multiply Routine | 77 |
| Changing Between Capture Prescalers | 125 |
| Computed GOTO Using an Offset Value | 50 |
| Erasing a Flash Program Memory Row | 72 |
| Fast Register Stack | 50 |
| How to Clear RAM (Bank 1) Using Indirect Addressing | 61 |
| Implementing a Real-Time Clock Using a Timer1 Interrupt Service | 119 |
| Initializing PORTA | 94 |
| Initializing PORTB | 97 |
| Initializing PORTC | 100 |
| Initializing PORTD | 103 |
| Initializing PORTE | 106 |
| Loading the SSP1BUF (SSP1SR) Register | 148 |
| Reading a Flash Program Memory Word | 71 |
| Saving STATUS, WREG and BSR Registers in RAM | 91 |
| Writing to Flash Program Memory | 74 |
| Code Protection | 229 |
| COMF | 258 |

| | |
|--|----------|
| Comparator | 219 |
| Analog Input Connection Considerations | 223 |
| Associated Registers | 223 |
| Configuration | 220 |
| Effects of a Reset | 222 |
| Interrupts | 222 |
| Operation | 221 |
| Operation During Sleep | 222 |
| Outputs | 221 |
| Reference | 221 |
| External Signal | 221 |
| Internal Signal | 221 |
| Response Time | 221 |
| Comparator Specifications | 310 |
| Comparator Voltage Reference | 225 |
| Accuracy and Error | 226 |
| Associated Registers | 227 |
| Configuring | 225 |
| Connection Considerations | 226 |
| Effects of a Reset | 226 |
| Operation During Sleep | 226 |
| Compare (CCP Module) | 126 |
| Associated Registers | 127 |
| CCPRx Register | 126 |
| Pin Configuration | 126 |
| Software Interrupt | 126 |
| Special Event Trigger | 126, 218 |
| Timer1 Mode Selection | 126 |
| Compare (ECCP Module) | 132 |
| Special Event Trigger | 132 |
| Computed GOTO | 50 |
| Configuration Bits | 229 |
| Configuration Register Protection | 240 |
| Context Saving During Interrupts | 91 |
| CPFSEQ | 258 |
| CPFSGT | 259 |
| CPFSLT | 259 |
| Crystal Oscillator/Ceramic Resonator | 23 |
| Customer Change Notification Service | 355 |
| Customer Notification Service | 355 |
| Customer Support | 355 |

D

| | |
|---|-----|
| Data Addressing Modes | 61 |
| Comparing Addressing Modes with the Extended Instruction Set Enabled | 64 |
| Direct | 61 |
| Indexed Literal Offset | 63 |
| Instructions Affected | 63 |
| Indirect | 61 |
| Inherent and Literal | 61 |
| Data Memory | 53 |
| Access Bank | 55 |
| and the Extended Instruction Set | 63 |
| Bank Select Register (BSR) | 53 |
| General Purpose Registers | 55 |
| Map for PIC18F24J10/44J10 | 54 |
| Special Function Registers | 56 |
| DAW | 260 |
| DC and AC Characteristics | |
| Graphs and Tables | 331 |

PIC18F45J10 FAMILY

| | | | |
|---|-----|---|----------|
| DC Characteristics | 307 | Synchronous Master Mode | 203 |
| Power-Down and Supply Current | 300 | Associated Registers, Receive | 205 |
| Supply Voltage | 299 | Associated Registers, Transmit | 204 |
| DCFSNZ | 261 | Reception | 205 |
| DECF | 260 | Transmission | 203 |
| DECFSZ | 261 | Synchronous Slave Mode | 206 |
| Default System Clock | 27 | Associated Registers, Receive | 207 |
| Demonstration Boards | | Associated Registers, Transmit | 206 |
| PICDEM 1 | 294 | Reception | 207 |
| PICDEM 17 | 295 | Transmission | 206 |
| PICDEM 18R | 295 | Evaluation and Programming Tools | 295 |
| PICDEM 2 Plus | 294 | Extended Instruction Set | |
| PICDEM 3 | 294 | ADDFSR | 284 |
| PICDEM 4 | 294 | ADDULNK | 284 |
| PICDEM LIN | 295 | and Using MPLAB Tools | 290 |
| PICDEM USB | 295 | CALLW | 285 |
| PICDEM.net Internet/Ethernet | 294 | Considerations for Use | 288 |
| Development Support | 291 | MOVSF | 285 |
| Device Overview | 7 | MOVSS | 286 |
| Details on Individual Family Members | 8 | PUSHL | 286 |
| Features (table) | 9 | SUBFSR | 287 |
| New Core Features | 7 | SUBULNK | 287 |
| Other Special Features | 8 | Syntax | 283 |
| Direct Addressing | 62 | External Clock Input (EC Modes) | 24 |
| E | | F | |
| Effect on Standard PIC Instructions | 288 | Fail-Safe Clock Monitor | 229, 238 |
| Effects of Power-Managed Modes on | | Interrupts in Power-Managed Modes | 239 |
| Various Clock Sources | 28 | POR or Wake-up from Sleep | 239 |
| Electrical Characteristics | 297 | WDT During Oscillator Failure | 238 |
| Enhanced Capture/Compare/PWM (ECCP) | 131 | Fast Register Stack | 50 |
| Associated Registers | 144 | Firmware Instructions | 241 |
| Capture and Compare Modes | 132 | Flash Configuration Words | 229 |
| Capture Mode. <i>See</i> Capture (ECCP Module). | | Flash Program Memory | 67 |
| Outputs and Configuration | 132 | Associated Registers | 75 |
| Pin Configurations for ECCP1 Modes | 132 | Control Registers | 68 |
| PWM Mode. <i>See</i> PWM (ECCP Module). | | EECON1 and EECON2 | 68 |
| Standard PWM Mode | 132 | TABLAT (Table Latch) | 70 |
| Timer Resources | 132 | TBLPTR (Table Pointer) | 70 |
| Enhanced PWM Mode. <i>See</i> PWM (ECCP Module). | | Erase Sequence | 72 |
| Enhanced Universal Synchronous Asynchronous | | Erasing | 72 |
| Receiver Transmitter (EUSART). <i>See</i> EUSART. | | Operation During Code-Protect | 75 |
| Equations | | Reading | 71 |
| A/D Acquisition Time | 214 | Table Pointer | |
| A/D Minimum Charging Time | 214 | Boundaries Based on Operation | 70 |
| Errata | 5 | Table Pointer Boundaries | 70 |
| EUSART | | Table Reads and Table Writes | 67 |
| Asynchronous Mode | 197 | Write Sequence | 73 |
| 12-Bit Break Transmit and Receive | 202 | Writing To | 73 |
| Associated Registers, Receive | 200 | Protection Against Spurious Writes | 75 |
| Associated Registers, Transmit | 198 | Unexpected Termination | 75 |
| Auto-Wake-up on Sync Break | 200 | Write Verify | 75 |
| Receiver | 199 | FSCM. <i>See</i> Fail-Safe Clock Monitor. | |
| Setting Up 9-Bit Mode with | | G | |
| Address Detect | 199 | GOTO | 262 |
| Transmitter | 197 | H | |
| Baud Rate Generator | | Hardware Multiplier | 77 |
| Operation in Power-Managed Mode | 191 | Introduction | 77 |
| Baud Rate Generator (BRG) | 191 | Operation | 77 |
| Associated Registers | 192 | Performance Comparison | 77 |
| Auto-Baud Rate Detect | 195 | | |
| Baud Rate Error, Calculating | 192 | | |
| Baud Rates, Asynchronous Modes | 193 | | |
| High Baud Rate Select (BRGH Bit) | 191 | | |
| Sampling | 191 | | |

PIC18F45J10 FAMILY

| | |
|--|----------|
| I | |
| I/O Ports | 93 |
| I ² C Mode (MSSP) | |
| Acknowledge Sequence Timing | 179 |
| Associated Registers | 185 |
| Baud Rate Generator | 172 |
| Bus Collision | |
| During a Repeated Start Condition | 183 |
| During a Stop Condition | 184 |
| Clock Arbitration | 173 |
| Clock Stretching | 165 |
| 10-Bit Slave Receive Mode (SEN = 1) | 165 |
| 10-Bit Slave Transmit Mode | 165 |
| 7-Bit Slave Receive Mode (SEN = 1) | 165 |
| 7-Bit Slave Transmit Mode | 165 |
| Clock Synchronization and the CKP Bit | 166 |
| Effects of a Reset | 180 |
| General Call Address Support | 169 |
| I ² C Clock Rate w/BRG | 172 |
| Master Mode | 170 |
| Baud Rate Generator | 172 |
| Operation | 171 |
| Reception | 176 |
| Repeated Start Condition Timing | 175 |
| Start Condition Timing | 174 |
| Transmission | 176 |
| Multi-Master Communication, Bus Collision and Arbitration | 180 |
| Multi-Master Mode | 180 |
| Operation | 159 |
| Read/Write Bit Information (R/W Bit) | 159, 160 |
| Registers | 155 |
| Serial Clock (SCKx/SCLx) | 160 |
| Slave Mode | 159 |
| Addressing | 159 |
| Reception | 160 |
| Transmission | 160 |
| Sleep Operation | 180 |
| Stop Condition Timing | 179 |
| INCF | 262 |
| INCFSZ | 263 |
| In-Circuit Debugger | 240 |
| In-Circuit Serial Programming (ICSP) | 229, 240 |
| Indexed Literal Offset Addressing and Standard PIC18 Instructions | 288 |
| Indexed Literal Offset Mode | 288 |
| Indirect Addressing | 62 |
| INFSNZ | 263 |
| Initialization Conditions for All Registers | 43–46 |
| Instruction Cycle | 51 |
| Clocking Scheme | 51 |
| Instruction Flow/Pipelining | 51 |
| Instruction Set | 241 |
| ADDLW | 247 |
| ADDWF | 247 |
| ADDWF (Indexed Literal Offset Mode) | 289 |
| ADDWFC | 248 |
| ANDLW | 248 |
| ANDWF | 249 |
| BC | 249 |
| BCF | 250 |
| BN | 250 |
| BNC | 251 |
| BNN | 251 |
| BNOV | 252 |
| BNZ | 252 |
| BOV | 255 |
| BRA | 253 |
| BSF | 253 |
| BSF (Indexed Literal Offset Mode) | 289 |
| BTFSC | 254 |
| BTFSS | 254 |
| BTG | 255 |
| BZ | 256 |
| CALL | 256 |
| CLRF | 257 |
| CLRWDT | 257 |
| COMF | 258 |
| CPFSEQ | 258 |
| CPFSGT | 259 |
| CPFSLT | 259 |
| DAW | 260 |
| DCFSNZ | 261 |
| DECf | 260 |
| DECFSZ | 261 |
| Extended Instruction Set | 283 |
| General Format | 243 |
| GOTO | 262 |
| INCF | 262 |
| INCFSZ | 263 |
| INFSNZ | 263 |
| IORLW | 264 |
| IORWF | 264 |
| LFSR | 265 |
| MOVF | 265 |
| MOVFF | 266 |
| MOVLB | 266 |
| MOVLW | 267 |
| MOVWF | 267 |
| MULLW | 268 |
| MULWF | 268 |
| NEGF | 269 |
| NOP | 269 |
| Opcode Field Descriptions | 242 |
| POP | 270 |
| PUSH | 270 |
| RCALL | 271 |
| RESET | 271 |
| RETFIE | 272 |
| RETLW | 272 |
| RETURN | 273 |
| RLCF | 273 |
| RLNCF | 274 |
| RRCF | 274 |
| RRNCF | 275 |
| SETF | 275 |
| SETF (Indexed Literal Offset Mode) | 289 |
| SLEEP | 276 |
| Standard Instructions | 241 |
| SUBFWB | 276 |
| SUBLW | 277 |
| SUBWF | 277 |
| SUBWFB | 278 |
| SWAPF | 278 |
| TBLRD | 279 |
| TBLWT | 280 |
| TSTFSZ | 281 |
| XORLW | 281 |
| XORWF | 282 |

PIC18F45J10 FAMILY

| | |
|---|----------|
| INTCON Registers | 81 |
| Inter-Integrated Circuit. <i>See</i> I ² C Mode. | |
| Internal Oscillator Block | 26 |
| Internal RC Oscillator | |
| Use with WDT | 235 |
| Internet Address | 355 |
| Interrupt Sources | 229 |
| A/D Conversion Complete | 213 |
| Capture Complete (CCP) | 125 |
| Compare Complete (CCP) | 126 |
| Interrupt-on-Change (RB7:RB4) | 97 |
| INTn Pin | 91 |
| PORTB, Interrupt-on-Change | 91 |
| TMR0 | 91 |
| TMR0 Overflow | 113 |
| TMR1 Overflow | 115 |
| TMR2-to-PR2 Match (PWM) | 128, 133 |
| Interrupts | 79 |
| Interrupts, Flag Bits | |
| Interrupt-on-Change (RB7:RB4) | |
| Flag (RBIF Bit) | 97 |
| INTOSC, INTRC. <i>See</i> Internal Oscillator Block. | |
| IORLW | 264 |
| IORWF | 264 |
| IPR Registers | 88 |
| L | |
| LFSR | 265 |
| M | |
| Master Clear (MCLR) | 39 |
| Master Synchronous Serial Port (MSSP). <i>See</i> MSSP. | |
| Memory Organization | 47 |
| Data Memory | 53 |
| Program Memory | 47 |
| Memory Programming Requirements | 309 |
| Microchip Internet Web Site | 355 |
| MOVF | 265 |
| MOVFF | 266 |
| MOVLB | 266 |
| MOVLW | 267 |
| MOVSF | 285 |
| MOVSS | 286 |
| MOVWF | 267 |
| MPLAB ASM30 Assembler, Linker, Librarian | 292 |
| MPLAB ICD 2 In-Circuit Debugger | 293 |
| MPLAB ICE 2000 High-Performance | |
| Universal In-Circuit Emulator | 293 |
| MPLAB ICE 4000 High-Performance | |
| Universal In-Circuit Emulator | 293 |
| MPLAB Integrated Development | |
| Environment Software | 291 |
| MPLAB PM3 Device Programmer | 293 |
| MPLINK Object Linker/MPLIB Object Librarian | 292 |
| MSSP | |
| ACK Pulse | 159, 160 |
| Control Registers (general) | 145 |
| Module Overview | 145 |
| SPI Master/Slave Connection | 149 |
| SSPxBUF Register | 150 |
| SSPxSR Register | 150 |
| MULLW | 268 |
| MULWF | 268 |

| | |
|--|----------|
| N | |
| NEGF | 269 |
| NOP | 269 |
| Notable Differences Between PIC18F4520 | |
| and PIC18F45J10 Families | 343 |
| Oscillator Options | 344 |
| Peripherals | 344 |
| Pinouts | 344 |
| Power Requirements | 344 |
| O | |
| Oscillator Configuration | 23 |
| EC | 23 |
| ECPLL | 23 |
| HS | 23 |
| HS Modes | 23 |
| HSPLL | 23 |
| Internal Oscillator Block | 26 |
| INTRC | 23 |
| Oscillator Selection | 229 |
| Oscillator Start-up Timer (OST) | 29 |
| Oscillator Switching | 26 |
| Oscillator Transitions | 27 |
| Oscillator, Timer1 | 115 |
| P | |
| Packaging Information | 333 |
| Marking | 333 |
| Parallel Slave Port (PSP) | 103, 109 |
| Associated Registers | 110 |
| CS (Chip Select) | 109 |
| PORTD | 109 |
| RD (Read Input) | 109 |
| Select (PSPMODE Bit) | 103, 109 |
| WR (Write Input) | 109 |
| PICKIT 1 Flash Starter Kit | 295 |
| PICSTART Plus Development Programmer | 294 |
| PIE Registers | 86 |
| Pin Functions | |
| MCLR | 12, 16 |
| OSC1/CLKI | 12, 16 |
| OSC2/CLKO | 12, 16 |
| RA0/AN0 | 13, 17 |
| RA1/AN1 | 13, 17 |
| RA2/AN2/VREF-/CVREF | 13, 17 |
| RA3/AN3/VREF+ | 13, 17 |
| RA5/AN4/SS1/C2OUT | 13, 17 |
| RB0/INT0/FLT0/AN12 | 14, 18 |
| RB1/INT1/AN10 | 14, 18 |
| RB2/INT2/AN8 | 14, 18 |
| RB3/AN9/CCP2 | 14, 18 |
| RB4/KBI0/AN11 | 14, 18 |
| RB5/KBI1/C1OUT | 18 |
| RB5/KBI1/T0CKI/C1OUT | 14 |
| RB6/KBI2/PGC | 14, 18 |
| RB7/KBI3/PGD | 14, 18 |
| RC0/T1OSO/T1CKI | 15, 19 |
| RC1/T1OSI/CCP2 | 15, 19 |
| RC2/CCP1 | 15 |
| RC2/CCP1/P1A | 19 |
| RC3/SCK1/SCL1 | 15, 19 |
| RC4/SDI1/SDA1 | 15, 19 |
| RC5/SDO1 | 15, 19 |

PIC18F45J10 FAMILY

| | | | |
|--|--------|---|----------|
| RC6/TX/CK | 15, 19 | Clock Transitions and Status Indicators | 32 |
| RC7/RX/DT | 15, 19 | Entering | 31 |
| RD0/PSP0/SCK2/SCL2 | 20 | Exiting Idle and Sleep Modes | 36 |
| RD1/PSP1/SDI2/SDA2 | 20 | by Reset | 36 |
| RD2/PSP2/SDO2 | 20 | by WDT Time-out | 36 |
| RD3/PSP3/SS2 | 20 | Without an Oscillator | |
| RD4/PSP4 | 20 | Start-up Delay | 36 |
| RD5/PSP5/P1B | 20 | Idle Modes | 34 |
| RD6/PSP6/P1C | 20 | PRI_IDLE | 35 |
| RD7/PSP7/P1D | 20 | RC_IDLE | 36 |
| RE0/RD/AN5 | 21 | SEC_IDLE | 35 |
| RE1/WR/AN6 | 21 | Run Modes | 32 |
| RE2/CS/AN7 | 21 | PRI_RUN | 32 |
| VDD | 15, 21 | RC_RUN | 33 |
| VDDCORE/VCAP | 15, 21 | SEC_RUN | 32 |
| VSS | 15, 21 | Selecting | 31 |
| Pinout I/O Descriptions | | Sleep Mode | 34 |
| PIC18F24J10/25J10 | 12 | Summary (table) | 31 |
| PIC18F44J10/45J10 | 16 | Power-on Reset (POR) | 39 |
| PIR Registers | 84 | Power-up Timer (PWRT) | 40 |
| PLL Frequency Multiplier | 25 | Time-out Sequence | 40 |
| ECPLL Oscillator Mode | 25 | Power-up Delays | 29 |
| HSPLL Oscillator Mode | 25 | Power-up Timer (PWRT) | 29, 40 |
| POP | 270 | Prescaler | |
| POR. See Power-on Reset. | | Timer2 | 134 |
| PORTA | | Prescaler, Timer0 | 113 |
| Associated Registers | 96 | Prescaler, Timer2 | 129 |
| LATA Register | 94 | PRI_IDLE Mode | 35 |
| PORTA Register | 94 | PRI_RUN Mode | 32 |
| TRISA Register | 94 | PRO MATE II Universal Device | |
| PORTB | | Programmer | 293 |
| Associated Registers | 99 | Program Counter | 48 |
| LATB Register | 97 | PCL, PCH and PCU Registers | 48 |
| PORTB Register | 97 | PCLATH and PCLATU Registers | 48 |
| RB7:RB4 Interrupt-on-Change | | Program Memory | |
| Flag (RBIF Bit) | 97 | and Extended Instruction Set | 65 |
| TRISB Register | 97 | Instructions | 52 |
| PORTC | | Two-Word | 52 |
| Associated Registers | 102 | Interrupt Vector | 47 |
| LATC Register | 100 | Look-up Tables | 50 |
| PORTC Register | 100 | Map and Stack (diagram) | 47 |
| RC3/SCK1/SCL1 Pin | 160 | Reset Vector | 47 |
| TRISC Register | 100 | Program Verification and | |
| PORTD | | Code Protection | 240 |
| Associated Registers | 105 | Programming, Device Instructions | 241 |
| LATD Register | 103 | PSP. See Parallel Slave Port. | |
| Parallel Slave Port (PSP) Function | 103 | Pulse-Width Modulation. See PWM (CCP Module) | |
| PORTD Register | 103 | and PWM (ECCP Module). | |
| TRISD Register | 103 | PUSH | 270 |
| PORTE | | PUSH and POP Instructions | 49 |
| Associated Registers | 108 | PUSHL | 286 |
| LATE Register | 106 | PWM (CCP Module) | |
| PORTE Register | 106 | Associated Registers | 130 |
| PSP Mode Select (PSPMODE Bit) | 103 | Auto-Shutdown (CCP1 Only) | 129 |
| TRISE Register | 106 | CCPR1H:CCPR1L Registers | 133 |
| Power-Managed Modes | 31 | Duty Cycle | 128, 134 |
| and A/D Operation | 216 | Example Frequencies/Resolutions | 129, 134 |
| and EUSART Operation | 191 | Period | 128, 133 |
| and Multiple Sleep Commands | 32 | Setup for Operation | 129 |
| and PWM Operation | 143 | TMR2-to-PR2 Match | 128, 133 |
| and SPI Operation | 153 | | |

PIC18F45J10 FAMILY

| | | | |
|--|----------|---|--------|
| PWM (ECCP Module) | 133 | OSCTUNE (PLL Control) | 25 |
| Direction Change in Full-Bridge | | PIE1 (Peripheral Interrupt Enable 1) | 86 |
| Output Mode | 138 | PIE2 (Peripheral Interrupt Enable 2) | 87 |
| Effects of a Reset | 143 | PIE3 (Peripheral Interrupt Enable 3) | 87 |
| Enhanced PWM Auto-Shutdown | 140 | PIR1 (Peripheral Interrupt Request | |
| Full-Bridge Application Example | 138 | (Flag) 1) | 84 |
| Full-Bridge Mode | 137 | PIR2 (Peripheral Interrupt Request | |
| Half-Bridge Mode | 136 | (Flag) 2) | 85 |
| Half-Bridge Output Mode | | PIR3 (Peripheral Interrupt Request | |
| Applications Example | 136 | (Flag) 3) | 85 |
| Operation in Power-Managed Modes | 143 | RCON (Reset Control) | 38, 90 |
| Operation with Fail-Safe Clock Monitor | 143 | RCSTA (Receive Status and Control) | 189 |
| Output Configurations | 134 | SSPxCON1 (MSSPx Control 1, I ² C Mode) | 157 |
| Output Relationships (Active-High) | 135 | SSPxCON1 (MSSPx Control 1, SPI Mode) | 147 |
| Output Relationships (Active-Low) | 135 | SSPxCON2 (MSSPx Control 2, I ² C Mode) | 158 |
| Programmable Dead-Band Delay | 140 | SSPxSTAT (MSSPx Status, I ² C Mode) | 156 |
| Setup for PWM Operation | 143 | SSPxSTAT (MSSPx Status, SPI Mode) | 146 |
| Start-up Considerations | 142 | STATUS | 60 |
| Q | | STKPTR (Stack Pointer) | 49 |
| Q Clock | 129, 134 | T0CON (Timer0 Control) | 111 |
| R | | T1CON (Timer1 Control) | 115 |
| RAM. <i>See</i> Data Memory. | | T2CON (Timer2 Control) | 121 |
| RBIF Bit | 97 | TRISE (PORTE/PSP Control) | 107 |
| RC_IDLE Mode | 36 | TXSTA (Transmit Status and Control) | 188 |
| RC_RUN Mode | 33 | WDTCON (Watchdog Timer Control) | 235 |
| RCALL | 271 | RESET | 271 |
| RCON Register | | Reset | 37 |
| Bit Status During Initialization | 42 | MCLR Reset, During Power-Managed Modes | 37 |
| Reader Response | 356 | MCLR Reset, Normal Operation | 37 |
| Register File | 55 | Power-on Reset (POR) | 37 |
| Register File Summary | 57–59 | Programmable Brown-out Reset (BOR) | 37 |
| Registers | | Reset Instruction | 37 |
| ADCON0 (A/D Control 0) | 209 | Stack Full Reset | 37 |
| ADCON1 (A/D Control 1) | 210 | Stack Underflow Reset | 37 |
| ADCON2 (A/D Control 2) | 211 | Watchdog Timer (WDT) Reset | 37 |
| BAUDCON (Baud Rate Control) | 190 | Resets | 229 |
| CCP1CON (Enhanced | | Brown-out Reset (BOR) | 229 |
| Capture/Compare/PWM Control 1) | 131 | Oscillator Start-up Timer (OST) | 229 |
| CCPxCON (Standard | | Power-on Reset (POR) | 229 |
| Capture/Compare/PWM Control) | 123 | Power-up Timer (PWRT) | 229 |
| CMCON (Comparator Control) | 219 | RETFIE | 272 |
| CONFIG1H (Configuration 1 High) | 231 | RETLW | 272 |
| CONFIG1L (Configuration 1 Low) | 231 | RETURN | 273 |
| CONFIG2H (Configuration 2 High) | 232 | Return Address Stack | 48 |
| CONFIG2L (Configuration 2 Low) | 232 | Return Stack Pointer (STKPTR) | 49 |
| CONFIG3H (Configuration 3 High) | 233 | Revision History | 343 |
| CONFIG3L (Configuration 3 Low) | 233 | RLCF | 273 |
| CVRCON (Comparator Voltage | | RLNCF | 274 |
| Reference Control) | 225 | RRCF | 274 |
| Device ID Register 1 | 234 | RRNCF | 275 |
| Device ID Register 2 | 234 | S | |
| ECCP1AS (ECCP | | SCKx | 145 |
| Auto-Shutdown Control) | 141 | SDIx | 145 |
| ECCP1DEL (PWM Dead-Band Delay) | 140 | SDOx | 145 |
| EECON1 (Data EEPROM Control 1) | 69 | SEC_IDLE Mode | 35 |
| INTCON (Interrupt Control) | 81 | SEC_RUN Mode | 32 |
| INTCON2 (Interrupt Control 2) | 82 | Serial Clock, SCKx | 145 |
| INTCON3 (Interrupt Control 3) | 83 | Serial Data In (SDIx) | 145 |
| IPR1 (Peripheral Interrupt Priority 1) | 88 | Serial Data Out (SDOx) | 145 |
| IPR2 (Peripheral Interrupt Priority 2) | 89 | Serial Peripheral Interface. <i>See</i> SPI Mode. | |
| IPR3 (Peripheral Interrupt Priority 3) | 89 | SETF | 275 |
| OSCCON (Oscillator Control) | 28 | Slave Select (SSx) | 145 |

PIC18F45J10 FAMILY

| | | | |
|--|----------|--|----------|
| SLEEP | 276 | Timer1 | 115 |
| Sleep | | 16-Bit Read/Write Mode | 117 |
| OSC1 and OSC2 Pin States | 29 | Associated Registers | 119 |
| Software Simulator (MPLAB SIM) | 292 | Interrupt | 118 |
| Software Simulator (MPLAB SIM30) | 292 | Operation | 116 |
| Special Event Trigger. <i>See</i> Compare (ECCP/CCP Modules). | | Oscillator | 115, 117 |
| Special Features of the CPU | 229 | Layout Considerations | 118 |
| Special Function Registers | 56 | Oscillator, as Secondary Clock | 26 |
| Map | 56 | Overflow Interrupt | 115 |
| SPI Mode (MSSP) | | Resetting, Using the ECCP/CCP | |
| Associated Registers | 154 | Special Event Trigger | 118 |
| Bus Mode Compatibility | 153 | Special Event Trigger (ECCP) | 132 |
| Clock Speed and Module | | TMR1H Register | 115 |
| Interactions | 153 | TMR1L Register | 115 |
| Effects of a Reset | 153 | Use as a Clock Source | 117 |
| Enabling SPI I/O | 149 | Use as a Real-Time Clock | 118 |
| Master Mode | 150 | Timer2 | 121 |
| Master/Slave Connection | 149 | Associated Registers | 122 |
| Operation | 148 | Interrupt | 122 |
| Operation in Power-Managed | | Operation | 121 |
| Modes | 153 | Output | 122 |
| Serial Clock | 145 | PR2 Register | 128, 133 |
| Serial Data In | 145 | TMR2-to-PR2 Match Interrupt | 128, 133 |
| Serial Data Out | 145 | Timing Diagrams | |
| Slave Mode | 151 | A/D Conversion | 328 |
| Slave Select | 145 | Acknowledge Sequence | 179 |
| Slave Select Synchronization | 151 | Asynchronous Reception | 200 |
| SPI Clock | 150 | Asynchronous Transmission | 198 |
| Typical Connection | 149 | Asynchronous Transmission | |
| SSPOV | 176 | (Back to Back) | 198 |
| SSPOV Status Flag | 176 | Automatic Baud Rate Calculation | 196 |
| SSPxSTAT Register | | Auto-Wake-up Bit (WUE) During | |
| R/W Bit | 159, 160 | Normal Operation | 201 |
| SSx | 145 | Auto-Wake-up Bit (WUE) During Sleep | 201 |
| Stack Full/Underflow Resets | 50 | Baud Rate Generator with Clock Arbitration | 173 |
| SUBFSR | 287 | BRG Overflow Sequence | 196 |
| SUBFWB | 276 | BRG Reset Due to SDAX Arbitration | |
| SUBLW | 277 | During Start Condition | 182 |
| SUBULNK | 287 | Brown-out Reset (BOR) | 316 |
| SUBWF | 277 | Bus Collision During a Repeated | |
| SUBWFB | 278 | Start Condition (Case 1) | 183 |
| SWAPF | 278 | Bus Collision During a Repeated | |
| T | | Start Condition (Case 2) | 183 |
| Table Pointer Operations (table) | 70 | Bus Collision During a Start Condition | |
| Table Reads/Table Writes | 50 | (SCLx = 0) | 182 |
| TBLRD | 279 | Bus Collision During a Stop Condition | |
| TBLWT | 280 | (Case 1) | 184 |
| Timer0 | 111 | Bus Collision During a Stop Condition | |
| Associated Registers | 113 | (Case 2) | 184 |
| Clock Source Select (T0CS Bit) | 112 | Bus Collision During Start Condition | |
| Operation | 112 | (SDAX Only) | 181 |
| Overflow Interrupt | 113 | Bus Collision for Transmit and | |
| Prescaler | 113 | Acknowledge | 180 |
| Prescaler Assignment (PSA Bit) | 113 | Capture/Compare/PWM | |
| Prescaler Select | | (Including ECCP Module) | 318 |
| (T0PS2:T0PS0 Bits) | 113 | CLKO and I/O | 315 |
| Prescaler. <i>See</i> Prescaler, Timer0. | | Clock Synchronization | 166 |
| Reads and Writes in 16-Bit Mode | 112 | Clock/Instruction Cycle | 51 |
| Source Edge Select (T0SE Bit) | 112 | EUSART Synchronous | |
| Switching Prescaler Assignment | 113 | Receive (Master/Slave) | 327 |
| | | EUSART Synchronous | |
| | | Transmission (Master/Slave) | 327 |

PIC18F45J10 FAMILY

| | | | |
|--|-----|---|--------------------|
| Example SPI Master Mode (CKE = 0) | 319 | Transition for Wake from Sleep | 34 |
| Example SPI Master Mode (CKE = 1) | 320 | Transition from RC_RUN Mode to | |
| Example SPI Slave Mode (CKE = 0) | 321 | PRI_RUN Mode | 33 |
| Example SPI Slave Mode (CKE = 1) | 322 | Transition to RC_RUN Mode | 33 |
| External Clock (All Modes Except PLL) | 313 | Timing Diagrams and Specifications | |
| Fail-Safe Clock Monitor | 239 | A/D Conversion Requirements | 329 |
| First Start Bit Timing | 174 | AC Characteristics | |
| Full-Bridge PWM Output | 137 | Internal RC Accuracy | 314 |
| Half-Bridge PWM Output | 136 | Capture/Compare/PWM Requirements | |
| I ² C Bus Data | 323 | (Including ECCP Module) | 318 |
| I ² C Bus Start/Stop Bits | 323 | CLKO and I/O Requirements | 315 |
| I ² C Master Mode (7 or | | EUSART Synchronous Receive | |
| 10-Bit Transmission) | 177 | Requirements | 327 |
| I ² C Master Mode (7-Bit Reception) | 178 | EUSART Synchronous Transmission | |
| I ² C Slave Mode | | Requirements | 327 |
| (10-Bit Reception, SEN = 0) | 163 | Example SPI Mode Requirements | |
| I ² C Slave Mode | | (Master Mode, CKE = 0) | 319 |
| (10-Bit Reception, SEN = 1) | 168 | Example SPI Mode Requirements | |
| I ² C Slave Mode (10-Bit Transmission) | 164 | (Master Mode, CKE = 1) | 320 |
| I ² C Slave Mode (7-bit Reception, SEN = 0) | 161 | Example SPI Mode Requirements | |
| I ² C Slave Mode (7-Bit Reception, SEN = 1) | 167 | (Slave Mode, CKE = 0) | 321 |
| I ² C Slave Mode (7-Bit Transmission) | 162 | Example SPI Slave Mode | |
| I ² C Slave Mode General Call Address | | Requirements (CKE = 1) | 322 |
| Sequence (7 or 10-Bit Address Mode) | 169 | External Clock Requirements | 313 |
| I ² C Stop Condition Receive or | | I ² C Bus Data Requirements (Slave Mode) | 324 |
| Transmit Mode | 179 | I ² C Bus Start/Stop Bits Requirements | |
| Master SSP I ² C Bus Data | 325 | (Slave Mode) | 323 |
| Master SSP I ² C Bus Start/Stop Bits | 325 | Master SSP I ² C Bus Data Requirements | 326 |
| Parallel Slave Port (PSP) Read | 110 | Master SSP I ² C Bus Start/Stop Bits | |
| Parallel Slave Port (PSP) Write | 110 | Requirements | 325 |
| PWM Auto-Shutdown (PRSEN = 0, | | Parallel Slave Port Requirements | 318 |
| Auto-Restart Disabled) | 142 | PLL Clock | 314 |
| PWM Auto-Shutdown (PRSEN = 1, | | Reset, Watchdog Timer, | |
| Auto-Restart Enabled) | 142 | Oscillator Start-up Timer, Power-up Timer | |
| PWM Direction Change | 139 | and Brown-out Reset Requirements | 316 |
| PWM Direction Change at | | Timer0 and Timer1 External | |
| Near 100% Duty Cycle | 139 | Clock Requirements | 317 |
| PWM Output | 128 | Top-of-Stack Access | 48 |
| Repeated Start Condition | 175 | TRISE Register | |
| Reset, Watchdog Timer (WDT), | | PSPMODE Bit | 103 |
| Oscillator Start-up Timer (OST) and | | TSTFSZ | 281 |
| Power-up Timer (PWRT) | 316 | Two-Speed Start-up | 229, 237 |
| Send Break Character Sequence | 202 | Two-Word Instructions | |
| Slave Synchronization | 151 | Example Cases | 52 |
| Slow Rise Time (MCLR Tied to VDD, | | TXSTA Register | |
| VDD Rise > TPWRT) | 41 | BRGH Bit | 191 |
| SPI Mode (Master Mode) | 150 | V | |
| SPI Mode (Slave Mode, CKE = 0) | 152 | Voltage Reference Specifications | 310 |
| SPI Mode (Slave Mode, CKE = 1) | 152 | Voltage Regulator (On-Chip) | 236 |
| Synchronous Reception | | W | |
| (Master Mode, SREN) | 205 | Watchdog Timer (WDT) | 229, 235 |
| Synchronous Transmission | 203 | Associated Registers | 235 |
| Synchronous Transmission (Through TXEN) | 204 | Control Register | 235 |
| Time-out Sequence on Power-up | | During Oscillator Failure | 238 |
| (MCLR Not Tied to VDD), Case 1 | 40 | Programming Considerations | 235 |
| Time-out Sequence on Power-up | | WCOL | 174, 175, 176, 179 |
| (MCLR Not Tied to VDD), Case 2 | 41 | WCOL Status Flag | 174, 175, 176, 179 |
| Time-out Sequence on Power-up | | WWW Address | 355 |
| (MCLR Tied to VDD, VDD Rise TPWRT) | 40 | WWW, On-Line Support | 5 |
| Timer0 and Timer1 External Clock | 317 | X | |
| Transition for Entry to Idle Mode | 35 | XORLW | 281 |
| Transition for Entry to SEC_RUN Mode | 32 | XORWF | 282 |
| Transition for Entry to Sleep Mode | 34 | | |
| Transition for Two-Speed Start-up (INTRC) | 237 | | |
| Transition for Wake from Idle to Run Mode | 35 | | |

PIC18F45J10 FAMILY

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: <http://support.microchip.com>

In addition, there is a Development Systems Information Line which lists the latest versions of Microchip's development systems software products. This line also provides information on how customers can receive currently available upgrade kits.

The Development Systems Information Line numbers are:

1-800-755-2345 – United States and most of Canada

1-480-792-7302 – Other International Locations

PIC18F45J10 FAMILY

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
RE: Reader Response
Total Pages Sent _____

From: Name _____
Company _____
Address _____
City / State / ZIP / Country _____
Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC18F45J10 family Literature Number: DS39682A

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?

PIC18F45J10 FAMILY

PIC18F45J10 FAMILY PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

| <u>PART NO.</u> | <u>X</u> | <u>/XX</u> | <u>XXX</u> |
|-------------------|--|------------|------------|
| Device | Temperature Range | Package | Pattern |
| Device | PIC18F24J10/25J10 ⁽¹⁾ , PIC18F44J10/45J10 ⁽¹⁾ , PIC18F24J10/25J10T ⁽²⁾ , PIC18F44J10/45J10T ⁽²⁾ ; VDD range 4.2V to 5.5V PIC18LF24J10/25J10 ⁽¹⁾ , PIC18LF44J10/45J10 ⁽¹⁾ , PIC18LF24J10/25J10T ⁽²⁾ , PIC18LF44J10/45J10T ⁽²⁾ ; VDD range 2.0V to 5.5V | | |
| Temperature Range | I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended) | | |
| Package | PT = TQFP (Thin Quad Flatpack) SO = SOIC SP = Skinny Plastic DIP P = PDIP ML = QFN | | |
| Pattern | QTP, SQTP, Code or Special Requirements (blank otherwise) | | |

Examples:

- a) PIC18LF45J10-I/P 301 = Industrial temp., PDIP package, Extended VDD limits, QTP pattern #301.
- b) PIC18LF24J10-I/SO = Industrial temp., SOIC package, Extended VDD limits.
- c) PIC18LF44J10-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 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:
<http://support.microchip.com>
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Qingdao
Tel: 86-532-502-7355
Fax: 86-532-502-7205

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

Japan - Kanagawa
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Taiwan - Hsinchu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

EUROPE

Austria - Weis
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark - Ballerup
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Massy
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Ismaning
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

England - Berkshire
Tel: 44-118-921-5869
Fax: 44-118-921-5820

03/01/05