



# **PIC18F46J11 Family Data Sheet**

28/44-Pin, Low-Power,  
High-Performance Microcontrollers  
with nanoWatt XLP 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in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates.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 devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

**Trademarks**

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, rPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL 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, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Octopus, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICKit, PICTail, PIC<sup>32</sup> logo, REAL ICE, rLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA 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.

© 2009, 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 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*



# MICROCHIP PIC18F46J11 FAMILY

## 28/44-Pin, Low-Power, High-Performance Microcontrollers

### Power Management Features with nanoWatt XLP™ for Extreme Low Power:

- Deep Sleep mode: CPU off, Peripherals off, Currents Down to 13 nA and 850 nA with RTCC
  - Able to wake-up on external triggers, programmable WDT or RTCC alarm
  - Ultra Low-Power Wake-up (ULPWU)
- Sleep mode: CPU off, Peripherals off, SRAM on, Fast Wake-up, Currents Down to 105 nA Typical
- Idle: CPU off, Peripherals on, Currents Down to 2.3  $\mu$ A Typical
- Run: CPU on, Peripherals on, Currents Down to 6.2  $\mu$ A Typical
- Timer1 Oscillator/w RTCC: 1  $\mu$ A, 32 kHz Typical
- Watchdog Timer: 2.2  $\mu$ A, 2V Typical

### Special Microcontroller Features:

- 5.5V Tolerant Inputs (digital only pins)
- Low-Power, High-Speed CMOS Flash Technology
- C Compiler Optimized Architecture for Re-Entrant Code
- Priority Levels for Interrupts
- Self-Programmable under Software Control
- 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 Breakpoints via Two Pins
- Operating Voltage Range of 2.0V to 3.6V
- On-Chip 2.5V Regulator
- Flash Program Memory of 10,000 Erase/Write Cycles Minimum and 20-Year Data Retention

### Peripheral Highlights:

- Peripheral Pin Select:
  - Allows independent I/O mapping of many peripherals
  - Continuous hardware integrity checking and safety interlocks prevent unintentional configuration changes
- Hardware Real-Time Clock and Calendar (RTCC):
  - Provides clock, calendar and alarm functions
- High-Current Sink/Source 25 mA/25 mA (PORTB and PORTC)

### Peripheral Highlights (Continued):

- Four Programmable External Interrupts
- Four Input Change Interrupts
- Two Enhanced Capture/Compare/PWM (ECCP) modules:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
  - Pulse steering control
- Two Master Synchronous Serial Port (MSSP) modules featuring:
  - 3-wire SPI (all 4 modes)
  - 1024-byte SPI Direct Memory Access (DMA) channel
  - I<sup>2</sup>C™ Master and Slave modes
- 8-Bit Parallel Master Port/Enhanced Parallel Slave Port
- Two-Rail – Rail Analog Comparators with Input Multiplexing
- 10-Bit, up to 13-Channel Analog-to-Digital (A/D) Converter module:
  - Auto-acquisition capability
  - Conversion available during Sleep
  - Self-Calibration
- High/Low-Voltage Detect module
- Charge Time Measurement Unit (CTMU):
  - Supports capacitive touch sensing for touch screens and capacitive switches
  - Provides a Precise Resolution Time Measurement for Both Flow Measurement and Simple Temperature Sensing
- Two Enhanced USART modules:
  - Supports RS-485, RS-232 and LIN/J2602
  - Auto-wake-up on Start bit
- Auto-Baud Detect

### Flexible Oscillator Structure:

- 1% Accurate High-Precision Internal Oscillator
- Two External Clock modes, up to 48 MHz (12 MIPS)
- Internal 31 kHz Oscillator, Internal Oscillators Tunable at 31 kHz and 8 MHz or 48 MHz with PLL
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if any clock stops
- Two-Speed Oscillator Start-up
- Programmable Reference Clock Output Generator

# PIC18F46J11 FAMILY

PIC18F/LF <sup>(1)</sup> Device	Pins	Program Memory (bytes)	SRAM (bytes)	Remappable Pins	Timers 8/16-Bit	ECCP/(PWM)	EUSART	MSSP		10-Bit A/D (ch)	Comparators	Deep Sleep	PMP/PSP	CTMU	RTCC	
								SPI w/DMA	I <sup>2</sup> C™							
PIC18F24J11	28	16K	3776	16	2/3	2	2	2	Y	Y	10	2	Y	N	Y	Y
PIC18F25J11	28	32K	3776	16	2/3	2	2	2	Y	Y	10	2	Y	N	Y	Y
PIC18F26J11	28	64K	3776	16	2/3	2	2	2	Y	Y	10	2	Y	N	Y	Y
PIC18F44J11	44	16K	3776	22	2/3	2	2	2	Y	Y	13	2	Y	Y	Y	Y
PIC18F45J11	44	32K	3776	22	2/3	2	2	2	Y	Y	13	2	Y	Y	Y	Y
PIC18F46J11	44	64K	3776	22	2/3	2	2	2	Y	Y	13	2	Y	Y	Y	Y
PIC18LF24J11	28	16K	3776	16	2/3	2	2	2	Y	Y	10	2	N	N	Y	Y
PIC18LF25J11	28	32K	3776	16	2/3	2	2	2	Y	Y	10	2	N	N	Y	Y
PIC18LF26J11	28	64K	3776	16	2/3	2	2	2	Y	Y	10	2	N	N	Y	Y
PIC18LF44J11	44	16K	3776	22	2/3	2	2	2	Y	Y	13	2	N	Y	Y	Y
PIC18LF45J11	44	32K	3776	22	2/3	2	2	2	Y	Y	13	2	N	Y	Y	Y
PIC18LF46J11	44	64K	3776	22	2/3	2	2	2	Y	Y	13	2	N	Y	Y	Y

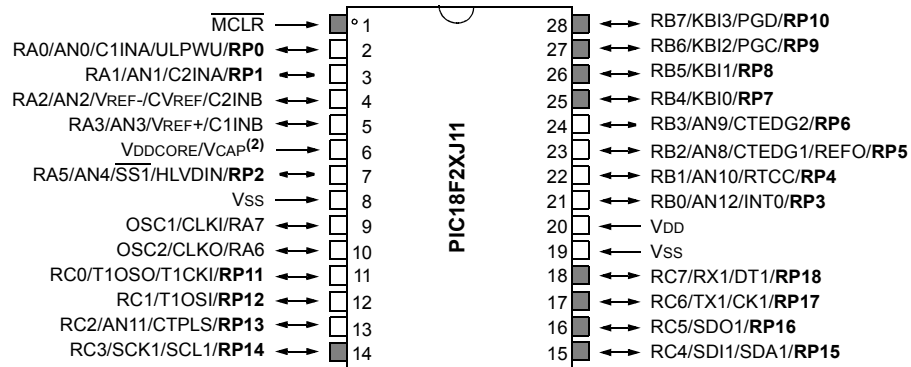
**Note 1:** See Section 1.3 “Details on Individual Family Devices”, Section 3.6 “Deep Sleep Mode” and Section 25.3 “On-Chip Voltage Regulator” for details describing the functional differences between PIC18F and PIC18LF variants in this device family.

# PIC18F46J11 FAMILY

## Pin Diagrams

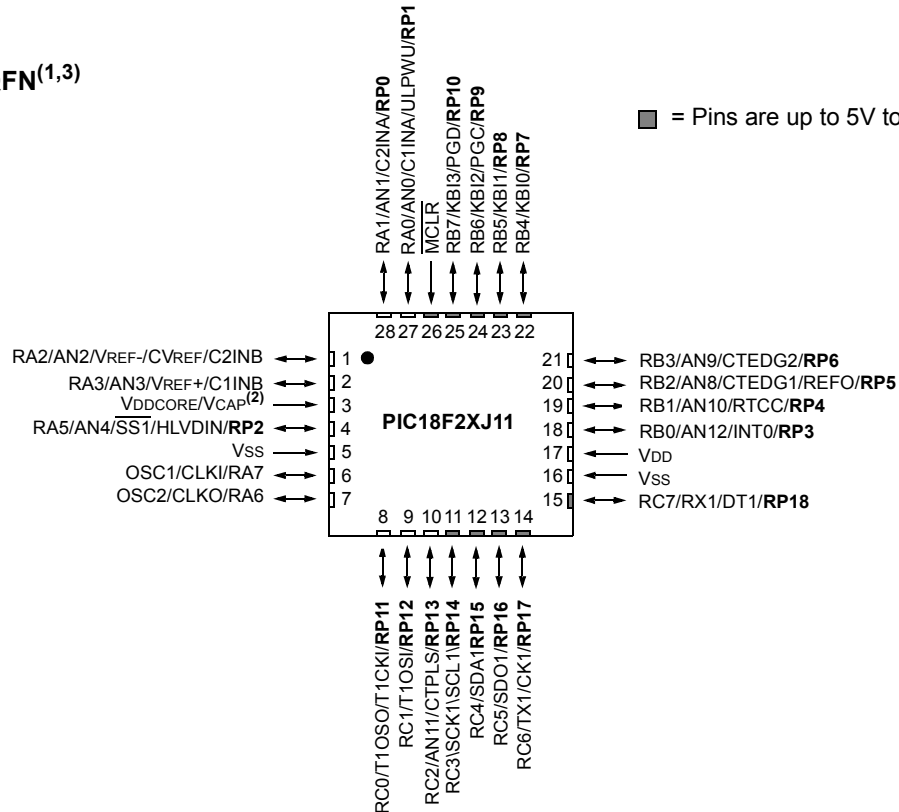
### 28-Pin SPDIP/SOIC/SSOP<sup>(1)</sup>

■ = Pins are up to 5V tolerant



### 28-Pin QFN<sup>(1,3)</sup>

■ = Pins are up to 5V tolerant



**Legend:** RPn represents remappable pins.

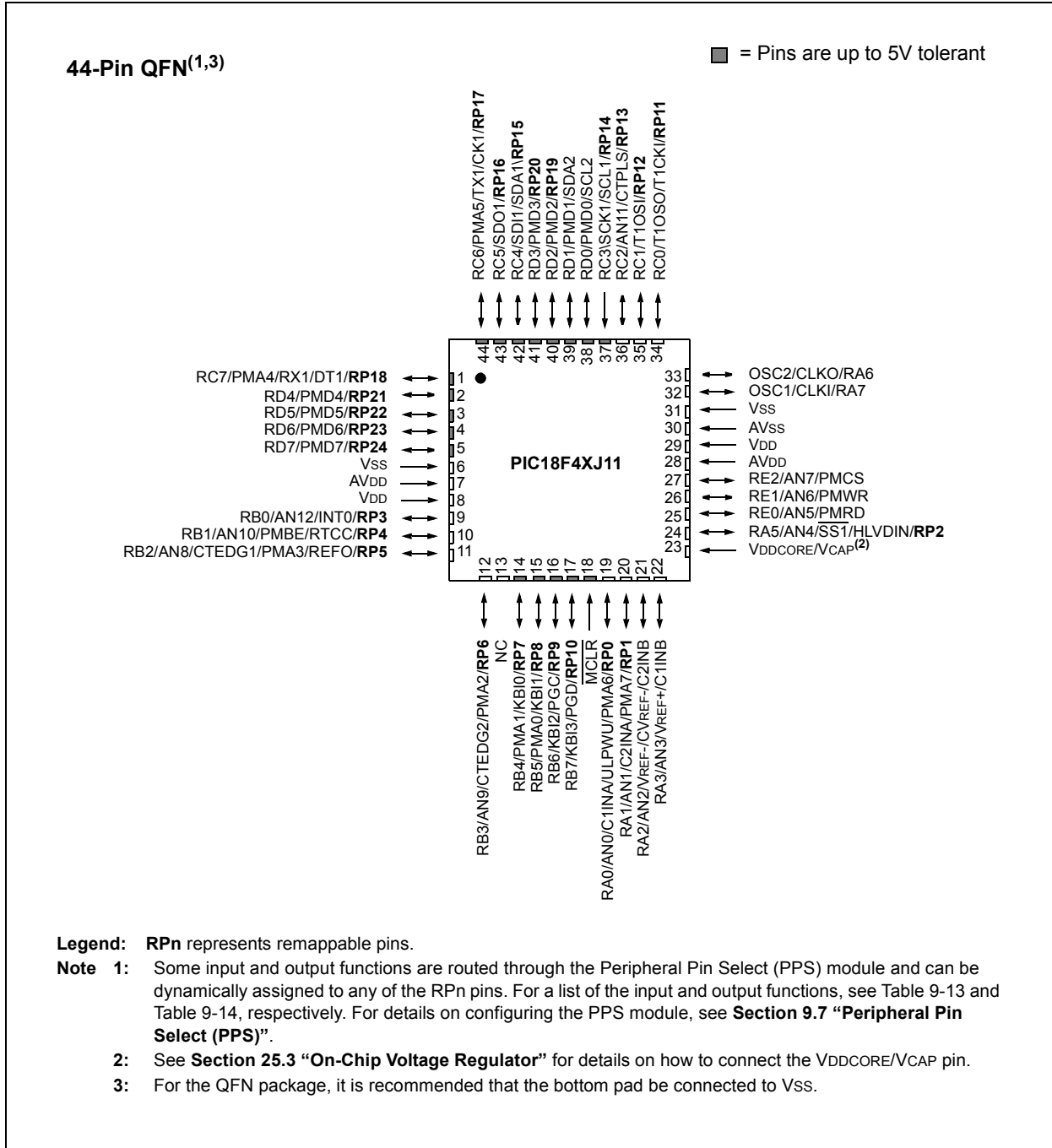
**Note 1:** Some input and output functions are routed through the Peripheral Pin Select (PPS) module and can be dynamically assigned to any of the RPn pins. For a list of the input and output functions, see Table 9-13 and Table 9-14, respectively. For details on configuring the PPS module, see **Section 9.7 "Peripheral Pin Select (PPS)"**.

**2:** See **Section 25.3 "On-Chip Voltage Regulator"** for details on how to connect the VDDCORE/VCAP pin.

**3:** For the QFN package, it is recommended that the bottom pad be connected to VSS.

# PIC18F46J11 FAMILY

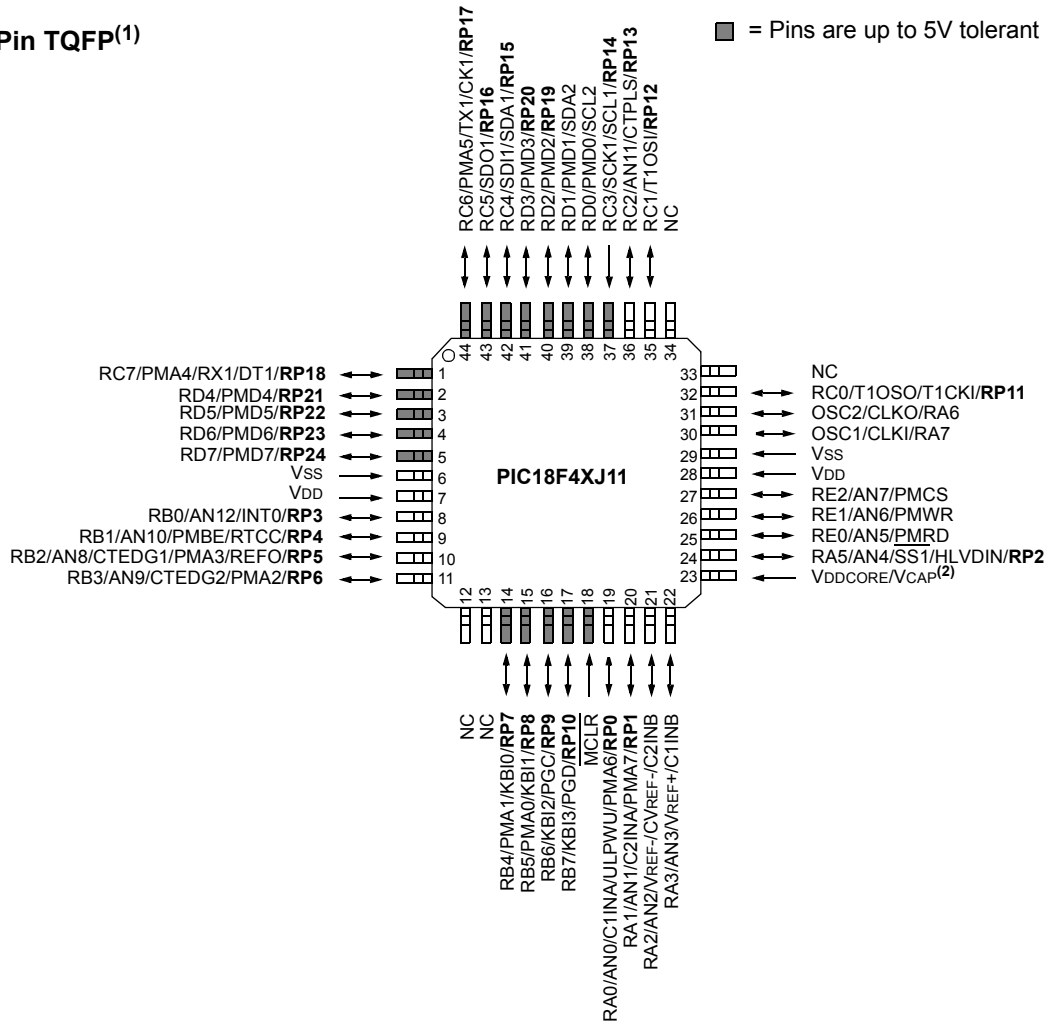
## Pin Diagrams (Continued)



# PIC18F46J11 FAMILY

## Pin Diagrams (Continued)

### 44-Pin TQFP<sup>(1)</sup>



**Legend:** RPn represents remappable pins.

**Note 1:** Some input and output functions are routed through the Peripheral Pin Select (PPS) module and can be dynamically assigned to any of the RPn pins. For a list of the input and output functions, see Table 9-13 and Table 9-14, respectively. For details on configuring the PPS module, see **Section 9.7 "Peripheral Pin Select (PPS)"**.

**Note 2:** See **Section 25.3 "On-Chip Voltage Regulator"** for details on how to connect the VDDCORE/VCAP pin.

# PIC18F46J11 FAMILY

---

## Table of Contents

1.0	Device Overview .....	11
2.0	Oscillator Configurations .....	31
3.0	Low-Power Modes .....	41
4.0	Reset .....	57
5.0	Memory Organization .....	71
6.0	Flash Program Memory .....	97
7.0	8 x 8 Hardware Multiplier .....	107
8.0	Interrupts .....	109
9.0	I/O Ports .....	125
10.0	Parallel Master Port (PMP) .....	165
11.0	Timer0 Module .....	191
12.0	Timer1 Module .....	195
13.0	Timer2 Module .....	207
14.0	Timer3 Module .....	209
15.0	Timer4 Module .....	219
16.0	Real-Time Clock and Calendar (RTCC) .....	221
17.0	Enhanced Capture/Compare/PWM (ECCP) Module .....	241
18.0	Master Synchronous Serial Port (MSSP) Module .....	265
19.0	Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) .....	321
20.0	10-bit Analog-to-Digital Converter (A/D) Module .....	345
21.0	Comparator Module .....	355
22.0	Comparator Voltage Reference Module .....	363
23.0	High/Low Voltage Detect (HLVD) .....	367
24.0	Charge Time Measurement Unit (CTMU) .....	373
25.0	Special Features of the CPU .....	389
26.0	Instruction Set Summary .....	407
27.0	Development Support .....	457
28.0	Electrical Characteristics .....	461
29.0	Packaging Information .....	499
	Appendix A: Revision History .....	511
	Appendix B: Device Differences .....	511
	The Microchip Web Site .....	525
	Customer Change Notification Service .....	525
	Customer Support .....	525
	Reader Response .....	526
	Product Identification System .....	527



## 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](mailto: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](http://www.microchip.com) to receive the most current information on all of our products.

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 1.0 DEVICE OVERVIEW

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

- PIC18F24J11
- PIC18F25J11
- PIC18F26J11
- PIC18F44J11
- PIC18F45J11
- PIC18F46J11
- PIC18LF24J11
- PIC18LF25J11
- PIC18LF26J11
- PIC18LF44J11
- PIC18LF45J11
- PIC18LF46J11

## 1.1 Core Features

### 1.1.1 nanoWatt TECHNOLOGY

All of the devices in the PIC18F46J11 family incorporate a range of features that can significantly reduce power consumption during operation. Key features are:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal RC oscillator, 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 operational requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the users to incorporate power-saving ideas into their application's software design.

### 1.1.2 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F46J11 family offer five different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes using crystals or ceramic resonators.
- Two External Clock modes offering the option of a divide-by-4 clock output.
- An internal oscillator block, which provides an 8 MHz clock and an INTRC source (approximately 31 kHz, stable over temperature and VDD), as well as a range of six user-selectable clock frequencies, between 125 kHz to 4 MHz, for a total of eight clock frequencies. This option frees an oscillator pin for use as an additional general purpose I/O.
- A Phase Lock Loop (PLL) frequency multiplier, available to the high-speed crystal, and external and internal oscillators, providing a clock speed up to 48 MHz.

The internal oscillator block provides a stable reference source that gives the PIC18F46J11 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, 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 (POR), or wake-up from Sleep mode, until the primary clock source is available.

### 1.1.3 EXPANDED MEMORY

The PIC18F46J11 family provides ample room for application code, from 16 Kbytes to 64 Kbytes of code space. The Flash cells for program memory are rated to last in excess of 10000 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 20 years.

The Flash program memory is readable and writable during normal operation. The PIC18F46J11 family also provides plenty of room for dynamic application data with up to 3.8 Kbytes of data RAM.

# PIC18F46J11 FAMILY

---

## 1.1.4 EXTENDED INSTRUCTION SET

The PIC18F46J11 family implements the optional extension to the PIC18 instruction set, adding eight new instructions and an Indexed Addressing mode. Enabled as a device configuration option, the extension has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as C.

## 1.1.5 EASY MIGRATION

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

The consistent pinout scheme used throughout the entire family also aids in migrating to the next larger device.

The PIC18F46J11 family is also pin compatible with other PIC18 families, such as the PIC18F4620, PIC18F4520 and PIC18F45J10. This allows a new dimension to the evolution of applications, allowing developers to select different price points within Microchip's PIC18 portfolio, while maintaining the same feature set.

## 1.2 Other Special Features

- **Communications:** The PIC18F46J11 family incorporates a range of serial and parallel communication peripherals. This device also includes two independent Enhanced USARTs and two Master Synchronous Serial Port (MSSP) modules, capable of both Serial Peripheral Interface (SPI) and I<sup>2</sup>C™ (Master and Slave) modes of operation. The device also has a parallel port and can be configured to serve as either a Parallel Master Port (PMP) or as a Parallel Slave Port (PSP).
- **ECCP Modules:** All devices in the family incorporate three Enhanced Capture/Compare/PWM (ECCP) modules to maximize flexibility in control applications. Up to four different time bases may be used to perform several different operations at once. Each of the ECCPs offers up to four PWM outputs, allowing for a total of eight PWMs. The ECCPs also offer many beneficial features, including polarity selection, programmable dead time, auto-shutdown and restart and Half-Bridge and Full-Bridge Output modes.

- **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, reducing 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 28.0 “Electrical Characteristics”** for time-out periods.

## 1.3 Details on Individual Family Devices

Devices in the PIC18F46J11 family are available in 28-pin and 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 two ways:

- Flash program memory (three sizes: 16 Kbytes for the PIC18FX4J11, 32 Kbytes for PIC18FX5J11 devices and 64 Kbytes for PIC18FX6J11)
- I/O ports (three bidirectional ports on 28-pin devices, five bidirectional ports on 44-pin devices)

All other features for devices in this family are identical. These are summarized in Table 1-1 and Table 1-2.

The pinouts for the PIC18F2XJ11 devices are listed in Table 1-3 and the pinouts for the PIC18F4XJ11 devices are listed in Table 1-4.

The PIC18F46J11 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 PIC18F46J11) have the voltage regulator enabled.

These parts can run from 2.15V-3.6V on VDD, but should have the VDDCORE pin connected to VSS through a low-ESR capacitor. Parts designated with an “LF” part number (such as PIC18LF46J11) do not enable the voltage regulator. For “LF” parts, an external supply of 2.0V-2.7V has to be supplied to the VDDCORE pin with 2.0V-3.6V supplied to VDD (VDDCORE should never exceed VDD).

For more details about the internal voltage regulator, see **Section 25.3 “On-Chip Voltage Regulator”**.

# PIC18F46J11 FAMILY

**TABLE 1-1: DEVICE FEATURES FOR THE PIC18F2XJ11 (28-PIN DEVICES)**

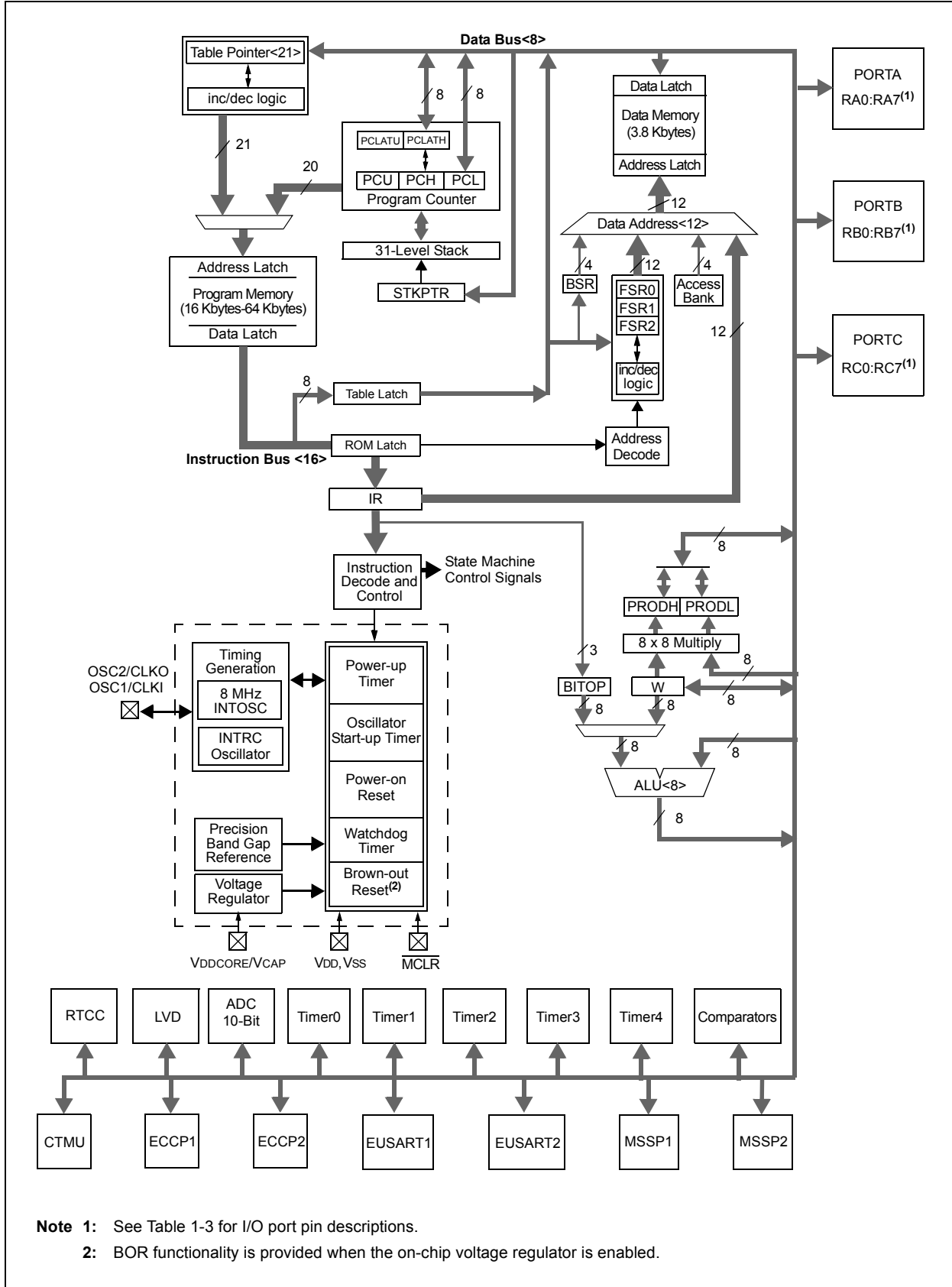
Features	PIC18F24J11	PIC18F25J11	PIC18F26J11
Operating Frequency	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
Program Memory (Bytes)	16K	32K	64K
Program Memory (Instructions)	8,192	16,384	32,768
Data Memory (Bytes)	3.8K	3.8K	3.8K
Interrupt Sources	30		
I/O Ports	Ports A, B, C		
Timers	5		
Enhanced Capture/Compare/PWM Modules	2		
Serial Communications	MSSP (2), Enhanced USART (2)		
Parallel Communications (PMP/PSP)	No		
10-Bit Analog-to-Digital Module	10 Input Channels		
Resets (and Delays)	POR, BOR, $\overline{\text{RESET}}$ Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$ , WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	28-Pin QFN, SOIC, SSOP and SPDIP (300 mil)		

**TABLE 1-2: DEVICE FEATURES FOR THE PIC18F4XJ11 (44-PIN DEVICES)**

Features	PIC18F44J11	PIC18F45J11	PIC18F46J11
Operating Frequency	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
Program Memory (Bytes)	16K	32K	64K
Program Memory (Instructions)	8,192	16,384	32,768
Data Memory (Bytes)	3.8K	3.8K	3.8K
Interrupt Sources	30		
I/O Ports	Ports A, B, C, D, E		
Timers	5		
Enhanced Capture/Compare/PWM Modules	2		
Serial Communications	MSSP (2), Enhanced USART (2)		
Parallel Communications (PMP/PSP)	Yes		
10-Bit Analog-to-Digital Module	13 Input Channels		
Resets (and Delays)	POR, BOR, $\overline{\text{RESET}}$ Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$ , WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	44-Pin QFN and TQFP		

# PIC18F46J11 FAMILY

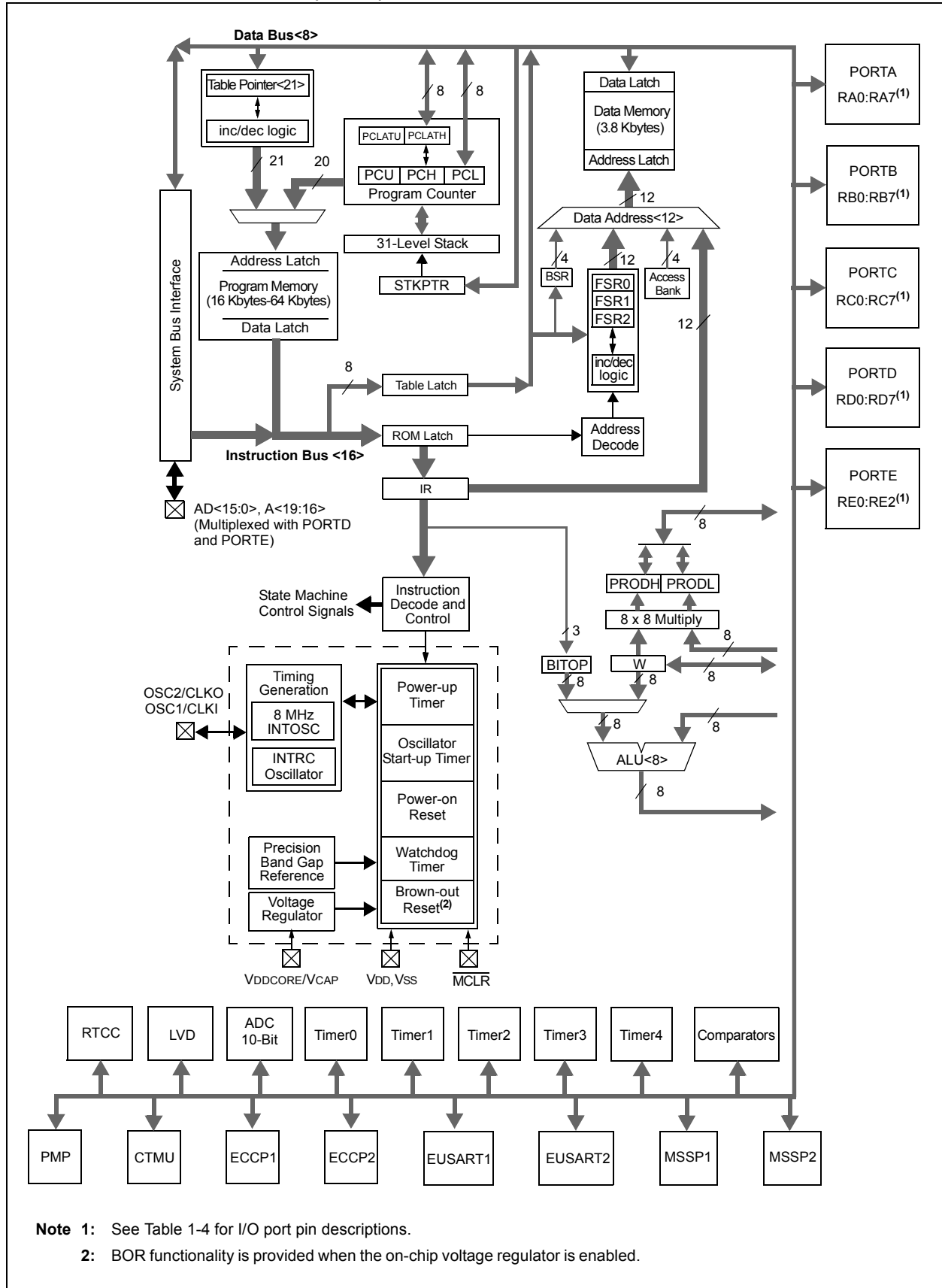
FIGURE 1-1: PIC18F2XJ11 (28-PIN) BLOCK DIAGRAM



- Note 1:** See Table 1-3 for I/O port pin descriptions.  
**Note 2:** BOR functionality is provided when the on-chip voltage regulator is enabled.

# PIC18F46J11 FAMILY

FIGURE 1-2: PIC18F4XJ11 (44-PIN) BLOCK DIAGRAM



# PIC18F46J11 FAMILY

**TABLE 1-3: PIC18F2XJ11 PINOUT I/O DESCRIPTIONS**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
MCLR	1	26	I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OSC1/CLKI/RA7 OSC1	9	6	I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; CMOS otherwise. Main oscillator input connection. External clock source input; always associated with pin function OSC1 (see related OSC1/CLKI pins). Digital I/O.
CLKI			I	CMOS	
RA7 <sup>(1)</sup>			I/O	TTL	
OSC2/CLKO/RA6 OSC2	10	7	O	—	Oscillator crystal or clock output. Connects to crystal or resonator in Crystal Oscillator mode. Main oscillator feedback output connection. In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. Digital I/O.
CLKO			O	—	
RA6 <sup>(1)</sup>			I/O	TTL	

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.



# PIC18F46J11 FAMILY

**TABLE 1-3: PIC18F2XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
RA0/AN0/C1INA/ULPWU/RP0 RA0 AN0 C1INA ULPWU RP0	2	27	I/O I I I I/O	DIG Analog Analog Analog DIG	PORTA is a bidirectional I/O port.  Digital I/O. Analog input 0. Comparator 1 input A. Ultra low-power wake-up input. Remappable peripheral pin 0.
RA1/AN1/C2INA/RP1 RA1 AN1 C2INA RP1	3	28	I O I I/O	DIG Analog Analog DIG	 Digital I/O. Analog input 1. Comparator 2 input A. Remappable peripheral pin 1.
RA2/AN2/VREF-/CVREF/C2INB RA2 AN2 VREF- CVREF C2INB	4	1	I/O I O I I	DIG Analog Analog Analog Analog	 Digital I/O. Analog input 2. A/D reference voltage (low) input. Comparator reference voltage output. Comparator 2 input B.
RA3/AN3/VREF+/C1INB RA3 AN3 VREF+ C1INB	5	2	I/O I I I	DIG Analog Analog Analog	 Digital I/O Analog input 3 A/D reference voltage (high) input Comparator 1 input B
RA5/AN4/ $\overline{SS1}$ /HLVDIN/ RP2 RA5 AN4 $\overline{SS1}$ HLVDIN RP2	7	4	I/O I I I I/O	DIG Analog TTL Analog DIG	 Digital I/O. Analog input 4. SPI slave select input. High/low-voltage detect input. Remappable peripheral pin 2.
RA6 <sup>(1)</sup> RA7 <sup>(1)</sup>					See the OSC2/CLKO/RA6 pin. See the OSC1/CLKI/RA7 pin.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

# PIC18F46J11 FAMILY

**TABLE 1-3: PIC18F2XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
RB0/AN12/INT0/RP3 RB0 AN12 INT0 RP3	21	18	I/O I I I/O	DIG Analog ST DIG	PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.  Digital I/O. Analog input 12. External interrupt 0. Remappable peripheral pin 3.
RB1/AN10/RTCC/RP4 RB1 AN10 RTCC RP4	22	19	I/O I O I/O	DIG Analog DIG DIG	Digital I/O. Analog input 10. Asynchronous serial transmit data output. Remappable peripheral pin 4.
RB2/AN8/CTEDG1/ REFO/RP5 RB2 AN8 CTEDG1 REFO RP5	23	20	I/O I I O I/O	DIG Analog ST DIG DIG	Digital I/O. Analog input 8. CTMU edge 1 input. Reference output clock. Remappable peripheral pin 5.
RB3/AN9/CTEDG2/RP6 RB3 AN9 CTEDG2 RP6	24	21	I/O I I/O I	DIG Analog ST DIG	Digital I/O. Analog input 9. CTMU edge 2 input. Remappable peripheral pin 6.
RB4/KBI0/RP7 RB4 KBI0 RP7	25	22	I/O I I/O	DIG TTL DIG	Digital I/O. Interrupt-on-change pin. Remappable peripheral pin 7.
RB5/KBI1/RP8 RB5 KBI1 RP8	26	23	I/O I/O I/O	DIG DIG DIG	Digital I/O. Parallel Master Port address. Remappable peripheral pin 8.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.





# PIC18F46J11 FAMILY

**TABLE 1-3: PIC18F2XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	28-SPDIP/ SSOP/ SOIC	28-QFN			
VSS1	8	5	P	—	Ground reference for logic and I/O pins.
VSS2	19	16	—	—	
VDD	20	17	P	—	Positive supply for peripheral digital logic and I/O pins.
VDDCORE/VCAP	6	3			Core logic power or external filter capacitor connection.
VDDCORE			P	—	Positive supply for microcontroller core logic (regulator disabled).
VCAP			P	—	External filter capacitor connection (regulator enabled).

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

# PIC18F46J11 FAMILY

**TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
MCLR	18	18	I	ST	Master Clear (Reset) input; this is an active-low Reset to the device.
OSC1/CLKI/RA7 OSC1	32	30	I	ST	Oscillator crystal or external clock input. Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS. Main oscillator input connection.
CLKI			I	CMOS	External clock source input; always associated with pin function OSC1 (see related OSC1/CLKI pins).
RA7 <sup>(1)</sup>			I/O	TTL	Digital I/O.
OSC2/CLKO/RA6 OSC2	33	31	O	—	Oscillator crystal or clock output Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	—	Main oscillator feedback output connection in RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6 <sup>(1)</sup>			I/O	TTL	Digital I/O.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

# PIC18F46J11 FAMILY

**TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
RA0/AN0/C1INA/ULPWU/PMA6/RP0	19	19			PORTA is a bidirectional I/O port.
RA0			I/O	DIG	Digital I/O.
AN0			I	Analog	Analog input 0.
C1INA			I	Analog	Comparator 1 input A.
ULPWU			I	Analog	Ultra low-power wake-up input.
PMA6			I/O	DIG	Parallel Master Port digital I/O.
RP0			I/O	DIG	Remappable peripheral pin 0.
RA1/AN1/C2INA/PMA7/RP1	20	20			
RA1			I	DIG	Digital I/O.
AN1			O	Analog	Analog input 1.
C2INA			I	Analog	Comparator 2 input A.
PMA7			I/O	DIG	Parallel Master Port digital I/O.
RP1			I/O	DIG	Remappable peripheral pin 1.
RA2/AN2/VREF-/CVREF/C2INB	21	21			
RA2			I/O	DIG	Digital I/O.
AN2			I	Analog	Analog input 2.
VREF-			O	Analog	A/D reference voltage (low) input.
CVREF			I	Analog	Comparator reference voltage output.
C2INB			I	Analog	Comparator 2 input B.
RA3/AN3/VREF+/C1INB	22	22			
RA3			I/O	DIG	Digital I/O.
AN3			I	Analog	Analog input 3.
VREF+			I	Analog	A/D reference voltage (high) input.
C1INB			I	Analog	Comparator 1 input B.
RA5/AN4/ $\overline{SS1}$ /HLVDIN/RP2	24	24			
RA5			I/O	DIG	Digital I/O.
AN4			I	Analog	Analog input 4.
$\overline{SS1}$			I	TTL	SPI slave select input.
HLVDIN			I	Analog	High/low-voltage detect input.
RP2			I/O	DIG	Remappable peripheral pin 2.
RA6 <sup>(1)</sup>					See the OSC2/CLKO/RA6 pin.
RA7 <sup>(1)</sup>					See the OSC1/CLKI/RA7 pin.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

# PIC18F46J11 FAMILY

**TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
RB0/AN12/INT0/RP3 RB0 AN12 INT0 RP3	9	8	I/O	DIG	PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.  Digital I/O.
RB1/AN10/PMBE/RTCC/RP4 RB1 AN10 PMBE RTCC RP4	10	9	I/O	DIG	Analog input 12. External interrupt 0. Remappable peripheral pin 3.  Digital I/O.
RB2/AN8/CTEDG1/PMA3/REFO/ RP5 RB2 AN8 CTEDG1 PMA3 REFO RP5	11	10	I/O	DIG	Analog input 10. Parallel Master Port byte enable. Asynchronous serial transmit data output. Remappable peripheral pin 4.  Digital I/O.
RB3/AN9/CTEDG2/PMA2/ RP6 RB3 AN9 CTEDG2 PMA2 RP6	12	11	I/O	DIG	Analog input 8. CTMU edge 1 input. Parallel Master Port address. Reference output clock. Remappable peripheral pin 5.  Digital I/O.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.







# PIC18F46J11 FAMILY

**TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
RC6/PMA5/TX1/CK1/RP17	44	44			PORTC (continued)
RC6			I/O	ST	Digital I/O.
PMA5			I/O	DIG	Parallel Master Port address.
TX1			O	DIG	EUSART1 asynchronous transmit.
CK1			I/O	ST	EUSART1 synchronous clock (see related RX1/DT1).
RP17			I/O	DIG	Remappable peripheral pin 17.
RC7/PMA4/RX1/DT1/RP18	1	1			
RC7			I/O	ST	EUSART1 asynchronous receive.
PMA4			I/O	DIG	Parallel Master Port address.
RX1			I	ST	EUSART1 synchronous data (see related TX1/CK1).
DT1			I/O	ST	Synchronous serial data output/input.
RP18			I/O	DIG	Remappable peripheral pin 18.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

# PIC18F46J11 FAMILY

**TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
RD0/PMD0/SCL2 RD0 PMD0 SCL2	38	38	I/O I/O I/O	ST DIG I <sup>2</sup> C	PORTD is a bidirectional I/O port.  Digital I/O. Parallel Master Port data. I <sup>2</sup> C™ data input/output.
RD1/PMD1/SDA2 RD1 PMD1 SDA2	39	39	I/O I/O I/O	ST DIG I <sup>2</sup> C	Digital I/O. Parallel Master Port data. I <sup>2</sup> C data input/output.
RD2/PMD2/RP19 RD2 PMD2 RP19	40	40	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 19.
RD3/PMD3/RP20 RD3 PMD3 RP20	41	41	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 20.
RD4/PMD4/RP21 RD4 PMD4 RP21	2	2	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 21.
RD5/PMD5/RP22 RD5 PMD5 RP22	3	3	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 22.
RD6/PMD6/RP23 RD6 PMD6 RP23	4	4	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 23.
RD7/PMD7/RP24 RD7 PMD7 RP24	5	5	I/O I/O I/O	ST DIG DIG	Digital I/O. Parallel Master Port data. Remappable peripheral pin 24.

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

**Note 1:** RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.



# PIC18F46J11 FAMILY

---

NOTES:

## 2.0 OSCILLATOR CONFIGURATIONS

### 2.1 Overview

Devices in the PIC18F46J11 family incorporate a different oscillator and microcontroller clock system than general purpose PIC18F devices.

The PIC18F46J11 family has additional prescalers and postscalers, which have been added to accommodate a wide range of oscillator frequencies. Figure 2-1 provides an overview of the oscillator structure.

Other oscillator features used in PIC18 enhanced microcontrollers, such as the internal oscillator block and clock switching, remain the same. They are discussed later in this chapter.

#### 2.1.1 OSCILLATOR CONTROL

The operation of the oscillator in PIC18F46J11 family devices is controlled through three Configuration registers and two control registers. Configuration registers, CONFIG1L, CONFIG1H and CONFIG2L, select the oscillator mode, PLL prescaler and CPU divider options. As Configuration bits, these are set when the device is programmed and left in that configuration until the device is reprogrammed.

The OSCCON register (Register 2-2) selects the Active Clock mode; it is primarily used in controlling clock switching in power-managed modes. Its use is discussed in **Section 2.3.1 “Oscillator Control Register”**.

The OSCTUNE register (Register 2-1) is used to trim the INTOSC frequency source and select the low-frequency clock source that drives several special features. The OSCTUNE register is also used to activate or disable the Phase Locked Loop (PLL). Its use is described in **Section 2.2.5.1 “OSCTUNE Register”**.

### 2.2 Oscillator Types

PIC18F46J11 family devices can be operated in eight distinct oscillator modes. Users can program the FOSC<2:0> Configuration bits to select one of the modes listed in Table 2-1. For oscillator modes which produce a clock output (CLKO) on pin RA6, the output frequency will be one fourth of the peripheral clock frequency. The clock output stops when in Sleep mode, but will continue during Idle mode (see Figure 2-1).

TABLE 2-1: OSCILLATOR MODES

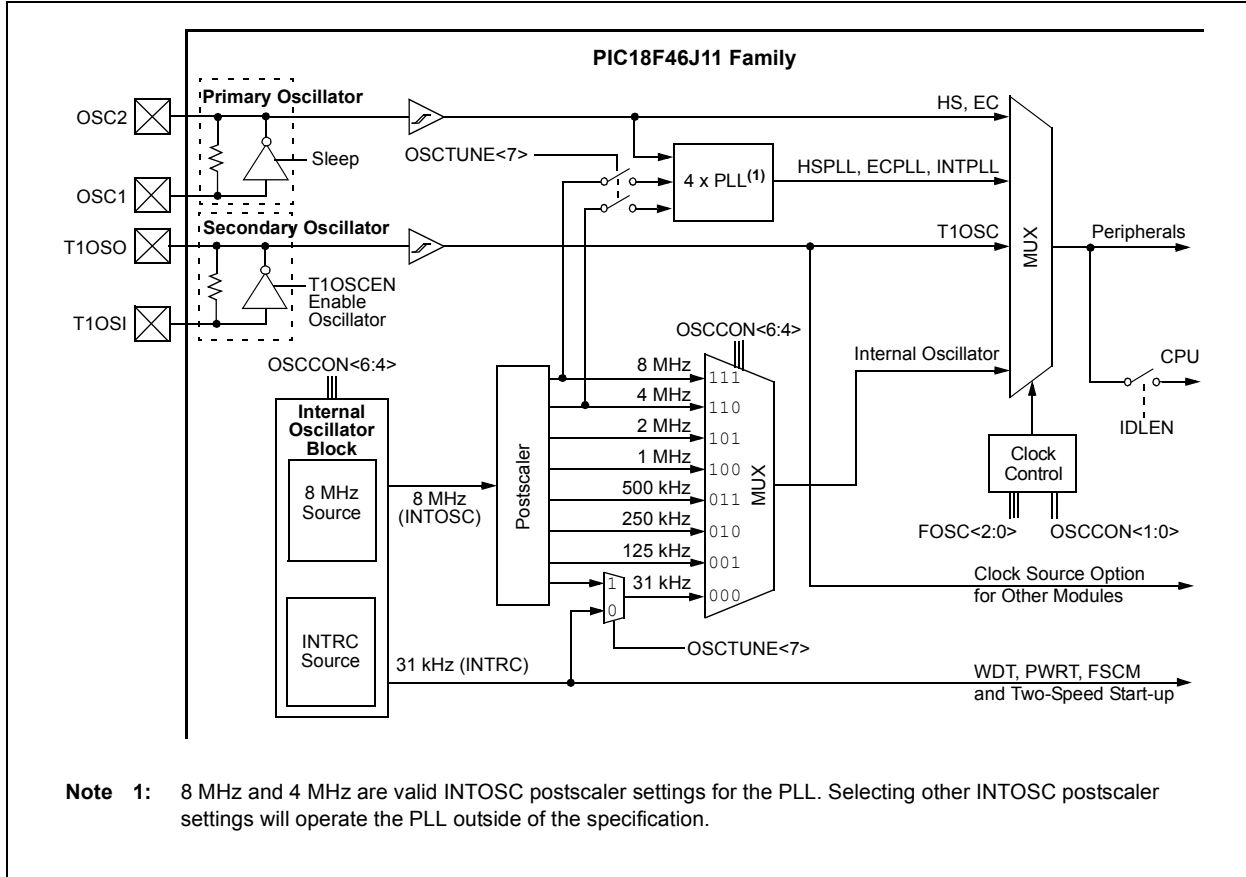
Mode	Description
ECPLL	External Clock Input mode, the PLL can be enabled or disabled in software, CLKO on RA6, apply external clock signal to RA7.
EC	External Clock Input mode, the PLL is always disabled, CLKO on RA6, apply external clock signal to RA7.
HSPLL	High-Speed Crystal/Resonator mode, PLL can be enabled or disabled in software, crystal/resonator connected between RA6 and RA7.
HS	High-Speed Crystal/Resonator mode, PLL always disabled, crystal/resonator connected between RA6 and RA7.
INTOSCPLLO	Internal Oscillator mode, PLL can be enabled or disabled in software, CLKO on RA6, port function on RA7, the internal oscillator block is used to derive both the primary clock source and the postscaled internal clock.
INTOSCPLL	Internal Oscillator mode, PLL can be enabled or disabled in software, port function on RA6 and RA7, the internal oscillator block is used to derive both the primary clock source and the postscaled internal clock.
INTOSCO	Internal Oscillator mode, PLL is always disabled, CLKO on RA6, port function on RA7, the output of the INTOSC postscaler serves as both the postscaled internal clock and the primary clock source.
INTOSC	Internal Oscillator mode, PLL is always disabled, port function on RA6 and RA7, the output of the INTOSC postscaler serves as both the postscaled internal clock and the primary clock source.

# PIC18F46J11 FAMILY

## 2.2.1 OSCILLATOR MODES

Figure 2-1 helps in understanding the oscillator structure of the PIC18F46J11 family of devices.

**FIGURE 2-1: PIC18F46J11 FAMILY CLOCK DIAGRAM**





# PIC18F46J11 FAMILY

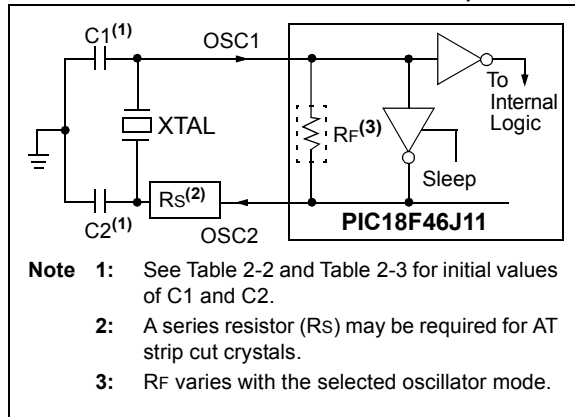
## 2.2.2 CRYSTAL OSCILLATOR/CERAMIC RESONATORS

In HS and HSPLL Oscillator modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. Figure 2-2 displays 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 2-2: CRYSTAL/CERAMIC RESONATOR OPERATION (HS OR HSPLL CONFIGURATION)**



**TABLE 2-2: 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
<b>Capacitor values are for design guidance only.</b>			
These capacitors were tested with the resonators listed below for basic start-up and operation. <b>These values are not optimized.</b>			
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 2-3 for additional information.			
<b>Resonators Used:</b>			
4.0 MHz			
8.0 MHz			
16.0 MHz			

**TABLE 2-3: 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 not only 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.

An internal postscaler allows users to select a clock frequency other than that of the crystal or resonator. Frequency division is determined by the CPDIV Configuration bits. Users may select a clock frequency of the oscillator frequency, or 1/2, 1/3 or 1/6 of the frequency.

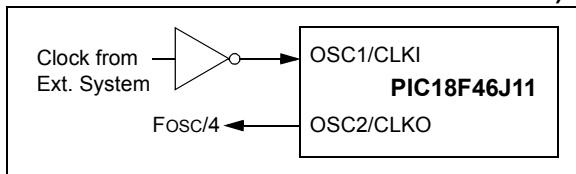
# PIC18F46J11 FAMILY

## 2.2.3 EXTERNAL CLOCK INPUT

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 (POR) or after an exit from Sleep mode.

In the EC Oscillator mode, the oscillator frequency divided-by-4 is available on the OSC2 pin. In the ECPLL Oscillator mode, the PLL output divided-by-4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-3 displays the pin connections for the EC Oscillator mode.

**FIGURE 2-3: EXTERNAL CLOCK INPUT OPERATION (EC AND ECPLL CONFIGURATION)**



## 2.2.4 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.

## 2.2.5 INTERNAL OSCILLATOR BLOCK

The PIC18F46J11 family devices include an internal oscillator block which generates two different clock signals; either can be used as the microcontroller's clock source. The internal oscillator may eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

The main output (INTOSC) is an 8 MHz clock source which can be used to directly drive the device clock. It also drives the INTOSC postscaler, which can provide a range of clock frequencies from 31 kHz to 8 MHz. Additionally, the INTOSC may be used in conjunction with the PLL to generate clock frequencies up to 32 MHz.

The other clock source is the internal RC oscillator (INTRC), which provides a nominal 31 kHz output. INTRC is enabled if it is selected as the device clock source. It is also enabled automatically when any of the following are enabled:

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

These features are discussed in more detail in **Section 25.0 "Special Features of the CPU"**.

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (page 38).

## 2.2.5.1 OSCTUNE Register

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

When the OSCTUNE register is modified, the INTOSC frequency will begin shifting to the new frequency. The INTOSC clock will stabilize, typically within 1  $\mu$ s. Code execution continues during this shift. There is no indication that the shift has occurred.

The OSCTUNE register also contains the INTSRC bit. The INTSRC bit allows users to select which internal oscillator provides the clock source when the 31 kHz frequency option is selected. This is covered in more detail in **Section 2.3.1 "Oscillator Control Register"**.

The 4x Phase Locked Loop (PLL) can be used with the internal oscillator block to produce faster device clock speeds than are normally possible with the internal oscillator sources. When enabled, the PLL produces a clock speed up to 32 MHz.

PLL operation is controlled through software. The control bit, PLEN (OSCTUNE<6>), is used to enable or disable its operation. The PLL is available only to INTOSC when the device is configured to use one of the INTPLL modes as the primary clock source, SCS<1:0> = 00 (FOSC<2:0> = 011 or 010). Additionally, the PLL will only function when the selected output frequency is either 4 MHz or 8 MHz (OSCCON<6:4> = 111 or 110).

When configured for one of the PLL enabled modes, setting the PLEN bit does not immediately switch the device clock to the PLL output. The PLL requires up to two milliseconds to start-up and lock, during which time, the device continues to be clocked. Once the PLL output is ready, the microcontroller core will automatically switch to the PLL derived frequency.

## 2.2.5.2 Internal Oscillator Output Frequency and Drift

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8.0 MHz. However, this frequency may drift as V<sub>DD</sub> or temperature changes, which can affect the controller operation in a variety of ways.

The low-frequency INTRC oscillator operates independently of the INTOSC source. Any changes in INTOSC across voltage and temperature are not necessarily reflected by changes in INTRC and vice versa.

## 2.2.5.3 Compensating for INTOSC Drift

It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This has no effect on the INTRC clock source frequency.

Tuning the INTOSC source requires knowing when to make the adjustment, in which direction it should be made, and in some cases, how large a change is needed. When using the EUSART, for example, an adjustment may be required when it begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the device clock frequency is too high; to adjust for this, decrement the value in OSCTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low; to compensate, increment OSCTUNE to increase the clock frequency.

It is also possible to verify device clock speed against a reference clock. Two timers may be used: one timer is clocked by the peripheral clock, while the other is clocked by a fixed reference source, such as the Timer1 oscillator. Both timers are cleared, but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is greater than expected, then the internal oscillator block is running too fast. To adjust for this, decrement the OSCTUNE register.

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

If the measured time is greater than the calculated time, the internal oscillator block is running too fast; to compensate, decrement the OSCTUNE register. If the measured time is less than the calculated time, the internal oscillator block is running too slow; to compensate, increment the OSCTUNE register.

# PIC18F46J11 FAMILY

## REGISTER 2-1: OSCTUNE: OSCILLATOR TUNING REGISTER (ACCESS F9Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

- bit 7      **INTSRC:** Internal Oscillator Low-Frequency Source Select bit  
           1 = 31.25 kHz device clock derived from 8 MHz INTOSC source (divide-by-256 enabled)  
           0 = 31 kHz device clock derived directly from INTRC internal oscillator
- bit 6      **PLLEN:** Frequency Multiplier Enable bit  
           1 = PLL enabled  
           0 = PLL disabled
- bit 5-0    **TUN<5:0>:** Frequency Tuning bits  
           011111 = Maximum frequency  
           011110  
           •  
           •  
           •  
           000001  
           000000 = Center frequency; oscillator module is running at the calibrated frequency  
           111111  
           •  
           •  
           •  
           100000 = Minimum frequency

## 2.3 Clock Sources and Oscillator Switching

Like previous PIC18 enhanced devices, the PIC18F46J11 family includes a feature that allows the device clock source to be switched from the main oscillator to an alternate, low-frequency clock source. PIC18F46J11 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 Block

The **Primary Oscillators** include the External Crystal and Resonator modes, the External Clock modes and the internal oscillator block. The particular mode is defined by the FOSC<2:0> Configuration bits. The details of these modes are covered earlier in this chapter.

The **Secondary Oscillators** are external sources that are 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.

PIC18F46J11 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 (RTC). Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T1CKI/RP11 and RC1/T1OSI/RP12 pins. Like the HS Oscillator mode circuits, loading capacitors are also connected from each pin to ground. The Timer1 oscillator is discussed in more detail in **Section 12.5 “Timer1 Oscillator”**.

In addition to being a primary clock source, the **postscaled internal clock** 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 (FSCM).

## 2.3.1 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, SCS<1:0>, select the clock source. The available clock sources are the primary clock (defined by the FOSC<2:0> Configuration bits), the secondary clock (Timer1 oscillator) and the postscaled internal clock. The clock source changes immediately, after one or more of the bits is written to, following a brief clock transition interval. The SCS bits are cleared on all forms of Reset.

The Internal Oscillator Frequency Select bits, IRCF<2:0>, select the frequency output provided on the postscaled internal clock line. The choices are the INTRC source, the INTOSC source (8 MHz) or one of the frequencies derived from the INTOSC postscaler (31 kHz to 4 MHz). If the postscaled internal clock is supplying the device clock, changing the states of these bits will have an immediate change on the internal oscillator's output. On device Resets, the default output frequency of the INTOSC postscaler is set at 4 MHz.

When an output frequency of 31 kHz is selected (IRCF<2:0> = 000), users may choose the internal oscillator, which acts as the source. This is done with the INTSRC bit in the OSCTUNE register (OSCTUNE<7>). Setting this bit selects INTOSC as a 31.25 kHz clock source by enabling the divide-by-256 output of the INTOSC postscaler. Clearing INTSRC selects INTRC (nominally 31 kHz) as the clock source.

This option allows users to select the tunable and more precise INTOSC as a clock source, while maintaining power savings with a very low clock speed. Regardless of the setting of INTSRC, INTRC always remains the clock source for features such as the WDT and the FSCM.

The OSTS and T1RUN 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 (T1CON<6>) 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 none of these bits are set, the INTRC is providing the clock or the internal oscillator block 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 "Low-Power Modes"**.

**Note 1:** The Timer1 crystal driver 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 the Timer1 clock source will be ignored, unless the CONFIG2L register's T1DIG bit is set.

**2:** If Timer1 is driving a crystal, it is recommended that the Timer1 oscillator be operating and stable prior to switching to it as the clock source; otherwise, a very long delay may occur while the Timer1 oscillator starts.

## 2.3.2 OSCILLATOR TRANSITIONS

PIC18F46J11 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 more detail in **Section 3.1.2 "Entering Power-Managed Modes"**.

# PIC18F46J11 FAMILY

## REGISTER 2-2: OSCCON: OSCILLATOR CONTROL REGISTER (ACCESS FD3h)

R/W-0	R/W-1	R/W-1	R/W-0	R-1 <sup>(1)</sup>	U-1	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	—	SCS1	SCS0
bit 7						bit 0	

### Legend:

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

- bit 7      **IDLEN:** Idle Enable bit  
 1 = Device enters Idle mode on `SLEEP` instruction  
 0 = Device enters Sleep mode on `SLEEP` instruction
- bit 6-4    **IRCF<2:0>:** Internal Oscillator Frequency Select bits  
 111 = 8 MHz (INTOSC drives clock directly)  
 110 = 4 MHz<sup>(2)</sup>  
 101 = 2 MHz  
 100 = 1 MHz  
 011 = 500 kHz  
 010 = 250 kHz  
 001 = 125 kHz  
 000 = 31 kHz (from either INTOSC/256 or INTRC directly)<sup>(3)</sup>
- bit 3      **OSTS:** Oscillator Start-up Time-out Status bit<sup>(1)</sup>  
 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
- bit 2      **Unimplemented:** Read as '1'
- bit 1-0    **SCS<1:0>:** System Clock Select bits  
 11 = Postscaled internal clock (INTRC/INTOSC derived)  
 10 = Reserved  
 01 = Timer1 oscillator  
 00 = Primary clock source (INTOSC postscaler output when FOSC<2:0> = 001 or 000)  
 00 = Primary clock source (CPU divider output for other values of FOSC<2:0>)

- Note 1:** Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.  
**Note 2:** Default output frequency of INTOSC on Reset (4 MHz).  
**Note 3:** Source selected by the INTSRC bit (OSCTUNE<7>).

# PIC18F46J11 FAMILY

## 2.4 Reference Clock Output

In addition to the peripheral clock/4 output in certain oscillator modes, the device clock in the PIC18F46J11 family can also be configured to provide a reference clock output signal to a port pin. This feature is available in all oscillator configurations and allows the user to select a greater range of clock submultiples to drive external devices in the application.

This reference clock output is controlled by the REFOCON register (Register 2-3). Setting the ROON bit (REFOCON<7>) makes the clock signal available on the REFO (RB2) pin. The RODIV<3:0> bits enable the selection of 16 different clock divider options.

The ROSSLP and ROSEL bits (REFOCON<5:4>) control the availability of the reference output during Sleep mode. The ROSEL bit determines if the oscillator is on OSC1 and OSC2, or the current system clock source is used for the reference clock output. The ROSSLP bit determines if the reference source is available on RB2 when the device is in Sleep mode.

To use the reference clock output in Sleep mode, both the ROSSLP and ROSEL bits must be set. The device clock must also be configured for an EC or HS mode; otherwise, the oscillator on OSC1 and OSC2 will be powered down when the device enters Sleep mode. Clearing the ROSEL bit allows the reference output frequency to change as the system clock changes during any clock switches.

**REGISTER 2-3: REFOCON: REFERENCE OSCILLATOR CONTROL REGISTER (BANKED F3Dh)**

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ROON	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0
bit 7							bit 0

**Legend:**

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

- bit 7      **ROON:** Reference Oscillator Output Enable bit  
           1 = Reference oscillator enabled on REFO pin  
           0 = Reference oscillator disabled
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **ROSSLP:** Reference Oscillator Output Stop in Sleep bit  
           1 = Reference oscillator continues to run in Sleep  
           0 = Reference oscillator is disabled in Sleep
- bit 4      **ROSEL:** Reference Oscillator Source Select bit  
           1 = Primary oscillator used as the base clock<sup>(1)</sup>  
           0 = System clock used as the base clock; base clock reflects any clock switching of the device
- bit 3-0    **RODIV<3:0>:** Reference Oscillator Divisor Select bits  
           1111 = Base clock value divided by 32,768  
           1110 = Base clock value divided by 16,384  
           1101 = Base clock value divided by 8,192  
           1100 = Base clock value divided by 4,096  
           1011 = Base clock value divided by 2,048  
           1010 = Base clock value divided by 1,024  
           1001 = Base clock value divided by 512  
           1000 = Base clock value divided by 256  
           0111 = Base clock value divided by 128  
           0110 = Base clock value divided by 64  
           0101 = Base clock value divided by 32  
           0100 = Base clock value divided by 16  
           0011 = Base clock value divided by 8  
           0010 = Base clock value divided by 4  
           0001 = Base clock value divided by 2  
           0000 = Base clock value

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

# PIC18F46J11 FAMILY

---

## 2.5 Effects of Power-Managed Modes on Various Clock Sources

When the PRI\_IDLE mode is selected, the designated primary oscillator continues to run without interruption. 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 internal oscillator modes (RC\_RUN and RC\_IDLE), the internal oscillator block 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 25.2 “Watchdog Timer (WDT)”**, **Section 25.4 “Two-Speed Start-up”** and **Section 25.5 “Fail-Safe Clock Monitor”** for more information on WDT, FSCM and Two-Speed Start-up). The INTOSC output at 8 MHz may be used directly to clock the device or may be divided down by the postscaler. The INTOSC output is disabled if the clock is provided directly from the INTRC output.

If Sleep mode is selected, all clock sources, which are no longer required, are stopped. Since all the transistor switching currents have been stopped, Sleep mode achieves the lowest current consumption of the device (only leakage currents) outside of Deep Sleep mode.

Enabling any on-chip feature that will operate during Sleep mode increases the current consumed during Sleep mode. The INTRC is required to support WDT operation. The Timer1 oscillator may be operating to support an RTC. Other features may be operating that do not require a device clock source (i.e., MSSP slave, PMP, INTx pins, etc.). Peripherals that may add significant current consumption are listed in **Section 28.2 “DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial)”**.

## 2.6 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 4.6 “Power-up Timer (PWRT)”**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on power-up (parameter 33, Table 28-13).

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

There is a delay of interval, TcSD (parameter 38, Table 28-13), following POR, while the controller becomes ready to execute instructions. This delay runs concurrently with any other delays. This may be the only delay that occurs when any of the internal oscillator or EC modes are used as the primary clock source.



## 3.0 LOW-POWER MODES

The PIC18F46J11 family devices can manage power consumption through clocking to the CPU and the peripherals. In general, reducing the clock frequency and the amount of circuitry being clocked reduces power consumption.

For managing power in an application, the primary modes of operation are:

- Run Mode
- Idle Mode
- Sleep Mode
- Deep Sleep Mode

Additionally, there is an Ultra Low-Power Wake-up (ULPWU) mode for generating an interrupt-on-change on RA0.

These modes define which portions of the device are clocked and at what speed.

- The Run and Idle modes can use any of the three available clock sources (primary, secondary or internal oscillator blocks).
- The Sleep mode does not use a clock source.

The ULPWU mode on RA0 allows a slow falling voltage to generate an interrupt-on-change on RA0 without excess current consumption. See **Section 3.7 “Ultra Low-Power Wake-up”**.

The power-managed modes include several power-saving features offered on previous PIC® devices, such as clock switching, ULPWU and Sleep mode. In addition, the PIC18F46J11 family devices add a new power-managed Deep Sleep mode.

### 3.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires these decisions:

- Will the CPU be clocked?
- If so, which clock source will be used?

The IDLEN bit (OSCCON<7>) controls CPU clocking and the SCS<1:0> bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 3-1.

#### 3.1.1 CLOCK SOURCES

The SCS<1:0> bits allow the selection of one of three clock sources for power-managed modes. They are:

- Primary clock source – Defined by the FOSC<2:0> Configuration bits
- Timer1 clock – Provided by the secondary oscillator
- Postscaled internal clock – Derived from the internal oscillator block

#### 3.1.2 ENTERING POWER-MANAGED MODES

Switching from one clock source to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source.

Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch also may be subject to clock transition delays. These delays 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, the IDLEN bit or the DSEN bit prior to issuing a SLEEP instruction.

If the IDLEN and DSEN bits are already configured correctly, it may only be necessary to perform a SLEEP instruction to switch to the desired mode.

# PIC18F46J11 FAMILY

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

Mode	DSCONH<7>	OSCCON<7,1:0>		Module Clocking		Available Clock and Oscillator Source
	DSEN <sup>(1)</sup>	IDLEN <sup>(1)</sup>	SCS<1:0>	CPU	Peripherals	
Sleep	0	0	N/A	Off	Off	Timer1 oscillator and/or RTCC optionally enabled
Deep Sleep <sup>(2)</sup>	1	0	N/A	Off	—	RTCC can run uninterrupted using the Timer1 or internal low-power RC oscillator
PRI_RUN	0	N/A	00	Clocked	Clocked	The normal, full-power execution mode. Primary clock source (defined by FOSC<2:0>)
SEC_RUN	0	N/A	01	Clocked	Clocked	Secondary – Timer1 oscillator
RC_RUN	0	N/A	11	Clocked	Clocked	Postscaled internal clock
PRI_IDLE	0	1	00	Off	Clocked	Primary clock source (defined by FOSC<2:0>)
SEC_IDLE	0	1	01	Off	Clocked	Secondary – Timer1 oscillator
RC_IDLE	0	1	11	Off	Clocked	Postscaled internal clock

**Note 1:** IDLEN and DSEN reflect their values when the SLEEP instruction is executed.

**Note 2:** Deep Sleep entirely shuts off the voltage regulator for ultra low-power consumption. See **Section 3.6 “Deep Sleep Mode”** for more information.

### 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 in a given power-managed mode. When the OSTS bit is set, the primary clock would be providing the device clock. When the T1RUN bit is set, the Timer1 oscillator would be providing the clock. If neither of these bits is set, INTRC would be 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 or Deep 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 and DSEN bits at the time the instruction is executed. If another SLEEP instruction is executed, the device will enter the power-managed mode specified by IDLEN and DSEN at that time. If IDLEN or DSEN have 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 25.4 “Two-Speed Start-up”** for details). In this mode, the OSTS bit is set (see **Section 2.3.1 “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 low-power consumption while still using a high-accuracy clock source.

SEC\_RUN mode is entered by setting the SCS<1:0> 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.

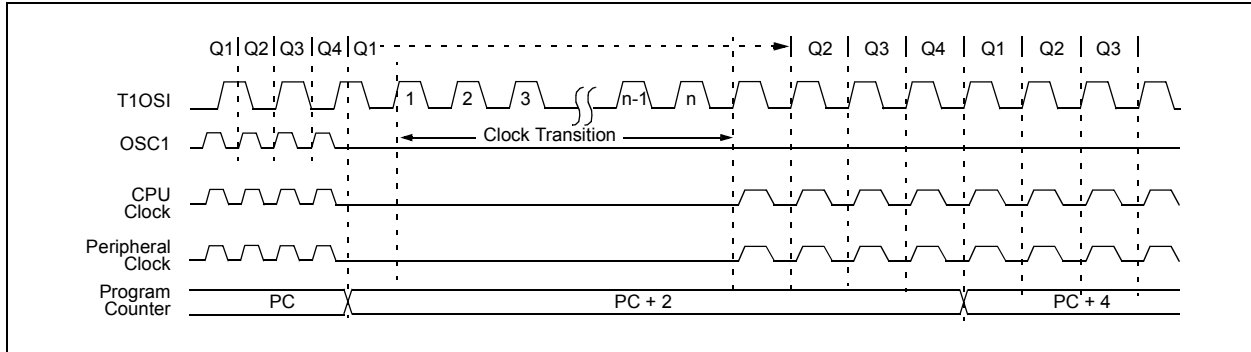
**Note:** The Timer1 oscillator should already be running prior to entering SEC\_RUN mode. If the T1OSCEN bit is not set when the SCS<1:0> bits are set to ‘01’, entry to SEC\_RUN mode will not occur. If the Timer1 oscillator is enabled, but not yet running, device clocks will be delayed until the oscillator has started. In such situations, initial oscillator operation is far from stable and unpredictable operation may result.

# PIC18F46J11 FAMILY

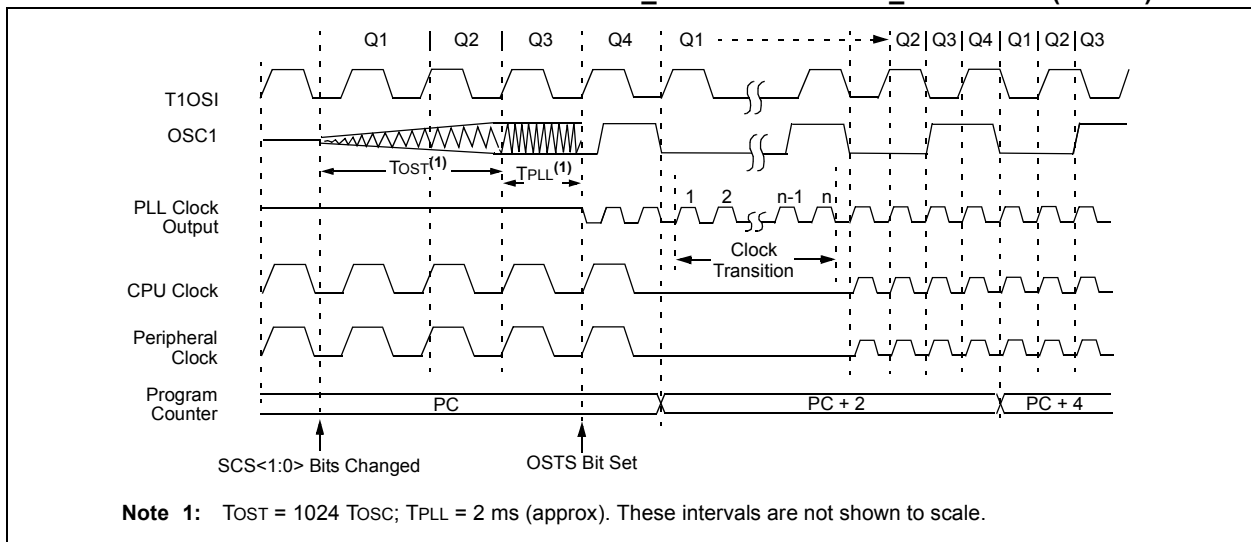
On transitions from SEC\_RUN mode to PRI\_RUN mode, the peripherals and CPU continue to be clocked from the Timer1 oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see

Figure 3-2). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock would be 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**



**FIGURE 3-2: TRANSITION TIMING FROM SEC\_RUN MODE TO PRI\_RUN MODE (HSPLL)**



# PIC18F46J11 FAMILY

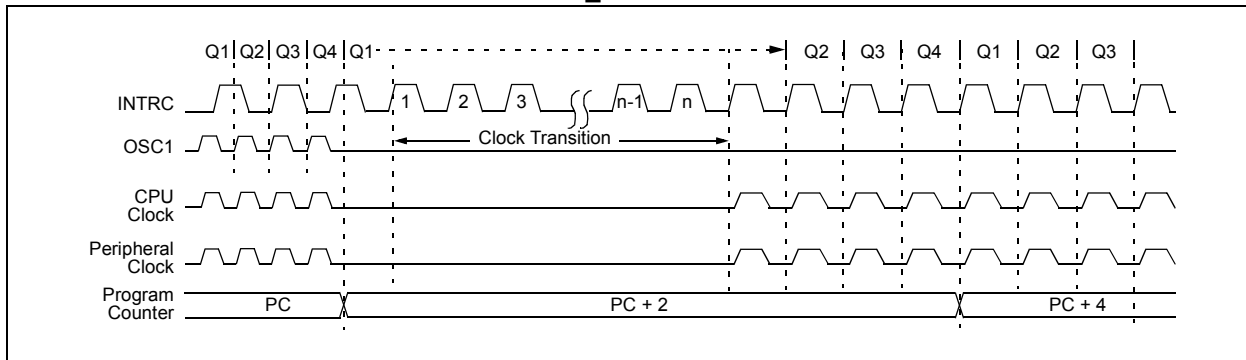
## 3.2.3 RC\_RUN MODE

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.

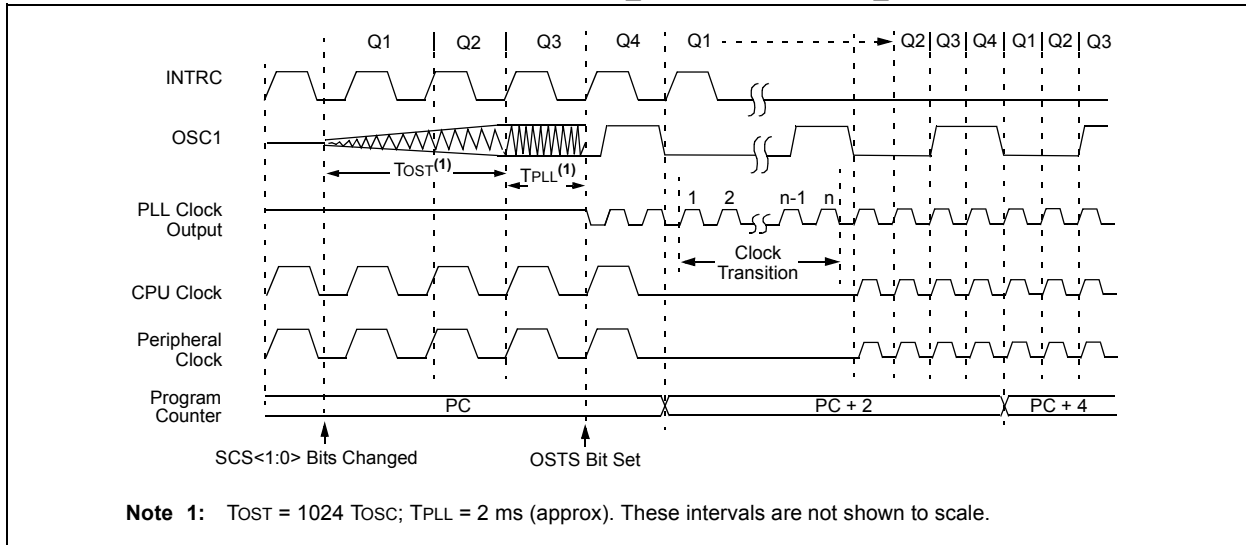
This mode is entered by setting the SCS<1:0> bits (OSCCON<1:0>) to '11'. When the clock source is switched to the internal oscillator block (see Figure 3-3), the primary oscillator is shut down and the OSTS bit is cleared.

On transitions from RC\_RUN mode to PRI\_RUN mode, the device continues to be clocked from the INTOSC block while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-4). 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 clock source will continue to run if either the WDT or the FSCM is enabled.

**FIGURE 3-3: TRANSITION TIMING TO RC\_RUN MODE**



**FIGURE 3-4: TRANSITION TIMING FROM RC\_RUN MODE TO PRI\_RUN MODE**



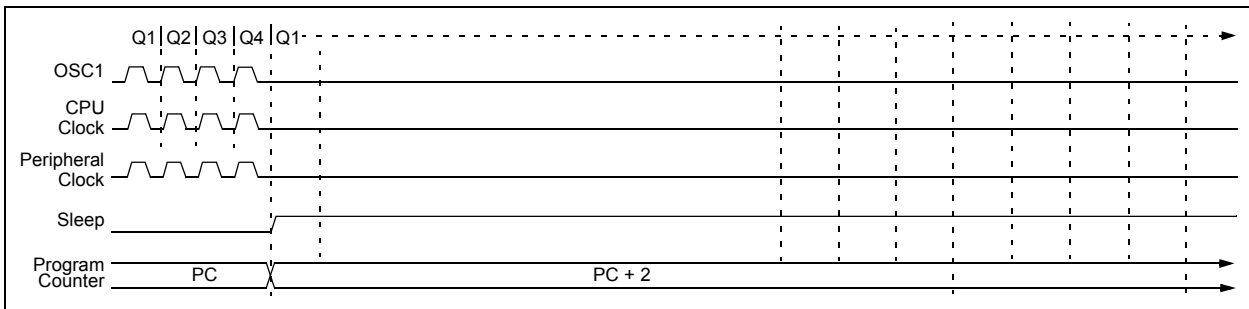
## 3.3 Sleep Mode

The power-managed Sleep mode is identical to the legacy Sleep mode offered in all other PIC 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 3-5). All clock source status bits are cleared.

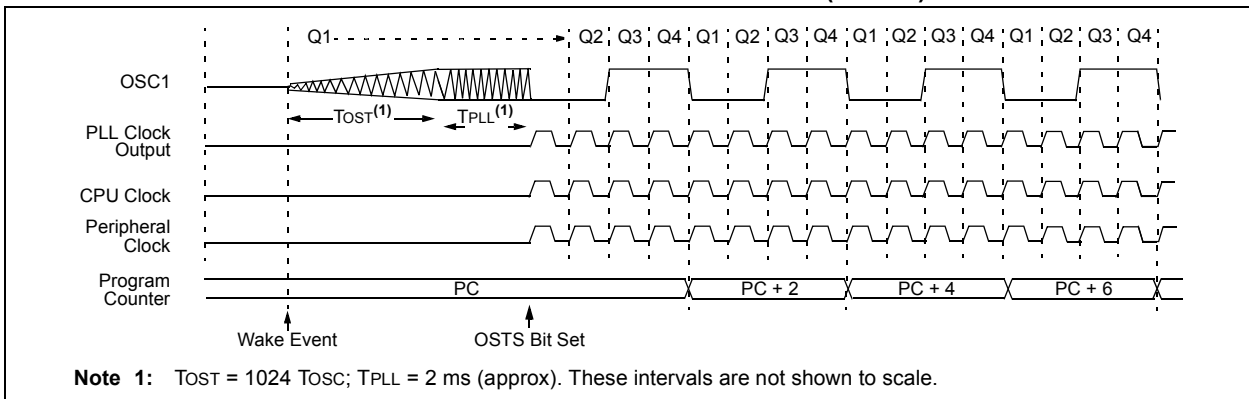
Entering the Sleep mode from any other mode does not require a clock switch. This is because no clocks are needed once the controller has entered Sleep mode. If the WDT is selected, the INTRC source will continue to operate. If the Timer1 oscillator is enabled, it will also continue to run.

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 `SCS<1:0>` bits becomes ready (see Figure 3-6), or it will be clocked from the internal oscillator if either the Two-Speed Start-up or the FSCM is enabled (see **Section 25.0 “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.

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



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



**Note 1:**  $T_{OST} = 1024 T_{OSC}$ ;  $T_{PLL} = 2 \text{ ms}$  (approx). These intervals are not shown to scale.

# PIC18F46J11 FAMILY

## 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 '1' when a `SLEEP` instruction is executed, the peripherals will be clocked from the clock source selected using the `SCS<1:0>` 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 `TcSD` (parameter 38, Table 28-13) 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 or Sleep mode, a WDT time-out will result in a WDT wake-up to the Run mode currently specified by the `SCS<1:0>` bits.

### 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 '00' and execute `SLEEP`. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the `FOSC<1:0>` Configuration bits. The OSTS bit remains set (see Figure 3-7).

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 3-8).

### 3.4.2 SEC\_IDLE MODE

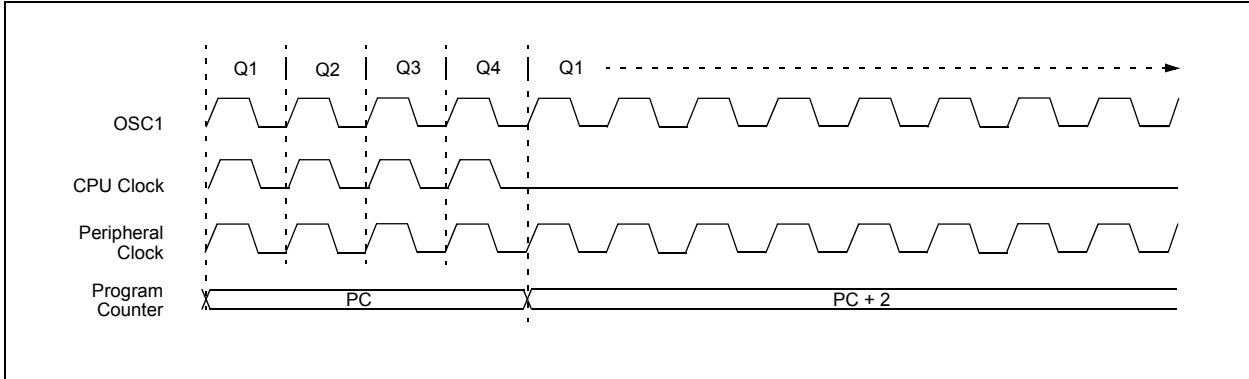
In `SEC_IDLE` mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered from `SEC_RUN` by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then set `SCS<1:0>` to '01' and execute `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 3-8).

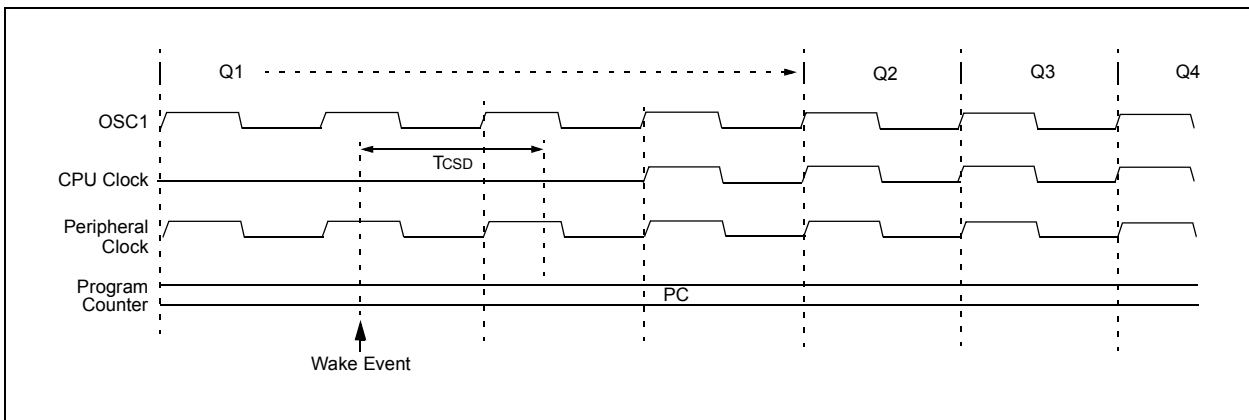
**Note:** The Timer1 oscillator should already be running prior to entering `SEC_IDLE` mode. If the `T1OSCEN` bit is not set when the `SLEEP` instruction is executed, the `SLEEP` instruction will be ignored and entry to `SEC_IDLE` mode will not occur. If the Timer1 oscillator is enabled, but not yet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

# PIC18F46J11 FAMILY

**FIGURE 3-7: TRANSITION TIMING FOR ENTRY TO IDLE MODE**



**FIGURE 3-8: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE**



# PIC18F46J11 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 block. 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 INTOSC block, the primary oscillator is shut down and the OSTS bit is cleared.

When a wake event occurs, the peripherals continue to be clocked from the internal oscillator block. After a delay of T<sub>CSD</sub> 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 FSCM 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<sub>CSD</sub>, following the wake event, is required when leaving Sleep and Idle modes. This delay is required for the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

### 3.5.2 EXIT BY WDT TIME-OUT

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

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

The WDT 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 FSCM 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
- PRI\_IDLE mode and the primary clock source is the ECPLL 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<sub>CSD</sub>, 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.

## 3.6 Deep Sleep Mode

Deep Sleep mode brings the device into its lowest power consumption state without requiring the use of external switches to remove power from the device. During deep sleep, the on-chip VDDCORE voltage regulator is powered down, effectively disconnecting power to the core logic of the microcontroller.

**Note:** Since Deep Sleep mode powers down the microcontroller by turning off the on-chip VDDCORE voltage regulator, Deep Sleep capability is available only on PIC18FXXJ members in the device family. The on-chip voltage regulator is not available in PIC18LFXXJ members of the device family, and therefore, they do not support Deep Sleep.



On devices that support it, the Deep Sleep mode is entered by:

- Setting the REGSLP (WDTCON<7>) bit (the default state on device Reset)
- Clearing the IDLEN bit (the default state on device Reset)
- Setting the DSEN bit (DSCONH<7>)
- Executing the `SLEEP` instruction immediately after setting DSEN (no delay in between)

In order to minimize the possibility of inadvertently entering Deep Sleep, the DSEN bit is cleared in hardware two instruction cycles after having been set. Therefore, in order to enter Deep Sleep, the `SLEEP` instruction must be executed in the immediate instruction cycle after setting DSEN. If DSEN is not set when Sleep is executed, the device will enter conventional Sleep mode instead.

During Deep Sleep, the core logic circuitry of the microcontroller is powered down to reduce leakage current. Therefore, most peripherals and functions of the microcontroller become unavailable during Deep Sleep. However, a few specific peripherals and functions are powered directly from the VDD supply rail of the microcontroller, and therefore, can continue to function in Deep Sleep.

Entering Deep Sleep mode clears the DSWAKEL register. However, if the Real-Time Clock and Calendar (RTCC) is enabled prior to entering Deep Sleep, it will continue to operate uninterrupted.

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

When a wake event occurs in Deep Sleep mode (by MCLR Reset, RTCC alarm, INT0 interrupt, ULPWU or DSWDT), the device will exit Deep Sleep mode and perform a Power-on Reset (POR). When the device is released from Reset, code execution will resume at the device's Reset vector.

### 3.6.1 PREPARING FOR DEEP SLEEP

Because VDDCORE could fall below the SRAM retention voltage while in Deep Sleep mode, SRAM data could be lost in Deep Sleep. Exiting Deep Sleep mode causes a POR; as a result, most Special Function Registers will reset to their default POR values.

Applications needing to save a small amount of data throughout a Deep Sleep cycle can save the data to the general purpose DSGPR0 and DSGPR1 registers. The contents of these registers are preserved while the device is in Deep Sleep, and will remain valid throughout an entire Deep Sleep entry and wake-up sequence.

### 3.6.2 I/O PINS DURING DEEP SLEEP

During Deep Sleep, the general purpose I/O pins will retain their previous states.

Pins that are configured as inputs (TRIS bit set) prior to entry into Deep Sleep will remain high-impedance during Deep Sleep.

Pins that are configured as outputs (TRIS bit clear) prior to entry into Deep Sleep will remain as output pins during Deep Sleep. While in this mode, they will drive the output level determined by their corresponding LAT bit at the time of entry into Deep Sleep.

When the device wakes back up, the I/O pin behavior depends on the type of wake-up source.

If the device wakes back up by an RTCC alarm, INTO interrupt, DSWDT or ULPWU event, all I/O pins will continue to maintain their previous states, even after the device has finished the POR sequence and is executing application code again. Pins configured as inputs during Deep Sleep will remain high-impedance, and pins configured as outputs will continue to drive their previous value.

After waking up, the TRIS and LAT registers will be reset, but the I/O pins will still maintain their previous states. If firmware modifies the TRIS and LAT values for the I/O pins, they will not immediately go to the newly configured states. Once the firmware clears the RELEASE bit (DSCONL<0>), the I/O pins will be "released". This causes the I/O pins to take the states configured by their respective TRIS and LAT bit values.

If the Deep Sleep BOR (DSBOR) circuit is enabled, and VDD drops below the DSBOR and VDD rail POR thresholds, the I/O pins will be immediately released similar to clearing the RELEASE bit. All previous state information will be lost, including the general purpose DSGPR0 and DSGPR1 contents. See **Section 3.6.5 "Deep Sleep Brown Out Reset (DSBOR)"** for additional details about this scenario.

If a MCLR Reset event occurs during Deep Sleep, the I/O pins will also be released automatically, but in this case, the DSGPR0 and DSGPR1 contents will remain valid.

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

# PIC18F46J11 FAMILY

---

## 3.6.3 DEEP SLEEP WAKE-UP SOURCES

While in Deep Sleep mode, the device can be awakened by a MCLR, POR, RTCC, INT0 I/O pin interrupt, DSWDT or ULPWU event. After waking, the device performs a POR. When the device is released from Reset, code execution will begin at the device's Reset vector.

The software can determine if the wake-up was caused from an exit from Deep Sleep mode by reading the DS bit (WDTCN<3>). If this bit is set, the POR was caused by a Deep Sleep exit. The DS bit must be manually cleared by the software.

The software can determine the wake event source by reading the DSWAKEH and DSWAKEL registers. When the application firmware is done using the DSWAKEH and DSWAKEL status registers, individual bits do not need to be manually cleared before entering Deep Sleep again. When entering Deep Sleep mode, these registers are automatically cleared.

### 3.6.3.1 Wake-up Event Considerations

Deep Sleep wake-up events are only monitored while the processor is fully in Deep Sleep mode. If a wake-up event occurs before Deep Sleep mode is entered, the event status will not be reflected in the DSWAKE registers. If the wake-up source asserts prior to entering Deep Sleep, the CPU may go to the interrupt vector (if the wake source has an interrupt bit and the interrupt is fully enabled), and may abort the Deep Sleep entry sequence by executing past the SLEEP instruction. In this case, a wake-up event handler should be placed after the SLEEP instruction to process the event and re-attempt entry into Deep Sleep if desired.

When the device is in Deep Sleep with more than one wake-up source simultaneously enabled, only the first wake-up source to assert will be detected and logged in the DSWAKEH/DSWAKEL status registers.

## 3.6.4 DEEP SLEEP WATCHDOG TIMER (DSWDT)

Deep Sleep has its own dedicated WDT (DSWDT) with a postscaler for time-outs of 2.1 ms to 25.7 days, configurable through the bits, DSWDTPS<3:0> (CONFIG3L<7:4>).

The DSWDT can be clocked from either the INTRC or the T1OSC/T1CKI input. If the T1OSC/T1CKI source will be used with a crystal, the T1OSCN bit in the T1CON register needs to be set prior to entering Deep Sleep. The reference clock source is configured through the DSWDTOSC bit (CONFIG3L<0>).

DSWDT is enabled through the DSWDTEN bit (CONFIG3L<3>). Entering Deep Sleep mode automatically clears the DSWDT. See **Section 25.0 “Special Features of the CPU”** for more information.

## 3.6.5 DEEP SLEEP BROWN OUT RESET (DSBOR)

The Deep Sleep module contains a dedicated Deep Sleep BOR (DSBOR) circuit. This circuit may be optionally enabled through the DSBOREN Configuration bit (CONFIG3L<2>).

The DSBOR circuit monitors the VDD supply rail voltage. The behavior of the DSBOR circuit is described in **Section 4.4 “Brown-out Reset (BOR)”**.

## 3.6.6 RTCC PERIPHERAL AND DEEP SLEEP

The RTCC can operate uninterrupted during Deep Sleep mode. It can wake the device from Deep Sleep by configuring an alarm.

The RTCC clock source is configured with the RTCOSC bit (CONFIG3L<1>). The available reference clock sources are the INTRC and T1OSC/T1CKI. If the INTRC is used, the RTCC accuracy will directly depend on the INTRC tolerance. For more information on configuring the RTCC peripheral, see **Section 16.0 “Real-Time Clock and Calendar (RTCC)”**.

## 3.6.7 TYPICAL DEEP SLEEP SEQUENCE

This section gives the typical sequence for using the Deep Sleep mode. Optional steps are indicated, and additional information is given in notes at the end of the procedure.

1. Enable DSWDT (optional).<sup>(1)</sup>
2. Configure DSWDT clock source (optional).<sup>(2)</sup>
3. Enable DSBOR (optional).<sup>(1)</sup>
4. Enable RTCC (optional).<sup>(3)</sup>
5. Configure the RTCC peripheral (optional).<sup>(3)</sup>
6. Configure the ULPWU peripheral (optional).<sup>(4)</sup>
7. Enable the INT0 Interrupt (optional).<sup>(4)</sup>
8. Context save SRAM data by writing to the DSGPR0 and DSGPR1 registers (optional).
9. Set the REGSLP bit (WDTCON<7>) and clear the IDLEN bit (OSCCON<7>).
10. If using an RTCC alarm for wake-up, wait until the RTCSYNC (RTCCCFG<4>) bit is clear.
11. Enter Deep Sleep mode by setting the DSEN bit (DSCONH<7>) and issuing a `SLEEP` instruction. These two instructions must be executed back to back.
12. Once a wake-up event occurs, the device will perform a POR reset sequence. Code execution resumes at the device's Reset vector.
13. Determine if the device exited Deep Sleep by reading the Deep Sleep bit, DS (WDTCON<3>). This bit will be set if there was an exit from Deep Sleep mode.
14. Clear the Deep Sleep bit, DS (WDTCON<3>).
15. Determine the wake-up source by reading the DSWAKEH and DSWAKEL registers.
16. Determine if a DSBOR event occurred during Deep Sleep mode by reading the DSBOR bit (DSCONL<1>).
17. Read the DSGPR0 and DSGPR1 context save registers (optional).
18. Clear the RELEASE bit (DSCONL<0>).

**Note 1:** DSWDT and DSBOR are enabled through the devices' Configuration bits. For more information, see **Section 25.1 "Configuration Bits"**.

**2:** The DSWDT and RTCC clock sources are selected through the devices' Configuration bits. For more information, see **Section 25.1 "Configuration Bits"**.

**3:** For more information, see **Section 16.0 "Real-Time Clock and Calendar (RTCC)"**.

**4:** For more information on configuring this peripheral, see **Section 3.7 "Ultra Low-Power Wake-up"**.

## 3.6.8 DEEP SLEEP FAULT DETECTION

If during Deep Sleep the device is subjected to unusual operating conditions, such as an Electrostatic Discharge (ESD) event, it is possible that the internal circuit states used by the Deep Sleep module could become corrupted. If this were to happen, the device may exhibit unexpected behavior, such as a failure to wake back up.

In order to prevent this type of scenario from occurring, the Deep Sleep module includes automatic self-monitoring capability. During Deep Sleep, critical internal nodes are continuously monitored in order to detect possible Fault conditions (which would not ordinarily occur). If a Fault condition is detected, the circuitry will set the DSFLT status bit (DSWAKEL<7>) and automatically wake the microcontroller from Deep Sleep, causing a POR Reset.

During Deep Sleep, the Fault detection circuitry is always enabled and does not require any specific configuration prior to entering Deep Sleep.

# PIC18F46J11 FAMILY

## 3.6.9 DEEP SLEEP MODE REGISTERS

Deep Sleep mode registers are provided in Register 3-1 through Register 3-6.

### REGISTER 3-1: DSCONH: DEEP SLEEP CONTROL HIGH BYTE REGISTER (BANKED F4Dh)

R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
DSEN <sup>(1)</sup>	—	—	—	—	(Reserved)	DSULPEN	RTCWDIS
bit 7							bit 0

#### Legend:

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

- bit 7                      **DSEN:** Deep Sleep Enable bit<sup>(1)</sup>  
                                  1 = Deep Sleep mode is entered on a SLEEP command  
                                  0 = Sleep mode is entered on a SLEEP command
- bit 6-3                      **Unimplemented:** Read as '0'
- bit 2                      **(Reserved):** Always write '0' to this bit
- bit 1                      **DSULPEN:** Ultra Low-Power Wake-up Module Enable bit  
                                  1 = ULPWU module is enabled in Deep Sleep  
                                  0 = ULPWU module is disabled in Deep Sleep
- bit 0                      **RTCWDIS:** RTCC Wake-up Disable bit  
                                  1 = Wake-up from RTCC is disabled  
                                  0 = Wake-up from RTCC is enabled

**Note 1:** In order to enter Deep Sleep, Sleep must be executed immediately after setting DSEN.

### REGISTER 3-2: DSCONL: DEEP SLEEP CONTROL LOW BYTE REGISTER (BANKED F4Ch)

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>
—	—	—	—	—	ULPWDIS	DSBOR	RELEASE
bit 7							bit 0

#### Legend:

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

- bit 7-3                      **Unimplemented:** Read as '0'
- bit 2                      **ULPWDIS:** Ultra Low-Power Wake-up Disable bit  
                                  1 = ULPWU wake-up source is disabled  
                                  0 = ULPWU wake-up source is enabled (must also set DSULPEN = 1)
- bit 1                      **DSBOR:** Deep Sleep BOR Event Status bit  
                                  1 = DSBOR was enabled and VDD dropped below the DSBOR arming voltage during Deep Sleep, but did not fall below VDSBOR  
                                  0 = DSBOR was disabled or VDD did not drop below the DSBOR arming voltage during Deep Sleep
- bit 0                      **RELEASE:** I/O Pin State Release bit  
                                  Upon waking from Deep Sleep, the I/O pins maintain their previous states. Clearing this bit will release the I/O pins and allow their respective TRIS and LAT bits to control their states.

**Note 1:** This is the value when VDD is initially applied.

# PIC18F46J11 FAMILY

## REGISTER 3-3: DSGPR0: DEEP SLEEP PERSISTENT GENERAL PURPOSE REGISTER 0 (BANKED F4Eh)

R/W-xxxx <sup>(1)</sup>	
Deep Sleep Persistent General Purpose bits	
bit 7	bit 0

### Legend:

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

bit 7-0      Deep Sleep Persistent General Purpose bits  
                 Contents are retained even in Deep Sleep mode.

**Note 1:** All register bits are maintained unless: VDDCORE drops below the normal BOR threshold outside of Deep Sleep, or the device is in Deep Sleep and the dedicated DSBOR is enabled and VDD drops below the DSBOR threshold, or DSBOR is enabled or disabled, but VDD is hard cycled to near Vss.

## REGISTER 3-4: DSGPR1: DEEP SLEEP PERSISTENT GENERAL PURPOSE REGISTER 1 (BANKED F4Fh)

R/W-xxxx <sup>(1)</sup>	
Deep Sleep Persistent General Purpose bits	
bit 7	bit 0

### Legend:

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

bit 7-0      Deep Sleep Persistent General Purpose bits  
                 Contents are retained even in Deep Sleep mode.

**Note 1:** All register bits are maintained unless: VDDCORE drops below the normal BOR threshold outside of Deep Sleep, or the device is in Deep Sleep and the dedicated DSBOR is enabled and VDD drops below the DSBOR threshold, or DSBOR is enabled or disabled, but VDD is hard cycled to near Vss.

# PIC18F46J11 FAMILY

## REGISTER 3-5: DSWAKEH: DEEP SLEEP WAKE HIGH BYTE REGISTER (BANKED F4Bh)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	DSINT0
bit 7							bit 0

### Legend:

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

bit 7-1                      **Unimplemented:** Read as '0'  
 bit 0                      **DSINT0:** Interrupt-on-Change bit  
                                  1 = Interrupt-on-change was asserted during Deep Sleep  
                                  0 = Interrupt-on-change was not asserted during Deep Sleep

## REGISTER 3-6: DSWAKEL: DEEP SLEEP WAKE LOW BYTE REGISTER (BANKED F4Ah)

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-1
DSFLT	—	DSULP <sup>(2)</sup>	DSWDT <sup>(2)</sup>	DSRTC <sup>(2)</sup>	DSMCLR <sup>(2)</sup>	—	DSPOR
bit 7							bit 0

### Legend:

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

bit 7                      **DSFLT:** Deep Sleep Fault Detected bit  
                                  1 = A Deep Sleep Fault was detected during Deep Sleep  
                                  0 = A Deep Sleep fault was not detected during Deep Sleep

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

bit 5                      **DSULP:** Ultra Low-Power Wake-up status bit<sup>(2)</sup>  
                                  1 = An Ultra Low-Power Wake-up event occurred during Deep Sleep  
                                  0 = An Ultra Low-Power Wake-up event did not occur during Deep Sleep

bit 4                      **DSWDT:** Deep Sleep Watchdog Timer Time-out bit<sup>(2)</sup>  
                                  1 = The Deep Sleep Watchdog Timer timed out during Deep Sleep  
                                  0 = The Deep Sleep Watchdog Timer did not time out during Deep Sleep

bit 3                      **DSRTC:** Real-Time Clock and Calendar Alarm bit<sup>(2)</sup>  
                                  1 = The Real-Time Clock/Calendar triggered an alarm during Deep Sleep  
                                  0 = The Real-Time Clock /Calendar did not trigger an alarm during Deep Sleep

bit 2                      **DSMCLR:**  $\overline{\text{MCLR}}$  Event bit<sup>(2)</sup>  
                                  1 = The  $\overline{\text{MCLR}}$  pin was asserted during Deep Sleep  
                                  0 = The  $\overline{\text{MCLR}}$  pin was not asserted during Deep Sleep

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

bit 0                      **DSPOR:** Power-on Reset Event bit  
                                  1 = The VDD supply POR circuit was active and a POR event was detected<sup>(1)</sup>  
                                  0 = The VDD supply POR circuit was not active, or was active, but did not detect a POR event

**Note 1:** Unlike the other bits in this register, this bit can be set outside of Deep Sleep.  
**Note 2:** If multiple wake-up triggers are fired around the same time, only the first wake-up event triggered will have its wake-up status bit set.

## 3.7 Ultra Low-Power Wake-up

The Ultra Low-Power Wake-up (ULPWU) on RA0 allows a slow falling voltage to generate an interrupt without excess current consumption.

Follow these steps to use this feature:

1. Configure a remappable output pin to output the ULPOUT signal.
2. Map an INTx interrupt-on-change input function to the same pin as used for the ULPOUT output function. Alternatively, in step 1, configure ULPOUT to output onto a PORTB interrupt-on-change pin.
3. Charge the capacitor on RA0 by configuring the RA0 pin to an output and setting it to '1'.
4. Enable interrupt for the corresponding pin selected in step 2.
5. Stop charging the capacitor by configuring RA0 as an input.
6. Discharge the capacitor by setting the ULPEN and ULPSINK bits in the WDTCON register.
7. Configure Sleep mode.
8. Enter Sleep mode.

When the voltage on RA0 drops below  $V_{IL}$ , an interrupt will be generated, which will cause the device to wake-up and execute the next instruction.

This feature provides a low-power technique for periodically waking up the device from Sleep mode. The time-out is dependent on the discharge time of the RC circuit on RA0.

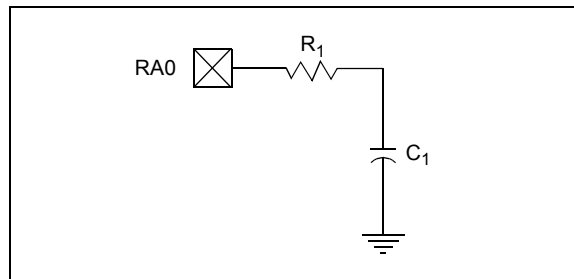
When the ULPWU module causes the device to wake-up from Sleep mode, the WDTCON<ULPLVL> bit is set. When the ULPWU module causes the device to wake-up from Deep Sleep, the DSULP (DSWAKEL<5>) bit is set. Software can check these bits upon wake-up to determine the wake-up source. Also in Sleep mode, only the remappable output function, ULPWU, will output this bit value to an RPN pin for externally detecting wake-up events.

See Example 3-1 for initializing the ULPWU module.

**Note:** For module-related bit definitions, see the WDTCON register in **Section 25.2 “Watchdog Timer (WDT)”** and the DSWAKEL register (Register 3-6).

A series resistor between RA0 and the external capacitor provides overcurrent protection for the RA0/AN0/C1INA/ULPWU/RP0 pin and can allow for software calibration of the time-out (see Figure 3-9).

**FIGURE 3-9: SERIAL RESISTOR**



A timer can be used to measure the charge time and discharge time of the capacitor. The charge time can then be adjusted to provide the desired interrupt delay. This technique will compensate for the affects of temperature, voltage and component accuracy. The peripheral can also be configured as a simple Programmable Low-Voltage Detect (LVD) or temperature sensor.

**Note:** For more information, refer to AN879, “Using the Microchip Ultra Low-Power Wake-up Module” application note (DS00879).

# PIC18F46J11 FAMILY

## EXAMPLE 3-1: ULTRA LOW-POWER WAKE-UP INITIALIZATION

```

//*****
//Configure a remappable output pin with interrupt capability
//for ULPWU function (RP21 => RD4/INT1 in this example)
//*****
RPOR21 = 13;// ULPWU function mapped to RP21/RD4
RPINR1 = 21;// INT1 mapped to RP21 (RD4)

//*****
//Charge the capacitor on RA0
//*****
TRISAbits.TRISA0 = 0;
PORTAbits.RA0 = 1;
for(i = 0; i < 10000; i++) Nop();

//*****
//Stop Charging the capacitor on RA0
//*****
TRISAbits.TRISA0 = 1;

//*****
//Enable the Ultra Low Power Wakeup module
//and allow capacitor discharge
//*****
WDTCONbits.ULPEN = 1;
WDTCONbits.ULPSINK = 1;

//*****
//For Sleep, Enable Interrupt for ULPW.
//*****
INTCON3bits.INT1IF = 0;
INTCON3bits.INT1IE = 1;

//*****
//Configure Sleep Mode
//*****
//For Sleep
OSCCONbits.IDLEN = 0;

//For Deep Sleep
OSCCONbits.IDLEN = 0;// enable deep sleep
DSCONHbits.DSEN = 1;// Note: must be set just before executing Sleep();
//*****
//Enter Sleep Mode
//*****
Sleep();
    // for sleep, execution will resume here
    // for deep sleep, execution will restart at reset vector (use WDTCONbits.DS to detect)

```



# PIC18F46J11 FAMILY

## 4.0 RESET

The PIC18F46J11 family of devices differentiates among 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)
- Configuration Mismatch (CM)
- Brown-out Reset (BOR)
- RESET Instruction
- Stack Full Reset
- Stack Underflow Reset
- Deep Sleep Reset

This section discusses Resets generated by  $\overline{\text{MCLR}}$ , POR and BOR, and covers the operation of the various start-up timers.

For information on WDT Resets, see **Section 25.2 “Watchdog Timer (WDT)”**. For Stack Reset events, see **Section 5.1.4.4 “Stack Full and Underflow Resets”** and for Deep Sleep mode, see **Section 3.6 “Deep Sleep Mode”**.

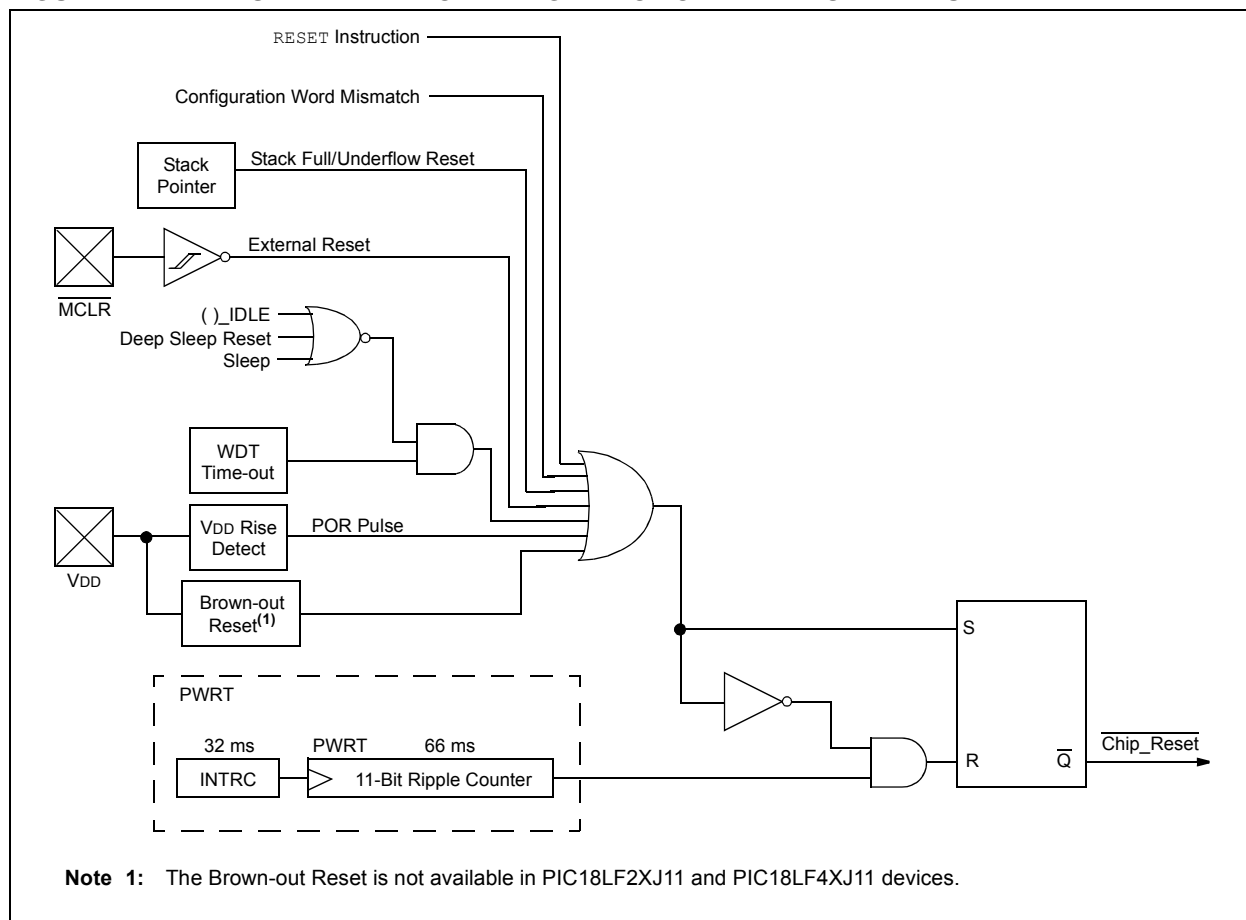
Figure 4-1 provides a simplified block diagram of the on-chip Reset circuit.

## 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 4.7 “Reset State of Registers”**.

The ECON 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**



# PIC18F46J11 FAMILY

## REGISTER 4-1: RCON: RESET CONTROL REGISTER (ACCESS FD0h)

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

### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

- bit 7      **IPEN:** Interrupt Priority Enable bit  
           1 = Enable priority levels on interrupts  
           0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)
- bit 6      **Unimplemented:** Read as '0'
- bit 5       **$\overline{\text{CM}}$ :** Configuration Mismatch Flag bit  
           1 = A Configuration Mismatch Reset has not occurred  
           0 = A Configuration Mismatch Reset has occurred (must be set in software after a Configuration Mismatch Reset occurs)
- bit 4       **$\overline{\text{RI}}$ :** 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       **$\overline{\text{TO}}$ :** Watchdog Time-out Flag bit  
           1 = Set by power-up, CLRWD $\overline{\text{T}}$  instruction or SLEEP instruction  
           0 = A WDT time-out occurred
- bit 2       **$\overline{\text{PD}}$ :** Power-Down Detection Flag bit  
           1 = Set by power-up or by the CLRWD $\overline{\text{T}}$  instruction  
           0 = Set by execution of the SLEEP instruction
- bit 1       **$\overline{\text{POR}}$ :** 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       **$\overline{\text{BOR}}$ :** 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 1:** It is recommended that the  $\overline{\text{POR}}$  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,  $\overline{\text{BOR}}$  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  $\overline{\text{BOR}}$  is '0' and  $\overline{\text{POR}}$  is '1' (assuming that  $\overline{\text{POR}}$  was set to '1' by software immediately after a Power-on Reset).

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

The Master Clear Reset ( $\overline{\text{MCLR}}$ ) 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 POR condition is generated on-chip whenever VDD rises above a certain threshold. This allows the device to start in the initialized state when VDD is adequate for operation.

To take advantage of the POR circuitry, tie the  $\overline{\text{MCLR}}$  pin through a resistor (1 k $\Omega$  to 10 k $\Omega$ ) to VDD. This will eliminate external RC components usually needed to create a POR delay.

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 Power-on Reset occurs; it does not change for any other Reset event.  $\overline{\text{POR}}$  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)

"F" devices incorporate two types of BOR circuits: one which monitors VDDCORE and one which monitors VDD. Only one BOR circuit can be active at a time. When in normal Run mode, Idle or normal Sleep modes, the BOR circuit that monitors VDDCORE is active and will cause the device to be held in BOR if VDDCORE drops below VBOR (parameter D005). Once VDDCORE rises back above VBOR, the device will be held in Reset until the expiration of the Power-up Timer, with period, TPWRT (parameter 33).

During Deep Sleep operation, the on-chip core voltage regulator is disabled and VDDCORE is allowed to drop to ground levels. If the Deep Sleep BOR circuit is enabled by the DSBOREN Configuration bit (CONFIG3L<2> = 1), it will monitor VDD. If VDD drops below the VDSBOR threshold, the device will be held in a Reset state similar to POR. All registers will be set back to their POR Reset values and the contents of the DSGPR0 and DSGPR1 holding registers will be lost.

Additionally, if any I/O pins had been configured as outputs during Deep Sleep, these pins will be tri-stated and the device will no longer be held in Deep Sleep. Once the VDD voltage recovers back above the VDSBOR threshold, and once the core voltage regulator achieves a VDDCORE voltage above VBOR, the device will begin executing code again normally, but the DS bit in the WDTCON register will not be set. The device behavior will be similar to hard cycling all power to the device.

On "LF" devices, the VDDCORE BOR circuit is always disabled because the internal core voltage regulator is disabled. Instead of monitoring VDDCORE, PIC18LF devices in this family can use the VDD BOR circuit to monitor VDD excursions below the VDSBOR threshold. The VDD BOR circuit can be disabled by setting the DSBOREN bit = 0.

The VDD BOR circuit is enabled when DSBOREN = 1 on "LF" devices, or on "F" devices while in Deep Sleep with DSBOREN = 1. When enabled, the VDD BOR circuit is extremely low power (typ. 200 nA) during normal operation above ~2.3V on VDD. If VDD drops below this DSBOR arming level when the VDD BOR circuit is enabled, the device may begin to consume additional current (typ. 50  $\mu$ A) as internal features of the circuit power up. The higher current is necessary to achieve more accurate sensing of the VDD level. However, the device will not enter Reset until VDD falls below the VDSBOR threshold.

### 4.4.1 DETECTING BOR

The  $\overline{\text{BOR}}$  bit always resets to '0' on any VDDCORE, BOR or POR event. This makes it difficult to determine if a Brown-out Reset 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 Power-on Reset event. If  $\overline{\text{BOR}}$  is '0' while  $\overline{\text{POR}}$  is '1', it can be reliably assumed that a Brown-out Reset event has occurred.

If the voltage regulator is disabled (LF devices), the VDDCORE BOR functionality is disabled. In this case, the  $\overline{\text{BOR}}$  bit cannot be used to determine a Brown-out Reset event. The  $\overline{\text{BOR}}$  bit is still cleared by a Power-on Reset event.

# PIC18F46J11 FAMILY

## 4.5 Configuration Mismatch (CM)

The Configuration Mismatch (CM) Reset is designed to detect, and attempt to recover from, random memory corrupting events. These include Electrostatic Discharge (ESD) events, which can cause widespread single bit changes throughout the device and result in catastrophic failure.

In PIC18FXXJ Flash devices, the device Configuration registers (located in the configuration memory space) are continuously monitored during operation by comparing their values to complimentary shadow registers. If a mismatch is detected between the two sets of registers, a CM Reset automatically occurs. These events are captured by the  $\overline{\text{CM}}$  bit (RCON<5>). The state of the bit is set to '0' whenever a CM event occurs; it does not change for any other Reset event.

A CM Reset behaves similarly to a  $\overline{\text{MCLR}}$ ,  $\overline{\text{RESET}}$  instruction, WDT time-out or Stack Event Resets. As with all hard and power Reset events, the device Configuration Words are reloaded from the Flash Configuration Words in program memory as the device restarts.

## 4.6 Power-up Timer (PWRT)

PIC18F46J11 family devices incorporate an on-chip PWRT to help regulate the POR 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 PIC18F46J11 family devices is a 5-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of  $32 \times 32 \mu\text{s} = 1 \text{ ms}$ . While the PWRT is counting, the device is held in Reset.

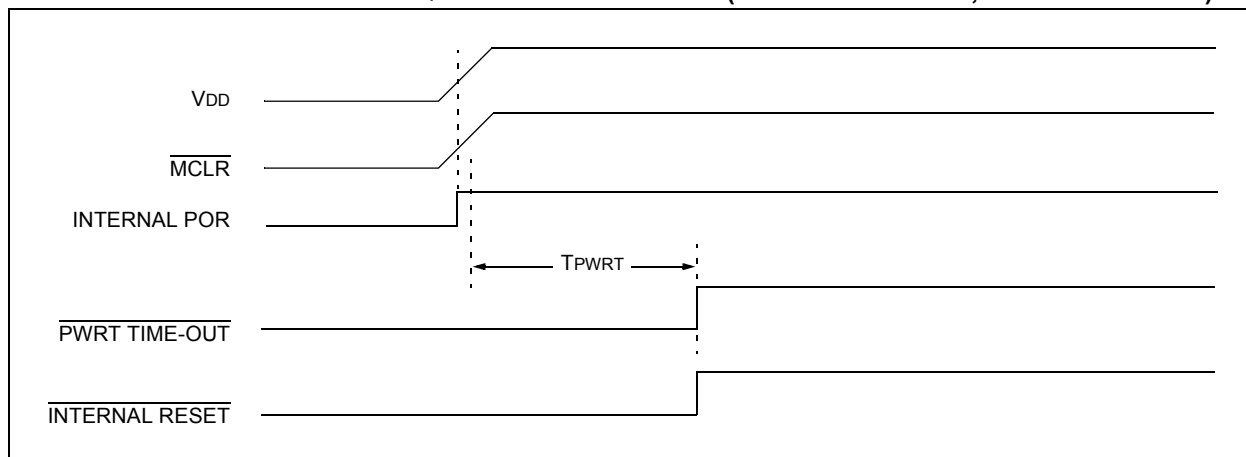
The power-up time delay depends on the INTRC clock and will vary from chip-to-chip due to temperature and process variation. See DC parameter 33 (TPWRT) for details.

### 4.6.1 TIME-OUT SEQUENCE

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 4-2, Figure 4-3, Figure 4-4 and Figure 4-5 all depict time-out sequences on power-up with the PWRT.

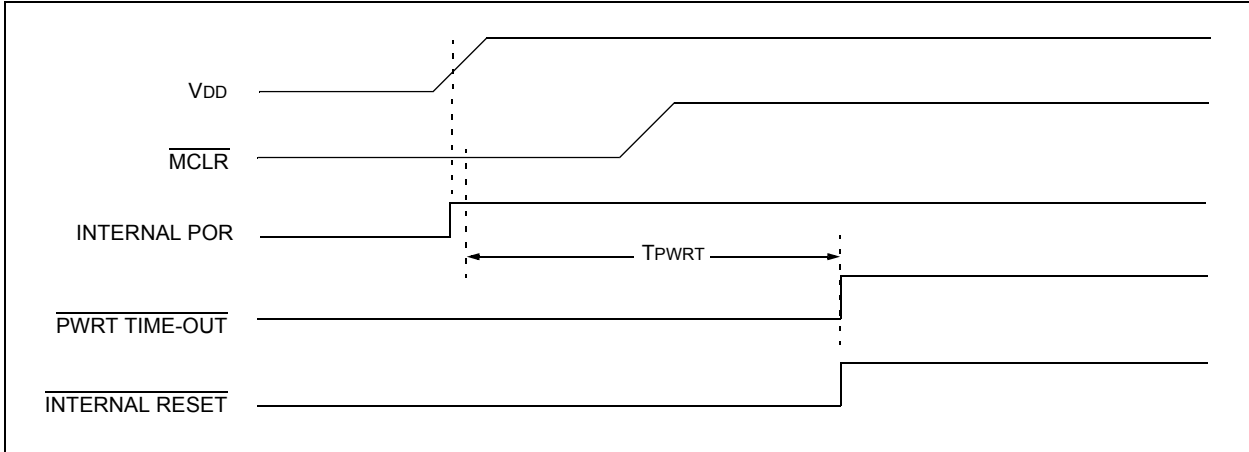
Since the time-outs occur from the POR pulse, if  $\overline{\text{MCLR}}$  is kept low long enough, the PWRT will expire. Bringing  $\overline{\text{MCLR}}$  high will begin execution immediately if a clock source is available (Figure 4-4). This is useful for testing purposes, or to synchronize more than one PIC18FXXX device operating in parallel.

**FIGURE 4-2: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{\text{DD}}$ ,  $V_{\text{DD}}$  RISE <  $T_{\text{PWRT}}$ )**

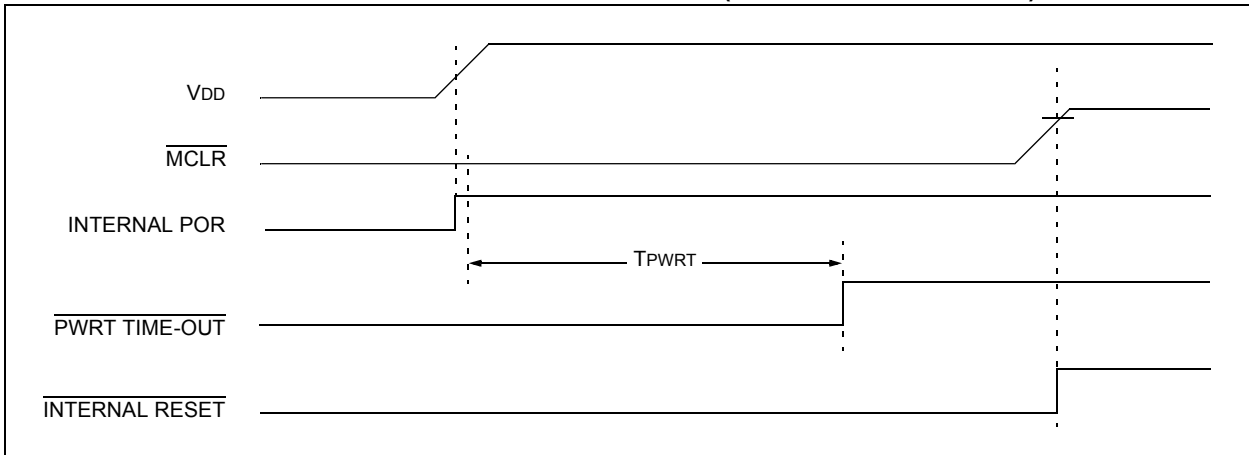


# PIC18F46J11 FAMILY

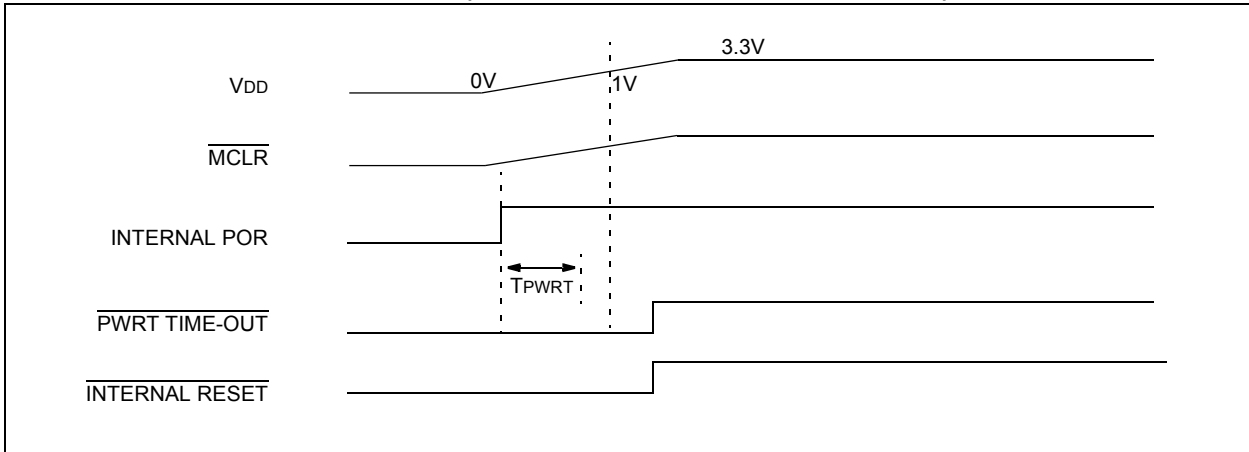
**FIGURE 4-3: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{\text{DD}}$ ): CASE 1**



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



**FIGURE 4-5: SLOW RISE TIME ( $\overline{\text{MCLR}}$  TIED TO  $V_{\text{DD}}$ ,  $V_{\text{DD}}$  RISE  $>$   $T_{\text{PWRT}}$ )**



# PIC18F46J11 FAMILY

## 4.7 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 ( $\overline{CM}$ ,  $\overline{RI}$ ,

$\overline{TO}$ ,  $\overline{PD}$ ,  $\overline{POR}$  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 POR and BOR,  $\overline{MCLR}$  and WDT Resets, and WDT wake-ups.

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

Condition	Program Counter <sup>(1)</sup>	RCON Register						STKPTR Register	
		$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
Power-on Reset	0000h	1	1	1	1	0	0	0	0
RESET instruction	0000h	u	0	u	u	u	u	u	u
Brown-out Reset	0000h	1	1	1	1	u	0	u	u
Configuration Mismatch Reset	0000h	0	u	u	u	u	u	u	u
$\overline{MCLR}$ Reset during power-managed Run modes	0000h	u	u	1	u	u	u	u	u
$\overline{MCLR}$ Reset during power-managed Idle modes and Sleep mode	0000h	u	u	1	0	u	u	u	u
$\overline{MCLR}$ Reset during full-power execution	0000h	u	u	u	u	u	u	u	u
Stack Full Reset (STVREN = 1)	0000h	u	u	u	u	u	u	1	u
Stack Underflow Reset (STVREN = 1)	0000h	u	u	u	u	u	u	u	1
Stack Underflow Error (not an actual Reset, STVREN = 0)	0000h	u	u	u	u	u	u	u	1
WDT time-out during full-power or power-managed Run modes	0000h	u	u	0	u	u	u	u	u
WDT time-out during power-managed Idle or Sleep modes	PC + 2	u	u	0	0	u	u	u	u
Interrupt exit from power-managed modes	PC + 2	u	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 bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

# PIC18F46J11 FAMILY

**TABLE 4-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
TOSU	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---0 0000 <sup>(1)</sup>
TOSH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu <sup>(1)</sup>
TOSL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu <sup>(1)</sup>
STKPTR	PIC18F2XJ11	PIC18F4XJ11	00-0 0000	uu-0 0000	uu-u uuuu <sup>(1)</sup>
PCLATU	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	PC + 2 <sup>(2)</sup>
TBLPTRU	PIC18F2XJ11	PIC18F4XJ11	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F2XJ11	PIC18F4XJ11	0000 000x	0000 000u	uuuu uuuu <sup>(3)</sup>
INTCON2	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu <sup>(3)</sup>
INTCON3	PIC18F2XJ11	PIC18F4XJ11	1100 0000	1100 0000	uuuu uuuu <sup>(3)</sup>
INDF0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTINC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTDEC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PREINC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PLUSW0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
FSR0H	PIC18F2XJ11	PIC18F4XJ11	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTINC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTDEC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PREINC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PLUSW1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
FSR1H	PIC18F2XJ11	PIC18F4XJ11	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
POSTINC2	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTDEC2	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PREINC2	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PLUSW2	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
FSR2H	PIC18F2XJ11	PIC18F4XJ11	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F2XJ11	PIC18F4XJ11	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F2XJ11	PIC18F4XJ11	0110 q000	0110 q000	0110 q00u
CM1CON	PIC18F2XJ11	PIC18F4XJ11	0001 1111	uuuu uuuu	uuuu uuuu
CM2CON	PIC18F2XJ11	PIC18F4XJ11	0001 1111	uuuu uuuu	uuuu uuuu
RCON <sup>(4)</sup>	PIC18F2XJ11	PIC18F4XJ11	0-11 1100	0-qq qquu	u-qq qquu
TMR1H	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F2XJ11	PIC18F4XJ11	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
T2CON	PIC18F2XJ11	PIC18F4XJ11	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP1MSK	PIC18F2XJ11	PIC18F4XJ11	1111 1111	uuuu uuuu	uuuu uuuu
SSP1STAT	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
ADCON1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
WDTCON	PIC18F2XJ11	PIC18F4XJ11	1qq- 0000	0qq- 0000	uqq- uuuu
PSTR1CON	PIC18F2XJ11	PIC18F4XJ11	00-0 0001	00-0 0001	uu-u uuuu
ECCP1AS	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.



# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
ECCP1DEL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
CCPR1H	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PSTR2CON	PIC18F2XJ11	PIC18F4XJ11	00-0 0001	00-0 0001	uu-u uuuu
ECCP2AS	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
CTMUCONH	PIC18F2XJ11	PIC18F4XJ11	0-00 000-	0-00 000-	u-uu uuu-
CTMUCONL	PIC18F2XJ11	PIC18F4XJ11	0000 00xx	0000 00xx	uuuu uuuu
CTMUICON	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SPBRG1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXSTA1	PIC18F2XJ11	PIC18F4XJ11	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F2XJ11	PIC18F4XJ11	0000 000x	0000 000x	uuuu uuuu
SPBRG2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F2XJ11	PIC18F4XJ11	0000 0010	0000 0010	uuuu uuuu
EECON2	PIC18F2XJ11	PIC18F4XJ11	---- ----	---- ----	---- ----
EECON1	PIC18F2XJ11	PIC18F4XJ11	--00 x00-	--00 u00-	--00 u00-
IPR3	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
PIR3	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu <sup>(3)</sup>
PIE3	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
PIR2	PIC18F2XJ11	PIC18F4XJ11	000- 0000	000- 0000	uuu- uuuu <sup>(3)</sup>
PIE2	PIC18F2XJ11	PIC18F4XJ11	000- 0000	000- 0000	uuu- uuuu
IPR1	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu <sup>(3)</sup>
PIE1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
RCSTA2	PIC18F2XJ11	PIC18F4XJ11	0000 000x	0000 000x	uuuu uuuu
OSCTUNE	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
T1GCON	PIC18F2XJ11	PIC18F4XJ11	0000 0x00	0000 0x00	uuuu uuuu
RTCVALH	PIC18F2XJ11	PIC18F4XJ11	0xxx xxxx	0uuu uuuu	0uuu uuuu
RTCVALL	PIC18F2XJ11	PIC18F4XJ11	0xxx xxx	0uuu uuuu	0uuu uuuu
T3GCON	PIC18F2XJ11	PIC18F4XJ11	0000 0x00	uuuu uxuu	uuuu uxuu
TRISE <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	---- -111	---- -111	---- -uuu
TRISD <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
TRISA	PIC18F2XJ11	PIC18F4XJ11	111- 1111	111- 1111	uuu- uuuu
ALRMCFG	PIC18F2XJ11	PIC18F4XJ11	0000 0000	uuuu uuuu	uuuu uuuu
ALMRPT	PIC18F2XJ11	PIC18F4XJ11	0000 0000	uuuu uuuu	uuuu uuuu
ALRMVALH	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
ALRMVALL	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	---- -xxx	---- -uuu	---- -uuu
LATD <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	PIC18F2XJ11	PIC18F4XJ11	xxx- xxxx	uuu- uuuu	uuu- uuuu
DMACON1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
DMACON2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
HLVDCON	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PORTE <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	00-- -xxx	uu-- -uuu	uu-- -uuu
PORTD <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	PIC18F2XJ11	PIC18F4XJ11	xxx- xxxx	uuu- uuuu	uuu- uuuu
SPBRGH1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F2XJ11	PIC18F4XJ11	0100 0-00	0100 0-00	uuuu u-uu
SPBRGH2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
BAUDCON2	PIC18F2XJ11	PIC18F4XJ11	0100 0-00	0100 0-00	uuuu u-uu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
TMR3H	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F2XJ11	PIC18F4XJ11	0000 -000	uuuu -uuu	uuuu -uuu
TMR4	PIC18F2XJ11	PIC18F4XJ11	0000 0000	uuuu uuuu	uuuu uuuu
PR4	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
T4CON	PIC18F2XJ11	PIC18F4XJ11	-000 0000	-000 0000	-uuu uuuu
SSP2BUF	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP2ADD	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP2MASK	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP2STAT	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu
SSP2CON1	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
SSP2CON2	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
CMSTAT	PIC18F2XJ11	PIC18F4XJ11	---- --11	---- --11	---- --uu
PMADDRH <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	-000 0000	-000 0000	-uuu uuuu
PMDOUT1H <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMADDRL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDOUT1L	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDIN1H	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDIN1L	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TXADDRL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TXADDRH	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu
RXADDRL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
RXADDRH	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu
DMABCL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
DMABCH	PIC18F2XJ11	PIC18F4XJ11	---- --00	---- --00	---- --uu
PMCONH	PIC18F2XJ11	PIC18F4XJ11	0-00 0000	0-00 0000	u-uu uuuu
PMCONL	PIC18F2XJ11	PIC18F4XJ11	000- 0000	000- 0000	uuu- uuuu
PMMODEH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMMODEL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDOUT2H	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDOUT2L	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDIN2H	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMDIN2L	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
PMEH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMEL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PMSTATH	PIC18F2XJ11	PIC18F4XJ11	00-- 0000	00-- 0000	uu-- uuuu
PMSTATL	PIC18F2XJ11	PIC18F4XJ11	10-- 1111	10-- 1111	uu-- uuuu
CVRCON <sup>(5)</sup>	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
ANCON1	PIC18F2XJ11	PIC18F4XJ11	00-0 0000	uu-u uuuu	uu-u uuuu
ANCON0	PIC18F2XJ11	PIC18F4XJ11	0000 0000	uuuu uuuu	uuuu uuuu
ODCON1	PIC18F2XJ11	PIC18F4XJ11	---- --00	---- --uu	---- --uu
ODCON2	PIC18F2XJ11	PIC18F4XJ11	---- --00	---- --uu	---- --uu
ODCON3	PIC18F2XJ11	PIC18F4XJ11	---- --00	---- --uu	---- --uu
RTCCFG	PIC18F2XJ11	PIC18F4XJ11	0-00 0000	u-uu uuuu	u-uu uuuu
RTCCAL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	uuuu uuuu	uuuu uuuu
REFOCON	PIC18F2XJ11	PIC18F4XJ11	0-00 0000	u-uu uuuu	u-uu uuuu
PADCFG1	PIC18F2XJ11	PIC18F4XJ11	---- -000	---- -uuu	---- -uuu
RPINR24	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR23	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR22	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR21	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR17	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR16	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR13	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR12	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR8	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR7	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR6	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR4	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR3	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR2	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPINR1	PIC18F2XJ11	PIC18F4XJ11	---1 1111	---1 1111	---u uuuu
RPOR24	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR23	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR22	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR21	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.

**5:** Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
RPOR20	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR19	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR18	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR17	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR16	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR15	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR14	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR13	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR12	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR11	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR10	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR9	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR8	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR7	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR6	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR5	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR4	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR3	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR2	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR1	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
RPOR0	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

**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.
- 5: Not implemented for PIC18F2XJ11 devices.

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 5.0 MEMORY ORGANIZATION

There are two types of memory in PIC18 Flash microcontrollers:

- 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.

**Section 6.0 “Flash Program Memory”** provides additional information on the operation of the Flash program memory.

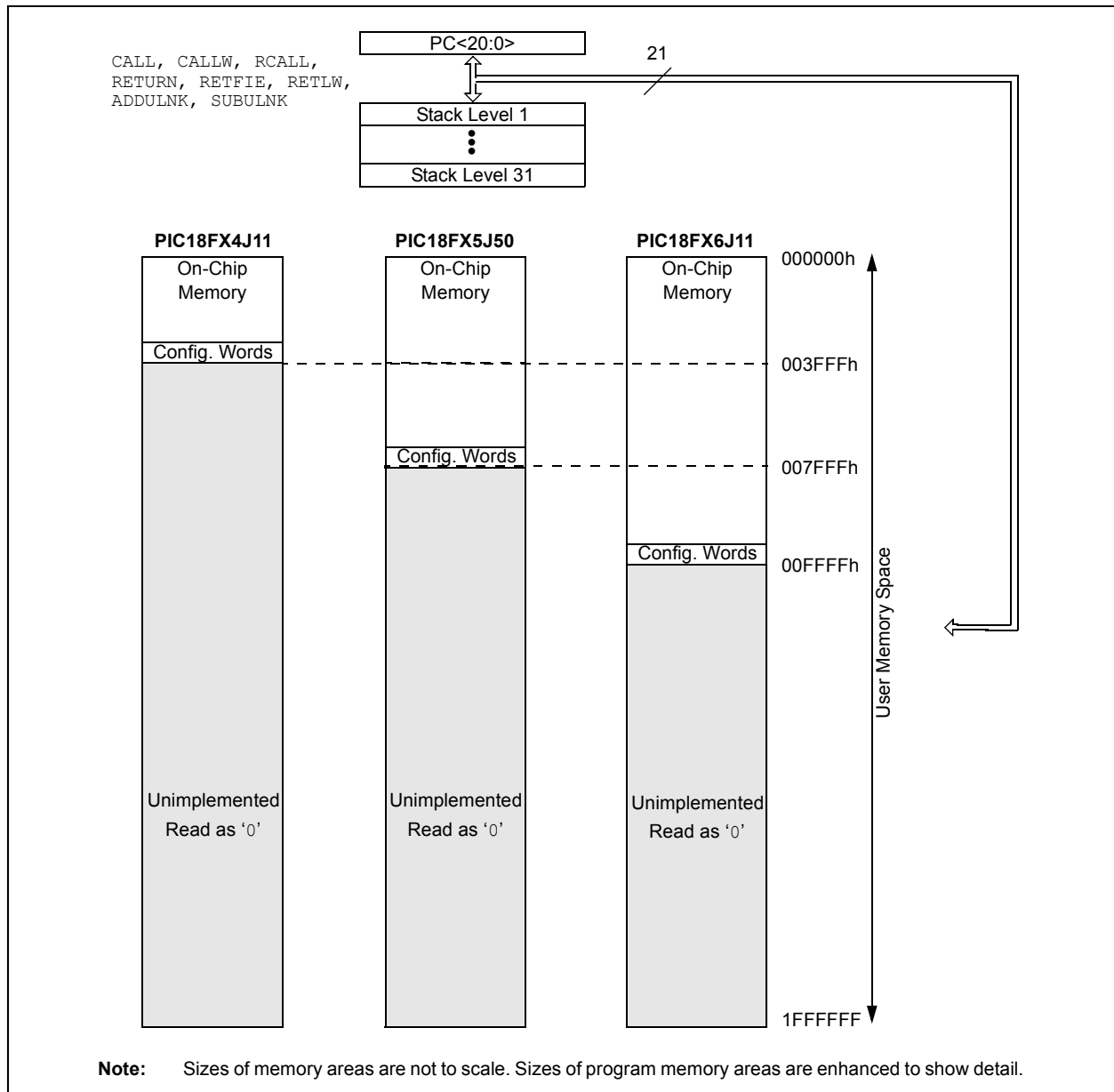
## 5.1 Program Memory Organization

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 returns all '0's (a NOP instruction).

The PIC18F46J11 family offers a range of on-chip Flash program memory sizes, from 16 Kbytes (up to 8,192 single-word instructions) to 64 Kbytes (32,768 single-word instructions).

Figure 5-1 provides the program memory maps for individual family devices.

**FIGURE 5-1: MEMORY MAPS FOR PIC18F46J11 FAMILY DEVICES**



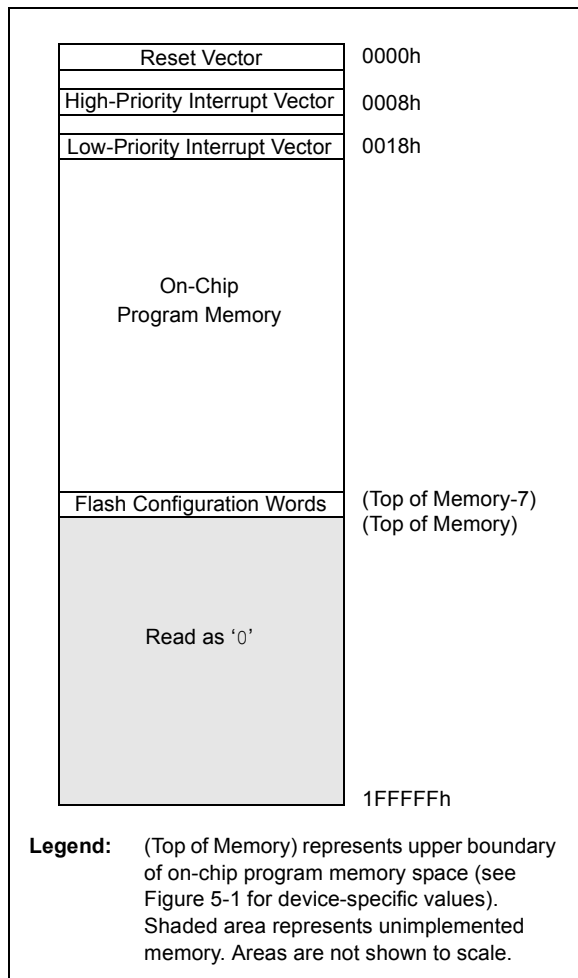
# PIC18F46J11 FAMILY

## 5.1.1 HARD MEMORY VECTORS

All PIC18 devices have a total of three hard-coded return vectors in their program memory space. The Reset vector address is the default value to which the program counter returns on all device Resets; it is located at 0000h.

PIC18 devices also have two interrupt vector addresses for handling high-priority and low-priority interrupts. The high-priority interrupt vector is located at 0008h and the low-priority interrupt vector at 0018h. Figure 5-2 provides their locations in relation to the program memory map.

**FIGURE 5-2: HARD VECTOR AND CONFIGURATION WORD LOCATIONS FOR PIC18F46J11 FAMILY DEVICES**



## 5.1.2 FLASH CONFIGURATION WORDS

Because PIC18F46J11 family devices do not have persistent configuration memory, the top four words of on-chip program memory are reserved for configuration information. On Reset, the configuration information is copied into the Configuration registers.

The Configuration Words are stored in their program memory location in numerical order, starting with the lower byte of CONFIG1 at the lowest address and ending with the upper byte of CONFIG4.

Table 5-1 provides the actual addresses of the Flash Configuration Word for devices in the PIC18F46J11 family. Figure 5-2 displays their location in the memory map with other memory vectors.

Additional details on the device Configuration Words are provided in Section 25.1 "Configuration Bits".

**TABLE 5-1: FLASH CONFIGURATION WORD FOR PIC18F46J11 FAMILY DEVICES**

Device	Program Memory (Kbytes)	Configuration Word Addresses
PIC18F24J11	16	3FF8h to 3FFFh
PIC18F44J11		
PIC18F25J11	32	7FF8h to 7FFFh
PIC18F45J11		
PIC18F26J11	64	FFF8h to FFFFh
PIC18F46J11		



## 5.1.3 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 to PCL. Similarly, the upper 2 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 5.1.6.1 “Computed GOTO”**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit (LSb) of PCL is fixed to a value of ‘0’. The PC increments by two 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.

## 5.1.4 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 a RETFIE instruction (and on ADDULNK and SUBULNK instructions if the extended instruction set is enabled). 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 (SP), 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 Function Registers (SFRs). 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.

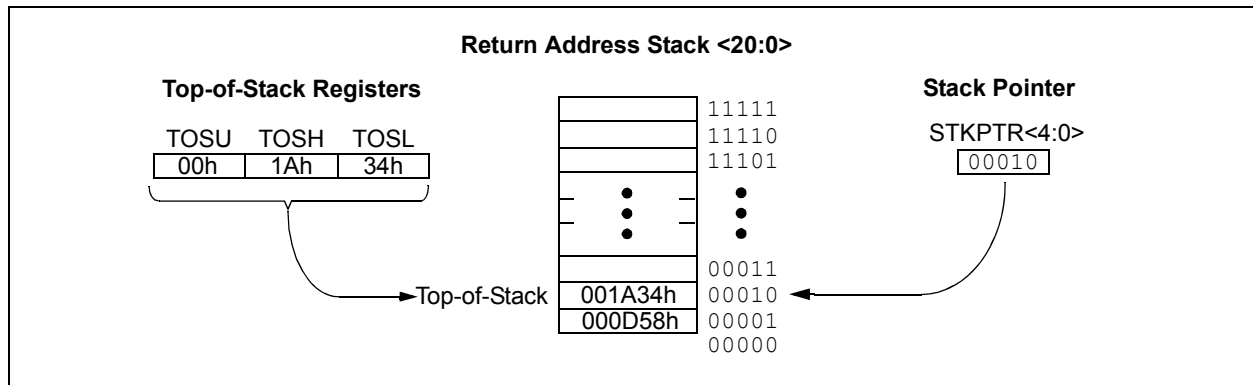
The Stack Pointer is initialized to ‘00000’ after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of ‘00000’; this is only a Reset value. Status bits indicate if the stack is full, has overflowed or has underflowed.

### 5.1.4.1 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, holds the contents of the stack location pointed to by the STKPTR register (Figure 5-3). This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt (and ADDULNK and SUBULNK instructions if the extended instruction set is enabled), 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 5-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS**



# PIC18F46J11 FAMILY

## 5.1.4.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 5-1) contains the Stack Pointer value, the STKFUL (Stack Full) 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 Power-on Reset (POR).

The action that takes place when the stack becomes full depends on the state of the Stack Overflow Reset Enable (STVREN) Configuration bit.

Refer to **Section 25.1 “Configuration Bits”** for device Configuration bits’ description.

If STVREN is set (default), the 31<sup>st</sup> 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 31<sup>st</sup> push and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31<sup>st</sup> 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 zero to the PC and set 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.

## 5.1.4.3 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 necessary. 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 (ACCESS FFCh)

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL <sup>(1)</sup>	STKUNF <sup>(1)</sup>	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

<b>Legend:</b>	C = Clearable bit	U = Unimplemented bit, read as ‘0’
R = Readable bit	W = Writable bit	‘0’ = Bit is cleared
-n = Value at POR	‘1’ = Bit is set	x = Bit is unknown

- bit 7      **STKFUL:** Stack Full Flag bit<sup>(1)</sup>  
             1 = Stack became full or overflowed  
             0 = Stack has not become full or overflowed
- bit 6      **STKUNF:** Stack Underflow Flag bit<sup>(1)</sup>  
             1 = Stack underflow occurred  
             0 = Stack underflow did not occur
- bit 5      **Unimplemented:** Read as ‘0’
- bit 4-0    **SP<4:0>:** Stack Pointer Location bits

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

## 5.1.4.4 Stack Full and Underflow Resets

Device Resets on stack overflow and stack underflow conditions are enabled by setting the STVREN bit in Configuration register 1L. When STVREN is set, a full or underflow condition sets the appropriate STKFUL or STKUNF bit and then causes a device Reset. When STVREN is cleared, a full or underflow condition sets the appropriate STKFUL or STKUNF bit, but does not cause a device Reset. The STKFUL or STKUNF bits are cleared by the user software or a POR.

## 5.1.5 FAST REGISTER STACK (FRS)

A Fast Register Stack (FRS) is provided for the STATUS, WREG and BSR registers to provide a “fast return” option for interrupts. This stack 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 push values into the Stack registers. The values in the registers are then loaded back into the working registers if the RETFIE, FAST instruction is used to return from the interrupt.

If both low-priority 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 FRS for returns from interrupt. If no interrupts are used, the FRS 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 FRS.

Example 5-1 provides a source code example that uses the FRS 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
```

## 5.1.6 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

### 5.1.6.1 Computed GOTO

A computed GOTO is accomplished by adding an offset to the PC. 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 executed instruction 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 PC should advance and should be multiples of 2 (LSb = 0).

In this method, only one byte may be stored in each instruction location; 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
      .
      .
      .
```

### 5.1.6.2 Table Reads

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

Look-up table data may be stored two bytes per program word while programming. The Table Pointer (TBLPTR) specifies the byte address and the Table Latch (TABLAT) contains the data that is read from the program memory. Data is transferred from program memory one byte at a time.

Table read operation is discussed further in Section 6.1 “Table Reads and Table Writes”.

# PIC18F46J11 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 '4' to generate four non-overlapping quadrature clocks (Q1, Q2, Q3 and Q4). Internally, the PC is incremented on every Q1; the instruction is fetched from the program memory and latched into the Instruction Register (IR) during Q4. The instruction is decoded and executed during the following Q1 through Q4. Figure 5-4 illustrates the clocks and instruction execution flow.

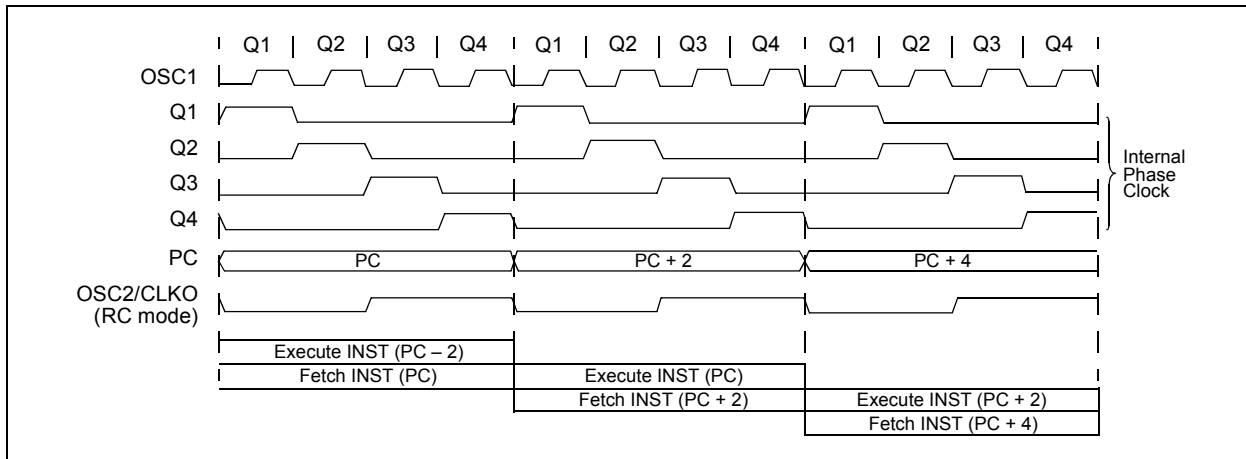
### 5.2.2 INSTRUCTION FLOW/PIPELINING

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 PC to change (e.g., GOTO), then two cycles are required to complete the instruction (Example 5-3).

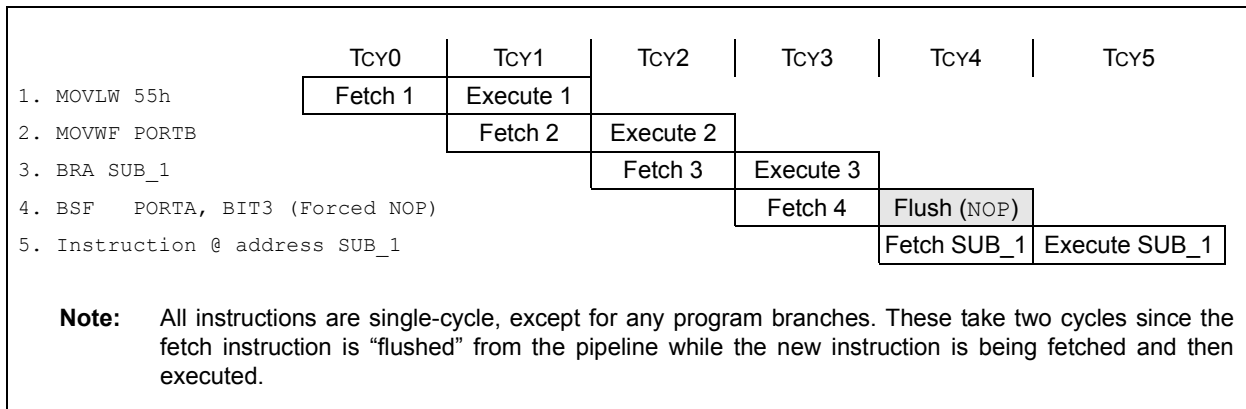
A fetch cycle begins with the PC incrementing in Q1.

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

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



**EXAMPLE 5-3: INSTRUCTION PIPELINE FLOW**



# PIC18F46J11 FAMILY

## 5.2.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as 2 bytes or 4 bytes in program memory. The Least Significant Byte (LSB) 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 5.1.3 "Program Counter").

Figure 5-5 provides 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 5-5 displays 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 26.0 "Instruction Set Summary" provides further details of the instruction set.

**FIGURE 5-5: 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
			C1h	23h	00000Eh
Instruction 3:	MOVFF	123h, 456h	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 LSRF. In all cases, the second word of the instructions always has '1111' as its four Most Significant Bits (MSBs); 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 illustrates how this works.

**Note:** See Section 5.5 "Program Memory 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

# PIC18F46J11 FAMILY

## 5.3 Data Memory Organization

**Note:** The operation of some aspects of data memory is changed when the PIC18 extended instruction set is enabled. See **Section 5.6 “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. The PIC18F46J11 family implements all available banks and provides 3.8 Kbytes of data memory available to the user. Figure 5-6 provides the data memory organization for the 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 section.

To ensure that commonly used registers (select 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 select 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

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 MSBs of a location's address; the instruction itself includes the 8 LSbs. Only the four lower bits of the BSR are implemented (BSR<3:0>). 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 illustrated in Figure 5-7.

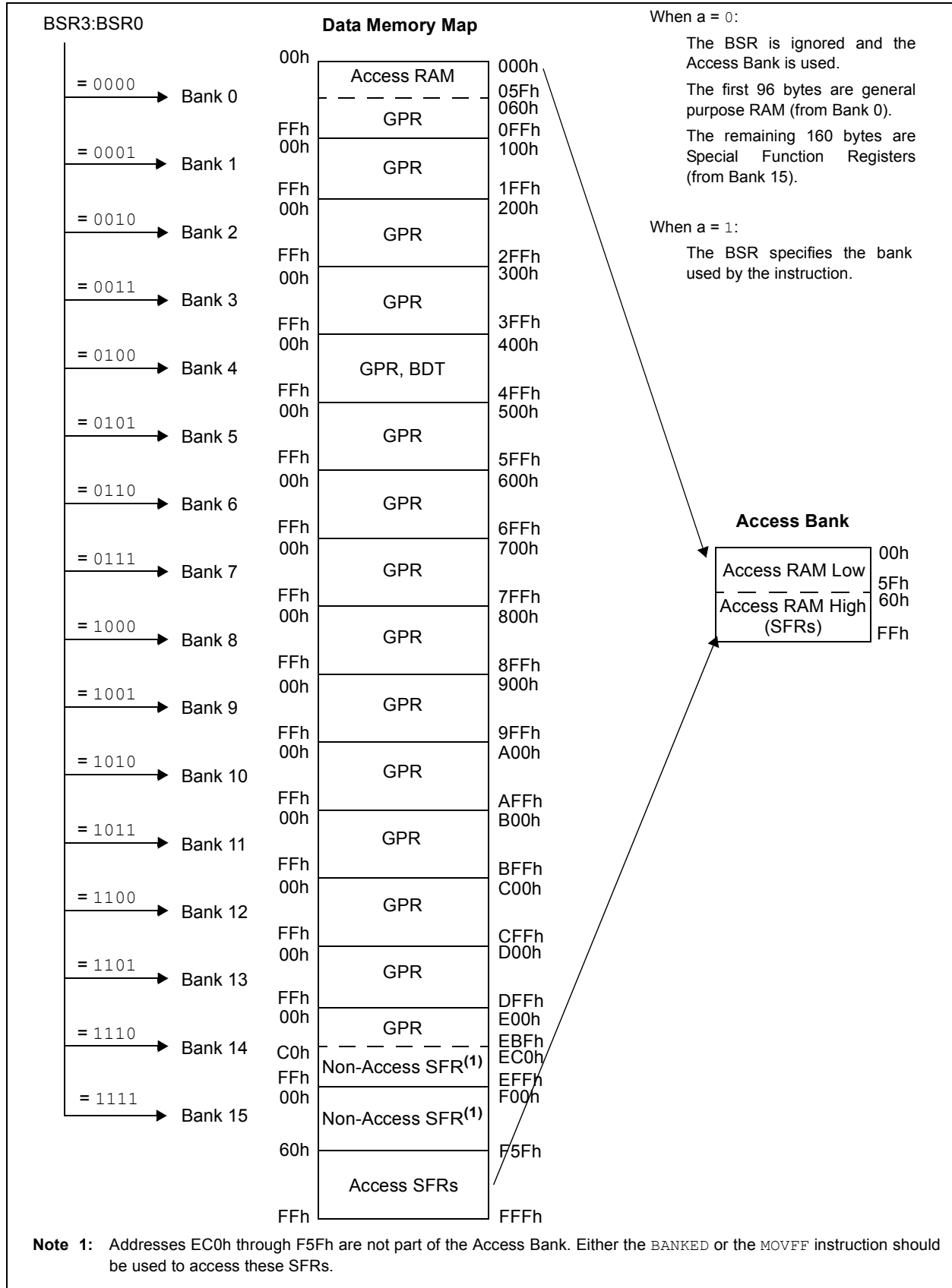
Since, up to 16 registers may share the same low-order address, the user must always be careful to ensure that the proper bank is selected before performing a data read or write. For example, writing what should be program data to an 8-bit address of F9h while the BSR is 0Fh, will end up resetting the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

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 5-6 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.

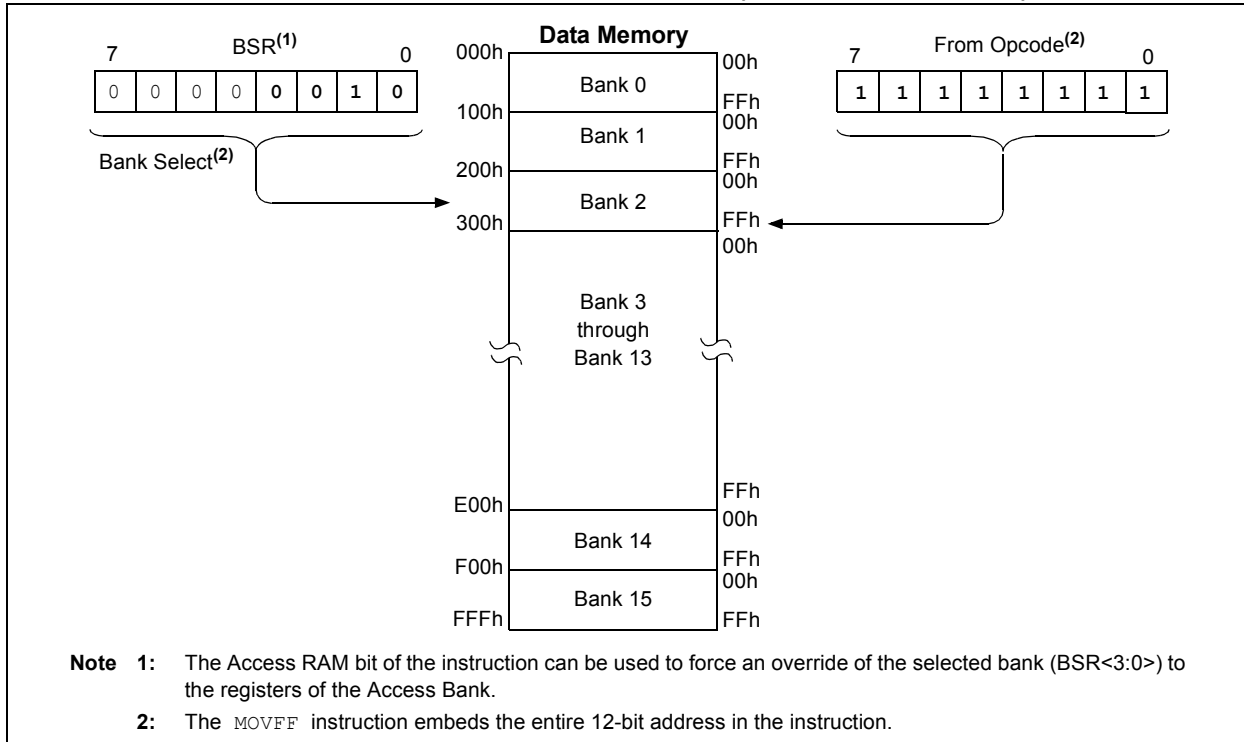
# PIC18F46J11 FAMILY

**FIGURE 5-6: DATA MEMORY MAP FOR PIC18F46J11 FAMILY DEVICES**



# PIC18F46J11 FAMILY

**FIGURE 5-7: 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 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Bank 15. The lower half is known as the Access RAM and is composed of GPRs. The upper half is 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 5-6).

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 60h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 60h 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 5.6.3 "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 upward toward the bottom of the SFR area. GPRs are not initialized by a POR and are unchanged on all other Resets.



# PIC18F46J11 FAMILY

## 5.3.4 SPECIAL FUNCTION REGISTERS

The 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 more than the top half of Bank 15 (F40h to FFFh). Table 5-2, Table 5-3 and Table 5-4 provide a list of these registers.

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 corresponding chapters, while the

ALU's STATUS register is described later in this section. Registers related to the operation of the peripheral features 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

**Note:** The SFRs located between EC0h and F5Fh are not part of the Access Bank. Either banked instructions (using BSR) or the MOVFF instruction should be used to access these locations. When programming in MPLAB® C18, the compiler will automatically use the appropriate addressing mode.

**TABLE 5-2: ACCESS BANK SPECIAL FUNCTION REGISTER MAP**

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2 <sup>(1)</sup>	FBFh	PSTR1CON	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2 <sup>(1)</sup>	FBEh	ECCP1AS	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2 <sup>(1)</sup>	FBDh	ECCP1DEL	F9Dh	PIE1	F7Dh	SPBRGH2
FFCh	STKPTR	FDCh	PREINC2 <sup>(1)</sup>	FBCh	CCPR1H	F9Ch	RCSTA2	F7Ch	BAUDCON2
FFBh	PCLATU	FDBh	PLUSW2 <sup>(1)</sup>	FBBh	CCPR1L	F9Bh	OSCTUNE	F7Bh	TMR3H
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP1CON	F9Ah	T1GCON	F7Ah	TMR3L
FF9h	PCL	FD9h	FSR2L	FB9h	PSTR2CON	F99h	RTCV ALH	F79h	T3CON
FF8h	TBLPTRU	FD8h	STATUS	FB8h	ECCP2AS	F98h	RTCVALL	F78h	TMR4
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	ECCP2DEL	F97h	T3GCON	F77h	PR4
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	CCPR2H	F96h	TRISE	F76h	T4CON
FF5h	TABLAT	FD5h	T0CON	FB5h	CCPR2L	F95h	TRISD	F75h	SSP2BUF
FF4h	PRODH	FD4h	__ <sup>(5)</sup>	FB4h	CCP2CON	F94h	TRISC	F74h	SSP2ADD <sup>(3)</sup>
FF3h	PRODL	FD3h	OSCCON	FB3h	CTMUCONH	F93h	TRISB	F73h	SSP2STAT
FF2h	INTCON	FD2h	CM1CON	FB2h	CTMUCONL	F92h	TRISA	F72h	SSP2CON1
FF1h	INTCON2	FD1h	CM2CON	FB1h	CTMUICON	F91h	ALRMCFG	F71h	SSP2CON2
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRG1	F90h	ALRMRPT	F70h	CMSTAT
FEFh	INDF0 <sup>(1)</sup>	FCFh	TMR1H	FAFh	RCREG1	F8Fh	ALRMVALH	F6Fh	PMADDRH <sup>(2,4)</sup>
FEEh	POSTINC0 <sup>(1)</sup>	FCEh	TMR1L	FAEh	TXREG1	F8Eh	ALRMVALL	F6Eh	PMADDRL <sup>(2,4)</sup>
FEDh	POSTDEC0 <sup>(1)</sup>	FCDh	T1CON	FADh	TXSTA1	F8Dh	LATE <sup>(2)</sup>	F6Dh	PMDIN1H <sup>(2)</sup>
FECh	PREINC0 <sup>(1)</sup>	FCCh	TMR2	FACH	RCSTA1	F8Ch	LATD <sup>(2)</sup>	F6Ch	PMDIN1L <sup>(2)</sup>
FEBh	PLUSW0 <sup>(1)</sup>	FCBh	PR2	FABh	SPBRG2	F8Bh	LATC	F6Bh	TXADDRL
FEAh	FSR0H	FCAh	T2CON	FAAh	RCREG2	F8Ah	LATB	F6Ah	TXADDRH
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	TXREG2	F89h	LATA	F69h	RXADDRL
FE8h	WREG	FC8h	SSP1ADD <sup>(3)</sup>	FA8h	TXSTA2	F88h	DMACON1	F68h	RXADDRH
FE7h	INDF1 <sup>(1)</sup>	FC7h	SSP1STAT	FA7h	EECON2	F87h	__ <sup>(5)</sup>	F67h	DMABCL
FE6h	POSTINC1 <sup>(1)</sup>	FC6h	SSP1CON1	FA6h	EECON1	F86h	DMACON2	F66h	DMABCH
FE5h	POSTDEC1 <sup>(1)</sup>	FC5h	SSP1CON2	FA5h	IPR3	F85h	HLVDCON	F65h	__ <sup>(5)</sup>
FE4h	PREINC1 <sup>(1)</sup>	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE <sup>(2)</sup>	F64h	__ <sup>(5)</sup>
FE3h	PLUSW1 <sup>(1)</sup>	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD <sup>(2)</sup>	F63h	__ <sup>(5)</sup>
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	__ <sup>(5)</sup>
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	__ <sup>(5)</sup>
FE0h	BSR	FC0h	WDTCON	FA0h	PIE2	F80h	PORTA	F60h	__ <sup>(5)</sup>

- Note**
- 1: This is not a physical register.
  - 2: This register is not available on 28-pin devices.
  - 3: SSPxADD and SSPxMSK share the same address.
  - 4: PMADDRH and PMDOUTH share the same address and PMADDRL and PMDOUTL share the same address. PMADDRx is used in Master modes and PMDOUTx is used in Slave modes.
  - 5: Reserved: Do not write to this location.

# PIC18F46J11 FAMILY

**TABLE 5-3: NON-ACCESS BANK SPECIAL FUNCTION REGISTER MAP**

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
F5Fh	PMCONH	F3Fh	RTCCFG	F1Fh	—	EFFh	PPSCON	EDFh	—
F5Eh	PMCONL	F3Eh	RTCCAL	F1Eh	—	EFEh	RPINR24	EDEh	RPOR24 <sup>(1)</sup>
F5Dh	PMMODEH	F3Dh	REFOCON	F1Dh	—	EFDh	RPINR23	EDDh	RPOR23 <sup>(1)</sup>
F5Ch	PMMODEL	F3Ch	PADCFG1	F1Ch	—	EFCh	RPINR22	EDCh	RPOR22 <sup>(1)</sup>
F5Bh	PMDOUT2H	F3Bh	—	F1Bh	—	EFBh	RPINR21	EDBh	RPOR21 <sup>(1)</sup>
F5Ah	PMDOUT2L	F3Ah	—	F1Ah	—	EFAh	—	EDAh	RPOR20 <sup>(1)</sup>
F59h	PMDIN2H	F39h	—	F19h	—	EF9h	—	ED9h	RPOR19 <sup>(1)</sup>
F58h	PMDIN2L	F38h	—	F18h	—	EF8h	—	ED8h	RPOR18
F57h	PMEH	F37h	—	F17h	—	EF7h	RPINR17	ED7h	RPOR17
F56h	PMEL	F36h	—	F16h	—	EF6h	RPINR16	ED6h	RPOR16
F55h	PMSTATH	F35h	—	F15h	—	EF5h	—	ED5h	RPOR15
F54h	PMSTATL	F34h	—	F14h	—	EF4h	—	ED4h	RPOR14
F53h	CVRCON	F33h	—	F13h	—	EF3h	—	ED3h	RPOR13
F52h	TCLKCON	F32h	—	F12h	—	EF2h	—	ED2h	RPOR12
F51h	—	F31h	—	F11h	—	EF1h	—	ED1h	RPOR11
F50h	—	F30h	—	F10h	—	EF0h	—	ED0h	RPOR10
F4Fh	DSGPR1 <sup>(2)</sup>	F2Fh	—	F0Fh	—	EEFh	—	ECFh	RPOR9
F4Eh	DSGPR0 <sup>(2)</sup>	F2Eh	—	F0Eh	—	EEEh	RPINR8	ECEh	RPOR8
F4Dh	DSCONH <sup>(2)</sup>	F2Dh	—	F0Dh	—	EEDh	RPINR7	ECDh	RPOR7
F4Ch	DSCONL <sup>(2)</sup>	F2Ch	—	F0Ch	—	EECh	RPINR6	ECCh	RPOR6
F4Bh	DSWAKEH <sup>(2)</sup>	F2Bh	—	F0Bh	—	EEBh	—	ECBh	RPOR5
F4Ah	DSWAKEL <sup>(2)</sup>	F2Ah	—	F0Ah	—	EEAh	RPINR4	ECAh	RPOR4
F49h	ANCON1	F29h	—	F09h	—	EE9h	RPINR3	EC9h	RPOR3
F48h	ANCON0	F28h	—	F08h	—	EE8h	RPINR2	EC8h	RPOR2
F47h	—	F27h	—	F07h	—	EE7h	RPINR1	EC7h	RPOR1
F46h	—	F26h	—	F06h	—	EE6h	RPINR0	EC6h	RPOR0
F45h	—	F25h	—	F05h	—	EE5h	—	EC5h	—
F44h	—	F24h	—	F04h	—	EE4h	—	EC4h	—
F43h	—	F23h	—	F03h	—	EE3h	—	EC3h	—
F42h	ODCON1	F22h	—	F02h	—	EE2h	—	EC2h	—
F41h	ODCON2	F21h	—	F01h	—	EE1h	—	EC1h	—
F40h	ODCON3	F20h	—	F00h	—	EE0h	—	EC0h	—

- Note 1:** This register is not available on 28-pin devices.  
**Note 2:** Deep Sleep registers are not available on LF devices.

# PIC18F46J11 FAMILY

## 5.3.4.1 Context Defined SFRs

There are several registers that share the same address in the SFR space. The register's definition and usage depends on the operating mode of its associated peripheral. These registers are:

- SSPxADD and SSPxMSK: These are two separate hardware registers, accessed through a single SFR address. The operating mode of the MSSP modules determines which register is being accessed. See **Section 18.5.3.4 “7-Bit Address Masking Mode”** for additional details.

- PMADDRH/L and PMDOUT2H/L: In this case, these named buffer pairs are actually the same physical registers. The Parallel Master Port (PMP) module's operating mode determines what function the registers take on. See **Section 10.1.2 “Data Registers”** for additional details.

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY)**

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	63, 75
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	63, 73
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	63, 73
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	63, 74
PCLATU	—	—	bit 21 <sup>(1)</sup>	Holding Register for PC<20:16>					---0 0000	63, 73
PCLATH	Holding Register for PC<15:8>								0000 0000	63, 73
PCL	PC Low Byte (PC<7:0>)								0000 0000	63, 73
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	63, 106
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	63, 106
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	63, 106
TABLAT	Program Memory Table Latch								0000 0000	63, 106
PRODH	Product Register High Byte								xxxx xxxx	63, 63
PRODL	Product Register Low Byte								xxxx xxxx	63, 107
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	63, 111
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	63, 112
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	63, 113
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	63, 91
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	63, 92
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	63, 92
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	63, 92
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	63, 92
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- xxxx	63, 91
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	63, 91
WREG	Working Register								xxxx xxxx	63, 75
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	63, 91
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	63, 92
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	63, 92
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	63, 92
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	63, 92

**Legend:** x = unknown, u = unchanged, – = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note 1:** Bit 21 of the PC is only available in Serial Programming (SP) modes.
- Note 2:** Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
- Note 3:** The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
- Note 4:** Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See **Section 18.5.3.2 “Address Masking Modes”** for details.
- Note 5:** These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
- Note 6:** The PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See **Section 10.1.2 “Data Registers”** for more information.

# PIC18F46J11 FAMILY

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY) (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:
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- xxxx	63, 91
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	63, 91
BSR	—	—	—	—	Bank Select Register				---- 0000	63, 78
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	63, 91
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	64, 92
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	64, 92
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	64, 92
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	64, 92
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- xxxx	64, 91
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	64, 91
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	64, 89
TMR0H	Timer0 Register High Byte								0000 0000	64
TMR0L	Timer0 Register Low Byte								xxxx xxxx	64
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	64, 191
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS <sup>(2)</sup>	—	SCS1	SCS0	0110 q-00	64, 38
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	0001 1111	64, 356
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	0001 1111	64, 356
RCON	IPEN	—	CM	RI	TO	PD	POR	BOR	0-11 1100	64, 123
TMR1H	Timer1 Register High Byte								xxxx xxxx	64
TMR1L	Timer1 Register Low Byte								xxxx xxxx	64
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	T1SYN <sup>C</sup>	RD16	TMR1ON	0000 0000	64, 195
TMR2	Timer2 Register								0000 0000	64
PR2	Timer2 Period Register								1111 1111	64
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	64, 207
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								xxxx xxxx	64
SSP1ADD	MSSP1 Address Register (I <sup>2</sup> C™ Slave mode), MSSP1 Baud Rate Reload Register (I <sup>2</sup> C Master mode)								0000 0000	64
SSP1MSK <sup>(4)</sup>	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	1111 1111	64, 289
SSP1STAT	SMP	CKE	D/A	P	S	R/W	UA	BF	0000 0000	64, 286
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	64, 287
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	64, 288
	GCEN	ACKSTAT	ADMSK5 <sup>(4)</sup>	ADMSK4 <sup>(4)</sup>	ADMSK3 <sup>(4)</sup>	ADMSK2 <sup>(4)</sup>	ADMSK1 <sup>(4)</sup>	SEN		
ADRESH	A/D Result Register High Byte								xxxx xxxx	64
ADRESL	A/D Result Register Low Byte								xxxx xxxx	64
ADCON0	VCFG1	VCFG0	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	0000 0000	64, 345
ADCON1	ADFM	ADCAL	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0000 0000	64, 346
WDTCON	REGSLP	LVDSTAT	ULPLVL	—	DS	ULPEN	ULPSINK	SWDTEN	1xx- 0000	64, 400
PSTR1CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	00-0 0001	64, 261
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	0000 0000	64
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000	65
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	65
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	65
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	65

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note 1:** Bit 21 of the PC is only available in Serial Programming (SP) modes.
- 2:** Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
- 3:** The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
- 4:** Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See **Section 18.5.3.2 "Address Masking Modes"** for details.
- 5:** These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
- 6:** The PMADDRH/PMOUT1H and PMADDRH/PMOUT1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See **Section 10.1.2 "Data Registers"** for more information.

# PIC18F46J11 FAMILY

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY) (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:
PSTR2CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	00-0 0001	65, 261
ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000	65
ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000	65
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	65
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	65
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000	65
CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	—	0-00 000-	65
CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	0000 0000	65
CTMUICON	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	0000 0000	65
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								0000 0000	65
RCREG1	EUSART1 Receive Register								0000 0000	65
TXREG1	EUSART1 Transmit Register								xxxx xxxx	65
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	65, 322
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	65, 323
SPBRG2	EUSART2 Baud Rate Generator Register Low Byte								0000 0000	65
RCREG2	EUSART2 Receive Register								0000 0000	65
TXREG2	EUSART2 Transmit Register								0000 0000	65
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	65, 322
EECON2	Program Memory Control Register 2 (not a physical register)								---- ----	65
EECON1	—	—	WPROG	FREE	WRERR	WREN	WR	—	--00 x00-	65, 99
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	1111 1111	65, 122
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	0000 0000	65, 116
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	0000 0000	65, 119
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	111- 1111	65, 121
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	000- 0000	65, 115
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	000- 0000	65, 118
IPR1	PMPIP <sup>(5)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	1111 1111	65, 120
PIR1	PMPIF <sup>(5)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	65, 114
PIE1	PMPIE <sup>(5)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	0000 0000	65, 117
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	66, 323
OSCTUNE	INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000 0000	66, 36
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ T1DONE	T1GVAL	T1GSS1	T1GSS0	0000 0x00	66, 196
RTCVALH	RTCC Value Register Window High Byte, Based on RTCPTR<1:0>								0xxx xxxx	66
RTCVALL	RTCC Value Register Window Low Byte, Based on RTCPTR<1:0>								0xxx xxxx	66
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/ T3DONE	T3GVAL	T3GSS1	T3GSS0	0000 0x00	66, 210
TRISE	—	—	—	—	—	TRISE2	TRISE1	TRISE0	---- -111	66
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111	66
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	66
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	66
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	111- 1111	66
ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	0000 0000	66, 225

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note**
- 1: Bit 21 of the PC is only available in Serial Programming (SP) modes.
  - 2: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
  - 3: The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
  - 4: Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See Section 18.5.3.2 "Address Masking Modes" for details.
  - 5: These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
  - 6: The PMADDRH/PMDOUB1H and PMADDRL/PMDOUB1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See Section 10.1.2 "Data Registers" for more information.

# PIC18F46J11 FAMILY

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY) (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:
ALRM RPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000 0000	66, 226
ALRMVALH	Alarm Value Register Window High Byte, Based on ALRMPTR<1:0>								xxxx xxxx	66
ALRMVALL	Alarm Value Register Window Low Byte, Based on ALRMPTR<1:0>								xxxx xxxx	66
LATE	—	—	—	—	—	LATE2	LATE1	LATE0	---- -xxx	66
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx	66
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	66
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	66
LATA	LATA7	LATA6	LATA5	—	LATA3	LATA2	LATA1	LATA0	xxx- xxxx	66
DMA CON1	SSCON1	SSCON0	TXINC	RXINC	DUPLEX1	DUPLEX0	DLYINTEN	DMAEN	0000 0000	66, 278
DMATXBUF	SPI DMA Transmit Buffer								xxxx xxxx	66
DMA CON2	DLYCYC3	DLYCYC2	DLYCYC1	DLYCYC0	INTLVL3	INTLVL2	INTLVL1	INTLVL0	0000 0000	66, 279
HLVD CON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0000 0000	66
PORTE	RDPU	REPU	—	—	—	RE2	RE1	RE0	00-- -xxx	66
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	66
PORTC	RC7	RC6	RC5	RC4	RC4	RC2	RC1	RC0	xxxx xxxx	66
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	66
PORTA	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0	xxx- xxxx	66
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								0000 0000	66
BAUD CON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	66, 324
SPBRGH2	EUSART2 Baud Rate Generator Register High Byte								0000 0000	66
BAUD CON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	66, 324
TMR3H	Timer3 Register High Byte								xxxx xxxx	67
TMR3L	Timer3 Register Low Byte								xxxx xxxx	67
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	—	T3SYNC	RD16	TMR3ON	0000 -000	67, 209
TMR4	Timer4 Register								0000 0000	67
PR4	Timer4 Period Register								1111 1111	67
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	67, 219
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								xxxx xxxx	67
SSP2ADD/	MSSP2 Address Register (I <sup>2</sup> C™ Slave mode), MSSP2 Baud Rate Reload Register (I <sup>2</sup> C Master mode)								0000 0000	67, 289
SSP2MSK <sup>(4)</sup>	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	0000 0000	67, 289
SSP2STAT	SMP	CKE	D/Ā	P	S	R/W	UA	BF	1111 1111	67, 267
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	67, 287
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	67, 288
	GCEN	ACKSTAT	ADMSK5 <sup>(4)</sup>	ADMSK4 <sup>(4)</sup>	ADMSK3 <sup>(4)</sup>	ADMSK2 <sup>(4)</sup>	ADMSK1 <sup>(4)</sup>	SEN		
CMSTAT	—	—	—	—	—	—	COUT2	COUT1	---- --11	67, 357
PMADDRH/	—	CS1	Parallel Master Port Address High Byte						-000 0000	67, 173
PMDOUT1H <sup>(5)</sup>	Parallel Port Out Data High Byte (Buffer 1)								0000 0000	67, 173
PMADDRL/	Parallel Master Port Address Low Byte								0000 0000	67, 173
PMDOUT1L <sup>(5)</sup>	Parallel Port Out Data Low Byte (Buffer 0)								0000 0000	67, 173
PMDIN1H <sup>(5)</sup>	Parallel Port In Data High Byte (Buffer 1)								0000 0000	67
PMDIN1L <sup>(5)</sup>	Parallel Port In Data Low Byte (Buffer 0)								0000 0000	67
TXADDRL	SPI DMA Transit Data Pointer Low Byte								xxxx xxxx	67

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note 1:** Bit 21 of the PC is only available in Serial Programming (SP) modes.
- Note 2:** Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
- Note 3:** The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
- Note 4:** Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See **Section 18.5.3.2 "Address Masking Modes"** for details.
- Note 5:** These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
- Note 6:** The PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See **Section 10.1.2 "Data Registers"** for more information.

# PIC18F46J11 FAMILY

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY) (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:	
TXADDRH	—	—	—	—	SPI DMA Transit Data Pointer High Byte				---- xxxx	67	
RXADDRL	SPI DMA Receive Data Pointer Low Byte								xxxx xxxx	67	
RXADDRH	—	—	—	—	SPI DMA Receive Data Pointer High Byte				---- xxxx	67	
DMABCL	SPI DMA Byte Count Low Byte								xxxx xxxx	67	
DMABCH	—	—	—	—	—	—	SPI DMA Receive Data Pointer High Byte		---- --xx	67	
PMCONH <sup>(5)</sup>	PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	0-00 0000	67, 166	
PMCONL <sup>(5)</sup>	CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP	000- 0000	67, 167	
PMODEH <sup>(5)</sup>	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	0000 0000	67, 168	
PMODEL <sup>(5)</sup>	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	0000 0000	67, 169	
PMDOUT2H <sup>(5)</sup>	Parallel Port Out Data High Byte (Buffer 3)								0000 0000	67	
PMDOUT2L <sup>(5)</sup>	Parallel Port Out Data Low Byte (Buffer 2)								0000 0000	67	
PMDIN2H <sup>(5)</sup>	Parallel Port In Data High Byte (Buffer 3)								0000 0000	67	
PMDIN2L <sup>(5)</sup>	Parallel Port In Data Low Byte (Buffer 2)								0000 0000	67	
PMEH <sup>(5)</sup>	PTEN15	PTEN14	PTEN13	PTEN12	PTEN11	PTEN10	PTEN9	PTEN8	0000 0000	67, 170	
PMEL <sup>(5)</sup>	PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0	0000 0000	67, 170	
PMSTATH <sup>(5)</sup>	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	00-- 0000	67, 171	
PMSTATL <sup>(5)</sup>	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	10-- 1111	67, 171	
CVRCON <sup>(5)</sup>	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	67, 364	
TCLKCON	—	—	—	T1RUN	—	—	T3CCP2	T3CCP1	---0 --00	197	
DSGPR1	Deep Sleep Persistent General Purpose Register (contents retained even in Deep Sleep)								xxxx xxxx	53	
DSGPR0	Deep Sleep Persistent General Purpose Register (contents retained even in Deep Sleep)								xxxx xxxx	53	
DSCONH	DSEN	—	—	—	—	(Reserved)	DSULPEN	RTCWDIS	0--- -000	52	
DSCONL	—	—	—	—	—	—	ULPWDIS	DSBOR	RELEASE	---- -000	52
DSWAKEH	—	—	—	—	—	—	—	DSINT0	---- ---0	54	
DSWAKEL	DSFLT	—	DSULP	DSWDT	DSRTC	DSMCLR	—	DSPOR	0-00 00-1	54	
ANCON1	VBGEN	r	—	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	00-0 0000	67, 347	
ANCON0	PCFG7 <sup>(5)</sup>	PCFG6 <sup>(5)</sup>	PCFG5 <sup>(5)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000 0000	67, 347	
ODCON1	—	—	—	—	—	—	ECCP20D	ECCP10D	---- --00	67, 127	
ODCON2	—	—	—	—	—	—	U20D	U10D	---- --00	67, 127	
ODCON3	—	—	—	—	—	—	SPI20D	SPI10D	---- --00	67, 128	
RTCCFG	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPR1	RTCPR0	0-00 0000	67, 223	
RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000 0000	67, 224	
REFOCON	ROON	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0	0-00 0000	67, 39	
PADCFG1	—	—	—	—	—	—	RTSECSEL1	RTSECSEL0	PMP TTL	---- -000	67, 128
PPSCON	—	—	—	—	—	—	—	IOLOCK	---- ---0	149	
RPINR24	—	—	—	Input Function FLT0 to Input Pin Mapping Bits					---1 1111	68, 154	
RPINR23	—	—	—	Input Function SS2 to Input Pin Mapping Bits					---1 1111	68, 154	
RPINR22	—	—	—	Input Function SCK2 to Input Pin Mapping Bits					---1 1111	68, 154	
RPINR21	—	—	—	Input Function SDI2 to Input Pin Mapping Bits					---1 1111	68, 153	
RPINR17	—	—	—	Input Function CK2 to Input Pin Mapping Bits					---1 1111	68, 153	
RPINR16	—	—	—	Input Function RX2DT2 to Input Pin Mapping Bits					---1 1111	153	
RPINR13	—	—	—	Input Function T3G to Input Pin Mapping Bits					---1 1111	68, 152	

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note 1:** Bit 21 of the PC is only available in Serial Programming (SP) modes.
- 2:** Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
- 3:** The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
- 4:** Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See **Section 18.5.3.2 "Address Masking Modes"** for details.
- 5:** These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
- 6:** The PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See **Section 10.1.2 "Data Registers"** for more information.

# PIC18F46J11 FAMILY

**TABLE 5-4: REGISTER FILE SUMMARY (PIC18F46J11 FAMILY) (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:
RPINR12	—	—	—	Input Function T1G to Input Pin Mapping Bits				---	1 1111	69, 152
RPINR8	—	—	—	Input Function IC2 to Input Pin Mapping Bits				---	1 1111	68, 152
RPINR7	—	—	—	Input Function IC1 to Input Pin Mapping Bits				---	1 1111	68, 151
RPINR6	—	—	—	Input Function T3CKI to Input Pin Mapping Bits				---	1 1111	68, 151
RPINR4	—	—	—	Input Function T0CKI to Input Pin Mapping Bits				---	1 1111	68, 151
RPINR3	—	—	—	Input Function INT3 to Input Pin Mapping Bits				---	1 1111	68, 150
RPINR2	—	—	—	Input Function INT2 to Input Pin Mapping Bits				---	1 1111	68
RPINR1	—	—	—	Input Function INT1 to Input Pin Mapping Bits				---	1 1111	68, 150
RPOR24 <sup>(5)</sup>	—	—	—	Remappable Pin RP24 Output Signal Select Bits				---	0 0000	68, 163
RPOR23 <sup>(5)</sup>	—	—	—	Remappable Pin RP23 Output Signal Select Bits				---	0 0000	68, 163
RPOR22 <sup>(5)</sup>	—	—	—	Remappable Pin RP22 Output Signal Select Bits				---	0 0000	68, 162
RPOR21 <sup>(5)</sup>	—	—	—	Remappable Pin RP21 Output Signal Select Bits				---	0 0000	68, 162
RPOR20 <sup>(5)</sup>	—	—	—	Remappable Pin RP20 Output Signal Select Bits				---	0 0000	68, 162
RPOR19 <sup>(5)</sup>	—	—	—	Remappable Pin RP19 Output Signal Select Bits				---	0 0000	68, 161
RPOR18	—	—	—	Remappable Pin RP18 Output Signal Select Bits				---	0 0000	68, 161
RPOR17	—	—	—	Remappable Pin RP17 Output Signal Select Bits				---	0 0000	69, 161
RPOR16	—	—	—	Remappable Pin RP13 Output Signal Select Bits				---	0 0000	69, 160
RPOR15	—	—	—	Remappable Pin RP13 Output Signal Select Bits				---	0 0000	69, 160
RPOR14	—	—	—	Remappable Pin RP13 Output Signal Select Bits				---	0 0000	69, 160
RPOR13	—	—	—	Remappable Pin RP13 Output Signal Select Bits				---	0 0000	69, 159
RPOR12	—	—	—	Remappable Pin RP12 Output Signal Select Bits				---	0 0000	69, 159
RPOR11	—	—	—	Remappable Pin RP11 Output Signal Select Bits				---	0 0000	69, 159
RPOR10	—	—	—	Remappable Pin RP10 Output Signal Select Bits				---	0 0000	69, 158
RPOR9	—	—	—	Remappable Pin RP9 Output Signal Select Bits				---	0 0000	69, 158
RPOR8	—	—	—	Remappable Pin RP8 Output Signal Select Bits				---	0 0000	69, 157
RPOR7	—	—	—	Remappable Pin RP7 Output Signal Select Bits				---	0 0000	69, 157
RPOR6	—	—	—	Remappable Pin RP6 Output Signal Select Bits				---	0 0000	69, 157
RPOR5	—	—	—	Remappable Pin RP5 Output Signal Select Bits				---	0 0000	69, 156
RPOR4	—	—	—	Remappable Pin RP4 Output Signal Select Bits				---	0 0000	69, 156
RPOR3	—	—	—	Remappable Pin RP3 Output Signal Select Bits				---	0 0000	69, 156
RPOR2	—	—	—	Remappable Pin RP2 Output Signal Select Bits				---	0 0000	69, 155
RPOR1	—	—	—	Remappable Pin RP1 Output Signal Select Bits				---	0 0000	69, 155
RPOR0	—	—	—	Remappable Pin RP0 Output Signal Select Bits				---	0 0000	69, 155

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved. **Bold** indicates shared access SFRs.

- Note**
- 1: Bit 21 of the PC is only available in Serial Programming (SP) modes.
  - 2: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.
  - 3: The SSPxMSK registers are only accessible when SSPxCON2<3:0> = 1001.
  - 4: Alternate names and definitions for these bits when the MSSP module is operating in I<sup>2</sup>C™ Slave mode. See **Section 18.5.3.2 "Address Masking Modes"** for details.
  - 5: These bits and/or registers are only available in 44-pin devices; otherwise, they are unimplemented and read as '0'. Reset values are shown for 44-pin devices.
  - 6: The PMADDRH/PMDOOUT1H and PMADDRL/PMDOOUT1L register pairs share the same physical registers and addresses, but have different functions determined by the module's operating mode. See **Section 10.1.2 "Data Registers"** for more information.



## 5.3.5 STATUS REGISTER

The STATUS register in Register 5-2, contains the arithmetic status of the ALU. The STATUS register can be the operand for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, then the write to these five bits is disabled.

These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the STATUS register as destination may be different than intended. For example, `CLRF STATUS` will set the Z bit but leave the other bits unchanged. The STATUS

register then reads back as '000u u1uu'. It is recommended, therefore, that only `BCF`, `BSE`, `SWAPF`, `MOVFF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions not affecting any Status bits, see the instruction set summary in Table 26-2 and Table 26-3.

**Note:** The C and DC bits operate as a borrow and digit borrow bits respectively, in subtraction.

### REGISTER 5-2: STATUS REGISTER (ACCESS FD8h)

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 <sup>(1)</sup>	C <sup>(2)</sup>
bit 7							bit 0

#### Legend:

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

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

bit 4 **N:** Negative bit

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

1 = Result was negative  
0 = Result was positive

bit 3 **OV:** Overflow bit

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

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

bit 2 **Z:** Zero bit

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

bit 1 **DC:** Digit carry/borrow bit<sup>(1)</sup>

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions:  
1 = A carry-out from the 4<sup>th</sup> low-order bit of the result occurred  
0 = No carry-out from the 4<sup>th</sup> low-order bit of the result

bit 0 **C:** Carry/borrow bit<sup>(2)</sup>

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions:  
1 = A carry-out from the MSb of the result occurred  
0 = No carry-out from the MSb of the result occurred

**Note 1:** 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.

**2:** 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.

# PIC18F46J11 FAMILY

## 5.4 Data Addressing Modes

**Note:** The execution of some instructions in the core PIC18 instruction set is changed when the PIC18 extended instruction set is enabled. See **Section 5.6 “Data Memory and the Extended Instruction Set”** for more information.

While the program memory can be addressed in only one way, through the PC, 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 more detail in **Section 5.6.1 “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 LSB. 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”**) 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 SFRs, 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. It also enables users to perform Indexed Addressing and other Stack Pointer operations for program memory in data memory.

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

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

## 5.4.3.1 FSR Registers and the INDF Operand (INDF)

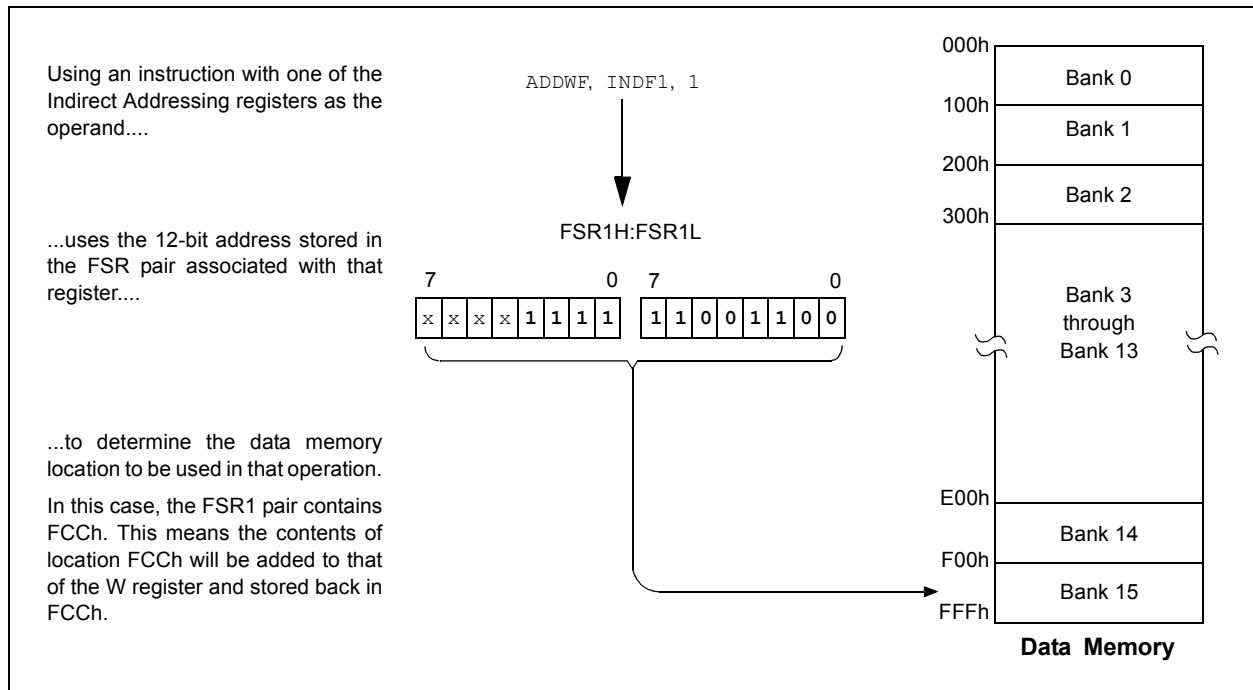
At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. The four upper bits of the FSRnH register are not used, so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of 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 INDF operands, INDF0 through INDF2. These can be presumed to be “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.

**FIGURE 5-8: INDIRECT ADDRESSING**



# PIC18F46J11 FAMILY

---

## 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’ thereafter
- POSTINC: accesses the FSR value, then automatically increments it by ‘1’ thereafter
- 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 the value 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.).

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 *NOE*.

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 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 appropriate caution that they do not inadvertently change settings that might affect the operation of the device.

## 5.5 Program Memory and the Extended Instruction Set

The operation of program memory is unaffected by the use of the extended instruction set.

Enabling the extended instruction set adds five additional two-word commands to the existing PIC18 instruction set: *ADDFSR*, *CALLW*, *MOVSE*, *MOVSS* and *SUBFSR*. These instructions are executed as described in **Section 5.2.4 “Two-Word Instructions”**.

## 5.6 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. This mode also alters the behavior of Indirect Addressing using FSR2 and its associated operands.

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 remains unchanged.

## 5.6.1 INDEXED ADDRESSING WITH LITERAL OFFSET

Enabling the PIC18 extended instruction set changes the behavior of Indirect Addressing using the FSR2 register pair and its associated file operands. Under proper conditions, instructions that use the Access Bank, that is, most bit 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.

## 5.6.2 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 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 and bit-oriented instructions are not affected if they 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 provided in Figure 5-9.

Those who desire to use byte 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 26.2.1 "Extended Instruction Syntax"**.

# PIC18F46J11 FAMILY

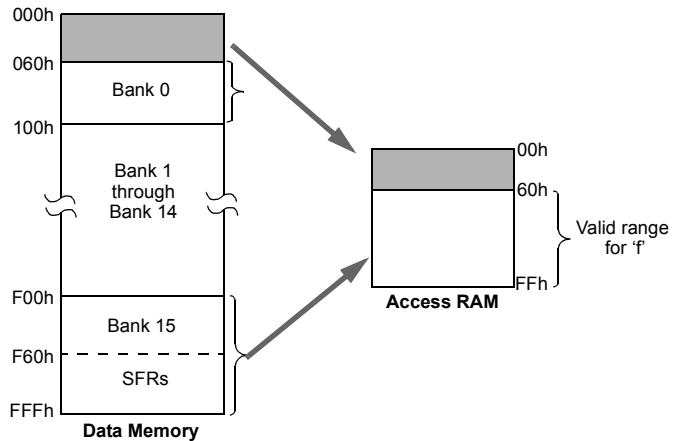
**FIGURE 5-9: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)**

**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 FFFh. This is the same as locations F60h to FFFh (Bank 15) of data memory.

Locations below 060h are not available in this addressing mode.

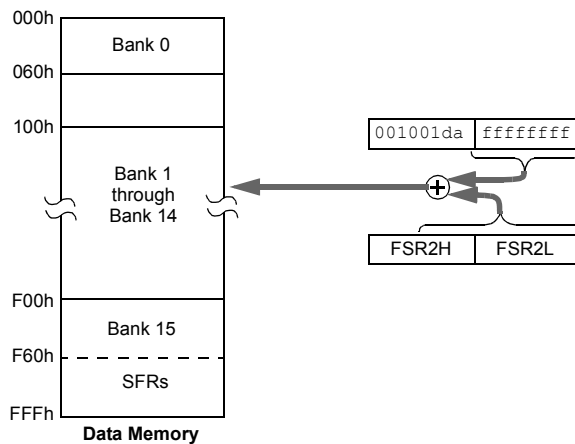


**When a = 0 and f ≤ 5Fh:**

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.

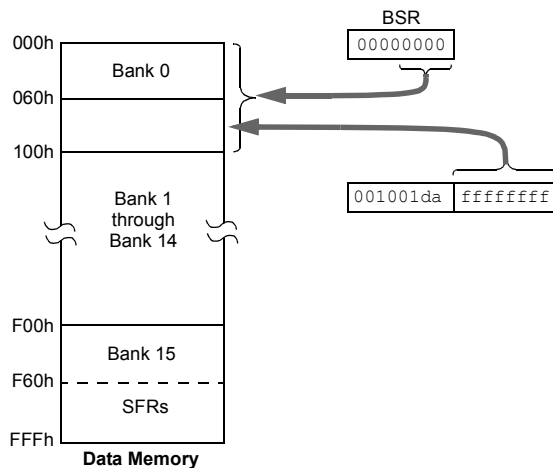
Note that in this mode, the correct syntax is:

```
ADDWF [k], d
where 'k' is 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.



## 5.6.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

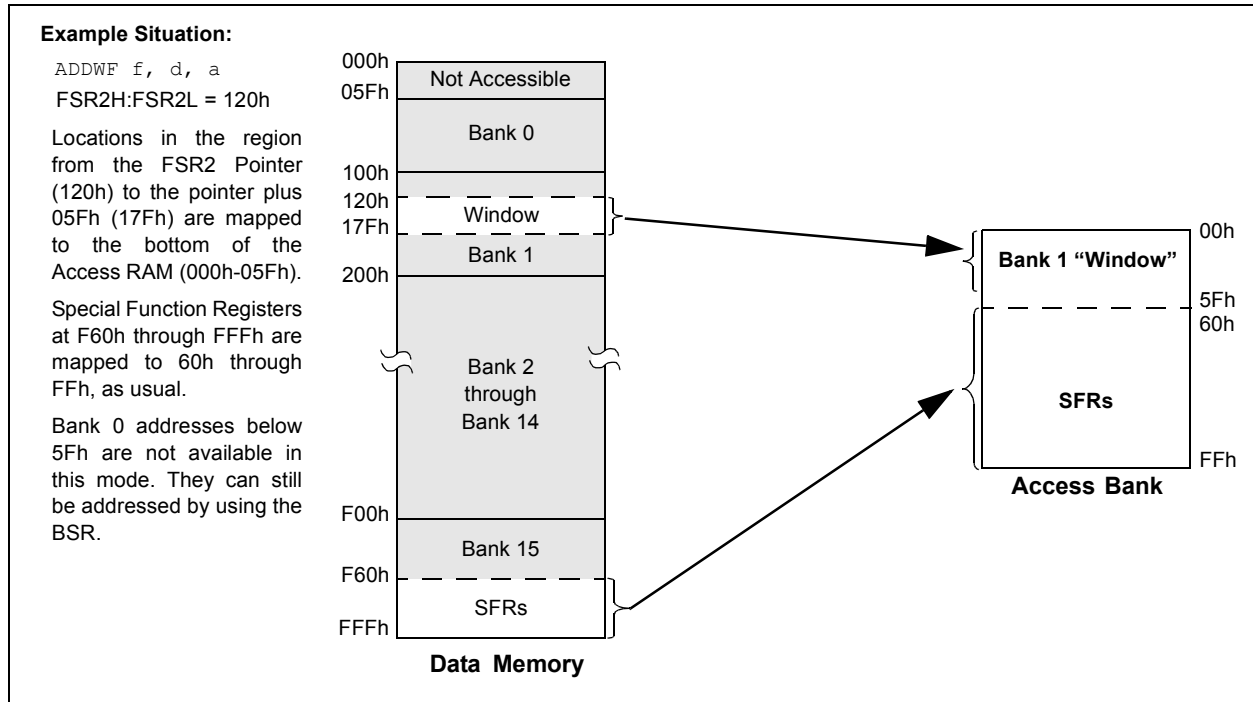
The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined “window” that can be located anywhere in the data memory space. The value of FSR2 establishes the lower boundary of the addresses mapped to 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”**). Figure 5-10 provides an example of Access Bank remapping in this addressing mode.

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. Any Indirect or Indexed Addressing operation that explicitly uses any of the indirect file operands (including FSR2) will continue to operate as standard Indirect Addressing. Any instruction that uses the Access Bank, but includes a register address of greater than 05Fh, will use Direct Addressing and the normal Access Bank map.

## 5.6.4 BSR IN INDEXED LITERAL OFFSET MODE

Although the Access Bank is remapped when the extended instruction set is enabled, the operation of the BSR remains unchanged. Direct Addressing, using the BSR to select the data memory bank, operates in the same manner as previously described.

**FIGURE 5-10: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING**



# PIC18F46J11 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 1 byte at a time. A write to program memory is executed on blocks of 64 bytes at a time or 2 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)

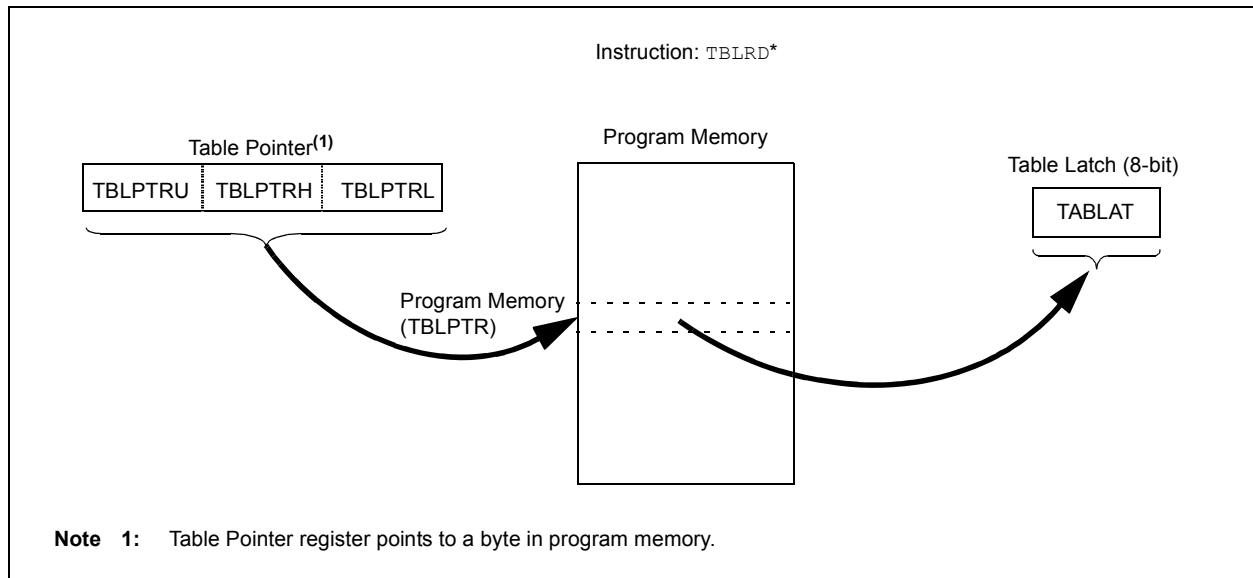
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

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

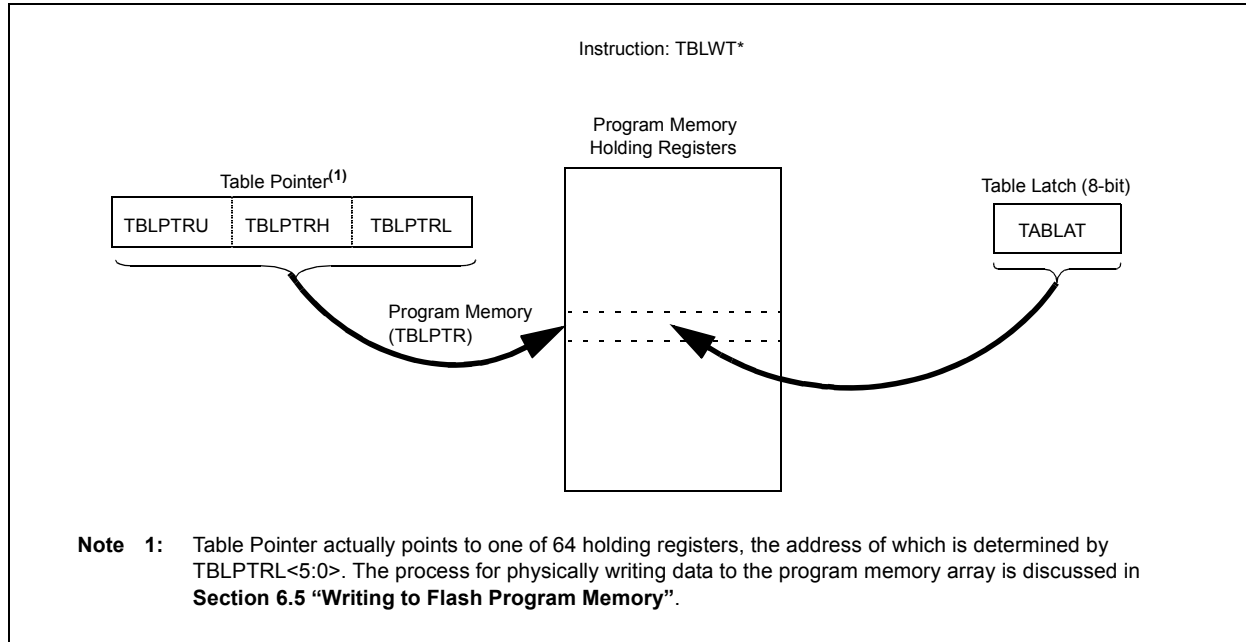
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions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

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



# PIC18F46J11 FAMILY

FIGURE 6-2: TABLE WRITE OPERATION



## 6.2 Control Registers

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

- 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 WPROG bit, when set, will allow programming two bytes per word on the execution of the WR command. If this bit is cleared, the WR command will result in programming on a block of 64 bytes.

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.

# PIC18F46J11 FAMILY

## REGISTER 6-1: EECON1: EEPROM CONTROL REGISTER 1 (ACCESS FA6h)

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

<b>Legend:</b>	S = Settable bit (cannot be cleared in software)
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	U = Unimplemented bit, read as '0'
	'0' = Bit is cleared
	x = Bit is unknown

- bit 7-6     **Unimplemented:** Read as '0'
- bit 5       **WPROG:** One Word-Wide Program bit  
             1 = Program 2 bytes on the next WR command  
             0 = Program 64 bytes on the next WR command
- bit 4       **FREE:** Flash Erase Enable bit  
             1 = Perform an erase operation on the next WR command (cleared by hardware after completion of erase)  
             0 = Perform write only
- bit 3       **WRERR:** Flash Program 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 Write Enable bit  
             1 = Allows write cycles to Flash program memory  
             0 = Inhibits write cycles to Flash program memory
- bit 1       **WR:** Write Control bit  
             1 = Initiates a program memory erase cycle or write cycle  
                 (The operation is self-timed and the bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)  
             0 = Write cycle is complete
- bit 0       **Unimplemented:** Read as '0'

# PIC18F46J11 FAMILY

## 6.2.2 TABLE LATCH REGISTER (TABLAT)

The Table Latch (TABLAT) is an 8-bit register mapped into the Special Function Register (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 TABLE POINTER REGISTER (TBLPTR)

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 22<sup>nd</sup> 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.

Table 6-1 provides these operations. 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 seven Least Significant bits (LSbs) of the Table Pointer register (TBLPTR<6:0>) 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 12 Most Significant bits (MSBs) of the TBLPTR (TBLPTR<21:10>) determine which program memory block of 1024 bytes is written to. For more information, see **Section 6.5 “Writing to Flash Program Memory”**.

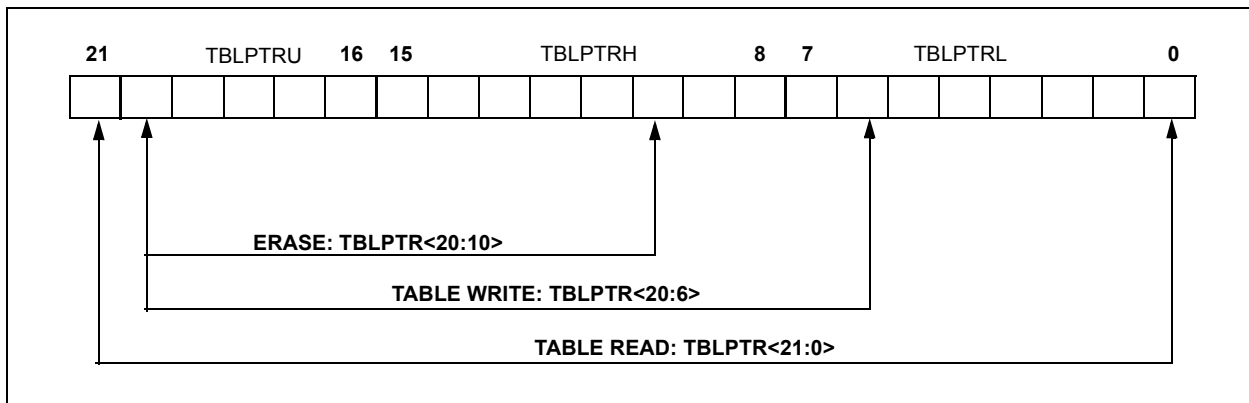
When an erase of program memory is executed, the 12 MSBs of the Table Pointer register point to the 1024-byte block that will be erased. The LSbs are ignored.

Figure 6-3 illustrates 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

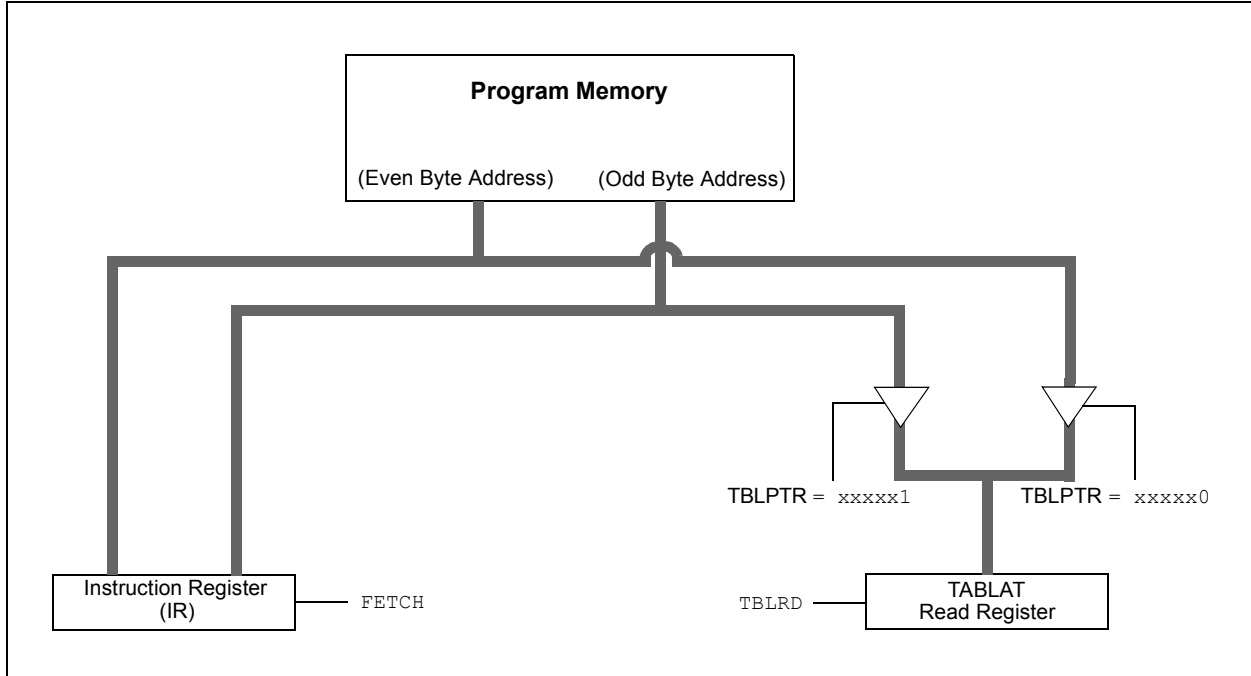
The `TBLRD` instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

`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 LSB of the address selects between the high and low bytes of the word.

Figure 6-4 illustrates 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
    MOVF   TABLAT, W           ; get data
    MOVWF  WORD_ODD
    
```

# PIC18F46J11 FAMILY

## 6.4 Erasing Flash Program Memory

The minimum erase block is 512 words or 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 12 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 WREN and FREE bits (EECON1<2,4>) to enable the erase operation.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit; this will begin the erase cycle.
7. The CPU will stall for the duration of the erase for TIE (see parameter D133B).
8. Re-enable interrupts.

#### EXAMPLE 6-2: ERASING 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_ROW				
	BSF	EECON1, WREN		; enable write to memory
	BSF	EECON1, FREE		; enable Erase operation
	BCF	INTCON, GIE		; disable interrupts
<b>Required Sequence</b>	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 programming block is 32 words or 64 bytes. Programming one word or 2 bytes at a time is also 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 (if WPROG = 0). 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.

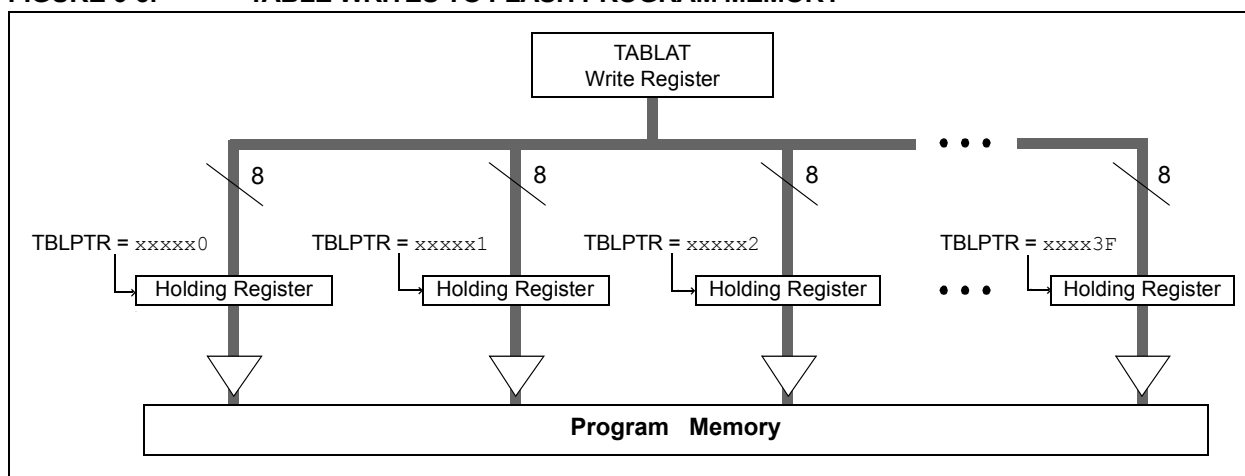
The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The 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 1:** Unlike previous PIC® devices, devices of the PIC18F46J11 family do not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence.

**2:** To maintain the endurance of the program memory cells, each Flash byte should not be programmed more than once between erase operations. Before attempting to modify the contents of the target cell a second time, an erase of the target page, or a bulk erase of the entire memory, must be performed.

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



### 6.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

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

1. Read 1024 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer register with address being erased.
4. Execute the erase procedure.
5. Load Table Pointer register with address of first byte being written, minus 1.
6. Write the 64 bytes into the holding registers with auto-increment.
7. Set the WREN bit (EECON1<2>) to enable byte writes.

8. Disable interrupts.
9. Write 55h to EECON2.
10. Write 0AAh to EECON2.
11. Set the WR bit. This will begin the write cycle.
12. The CPU will stall for the duration of the write for T<sub>W</sub> (see parameter D133A).
13. Re-enable interrupts.
14. Repeat steps 6 through 13 until all 1024 bytes are written to program memory.
15. Verify the memory (table read).

An example of the required code is provided in Example 6-3 on the following page.

**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.

# PIC18F46J11 FAMILY

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

	MOVLW	CODE_ADDR_UPPER		; Load TBLPTR with the base address
	MOVWF	TBLPTRU		; of the memory block, minus 1
	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 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_BYTE_TO_HREGS		
PROGRAM_MEMORY				
	BSF	EECON1, WREN		; enable write to memory
	BCF	INTCON, GIE		; disable interrupts
	MOVLW	55h		
<b>Required Sequence</b>	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



# PIC18F46J11 FAMILY

## 6.5.2 FLASH PROGRAM MEMORY WRITE SEQUENCE (WORD PRORAMMING).

The PIC18F46J11 family of devices has a feature that allows programming a single word (two bytes). This feature is enabled when the WPROG bit is set. If the memory location is already erased, the following sequence is required to enable this feature:

1. Load the Table Pointer register with the address of the data to be written. (It must be an even address.)
2. Write the 2 bytes into the holding registers by performing table writes. (Do not post-increment on the second table write.)
3. Set the WREN bit (EECON1<2>) to enable writes and the WPROG bit (EECON1<5>) to select Word Write mode.
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 the duration of the write for  $T_{iw}$  (see parameter D133A).
9. Re-enable interrupts.

### EXAMPLE 6-4: SINGLE-WORD WRITE TO FLASH PROGRAM MEMORY

```
MOVLW CODE_ADDR_UPPER      ; Load TBLPTR with the base address
MOVWF TBLPTRU
MOVLW CODE_ADDR_HIGH
MOVWF TBLPTRH
MOVLW CODE_ADDR_LOW        ; The table pointer must be loaded with an even
                           ; address
MOVWF TBLPTRL

MOVLW DATA0                ; LSB of word to be written
MOVWF TABLAT
TBLWT*+
MOVLW DATA1                ; MSB of word to be written
MOVWF TABLAT
TBLWT*                      ; The last table write must not increment the table
                           ; pointer! The table pointer needs to point to the
                           ; MSB before starting the write operation.

PROGRAM_MEMORY
BSF EECON1, WPROG           ; enable single word write
BSF EECON1, WREN           ; enable write to memory
BCF INTCON, GIE            ; disable interrupts
MOVLW 55h
Required MOVWF EECON2        ; write 55h
Sequence MOVLW 0AAh
MOVWF EECON2                ; write AAh
BSF EECON1, WR             ; start program (CPU stall)
BSF INTCON, GIE           ; re-enable interrupts
BCF EECON1, WPROG         ; disable single word write
BCF EECON1, WREN          ; disable write to memory
```

# PIC18F46J11 FAMILY

## 6.5.3 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.

## 6.5.4 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.

## 6.6 Flash Program Operation During Code Protection

See **Section 25.6 “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>)					63
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								63
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								63
TABLAT	Program Memory Table Latch								63
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	63
EECON2	Program Memory Control Register 2 (not a physical register)								65
EECON1	—	—	WPROG	FREE	WRERR	WREN	WR	—	65

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used during Flash program memory access.

## 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. Table 7-1 provides a comparison of various hardware and software multiply operations, along with the savings in memory and execution time.

### 7.2 Operation

Example 7-1 provides 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 provides the instruction sequence for 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		
				@ 48 MHz	@ 10 MHz	@ 4 MHz
8 x 8 unsigned	Without hardware multiply	13	69	5.7 $\mu$ s	27.6 $\mu$ s	69 $\mu$ s
	Hardware multiply	1	1	83.3 ns	400 ns	1 $\mu$ s
8 x 8 signed	Without hardware multiply	33	91	7.5 $\mu$ s	36.4 $\mu$ s	91 $\mu$ s
	Hardware multiply	6	6	500 ns	2.4 $\mu$ s	6 $\mu$ s
16 x 16 unsigned	Without hardware multiply	21	242	20.1 $\mu$ s	96.8 $\mu$ s	242 $\mu$ s
	Hardware multiply	28	28	2.3 $\mu$ s	11.2 $\mu$ s	28 $\mu$ s
16 x 16 signed	Without hardware multiply	52	254	21.6 $\mu$ s	102.6 $\mu$ s	254 $\mu$ s
	Hardware multiply	35	40	3.3 $\mu$ s	16.0 $\mu$ s	40 $\mu$ s

# PIC18F46J11 FAMILY

Example 7-3 provides the instruction sequence for a 16 x 16 unsigned multiplication. Equation 7-1 provides the algorithm that is used. The 32-bit result is stored in four registers (RES<3:0>).

## 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 provides the sequence to do a 16 x 16 signed multiply. Equation 7-2 provides the algorithm used. The 32-bit result is stored in four registers (RES<3:0>). 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

Devices of the PIC18F46J11 family 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 13 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 PIC® 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.

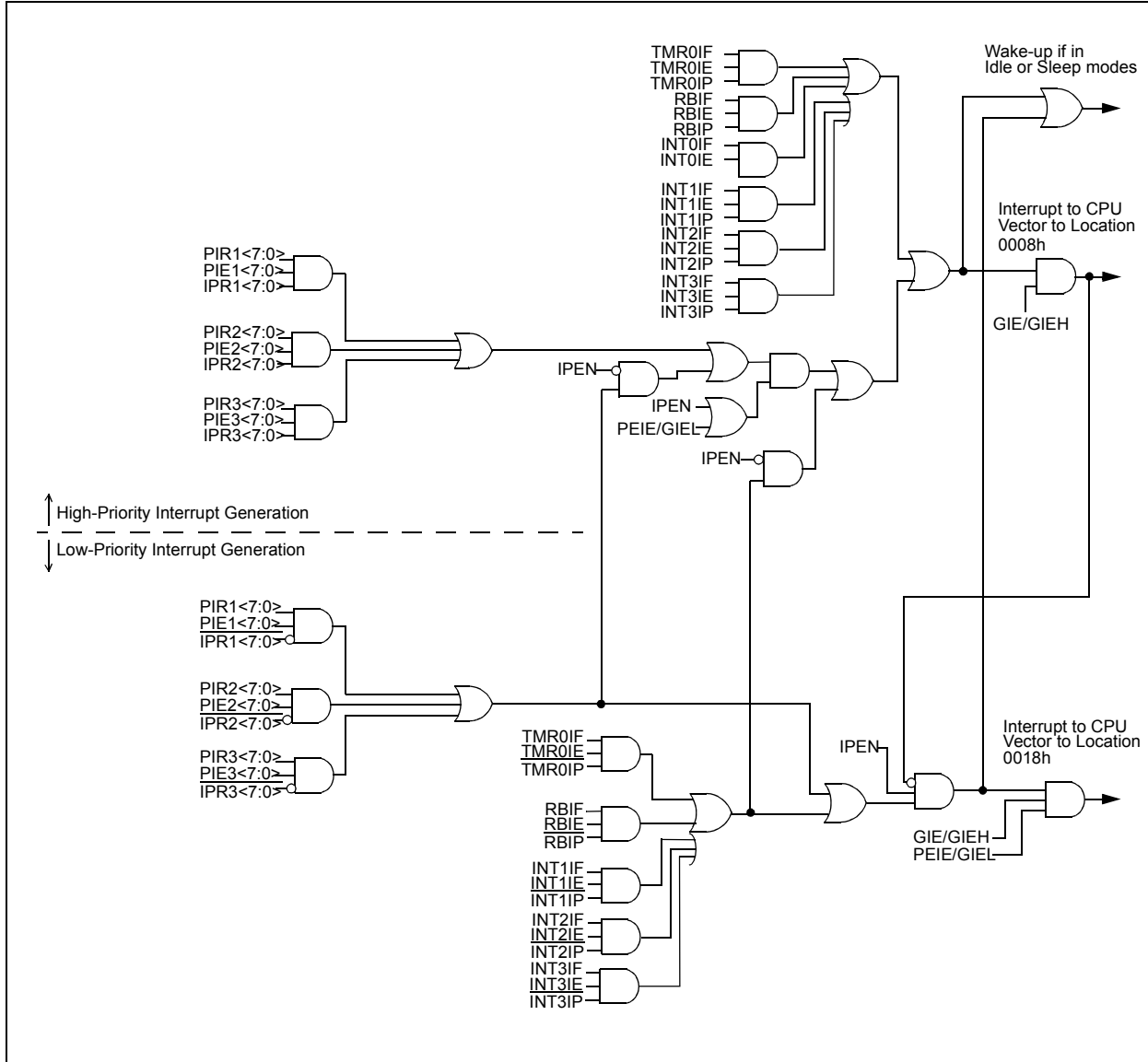
The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INTx 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.

# PIC18F46J11 FAMILY

FIGURE 8-1: PIC18F46J11 FAMILY INTERRUPT LOGIC



# PIC18F46J11 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 (ACCESS FF2h)

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 <sup>(1)</sup>
bit 7							bit 0

#### Legend:

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

- bit 7      **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<sup>(1)</sup>  
 1 = At least one of the RB<7:4> pins changed state (must be cleared in software)  
 0 = None of the RB<7:4> pins have changed state

**Note 1:** A mismatch condition will continue to set this bit. Reading PORTB and waiting 1 T<sub>cy</sub> will end the mismatch condition and allow the bit to be cleared.

# PIC18F46J11 FAMILY

## REGISTER 8-2: INTCON2: INTERRUPT CONTROL REGISTER 2 (ACCESS FF1h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
bit 7							bit 0

### Legend:

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

- bit 7  **$\overline{\text{RBPU}}$** : 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 **INTEDG3**: External Interrupt 3 Edge Select bit  
 1 = Interrupt on rising edge  
 0 = Interrupt on falling edge
- bit 2 **TMR0IP**: TMR0 Overflow Interrupt Priority bit  
 1 = High priority  
 0 = Low priority
- bit 1 **INT3IP**: INT3 External Interrupt Priority bit  
 1 = High priority  
 0 = Low priority
- bit 0 **RBIP**: RB Port Change Interrupt Priority bit  
 1 = High priority  
 0 = Low priority

**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.



# PIC18F46J11 FAMILY

## REGISTER 8-3: INTCON3: INTERRUPT CONTROL REGISTER 3 (ACCESS FF0h)

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **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      **INT3IE:** INT3 External Interrupt Enable bit  
          1 = Enables the INT3 external interrupt  
          0 = Disables the INT3 external interrupt
- 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      **INT3IF:** INT3 External Interrupt Flag bit  
          1 = The INT3 external interrupt occurred (must be cleared in software)  
          0 = The INT3 external interrupt did not occur
- 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

**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.

# PIC18F46J11 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 (ACCESS F9Eh)

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

#### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared

x = Bit is unknown

- bit 7 **PMPIF:** Parallel Master Port Read/Write Interrupt Flag bit<sup>(1)</sup>  
1 = A read or a write operation has taken place (must be cleared in software)  
0 = No read or write has occurred
- 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 **RC1IF:** EUSART1 Receive Interrupt Flag bit  
1 = The EUSART1 receive buffer, RCREG1, is full (cleared when RCREG1 is read)  
0 = The EUSART1 receive buffer is empty
- bit 4 **TX1IF:** EUSART1 Transmit Interrupt Flag bit  
1 = The EUSART1 transmit buffer, TXREG1, is empty (cleared when TXREG1 is written)  
0 = The EUSART1 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 Interrupt Flag bit  
Capture mode:  
1 = A TMR1/TMR3 register capture occurred (must be cleared in software)  
0 = No TMR1/TMR3 register capture occurred  
Compare mode:  
1 = A TMR1/TMR3 register compare match occurred (must be cleared in software)  
0 = No TMR1/TMR3 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

**Note 1:** These bits are unimplemented on 28-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 8-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2 (ACCESS FA1h)

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF
bit 7							bit 0

### Legend:

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

- bit 7      **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      **CM2IF:** Comparator 2 Interrupt Flag bit  
 1 = Comparator input has changed (must be cleared in software)  
 0 = Comparator input has not changed
- bit 5      **CM1IF:** Comparator 1 Interrupt Flag bit  
 1 = Comparator input has changed (must be cleared in software)  
 0 = Comparator input has not changed
- bit 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      **LVDIF:** High/Low-Voltage Detect (HLVD) Interrupt Flag bit  
 1 = A high/low-voltage condition occurred (must be cleared in software)  
 0 = An HLVD event has not occurred
- bit 1      **TMR3IF:** TMR3 Overflow Interrupt Flag bit  
 1 = TMR3 register overflowed (must be cleared in software)  
 0 = TMR3 register did not overflow
- bit 0      **CCP2IF:** ECCP2 Interrupt Flag bit  
Capture mode:  
 1 = A TMR1/TMR3 register capture occurred (must be cleared in software)  
 0 = No TMR1/TMR3 register capture occurred  
Compare mode:  
 1 = A TMR1/TMR3 register compare match occurred (must be cleared in software)  
 0 = No TMR1/TMR3 register compare match occurred  
PWM mode:  
 Unused in this mode.

# PIC18F46J11 FAMILY

## REGISTER 8-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3 (ACCESS FA4h)

R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **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      **RC2IF:** EUSART2 Receive Interrupt Flag bit  
1 = The EUSART2 receive buffer, RCREG2, is full (cleared when RCREG2 is read)  
0 = The EUSART2 receive buffer is empty
- bit 4      **TX2IF:** EUSART2 Transmit Interrupt Flag bit  
1 = The EUSART2 transmit buffer, TXREG2, is empty (cleared when TXREG2 is written)  
0 = The EUSART2 transmit buffer is full
- bit 3      **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit  
1 = TMR4 to PR4 match occurred (must be cleared in software)  
0 = No TMR4 to PR4 match occurred
- bit 2      **CTMUIF:** Charge Time Measurement Unit Interrupt Flag bit  
1 = A CTMU event has occurred (must be cleared in software)  
0 = CTMU event has not occurred
- bit 1      **TMR3GIF:** Timer3 Gate Event Interrupt Flag bit  
1 = A Timer3 gate event completed (must be cleared in software)  
0 = No Timer3 gate event completed
- bit 0      **RTCCIF:** RTCC Interrupt Flag bit  
1 = RTCC interrupt occurred (must be cleared in software)  
0 = No RTCC interrupt occurred

## 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 (ACCESS F9Dh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

#### Legend:

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

- bit 7      **PMPIE:** Parallel Master Port Read/Write Interrupt Enable bit<sup>(1)</sup>  
           1 = Enables the PMP read/write interrupt  
           0 = Disables the PMP read/write interrupt
- bit 6      **ADIE:** A/D Converter Interrupt Enable bit  
           1 = Enables the A/D interrupt  
           0 = Disables the A/D interrupt
- bit 5      **RC1IE:** EUSART1 Receive Interrupt Enable bit  
           1 = Enables the EUSART1 receive interrupt  
           0 = Disables the EUSART1 receive interrupt
- bit 4      **TX1IE:** EUSART1 Transmit Interrupt Enable bit  
           1 = Enables the EUSART1 transmit interrupt  
           0 = Disables the EUSART1 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 Interrupt Enable bit  
           1 = Enables the ECCP1 interrupt  
           0 = Disables the ECCP1 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

**Note 1:** These bits are unimplemented on 28-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 8-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2 (ACCESS FA0h)

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **OSCFIE:** Oscillator Fail Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled
- bit 6      **CM2IE:** Comparator 2 Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled
- bit 5      **CM1IE:** Comparator 1 Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled
- bit 4      **Unimplemented:** Read as '0'
- bit 3      **BCL1IE:** Bus Collision Interrupt Enable bit (MSSP1 module)  
           1 = Enabled  
           0 = Disabled
- bit 2      **LVDIE:** High/Low-Voltage Detect Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled
- bit 1      **TMR3IE:** TMR3 Overflow Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled
- bit 0      **CCP2IE:** ECCP2 Interrupt Enable bit  
           1 = Enabled  
           0 = Disabled

# PIC18F46J11 FAMILY

## REGISTER 8-9: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3 (ACCESS FA3h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE
bit 7							bit 0

### Legend:

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

- bit 7      **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      **RC2IE:** EUSART2 Receive Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 4      **TX2IE:** EUSART2 Transmit Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 3      **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 2      **CTMUIE:** Charge Time Measurement Unit (CTMU) Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 1      **TMR3GIE:** Timer3 Gate Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 0      **RTCCIE:** RTCC Interrupt Enable bit  
1 = Enabled  
0 = Disabled

# PIC18F46J11 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 (ACCESS F9Fh)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PMPIP <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **PMPIP:** Parallel Master Port Read/Write Interrupt Priority bit<sup>(1)</sup>  
1 = High priority  
0 = Low priority
- bit 6      **ADIP:** A/D Converter Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 5      **RC1IP:** EUSART1 Receive Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 4      **TX1IP:** EUSART1 Transmit Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 3      **SSP1IP:** Master Synchronous Serial Port Interrupt Priority bit (MSSP1 module)  
1 = High priority  
0 = Low priority
- bit 2      **CCP1IP:** ECCP1 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

**Note 1:** These bits are unimplemented on 28-pin devices.



# PIC18F46J11 FAMILY

## REGISTER 8-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2 (ACCESS FA2h)

R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP
bit 7							bit 0

### Legend:

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

- bit 7      **OSCFIP:** Oscillator Fail Interrupt Priority bit  
           1 = High priority  
           0 = Low priority
- bit 6      **CM2IP:** Comparator 2 Interrupt Priority bit  
           1 = High priority  
           0 = Low priority
- bit 5      **C12IP:** Comparator 1 Interrupt Priority bit  
           1 = High priority  
           0 = Low priority
- bit 4      **Unimplemented:** Read as '0'
- bit 3      **BCL1IP:** Bus Collision Interrupt Priority bit (MSSP1 module)  
           1 = High priority  
           0 = Low priority
- bit 2      **LVDIP:** High/Low-Voltage Detect Interrupt Priority bit  
           1 = High priority  
           0 = Low priority
- bit 1      **TMR3IP:** TMR3 Overflow Interrupt Priority bit  
           1 = High priority  
           0 = Low priority
- bit 0      **CCP2IP:** ECCP2 Interrupt Priority bit  
           1 = High priority  
           0 = Low priority

# PIC18F46J11 FAMILY

## REGISTER 8-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3 (ACCESS FA5h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **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      **RC2IP:** EUSART2 Receive Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 4      **TX2IP:** EUSART2 Transmit Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 3      **TMR4IE:** TMR4 to PR4 Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 2      **CTMUIP:** Charge Time Measurement Unit (CTMU) Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 1      **TMR3GIP:** Timer3 Gate Interrupt Priority bit  
1 = High priority  
0 = Low priority
- bit 0      **RTCCIP:** RTCC Interrupt Priority bit  
1 = High priority  
0 = Low priority

## 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 mode. RCON also contains the bit that enables interrupt priorities (IPEN).

**REGISTER 8-13: RCON: RESET CONTROL REGISTER (ACCESS FD0h)**

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

**Legend:**

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

- bit 7      **IPEN:** Interrupt Priority Enable bit  
           1 = Enable priority levels on interrupts  
           0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **CM:** Configuration Mismatch Flag bit  
           For details on bit operation, see Register 4-1.
- bit 4      **RI:** RESET Instruction Flag bit  
           For details on bit operation, see Register 4-1.
- bit 3      **TO:** Watchdog Timer Time-out Flag bit  
           For details on bit operation, see Register 4-1.
- bit 2      **PD:** Power-Down Detection Flag bit  
           For details on bit operation, see Register 4-1.
- bit 1      **POR:** Power-on Reset Status bit  
           For details on bit operation, see Register 4-1.
- bit 0      **BOR:** Brown-out Reset Status bit  
           For details on bit operation, see Register 4-1.

# PIC18F46J11 FAMILY

## 8.6 INTx Pin Interrupts

External interrupts on the INT0, INT1, INT2 and INT3 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 INTx pin, the corresponding flag bit and INTxIF are 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, INT2 and INT3) 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, INT2 and INT3 is determined by the value contained in the Interrupt Priority bits, INT1IP (INTCON3<6>), INT2IP (INTCON3<7>) and INT3IP (INTCON2<1>). 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 11.0 “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
MOVF  W_TEMP, W        ; Restore WREG
MOVFF  STATUS_TEMP, STATUS ; Restore STATUS
```

## 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.

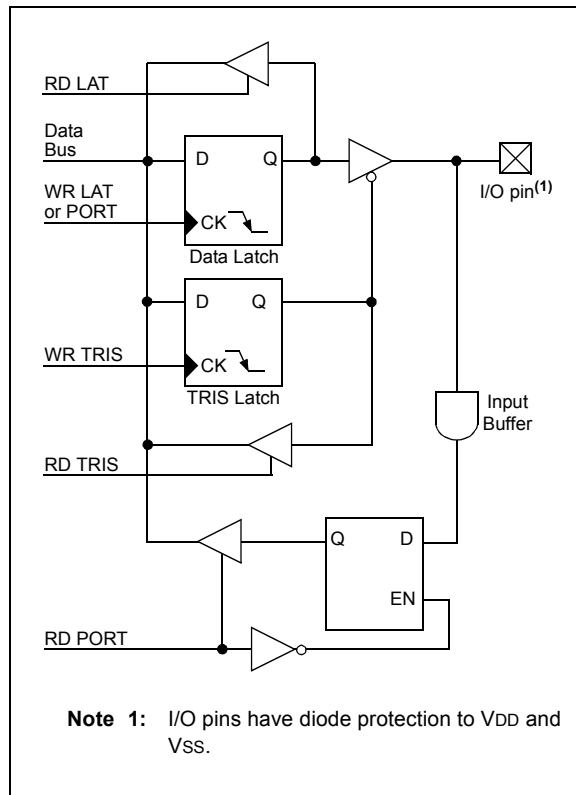
Each port has three registers for its operation. These registers are:

- TRIS register (Data Direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (Data Latch)

The Data Latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving.

Figure 9-1 displays a simplified model of a generic I/O port, without the interfaces to other peripherals.

**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 28.0 “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 28.0 “Electrical Characteristics”** for more details.

**TABLE 9-2: INPUT VOLTAGE LEVELS**

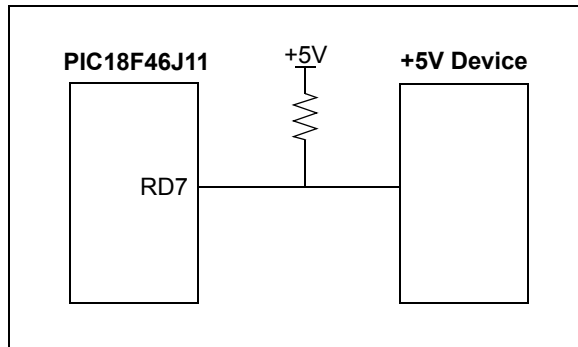
Port or Pin	Tolerated Input	Description
PORTA<7:0>	VDD	Only VDD input levels tolerated.
PORTB<3:0>		
PORTC<2:0>		
PORTE<2:0>	5.5V	Tolerates input levels above VDD, useful for most standard logic.
PORTB<7:4>		
PORTC<7:3>		
PORTD<7:0>		

# PIC18F46J11 FAMILY

## 9.1.3 INTERFACING TO A 5V SYSTEM

Though the  $V_{DDMAX}$  of the PIC18F46J11 family is 3.6V, these devices are still capable of interfacing with 5V systems, even if the  $V_{IH}$  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
```

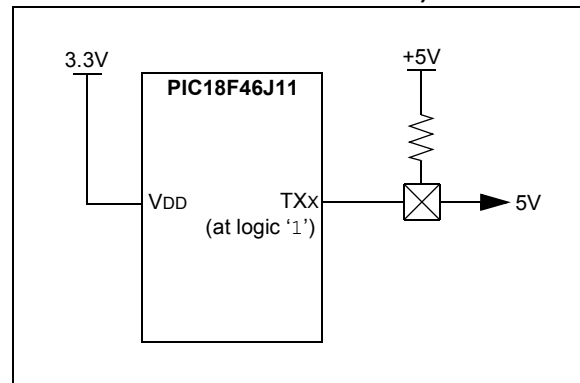
## 9.1.4 OPEN-DRAIN OUTPUTS

The output pins for several peripherals are also equipped with a configurable open-drain output option. This allows the peripherals to communicate with external digital logic operating at a higher voltage level, without the use of level translators.

The open-drain option is implemented on port pins specifically associated with the data and clock outputs of the EUSARTs, the MSSP modules (in SPI mode) and the ECCP modules. It is selectively enabled by setting the open-drain control bit for the corresponding module in the ODCON registers (Register 9-1, Register 9-2 and Register 9-3). Their configuration is discussed in more detail with the individual port where these peripherals are multiplexed.

When the open-drain option is required, the output pin must also be tied through an external pull-up resistor provided by the user to a higher voltage level, up to 5.5V (Figure 9-3). When a digital logic high signal is output, it is pulled up to the higher voltage level.

**FIGURE 9-3: USING THE OPEN-DRAIN OUTPUT (USART SHOWN AS EXAMPLE)**



## 9.1.5 TTL INPUT BUFFER OPTION

Many of the digital I/O ports use Schmitt Trigger (ST) input buffers. While this form of buffering works well with many types of input, some applications may require TTL level signals to interface with external logic devices. This is particularly true for the Parallel Master Port (PMP), which is likely to be interfaced to TTL level logic or memory devices.

The inputs for the PMP can be optionally configured for TTL buffers with the PMPTTL bit in the PADCFG1 register (Register 9-4). Setting this bit configures all data and control input pins for the PMP to use TTL buffers. By default, these PMP inputs use the port's ST buffers.

# PIC18F46J11 FAMILY

## REGISTER 9-1: ODCON1: PERIPHERAL OPEN-DRAIN CONTROL REGISTER 1 (BANKED F42h)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	ECCP2OD	ECCP1OD
bit 7						bit 0	

### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

- bit 7-2     **Unimplemented:** Read as '0'
- bit 1     **ECCP2OD:** ECCP2 Open-Drain Output Enable bit  
1 = Open-drain capability enabled  
0 = Open-drain capability disabled
- bit 0     **ECCP1OD:** ECCP1 Open-Drain Output Enable bit  
1 = Open-drain capability enabled  
0 = Open-drain capability disabled

## REGISTER 9-2: ODCON2: PERIPHERAL OPEN-DRAIN CONTROL REGISTER 2 (BANKED F41h)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	U2OD	U1OD
bit 7						bit 0	

### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

- bit 7-2     **Unimplemented:** Read as '0'
- bit 1     **U2OD:** USART2 Open-Drain Output Enable bit  
1 = Open-drain capability enabled  
0 = Open-drain capability disabled
- bit 0     **U1OD:** USART1 Open-Drain Output Enable bit  
1 = Open-drain capability enabled  
0 = Open-drain capability disabled

# PIC18F46J11 FAMILY

## REGISTER 9-3: ODCON3: PERIPHERAL OPEN-DRAIN CONTROL REGISTER 3 (BANKED F40h)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPI2OD	SPI1OD
bit 7						bit 0	

### Legend:

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

- bit 7-2            **Unimplemented:** Read as '0'
- bit 1            **SPI2OD:** SPI2 Open-Drain Output Enable bit  
                   1 = Open-drain capability enabled  
                   0 = Open-drain capability disabled
- bit 0            **SPI1OD:** SPI1 Open-Drain Output Enable bit  
                   1 = Open-drain capability enabled  
                   0 = Open-drain capability disabled

## REGISTER 9-4: PADCFG1: PAD CONFIGURATION CONTROL REGISTER 1 (BANKED F3Ch)

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	RTSECSEL1 <sup>(1)</sup>	RTSECSEL0 <sup>(1)</sup>	PMPTTL
bit 7						bit 0	

### Legend:

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

- bit 7-3            **Unimplemented:** Read as '0'
- bit 2-1            **RTSECSEL<1:0>:** RTCC Seconds Clock Output Select bits<sup>(1)</sup>  
                   11 = Reserved; do not use  
                   10 = RTCC source clock is selected for the RTCC pin (can be INTRC or T1OSC, depending on the  
                                      RTCOSC (CONFIG3L<1>) setting)  
                   01 = RTCC seconds clock is selected for the RTCC pin  
                   00 = RTCC alarm pulse is selected for the RTCC pin
- bit 0            **PMPTTL:** PMP Module TTL Input Buffer Select bit  
                   1 = PMP module uses TTL input buffers  
                   0 = PMP module uses Schmitt Trigger input buffers

**Note 1:** To enable the actual RTCC output, the RTCOE (RTCCFG<2>) bit needs to be set.



## 9.2 PORTA, TRISA and LATA Registers

PORTA is a 7-bit wide, bidirectional port. It may function as a 5-bit port, depending on the oscillator mode selected. 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, RA<3:0> 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 by setting the appropriate bits in the CMCON register. To use RA<3:0> as digital inputs, it is also necessary to turn off the comparators.

**Note:** On a Power-on Reset (POR), RA5 and RA<3:0> 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
```

# PIC18F46J11 FAMILY

**TABLE 9-3: PORTA I/O SUMMARY**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RA0/AN0/C1INA/ ULPWU/RP0	RA0	1	I/O	DIG	PORTA<0> data input; disabled when analog input enabled.
		0		TTL	LATA<0> data output; not affected by analog input.
	AN0	1	I	ANA	A/D input channel 0 and Comparator C1- input. Default input configuration on POR; does not affect digital output.
	C1INA	1	I	ANA	Comparator 1 input A.
	ULPWU	1	I	ANA	Ultra low-power wake-up input.
	RP0	1	I/O	ST	Remappable peripheral pin 0 input.
0		DIG		Remappable peripheral pin 0 output.	
RA1/AN1/C2INA/ PMA7/RP1	RA1	1	I	DIG	PORTA<1> data input; disabled when analog input enabled.
		0	O	TTL	LATA<1> data output; not affected by analog input.
	AN1	1	I	ANA	A/D input channel 1 and Comparator C2- input. Default input configuration on POR; does not affect digital output.
	C2INA	1	I	ANA	Comparator 1 input A.
	PMA7 <sup>(1)</sup>	1	I/O	ST/ TTL	Parallel Master Port.
		0		DIG	Parallel Master Port address.
RP1	1	I/O	ST	Remappable peripheral pin 1 input.	
	0		DIG	Remappable peripheral pin 1 output	
RA2/AN2/ VREF-/CVREF/ C2INB	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.
	C2INB	1	I	ANA	Comparator 2 input B.
0		O	ANA	CTMU pulse generator charger for the C2INB comparator input.	
RA3/AN3/VREF+/ C1INB	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.
	C1INB	1	I	ANA	Comparator 1 input B

**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:** This bit is only available on 44-pin devices.

# PIC18F46J11 FAMILY

**TABLE 9-3: PORTA I/O SUMMARY (CONTINUED)**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RA5/AN4/SS1/ HLVDIN/RP2	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.
	SS1	1	I	TTL	Slave select input for MSSP1.
	HLVDIN	1	I	ANA	High/Low-Voltage Detect external trip point reference input.
	RP2	1	I	ST	Remappable Peripheral pin 2 input.
0		O	DIG	Remappable Peripheral pin 2 output.	
OSC2/CLKO/ RA6	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.
	RA6	1	I	TTL	PORTA<6> data input.
		0	O	DIG	LATA<6> data output.
OSC1/CLKI/RA7	OSC1	1	I	ANA	Main oscillator input connection.
	CLKI	1	I	ANA	Main clock input connection.
	RA7	1	I	TTL	PORTA<6> data input.
		0	O	DIG	LATA<6> data output.

**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:** This bit is only available on 44-pin devices.

**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	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0	81
LATA	LAT7	LAT6	LAT5	—	LAT3	LAT2	LAT1	LAT0	81
TRISA	TRIS7	TRIS6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	81
ANCON0	PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5 <sup>(1)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	82
CMxCON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	81
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	82

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

**Note 1:** These bits are only available in 44-pin devices.

# PIC18F46J11 FAMILY

## 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 0x3F      ; Configure as digital I/O
MOVWF WREG ADCON1 ; pins in this example

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 POR.

**Note:** On a POR, the RB<3:0> bits are configured as analog inputs by default and read as '0'; RB<7:4> bits are configured as digital inputs.

Four of the PORTB pins (RB<7:4>) have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB<7:4> pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB<7:4>) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB<7:4> 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 (ISR), can clear the interrupt using the following steps:

1. Any read or write of PORTB (except with the MOVFF (ANY), PORTB instruction).
2. Wait one instruction cycle (such as executing a NOP instruction).
3. Clear flag bit, RBIF.

A mismatch condition continues to set flag bit, RBIF. Reading PORTB will end the mismatch condition and allow flag bit, RBIF, to be cleared after one instruction cycle of delay.

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.

The RB5 pin is multiplexed with the Timer0 module clock input and one of the comparator outputs to become the RB5/KBI1/SDI1/SDA1/RP8 pin.

# PIC18F46J11 FAMILY

**TABLE 9-5: PORTB I/O SUMMARY**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RB0/AN12/ INT0/RP3	RB0	1	1	TTL	PORTB<0> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. <sup>(1)</sup>
		0	O	DIG	LATB<0> data output; not affected by analog input.
	AN12	1	I	ANA	A/D input channel 12. <sup>(1)</sup>
	INT0	1	I	ST	External interrupt 0 input.
	RP3	1	I	ST	Remappable peripheral pin 3 input.
0		O	DIG	Remappable peripheral pin 3 output.	
RB1/AN10/ RTCC/RP4	RB1	1	I	TTL	PORTB<1> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. <sup>(1)</sup>
		0	O	DIG	LATB<1> data output; not affected by analog input.
	AN10	1	I	ANA	A/D input channel 10. <sup>(1)</sup>
	RTCC	0	O	DIG	Asynchronous serial transmit data output (USART module).
	RP4	1	I	ST	Remappable peripheral pin 4 input.
0		O	DIG	Remappable peripheral pin 4 output.	
RB2/AN8/ CTEDG1/ REFO/RP5	RB2	1	I	TTL	PORTB<2> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared. Disabled when analog input enabled. <sup>(1)</sup>
		0	O	DIG	LATB<2> data output; not affected by analog input.
	AN8	1	I	ANA	A/D input channel 8. <sup>(1)</sup>
	CTEDG1	1	I	ST	CTMU Edge 1 input.
	REFO	0	O	DIG	Reference output clock.
RP5	1	I	ST	Remappable peripheral pin 5 input.	
	0	O	DIG	Remappable peripheral pin 5 output.	
RB3/AN9/ CTEDG2/ PMA2/RP6	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. <sup>(1)</sup>
	AN9	1	I	ANA	A/D input channel 9. <sup>(1)</sup>
	CTEDG2	1	I	ST	CTMU edge 2 input.
	PMA2 <sup>(3)</sup>	0	O	DIG	Parallel Master Port address.
	RP6	1	I	ST	Remappable peripheral pin 6 input.
0		O	DIG	Remappable peripheral pin 6 output.	

**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 P $\overline{\text{BADEN}}$  Configuration bit. Pins are configured as analog inputs by default when P $\overline{\text{BADEN}}$  is set and digital inputs when P $\overline{\text{BADEN}}$  is cleared.

**2:** All other pin functions are disabled when ICSP™ or ICD are enabled.

**3:** This bit is not available on 28-pin devices.

# PIC18F46J11 FAMILY

**TABLE 9-5: PORTB I/O SUMMARY (CONTINUED)**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RB4/PMA1/ KBI0/RP7	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. <sup>(1)</sup>
	PMA1 <sup>(3)</sup>	0	O	DIG	Parallel Master Port address.
	KBI0	1	I	TTL	Interrupt-on-change pin.
	RP7	1	I	ST	Remappable peripheral pin 7 input.
		0	O	DIG	Remappable peripheral pin 7 output.
RB5/PMA0/ KBI1/RP8	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.
	PMA0 <sup>(3)</sup>	0	O	DIG	Parallel Master Port address.
	KBI1	1	I	TTL	Interrupt-on-change pin.
	RP8	1	I	ST	Remappable peripheral pin 8 input.
		0	O	DIG	Remappable peripheral pin 8 output.
RB6/KBI2/ PGC/RP9	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. <sup>(2)</sup>
	RP9	1	I	ST	Remappable peripheral pin 9 input.
		0	O	DIG	Remappable peripheral pin 9 output.
RB7/KBI3/ PGD/RP10	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. <sup>(2)</sup>
		x	I	ST	Serial execution data input for ICSP and ICD operation. <sup>(2)</sup>
	RP10	1	I	ST	Remappable peripheral pin 10 input.
		0	O	ST	Remappable peripheral pin 10 output.

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

**3:** This bit is not available on 28-pin devices.

# PIC18F46J11 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	81
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	81
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	81
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	81
INTCON2	RBP $\bar{U}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	81
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	81
ADCON0	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	81

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

# PIC18F46J11 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 (see Table ). The pins have Schmitt Trigger input buffers.

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.

Unlike other PORTC pins, RC4 and RC5 do not have TRISC bits associated with them. As digital ports, they can only function as digital inputs. If an external transceiver is used, RC4 and RC5 always function as inputs from the transceiver. If the on-chip transceiver is used, the data direction is determined by the operation being performed by the module at that time.

**Note:** On a Power-on Reset, PORTC pins (except RC2, RC4 and RC5) are configured as digital inputs. RC2 will default as an analog input (controlled by the ANCON1 register).

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 0x3F    ; Value used to
                ; initialize data
                ; direction
MOVWF TRISC   ; Set RC<5:0> as inputs
                ; RC<7:6> as outputs
MOVLB 0x0F    ; ANCON register is not in
                ; Access Bank
BSF   ANCON1,PCFG11
                ;Configure RC2/AN11 as
                ; digital input
```



# PIC18F46J11 FAMILY

**TABLE 9-7: PORTC I/O SUMMARY<sup>(1)</sup>**

Pin	Function	TRIS Setting	I/O	I/O Type	Description	
RC0/T1OSO/ T1CKI/RP11	RC0	1	I	ST	PORTC<0> data input.	
		0	O	DIG	LATC<0> data output.	
	T1OSO	x	O	ANA	Timer1 oscillator output; enabled when Timer1 oscillator enabled. Disables digital I/O.	
	T1CKI	1	I	ST	Timer1 counter input.	
	RP11	1	I	ST	Remappable peripheral pin 11 input.	
0		O	DIG	Remappable peripheral pin 11 output.		
RC1/T1OSI/ RP12	RC1	1	I	ST	PORTC<1> data input.	
		0	O	DIG	LATC<1> data output.	
	T1OSI	x	I	ANA	Timer1 oscillator input; enabled when Timer1 oscillator enabled. Disables digital I/O.	
	RP12	1	I	ST	Remappable peripheral pin 12 input.	
0		O	DIG	Remappable peripheral pin 12 output.		
RC2/AN11/ CTPLS/RP13	RC2	1	I	ST	PORTC<2> data input.	
		0	O	DIG	LATC<2> data output.	
	AN11	1	I	ANA	A/D input channel 11.	
	CTPLS	0	O	DIG	CTMU pulse generator output.	
		1	I	ST	Remappable peripheral pin 13 input.	
RC3/SCK1/ SCL1/RP14	RC3	1	I	ST	PORTC<3> data input.	
		0	O	DIG	LATC<3> data output.	
	SCK1	1	I	ST/TTL	Parallel Master Port <code>io_addr_in&lt;1&gt;</code> .	
		0	O	DIG	Parallel Master Port address.	
	SCL1	1	I	I <sup>2</sup> C/ SMBus	I <sup>2</sup> C™ clock input (MSSP1 module).	
		0	O	DIG	I <sup>2</sup> C clock output (MSSP1 module).	
	RP14	1	I	ST	Remappable peripheral pin 14 input.	
		0	O	DIG	Remappable peripheral pin 14 output.	
	RC4/SDI1/ SDA1/RP15	RC4	1	I	ST	PORTC<4> data input.
			0	O	DIG	LATC<4> data output.
SDI1		1	I	ST	SPI data input (MSSP1 module).	
SDA1		1	I	I <sup>2</sup> C/ SMBus	I <sup>2</sup> C data input (MSSP1 module).	
		0	O	DIG	I <sup>2</sup> C/SMBus.	
RP15		1	I	ST	Remappable peripheral pin 15 input.	
	0	O	DIG	Remappable peripheral pin 15 output.		

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; I<sup>2</sup>C/SMB = I<sup>2</sup>C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option)

**Note 1:** Enhanced PWM output is available only on PIC18F4XJ11 devices.

**2:** This bit is only available on 44-pin devices.

# PIC18F46J11 FAMILY

**TABLE 9-7: PORTC I/O SUMMARY<sup>(1)</sup> (CONTINUED)**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RC5/SDO1/ RP16	RC5	1	I	DIG	PORTC<5> data input.
		0	O	DIG	LATC<5> data output.
	SDO1	0	O	DIG	SPI data output (MSSP1 module).
		RP16	1	I	ST
	0		O	DIG	Remappable peripheral pin 16 output.
RC6/PMA5/ TX1/CK1/RP17	RC6	1	I	ST	PORTC<6> data input.
		0	O	DIG	LATC<6> data output.
	PMA5 <sup>(2)</sup>	1	I	ST/TTL	Parallel Master Port <code>io_addr_in&lt;5&gt;</code> .
		0	O	DIG	Parallel Master Port address.
	TX1	0	O	DIG	Asynchronous serial transmit data output (EUSART module); takes priority over port data. User must configure as output.
		CK1	1	I	ST
			0	O	DIG
	RP17	1	I	ST	Remappable peripheral pin 17 input.
0		O	DIG	Remappable peripheral pin 17 output.	
RC7/RX1/DT1/ RP18	RC7	1	I	ST	PORTC<7> data input.
		0	O	DIG	LATC<7> data output.
	RX1	1	I	ST	Asynchronous serial receive data input (EUSART module).
	DT1	1	I	ST	Synchronous serial data input (EUSART module). User must configure as an input.
		0	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
	RP18	1	I	ST	Remappable peripheral pin 18 input.
0		O	DIG	Remappable peripheral pin 18 output.	

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; I<sup>2</sup>C/SMB = I<sup>2</sup>C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option)

**Note 1:** Enhanced PWM output is available only on PIC18F4XJ11 devices.

**2:** This bit is only available on 44-pin devices.

**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	81
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	81
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	81

## 9.5 PORTD, TRISD and LATD Registers

**Note:** PORTD is available only in 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.

**Note:** On a POR, these pins are configured as digital inputs.

### 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
```

Each of the PORTD pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by setting bit, RDPUL (PORTE<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 POR.

Note that the pull-ups can be used for any set of features, similar to the pull-ups found on PORTB.

# PIC18F46J11 FAMILY

**TABLE 9-9: PORTD I/O SUMMARY**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RD0/PMD0/ SCL2	RD0	1	I	ST	PORTD<0> data input.
		0	O	DIG	LATD<0> data output.
	PMD0	1	I	ST/TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	SCL2	1	I	I <sup>2</sup> C/ SMB	I <sup>2</sup> C™ clock input (MSSP2 module); input type depends on module setting.
0	O	DIG	I <sup>2</sup> C™ clock output (MSSP2 module); takes priority over port data.		
RD1/PMD1/ SDA2	RD1	1	I	ST	PORTD<1> data input.
		0	O	DIG	LATD<1> data output.
	PMD1	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	SDA2	1	I	I <sup>2</sup> C/ SMB	I <sup>2</sup> C data input (MSSP2 module); input type depends on module setting.
0	O	DIG	I <sup>2</sup> C data output (MSSP2 module); takes priority over port data.		
RD2/PMD2/ RP19	RD2	1	I	ST	PORTD<2> data input.
		0	O	DIG	LATD<2> data output.
	PMD2	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP19	1	I	ST	Remappable peripheral pin 19 input.
0	O	DIG	DIG	Remappable peripheral pin 19 output.	
RD3/PMD3/ RP20	RD3	1	I	DIG	PORTD<3> data input.
		0	O	DIG	LATD<3> data output.
	PMD3	1	I	ST/TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP20	1	I	ST	Remappable peripheral pin 20 input.
0	O	DIG	DIG	Remappable peripheral pin 20 output.	
RD4/PMD4/ RP21	RD4	1	I	ST	PORTD<4> data input.
		0	O	DIG	LATD<4> data output.
	PMD4	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP21	1	I	ST	Remappable peripheral pin 21 input.
0	O	DIG	DIG	Remappable peripheral pin 21 output.	
RD5/PMD5/ RP22	RD5	1	I	ST	PORTD<5> data input.
		0	O	DIG	LATD<5> data output.
	PMD5	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP22	1	I	ST	Remappable peripheral pin 22 input.
0	O	DIG	DIG	Remappable peripheral pin 22 output.	

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; I<sup>2</sup>C/SMB = I<sup>2</sup>C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

# PIC18F46J11 FAMILY

**TABLE 9-9: PORTD I/O SUMMARY (CONTINUED)**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RD6/PMD6/ RP23	RD6	1	I	ST	PORTD<6> data input.
		0	O	DIG	LATD<6> data output.
	PMD6	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP23	1	I	ST	Remappable peripheral pin 23 input.
		0	O	DIG	Remappable peripheral pin 23 output.
RD7/PMD7/ RP24	RD7	1	I	ST	PORTD<7> data input.
		0	O	DIG	LATD<7> data output.
	PMD7	1	I	TTL	Parallel Master Port data in.
		0	O	DIG	Parallel Master Port data out.
	RP24	1	I	ST	Remappable peripheral pin 24 input.
		0	O	DIG	Remappable peripheral pin 24 output.

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; I<sup>2</sup>C/SMB = I<sup>2</sup>C/SMBus input buffer; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**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 <sup>(1)</sup>	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	86
LATD <sup>(1)</sup>	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	86
TRISD <sup>(1)</sup>	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	86

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

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

# PIC18F46J11 FAMILY

## 9.6 PORTE, TRISE and LATE Registers

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

Depending on the particular PIC18F46J11 family device selected, PORTE is implemented in two different ways.

For 44-pin devices, PORTE is a 3-bit wide port. Three pins (RE0/AN5/PMRD, RE1/AN6/PMWR and RE2/AN7/PMCS) 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 POR, RE<2:0> are configured as analog inputs.

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

### 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
```

Each of the PORTE pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by setting bit, REPU (PORTE<6>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a POR.

Note that the pull-ups can be used for any set of features, similar to the pull-ups found on PORTB.

# PIC18F46J11 FAMILY

**TABLE 9-11: PORTE I/O SUMMARY**

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RE0/AN5/ PMRD	RE0	1	I	ST	PORTE<0> data input; disabled when analog input enabled.
		0	O	DIG	LATE<0> data output; not affected by analog input.
	AN5	1	I	ANA	A/D input channel 5; default input configuration on POR.
		PMRD	1	I	ST/TTL
	0		O	DIG	Parallel Master Port read strobe.
RE1/AN6/ PMWR	RE1	1	I	ST	PORTE<1> data input; disabled when analog input enabled.
		0	O	DIG	LATE<1> data output; not affected by analog input.
	AN6	1	I	ANA	A/D input channel 6; default input configuration on POR.
		PMWR	1	I	ST/TTL
	0		O	DIG	Parallel Master Port write strobe.
RE2/AN7/ PMCS	RE2	1	I	ST	PORTE<2> data input; disabled when analog input enabled.
		0	O	DIG	LATE<2> data output; not affected by analog input.
	AN7	1	I	ANA	A/D input channel 7; default input configuration on POR.
		PMCS	0	O	DIG
Vss1	—		—	P	—
Vss2					
AVss1	—	—	P	—	Ground reference for analog modules.
VDD1	—	—	P	—	Positive supply for peripheral digital logic and I/O pins.
VDD2					
VDDCORE/VCAP	VDDCORE	—	P	—	Positive supply for microcontroller core logic (regulator disabled).
	VCAP	—	P	—	External filter capacitor connection (regulator enabled).
AVDD1	—	—	P	—	Positive supply for analog modules.
AVDD2			—		

**Legend:** DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level  
I = Input; O = Output; P = Power

**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 <sup>(1)</sup>	RDPU <sup>(3)</sup>	REPU <sup>(4)</sup>	—	—	—	RE2	RE1	RE0	86
LATE <sup>(1)</sup>	—	—	—	—	—	LATE2	LATE1	LATE0	86
TRISE <sup>(1)</sup>	—	—	—	—	—	TRISE2	TRISE1	TRISE0	85
ANCON0	PCFG7 <sup>(2)</sup>	PCFG6 <sup>(2)</sup>	PCFG5 <sup>(2)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	87

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

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

**2:** These bits are only available in 44-pin devices.

**Note 1:** PORTD Pull-up Enable bit  
0 = All PORTD pull-ups are disabled  
1 = PORTD pull-ups are enabled for any input pad

**2:** PORTE Pull-up Enable bit  
0 = All PORTE pull-ups are disabled  
1 = PORTE pull-ups are enabled for any input pad

# PIC18F46J11 FAMILY

---

## 9.7 Peripheral Pin Select (PPS)

A major challenge in general purpose devices is providing the largest possible set of peripheral features while minimizing the conflict of features on I/O pins. The challenge is even greater on low pin count devices similar to the PIC18F46J11 family. In an application that needs to use more than one peripheral multiplexed on single pin, inconvenient workarounds in application code or a complete redesign may be the only option.

The Peripheral Pin Select (PPS) feature provides an alternative to these choices by enabling the user's peripheral set selection and their placement on a wide range of I/O pins. By increasing the pinout options available on a particular device, users can better tailor the microcontroller to their entire application, rather than trimming the application to fit the device.

The PPS feature operates over a fixed subset of digital I/O pins. Users may independently map the input and/or output of any one of the many digital peripherals to any one of these I/O pins. PPS is performed in software and generally does not require the device to be reprogrammed. Hardware safeguards are included that prevent accidental or spurious changes to the peripheral mapping once it has been established.

### 9.7.1 AVAILABLE PINS

The PPS feature is used with a range of up to 22 pins; the number of available pins is dependent on the particular device and its pin count. Pins that support the PPS feature include the designation "RPn" in their full pin designation, where "RP" designates a remappable peripheral and "n" is the remappable pin number. See Table 1-2 for pinout options in each package offering.

### 9.7.2 AVAILABLE PERIPHERALS

The peripherals managed by the PPS are all digital only peripherals. These include general serial communications (UART and SPI), general purpose timer clock inputs, timer-related peripherals (input capture and output compare) and external interrupt inputs. Also included are the outputs of the comparator module, since these are discrete digital signals.

The PPS module is not applied to I<sup>2</sup>C, change notification inputs, RTCC alarm outputs or peripherals with analog inputs. Additionally, the MSSP1 and EUSART1 modules are not routed through the PPS module.

A key difference between pin select and non-pin select peripherals is that pin select peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non PPS peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

#### 9.7.2.1 Peripheral Pin Select Function Priority

When a pin selectable peripheral is active on a given I/O pin, it takes priority over all other digital I/O and digital communication peripherals associated with the pin. Priority is given regardless of the type of peripheral that is mapped. Pin select peripherals never take priority over any analog functions associated with the pin.

### 9.7.3 CONTROLLING PERIPHERAL PIN SELECT

PPS features are controlled through two sets of Special Function Registers (SFRs): one to map peripheral inputs and the other to map outputs. Because they are separately controlled, a particular peripheral's input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral selectable pin is handled in two different ways, depending on whether an input or an output is being mapped.



# PIC18F46J11 FAMILY

## 9.7.3.1 Input Mapping

The inputs of the PPS options are mapped on the basis of the peripheral; that is, a control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping (see Register 9-6 through Register 9-20). Each register contains a 5-bit field, which is associated

with one of the pin selectable peripherals. Programming a given peripheral's bit field with an appropriate 5-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any of the bit fields corresponds to the maximum number of peripheral pin selections supported by the device.

**TABLE 9-13: SELECTABLE INPUT SOURCES (MAPS INPUT TO FUNCTION)<sup>(1)</sup>**

Input Name	Function Name	Register	Configuration Bits
External Interrupt 1	INT1	RPINR1	INTR1R<4:0>
External Interrupt 2	INT2	RPINR2	INTR2R<4:0>
External Interrupt 3	INT3	RPINR3	INTR3R<4:0>
Timer0 External Clock Input	T0CKI	RPINR4	T0CKR<4:0>
Timer3 External Clock Input	T3CKI	RPINR6	T3CKR<4:0>
Input Capture 1	CCP1	RPINR7	IC1R<4:0>
Input Capture 2	CCP2	RPINR8	IC2R<4:0>
Timer1 Gate Input	T1G	RPINR12	T1GR<4:0>
Timer3 Gate Input	T3G	RPINR13	T3GR<4:0>
EUSART2 Asynchronous Receive/Synchronous Receive	RX2/DT2	RPINR16	RX2DT2R<4:0>
EUSART2 Asynchronous Clock Input	CK2	RPINR17	CK2R<4:0>
SPI2 Data Input	SDI2	RPINR21	SDI2R<4:0>
SPI2 Clock Input	SCK2IN	RPINR22	SCK2R<4:0>
SPI2 Slave Select Input	SS2IN	RPINR23	SS2R<4:0>
PWM Fault Input	FLT0	RPINR24	OCFAR<4:0>

**Note 1:** Unless otherwise noted, all inputs use the Schmitt Trigger input buffers.

# PIC18F46J11 FAMILY

## 9.7.3.2 Output Mapping

In contrast to inputs, the outputs of the PPS options are mapped on the basis of the pin. In this case, a control register associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers are used to control output mapping. The value of the bit field corresponds to one of the peripherals and that peripheral's output is mapped to the pin (see Table 9-14).

Because of the mapping technique, the list of peripherals for output mapping also includes a null value of '00000'. This permits any given pin to remain disconnected from the output of any of the pin selectable peripherals.

**TABLE 9-14: SELECTABLE OUTPUT SOURCES (MAPS FUNCTION TO OUTPUT)**

Function	Output Function Number <sup>(1)</sup>	Output Name
NULL	0	NULL <sup>(2)</sup>
C1OUT	1	Comparator 1 Output
C2OUT	2	Comparator 2 Output
TX2/CK2	5	EUSART2 Asynchronous Transmit/Asynchronous Clock Output
DT2	6	EUSART2 Synchronous Transmit
SDO2	9	SPI2 Data Output
SCK2	10	SPI2 Clock Output
SSDMA	12	SPI DMA Slave Select
ULPOUT	13	Ultra Low-Power Wake-up Event
CCP1/P1A	14	ECCP1 Compare or PWM Output Channel A
P1B	15	ECCP1 Enhanced PWM Output, Channel B
P1C	16	ECCP1 Enhanced PWM Output, Channel C
P1D	17	ECCP1 Enhanced PWM Output, Channel D
CCP2/P2A	18	ECCP2 Compare or PWM Output
P2B	19	ECCP2 Enhanced PWM Output, Channel B
P2C	20	ECCP2 Enhanced PWM Output, Channel C
P2D	21	ECCP2 Enhanced PWM Output, Channel D

**Note 1:** Value assigned to the RPn<4:0> pins corresponds to the peripheral output function number.

**2:** The NULL function is assigned to all RPn outputs at device Reset and disables the RPn output function.

### 9.7.3.3 Mapping Limitations

The control schema of the PPS is extremely flexible. Other than systematic blocks that prevent signal contention caused by two physical pins being configured as the same functional input or two functional outputs configured as the same pin, there are no hardware enforced lock outs. The flexibility extends to the point of allowing a single input to drive multiple peripherals or a single functional output to drive multiple output pins.

### 9.7.4 CONTROLLING CONFIGURATION CHANGES

Because peripheral remapping can be changed during run time, some restrictions on peripheral remapping are needed to prevent accidental configuration changes. PIC18F devices include three features to prevent alterations to the peripheral map:

- Control register lock sequence
- Continuous state monitoring
- Configuration bit remapping lock

#### 9.7.4.1 Control Register Lock

Under normal operation, writes to the RPINRx and RPORx registers are not allowed. Attempted writes will appear to execute normally, but the contents of the registers will remain unchanged. To change these registers, they must be unlocked in hardware. The register lock is controlled by the IOLOCK bit (PPSCON<0>). Setting IOLOCK prevents writes to the control registers; clearing IOLOCK allows writes.

To set or clear IOLOCK, a specific command sequence must be executed:

1. Write 55h to EECON2<7:0>.
2. Write AAh to EECON2<7:0>.
3. Clear (or set) IOLOCK as a single operation.

IOLOCK remains in one state until changed. This allows all of the PPS registers to be configured with a single unlock sequence followed by an update to all control registers, then locked with a second lock sequence.

#### 9.7.4.2 Continuous State Monitoring

In addition to being protected from direct writes, the contents of the RPINRx and RPORx registers are constantly monitored in hardware by shadow registers. If an unexpected change in any of the registers occurs (such as cell disturbances caused by ESD or other external events), a Configuration Mismatch Reset will be triggered.

### 9.7.4.3 Configuration Bit Pin Select Lock

As an additional level of safety, the device can be configured to prevent more than one write session to the RPINRx and RPORx registers. The IOL1WAY (CONFIG3H<0>) Configuration bit blocks the IOLOCK bit from being cleared after it has been set once. If IOLOCK remains set, the register unlock procedure will not execute and the PPS control registers cannot be written to. The only way to clear the bit and re-enable peripheral remapping is to perform a device Reset.

In the default (unprogrammed) state, IOL1WAY is set, restricting users to one write session. Programming IOL1WAY allows users unlimited access (with the proper use of the unlock sequence) to the PPS registers.

### 9.7.5 CONSIDERATIONS FOR PERIPHERAL PIN SELECTION

The ability to control peripheral pin selection introduces several considerations into application design that could be overlooked. This is particularly true for several common peripherals that are available only as remappable peripherals.

The main consideration is that the PPS is not available on default pins in the device's default (Reset) state. Since all RPINRx registers reset to '11111' and all RPORx registers reset to '00000', all PPS inputs are tied to RP31 and all PPS outputs are disconnected.

**Note:** In tying PPS inputs to RP31, RP31 does not have to exist on a device for the registers to be reset to it.

This situation requires the user to initialize the device with the proper peripheral configuration before any other application code is executed. Since the IOLOCK bit resets in the unlocked state, it is not necessary to execute the unlock sequence after the device has come out of Reset.

For application safety, however, it is best to set IOLOCK and lock the configuration after writing to the control registers.

Because the unlock sequence is timing critical, it must be executed as an assembly language routine. If the bulk of the application is written in C or another high-level language, the unlock sequence should be performed by writing in-line assembly.

# PIC18F46J11 FAMILY

Choosing the configuration requires the review of all PPSs and their pin assignments, especially those that will not be used in the application. In all cases, unused pin selectable peripherals should be disabled completely. Unused peripherals should have their inputs assigned to an unused RPn pin function. I/O pins with unused RPn functions should be configured with the null peripheral output.

The assignment of a peripheral to a particular pin does not automatically perform any other configuration of the pin's I/O circuitry. In theory, this means adding a pin selectable output to a pin may mean inadvertently driving an existing peripheral input when the output is driven. Users must be familiar with the behavior of other fixed peripherals that share a remappable pin and know when to enable or disable them. To be safe, fixed digital peripherals that share the same pin should be disabled when not in use.

Along these lines, configuring a remappable pin for a specific peripheral does not automatically turn that feature on. The peripheral must be specifically configured for operation and enabled, as if it were tied to a fixed pin. Where this happens in the application code (immediately following device Reset and peripheral configuration or inside the main application routine) depends on the peripheral and its use in the application.

A final consideration is that the PPS functions neither override analog inputs nor reconfigure pins with analog functions for digital I/O. If a pin is configured as an analog input on device Reset, it must be explicitly reconfigured as digital I/O when used with a PPS.

Example 9-7 provides a configuration for bidirectional communication with flow control using EUSART2. The following input and output functions are used:

- Input Function RX2
- Output Function TX2

## EXAMPLE 9-7: CONFIGURING EUSART2 INPUT AND OUTPUT FUNCTIONS

```

//*****
// Unlock Registers
//*****
_asm
; PPS registers are in BANK 14
MOVLB      0x0E
MOVLW      0x55
MOVWF      EECON2, 0
MOVLW      0xAA
MOVWF      EECON2, 0

; PPS Write Protect off
BCF        PPSCON, IOLOCK, BANKED
_endasm
//*****
// Configure Input Functions
// (See Table 9-13)
//*****
// Assign RX2 To Pin RP0
//*****
_asm
MOVLW      0x00
MOVWF      RPINR16, BANKED
_endasm
//*****
// Configure Output Functions
// (See Table 9-14)
//*****
// Assign TX2 To Pin RP1
//*****
_asm
MOVLW      0x05
MOVWF      RPOR1, BANKED
_endasm
//*****
// Lock Registers
//*****
_asm
MOVLW      0x55
MOVWF      EECON2, 0
MOVLW      0xAA
MOVWF      EECON2, 0

; PPS Write Protected
BSF        PPSCON, IOLOCK, BANKED
_endasm

```

**Note:** If the Configuration bit, IOL1WAY = 1, once the IOLOCK bit is set, it cannot be cleared, preventing any future RP register changes. The IOLOCK bit is cleared back to '0' on any device Reset.

# PIC18F46J11 FAMILY

## 9.7.6 PERIPHERAL PIN SELECT REGISTERS

The PIC18F46J11 family of devices implements a total of 37 registers for remappable peripheral configuration of 44-pin devices. The 28-pin devices have 31 registers for remappable peripheral configuration.

**Note:** Input and output register values can only be changed if PPS<IOLOCK> = 0. See Example 9-7 for a specific command sequence.

### REGISTER 9-5: PPSCON: PERIPHERAL PIN SELECT INPUT REGISTER 0 (BANKED EFFh)<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	IOLOCK
bit 7							bit 0

#### Legend:

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

bit 7-1                      **Unimplemented:** Read as '0'

bit 0                      **IOLOCK:** I/O Lock Enable bit

1 = I/O lock active, RPORx and RPINRx registers are write-protected

0 = I/O lock not active, pin configurations can be changed

**Note 1:** Register values can only be changed if PPSCON<IOLOCK> = 0.

# PIC18F46J11 FAMILY

## REGISTER 9-6: RPINR1: PERIPHERAL PIN SELECT INPUT REGISTER 1 (BANKED EE7h)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INTR1R4	INTR1R3	INTR1R2	INTR1R1	INTR1R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **INTR1R<4:0>:** Assign External Interrupt 1 (INT1) to the Corresponding RPn Pin bits

## REGISTER 9-7: RPINR2: PERIPHERAL PIN SELECT INPUT REGISTER 2 (BANKED EE8h)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INTR2R4	INTR2R3	INTR2R2	INTR2R1	INTR2R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **INTR2R<4:0>:** Assign External Interrupt 2 (INT2) to the Corresponding RPn pin bits

## REGISTER 9-8: RPINR3: PERIPHERAL PIN SELECT INPUT REGISTER 3 (BANKED EE9h)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INTR3R4	INTR3R3	INTR3R2	INTR3R1	INTR3R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **INTR3R<4:0>:** Assign External Interrupt 3 (INT3) to the Corresponding RPn Pin bits

# PIC18F46J11 FAMILY

## REGISTER 9-9: RPINR4: PERIPHERAL PIN SELECT INPUT REGISTER 4 (BANKED EEAh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T0CKR4	T0CKR3	T0CKR2	T0CKR1	T0CKR0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **T0CKR<4:0>:** Timer0 External Clock Input (T0CKI) to the Corresponding RPn Pin bits

## REGISTER 9-10: RPINR6: PERIPHERAL PIN SELECT INPUT REGISTER 6 (BANKED EECh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T3CKR4	T3CKR3	T3CKR2	T3CKR1	T3CKR0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **T3CKR<4:0>:** Timer 3 External Clock Input (T3CKI) to the Corresponding RPn Pin bits

## REGISTER 9-11: RPINR7: PERIPHERAL PIN SELECT INPUT REGISTER 7 (BANKED EEDh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC1R4	IC1R3	IC1R2	IC1R1	IC1R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **IC1R<4:0>:** Assign Input Capture 1 (ECCP1) to the Corresponding RPn Pin bits

# PIC18F46J11 FAMILY

## REGISTER 9-12: RPINR8: PERIPHERAL PIN SELECT INPUT REGISTER 8 (BANKED EEEh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC2R4	IC2R3	IC2R2	IC2R1	IC2R0
bit 7							bit 0

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

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

bit 4-0      **IC2R<4:0>:** Assign Input Capture 2 (ECCP2) to the Corresponding RPn Pin bits

## REGISTER 9-13: RPINR12: PERIPHERAL PIN SELECT INPUT REGISTER 12 (BANKED EF2h)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T1GR4	T1GR3	T1GR2	T1GR1	T1GR0
bit 7							bit 0

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

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

bit 4-0      **T1GR<4:0>:** Timer1 Gate Input (T1G) to the Corresponding RPn Pin bits

## REGISTER 9-14: RPINR13: PERIPHERAL PIN SELECT INPUT REGISTER 13 (BANKED EF3h)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T3GR4	T3GR3	T3GR2	T3GR1	T3GR0
bit 7							bit 0

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

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

bit 4-0      **T3GR<4:0>:** Timer3 Gate Input (T3G) to the Corresponding RPn Pin bits



# PIC18F46J11 FAMILY

## REGISTER 9-15: RPINR16: PERIPHERAL PIN SELECT INPUT REGISTER 16 (BANKED EF6h)

U-0	U-0	U-0	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1
—	—	—	RX2DT2R4	RX2DT2R3	RX2DT2R2	RX2DT2R1	RX2DT2R0
bit 7							bit 0

**Legend:** R $\bar{W}$  = Readable, Writable if IOLOCK = 0  
 R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
 -n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RX2DT2R<4:0>:** EUSART2 Synchronous/Asynchronous Receive (RX2/DT2) to the Corresponding RPn Pin bits

## REGISTER 9-16: RPINR17: PERIPHERAL PIN SELECT INPUT REGISTER 17 (BANKED EF7h)

U-0	U-0	U-0	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1
—	—	—	CK2R4	CK2R3	CK2R2	CK2R1	CK2R0
bit 7							bit 0

**Legend:** R $\bar{W}$  = Readable, Writable if IOLOCK = 0  
 R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
 -n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **CK2R<4:0>:** EUSART2 Clock Input (CK2) to the Corresponding RPn Pin bits

## REGISTER 9-17: RPINR21: PERIPHERAL PIN SELECT INPUT REGISTER 21 (BANKED EFBh)

U-0	U-0	U-0	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1	R $\bar{W}$ -1
—	—	—	SDI2R4	SDI2R3	SDI2R2	SDI2R1	SDI2R0
bit 7							bit 0

**Legend:** R $\bar{W}$  = Readable, Writable if IOLOCK = 0  
 R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
 -n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **SDI2R<4:0>:** Assign SPI2 Data Input (SDI2) to the Corresponding RPn Pin bits

# PIC18F46J11 FAMILY

## REGISTER 9-18: RPINR22: PERIPHERAL PIN SELECT INPUT REGISTER 22 (BANKED EFCh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	
—	—	—	SCK2R4	SCK2R3	SCK2R2	SCK2R1	SCK2R0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **SCK2R<4:0>:** Assign SPI2 Data Input (SDI2) to the Corresponding RPn Pin bits

## REGISTER 9-19: RPINR23: PERIPHERAL PIN SELECT INPUT REGISTER 23 (BANKED EFDh)

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	
—	—	—	SS2R4	SS2R3	SS2R2	SS2R1	SS2R0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **SS2R<4:0>:** Assign SPI2 Slave Select Input (SS2IN) to the Corresponding RPn Pin bits

## REGISTER 9-20: RPINR24: PERIPHERAL PIN SELECT INPUT REGISTER 24 (BANKED EFEh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	—	OCFAR4	OCFAR3	OCFAR2	OCFAR1	OCFAR0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **OCFAR<4:0>:** Assign PWM Fault Input (FLT0) to the Corresponding RPn Pin bits

# PIC18F46J11 FAMILY

## REGISTER 9-21: RPOR0: PERIPHERAL PIN SELECT OUTPUT REGISTER 0 (BANKED EC6h)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	—	RP0R4	RP0R3	RP0R2	RP0R1	RP0R0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP0R<4:0>:** Peripheral Output Function is Assigned to RP0 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** Register values can be changed only if PPSCON<IOLOCK> = 0.

## REGISTER 9-22: RPOR1: PERIPHERAL PIN SELECT OUTPUT REGISTER 1 (BANKED EC7h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	—	RP1R4	RP1R3	RP1R2	RP1R1	RP1R0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP1R<4:0>:** Peripheral Output Function is Assigned to RP1 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-23: RPOR2: PERIPHERAL PIN SELECT OUTPUT REGISTER 2 (BANKED EC8h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	—	RP2R4	RP2R3	RP2R2	RP2R1	RP2R0	
bit 7								bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP2R<4:0>:** Peripheral Output Function is Assigned to RP2 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

# PIC18F46J11 FAMILY

## REGISTER 9-24: RPOR3: PERIPHERAL PIN SELECT OUTPUT REGISTER 3 (BANKED EC9h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP3R4	RP3R3	RP3R2	RP3R1	RP3R0
bit 7							bit 0

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

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

bit 4-0      **RP3R<4:0>:** Peripheral Output Function is Assigned to RP3 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-25: RPOR4: PERIPHERAL PIN SELECT OUTPUT REGISTER 4 (BANKED ECAh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP4R4	RP4R3	RP4R2	RP4R1	RP4R0
bit 7							bit 0

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

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

bit 4-0      **RP4R<4:0>:** Peripheral Output Function is Assigned to RP4 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-26: RPOR5: PERIPHERAL PIN SELECT OUTPUT REGISTER 5 (BANKED ECBh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP5R4	RP5R3	RP5R2	RP5R1	RP5R0
bit 7							bit 0

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

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

bit 4-0      **RP5R<4:0>:** Peripheral Output Function is Assigned to RP5 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

# PIC18F46J11 FAMILY

## REGISTER 9-27: RPOR6: PERIPHERAL PIN SELECT OUTPUT REGISTER 6 (BANKED ECCh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP6R4	RP6R3	RP6R2	RP6R1	RP6R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0      **RP6R<4:0>:** Peripheral Output Function is Assigned to RP6 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-28: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTER 7 (BANKED ECDh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP7R4	RP7R3	RP7R2	RP7R1	RP7R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0      **RP7R<4:0>:** Peripheral Output Function is Assigned to RP7 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-29: RPOR8: PERIPHERAL PIN SELECT OUTPUT REGISTER 8 (BANKED ECEh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP8R4	RP8R3	RP8R2	RP8R1	RP8R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0      **RP8R<4:0>:** Peripheral Output Function is Assigned to RP8 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

# PIC18F46J11 FAMILY

## REGISTER 9-30: RPOR9: PERIPHERAL PIN SELECT OUTPUT REGISTER 9 (BANKED ECFh)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP9R4	RP9R3	RP9R2	RP9R1	RP9R0
bit 7							bit 0

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

bit 7-5      **Unimplemented:** Read as '0'  
 bit 4-0      **RP9R<4:0>:** Peripheral Output Function is Assigned to RP9 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)

## REGISTER 9-31: RPOR10: PERIPHERAL PIN SELECT OUTPUT REGISTER 10 (BANKED ED0h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP10R4	RP10R3	RP10R2	RP10R1	RP10R0
bit 7							bit 0

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

bit 7-5      **Unimplemented:** Read as '0'  
 bit 4-0      **RP10R<4:0>:** Peripheral Output Function is Assigned to RP10 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)

## REGISTER 9-32: RPOR11: PERIPHERAL PIN SELECT OUTPUT REGISTER 11 (BANKED ED1h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP11R4	RP11R3	RP11R2	RP11R1	RP11R0
bit 7							bit 0

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

bit 7-5      **Unimplemented:** Read as '0'  
 bit 4-0      **RP11R<4:0>:** Peripheral Output Function is Assigned to RP11 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)

# PIC18F46J11 FAMILY

## REGISTER 9-33: RPOR12: PERIPHERAL PIN SELECT OUTPUT REGISTER 12 (BANKED ED2h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP12R4	RP12R3	RP12R2	RP12R1	RP12R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0 **RP12R<4:0>:** Peripheral Output Function is Assigned to RP12 Output Pin bits (see Table 9-14 for peripheral function numbers)

## REGISTER 9-34: RPOR13: PERIPHERAL PIN SELECT OUTPUT REGISTER 13 (BANKED ED3h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP13R4	RP13R3	RP13R2	RP13R1	RP13R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0 **RP13R<4:0>:** Peripheral Output Function is Assigned to RP13 Output Pin bits (see Table 9-14 for peripheral function numbers)

## REGISTER 9-35: RPOR14: PERIPHERAL PIN SELECT OUTPUT REGISTER 14 (BANKED ED4h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP14R4	RP14R3	RP14R2	RP14R1	RP14R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown
	U = Unimplemented bit, read as '0'

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

bit 4-0 **RP14R<4:0>:** Peripheral Output Function is Assigned to RP14 Output Pin bits (see Table 9-14 for peripheral function numbers)

# PIC18F46J11 FAMILY

## REGISTER 9-36: RPOR15: PERIPHERAL PIN SELECT OUTPUT REGISTER 15 (BANKED ED5h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP15R4	RP15R3	RP15R2	RP15R1	RP15R0
bit 7							bit 0

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

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

bit 4-0      **RP15R<4:0>:** Peripheral Output Function is Assigned to RP15 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)

## REGISTER 9-37: RPOR16: PERIPHERAL PIN SELECT OUTPUT REGISTER 16 (BANKED ED6h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP16R4	RP16R3	RP16R2	RP16R1	RP16R0
bit 7							bit 0

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

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

bit 4-0      **RP16R<4:0>:** Peripheral Output Function is Assigned to RP16 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)

## REGISTER 9-38: RPOR17: PERIPHERAL PIN SELECT OUTPUT REGISTER 17 (BANKED ED7h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP17R4	RP17R3	RP17R2	RP17R1	RP17R0
bit 7							bit 0

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

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

bit 4-0      **RP17R<4:0>:** Peripheral Output Function is Assigned to RP17 Output Pin bits  
 (see Table 9-14 for peripheral function numbers)



# PIC18F46J11 FAMILY

## REGISTER 9-39: RPOR18: PERIPHERAL PIN SELECT OUTPUT REGISTER 18 (BANKED ED8h)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP18R4	RP18R3	RP18R2	RP18R1	RP18R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown

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

bit 4-0      **RP18R<4:0>:** Peripheral Output Function is Assigned to RP18 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

## REGISTER 9-40: RPOR19: PERIPHERAL PIN SELECT OUTPUT REGISTER 19 (BANKED ED9h)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP19R4	RP19R3	RP19R2	RP19R1	RP19R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown

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

bit 4-0      **RP19R<4:0>:** Peripheral Output Function is Assigned to RP19 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP19 pins are not available on 28-pin devices.

## REGISTER 9-41: RPOR20: PERIPHERAL PIN SELECT OUTPUT REGISTER 20 (BANKED EDAh)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP20R4	RP20R3	RP20R2	RP20R1	RP20R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	'0' = Bit is cleared
	x = Bit is unknown

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

bit 4-0      **RP20R<4:0>:** Peripheral Output Function is Assigned to RP20 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP20 pins are not available on 28-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 9-42: RPOR21: PERIPHERAL PIN SELECT OUTPUT REGISTER 21 (BANKED EDBh)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP21R4	RP21R3	RP21R2	RP21R1	RP21R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP21R<4:0>:** Peripheral Output Function is Assigned to RP21 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP21 pins are not available on 28-pin devices.

## REGISTER 9-43: RPOR22: PERIPHERAL PIN SELECT OUTPUT REGISTER 22 (BANKED EDCh)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP22R4	RP22R3	RP22R2	RP22R1	RP22R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP22R<4:0>:** Peripheral Output Function is Assigned to RP22 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP22 pins are not available on 28-pin devices.

## REGISTER 9-44: RPOR23: PERIPHERAL PIN SELECT OUTPUT REGISTER 23 (BANKED EDDh)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP23R4	RP23R3	RP23R2	RP23R1	RP23R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP23R<4:0>:** Peripheral Output Function is Assigned to RP23 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP23 pins are not available on 28-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 9-45: RPOR24: PERIPHERAL PIN SELECT OUTPUT REGISTER 24 (BANKED EDEh)<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP24R4	RP24R3	RP24R2	RP24R1	RP24R0
bit 7							bit 0

<b>Legend:</b>	R/W = Readable, Writable if IOLOCK = 0
R = Readable bit	W = Writable bit      U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

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

bit 4-0      **RP24R<4:0>:** Peripheral Output Function is Assigned to RP24 Output Pin bits  
(see Table 9-14 for peripheral function numbers)

**Note 1:** RP24 pins are not available on 28-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:

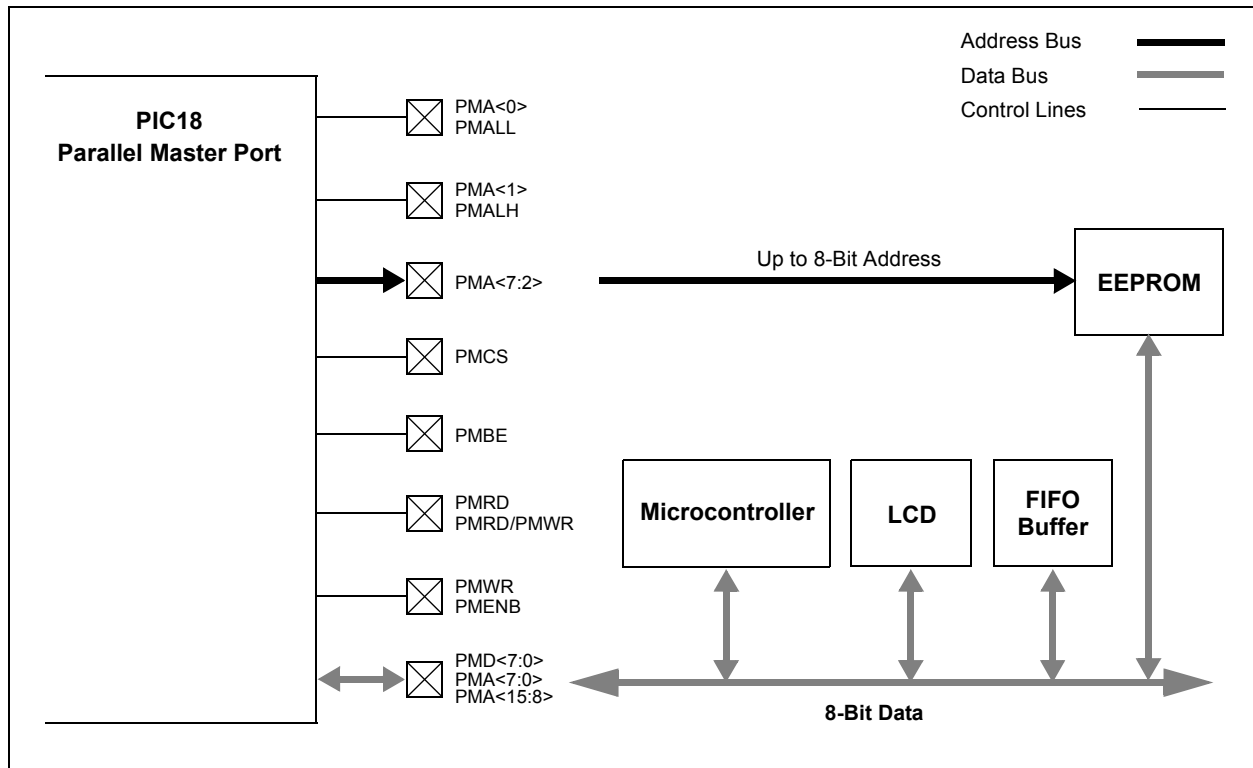
## 10.0 PARALLEL MASTER PORT (PMP)

The Parallel Master Port module (PMP) is an 8-bit parallel I/O module, specifically designed to communicate with a wide variety of parallel devices, such as communication peripherals, LCDs, external memory devices and microcontrollers. Because the interface to parallel peripherals varies significantly, the PMP is highly configurable. The PMP module can be configured to serve as either a PMP or as a Parallel Slave Port (PSP).

Key features of the PMP module are:

- Up to 16 bits of Addressing when Using Data/Address Multiplexing
- Up to 8 Programmable Address Lines
- One Chip Select Line
- Programmable Strobe Options:
  - Individual Read and Write Strobes or;
  - Read/Write Strobe with Enable Strobe
- Address Auto-Increment/Auto-Decrement
- Programmable Address/Data Multiplexing
- Programmable Polarity on Control Signals
- Legacy Parallel Slave Port Support
- Enhanced Parallel Slave Support:
  - Address Support
  - 4-Byte Deep, Auto-Incrementing Buffer
- Programmable Wait States
- Selectable Input Voltage Levels

FIGURE 10-1: PMP MODULE OVERVIEW



# PIC18F46J11 FAMILY

## 10.1 Module Registers

The PMP module has a total of 14 Special Function Registers (SFRs) for its operation, plus one additional register to set configuration options. Of these, eight registers are used for control and six are used for PMP data transfer.

### 10.1.1 CONTROL REGISTERS

The eight PMP Control registers are:

- PMCONH and PMCONL
- PMMODEH and PMMODEL
- PMSTATL and PMSTATH
- PMEHL and PМЕL

The PMCON registers (Register 10-1 and Register 10-2) control basic module operations, including turning the module on or off. They also configure address multiplexing and control strobe configuration.

The PMMODE registers (Register 10-3 and Register 10-4) configure the various Master and Slave modes, the data width and interrupt generation.

The PMEHL and PМЕL registers (Register 10-5 and Register 10-6) configure the module's operation at the hardware (I/O pin) level.

The PMSTAT registers (Register 10-5 and Register 10-6) provide status flags for the module's input and output buffers, depending on the operating mode.

### REGISTER 10-1: PMCONH: PARALLEL PORT CONTROL REGISTER HIGH BYTE (BANKED F5Fh)<sup>(1)</sup>

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN
bit 7							bit 0

#### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

- bit 7      **PMPEN:** Parallel Master Port Enable bit  
1 = PMP enabled  
0 = PMP disabled, no off-chip access performed
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **PSIDL:** Stop in Idle Mode bit  
1 = Discontinue module operation when device enters Idle mode  
0 = Continue module operation in Idle mode
- bit 4-3    **ADRMUX<1:0>:** Address/Data Multiplexing Selection bits  
11 = Reserved  
10 = All 16 bits of address are multiplexed on PMD<7:0> pins  
01 = Lower 8 bits of address are multiplexed on PMD<7:0> pins (only eight bits of address are available in this mode)  
00 = Address and data appear on separate pins (only eight bits of address are available in this mode)
- bit 2      **PTBEEN:** Byte Enable Port Enable bit (16-Bit Master mode)  
1 = PMBE port enabled  
0 = PMBE port disabled
- bit 1      **PTWREN:** Write Enable Strobe Port Enable bit  
1 = PMWR/PMENB port enabled  
0 = PMWR/PMENB port disabled
- bit 0      **PTRDEN:** Read/Write Strobe Port Enable bit  
1 = PMRD/PMWR port enabled  
0 = PMRD/PMWR port disabled

**Note 1:** This register is only available in 44-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 10-2: PMCONL: PARALLEL PORT CONTROL REGISTER LOW BYTE (BANKED F5Eh)<sup>(1)</sup>

R/W-0	R/W-0	R/W-0 <sup>(2)</sup>	U-0	R/W-0 <sup>(2)</sup>	R/W-0	R/W-0	R/W-0
CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP
bit 7							bit 0

### Legend:

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

bit 7-6 **CSF<1:0>**: Chip Select Function bits

11 = Reserved

10 = Chip select function is enabled and PMCS1 acts as chip select (in Master mode). Up to 13 address bits only can be generated.

01 = Reserved

00 = Chip select function is disabled (in Master mode). All 16 address bits can be generated.

bit 5 **ALP**: Address Latch Polarity bit<sup>(2)</sup>

1 = Active-high (PMALL and PMALH)

0 = Active-low (PMALL and PMALH)

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

bit 3 **CS1P**: Chip Select Polarity bit<sup>(2)</sup>

1 = Active-high (PMCS)

0 = Active-low (PMCS)

bit 2 **BEP**: Byte Enable Polarity bit

1 = Byte enable active-high (PMBE)

0 = Byte enable active-low (PMBE)

bit 1 **WRSP**: Write Strobe Polarity bit

For Slave modes and Master Mode 2 (PMMODEH<1:0> = 00, 01, 10):

1 = Write strobe active-high (PMWR)

0 = Write strobe active-low (PMWR)

For Master Mode 1 (PMMODEH<1:0> = 11):

1 = Enable strobe active-high (PMENB)

0 = Enable strobe active-low (PMENB)

bit 0 **RDSP**: Read Strobe Polarity bit

For Slave modes and Master Mode 2 (PMMODEH<1:0> = 00, 01, 10):

1 = Read strobe active-high (PMRD)

0 = Read strobe active-low (PMRD)

For Master Mode 1 (PMMODEH<1:0> = 11):

1 = Read/write strobe active-high (PMRD/PMWR)

0 = Read/write strobe active-low (PMRD/PMWR)

**Note 1:** This register is only available in 44-pin devices.

**Note 2:** These bits have no effect when their corresponding pins are used as address lines.

# PIC18F46J11 FAMILY

## REGISTER 10-3: PMMODEH: PARALLEL PORT MODE REGISTER HIGH BYTE (BANKED F5Dh)<sup>(1)</sup>

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0
bit 7							bit 0

### Legend:

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

- bit 7      **BUSY:** Busy bit (Master mode only)  
 1 = Port is busy  
 0 = Port is not busy
- bit 6-5    **IRQM<1:0>:** Interrupt Request Mode bits  
 11 = Interrupt generated when Read Buffer 3 is read or Write Buffer 3 is written (Buffered PSP mode) or on a read or write operation when PMA<1:0> = 11 (Addressable PSP mode only)  
 10 = No interrupt generated, processor stall activated  
 01 = Interrupt generated at the end of the read/write cycle  
 00 = No interrupt generated
- bit 4-3    **INCM<1:0>:** Increment Mode bits  
 11 = PSP read and write buffers auto-increment (Legacy PSP mode only)  
 10 = Decrement ADDR<15,13:0> by 1 every read/write cycle  
 01 = Increment ADDR<15,13:0> by 1 every read/write cycle  
 00 = No increment or decrement of address
- bit 2      **MODE16:** 8/16-Bit Mode bit  
 1 = 16-bit mode: Data register is 16 bits, a read or write to the Data register invokes two 8-bit transfers  
 0 = 8-bit mode: Data register is 8 bits, a read or write to the Data register invokes one 8-bit transfer
- bit 1-0    **MODE<1:0>:** Parallel Port Mode Select bits  
 11 = Master Mode 1 (PMCS, PMRD/PMWR, PMENB, PMBE, PMA<x:0> and PMD<7:0>)  
 10 = Master Mode 2 (PMCS, PMRD, PMWR, PMBE, PMA<x:0> and PMD<7:0>)  
 01 = Enhanced PSP, control signals (PMRD, PMWR, PMCS, PMD<7:0> and PMA<1:0>)  
 00 = Legacy Parallel Slave Port, control signals (PMRD, PMWR, PMCS and PMD<7:0>)

**Note 1:** This register is only available in 44-pin devices.



# PIC18F46J11 FAMILY

## REGISTER 10-4: PMMODEL: PARALLEL PORT MODE REGISTER LOW BYTE (BANKED F5Ch)<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAITB1 <sup>(2)</sup>	WAITB0 <sup>(2)</sup>	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1 <sup>(2)</sup>	WAITE0 <sup>(2)</sup>
bit 7						bit 0	

### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared

x = Bit is unknown

bit 7-6 **WAITB<1:0>**: Data Setup to Read/Write Wait State Configuration bits<sup>(2)</sup>

11 = Data wait of 4 Tcy; multiplexed address phase of 4 Tcy  
10 = Data wait of 3 Tcy; multiplexed address phase of 3 Tcy  
01 = Data wait of 2 Tcy; multiplexed address phase of 2 Tcy  
00 = Data wait of 1 Tcy; multiplexed address phase of 1 Tcy

bit 5-2 **WAITM<3:0>**: Read to Byte Enable Strobe Wait State Configuration bits

1111 = Wait of additional 15 Tcy

.  
.  
.

0001 = Wait of additional 1 Tcy

0000 = No additional Wait cycles (operation forced into one Tcy)

bit 1-0 **WAITE<1:0>**: Data Hold After Strobe Wait State Configuration bits<sup>(2)</sup>

11 = Wait of 4 Tcy  
10 = Wait of 3 Tcy  
01 = Wait of 2 Tcy  
00 = Wait of 1 Tcy

**Note 1:** This register is only available in 44-pin devices.

**2:** WAITBx and WAITEx bits are ignored whenever WAITM<3:0> = 0000.

# PIC18F46J11 FAMILY

## REGISTER 10-5: PМЕH: PARALLEL PORT ENABLE REGISTER HIGH BYTE (BANKED F57h)<sup>(1)</sup>

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	PTEN14	—	—	—	—	—	—
bit 7							bit 0

### Legend:

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

- bit 7                      **Unimplemented:** Read as '0'
- bit 6                      **PTEN14:** PMCS1 Port Enable bit  
                                  1 = PMCS1 chip select line  
                                  0 = PMCS1 functions as port I/O
- bit 5-0                      **Unimplemented:** Read as '0'

**Note 1:** This register is only available in 44-pin devices.

## REGISTER 10-6: PМЕL: PARALLEL PORT ENABLE REGISTER LOW BYTE (BANKED F56h)<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0
bit 7							bit 0

### Legend:

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

- bit 7-2                      **PTEN<7:2>:** PMP Address Port Enable bits  
                                  1 = PMA<7:2> function as PMP address lines  
                                  0 = PMA<7:2> function as port I/O
- bit 1-0                      **PTEN<1:0>:** PMALH/PMALL Strobe Enable bits  
                                  1 = PMA<1:0> function as either PMA<1:0> or PMALH and PMALL  
                                  0 = PMA<1:0> pads functions as port I/O

**Note 1:** This register is only available in 44-pin devices.

# PIC18F46J11 FAMILY

## REGISTER 10-7: PMSTATH: PARALLEL PORT STATUS REGISTER HIGH BYTE (BANKED F55h)<sup>(1)</sup>

R-0	R/W-0	U-0	U-0	R-0	R-0	R-0	R-0
IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F
bit 7							bit 0

### Legend:

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

- bit 7            **IBF:** Input Buffer Full Status bit  
                   1 = All writable input buffer registers are full  
                   0 = Some or all of the writable input buffer registers are empty
- bit 6            **IBOV:** Input Buffer Overflow Status bit  
                   1 = A write attempt to a full input byte register occurred (must be cleared in software)  
                   0 = No overflow occurred
- bit 5-4        **Unimplemented:** Read as '0'
- bit 3-0        **IB3F:IB0F:** Input Buffer x Status Full bits  
                   1 = Input buffer contains data that has not been read (reading buffer will clear this bit)  
                   0 = Input buffer does not contain any unread data

**Note 1:** This register is only available in 44-pin devices.

## REGISTER 10-8: PMSTATL: PARALLEL PORT STATUS REGISTER LOW BYTE (BANKED F54h)<sup>(1)</sup>

R-1	R/W-0	U-0	U-0	R-1	R-1	R-1	R-1
OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E
bit 7							bit 0

### Legend:

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

- bit 7            **OBE:** Output Buffer Empty Status bit  
                   1 = All readable output buffer registers are empty  
                   0 = Some or all of the readable output buffer registers are full
- bit 6            **OBUF:** Output Buffer Underflow Status bit  
                   1 = A read occurred from an empty output byte register (must be cleared in software)  
                   0 = No underflow occurred
- bit 5-4        **Unimplemented:** Read as '0'
- bit 3-0        **OB3E:OB0E:** Output Buffer x Status Empty bits  
                   1 = Output buffer is empty (writing data to the buffer will clear this bit)  
                   0 = Output buffer contains data that has not been transmitted

**Note 1:** This register is only available in 44-pin devices.

# PIC18F46J11 FAMILY

---

## 10.1.2 DATA REGISTERS

The PMP module uses eight registers for transferring data into and out of the microcontroller. They are arranged as four pairs to allow the option of 16-bit data operations:

- PMDIN1H and PMDIN1L
- PMDIN2H and PMDIN2L
- PMADDRH/PMDOOUT1H and PMADDRL/PMDOOUT1L
- PMDOOUT2H and PMDOOUT2L

The PMDIN1 register is used for incoming data in Slave modes and both input and output data in Master modes. The PMDIN2 register is used for buffering input data in select Slave modes.

The PMADDR/PMDOOUT1 registers are actually a single register pair; the name and function are dictated by the module's operating mode. In Master modes, the registers function as the PMADDRH and PMADDRL registers and contain the address of any incoming or outgoing data. In Slave modes, the registers function as PMDOOUT1H and PMDOOUT1L and are used for outgoing data.

PMADDRH differs from PMADDRL in that it can also have limited PMP control functions. When the module is operating in select Master mode configurations, the upper two bits of the register can be used to determine the operation of chip select signals. If these are not used, PMADDR simply functions to hold the upper 8 bits of the address. Register 10-9 provides the function of the individual bits in PMADDRH.

The PMDOOUT2H and PMDOOUT2L registers are only used in Buffered Slave modes and serve as a buffer for outgoing data.

## 10.1.3 PAD CONFIGURATION CONTROL REGISTER

In addition to the module level configuration options, the PMP module can also be configured at the I/O pin for electrical operation. This option allows users to select either the normal Schmitt Trigger input buffer on digital I/O pins shared with the PMP, or use TTL level compatible buffers instead. Buffer configuration is controlled by the PMPTTL bit in the PADCFG1 register.

# PIC18F46J11 FAMILY

## REGISTER 10-9: PMADDRH: PARALLEL PORT ADDRESS REGISTER HIGH BYTE – MASTER MODES ONLY (ACCESS F6Fh)<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CS1	Parallel Master Port Address High Byte<13:8>					
bit 7							bit 0

### Legend:

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

bit 7                      **Unimplemented:** Read as '0'  
 bit 6                      **CS1:** Chip Select bit  
                             If PMCON<7:6> = 10:  
                             1 = Chip select is active  
                             0 = Chip select is inactive  
                             If PMCON<7:6> = 11 or 00:  
                             Bit functions as ADDR<14>.  
 bit 5-0                      **Parallel Master Port Address:** High Byte<13:8> bits

**Note 1:** In Enhanced Slave mode, PMADDRH functions as PMDOUT1H, one of the Output Data Buffer registers.

## REGISTER 10-10: PMADDRL: PARALLEL PORT ADDRESS REGISTER LOW BYTE – MASTER MODES ONLY (ACCESS F6Eh)<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
Parallel Master Port Address Low Byte<7:0>							
bit 7							bit 0

### Legend:

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

bit 7-0                      **Parallel Master Port Address:** Low Byte<7:0> bits

**Note 1:** In Enhanced Slave mode, PMADDRL functions as PMDOUT1L, one of the Output Data Buffer registers.

# PIC18F46J11 FAMILY

## 10.2 Slave Port Modes

The primary mode of operation for the module is configured using the MODE<1:0> bits in the PMMODEH register. The setting affects whether the module acts as a slave or a master, and it determines the usage of the control pins.

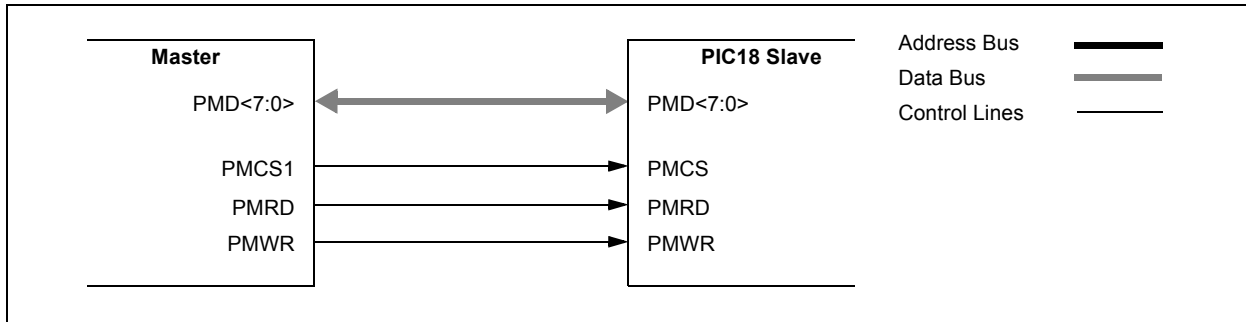
### 10.2.1 LEGACY MODE (PSP)

In Legacy mode (PMMODEH<1:0> = 00 and PMPEN = 1), the module is configured as a Parallel Slave Port (PSP) with the associated enabled module

pins dedicated to the module. In this mode, an external device, such as another microcontroller or micro-processor, can asynchronously read and write data using the 8-bit data bus (PMD<7:0>), the read (PMRD), write (PMWR) and chip select (PMCS1) inputs. It acts as a slave on the bus and responds to the read/write control signals.

Figure 10-2 displays the connection of the PSP. When chip select is active and a write strobe occurs (PMCS = 1 and PMWR = 1), the data from PMD<7:0> is captured into the PMDIN1L register.

**FIGURE 10-2: LEGACY PARALLEL SLAVE PORT EXAMPLE**



# PIC18F46J11 FAMILY

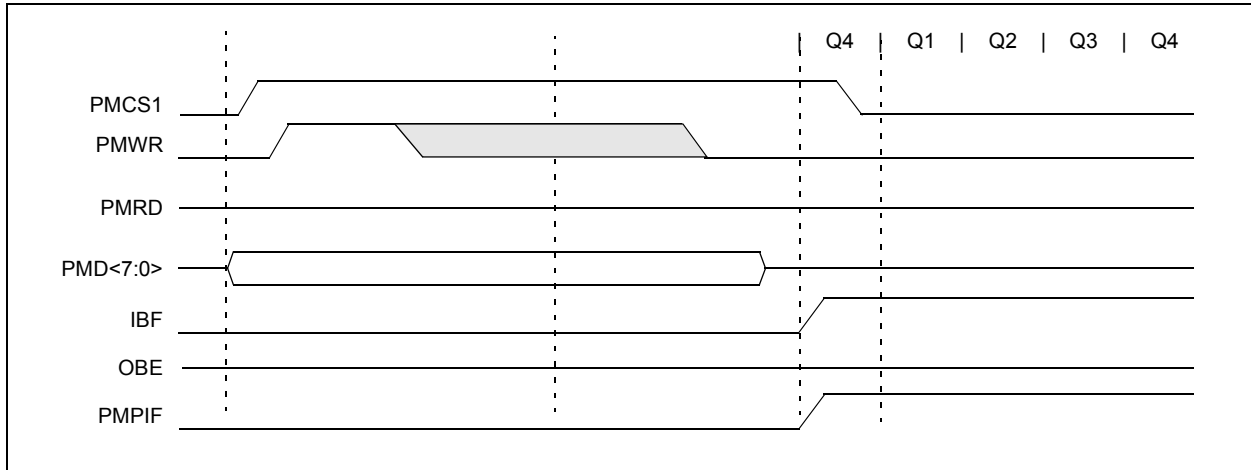
## 10.2.2 WRITE TO SLAVE PORT

When chip select is active and a write strobe occurs ( $PMCS = 1$  and  $PMWR = 1$ ), the data from  $PMD<7:0>$  is captured into the lower  $PMDIN1L$  register. The  $PMPIF$  and  $IBF$  flag bits are set when the write ends. The timing for the control signals in Write mode is displayed in Figure 10-3. The polarity of the control signals are configurable.

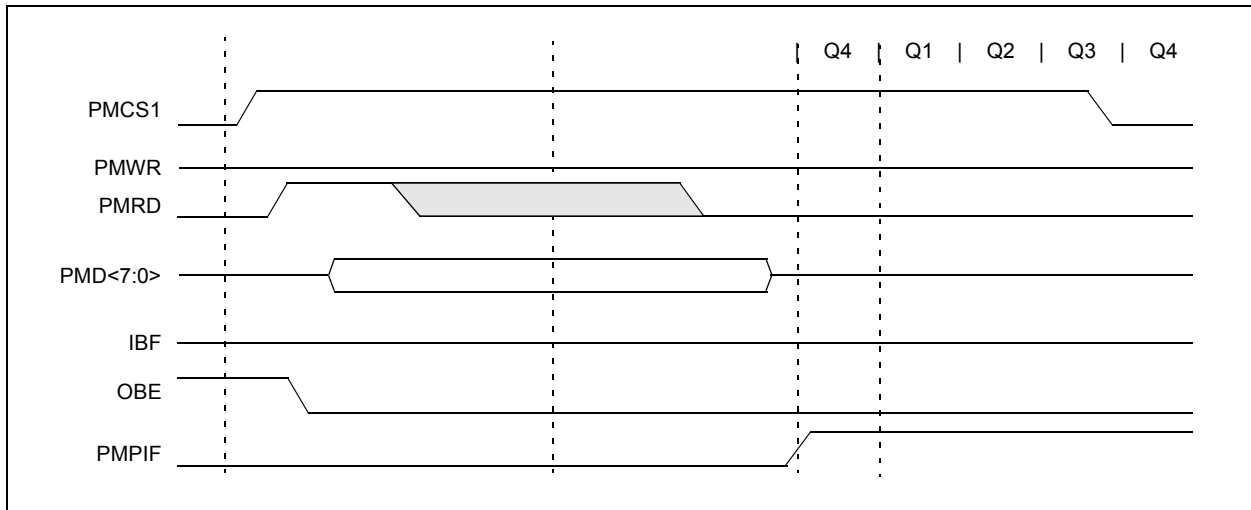
## 10.2.3 READ FROM SLAVE PORT

When chip select is active and a read strobe occurs ( $PMCS = 1$  and  $PMRD = 1$ ), the data from the  $PMDOUT1L$  register ( $PMDOUT1L<7:0>$ ) is presented onto  $PMD<7:0>$ . Figure 10-4 provides the timing for the control signals in Read mode.

**FIGURE 10-3: PARALLEL SLAVE PORT WRITE WAVEFORMS**



**FIGURE 10-4: PARALLEL SLAVE PORT READ WAVEFORMS**



# PIC18F46J11 FAMILY

## 10.2.4 BUFFERED PARALLEL SLAVE PORT MODE

Buffered Parallel Slave Port mode is functionally identical to the legacy PSP mode with one exception, the implementation of 4-level read and write buffers. Buffered PSP mode is enabled by setting the INCM bits in the PMMODEH register. If the INCM<1:0> bits are set to '11', the PMP module will act as the Buffered PSP.

When the Buffered mode is active, the PMDIN1L, PMDIN1H, PMDIN2L and PMDIN2H registers become the write buffers and the PMDOUT1L, PMDOUT1H, PMDOUT2L and PMDOUT2H registers become the read buffers. Buffers are numbered 0 through 3, starting with the lower byte of PMDIN1L to PMDIN2H as the read buffers and PMDOUT1L to PMDOUT2H as the write buffers.

### 10.2.4.1 READ FROM SLAVE PORT

For read operations, the bytes will be sent out sequentially, starting with Buffer 0 (PMDOUT1L<7:0>) and ending with Buffer 3 (PMDOUT2H<7:0>) for every read strobe. The module maintains an internal pointer to keep track of which buffer is to be read. Each buffer has a corresponding read status bit, OBxE, in the PMSTATL register. This bit is cleared when a buffer contains data that has not been written to the bus, and is set when data is written to the bus. If the current buffer location being read from is empty, a buffer underflow is generated, and the Buffer Overflow flag bit, OBUF, is set. If all four OBxE status bits are set, then the Output Buffer Empty flag (OBE) will also be set.

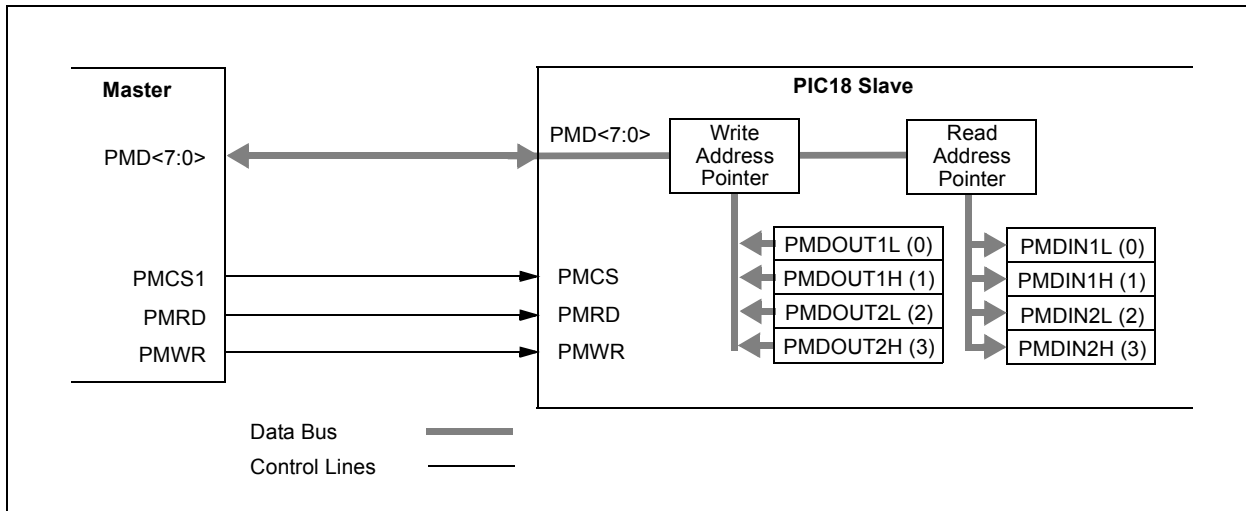
### 10.2.4.2 WRITE TO SLAVE PORT

For write operations, the data has to be stored sequentially, starting with Buffer 0 (PMDIN1L<7:0>) and ending with Buffer 3 (PMDIN2H<7:0>). As with read operations, the module maintains an internal pointer to the buffer that is to be written next.

The input buffers have their own write status bits, IBxF in the PMSTATH register. The bit is set when the buffer contains unread incoming data, and cleared when the data has been read. The flag bit is set on the write strobe. If a write occurs on a buffer when its associated IBxF bit is set, the Buffer Overflow flag, IBOV, is set; any incoming data in the buffer will be lost. If all four IBxF flags are set, the Input Buffer Full Flag (IBF) is set.

In Buffered Slave mode, the module can be configured to generate an interrupt on every read or write strobe (IRQM<1:0> = 01). It can be configured to generate an interrupt on a read from Read Buffer 3 or a write to Write Buffer 3, which is essentially an interrupt every fourth read or write strobe (RQM<1:0> = 11). When interrupting every fourth byte for input data, all input buffer registers should be read to clear the IBxF flags. If these flags are not cleared, then there is a risk of hitting an overflow condition.

**FIGURE 10-5: PARALLEL MASTER/SLAVE CONNECTION BUFFERED EXAMPLE**





# PIC18F46J11 FAMILY

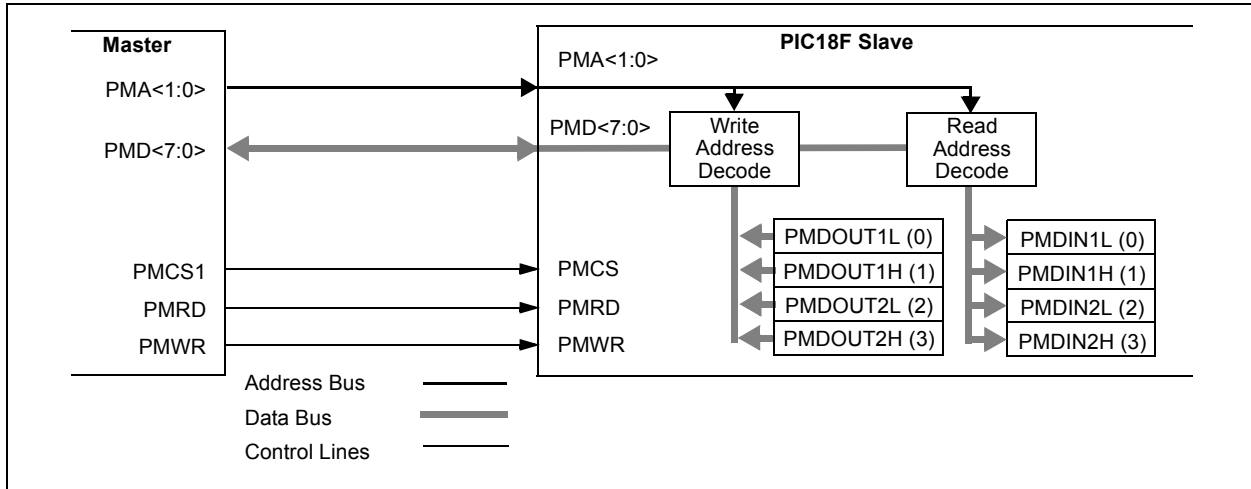
## 10.2.5 ADDRESSABLE PARALLEL SLAVE PORT MODE

In the Addressable Parallel Slave Port mode (PMMODEH<1:0> = 01), the module is configured with two extra inputs, PMA<1:0>, which are the address lines 1 and 0. This makes the 4-byte buffer space directly addressable as fixed pairs of read and write buffers. As with Legacy Buffered mode, data is output from PMDOUT1L, PMDOUT1H, PMDOUT2L and PMDOUT2H, and is read in on PMDIN1L, PMDIN1H, PMDIN2L and PMDIN2H. Table 10-1 provides the buffer addressing for the incoming address to the input and output registers.

**TABLE 10-1: SLAVE MODE BUFFER ADDRESSING**

PMA<1:0>	Output Register (Buffer)	Input Register (Buffer)
00	PMDOUT1L (0)	PMDIN1L (0)
01	PMDOUT1H (1)	PMDIN1H (1)
10	PMDOUT2L (2)	PMDIN2L (2)
11	PMDOUT2H(3)	PMDIN2H (3)

**FIGURE 10-6: PARALLEL MASTER/SLAVE CONNECTION ADDRESSED BUFFER EXAMPLE**

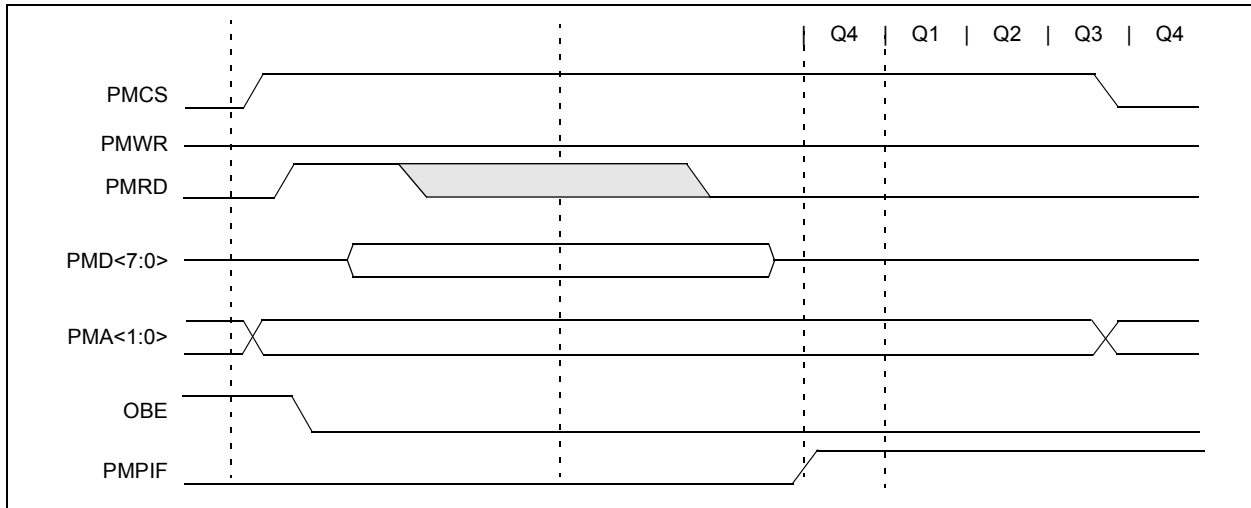


### 10.2.5.1 READ FROM SLAVE PORT

When chip select is active and a read strobe occurs (PMCS = 1 and PMRD = 1), the data from one of the four output bytes is presented onto PMD<7:0>. Which byte is read depends on the 2-bit address placed on ADDR<1:0>. Table 10-1 provides the corresponding

output registers and their associated address. When an output buffer is read, the corresponding OBxE bit is set. The OBxE flag bit is set when all the buffers are empty. If any buffer is already empty, OBxE = 1, the next read to that buffer will generate an OBUF event.

**FIGURE 10-7: PARALLEL SLAVE PORT READ WAVEFORMS**



# PIC18F46J11 FAMILY

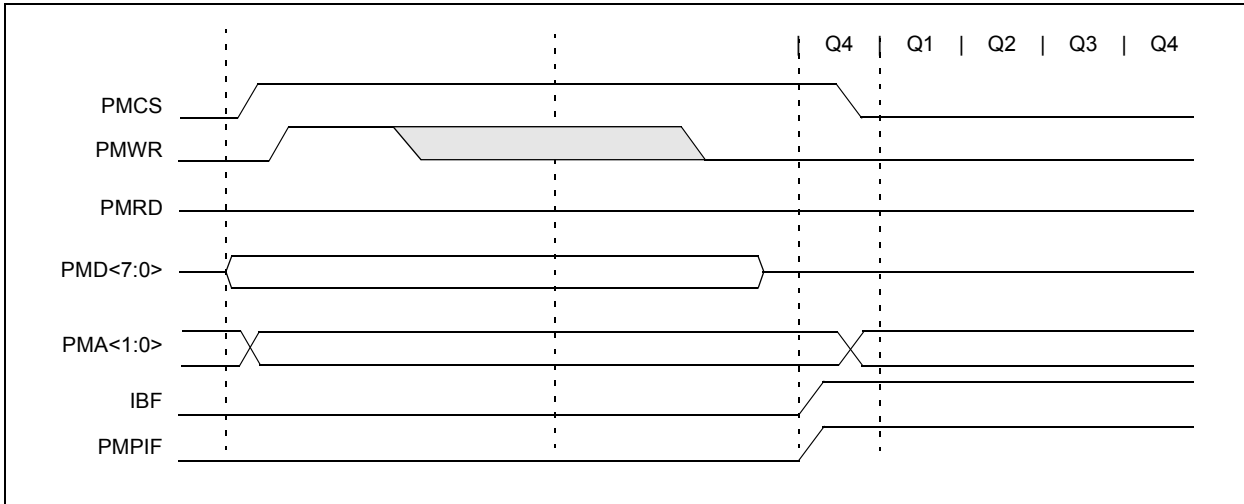
## 10.2.5.2 WRITE TO SLAVE PORT

When chip select is active and a write strobe occurs (PMCS = 1 and PMWR = 1), the data from PMD<7:0> is captured into one of the four input buffer bytes. Which byte is written depends on the 2-bit address placed on ADDRL<1:0>.

Table 10-1 provides the corresponding input registers and their associated address.

When an input buffer is written, the corresponding IBxF bit is set. The IBF flag bit is set when all the buffers are written. If any buffer is already written (IBxF = 1), the next write strobe to that buffer will generate an OBUF event and the byte will be discarded.

**FIGURE 10-8: PARALLEL SLAVE PORT WRITE WAVEFORMS**



## 10.3 MASTER PORT MODES

In its Master modes, the PMP module provides an 8-bit data bus, up to 16 bits of address, and all the necessary control signals to operate a variety of external parallel devices, such as memory devices, peripherals and slave microcontrollers. To use the PMP as a master, the module must be enabled ( $PMPEN = 1$ ) and the mode must be set to one of the two possible Master modes ( $PMMODEH<1:0> = 10$  or  $11$ ).

Because there are a number of parallel devices with a variety of control methods, the PMP module is designed to be extremely flexible to accommodate a range of configurations. Some of these features include:

- 8-Bit and 16-Bit Data modes on an 8-bit data bus
- Configurable address/data multiplexing
- Up to two chip select lines
- Up to 16 selectable address lines
- Address auto-increment and auto-decrement
- Selectable polarity on all control lines
- Configurable Wait states at different stages of the read/write cycle

### 10.3.1 PMP AND I/O PIN CONTROL

Multiple control bits are used to configure the presence or absence of control and address signals in the module. These bits are  $PTBEEN$ ,  $PTWREN$ ,  $PTRDEN$  and  $PTEN<15:0>$ . They give the user the ability to conserve pins for other functions and allow flexibility to control the external address. When any one of these bits is set, the associated function is present on its associated pin; when clear, the associated pin reverts to its defined I/O port function.

Setting a  $PTENx$  bit will enable the associated pin as an address pin and drive the corresponding data contained in the  $PMADDR$  register. Clearing a  $PTENx$  bit will force the pin to revert to its original I/O function.

For the pins configured as chip select ( $PMCS1$  or  $PMCS2$ ) with the corresponding  $PTENx$  bit set, the  $PTEN0$  and  $PTEN1$  bits will also control the  $PMALL$  and  $PMALH$  signals. When multiplexing is used, the associated address latch signals should be enabled.

### 10.3.2 READ/WRITE CONTROL

The PMP module supports two distinct read/write signaling methods. In Master Mode 1, read and write strobes are combined into a single control line,  $PMRD/PMWR$ . A second control line,  $PMENB$ , determines when a read or write action is to be taken. In Master Mode 2, separate read and write strobes ( $PMRD$  and  $PMWR$ ) are supplied on separate pins.

All control signals ( $PMRD$ ,  $PMWR$ ,  $PMBE$ ,  $PMENB$ ,  $PMAL$  and  $PMCSx$ ) can be individually configured as either positive or negative polarity. Configuration is controlled by separate bits in the  $PMCONL$  register.

Note that the polarity of control signals that share the same output pin (for example,  $PMWR$  and  $PMENB$ ) are controlled by the same bit; the configuration depends on which Master Port mode is being used.

### 10.3.3 DATA WIDTH

The PMP supports data widths of both 8 bits and 16 bits. The data width is selected by the  $MODE16$  bit ( $PMMODEH<2>$ ). Because the data path into and out of the module is only 8 bits wide, 16-bit operations are always handled in a multiplexed fashion, with the Least Significant Byte (LSB) of data being presented first. To differentiate data bytes, the byte enable control strobe,  $PMBE$ , is used to signal when the Most Significant Byte (MSB) of data is being presented on the data lines.

### 10.3.4 ADDRESS MULTIPLEXING

In either of the Master modes ( $PMMODEH<1:0> = 1x$ ), the user can configure the address bus to be multiplexed together with the data bus. This is accomplished by using the  $ADRMUX<1:0>$  bits ( $PMCONH<4:3>$ ). There are three address multiplexing modes available; typical pinout configurations for these modes are displayed in Figure 10-9, Figure 10-10 and Figure 10-11.

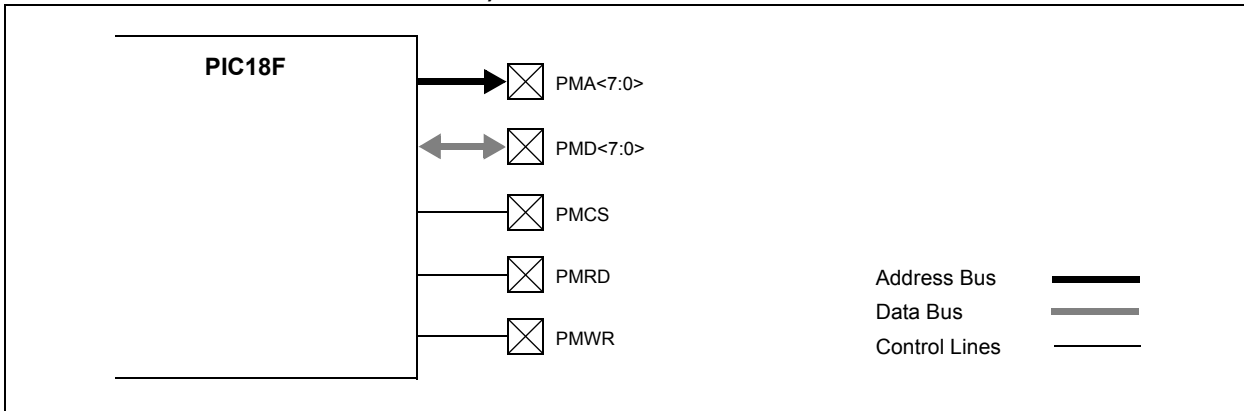
In Demultiplexed mode ( $PMCONH<4:3> = 00$ ), data and address information are completely separated. Data bits are presented on  $PMD<7:0>$  and address bits are presented on  $PMADDRH<6:0>$  and  $PMADDRL<7:0>$ .

In Partially Multiplexed mode ( $PMCONH<4:3> = 01$ ), the lower eight bits of the address are multiplexed with the data pins on  $PMD<7:0>$ . The upper eight bits of address are unaffected and are presented on  $PMADDRH<6:0>$ . The  $PMA0$  pin is used as an address latch, and presents the address latch low enable strobe ( $PMALL$ ). The read and write sequences are extended by a complete CPU cycle during which the address is presented on the  $PMD<7:0>$  pins.

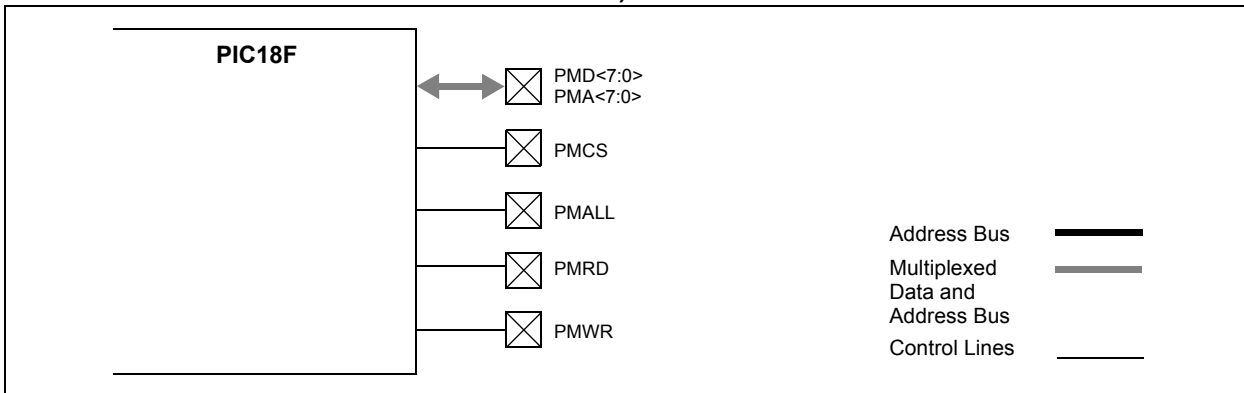
In Fully Multiplexed mode ( $PMCONH<4:3> = 10$ ), the entire 16 bits of the address are multiplexed with the data pins on  $PMD<7:0>$ . The  $PMA0$  and  $PMA1$  pins are used to present address latch low enable ( $PMALL$ ) and address latch high enable ( $PMALH$ ) strobes, respectively. The read and write sequences are extended by two complete CPU cycles. During the first cycle, the lower eight bits of the address are presented on the  $PMD<7:0>$  pins with the  $PMALL$  strobe active. During the second cycle, the upper eight bits of the address are presented on the  $PMD<7:0>$  pins with the  $PMALH$  strobe active. In the event the upper address bits are configured as chip select pins, the corresponding address bits are automatically forced to '0'.

# PIC18F46J11 FAMILY

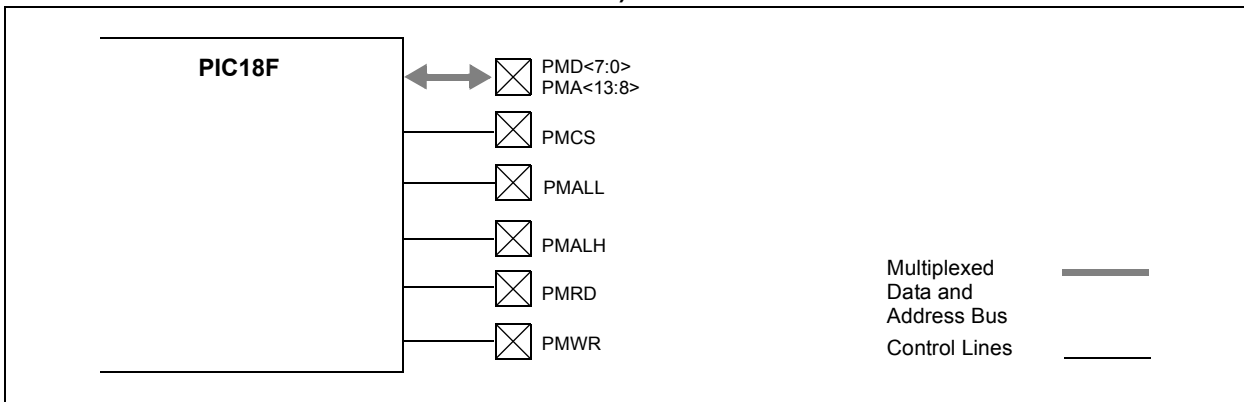
**FIGURE 10-9: DEMULTIPLEXED ADDRESSING MODE (SEPARATE READ AND WRITE STROBES WITH CHIP SELECT)**



**FIGURE 10-10: PARTIALLY MULTIPLEXED ADDRESSING MODE (SEPARATE READ AND WRITE STROBES WITH CHIP SELECT)**



**FIGURE 10-11: FULLY MULTIPLEXED ADDRESSING MODE (SEPARATE READ AND WRITE STROBES WITH CHIP SELECT)**



## 10.3.5 CHIP SELECT FEATURES

Up to two chip select lines, PMCS1 and PMCS2, are available for the Master modes of the PMP. The two chip select lines are multiplexed with the Most Significant bit (MSb) of the address bus (PMADDRH<6>). When a pin is configured as a chip select, it is not included in any address auto-increment/decrement. The function of the chip select signals is configured using the chip select function bits (PMCONL<7:6>).

## 10.3.6 AUTO-INCREMENT/DECREMENT

While the module is operating in one of the Master modes, the INCMx bits (PMMODEH<4:3>) control the behavior of the address value. The address can be made to automatically increment or decrement after each read and write operation. The address increments once each operation is completed and the BUSY bit goes to '0'. If the chip select signals are disabled and configured as address bits, the bits will participate in the increment and decrement operations; otherwise, the CS1 bit values will be unaffected.

## 10.3.7 WAIT STATES

In Master mode, the user has control over the duration of the read, write and address cycles by configuring the module Wait states. Three portions of the cycle, the beginning, middle and end, are configured using the corresponding WAITBx, WAITMx and WAITEx bits in the PMMODEL register.

The WAITBx bits (PMMODEL<7:6>) set the number of Wait cycles for the data setup prior to the PMRD/PMWT strobe in Mode 10, or prior to the PMENB strobe in Mode 11. The WAITMx bits (PMMODEL<5:2>) set the number of Wait cycles for the PMRD/PMWT strobe in Mode 10, or for the PMENB strobe in Mode 11. When this Wait state setting is '0', then WAITB and WAITE have no effect. The WAITE bits (PMMODEL<1:0>) define the number of Wait cycles for the data hold time after the PMRD/PMWT strobe in Mode 10, or after the PMENB strobe in Mode 11.

## 10.3.8 READ OPERATION

To perform a read on the PMP, the user reads the PMDIN1L register. This causes the PMP to output the desired values on the chip select lines and the address bus. Then the read line (PMRD) is strobed. The read data is placed into the PMDIN1L register.

If the 16-bit mode is enabled (MODE16 = 1), the read of the low byte of the PMDIN1L register will initiate two bus reads. The first read data byte is placed into the PMDIN1L register, and the second read data is placed into the PMDIN1H.

Note that the read data obtained from the PMDIN1L register is actually the read value from the previous read operation. Hence, the first user read will be a dummy read to initiate the first bus read and fill the read register. Also, the requested read value will not be ready until after the BUSY bit is observed low. Thus, in a back-to-back read operation, the data read from the register will be the same for both reads. The next read of the register will yield the new value.

## 10.3.9 WRITE OPERATION

To perform a write onto the parallel bus, the user writes to the PMDIN1L register. This causes the module to first output the desired values on the chip select lines and the address bus. The write data from the PMDIN1L register is placed onto the PMD<7:0> data bus. Then the write line (PMWR) is strobed. If the 16-bit mode is enabled (MODE16 = 1), the write to the PMDIN1L register will initiate two bus writes. First write will consist of the data contained in PMDIN1L and the second write will contain the PMDIN1H.

## 10.3.10 PARALLEL MASTER PORT STATUS

### 10.3.10.1 The BUSY Bit

In addition to the PMP interrupt, a BUSY bit is provided to indicate the status of the module. This bit is used only in Master mode. While any read or write operation is in progress, the BUSY bit is set for all but the very last CPU cycle of the operation. In effect, if a single-cycle read or write operation is requested, the BUSY bit will never be active. This allows back-to-back transfers. While the bit is set, any request by the user to initiate a new operation will be ignored (i.e., writing or reading the lower byte of the PMDIN1L register will neither initiate a read nor a write).

### 10.3.10.2 Interrupts

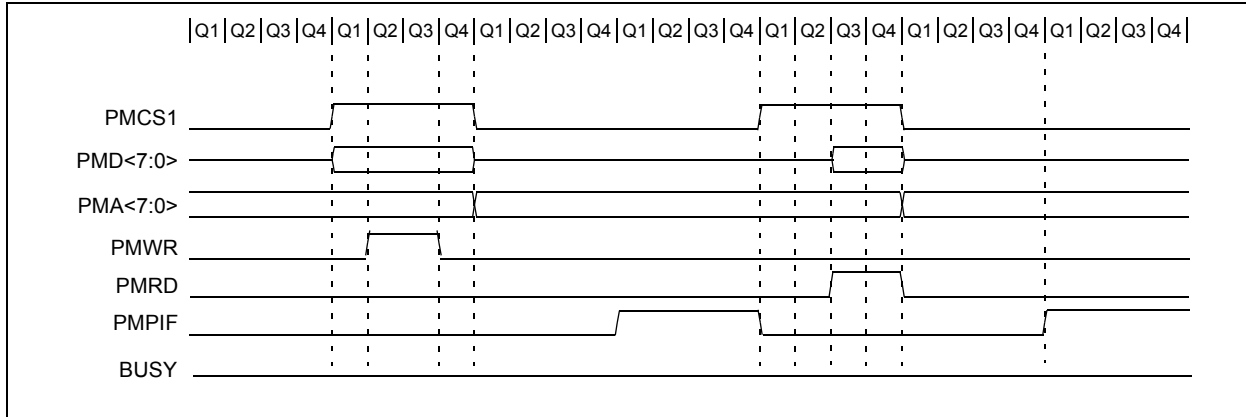
When the PMP module interrupt is enabled for Master mode, the module will interrupt on every completed read or write cycle; otherwise, the BUSY bit is available to query the status of the module.

# PIC18F46J11 FAMILY

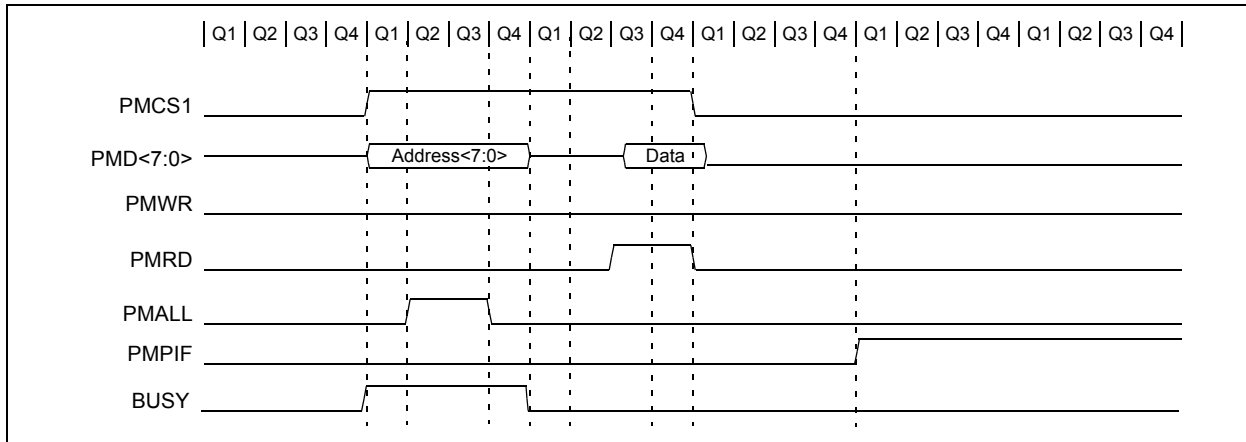
## 10.3.11 MASTER MODE TIMING

This section contains a number of timing examples that represent the common Master mode configuration options. These options vary from 8-bit to 16-bit data, fully demultiplexed to fully multiplexed address and Wait states.

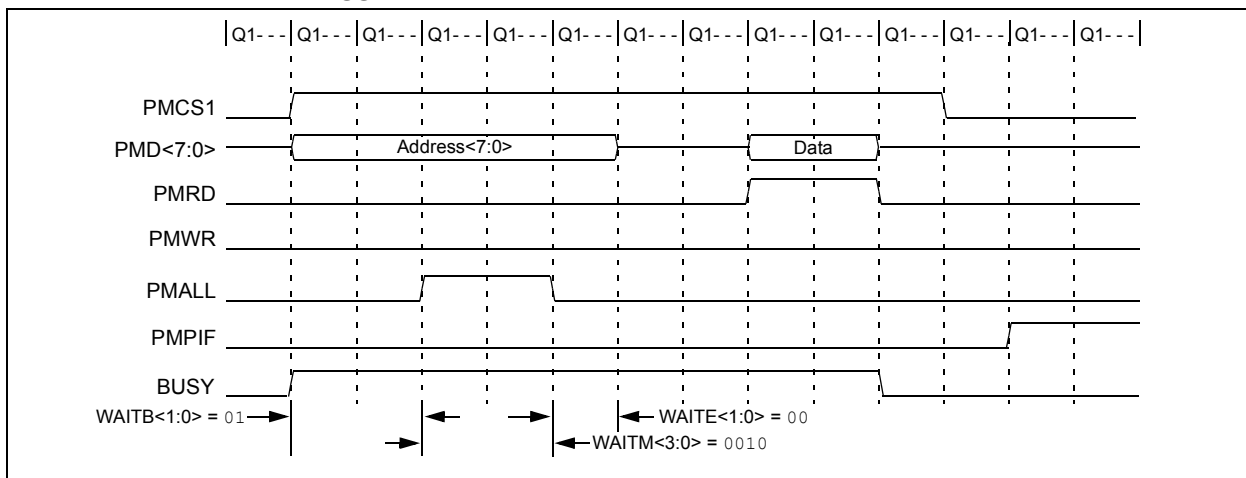
**FIGURE 10-12: READ AND WRITE TIMING, 8-BIT DATA, DEMULTIPLEXED ADDRESS**



**FIGURE 10-13: READ TIMING, 8-BIT DATA, PARTIALLY MULTIPLEXED ADDRESS**

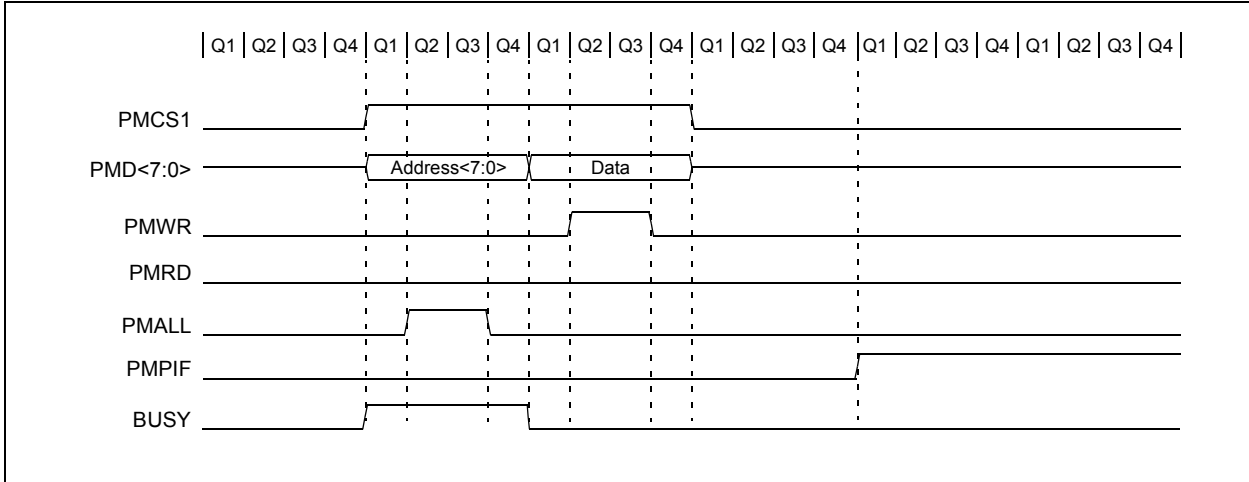


**FIGURE 10-14: READ TIMING, 8-BIT DATA, WAIT STATES ENABLED, PARTIALLY MULTIPLEXED ADDRESS**

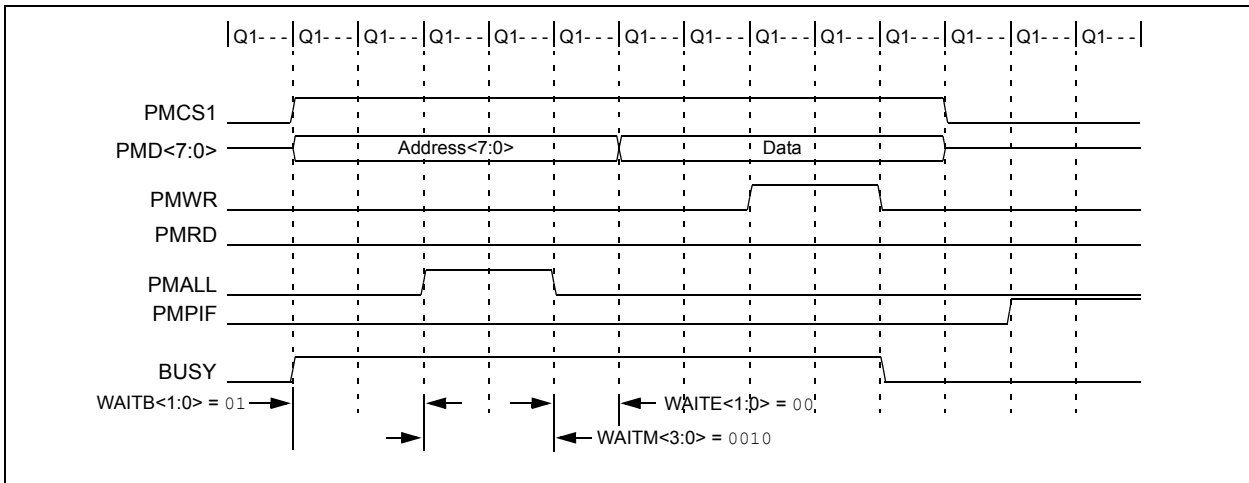


# PIC18F46J11 FAMILY

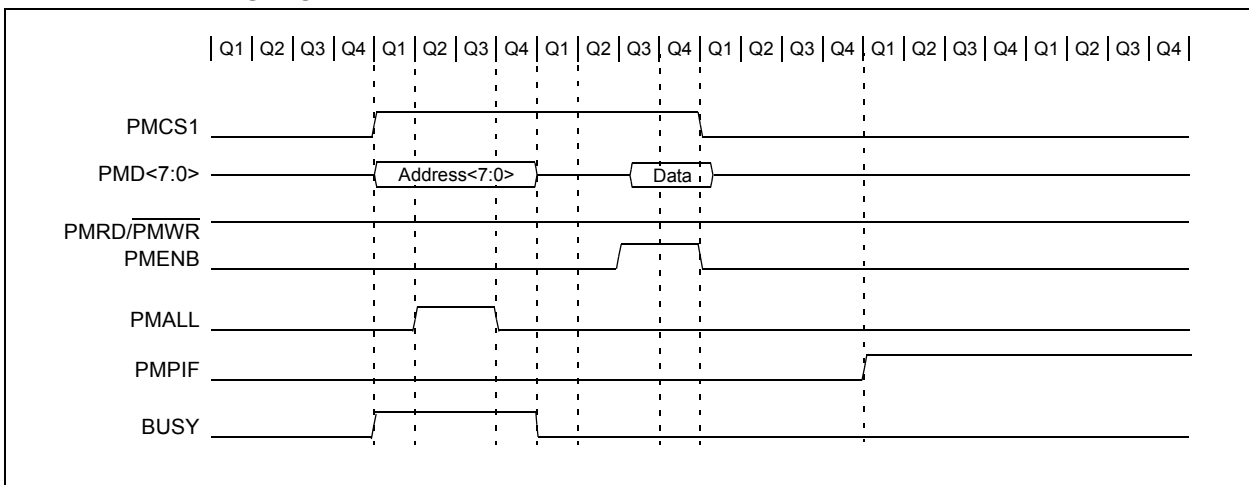
**FIGURE 10-15: WRITE TIMING, 8-BIT DATA, PARTIALLY MULTIPLEXED ADDRESS**



**FIGURE 10-16: WRITE TIMING, 8-BIT DATA, WAIT STATES ENABLED, PARTIALLY MULTIPLEXED ADDRESS**

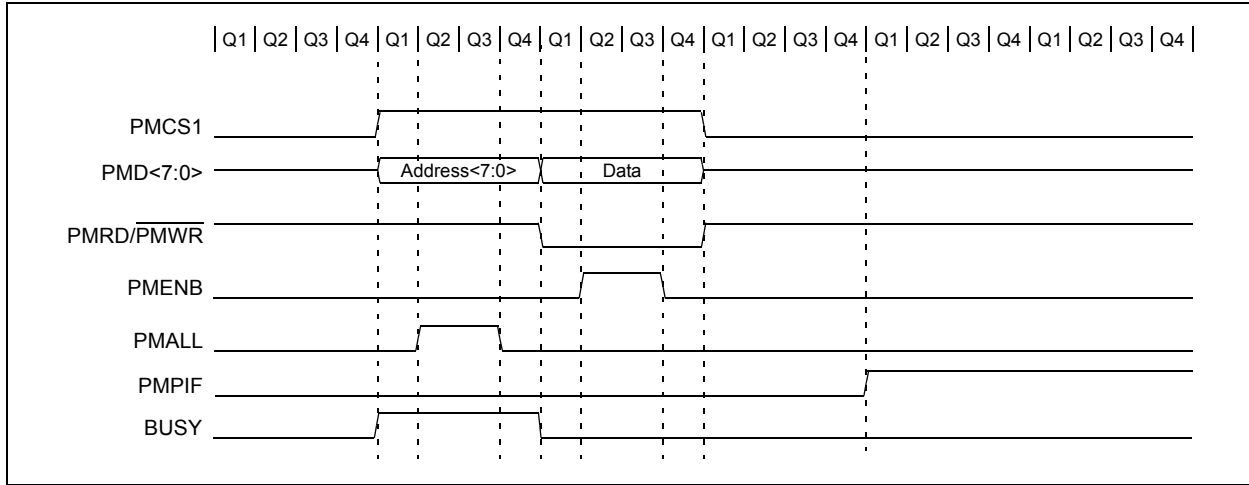


**FIGURE 10-17: READ TIMING, 8-BIT DATA, PARTIALLY MULTIPLEXED ADDRESS, ENABLE STROBE**

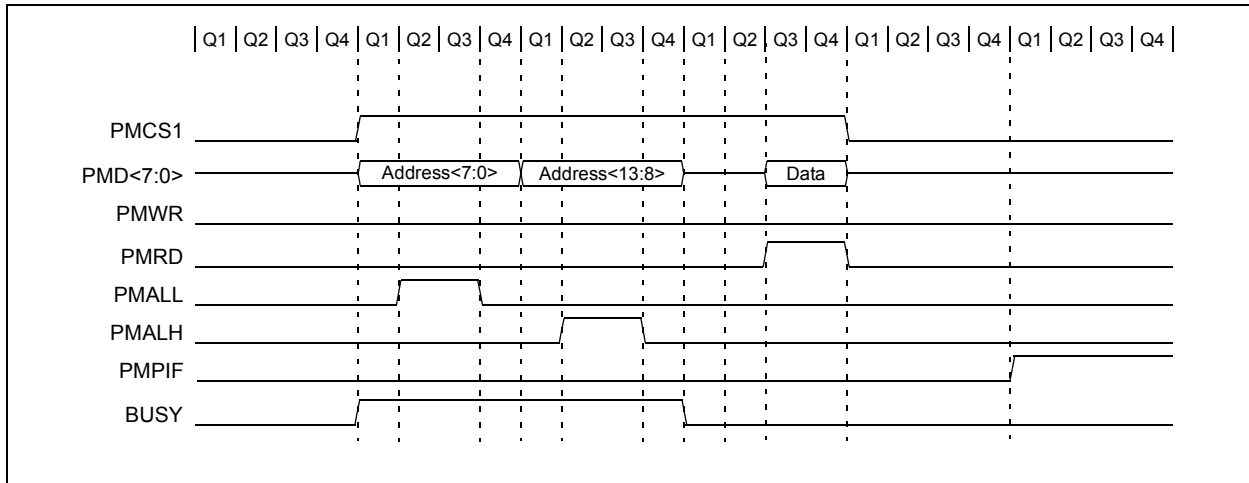


# PIC18F46J11 FAMILY

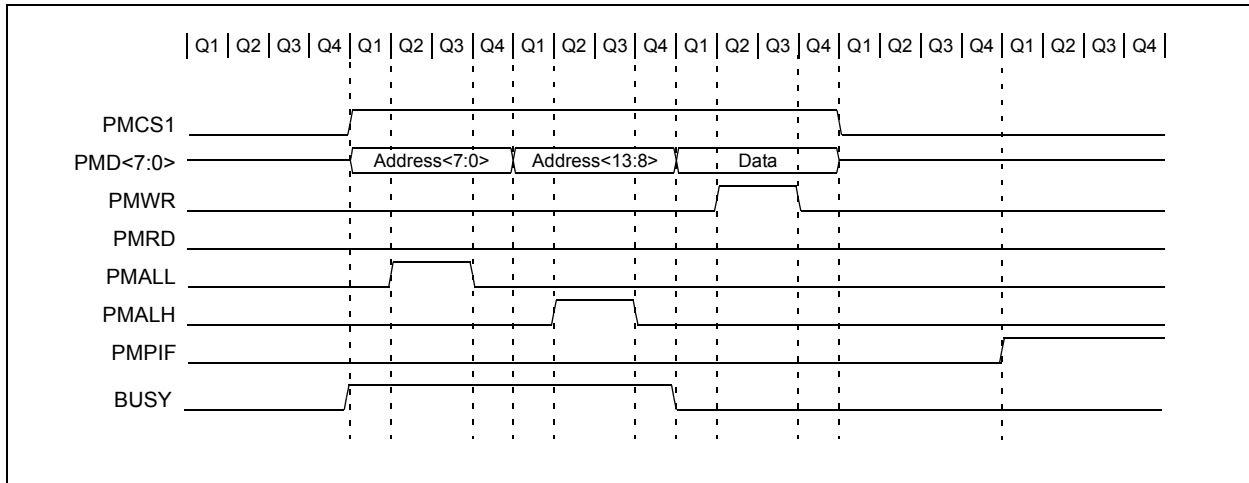
**FIGURE 10-18: WRITE TIMING, 8-BIT DATA, PARTIALLY MULTIPLEXED ADDRESS, ENABLE STROBE**



**FIGURE 10-19: READ TIMING, 8-BIT DATA, FULLY MULTIPLEXED 16-BIT ADDRESS**



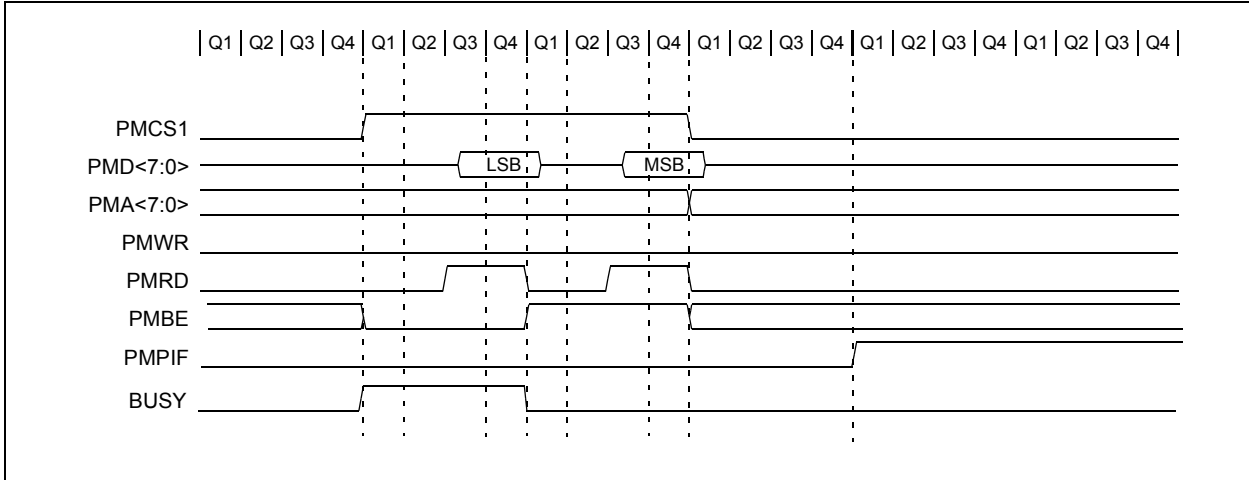
**FIGURE 10-20: WRITE TIMING, 8-BIT DATA, FULLY MULTIPLEXED 16-BIT ADDRESS**



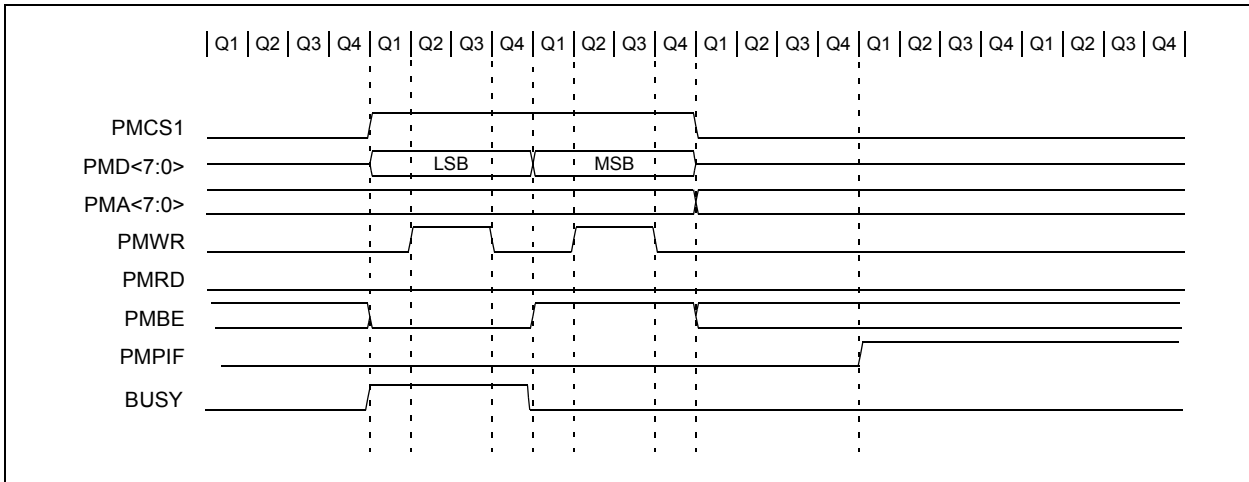


# PIC18F46J11 FAMILY

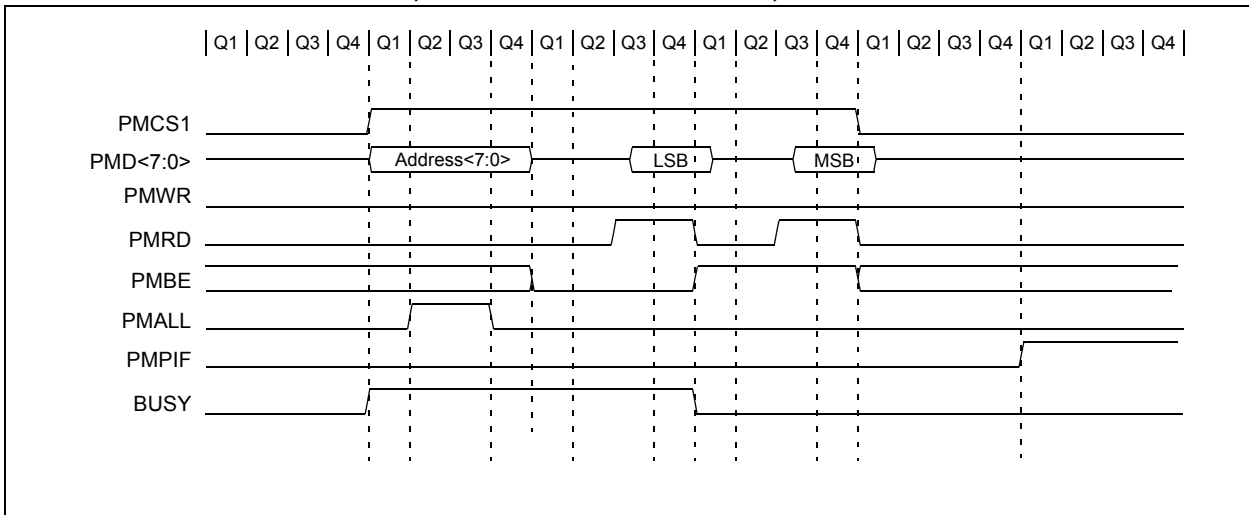
**FIGURE 10-21: READ TIMING, 16-BIT DATA, DEMULTIPLEXED ADDRESS**



**FIGURE 10-22: WRITE TIMING, 16-BIT DATA, DEMULTIPLEXED ADDRESS**

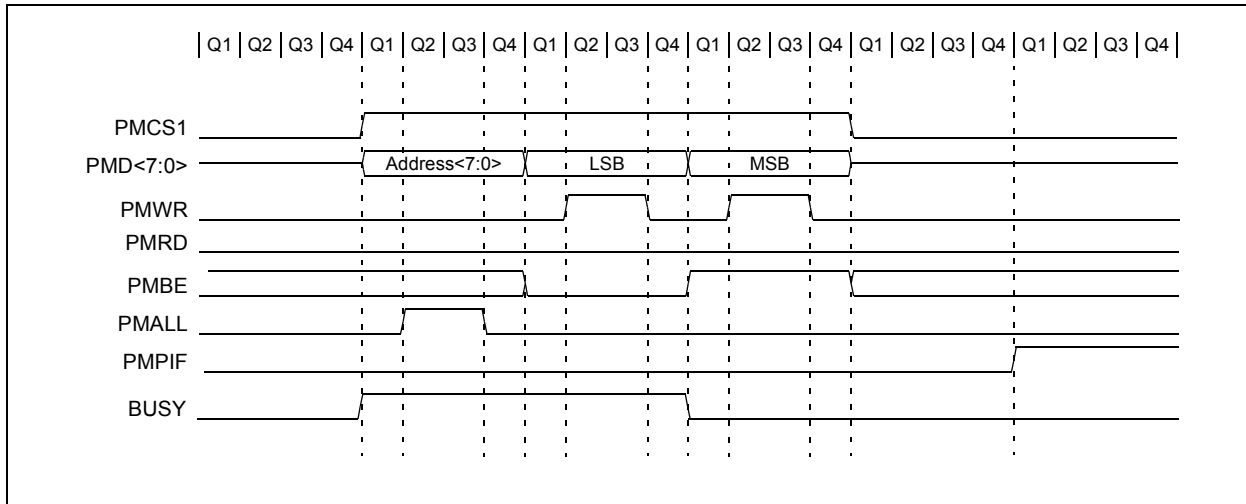


**FIGURE 10-23: READ TIMING, 16-BIT MULTIPLEXED DATA, PARTIALLY MULTIPLEXED ADDRESS**

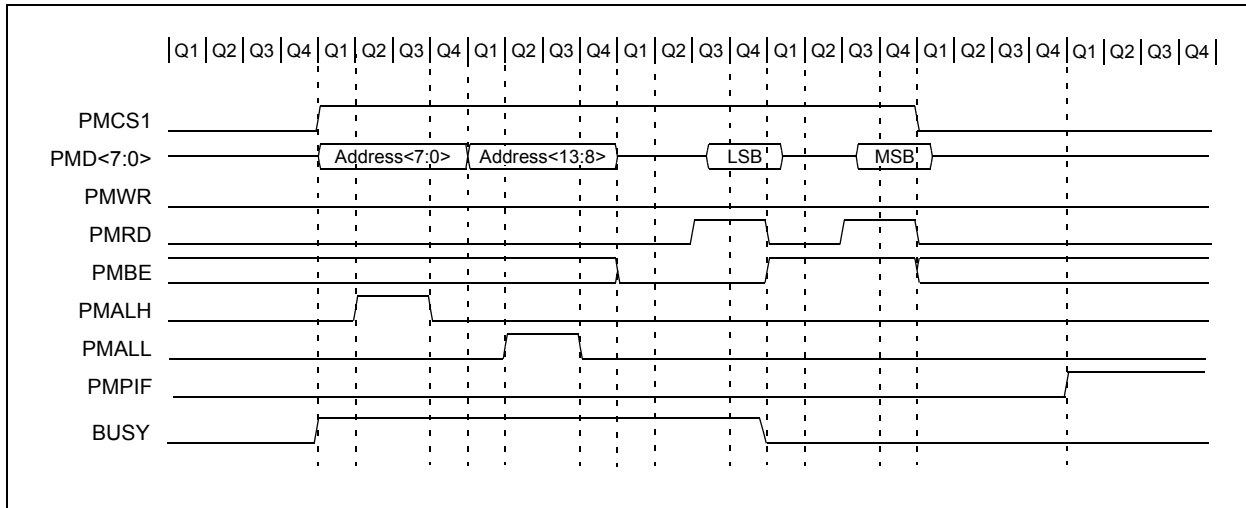


# PIC18F46J11 FAMILY

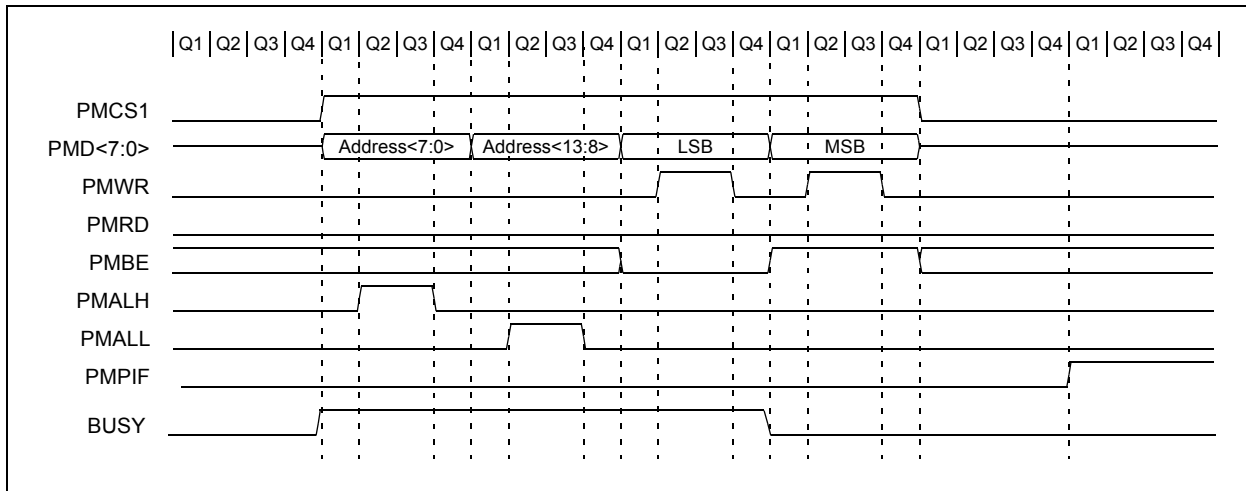
**FIGURE 10-24: WRITE TIMING, 16-BIT MULTIPLEXED DATA, PARTIALLY MULTIPLEXED ADDRESS**



**FIGURE 10-25: READ TIMING, 16-BIT MULTIPLEXED DATA, FULLY MULTIPLEXED 16-BIT ADDRESS**



**FIGURE 10-26: WRITE TIMING, 16-BIT MULTIPLEXED DATA, FULLY MULTIPLEXED 16-BIT ADDRESS**



# PIC18F46J11 FAMILY

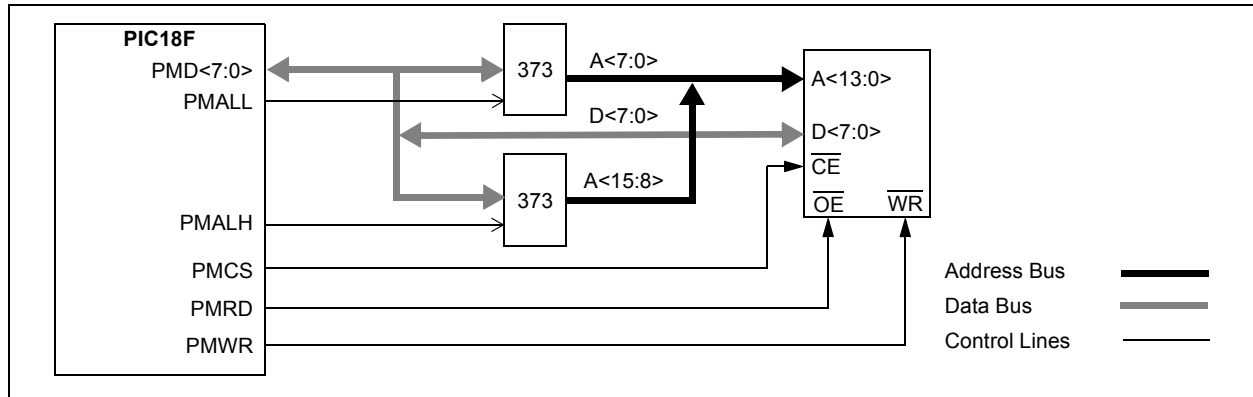
## 10.4 Application Examples

This section introduces some potential applications for the PMP module.

### 10.4.1 MULTIPLEXED MEMORY OR PERIPHERAL

Figure 10-27 demonstrates the hookup of a memory or another addressable peripheral in Full Multiplex mode. Consequently, this mode achieves the best pin saving from the microcontroller perspective. However, for this configuration, there needs to be some external latches to maintain the address.

**FIGURE 10-27: EXAMPLE – MULTIPLEXED ADDRESSING APPLICATION**

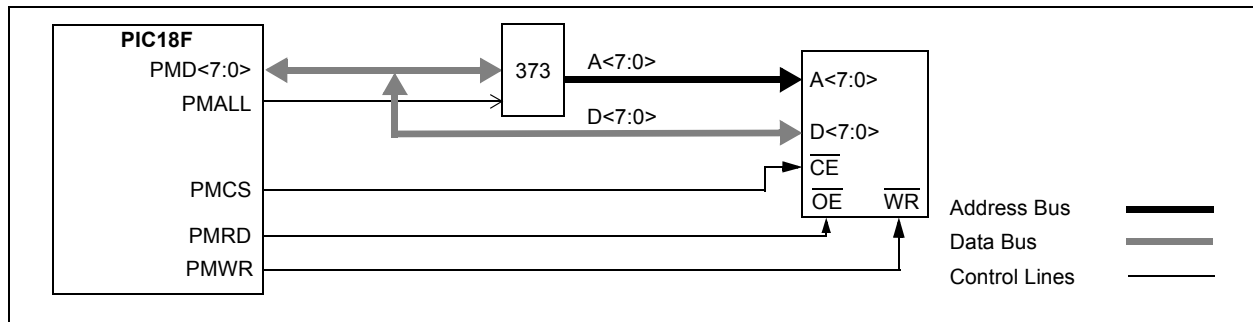


### 10.4.2 PARTIALLY MULTIPLEXED MEMORY OR PERIPHERAL

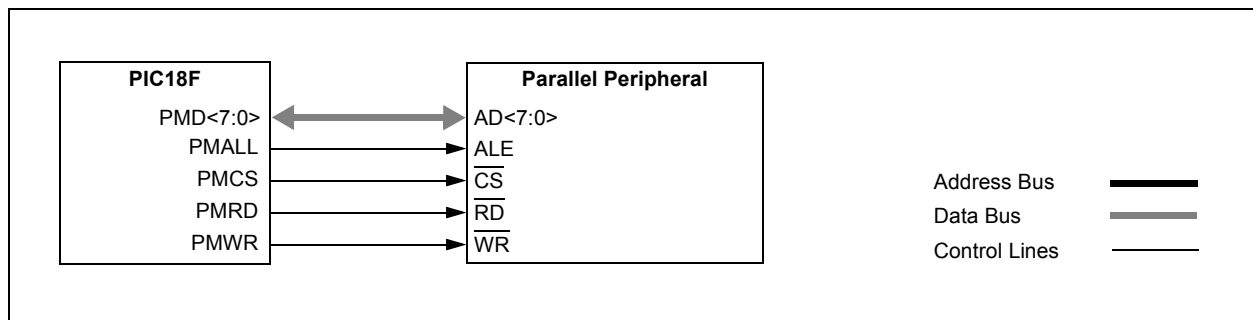
Partial multiplexing implies using more pins; however, for a few extra pins, some extra performance can be achieved. Figure 10-28 provides an example of a memory or peripheral that is partially multiplexed with

an external latch. If the peripheral has internal latches, as displayed in Figure 10-29, then no extra circuitry is required except for the peripheral itself.

**FIGURE 10-28: EXAMPLE OF A PARTIALLY MULTIPLEXED ADDRESSING APPLICATION**



**FIGURE 10-29: EXAMPLE OF AN 8-BIT MULTIPLEXED ADDRESS AND DATA APPLICATION**

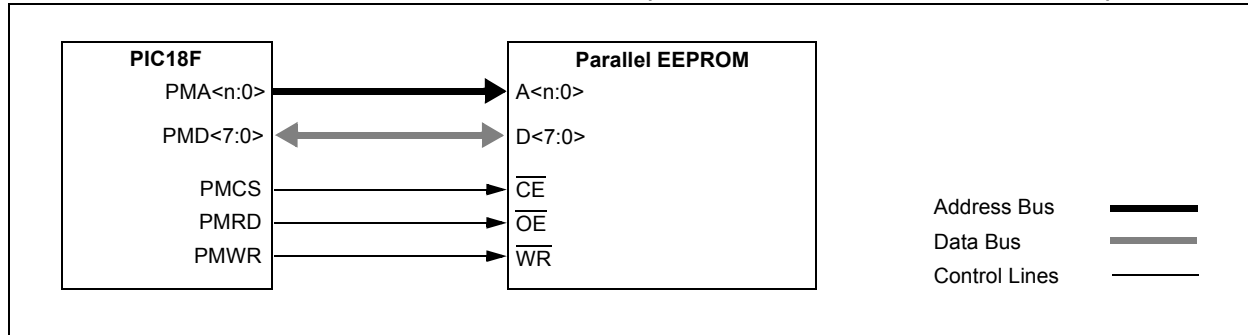


# PIC18F46J11 FAMILY

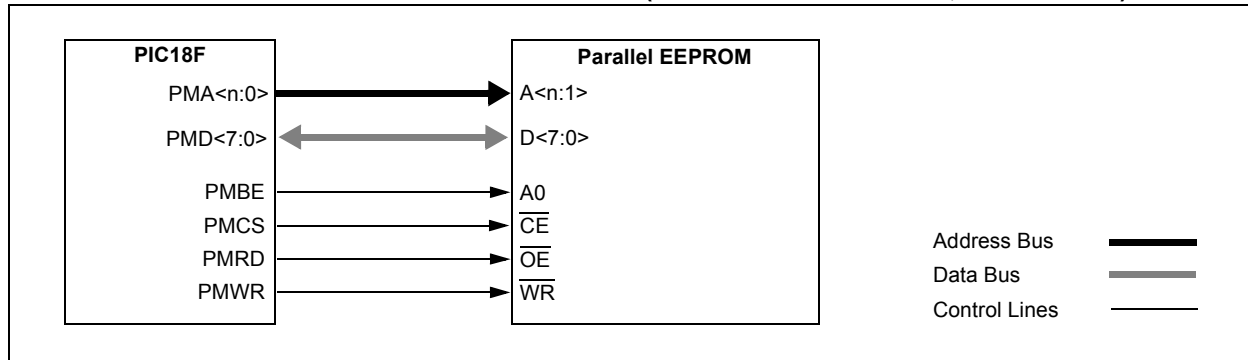
## 10.4.3 PARALLEL EEPROM EXAMPLE

Figure 10-30 provides an example connecting parallel EEPROM to the PMP. Figure 10-31 demonstrates a slight variation to this, configuring the connection for 16-bit data from a single EEPROM.

**FIGURE 10-30: PARALLEL EEPROM EXAMPLE (UP TO 15-BIT ADDRESS, 8-BIT DATA)**



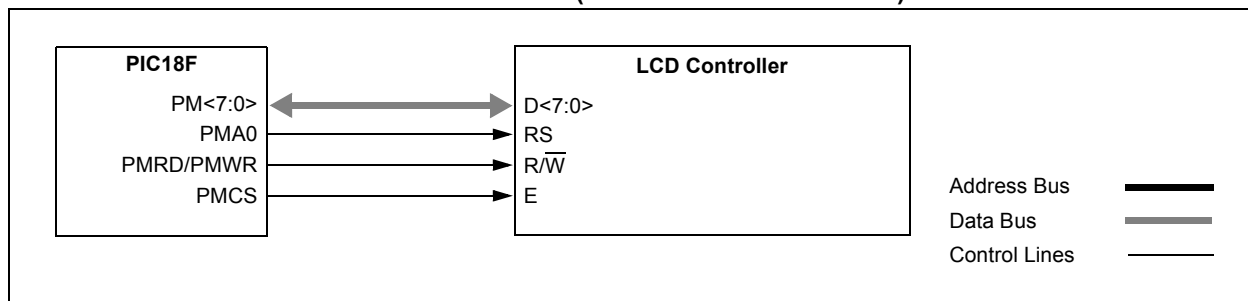
**FIGURE 10-31: PARALLEL EEPROM EXAMPLE (UP TO 15-BIT ADDRESS, 16-BIT DATA)**



## 10.4.4 LCD CONTROLLER EXAMPLE

The PMP module can be configured to connect to a typical LCD controller interface, as displayed in Figure 10-32. In this case, the PMP module is configured for active-high control signals since common LCD displays require active-high control.

**FIGURE 10-32: LCD CONTROL EXAMPLE (BYTE MODE OPERATION)**



# PIC18F46J11 FAMILY

**TABLE 10-2: REGISTERS ASSOCIATED WITH PMP MODULE**

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	63
PIR1	PMPIF <sup>(2)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(2)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPIP <sup>(2)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PMCONH <sup>(2)</sup>	PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	67
PMCONL <sup>(2)</sup>	CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP	67
PMADDRH <sup>(1,2)</sup> /	—	CS1	Parallel Master Port Address High Byte						67
PMDOUT1H <sup>(1,2)</sup>	Parallel Port Out Data High Byte (Buffer 1)								67
PMADDRL <sup>(1,2)</sup> /	Parallel Master Port Address Low Byte								67
PMDOUT1L <sup>(1,2)</sup>	Parallel Port Out Data Low Byte (Buffer 0)								67
PMDOUT2H <sup>(2)</sup>	Parallel Port Out Data High Byte (Buffer 3)								67
PMDOUT2L <sup>(2)</sup>	Parallel Port Out Data Low Byte (Buffer 2)								67
PMDIN1H <sup>(2)</sup>	Parallel Port In Data High Byte (Buffer 1)								67
PMDIN1L <sup>(2)</sup>	Parallel Port In Data Low Byte (Buffer 0)								67
PMDIN2H <sup>(2)</sup>	Parallel Port In Data High Byte (Buffer 3)								67
PMDIN2L <sup>(2)</sup>	Parallel Port In Data Low Byte (Buffer 2)								67
PMMODEH <sup>(2)</sup>	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	67
PMMODEL <sup>(2)</sup>	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	67
PMEH <sup>(2)</sup>	—	PTEN14	—	—	—	—	—	—	68
PMEL <sup>(2)</sup>	PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0	68
PMSTATH <sup>(2)</sup>	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	68
PMSTATL <sup>(2)</sup>	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	68
PADCFG1	—	—	—	—	—	RTSECSEL1	RTSECSEL0	PMPTTL	68

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

**Note 1:** The PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L register pairs share the physical registers and addresses, but have different functions determined by the module's operating mode.

**2:** These bits and/or registers are only available in 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 11.0 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 11-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

Figure 11-1 provides a simplified block diagram of the Timer0 module in 8-bit mode. Figure 11-2 provides a simplified block diagram of the Timer0 module in 16-bit mode.

**REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER (ACCESS FD5h)**

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

**Legend:**

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

- bit 7 **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 **T0PS<2:0>:** 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

# PIC18F46J11 FAMILY

## 11.1 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 11.3 “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 edge or falling edge of pin, 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

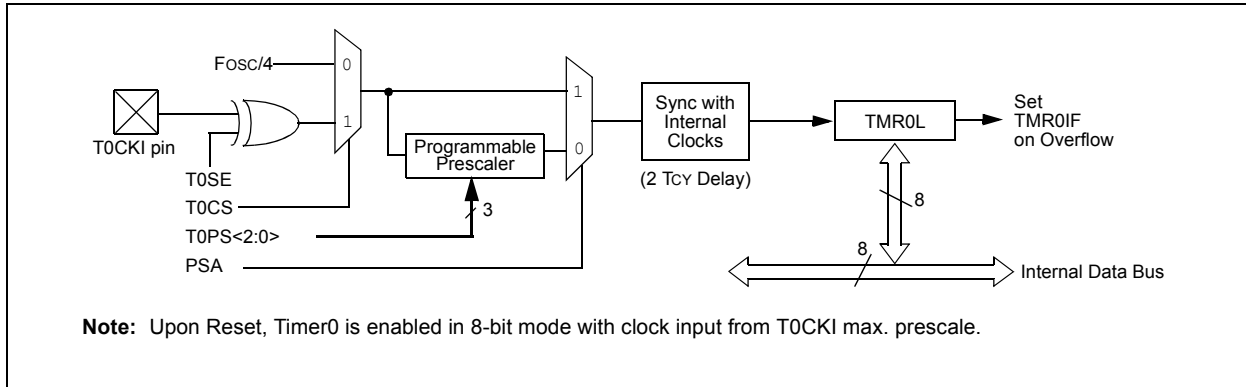
internal phase clock (Tosc). There is a delay between synchronization and the onset of incrementing the timer/counter.

## 11.2 Timer0 Reads and Writes in 16-Bit Mode

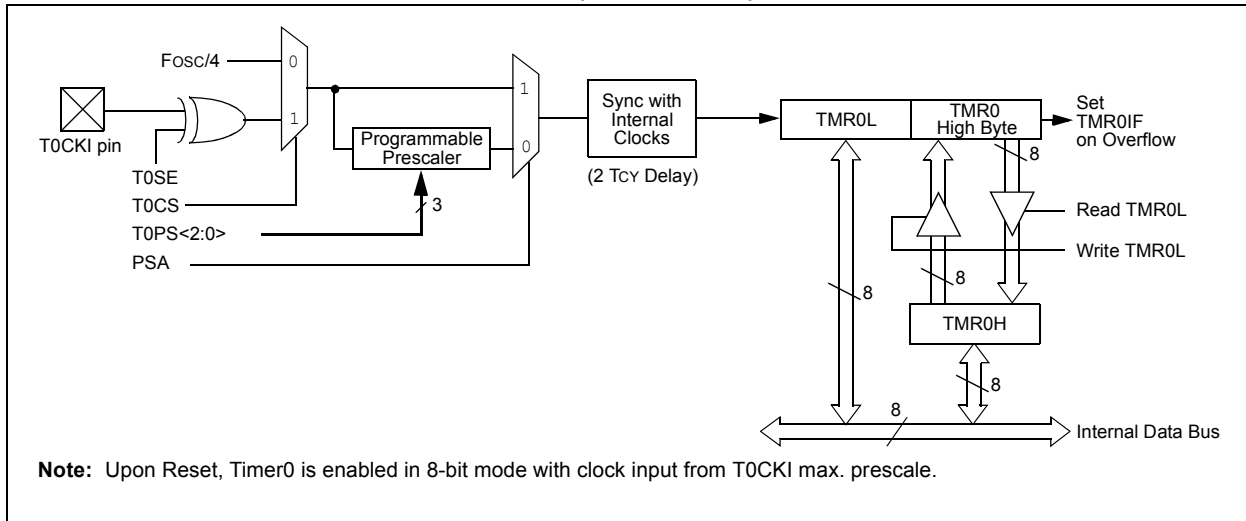
TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0, which is not directly readable nor writable (refer to Figure 11-2). TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability 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 11-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)**



**FIGURE 11-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)**





## 11.3 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 T0PS<2:0> 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.

### 11.3.1 SWITCHING PRESCALER ASSIGNMENT

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

## 11.4 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 (ISR).

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

**TABLE 11-1: 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								84
TMR0H	Timer0 Register High Byte								84
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	83
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	84

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

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 12.0 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 ECCP Special Event Trigger
- Device clock status flag (T1RUN)
- Timer with gated control

Figure 12-1 displays a simplified block diagram of the Timer1 module.

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 is controlled through the T1CON Control register (Register 12-1). It also contains the Timer1 oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

The Fosc clock source (TMR1CS<1:0> = 01) should not be used with the ECCP capture/compare features. If the timer will be used with the capture or compare features, always select one of the other timer clocking options.

### REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER (ACCESS FCDh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	RD16	TMR1ON
bit 7							bit 0

#### Legend:

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

bit 7-6 **TMR1CS<1:0>**: Timer1 Clock Source Select bits

- 10 = Timer1 clock source is T1OSC or T1CKI pin
- 01 = Timer1 clock source is system clock (Fosc)<sup>(1)</sup>
- 00 = Timer1 clock source is instruction clock (Fosc/4)

bit 5-4 **T1CKPS<1:0>**: Timer1 Input Clock Prescale Select bits

- 11 = 1:8 Prescale value
- 10 = 1:4 Prescale value
- 01 = 1:2 Prescale value
- 00 = 1:1 Prescale value

bit 3 **T1OSCEN**: Timer1 Crystal Oscillator Enable bit

- 1 = Timer1 oscillator circuit enabled
- 0 = Timer1 oscillator circuit disabled

The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **T1SYNC**: Timer1 External Clock Input Synchronization Select bit

TMR1CS<1:0> = 10:

- 1 = Do not synchronize external clock input
- 0 = Synchronize external clock input

TMR1CS<1:0> = 0x:

This bit is ignored. Timer1 uses the internal clock when TMR1CS<1:0> = 0x.

bit 1 **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 0 **TMR1ON**: Timer1 On bit

- 1 = Enables Timer1
- 0 = Stops Timer1

**Note 1:** The Fosc clock source should not be selected if the timer will be used with the ECCP capture/compare features.

# PIC18F46J11 FAMILY

## 12.1 Timer1 Gate Control Register

The Timer1 Gate Control register (T1GCON), displayed in Register 12-2, is used to control the Timer1 gate.

**REGISTER 12-2: T1GCON: TIMER1 GATE CONTROL REGISTER (F9Ah)<sup>(1)</sup>**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/T1DONE	T1GVAL	T1GSS1	T1GSS0
bit 7							bit 0

**Legend:**

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

- bit 7      **TMR1GE:** Timer1 Gate Enable bit  
           If TMR1ON = 0:  
           This bit is ignored.  
           If TMR1ON = 1:  
           1 = Timer1 counting is controlled by the Timer1 gate function  
           0 = Timer1 counts regardless of Timer1 gate function
- bit 6      **T1GPOL:** Timer1 Gate Polarity bit  
           1 = Timer1 gate is active-high (Timer1 counts when gate is high)  
           0 = Timer1 gate is active-low (Timer1 counts when gate is low)
- bit 5      **T1GTM:** Timer1 Gate Toggle Mode bit  
           1 = Timer1 Gate Toggle mode is enabled  
           0 = Timer1 Gate Toggle mode is disabled and toggle flip-flop is cleared  
           Timer1 gate flip-flop toggles on every rising edge.
- bit 4      **T1GSPM:** Timer1 Gate Single Pulse Mode bit  
           1 = Timer1 Gate Single Pulse mode is enabled and is controlling Timer1 gate  
           0 = Timer1 Gate Single Pulse mode is disabled
- bit 3      **T1GGO/T1DONE:** Timer1 Gate Single Pulse Acquisition Status bit  
           1 = Timer1 gate single pulse acquisition is ready, waiting for an edge  
           0 = Timer1 gate single pulse acquisition has completed or has not been started  
           This bit is automatically cleared when T1GSPM is cleared.
- bit 2      **T1GVAL:** Timer1 Gate Current State bit  
           Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L; unaffected by  
           Timer1 Gate Enable (TMR1GE) bit.
- bit 1-0    **T1GSS<1:0>:** Timer1 Gate Source Select bits  
           00 = Timer1 gate pin  
           01 = Timer0 overflow output  
           10 = TMR2 to match PR2 output

**Note 1:** Programming the T1GCON prior to T1CON is recommended.

# PIC18F46J11 FAMILY

## REGISTER 12-3: TCLKCON: TIMER CLOCK CONTROL REGISTER (BANKED F52h)

U-0	U-0	U-0	R-0	U-0	U-0	R/W-0	R/W-0
—	—	—	T1RUN	—	—	T3CCP2	T3CCP1
bit 7						bit 0	

### Legend:

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

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

bit 4 **T1RUN:** Timer1 Run Status bit

1 = Device is currently clocked by T1OSC/T1CKI

0 = System clock comes from an oscillator other than T1OSC/T1CKI

bit 3-2 **Unimplemented:** Read as '0'

bit 1-0 **T3CCP<2:1>:** ECCP Timer Assignment bits

10 = ECCP1 and ECCP2 both use Timer3 (capture/compare) and Timer4 (PWM)

01 = ECCP1 uses Timer1 (compare/capture) and Timer2 (PWM); ECCP2 uses Timer3 (capture/compare) and Timer4 (PWM)

00 = ECCP1 and ECCP2 both use Timer1 (capture/compare) and Timer2 (PWM)

# PIC18F46J11 FAMILY

## 12.2 Timer1 Operation

The Timer1 module is an 8-bit or 16-bit incrementing counter, which is accessed through the TMR1H:TMR1L register pair.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1 is enabled by configuring the TMR1ON and TMR1GE bits in the T1CON and T1GCON registers, respectively.

When Timer1 is enabled, the RC1/T1OSI/RP12 and RC0/T1OSO/T1CKI/RP11 pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

## 12.3 Clock Source Selection

The TMR1CS<1:0> and T1OSCEN bits of the T1CON register are used to select the clock source for Timer1. Register 12-1 displays the clock source selections.

### 12.3.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, the TMR1H:TMR1L register pair will increment on multiples of FOSC as determined by the Timer1 prescaler.

### 12.3.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input, T1CKI, or the capacitive sensing oscillator signal. Either of these external clock sources can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

**Note:** In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

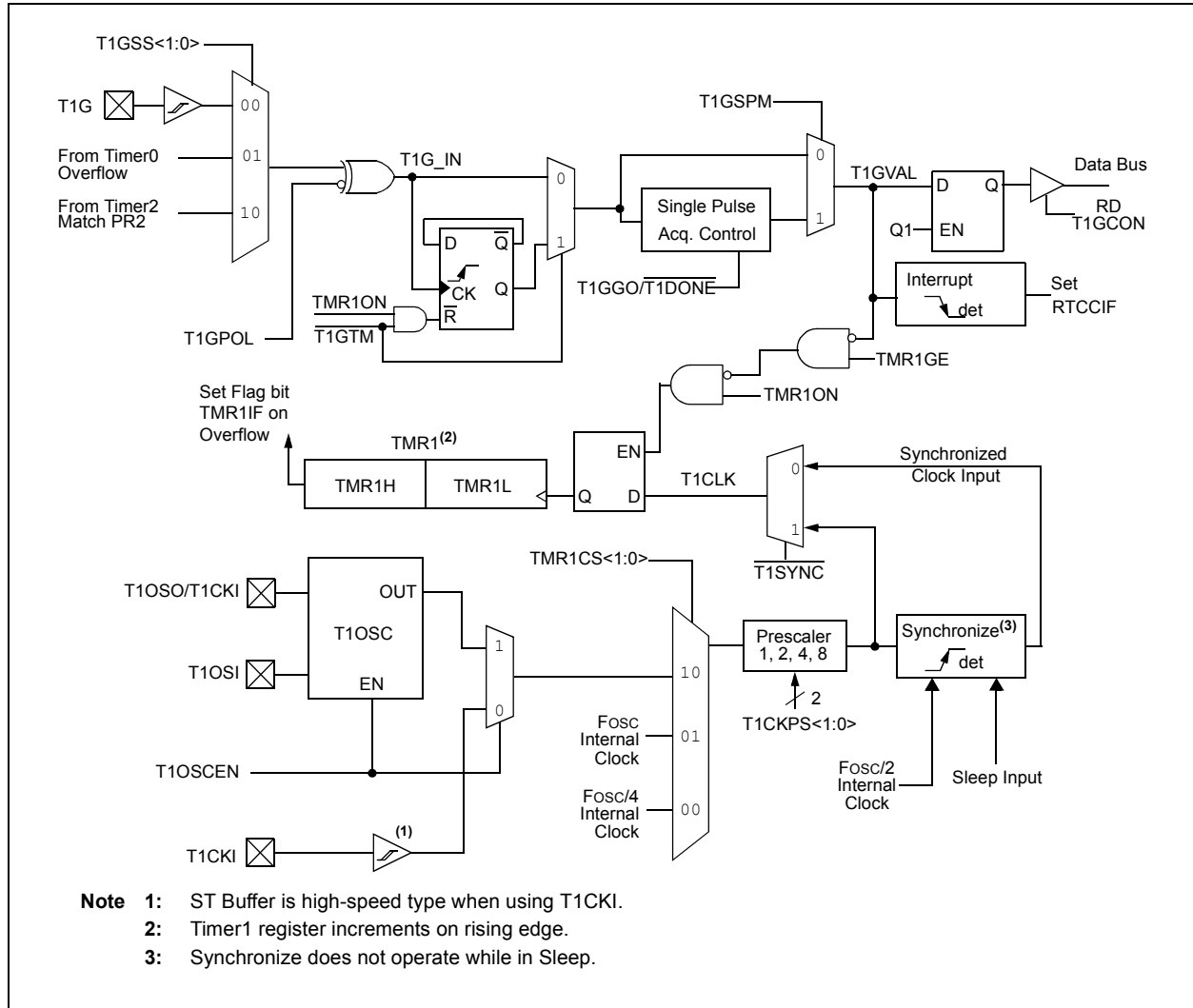
- Timer1 enabled after POR Reset
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = 0)

when T1CKI is high, then Timer1 is enabled (TMR1ON = 1) when T1CKI is low.

TABLE 12-1: TIMER1 CLOCK SOURCE SELECTION

TMR1CS1	TMR1CS0	T1OSCEN	Clock Source
0	1	x	Clock Source (FOSC)
0	0	x	Instruction Clock (FOSC/4)
1	0	0	External Clock on T1CKI Pin
1	0	1	Oscillator Circuit on T1OSI/T1OSO Pin

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



# PIC18F46J11 FAMILY

## 12.4 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes. When the RD16 control bit (T1CON<1>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L loads 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.

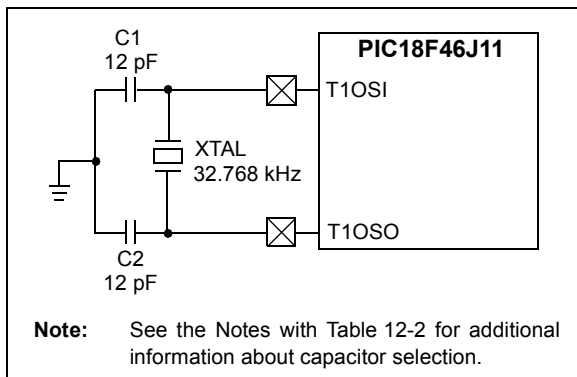
The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

## 12.5 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 LP oscillator is depicted in Figure 12-2. Table 12-2 provides the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

**FIGURE 12-2: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR**



**TABLE 12-2: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR<sup>(2,3,4,5)</sup>**

Oscillator Type	Freq.	C1	C2
LP	32 kHz	12 pF <sup>(1)</sup>	12 pF <sup>(1)</sup>

- 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. Values listed would be typical of a CL = 10 pF rated crystal, when LPT1OSC = 1.
- 5:** Incorrect capacitance value may result in a frequency not meeting the crystal manufacturer's tolerance specification.

The Timer1 crystal oscillator drive level is determined based on the LPT1OSC (CONFIG2L<4>) Configuration bit. The higher drive level mode, LPT1OSC = 1, is intended to drive a wide variety of 32.768 kHz crystals with a variety of load capacitance (CL) ratings.

The lower drive level mode is highly optimized for extremely low-power consumption. It is not intended to drive all types of 32.768 kHz crystals. In the low drive level mode, the crystal oscillator circuit may not work if excessively large discrete capacitors are placed on the T1OSI and T1OSO pins. This mode is only designed to work with discrete capacitances of approximately 3 pF-10 pF on each pin.

Crystal manufacturers usually specify a CL (load capacitance) rating for their crystals. This value is related to, but not necessarily the same as, the values that should be used for C1 and C2 in Figure 12-2. See the crystal manufacturer's applications' information for more details on how to select the optimum C1 and C2 for a given crystal. The optimum value depends in part on the amount of parasitic capacitance in the circuit, which is often unknown. Therefore, after values have been selected, it is highly recommended that thorough testing and validation of the oscillator be performed.



## 12.5.1 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, SCS<1:0> (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 "Low-Power Modes"**.

Whenever the Timer1 oscillator is providing the clock source, the Timer1 system clock status flag, T1RUN (TCLKCON<4>), is set. This can be used to determine the controller's current clocking mode. It can also indicate the clock source currently being 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.

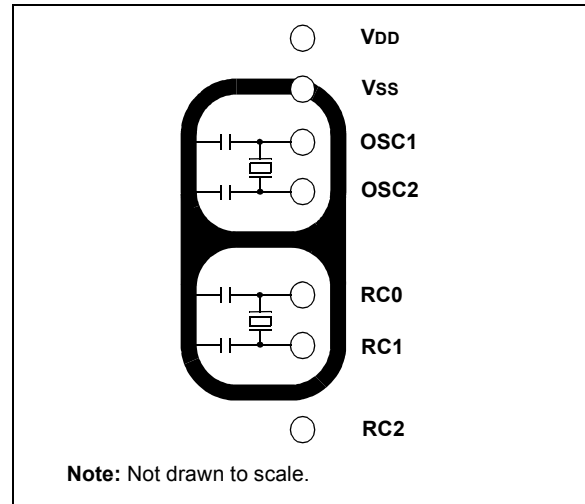
## 12.5.2 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

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. This is especially true when the oscillator is configured for extremely low power mode (LPT1OSC = 0).

The oscillator circuit, displayed in Figure 12-2, 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 ECCP1 pin in Output Compare or PWM mode, or the primary oscillator using the OSC2 pin), a grounded guard ring around the oscillator circuit, as displayed in Figure 12-3, may be helpful when used on a single-sided PCB or in addition to a ground plane.

**FIGURE 12-3: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING**



In the low drive level mode, LPT1OSC = 0, it is critical that RC2 I/O pin signals be kept away from the oscillator circuit. Configuring RC2 as a digital output, and toggling it, can potentially disturb the oscillator circuit, even with relatively good PCB layout. If possible, it is recommended to either leave RC2 unused, or use it as an input pin with a slew rate limited signal source. If RC2 must be used as a digital output, it may be necessary to use the higher drive level oscillator mode (LPT1OSC = 1) with many PCB layouts. Even in the higher drive level mode, careful layout procedures should still be followed when designing the oscillator circuit.

In addition to dV/dt induced noise considerations, it is also important to ensure that the circuit board is clean. Even a very small amount of conductive soldering flux residue can cause PCB leakage currents, which can overwhelm the oscillator circuit.

## 12.6 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>).

# PIC18F46J11 FAMILY

## 12.7 Resetting Timer1 Using the ECCP Special Event Trigger

If ECCP1 or ECCP2 is configured to use Timer1 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3. The trigger from ECCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 17.3.4 “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 CCPxH:CCPxL 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 Trigger from the ECCPx module will not set the TMR1IF interrupt flag bit (PIR1<0>).

## 12.8 Timer1 Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using the Timer1 gate circuitry. This is also referred to as Timer1 gate count enable.

The Timer1 gate can also be driven by multiple selectable sources.

### 12.8.1 TIMER1 GATE COUNT ENABLE

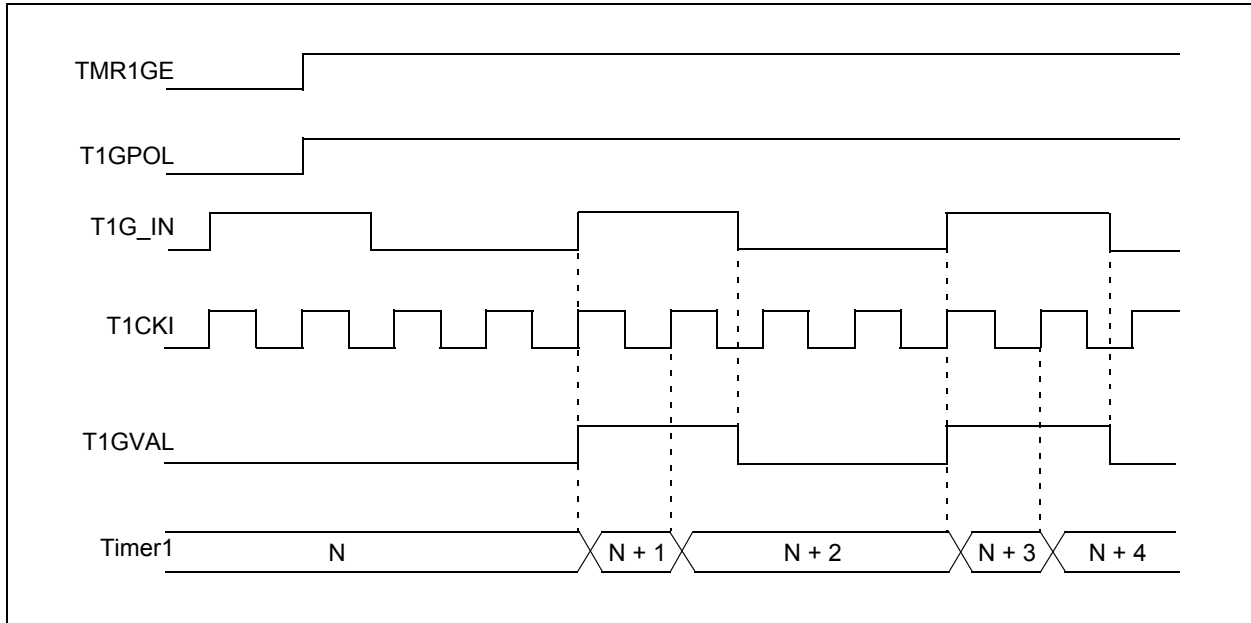
The Timer1 Gate Enable mode is enabled by setting the TMR1GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the T1GPOL bit of the T1GCON register.

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See Figure 12-4 for timing details.

**TABLE 12-3: TIMER1 GATE ENABLE SELECTIONS**

T1CLK	T1GPOL	T1G	Timer1 Operation
↑	0	0	Counts
↑	0	1	Holds Count
↑	1	0	Holds Count
↑	1	1	Counts

**FIGURE 12-4: TIMER1 GATE COUNT ENABLE MODE**



## 12.8.2 TIMER1 GATE SOURCE SELECTION

The Timer1 gate source can be selected from one of four different sources. Source selection is controlled by the T1GSSx bits of the T1GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T1GPOL bit of the T1GCON register.

**TABLE 12-4: TIMER1 GATE SOURCES**

T1GSS<1:0>	Timer1 Gate Source
00	Timer1 Gate Pin
01	Overflow of Timer0 (TMR0 increments from FFh to 00h)
10	TMR2 to Match PR2 (TMR2 increments to match PR2)

### 12.8.2.1 T1G Pin Gate Operation

The T1G pin is one source for Timer1 gate control. It can be used to supply an external source to the Timer1 gate circuitry.

### 12.8.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

### 12.8.2.3 Timer2 Match Gate Operation

The TMR2 register will increment until it matches the value in the PR2 register. On the very next increment cycle, TMR2 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

# PIC18F46J11 FAMILY

## 12.8.3 TIMER1 GATE TOGGLE MODE

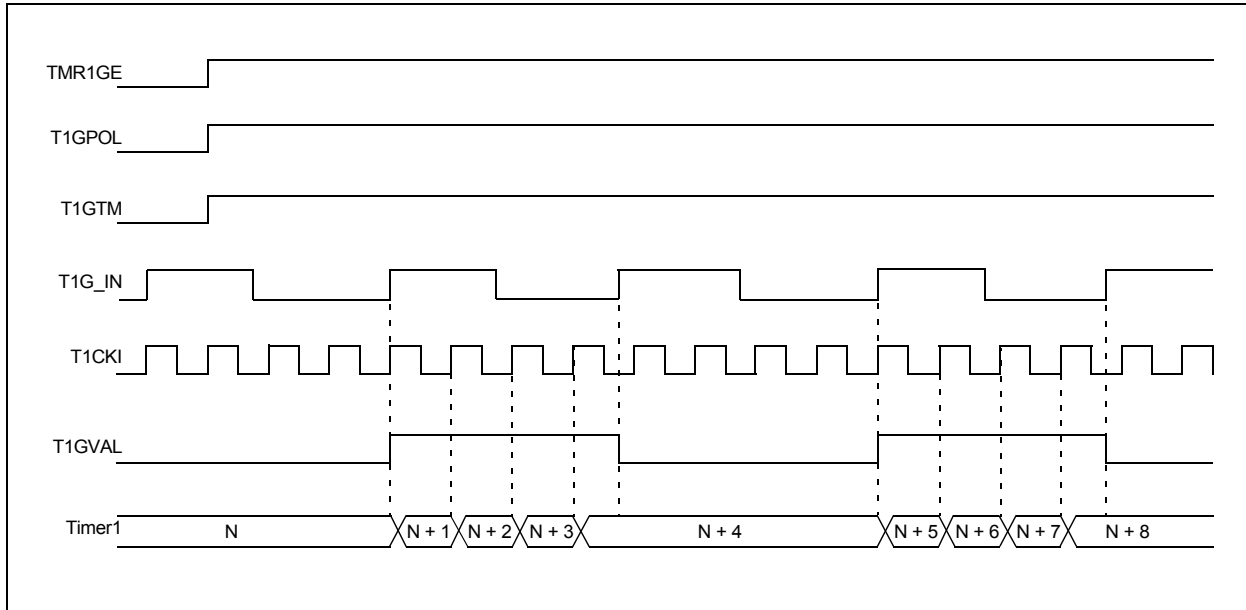
When Timer1 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer1 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See Figure 12-5 for timing details.

The T1GVAL bit will indicate when the Toggled mode is active and the timer is counting.

The Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When the T1GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

**FIGURE 12-5: TIMER1 GATE TOGGLE MODE**



## 12.8.4 TIMER1 GATE SINGLE PULSE MODE

When Timer1 Gate Single Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer1 Gate Single Pulse mode is first enabled by setting the T1GSPM bit in the T1GCON register. Next, the T1GGO/T1DONE bit in the T1GCON register must be set. The Timer1 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the T1GGO/T1DONE bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/T1DONE bit is once again set in software.

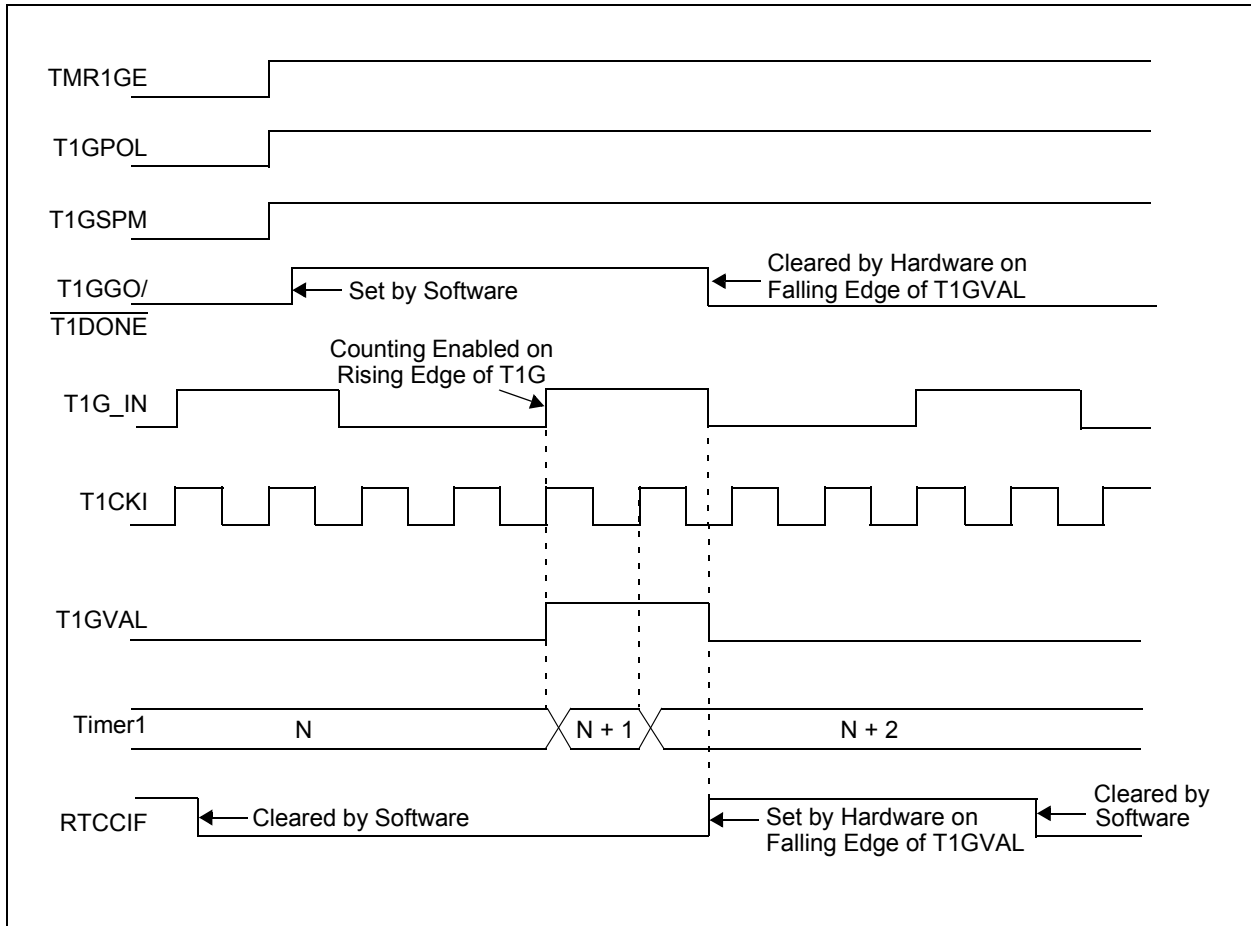
Clearing the T1GSPM bit of the T1GCON register will also clear the T1GGO/T1DONE bit. See Figure 12-6 for timing details.

Enabling the Toggle mode and the Single Pulse mode, simultaneously, will permit both sections to work together. This allows the cycle times on the Timer1 gate source to be measured. See Figure 12-7 for timing details.

## 12.8.5 TIMER1 GATE VALUE STATUS

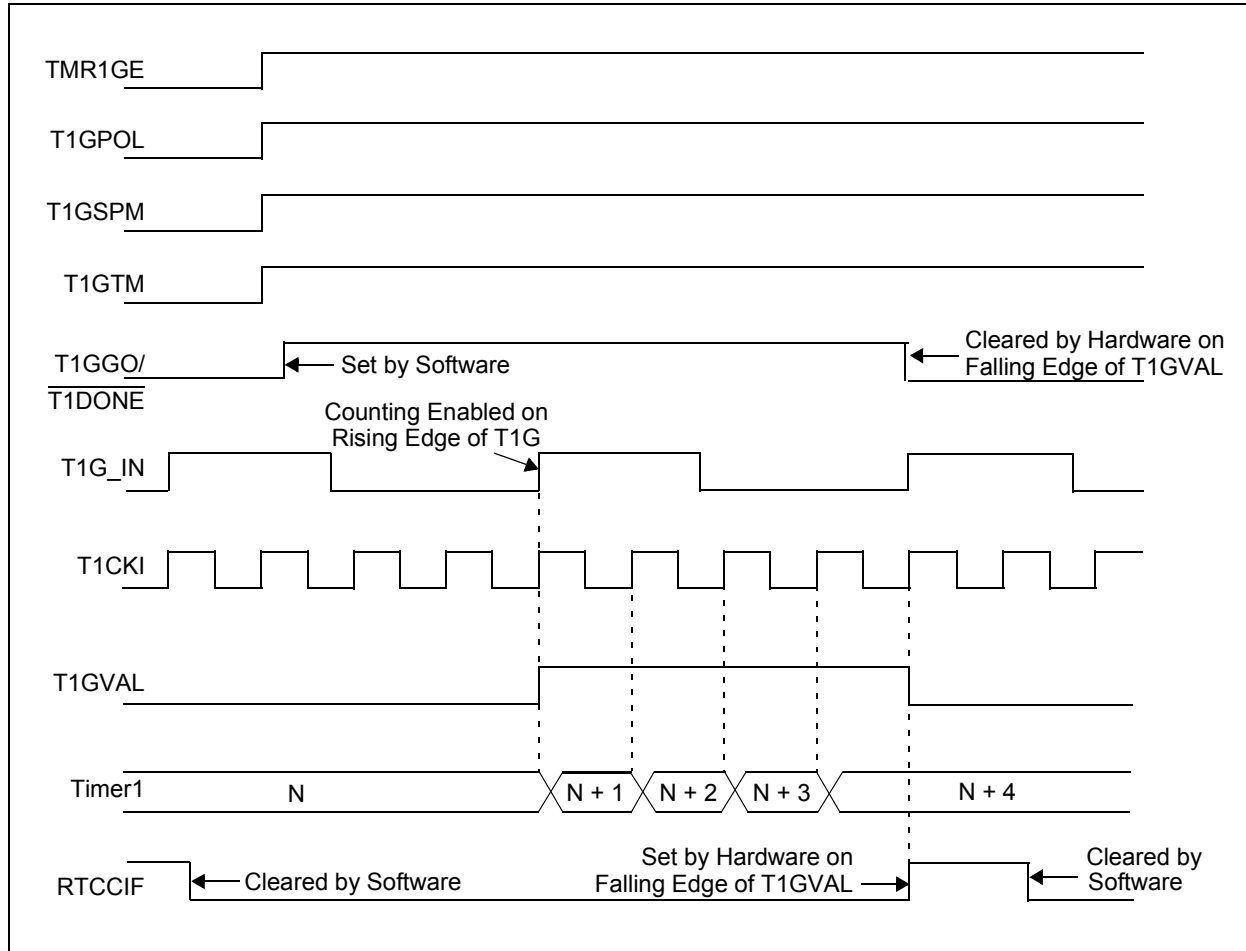
When the Timer1 gate value status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the T1GVAL bit in the T1GCON register. The T1GVAL bit is valid even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

**FIGURE 12-6: TIMER1 GATE SINGLE PULSE MODE**



# PIC18F46J11 FAMILY

**FIGURE 12-7: TIMER1 GATE SINGLE PULSE AND TOGGLE COMBINED MODE**



**TABLE 12-5: 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	83
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	85
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	85
IPR1	PMPPI <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	85
TMR1L	Timer1 Register Low Byte								84
TMR1H	Timer1 Register High Byte								84
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	RD16	TMR1ON	84
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	$\overline{T1GGO/}$ T1DONE	T1GVAL	T1GSS1	T1GSS0	85
TCLKCON	—	—	—	T1RUN	—	—	T3CCP2	T3CCP1	87

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

**Note 1:** These bits are only available in 44-pin devices.

# PIC18F46J11 FAMILY

## 13.0 TIMER2 MODULE

The Timer2 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 modules

The module is controlled through the T2CON register (Register 13-1) 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 13-1.

## 13.1 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, T2CKPS<1:0> (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 13.2 “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 (POR),  $\overline{MCLR}$  Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR))

TMR2 is not cleared when T2CON is written.

### REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER (ACCESS FCAh)

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

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 6-3 **T2OUTPS<3:0>:** 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 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits

00 = Prescaler is 1

01 = Prescaler is 4

1x = Prescaler is 16

# PIC18F46J11 FAMILY

## 13.2 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>).

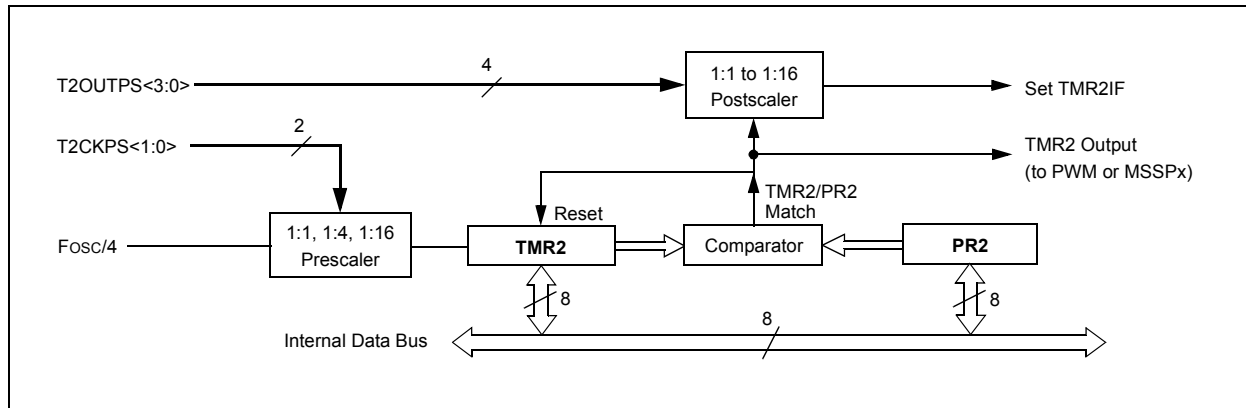
A range of 16 postscaler options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, T2OUTPS<3:0> (T2CON<6:3>).

## 13.3 Timer2 Output

The unscaled output of TMR2 is available primarily to the ECCP 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 modules operating in SPI mode. Additional information is provided in **Section 18.0 “Master Synchronous Serial Port (MSSP) Module”**.

**FIGURE 13-1: TIMER2 BLOCK DIAGRAM**



**TABLE 13-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	83
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	85
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	85
IPR1	PMP1P <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	85
TMR2	Timer2 Register								84
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	84
PR2	Timer2 Period Register								84

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

**Note 1:** These bits are only available in 44-pin devices.



# PIC18F46J11 FAMILY

## 14.0 TIMER3 MODULE

The Timer3 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR3H and TMR3L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Module Reset on ECCP Special Event Trigger

A simplified block diagram of the Timer3 module is shown in Figure 14-1.

The Timer3 module is controlled through the T3CON register (Register 14-1). It also selects the clock source options for the ECCP modules; see **Section 17.1.1 “ECCP Module and Timer Resources”** for more information.

The FOSC clock source (TMR3CS<1:0> = 01) should not be used with the ECCP capture/compare features. If the timer will be used with the capture or compare features, always select one of the other timer clocking options.

**REGISTER 14-1: T3CON: TIMER3 CONTROL REGISTER (ACCESS F79h)**

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	—	T3SYNC	RD16	TMR3ON
bit 7							bit 0

**Legend:**

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

- bit 7-6     **TMR3CS<1:0>**: Timer3 Clock Source Select bits  
 10 = Timer3 clock source is the T3CKI input pin (assigned in the PPS module)  
 01 = Timer3 clock source is the system clock (FOSC)<sup>(1)</sup>  
 00 = Timer3 clock source is the instruction clock (FOSC/4)
- bit 5-4     **T3CKPS<1:0>**: Timer3 Input Clock Prescale Select bits  
 11 = 1:8 Prescale value  
 10 = 1:4 Prescale value  
 01 = 1:2 Prescale value  
 00 = 1:1 Prescale value
- bit 3       **Reserved**: Program as ‘0’
- bit 2       **T3SYNC**: Timer3 External Clock Input Synchronization Control bit  
 When TMR3CS<1:0> = 10:  
 1 = Do not synchronize external clock input  
 0 = Synchronize external clock input  
 When TMR3CS<1:0> = 0x:  
 This bit is ignored; Timer3 uses the internal clock.
- bit 1       **RD16**: 16-Bit Read/Write Mode Enable bit  
 1 = Enables register read/write of Timer3 in one 16-bit operation  
 0 = Enables register read/write of Timer3 in two 8-bit operations
- bit 0       **TMR3ON**: Timer3 On bit  
 1 = Enables Timer3  
 0 = Stops Timer3

**Note 1:** The FOSC clock source should not be selected if the timer will be used with the ECCP capture/compare features.

# PIC18F46J11 FAMILY

## 14.1 Timer3 Gate Control Register

The Timer3 Gate Control register (T3GCON), provided in Register 14-2, is used to control the Timer3 gate.

**REGISTER 14-2: T3GCON: TIMER3 GATE CONTROL REGISTER (ACCESS F97h)<sup>(1)</sup>**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/T3DONE	T3GVAL	T3GSS1	T3GSS0
bit 7							bit 0

**Legend:**

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

- bit 7      **TMR3GE:** Timer3 Gate Enable bit  
             If TMR3ON = 0:  
             This bit is ignored.  
             If TMR3ON = 1:  
             1 = Timer3 counting is controlled by the Timer3 gate function  
             0 = Timer3 counts regardless of Timer3 gate function
- bit 6      **T3GPOL:** Timer3 Gate Polarity bit  
             1 = Timer3 gate is active-high (Timer3 counts when gate is high)  
             0 = Timer3 gate is active-low (Timer3 counts when gate is low)
- bit 5      **T3GTM:** Timer3 Gate Toggle Mode bit  
             1 = Timer3 Gate Toggle mode is enabled  
             0 = Timer3 Gate Toggle mode is disabled and toggle flip-flop is cleared  
             Timer3 gate flip-flop toggles on every rising edge.
- bit 4      **T3GSPM:** Timer3 Gate Single Pulse Mode bit  
             1 = Timer3 Gate Single Pulse mode is enabled and is controlling Timer3 gate  
             0 = Timer3 Gate Single Pulse mode is disabled
- bit 3      **T3GGO/T3DONE:** Timer3 Gate Single Pulse Acquisition Status bit  
             1 = Timer3 gate single pulse acquisition is ready, waiting for an edge  
             0 = Timer3 gate single pulse acquisition has completed or has not been started  
             This bit is automatically cleared when T3GSPM is cleared.
- bit 2      **T3GVAL:** Timer3 Gate Current State bit  
             Indicates the current state of the Timer3 gate that could be provided to TMR3H:TMR3L. Unaffected by  
             Timer3 Gate Enable bit (TMR3GE).
- bit 1-0    **T3GSS<1:0>:** Timer3 Gate Source Select bits  
             10 = TMR2 to match PR2 output  
             01 = Timer0 overflow output  
             00 = Timer3 gate pin (T3G)

**Note 1:** Programming the T3GCON prior to T3CON is recommended.

# PIC18F46J11 FAMILY

## REGISTER 14-3: TCLKCON: TIMER CLOCK CONTROL REGISTER (BANKED F52h)

U-0	U-0	U-0	R-0	U-0	U-0	R/W-0	R/W-0
—	—	—	T1RUN	—	—	T3CCP2	T3CCP1
bit 7						bit 0	

### Legend:

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

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

bit 4      **T1RUN:** Timer1 Run Status bit

1 = Device is currently clocked by T1OSC/T1CKI

0 = System clock comes from an oscillator other than T1OSC/T1CKI

bit 3-2      **Unimplemented:** Read as '0'

bit 1-0      **T3CCP<2:1>:** ECCP Timer Assignment bits

10 = ECCP1 and ECCP2 both use Timer3 (capture/compare) and Timer4 (PWM)

01 = ECCP1 uses Timer1 (compare/capture) and Timer2 (PWM); ECCP2 uses Timer3 (capture/compare) and Timer4 (PWM)

00 = ECCP1 and ECCP2 both use Timer1 (capture/compare) and Timer2 (PWM)

# PIC18F46J11 FAMILY

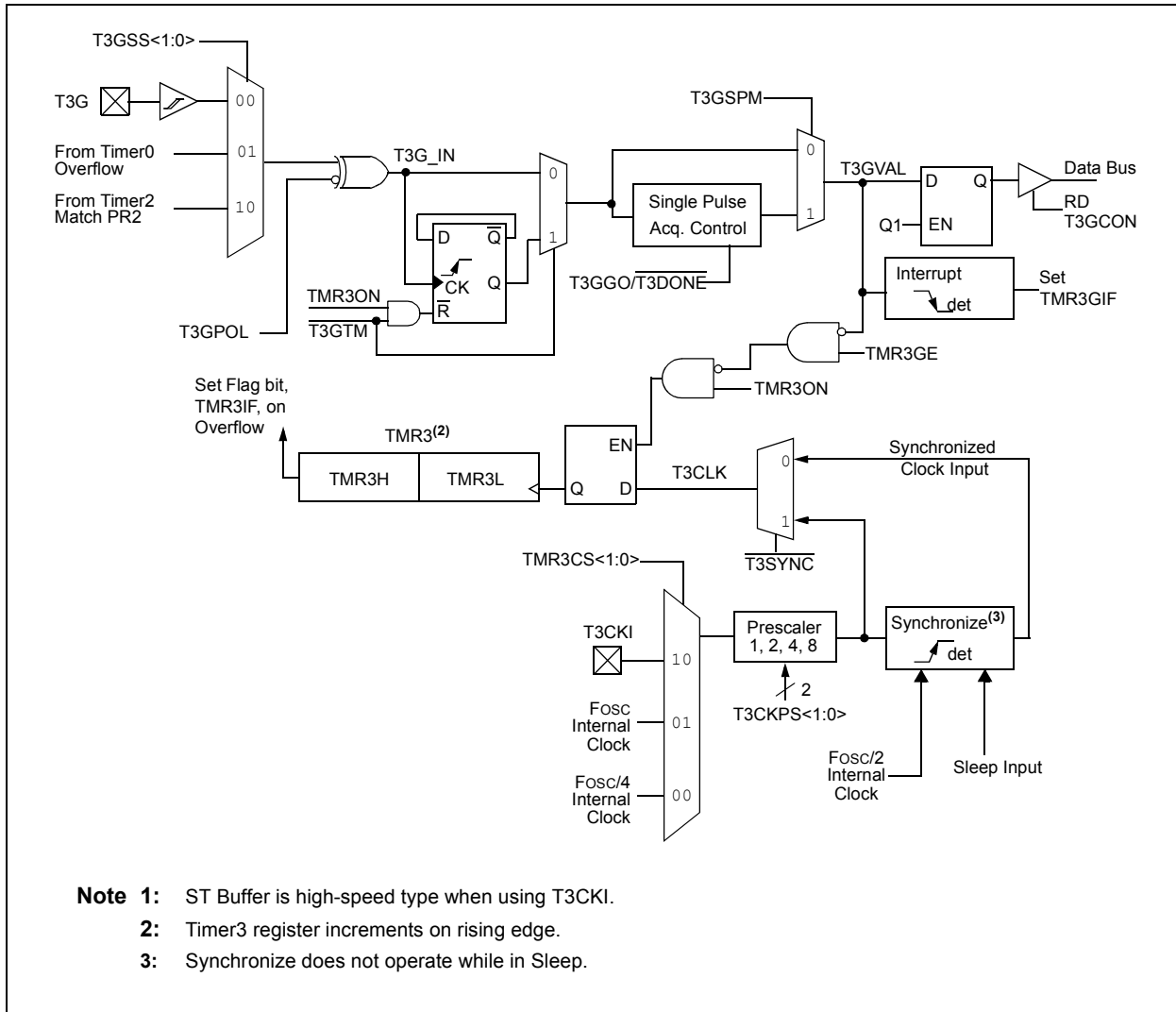
## 14.2 Timer3 Operation

Timer3 can operate in one of three modes:

- Timer
- Synchronous Counter
- Asynchronous Counter
- Timer with Gated Control

The operating mode is determined by the clock select bits, TMR3CSx (T3CON<7:6>). When the TMR3CSx bits are cleared (= 00), Timer3 increments on every internal instruction cycle (Fosc/4). When TMR3CSx = 01, the Timer3 clock source is the system clock (Fosc), and when it is '10', Timer3 works as a counter from the external clock from the T3CKI pin (on the rising edge after the first falling edge) or the Timer1 oscillator.

**FIGURE 14-1: TIMER3 BLOCK DIAGRAM**



## 14.3 Timer3 16-Bit Read/Write Mode

Timer3 can be configured for 16-bit reads and writes (see **Section 14.3 “Timer3 16-Bit Read/Write Mode”**). When the RD16 control bit (T3CON<1>) is set, the address for TMR3H is mapped to a buffer register for the high byte of Timer3. A read from TMR3L will load the contents of the high byte of Timer3 into the Timer3 High Byte Buffer register. This provides the user with the ability to accurately read all 16 bits of Timer3 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 Timer3 must also take place through the TMR3H Buffer register. The Timer3 high byte is updated with the contents of TMR3H when a write occurs to TMR3L. This allows a user to write all 16 bits to both the high and low bytes of Timer3 at once.

The high byte of Timer3 is not directly readable or writable in this mode. All reads and writes must take place through the Timer3 High Byte Buffer register.

Writes to TMR3H do not clear the Timer3 prescaler. The prescaler is only cleared on writes to TMR3L.

## 14.4 Using the Timer1 Oscillator as the Timer3 Clock Source

The Timer1 internal oscillator may be used as the clock source for Timer3. The Timer1 oscillator is enabled by setting the T1OSCEN (T1CON<3>) bit. To use it as the Timer3 clock source, the TMR3CS bit must also be set. As previously noted, this also configures Timer3 to increment on every rising edge of the oscillator source.

The Timer1 oscillator is described in **Section 12.0 “Timer1 Module”**.

## 14.5 Timer3 Gate

Timer3 can be configured to count freely, or the count can be enabled and disabled using Timer3 gate circuitry. This is also referred to as Timer3 gate count enable.

Timer3 gate can also be driven by multiple selectable sources.

### 14.5.1 TIMER3 GATE COUNT ENABLE

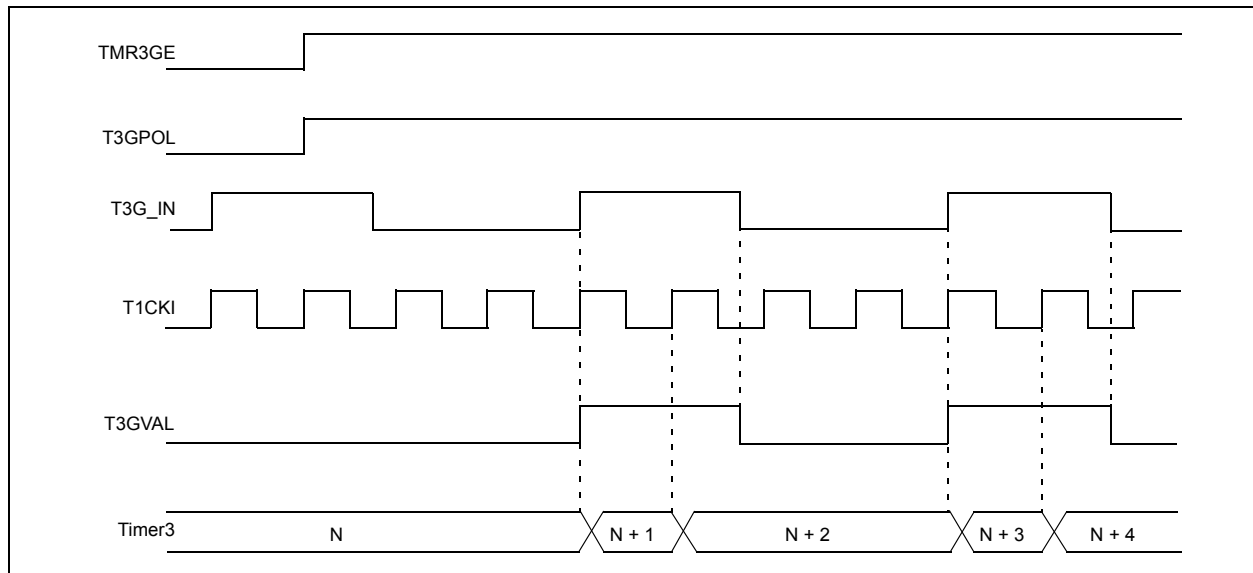
The Timer3 Gate Enable mode is enabled by setting the TMR3GE bit of the T3GCON register. The polarity of the Timer3 Gate Enable mode is configured using the T3GPOL bit of the T3GCON register.

When Timer3 Gate Enable mode is enabled, Timer3 will increment on the rising edge of the Timer3 clock source. When Timer3 Gate Enable mode is disabled, no incrementing will occur and Timer3 will hold the current count. See Figure 14-2 for timing details.

**TABLE 14-1: TIMER3 GATE ENABLE SELECTIONS**

T3CLK	T3GPOL	T3G	Timer3 Operation
↑	0	0	Counts
↑	0	1	Holds Count
↑	1	0	Holds Count
↑	1	1	Counts

**FIGURE 14-2: TIMER3 GATE COUNT ENABLE MODE**



# PIC18F46J11 FAMILY

## 14.5.2 TIMER3 GATE SOURCE SELECTION

The Timer3 gate source can be selected from one of four different sources. Source selection is controlled by the T3GSSx bits of the T3GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T3GPOL bit of the T3GCON register.

**TABLE 14-2: TIMER3 GATE SOURCES**

T3GSS<1:0>	Timer3 Gate Source
00	Timer3 Gate Pin
01	Overflow of Timer0 (TMR0 increments from FFh to 00h)
10	TMR2 to Match PR2 (TMR2 increments to match PR2)
11	Reserved

### 14.5.2.1 T3G Pin Gate Operation

The T3G pin is one source for Timer3 gate control. It can be used to supply an external source to the Timer3 gate circuitry.

### 14.5.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer3 gate circuitry.

### 14.5.2.3 Timer2 Match Gate Operation

The TMR2 register will increment until it matches the value in the PR2 register. On the very next increment cycle, TMR2 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timer3 gate circuitry.

## 14.5.3 TIMER3 GATE TOGGLE MODE

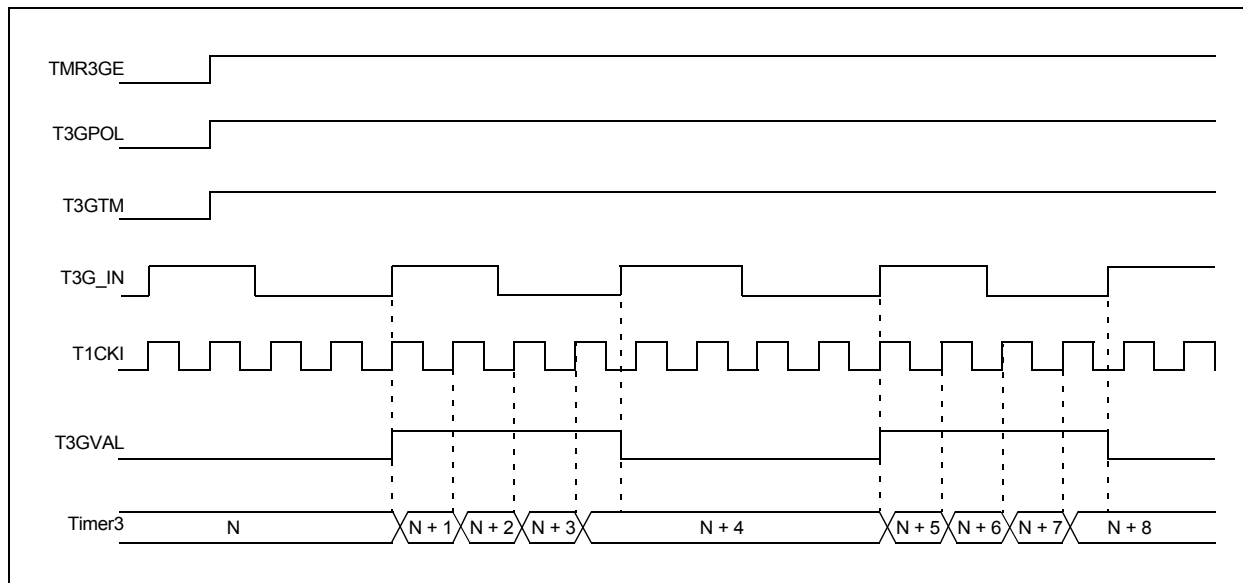
When Timer3 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer3 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See Figure 14-3 for timing details.

The T3GVAL bit will indicate when the Toggled mode is active and the timer is counting.

Timer3 Gate Toggle mode is enabled by setting the T3GTM bit of the T3GCON register. When the T3GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

**FIGURE 14-3: TIMER3 GATE TOGGLE MODE**



## 14.5.4 TIMER3 GATE SINGLE PULSE MODE

When Timer3 Gate Single Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer3 Gate Single Pulse mode is first enabled by setting the T3GSPM bit in the T3GCON register. Next, the T3GGO/T3DONE bit in the T3GCON register must be set.

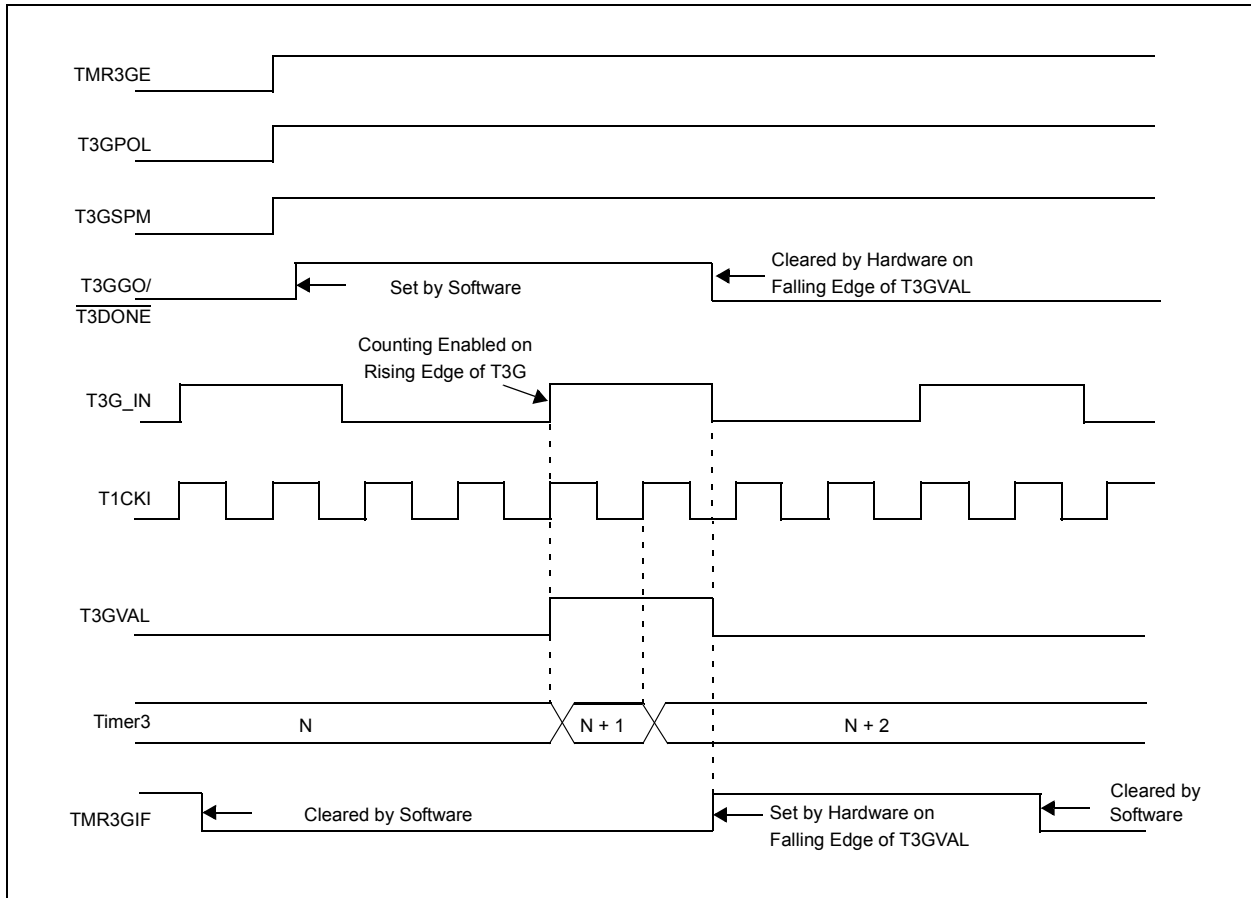
The Timer3 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the T3GGO/T3DONE bit will automatically be cleared. No

other gate events will be allowed to increment Timer3 until the T3GGO/T3DONE bit is once again set in software.

Clearing the T3GSPM bit of the T3GCON register will also clear the T3GGO/T3DONE bit. See Figure 14-4 for timing details.

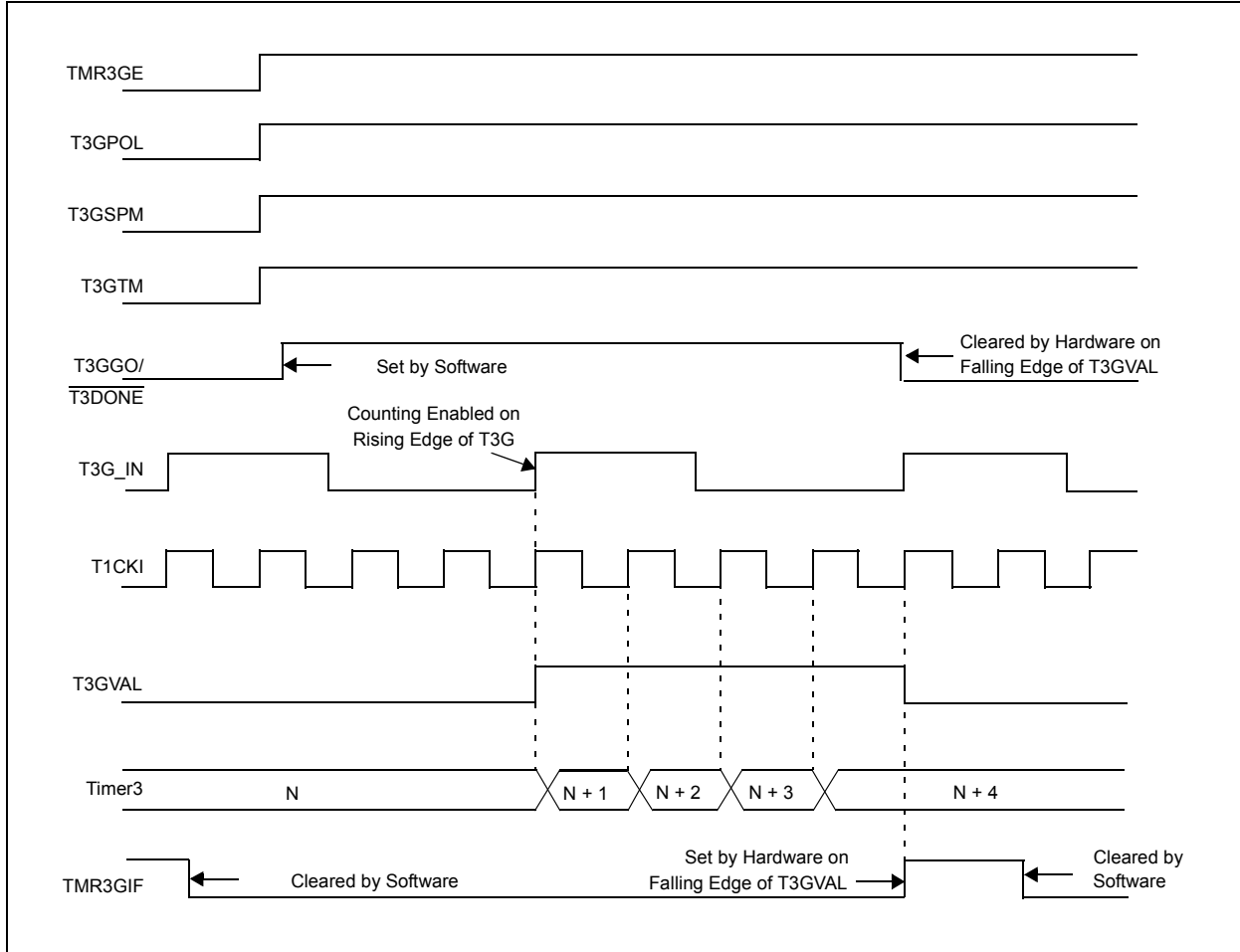
Enabling the Toggle mode and the Single Pulse mode, simultaneously, will permit both sections to work together. This allows the cycle times on the Timer3 gate source to be measured. See Figure 14-5 for timing details.

**FIGURE 14-4: TIMER3 GATE SINGLE PULSE MODE**



# PIC18F46J11 FAMILY

**FIGURE 14-5: TIMER3 GATE SINGLE PULSE AND TOGGLE COMBINED MODE**



## 14.5.5 TIMER3 GATE VALUE STATUS

When Timer3 gate value status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the T3GVAL bit in the T3GCON register. The T3GVAL bit is valid even when the Timer3 gate is not enabled (TMR3GE bit is cleared).

## 14.5.6 TIMER3 GATE EVENT INTERRUPT

When the Timer3 gate event interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of T3GVAL occurs, the TMR3GIF flag bit in the PIR3 register will be set. If the TMR3GIE bit in the PIE3 register is set, then an interrupt will be recognized.

The TMR3GIF flag bit operates even when the Timer3 gate is not enabled (TMR3GE bit is cleared).



## 14.6 Timer3 Interrupt

The TMR3 register pair (TMR3H:TMR3L) increments from 0000h to FFFFh and overflows to 0000h. The Timer3 interrupt, if enabled, is generated on overflow and is latched in interrupt flag bit, TMR3IF (PIR2<1>). This interrupt can be enabled or disabled by setting or clearing the Timer3 Interrupt Enable bit, TMR3IE (PIE2<1>).

## 14.7 Resetting Timer3 Using the ECCP Special Event Trigger

If ECCP1 or ECCP2 is configured to use Timer3 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3.

The trigger from ECCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 17.3.4 “Special Event Trigger”** for more information).

The module must be configured as either a timer or synchronous counter to take advantage of this feature. When used this way, the CCPRxH:CCPRxL register pair effectively becomes a Period register for Timer3.

If Timer3 is running in Asynchronous Counter mode, the Reset operation may not work.

In the event that a write to Timer3 coincides with a Special Event Trigger from an ECCP module, the write will take precedence.

**Note:** The Special Event Triggers from the ECCPx module will not set the TMR3IF interrupt flag bit (PIR1<0>).

**TABLE 14-3: REGISTERS ASSOCIATED WITH TIMER3 AS A TIMER/COUNTER**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	83
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	85
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	85
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	85
TMR3L	Timer3 Register Low Byte								86
TMR3H	Timer3 Register High Byte								86
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	RD16	TMR1ON	84
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	—	$\overline{T3SYNC}$	RD16	TMR3ON	86
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	$\overline{T3GGO/}$ $\overline{T3DONE}$	T3GVAL	T3GSS1	T3GSS0	85
TCLKCON	—	—	—	T1RUN	—	—	T3CCP2	T3CCP1	87
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	85
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	85
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	85

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the Timer3 module.

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 15.0 TIMER4 MODULE

The Timer4 timer module has the following features:

- 8-Bit Timer register (TMR4)
- 8-Bit Period register (PR4)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR4 match of PR4

Timer4 has a control register shown in Register 15-1. Timer4 can be shut off by clearing control bit, TMR4ON (T4CON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer4 is also controlled by this register. Figure 15-1 is a simplified block diagram of the Timer4 module.

## 15.1 Timer4 Operation

Timer4 can be used as the PWM time base for the PWM mode of the ECCP modules. The TMR4 register is readable and writable and is cleared on any device Reset. The input clock ( $F_{osc}/4$ ) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T4CKPS<1:0> (T4CON<1:0>). The match output of TMR4 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR4 interrupt, latched in flag bit, TMR4IF (PIR3<3>).

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

- a write to the TMR4 register
- a write to the T4CON register
- any device Reset (Power-on Reset (POR),  $\overline{MCLR}$  Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR))

TMR4 is not cleared when T4CON is written.

**REGISTER 15-1: T4CON: TIMER4 CONTROL REGISTER (ACCESS F76h)**

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
bit 7							bit 0

**Legend:**

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

- bit 7 **Unimplemented:** Read as '0'
- bit 6-3 **T4OUTPS<3:0>:** Timer4 Output Postscale Select bits
  - 0000 = 1:1 Postscale
  - 0001 = 1:2 Postscale
  - 
  - 
  - 
  - 1111 = 1:16 Postscale
- bit 2 **TMR4ON:** Timer4 On bit
  - 1 = Timer4 is on
  - 0 = Timer4 is off
- bit 1-0 **T4CKPS<1:0>:** Timer4 Clock Prescale Select bits
  - 00 = Prescaler is 1
  - 01 = Prescaler is 4
  - 1x = Prescaler is 16

# PIC18F46J11 FAMILY

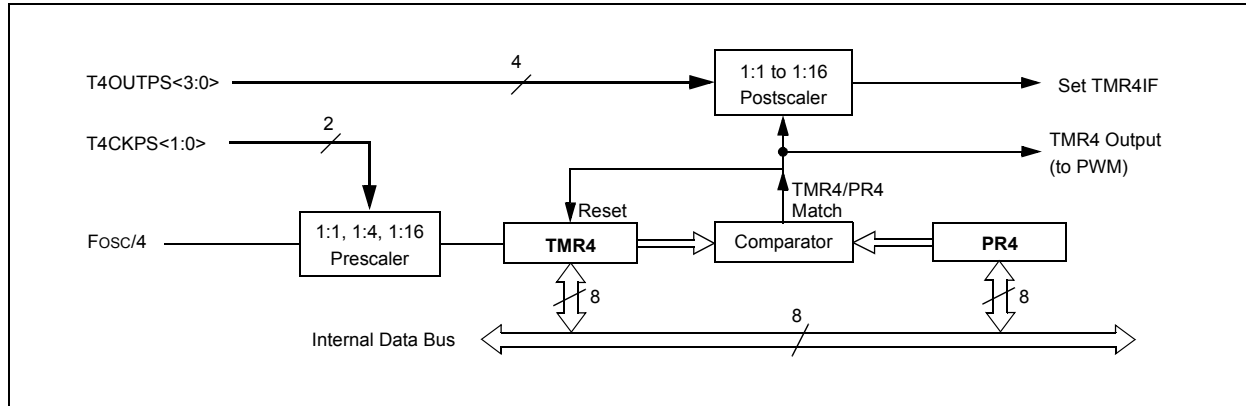
## 15.2 Timer4 Interrupt

The Timer4 module has an 8-bit Period register, PR4, which is both readable and writable. Timer4 increments from 00h until it matches PR4 and then resets to 00h on the next increment cycle. The PR4 register is initialized to FFh upon Reset.

## 15.3 Output of TMR4

The output of TMR4 (before the postscaler) is used only as a PWM time base for the ECCP modules. It is not used as a baud rate clock for the MSSP modules as is the Timer2 output.

**FIGURE 15-1: TIMER4 BLOCK DIAGRAM**



**TABLE 15-1: REGISTERS ASSOCIATED WITH TIMER4 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	83
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCIP	85
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCIF	85
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCIE	85
TMR4	Timer4 Register								86
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	86
PR4	Timer4 Period Register								86

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the Timer4 module.

## 16.0 REAL-TIME CLOCK AND CALENDAR (RTCC)

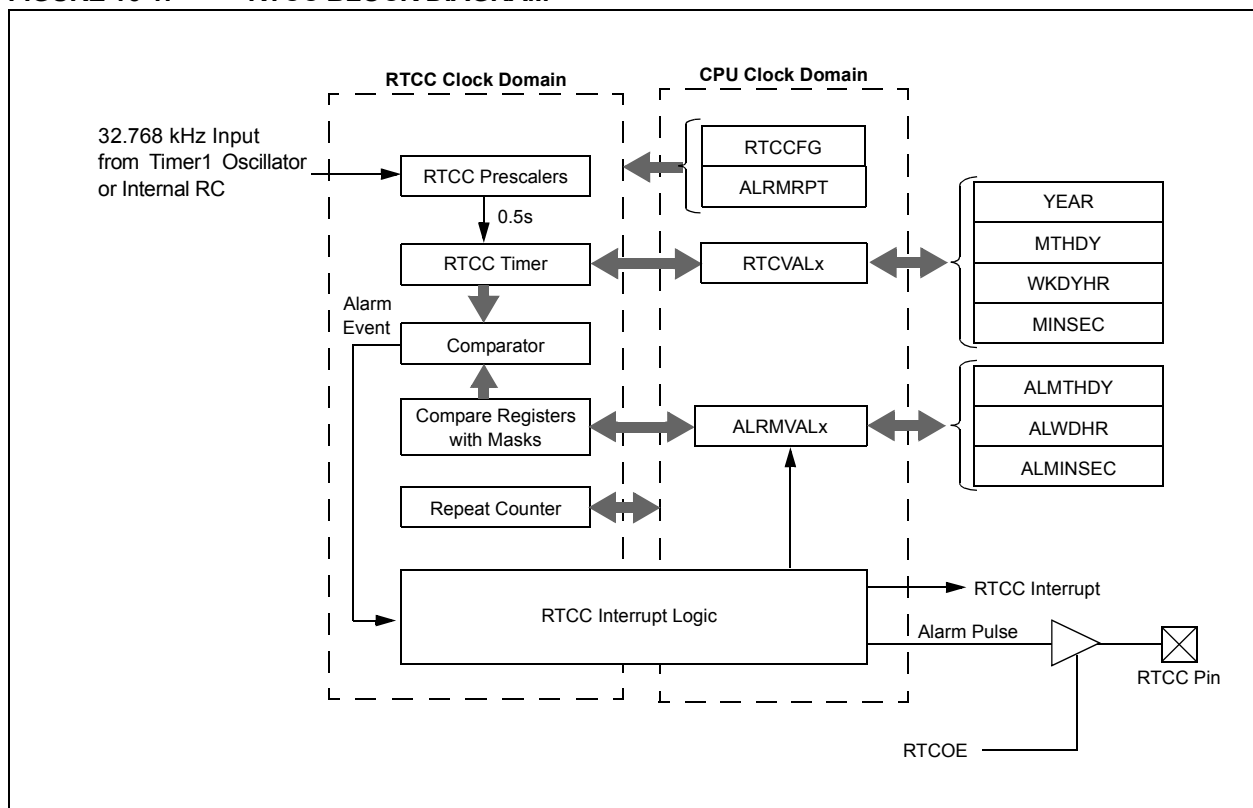
The key features of the Real-Time Clock and Calendar (RTCC) module are:

- Time: hours, minutes and seconds
- 24-hour format (military time)
- Calendar: weekday, date, month and year
- Alarm configurable
- Year range: 2000 to 2099
- Leap year correction
- BCD format for compact firmware
- Optimized for low-power operation
- User calibration with auto-adjust
- Calibration range:  $\pm 2.64$  seconds error per month
- Requirements: external 32.768 kHz clock crystal
- Alarm pulse or seconds clock output on RTCC pin

The RTCC module is intended for applications where accurate time must be maintained for an extended period with minimum to no intervention from the CPU. The module is optimized for low-power usage in order to provide extended battery life while keeping track of time.

The module is a 100-year clock and calendar with automatic leap year detection. The range of the clock is from 00:00:00 (midnight) on January 1, 2000 to 23:59:59 on December 31, 2099. Hours are measured in 24-hour (military time) format. The clock provides a granularity of one second with half-second visibility to the user.

**FIGURE 16-1: RTCC BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

---

## 16.1 RTCC MODULE REGISTERS

The RTCC module registers are divided into following categories:

### RTCC Control Registers

- RTCCFG
- RTCCAL
- PADCFG1
- ALRMCFG
- ALMRPT

### RTCC Value Registers

- RTCVALH and RTCVALL – Can access the following registers
  - YEAR
  - MONTH
  - DAY
  - WEEKDAY
  - HOUR
  - MINUTE
  - SECOND

### Alarm Value Registers

- ALRMVALH and ALRMVALL – Can access the following registers:
  - ALRMMNTH
  - ALRMDAY
  - ALRMWD
  - ALRMHR
  - ALRMMIN
  - ALRMSEC

**Note:** The RTCVALH and RTCVALL registers can be accessed through RTCRPT<1:0>. ALRMVALH and ALRMVALL can be accessed through ALRMPTR<1:0>.

# PIC18F46J11 FAMILY

## 16.1.1 RTCC CONTROL REGISTERS

### REGISTER 16-1: RTCCFG: RTCC CONFIGURATION REGISTER (BANKED F3Fh)<sup>(1)</sup>

R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN <sup>(2)</sup>	—	RTCWREN	RTCSYNC	HALFSEC <sup>(3)</sup>	RTCOE	RTCPTR1	RTCPTR0
bit 7							bit 0

#### Legend:

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

- bit 7      **RTCEN:** RTCC Enable bit<sup>(2)</sup>  
             1 = RTCC module is enabled  
             0 = RTCC module is disabled
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **RTCWREN:** RTCC Value Registers Write Enable bit  
             1 = RTCVALH and RTCVALL registers can be written to by the user  
             0 = RTCVALH and RTCVALL registers are locked out from being written to by the user
- bit 4      **RTCSYNC:** RTCC Value Registers Read Synchronization bit  
             1 = RTCVALH, RTCVALL and ALMRPT registers can change while reading due to a rollover ripple resulting in an invalid data read  
             If the register is read twice and results in the same data, the data can be assumed to be valid.  
             0 = RTCVALH, RTCVALL or ALCFGRPT registers can be read without concern over a rollover ripple
- bit 3      **HALFSEC:** Half-Second Status bit<sup>(3)</sup>  
             1 = Second half period of a second  
             0 = First half period of a second
- bit 2      **RTCOE:** RTCC Output Enable bit  
             1 = RTCC clock output enabled  
             0 = RTCC clock output disabled
- bit 1-0    **RTCPTR<1:0>:** RTCC Value Register Window Pointer bits  
             Points to the corresponding RTCC Value registers when reading the RTCVALH<7:0> and RTCVALL<7:0> registers; the RTCPTR<1:0> value decrements on every read or write of RTCVALH<7:0> until it reaches '00'.  
             RTCVALH<7:0>:  
             00 = Minutes  
             01 = Weekday  
             10 = Month  
             11 = Reserved  
             RTCVALL<7:0>:  
             00 = Seconds  
             01 = Hours  
             10 = Day  
             11 = Year

- Note 1:** The RTCCFG register is only affected by a POR.  
**2:** A write to the RTCEN bit is only allowed when RTCWREN = 1.  
**3:** This bit is read-only. It is cleared to '0' on a write to the lower half of the MINSEC register.

# PIC18F46J11 FAMILY

## REGISTER 16-2: RTCCAL: RTCC CALIBRATION REGISTER (BANKED F3Eh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

### Legend:

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

bit 7-0                      **CAL<7:0>**: RTC Drift Calibration bits  
 01111111 = Maximum positive adjustment; adds 508 RTC clock pulses every minute  
 .  
 .  
 .  
 00000001 = Minimum positive adjustment; adds four RTC clock pulses every minute  
 00000000 = No adjustment  
 11111111 = Minimum negative adjustment; subtracts four RTC clock pulses every minute  
 .  
 .  
 .  
 10000000 = Maximum negative adjustment; subtracts 512 RTC clock pulses every minute

## REGISTER 16-3: PADCFG1: PAD CONFIGURATION REGISTER (BANKED F3Ch)

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	RTSECSEL1 <sup>(1)</sup>	RTSECSEL0 <sup>(1)</sup>	PMPTTL
bit 7							bit 0

### Legend:

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

bit 7-3                      **Unimplemented:** Read as '0'  
 bit 2-1                      **RTSECSEL<1:0>**: RTCC Seconds Clock Output Select bits<sup>(1)</sup>  
 11 = Reserved, do not use  
 10 = RTCC source clock is selected for the RTCC pin (pin can be INTRC or T1OSC, depending on the RTCOSC (CONFIG3L<1>) setting)  
 01 = RTCC seconds clock is selected for the RTCC pin  
 00 = RTCC alarm pulse is selected for the RTCC pin  
 bit 0                      **PMPTTL**: PMP Module TTL Input Buffer Select bit  
 1 = PMP module uses TTL input buffers  
 0 = PMP module uses Schmitt input buffers

**Note 1:** To enable the actual RTCC output, the RTCOE (RTCCFG<2>) bit must be set.



# PIC18F46J11 FAMILY

**REGISTER 16-4: ALRMCFG: ALARM CONFIGURATION REGISTER (ACCESS F91h)**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
bit 7							bit 0

**Legend:**

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

- bit 7      **ALRMEN:** Alarm Enable bit  
 1 = Alarm is enabled (cleared automatically after an alarm event whenever ARPT<7:0> = 0000 0000 and CHIME = 0)  
 0 = Alarm is disabled
- bit 6      **CHIME:** Chime Enable bit  
 1 = Chime is enabled; ALRMRPT<7:0> bits are allowed to roll over from 00h to FFh  
 0 = Chime is disabled; ALRMRPT<7:0> bits stop once they reach 00h
- bit 5-2    **AMASK<3:0>:** Alarm Mask Configuration bits  
 0000 = Every half second  
 0001 = Every second  
 0010 = Every 10 seconds  
 0011 = Every minute  
 0100 = Every 10 minutes  
 0101 = Every hour  
 0110 = Once a day  
 0111 = Once a week  
 1000 = Once a month  
 1001 = Once a year (except when configured for February 29<sup>th</sup>, once every four years)  
 101x = Reserved – do not use  
 11xx = Reserved – do not use
- bit 1-0    **ALRMPTR<1:0>:** Alarm Value Register Window Pointer bits  
 Points to the corresponding Alarm Value registers when reading the ALRMVALH and ALRMVALL registers. The ALRMPTR<1:0> value decrements on every read or write of ALRMVALH until it reaches '00'.  
ALRMVALH<15:8>:  
 00 = ALRMMIN  
 01 = ALRMWD  
 10 = ALRMMNTH  
 11 = Unimplemented  
ALRMVALL<7:0>:  
 00 = ALRMSEC  
 01 = ALRMHR  
 10 = ALRMDAY  
 11 = Unimplemented

# PIC18F46J11 FAMILY

---

## REGISTER 16-5: ALMRPT: ALARM CALIBRATION REGISTER (ACCESS F90h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 7-0      **ARPT<7:0>**: Alarm Repeat Counter Value bits

11111111 = Alarm will repeat 255 more times

.

.

.

00000000 = Alarm will not repeat

The counter decrements on any alarm event. The counter is prevented from rolling over from 00h to FFh unless CHIME = 1.

# PIC18F46J11 FAMILY

## 16.1.2 RTCVALH AND RTCVALL REGISTER MAPPINGS

### REGISTER 16-6: RESERVED REGISTER (ACCESS F99h, PTR 11b)

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

#### Legend:

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

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

### REGISTER 16-7: YEAR: YEAR VALUE REGISTER (ACCESS F98h, PTR 11b)<sup>(1)</sup>

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
bit 7							bit 0

#### Legend:

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

bit 7-4                      **YRTEN<3:0>:** Binary Coded Decimal Value of Year's Tens Digit bits  
 Contains a value from 0 to 9.

bit 3-0                      **YRONE<3:0>:** Binary Coded Decimal Value of Year's Ones Digit bits  
 Contains a value from 0 to 9.

**Note 1:** A write to the YEAR register is only allowed when RTCWREN = 1.

### REGISTER 16-8: MONTH: MONTH VALUE REGISTER (ACCESS F99h, PTR 10b)<sup>(1)</sup>

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MHTTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

#### Legend:

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

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

bit 4                      **MHTTEN0:** Binary Coded Decimal Value of Month's Tens Digit bit  
 Contains a value of 0 or 1.

bit 3-0                      **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits  
 Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

# PIC18F46J11 FAMILY

## REGISTER 16-9: DAY: DAY VALUE REGISTER (ACCESS F98h, PTR 10b)<sup>(1)</sup>

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

### Legend:

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

- bit 7-6            **Unimplemented:** Read as '0'
- bit 5-4            **DAYTEN<1:0>:** Binary Coded Decimal value of Day's Tens Digit bits  
Contains a value from 0 to 3.
- bit 3-0            **DAYONE<3:0>:** Binary Coded Decimal Value of Day's Ones Digit bits  
Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

## REGISTER 16-10: WKDY: WEEKDAY VALUE REGISTER (ACCESS F99h, PTR 01b)<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7							bit 0

### Legend:

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

- bit 7-3            **Unimplemented:** Read as '0'
- bit 2-0            **WDAY<2:0>:** Binary Coded Decimal Value of Weekday Digit bits  
Contains a value from 0 to 6.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

# PIC18F46J11 FAMILY

## REGISTER 16-11: HOURS: HOURS VALUE REGISTER (ACCESS F98h, PTR 01b)<sup>(1)</sup>

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

### Legend:

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

- bit 7-6            **Unimplemented:** Read as '0'
- bit 5-4            **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits  
Contains a value from 0 to 2.
- bit 3-0            **HRONE<3:0>:** Binary Coded Decimal Value of Hour's Ones Digit bits  
Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

## REGISTER 16-12: MINUTES: MINUTES VALUE REGISTER (ACCESS F99h, PTR 00b)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

### Legend:

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

- bit 7                **Unimplemented:** Read as '0'
- bit 6-4            **MINTEN<2:0>:** Binary Coded Decimal Value of Minute's Tens Digit bits  
Contains a value from 0 to 5.
- bit 3-0            **MINONE<3:0>:** Binary Coded Decimal Value of Minute's Ones Digit bits  
Contains a value from 0 to 9.

## REGISTER 16-13: SECONDS: SECONDS VALUE REGISTER (ACCESS F98h, PTR 00b)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

### Legend:

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

- bit 7                **Unimplemented:** Read as '0'
- bit 6-4            **SECTEN<2:0>:** Binary Coded Decimal Value of Second's Tens Digit bits  
Contains a value from 0 to 5.
- bit 3-0            **SECONE<3:0>:** Binary Coded Decimal Value of Second's Ones Digit bits  
Contains a value from 0 to 9.

# PIC18F46J11 FAMILY

## 16.1.3 ALRMVALH AND ALRMVALL REGISTER MAPPINGS

### REGISTER 16-14: ALRMMNTH: ALARM MONTH VALUE REGISTER (ACCESS F8Fh, PTR 10b)<sup>(1)</sup>

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MHTTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

**Legend:**

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

- bit 7-5            **Unimplemented:** Read as '0'
- bit 4             **MHTTEN0:** Binary Coded Decimal Value of Month's Tens Digit bit  
Contains a value of 0 or 1.
- bit 3-0           **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits  
Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

### REGISTER 16-15: ALRMDAY: ALARM DAY VALUE REGISTER (ACCESS F8Eh, PTR 10b)<sup>(1)</sup>

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

**Legend:**

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

- bit 7-6            **Unimplemented:** Read as '0'
- bit 5-4            **DAYTEN<1:0>:** Binary Coded Decimal Value of Day's Tens Digit bits  
Contains a value from 0 to 3.
- bit 3-0            **DAYONE<3:0>:** Binary Coded Decimal Value of Day's Ones Digit bits  
Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

# PIC18F46J11 FAMILY

## REGISTER 16-16: ALRMWD: ALARM WEEKDAY VALUE REGISTER (ACCESS F8Fh, PTR 01b)<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7						bit 0	

### Legend:

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

bit 7-3                      **Unimplemented:** Read as '0'

bit 2-0                      **WDAY<2:0>:** Binary Coded Decimal Value of Weekday Digit bits  
 Contains a value from 0 to 6.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

## REGISTER 16-17: ALRMHR: ALARM HOURS VALUE REGISTER (ACCESS F8Eh, PTR 01b)<sup>(1)</sup>

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7						bit 0	

### Legend:

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

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

bit 5-4                      **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits  
 Contains a value from 0 to 2.

bit 3-0                      **HRONE3:HRONE0:** Binary Coded Decimal Value of Hour's Ones Digit bits  
 Contains a value from 0 to 9.

**Note 1:** A write to this register is only allowed when RTCWREN = 1.

# PIC18F46J11 FAMILY

## REGISTER 16-18: ALRMMIN: ALARM MINUTES VALUE REGISTER (ACCESS F8Fh, PTR 00b)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

### Legend:

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

- bit 7                      **Unimplemented:** Read as '0'
- bit 6-4                      **MINTEN<2:0>:** Binary Coded Decimal Value of Minute's Tens Digit bits  
Contains a value from 0 to 5.
- bit 3-0                      **MINONE<3:0>:** Binary Coded Decimal Value of Minute's Ones Digit bits  
Contains a value from 0 to 9.

## REGISTER 16-19: ALRMSEC: ALARM SECONDS VALUE REGISTER (ACCESS F8Eh, PTR 00b)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

### Legend:

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

- bit 7                      **Unimplemented:** Read as '0'
- bit 6-4                      **SECTEN<2:0>:** Binary Coded Decimal Value of Second's Tens Digit bits  
Contains a value from 0 to 5.
- bit 3-0                      **SECONE<3:0>:** Binary Coded Decimal Value of Second's Ones Digit bits  
Contains a value from 0 to 9.



## 16.1.4 RTCEN BIT WRITE

An attempt to write to the RTCEN bit while RTCWREN = 0 will be ignored. RTCWREN must be set before a write to RTCEN can take place.

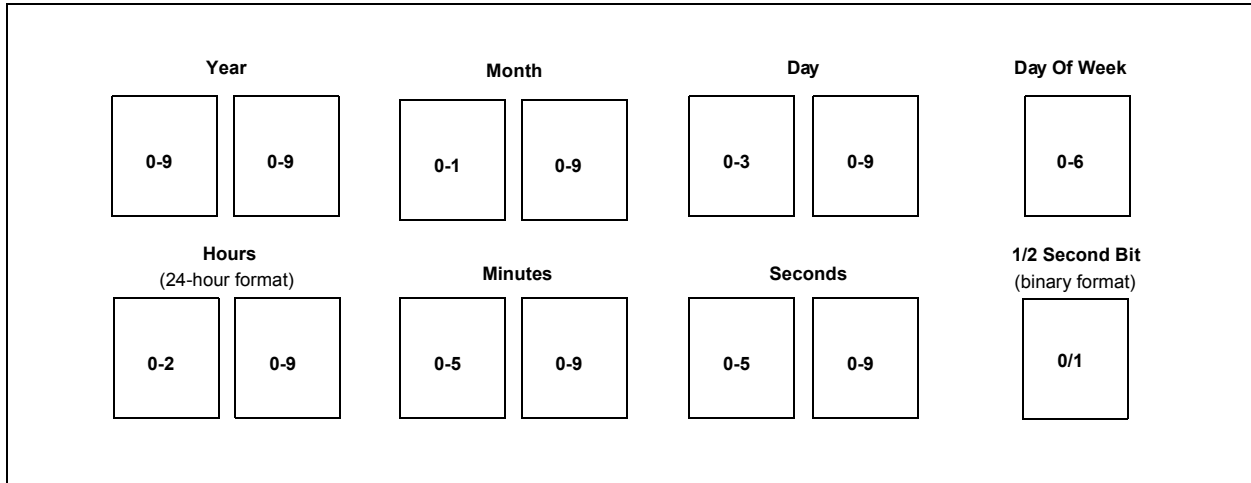
Like the RTCEN bit, the RTCVALH<15:8> and RTCVALL<7:0> registers can only be written to when RTCWREN = 1. A write to these registers, while RTCWREN = 0, will be ignored.

## 16.2 Operation

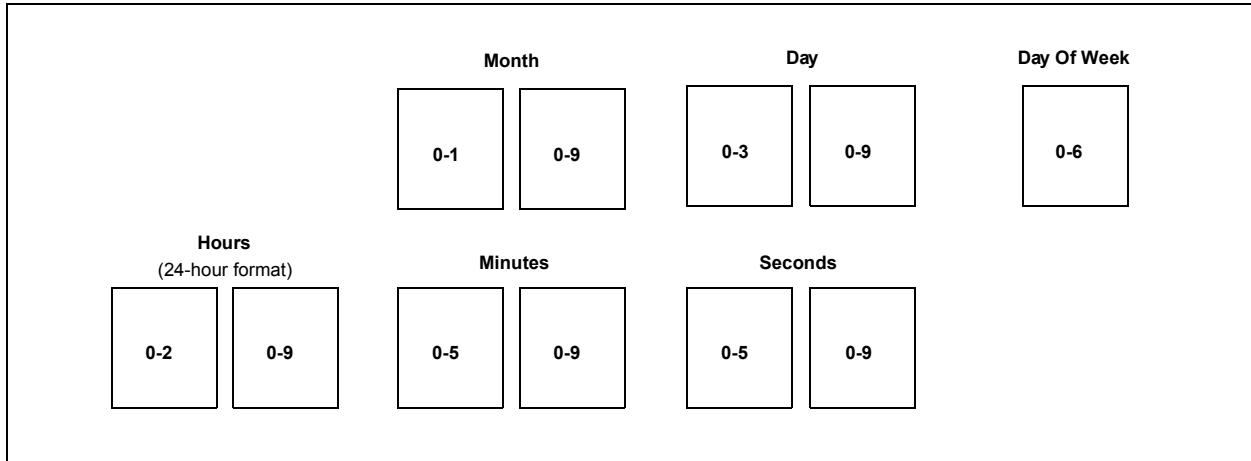
### 16.2.1 REGISTER INTERFACE

The register interface for the RTCC and alarm values is implemented using the Binary Coded Decimal (BCD) format. This simplifies the firmware, when using the module, as each of the digits is contained within its own 4-bit value (see Figure 16-2 and Figure 16-3).

**FIGURE 16-2: TIMER DIGIT FORMAT**



**FIGURE 16-3: ALARM DIGIT FORMAT**



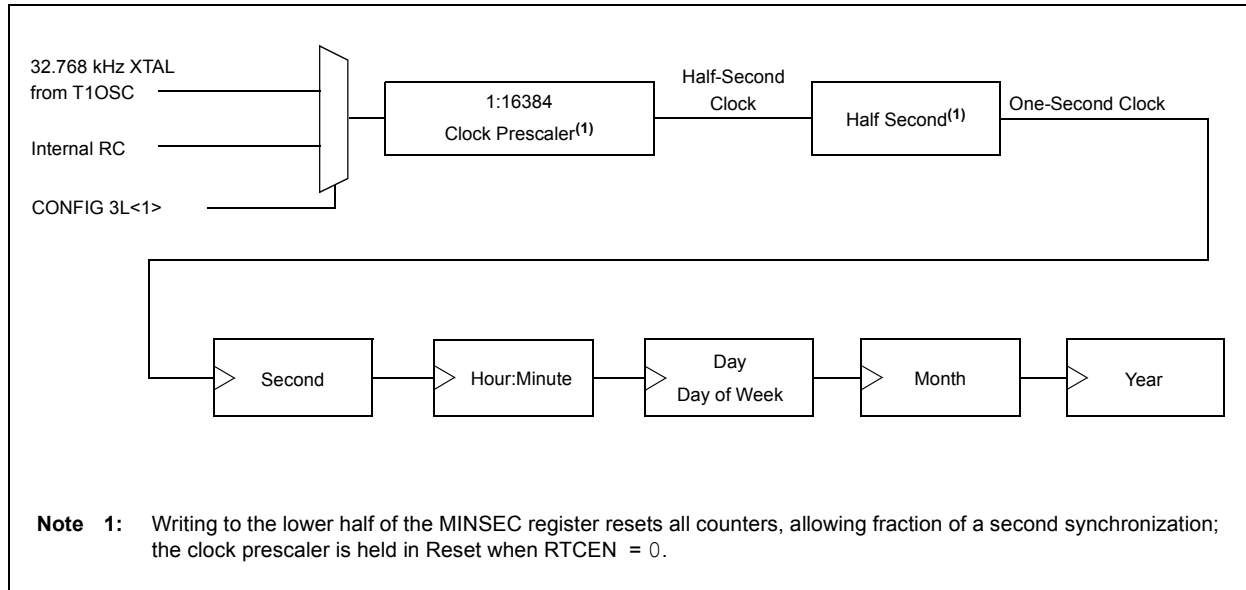
# PIC18F46J11 FAMILY

## 16.2.2 CLOCK SOURCE

As mentioned earlier, the RTCC module is intended to be clocked by an external Real-Time Clock crystal oscillating at 32.768 kHz, but also can be clocked by the internal RC oscillator. The RTCC clock selection is decided by the RTCOSC bit (CONFIG3L<1>).

Calibration of the crystal can be done through this module to yield an error of 3 seconds or less per month. (For further details, see **Section 16.2.9 “Calibration”**.)

**FIGURE 16-4: CLOCK SOURCE MULTIPLEXING**



### 16.2.2.1 Real-Time Clock Enable

The RTCC module can be clocked by an external, 32.768 kHz crystal (Timer1 oscillator) or the internal RC oscillator, which can be selected in CONFIG3L<1>.

If the external clock is used, the Timer1 oscillator should be enabled by setting T1CON<3> (T1OSCEN). If INTRC is providing the clock, the INTRC clock can be brought out to the RTCC pin by the RTSECSEL<1:0> bits in the PADCFG register.

### 16.2.3 DIGIT CARRY RULES

This section explains which timer values are affected when there is a rollover.

- Time of Day: From 23:59:59 to 00:00:00 with a carry to the Day field
- Month: From 12/31 to 01/01 with a carry to the Year field
- Day of Week: From 6 to 0 with no carry (see Table 16-1)
- Year Carry: From 99 to 00; this also surpasses the use of the RTCC

For the day to month rollover schedule, see Table 16-2.

Considering that the following values are in BCD format, the carry to the upper BCD digit will occur at a count of 10 and not at 16 (SECONDS, MINUTES, HOURS, WEEKDAY, DAYS and MONTHS).

**TABLE 16-1: DAY OF WEEK SCHEDULE**

Day of Week	
Sunday	0
Monday	1
Tuesday	2
Wednesday	3
Thursday	4
Friday	5
Saturday	6

**TABLE 16-2: DAY TO MONTH ROLLOVER SCHEDULE**

Month	Maximum Day Field
01 (January)	31
02 (February)	28 or 29 <sup>(1)</sup>
03 (March)	31
04 (April)	30
05 (May)	31
06 (June)	30
07 (July)	31
08 (August)	31
09 (September)	30
10 (October)	31
11 (November)	30
12 (December)	31

**Note 1:** See Section 16.2.4 “Leap Year”.

## 16.2.4 LEAP YEAR

Since the year range on the RTCC module is 2000 to 2099, the leap year calculation is determined by any year divisible by ‘4’ in the above range. Only February is effected in a leap year.

February will have 29 days in a leap year and 28 days in any other year.

## 16.2.5 GENERAL FUNCTIONALITY

All Timer registers containing a time value of seconds or greater are writable. The user configures the time by writing the required year, month, day, hour, minutes and seconds to the Timer registers, via Register Pointers (see Section 16.2.8 “Register Mapping”).

The timer uses the newly written values and proceeds with the count from the required starting point.

The RTCC is enabled by setting the RTCEN bit (RTCCFG<7>). If enabled, while adjusting these registers, the timer still continues to increment. However, any time the MINSEC register is written to, both of the timer prescalers are reset to ‘0’. This allows fraction of a second synchronization.

The Timer registers are updated in the same cycle as the write instruction’s execution by the CPU. The user must ensure that when RTCEN = 1, the updated registers will not be incremented at the same time. This can be accomplished in several ways:

- By checking the RTCSYNC bit (RTCCFG<4>)
- By checking the preceding digits from which a carry can occur
- By updating the registers immediately following the seconds pulse (or alarm interrupt)

The user has visibility to the half-second field of the counter. This value is read-only and can be reset only by writing to the lower half of the SECONDS register.

## 16.2.6 SAFETY WINDOW FOR REGISTER READS AND WRITES

The RTCSYNC bit indicates a time window during which the RTCC Clock Domain registers can be safely read and written without concern about a rollover. When RTCSYNC = 0, the registers can be safely accessed by the CPU.

Whether RTCSYNC = 1 or 0, the user should employ a firmware solution to ensure that the data read did not fall on a rollover boundary, resulting in an invalid or partial read. This firmware solution would consist of reading each register twice and then comparing the two values. If the two values matched, then, a rollover did not occur.

## 16.2.7 WRITE LOCK

In order to perform a write to any of the RTCC Timer registers, the RTCWREN bit (RTCCFG<5>) must be set.

To avoid accidental writes to the RTCC Timer register, it is recommended that the RTCWREN bit (RTCCFG<5>) be kept clear at any time other than while writing to. For the RTCWREN bit to be set, there is only one instruction cycle time window allowed between the 55h/AA sequence and the setting of RTCWREN. For that reason, it is recommended that users follow the code example in Example 16-1.

### EXAMPLE 16-1: SETTING THE RTCWREN BIT

```
movlb    0x0f
movlw    0x55
movwf    EECON2, 0
movlw    0xAA
movwf    EECON2, 0
bsf      RTCCFG, 5, 1
```

## 16.2.8 REGISTER MAPPING

To limit the register interface, the RTCC Timer and Alarm Timer registers are accessed through corresponding register pointers. The RTCC Value register window (RTCVALH and RTCVALL) uses the RTCPTR bits (RTCCFG<1:0>) to select the required Timer register pair.

By reading or writing to the RTCVALH register, the RTCC Pointer value (RTCPTR<1:0>) decrements by 1 until it reaches ‘00’. Once it reaches ‘00’, the MINUTES and SECONDS value will be accessible through RTCVALH and RTCVALL until the pointer value is manually changed.

# PIC18F46J11 FAMILY

**TABLE 16-3: RTCVLH AND RTCVALL REGISTER MAPPING**

RTCPTR<1:0>	RTCC Value Register Window	
	RTCVLH<15:8>	RTCVALL<7:0>
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

The Alarm Value register window (ALRMVALH and ALRMVALL) uses the ALRMPTR bits (ALRMCFG<1:0>) to select the desired Alarm register pair.

By reading or writing to the ALRMVALH register, the Alarm Pointer value, ALRMPTR<1:0>, decrements by 1 until it reaches '00'. Once it reaches '00', the ALRMMIN and ALRMSEC value will be accessible through ALRMVALH and ALRMVALL until the pointer value is manually changed.

**TABLE 16-4: ALRMVAL REGISTER MAPPING**

ALRMPTR<1:0>	Alarm Value Register Window	
	ALRMVALH<15:8>	ALRMVALL<7:0>
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

## 16.2.9 CALIBRATION

The real-time crystal input can be calibrated using the periodic auto-adjust feature. When properly calibrated, the RTCC can provide an error of less than three seconds per month.

To perform this calibration, find the number of error clock pulses and store the value in the lower half of the RTCCAL register. The 8-bit, signed value – loaded into RTCCAL – is multiplied by '4' and will either be added or subtracted from the RTCC timer, once every minute.

To calibrate the RTCC module:

1. Use another timer resource on the device to find the error of the 32.768 kHz crystal.
2. Convert the number of error clock pulses per minute (see Equation 16-1).

**EQUATION 16-1: CONVERTING ERROR CLOCK PULSES**

$$(Ideal\ Frequency\ (32,758) - Measured\ Frequency) * 60 = Error\ Clocks\ per\ Minute$$

- If the oscillator is *faster* than ideal (negative result from step 2), the RTCCALL register value needs to be negative. This causes the specified number of clock pulses to be subtracted from the timer counter once every minute.
  - If the oscillator is *slower* than ideal (positive result from step 2), the RTCCALL register value needs to be positive. This causes the specified number of clock pulses to be added to the timer counter once every minute.
3. Load the RTCCAL register with the correct value.

Writes to the RTCCAL register should occur only when the timer is turned off, or immediately after the rising edge of the seconds pulse.

**Note:** In determining the crystal's error value, it is the user's responsibility to include the crystal's initial error from drift due to temperature or crystal aging.

## 16.3 Alarm

The alarm features and characteristics are:

- Configurable from half a second to one year
- Enabled using the ALRMEN bit (ALRMCFG<7>, Register 16-4)
- Offers one-time and repeat alarm options

### 16.3.1 CONFIGURING THE ALARM

The alarm feature is enabled using the ALRMEN bit.

This bit is cleared when an alarm is issued. The bit will not be cleared if the CHIME bit = 1 or if ALMRPT ≠ 0.

The interval selection of the alarm is configured through the ALRMCFG bits (AMASK<3:0>). (See Figure 16-5.) These bits determine which and how many digits of the alarm must match the clock value for the alarm to occur.

The alarm can also be configured to repeat based on a preconfigured interval. The number of times this occurs after the alarm is enabled is stored in the ALMRPT register.

**Note:** While the alarm is enabled (ALRMEN = 1), changing any of the registers – other than the RTCCAL, ALRMCFG and ALMRPT registers and the CHIME bit – can result in a false alarm event leading to a false alarm interrupt. To avoid this, only change the timer and alarm values while the alarm is disabled (ALRMEN = 0). It is recommended that the ALRMCFG and ALMRPT registers and CHIME bit be changed when RTCSYNC = 0.

**FIGURE 16-5: ALARM MASK SETTINGS**

Alarm Mask Setting AMASK<3:0>	Day of the Week	Month	Day	Hours	Minutes	Seconds
0000 – Every half second	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
0001 – Every second	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>
0010 – Every 10 seconds	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>s</b>
0011 – Every minute	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
0100 – Every 10 minutes	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
0101 – Every hour	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> : <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
0110 – Every day	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>h</b> <input type="checkbox"/> <b>h</b>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
0111 – Every week	<input type="checkbox"/> <b>d</b>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>h</b> <input type="checkbox"/> <b>h</b>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
1000 – Every month	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> / <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> <b>d</b> <input type="checkbox"/> <b>d</b>	<input type="checkbox"/> <b>h</b> <input type="checkbox"/> <b>h</b>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>
1001 – Every year <sup>(1)</sup>	<input type="checkbox"/>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>d</b> <input type="checkbox"/> <b>d</b>	<input type="checkbox"/> <b>h</b> <input type="checkbox"/> <b>h</b>	<input type="checkbox"/> <b>m</b> <input type="checkbox"/> <b>m</b>	<input type="checkbox"/> <b>s</b> <input type="checkbox"/> <b>s</b>

**Note 1:** Annually, except when configured for February 29.

# PIC18F46J11 FAMILY

When  $ALRMCFG = 00$  and the CHIME bit = 0 ( $ALRMCFG<6>$ ), the repeat function is disabled and only a single alarm will occur. The alarm can be repeated up to 255 times by loading the ALMRPT register with FFh.

After each alarm is issued, the ALMRPT register is decremented by one. Once the register has reached '00', the alarm will be issued one last time.

After the alarm is issued a last time, the ALRMEN bit is cleared automatically and the alarm turned off. Indefinite repetition of the alarm can occur if the CHIME bit = 1.

When CHIME = 1, the alarm is not disabled when the ALMRPT register reaches '00', but it rolls over to FF and continues counting indefinitely.

## 16.3.2 ALARM INTERRUPT

At every alarm event, an interrupt is generated. Additionally, an alarm pulse output is provided that operates at half the frequency of the alarm.

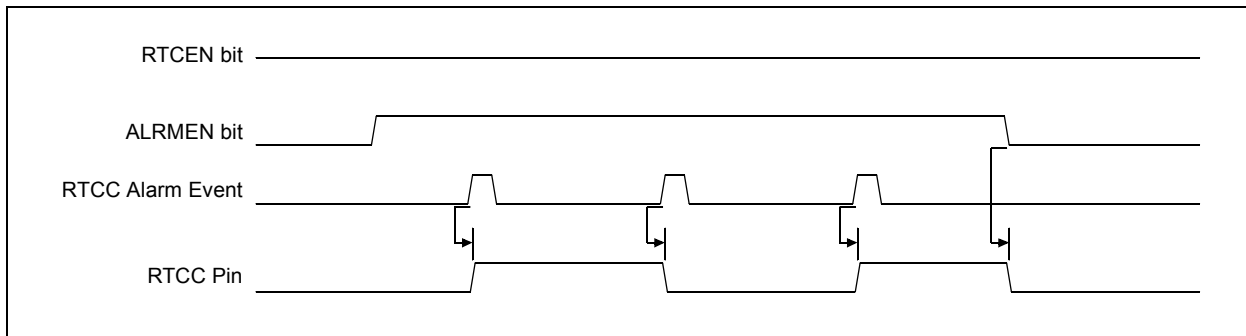
The alarm pulse output is completely synchronous with the RTCC clock and can be used as a trigger clock to other peripherals. This output is available on the RTCC pin. The output pulse is a clock with a 50% duty cycle and a frequency half that of the alarm event (see Figure 16-6).

The RTCC pin also can output the seconds clock. The user can select between the alarm pulse, generated by the RTCC module, or the seconds clock output.

The RTSECSEL ( $PADCFG1<1:0>$ ) bits select between these two outputs:

- Alarm pulse –  $RTSECSEL<1:0> = 00$
- Seconds clock –  $RTSECSEL<1:0> = 01$

**FIGURE 16-6: TIMER PULSE GENERATION**



## 16.4 Low-Power Modes

The timer and alarm can optionally continue to operate while in Sleep, Idle and even Deep Sleep mode. An alarm event can be used to wake-up the microcontroller from any of these Low-Power modes.

## 16.5 Reset

### 16.5.1 DEVICE RESET

When a device Reset occurs, the ALCFGRPT register is forced to its Reset state causing the alarm to be disabled (if enabled prior to the Reset). If the RTCC was enabled, it will continue to operate when a basic device Reset occurs.

### 16.5.2 POWER-ON RESET (POR)

The RTCCFG and ALMRPT registers are reset only on a POR. Once the device exits the POR state, the clock registers should be reloaded with the desired values.

The timer prescaler values can be reset only by writing to the SECONDS register. No device Reset can affect the prescalers.

# PIC18F46J11 FAMILY

## 16.6 Register Maps

Table 16-5, Table 16-6 and Table 16-7 summarize the registers associated with the RTCC module.

**TABLE 16-5: RTCC CONTROL REGISTERS**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
RTCCFG	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	0000
RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000
PADCFG1	—	—	—	—	—	RTSECSEL1	RTSECSEL0	PMPTTL	0000
ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	0000
ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	1111
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	0000
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	0000

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 44-pin devices.

**TABLE 16-6: RTCC VALUE REGISTERS**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
RTCVALH	RTCC Value Register Window High Byte, Based on RTCPTR<1:0>								xxxx
RTCVALL	RTCC Value Register Window Low Byte, Based on RTCPTR<1:0>								xxxx
RTCCFG	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	0000
ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	0000
ALRMVALH	Alarm Value Register Window High Byte, Based on ALRMPTR<1:0>								xxxx
ALRMVALL	Alarm Value Register Window Low Byte, Based on ALRMPTR<1:0>								xxxx

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 44-pin devices.

**TABLE 16-7: ALARM VALUE REGISTERS**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
ALRMVALH	Alarm Value Register Window High Byte, Based on ALRMPTR<1:0>								xxxx
ALRMVALL	Alarm Value Register Window Low Byte, Based on ALRMPTR<1:0>								xxxx
RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000
RTCVALH	RTCC Value Register Window High Byte, Based on RTCPTR<1:0>								xxxx
RTCVALL	RTCC Value Register Window Low Byte, Based on RTCPTR<1:0>								xxxx

**Legend:** — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:



## 17.0 ENHANCED CAPTURE/COMPARE/PWM (ECCP) MODULE

PIC18F46J11 family devices have two Enhanced Capture/Compare/PWM (ECCP) modules: ECCP1 and ECCP2. These modules contain 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. These ECCP modules are upward compatible with CCP

**Note:** Register and bit names referencing one of the two ECCP modules substitute an 'x' for the module number. For example, registers CCP1CON and CCP2CON, which have the same definitions, are called CCPxCON. Figures and diagrams use ECCP1-based names, but those names also apply to ECCP2, with a "2" replacing the illustration name's "1".  
When writing firmware, the "x" in register and bit names must be replaced with the appropriate module number.

ECCP1 and ECCP2 are implemented as standard CCP modules with enhanced PWM capabilities. These include:

- Provision for two or four output channels
- Output Steering modes
- Programmable polarity
- Programmable dead-band control
- Automatic shutdown and restart

The enhanced features are discussed in detail in **Section 17.5 "PWM (Enhanced Mode)"**.

**Note:** PxA, PxB, PxC and PxD are associated with the remappable pins (RPN).

# PIC18F46J11 FAMILY

## REGISTER 17-1: CCPxCON: ECCPx CONTROL (ACCESS FBAh/FB4h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

### Legend:

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

bit 7-6 **PxM<1:0>**: Enhanced PWM Output Configuration bits

If CCPxM<3:2> = 00, 01, 10:

xx = PxA assigned as capture/compare input/output; PxB, PxC and PxD assigned as port pins

If CCPxM<3:2> = 11:

00 = Single output: PxA, PxB, PxC and PxD controlled by steering (see **Section 17.5.7 “Pulse Steering Mode”**)

01 = Full-bridge output forward: PxD modulated; PxA active; PxB, PxC inactive

10 = Half-bridge output: PxA, PxB modulated with dead-band control; PxC and PxD assigned as port pins

11 = Full-bridge output reverse: PxB modulated; PxC active; PxA and PxD inactive

bit 5-4 **DCxB<1:0>**: 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 CCPRxL.

bit 3-0 **CCPxM<3:0>**: ECCPx Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCPx 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 4<sup>th</sup> rising edge

0111 = Capture mode, every 16<sup>th</sup> rising edge

1000 = Compare mode, initialize ECCPx pin low, set output on compare match (set CCPxIF)

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

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

1011 = Compare mode, trigger special event (ECCPx resets TMR1 or TMR3, starts A/D conversion, sets CCxIF bit)

1100 = PWM mode; PxA and PxC active-high; PxB and PxD active-high

1101 = PWM mode; PxA and PxC active-high; PxB and PxD active-low

1110 = PWM mode; PxA and PxC active-low; PxB and PxD active-high

1111 = PWM mode; PxA and PxC active-low; PxB and PxD active-low

In addition to the expanded range of modes available through the CCPxCON and ECCPxAS registers, the ECCP modules have two additional registers associated with Enhanced PWM operation and auto-shutdown features. They are:

- ECCPxDEL (Enhanced PWM Control)
- PSTRxCON (Pulse Steering Control)

## 17.1 ECCP Outputs and Configuration

The Enhanced CCP module may have up to four PWM outputs, depending on the selected operating mode. These outputs, designated PxA through PxD, are routed through the Peripheral Pin Select (PPS) module. Therefore, individual functions may be mapped to any of the remappable I/O pins, RPN. The outputs that are active depend on the ECCP operating mode selected. The pin assignments are summarized in Table 17-4.

To configure the I/O pins as PWM outputs, the proper PWM mode must be selected by setting the PxM<1:0> and CCPxM<3:0> bits. The appropriate TRIS direction bits for the port pins must also be set as outputs and the output functions need to be assigned to I/O pins in the PPS module. (For details on configuring the module, see **Section 9.7 “Peripheral Pin Select (PPS)”**.)

### 17.1.1 ECCP MODULE AND TIMER RESOURCES

The ECCP modules utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode.

**TABLE 17-1: ECCP MODE – TIMER RESOURCE**

ECCP Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3
PWM	Timer2 or Timer4

The assignment of a particular timer to a module is determined by the Timer-to-ECCP enable bits in the TCLKCON register (Register 12-3). The interactions between the two modules are depicted in Figure 17-1. Capture operations are designed to be used when the timer is configured for Synchronous Counter mode. Capture operations may not work as expected if the associated timer is configured for Asynchronous Counter mode.

## 17.2 Capture Mode

In Capture mode, the CCPRxH:CCPRxL register pair captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on the corresponding ECCPx pin. An event is defined as one of the following:

- Every falling edge
- Every rising edge
- Every 4<sup>th</sup> rising edge
- Every 16<sup>th</sup> rising edge

The event is selected by the mode select bits, CCPxM<3:0>, of the CCPxCON register. When a capture is made, the interrupt request flag bit, CCPxIF, is set; it must be cleared by software. If another capture occurs before the value in register CCPRx is read, the old captured value is overwritten by the new captured value.

### 17.2.1 ECCP PIN CONFIGURATION

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

Additionally, the ECCPx input function needs to be assigned to an I/O pin through the Peripheral Pin Select module. For details on setting up the remappable pins, see **Section 9.7 “Peripheral Pin Select (PPS)”**.

**Note:** If the ECCPx pin is configured as an output, a write to the port can cause a capture condition.

### 17.2.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (Timer1 and/or Timer3) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation may not work. The timer to be used with each ECCP module is selected in the TCLKCON register (Register 12-3).

### 17.2.3 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.

# PIC18F46J11 FAMILY

## 17.2.4 ECCP PRESCALER

There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCPxM<3:0>). Whenever the ECCP module is turned off, or Capture mode is disabled, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

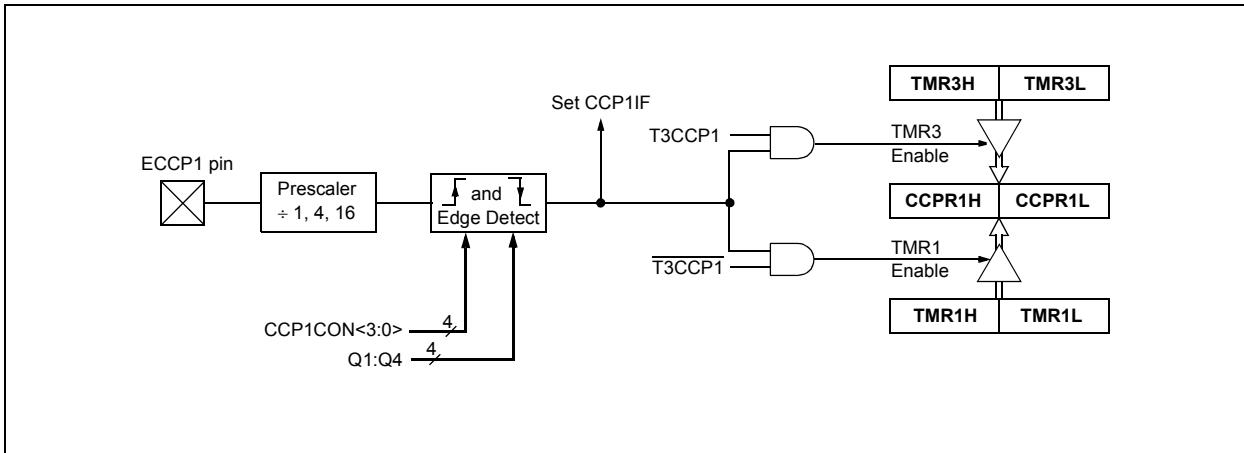
Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared; therefore, the first capture may be from a non-zero prescaler. Example 17-1 provides the

recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

### EXAMPLE 17-1: CHANGING BETWEEN CAPTURE PRESCALERS

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

FIGURE 17-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



## 17.3 Compare Mode

In Compare mode, the 16-bit CCPRx register value is constantly compared against either the TMR1 or TMR3 register pair value. When a match occurs, the ECCPx 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 (CCPxM<3:0>). At the same time, the interrupt flag bit, CCPxIF, is set.

### 17.3.1 ECCP PIN CONFIGURATION

Users must configure the ECCPx pin as an output by clearing the appropriate TRIS bit.

**Note:** Clearing the CCPxCON register will force the ECCPx compare output latch (depending on device configuration) to the default low level. This is not the PORTx I/O data latch.

### 17.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode or Synchronized Counter mode if the ECCP module is using the compare feature. In Asynchronous Counter mode, the compare operation will not work reliably.

### 17.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCPxM<3:0> = 1010), the ECCPx pin is not affected; only the CCPxIF interrupt flag is affected.

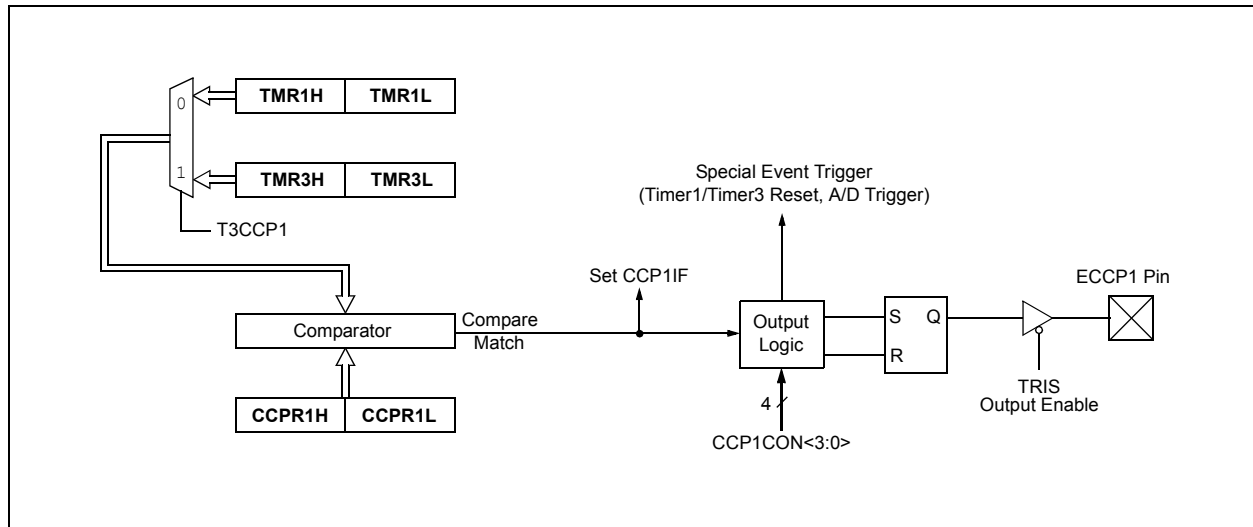
### 17.3.4 SPECIAL EVENT TRIGGER

The ECCP module is 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 (CCPxM<3:0> = 1011).

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 can also start an A/D conversion. In order to do this, the A/D converter must already be enabled.

**FIGURE 17-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

## 17.4 PWM Mode

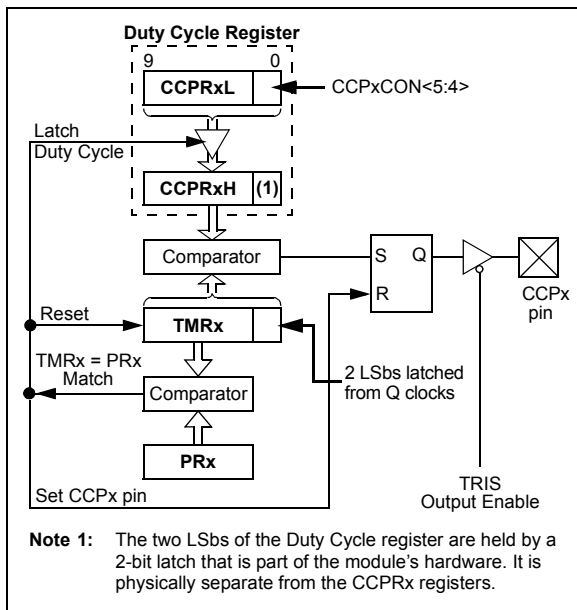
In Pulse-Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output.

**Note:** Clearing the CCPxCON register will force the output latch (depending on device configuration) to the default low level. This is not the LATx data latch.

Figure 17-3 shows a simplified block diagram of the CCP module in PWM mode.

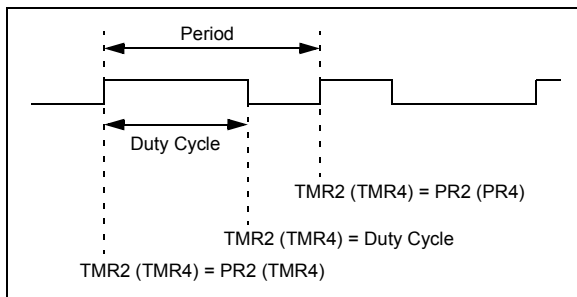
For a step-by-step procedure on how to set up a CCP module for PWM operation, see **Section 17.4.3 “Setup for PWM Operation”**.

**FIGURE 17-3: SIMPLIFIED PWM BLOCK DIAGRAM**



A PWM output (Figure 17-4) 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 17-4: PWM OUTPUT**



### 17.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 (PR4) register. The PWM period can be calculated using Equation 17-1:

**EQUATION 17-1:**

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

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

When TMR2 (TMR4) is equal to PR2 (PR4), the following three events occur on the next increment cycle:

- TMR2 (TMR4) 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 and Timer 4 postscalers (see **Section 13.0 “Timer2 Module”** and **Section 15.0 “Timer4 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.

### 17.4.2 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>. Equation 17-2 is used to calculate the PWM duty cycle in time.

**EQUATION 17-2:**

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

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 (PR4) and TMR2 (TMR4) occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

# PIC18F46J11 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 (TMR4), concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 (TMR4) prescaler, the CCPx pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by Equation 17-3:

**EQUATION 17-3:**

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

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

## 17.4.3 SETUP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 (PR4) 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 (TMR4) prescale value, then enable Timer2 (Timer4) by writing to T2CON (T4CON).
5. Configure the CCPx module for PWM operation.

**TABLE 17-2: 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

# PIC18F46J11 FAMILY

**TABLE 17-3: REGISTERS ASSOCIATED WITH PWM, TIMER2 AND TIMER4**

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	63
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	64
PIR1	PMPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMP1P	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	65
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	65
TMR2 <sup>(1)</sup>	Timer2 Register								64
PR2 <sup>(1)</sup>	Timer2 Period Register								64
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	64
TMR4	Timer4 Register								67
PR4 <sup>(1)</sup>	Timer4 Period Register								67
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	67
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								67
CCPR4H	Capture/Compare/PWM Register 4 High Byte								67
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								67
CCPR5H	Capture/Compare/PWM Register 5 High Byte								67
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	67
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	67
ODCON1 <sup>(2)</sup>	—	—	—	CCP5OD	CCP4OD	ECCP3OD	ECCP2OD	ECCP1OD	68

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by PWM, Timer2 or Timer4.

**Note 1:** Default (legacy) SFR at this address, available when WDTCON<4> = 0.

**2:** Configuration SFR, overlaps with default SFR at this address; available only when WDTCON<4> = 1.



## 17.5 PWM (Enhanced Mode)

The Enhanced PWM mode can generate a PWM signal on up to four different output pins with up to 10 bits of resolution. It can do this through four different PWM Output modes:

- Single PWM
- Half-Bridge PWM
- Full-Bridge PWM, Forward mode
- Full-Bridge PWM, Reverse mode

To select an Enhanced PWM mode, the PxM bits of the CCPxCON register must be set appropriately.

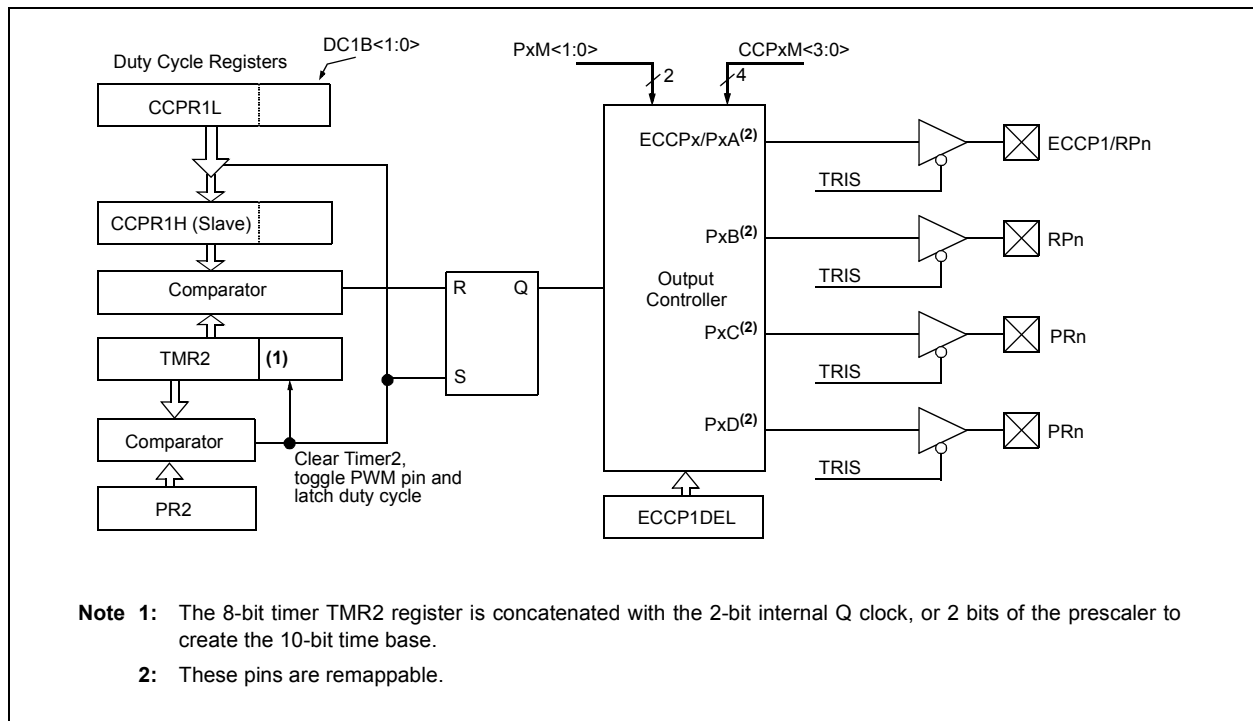
The PWM outputs are multiplexed with I/O pins and are designated: PxA, PxB, PxC and PxD. The polarity of the PWM pins is configurable and is selected by setting the CCPxM bits in the CCPxCON register appropriately.

Table 17-1 provides the pin assignments for each Enhanced PWM mode.

Figure 17-5 provides an example of a simplified block diagram of the Enhanced PWM module.

**Note:** To prevent the generation of an incomplete waveform when the PWM is first enabled, the ECCP module waits until the start of a new PWM period before generating a PWM signal.

**FIGURE 17-5: EXAMPLE SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODE**



**Note 1:** The TRIS register value for each PWM output must be configured appropriately.

**Note 2:** Any pin not used by an Enhanced PWM mode is available for alternate pin functions.

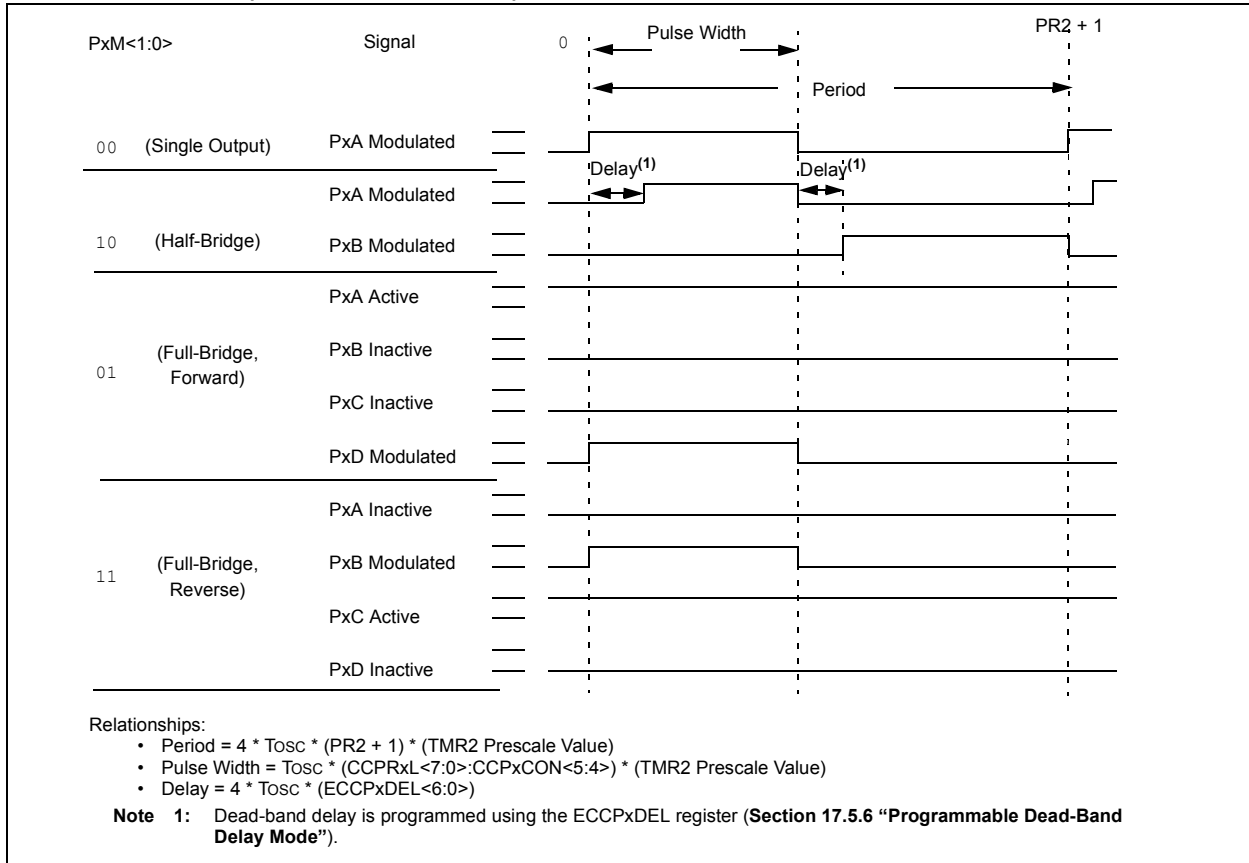
# PIC18F46J11 FAMILY

**TABLE 17-4: EXAMPLE PIN ASSIGNMENTS FOR VARIOUS PWM ENHANCED MODES**

ECCP Mode	PxM<1:0>	PxA	PxB	PxC	PxD
Single	00	Yes <sup>(1)</sup>	Yes <sup>(1)</sup>	Yes <sup>(1)</sup>	Yes <sup>(1)</sup>
Half-Bridge	10	Yes	Yes	No	No
Full-Bridge, Forward	01	Yes	Yes	Yes	Yes
Full-Bridge, Reverse	11	Yes	Yes	Yes	Yes

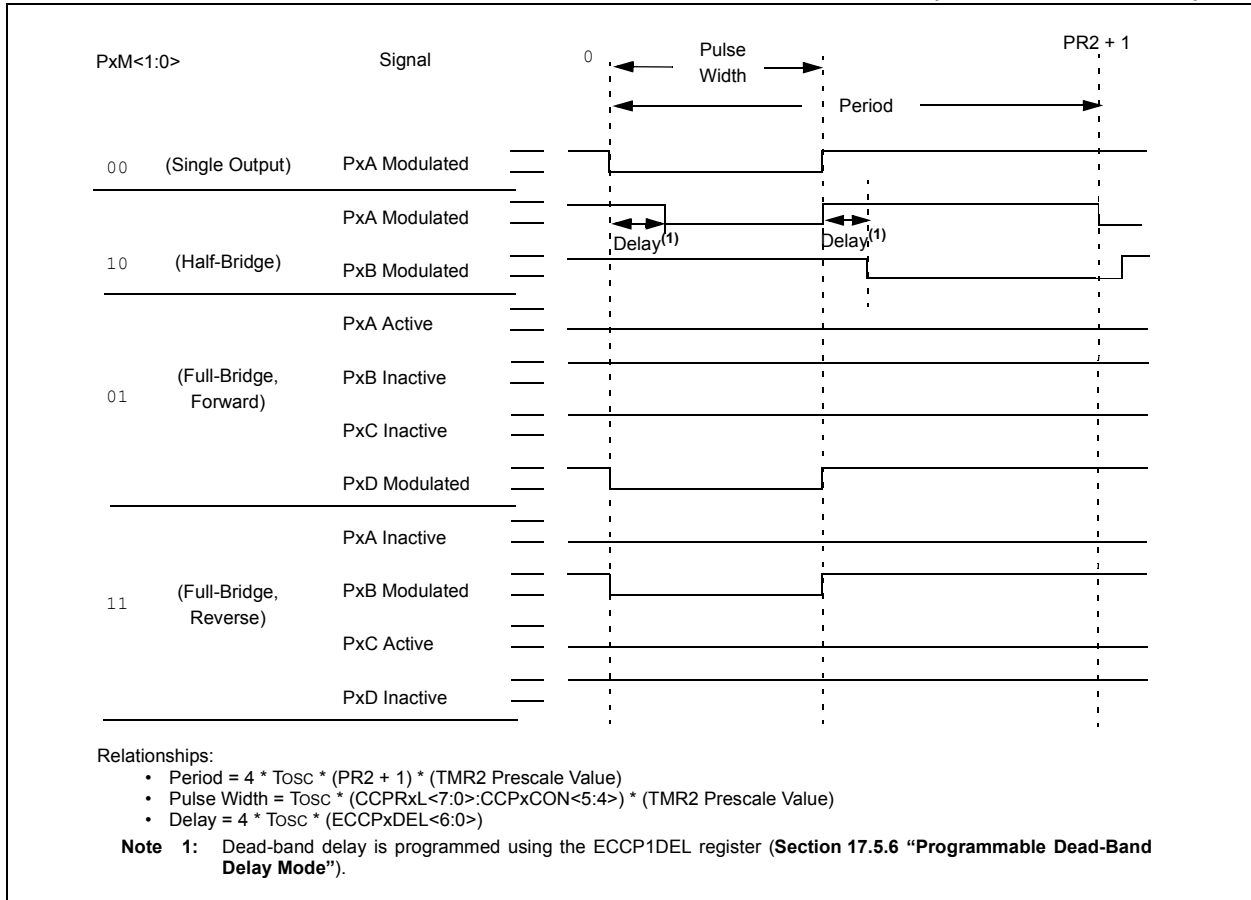
**Note 1:** Outputs are enabled by pulse steering in Single mode (see Register 17-4).

**FIGURE 17-6: EXAMPLE PWM (ENHANCED MODE) OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)**



# PIC18F46J11 FAMILY

**FIGURE 17-7: EXAMPLE ENHANCED PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)**



# PIC18F46J11 FAMILY

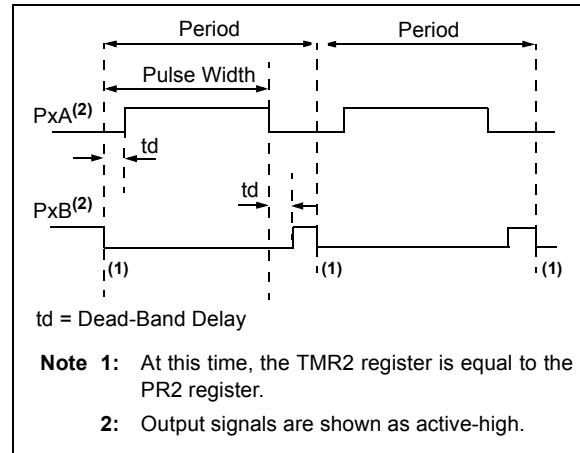
## 17.5.1 HALF-BRIDGE MODE

In Half-Bridge mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the PxA pin, while the complementary PWM output signal is output on the PxB pin (see Figure 17-8). This mode can be used for half-bridge applications, as shown in Figure 17-9, or for full-bridge applications, where four power switches are being modulated with two PWM signals.

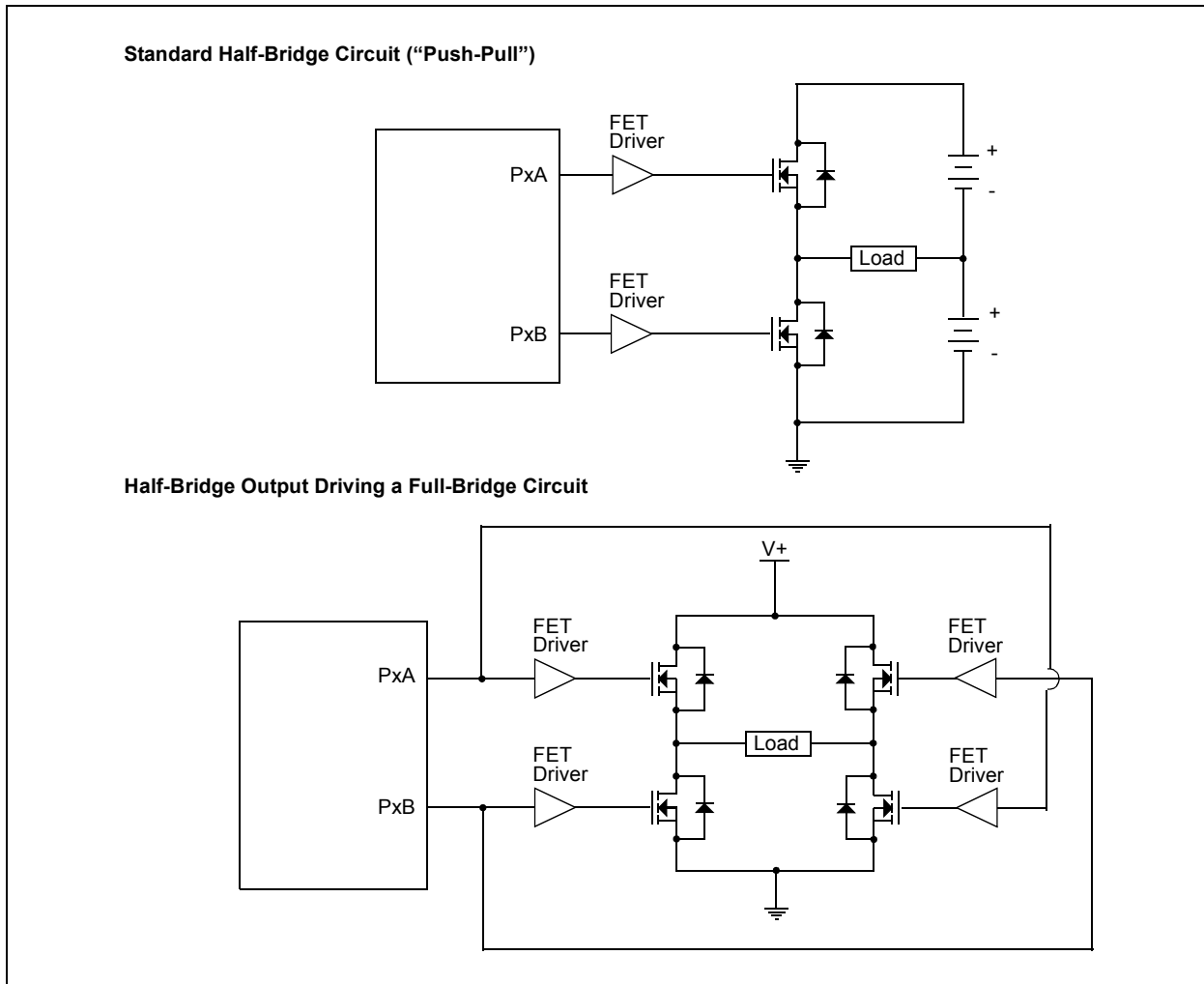
In Half-Bridge mode, the programmable dead-band delay can be used to prevent shoot-through current in half-bridge power devices. The value of the PxDC<6:0> bits of the ECCPxDEL register sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. See **Section 17.5.6 “Programmable Dead-Band Delay Mode”** for more details of the dead-band delay operations.

Since the PxA and PxB outputs are multiplexed with the PORT data latches, the associated TRIS bits must be cleared to configure PxA and PxB as outputs.

**FIGURE 17-8: EXAMPLE OF HALF-BRIDGE PWM OUTPUT**



**FIGURE 17-9: EXAMPLE OF HALF-BRIDGE APPLICATIONS**



## 17.5.2 FULL-BRIDGE MODE

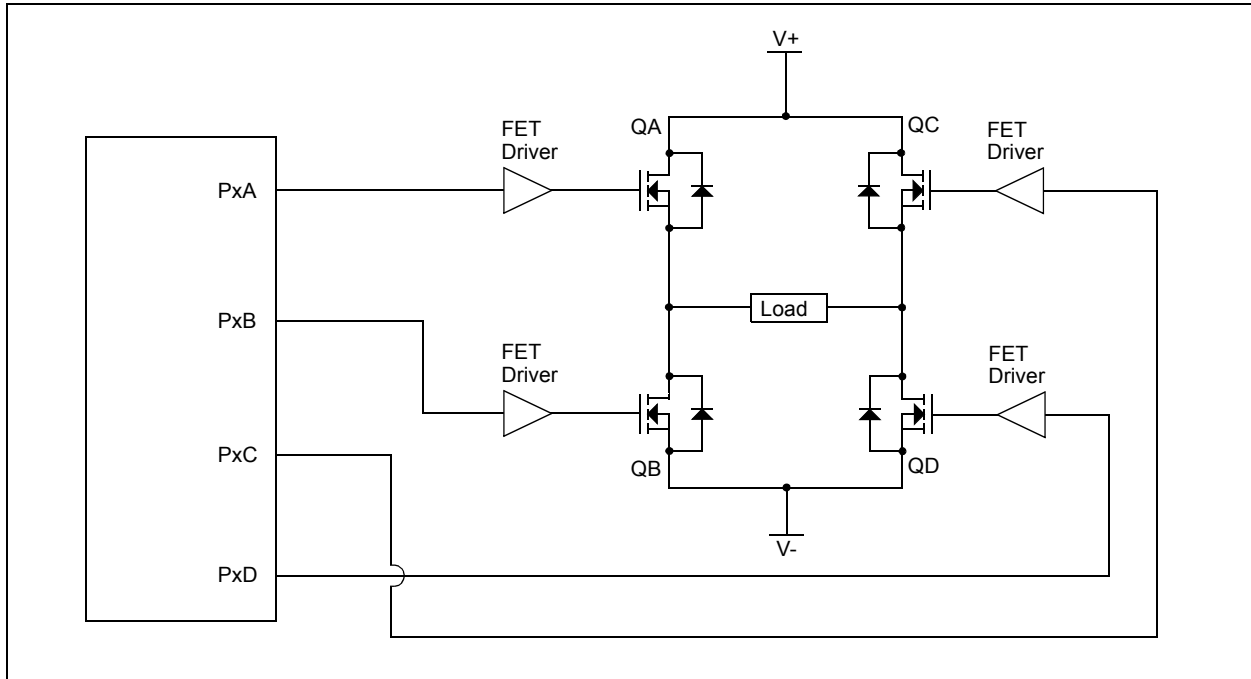
In Full-Bridge mode, all four pins are used as outputs. An example of a full-bridge application is provided in Figure 17-10.

In the Forward mode, the PxA pin is driven to its active state, the PxD pin is modulated, while the PxB and PxC pins will be driven to their inactive state as provided in Figure 17-11.

In the Reverse mode, the PxC pin is driven to its active state, the PxB pin is modulated, while the PxA and PxD pins will be driven to their inactive state as provided in Figure 17-11.

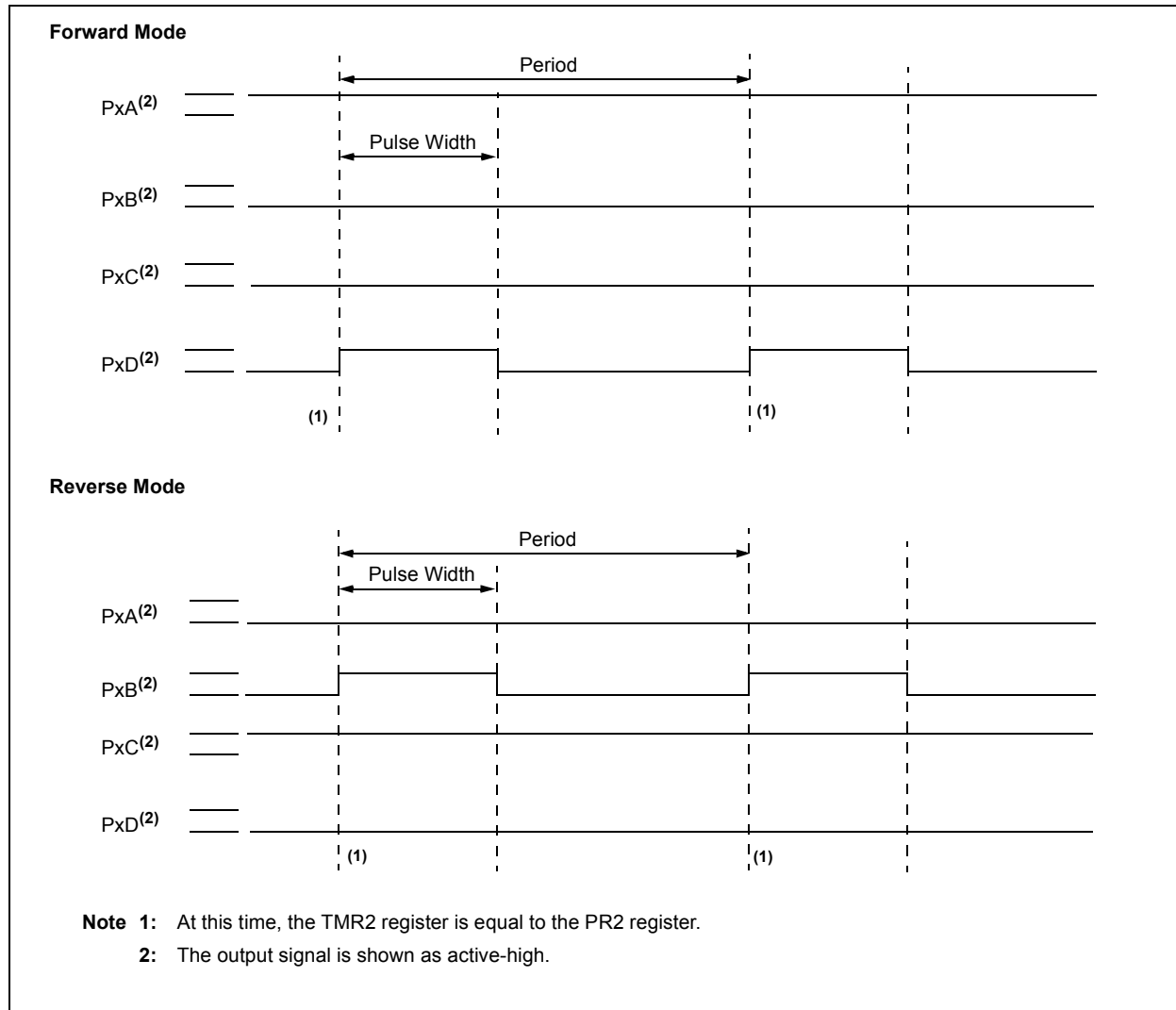
The PxA, PxB, PxC and PxD outputs are multiplexed with the PORT data latches. The associated TRIS bits must be cleared to configure the PxA, PxB, PxC and PxD pins as outputs.

**FIGURE 17-10: EXAMPLE OF FULL-BRIDGE APPLICATION**



# PIC18F46J11 FAMILY

FIGURE 17-11: EXAMPLE OF FULL-BRIDGE PWM OUTPUT



## 17.5.2.1 Direction Change in Full-Bridge Mode

In the Full-Bridge mode, the Pxm1 bit in the CCPxCON register allows users to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will change to the new direction on the next PWM cycle.

A direction change is initiated in software by changing the Pxm1 bit of the CCPxCON register. The following sequence occurs prior to the end of the current PWM period:

- The modulated outputs (PxB and PxD) are placed in their inactive state.
- The associated unmodulated outputs (PxA and PxC) are switched to drive in the opposite direction.
- PWM modulation resumes at the beginning of the next period.

See Figure 17-12 for an illustration of this sequence.

The Full-Bridge mode does not provide a dead-band delay. As one output is modulated at a time, a dead-band delay is generally not required. There is a situation where a dead-band delay is 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.

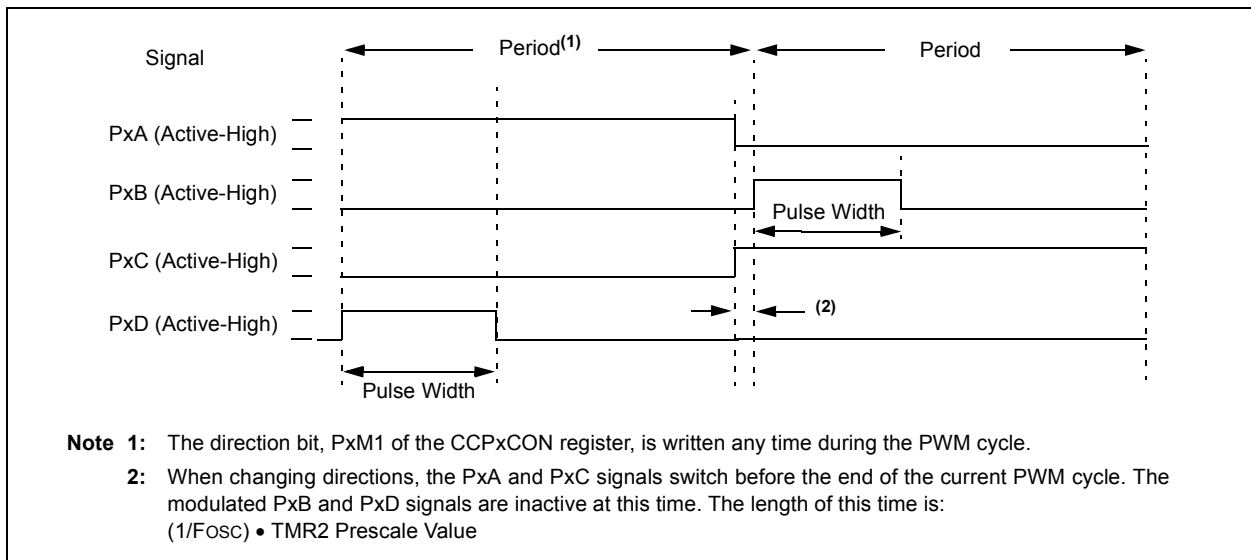
Figure 17-13 shows an example of the PWM direction changing from forward to reverse, at a near 100% duty cycle. In this example, at time, t1, the PxA and PxD outputs become inactive, while the PxC output becomes active. Since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current will flow through power devices, QC and QD (see Figure 17-10), 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, two possible solutions for eliminating the shoot-through current are:

1. Reduce PWM duty cycle for one PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

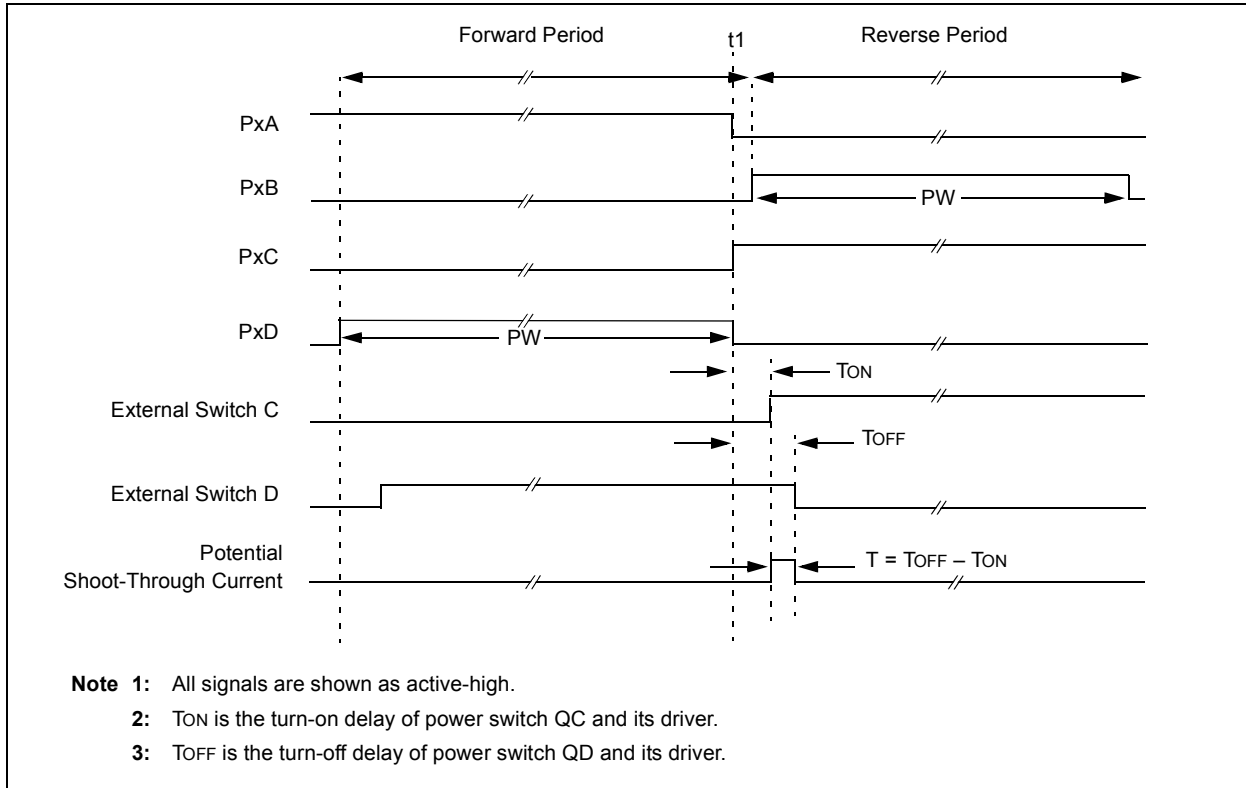
Other options to prevent shoot-through current may exist.

**FIGURE 17-12: EXAMPLE OF PWM DIRECTION CHANGE**



# PIC18F46J11 FAMILY

**FIGURE 17-13: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE**



## 17.5.3 START-UP CONSIDERATIONS

When any PWM mode is used, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins.

**Note:** 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  $CCPxM<1:0>$  bits of the  $CCPxCON$  register allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (PxA/PxC and PxB/PxD). The PWM output

polarities must be selected before the PWM pin output drivers are enabled. Changing the polarity configuration while the PWM pin output drivers are enabled is not recommended since it may result in damage to the application circuits.

The PxA, PxB, PxC and PxD output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pin output drivers at the same time as the Enhanced PWM modes may cause damage to the application circuit. The Enhanced PWM modes must be enabled in the proper Output mode and complete a full PWM cycle before enabling the PWM pin output drivers. The completion of a full PWM cycle is indicated by the TMR2IF or TMR4IF bit of the PIR1 or PIR3 register being set as the second PWM period begins.



# PIC18F46J11 FAMILY

## 17.5.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCPxAS<2:0> bits of the ECCPAS register. A shutdown event may be generated by:

- A logic '0' on the pin that is assigned the FLT0 input function
- Comparator C1
- Comparator C2
- Setting the ECCPxASE bit in firmware

A shutdown condition is indicated by the ECCPxASE (Auto-Shutdown Event Status) bit of the ECCPxAS register. If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

When a shutdown event occurs, two things happen:

The ECCPxASE bit is set to '1'. The ECCPxASE will remain set until cleared in firmware or an auto-restart occurs (see **Section 17.5.5 "Auto-Restart Mode"**).

The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs [PxA/PxC] and [PxB/PxD]. The state of each pin pair is determined by the PSSxAC and PSSxBD bits of the ECCPxAS register. Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

### REGISTER 17-2: ECCPxAS: ECCPx AUTO-SHUTDOWN CONTROL REGISTER (ACCESS FBEh/FB8h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
bit 7							bit 0

#### Legend:

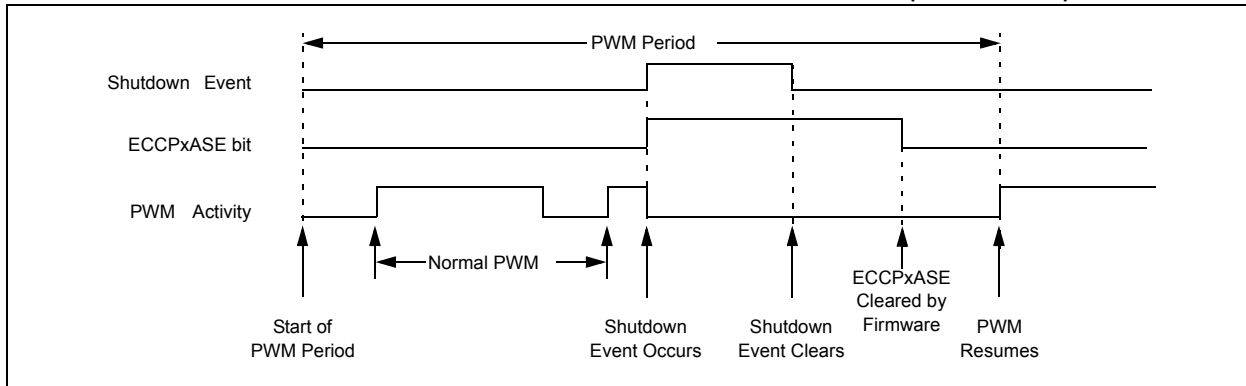
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

bit 7	<b>ECCPxASE:</b> ECCP Auto-Shutdown Event Status bit 1 = A shutdown event has occurred; ECCP outputs are in a shutdown state 0 = ECCP outputs are operating
bit 6-4	<b>ECCPxAS&lt;2:0&gt;:</b> ECCP Auto-Shutdown Source Select bits 000 = Auto-shutdown is disabled 001 = Comparator C1OUT output is high 010 = Comparator C2OUT output is high 011 = Either Comparator C1OUT or C2OUT is high 100 = VIL on FLT0 pin 101 = VIL on FLT0 pin or Comparator C1OUT output is high 110 = VIL on FLT0 pin or Comparator C2OUT output is high 111 = VIL on FLT0 pin or Comparator C1OUT or Comparator C2OUT is high
bit 3-2	<b>PSSxAC&lt;1:0&gt;:</b> Pins PxA and PxC Shutdown State Control bits 00 = Drive pins PxA and PxC to '0' 01 = Drive pins PxA and PxC to '1' 1x = Pins PxA and PxC tri-state
bit 1-0	<b>PSSxBD&lt;1:0&gt;:</b> Pins PxB and PxD Shutdown State Control bits 00 = Drive pins PxB and PxD to '0' 01 = Drive pins PxB and PxD to '1' 1x = Pins PxB and PxD tri-state

- Note 1:** The auto-shutdown condition is a level-based signal, not an edge-based signal. As long as the level is present, the auto-shutdown will persist.
- 2:** Writing to the ECCPxASE bit is disabled while an auto-shutdown condition persists.
- 3:** Once the auto-shutdown condition has been removed and the PWM restarted (either through firmware or auto-restart), the PWM signal will always restart at the beginning of the next PWM period.

# PIC18F46J11 FAMILY

**FIGURE 17-14: PWM AUTO-SHUTDOWN WITH FIRMWARE RESTART (PxRSEN = 0)**



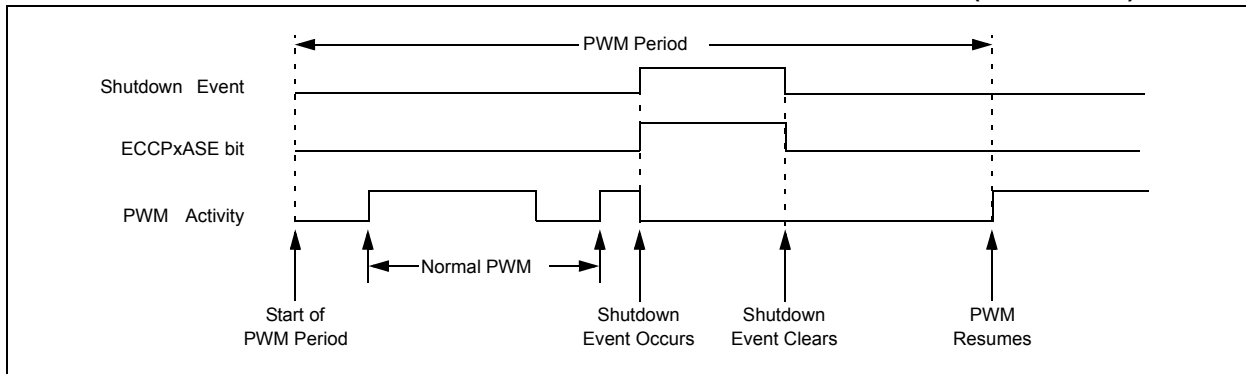
## 17.5.5 AUTO-RESTART MODE

The Enhanced PWM can be configured to automatically restart the PWM signal once the auto-shutdown condition has been removed. Auto-restart is enabled by setting the PxRSEN bit in the ECCPxDEL register.

If auto-restart is enabled, the ECCPxASE bit will remain set as long as the auto-shutdown condition is active. When the auto-shutdown condition is removed, the ECCPxASE bit will be cleared via hardware and normal operation will resume.

The module will wait until the next PWM period begins, however, before re-enabling the output pin. This behavior allows the auto-shutdown with auto-restart features to be used in applications based on current mode PWM control.

**FIGURE 17-15: PWM AUTO-SHUTDOWN WITH AUTO-RESTART ENABLED (PxRSEN = 1)**

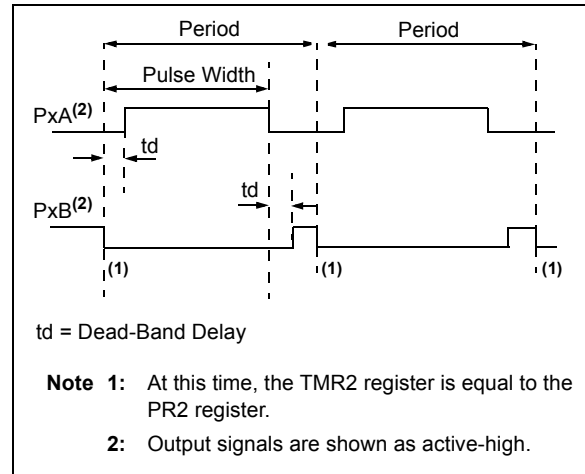


## 17.5.6 PROGRAMMABLE DEAD-BAND DELAY MODE

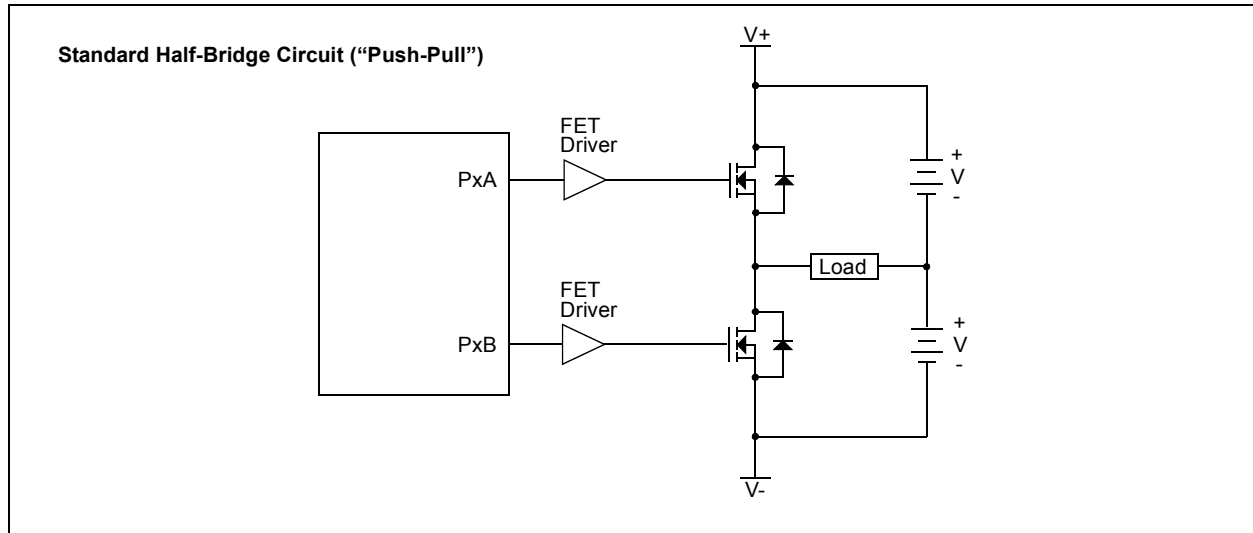
In half-bridge applications, where all power switches are modulated at the PWM frequency, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period until one switch completely turns off. During this brief interval, a very high current (shoot-through current) will 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 Half-Bridge mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 17-16 for illustration. The lower seven bits of the associated ECCPxDEL register (Register 17-3) sets the delay period in terms of microcontroller instruction cycles ( $T_{cy}$  or  $4 T_{osc}$ ).

**FIGURE 17-16: EXAMPLE OF HALF-BRIDGE PWM OUTPUT**



**FIGURE 17-17: EXAMPLE OF HALF-BRIDGE APPLICATIONS**



# PIC18F46J11 FAMILY

## REGISTER 17-3: ECCPxDEL: ENHANCED PWM CONTROL REGISTER (ACCESS FBDh/FB7h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
bit 7							bit 0

### Legend:

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

- bit 7      **PxRSEN:** PWM Restart Enable bit  
 1 = Upon auto-shutdown, the ECCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically  
 0 = Upon auto-shutdown, ECCPxASE must be cleared by software to restart the PWM
- bit 6-0      **PxDC<6:0>:** PWM Delay Count bits  
 PxDCn = Number of Fosc/4 (4 \* Tosc) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active.

### 17.5.7 PULSE STEERING MODE

In Single Output mode, pulse steering allows any of the PWM pins to be the modulated signal. Additionally, the same PWM signal can simultaneously be available on multiple pins.

Once the Single Output mode is selected (CCPxM<3:2> = 11 and PxM<1:0> = 00 of the CCPxCON register), the user firmware can bring out the same PWM signal to one, two, three or four output pins by setting the appropriate STR<D:A> bits of the PSTRxCON register, as provided in Table 17-4.

While the PWM Steering mode is active, the CCPxM<1:0> bits of the CCPxCON register select the PWM output polarity for the Px<D:A> pins.

The PWM auto-shutdown operation also applies to PWM Steering mode as described in **Section 17.5.4 “Enhanced PWM Auto-shutdown mode”**. An auto-shutdown event will only affect pins that have PWM outputs enabled.

**Note:** The associated TRIS bits must be set to output ('0') to enable the pin output driver in order to see the PWM signal on the pin.

# PIC18F46J11 FAMILY

## REGISTER 17-4: PSTRxCON: PULSE STEERING CONTROL (ACCESS FBFh/FB9h)<sup>(1)</sup>

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
bit 7							bit 0

### Legend:

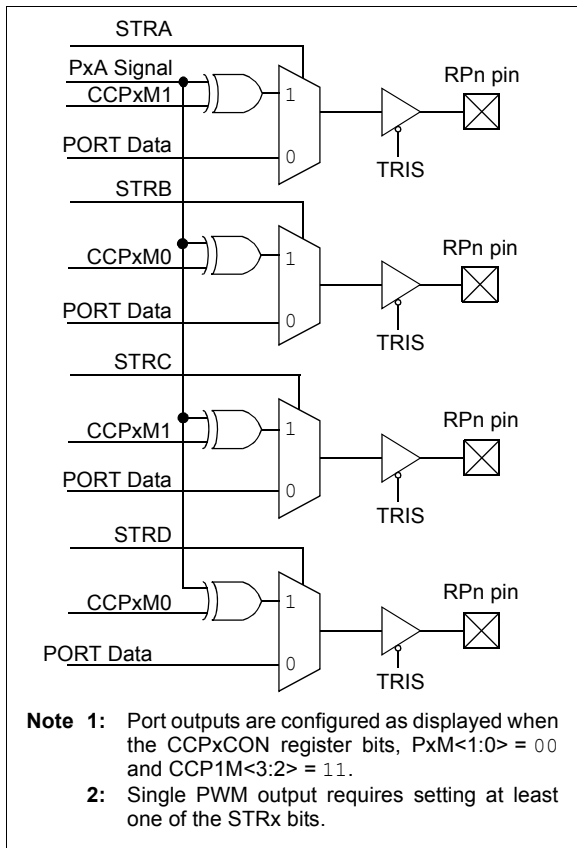
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared      x = Bit is unknown

- bit 7-6      **CMPL<1:0>**: Complementary Mode Output Assignment Steering Sync bits  
 1 = Modulated output pin toggles between PxA and PxB for each period  
 0 = Complementary output assignment disabled; STRD:STRA bits used to determine Steering mode
- bit 5      **Unimplemented**: Read as '0'
- bit 4      **STRSYNC**: Steering Sync bit  
 1 = Output steering update occurs on next PWM period  
 0 = Output steering update occurs at the beginning of the instruction cycle boundary
- bit 3      **STRD**: Steering Enable bit D  
 1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>  
 0 = PxD pin is assigned to port pin
- bit 2      **STRC**: Steering Enable bit C  
 1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>  
 0 = PxC pin is assigned to port pin
- bit 1      **STRB**: Steering Enable bit B  
 1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>  
 0 = PxB pin is assigned to port pin
- bit 0      **STRA**: Steering Enable bit A  
 1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>  
 0 = PxA pin is assigned to port pin

**Note 1:** The PWM Steering mode is available only when the CCPxCON register bits, CCPxM<3:2> = 11 and Pxm<1:0> = 00.

# PIC18F46J11 FAMILY

**FIGURE 17-18: SIMPLIFIED STEERING BLOCK DIAGRAM**



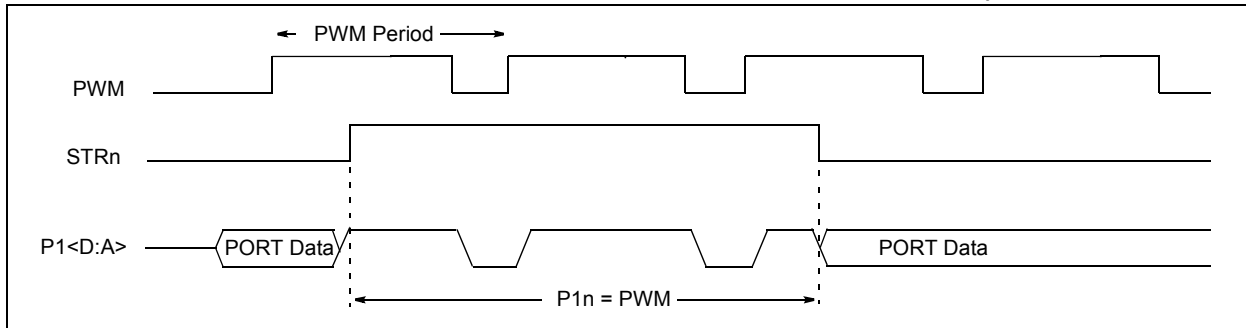
## 17.5.7.1 Steering Synchronization

The STRSYNC bit of the PSTRxCON register gives the user two selections of when the steering event will happen. When the STRSYNC bit is '0', the steering event will happen at the end of the instruction that writes to the PSTRxCON register. In this case, the output signal at the Px<D:A> pins may be an incomplete PWM waveform. This operation is useful when the user firmware needs to immediately remove a PWM signal from the pin.

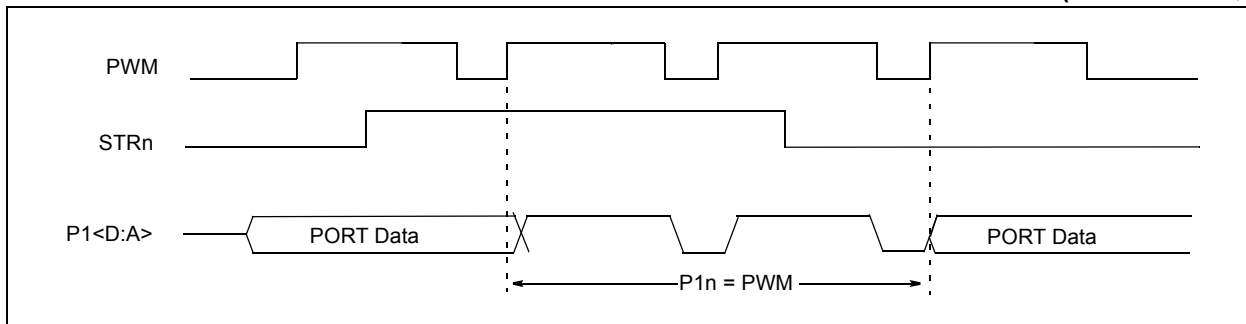
When the STRSYNC bit is '1', the effective steering update will happen at the beginning of the next PWM period. In this case, steering on/off the PWM output will always produce a complete PWM waveform.

Figures 17-19 and 17-20 illustrate the timing diagrams of the PWM steering depending on the STRSYNC setting.

**FIGURE 17-19: EXAMPLE OF STEERING EVENT AT END OF INSTRUCTION (STRSYNC = 0)**



**FIGURE 17-20: EXAMPLE OF STEERING EVENT AT BEGINNING OF INSTRUCTION (STRSYNC = 1)**



# PIC18F46J11 FAMILY

## 17.5.8 OPERATION IN POWER-MANAGED MODES

In Sleep mode, all clock sources are disabled. Timer2 will not increment and the state of the module will not change. If the ECCPx 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 HFINTOSC and the postscaler may not be stable immediately.

In PRI\_IDLE mode, the primary clock will continue to clock the ECCPx module without change.

### 17.5.8.1 Operation with Fail-Safe Clock Monitor (FSCM)

If the Fail-Safe Clock Monitor (FSCM) is enabled, a clock failure will force the device into the power-managed RC\_RUN mode and the OSCFIF bit of

the PIR2 register will be set. The ECCPx will then be clocked from the internal oscillator clock source, which may have a different clock frequency than the primary clock.

## 17.5.9 EFFECTS OF A RESET

Both Power-on Reset and subsequent Resets will force all ports to Input mode and the ECCP registers to their Reset states.

This forces the ECCP module to reset to a state compatible with previous, non-enhanced ECCP modules used on other PIC18 and PIC16 devices.

**TABLE 17-5: REGISTERS ASSOCIATED WITH ECCP1 MODULE AND TIMER1 TO TIMER3**

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	RABIE	TMR0IF	INT0IF	RABIF	63
RCON	IPEN	SBOREN	—	RI	TO	PD	POR	BOR	64
PIR1	PMPIF <sup>(1)</sup>	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPIP <sup>(1)</sup>	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	65
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	65
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	65
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	65
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	66
TMR1L	Timer1 Register Low Byte								64
TMR1H	Timer1 Register High Byte								64
TCLKCON	—	—	—	T1RUN	—	—	T3CCP2	T3CCP1	87
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	T1SYN $\bar{C}$	RD16	TMR1ON	64
TMR2	Timer2 Register								64
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	64
PR2	Timer2 Period Register								64
TMR3L	Timer3 Register Low Byte								67
TMR3H	Timer3 Register High Byte								67
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	—	T3SYN $\bar{C}$	RD16	TMR3ON	67
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								65
CCPR1H	Capture/Compare/PWM Register 1 High Byte								65
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	65
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	64
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	65

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

**Note 1:** These bits are only available on 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:



## 18.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices include serial EEPROMs, shift registers, display drivers and A/D Converters.

### 18.1 Master SSP (MSSP) Module Overview

The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I<sup>2</sup>C™)
  - Full Master mode
  - Slave mode (with general address call)

The I<sup>2</sup>C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode with 5-bit and 7-bit address masking (with address masking for both 10-bit and 7-bit addressing)

All members of the PIC18F46J11 family have two MSSP modules, designated as MSSP1 and MSSP2. The modules operate independently:

- PIC18F4XJ11 devices – Both modules can be configured for either I<sup>2</sup>C or SPI communication
- PIC18F2XJ11 devices:
  - MSSP1 can be used for either I<sup>2</sup>C or SPI communication
  - MSSP2 can be used only for SPI communication

All of the MSSP1 module-related SPI and I<sup>2</sup>C I/O functions are hard-mapped to specific I/O pins.

For MSSP2 functions:

- SPI I/O functions (SDO2, SDI2, SCK2 and  $\overline{SS2}$ ) are all routed through the Peripheral Pin Select (PPS) module.

These functions may be configured to use any of the RPN remappable pins, as described in **Section 9.7 “Peripheral Pin Select (PPS)”**.

- I<sup>2</sup>C functions (SCL2 and SDA2) have fixed pin locations.

On all PIC18F46J11 family devices, the SPI DMA capability can only be used in conjunction with MSSP2. The SPI DMA feature is described in **Section 18.4 “SPI DMA Module”**.

**Note:** Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names and module I/O signals use the generic designator ‘x’ to indicate the use of a numeral to distinguish a particular module when required. Control bit names are not individuated.



# PIC18F46J11 FAMILY

## 18.3.1 REGISTERS

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

- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Status Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSPx 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 six 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 18-1: SSPxSTAT: MSSPx STATUS REGISTER – SPI MODE (ACCESS FC7h/F73h)**

R/W-1	R/W-1	R-1	R-1	R-1	R-1	R-1	R-1
SMP	CKE <sup>(1)</sup>	D/A	P	S	R/W	UA	BF
bit 7							bit 0

**Legend:**

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

- bit 7 **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<sup>(1)</sup>  
 1 = Transmit occurs on transition from active to Idle clock state  
 0 = Transmit occurs on transition from Idle to active clock state
- bit 5 **D/A:** Data/Address bit  
 Used in I<sup>2</sup>C™ mode only.
- bit 4 **P:** Stop bit  
 Used in I<sup>2</sup>C mode only; this bit is cleared when the MSSP module is disabled, SSPEN is cleared.
- bit 3 **S:** Start bit  
 Used in I<sup>2</sup>C mode only.
- bit 2 **R/W:** Read/Write Information bit  
 Used in I<sup>2</sup>C mode only.
- bit 1 **UA:** Update Address bit  
 Used in I<sup>2</sup>C mode only.
- bit 0 **BF:** Buffer Full Status bit  
 1 = Receive complete, SSPxBUF is full  
 0 = Receive not complete, SSPxBUF is empty

**Note 1:** Polarity of clock state is set by the CKP bit (SSPxCON1<4>).

# PIC18F46J11 FAMILY

## REGISTER 18-2: SSPxCON1: MSSPx CONTROL REGISTER 1 – SPI MODE (ACCESS FC6H/F72h)

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 <sup>(1)</sup>	SSPEN <sup>(2)</sup>	CKP	SSPM3 <sup>(3)</sup>	SSPM2 <sup>(3)</sup>	SSPM1 <sup>(3)</sup>	SSPM0 <sup>(3)</sup>
bit 7							bit 0

### Legend:

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

- bit 7      **WCOL:** Write Collision Detect bit  
 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<sup>(1)</sup>  
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
- bit 5      **SSPEN:** Master Synchronous Serial Port Enable bit<sup>(2)</sup>  
 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
- 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    **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits<sup>(3)</sup>  
 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 1:** In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
- 2:** When enabled, this pin must be properly configured as input or output.
- 3:** Bit combinations not specifically listed here, are either reserved or implemented in I<sup>2</sup>C™ mode only.

## 18.3.2 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 module 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 (BF) detect bit (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.

**Note:** When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of transfer data is written to the SSPxBUF. Application software should follow this process even when the current contents of SSPxBUF are not important.

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. 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 18-1 provides the loading of the SSPxBUF (SSPxSR) 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.

## 18.3.3 OPEN-DRAIN OUTPUT OPTION

The drivers for the SDOx output and SCKx clock pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, provided the SDOx or SCKx pin is not multiplexed with an ANx analog function. This allows the output to communicate with external circuits without the need for additional level shifters. For more information, see **Section 9.1.4 “Open-Drain Outputs”**.

The open-drain output option is controlled by the SPI2OD and SPI1OD bits (ODCON3<1:0>). Setting an SPIxOD bit configures both SDOx and SCKx pins for the corresponding open-drain operation.

### EXAMPLE 18-1: 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
    
```

# PIC18F46J11 FAMILY

## 18.3.4 ENABLING SPI I/O

To enable the serial port, MSSP Enable bit, SSPEN (SSPxCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPxCON1 registers and then set the SSPEN bit. This configures the SDIx, SDOx, SCKx and SSx pins as serial port pins. For the pins to behave as the serial port function, the appropriate TRIS bits, ANCON/PCFG bits and Peripheral Pin Select registers (if using MSSP2) should be correctly initialized prior to setting the SSPEN bit.

A typical SPI serial port initialization process follows:

- Initialize ODCON3 register (optional open-drain output control)
- Initialize remappable pin functions (if using MSSP2, see **Section 9.7 “Peripheral Pin Select (PPS)”**)
- Initialize SCKx LAT value to desired Idle SCK level (if master device)
- Initialize SCKx ANCON/PCFG bit (if Slave mode and multiplexed with ANx function)
- Initialize SCKx TRIS bit as output (Master mode) or input (Slave mode)
- Initialize SDIx ANCON/PCFG bit (if SDIx is multiplexed with ANx function)
- Initialize SDIx TRIS bit
- Initialize SSx ANCON/PCFG bit (if Slave mode and multiplexed with ANx function)
- Initialize SSx TRIS bit (Slave modes)
- Initialize SDOx TRIS bit
- Initialize SSPxSTAT register
- Initialize SSPxCON1 register
- Set SSPEN bit to enable the module

Any MSSP1 serial port function that is not desired may be overridden by programming the corresponding Data Direction (TRIS) register to the opposite value. If individual MSSP2 serial port functions will not be used, they may be left unmapped.

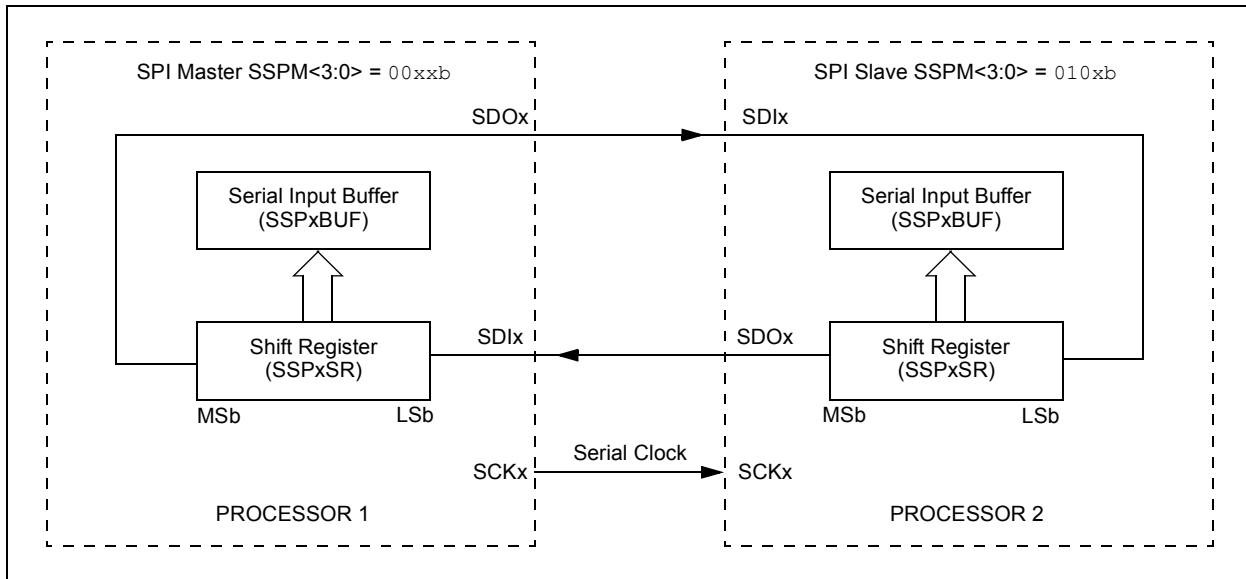
**Note:** When MSSP2 is used in SPI Master mode, the SCK2 function must be configured as both an output and input in the PPS module. SCK2 must be initialized as an output pin (by writing 0x0A to one of the RPORx registers). Additionally, SCK2IN must also be mapped to the same pin, by initializing the RPINR22 register. Failure to initialize SCK2/SCK2IN as both output and input will prevent the module from receiving data on the SDI2 pin, as the module uses the SCK2IN signal to latch the received data.

## 18.3.5 TYPICAL CONNECTION

Figure 18-2 illustrates a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCKx signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends valid data – Slave sends dummy data
- Master sends valid data – Slave sends valid data
- Master sends dummy data – Slave sends valid data

**FIGURE 18-2: SPI MASTER/SLAVE CONNECTION**



## 18.3.6 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 18-2) is to 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 CKP is selected by appropriately programming the CKP bit (SSPxCON1<4>). This then, would give waveforms for SPI communication as illustrated in Figure 18-3, Figure 18-5 and Figure 18-6, where the

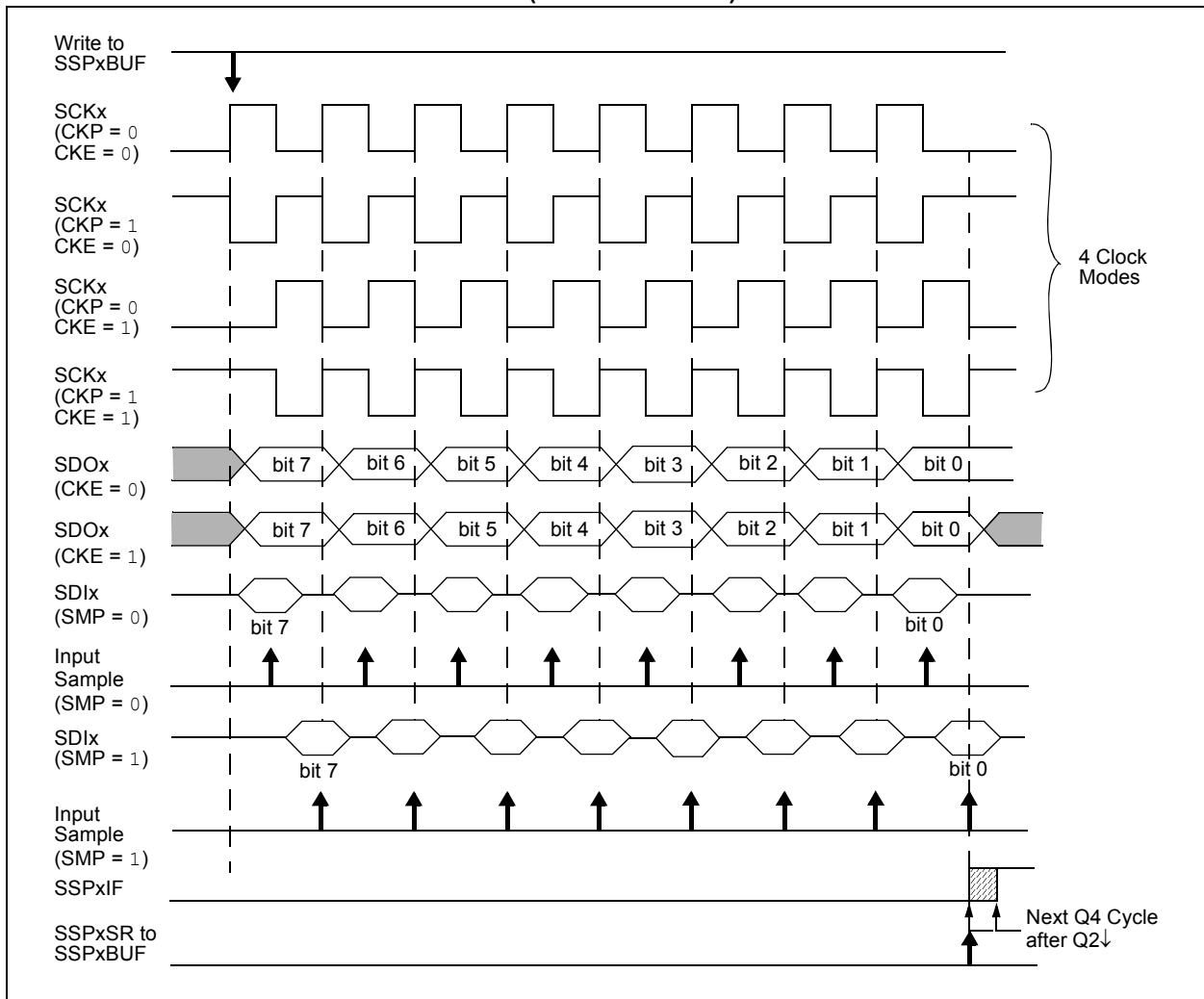
Most Significant Byte (MSB) is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- FOSC/4 (or TcY)
- FOSC/16 (or 4 • TcY)
- FOSC/64 (or 16 • TcY)
- Timer2 output/2

When using the Timer2 output/2 option, the Period Register 2 (PR2) can be used to determine the SPI bit rate. However, only PR2 values of 0x01 to 0xFF are valid in this mode.

Figure 18-3 illustrates 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 18-3: SPI MODE WAVEFORM (MASTER MODE)**



# PIC18F46J11 FAMILY

## 18.3.7 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.

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 can be configured to wake-up from Sleep.

## 18.3.8 SLAVE SELECT SYNCHRONIZATION

The  $\overline{SSx}$  pin allows a Synchronous Slave mode. The SPI must be in Slave mode with the  $\overline{SSx}$  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.

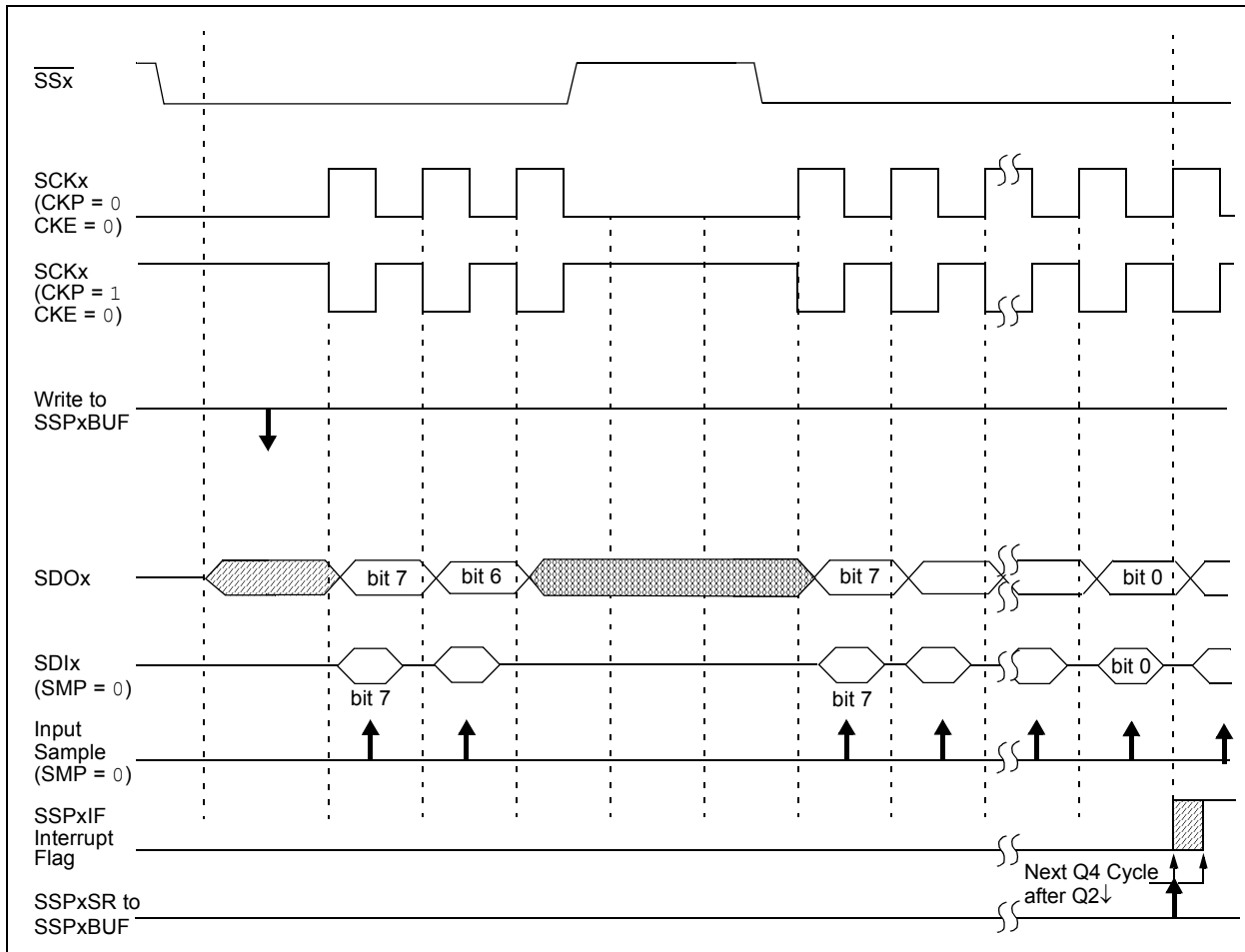
**Note 1:** When the SPI is in Slave mode with the  $\overline{SSx}$  pin control enabled ( $SSPxCON1<3:0> = 0100$ ), the SPI module will reset if the  $\overline{SSx}$  pin is set to VDD.

**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.

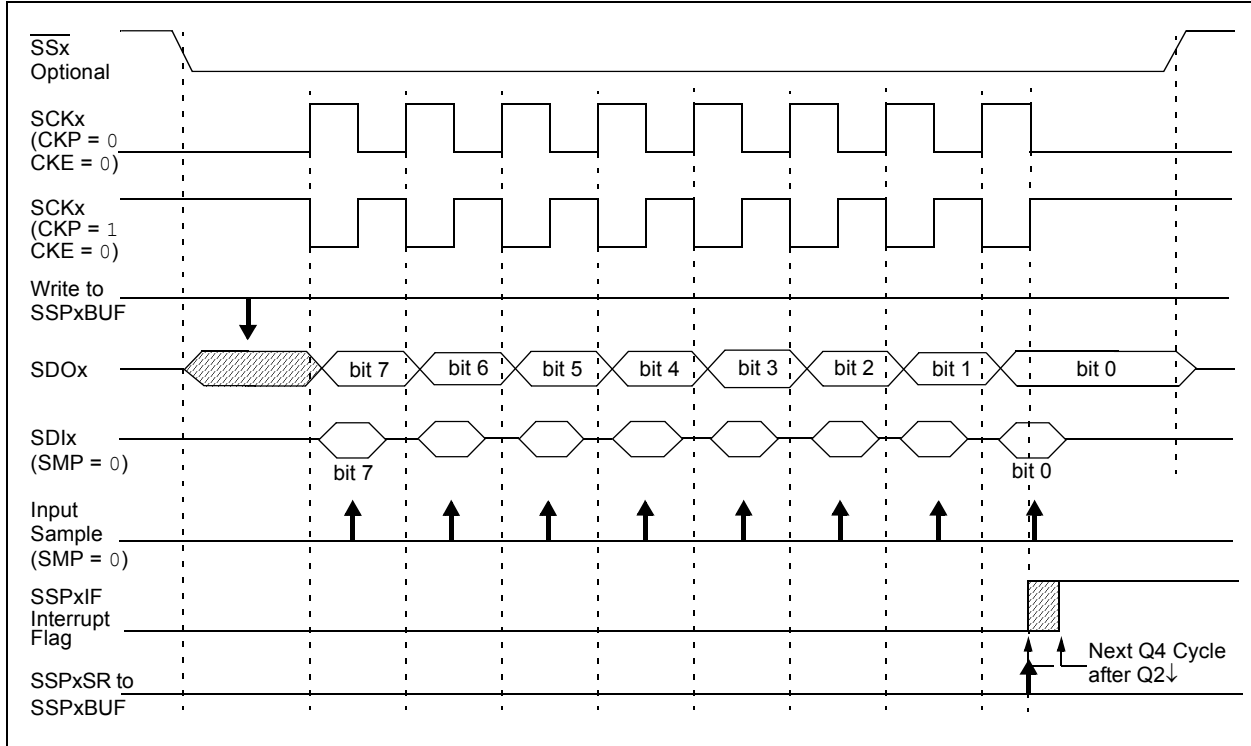
**FIGURE 18-4: SLAVE SYNCHRONIZATION WAVEFORM**



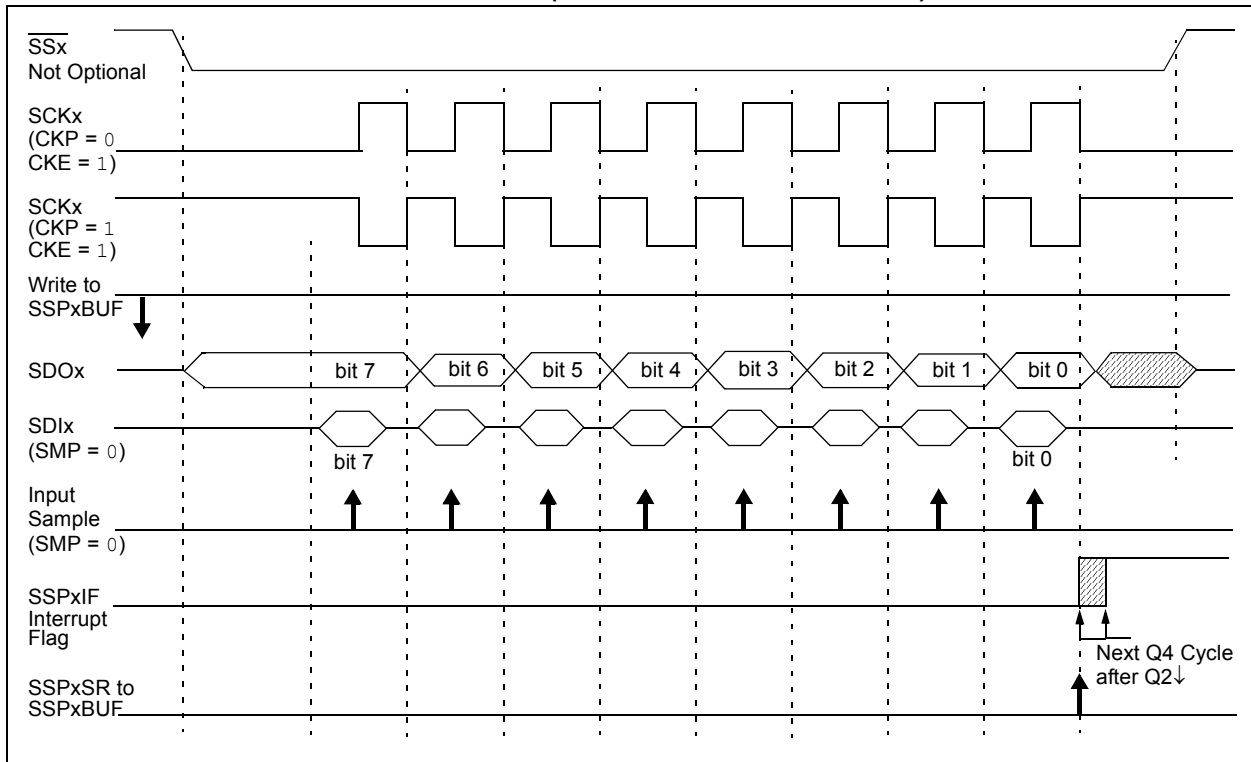


# PIC18F46J11 FAMILY

**FIGURE 18-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)**



**FIGURE 18-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)**



# PIC18F46J11 FAMILY

## 18.3.9 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 can be from the primary clock source, the secondary clock (Timer1 oscillator) or the INTOSC source. See **Section 2.3 “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.

## 18.3.10 EFFECTS OF A RESET

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

## 18.3.11 BUS MODE COMPATIBILITY

Table 18-1 provides the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

**TABLE 18-1: 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.

## 18.3.12 SPI CLOCK SPEED AND MODULE INTERACTIONS

Because MSSP1 and MSSP2 are independent modules, they can operate simultaneously at different data rates. Setting the SSPM<3:0> 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 module's 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.

# PIC18F46J11 FAMILY

**TABLE 18-2: 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	63
PIR1	PMPIF <sup>(2)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(2)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPPIF <sup>(2)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	66
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	66
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	66
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								64
SSPxCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	64
SSPxSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	64
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								67
ODCON3 <sup>(1)</sup>	—	—	—	—	—	—	SPI2OD	SPI1OD	68

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

**Note 1:** Configuration SFR overlaps with default SFR at this address; available only when WDTCON<4> = 1.

**2:** These bits are only available on 44-pin devices.

# PIC18F46J11 FAMILY

## 18.4 SPI DMA Module

The SPI DMA module contains control logic to allow the MSSP2 module to perform SPI direct memory access transfers. This enables the module to quickly transmit or receive large amounts of data with relatively little CPU intervention. When the SPI DMA module is used, MSSP2 can directly read and write to general purpose SRAM. When the SPI DMA module is not enabled, MSSP2 functions normally, but without DMA capability.

The SPI DMA module is composed of control logic, a Destination Receive Address Pointer, a Transmit Source Address Pointer, an interrupt manager and a Byte Count register for setting the size of each DMA transfer. The DMA module may be used with all SPI Master and Slave modes, and supports both half-duplex and full-duplex transfers.

### 18.4.1 I/O PIN CONSIDERATIONS

When enabled, the SPI DMA module uses the MSSP2 module. All SPI related input and output signals related to MSSP2 are routed through the Peripheral Pin Select module. The appropriate initialization procedure as described in **Section 18.4.6 “Using the SPI DMA Module”** will need to be followed prior to using the SPI DMA module. The output pins assigned to the SDO2 and SCK2 functions can optionally be configured as open-drain outputs, such as for level shifting operations mentioned in the same section.

### 18.4.2 RAM TO RAM COPY OPERATIONS

Although the SPI DMA module is primarily intended to be used for SPI communication purposes, the module can also be used to perform RAM to RAM copy operations. To do this, configure the module for Full-Duplex Master mode operation, but assign the SDO2 output and SDI2 input functions onto the same RPn pin in the PPS module. This will allow the module to operate in Loopback mode, providing RAM copy capability.

### 18.4.3 IDLE AND SLEEP CONSIDERATIONS

The SPI DMA module remains fully functional when the microcontroller is in Idle mode.

During normal sleep, the SPI DMA module is not functional and should not be used. To avoid corrupting a transfer, user firmware should be careful to make certain that pending DMA operations are complete by polling the DMAEN bit in the DMACON1 register prior to putting the microcontroller into Sleep.

In SPI Slave modes, the MSSP2 module is capable of transmitting and/or receiving one byte of data while in Sleep mode. This allows the SSP2IF flag in the PIR3 register to be used as a wake-up source. When the DMAEN bit is cleared, the SPI DMA module is effectively disabled, and the MSSP2 module functions normally, but without DMA capabilities. If the DMAEN bit is clear prior to entering Sleep, it is still possible to use the SSP2IF as a wake-up source without any data loss.

Neither MSSP2 nor the SPI DMA module will provide any functionality in Deep Sleep. Upon exiting from Deep Sleep, all of the I/O pins, MSSP2 and SPI DMA related registers will need to be fully reinitialized before the SPI DMA module can be used again.

### 18.4.4 REGISTERS

The SPI DMA engine is enabled and controlled by the following Special Function Registers:

- DMACON1
- DMACON2
- TXADDRH
- TXADDRL
- RXADDRH
- RXADDRL
- DMABCH
- DMABCL

#### 18.4.4.1 DMACON1

The DMACON1 register is used to select the main operating mode of the SPI DMA module. The SSSCON1 and SSSCON0 bits are used to control the slave select pin.

When MSSP2 is used in SPI Master mode with the SPI DMA module,  $\overline{\text{SSDMA}}$  can be controlled by the DMA module as an output pin. If MSSP2 will be used to communicate with an SPI slave device that needs the  $\overline{\text{SS}}$  pin to be toggled periodically, the SPI DMA hardware can automatically be used to deassert  $\overline{\text{SS}}$  between each byte, every two bytes or every four bytes.

Alternatively, user firmware can manually generate slave select signals with normal general purpose I/O pins, if required by the slave device(s).

When the TXINC bit is set, the TXADDR register will automatically increment after each transmitted byte. Automatic transmit address increment can be disabled by clearing the TXINC bit. If the automatic transmit address increment is disabled, each byte which is output on SDO2, will be the same (the contents of the SRAM pointed to by the TXADDR register) for the entire DMA transaction.

When the RXINC bit is set, the RXADDR register will automatically increment after each received byte. Automatic receive address increment can be disabled by clearing the RXINC bit. If RXINC is disabled in Full-Duplex or Half-Duplex Receive modes, all incoming data bytes on SDI2 will overwrite the same memory location pointed to by the RXADDR register. After the SPI DMA transaction has completed, the last received byte will reside in the memory location pointed to by the RXADDR register.

The SPI DMA module can be used for either half-duplex receive only communication, half-duplex transmit only communication or full-duplex simultaneous transmit and receive operations. All modes are available for both SPI master and SPI slave configurations. The DUPLEX0 and DUPLEX1 bits can be used to select the desired operating mode.

The behavior of the DLYINTEN bit varies greatly depending on the SPI operating mode. For example behavior for each of the modes, see Figure 18-3 through Figure 18-6.

**SPI Slave mode, DLYINTEN = 1:** In this mode, an SSP2IF interrupt will be generated during a transfer if the time between successful byte transmission events is longer than the value set by the DLYCYC<3:0> bits in the DMACON2 register. This interrupt allows slave firmware to know that the master device is taking an unusually large amount of time between byte transmissions. For example, this information may be useful for implementing application-defined communication protocols involving time-outs if the bus remains Idle for too long. When DLYINTEN = 1, the DLYLVL<3:0> interrupts occur normally according to the selected setting.

**SPI Slave mode, DLYINTEN = 0:** In this mode, the time-out based interrupt is disabled. No additional SSP2IF interrupt events will be generated by the SPI DMA module, other than those indicated by the INTLVL<3:0> bits in the DMACON2 register. In this mode, always set DLYCYC<3:0> = 0000.

**SPI Master mode, DLYINTEN = 0:** The DLYCYC<3:0> bits in the DMACON2 register determine the amount of additional inter-byte delay, which is added by the SPI DMA module during a transfer. The Master mode SS2 output feature may be used.

**SPI Master mode, DLYINTEN = 1:** The amount of hardware overhead is slightly reduced in this mode, and the minimum inter-byte delay is 8 Tcy for Fosc/4, 9 Tcy for Fosc/16 and 15 Tcy for Fosc/64. This mode can potentially be used to obtain slightly higher effective SPI bandwidth. In this mode, the SS2 control feature cannot be used, and should always be disabled (DMACON1<7:6> = 00). Additionally, the interrupt generating hardware (used in Slave mode) remains active. To avoid extraneous SSP2IF interrupt events, set the DMACON2 delay bits, DLYCYC<3:0> = 1111, and ensure that the SPI serial clock rate is no slower than Fosc/64.

In SPI Master modes, the DMAEN bit is used to enable the SPI DMA module and to initiate an SPI DMA transaction. After user firmware sets the DMAEN bit, the DMA hardware will begin transmitting and/or receiving data bytes according to the configuration used. In SPI Slave modes, setting the DMAEN bit will finish the initialization steps needed to prepare the SPI DMA module for communication (which must still be initiated by the master device).

To avoid possible data corruption, once the DMAEN bit is set, user firmware should not attempt to modify any of the MSSP2 or SPI DMA related registers, with the exception of the INTLVL bits in the DMACON2 register.

If user firmware wants to halt an ongoing DMA transaction, the DMAEN bit can be manually cleared by the firmware. Clearing the DMAEN bit while a byte is currently being transmitted will not immediately halt the byte in progress. Instead, any byte currently in progress will be completed before the MSSP2 and SPI DMA modules go back to their Idle conditions. If user firmware clears the DMAEN bit, the TXADDR, RXADDR and DMABC registers will no longer update, and the DMA module will no longer make any additional read or writes to SRAM; therefore, state information can be lost.

# PIC18F46J11 FAMILY

## REGISTER 18-3: DMACON1: DMA CONTROL REGISTER 1 (ACCESS F88h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSCON1	SSCON0	TXINC	RXINC	DUPLEX1	DUPLEX0	DLYINTEN	DMAEN
bit 7							bit 0

### Legend:

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

- bit 7-6     **SSCON<1:0>**: SSDMA Output Control bits (Master modes only)  
 11 = SSDMA is asserted for the duration of 4 bytes; DLYINTEN is always reset low  
 01 = SSDMA is asserted for the duration of 2 bytes; DLYINTEN is always reset low  
 10 = SSDMA is asserted for the duration of 1 byte; DLYINTEN is always reset low  
 00 = SSDMA is not controlled by the DMA module; DLYINTEN bit is software programmable
- bit 5     **TXINC**: Transmit Address Increment Enable bit  
 Allows the transmit address to increment as the transfer progresses.  
 1 = The transmit address is to be incremented from the initial value of TXADDR<11:0>  
 0 = The transmit address is always set to the initial value of TXADDR<11:0>
- bit 4     **RXINC**: Receive Address Increment Enable bit  
 Allows the receive address to increment as the transfer progresses.  
 1 = The received address is to be incremented from the initial value of RXADDR<11:0>  
 0 = The received address is always set to the initial value of RXADDR<11:0>
- bit 3-2   **DUPLEX<1:0>**: Transmit/Receive Operating Mode Select bits  
 10 = SPI DMA operates in Full-Duplex mode, data is simultaneously transmitted and received  
 01 = DMA operates in Half-Duplex mode, data is transmitted only  
 00 = DMA operates in Half-Duplex mode, data is received only
- bit 1     **DLYINTEN**: Delay Interrupt Enable bit  
 Enables the interrupt to be invoked after the number of SCK cycles specified in DLYCYC<2:0> has elapsed from the latest completed transfer.  
 1 = The interrupt is enabled, SSCON<1:0> must be set to '00'  
 0 = The interrupt is disabled
- bit 0     **DMAEN**: DMA Operation Start/Stop bit  
 This bit is set by the users' software to start the DMA operation. It is reset back to zero by the DMA engine when the DMA operation is completed or aborted.  
 1 = DMA is in session  
 0 = DMA is not in session

# PIC18F46J11 FAMILY

## 18.4.4.2 DMACON2

The DMACON2 register contains control bits for controlling interrupt generation and inter-byte delay behavior. The INTLVL<3:0> bits are used to select when an SSP2IF interrupt should be generated. The function of the DLYCYC<3:0> bits depends on the SPI operating mode (Master/Slave), as well as the DLYINTEN setting. In SPI Master mode, the DLYCYC<3:0> bits can be used

to control how much time the module will idle between bytes in a transfer. By default, the hardware requires a minimum delay of: 8 T<sub>cy</sub> for F<sub>osc</sub>/4, 9 T<sub>cy</sub> for F<sub>osc</sub>/16 and 15 T<sub>cy</sub> for F<sub>osc</sub>/64. Additional delays can be added with the DLYCYC bits. In SPI Slave modes, the DLYCYC<3:0> bits may optionally be used to trigger an additional time-out based interrupt.

**REGISTER 18-4: DMACON2: DMA CONTROL REGISTER 2 (ACCESS F86h)**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DLYCYC3	DLYCYC2	DLYCYC1	DLYCYC0	INTLVL3	INTLVL2	INTLVL1	INTLVL0
bit 7							bit 0

**Legend:**

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

bit 7-4      **DLYCYC<3:0>**: Delay Cycle Selection bits

When DLYINTEN = 0, these bits specify the additional delay (above the base overhead of the hardware) in number of T<sub>cy</sub> cycles before the SSP2BUF register is written again for the next transfer. When DLYINTEN = 1, these bits specify the additional delay in number of T<sub>cy</sub> cycles from the latest completed transfer before an interrupt to the CPU is invoked. In this case, the delay before the SSP2BUF register is written again is 1 T<sub>cy</sub> + (base overhead of hardware).

- 1111 = Delay time in number of instruction cycles is 2,048 cycles
- 1110 = Delay time in number of instruction cycles is 1,024 cycles
- 1101 = Delay time in number of instruction cycles is 896 cycles
- 1100 = Delay time in number of instruction cycles is 768 cycles
- 1011 = Delay time in number of instruction cycles is 640 cycles
- 1010 = Delay time in number of instruction cycles is 512 cycles
- 1001 = Delay time in number of instruction cycles is 384 cycles
- 1000 = Delay time in number of instruction cycles is 256 cycles
- 0111 = Delay time in number of instruction cycles is 128 cycles
- 0110 = Delay time in number of instruction cycles is 64 cycles
- 0101 = Delay time in number of instruction cycles is 32 cycles
- 0100 = Delay time in number of instruction cycles is 16 cycles
- 0011 = Delay time in number of instruction cycles is 8 cycles
- 0010 = Delay time in number of instruction cycles is 4 cycles
- 0001 = Delay time in number of instruction cycles is 2 cycles
- 0000 = Delay time in number of instruction cycles is 1 cycle

# PIC18F46J11 FAMILY

---

## REGISTER 18-4: DMACON2: DMA CONTROL REGISTER 2 (ACCESS F86h) (CONTINUED)

bit 3-0 **INTLVL<3:0>**: Watermark Interrupt Enable bits

These bits specify the amount of remaining data yet to be transferred (transmitted and/or received) upon which an interrupt is generated.

1111 = Amount of remaining data to be transferred is 576 bytes  
1110 = Amount of remaining data to be transferred is 512 bytes  
1101 = Amount of remaining data to be transferred is 448 bytes  
1100 = Amount of remaining data to be transferred is 384 bytes  
1011 = Amount of remaining data to be transferred is 320 bytes  
1010 = Amount of remaining data to be transferred is 256 bytes  
1001 = Amount of remaining data to be transferred is 192 bytes  
1000 = Amount of remaining data to be transferred is 128 bytes  
0111 = Amount of remaining data to be transferred is 67 bytes  
0110 = Amount of remaining data to be transferred is 32 bytes  
0101 = Amount of remaining data to be transferred is 16 bytes  
0100 = Amount of remaining data to be transferred is 8 bytes  
0011 = Amount of remaining data to be transferred is 4 bytes  
0010 = Amount of remaining data to be transferred is 2 bytes  
0001 = Amount of remaining data to be transferred is 1 byte  
0000 = Transfer complete



### 18.4.4.3 DMABCH and DMABCL

The DMABCH and DMABCL register pair forms a 10-bit Byte Count register, which is used by the SPI DMA module to send/receive up to 1,024 bytes for each DMA transaction. When the DMA module is actively running (DMAEN = 1), the DMA Byte Count register decrements after each byte is transmitted/received. The DMA transaction will halt and the DMAEN bit will be automatically cleared by hardware after the last byte has completed. After a DMA transaction is complete, the DMABC register will read 0x000.

Prior to initiating a DMA transaction by setting the DMAEN bit, user firmware should load the appropriate value into the DMABCH/DMABCL registers. The DMABC is a “base zero” counter, so the actual number of bytes which will be transmitted follows in Equation 18-1.

For example, if user firmware wants to transmit 7 bytes in one transaction, DMABC should be loaded with 006h. Similarly, if user firmware wishes to transmit 1,024 bytes, DMABC should be loaded with 3FFh.

#### EQUATION 18-1: BYTES TRANSMITTED FOR A GIVEN DMABC

$$\text{Bytes}_{\text{XMIT}} = (\text{DMABC} + 1)$$

### 18.4.4.4 TXADDRH and TXADDRL

The TXADDRH and TXADDRL registers pair together to form a 12-bit Transmit Source Address Pointer register. In modes that use TXADDR (Full-Duplex and Half-Duplex Transmit), the TXADDR will be incremented after each byte is transmitted. Transmitted data bytes will be taken from the memory location pointed to by the TXADDR register. The contents of the memory locations pointed to by TXADDR will not be modified by the DMA module during a transmission.

The SPI DMA module can read from and transmit data from all general purpose memory on the device. The SPI DMA module cannot be used to read from the Special Function Registers (SFRs) contained in banks 14 and 15.

### 18.4.4.5 RXADDRH and RXADDRL

The RXADDRH and RXADDRL register pair together to form a 12-bit Receive Destination Address Pointer. In modes that use RXADDR (Full-Duplex and Half-Duplex Receive), the RXADDR register will be incremented after each byte is received. Received data bytes will be stored at the memory location pointed to by the RXADDR register.

The SPI DMA module can write received data to all general purpose memory on the device. The SPI DMA module cannot be used to modify the Special Function Registers contained in banks 14 and 15.

### 18.4.5 INTERRUPTS

The SPI DMA module alters the behavior of the SSP2IF interrupt flag. In normal/non-DMA modes, the SSP2IF is set once after every single byte is transmitted/received through the MSSP2 module. When MSSP2 is used with the SPI DMA module, the SSP2IF interrupt flag will be set according to the user-selected INTLVL<3:0> value specified in the DMACON2 register. The SSP2IF interrupt condition will also be generated once the SPI DMA transaction has fully completed, and the DMAEN bit has been cleared by hardware.

The SSP2IF flag becomes set once the DMA byte count value indicates that the specified INTLVL has been reached. For example, if DMACON2<3:0> = 0101 (16 bytes remaining), the SSP2IF interrupt flag will become set once DMABC reaches 00Fh. If user firmware then clears the SSP2IF interrupt flag, the flag will not be set again by the hardware until after all bytes have been fully transmitted and the DMA transaction is complete.

**Note:** User firmware may modify the INTLVL bits while a DMA transaction is in progress (DMAEN = 1). If an INTLVL value is selected which is higher than the actual remaining number of bytes (indicated by DMABC + 1), the SSP2IF interrupt flag will immediately become set.

For example, if DMABC = 00Fh (implying 16 bytes are remaining) and user firmware writes ‘1111’ to INTLVL<3:0> (interrupt when 576 bytes remaining), the SSP2IF interrupt flag will immediately become set. If user firmware clears this interrupt flag, a new interrupt condition will not be generated until either: user firmware again writes INTLVL with an interrupt level higher than the actual remaining level, or the DMA transaction completes and the DMAEN bit is cleared.

**Note:** If the INTLVL bits are modified while a DMA transaction is in progress, care should be taken to avoid inadvertently changing the DLYCYC<3:0> value.

# PIC18F46J11 FAMILY

---

## 18.4.6 USING THE SPI DMA MODULE

The following steps would typically be taken to enable and use the SPI DMA module:

1. Configure the I/O pins, which will be used by MSSP2.
  - a) Assign SCK2, SDO2, SDI2 and  $\overline{SS2}$  to RPN pins as appropriate for the SPI mode which will be used. Only functions which will be used need to be assigned to a pin.
  - b) Initialize the associated LATx registers for the desired Idle SPI bus state.
  - c) If Open-Drain Output mode on SDO2 and SCK2 (Master mode) is desired, set ODCON3<1>.
  - d) Configure corresponding TRISx bits for each I/O pin used
2. Configure and enable MSSP2 for the desired SPI operating mode.
  - a) Select the desired operating mode (Master or Slave, SPI Mode 0, 1, 2 and 3) and configure the module by writing to the SSP2STAT and SSP2CON1 registers.
  - b) Enable MSSP2 by setting SSP2CON1<5> = 1.
3. Configure the SPI DMA engine.
  - a) Select the desired operating mode by writing the appropriate values to DMACON2 and DMACON1.
  - b) Initialize the TXADDRH/TXADDRL Pointer (Full-Duplex or Half-Duplex Transmit Only mode).
  - c) Initialize the RXADDRH/RXADDRL Pointer (Full-Duplex or Half-Duplex Receive Only mode).
  - d) Initialize the DMABCH/DMABCL Byte Count register with the number of bytes to be transferred in the next SPI DMA operation.
  - e) Set the DMAEN bit (DMACON1<0>).

In SPI Master modes, this will initiate a DMA transaction. In SPI Slave modes, this will complete the initialization process, and the module will now be ready to begin receiving and/or transmitting data to the master device once the master starts the transaction.

4. Detect the SSP2IF interrupt condition (PIR3<7>).
  - a) If the interrupt was configured to occur at the completion of the SPI DMA transaction, the DMAEN bit (DMACON1<0>) will be clear. User firmware may prepare the module for another transaction by repeating steps 3.b through 3.e.
  - b) If the interrupt was configured to occur prior to the completion of the SPI DMA transaction, the DMAEN bit may still be set, indicating the transaction is still in progress. User firmware would typically use this interrupt condition to begin preparing new data for the next DMA transaction. Firmware should not repeat steps 3.b. through 3.e. until the DMAEN bit is cleared by the hardware, indicating the transaction is complete.

Example 18-2 provides example code demonstrating the initialization process and the steps needed to use the SPI DMA module to perform a 512-byte Full-Duplex, Master mode transfer.

# PIC18F46J11 FAMILY

## EXAMPLE 18-2: 512-BYTE SPI MASTER MODE Init AND TRANSFER

```
                                ;For this example, let's use RP5(RB2) for SCK2,
                                ;RP4(RB1) for SDO2, and RP3(RB0) for SDI2

                                ;Let's use SPI master mode, CKE = 0, CKP = 0,
                                ;without using slave select signalling.

InitSPI Pins:
    movlb    0x0F                ;Select bank 15, for access to ODCON3 register
    bcf     ODCON3, SPI2OD       ;Let's not use open drain outputs in this example

    bcf     LATB, RB2            ;Initialize our (to be) SCK2 pin low (idle).
    bcf     LATB, RB1            ;Initialize our (to be) SDO2 pin to an idle state
    bcf     TRISB, RB1           ;Make SDO2 output, and drive low
    bcf     TRISB, RB2           ;Make SCK2 output, and drive low (idle state)
    bsf     TRISB, RB0           ;SDI2 is an input, make sure it is tri-stated

                                ;Now we should unlock the PPS registers, so we can
                                ;assign the MSSP2 functions to our desired I/O pins.

    movlb    0x0E                ;Select bank 14 for access to PPS registers
    bcf     INTCON, GIE          ;I/O Pin unlock sequence will not work if CPU
                                ;services an interrupt during the sequence
                                ;Unlock sequence consists of writing 0x55
                                ;and 0xAA to the EECON2 register.

    movlw    0x55
    movwf    EECON2
    movlw    0xAA
    movwf    EECON2
    bcf     PPSCON, IOLOCK       ;We may now write to RPINRx and RPORx registers
    bsf     INTCON, GIE          ;May now turn back on interrupts if desired

    movlw    0x03                ;0x0A is SCK2 output signal
    movwf    RPINR21             ;Assign the SDI2 function to pin RP3

    movlw    0x0A                ;Let's assign SCK2 output to pin RP4
    movwf    RPOR4               ;RPOR4 maps output signals to RP4 pin
    movlw    0x04                ;SCK2 also needs to be configured as an input on the
                                ;same pin
    movwf    RPINR22             ;SCK2 input function taken from RP4 pin
    movlw    0x09                ;0x09 is SDO2 output
    movwf    RPOR5               ;Assign SDO2 output signal to the RP5 (RB2) pin
    bsf     PPSCON, IOLOCK       ;Lock the PPS registers to prevent changes
    movlb    0x0F                ;Done with PPS registers, bank 15 has other SFRs

InitMSSP2:
    clrf     SSP2STAT            ;CKE = 0, SMP = 0 (sampled at middle of bit)
    movlw    b'00000000'         ;CKP = 0, SPI Master mode, Fosc/4
    movwf    SSP2CON1            ;MSSP2 initialized
    bsf     SSP2CON1, SSPEN       ;Enable the MSSP2 module

InitSPIDMA:
    movlw    b'00111110'         ;Full duplex, RX/TXINC enabled, no SSCON
    movwf    DMACON1             ;DLYINTEN is set, so DLYCYC3:DLYCYC0 = 1111
    movlw    b'11110000'         ;Minimum delay between bytes, interrupt
    movwf    DMACON2             ;only once when the transaction is complete
```

# PIC18F46J11 FAMILY

## EXAMPLE 18-2: 512-BYTE SPI MASTER MODE Init AND TRANSFER (CONTINUED)

```
                                ;Somewhere else in our project, lets assume we have
                                ;allocated some RAM for use as SPI receive and
                                ;transmit buffers.

;                               udata      0x500
;DestBuf      res                0x200    ;Let's reserve 0x500-0x6FF for use as our SPI
;                               ;receive data buffer in this example
;SrcBuf       res                0x200    ;Lets reserve 0x700-0x8FF for use as our SPI
;                               ;transmit data buffer in this example

PrepareTransfer:
    movlw      HIGH(DestBuf)          ;Get high byte of DestBuf address (0x05)
    movwf     RXADDRH                 ;Load upper four bits of the RXADDR register
    movlw     LOW(DestBuf)            ;Get low byte of the DestBuf address (0x00)
    movwf     RXADDRL                 ;Load lower eight bits of the RXADDR register

    movlw     HIGH(SrcBuf)            ;Get high byte of SrcBuf address (0x07)
    movwf     TXADDRH                 ;Load upper four bits of the TXADDR register
    movlw     LOW(SrcBuf)             ;Get low byte of the SrcBuf address (0x00)
    movwf     TXADDRL                 ;Load lower eight bits of the TXADDR register

    movlw     0x01                    ;Lets move 0x200 (512) bytes in one DMA xfer
    movwf     DMABCH                  ;Load the upper two bits of DMABC register
    movlw     0xFF                    ;Actual bytes transferred is (DMABC + 1), so
    movwf     DMABCL                  ;we load 0x01FF into DMABC to xfer 0x200 bytes

BeginXfer:
    bsf       DMACON1, DMAEN          ;The SPI DMA module will now begin transferring
                                ;the data taken from SrcBuf, and will store
                                ;received bytes into DestBuf.

;Execute whatever                  ;CPU is now free to do whatever it wants to
                                ;and the DMA operation will continue without
                                ;intervention, until it completes.

                                ;When the transfer is complete, the SSP2IF flag in
                                ;the PIR3 register will become set, and the DMAEN bit
                                ;is automatically cleared by the hardware.
                                ;The DestBuf (0x500-0x7FF) will contain the received
                                ;data. To start another transfer, firmware will need
                                ;to reinitialize RXADDR, TXADDR, DMABC and then
                                ;set the DMAEN bit.
```

## 18.5 I<sup>2</sup>C Mode

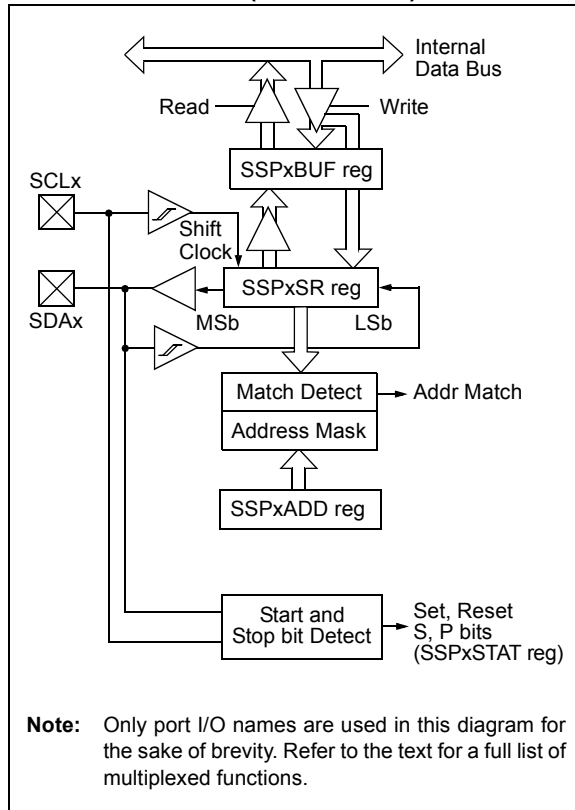
The MSSP module in I<sup>2</sup>C mode fully implements all master and slave functions (including general call support), and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications and 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial Clock (SCLx) – RB4/PMA1/KBI0/SCK1/SCL1/RP7 or RD0/PMD0/SCL2
- Serial Data (SDAx) – RB5/PMA0/KBI1/SDI1/SDA1/RP8 or RD1/PMD1/SDA2

The user must configure these pins as inputs by setting the associated TRIS bits.

**FIGURE 18-7: MSSPx BLOCK DIAGRAM (I<sup>2</sup>C™ MODE)**



### 18.5.1 REGISTERS

The MSSP module has six registers for I<sup>2</sup>C operation. These are:

- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Control Register 2 (SSPxCON2)
- MSSPx Status Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSPx Shift Register (SSPxSR) – Not directly accessible
- MSSPx Address Register (SSPxADD)
- MSSPx 7-Bit Address Mask Register (SSPxMSK)

SSPxCON1, SSPxCON2 and SSPxSTAT are the control and status registers in I<sup>2</sup>C mode operation. The SSPxCON1 and SSPxCON2 registers are readable and writable. The lower six 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.

SSPxADD contains the slave device address when the MSSP is configured in I<sup>2</sup>C Slave mode. When the MSSP is configured in Master mode, the lower seven bits of SSPxADD act as the Baud Rate Generator (BRG) reload value.

SSPxMSK holds the slave address mask value when the module is configured for 7-Bit Address Masking mode. While it is a separate register, it shares the same SFR address as SSPxADD; it is only accessible when the SSPM<3:0> bits are specifically set to permit access. Additional details are provided in **Section 18.5.3.4 “7-Bit Address Masking Mode”**.

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.

# PIC18F46J11 FAMILY

## REGISTER 18-5: SSPxSTAT: MSSPx STATUS REGISTER – I<sup>2</sup>C™ MODE (ACCESS FC7h/F73h)

R/W-1	R/W-1	R-1	R-1	R-1	R-1	R-1	R-1
SMP	CKE	D/A	P <sup>(1)</sup>	S <sup>(1)</sup>	R/W <sup>(2,3)</sup>	UA	BF
bit 7							bit 0

### Legend:

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

- bit 7            **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/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<sup>(1)</sup>  
 1 = Indicates that a Stop bit has been detected last  
 0 = Stop bit was not detected last
- bit 3            **S:** Start bit<sup>(1)</sup>  
 1 = Indicates that a Start bit has been detected last  
 0 = Start bit was not detected last
- bit 2            **R/W:** Read/Write Information bit<sup>(2,3)</sup>  
In Slave mode:  
 1 = Read  
 0 = Write  
In Master mode:  
 1 = Transmit is in progress  
 0 = Transmit is not in progress
- 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  $\overline{\text{ACK}}$  and Stop bits)  
 0 = SSPxBUF is empty (does not include the  $\overline{\text{ACK}}$  and Stop bits)

- Note 1:** This bit is cleared on Reset and when SSPEN is cleared.  
**Note 2:** This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not  $\overline{\text{ACK}}$  bit.  
**Note 3:** ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Active mode.

# PIC18F46J11 FAMILY

**REGISTER 18-6: SSPxCON1: MSSPx CONTROL REGISTER 1 – I<sup>2</sup>C™ MODE (ACCESS FC6h/F72h)**

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 <sup>(1)</sup>	CKP	SSPM3 <sup>(2)</sup>	SSPM2 <sup>(2)</sup>	SSPM1 <sup>(2)</sup>	SSPM0 <sup>(2)</sup>
bit 7							bit 0

**Legend:**

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

- bit 7      **WCOL:** Write Collision Detect bit  
In Master Transmit mode:  
 1 = A write to the SSPxBUF register was attempted while the I<sup>2</sup>C 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<sup>(1)</sup>  
 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
- bit 4      **CKP:** SCKx Release Control bit  
In Slave mode:  
 1 = Releases clock  
 0 = Holds clock low (clock stretch); used to ensure data setup time  
In Master mode:  
 Unused in this mode.
- bit 3-0    **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits<sup>(2)</sup>  
 1111 = I<sup>2</sup>C Slave mode, 10-bit address with Start and Stop bit interrupts enabled  
 1110 = I<sup>2</sup>C Slave mode, 7-bit address with Start and Stop bit interrupts enabled  
 1011 = I<sup>2</sup>C Firmware Controlled Master mode (slave Idle)  
 1001 = Load SSPxMSK register at SSPxADD SFR address<sup>(3,4)</sup>  
 1000 = I<sup>2</sup>C Master mode, clock = Fosc/(4 \* (SSPxADD + 1))  
 0111 = I<sup>2</sup>C Slave mode, 10-bit address  
 0110 = I<sup>2</sup>C Slave mode, 7-bit address

- Note 1:** When enabled, the SDAx and SCLx pins must be configured as inputs.  
**2:** Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.  
**3:** When SSPM<3:0> = 1001, any reads or writes to the SSPxADD SFR address actually accesses the SSPxMSK register.  
**4:** This mode is only available when 7-Bit Address Masking mode is selected (MSSPMSK Configuration bit is '1').

# PIC18F46J11 FAMILY

## REGISTER 18-7: SSPxCON2: MSSPx CONTROL REGISTER 2 –I<sup>2</sup>C™ MASTER MODE (ACCESS FC5h/F71h)

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 <sup>(3)</sup>	ACKSTAT	ACKDT <sup>(1)</sup>	ACKEN <sup>(2)</sup>	RCEN <sup>(2)</sup>	PEN <sup>(2)</sup>	RSEN <sup>(2)</sup>	SEN <sup>(2)</sup>
bit 7							bit 0

### Legend:

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

- bit 7      **GCEN:** General Call Enable bit (Slave mode only)<sup>(3)</sup>  
 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)<sup>(1)</sup>  
 1 = Not Acknowledge  
 0 = Acknowledge
- bit 4      **ACKEN:** Acknowledge Sequence Enable bit<sup>(2)</sup>  
 1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit; automatically cleared by hardware  
 0 = Acknowledge sequence Idle
- bit 3      **RCEN:** Receive Enable bit (Master Receive mode only)<sup>(2)</sup>  
 1 = Enables Receive mode for I<sup>2</sup>C  
 0 = Receive Idle
- bit 2      **PEN:** Stop Condition Enable bit<sup>(2)</sup>  
 1 = Initiates Stop condition on SDAx and SCLx pins; automatically cleared by hardware  
 0 = Stop condition Idle
- bit 1      **RSEN:** Repeated Start Condition Enable bit<sup>(2)</sup>  
 1 = Initiates Repeated Start condition on SDAx and SCLx pins; automatically cleared by hardware  
 0 = Repeated Start condition Idle
- bit 0      **SEN:** Start Condition Enable bit<sup>(2)</sup>  
 1 = Initiates Start condition on SDAx and SCLx pins; automatically cleared by hardware  
 0 = Start condition Idle

- Note 1:** Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.  
**Note 2:** If the I<sup>2</sup>C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).  
**Note 3:** This bit is not implemented in I<sup>2</sup>C Master mode.



# PIC18F46J11 FAMILY

## REGISTER 18-8: SSPxCON2: MSSPx CONTROL REGISTER 2 – I<sup>2</sup>C™ SLAVE MODE (ACCESS FC5h/F71h)

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 <sup>(2)</sup>	ADMSK5	ADMSK4	ADMSK3	ADMSK2	ADMSK1	SEN <sup>(1)</sup>
bit 7							bit 0

### Legend:

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

- bit 7            **GCEN:** General Call Enable bit (Slave mode only)  
                   1 = Enables interrupt when a general call address (0000h) is received in the SSPxSR  
                   0 = General call address disabled
- bit 6            **ACKSTAT:** Acknowledge Status bit<sup>(2)</sup>  
                   Unused in Slave mode.
- bit 5-2        **ADMSK<5:2>:** Slave Address Mask Select bits (5-Bit Address Masking)  
                   1 = Masking of corresponding bits of SSPxADD enabled  
                   0 = Masking of corresponding bits of SSPxADD disabled
- bit 1            **ADMSK1:** Slave Address Least Significant bit(s) Mask Select bit  
                   In 7-Bit Addressing mode:  
                   1 = Masking of SSPxADD<1> only enabled  
                   0 = Masking of SSPxADD<1> only disabled  
                   In 10-Bit Addressing mode:  
                   1 = Masking of SSPxADD<1:0> enabled  
                   0 = Masking of SSPxADD<1:0> disabled
- bit 0            **SEN:** Start Condition Enable/Stretch Enable bit<sup>(1)</sup>  
                   1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)  
                   0 = Clock stretching is disabled

- Note 1:** If the I<sup>2</sup>C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).
- 2:** This bit is unimplemented in I<sup>2</sup>C Slave mode.

## REGISTER 18-9: SSPxMSK: I<sup>2</sup>C™ SLAVE ADDRESS MASK REGISTER – 7-BIT MASKING MODE (ACCESS FC8h/F74h)<sup>(1)</sup>

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0 <sup>(2)</sup>
bit 7							bit 0

### Legend:

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

- bit 7-0        **MSK<7:0>:** Slave Address Mask Select bits  
                   1 = Masking of corresponding bit of SSPxADD enabled  
                   0 = Masking of corresponding bit of SSPxADD disabled

- Note 1:** This register shares the same SFR address as SSPxADD and is only addressable in select MSSP operating modes. See **Section 18.5.3.4 “7-Bit Address Masking Mode”** for more details.
- 2:** MSK0 is not used as a mask bit in 7-bit addressing.

# PIC18F46J11 FAMILY

## 18.5.2 OPERATION

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

The SSPxCON1 register allows control of the I<sup>2</sup>C operation. Four mode selection bits (SSPxCON1<3:0>) allow one of the following I<sup>2</sup>C modes to be selected:

- I<sup>2</sup>C Master mode, clock
- I<sup>2</sup>C Slave mode (7-bit address)
- I<sup>2</sup>C Slave mode (10-bit address)
- I<sup>2</sup>C Slave mode (7-bit address) with Start and Stop bit interrupts enabled
- I<sup>2</sup>C Slave mode (10-bit address) with Start and Stop bit interrupts enabled
- I<sup>2</sup>C Firmware Controlled Master mode, slave is Idle

Selection of any I<sup>2</sup>C mode with the SSPEN bit set forces the SCLx and SDAx pins to be open-drain, provided these pins are programmed as inputs by setting the appropriate TRISB or TRISD bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCLx and SDAx pins.

## 18.5.3 SLAVE MODE

In Slave mode, the SCLx and SDAx pins must be configured as inputs (TRISB<5:4> set). The MSSP module will override the input state with the output data when required (slave-transmitter).

The I<sup>2</sup>C Slave mode hardware will always generate an interrupt on an address match. Address masking will allow the hardware to generate an interrupt for more than one address (up to 31 in 7-bit addressing and up to 63 in 10-bit addressing). 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 I<sup>2</sup>C specification, as well as the requirement of the MSSP module, are shown in timing parameter 100 and parameter 101.

### 18.5.3.1 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  $\overline{\text{ACK}}$  pulse is generated.
4. The MSSPx Interrupt Flag bit, SSPxIF, is set (and interrupt is generated, if enabled) on the falling edge of the ninth SCLx pulse.

In 10-Bit Addressing 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 addressing is as follows, with steps 7 through 9 for the slave-transmitter:

1. Receive first (high) byte of address (bits, SSPxIF, BF and UA, are set on address match).
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.

## 18.5.3.2 Address Masking Modes

Masking an address bit causes that bit to become a “don’t care”. When one address bit is masked, two addresses will be Acknowledged and cause an interrupt. It is possible to mask more than one address bit at a time, which greatly expands the number of addresses Acknowledged.

The I<sup>2</sup>C slave behaves the same way, whether address masking is used or not. However, when address masking is used, the I<sup>2</sup>C slave can Acknowledge multiple addresses and cause interrupts. When this occurs, it is necessary to determine which address caused the interrupt by checking SSPXBUF.

The PIC18F46J11 family of devices is capable of using two different Address Masking modes in I<sup>2</sup>C slave operation: 5-Bit Address Masking and 7-Bit Address Masking. The Masking mode is selected at device configuration using the MSSPMSK Configuration bit. The default device configuration is 7-Bit Address Masking.

Both Masking modes, in turn, support address masking of 7-bit and 10-bit addresses. The combination of Masking modes and addresses provide different ranges of Acknowledgable addresses for each combination.

While both Masking modes function in roughly the same manner, the way they use address masks is different.

## 18.5.3.3 5-Bit Address Masking Mode

As the name implies, 5-Bit Address Masking mode uses an address mask of up to five bits to create a range of addresses to be Acknowledged, using bits 5 through 1 of

the incoming address. This allows the module to Acknowledge up to 31 addresses when using 7-bit addressing, or 63 addresses with 10-bit addressing (see Example 18-3). This Masking mode is selected when the MSSPMSK Configuration bit is programmed ('0').

The address mask in this mode is stored in the SSPxCON2 register, which stops functioning as a control register in I<sup>2</sup>C Slave mode (Register 18-8). In 7-Bit Address Masking mode, address mask bits, ADMSK<5:1> (SSPxCON2<5:1>), mask the corresponding address bits in the SSPxADD register. For any ADMSK bits that are set (ADMSK<n> = 1), the corresponding address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

In 10-Bit Address Masking mode, bits, ADMSK<5:2>, mask the corresponding address bits in the SSPxADD register. In addition, ADMSK1 simultaneously masks the two LSBs of the address (SSPxADD<1:0>). For any ADMSK bits that are active (ADMSK<n> = 1), the corresponding address bit is ignored (SPxADD<n> = x). Also note, that although in 10-Bit Address Masking mode, the upper address bits reuse part of the SSPxADD register bits. The address mask bits do not interact with those bits; they only affect the lower address bits.

**Note 1:** ADMSK1 masks the two Least Significant bits of the address.  
**2:** The two MSBs of the address are not affected by address masking.

## EXAMPLE 18-3: ADDRESS MASKING EXAMPLES IN 5-BIT MASKING MODE

### 7-Bit Addressing:

SSPxADD<7:1> = A0h (1010000) (SSPxADD<0> is assumed to be '0')

ADMSK<5:1> = 00111

Addresses Acknowledged: A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

### 10-Bit Addressing:

SSPxADD<7:0> = A0h (10100000) (The two MSBs of the address are ignored in this example, since they are not affected by masking)

ADMSK<5:1> = 00111

Addresses Acknowledged: A0h, A1h, A2h, A3h, A4h, A5h, A6h, A7h, A8h, A9h, AAh, ABh, ACh, ADh, AEh, AFh

# PIC18F46J11 FAMILY

## 18.5.3.4 7-Bit Address Masking Mode

Unlike 5-Bit Address Masking mode, 7-Bit Address Masking mode uses a mask of up to eight bits (in 10-bit addressing) to define a range of addresses that can be Acknowledged, using the lowest bits of the incoming address. This allows the module to Acknowledge up to 127 different addresses with 7-bit addressing, or 255 with 10-bit addressing (see Example 18-4). This mode is the default configuration of the module, and is selected when MSSPMSK is unprogrammed ('1').

The address mask for 7-Bit Address Masking mode is stored in the SSPxMSK register, instead of the SSPxCON2 register. SSPxMSK is a separate hardware register within the module, but it is not directly addressable. Instead, it shares an address in the SFR space with the SSPxADD register. To access the SSPxMSK register, it is necessary to select MSSP mode, '1001' (SSPCON1<3:0> = 1001), and then read or write to the location of SSPxADD.

To use 7-Bit Address Masking mode, it is necessary to initialize SSPxMSK with a value before selecting the I<sup>2</sup>C Slave Addressing mode. Thus, the required sequence of events is:

1. Select SSPxMSK Access mode (SSPxCON2<3:0> = 1001).
2. Write the mask value to the appropriate SSPxADD register address (FC8h for MSSP1, F6Eh for MSSP2).
3. Set the appropriate I<sup>2</sup>C Slave mode (SSPxCON2<3:0> = 0111 for 10-bit addressing, 0110 for 7-bit addressing).

Setting or clearing mask bits in SSPxMSK behaves in the opposite manner of the ADMSK bits in 5-Bit Address Masking mode. That is, clearing a bit in SSPxMSK causes the corresponding address bit to be masked; setting the bit requires a match in that position. SSPxMSK resets to all '1's upon any Reset condition and, therefore, has no effect on the standard MSSP operation until written with a mask value.

With 7-Bit Address Masking mode, SSPxMSK<7:1> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (SSPxMSK<n> = 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

With 10-Bit Address Masking mode, SSPxMSK<7:0> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (= 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x).

**Note:** The two MSBs of the address are not affected by address masking.

### EXAMPLE 18-4: ADDRESS MASKING EXAMPLES IN 7-BIT MASKING MODE

#### 7-Bit Addressing:

SSPxADD<7:1> = 1010 000

SSPxMSK<7:1> = 1111 001

Addresses Acknowledged = A8h, A6h, A4h, A0h

#### 10-Bit Addressing:

SSPxADD<7:0> = 1010 0000 (The two MSBs are ignored in this example since they are not affected)

SSPxMSK<5:1> = 1111 0

Addresses Acknowledged = A8h, A6h, A4h, A0h

## 18.5.3.5 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), SCLx will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPxCON1<4>). See **Section 18.5.4 “Clock Stretching”** for more details.

## 18.5.3.6 Transmission

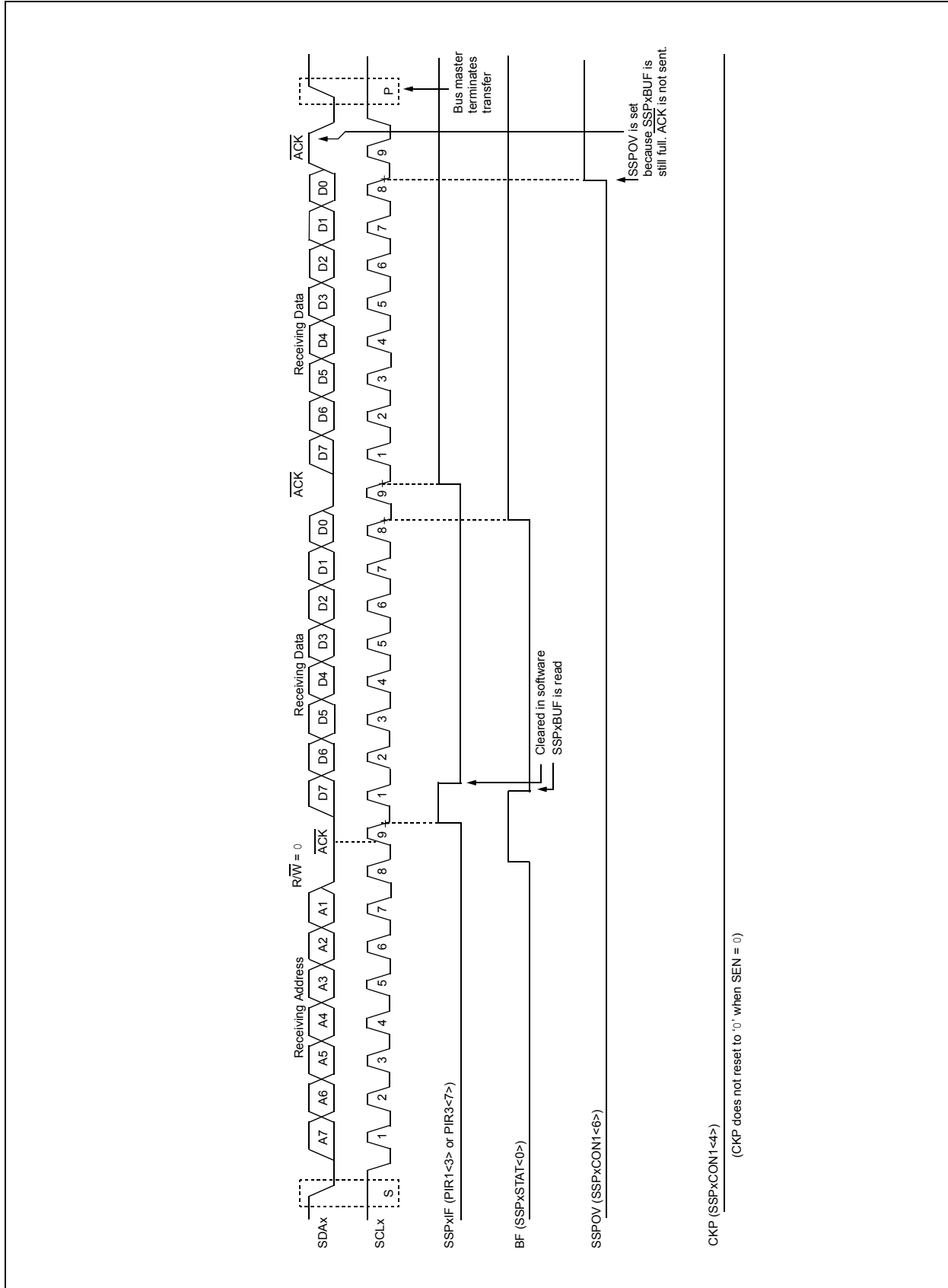
When the  $\overline{R/W}$  bit of the incoming address byte is set and an address match occurs, the  $\overline{R/W}$  bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register. The  $\overline{ACK}$  pulse will be sent on the ninth bit and pin SCLx is held low regardless of SEN (see **Section 18.5.4 “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, the SCLx pin 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 18-10).

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 the 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, the SCLx pin must be enabled by setting bit, CKP.

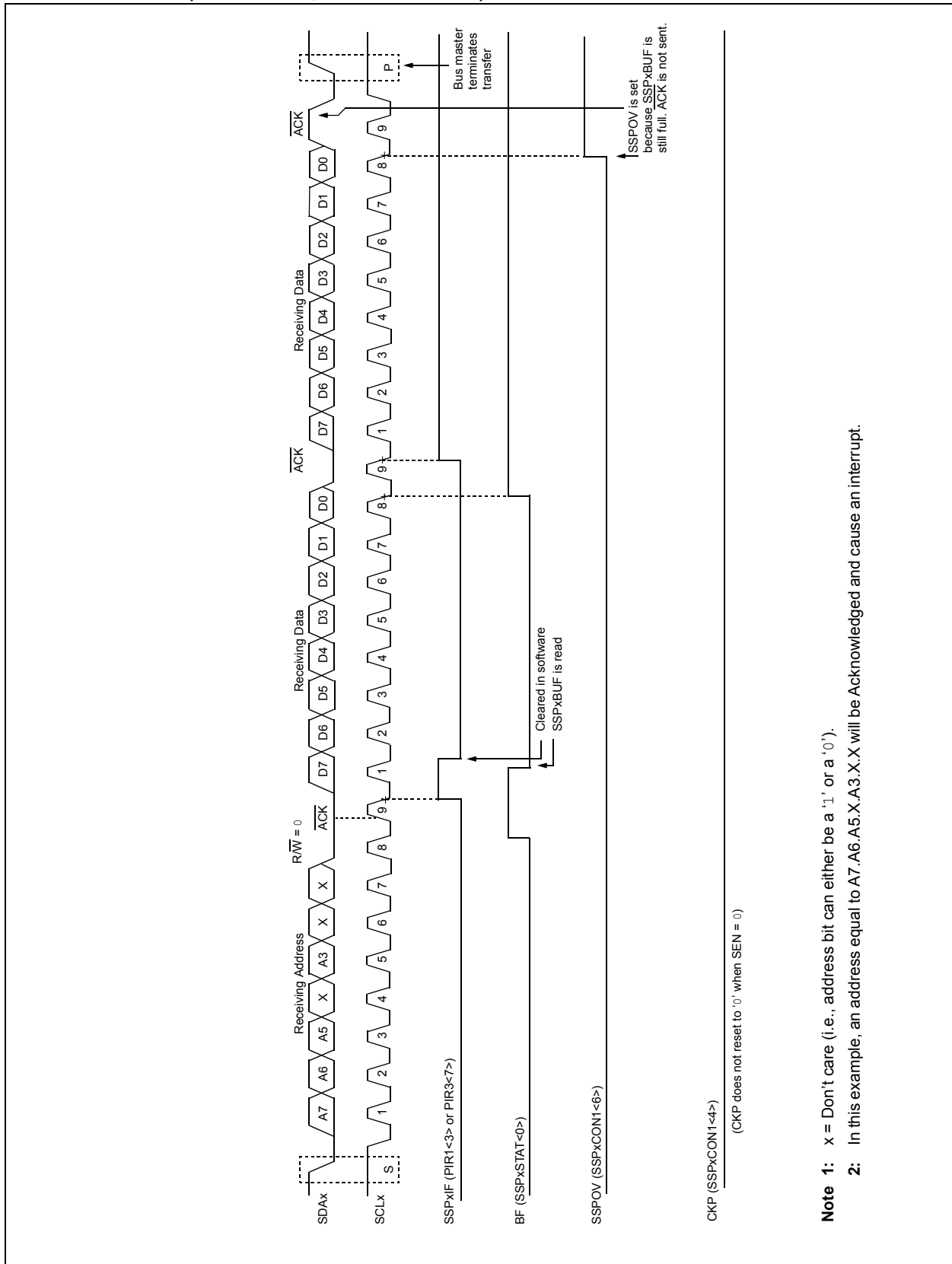
An MSSP interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared in software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

# PIC18F46J11 FAMILY

FIGURE 18-8: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)

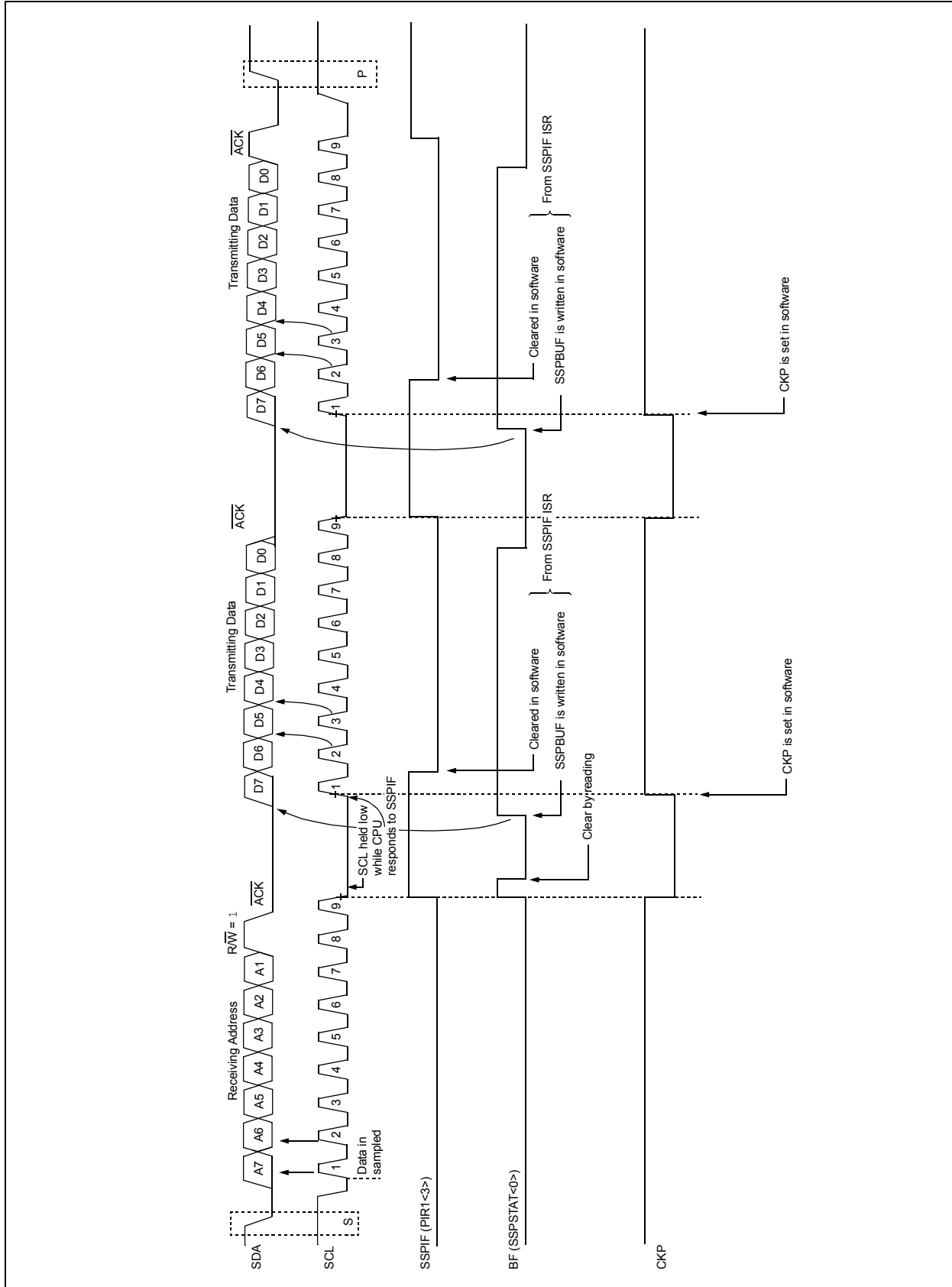


**FIGURE 18-9: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01011 (RECEPTION, 7-BIT ADDRESS)**



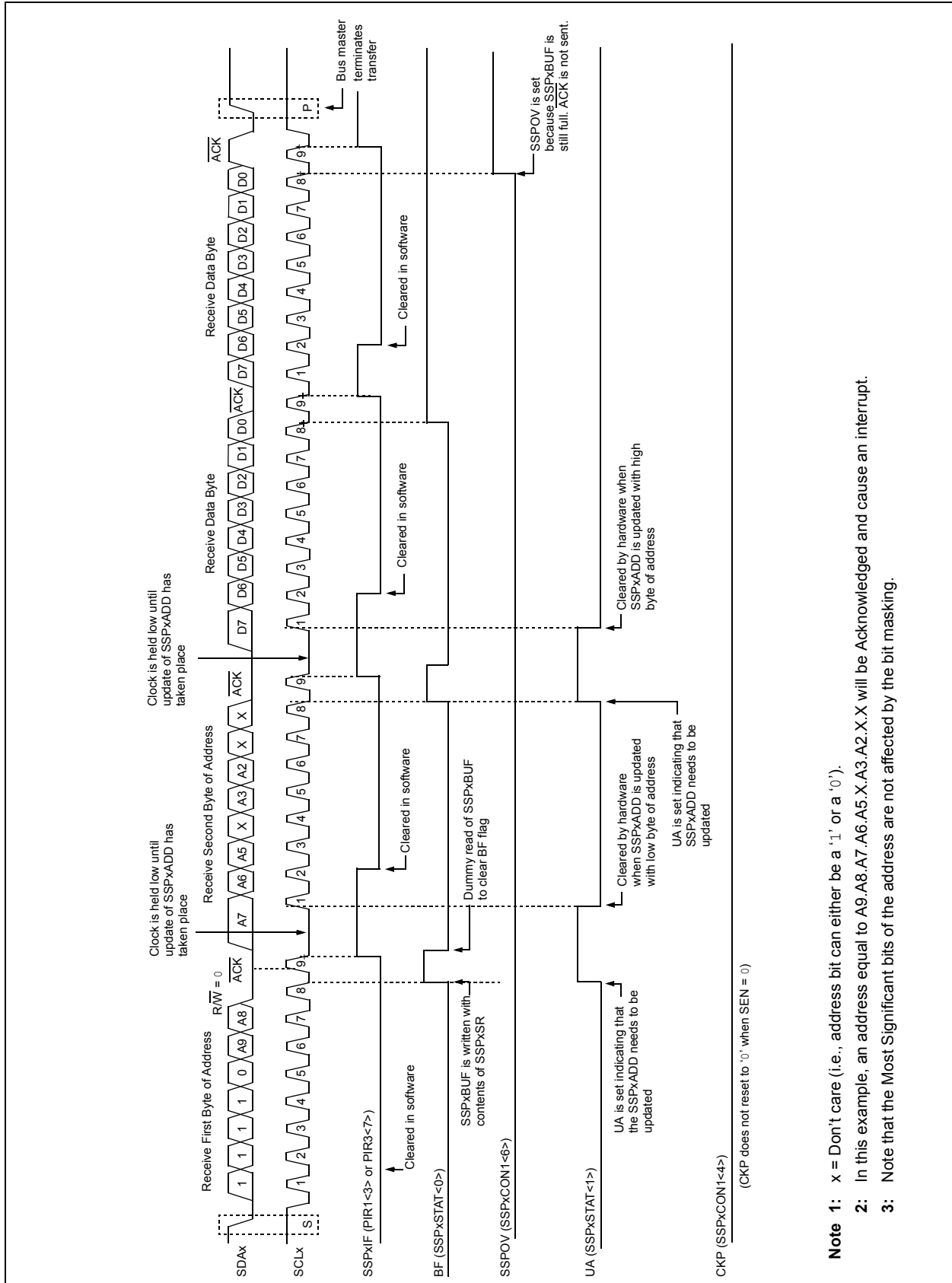
# PIC18F46J11 FAMILY

FIGURE 18-10: I<sup>2</sup>C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)





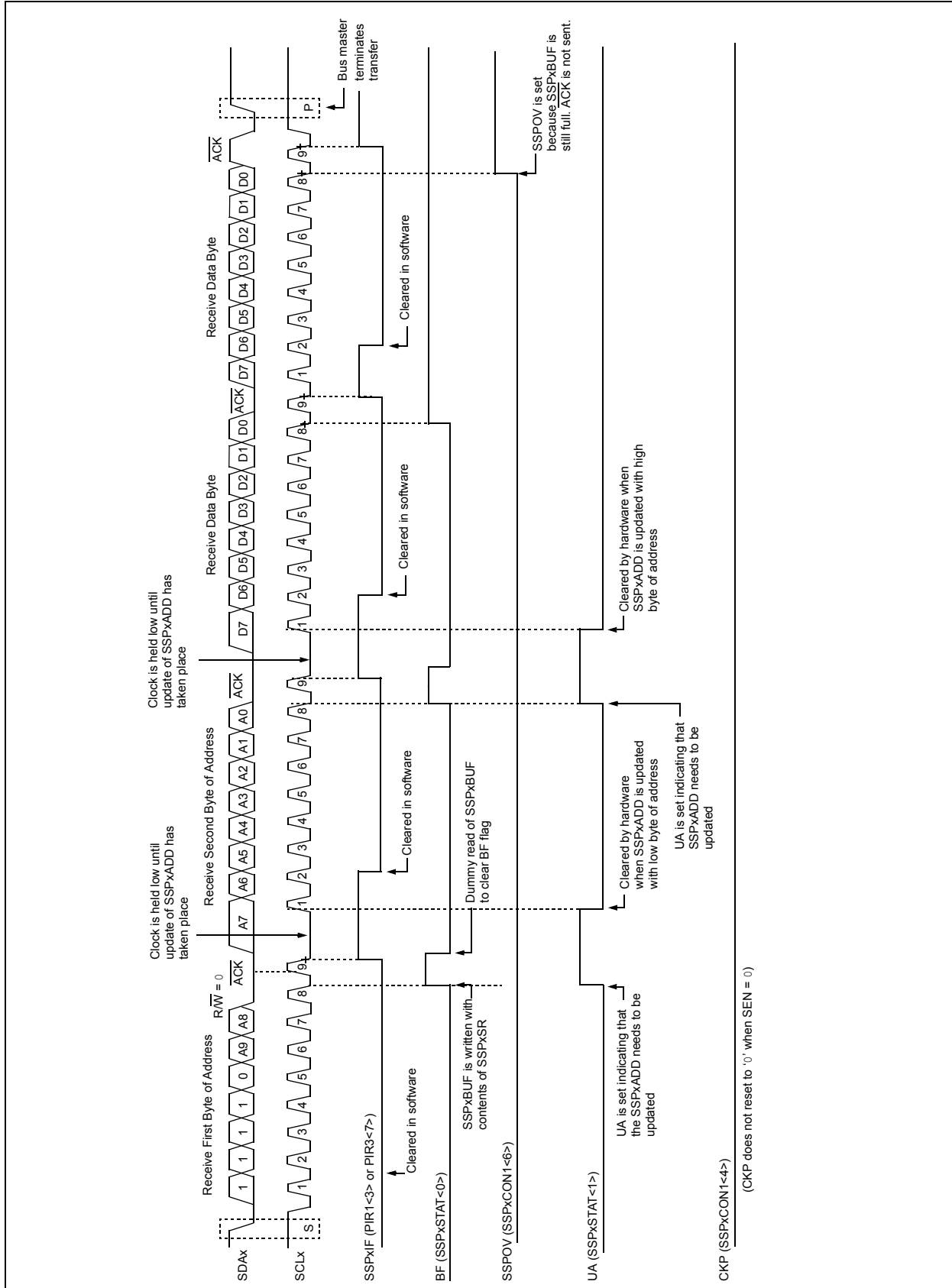
**FIGURE 18-11: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01001 (RECEPTION, 10-BIT ADDRESS)**



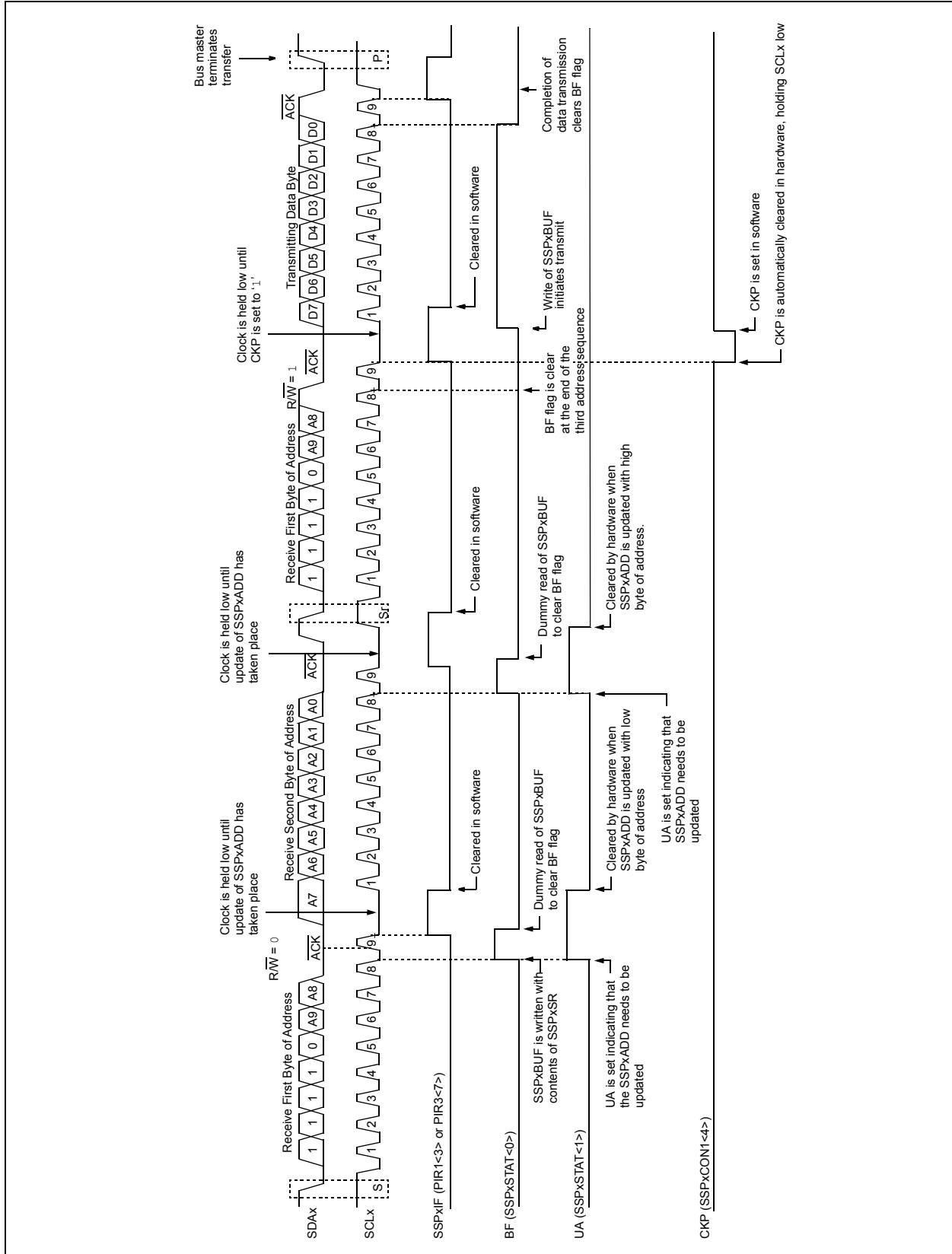
- Note 1:** x = Don't care (i.e., address bit can either be a '1' or a '0').
- 2:** In this example, an address equal to A9.A8.A7.A6.A5.X.A3.A2.X.X will be Acknowledged and cause an interrupt.
- 3:** Note that the Most Significant bits of the address are not affected by the bit masking.

# PIC18F46J11 FAMILY

FIGURE 18-12: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)



**FIGURE 18-13: I<sup>2</sup>C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)**



# PIC18F46J11 FAMILY

## 18.5.4 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.

### 18.5.4.1 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 bit 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 18-15).

**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.

### 18.5.4.2 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 has not 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.

### 18.5.4.3 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 Interrupt Service Routine (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 18-10).

**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.

### 18.5.4.4 Clock Stretching for 10-Bit Slave Transmit Mode

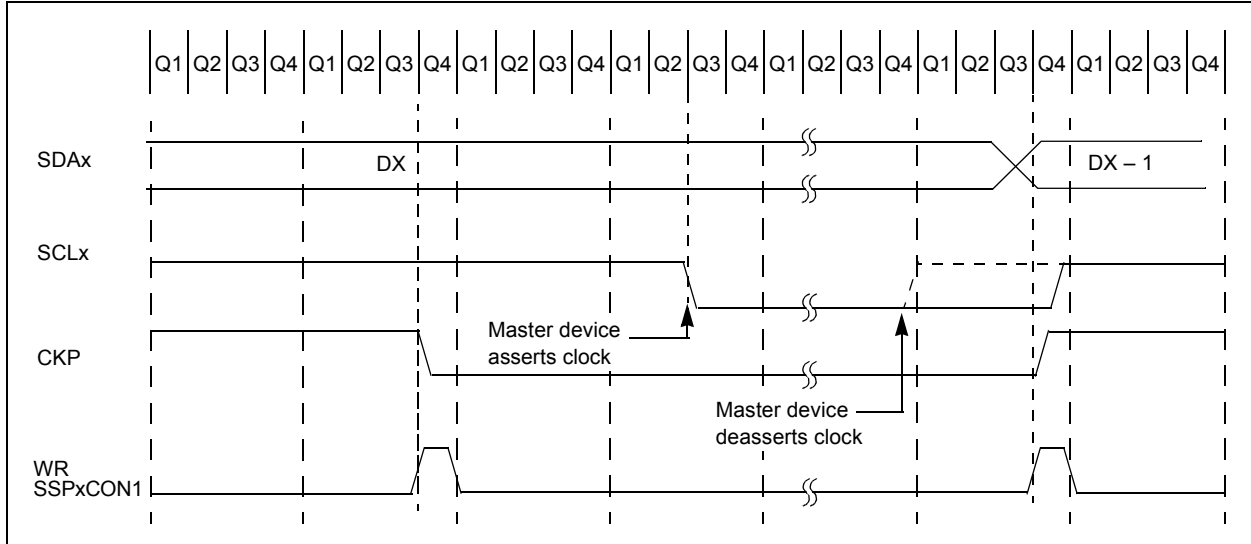
In 10-Bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-Bit Slave Receive mode. The first two addresses are followed by a third address sequence, which contains the high-order bits of the 10-bit address and the 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 18-13).

## 18.5.4.5 Clock Synchronization and 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 I<sup>2</sup>C master device has

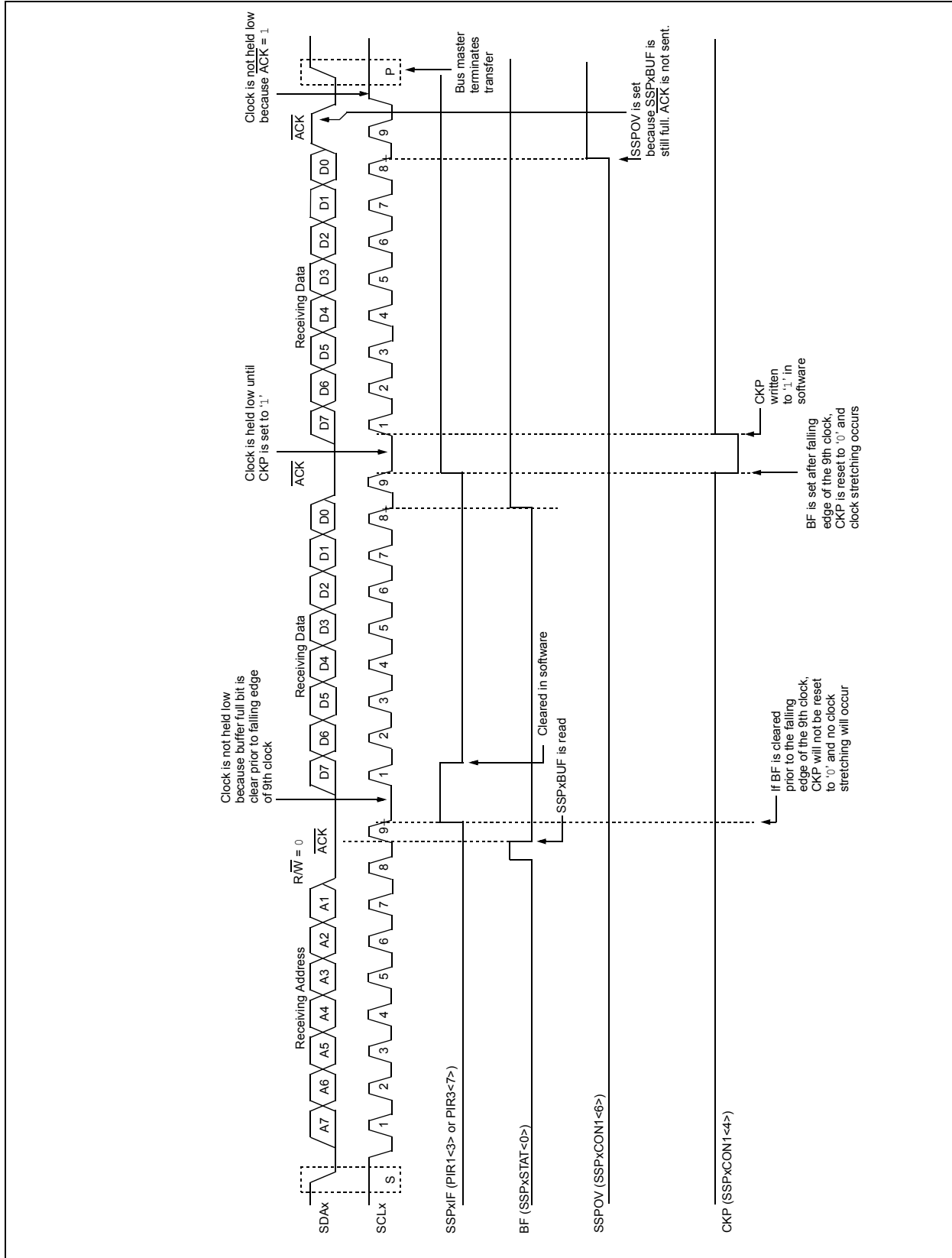
already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I<sup>2</sup>C bus have deasserted SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 18-14).

**FIGURE 18-14: CLOCK SYNCHRONIZATION TIMING**

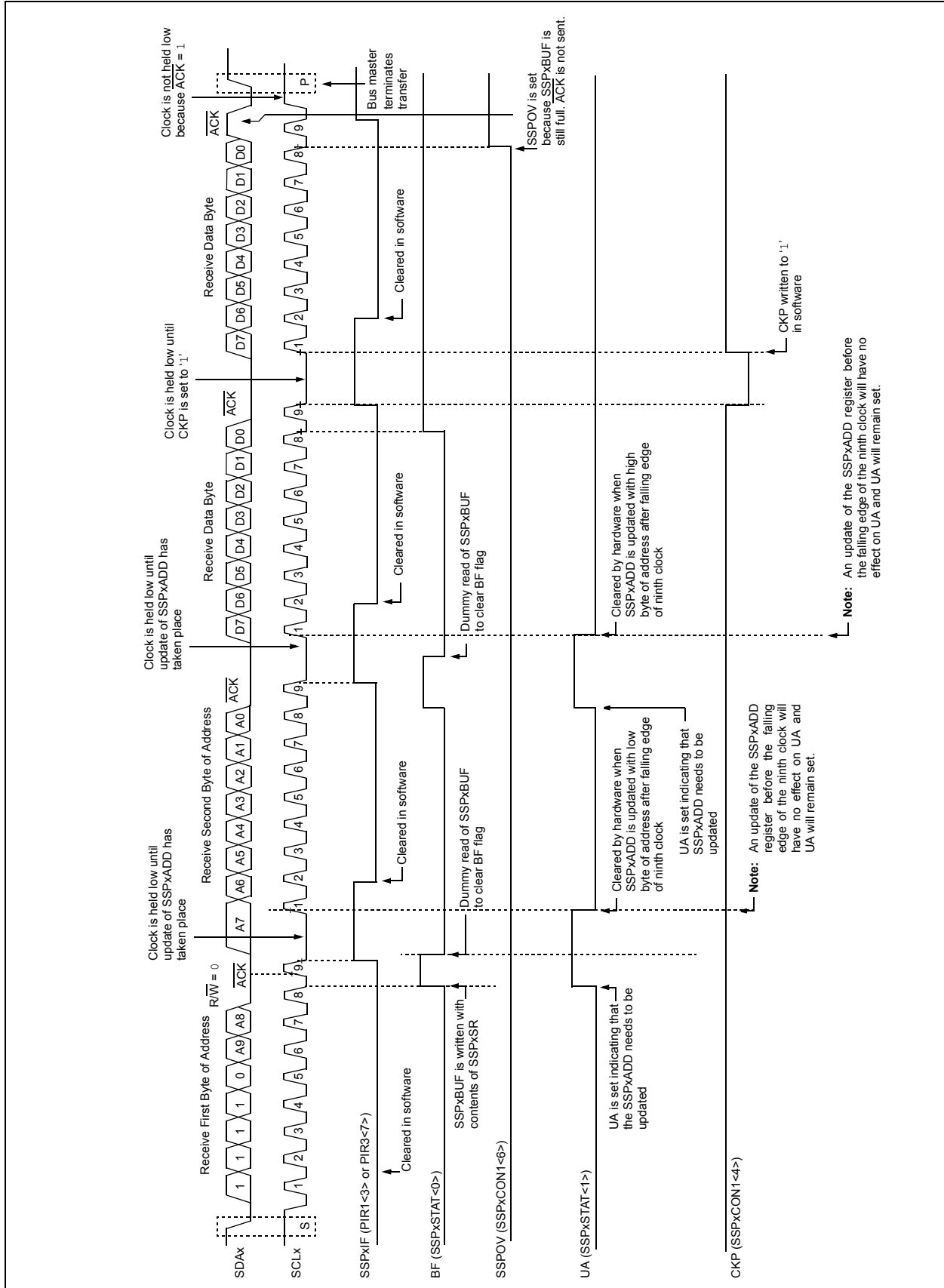


# PIC18F46J11 FAMILY

FIGURE 18-15: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)



**FIGURE 18-16: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)**



# PIC18F46J11 FAMILY

## 18.5.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I<sup>2</sup>C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I<sup>2</sup>C protocol. It consists of all '0's with R/W = 0.

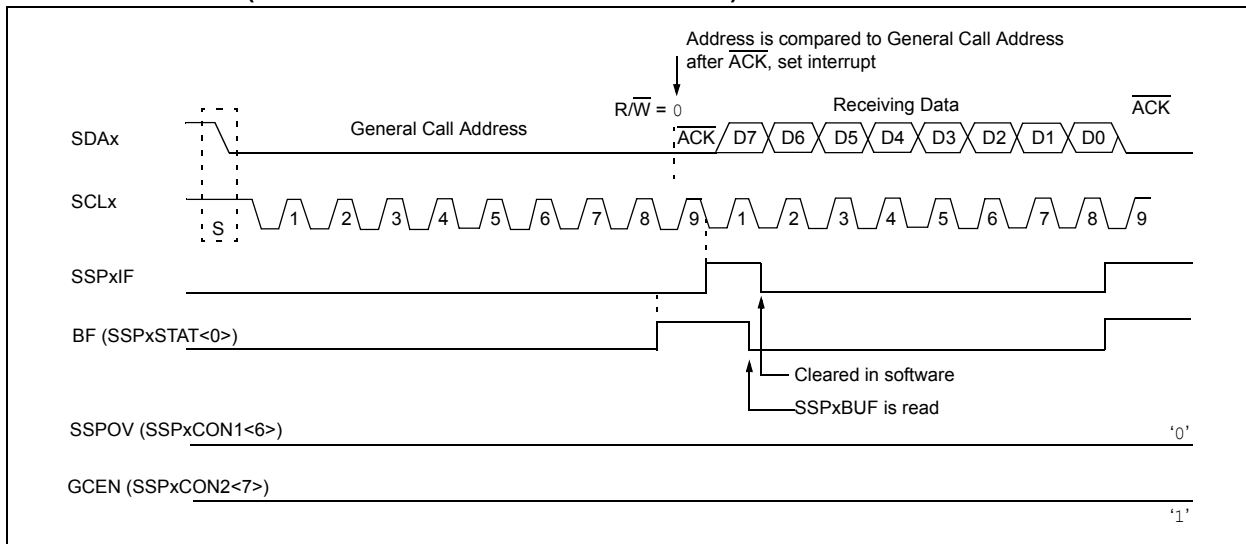
The general call address is recognized when the General Call Enable bit, GCEN, is enabled (SSPxCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPxSR and the address is compared against the SSPxADD. It is also compared to the general call address and fixed in hardware.

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 ( $\overline{\text{ACK}}$  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 Addressing 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 18-17).

**FIGURE 18-17: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7-BIT OR 10-BIT ADDRESSING MODE)**



## 18.5.6 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 if the TRIS bits are set.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Start (S) and Stop (P) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I<sup>2</sup>C bus may be taken when the Stop bit is set, or the bus is Idle, with both the Start and Stop bits clear.

In Firmware Controlled Master mode, user code conducts all I<sup>2</sup>C 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 I<sup>2</sup>C 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.

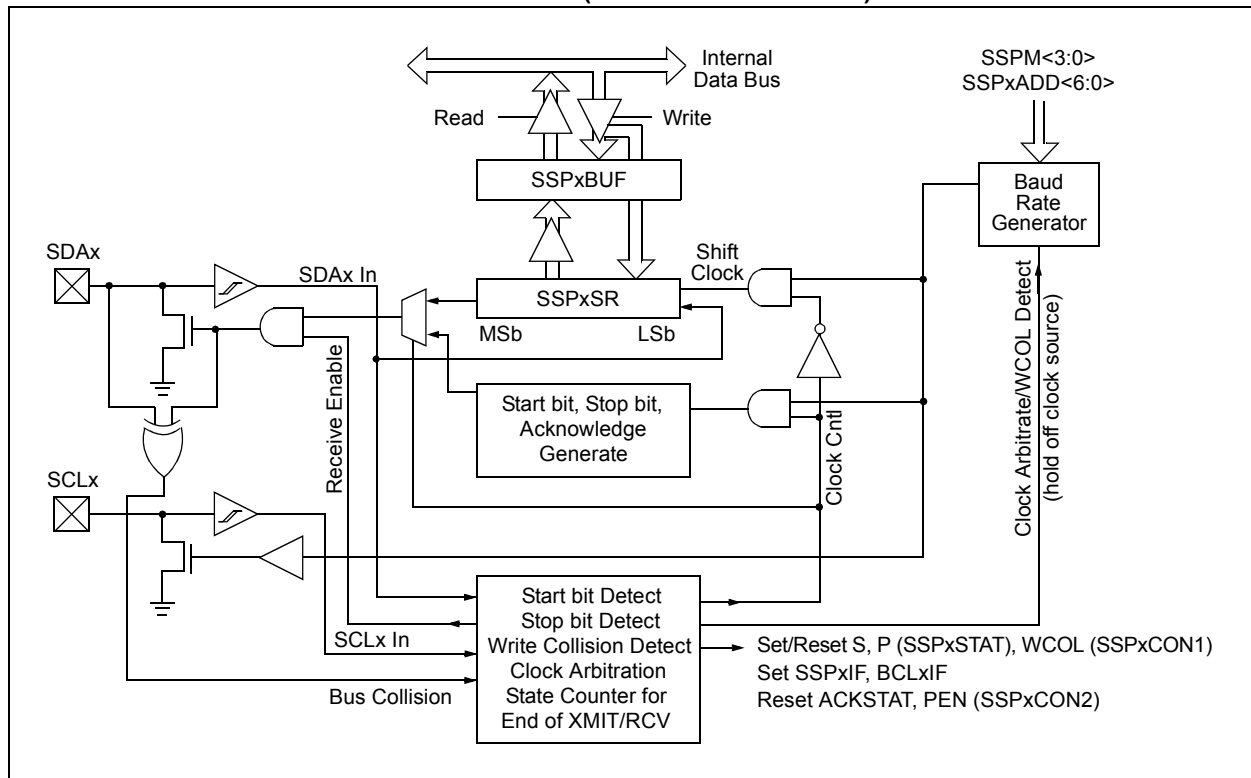


**Note:** The MSSP module, when configured in I<sup>2</sup>C Master mode, does not allow queuing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the 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 transmitted
- Repeated Start

**FIGURE 18-18: MSSPx BLOCK DIAGRAM (I<sup>2</sup>C™ MASTER MODE)**



## 18.5.6.1 I<sup>2</sup>C 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 I<sup>2</sup>C bus will not be released.

In Master Transmitter mode, serial data is output through SDAx while SCLx 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. S and P 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 SDAx, while SCLx outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. S and P conditions indicate the beginning and end of transmission.

The BRG, used for the SPI mode operation, is used to set the SCLx clock frequency for either 100 kHz, 400 kHz or 1 MHz I<sup>2</sup>C operation. See **Section 18.5.7 "Baud Rate"** for more details.

# PIC18F46J11 FAMILY

---

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 for 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 of the SDAx 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 8 bits of data.
8. Data is shifted out the SDAx 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.

## 18.5.7 BAUD RATE

In I<sup>2</sup>C Master mode, the BRG reload value is placed in the lower seven bits of the SSPxADD register (Figure 18-19). 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 (T<sub>CY</sub>) on the Q2 and Q4 clocks. In I<sup>2</sup>C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by  $\overline{ACK}$ ), the internal clock will automatically stop counting and the SCLx pin will remain in its last state.

Table 18-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

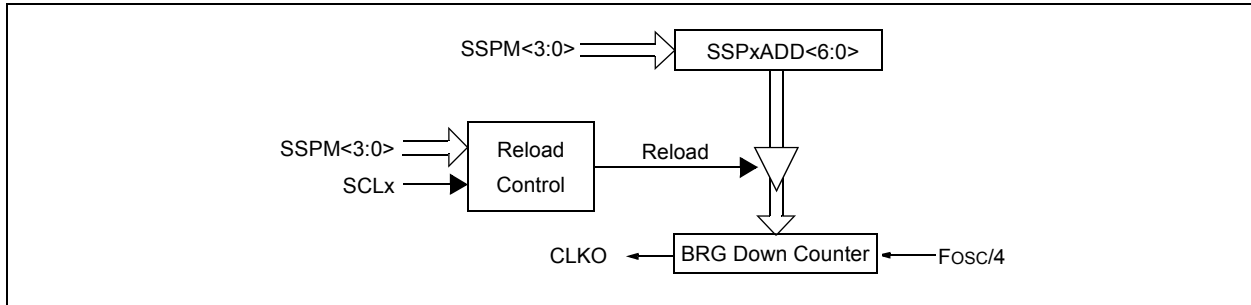
# PIC18F46J11 FAMILY

## 18.5.7.1 Baud Rate and Module Interdependence

Because MSSP1 and MSSP2 are independent, they can operate simultaneously in I<sup>2</sup>C 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 18-19: BAUD RATE GENERATOR BLOCK DIAGRAM**



**TABLE 18-3: I<sup>2</sup>C™ CLOCK RATE w/BRG**

Fosc	Fcy	Fcy * 2	BRG Value	FsCL (2 Rollovers of BRG)
40 MHz	10 MHz	20 MHz	18h	400 kHz <sup>(1)</sup>
40 MHz	10 MHz	20 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	20 MHz	63h	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz <sup>(1)</sup>
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz <sup>(1)</sup>
4 MHz	1 MHz	2 MHz	09h	100 kHz
4 MHz	1 MHz	2 MHz	00h	1 MHz <sup>(1)</sup>

**Note 1:** The I<sup>2</sup>C interface does not conform to the 400 kHz I<sup>2</sup>C 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.

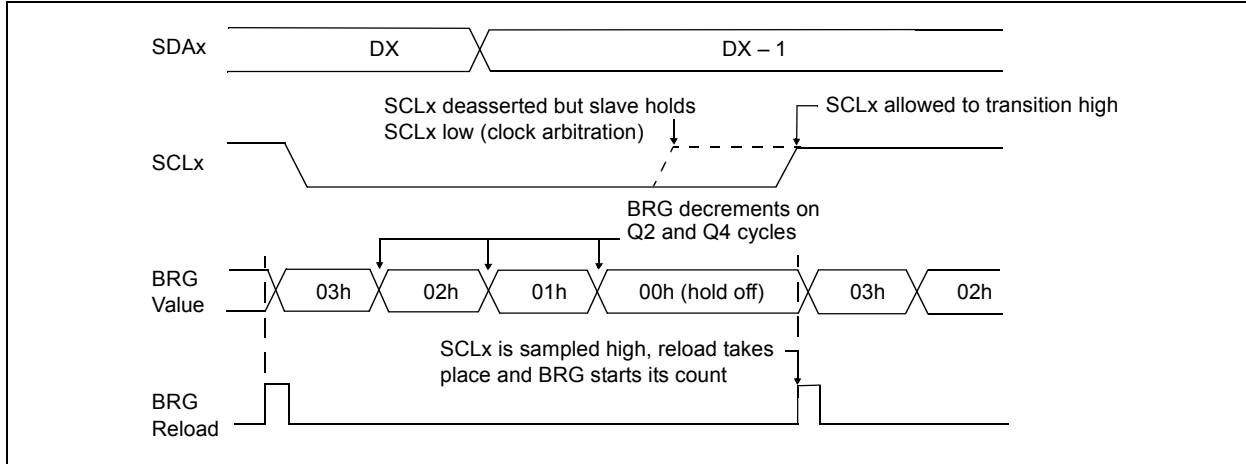
# PIC18F46J11 FAMILY

## 18.5.7.2 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 BRG is suspended from counting until the SCLx pin is actually

sampled high. When the SCLx pin is sampled high, the BRG 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 18-20).

**FIGURE 18-20: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**



## 18.5.8 I<sup>2</sup>C 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 BRG 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 Start bit (SSPxSTAT<3>) to be set. Following this, the BRG is reloaded with the contents of SSPxADD<6:0> and resumes its count. When the BRG times out (TBRG), the SEN bit (SSPxCON2<0>) will be automatically cleared by hardware. The BRG is suspended, leaving the SDAx line held low and the Start condition is complete.

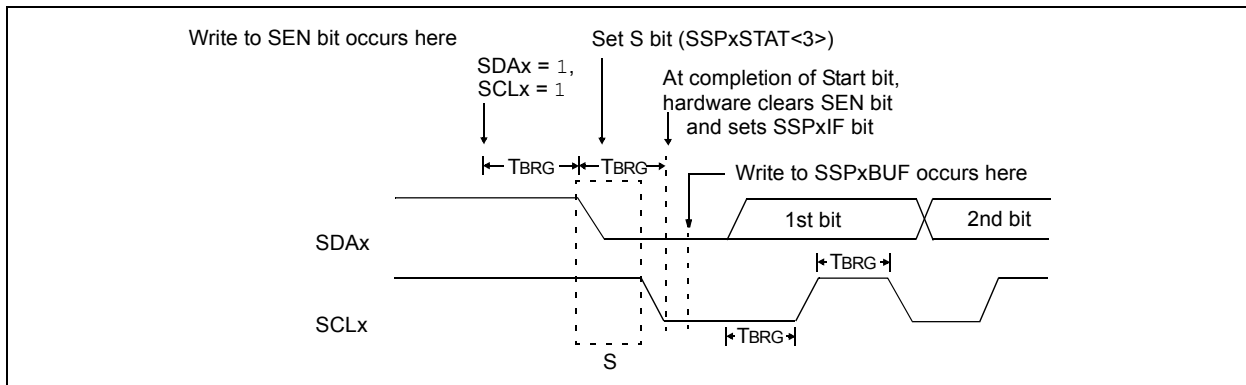
**Note:** If, at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low or if during the Start condition, the SCLx line is sampled low, before the SDAx line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I<sup>2</sup>C module is reset into its Idle state.

### 18.5.8.1 WCOL Status Flag

If the user writes the SSPxBUF when a Start sequence is in progress, the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

**Note:** Because queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete.

**FIGURE 18-21: FIRST START BIT TIMING**



## 18.5.9 I<sup>2</sup>C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPxCON2<1>) is programmed high and the I<sup>2</sup>C 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 BRG is loaded with the contents of SSPxADD<5:0> and begins counting. The SDAx pin is released (brought high) for one BRG count (TBRG). When the BRG times out, and if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the BRG 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 BRG 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 Start bit (SSPxSTAT<3>) will be set. The SSPxIF bit will not be set until the BRG 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'.

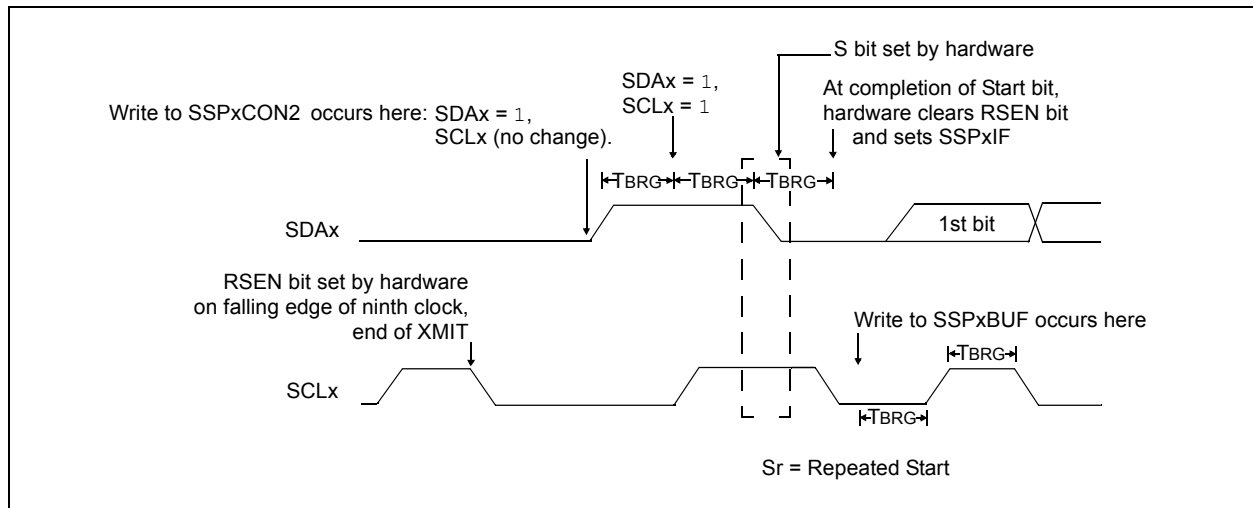
Immediately following the SSPxIF bit getting set, the user may write the SSPxBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional 8 bits of address (10-bit mode) or 8 bits of data (7-bit mode).

### 18.5.9.1 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 does not occur).

**Note:** Because queueing of events is not allowed, writing of the lower five bits of SSPxCON2 is disabled until the Repeated Start condition is complete.

**FIGURE 18-22: REPEATED START CONDITION WAVEFORM**



# PIC18F46J11 FAMILY

## 18.5.10 I<sup>2</sup>C 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 BRG 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 BRG 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  $\overline{\text{ACK}}$  bit during the ninth bit time if an address match occurred, or if data was received properly. The status of  $\overline{\text{ACK}}$  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 (BRG) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 18-23).

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  $\overline{\text{ACK}}$  bit is loaded into the ACKSTAT status bit (SSPxCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPxIF flag is set, the BF flag is cleared and the BRG is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

### 18.5.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPxSTAT<0>) is set when the CPU writes to SSPxBUF and is cleared when all eight bits are shifted out.

### 18.5.10.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur) after 2 T<sub>CY</sub> after the SSPxBUF write. If SSPxBUF is rewritten within 2 T<sub>CY</sub>, the WCOL bit is set and SSPxBUF is updated. This may result in a corrupted transfer.

The user should verify that the WCOL bit is clear after each write to SSPxBUF to ensure the transfer is correct. In all cases, WCOL must be cleared in software.

### 18.5.10.3 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.

## 18.5.11 I<sup>2</sup>C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPxCON2<3>).

<b>Note:</b> The MSSP module must be in an inactive state before the RCEN bit is set or the RCEN bit will be disregarded.
---

The BRG 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 BRG 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>).

### 18.5.11.1 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.

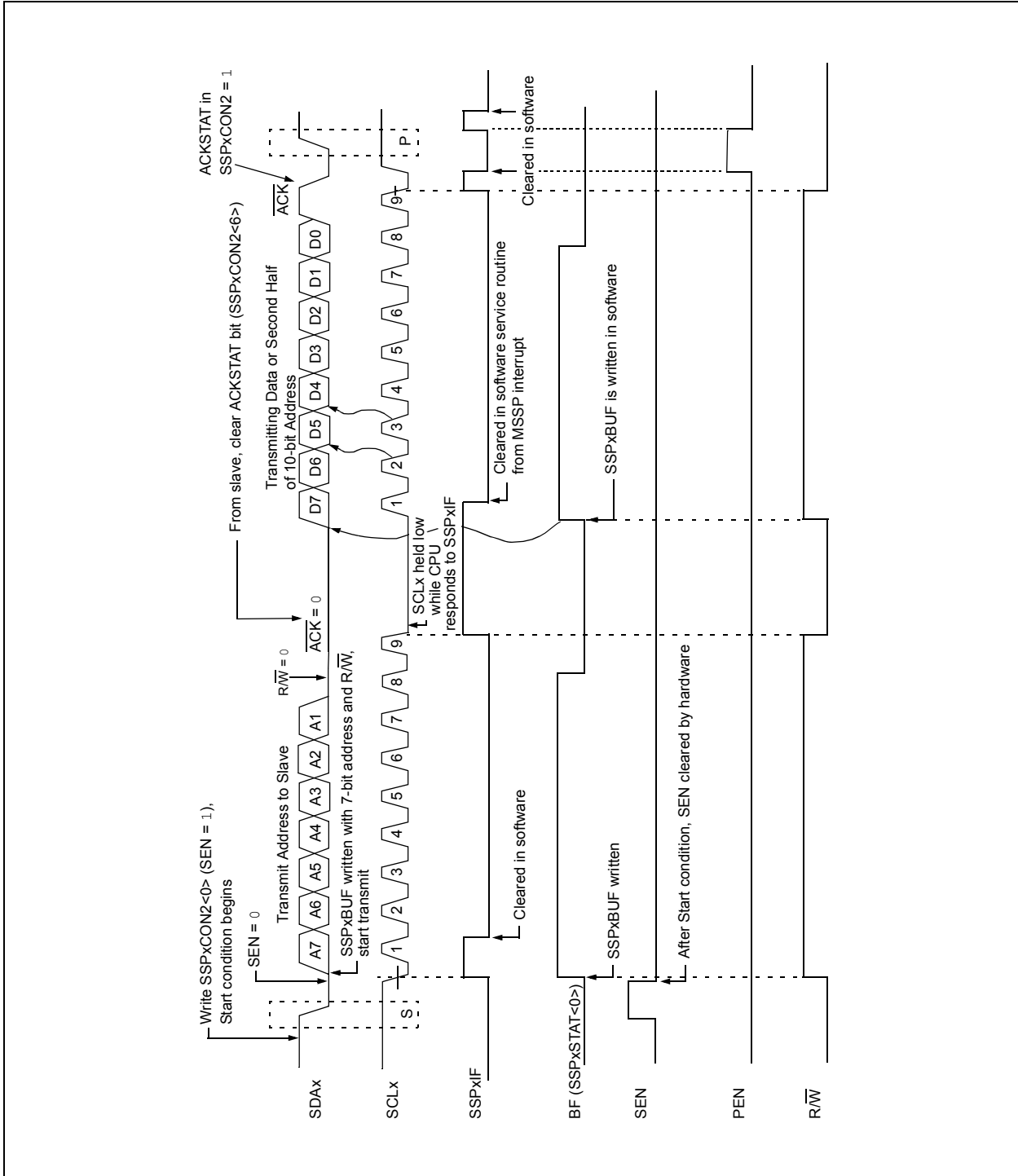
### 18.5.11.2 SSPOV Status Flag

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.

### 18.5.11.3 WCOL Status Flag

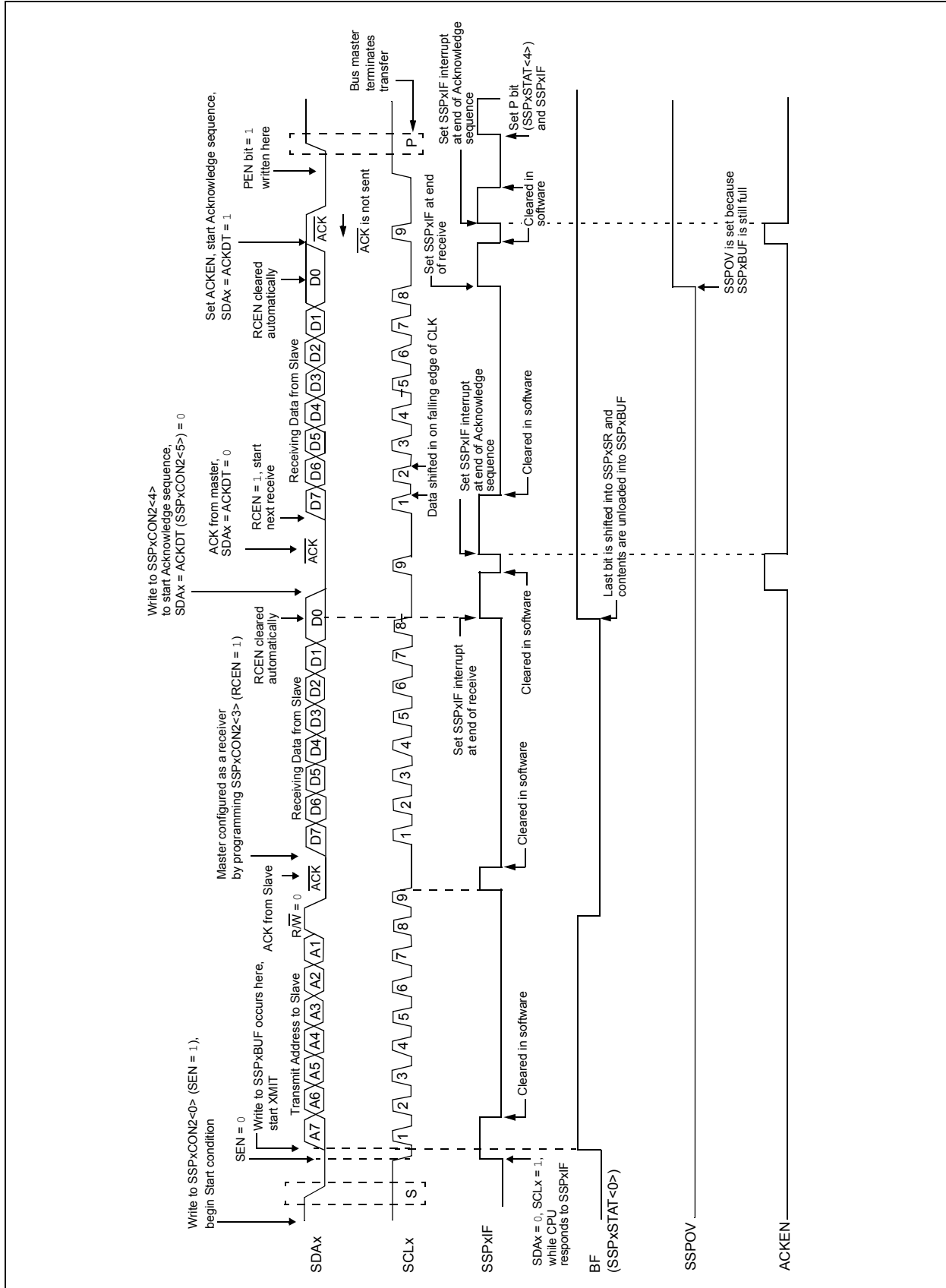
If users write the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

**FIGURE 18-23: I<sup>2</sup>C™ MASTER MODE WAVEFORM (TRANSMISSION, 7-BIT OR 10-BIT ADDRESS)**



# PIC18F46J11 FAMILY

**FIGURE 18-24: I<sup>2</sup>C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)**





## 18.5.12 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 BRG 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 BRG counts for TBRG; the SCLx pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the BRG is turned off and the MSSP module then goes into an inactive state (Figure 18-25).

### 18.5.12.1 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 does not occur).

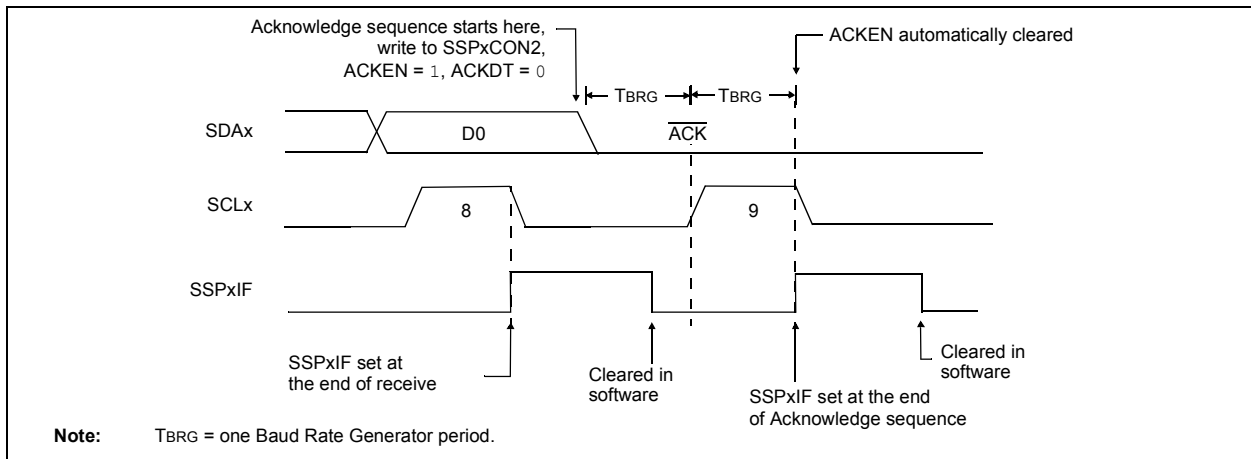
## 18.5.13 STOP CONDITION TIMING

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 BRG is reloaded and counts down to 0. When the BRG times out, the SCLx pin will be brought high and one Baud Rate Generator rollover count (TBRG) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the Stop bit (SSPxSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 18-26).

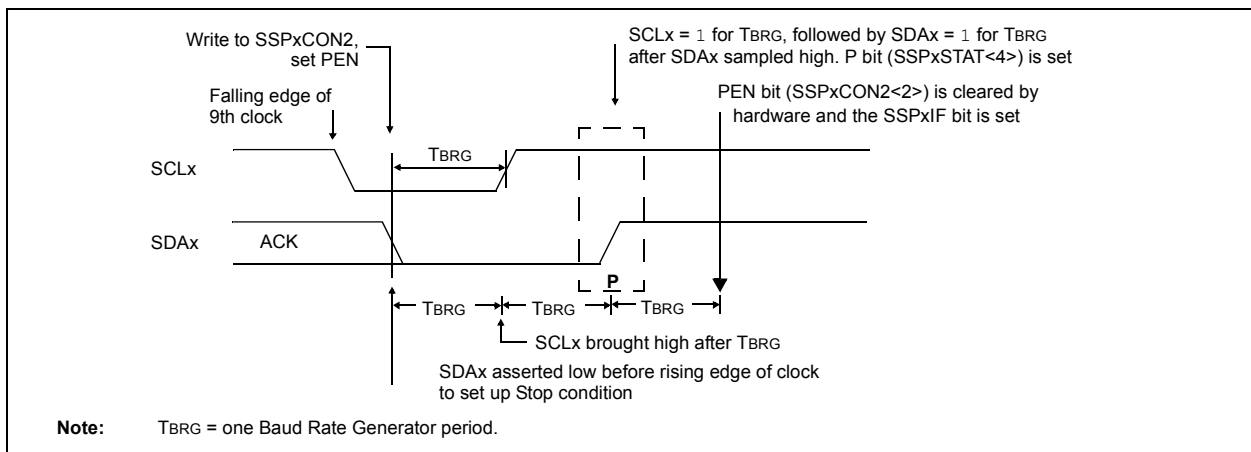
### 18.5.13.1 WCOL Status Flag

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 does not occur).

**FIGURE 18-25: ACKNOWLEDGE SEQUENCE WAVEFORM**



**FIGURE 18-26: STOP CONDITION RECEIVE OR TRANSMIT MODE**



# PIC18F46J11 FAMILY

## 18.5.14 SLEEP OPERATION

While in Sleep mode, the I<sup>2</sup>C 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).

## 18.5.15 EFFECTS OF A RESET

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

## 18.5.16 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 Start and Stop bits are cleared from a Reset or when the MSSP module is disabled. Control of the I<sup>2</sup>C bus may be taken when the P bit (SSPxSTAT<4>) is set, or the bus is Idle, with both the Start and Stop 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

## 18.5.17 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 out-

puts 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 I<sup>2</sup>C port to its Idle state (Figure 18-27).

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 I<sup>2</sup>C bus is free, the user can resume communication by asserting a Start condition.

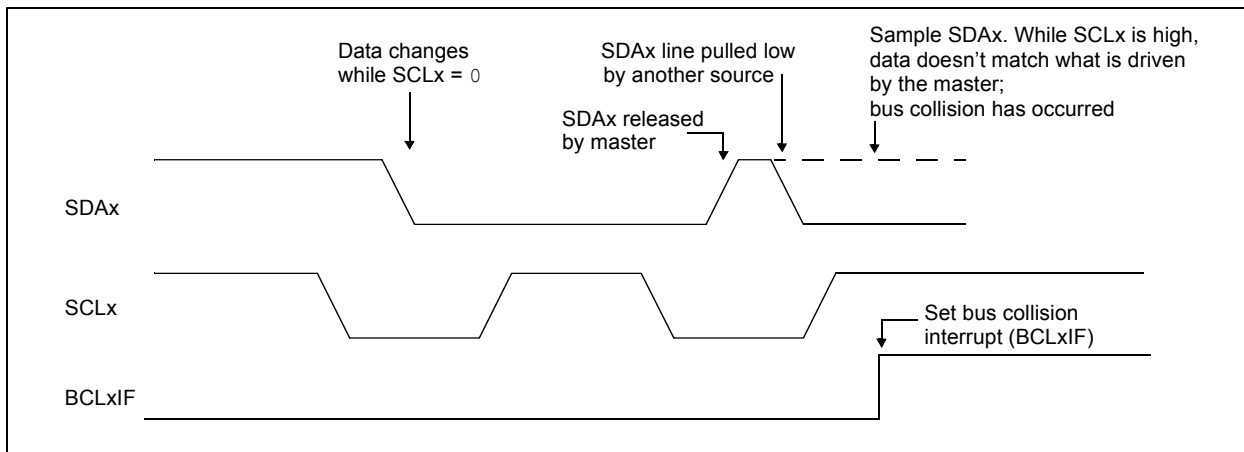
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine (ISR), and if the I<sup>2</sup>C 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 I<sup>2</sup>C bus can be taken when the Stop bit is set in the SSPxSTAT register, or the bus is Idle and the Start and Stop bits are cleared.

**FIGURE 18-27: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**



## 18.5.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

- SDAx or SCLx is sampled low at the beginning of the Start condition (Figure 18-28).
- SCLx is sampled low before SDAx is asserted low (Figure 18-29).

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:

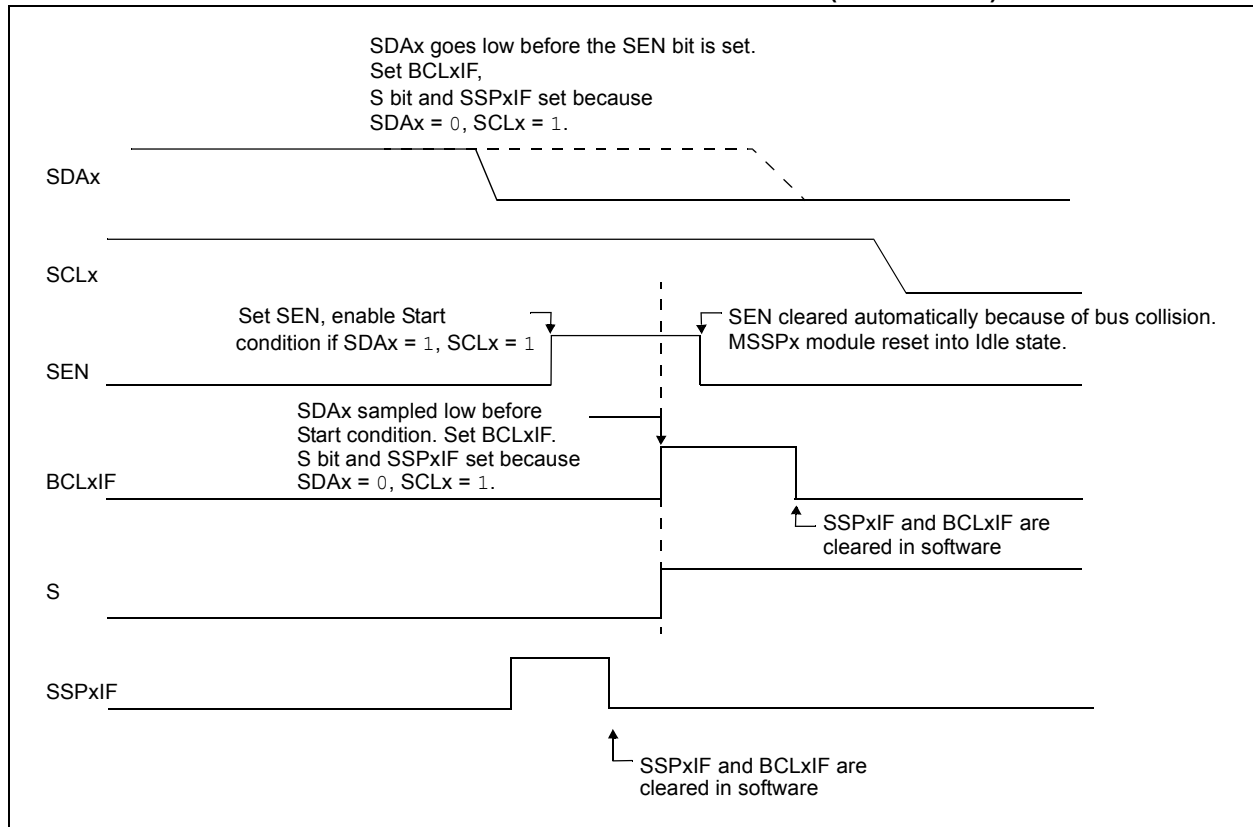
- The Start condition is aborted
- The BCLxIF flag is set
- The MSSP module is reset to its inactive state (Figure 18-28)

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the BRG 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 18-30). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The BRG 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.

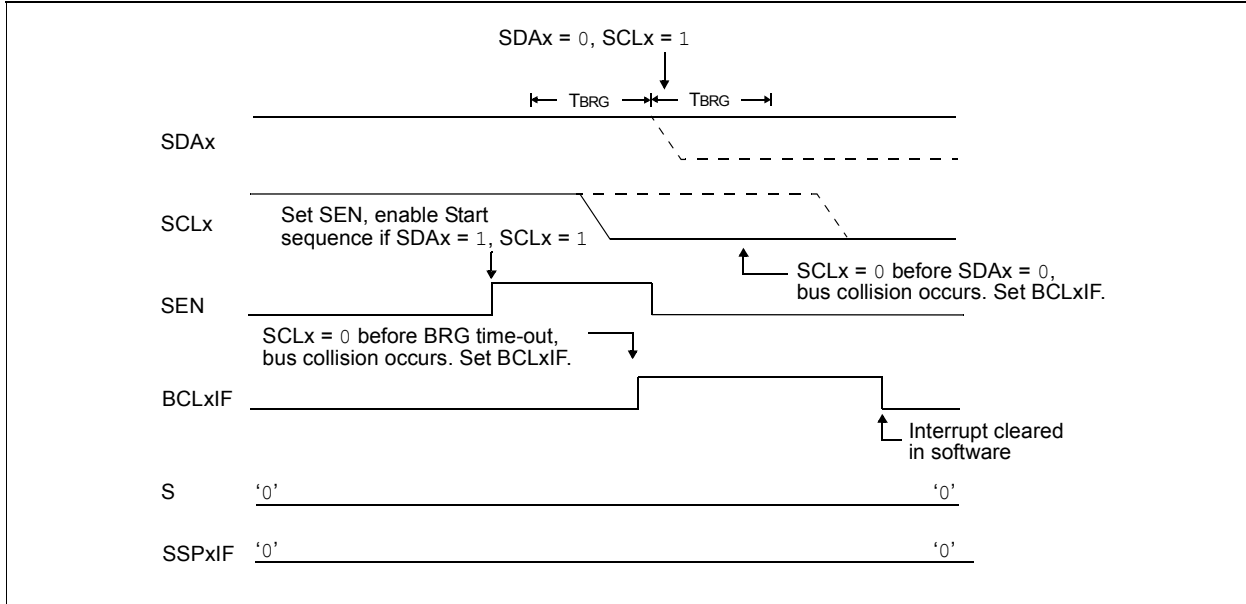
**Note:** The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be 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 18-28: BUS COLLISION DURING START CONDITION (SDAx ONLY)**

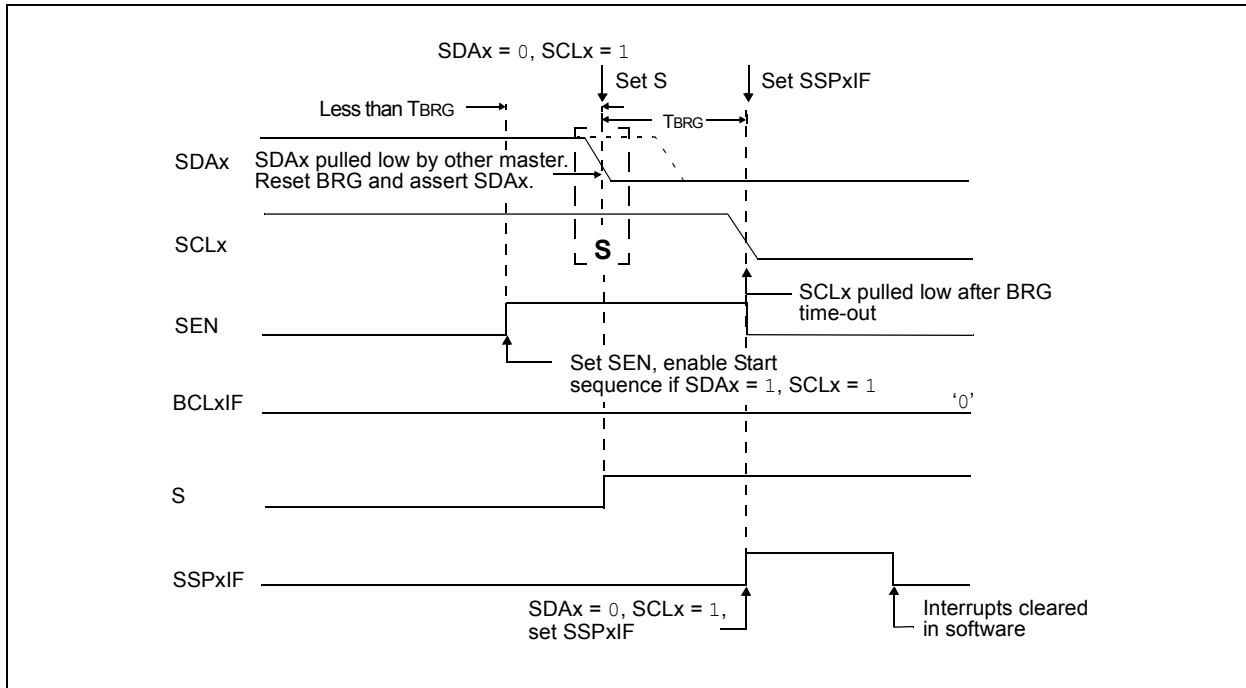


# PIC18F46J11 FAMILY

**FIGURE 18-29: BUS COLLISION DURING START CONDITION (SCLx = 0)**



**FIGURE 18-30: BRG RESET DUE TO SDAx ARBITRATION DURING START CONDITION**



## 18.5.17.2 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 a low level to a high level.
- SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1'.

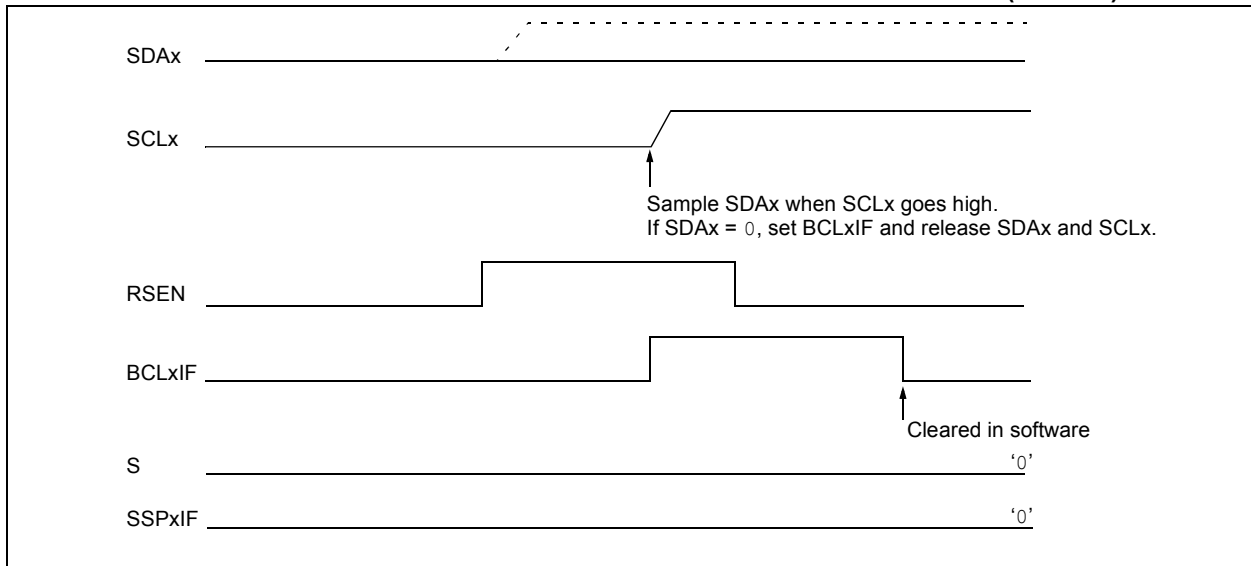
When the user deasserts SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD<6:0> and counts down to 0. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', see Figure 18-31). 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.

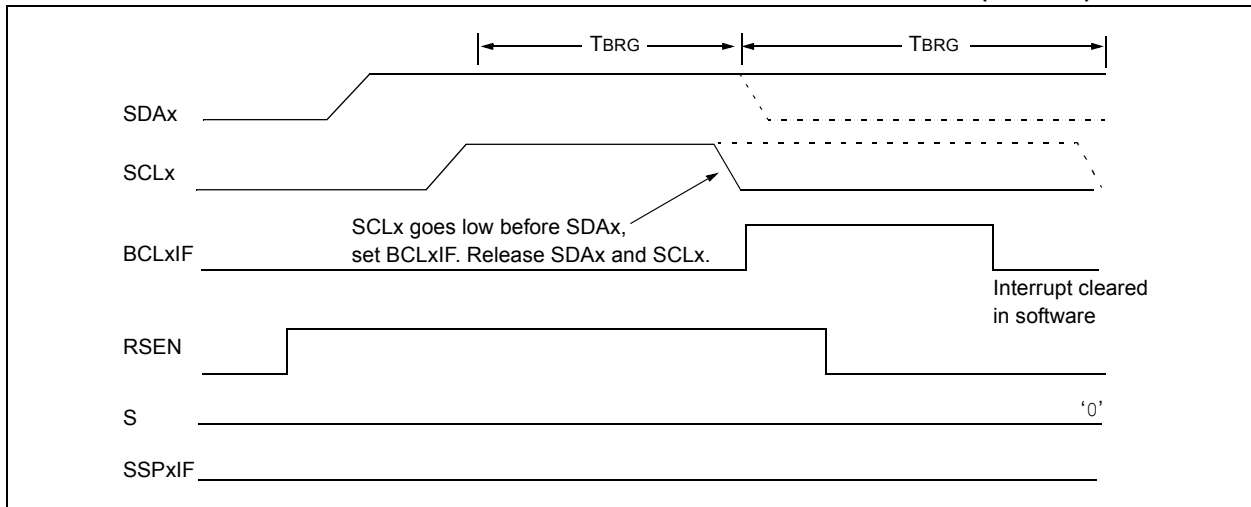
If SCLx goes from high-to-low before the BRG times out and SDAx has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 18-32).

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 18-31: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)**



**FIGURE 18-32: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)**



# PIC18F46J11 FAMILY

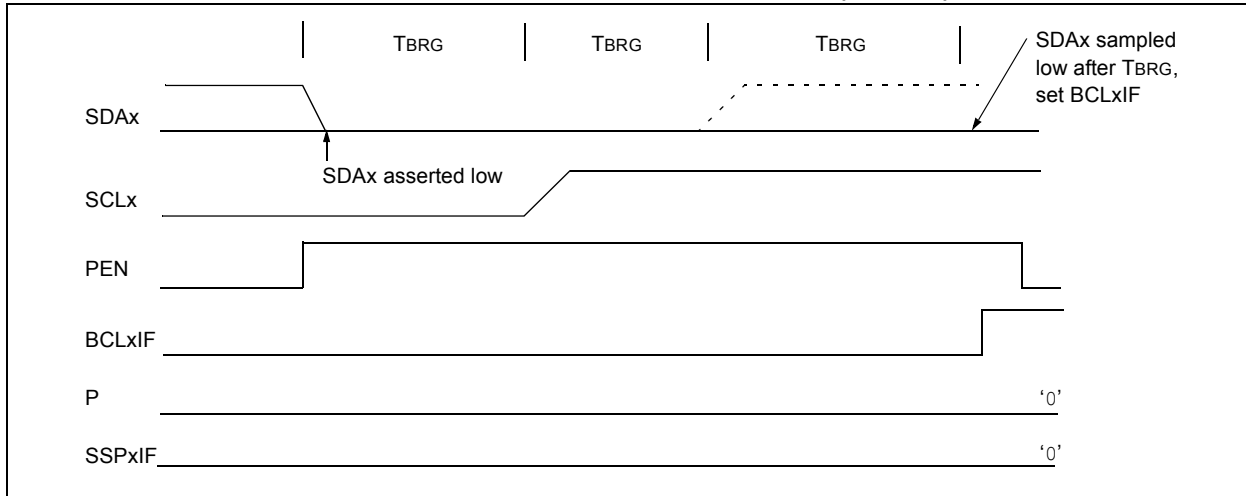
## 18.5.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

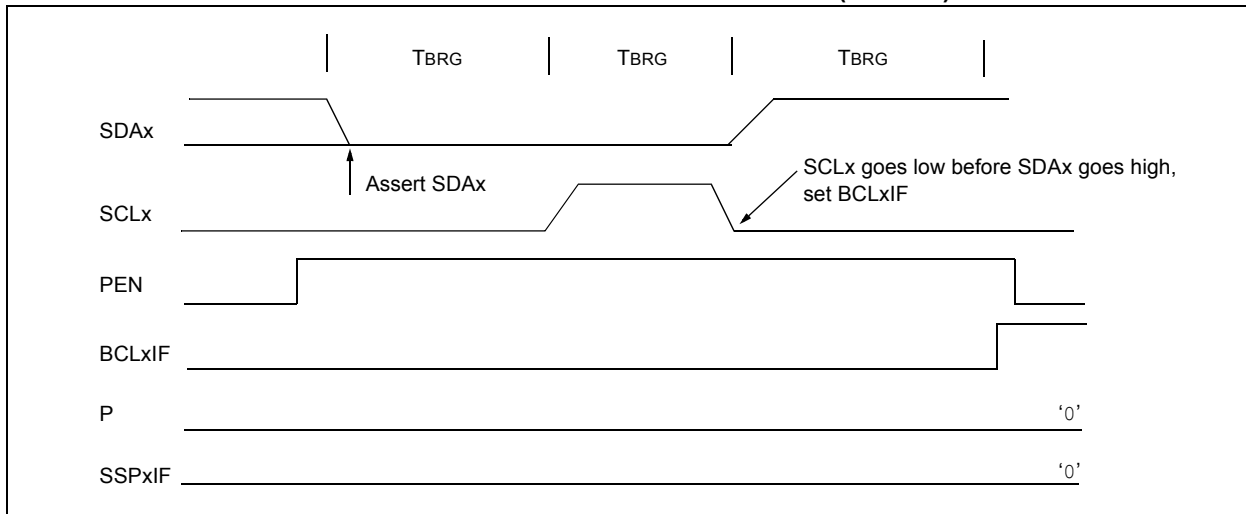
- a) After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out.
- b) After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high.

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 BRG 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 18-33). 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 18-34).

**FIGURE 18-33: BUS COLLISION DURING A STOP CONDITION (CASE 1)**



**FIGURE 18-34: BUS COLLISION DURING A STOP CONDITION (CASE 2)**



# PIC18F46J11 FAMILY

**TABLE 18-4: REGISTERS ASSOCIATED WITH I<sup>2</sup>C™ 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	63
PIR1	PMPIF <sup>(3)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(3)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPPIF <sup>(3)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	65
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	65
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCIP	65
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	66
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	66
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								64
SSPxADD	MSSP1 Address Register (I <sup>2</sup> C™ Slave mode), MSSP1 Baud Rate Reload Register (I <sup>2</sup> C Master mode)								64, 67
SSPxMSK <sup>(1)</sup>	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	64, 67
SSPxCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	64, 67
SSPxCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	64, 67
	GCEN	ACKSTAT	ADMSK5 <sup>(2)</sup>	ADMSK4 <sup>(2)</sup>	ADMSK3 <sup>(2)</sup>	ADMSK2 <sup>(2)</sup>	ADMSK1 <sup>(2)</sup>	SEN	
SSPxSTAT	SMP	CKE	D $\bar{A}$	P	S	R $\bar{W}$	UA	BF	64, 67
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								67
SSP2ADD	MSSP2 Address Register (I <sup>2</sup> C Slave mode), MSSP2 Baud Rate Reload Register (I <sup>2</sup> C Master mode)								67

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the MSSPx module in I<sup>2</sup>C™ mode.

- Note 1:** SSPxMSK shares the same address in SFR space as SSPxADD, but is only accessible in certain I<sup>2</sup>C Slave mode operations in 7-Bit Masking mode. See **Section 18.5.3.4 “7-Bit Address Masking Mode”** for more details.
- 2:** Alternate bit definitions for use in I<sup>2</sup>C Slave mode operations only.
- 3:** These bits are only available on 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:



## 19.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of two serial I/O modules. (Generically, the EUSART 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 and so on.

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.

All members of the PIC18F46J11 family are equipped with two independent EUSART modules, referred to as EUSART1 and EUSART2. They 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 EUSART1 and EUSART2 are multiplexed with the functions of PORTC (RC6/PMA5/TX1/CK1/RP17 and RC7/PMA4/RX1/DT1/SDO1/RP18) and remapped (RPn1/TX2/CK2 and RPn2/RX2/DT2), respectively. In order to configure these pins as an EUSART:

- For EUSART1:
  - SPEN bit (RCSTA1<7>) must be set (= 1)
  - TRISC<7> bit must be set (= 1)
  - TRISC<6> bit must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - TRISC<6> bit must be set (= 1) for Synchronous Slave mode
- For EUSART2:
  - SPEN bit (RCSTA2<7>) must be set (= 1)
  - TRIS bit for RPn2/RX2/DT2 = 1
  - TRIS bit for RPn1/TX2/CK2 = 0 for Asynchronous and Synchronous Master modes
  - TRISC<6> bit must be set (= 1) for Synchronous Slave mode

**Note:** The EUSART control will automatically reconfigure the pin from input to output as needed.

The TXx/CKx I/O pins have an optional open-drain output capability. By default, when this pin is used by the EUSART as an output, it will function as a standard push-pull CMOS output. The TXx/CKx I/O pins' open-drain, output feature can be enabled by setting the corresponding UxOD bit in the ODCON2 register. For more details, see **Section 18.3.3 "Open-Drain Output Option"**.

The operation of each Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTAx)
- Receive Status and Control (RCSTAx)
- Baud Rate Control (BAUDCONx)

These are covered in detail in Register 19-1, Register 19-2 and Register 19-3, respectively.

**Note:** Throughout this section, references to register and bit names that may be associated with a specific EUSART module are referred to generically by the use of 'x' in place of the specific module number. Thus, "RCSTAx" might refer to the Receive Status register for either EUSART1 or EUSART2.

# PIC18F46J11 FAMILY

**REGISTER 19-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER (ACCESS FADh/FA8h)**

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 <sup>(1)</sup>	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7						bit 0	

**Legend:**

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

- bit 7      **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<sup>(1)</sup>  
 1 = Transmit enabled  
 0 = Transmit disabled
- 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:** 9<sup>th</sup> bit of Transmit Data  
 Can be address/data bit or a parity bit.

**Note 1:** SREN/CREN overrides TXEN in Sync mode.

# PIC18F46J11 FAMILY

**REGISTER 19-2: RCSTAx: RECEIVE STATUS AND CONTROL REGISTER (ACCESS FACH/F9Ch)**

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

**Legend:**

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

- bit 7      **SPEN:** Serial Port Enable bit  
           1 = Serial port enabled (configures RXx/DTx and TXx/CKx 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 RCREGx 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:** 9<sup>th</sup> bit of Received Data  
           This can be address/data bit or a parity bit and must be calculated by user firmware.

# PIC18F46J11 FAMILY

## REGISTER 19-3: BAUDCONx: BAUD RATE CONTROL REGISTER (ACCESS F7Eh/F7Ch)

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

### Legend:

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

- bit 7      **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      **RXDTP:** Data/Receive Polarity Select bit  
Asynchronous mode:  
 1 = Receive data (RXx) is inverted (active-low)  
 0 = Receive data (RXx) is not inverted (active-high)  
Synchronous mode:  
 1 = Data (DTx) is inverted (active-low)  
 0 = Data (DTx) is not inverted (active-high)
- bit 4      **TXCKP:** Synchronous Clock Polarity Select bit  
Asynchronous mode:  
 1 = Idle state for transmit (TXx) is a low level  
 0 = Idle state for transmit (TXx) is a high level  
Synchronous mode:  
 1 = Idle state for clock (CKx) is a high level  
 0 = Idle state for clock (CKx) is a low level
- bit 3      **BRG16:** 16-Bit Baud Rate Register Enable bit  
 1 = 16-bit Baud Rate Generator – SPBRGHx and SPBRGx  
 0 = 8-bit Baud Rate Generator – SPBRGx only (Compatible mode), SPBRGHx value ignored
- bit 2      **Unimplemented:** Read as '0'
- bit 1      **WUE:** Wake-up Enable bit  
Asynchronous mode:  
 1 = EUSART will continue to sample the RXx pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge  
 0 = RXx 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.

## 19.1 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 (BAUDCONx<3>) selects 16-bit mode.

The SPBRGHx:SPBRGx register pair controls the period of a free-running timer. In Asynchronous mode, bits, BRGH (TXSTAx<2>) and BRG16 (BAUDCONx<3>), also control the baud rate. In Synchronous mode, BRGH is ignored.

Table 19-1 provides 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 SPBRGHx:SPBRGx registers can be calculated using the formulas in Table 19-1. From this, the error in baud rate can be determined. An example calculation is provided in Example 19-1. Typical baud rates and error values for the various Asynchronous modes are provided in Table 19-2. It may be advantageous 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 SPBRGHx:SPBRGx 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.

### 19.1.1 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 SPBRGx register pair.

### 19.1.2 SAMPLING

The data on the RXx pin (either RC7/PMA4/RX1/DT1/SDO1/RP18 or RPn2/RX2/DT2) is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RXx pin.

**TABLE 19-1: 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 SPBRGHx:SPBRGx register pair

# PIC18F46J11 FAMILY

## EXAMPLE 19-1: CALCULATING BAUD RATE ERROR

For a device with Fosc of 16 MHz, desired baud rate of 9600, Asynchronous mode, and 8-bit BRG:

$$\text{Desired Baud Rate} = \text{Fosc} / (64 ([\text{SPBRGHx}:\text{SPBRGx}] + 1))$$

Solving for SPBRGHx:SPBRGx:

$$\begin{aligned} X &= ((\text{Fosc}/\text{Desired Baud Rate})/64) - 1 \\ &= ((16000000/9600)/64) - 1 \\ &= [25.042] = 25 \end{aligned}$$

$$\begin{aligned} \text{Calculated Baud Rate} &= 16000000 / (64 (25 + 1)) \\ &= 9615 \end{aligned}$$

$$\begin{aligned} \text{Error} &= (\text{Calculated Baud Rate} - \text{Desired Baud Rate}) / \text{Desired Baud Rate} \\ &= (9615 - 9600) / 9600 = 0.16\% \end{aligned}$$

**TABLE 19-2: 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:
TXSTAx	CSRC	TX9	TXEN	SYNC	SEnDB	BRGH	TRMT	TX9D	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65

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

# PIC18F46J11 FAMILY

**TABLE 19-3: 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	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-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	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-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	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	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	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

# PIC18F46J11 FAMILY

**TABLE 19-3: 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	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	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	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-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	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-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	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—



## 19.1.3 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 19-1) 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 RXx signal, the RXx signal is timing the BRG. In ABD mode, the internal BRG 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 ABD 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 high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRGx begins counting up, using the preselected clock source on the first rising edge of RXx. After eight bits on the RXx pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGHx:SPBRGx register pair. Once the 5<sup>th</sup> 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 (BAUDCONx<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 19-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8<sup>th</sup> 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 SPBRGx and SPBRGHx 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 SPBRGHx register.

Refer to Table 19-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSART state machine is held in Idle. The RCxIF interrupt is set once the fifth rising edge on RXx is detected. The value in the RCREGx needs to be read to clear the RCxIF interrupt. The contents of RCREGx 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 19-4: 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

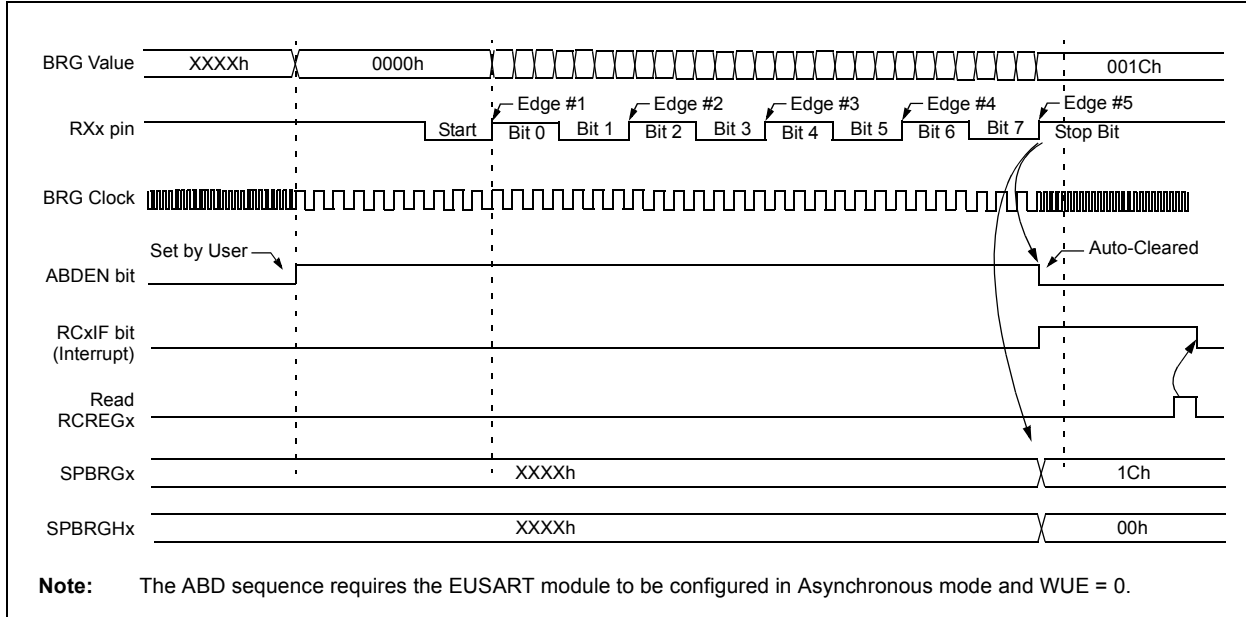
**Note:** During the ABD sequence, SPBRGx and SPBRGHx are both used as a 16-bit counter, independent of BRG16 setting.

### 19.1.3.1 ABD and EUSART Transmission

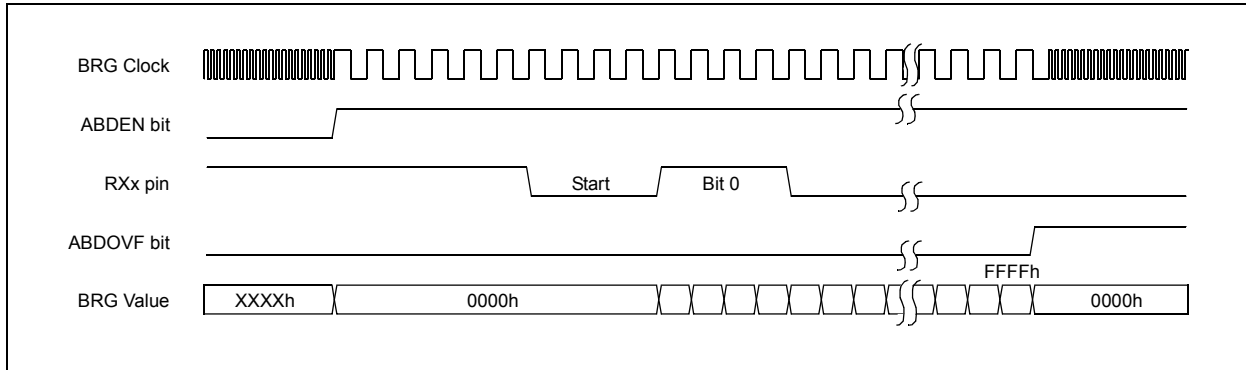
Since the BRG clock is reversed during ABD acquisition, the EUSART transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREGx cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSART operation.

# PIC18F46J11 FAMILY

**FIGURE 19-1: AUTOMATIC BAUD RATE CALCULATION**



**FIGURE 19-2: BRG OVERFLOW SEQUENCE**



## 19.2 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 BRG 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 BRG 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 ninth 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

### 19.2.1 EUSART ASYNCHRONOUS TRANSMITTER

Figure 19-3 displays the EUSART transmitter block diagram.

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, TXREGx. The TXREGx 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 TXREGx register (if available).

Once the TXREGx register transfers the data to the TSR register (occurs in one T<sub>cy</sub>), the TXREGx register is empty and the TXxIF flag bit is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF will be set regardless of the state of TXxIE; it cannot be cleared in software. TXxIF is also not cleared immediately upon loading TXREGx, but becomes valid in the second instruction cycle following the load instruction. Polling TXxIF immediately following a load of TXREGx will return invalid results.

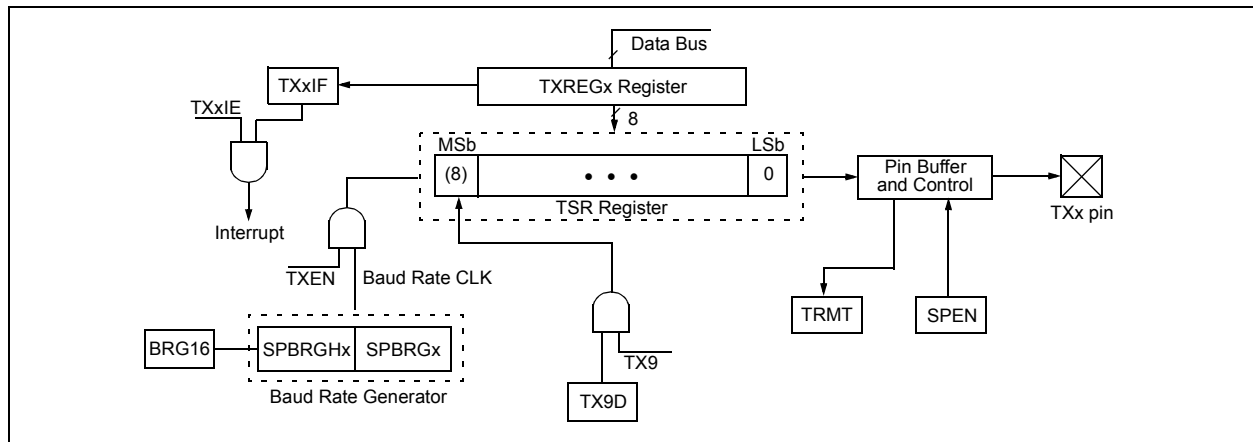
While TXxIF indicates the status of the TXREGx register; another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit, which is set when the TSR register is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register 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, TXxIF, is set when enable bit, TXEN, is set.

To set up an Asynchronous Transmission:

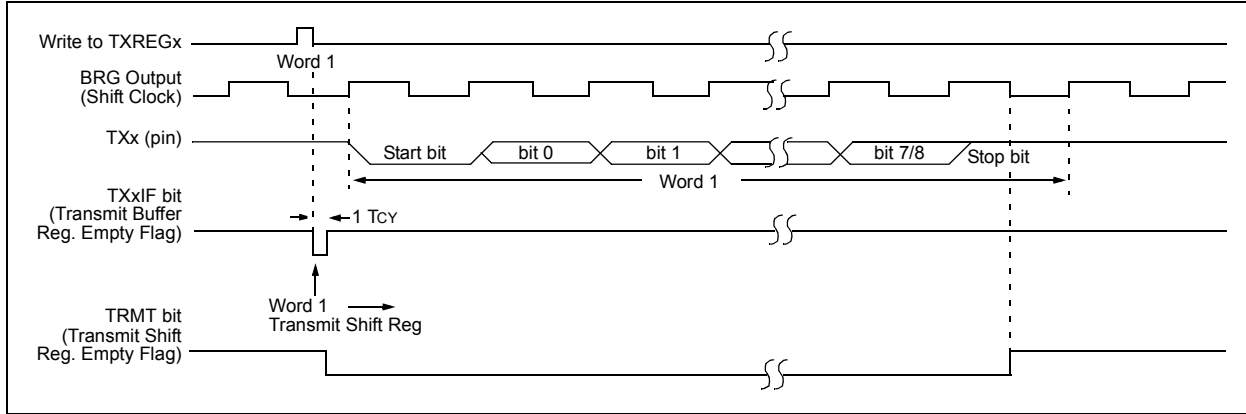
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 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, TXxIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREGx register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 19-3: EUSART TRANSMIT BLOCK DIAGRAM**

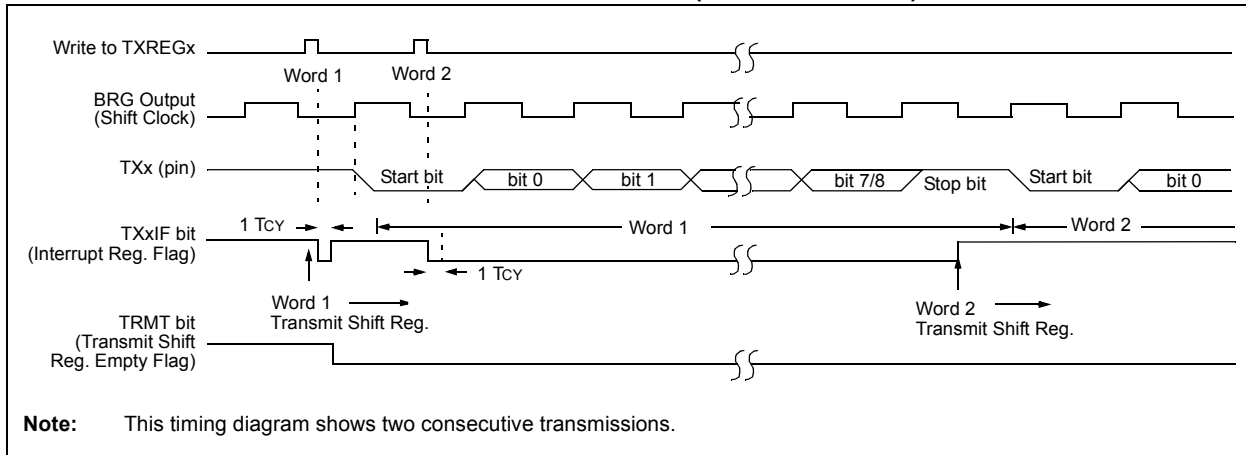


# PIC18F46J11 FAMILY

**FIGURE 19-4: ASYNCHRONOUS TRANSMISSION**



**FIGURE 19-5: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)**



**TABLE 19-5: 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	63
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPIP <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
TXREGx	EUSARTx Transmit Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXDTP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65
ODCON2	—	—	—	—	—	—	U2OD	U1OD	68

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

**Note 1:** These bits are only available on 44-pin devices.

## 19.2.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is displayed in Figure 19-6. The data is received on the RXx 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:

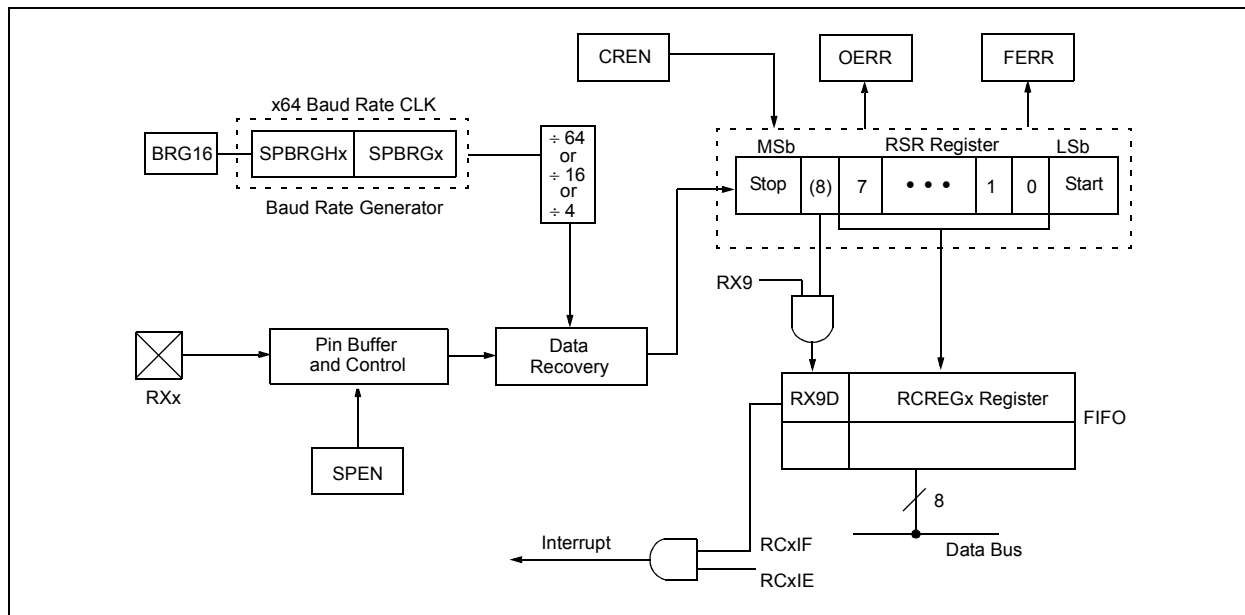
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, RCxIE.
4. If 9-bit reception is desired, set bit, RX9.
5. Enable the reception by setting bit, CREN.
6. Flag bit, RCxIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCxIE, was set.
7. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREGx 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.

## 19.2.3 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:

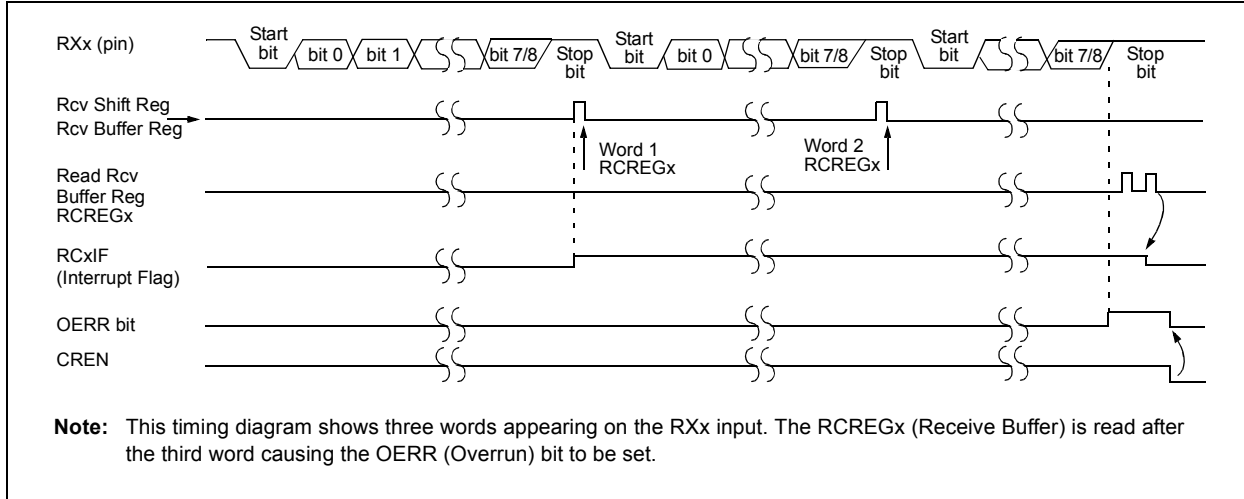
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCxIP 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 RCxIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCxIE and GIE bits are set.
8. Read the RCSTAx register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREGx 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 19-6: EUSARTx RECEIVE BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

**FIGURE 19-7: ASYNCHRONOUS RECEPTION**



**TABLE 19-6: 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	63
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMP1P <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
RCREGx	EUSARTx Receive Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65

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

**Note 1:** These bits are only available on 44-pin devices.

## 19.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the BRG 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 RXx/DTx line while the EUSART is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCONx<1>). Once set, the typical receive sequence on RXx/DTx 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 RXx/DTx 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 RCxIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 19-8) and asynchronously if the device is in Sleep mode (Figure 19-9). The interrupt condition is cleared by reading the RCREGx register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RXx 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.

### 19.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RXx/DTx, information with any state changes before the Stop bit may signal a false End-Of-Character (EOC) 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 or HSPLL 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.

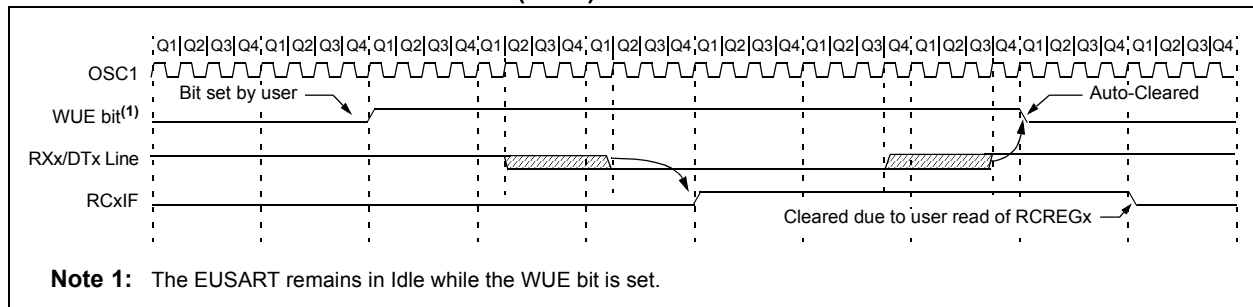
### 19.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCxIF 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 RCxIF bit. The WUE bit is cleared after this when a rising edge is seen on RXx/DTx. The interrupt condition is then cleared by reading the RCREGx register. Ordinarily, the data in RCREGx will be dummy data and should be discarded.

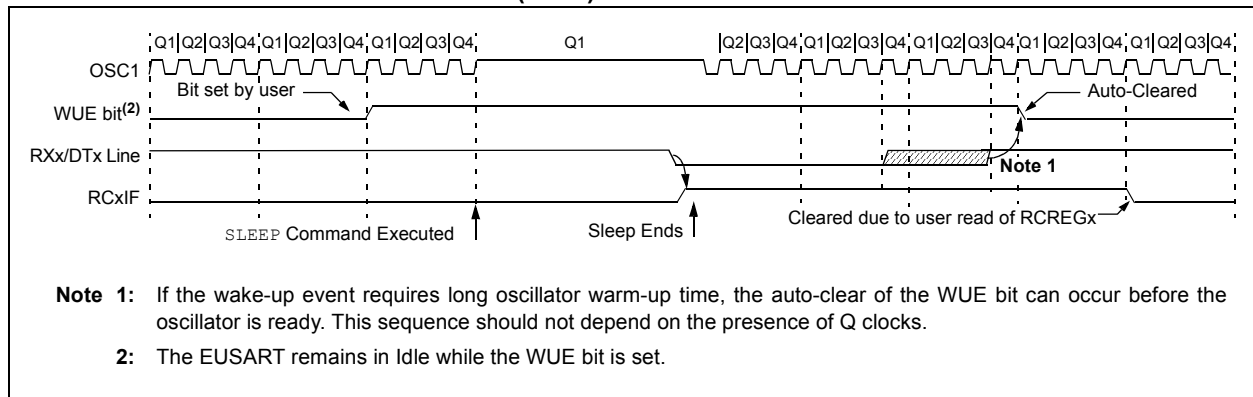
The fact that the WUE bit has been cleared (or is still set) and the RCxIF flag is set should not be used as an indicator of the integrity of the data in RCREGx. Users should consider implementing a parallel method in firmware to verify received data integrity.

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 19-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION**



**FIGURE 19-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP**



# PIC18F46J11 FAMILY

## 19.2.5 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 (TXSTAx<3> and TXSTAx<5>) are set while the Transmit Shift Register is loaded with data.

Note that the value of data written to TXREGx 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 TXREGx 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 19-10 for the timing of the Break character sequence.

### 19.2.5.1 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 TXREGx with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREGx 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 TXREGx becomes empty, as indicated by the TXxIF, the next data byte can be written to TXREGx.

## 19.2.6 RECEIVING A BREAK CHARACTER

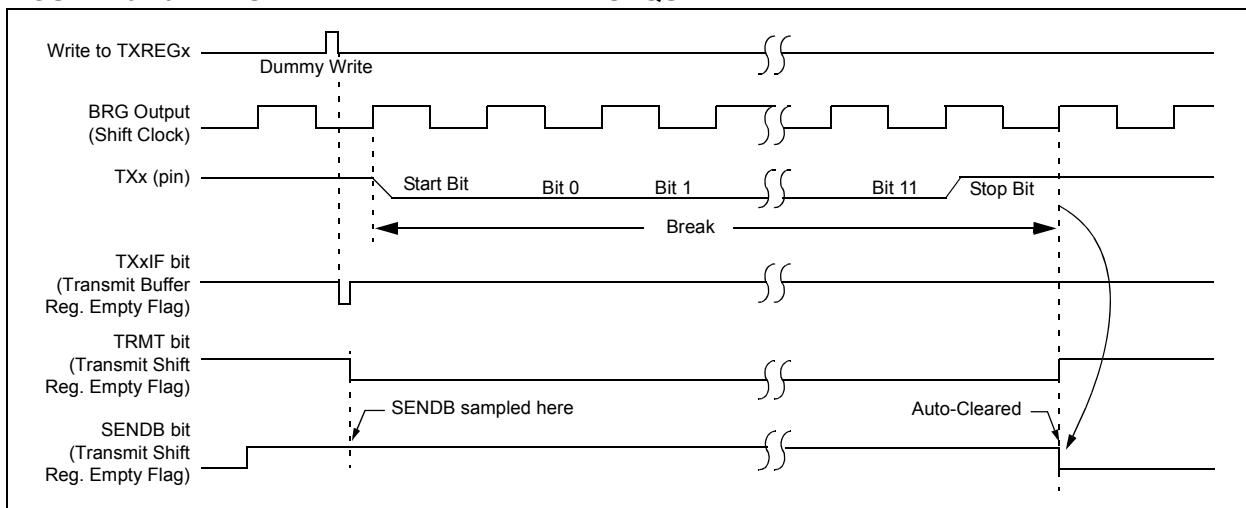
The Enhanced USART module can receive a Break character in two ways.

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 19.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RXx/DTx, cause an RCxIF 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 ABDEN bit once the TXxIF interrupt is observed.

**FIGURE 19-10: SEND BREAK CHARACTER SEQUENCE**





## 19.3 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTAx<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 (TXSTAx<4>). In addition, enable bit, SPEN (RCSTAx<7>), is set in order to configure the TXx and RXx pins to CKx (clock) and DTx (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CKx line. Clock polarity is selected with the TXCKP bit (BAUDCONx<4>). Setting TXCKP sets the Idle state on CKx as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

### 19.3.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 19-3. 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, TXREGx. The TXREGx 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 TXREGx (if available).

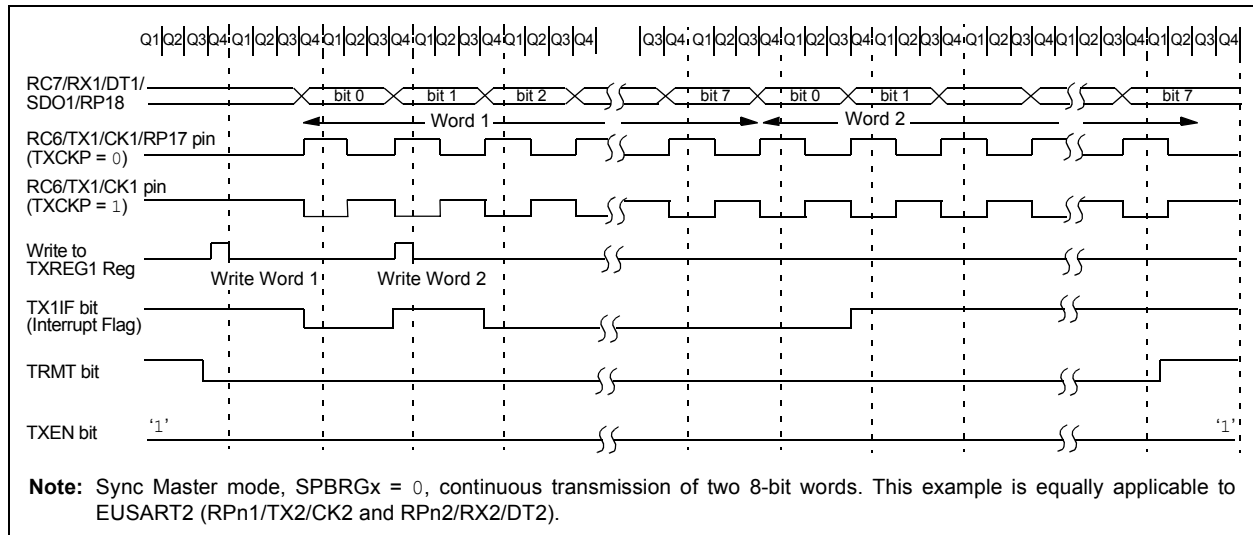
Once the TXREGx register transfers the data to the TSR register (occurs in one Tcy), the TXREGx is empty and the TXxIF flag bit is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF is set regardless of the state of enable bit, TXxIE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREGx register.

While flag bit, TXxIF, indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<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 must 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:

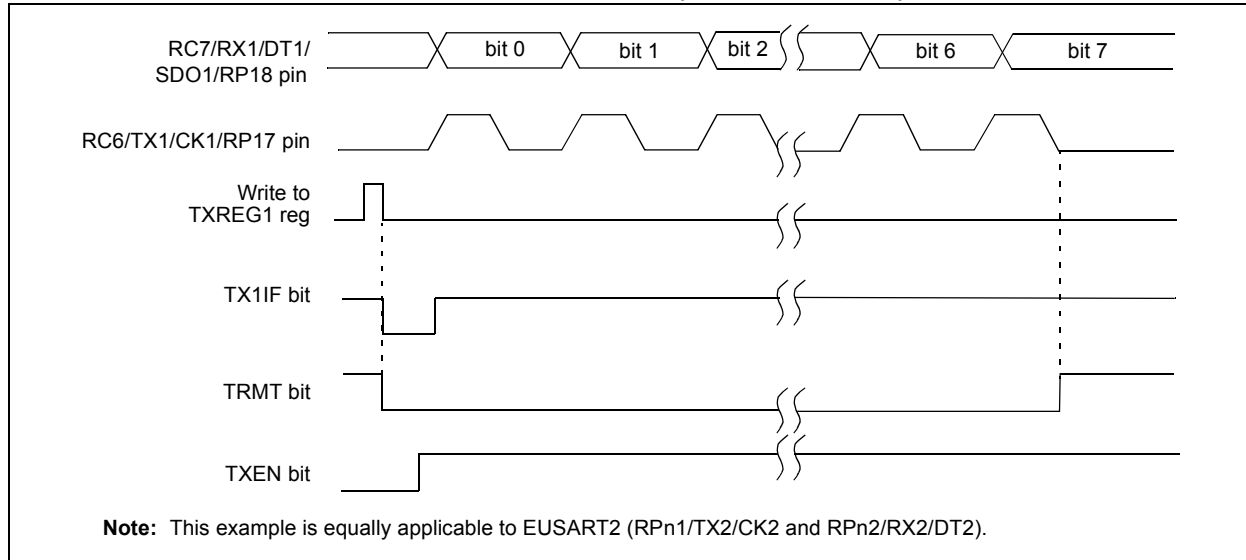
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the required baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission is required, 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 TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 19-11: SYNCHRONOUS TRANSMISSION**



# PIC18F46J11 FAMILY

**FIGURE 19-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



**TABLE 19-7: 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	63
PIR1	PMP1F <sup>(1)</sup>	AD1F	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMP1E <sup>(1)</sup>	AD1E	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMP1P <sup>(1)</sup>	AD1P	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
TXREGx	EUSARTx Transmit Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65
ODCON2	—	—	—	—	—	—	U2OD	U1OD	68

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

**Note 1:** These pins are only available on 44-pin devices.

## 19.3.2 EUSART SYNCHRONOUS MASTER RECEPTION

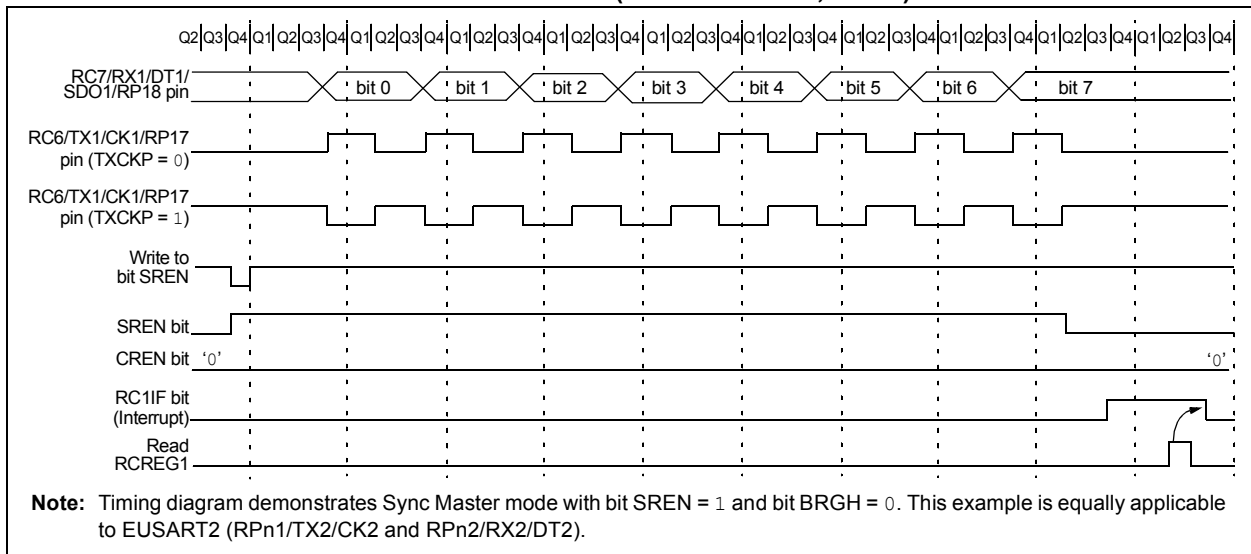
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTAx<5>) or the Continuous Receive Enable bit, CREN (RCSTAx<4>). Data is sampled on the RXx pin on the falling edge of the clock.

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 SPBRGHx:SPBRGx 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, RCxIE.
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, RCxIF, will be set when reception is complete and an interrupt will be generated if the enable bit, RCxIE, was set.
8. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREGx 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 19-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**



# PIC18F46J11 FAMILY

**TABLE 19-8: 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	RBIF	63
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPPIF <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
RCREGx	EUSARTx Receive Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65
ODCON2	—	—	—	—			U2OD	U1OD	68

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

**Note 1:** These pins are only available on 44-pin devices.

## 19.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

### 19.4.1 EUSART SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep mode.

If two words are written to the TXREGx 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 TXREGx register.
- c) Flag bit, TXxIF, will not be set.
- d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit, TXxIF, will now be set.
- e) If enable bit, TXxIE, 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, TXxIE.
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 TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

# PIC18F46J11 FAMILY

**TABLE 19-9: 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	63
PIR1	PMP1F <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMP1E <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMP1P <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
TXREGx	EUSARTx Transmit Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

**Note 1:** These pins are only available on 44-pin devices.

# PIC18F46J11 FAMILY

## 19.4.2 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 RCREGx register. If the RCxIE 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, RCxIE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RCxIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCxIE, was set.
6. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREGx 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 19-10: 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	63
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPPIF <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	65
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	65
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	65
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	65
RCREGx	EUSARTx Receive Register								65
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	65
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	66
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								66
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								65

**Legend:** — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

**Note 1:** These pins are only available on 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:



# PIC18F46J11 FAMILY

## 20.0 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 44-pin devices. Additionally, two internal channels are available for sampling the VDDCORE and VBG absolute reference voltage. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has six registers:

- A/D Control Register 0 (ADCON0)

- A/D Control Register 1 (ADCON1)
- A/D Port Configuration Register 2 (ANCON0)
- A/D Port Configuration Register 1 (ANCON1)
- A/D Result Registers (ADRESH and ADRESL)

The ADCON0 register, in Register 20-1, controls the operation of the A/D module. The ADCON1 register, in Register 20-2, configures the A/D clock source, programmed acquisition time and justification.

The ANCON0 and ANCON1 registers, in Register 20-3 and Register 20-4, configure the functions of the port pins.

### REGISTER 20-1: ADCON0: A/D CONTROL REGISTER 0 (ACCESS FC2h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
VCFG1	VCFG0	CHS3 <sup>(2)</sup>	CHS2 <sup>(2)</sup>	CHS1 <sup>(2)</sup>	CHS0 <sup>(2)</sup>	GO/DONE	ADON
bit 7							bit 0

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 7 **VCFG1:** Voltage Reference Configuration bit (VREF- source)

1 = VREF- (AN2)

0 = AVSS

bit 6 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)

1 = VREF+ (AN3)

0 = AVDD

bit 5-2 **CHS<3:0>:** Analog Channel Select bits<sup>(2)</sup>

0000 = Channel 00 (AN0)

0001 = Channel 01 (AN1)

0010 = Channel 02 (AN2)

0011 = Channel 03 (AN3)

0100 = Channel 04 (AN4)

0101 = Channel 05 (AN5)<sup>(1)</sup>

0110 = Channel 06 (AN6)<sup>(1)</sup>

0111 = Channel 07 (AN7)<sup>(1)</sup>

1000 = Channel 08 (AN8)

1001 = Channel 09 (AN9)

1010 = Channel 10 (AN10)

1011 = Channel 11 (AN11)

1100 = Channel 12 (AN12)

1101 = (Reserved)

1110 = VDDCORE

1111 = VBG Absolute Reference (~1.2V)<sup>(3)</sup>

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

**Note 1:** These channels are not implemented on 28-pin devices.

**2:** Performing a conversion on unimplemented channels will return random values.

**3:** For best accuracy, the band gap reference circuit should be enabled (ANCON1<7> = 1) at least 10 ms before performing a conversion on this channel.

# PIC18F46J11 FAMILY

## REGISTER 20-2: ADCON1: A/D CONTROL REGISTER 1 (ACCESS FC1h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCAL	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

### Legend:

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

- bit 7      **ADFM:** A/D Result Format Select bit  
           1 = Right justified  
           0 = Left justified
- bit 6      **ADCAL:** A/D Calibration bit  
           1 = Calibration is performed on next A/D conversion  
           0 = Normal A/D Converter operation
- bit 5-3    **ACQT<2:0>:** 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
- bit 2-0    **ADCS<2:0>:** A/D Conversion Clock Select bits  
           110 = FOSC/64  
           101 = FOSC/16  
           100 = FOSC/4  
           011 = FRC (clock derived from A/D RC oscillator)<sup>(1)</sup>  
           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<sub>cy</sub> (instruction cycle) is added before the A/D clock starts. This allows the *SLEEP* instruction to be executed before starting a conversion.

# PIC18F46J11 FAMILY

The ANCON0 and ANCON1 registers are used to configure the operation of the I/O pin associated with each analog channel. Setting any one of the PCFG bits configures the corresponding pin to operate as a digital only I/O. Clearing a bit configures the pin to operate as an analog input for either the A/D Converter or the comparator module; all digital peripherals are disabled and digital inputs read as '0'. As a rule, I/O pins that are multiplexed with analog inputs default to analog operation on device Resets.

In order to correctly perform A/D conversions on the V<sub>BG</sub> band gap reference (ADCON0<5:2> = 1111), the reference circuit must be powered on first. The VBGEN bit in the ANCON1 register allows the firmware to manually

request that the band gap reference circuit should be enabled. For best accuracy, firmware should allow a settling time of at least 10 ms prior to performing the first acquisition on this channel after enabling the band gap reference.

The reference circuit may already have been turned on if some other hardware module (such as the on-chip voltage regulator, comparators or HLVD) has already requested it. In this case, the initial turn-on settling time may have already elapsed and firmware does not need to wait as long before measuring V<sub>BG</sub>. Once the acquisition is complete, firmware may clear the VBGEN bit, which will save a small amount of power if no other modules are still requesting the V<sub>BG</sub> reference.

## REGISTER 20-3: ANCON0: A/D PORT CONFIGURATION REGISTER 2 (BANKED F48h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5 <sup>(1)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

### Legend:

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

bit 7-0            **PCFG<7:0>**: Analog Port Configuration bits (AN<7:0>)  
                     1 = Pin configured as a digital port  
                     0 = Pin configured as an analog channel – digital input disabled and reads '0'

**Note 1:** These bits are not implemented on 28-pin devices.

## REGISTER 20-4: ANCON1: A/D PORT CONFIGURATION REGISTER 1 (BANKED F49h)

R/W-0	r	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
VBGEN	—	—	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8
bit 7							bit 0

### Legend:

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

bit 7            **VBGEN**: 1.2V Band Gap Reference Enable bit  
                     1 = 1.2V band gap reference is powered on  
                     0 = 1.2V band gap reference is turned off to save power (if no other modules are requesting it)

bit 6            **Reserved**: Always maintain as '0' for lowest power consumption

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

bit 4-0        **PCFG<12:8>**: Analog Port Configuration bits (AN<12:8>)  
                     1 = Pin configured as a digital port  
                     0 = Pin configured as an analog channel – digital input disabled and reads '0'

# PIC18F46J11 FAMILY

The analog reference voltage is software selectable to either the device's positive and negative supply voltage (AVDD and AVSS), or the voltage level on the RA3/AN3/VREF+/C1INB and RA2/AN2/VREF-/CVREF/C2INB 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.

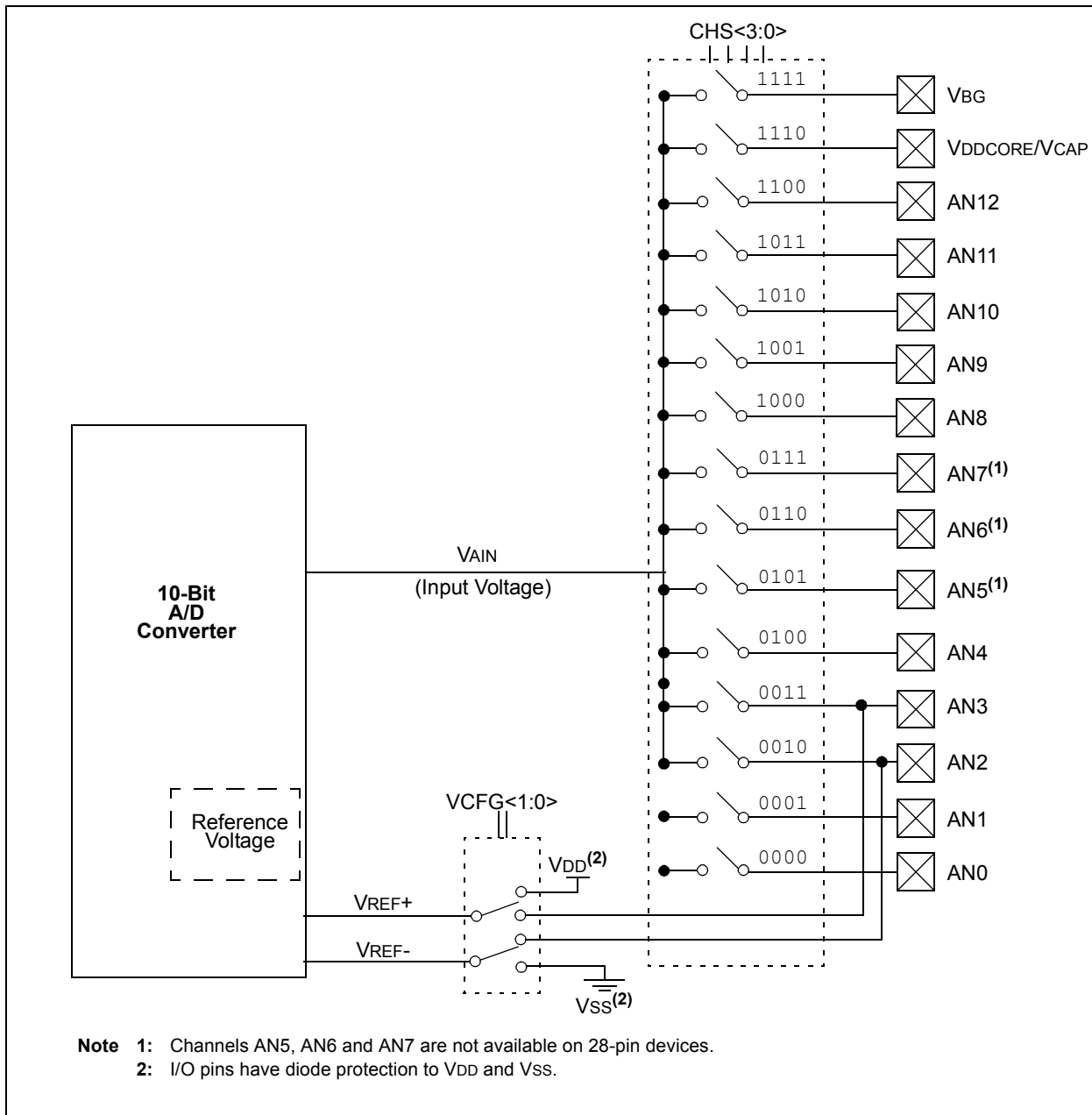
The output of the sample and hold is the input into the Converter, which generates the result via successive approximation.

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<1>) is cleared and the A/D Interrupt Flag bit, ADIF, is set.

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. The value in the ADRESH:ADRESL register pair is not modified for a Power-on Reset (POR). These registers will contain unknown data after a POR.

Figure 20-1 provides the block diagram of the A/D module.

**FIGURE 20-1: A/D BLOCK DIAGRAM**



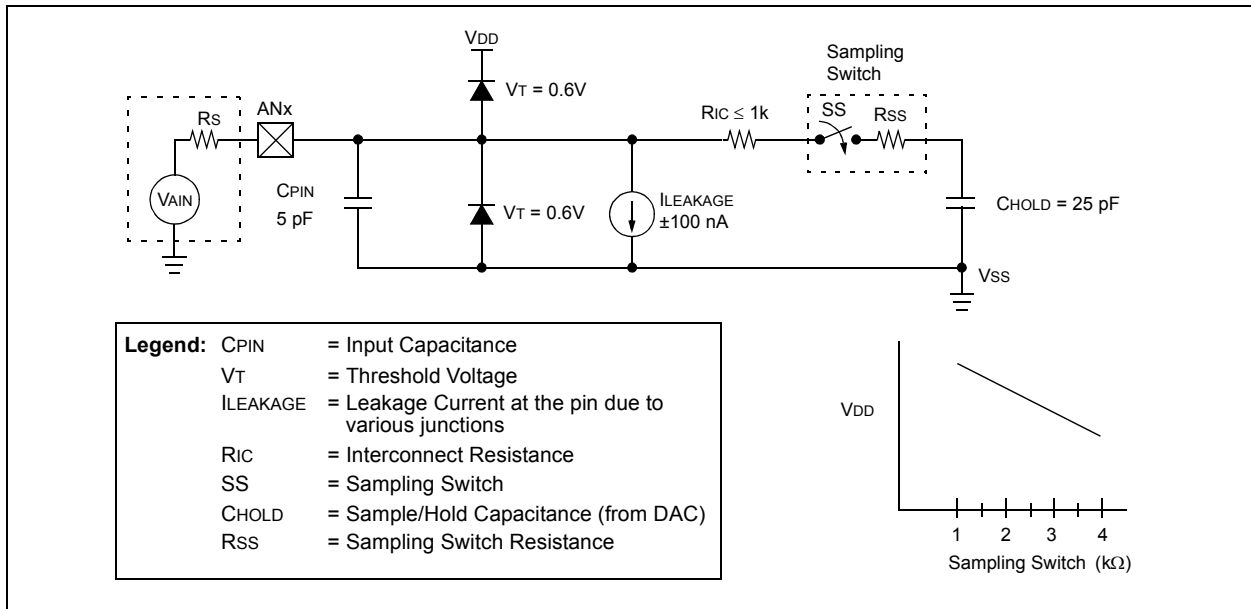
# PIC18F46J11 FAMILY

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 20.1 “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  $\overline{\text{GO/DONE}}$  bit and the actual start of the conversion.

The following steps should be followed to do an A/D conversion:

1. Configure the A/D module:
  - Configure the required ADC pins as analog pins using ANCON0, ANCON1
  - Set voltage reference using ADCON0
  - Select A/D input channel (ADCON0)
  - Select A/D acquisition time (ADCON1)
  - Select A/D conversion clock (ADCON1)
  - 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  $\overline{\text{GO/DONE}}$  bit (ADCON0<1>)
5. Wait for A/D conversion to complete, by either:
  - Polling for the  $\overline{\text{GO/DONE}}$  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 next acquisition starts.

**FIGURE 20-2: ANALOG INPUT MODEL**



# PIC18F46J11 FAMILY

## 20.1 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 illustrated in Figure 20-2. The source impedance (Rs) and the internal sampling switch (Rss) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (Rss) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 10 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 20-1 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.

Equation 20-3 provides 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	=	3V → Rss = 2 kΩ
Temperature	=	85°C (system max.)

### EQUATION 20-1: 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 20-2: 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 20-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$\begin{aligned} \text{TACQ} &= \text{TAMP} + \text{TC} + \text{TCOFF} \\ \text{TAMP} &= 0.2 \text{ ms} \\ \text{TCOFF} &= (\text{Temp} - 25^\circ\text{C})(0.02 \text{ } \mu\text{s}/^\circ\text{C}) \\ & \quad (85^\circ\text{C} - 25^\circ\text{C})(0.02 \text{ } \mu\text{s}/^\circ\text{C}) \\ & \quad 1.2 \text{ } \mu\text{s} \end{aligned}$$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, TCOFF = 0 μs.

$$\begin{aligned} \text{TC} &= -(\text{CHOLD})(\text{RIC} + \text{RSS} + \text{RS}) \ln(1/2048) \text{ } \mu\text{s} \\ & \quad -(25 \text{ pF})(1 \text{ k}\Omega + 2 \text{ k}\Omega + 2.5 \text{ k}\Omega) \ln(0.0004883) \text{ } \mu\text{s} \\ & \quad 1.05 \text{ } \mu\text{s} \\ \text{TACQ} &= 0.2 \text{ } \mu\text{s} + 1.05 \text{ } \mu\text{s} + 1.2 \text{ } \mu\text{s} \\ & \quad 2.45 \text{ } \mu\text{s} \end{aligned}$$

## 20.2 Selecting and Configuring Automatic Acquisition Time

The ADCON1 register allows the user to select an acquisition time that occurs each time the GO/DONE bit is set.

When the 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 GO/DONE bit. This occurs when the ACQT<2:0> bits (ADCON1<5:3>) remain in their Reset state ('000') and is compatible with devices that do not offer programmable acquisition times.

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When the GO/DONE bit is set, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and setting the GO/DONE bit.

In either case, when the conversion is completed, the 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.

## 20.3 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 in Table 28-28 for more information).

Table 20-1 provides the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 20-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency
Operation	ADCS<2:0>	
2 TOSC	000	2.86 MHz
4 TOSC	100	5.71 MHz
8 TOSC	001	11.43 MHz
16 TOSC	101	22.86 MHz
32 TOSC	010	45.71 MHz
64 TOSC	110	48.0 MHz
RC <sup>(2)</sup>	011	1.00 MHz <sup>(1)</sup>

- Note 1:** The RC source has a typical TAD time of 4  $\mu$ s.
- 2:** For device frequencies above 1 MHz, the device must be in Sleep mode for the entire conversion or the A/D accuracy may be out of specification.

## 20.4 Configuring Analog Port Pins

The ANCON0, ANCON1 and TRISA registers control the operation of 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 CHS<3:0> bits and the TRIS bits.

- Note 1:** When reading the PORT register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will 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.

# PIC18F46J11 FAMILY

## 20.5 A/D Conversions

Figure 20-3 displays the operation of the A/D Converter after the  $\overline{\text{GO/DONE}}$  bit has been set and the  $\text{ACQT}<2:0>$  bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins.

Figure 20-4 displays the operation of the A/D Converter after the  $\overline{\text{GO/DONE}}$  bit has been set, the  $\text{ACQT}<2:0>$  bits are set to '010' and a 4  $\text{TAD}$  acquisition time has been selected 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  $\text{ADRESH:ADRESL}$  registers will continue to contain the value of the last completed conversion (or the last value written to the  $\text{ADRESH:ADRESL}$  registers).

After the A/D conversion is completed or aborted, a 2  $\text{TAD}$  wait is required before the next acquisition can be started. After this wait, acquisition on the selected channel is automatically started.

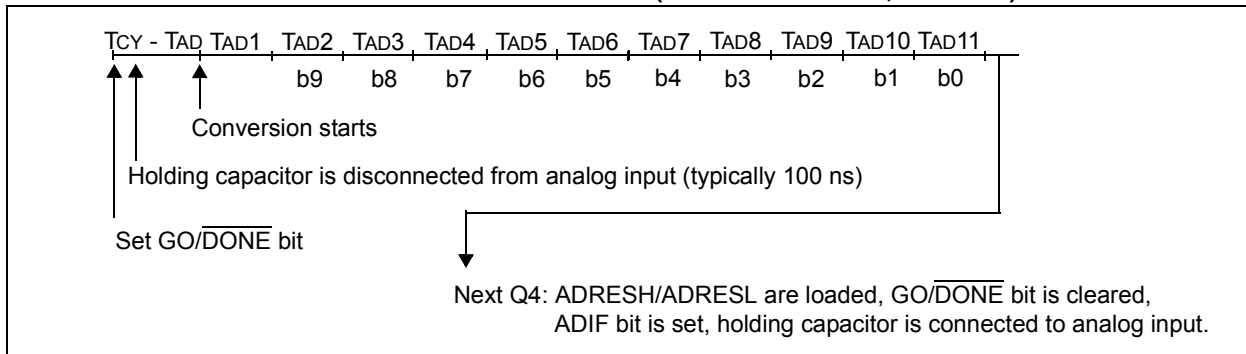
**Note:** The  $\overline{\text{GO/DONE}}$  bit should **NOT** be set in the same instruction that turns on the A/D.

## 20.6 Use of the ECCP2 Trigger

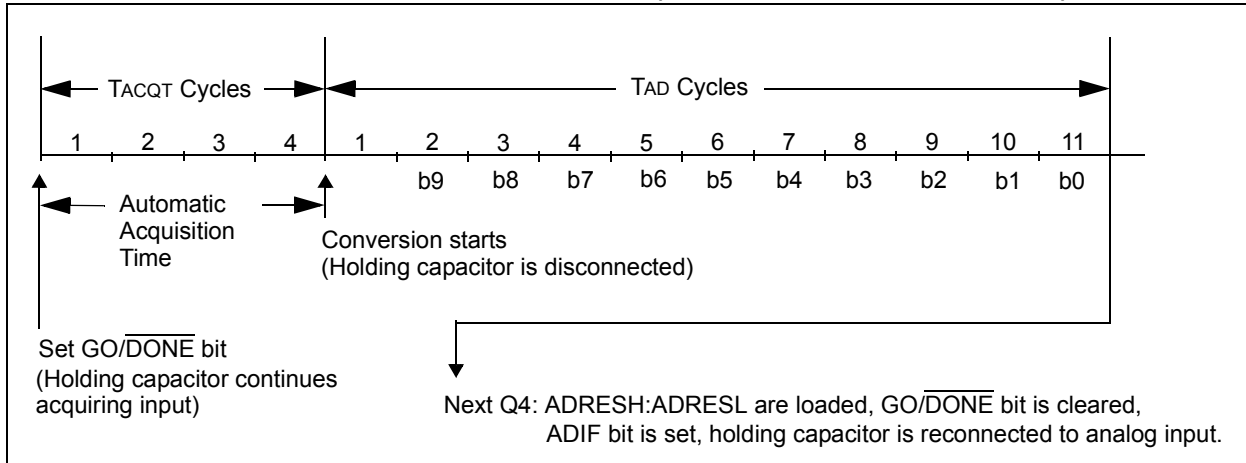
An A/D conversion can be started by the Special Event Trigger of the ECCP2 module. This requires that the  $\text{CCP2M}<3:0>$  bits ( $\text{CCP2CON}<3:0>$ ) be programmed as '1011' and that the A/D module is enabled ( $\text{ADON}$  bit is set). When the trigger occurs, the  $\overline{\text{GO/DONE}}$  bit will be set, starting the A/D acquisition and conversion, and the Timer1 (or Timer3) counter will be reset to zero. Timer1 (or Timer3) is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving  $\text{ADRESH/ADRESL}$  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  $\text{TACQ}$  time is selected before the Special Event Trigger sets the  $\overline{\text{GO/DONE}}$  bit (starts a conversion).

If the A/D module is not enabled ( $\text{ADON}$  is cleared), the Special Event Trigger will be ignored by the A/D module but will still reset the Timer1 (or Timer3) counter.

**FIGURE 20-3: A/D CONVERSION  $\text{TAD}$  CYCLES ( $\text{ACQT}<2:0> = 000, \text{TACQ} = 0$ )**



**FIGURE 20-4: A/D CONVERSION  $\text{TAD}$  CYCLES ( $\text{ACQT}<2:0> = 010, \text{TACQ} = 4 \text{TAD}$ )**





## 20.7 A/D Converter Calibration

The A/D Converter in the PIC18F46J11 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 (ADCON1<6>). 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 the offset. Thus, subsequent offsets will be compensated.

Example 20-1 provides an example of a calibration routine.

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.

## 20.8 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 ACQT<2:0> and ADCS<2:0> bits in ADCON1 should be updated in accordance with the power-managed mode clock that will be used. After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power-managed Idle mode during the conversion.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in the Sleep mode requires the A/D RC clock to be selected. If bits, ACQT<2:0>, are set to ‘000’ and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS bits in the OSCCON register must have already been cleared prior to starting the conversion.

### EXAMPLE 20-1: SAMPLE A/D CALIBRATION ROUTINE

```
BCF    ANCON0,PCFG0    ;Make Channel 0 analog
BSF    ADCON0,ADON     ;Enable A/D module
BSF    ADCON1,ADCAL    ;Enable Calibration
BSF    ADCON0,GO       ;Start a dummy A/D conversion
CALIBRATION
BTFSC  ADCON0,GO       ;Wait for the dummy conversion to finish
BRA    CALIBRATION     ;
BCF    ADCON1,ADCAL    ;Calibration done, turn off calibration enable
                          ;Proceed with the actual A/D conversion
```

# PIC18F46J11 FAMILY

**TABLE 20-2: SUMMARY OF A/D REGISTERS**

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	63
PIR1	PMPIF <sup>(1)</sup>	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	65
PIE1	PMPIE <sup>(1)</sup>	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	65
IPR1	PMPIP <sup>(1)</sup>	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	65
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	65
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	65
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	65
ADRESH	A/D Result Register High Byte								64
ADRESL	A/D Result Register Low Byte								64
ADCON0	VCFG1	VCFG0	CHS3	CHS3	CHS1	CHS0	GO/DONE	ADON	64
ANCON0	PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5 <sup>(1)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	68
ADCON1	ADFM	ADCAL	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	64
ANCON1	VBGEN	r <sup>(2)</sup>	—	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	68
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	65
PORTA	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0	66
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	66

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

**Note 1:** These bits are only available on 44-pin devices.

**2:** Reserved. Always maintain as '0' for minimum power consumption.

## 21.0 COMPARATOR MODULE

The analog comparator module contains two comparators that can be independently configured in a variety of ways. The inputs can be selected from the analog inputs and two internal voltage references. The digital outputs are available at the pin level and can also be read through the control register. Multiple output and interrupt event generation is also available. Figure 21-1 provides a generic single comparator from the module.

Key features of the module are:

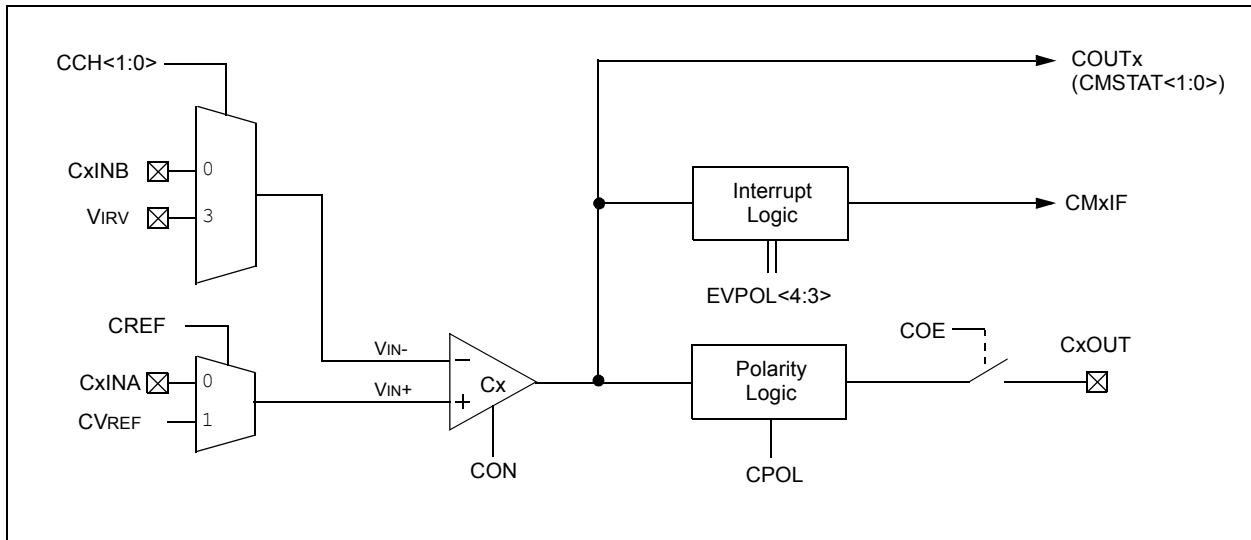
- Independent comparator control
- Programmable input configuration
- Output to both pin and register levels
- Programmable output polarity
- Independent interrupt generation for each comparator with configurable interrupt-on-change

## 21.1 Registers

The CMxCON registers (Register 21-1) select the input and output configuration for each comparator, as well as the settings for interrupt generation.

The CMSTAT register (Register 21-2) provides the output results of the comparators. The bits in this register are read-only.

**FIGURE 21-1: COMPARATOR SIMPLIFIED BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

## REGISTER 21-1: CMxCON: COMPARATOR CONTROL x REGISTER (ACCESS FD2h/FD1h)

R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
bit 7							bit 0

### Legend:

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

- bit 7      **CON:** Comparator Enable bit  
 1 = Comparator is enabled  
 0 = Comparator is disabled
- bit 6      **COE:** Comparator Output Enable bit  
 1 = Comparator output is present on the CxOUT pin (assigned in PPS module)  
 0 = Comparator output is internal only
- bit 5      **CPOL:** Comparator Output Polarity Select bit  
 1 = Comparator output is inverted  
 0 = Comparator output is not inverted
- bit 4-3    **EVPOL<1:0>:** Interrupt Polarity Select bits  
 11 = Interrupt generation on any change of the output<sup>(1)</sup>  
 10 = Interrupt generation only on high-to-low transition of the output  
 01 = Interrupt generation only on low-to-high transition of the output  
 00 = Interrupt generation is disabled
- bit 2      **CREF:** Comparator Reference Select bit (non-inverting input)  
 1 = Non-inverting input connects to internal CVREF voltage  
 0 = Non-inverting input connects to CxINA pin
- bit 1-0    **CCH<1:0>:** Comparator Channel Select bits  
 11 = Inverting input of comparator connects to VIRV  
 10 = For CM1CON, inverting input of comparator connects to C2INB pin; for CM2CON, reserved  
 01 = Reserved  
 00 = Inverting input of comparator connects to CxINB pin

**Note 1:** The CMxIF is automatically set any time this mode is selected and must be cleared by the application after the initial configuration.

# PIC18F46J11 FAMILY

## REGISTER 21-2: CMSTAT: COMPARATOR STATUS REGISTER (ACCESS F70h)

U-0	U-0	U-0	U-0	U-0	U-0	R-1	R-1
—	—	—	—	—	—	COUT2	COUT1
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 1-0      **COUT<2:1>:** Comparator x Status bits

If CPOL = 0 (non-inverted polarity):

1 = Comparator VIN+ > VIN-

0 = Comparator VIN+ < VIN-

If CPOL = 1 (inverted polarity):

1 = Comparator VIN+ < VIN-

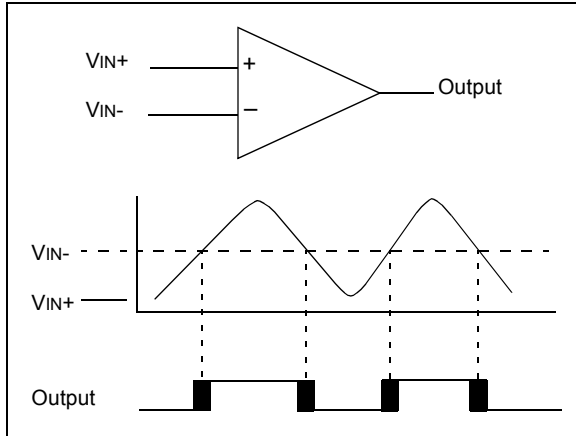
0 = Comparator VIN+ > VIN-

# PIC18F46J11 FAMILY

## 21.2 Comparator Operation

A single comparator is shown in Figure 21-2, 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 21-2 represent the uncertainty due to input offsets and response time.

**FIGURE 21-2: SINGLE COMPARATOR**



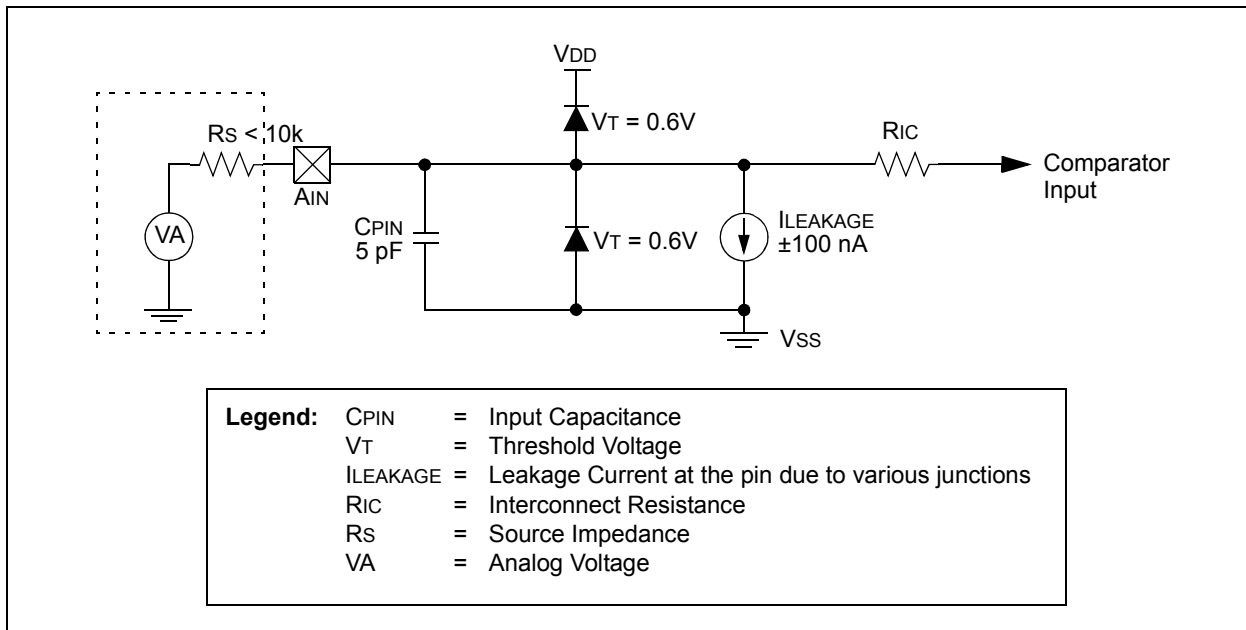
## 21.3 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response to a comparator input change. Otherwise, the maximum delay of the comparators should be used (see **Section 28.0 “Electrical Characteristics”**).

## 21.4 Analog Input Connection Considerations

Figure 21-3 provides a simplified circuit for an analog input. Since the analog pins are connected to a digital output, they have reverse biased diodes to  $V_{DD}$  and  $V_{SS}$ . The analog input, therefore, must be between  $V_{SS}$  and  $V_{DD}$ . 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 k $\Omega$  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 21-3: COMPARATOR ANALOG INPUT MODEL**



## 21.5 Comparator Control and Configuration

Each comparator has up to eight possible combinations of inputs: up to four external analog inputs, and one of two internal voltage references.

Both comparators allow a selection of the signal from pin, CxINA, or the voltage from the comparator reference (CVREF) on the non-inverting channel. This is compared to either CxINB, CTMU or the microcontroller's fixed internal reference voltage (VIRV, 0.6V nominal) on the inverting channel.

Table 21-1 provides the comparator inputs and outputs tied to fixed I/O pins.

Figure 21-4 illustrates the available comparator configurations and their corresponding bit settings.

**TABLE 21-1: COMPARATOR INPUTS AND OUTPUTS**

Comparator	Input or Output	I/O Pin
1	C1INA (VIN+)	RA0
	C1INB (VIN-)	RA3
	C1OUT	Remapped RPn
2	C2INA(VIN+)	RA1
	C2INB(VIN-)	RA2
	C2OUT	Remapped RPn

### 21.5.1 COMPARATOR ENABLE AND INPUT SELECTION

Setting the CON bit of the CMxCON register (CMxCON<7>) enables the comparator for operation. Clearing the CON bit disables the comparator, resulting in minimum current consumption.

The CCH<1:0> bits in the CMxCON register (CMxCON<1:0>) direct either one of three analog input pins, or the Internal Reference Voltage (VIRV), to the comparator VIN-. Depending on the comparator operating mode, either an external or internal voltage reference may be used. The analog signal present at VIN- is compared to the signal at VIN+ and the digital output of the comparator is adjusted accordingly.

The external reference is used when CREF = 0 (CMxCON<2>) and VIN+ is connected to the CxINA pin. When external voltage references are used, the comparator module can be configured to have the reference sources externally. The reference signal must be between VSS and VDD, and can be applied to either pin of the comparator.

The comparator module also allows the selection of an internally generated voltage reference (CVREF) from the comparator voltage reference module. This module is described in more detail in **Section 21.0 "Comparator Module"**. The reference from the comparator voltage reference module is only available when CREF = 1. In this mode, the internal voltage reference is applied to the comparator's VIN+ pin.

**Note:** The comparator input pin selected by CCH<1:0> must be configured as an input by setting both the corresponding TRIS and PCFG bits in the ANCON1 register.

### 21.5.2 COMPARATOR ENABLE AND OUTPUT SELECTION

The comparator outputs are read through the CMSTAT register. The CMSTAT<0> reads the Comparator 1 output and CMSTAT<1> reads the Comparator 2 output. These bits are read-only.

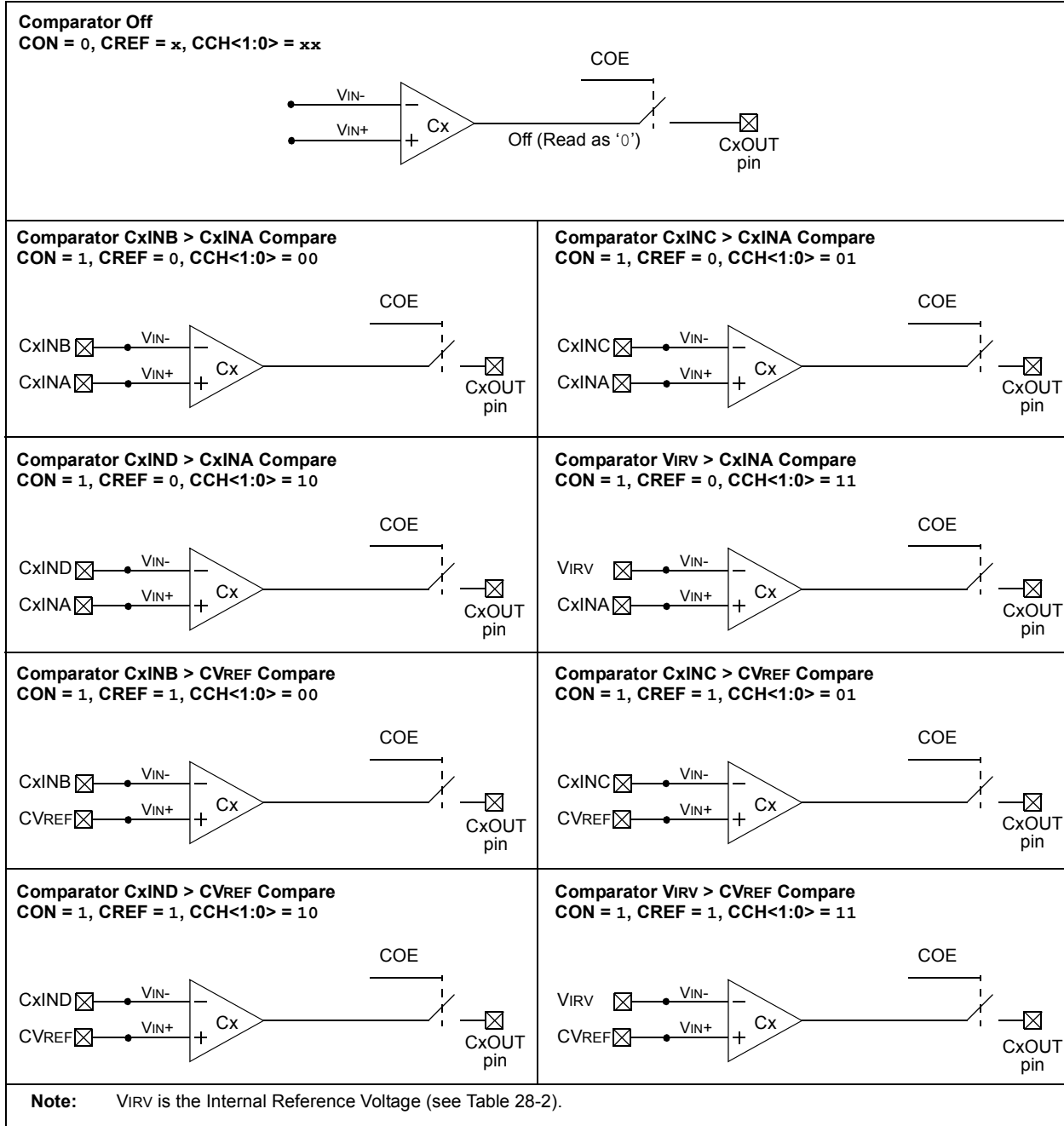
The comparator outputs may also be directly output to the RPn I/O pins by setting the COE bit (CMxCON<6>). When enabled, multiplexers in the output path of the pins switch to the output of the comparator.

By default, the comparator's output is at logic high whenever the voltage on VIN+ is greater than on VIN-. The polarity of the comparator outputs can be inverted using the CPOL bit (CMxCON<5>).

The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications, as discussed in **Section 21.2 "Comparator Operation"**.

# PIC18F46J11 FAMILY

**FIGURE 21-4: COMPARATOR CONFIGURATIONS**





## 21.6 Comparator Interrupts

The comparator interrupt flag is set whenever any of the following occurs:

- Low-to-high transition of the comparator output
- High-to-low transition of the comparator output
- Any change in the comparator output

The comparator interrupt selection is done by the  $EVPOL<1:0>$  bits in the  $CMxCON$  register ( $CMxCON<4:3>$ ).

In order to provide maximum flexibility, the output of the comparator may be inverted using the  $CPOL$  bit in the  $CMxCON$  register ( $CMxCON<5>$ ). This is functionally identical to reversing the inverting and non-inverting inputs of the comparator for a particular mode.

An interrupt is generated on the low-to-high or high-to-low transition of the comparator output. This mode of interrupt generation is dependent on  $EVPOL<1:0>$  in the  $CMxCON$  register. When  $EVPOL<1:0> = 01$  or  $10$ , the interrupt is generated on a low-to-high or high-to-low transition of the comparator output. Once the interrupt is generated, it is required to clear the interrupt flag by software.

When  $EVPOL<1:0> = 11$ , 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  $CMSTAT<1:0>$ , to determine the actual change that occurred. The  $CMxIF$  bits ( $PIR2<6:5>$ ) are the Comparator Interrupt Flags. The  $CMxIF$  bits must be reset by clearing them. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

Table 21-2 provides the interrupt generation corresponding to comparator input voltages and  $EVPOL$  bit settings.

Both the  $CMxIE$  bits ( $PIE2<6:5>$ ) 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  $CMxIF$  bits will still be set if an interrupt condition occurs.

Figure 21-1 provides a simplified diagram of the interrupt section.

**TABLE 21-2: COMPARATOR INTERRUPT GENERATION**

CPOL	EVPOL<1:0>	Comparator Input Change	COUTx Transition	Interrupt Generated
0	00	$V_{IN+} > V_{IN-}$	Low-to-High	No
		$V_{IN+} < V_{IN-}$	High-to-Low	No
	01	$V_{IN+} > V_{IN-}$	Low-to-High	Yes
		$V_{IN+} < V_{IN-}$	High-to-Low	No
	10	$V_{IN+} > V_{IN-}$	Low-to-High	No
		$V_{IN+} < V_{IN-}$	High-to-Low	Yes
	11	$V_{IN+} > V_{IN-}$	Low-to-High	Yes
		$V_{IN+} < V_{IN-}$	High-to-Low	Yes
1	00	$V_{IN+} > V_{IN-}$	High-to-Low	No
		$V_{IN+} < V_{IN-}$	Low-to-High	No
	01	$V_{IN+} > V_{IN-}$	High-to-Low	No
		$V_{IN+} < V_{IN-}$	Low-to-High	Yes
	10	$V_{IN+} > V_{IN-}$	High-to-Low	Yes
		$V_{IN+} < V_{IN-}$	Low-to-High	No
	11	$V_{IN+} > V_{IN-}$	High-to-Low	Yes
		$V_{IN+} < V_{IN-}$	Low-to-High	Yes

# PIC18F46J11 FAMILY

## 21.7 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. To minimize power consumption while in Sleep mode, turn off the comparators (CON = 0) before entering Sleep. If the device wakes up from Sleep, the contents of the CMxCON register are not affected.

## 21.8 Effects of a Reset

A device Reset forces the CMxCON registers to their Reset state. This forces both comparators and the voltage reference to the OFF state.

**TABLE 21-3: 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:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	63
PIR2	OSCFIF	CM2IF	CM1IF	—	BCL1IF	LVDIF	TMR3IF	CCP2IF	65
PIE2	OSCFIE	CM2IE	CM1IE	—	BCL1IE	LVDIE	TMR3IE	CCP2IE	65
IPR2	OSCFIP	CM2IP	CM1IP	—	BCL1IP	LVDIP	TMR3IP	CCP2IP	65
CMxCON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	64
CVRCON <sup>(1)</sup>	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	68
CMSTAT	—	—	—	—	—	—	COUT2	COUT1	67
ANCON0	PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5 <sup>(1)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	68
PORTA	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0	66
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	66

**Legend:** — = unimplemented, read as '0', r = reserved. Shaded cells are not related to comparator operation.

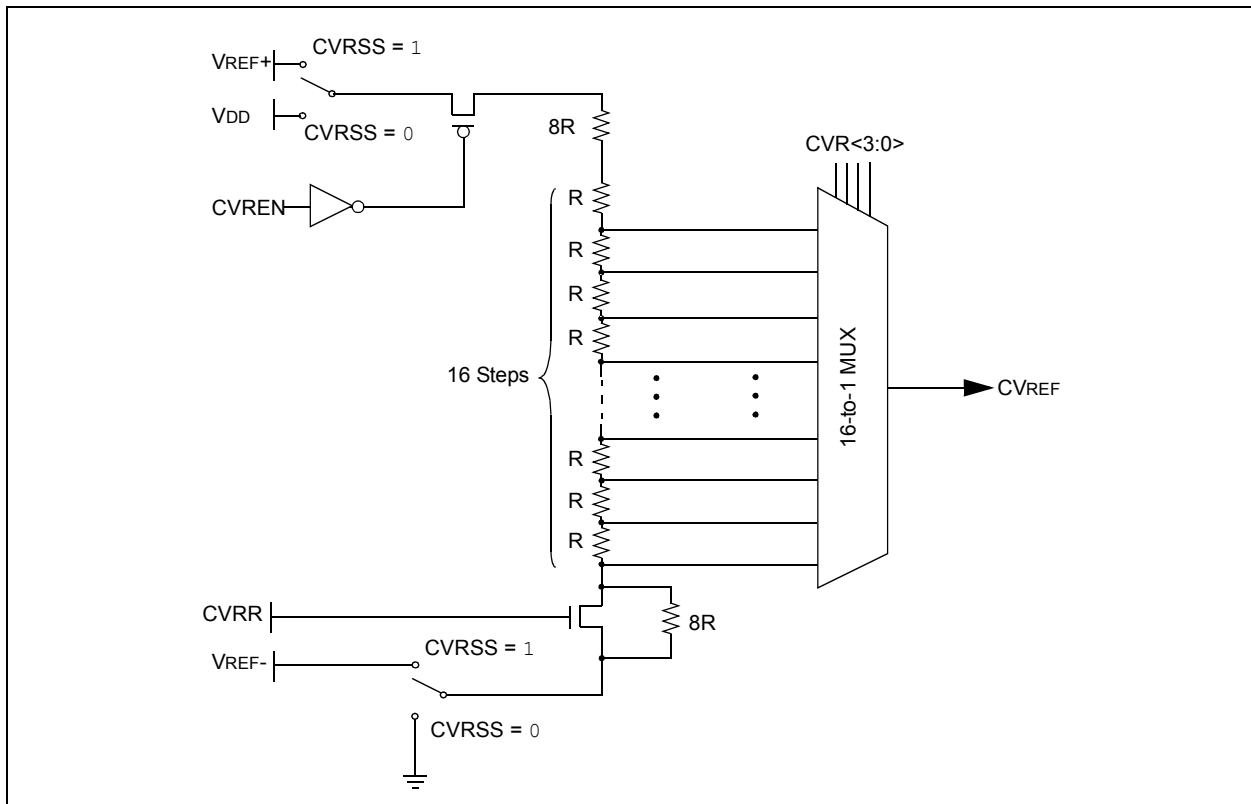
**Note 1:** These bits and/or registers are not implemented on 28-pin devices.

## 22.0 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.

Figure 22-1 provides a block diagram of the module. 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.

**FIGURE 22-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

## 22.1 Configuring the Comparator Voltage Reference

The comparator voltage reference module is controlled through the CVRCON register (Register 22-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR<3:0>), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

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

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 28-4 in **Section 28.0 “Electrical Characteristics”**).

### EQUATION 22-1: CALCULATING OUTPUT OF THE COMPARATOR VOLTAGE REFERENCE

<p>When CVRR = 1 and CVRSS = 0:  <math>CVREF = ((CVR&lt;3:0&gt;)/24) \times (CVRSRC)</math></p> <p>When CVRR = 0 and CVRSS = 0:  <math>CVREF = (CVRSRC/4) + ((CVR&lt;3:0&gt;)/32) \times (CVRSRC)</math></p> <p>When CVRR = 1 and CVRSS = 1:  <math>CVREF = ((CVR&lt;3:0&gt;)/24) \times (CVRSRC) + VREF-</math></p> <p>When CVRR = 0 and CVRSS = 1:  <math>CVREF = (CVRSRC/4) + ((CVR&lt;3:0&gt;)/32) \times (CVRSRC) + VREF-</math></p>
---

### REGISTER 22-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER (BANKED F53h)

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 <sup>(1)</sup>	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

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

- bit 7      **CVREN:** Comparator Voltage Reference Enable bit  
 1 = CVREF circuit powered on  
 0 = CVREF circuit powered down
- bit 6      **CVROE:** Comparator VREF Output Enable bit<sup>(1)</sup>  
 1 = CVREF voltage level is also output on the RA2/AN2/VREF-/CVREF/C2INB pin  
 0 = CVREF voltage is disconnected from the RA2/AN2/VREF-/CVREF/C2INB pin
- bit 5      **CVRR:** Comparator VREF Range Selection bit  
 1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)  
 0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)
- bit 4      **CVRSS:** Comparator VREF Source Selection bit  
 1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)  
 0 = Comparator reference source, CVRSRC = AVDD – AVSS
- bit 3-0    **CVR<3:0>:** Comparator VREF Value Selection bits ( $0 \leq (CVR<3:0>) \leq 15$ )  
 When CVRR = 1:  
 $CVREF = ((CVR<3:0>)/24) \bullet (CVRSRC)$   
 When CVRR = 0:  
 $CVREF = (CVRSRC/4) + ((CVR<3:0>)/32) \bullet (CVRSRC)$

**Note 1:** CVROE overrides the TRIS bit setting.

## 22.2 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 (see Figure 22-1) 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 28.0 “Electrical Characteristics”**.

## 22.3 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. See Figure 22-2 for an example buffering technique.

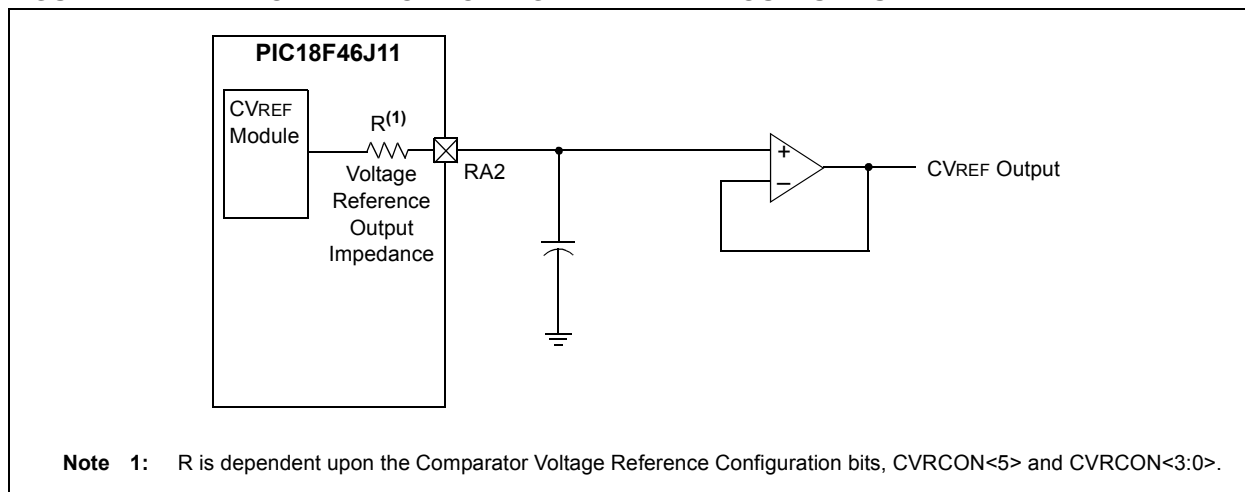
## 22.4 Operation During Sleep

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.

## 22.5 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.

**FIGURE 22-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



**TABLE 22-1: 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	68
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	64
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	64
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	66
ANCON0	PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5 <sup>(1)</sup>	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	68
ANCON1	VBGEN	r	—	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	68

**Legend:** — = unimplemented, read as '0', r = reserved. Shaded cells are not used with the comparator voltage reference.

**Note 1:** These bits are only available on 44-pin devices.

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## 23.0 HIGH/LOW VOLTAGE DETECT (HLVD)

PIC18F46J11 family devices (including PIC18LF46J11 family devices) have a High/Low Voltage Detect (HLVD) module for monitoring the absolute voltage on VDD or the HLVDIN pin. This is a programmable circuit that allows the user to specify both a device voltage trip point and the direction of change from that point.

If the module detects an excursion past the trip point in that direction, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to the interrupt.

The High/Low-Voltage Detect Control register (Register 23-1) completely controls the operation of the HLVD module. This allows the circuitry to be “turned off” by the user under software control, which minimizes the current consumption for the device.

Figure 23-1 provides a block diagram for the HLVD module.

**REGISTER 23-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER (ACCESS F85h)**

R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3 <sup>(1)</sup>	HLVDL2 <sup>(1)</sup>	HLVDL1 <sup>(1)</sup>	HLVDL0 <sup>(1)</sup>
bit 7							bit 0

**Legend:**

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

- bit 7      **VDIRMAG:** Voltage Direction Magnitude Select bit  
           1 = Event occurs when voltage equals or exceeds trip point (HLVDL<3:0>)  
           0 = Event occurs when voltage equals or falls below trip point (HLVDL<3:0>)
- bit 6      **BGVST:** Band Gap Reference Voltages Stable Status Flag bit  
           1 = Indicates internal band gap voltage references is stable  
           0 = Indicates internal band gap voltage reference is not stable
- bit 5      **IRVST:** Internal Reference Voltage Stable Flag bit  
           1 = Indicates that the voltage detect logic will generate the interrupt flag at the specified voltage range  
           0 = Indicates that the voltage detect logic will not generate the interrupt flag at the specified voltage range and the HLVD interrupt should not be enabled
- bit 4      **HLVDEN:** High/Low-Voltage Detect Power Enable bit  
           1 = HLVD enabled  
           0 = HLVD disabled
- bit 3-0    **HLVDL<3:0>:** Voltage Detection Limit bits<sup>(1)</sup>  
           1111 = External analog input is used (input comes from the HLVDIN pin)  
           1110 = Maximum setting  
           .  
           .  
           .  
           0000 = Minimum setting

**Note 1:** See Table 28-7 in Section 28.0 “Electrical Characteristics” for specifications.

The module is enabled by setting the HLVDEN bit. Each time the module is enabled, the circuitry requires some time to stabilize. The IRVST bit is a read-only bit that indicates when the circuit is stable. The module can generate an interrupt only after the circuit is stable and IRVST is set.

The VDIRMAG bit determines the overall operation of the module. When VDIRMAG is cleared, the module monitors for drops in VDD below a predetermined set point. When the bit is set, the module monitors for rises in VDD above the set point.

# PIC18F46J11 FAMILY

## 23.1 Operation

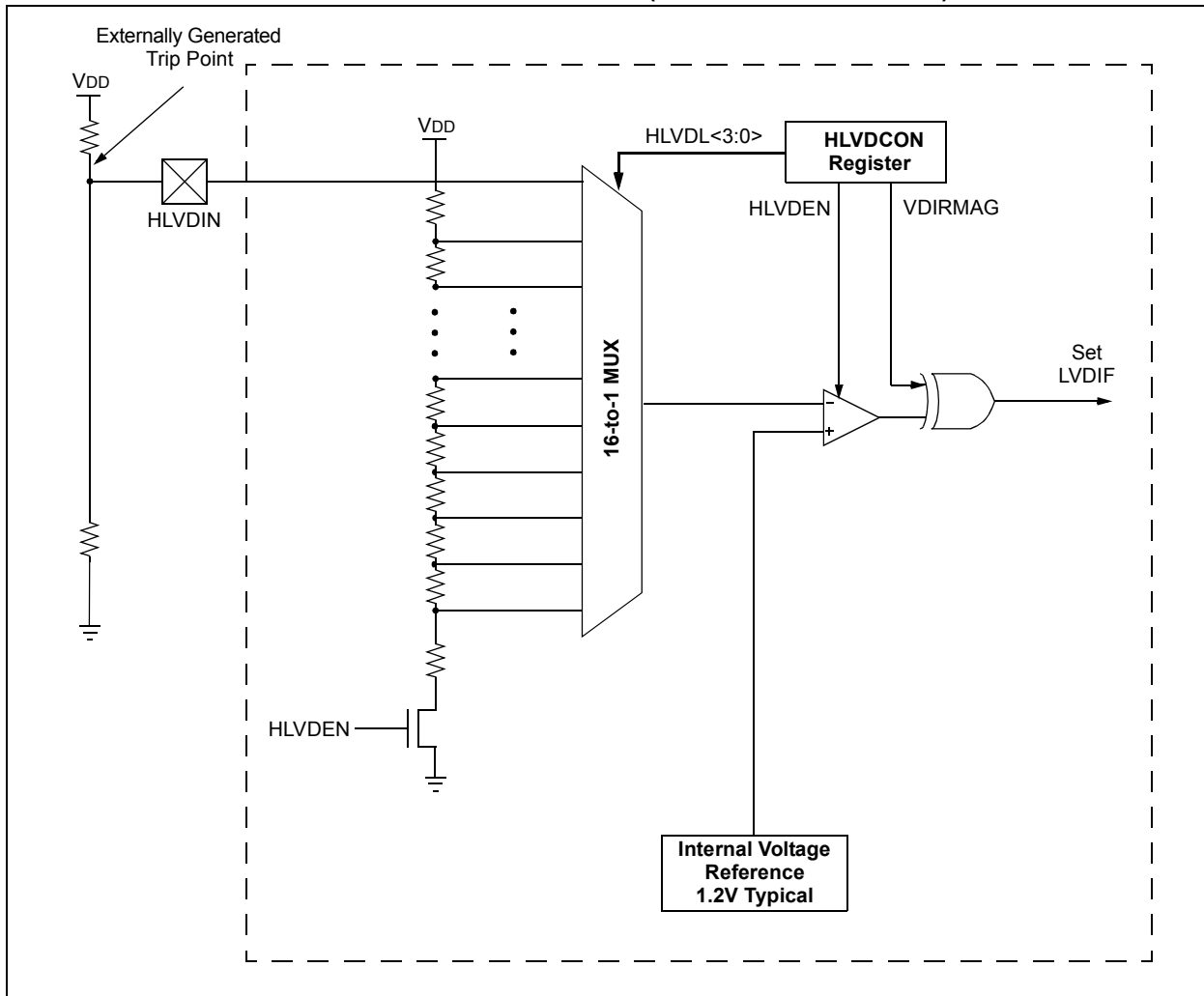
When the HLVD module is enabled, a comparator uses an internally generated reference voltage as the set point. The set point is compared with the trip point, where each node in the resistor divider represents a trip point voltage. The “trip point” voltage is the voltage level at which the device detects a high or low-voltage event, depending on the configuration of the module.

When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal by setting the LVDIF bit.

The trip point voltage is software programmable to any one of 16 values. The trip point is selected by programming the HLVDL<3:0> bits (HLVDCON<3:0>).

Additionally, the HLVD module allows the user to supply the trip voltage to the module from an external source. This mode is enabled when bits, HLVDL<3:0>, are set to ‘1111’. In this state, the comparator input is multiplexed from the external input pin, HLVDIN. This gives users flexibility because it allows them to configure the HLVD interrupt to occur at any voltage in the valid operating range.

FIGURE 23-1: HLVD MODULE BLOCK DIAGRAM (WITH EXTERNAL INPUT)





## 23.2 HLVD Setup

To set up the HLVD module:

1. Disable the module by clearing the HLVDEN bit (HLVDCON<4>).
2. Write the value to the HLVDL<3:0> bits that selects the desired HLVD trip point.
3. Set the VDIRMAG bit to detect one of the following:
  - High voltage (VDIRMAG = 1)
  - Low voltage (VDIRMAG = 0)
4. Enable the HLVD module by setting the HLVDEN bit.
5. Clear the HLVD Interrupt Flag, LVDIF (PIR2<2>), which may have been set from a previous interrupt.
6. If interrupts are desired, enable the HLVD interrupt by setting the HLVDIE and GIE/GIEH bits (PIE2<2> and INTCON<7>).

An interrupt will not be generated until the IRVST bit is set.

## 23.3 Current Consumption

When the module is enabled, the HLVD comparator and voltage divider are enabled and will consume static current. The total current consumption, when enabled, is specified in electrical specification parameter D022B ( $\Delta I_{HLVD}$ ) (**Section 28.2 “DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial)”**).

Depending on the application, the HLVD module does not need to operate constantly. To decrease the current requirements, the HLVD circuitry may only need to be enabled for short periods where the voltage is checked. After doing the check, the HLVD module may be disabled.

## 23.4 HLVD Start-up Time

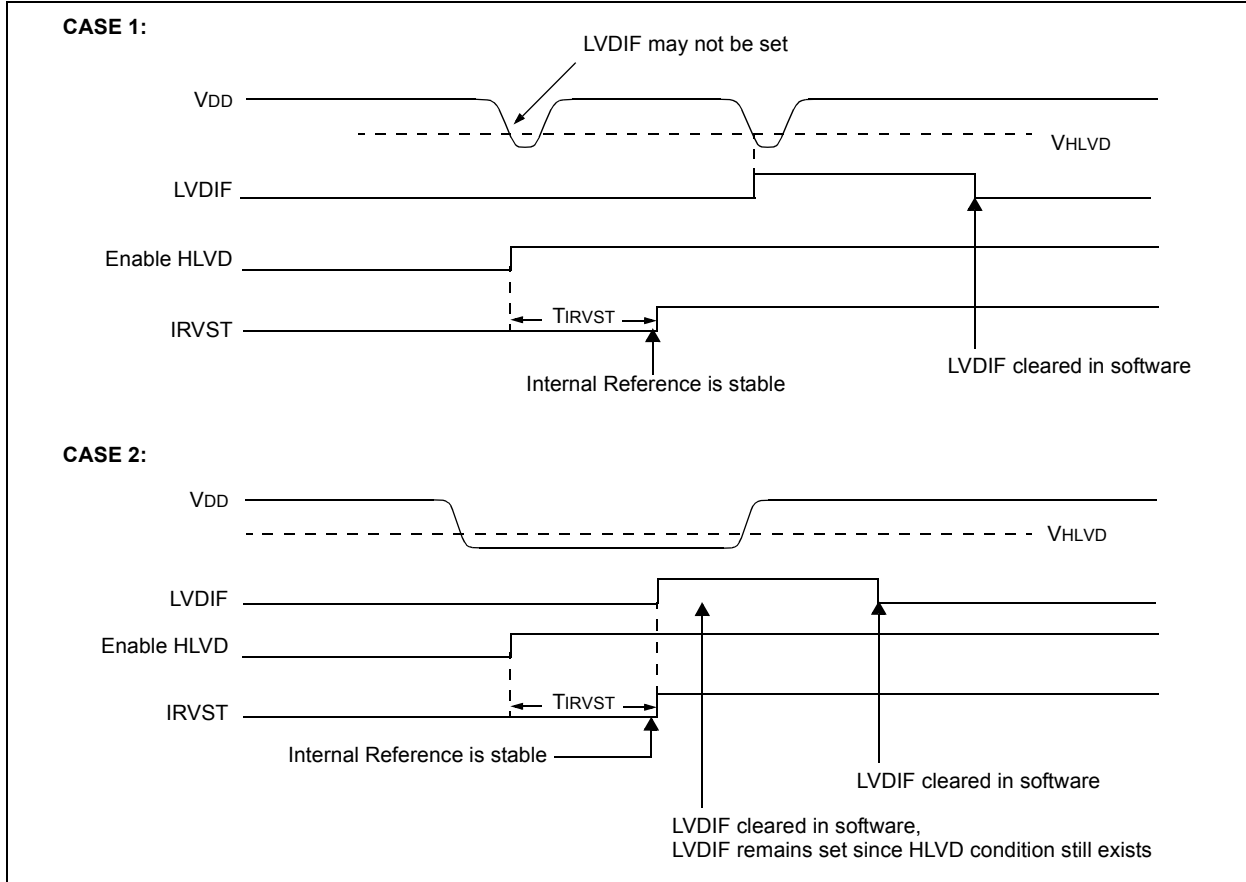
The internal reference voltage of the HLVD module, specified in electrical specification parameter D420 (see Table 28-7 in **Section 28.0 “Electrical Characteristics”**), may be used by other internal circuitry, such as the Programmable Brown-out Reset (BOR).

If the HLVD or other circuits using the voltage reference are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low or high-voltage condition can be reliably detected. This start-up time, T<sub>IRVST</sub>, is an interval that is independent of device clock speed. It is specified in electrical specification parameter 36 (Table 28-13).

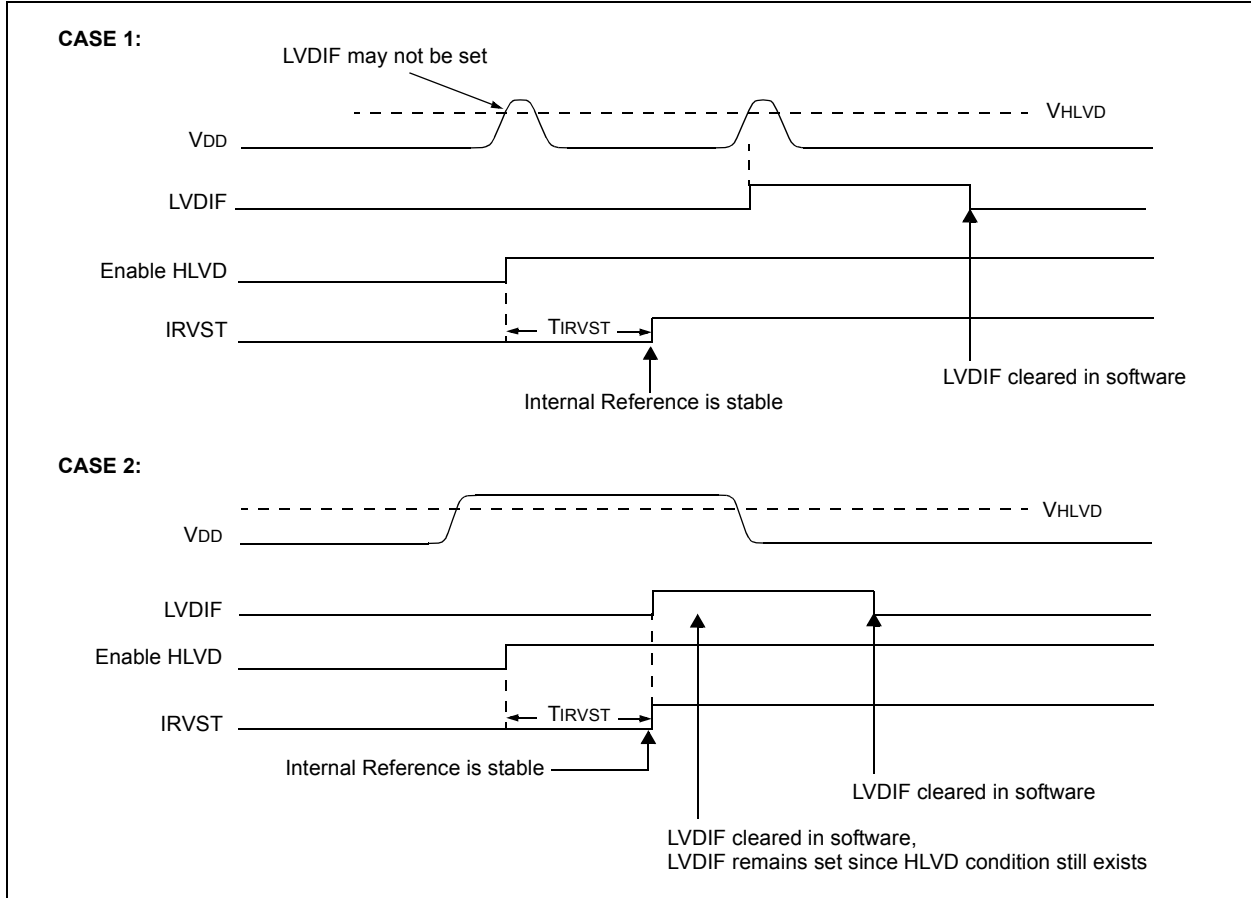
The HLVD interrupt flag is not enabled until T<sub>IRVST</sub> has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval. Refer to Figure 23-2 or Figure 23-3.

# PIC18F46J11 FAMILY

**FIGURE 23-2: LOW-VOLTAGE DETECT OPERATION (VDIRMAG = 0)**



**FIGURE 23-3: HIGH-VOLTAGE DETECT OPERATION (VDIRMAG = 1)**



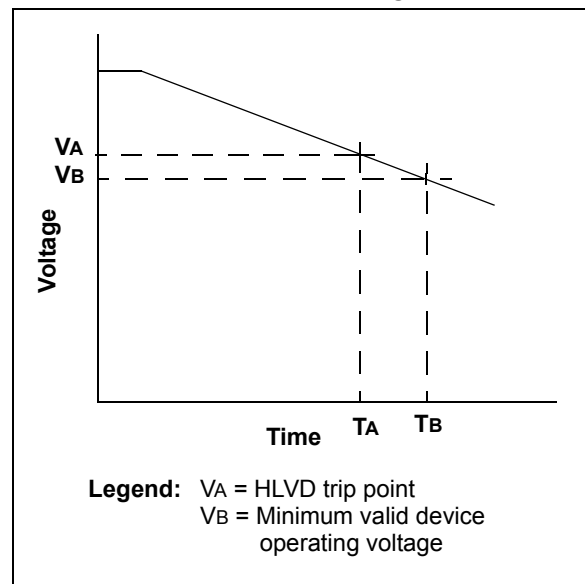
## 23.5 Applications

In many applications, it is desirable to have the ability to detect a drop below, or rise above, a particular threshold. For general battery applications, Figure 23-4 provides a possible voltage curve.

Over time, the device voltage decreases. When the device voltage reaches voltage,  $V_A$ , the HLVD logic generates an interrupt at time,  $T_A$ . The interrupt could cause the execution of an ISR, which would allow the application to perform “housekeeping tasks” and perform a controlled shutdown before the device voltage exits the valid operating range at  $T_B$ .

The HLVD, thus, would give the application a time window, represented by the difference between  $T_A$  and  $T_B$ , to safely exit.

**FIGURE 23-4: TYPICAL HIGH/LOW-VOLTAGE DETECT APPLICATION**



# PIC18F46J11 FAMILY

## 23.6 Operation During Sleep

When enabled, the HLVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the LVDIF bit will be set and the device will wake-up from Sleep. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

## 23.7 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the HLVD module to be turned off.

**TABLE 23-1: REGISTERS ASSOCIATED WITH HIGH/LOW-VOLTAGE DETECT MODULE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	66
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	63
PIR2	OSCFIF	CM1IF	CM2IF	—	BCLIF	LVDIF	TMR3IF	CCP2IF	65
PIE2	OSCFIE	CM1IE	CM2IE	—	BCLIE	LVDIE	TMR3IE	CCP2IE	65
IPR2	OSCFIP	CM1IP	CM2IP	—	BCLIP	LVDIP	TMR3IP	CCP2IP	65

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the HLVD module.

## 24.0 CHARGE TIME MEASUREMENT UNIT (CTMU)

The Charge Time Measurement Unit (CTMU) is a flexible analog module that provides accurate differential time measurement between pulse sources, as well as asynchronous pulse generation. By working with other on-chip analog modules, the CTMU can be used to precisely measure time, measure capacitance, measure relative changes in capacitance or generate output pulses with a specific time delay. The CTMU is ideal for interfacing with capacitive-based sensors.

The module includes the following key features:

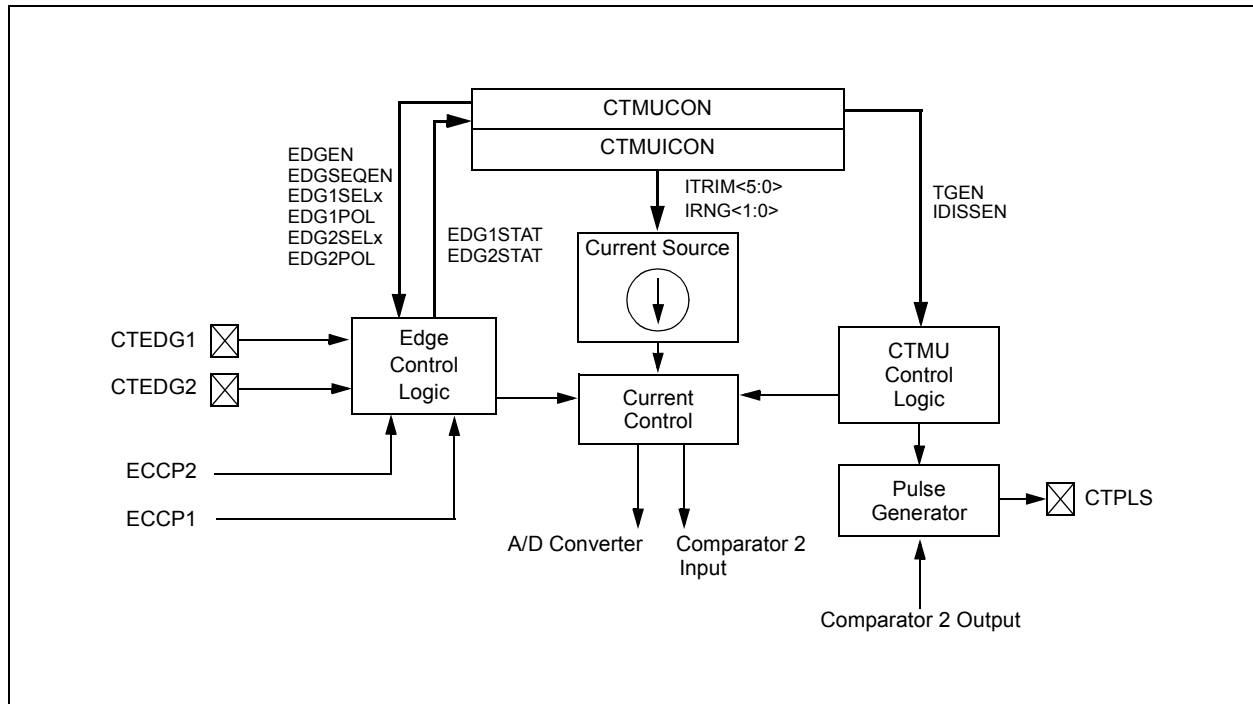
- Up to 13 channels available for capacitive or time measurement input
- On-chip precision current source
- Four-edge input trigger sources
- Polarity control for each edge source
- Control of edge sequence

- Control of response to edges
- Time measurement resolution of 1 nanosecond
- High precision time measurement
- Time delay of external or internal signal asynchronous to system clock
- Accurate current source suitable for capacitive measurement

The CTMU works in conjunction with the A/D Converter to provide up to 13 channels for time or charge measurement, depending on the specific device and the number of A/D channels available. When configured for time delay, the CTMU is connected to one of the analog comparators. The level-sensitive input edge sources can be selected from four sources: two external inputs or ECCP1/ECCP2 Special Event Triggers.

Figure 24-1 provides a block diagram of the CTMU.

**FIGURE 24-1: CTMU BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

## 24.1 CTMU Operation

The CTMU works by using a fixed current source to charge a circuit. The type of circuit depends on the type of measurement being made. In the case of charge measurement, the current is fixed, and the amount of time the current is applied to the circuit is fixed. The amount of voltage read by the A/D is then a measurement of the capacitance of the circuit. In the case of time measurement, the current, as well as the capacitance of the circuit, is fixed. In this case, the voltage read by the A/D is then representative of the amount of time elapsed from the time the current source starts and stops charging the circuit.

If the CTMU is being used as a time delay, both capacitance and current source are fixed, as well as the voltage supplied to the comparator circuit. The delay of a signal is determined by the amount of time it takes the voltage to charge to the comparator threshold voltage.

### 24.1.1 THEORY OF OPERATION

The operation of the CTMU is based on the equation for charge:

$$C = I \cdot \frac{dV}{dT}$$

More simply, the amount of charge measured in coulombs in a circuit is defined as current in amperes ( $I$ ) multiplied by the amount of time in seconds that the current flows ( $t$ ). Charge is also defined as the capacitance in farads ( $C$ ) multiplied by the voltage of the circuit ( $V$ ). It follows that:

$$I \cdot t = C \cdot V.$$

The CTMU module provides a constant, known current source. The A/D Converter is used to measure ( $V$ ) in the equation, leaving two unknowns: capacitance ( $C$ ) and time ( $t$ ). The above equation can be used to calculate capacitance or time, by either the relationship using the known fixed capacitance of the circuit:

$$t = (C \cdot V) / I$$

or by:

$$C = (I \cdot t) / V$$

using a fixed time that the current source is applied to the circuit.

### 24.1.2 CURRENT SOURCE

At the heart of the CTMU is a precision current source, designed to provide a constant reference for measurements. The level of current is user-selectable across three ranges or a total of two orders of magnitude, with the ability to trim the output in  $\pm 2\%$  increments (nominal). The current range is selected by the IRNG<1:0> bits (CTMUICON<9:8>), with a value of '00' representing the lowest range.

Current trim is provided by the ITRIM<5:0> bits (CTMUICON<7:2>). These six bits allow trimming of the current source in steps of approximately 2% per step. Note that half of the range adjusts the current source positively and the other half reduces the current source. A value of '000000' is the neutral position (no change). A value of '100000' is the maximum negative adjustment (approximately -62%) and '011111' is the maximum positive adjustment (approximately +62%).

### 24.1.3 EDGE SELECTION AND CONTROL

CTMU measurements are controlled by edge events occurring on the module's two input channels. Each channel, referred to as Edge 1 and Edge 2, can be configured to receive input pulses from one of the edge input pins (CTEDG1 and CTEDG2) or ECCPx Special Event Triggers. The input channels are level-sensitive, responding to the instantaneous level on the channel rather than a transition between levels. The inputs are selected using the EDG1SEL and EDG2SEL bit pairs (CTMUCONL<3:2 and 6:5>).

In addition to source, each channel can be configured for event polarity using the EDGE2POL and EDGE1POL bits (CTMUCONL<7,4>). The input channels can also be filtered for an edge event sequence (Edge 1 occurring before Edge 2) by setting the EDGSEQEN bit (CTMUCONH<2>).

### 24.1.4 EDGE STATUS

The CTMUCON register also contains two status bits: EDG2STAT and EDG1STAT (CTMUCONL<1:0>). Their primary function is to show if an edge response has occurred on the corresponding channel. The CTMU automatically sets a particular bit when an edge response is detected on its channel. The level-sensitive nature of the input channels also means that the status bits become set immediately if the channel's configuration is changed and is the same as the channel's current state.

The module uses the edge status bits to control the current source output to external analog modules (such as the A/D Converter). Current is only supplied to external modules when only one (but not both) of the status bits is set, and shuts current off when both bits are either set or cleared. This allows the CTMU to measure current only during the interval between edges. After both status bits are set, it is necessary to clear them before another measurement is taken. Both bits should be cleared simultaneously, if possible, to avoid re-enabling the CTMU current source.

In addition to being set by the CTMU hardware, the edge status bits can also be set by software. This is also the user's application to manually enable or disable the current source. Setting either one (but not both) of the bits enables the current source. Setting or clearing both bits at once disables the source.

## 24.1.5 INTERRUPTS

The CTMU sets its interrupt flag (PIR3<2>) whenever the current source is enabled, then disabled. An interrupt is generated only if the corresponding interrupt enable bit (PIE3<2>) is also set. If edge sequencing is not enabled (i.e., Edge 1 must occur before Edge 2), it is necessary to monitor the edge status bits and determine which edge occurred last and caused the interrupt.

## 24.2 CTMU Module Initialization

The following sequence is a general guideline used to initialize the CTMU module:

1. Select the current source range using the IRNG bits (CTMUICON<1:0>).
2. Adjust the current source trim using the ITRIM bits (CTMUICON<7:2>).
3. Configure the edge input sources for Edge 1 and Edge 2 by setting the EDG1SEL and EDG2SEL bits (CTMUCONL<3:2 and 6:5>).
4. Configure the input polarities for the edge inputs using the EDG1POL and EDG2POL bits (CTMUCONL<4,7>). The default configuration is for negative edge polarity (high-to-low transitions).
5. Enable edge sequencing using the EDGSEQEN bit (CTMUCONH<2>). By default, edge sequencing is disabled.
6. Select the operating mode (Measurement or Time Delay) with the TGEN bit. The default mode is Time/Capacitance Measurement.
7. Discharge the connected circuit by setting the IDISSEN bit (CTMUCONH<1>); after waiting a sufficient time for the circuit to discharge, clear IDISSEN.
8. Disable the module by clearing the CTMUEN bit (CTMUCONH<7>).
9. Enable the module by setting the CTMUEN bit.
10. Clear the Edge Status bits: EDG2STAT and EDG1STAT (CTMUCONL<1:0>).
11. Enable both edge inputs by setting the EDGEN bit (CTMUCONH<3>).

Depending on the type of measurement or pulse generation being performed, one or more additional modules may also need to be initialized and configured with the CTMU module:

- Edge Source Generation: In addition to the external edge input pins, both Timer1 and the Output Compare/PWM1 module can be used as edge sources for the CTMU.
- Capacitance or Time Measurement: The CTMU module uses the A/D Converter to measure the voltage across a capacitor that is connected to one of the analog input channels.
- Pulse Generation: When generating system clock independent output pulses, the CTMU module uses Comparator 2 and the associated comparator voltage reference.

## 24.3 Calibrating the CTMU Module

The CTMU requires calibration for precise measurements of capacitance and time, as well as for accurate time delay. If the application only requires measurement of a relative change in capacitance or time, calibration is usually not necessary. An example of this type of application would include a capacitive touch switch, in which the touch circuit has a baseline capacitance, and the added capacitance of the human body changes the overall capacitance of a circuit.

If actual capacitance or time measurement is required, two hardware calibrations must take place: the current source needs calibration to set it to a precise current, and the circuit being measured needs calibration to measure and/or nullify all other capacitance other than that to be measured.

### 24.3.1 CURRENT SOURCE CALIBRATION

The current source on board the CTMU module has a range of  $\pm 60\%$  nominal for each of three current ranges. Therefore, for precise measurements, it is possible to measure and adjust this current source by placing a high precision resistor,  $R_{CAL}$ , onto an unused analog channel. An example circuit is shown in Figure 24-2. The current source measurement is performed using the following steps:

1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Enable the current source by setting EDG1STAT (CTMUCONL<0>).
4. Issue settling time delay.
5. Perform A/D conversion.
6. Calculate the current source current using  $I = V/R_{CAL}$ , where  $R_{CAL}$  is a high precision resistance and  $V$  is measured by performing an A/D conversion.

# PIC18F46J11 FAMILY

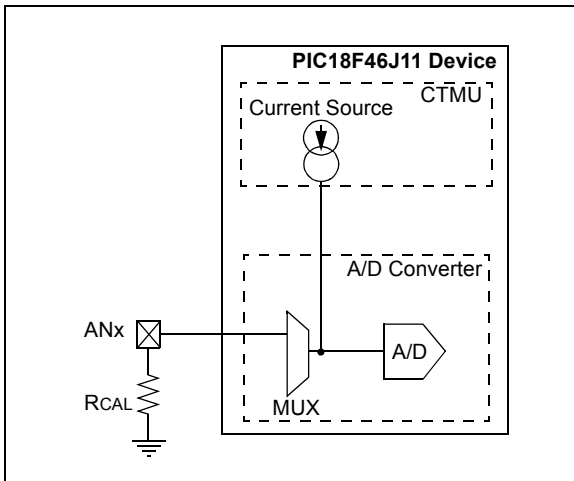
The CTMU current source may be trimmed with the trim bits in CTMUICON using an iterative process to get an exact desired current. Alternatively, the nominal value without adjustment may be used; it may be stored by the software for use in all subsequent capacitive or time measurements.

To calculate the value for  $R_{CAL}$ , the nominal current must be chosen, and then the resistance can be calculated. For example, if the A/D Converter reference voltage is 3.3V, use 70% of full scale, or 2.31V as the desired approximate voltage to be read by the A/D Converter. If the range of the CTMU current source is selected to be 0.55  $\mu\text{A}$ , the resistor value needed is calculated as  $R_{CAL} = 2.31\text{V}/0.55 \mu\text{A}$ , for a value of 4.2 M $\Omega$ . Similarly, if the current source is chosen to be 5.5  $\mu\text{A}$ ,  $R_{CAL}$  would be 420,000 $\Omega$ , and 42,000 $\Omega$  if the current source is set to 55  $\mu\text{A}$ .

A value of 70% of full-scale voltage is chosen to make sure that the A/D Converter was in a range that is well above the noise floor. Keep in mind that if an exact current is chosen, that is to incorporate the trimming bits from CTMUICON, the resistor value of  $R_{CAL}$  may need to be adjusted accordingly.  $R_{CAL}$  may also be adjusted to allow for available resistor values.  $R_{CAL}$  should be of the highest precision available, keeping in mind the amount of precision needed for the circuit that the CTMU will be used to measure. A recommended minimum would be 0.1% tolerance.

The following examples show one typical method for performing a CTMU current calibration. Example 24-1 demonstrates how to initialize the A/D Converter and the CTMU; this routine is typical for applications using both modules. Example 24-2 demonstrates one method for the actual calibration routine.

**FIGURE 24-2: CTMU CURRENT SOURCE CALIBRATION CIRCUIT**





## EXAMPLE 24-1: SETUP FOR CTMU CALIBRATION ROUTINES

```
#include "p18cxxx.h"
/*****
/*Setup CTMU *****/
/*****
void setup(void)

{ //CTMUCON - CTMU Control register

    CTMUCONH = 0x00;    //make sure CTMU is disabled
    CTMUCONL = 0x90;
    //CTMU continues to run when emulator is stopped,CTMU continues
    //to run in idle mode,Time Generation mode disabled, Edges are blocked
    //No edge sequence order, Analog current source not grounded, trigger
    //output disabled, Edge2 polarity = positive level, Edge2 source =
    //source 0, Edge1 polarity = positive level, Edge1 source = source 0,

    //CTMUICON - CTMU Current Control Register
    CTMUICON = 0x01;    //0.55uA, Nominal - No Adjustment

/*****
//Setup AD converter;
/*****

    TRISA=0x04;        //set channel 2 as an input

    // Configured AN2 as an analog channel
    // ANCON0
    ANCON0 = 0XFB;
    // ANCON1
    ANCON1 = 0X1F;

    // ADCON1
    ADCON1bits.ADFM=1;    // Result format 1= Right justified
    ADCON1bits.ADCAL=0;  // Normal A/D conversion operation
    ADCON1bits.ACQT=1;   // Acquisition time 7 = 20TAD 2 = 4TAD 1=2TAD
    ADCON1bits.ADCS=2;   // Clock conversion bits 6= FOSC/64 2=FOSC/32

    ANCON1bits.VBGEN=1;  // Turn on the Bandgap needed for Rev A0 parts

    // ADCON0
    ADCON0bits.VCFG0 =0; // Vref+ = AVdd
    ADCON0bits.VCFG1 =0; // Vref- = AVss
    ADCON0bits.CHS=2;   // Select ADC channel

    ADCON0bits.ADON=1;  // Turn on ADC
}
```

# PIC18F46J11 FAMILY

## EXAMPLE 24-2: CURRENT CALIBRATION ROUTINE

```
#include "p18cxxx.h"

#define COUNT 500 //@ 8MHz = 125uS.
#define DELAY for (i=0;i<COUNT;i++)
#define RCAL .027 //R value is 4200000 (4.2M)
//scaled so that result is in
//1/100th of uA
#define ADSCALE 1023 //for unsigned conversion 10 sig bits
#define ADREF 3.3 //Vdd connected to A/D Vr+

int main(void)
{
    int i;
    int j = 0; //index for loop
    unsigned int Vread = 0;
    double VTot = 0;
    float Vavg=0, Vcal=0, CTMUISrc = 0; //float values stored for calcs

    //assume CTMU and A/D have been setup correctly
    //see Example 25-1 for CTMU & A/D setup
    setup();

    CTMUCONHbits.CTMUEN = 1; //Enable the CTMU
    CTMUCONLbits.EDG1STAT = 0; // Set Edge status bits to zero
    CTMUCONLbits.EDG2STAT = 0;
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1; //drain charge on the circuit
        DELAY; //wait 125us
        CTMUCONHbits.IDISSEN = 0; //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1; //Begin charging the circuit
        //using CTMU current source
        DELAY; //wait for 125us
        CTMUCONLbits.EDG1STAT = 0; //Stop charging circuit

        PIR1bits.ADIF = 0; //make sure A/D Int not set
        ADCON0bits.GO=1; //and begin A/D conv.
        while(!PIR1bits.ADIF); //Wait for A/D convert complete

        Vread = ADRES; //Get the value from the A/D
        PIR1bits.ADIF = 0; //Clear A/D Interrupt Flag
        VTot += Vread; //Add the reading to the total
    }

    Vavg = (float) (VTot/10.000); //Average of 10 readings
    Vcal = (float) (Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL; //CTMUISrc is in 1/100ths of uA
}
```

## 24.3.2 CAPACITANCE CALIBRATION

There is a small amount of capacitance from the internal A/D Converter sample capacitor as well as stray capacitance from the circuit board traces and pads that affect the precision of capacitance measurements. A measurement of the stray capacitance can be taken by making sure the desired capacitance to be measured has been removed. The measurement is then performed using the following steps:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT (= 1).
3. Wait for a fixed delay of time  $t$ .
4. Clear EDG1STAT.
5. Perform an A/D conversion.
6. Calculate the stray and A/D sample capacitances:

$$C_{\text{OFFSET}} = C_{\text{STRAY}} + C_{\text{AD}} = (I \cdot t) / V$$

where  $I$  is known from the current source measurement step,  $t$  is a fixed delay and  $V$  is measured by performing an A/D conversion.

This measured value is then stored and used for calculations of time measurement or subtracted for capacitance measurement. For calibration, it is expected that the capacitance of  $C_{\text{STRAY}} + C_{\text{AD}}$  is approximately known.  $C_{\text{AD}}$  is approximately 4 pF.

An iterative process may need to be used to adjust the time,  $t$ , that the circuit is charged to obtain a reasonable voltage reading from the A/D Converter. The value of  $t$  may be determined by setting  $C_{\text{OFFSET}}$  to a theoretical value, then solving for  $t$ . For example, if  $C_{\text{STRAY}}$  is theoretically calculated to be 11 pF, and  $V$  is expected to be 70% of  $V_{\text{DD}}$ , or 2.31V, then  $t$  would be:

$$(4 \text{ pF} + 11 \text{ pF}) \cdot 2.31\text{V} / 0.55 \mu\text{A}$$

or 63  $\mu\text{s}$ .

See Example 24-3 for a typical routine for CTMU capacitance calibration.

# PIC18F46J11 FAMILY

## EXAMPLE 24-3: CAPACITANCE CALIBRATION ROUTINE

```
#include "p18cxxx.h"

#define COUNT 25 //@ 8MHz INTFRC = 62.5 us.
#define ETIME COUNT*2.5 //time in uS
#define DELAY for(i=0;i<COUNT;i++)
#define ADSCALE 1023 //for unsigned conversion 10 sig
bits
#define ADREF 3.3 //Vdd connected to A/D Vr+
#define RCAL .027 //R value is 4200000 (4.2M)
//scaled so that result is in
//1/100th of uA

int main(void)
{
    int i;
    int j = 0; //index for loop
    unsigned int Vread = 0;
    float CTMUISrc, CTMUCap, Vavg, VTot, Vcal;

    //assume CTMU and A/D have been setup correctly
    //see Example 25-1 for CTMU & A/D setup
    setup();

    CTMUCONHbits.CTMUEN = 1; //Enable the CTMU
    CTMUCONLbits.EDG1STAT = 0; // Set Edge status bits to zero
    CTMUCONLbits.EDG2STAT = 0;
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1; //drain charge on the circuit
        DELAY; //wait 125us
        CTMUCONHbits.IDISSEN = 0; //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1; //Begin charging the circuit
        //using CTMU current source
        DELAY; //wait for 125us
        CTMUCONLbits.EDG1STAT = 0; //Stop charging circuit

        PIR1bits.ADIF = 0; //make sure A/D Int not set
        ADCON0bits.GO=1; //and begin A/D conv.
        while(!PIR1bits.ADIF); //Wait for A/D convert complete

        Vread = ADRES; //Get the value from the A/D
        PIR1bits.ADIF = 0; //Clear A/D Interrupt Flag
        VTot += Vread; //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000); //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL; //CTMUISrc is in 1/100ths of uA
    CTMUCap = (CTMUISrc*ETIME/Vcal)/100;
}
```

## 24.4 Measuring Capacitance with the CTMU

There are two separate methods of measuring capacitance with the CTMU. The first is the absolute method, in which the actual capacitance value is desired. The second is the relative method, in which the actual capacitance is not needed, rather an indication of a change in capacitance is required.

### 24.4.1 ABSOLUTE CAPACITANCE MEASUREMENT

For absolute capacitance measurements, both the current and capacitance calibration steps found in **Section 24.3 “Calibrating the CTMU Module”** should be followed. Capacitance measurements are then performed using the following steps:

1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Set EDG1STAT.
4. Wait for a fixed delay,  $T$ .
5. Clear EDG1STAT.
6. Perform an A/D conversion.
7. Calculate the total capacitance,  $C_{TOTAL} = (I * T)/V$ , where  $I$  is known from the current source measurement step (see **Section 24.3.1 “Current Source Calibration”**),  $T$  is a fixed delay and  $V$  is measured by performing an A/D conversion.
8. Subtract the stray and A/D capacitance ( $C_{OFFSET}$  from **Section 24.3.2 “Capacitance Calibration”**) from  $C_{TOTAL}$  to determine the measured capacitance.

### 24.4.2 RELATIVE CHARGE MEASUREMENT

An application may not require precise capacitance measurements. For example, when detecting a valid press of a capacitance-based switch, detecting a relative change of capacitance is of interest. In this type of application, when the switch is open (or not touched), the total capacitance is the capacitance of the combination of the board traces, the A/D Converter, etc. A larger voltage will be measured by the A/D Converter. When the switch is closed (or is touched), the total capacitance is larger due to the addition of the capacitance of the human body to the above listed capacitances, and a smaller voltage will be measured by the A/D Converter.

Detecting capacitance changes is easily accomplished with the CTMU using these steps:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Wait for a fixed delay.
4. Clear EDG1STAT.
5. Perform an A/D conversion.

The voltage measured by performing the A/D conversion is an indication of the relative capacitance. Note that in this case, no calibration of the current source or circuit capacitance measurement is needed. See Example 24-4 for a sample software routine for a capacitive touch switch.

# PIC18F46J11 FAMILY

---

## EXAMPLE 24-4: ROUTINE FOR CAPACITIVE TOUCH SWITCH

```
#include "p18cxxx.h"

#define COUNT 500                //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define OPENSW 1000              //Un-pressed switch value
#define TRIP 300                 //Difference between pressed
                                //and un-pressed switch
#define HYST 65                  //amount to change
                                //from pressed to un-pressed

#define PRESSED 1
#define UNPRESSED 0

int main(void)
{
    unsigned int Vread;          //storage for reading
    unsigned int switchState;
    int i;

    //assume CTMU and A/D have been setup correctly
    //see Example 25-1 for CTMU & A/D setup
    setup();

    CTMUCONHbits.CTMUEN = 1;     // Enable the CTMU
    CTMUCONLbits.EDG1STAT = 0;   // Set Edge status bits to zero
    CTMUCONLbits.EDG2STAT = 0;
    CTMUCONHbits.IDISSEN = 1;    //drain charge on the circuit
    DELAY;                       //wait 125us
    CTMUCONHbits.IDISSEN = 0;    //end drain of circuit

    CTMUCONLbits.EDG1STAT = 1;   //Begin charging the circuit
                                //using CTMU current source
    DELAY;                       //wait for 125us
    CTMUCONLbits.EDG1STAT = 0;   //Stop charging circuit

    PIR1bits.ADIF = 0;           //make sure A/D Int not set
    ADCON0bits.GO=1;             //and begin A/D conv.
    while(!PIR1bits.ADIF);      //Wait for A/D convert complete

    Vread = ADRES;               //Get the value from the A/D

    if(Vread < OPENSW - TRIP)
    {
        switchState = PRESSED;
    }
    else if(Vread > OPENSW - TRIP + HYST)
    {
        switchState = UNPRESSED;
    }
}
```

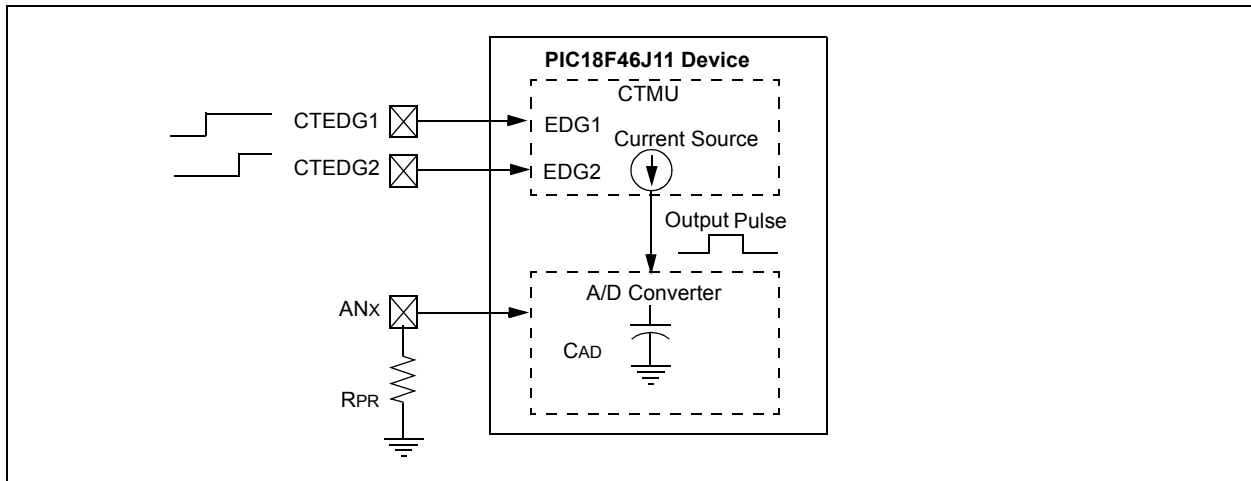
## 24.5 Measuring Time with the CTMU Module

Time can be precisely measured after the ratio ( $C/I$ ) is measured from the current and capacitance calibration step by following these steps:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Set EDG2STAT.
4. Perform an A/D conversion.
5. Calculate the time between edges as  $T = (C/I) * V$ , where  $I$  is calculated in the current calibration step (Section 24.3.1 “Current Source Calibration”),  $C$  is calculated in the capacitance calibration step (Section 24.3.2 “Capacitance Calibration”) and  $V$  is measured by performing the A/D conversion.

It is assumed that the time measured is small enough that the capacitance,  $C_{OFFSET}$ , provides a valid voltage to the A/D Converter. For the smallest time measurement, always set the A/D Channel Select register (AD1CHS) to an unused A/D channel; the corresponding pin for which is not connected to any circuit board trace. This minimizes added stray capacitance, keeping the total circuit capacitance close to that of the A/D Converter itself (4-5 pF). To measure longer time intervals, an external capacitor may be connected to an A/D channel and this channel selected when making a time measurement.

**FIGURE 24-3: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR TIME MEASUREMENT**



# PIC18F46J11 FAMILY

## 24.6 Creating a Delay with the CTMU Module

A unique feature on board the CTMU module is its ability to generate system clock independent output pulses based on an external capacitor value. This is accomplished using the internal comparator voltage reference module, Comparator 2 input pin and an external capacitor. The pulse is output onto the CTPLS pin. To enable this mode, set the TGEN bit.

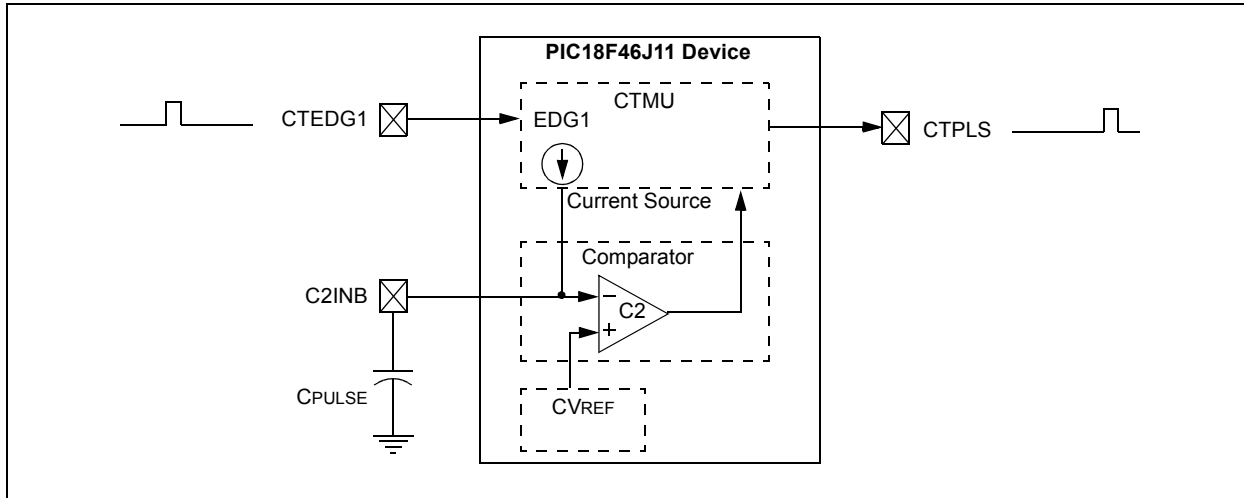
See Figure 24-4 for an example circuit.  $C_{PULSE}$  is chosen by the user to determine the output pulse width on CTPLS. The pulse width is calculated by  $T = (C_{PULSE}/I) * V$ , where  $I$  is known from the current source measurement step (Section 24.3.1 “Current Source Calibration”) and  $V$  is the internal reference voltage (CVREF).

An example use of this feature is for interfacing with variable capacitive-based sensors, such as a humidity sensor. As the humidity varies, the pulse width output on CTPLS will vary. The CTPLS output pin can be connected to an input capture pin and the varying pulse width is measured to determine the humidity in the application.

Follow these steps to use this feature:

1. Initialize Comparator 2.
2. Initialize the comparator voltage reference.
3. Initialize the CTMU and enable time delay generation by setting the TGEN bit.
4. Set EDG1STAT.
5. When  $C_{PULSE}$  charges to the value of the voltage reference trip point, an output pulse is generated on CTPLS.

**FIGURE 24-4: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR PULSE DELAY GENERATION**



## 24.7 Operation During Sleep/Idle Modes

### 24.7.1 SLEEP MODE AND DEEP SLEEP MODES

When the device enters any Sleep mode, the CTMU module current source is always disabled. If the CTMU is performing an operation that depends on the current source when Sleep mode is invoked, the operation may not terminate correctly. Capacitance and time measurements may return erroneous values.

### 24.7.2 IDLE MODE

The behavior of the CTMU in Idle mode is determined by the CTMUSIDL bit (CTMUCONH<5>). If CTMUSIDL is cleared, the module will continue to operate in Idle mode. If CTMUSIDL is set, the module's current source is disabled when the device enters Idle mode. If the

module is performing an operation when Idle mode is invoked, in this case, the results will be similar to those with Sleep mode.

## 24.8 Effects of a Reset on CTMU

Upon Reset, all registers of the CTMU are cleared. This leaves the CTMU module disabled, its current source is turned off and all configuration options return to their default settings. The module needs to be re-initialized following any Reset.

If the CTMU is in the process of taking a measurement at the time of Reset, the measurement will be lost. A partial charge may exist on the circuit that was being measured, and should be properly discharged before the CTMU makes subsequent attempts to make a measurement. The circuit is discharged by setting and then clearing the IDISSEN bit (CTMUCONH<1>) while the A/D Converter is connected to the appropriate channel.



# PIC18F46J11 FAMILY

## 24.9 Registers

There are three control registers for the CTMU:

- CTMUCONH
- CTMUCONL
- CTMUICON

The CTMUCONH and CTMUCONL registers (Register 24-1 and Register 24-2) contain control bits for configuring the CTMU module edge source selection, edge source polarity selection, edge sequencing, A/D trigger, analog circuit capacitor discharge and enables. The CTMUICON register (Register 24-3) has bits for selecting the current source range and current source trim.

### REGISTER 24-1: CTMUCONH: CTMU CONTROL REGISTER HIGH (ACCESS FB3h)

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	—
bit 7							bit 0

#### Legend:

R = Readable bit  
-n = Value at POR

W = Writable bit  
'1' = Bit is set

U = Unimplemented bit, read as '0'  
'0' = Bit is cleared  
x = Bit is unknown

bit 7	<b>CTMUEN:</b> CTMU Enable bit 1 = Module is enabled 0 = Module is disabled
bit 6	<b>Unimplemented:</b> Read as '0'
bit 5	<b>CTMUSIDL:</b> Stop in Idle Mode bit 1 = Discontinue module operation when device enters Idle mode 0 = Continue module operation in Idle mode
bit 4	<b>TGEN:</b> Time Generation Enable bit 1 = Enables edge delay generation 0 = Disables edge delay generation
bit 3	<b>EDGEN:</b> Edge Enable bit 1 = Edges are not blocked 0 = Edges are blocked
bit 2	<b>EDGSEQEN:</b> Edge Sequence Enable bit 1 = Edge 1 event must occur before Edge 2 event can occur 0 = No edge sequence is needed
bit 1	<b>IDISSEN:</b> Analog Current Source Control bit 1 = Analog current source output is grounded 0 = Analog current source output is not grounded
bit 0	<b>Reserved:</b> Write as '0'

# PIC18F46J11 FAMILY

## REGISTER 24-2: CTMUCONL: CTMU CONTROL REGISTER LOW (ACCESS FB2h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7      **EDG2POL:** Edge 2 Polarity Select bit  
 1 = Edge 2 programmed for a positive edge response  
 0 = Edge 2 programmed for a negative edge response
- bit 6-5    **EDG2SEL<1:0>:** Edge 2 Source Select bits  
 11 = CTEDG1 pin  
 10 = CTEDG2 pin  
 01 = ECCP1 Special Event Trigger  
 00 = ECCP2 Special Event Trigger
- bit 4      **EDG1POL:** Edge 1 Polarity Select bit  
 1 = Edge 1 programmed for a positive edge response  
 0 = Edge 1 programmed for a negative edge response
- bit 3-2    **EDG1SEL<1:0>:** Edge 1 Source Select bits  
 11 = CTEDG1 pin  
 10 = CTEDG2 pin  
 01 = ECCP1 Special Event Trigger  
 00 = ECCP2 Special Event Trigger
- bit 1      **EDG2STAT:** Edge 2 Status bit  
 1 = Edge 2 event has occurred  
 0 = Edge 2 event has not occurred
- bit 0      **EDG1STAT:** Edge 1 Status bit  
 1 = Edge 1 event has occurred  
 0 = Edge 1 event has not occurred

# PIC18F46J11 FAMILY

## REGISTER 24-3: CTMUICON: CTMU CURRENT CONTROL REGISTER (ACCESS FB1h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
bit 7							bit 0

### Legend:

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

bit 7-2            **ITRIM<5:0>**: Current Source Trim bits  
 011111 = Maximum positive change from nominal current  
 011110  
 .  
 .  
 .  
 000001 = Minimum positive change from nominal current  
 000000 = Nominal current output specified by IRNG<1:0>  
 111111 = Minimum negative change from nominal current  
 .  
 .  
 .  
 100010  
 100001 = Maximum negative change from nominal current

bit 1-0            **IRNG<1:0>**: Current Source Range Select bits  
 11 = 100 × Base current  
 10 = 10 × Base current  
 01 = Base current level (0.55 μA nominal)  
 00 = Current source disabled

**TABLE 24-1: REGISTERS ASSOCIATED WITH CTMU MODULE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page:
CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	—	65
CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	65
CTMUICON	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	65

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

# PIC18F46J11 FAMILY

---

NOTES:

## 25.0 SPECIAL FEATURES OF THE CPU

PIC18F46J11 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 (FSCM)
- Two-Speed Start-up
- Code Protection
- In-Circuit Serial Programming (ICSP)

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 PIC18F46J11 family of devices have a configurable Watchdog Timer (WDT), 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.

### 25.1 Configuration Bits

The Configuration bits can be programmed to select various device configurations. The configuration data is stored in the last four words of Flash program memory; Figure 5-1 depicts this. The configuration data gets loaded into the volatile Configuration registers, CONFIG1L through CONFIG4H, which are readable and mapped to program memory starting at location 300000h.

Table 25-2 provides a complete list. A detailed explanation of the various bit functions is provided in Register 25-1 through Register 25-6.

### 25.1.1 CONSIDERATIONS FOR CONFIGURING THE PIC18F46J11 FAMILY DEVICES

Unlike some previous PIC18 microcontrollers, devices of the PIC18F46J11 family do not use persistent memory registers to store configuration information. The Configuration registers, CONFIG1L through CONFIG4H, are implemented as volatile memory.

Immediately after power-up, or after a device Reset, the microcontroller hardware automatically loads the CONFIG1L through CONFIG4L registers with configuration data stored in nonvolatile Flash program memory. The last four words of Flash program memory, known as the Flash Configuration Words (FCW), are used to store the configuration data.

Table 25-1 provides the Flash program memory, which will be loaded into the corresponding Configuration register.

When creating applications for these devices, users should always specifically allocate the location of the FCW for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The four Most Significant bits (MSb) of the FCW corresponding to CONFIG1H, CONFIG2H, CONFIG3H and CONFIG4H should always be programmed to ‘1111’. This makes these FCWs appear to be NOP instructions in the remote event that their locations are ever executed by accident.

To prevent inadvertent configuration changes during code execution, the Configuration registers, CONFIG1L through CONFIG4L, are loaded only once per power-up or Reset cycle. User’s firmware can still change the configuration by using self-reprogramming to modify the contents of the FCW.

Modifying the FCW will not change the active contents being used in the CONFIG1L through CONFIG4H registers until after the device is reset.

# PIC18F46J11 FAMILY

**TABLE 25-1: MAPPING OF THE FLASH CONFIGURATION WORDS TO THE CONFIGURATION REGISTERS**

Configuration Register (Volatile)	Configuration Register Address	Flash Configuration Byte Address
CONFIG1L	300000h	XXXF8h
CONFIG1H	300001h	XXXF9h
CONFIG2L	300002h	XXXFAh
CONFIG2H	300003h	XXXFBh
CONFIG3L	300004h	XXXFCh
CONFIG3H	300005h	XXXFDh
CONFIG4L	300006h	XXXFEh
CONFIG4H	300007h	XXXFFh

**TABLE 25-2: CONFIGURATION BITS AND DEVICE IDs**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/Unprog. Value <sup>(1)</sup>	
300000h	CONFIG1L	DEBUG	XINST	STVREN	—	—	—	WDTEN	111- ---1	
300001h	CONFIG1H	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	—	CP0	—	1111 -1--	
300002h	CONFIG2L	IESO	FCMEN	—	LPT1OSC	T1DIG	FOSC2	FOSC1	FOSC0	11-1 1111
300003h	CONFIG2H	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	WDTPS3	WDTPS2	WDTPS1	WDTPS0	1111 1111
300004h	CONFIG3L	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0	DSWDTEN	DSBOREN	RTCOSC	DSWDTOSC	1111 1111
300005h	CONFIG3H	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	MSSPMSK	—	—	IOL1WAY	1111 1--1
300006h	CONFIG4L	WPCFG	WPEND	WPPF5	WPPF4	WPPF3	WPPF2	WPPF1	WPPF0	1111 1111
300007h	CONFIG4H	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	__ <sup>(2)</sup>	—	—	—	WPDIS	1111 ---1
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxx0 0000 <sup>(3)</sup>
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0100 00xx <sup>(3)</sup>

**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 programmed to '1'. This ensures that the location is executed as a NOP if it is accidentally executed.
- 3:** See Register 25-9 and Register 25-10 for DEVID values. These registers are read-only and cannot be programmed by the user.

# PIC18F46J11 FAMILY

## REGISTER 25-1: CONFIG1L: CONFIGURATION REGISTER 1 LOW (BYTE ADDRESS 300000h)

R/WO-1	R/WO-1	R/WO-1	U-0	U-1	U-1	U-1	R/WO-1
$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN
bit 7							bit 0

### Legend:

R = Readable bit

WO = Write-Once bit

U = Unimplemented bit, read as '0'

-n = Value at Reset

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

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)

# PIC18F46J11 FAMILY

## REGISTER 25-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-1	U-1	U-1	U-1	U-0	R/WO-1	U-0	U-0
—	—	—	—	—	CP0	—	—
bit 7							bit 0

### Legend:

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
-n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

bit 7-4                      **Unimplemented:** Program the corresponding Flash Configuration bit to '1'

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



# PIC18F46J11 FAMILY

**REGISTER 25-3: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)**

R/WO-1	R/WO-1	U-0	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
IESO	FCMEN	—	LPT1OSC	T1DIG	FOSC2	FOSC1	FOSC0
bit 7							bit 0

**Legend:**

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
 -n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

- 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                      **Unimplemented:** Read as '0'
- bit 4                      **LPT1OSC:** Low-Power Timer1 Oscillator Enable bit  
                             1 = Timer1 oscillator configured for high-power operation  
                             0 = Timer1 oscillator configured for low-power operation
- bit 3                      **T1DIG:** Secondary Clock Source T1OSCEN Enforcement bit  
                             1 = Secondary oscillator clock source may be selected (OSCCON<1:0> = 01) regardless of the  
                                     T1OSCEN (T1CON<3>) state  
                             0 = Secondary oscillator clock source may not be selected unless T1CON<3> = 1
- bit 2-0                      **FOSC<2:0>:** Oscillator Selection bits  
                             111 = ECPLL oscillator with PLL software controlled, CLKO on RA6  
                             110 = EC oscillator with CLKO on RA6  
                             101 = HSPLL oscillator with PLL software controlled  
                             100 = HS oscillator  
                             011 = INTOSCPLLO, internal oscillator with PLL software controlled, CLKO on RA6, port function on  
                                     RA7  
                             010 = INTOSCPLL, internal oscillator with PLL software controlled, port function on RA6 and RA7  
                             001 = INTOSCO internal oscillator block (INTRC/INTOSC) with CLKO on RA6, port function on RA7  
                             000 = INTOSC internal oscillator block (INTRC/INTOSC), port function on RA6 and RA7

# PIC18F46J11 FAMILY

## REGISTER 25-4: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

U-1	U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

### Legend:

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
 -n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

bit 7-4                      **Unimplemented:** Program the corresponding Flash Configuration bit to '1'

bit 3-0                      **WDTPS<3:0>:** Watchdog Timer Postscale Select bits

- 1111 = 1:32,768
- 1110 = 1:16,384
- 1101 = 1:8,192
- 1100 = 1:4,096
- 1011 = 1:2,048
- 1010 = 1:1,024
- 1001 = 1:512
- 1000 = 1:256
- 0111 = 1:128
- 0110 = 1:64
- 0101 = 1:32
- 0100 = 1:16
- 0011 = 1:8
- 0010 = 1:4
- 0001 = 1:2
- 0000 = 1:1

# PIC18F46J11 FAMILY

## REGISTER 25-5: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
DSWDTPS3 <sup>(1)</sup>	DSWDTPS2 <sup>(1)</sup>	DSWDTPS1 <sup>(1)</sup>	DSWDTPS0 <sup>(1)</sup>	DSWDTEN <sup>(1)</sup>	DSBOREN <sup>(1)</sup>	RTCOSC	DSWDTOSC <sup>(1)</sup>
bit 7							bit 0

### Legend:

R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value at Reset	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

bit 7-4 **DSWDTPS<3:0>**: Deep Sleep Watchdog Timer Postscale Select bits<sup>(1)</sup>

The DSWDT prescaler is 32. This creates an approximate base time unit of 1 ms.

1111 = 1:2,147,483,648 (25.7 days)

1110 = 1:536,870,912 (6.4 days)

1101 = 1:134,217,728 (38.5 hours)

1100 = 1:33,554,432 (9.6 hours)

1011 = 1:8,388,608 (2.4 hours)

1010 = 1:2,097,152 (36 minutes)

1001 = 1:524,288 (9 minutes)

1000 = 1:131,072 (135 seconds)

0111 = 1:32,768 (34 seconds)

0110 = 1:8,192 (8.5 seconds)

0101 = 1:2,048 (2.1 seconds)

0100 = 1:512 (528 ms)

0011 = 1:128 (132 ms)

0010 = 1:32 (33 ms)

0001 = 1:8 (8.3 ms)

0000 = 1:2 (2.1 ms)

bit 3 **DSWDTEN**: Deep Sleep Watchdog Timer Enable bit<sup>(1)</sup>

1 = DSWDT enabled

0 = DSWDT disabled

bit 2 **DSBOREN**: Deep Sleep BOR Enable bit<sup>(1)</sup>

1 = BOR enabled in Deep Sleep (when using PIC18FXXJXX device)

0 = BOR disabled in Deep Sleep (does not affect operation in non Deep Sleep modes)

bit 1 **RTCOSC**: RTCC Reference Clock Select bit

1 = RTCC uses T1OSC/T1CKI as reference clock

0 = RTCC uses INTRC as reference clock

bit 0 **DSWDTOSC**: DSWDT Reference Clock Select bit<sup>(1)</sup>

1 = DSWDT uses INTRC as reference clock

0 = DSWDT uses T1OSC/T1CKI as reference clock

**Note 1:** Deep Sleep bits are not available on "LF" devices.

# PIC18F46J11 FAMILY

## REGISTER 25-6: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

U-1	U-1	U-1	U-1	R/WO-1	U-0	U-0	R/WO-1
—	—	—	—	MSSPMSK	—	—	IOL1WAY
bit 7							bit 0

### Legend:

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
 -n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

- bit 7-4            **Unimplemented:** Program the corresponding Flash Configuration bit to '1'
- bit 3            **MSSPMSK:** MSSP 7-Bit Address Masking Mode Enable bit
  - 1 = 7-Bit Address Masking mode enabled
  - 0 = 5-Bit Address Masking mode enabled
- bit 2-1           **Unimplemented:** Read as '0'
- bit 0            **IOL1WAY:** IOLOCK One-Way Set Enable bit
  - 1 = IOLOCK bit (PPSCON<0>) can be set once, provided the unlock sequence has been completed. Once set, the Peripheral Pin Select registers cannot be written to a second time.
  - 0 = IOLOCK bit (PPSCON<0>) can be set and cleared as needed, provided the unlock sequence has been completed

## REGISTER 25-7: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
WPCFG	WPEND	WFPF5 <sup>(2)</sup>	WFPF4 <sup>(3)</sup>	WFPF3	WFPF2	WFPF1	WFPF0
bit 7							bit 0

### Legend:

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
 -n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

- bit 7            **WPCFG:** Write/Erase Protect Configuration Region Select bit (valid when WPDIS = 0)
  - 1 = Configuration Words page is not erase/write-protected, regardless of WPEND and WFPF<5:0> settings<sup>(1)</sup>
  - 0 = Configuration Words page is erase/write-protected, regardless of WPEND and WFPF<5:0><sup>(1)</sup>
- bit 6            **WPEND:** Write/Erase Protect Region Select bit (valid when WPDIS = 0)
  - 1 = Flash pages WFPF<5:0> to (Configuration Words page) are write/erase protected
  - 0 = Flash pages 0 to WFPF<5:0> are erase/write-protected
- bit 5-0           **WFPF<5:0>:** Write/Erase Protect Page Start/End Location bits<sup>(2,3)</sup>  
 Used with WPEND bit to define which pages in Flash will be erase/write-protected.

**Note 1:** The "Configuration Words page" contains the FCWs and is the last page of implemented Flash memory on a given device. Each page consists of 1,024 bytes. For example, on a device with 64 Kbytes of Flash, the first page is 0 and the last page (Configuration Words page) is 63 (3Fh).

**2:** Not available on 32K devices.

**3:** Not available on 32K and 16K devices.

# PIC18F46J11 FAMILY

## REGISTER 25-8: CONFIG4H: CONFIGURATION REGISTER 4 HIGH (BYTE ADDRESS 300007h)

U-1	U-1	U-1	U-1	U-0	U-0	U-0	R/WO-1
—	—	—	—	—	—	—	WPDIS
bit 7							bit 0

### Legend:

R = Readable bit                      WO = Write-Once bit                      U = Unimplemented bit, read as '0'  
-n = Value at Reset                      '1' = Bit is set                      '0' = Bit is cleared                      x = Bit is unknown

- bit 7-4                      **Unimplemented:** Program the corresponding Flash Configuration bit to '1'
- bit 3-1                      **Unimplemented:** Read as '0'
- bit 0                      **WPDIS:** Write-Protect Disable bit  
1 = WPPF<5:0>, WPEND and WPCFG bits ignored; all Flash memory may be erased or written  
0 = WPPF<5:0>, WPEND and WPCFG bits enabled; erase/write-protect active for the selected region(s)

## REGISTER 25-9: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F46J11 FAMILY DEVICES (BYTE ADDRESS 3FFFFEh)

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

### Legend:

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

- bit 7-5                      **DEV<2:0>:** Device ID bits  
These bits are used with DEV<10:3> bits in Device ID Register 2 to identify the part number. See Register 25-10.
- bit 4-0                      **REV<4:0>:** Revision ID bits  
These bits are used to indicate the device revision.

# PIC18F46J11 FAMILY

## REGISTER 25-10: DEVID2: DEVICE ID REGISTER 2 FOR PIC18F46J11 FAMILY DEVICES (BYTE ADDRESS 3FFFFFh)

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at Reset

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 7-0

**DEV<10:3>**: Device ID bits

These bits are used with the DEV<2:0> bits in the Device ID Register 1 to identify the part number.

DEV<10:3> (DEVID2<7:0>)	DEV<2:0> (DEVID2<7:5>)	Device
0100 1110	001	PIC18F46J11
0100 1110	000	PIC18F45J11
0100 1101	111	PIC18F44J11
0100 1101	110	PIC18F26J11
0100 1101	101	PIC18F25J11
0100 1101	100	PIC18F24J11
0100 1110	111	PIC18LF46J11
0100 1110	110	PIC18LF45J11
0100 1110	101	PIC18LF44J11
0100 1110	100	PIC18LF26J11
0100 1110	011	PIC18LF25J11
0100 1110	010	PIC18LF24J11

## 25.2 Watchdog Timer (WDT)

PIC18F46J11 family devices have both a conventional WDT circuit and a dedicated, Deep Sleep capable Watchdog Timer. When enabled, the conventional WDT operates in normal Run, Idle and Sleep modes. This data sheet section describes the conventional WDT circuit.

The dedicated, Deep Sleep capable WDT can only be enabled in Deep Sleep mode. This timer is described in **Section 3.6.4 “Deep Sleep Watchdog Timer (DSWDT)”**.

The conventional 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 multiplexer, controlled by the WDTPS bits in Configuration Register 2H. Available periods range from about 4 ms to 135 seconds (2.25 minutes depending on voltage, temperature and WDT postscaler). 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.

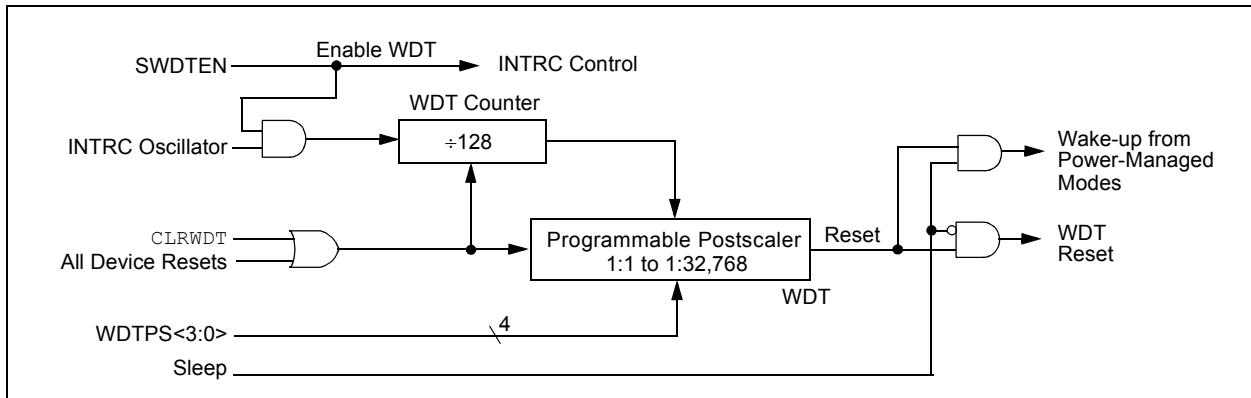
**2:** When a `CLRWDT` instruction is executed, the postscaler count will be cleared.

### 25.2.1 CONTROL REGISTER

The WDTCON register (Register 25-11) is a readable and writable register. The `SWDTEN` bit enables or disables WDT operation. This allows software to override the `WDTEN` Configuration bit and enable the WDT only if it has been disabled by the Configuration bit.

`LVDSTAT` is a read-only status bit that is continuously updated and provides information about the current level of `VDDCORE`. This bit is only valid when the on-chip voltage regulator is enabled.

**FIGURE 25-1: WDT BLOCK DIAGRAM**



# PIC18F46J11 FAMILY

## REGISTER 25-11: WDTCON: WATCHDOG TIMER CONTROL REGISTER (ACCESS FC0h)

R/W-1	R-x	R-x	U-0	R/W-0	R/W-0	R/W-0	R/W-0
REGSLP <sup>(2)</sup>	LVDSTAT <sup>(2)</sup>	ULPLVL	—	DS <sup>(2)</sup>	ULPEN	ULPSINK	SWDTEN <sup>(1)</sup>
bit 7							bit 0

### Legend:

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

- bit 7            **REGSLP:** Voltage Regulator Low-Power Operation Enable bit<sup>(2)</sup>  
 1 = On-chip regulator enters low-power operation when device enters Sleep mode  
 0 = On-chip regulator is active even in Sleep mode
- bit 6            **LVDSTAT:** Low-Voltage Detect Status bit<sup>(2)</sup>  
 1 = VDDCORE > 2.45V nominal  
 0 = VDDCORE < 2.45V nominal
- bit 5            **ULPLVL:** Ultra Low-Power Wake-up Output bit (not valid unless ULPEN = 1)  
 1 = Voltage on RA0 > ~0.5V  
 0 = Voltage on RA0 < ~0.5V
- bit 4            **Unimplemented:** Read as '0'
- bit 3            **DS:** Deep Sleep Wake-up Status bit (used in conjunction with RCON, POR and BOR bits to determine Reset source)<sup>(2)</sup>  
 1 = If the last exit from POR was caused by a normal wake-up from Deep Sleep  
 0 = If the last exit from POR was a result of hard cycling VDD, or if the Deep Sleep BOR was enabled and detected, a (VDD < VDSBOR) and (VDD < VPOR) condition
- bit 2            **ULPEN:** Ultra Low-Power Wake-up Module Enable bit  
 1 = Ultra Low-Power Wake-up module is enabled; ULPLVL bit indicates comparator output  
 0 = Ultra Low-Power Wake-up module is disabled
- bit 1            **ULPSINK:** Ultra Low-Power Wake-up Current Sink Enable bit  
 1 = Ultra Low-Power Wake-up current sink is enabled (if ULPEN = 1)  
 0 = Ultra Low-Power Wake-up current sink is disabled
- bit 0            **SWDTEN:** Software Controlled Watchdog Timer Enable bit<sup>(1)</sup>  
 1 = Watchdog Timer is on  
 0 = Watchdog Timer is off

- Note 1:** This bit has no effect if the Configuration bit, WDTEN, is enabled.  
**Note 2:** Not available on devices where the on-chip voltage regulator is disabled ("LF" devices).

**TABLE 25-3: 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	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	64
WDTCON	REGSLP	LVDSTAT	ULPLVL	—	DS	ULPEN	ULPSINK	SWDTEN	64

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the Watchdog Timer.



## 25.3 On-Chip Voltage Regulator

**Note 1:** The on-chip voltage regulator is only available in parts designated with an “F”, such as PIC18F25J11. The on-chip regulator is disabled on devices with “LF” in their part number.

**2:** The VDDCORE/VCAP pin must never be left floating. On “F” devices, it must be connected to a capacitor, of size CEFC, to ground. On “LF” devices, VDDCORE/VCAP must be connected to a power supply source between 2.0V and 2.7V.

The digital core logic of the PIC18F46J11 family devices is designed on an advanced manufacturing process, which requires 2.0V to 2.7V. The digital core logic obtains power from the VDDCORE/VCAP power supply pin.

However, in many applications it may be inconvenient to run the I/O pins at the same core logic voltage, as it would restrict the ability of the device to interface with other, higher voltage devices, such as those run at a nominal 3.3V. Therefore, all PIC18F46J11 family devices implement a dual power supply rail topology. The core logic obtains power from the VDDCORE/VCAP pin, while the general purpose I/O pins obtain power from the VDD pin of the microcontroller, which may be supplied with a voltage between 2.15V to 3.6V (“F” devices) or 2.0V to 3.6V (“LF” devices).

This dual supply topology allows the microcontroller to interface with standard 3.3V logic devices, while running the core logic at a lower voltage of nominally 2.5V.

In order to make the microcontroller more convenient to use, an integrated 2.5V low dropout, low quiescent current linear regulator has been integrated on the die inside PIC18F46J11 family devices. This regulator is designed specifically to supply the core logic of the device. It allows PIC18F46J11 family devices to effectively run from a single power supply rail, without the need for external regulators.

The on-chip voltage regulator is always enabled on “F” devices. The VDDCORE/VCAP pin serves simultaneously as the regulator output pin and the core logic supply power input pin. A capacitor should be connected to the VDDCORE/VCAP pin to ground and is necessary for regulator stability. For example connections for PIC18F and PIC18LF devices, see Figure 25-2.

On “LF” devices, the on-chip regulator is always disabled. This allows the device to save a small amount of quiescent current consumption, which may be advantageous in some types of applications, such as those which will entirely be running at a nominal 2.5V. On PIC18LF46J11 family devices, the VDDCORE/VCAP pin still serves as the core logic power supply input pin, and therefore, must be connected to a 2.0V to 2.7V supply rail at the application circuit board level. On these devices, the I/O pins may still optionally be supplied with a voltage between 2.0V to 3.6V, provided that VDD is always greater than, or equal to, VDDCORE/VCAP. For example connections for PIC18F and PIC18LF devices, see Figure 25-2.

**Note:** In parts designated with an “LF”, such as PIC18LF46J11, VDDCORE must never exceed VDD.

The specifications for core voltage and capacitance are listed in **Section 28.3 “DC Characteristics: PIC18F46J50 Family (Industrial)”**.

### 25.3.1 VOLTAGE REGULATOR TRACKING MODE AND LOW-VOLTAGE DETECTION

When it is enabled, the on-chip regulator provides a constant voltage of 2.5V nominal to the digital core logic. The regulator can provide this level from a VDD of about 2.5V, all the way up to the device’s VDDMAX. It does not have the capability to boost VDD levels below 2.5V. When the VDD supply input voltage drops too low to regulate to 2.5V, the regulator enters Tracking mode. In Tracking mode, the regulator output follows VDD, with a typical voltage drop of 100 mV or less.

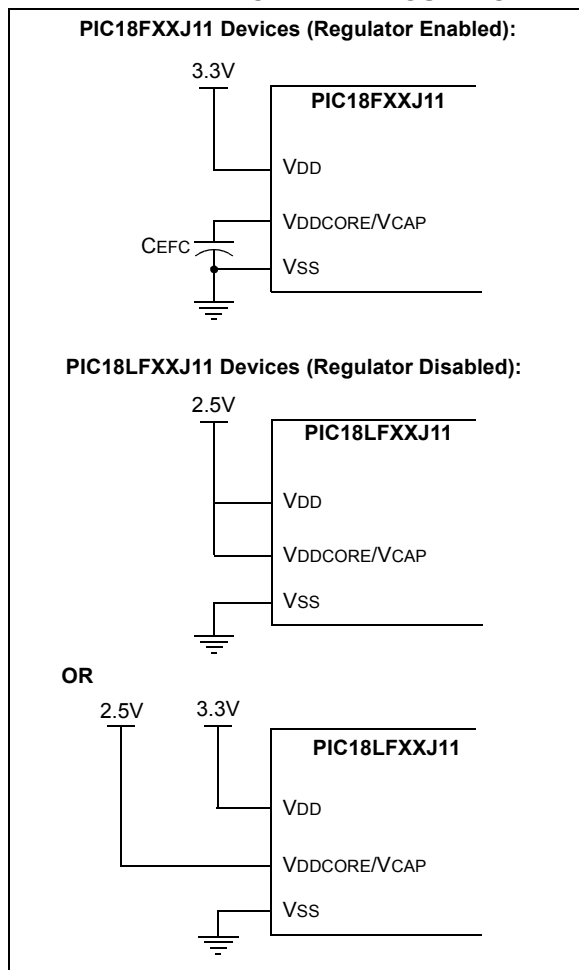
The on-chip regulator includes a simple, Low-Voltage Detect (LVD) circuit. This circuit is separate and independent of the High/Low-Voltage Detect (HLVD) module described in **Section 23.0 “High/Low Voltage Detect (HLVD)”**. The on-chip regulator LVD circuit continuously monitors the VDDCORE voltage level and updates the LVDSTAT bit in the WDTCON register. The LVD detect threshold is set slightly below the normal regulation set point of the on-chip regulator.

Application firmware may optionally poll the LVDSTAT bit to determine when it is safe to run at the maximum rated frequency, so as not to inadvertently violate the voltage versus frequency requirements provided by Figure 28-1.

The VDDCORE monitoring LVD circuit is only active when the on-chip regulator is enabled. On “LF” devices, the Analog-to-Digital Converter and the HLVD module can still be used to provide firmware with VDD and VDDCORE voltage level information.

# PIC18F46J11 FAMILY

**FIGURE 25-2: CONNECTIONS FOR THE ON-CHIP REGULATOR**



## 25.3.2 ON-CHIP REGULATOR AND BOR

When the on-chip regulator is enabled, PIC18F46J11 family devices also have a simple brown-out capability. If the voltage supplied to the regulator is inadequate to maintain a minimum output level; the regulator Reset circuitry will generate a **Brown-out Reset (BOR)**. This event is captured by the  $\overline{\text{BOR}}$  flag bit ( $\text{RCON}<0>$ ).

The operation of the BOR is described in more detail in **Section 4.4 “Brown-out Reset (BOR)”** and **Section 4.4.1 “Detecting BOR”**. The brown-out voltage levels are specific in **Section 28.1 “DC Characteristics: Supply Voltage PIC18F46J11 Family (Industrial)”**.

## 25.3.3 POWER-UP REQUIREMENTS

The on-chip regulator is designed to meet the power-up requirements for the device. If the application does not use the regulator, then strict power-up conditions must be adhered to. While powering up, VDDCORE should not exceed VDD by 0.3 volts.

## 25.3.4 OPERATION IN SLEEP MODE

When enabled, the on-chip regulator always consumes a small incremental amount of current over  $I_{DD}$ . This includes when the device is in Sleep mode, even though the core digital logic does not require much power. To provide additional savings in applications where power resources are critical, the regulator can be configured to automatically enter a lower quiescent draw standby mode whenever the device goes into Sleep mode. This feature is controlled by the REGSLP bit ( $\text{WDTCON}<7>$ , Register 25-11). If this bit is set upon entry into Sleep mode, the regulator will transition into a lower power state. In this state, the regulator still provides a regulated output voltage necessary to maintain SRAM state information, but consumes less quiescent current.

Substantial Sleep mode power savings can be obtained by setting the REGSLP bit, but device wake-up time will increase in order to insure the regulator has enough time to stabilize.

## 25.4 Two-Speed Start-up

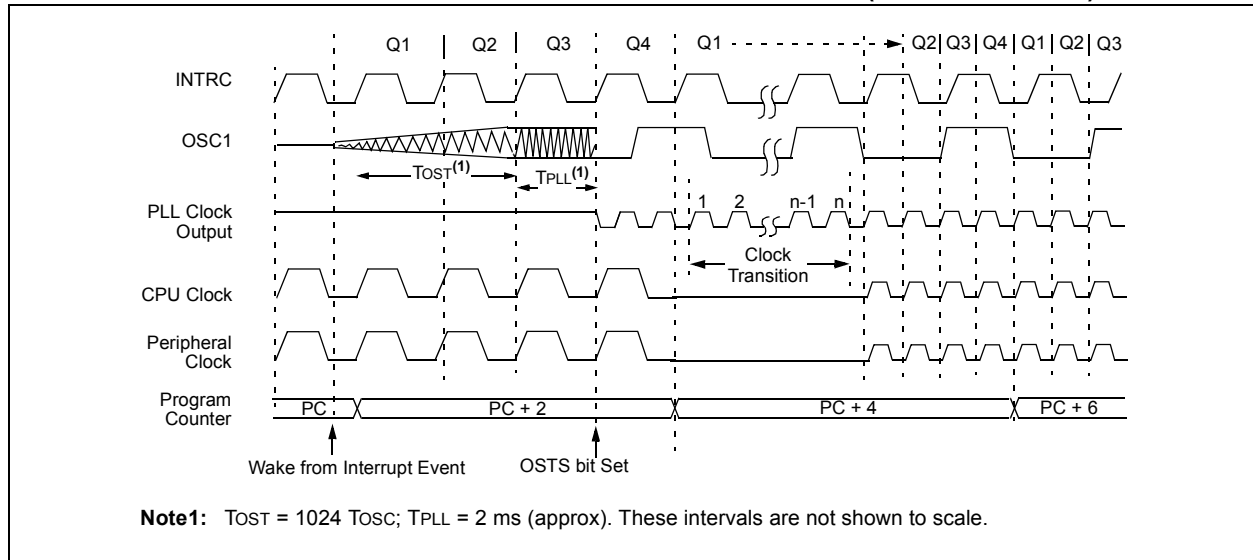
The Two-Speed Start-up feature helps to minimize the latency period, from oscillator start-up to code execution, by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is HS or HSPLL (Crystal-Based) modes. Since the EC and ECPLL modes do not require an Oscillator Start-up Timer (OST) 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 Power-on 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.

**FIGURE 25-3: TIMING TRANSITION FOR TWO-SPEED START-UP (INTRC TO HSPLL)**



### 25.4.1 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 `SCS<1:0>` 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.

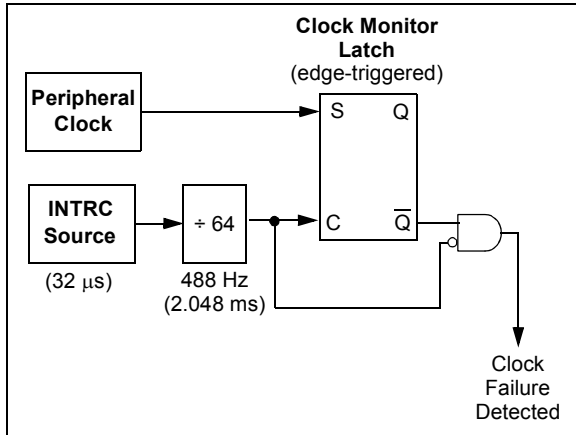
### 25.5 Fail-Safe Clock Monitor

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 25-4) 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. The clock monitor is set on the falling edge of the device clock source but cleared on the rising edge of the sample clock.

# PIC18F46J11 FAMILY

**FIGURE 25-4: FSCM BLOCK DIAGRAM**



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while the clock monitor is still set, and a clock failure has been detected (Figure 25-5), the following results:

- 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 25.4.1 “Special Considerations for Using Two-Speed Start-up”** for more details.

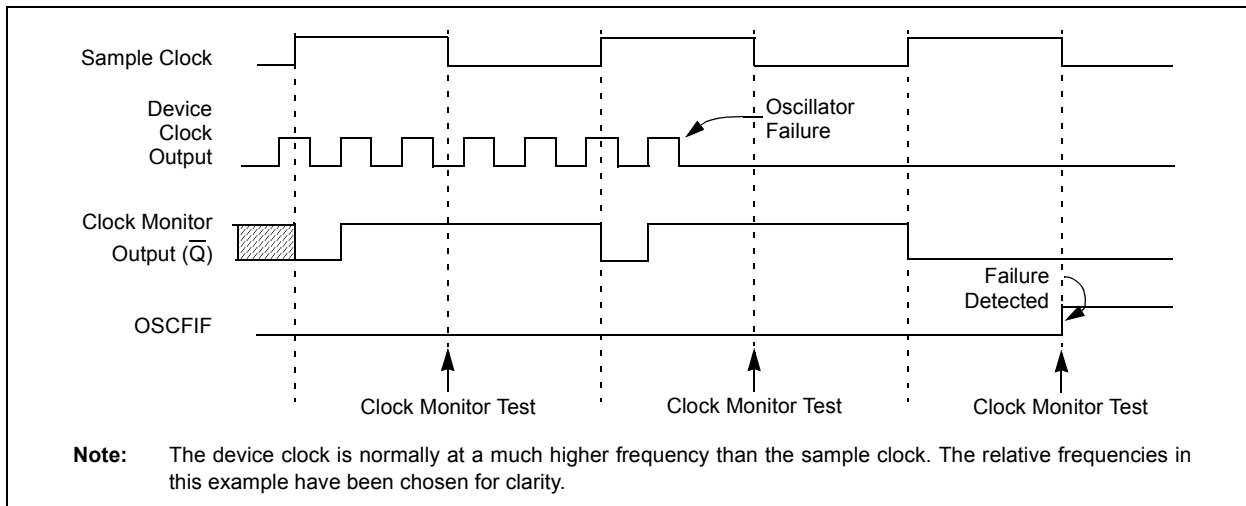
The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

## 25.5.1 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.

**FIGURE 25-5: FSCM TIMING DIAGRAM**



## 25.5.2 EXITING FAIL-SAFE OPERATION

The Fail-Safe Clock Monitor 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 any required start-up delays that are required for the oscillator mode, such as OST or PLL timer). 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 OSTS bit in the OSCCON register becoming set). The FSCM 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.

## 25.5.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. FSCM 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 INTRC multiplexer. 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 INTRC source.

## 25.5.4 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 the EC or INTRC modes, monitoring can begin immediately following these events.

For HS or HSPLL modes, 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 and PLL timers have timed

out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

**Note:** The same logic that prevents false oscillator failure interrupts on POR, or wake-up 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 25.4.1 “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.

## 25.6 Program Verification and Code Protection

For all devices in the PIC18F46J11 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.

### 25.6.1 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. This is seen by the user as a Configuration Mismatch (CM) Reset.

The data for the Configuration registers is derived from the FCW in program memory. When the CP0 bit is set, the source data for device configuration is also protected as a consequence.

# PIC18F46J11 FAMILY

---

## 25.7 In-Circuit Serial Programming (ICSP)

PIC18F46J11 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.

## 25.8 In-Circuit Debugger

When the  $\overline{\text{DEBUG}}$  Configuration bit is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB<sup>®</sup> IDE. When the microcontroller has this feature enabled, some resources are not available for general use.

Table 25-4 lists the resources required by the background debugger.

**TABLE 25-4: DEBUGGER RESOURCES**

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	512 bytes
Data Memory:	10 bytes

## 26.0 INSTRUCTION SET SUMMARY

The PIC18F46J11 family of devices incorporates the standard set of 75 PIC18 core instructions, and an extended set of eight 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.

### 26.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC<sup>®</sup> MCU instruction sets, while maintaining an easy migration from these PIC MCU 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 26-2 lists the **byte-oriented**, **bit-oriented**, **literal** and **control** operations.

Table 26-1 provides 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 '0', the result is placed in the WREG register. If 'd' is '1', 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 **NOE**.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter (PC) is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a **NOE**.

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  $\mu$ 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  $\mu$ s. Two-word branch instructions (if true) would take 3  $\mu$ s.

Figure 26-1 provides the general formats that the instructions can have. All examples use the convention 'nnh' to represent a hexadecimal number.

The instruction set summary, provided in Table 26-2, lists the standard instructions recognized by the Microchip MPASM<sup>™</sup> Assembler.

**Section 26.1.1 "Standard Instruction Set"** provides a description of each instruction.

# PIC18F46J11 FAMILY

**TABLE 26-1: 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: <b>C</b> arry, <b>D</b> igit Carry, <b>Z</b> ero, <b>O</b> verflow, <b>N</b> 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 <sub>s</sub>	12-bit register file address (000h to FFFh). This is the source address
f <sub>d</sub>	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 Used only 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 <sub>s</sub>	7-bit offset value for Indirect Addressing of register files (source)
z <sub>d</sub>	7-bit offset value for Indirect Addressing of register files (destination)
{ }	Optional argument
[text]	Indicates Indexed Addressing
(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 New)



# PIC18F46J11 FAMILY

## EXAMPLE 26-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations	Example Instruction																																								
<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">10</td> <td style="text-align: center;">9</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td style="border: 1px solid black; text-align: center;">d</td> <td style="border: 1px solid black; text-align: center;">a</td> <td colspan="2" style="border: 1px solid black;">f (FILE #)</td> </tr> </table> <p>d = 0 for result destination to be WREG register            d = 1 for result destination to be file register (f)            a = 0 to force Access Bank            a = 1 for BSR to select bank            f = 8-bit file register address</p>	15	10	9	8	7	0	OPCODE		d	a	f (FILE #)		ADDWF MYREG, W, B																												
15	10	9	8	7	0																																				
OPCODE		d	a	f (FILE #)																																					
<p><b>Byte to Byte move operations (2-word)</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">12</td> <td style="text-align: center;">11</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td colspan="2" style="border: 1px solid black;">f (Source FILE #)</td> </tr> </table> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">12</td> <td style="text-align: center;">11</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">1111</td> <td colspan="2" style="border: 1px solid black;">f (Destination FILE #)</td> </tr> </table> <p>f = 12-bit file register address</p>	15	12	11	0	OPCODE		f (Source FILE #)		15	12	11	0	1111		f (Destination FILE #)		MOVFF MYREG1, MYREG2																								
15	12	11	0																																						
OPCODE		f (Source FILE #)																																							
15	12	11	0																																						
1111		f (Destination FILE #)																																							
<p><b>Bit-oriented file register operations</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">12</td> <td style="text-align: center;">11</td> <td style="text-align: center;">9</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td style="border: 1px solid black; text-align: center;">b (BIT #)</td> <td style="border: 1px solid black; text-align: center;">a</td> <td colspan="3" style="border: 1px solid black;">f (FILE #)</td> </tr> </table> <p>b = 3-bit position of bit in file register (f)            a = 0 to force Access Bank            a = 1 for BSR to select bank            f = 8-bit file register address</p>	15	12	11	9	8	7	0	OPCODE		b (BIT #)	a	f (FILE #)			BSF MYREG, bit, B																										
15	12	11	9	8	7	0																																			
OPCODE		b (BIT #)	a	f (FILE #)																																					
<p><b>Literal operations</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td colspan="2" style="border: 1px solid black;">k (literal)</td> </tr> </table> <p>k = 8-bit immediate value</p>	15	8	7	0	OPCODE		k (literal)		MOVLW 7Fh																																
15	8	7	0																																						
OPCODE		k (literal)																																							
<p><b>Control operations</b></p> <p><b>CALL, GOTO and Branch operations</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td colspan="2" style="border: 1px solid black;">n&lt;7:0&gt; (literal)</td> </tr> </table> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">12</td> <td style="text-align: center;">11</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">1111</td> <td colspan="2" style="border: 1px solid black;">n&lt;19:8&gt; (literal)</td> </tr> </table> <p>n = 20-bit immediate value</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td style="border: 1px solid black; text-align: center;">S</td> <td style="border: 1px solid black;">n&lt;7:0&gt; (literal)</td> </tr> </table> <p>S = Fast bit</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">11</td> <td style="text-align: center;">10</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td colspan="2" style="border: 1px solid black;">n&lt;10:0&gt; (literal)</td> </tr> </table> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: right;">15</td> <td style="text-align: center;">8</td> <td style="text-align: center;">7</td> <td style="text-align: right;">0</td> </tr> <tr> <td colspan="2" style="border: 1px solid black;">OPCODE</td> <td colspan="2" style="border: 1px solid black;">n&lt;7:0&gt; (literal)</td> </tr> </table>	15	8	7	0	OPCODE		n<7:0> (literal)		15	12	11	0	1111		n<19:8> (literal)		15	8	7	0	OPCODE		S	n<7:0> (literal)	15	11	10	0	OPCODE		n<10:0> (literal)		15	8	7	0	OPCODE		n<7:0> (literal)		<p>GOTO Label</p> <p>CALL MYFUNC</p> <p>BRA MYFUNC</p> <p>BC MYFUNC</p>
15	8	7	0																																						
OPCODE		n<7:0> (literal)																																							
15	12	11	0																																						
1111		n<19:8> (literal)																																							
15	8	7	0																																						
OPCODE		S	n<7:0> (literal)																																						
15	11	10	0																																						
OPCODE		n<10:0> (literal)																																							
15	8	7	0																																						
OPCODE		n<7:0> (literal)																																							

# PIC18F46J11 FAMILY

**TABLE 26-2: PIC18F46J11 FAMILY INSTRUCTION SET**

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected	Notes	
			MSb			LSb			
<b>BYTE-ORIENTED OPERATIONS</b>									
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 <sub>s</sub> , f <sub>d</sub>	Move f <sub>s</sub> (source) to f <sub>d</sub> (destination)	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 the 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.

# PIC18F46J11 FAMILY

**TABLE 26-2: PIC18F46J11 FAMILY INSTRUCTION SET (CONTINUED)**

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected	Notes	
			MSb	LSb					
<b>BIT-ORIENTED OPERATIONS</b>									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, b, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2
<b>CONTROL OPERATIONS</b>									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	4
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	1 (2)	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	2	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call Subroutine	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	
POP	—	Pop Top of Return Stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software Device Reset	1	0000	0000	1111	1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$ , $\overline{PD}$	

- Note 1:** When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
- 2:** If this instruction is executed on the TMR0 register (and, where applicable,  $d = 1$ ), the prescaler will be cleared if assigned.
- 3:** If the 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.

# PIC18F46J11 FAMILY

**TABLE 26-2: PIC18F46J11 FAMILY INSTRUCTION SET (CONTINUED)**

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
			MSb			LSb		
<b>LITERAL OPERATIONS</b>								
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
<b>DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS</b>								
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 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 the 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.

# PIC18F46J11 FAMILY

## 26.1.1 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:	<table border="1"> <tr> <td>0000</td> <td>1111</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	0000	1111	kkkk	kkkk				
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:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>Decode</td> <td>Read literal 'k'</td> <td>Process Data</td> <td>Write to W</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	Decode	Read literal 'k'	Process Data	Write to W
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:	<table border="1"> <tr> <td>0010</td> <td>01da</td> <td>ffff</td> <td>ffff</td> </tr> </table>	0010	01da	ffff	ffff				
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 <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>Decode</td> <td>Read register 'f'</td> <td>Process Data</td> <td>Write to destination</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	Decode	Read register 'f'	Process Data	Write to destination
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).

# PIC18F46J11 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) → 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:            1

Cycles:           1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:**            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) .AND. k → W

Status Affected: N, Z

Encoding:        

0000	1011	kkkk	kkkk
------	------	------	------

Description:     The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words:            1

Cycles:           1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

**Example:**            ANDLW    05Fh

Before Instruction  
 W = A3h

After Instruction  
 W = 03h

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                        1

Cycles:                        1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:**                    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 → 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)

# PIC18F46J11 FAMILY

<b>BCF</b>	<b>Bit Clear f</b>								
Syntax:	BCF f, b {,a}								
Operands:	$0 \leq f \leq 255$ $0 \leq b \leq 7$ $a \in [0,1]$								
Operation:	$0 \rightarrow f < b >$								
Status Affected:	None								
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">1001</td> <td style="padding: 2px;">bbba</td> <td style="padding: 2px;">ffff</td> <td style="padding: 2px;">ffff</td> </tr> </table>	1001	bbba	ffff	ffff				
1001	bbba	ffff	ffff						
Description:	<p>Bit 'b' in register 'f' is cleared.</p> <p>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).</p> <p>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever <math>f \leq 95</math> (5Fh). See <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.</p>								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1" style="display: inline-table; border-collapse: collapse; text-align: center;"> <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:**            BCF        FLAG\_REG, 7, 0

Before Instruction  
                     FLAG\_REG = C7h  
 After Instruction  
                     FLAG\_REG = 47h

<b>BN</b>	<b>Branch if Negative</b>												
Syntax:	BN n												
Operands:	$-128 \leq n \leq 127$												
Operation:	if Negative bit is '1', $(PC) + 2 + 2n \rightarrow PC$												
Status Affected:	None												
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">1110</td> <td style="padding: 2px;">0110</td> <td style="padding: 2px;">nnnn</td> <td style="padding: 2px;">nnnn</td> </tr> </table>	1110	0110	nnnn	nnnn								
1110	0110	nnnn	nnnn										
Description:	<p>If the Negative bit is '1', then the program will branch.</p> <p>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 <math>PC + 2 + 2n</math>. This instruction is then a two-cycle instruction.</p>												
Words:	1												
Cycles:	1(2)												
Q Cycle Activity:													
If Jump:	<table border="1" style="display: inline-table; border-collapse: collapse; text-align: center;"> <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 literal 'n'</td> <td style="padding: 2px;">Process Data</td> <td style="padding: 2px;">Write to PC</td> </tr> <tr> <td style="padding: 2px;">No operation</td> <td style="padding: 2px;">No operation</td> <td style="padding: 2px;">No operation</td> <td style="padding: 2px;">No operation</td> </tr> </table>	Q1	Q2	Q3	Q4	Decode	Read literal 'n'	Process Data	Write to PC	No operation	No operation	No operation	No operation
Q1	Q2	Q3	Q4										
Decode	Read literal 'n'	Process Data	Write to PC										
No operation	No operation	No operation	No operation										
If No Jump:	<table border="1" style="display: inline-table; border-collapse: collapse; text-align: center;"> <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 literal 'n'</td> <td style="padding: 2px;">Process Data</td> <td style="padding: 2px;">No operation</td> </tr> </table>	Q1	Q2	Q3	Q4	Decode	Read literal 'n'	Process Data	No operation				
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)



# PIC18F46J11 FAMILY

**BNC**                      **Branch if Not Carry**

---

Syntax:                    BNC n

Operands:                -128 ≤ n ≤ 127

Operation:                if Carry bit is '0',  
(PC) + 2 + 2n → PC

Status Affected:        None

Encoding:                

1110	0011	nnnn	nnnn
------	------	------	------

Description:              If the Carry bit is '0', then the program will branch.

                              The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words:                    1

Cycles:                   1(2)

Q Cycle Activity:  
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:                    HERE            BNC    Jump

Before Instruction  
PC                    =    address (HERE)

After Instruction  
If Carry            =    0;  
PC                    =    address (Jump)

If Carry            =    1;  
PC                    =    address (HERE + 2)

**BNN**                      **Branch if Not Negative**

---

Syntax:                    BNN n

Operands:                -128 ≤ n ≤ 127

Operation:                if Negative bit is '0',  
(PC) + 2 + 2n → PC

Status Affected:        None

Encoding:                

1110	0111	nnnn	nnnn
------	------	------	------

Description:              If the Negative bit is '0', then the program will branch.

                              The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words:                    1

Cycles:                   1(2)

Q Cycle Activity:  
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:                    HERE            BNN    Jump

Before Instruction  
PC                    =    address (HERE)

After Instruction  
If Negative        =    0;  
PC                    =    address (Jump)

If Negative        =    1;  
PC                    =    address (HERE + 2)

# PIC18F46J11 FAMILY

## BNOV Branch if Not Overflow

Syntax: BNOV n

Operands:  $-128 \leq n \leq 127$

Operation: if Overflow bit is '0',  
(PC) + 2 + 2n → PC

Status Affected: None

Encoding: 

1110	0101	nnnn	nnnn
------	------	------	------

Description: If the Overflow bit is '0', then the program will branch.  
  
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:  
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:**                    HERE            BNOV Jump

Before Instruction  
PC = address (HERE)

After Instruction  
If Overflow = 0;  
PC = address (Jump)  
If Overflow = 1;  
PC = address (HERE + 2)

## BNZ Branch if Not Zero

Syntax: BNZ n

Operands:  $-128 \leq n \leq 127$

Operation: if Zero bit is '0',  
(PC) + 2 + 2n → PC

Status Affected: None

Encoding: 

1110	0001	nnnn	nnnn
------	------	------	------

Description: If the Zero bit is '0', then the program will branch.  
  
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:  
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:**                    HERE            BNZ Jump

Before Instruction  
PC = address (HERE)

After Instruction  
If Zero = 0;  
PC = address (Jump)  
If Zero = 1;  
PC = address (HERE + 2)

# PIC18F46J11 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:  $1 \rightarrow f < b >$

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

**Example:**

```

BSF    FLAG_REG, 7, 1
Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah
    
```

# PIC18F46J11 FAMILY

## BTFSC Bit Test File, Skip if Clear

**Syntax:** BTFSC f, b {,a}

**Operands:**  $0 \leq f \leq 255$   
 $0 \leq b \leq 7$   
 $a \in [0,1]$

**Operation:** skip if (f<b>) = 0

**Status Affected:** None

**Encoding:**

1011	bbba	ffff	ffff
------	------	------	------

**Description:** If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 26.2.3 "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    BTFSC    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)

## BTFSS Bit Test File, Skip if Set

**Syntax:** BTFSS f, b {,a}

**Operands:**  $0 \leq f \leq 255$   
 $0 \leq b < 7$   
 $a \in [0,1]$

**Operation:** skip if (f<b>) = 1

**Status Affected:** None

**Encoding:**

1010	bbba	ffff	ffff
------	------	------	------

**Description:** If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 26.2.3 "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    BTFSS    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)

# PIC18F46J11 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:	$\overline{f\langle b \rangle} \rightarrow f\langle b \rangle$								
Status Affected:	None								
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">0111</td> <td style="padding: 2px;">bbba</td> <td style="padding: 2px;">ffff</td> <td style="padding: 2px;">ffff</td> </tr> </table>	0111	bbba	ffff	ffff				
0111	bbba	ffff	ffff						
Description:	<p>Bit 'b' in data memory location 'f' is inverted.</p> <p>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).</p> <p>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever <math>f \leq 95</math> (5Fh). See <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.</p>								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1" style="display: inline-table; border-collapse: collapse; text-align: center;"> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> <tr> <td>Decode</td> <td>Read register 'f'</td> <td>Process Data</td> <td>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:** 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:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">1110</td> <td style="padding: 2px;">0100</td> <td style="padding: 2px;">nnnn</td> <td style="padding: 2px;">nnnn</td> </tr> </table>	1110	0100	nnnn	nnnn
1110	0100	nnnn	nnnn		
Description:	<p>If the Overflow bit is '1', then the program will branch.</p> <p>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 <math>PC + 2 + 2n</math>. This instruction is then a two-cycle instruction.</p>				
Words:	1				
Cycles:	1(2)				
Q Cycle Activity:	If Jump:				

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:** HERE BOV Jump

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Overflow = 1;  
 PC = address (Jump)  
 If Overflow = 0;  
 PC = address (HERE + 2)

# PIC18F46J11 FAMILY

## BZ Branch if Zero

Syntax: BZ n

Operands:  $-128 \leq n \leq 127$

Operation: if Zero bit is '1',  
(PC) + 2 + 2n → PC

Status Affected: None

Encoding: 

1110	0000	nnnn	nnnn
------	------	------	------

Description: If the Zero bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:  
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:**            HERE            BZ    Jump

Before Instruction  
PC = address (HERE)

After Instruction  
If Zero = 1;  
PC = address (Jump)

If Zero = 0;  
PC = address (HERE + 2)

## CALL Subroutine Call

Syntax: CALL k {,s}

Operands:  $0 \leq k \leq 1048575$   
s ∈ [0,1]

Operation: (PC) + 4 → TOS,  
k → PC<20:1>;  
if s = 1,  
(W) → WS,  
(STATUS) → STATUSs,  
(BSR) → BSRs

Status Affected: None

Encoding: 

1110	110s	k <sub>7</sub> kkk	kkkk <sub>0</sub>
1111	k <sub>19</sub> kkk	kkkk	kkkk <sub>8</sub>

1st word (k<7:0>)  
2nd word(k<19:8>)

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	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

# PIC18F46J11 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;"><tr><td>0110</td><td>101a</td><td>ffff</td><td>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 <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1" style="display: inline-table;"><thead><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr></thead><tbody><tr><td>Decode</td><td>Read register 'f'</td><td>Process Data</td><td>Write register 'f'</td></tr></tbody></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;"><tr><td>0000</td><td>0000</td><td>0000</td><td>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;"><thead><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr></thead><tbody><tr><td>Decode</td><td>No operation</td><td>Process Data</td><td>No operation</td></tr></tbody></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

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                        1

Cycles:                        1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example:                    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 26.2.3 "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        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)



# PIC18F46J11 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 Addressing mode whenever  $f \leq 95$  (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1(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)
```

# PIC18F46J11 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  $[W<7:4> > 9]$  or  $[C = 1]$  then,  
 $(W<7:4>) + 6 \rightarrow W<7:4>$ ;  
 $C = 1$ ;  
 else,  
 $(W<7:4>) \rightarrow W<7:4>$

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                        1

Cycles:                       1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:**                       $DECF \ CNT, \ 1, \ 0$

Before Instruction

CNT                    =    01h

Z                        =    0

After Instruction

CNT                    =    00h

Z                        =    1



# PIC18F46J11 FAMILY

## GOTO Unconditional Branch

Syntax: GOTO k

Operands:  $0 \leq k \leq 1048575$

Operation:  $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1110	1111	$k_7$ kkk	kkkk $k_0$
1111	$k_{19}$ kkk	kkkk	kkkk $k_8$

1st word ( $k<7:0>$ )  
2nd word ( $k<19:8>$ )

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: INCF CNT, 1, 0

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

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

# PIC18F46J11 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 26.2.3 "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  
                              :        :  
                              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 26.2.3 "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    :

Before Instruction  
PC = Address (HERE)

After Instruction  
REG = REG + 1  
If REG  $\neq$  0;  
PC = Address (NZERO)  
If REG = 0;  
PC = Address (ZERO)

# PIC18F46J11 FAMILY

<b>IORLW</b>	<b>Inclusive OR Literal with W</b>								
Syntax:	IORLW k								
Operands:	$0 \leq k \leq 255$								
Operation:	(W) .OR. k $\rightarrow$ W								
Status Affected:	N, Z								
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">1001</td> <td style="padding: 2px;">kkkk</td> <td style="padding: 2px;">kkkk</td> </tr> </table>	0000	1001	kkkk	kkkk				
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:	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse; text-align: center;"> <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 literal 'k'</td> <td style="padding: 2px;">Process Data</td> <td style="padding: 2px;">Write to W</td> </tr> </table>	Q1	Q2	Q3	Q4	Decode	Read literal 'k'	Process Data	Write to W
Q1	Q2	Q3	Q4						
Decode	Read literal 'k'	Process Data	Write to W						

**Example:**            IORLW    35h

Before Instruction  
W        =    9Ah

After Instruction  
W        =    BFh

<b>IORWF</b>	<b>Inclusive OR W with f</b>								
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:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">0001</td> <td style="padding: 2px;">00da</td> <td style="padding: 2px;">ffff</td> <td style="padding: 2px;">ffff</td> </tr> </table>	0001	00da	ffff	ffff				
0001	00da	ffff	ffff						
Description:	<p>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).</p> <p>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).</p> <p>If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever <math>f \leq 95</math> (5Fh). See <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.</p>								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse; text-align: center;"> <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 to destination</td> </tr> </table>	Q1	Q2	Q3	Q4	Decode	Read register 'f'	Process Data	Write to destination
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

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                        1

Cycles:                        1

Q Cycle Activity:

	Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W	

**Example:**                      MOVF REG, 0, 0

Before Instruction  
 REG                        = 22h  
 W                            = FFh

After Instruction  
 REG                        = 22h  
 W                            = 22h

# PIC18F46J11 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:

1100	ffff	ffff	ffff <sub>s</sub>
1111	ffff	ffff	ffff <sub>d</sub>

Description: The contents of source register ' $f_s$ ' are moved to destination register ' $f_d$ '. Location of source ' $f_s$ ' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination ' $f_d$ ' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation 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



# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: MOVWF REG, 0

Before Instruction

W = 4Fh  
REG = FFh

After Instruction

W = 4Fh  
REG = 4Fh

# PIC18F46J11 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 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write 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

# PIC18F46J11 FAMILY

NEGF	Negate f								
Syntax:	NEGF f {,a}								
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]								
Operation:	$\bar{f} + 1 \rightarrow f$								
Status Affected:	N, OV, C, DC, Z								
Encoding:	<table border="1"> <tr> <td>0110</td> <td>110a</td> <td>ffff</td> <td>ffff</td> </tr> </table>	0110	110a	ffff	ffff				
0110	110a	ffff	ffff						
Description:	<p>Location 'f' is negated using two's complement. The result is placed in the data memory location 'f'.</p> <p>If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).</p> <p>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 <b>Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.</p>								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>Decode</td> <td>Read register 'f'</td> <td>Process Data</td> <td>Write register 'f'</td> </tr> </tbody> </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:**           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:	<table border="1"> <tr> <td>0000</td> <td>0000</td> <td>0000</td> <td>0000</td> </tr> <tr> <td>1111</td> <td>xxxx</td> <td>xxxx</td> <td>xxxx</td> </tr> </table>	0000	0000	0000	0000	1111	xxxx	xxxx	xxxx
0000	0000	0000	0000						
1111	xxxx	xxxx	xxxx						
Description:	No operation.								
Words:	1								
Cycles:	1								
Q Cycle Activity:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>Decode</td> <td>No operation</td> <td>No operation</td> <td>No operation</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	Decode	No operation	No operation	No operation
Q1	Q2	Q3	Q4						
Decode	No operation	No operation	No operation						

**Example:**

None.

# PIC18F46J11 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      NEW
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    NEW

Before Instruction
TOS     = 345Ah
PC      = 0124h

After Instruction
PC      = 0126h
TOS     = 0126h
Stack (1 level down) = 345Ah
    
```

# PIC18F46J11 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:	<table border="1" style="display: inline-table;"><tr><td>1101</td><td>1nnn</td><td>nnnn</td><td>nnnn</td></tr></table>	1101	1nnn	nnnn	nnnn
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:	<table border="1" style="display: inline-table;"><tr><td>0000</td><td>0000</td><td>1111</td><td>1111</td></tr></table>	0000	0000	1111	1111				
0000	0000	1111	1111						
Description:	This instruction provides a way to execute a MCLR Reset in software.								
Words:	1								
Cycles:	1								
Q Cycle Activity:									
	<table border="1" style="display: inline-table;"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>Decode</td><td>Start reset</td><td>No operation</td><td>No operation</td></tr></table>	Q1	Q2	Q3	Q4	Decode	Start reset	No operation	No operation
Q1	Q2	Q3	Q4						
Decode	Start reset	No operation	No operation						

**Example:**           RESET

After Instruction

Registers = Reset Value

Flags\* = Reset Value

# PIC18F46J11 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.

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

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

# PIC18F46J11 FAMILY

**RETURN**                      **Return from Subroutine**

---

Syntax:                      RETURN {s}

Operands:                    s ∈ [0,1]

Operation:                    (TOS) → PC;  
                                   if s = 1,  
                                   (WS) → W,  
                                   (STATUS) → 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, STATUS and BSR 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                        1

Cycles:                        1

Q Cycle Activity:

	Q1	Q2	Q3	Q4
Decode		Read register 'f'	Process Data	Write to destination

**Example:**                      RLCF                      REG, 0, 0

Before Instruction

REG	=	1110 0110
C	=	0

After Instruction

REG	=	1110 0110
W	=	1100 1100
C	=	1

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1  
 Cycles: 1  
 Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** 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

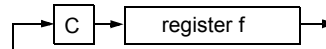
Encoding: 

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1  
 Cycles: 1  
 Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** RRCF REG, 0, 0

Before Instruction  
 REG = 1110 0110  
 C = 0  
 After Instruction  
 REG = 1110 0110  
 W = 0111 0011  
 C = 0



# PIC18F46J11 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:         $(f < n >) \rightarrow \text{dest} < n - 1 >$ ,  
 $(f < 0 >) \rightarrow \text{dest} < 7 >$

Status Affected:    N, Z

Encoding:        

0100	00da	ffff	ffff
------	------	------	------

Description:      The contents of register 'f' are rotated one bit to the right. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).

                    If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

                    If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever  $f \leq 95$  (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:            1

Cycles:           1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write 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:         $\text{FFh} \rightarrow f$

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:            1

Cycles:           1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

**Example:**            SETF REG, 1

Before Instruction  
REG = 5Ah  
After Instruction  
REG = FFh

# PIC18F46J11 FAMILY

**SLEEP**      **Enter Sleep Mode**

---

Syntax:            SLEEP

Operands:        None

Operation:        00h → WDT,  
                     0 → WDT postscaler,  
                     1 →  $\overline{TO}$ ,  
                     0 →  $\overline{PD}$

Status Affected:  $\overline{TO}$ ,  $\overline{PD}$

Encoding:        

0000	0000	0000	0011
------	------	------	------

Description:     The Power-Down status bit ( $\overline{PD}$ ) is cleared. The Time-out status bit ( $\overline{TO}$ ) is set. The 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:            1

Cycles:           1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 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

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words:                    1

Cycles:                    1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example 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

# PIC18F46J11 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1  
**Cycles:** 1  
**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1  
**Cycles:** 1  
**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** SWAPF REG, 1, 0

**Before Instruction**  
 REG = 53h

**After Instruction**  
 REG = 35h

# PIC18F46J11 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

# PIC18F46J11 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 5.0 “Memory Organization”** 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

# PIC18F46J11 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 26.2.3 "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

# PIC18F46J11 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) → 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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: XORWF REG, 1, 0

Before Instruction

REG = AFh  
W = B5h

After Instruction

REG = 1Ah  
W = B5h



## 26.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, the PIC18F46J11 family of devices also provides 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 for many of the standard PIC18 instructions.

The additional features of the extended instruction set are enabled by default on unprogrammed devices. Users must properly set or clear the XINST Configuration bit during programming to enable or disable these features.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers (FSR), or use them for Indexed Addressing. Two of the instructions, `ADDFSR` and `SUBFSR`, each have an additional special instantiation for using FSR2. These versions (`ADDLNLK` and `SUBLNLK`) 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 26-3. Detailed descriptions are provided in **Section 26.2.2 “Extended Instruction Set”**. The opcode field descriptions in Table 26-1 (page 408) 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.

### 26.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the FSRs 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. The 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 26.2.3.1 “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 26-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET**

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected
			MSb		LSb		
<code>ADDFSR</code> f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
<code>ADDLNLK</code> k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
<code>CALLW</code>	Call Subroutine using WREG	2	0000	0000	0001	0100	None
<code>MOVSF</code> z <sub>s</sub> , f <sub>d</sub>	Move z <sub>s</sub> (source) to 1st word f <sub>d</sub> (destination) 2nd word	2	1110	1011	0zzz	zzzz	None
<code>MOVSS</code> z <sub>s</sub> , z <sub>d</sub>	Move z <sub>s</sub> (source) to 1st word z <sub>d</sub> (destination) 2nd word	2	1110	1011	1zzz	zzzz	None
<code>PUSHL</code> k	Store Literal at FSR2, Decrement FSR2	1	1110	1010	kkkk	kkkk	None
<code>SUBFSR</code> f, k	Subtract Literal from FSR	1	1110	1001	ffkk	kkkk	None
<code>SUBLNLK</code> k	Subtract Literal from FSR2 and Return	2	1110	1001	11kk	kkkk	None

# PIC18F46J11 FAMILY

## 26.2.2 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

### ADDLNLK Add Literal to FSR2 and Return

Syntax: ADDLNLK 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:** ADDLNLK 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 format then becomes: {label} instruction argument(s).

# PIC18F46J11 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<sub>s</sub>], f<sub>d</sub>

**Operands:** 0 ≤ z<sub>s</sub> ≤ 127  
0 ≤ f<sub>d</sub> ≤ 4095

**Operation:** ((FSR2) + z<sub>s</sub>) → f<sub>d</sub>

**Status Affected:** None

**Encoding:**

1110	1011	0zzz	zzzz <sub>s</sub>
1111	ffff	ffff	ffff <sub>d</sub>

**Description:** The contents of the source register are moved to destination register 'f<sub>d</sub>'. The actual address of the source register is determined by adding the 7-bit literal offset 'z<sub>s</sub>' in the first word, to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f<sub>d</sub>' 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

# PIC18F46J11 FAMILY

## MOVSS Move Indexed to Indexed

Syntax: MOVSS [z<sub>s</sub>], [z<sub>d</sub>]

Operands: 0 ≤ z<sub>s</sub> ≤ 127  
0 ≤ z<sub>d</sub> ≤ 127

Operation: ((FSR2) + z<sub>s</sub>) → ((FSR2) + z<sub>d</sub>)

Status Affected: None

Encoding:

1110	1011	1zzz	zzzz <sub>s</sub>
1111	xxxx	xzzz	zzzz <sub>d</sub>

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<sub>s</sub>' or 'z<sub>d</sub>', 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:

1110	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

# PIC18F46J11 FAMILY

## **SUBFSR**      **Subtract Literal from FSR**

Syntax:            SUBFSR f, k  
 Operands:         $0 \leq k \leq 63$   
                      $f \in [0, 1, 2]$   
 Operation:       FSRf – k → 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 → FSR2,  
                     (TOS) → 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)

# PIC18F46J11 FAMILY

---

## 26.2.3 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 (**Section 5.6.1 “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 26.2.3.1 “Extended Instruction Syntax with Standard PIC18 Commands”**).

Although the Indexed Literal Offset 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 mode are provided on the following page to show how execution is affected. The operand conditions provided in the examples are applicable to all instructions of these types.

### 26.2.3.1 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 the 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.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

## 26.2.4 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 PIC18F46J11 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.

# PIC18F46J11 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:                 $(W) + ((FSR2) + k) \rightarrow \text{dest}$

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
-------------------	---	-----

# PIC18F46J11 FAMILY

---

## 26.2.5 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 for the PIC18F46J11 family. 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. 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.



## 27.0 DEVELOPMENT SUPPORT

The PIC<sup>®</sup> microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB<sup>®</sup> IDE Software
- Assemblers/Compilers/Linkers
  - MPASM<sup>™</sup> Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK<sup>™</sup> Object Linker/  
MPLIB<sup>™</sup> Object Librarian
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB REAL ICE<sup>™</sup> In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PICSTART<sup>®</sup> Plus Development Programmer
  - MPLAB PM3 Device Programmer
  - PICKit<sup>™</sup> 2 Development Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

## 27.1 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<sup>®</sup> operating system-based application that contains:

- A single graphical interface to all 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
- Visual device initializer for easy register initialization
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as HI-TECH Software C Compilers and IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PIC MCU 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 increased flexibility and power.

# PIC18F46J11 FAMILY

---

## 27.2 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for all PIC 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

## 27.3 MPLAB C18 and MPLAB C30 C Compilers

The MPLAB C18 and MPLAB C30 Code Development Systems are complete ANSI C compilers for Microchip's PIC18 and PIC24 families of microcontrollers and the dsPIC30 and dsPIC33 family of digital signal controllers. 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.

## 27.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

## 27.5 MPLAB ASM30 Assembler, Linker and Librarian

MPLAB ASM30 Assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 C 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

## 27.6 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC® DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB C18 and MPLAB C30 C Compilers, and the MPASM and MPLAB ASM30 Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

## 27.7 MPLAB ICE 2000 High-Performance In-Circuit Emulator

The MPLAB ICE 2000 In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PIC 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 architecture of the MPLAB ICE 2000 In-Circuit Emulator allows expansion to support new PIC 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.

## 27.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The MPLAB REAL ICE probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with the popular MPLAB ICD 2 system (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

MPLAB REAL ICE is field upgradeable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added, such as software breakpoints and assembly code trace. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, real-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

## 27.9 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 PIC MCUs and can be used to develop for these and other PIC MCUs and dsPIC DSCs. 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, and 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 PIC devices.

## 27.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer 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 PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an SD/MMC card for file storage and secure data applications.

# PIC18F46J11 FAMILY

---

## 27.11 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 PIC devices in DIP packages 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.

## 27.12 PICkit 2 Development Programmer

The PICkit™ 2 Development Programmer is a low-cost programmer and selected Flash device debugger with an easy-to-use interface for programming many of Microchip's baseline, mid-range and PIC18F families of Flash memory microcontrollers. The PICkit 2 Starter Kit includes a prototyping development board, twelve sequential lessons, software and HI-TECH's PICC™ Lite C compiler, and is designed to help get up to speed quickly using PIC® microcontrollers. The kit provides everything needed to program, evaluate and develop applications using Microchip's powerful, mid-range Flash memory family of microcontrollers.

## 27.13 Demonstration, Development and Evaluation Boards

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Check the Microchip web page ([www.microchip.com](http://www.microchip.com)) for the complete list of demonstration, development and evaluation kits.

# PIC18F46J11 FAMILY

## 28.0 ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings<sup>(†)</sup>

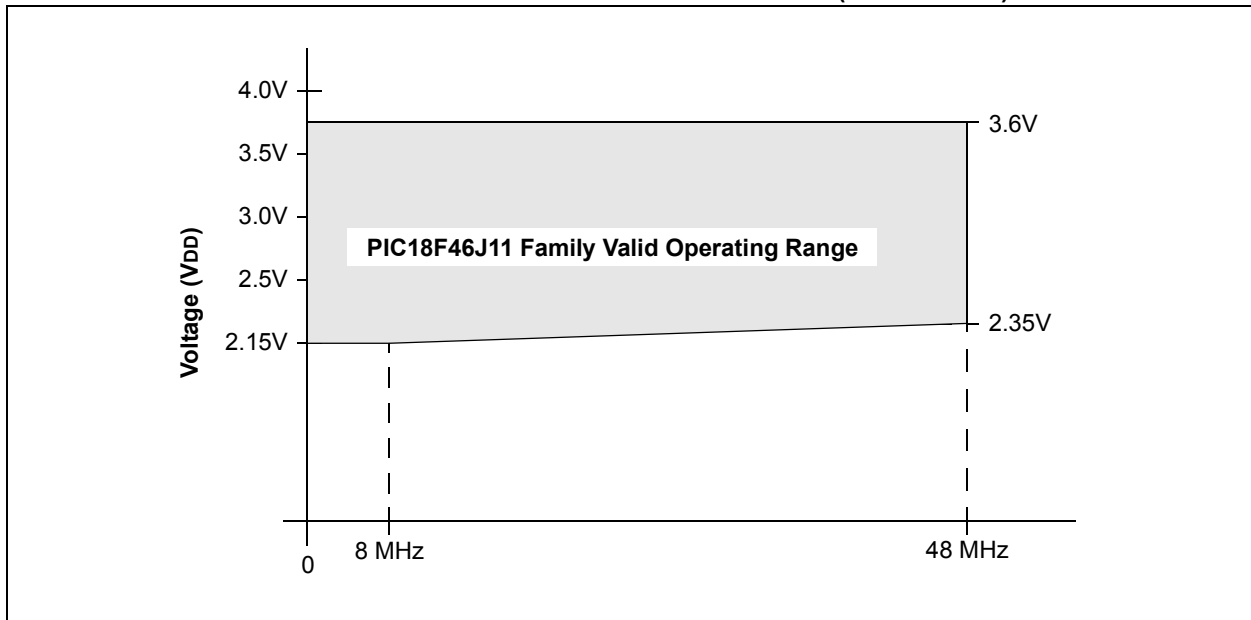
Ambient temperature under bias .....	-40°C to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on any digital only I/O pin or $\overline{\text{MCLR}}$ with respect to $V_{SS}$ (except $V_{DD}$ ) .....	-0.3V to 6.0V
Voltage on any combined digital and analog pin with respect to $V_{SS}$ (except $V_{DD}$ ).....	-0.3V to ( $V_{DD} + 0.3V$ )
Voltage on $V_{DDCORE}$ with respect to $V_{SS}$ .....	-0.3V to 2.75V
Voltage on $V_{DD}$ with respect to $V_{SS}$ .....	-0.3V to 4.0V
Total power dissipation ( <b>Note 1</b> ) .....	1.0W
Maximum current out of $V_{SS}$ pin .....	300 mA
Maximum current into $V_{DD}$ pin .....	250 mA
Maximum output current sunk by any PORTB, PORTC and RA6 I/O pin.....	25 mA
Maximum output current sunk by any PORTA (except RA6), PORTD and PORTE I/O pin.....	4 mA
Maximum output current sourced by any PORTB, PORTC and RA6 I/O pin .....	25 mA
Maximum output current sourced by any PORTA (except RA6), PORTD and PORTE I/O pin .....	4 mA
Maximum current sunk by all ports .....	200 mA
Maximum current sourced by all ports .....	200 mA

**Note 1:** Power dissipation is calculated as follows:

$$P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$$

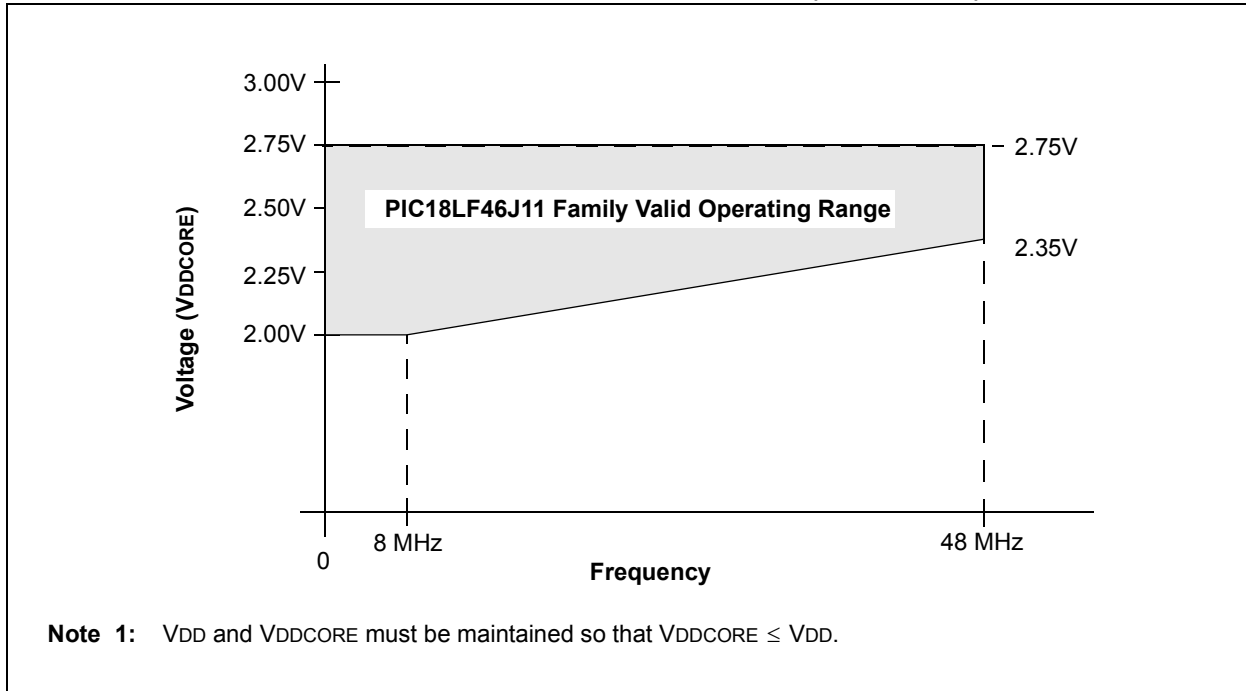
† **NOTICE:** Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

**FIGURE 28-1: PIC18F46J11 FAMILY  $V_{DD}$  FREQUENCY GRAPH (INDUSTRIAL)**



# PIC18F46J11 FAMILY

FIGURE 28-2: PIC18LF46J11 VDDCORE FREQUENCY GRAPH (INDUSTRIAL)<sup>(1)</sup>



# PIC18F46J11 FAMILY

## 28.1 DC Characteristics: Supply Voltage PIC18F46J11 Family (Industrial)

PIC18F46J11 Family			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	<b>Supply Voltage</b>	2.15 2.0	—	3.6 3.6	V V	PIC18F4XJ11, PIC18F2XJ11 PIC18LF4XJ11, PIC18LF2XJ11
D001B	VDDCORE	<b>External Supply for Microcontroller Core</b>	2.0	—	2.75	V	PIC18LF4XJ11, PIC18LF2XJ11
D001C	AVDD	<b>Analog Supply Voltage</b>	$V_{DD} - 0.3$	—	$V_{DD} + 0.3$	V	
D001D	AVSS	<b>Analog Ground Potential</b>	$V_{SS} - 0.3$	—	$V_{SS} + 0.3$	V	
D002	VDR	<b>RAM Data Retention Voltage<sup>(1)</sup></b>	1.5	—	—	V	
D003	VPOR	<b>V<sub>DD</sub> Start Voltage to Ensure Internal Power-on Reset Signal</b>	—	—	0.7	V	See Section 4.3 “Power-on Reset (POR)” for details
D004	SVDD	<b>V<sub>DD</sub> Rise Rate to Ensure Internal Power-on Reset Signal</b>	0.05	—	—	V/ms	See Section 4.3 “Power-on Reset (POR)” for details
D005	VBOR	<b>VDDCORE Brown-out Reset Voltage</b>	—	2.0	—	V	PIC18F4XJ11, PIC18F2XJ11 only (not used on “LF” devices)
D006	VDSBOR	<b>VDD Brown-out Reset Voltage</b>	—	1.8	—	V	DSBOREN = 1 on “LF” device, or “F” device In Deep Sleep

**Note 1:** This is the limit to which VDDCORE can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated)					
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18FXXJ11 Family		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		
<b>Power-Down Current (I<sub>PD</sub>)<sup>(1)</sup> – Sleep mode</b>							
	PIC18LFXXJ11	0.011	1.4	μA	-40°C	V <sub>DD</sub> = 2.0V, V <sub>DDCORE</sub> = 2.0V	<b>Sleep mode, REGSLP = 1</b>
		0.054	1.4	μA	+25°C		
		0.51	6	μA	+60°C		
		2.0	10.2	μA	+85°C		
	PIC18LFXXJ11	0.029	1.5	μA	-40°C	V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V	
		0.11	1.5	μA	+25°C		
		0.63	8	μA	+60°C		
	PIC18FXXJ11	2.5	5	μA	-40°C	V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 μF Capacitor	
		3.1	5	μA	+25°C		
		3.9	8	μA	+60°C		
		5.6	16	μA	+85°C		
	PIC18FXXJ11	4.1	7	μA	-40°C	V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 μF Capacitor	
3.3		7	μA	+25°C			
4.1		10	μA	+60°C			
6.0		19	μA	+85°C			
<b>Power-Down Current (I<sub>PD</sub>)<sup>(1)</sup> – Deep Sleep mode</b>							
	PIC18FXXJ11	1	25	nA	-40°C	V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 μF Capacitor	<b>Deep Sleep mode</b>
		13	100	nA	+25°C		
		108	250	nA	+60°C		
		428	1000	nA	+85°C		
	PIC18FXXJ11	3	50	nA	-40°C	V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 μF Capacitor	
		28	150	nA	+25°C		
		170	389	nA	+60°C		
		588	2000	nA	+85°C		

- 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<sub>DD</sub> or V<sub>SS</sub> 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. All features that add delta current are disabled (WDT, etc.). The test conditions for all I<sub>DD</sub> measurements in active operation mode are:  
 OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V<sub>DD</sub>/V<sub>SS</sub>;  
 MCLR = V<sub>DD</sub>; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with 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.



# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18FXXJ11 Family		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		
<b>Supply Current (<math>I_{DD}</math>)<sup>(2)</sup></b>							
	PIC18LFXXJ11	5.2	14.2	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.0\text{V}$ , $V_{DDCORE} = 2.0\text{V}$  $V_{DD} = 2.5\text{V}$ , $V_{DDCORE} = 2.5\text{V}$  $V_{DD} = 2.15\text{V}$ , $V_{DDCORE} = 10\ \mu\text{F}$  $V_{DD} = 3.3\text{V}$ , $V_{DDCORE} = 10\ \mu\text{F}$	
		6.2	14.2	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		8.6	19.0	$\mu\text{A}$	$+85^{\circ}\text{C}$		
	PIC18LFXXJ11	7.6	16.5	$\mu\text{A}$	$-40^{\circ}\text{C}$		
		8.5	16.5	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		11.3	22.4	$\mu\text{A}$	$+85^{\circ}\text{C}$		
	PIC18FXXJ11	37	77	$\mu\text{A}$	$-40^{\circ}\text{C}$		<b>FOSC = 31 kHz, RC_RUN mode, Internal RC Oscillator, INTSRC = 0</b>
		48	77	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		60	93	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18FXXJ11	52	84	$\mu\text{A}$	$-40^{\circ}\text{C}$			
	61	84	$\mu\text{A}$	$+25^{\circ}\text{C}$			
	70	108	$\mu\text{A}$	$+85^{\circ}\text{C}$			
	PIC18LFXXJ11	1.1	1.5	$\text{mA}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.0\text{V}$ , $V_{DDCORE} = 2.0$  $V_{DD} = 2.5\text{V}$ , $V_{DDCORE} = 2.5\text{V}$  $V_{DD} = 2.15\text{V}$ , $V_{DDCORE} = 10\ \mu\text{F}$  $V_{DD} = 3.3\text{V}$ , $V_{DDCORE} = 10\ \mu\text{F}$	
		1.1	1.5	$\text{mA}$	$+25^{\circ}\text{C}$		
		1.2	1.6	$\text{mA}$	$+85^{\circ}\text{C}$		
	PIC18LFXXJ11	1.5	1.7	$\text{mA}$	$-40^{\circ}\text{C}$		
		1.6	1.7	$\text{mA}$	$+25^{\circ}\text{C}$		
		1.6	1.9	$\text{mA}$	$+85^{\circ}\text{C}$		
	PIC18FXXJ11	1.3	2.6	$\text{mA}$	$-40^{\circ}\text{C}$		<b>FOSC = 4 MHz, RC_RUN mode, Internal RC Oscillator</b>
		1.4	2.6	$\text{mA}$	$+25^{\circ}\text{C}$		
		1.4	2.8	$\text{mA}$	$+85^{\circ}\text{C}$		
PIC18FXXJ11	1.6	2.9	$\text{mA}$	$-40^{\circ}\text{C}$			
	1.6	2.9	$\text{mA}$	$+25^{\circ}\text{C}$			
	1.6	3.0	$\text{mA}$	$+85^{\circ}\text{C}$			

- 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. All features that add delta current are disabled (WDT, etc.). 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  $V_{DD}/V_{SS}$ ;  
 $MCLR = V_{DD}$ ; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated)					
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18FXXJ11 Family		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		
<b>Supply Current (<math>I_{DD}</math>)<sup>(2)</sup></b>							
	PIC18LFXXJ11	1.9	3.6	mA	$-40^{\circ}\text{C}$	VDD = 2.0V, VDDCORE = 2.0V	
		2.0	3.8	mA	$+25^{\circ}\text{C}$		
		2.0	3.8	mA	$+85^{\circ}\text{C}$		
	PIC18LFXXJ11	2.8	4.8	mA	$-40^{\circ}\text{C}$		VDD = 2.5V, VDDCORE = 2.5V
		2.8	4.8	mA	$+25^{\circ}\text{C}$		
		2.8	4.9	mA	$+85^{\circ}\text{C}$		
	PIC18FXXJ11	2.3	4.2	mA	$-40^{\circ}\text{C}$		VDD = 2.15V, VDDCORE = 10 $\mu\text{F}$
		2.3	4.2	mA	$+25^{\circ}\text{C}$		
		2.4	4.5	mA	$+85^{\circ}\text{C}$		
PIC18FXXJ11	2.8	5.1	mA	$-40^{\circ}\text{C}$	VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$		
	2.8	5.1	mA	$+25^{\circ}\text{C}$			
	2.8	5.4	mA	$+85^{\circ}\text{C}$			
	PIC18LFXXJ11	1.9	9.4	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V, VDDCORE = 2.0V	
		2.3	9.4	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		4.5	17.2	$\mu\text{A}$	$+85^{\circ}\text{C}$		
	PIC18LFXXJ11	2.4	10.5	$\mu\text{A}$	$-40^{\circ}\text{C}$		VDD = 2.5V, VDDCORE = 2.5V
		2.8	10.5	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		5.4	19.5	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18FXXJ11	33.3	75	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.15V, VDDCORE = 10 $\mu\text{F}$		
	43.8	75	$\mu\text{A}$	$+25^{\circ}\text{C}$			
	55.3	92	$\mu\text{A}$	$+85^{\circ}\text{C}$			
PIC18FXXJ11	36.1	82	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$		
	44.5	82	$\mu\text{A}$	$+25^{\circ}\text{C}$			
	56.3	105	$\mu\text{A}$	$+85^{\circ}\text{C}$			

**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. All features that add delta current are disabled (WDT, etc.). 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/VSS;  
MCLR = VDD; WDT disabled unless otherwise specified.

- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated)				
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial				
PIC18FXXJ11 Family		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	
<b>Supply Current (I<sub>DD</sub>)<sup>(2)</sup></b>						
	PIC18LFXXJ11	0.531	0.980	mA	-40°C	V <sub>DD</sub> = 2.0V, V <sub>DDCORE</sub> = 2.0V  V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V  V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 μF  V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 μF  V <sub>DD</sub> = 2.0V, V <sub>DDCORE</sub> = 2.0V  V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V  V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 μF  V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 μF  V <sub>DD</sub> = 2.0V, V <sub>DDCORE</sub> = 2.0V  V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V  V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 μF  V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 μF
		0.571	0.980	mA	+25°C	
		0.608	1.12	mA	+85°C	
	PIC18LFXXJ11	0.625	1.14	mA	-40°C	
		0.681	1.14	mA	+25°C	
		0.725	1.25	mA	+85°C	
	PIC18FXXJ11	0.613	1.21	mA	-40°C	
		0.680	1.21	mA	+25°C	
		0.730	1.30	mA	+85°C	
PIC18FXXJ11	0.673	1.27	mA	-40°C		
	0.728	1.27	mA	+25°C		
	0.779	1.45	mA	+85°C		
	PIC18LFXXJ11	0.750	1.4	mA	-40°C	
		0.797	1.5	mA	+25°C	
		0.839	1.6	mA	+85°C	
	PIC18LFXXJ11	0.91	2.4	mA	-40°C	
		0.96	2.4	mA	+25°C	
		1.01	2.5	mA	+85°C	
	PIC18FXXJ11	0.87	2.1	mA	-40°C	
		0.93	2.1	mA	+25°C	
		0.98	2.3	mA	+85°C	
PIC18FXXJ11	0.95	2.6	mA	-40°C		
	1.01	2.6	mA	+25°C		
	1.06	2.7	mA	+85°C		

- 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<sub>DD</sub> or V<sub>SS</sub> 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. All features that add delta current are disabled (WDT, etc.). The test conditions for all I<sub>DD</sub> measurements in active operation mode are:  
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V<sub>DD</sub>/V<sub>SS</sub>;  
MCLR = V<sub>DD</sub>; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with 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.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated)				
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial				
PIC18FXXJ11 Family		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	
<b>Supply Current (<math>I_{DD}</math>)<sup>(2)</sup></b>						
	PIC18LFXXJ11	0.879	1.25	mA	$-40^{\circ}\text{C}$	VDD = 2.0V, VDDCORE = 2.0V
		0.881	1.25	mA	$+25^{\circ}\text{C}$	
		0.891	1.36	mA	$+85^{\circ}\text{C}$	
	PIC18LFXXJ11	1.35	1.70	mA	$-40^{\circ}\text{C}$	VDD = 2.0V, VDDCORE = 2.0V
		1.30	1.70	mA	$+25^{\circ}\text{C}$	
		1.27	1.82	mA	$+85^{\circ}\text{C}$	
	PIC18FXXJ11	1.09	1.60	mA	$-40^{\circ}\text{C}$	VDD = 2.15V, VDDCORE = 10 $\mu\text{F}$
		1.09	1.60	mA	$+25^{\circ}\text{C}$	
		1.11	1.70	mA	$+85^{\circ}\text{C}$	
	PIC18FXXJ11	1.36	1.95	mA	$-40^{\circ}\text{C}$	VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$
		1.36	1.89	mA	$+25^{\circ}\text{C}$	
		1.41	1.92	mA	$+85^{\circ}\text{C}$	
	PIC18LFXXJ11	10.9	14.8	mA	$-40^{\circ}\text{C}$	VDD = 2.5V, VDDCORE = 2.5V
		10.6	14.8	mA	$+25^{\circ}\text{C}$	
		10.6	15.2	mA	$+85^{\circ}\text{C}$	
	PIC18FXXJ11	12.9	23.2	mA	$-40^{\circ}\text{C}$	VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$
		12.8	22.7	mA	$+25^{\circ}\text{C}$	
		12.7	22.7	mA	$+85^{\circ}\text{C}$	

- 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. All features that add delta current are disabled (WDT, etc.). The test conditions for all  $I_{DD}$  measurements in active operation mode are:  
 $\overline{\text{OSC1}}$  = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD/VSS;  
 $\overline{\text{MCLR}}$  = VDD; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial				
PIC18FXXJ11 Family		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	
<b>Supply Current (<math>I_{DD}</math>)<sup>(2)</sup></b>						
	PIC18LFXXJ11	0.285	0.700	mA	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 2.0V, V <sub>DDCORE</sub> = 2.0V  V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V  V <sub>DD</sub> = 2.15V, V <sub>DDCORE</sub> = 10 $\mu\text{F}$  V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 $\mu\text{F}$  V <sub>DD</sub> = 2.5V, V <sub>DDCORE</sub> = 2.5V  V <sub>DD</sub> = 3.3V, V <sub>DDCORE</sub> = 10 $\mu\text{F}$
		0.300	0.700	mA	$+25^{\circ}\text{C}$	
		0.336	0.750	mA	$+85^{\circ}\text{C}$	
	PIC18LFXXJ11	0.372	1.00	mA	$-40^{\circ}\text{C}$	
		0.397	1.00	mA	$+25^{\circ}\text{C}$	
		0.495	1.10	mA	$+85^{\circ}\text{C}$	
	PIC18FXXJ11	0.357	0.850	mA	$-40^{\circ}\text{C}$	
		0.383	0.850	mA	$+25^{\circ}\text{C}$	
		0.407	0.900	mA	$+85^{\circ}\text{C}$	
PIC18FXXJ11	0.449	1.30	mA	$-40^{\circ}\text{C}$		
	0.488	1.20	mA	$+25^{\circ}\text{C}$		
	0.554	1.20	mA	$+85^{\circ}\text{C}$		
PIC18LFXXJ11	4.5	6.5	mA	$-40^{\circ}\text{C}$	F <sub>OSC</sub> = 4 MHz, <b>PRI_IDLE</b> mode, EC Oscillator        F <sub>OSC</sub> = 48 MHz <b>PRI_IDLE</b> mode, EC oscillator	
	4.5	6.5	mA	$+25^{\circ}\text{C}$		
	4.6	6.5	mA	$+85^{\circ}\text{C}$		
PIC18FXXJ11	4.9	12.4	mA	$-40^{\circ}\text{C}$		
	5.0	11.5	mA	$+25^{\circ}\text{C}$		
	5.1	11.5	mA	$+85^{\circ}\text{C}$		

- 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<sub>DD</sub> or V<sub>SS</sub> 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. All features that add delta current are disabled (WDT, etc.). The test conditions for all I<sub>DD</sub> measurements in active operation mode are:  
 $\overline{\text{OSC1}}$  = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V<sub>DD</sub>/V<sub>SS</sub>;  
 $\overline{\text{MCLR}}$  = V<sub>DD</sub>; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated)						
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial						
PIC18FXXJ11 Family		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			
	PIC18LFXXJ11	5.2	6.5	mA	$-40^{\circ}\text{C}$	VDD = 2.5V, VDDCORE = 2.5V	FOSC = 16 MHz <b>(PRI_RUN mode,</b> 4 MHz Internal Oscillator with PLL	
		5.1	6.4	mA	$+25^{\circ}\text{C}$			
		5.1	6.4	mA	$+85^{\circ}\text{C}$			
	PIC18FXXJ11	5.3	7.5	mA	$-40^{\circ}\text{C}$			VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$
		5.2	7.4	mA	$+25^{\circ}\text{C}$			
		5.2	7.4	mA	$+85^{\circ}\text{C}$			
	PIC18LFXXJ11	9.3	12.0	mA	$-40^{\circ}\text{C}$	VDD = 2.5V, VDDCORE = 2.5V	FOSC = 32 MHz, <b>PRI_RUN mode,</b> 8 MHz Internal Oscillator with PLL	
		9.2	11.8	mA	$+25^{\circ}\text{C}$			
		9.0	11.8	mA	$+85^{\circ}\text{C}$			
	PIC18FXXJ11	9.7	17.5	mA	$-40^{\circ}\text{C}$			VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$
		9.6	17.2	mA	$+25^{\circ}\text{C}$			
		9.6	17.2	mA	$+85^{\circ}\text{C}$			
	PIC18LFXXJ11	12.4	13.5	mA	$-40^{\circ}\text{C}$	VDD = 2.5V, VDDCORE = 2.5V	FOSC = 48 MHz, <b>PRI_RUN mode,</b> 12 MHz External Oscillator with PLL	
		12.2	13.5	mA	$+25^{\circ}\text{C}$			
		12.1	13.9	mA	$+85^{\circ}\text{C}$			
	PIC18FXXJ11	14.3	24.1	mA	$-40^{\circ}\text{C}$			VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$
		14.2	23.0	mA	$+25^{\circ}\text{C}$			
		14.2	23.0	mA	$+85^{\circ}\text{C}$			

- 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. All features that add delta current are disabled (WDT, etc.). 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/VSS;  
 MCLR = VDD; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial						
PIC18FXXJ11 Family		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			
	PIC18LFXXJ11	12.5	45	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{\text{DD}} = 2.5\text{V}$ , $V_{\text{DDCORE}} = 2.5\text{V}$  $V_{\text{DD}} = 2.15\text{V}$ , $V_{\text{DDCORE}} = 10\ \mu\text{F}$  $V_{\text{DD}} = 3.3\text{V}$ , $V_{\text{DDCORE}} = 10\ \mu\text{F}$  $V_{\text{DD}} = 2.5\text{V}$ , $V_{\text{DDCORE}} = 2.5\text{V}$  $V_{\text{DD}} = 2.15\text{V}$ , $V_{\text{DDCORE}} = 10\ \mu\text{F}$  $V_{\text{DD}} = 3.3\text{V}$ , $V_{\text{DDCORE}} = 10\ \mu\text{F}$		
		11.7	45	$\mu\text{A}$	$+25^{\circ}\text{C}$			
		5.2	61	$\mu\text{A}$	$+85^{\circ}\text{C}$			
	PIC18FXXJ11	40.2	95	$\mu\text{A}$	$-40^{\circ}\text{C}$		$F_{\text{OSC}} = 32\ \text{kHz}^{(3)}$ <b>SEC_RUN</b> mode, $\text{LPT1OSC} = 0$	
		50.2	95	$\mu\text{A}$	$+25^{\circ}\text{C}$			
		61.9	105	$\mu\text{A}$	$+85^{\circ}\text{C}$			
	PIC18LFXXJ11	44.4	110	$\mu\text{A}$	$-40^{\circ}\text{C}$			$F_{\text{OSC}} = 32\ \text{kHz}^{(3)}$ <b>SEC_IDLE</b> mode, $\text{LPT1OSC} = 0$
		53.1	110	$\mu\text{A}$	$+25^{\circ}\text{C}$			
		55.8	150	$\mu\text{A}$	$+85^{\circ}\text{C}$			
PIC18FXXJ11	4.5	31	$\mu\text{A}$	$-40^{\circ}\text{C}$	$F_{\text{OSC}} = 32\ \text{kHz}^{(3)}$ <b>SEC_IDLE</b> mode, $\text{LPT1OSC} = 0$			
	3.8	31	$\mu\text{A}$	$+25^{\circ}\text{C}$				
	4.1	50	$\mu\text{A}$	$+85^{\circ}\text{C}$				
PIC18FXXJ11	34.7	87	$\mu\text{A}$	$-40^{\circ}\text{C}$		$F_{\text{OSC}} = 32\ \text{kHz}^{(3)}$ <b>SEC_IDLE</b> mode, $\text{LPT1OSC} = 0$		
	44.6	89	$\mu\text{A}$	$+25^{\circ}\text{C}$				
	56.5	97	$\mu\text{A}$	$+85^{\circ}\text{C}$				
PIC18LFXXJ11	37.3	100	$\mu\text{A}$	$-40^{\circ}\text{C}$			$V_{\text{DD}} = 3.3\text{V}$ , $V_{\text{DDCORE}} = 10\ \mu\text{F}$	
	45.7	100	$\mu\text{A}$	$+25^{\circ}\text{C}$				
	54.6	140	$\mu\text{A}$	$+85^{\circ}\text{C}$				

- 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. All features that add delta current are disabled (WDT, etc.). The test conditions for all IDD measurements in active operation mode are:  
 $\text{OSC1} = \text{external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to } V_{\text{DD}}/V_{\text{SS}};$   
 $\text{MCLR} = V_{\text{DD}};$  WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial							
PIC18FXXJ11 Family		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 ( $\Delta I_{WDT}$ )	Module Differential Currents ( $\Delta I_{WDT}$ , $\Delta I_{OSCB}$ , $\Delta I_{AD}$ )  Watchdog Timer	0.86	8	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.5\text{V}$ , $V_{DDCORE} = 2.5\text{V}$	PIC18LFXXJ11		
		0.97	8	$\mu\text{A}$	$+25^{\circ}\text{C}$				
		0.98	10.4	$\mu\text{A}$	$+85^{\circ}\text{C}$				
		PIC18FXXJ11	0.71	7	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.15\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
			0.82	7	$\mu\text{A}$	$+25^{\circ}\text{C}$			
			0.65	10	$\mu\text{A}$	$+85^{\circ}\text{C}$			
			1.54	12.1	$\mu\text{A}$	$-40^{\circ}\text{C}$			
		PIC18FXXJ11	1.33	12.1	$\mu\text{A}$	$+25^{\circ}\text{C}$	$V_{DD} = 3.3\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
			1.16	13.6	$\mu\text{A}$	$+85^{\circ}\text{C}$			
			D022B ( $\Delta I_{HLVD}$ )	High/Low-Voltage Detect	3.9	8		$\mu\text{A}$	$-40^{\circ}\text{C}$
4.7	8				$\mu\text{A}$	$+25^{\circ}\text{C}$			
5.4	9	$\mu\text{A}$			$+85^{\circ}\text{C}$				
PIC18FXXJ11	2.7	6			$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.15\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
	3.2	6			$\mu\text{A}$	$+25^{\circ}\text{C}$			
	3.6	8			$\mu\text{A}$	$+85^{\circ}\text{C}$			
	3.5	9			$\mu\text{A}$	$-40^{\circ}\text{C}$			
PIC18FXXJ11	4.1	9			$\mu\text{A}$	$+25^{\circ}\text{C}$	$V_{DD} = 3.3\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
	4.5	12	$\mu\text{A}$	$+85^{\circ}\text{C}$					
	D025 ( $\Delta I_{OSCB}$ )	Real-Time Clock/Calendar with Low-Power Timer1 Oscillator	0.67	4.0	$\mu\text{A}$	$-40^{\circ}\text{C}$		$V_{DD} = 2.15\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$	PIC18FXXJ11 32.768 kHz, T1OSCEN = 1, LPT1OSC = 0
			0.83	4.5	$\mu\text{A}$	$+25^{\circ}\text{C}$			
0.95			4.5	$\mu\text{A}$	$+60^{\circ}\text{C}$				
1.10			4.5	$\mu\text{A}$	$+85^{\circ}\text{C}$				
PIC18FXXJ11			0.75	4.5	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 2.5\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
			0.92	5.0	$\mu\text{A}$	$+25^{\circ}\text{C}$			
			1.04	5.0	$\mu\text{A}$	$+60^{\circ}\text{C}$			
			1.21	5.0	$\mu\text{A}$	$+85^{\circ}\text{C}$			
PIC18FXXJ11			0.94	6.5	$\mu\text{A}$	$-40^{\circ}\text{C}$	$V_{DD} = 3.3\text{V}$ , $V_{DDCORE} = 10 \mu\text{F}$		
			1.11	6.5	$\mu\text{A}$	$+25^{\circ}\text{C}$			
			1.24	8.0	$\mu\text{A}$	$+60^{\circ}\text{C}$			
			1.43	8.0	$\mu\text{A}$	$+85^{\circ}\text{C}$			

- 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.).
- Note 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. All features that add delta current are disabled (WDT, etc.). 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  $V_{DD}/V_{SS}$ ;  
 $MCLR = V_{DD}$ ; WDT disabled unless otherwise specified.
- Note 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.



# PIC18F46J11 FAMILY

## 28.2 DC Characteristics: Power-Down and Supply Current PIC18F46J50 Family (Industrial) (Continued)

PIC18LFXXJ11 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18FXXJ11 Family		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		
D026 ( $\Delta I_{AD}$ )	A/D Converter	3.00	10	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.5V, VDDCORE = 2.5V	PIC18LFXXJ11 A/D on, not converting
		3.00	10	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		3.00	10	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		3.00	10	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.15V, VDDCORE = 10 $\mu\text{F}$	PIC18FXXJ11 A/D on, not converting
		3.00	10	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		3.00	10	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		3.20	11	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.3V, VDDCORE = 10 $\mu\text{F}$	
		3.20	11	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		3.20	11	$\mu\text{A}$	$+85^{\circ}\text{C}$		

- 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. All features that add delta current are disabled (WDT, etc.). 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/VSS;  
MCLR = VDD; WDT disabled unless otherwise specified.
- 3:** Low-Power Timer1 with standard, low-cost 32 kHz crystals have an operating temperature range of  $-10^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ . Extended temperature crystals are available at a much higher cost.

# PIC18F46J11 FAMILY

## 28.3 DC Characteristics: PIC18F46J50 Family (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 <sub>IL</sub>	<b>Input Low Voltage</b> All I/O ports: with TTL Buffer with Schmitt Trigger Buffer $\overline{\text{MCLR}}$ OSC1 OSC1 T1CKI	V <sub>SS</sub> — V <sub>SS</sub> V <sub>SS</sub> V <sub>SS</sub> V <sub>SS</sub> V <sub>SS</sub>	0.15 V <sub>DD</sub> 0.8 0.2 V <sub>DD</sub> 0.2 V <sub>DD</sub> 0.3 V <sub>DD</sub> 0.2 V <sub>DD</sub> 0.2 V <sub>DD</sub>	V V V V V V V	3.3V ≤ V <sub>DD</sub> ≤ 3.6V  HS, HSPLL modes EC, ECPLL modes
D040 D040A D041 Dxxx DxxxA Dxxx D042 D043 D043A D044	V <sub>IH</sub>	<b>Input High Voltage</b> I/O Ports with non 5.5V Tolerance: <sup>(4)</sup> with TTL Buffer with Schmitt Trigger Buffer I/O Ports with 5.5V Tolerance: <sup>(4)</sup> with TTL Buffer with Schmitt Trigger Buffer $\overline{\text{MCLR}}$ OSC1 OSC1 T1CKI	0.25 V <sub>DD</sub> + 0.8V 2.0 0.8 V <sub>DD</sub> 0.25 V <sub>DD</sub> + 0.8V 2.0 0.8 V <sub>DD</sub> 0.8 V <sub>DD</sub> 0.7 V <sub>DD</sub> 0.8 V <sub>DD</sub> 1.6	V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub> 5.5 5.5 5.5 5.5 V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub>	V V V V V V V V V V	V <sub>DD</sub> < 3.3V 3.3V ≤ V <sub>DD</sub> ≤ 3.6V  V <sub>DD</sub> < 3.3V 3.3V ≤ V <sub>DD</sub> ≤ 3.6V  HS, HSPLL modes EC, ECPLL modes
D060 D060A D061 D063	I <sub>IL</sub>	<b>Input Leakage Current</b> <sup>(1,2)</sup> I/O Ports with non 5.5V Tolerance: <sup>(4)</sup> I/O Ports with 5.5V Tolerance: <sup>(4)</sup> $\overline{\text{MCLR}}$ OSC1	— — — —	±0.2 ±0.2 ±0.2 ±0.2	μA μA μA μA	V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> , Pin at high-impedance 3.3V ≤ V <sub>PIN</sub> ≤ 5.5V, Pin at high-impedance V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub>
D070	I <sub>PU</sub> I <sub>PURB</sub>	<b>Weak Pull-up Current</b> PORTB, PORTD <sup>(3)</sup> and PORTE <sup>(3)</sup> Weak Pull-up Current	80	400	μA	V <sub>DD</sub> = 3.3V, V <sub>PIN</sub> = V <sub>SS</sub>

**Note 1:** The leakage current on the  $\overline{\text{MCLR}}$  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.

**2:** Negative current is defined as current sourced by the pin.

**3:** Only available in 44-pin devices.

**4:** Refer to Table 9-2 for the pins that have corresponding tolerance limits.

# PIC18F46J11 FAMILY

## 28.3 DC Characteristics: PIC18F46J50 Family (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
D080	VOL	<b>Output Low Voltage</b>				
		I/O Ports: PORTA (Except RA6), PORTD, PORTE PORTB, PORTC, RA6	—	0.4	V	$I_{OL} = 2\text{ mA}$ , $V_{DD} = 3.3\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D090	VOH	<b>Output High Voltage</b>				
		I/O Ports: PORTA (Except RA6), PORTD, PORTE PORTB, PORTC, RA6	2.4	—	V	$I_{OH} = -2$ , $V_{DD} = 3.3\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D101	C <sub>IO</sub>	<b>Capacitive Loading Specs on Output Pins</b> All I/O Pins and OSC2	—	50	pF	To meet the AC Timing Specifications
D102	C <sub>B</sub>	SCLx, SDAx	—	400	pF	I <sup>2</sup> C™ Specification

- Note 1:** The leakage current on the  $\overline{\text{MCLR}}$  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.
- 2:** Negative current is defined as current sourced by the pin.
- 3:** Only available in 44-pin devices.
- 4:** Refer to Table 9-2 for the pins that have corresponding tolerance limits.

# PIC18F46J11 FAMILY

**TABLE 28-1: 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
D100	IULP	<b>Ultra Low-Power Wake-up Current</b>	—	60	—	nA	Net of I/O leakage and current sink at 1.6V on pin, $V_{DD} = 3.3\text{V}$ See Application Note AN879, "Using the Microchip Ultra Low-Power Wake-up Module" (DS00879)
<b>Program Flash Memory</b>							
D130	EP	Cell Endurance	10K	—	—	E/W	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D131	VPR	VDDcore for Read	V <sub>MIN</sub>	—	2.75	V	V <sub>MIN</sub> = Minimum operating voltage
D132B	VPEW	VDDCORE for Self-Timed Erase or Write	2.25	—	2.75	V	
D133A	TIW	Self-Timed Write Cycle Time	—	2.8	—	ms	64 bytes
D133B	TIE	Self-Timed Block Erase Cycle Time	—	33.0	—	ms	
D134	TRETD	Characteristic Retention	20	—	—	Year	Provided no other specifications are violated
D135	IDDP	Supply Current during Programming	—	3	—	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.

**TABLE 28-2: COMPARATOR SPECIFICATIONS**

Operating Conditions: $3.0\text{V} < V_{DD} < 3.6\text{V}$ , $-40^{\circ}\text{C} < T_A < +85^{\circ}\text{C}$ (unless otherwise stated)							
Param No.	Sym	Characteristics	Min	Typ	Max	Units	Comments
D300	VIOFF	Input Offset Voltage	—	±5	±25	mV	
D301	VICM	Input Common Mode Voltage	0	—	V <sub>DD</sub>	V	
	VIRV	Internal Reference Voltage	0.57	0.60	0.63	V	
D302	CMRR	Common Mode Rejection Ratio	55	—	—	dB	
300	TRESP	Response Time <sup>(1)</sup>	—	150	400	ns	
301	TMC2OV	Comparator Mode Change to Output Valid	—	—	10	μs	

**Note 1:** Response time measured with one comparator input at  $V_{DD}/2$ , while the other input transitions from  $V_{SS}$  to  $V_{DD}$ .

**TABLE 28-3: CTMU CURRENT SOURCE SPECIFICATIONS**

DC CHARACTERISTICS			Standard Operating Conditions: 2.0V to 3.6V (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 <sup>(1)</sup>	Max	Units	Conditions
	IOUT1	CTMU Current Source, Base Range	—	550	—	nA	CTMUICON<1:0> = 01
	IOUT2	CTMU Current Source, 10x Range	—	5.5	—	μA	CTMUICON<1:0> = 10
	IOUT3	CTMU Current Source, 100x Range	—	55	—	μA	CTMUICON<1:0> = 11

**Note 1:** Nominal value at center point of current trim range (CTMUICON<7:2> = 000000).

# PIC18F46J11 FAMILY

**TABLE 28-4: VOLTAGE REFERENCE SPECIFICATIONS**

Operating Conditions: 3.0V < VDD < 3.6V, -40°C < TA < +85°C (unless otherwise stated)							
Param No.	Sym	Characteristics	Min	Typ	Max	Units	Comments
D310	VRES	Resolution	VDD/24	—	VDD/32	LSb	
D311	VRAA	Absolute Accuracy	—	—	1/2	LSb	
D312	VRUR	Unit Resistor Value (R)	—	2k	—	Ω	
310	TSET	Settling Time <sup>(1)</sup>	—	—	10	μs	

**Note 1:** Settling time measured while CVRR = 1 and CVR<3:0> bits transition from '0000' to '1111'.

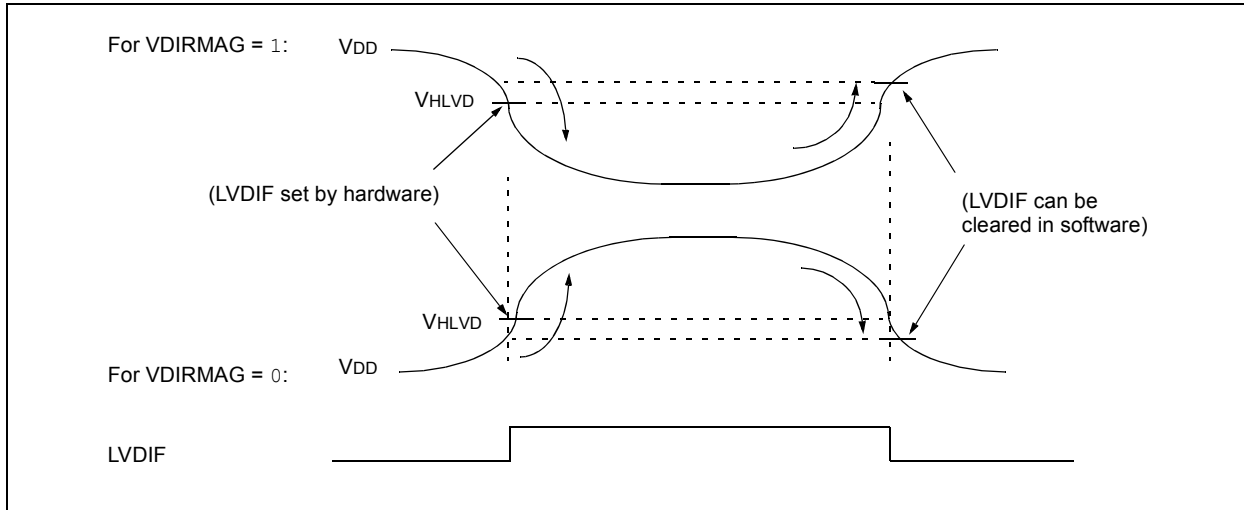
**TABLE 28-5: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS**

Operating Conditions: -40°C < TA < +85°C (unless otherwise stated)							
Param No.	Sym	Characteristics	Min	Typ	Max	Units	Comments
	VRGOUT	Regulator Output Voltage	2.35	2.5	2.7	V	Regulator enabled, VDD = 3.0V
	CEFC	External Filter Capacitor Value <sup>(1)</sup>	5.4	10	18	μF	ESR < 3Ω recommended ESR < 5Ω required

**Note 1:** CEFC applies for PIC18F devices in the family. For PIC18LF devices in the family, there is no specific minimum or maximum capacitance for VDDCORE, although proper supply rail bypassing should still be used.

# PIC18F46J11 FAMILY

**FIGURE 28-3: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS**



**TABLE 28-6: HIGH/LOW-VOLTAGE DETECT 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	Typ	Max	Units	Conditions
D420		HLVD Voltage on VDD Transition High-to-Low	HLVDL<3:0> = 1000	2.33	2.45	2.57	V	
			HLVDL<3:0> = 1001	2.47	2.60	2.73	V	
			HLVDL<3:0> = 1010	2.66	2.80	2.94	V	
			HLVDL<3:0> = 1011	2.76	2.90	3.05	V	
			HLVDL<3:0> = 1100	2.85	3.00	3.15	V	
			HLVDL<3:0> = 1101	2.97	3.13	3.29	V	
			HLVDL<3:0> = 1110	3.23	3.40	3.57	V	
D421	TIRVST	Time for Internal Reference Voltage to become Stable		—	20	—	$\mu\text{s}$	
D422	TLVD	High/Low-Voltage Detect Pulse Width		200	—	—	$\mu\text{s}$	

# PIC18F46J11 FAMILY

## 28.4 AC (Timing) Characteristics

### 28.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

- |             |           |  |
|-------------|-----------|--|
| 1. TppS2ppS | 3. TCC:ST | (I <sup>2</sup> C specifications only) |
| 2. TppS     | 4. Ts     | (I <sup>2</sup> 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	T13CKI
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 <sup>2</sup> C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I<sup>2</sup>C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	Stop condition
STA	Start condition		

# PIC18F46J11 FAMILY

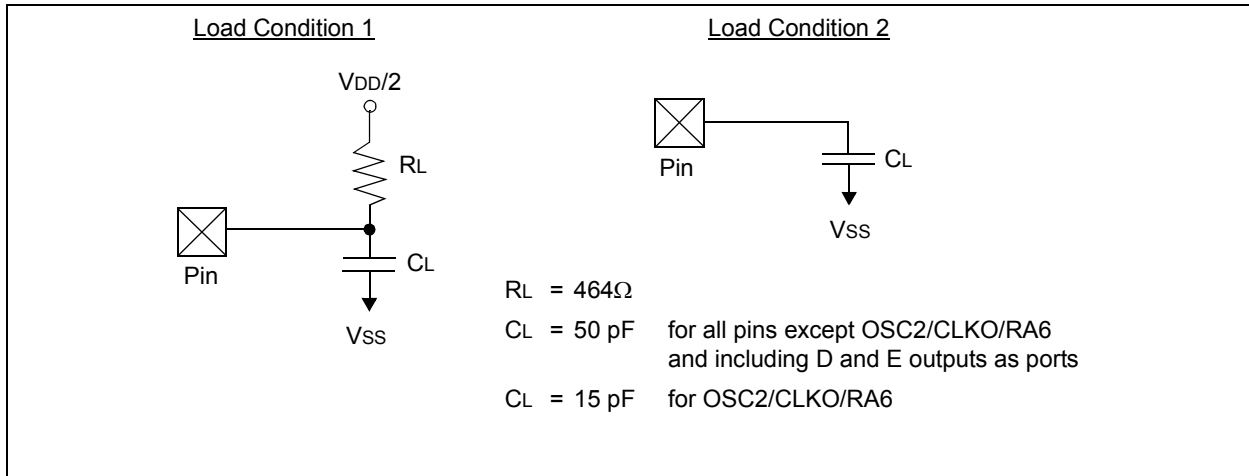
## 28.4.2 TIMING CONDITIONS

The temperature and voltages specified in Table 28-7 apply to all timing specifications unless otherwise noted. Figure 28-4 specifies the load conditions for the timing specifications.

**TABLE 28-7: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC**

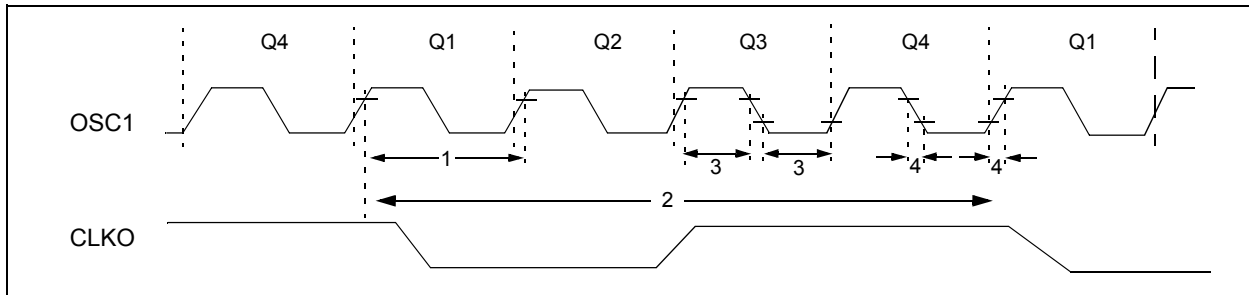
<b>AC CHARACTERISTICS</b>	<b>Standard Operating Conditions (unless otherwise stated)</b>	
	Operating temperature	$-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial
	Operating voltage $V_{DD}$ range	as described in <b>Section 28.1</b> and <b>Section 28.3</b> .

**FIGURE 28-4: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**



## 28.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

**FIGURE 28-5: EXTERNAL CLOCK TIMING**





# PIC18F46J11 FAMILY

**TABLE 28-8: EXTERNAL CLOCK TIMING REQUIREMENTS**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	FOSC	External CLKI Frequency <sup>(1)</sup>	DC	48	MHz	EC Oscillator mode
		Oscillator Frequency <sup>(1)</sup>	4	12	MHz	ECPLL Oscillator mode HS Oscillator mode HSPLL Oscillator mode
1	TOSC	External CLKI Period <sup>(1)</sup>	20.8	—	ns	EC Oscillator mode
		Oscillator Period <sup>(1)</sup>	83.3	—	ns	ECPLL Oscillator mode HS Oscillator mode HSPLL Oscillator mode
2	TCY	Instruction Cycle Time <sup>(1)</sup>	83.3	DC	ns	TCY = 4/FOSC, Industrial
3	TOSL, TOSH	External Clock in (OSC1) High or Low Time	10	—	ns	EC Oscillator mode
4	TOSR, TOSF	External Clock in (OSC1) Rise or Fall Time	—	7.5	ns	EC Oscillator mode

**Note 1:** Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

**TABLE 28-9: PLL CLOCK TIMING SPECIFICATIONS**

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
F10	FPLLIN	PLL Input Frequency Range	4	—	12	MHz	
F11	FPLLO	PLL Output Frequency (4x FPLLIN)	16	—	48	MHz	
F12	t <sub>rc</sub>	PLL Start-up Time (lock time)	—	—	2	ms	

† Data in “Typ” column is at 3.3V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

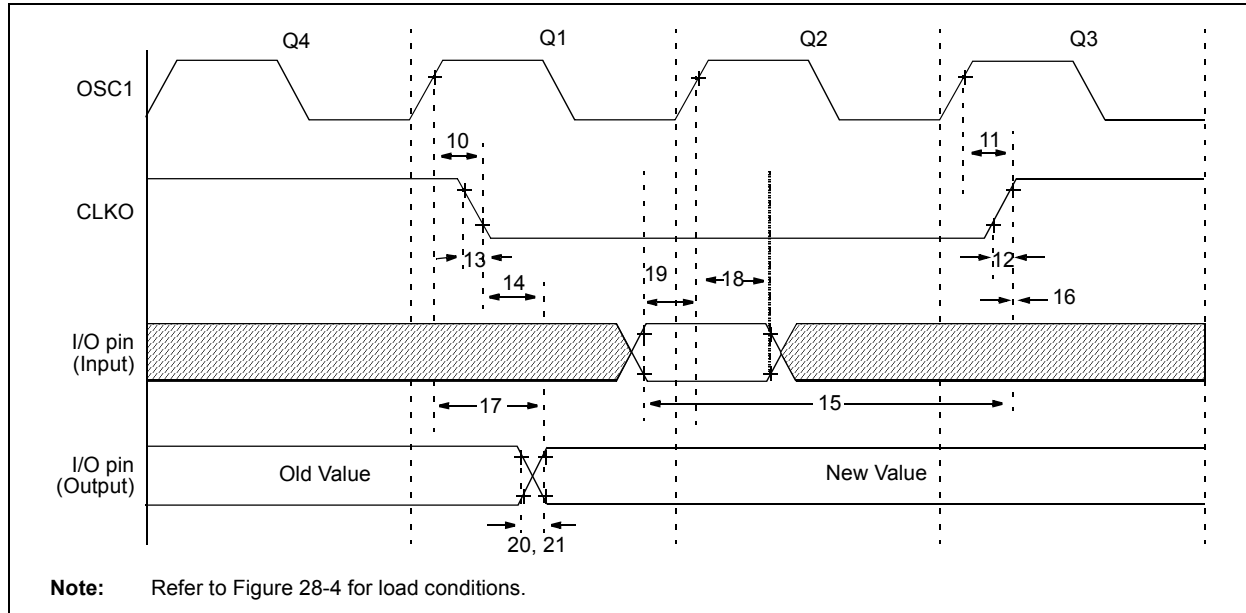
**TABLE 28-10: INTERNAL RC ACCURACY (INTOSC AND INTRC SOURCES)**

Param No.	Device	Min	Typ	Max	Units	Conditions	
INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz, 31 kHz <sup>(1)</sup>							
All Devices		-1	+/-0.15	+1	%	0°C to +85°C	VDD = 2.0-3.3V
		-1	+/-0.25	+1	%	-40°C to +85°C	VDD = 2.0-3.6V, VDDCORE = 2.0-2.7V
INTRC Accuracy @ Freq = 31 kHz <sup>(1)</sup>							
All Devices		20.3	—	42.2	kHz	-40°C to +85°C	VDD = 2.0-3.6V, VDDCORE = 2.0-2.7V

**Note 1:** The accuracy specification of the 31 kHz clock is determined by which source is providing it at a given time. When INTSRC (OSCTUNE<7>) is ‘1’, use the INTOSC accuracy specification. When INTSRC is ‘0’, use the INTRC accuracy specification.

# PIC18F46J11 FAMILY

**FIGURE 28-6: CLKO AND I/O TIMING**



**TABLE 28-11: CLKO AND I/O TIMING REQUIREMENTS**

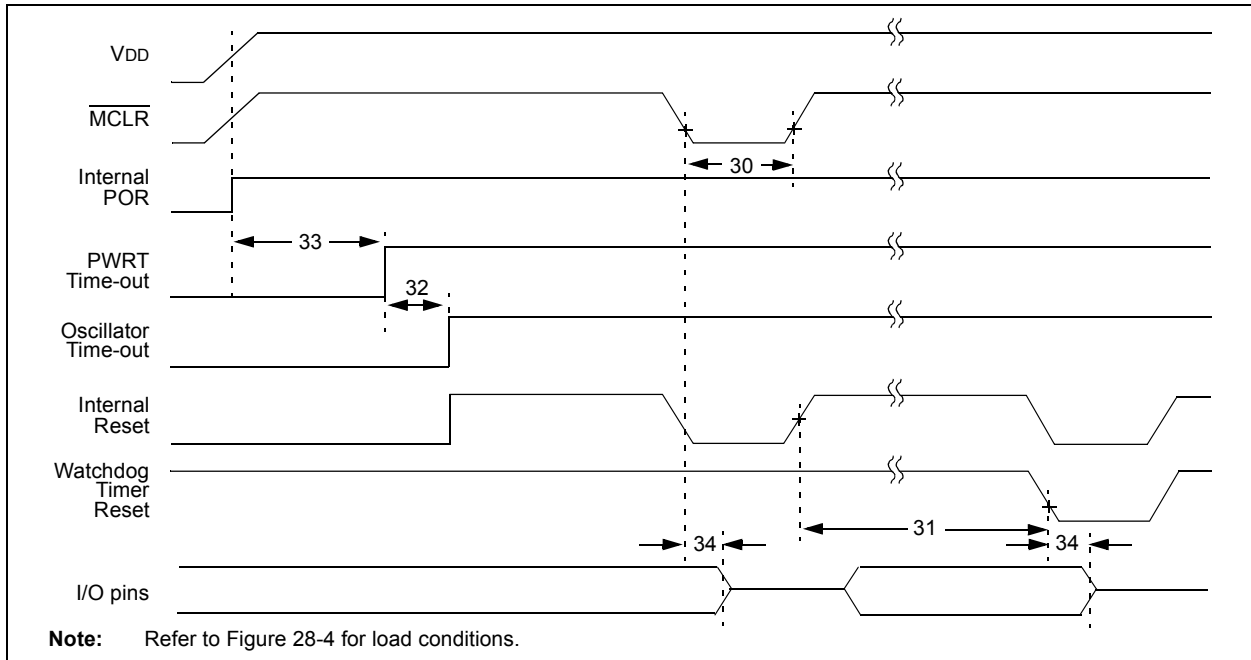
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
10	TosH2ckL	OSC1 $\uparrow$ to CLKO $\downarrow$	—	75	200	ns	(Note 1)
11	TosH2ckH	OSC1 $\uparrow$ to CLKO $\uparrow$	—	75	200	ns	(Note 1)
12	TckR	CLKO Rise Time	—	15	30	ns	(Note 1)
13	TckF	CLKO Fall Time	—	15	30	ns	(Note 1)
14	TckL2ioV	CLKO $\downarrow$ to Port Out Valid	—	—	$0.5 T_{CY} + 20$	ns	
15	TioV2ckH	Port In Valid before CLKO $\uparrow$	$0.25 T_{CY} + 25$	—	—	ns	
16	TckH2ioI	Port In Hold after CLKO $\uparrow$	0	—	—	ns	
17	TosH2ioV	OSC1 $\uparrow$ (Q1 cycle) to Port Out Valid	—	50	150	ns	
18	TosH2ioI	OSC1 $\uparrow$ (Q2 cycle) to Port Input Invalid (I/O in hold time)	100	—	—	ns	
19	TioV2osH	Port Input Valid to OSC1 $\uparrow$ (I/O in setup time)	0	—	—	ns	
20	TioR	Port Output Rise Time	—	—	6	ns	
21	TioF	Port Output Fall Time	—	—	5	ns	
22†	TINP	INTx pin High or Low Time	$T_{CY}$	—	—	ns	
23†	TRBP	RB7:RB4 Change INTx High or Low Time	$T_{CY}$	—	—	ns	

† These parameters are asynchronous events not related to any internal clock edges.

**Note 1:** Measurements are taken in EC mode, where CLKO output is  $4 \times T_{osc}$ .

# PIC18F46J11 FAMILY

**FIGURE 28-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**TABLE 28-12: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS**

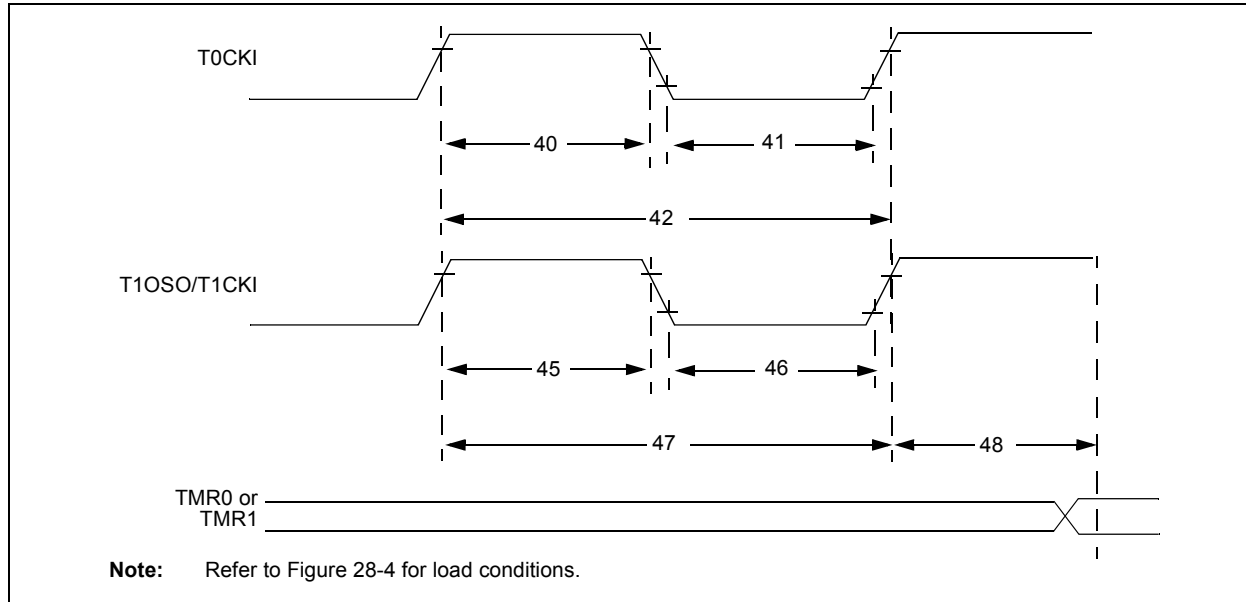
Param. No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	TMCL	MCLR Pulse-Width (low)	2	—	—	μs	—
31	TWDT	Watchdog Timer Time-out Period (no postscaler)	2.8	4.0	5.3	ms	—
32	TOST	Oscillator Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	—	1.0	—	ms	—
34	TIOZ	I/O High-Impedance from MCLR Low or Watchdog Timer Reset	—	—	3 TCY + 2	μs	<b>(Note 1)</b>
36	TIRVST	Time for Internal Reference Voltage to become Stable	—	20	—	μs	—
37	TLVD	High/Low-Voltage Detect Pulse Width	—	200	—	μs	—
38	TCSD	CPU Start-up Time	—	200	—	μs	<b>(Note 2)</b>

**Note 1:** The maximum TIOZ is the lesser of (3 TCY + 2 μs) or 700 μs.

**Note 2:** MCLR rising edge to code execution, assuming TPWRT (and TOST if applicable) has already expired.

# PIC18F46J11 FAMILY

**FIGURE 28-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**



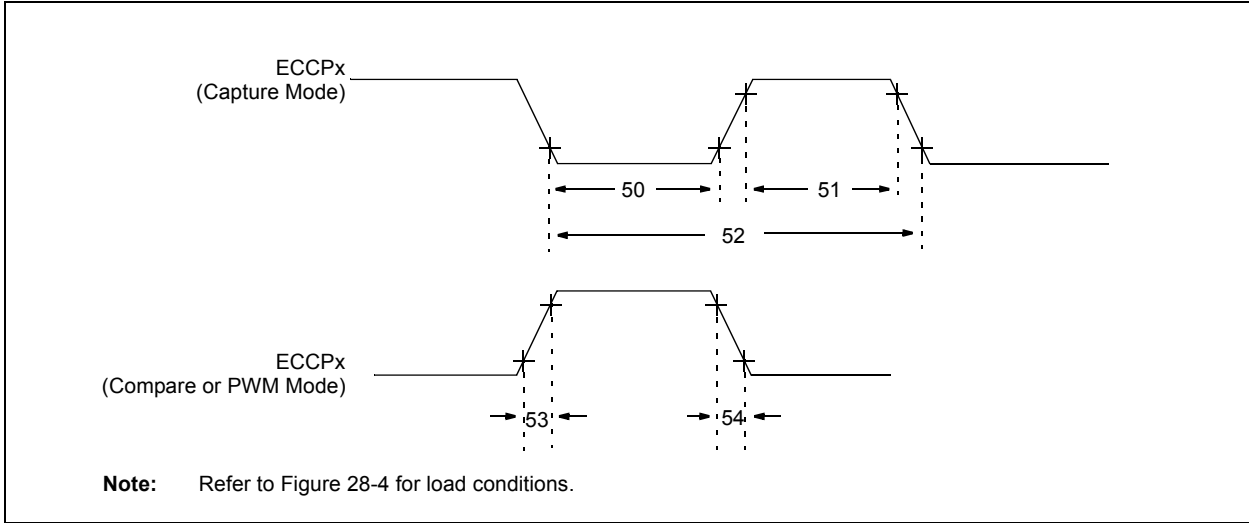
**TABLE 28-13: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
40	T <sub>T0H</sub>	T0CKI High Pulse Width	No prescaler	$0.5 T_{CY} + 20$	—	ns	
			With prescaler	10	—	ns	
41	T <sub>T0L</sub>	T0CKI Low Pulse Width	No prescaler	$0.5 T_{CY} + 20$	—	ns	
			With prescaler	10	—	ns	
42	T <sub>T0P</sub>	T0CKI Period	No prescaler	$T_{CY} + 10$	—	ns	
			With prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$	—	ns	
45	T <sub>T1H</sub>	T1CKI/T3CKI High Time	Synchronous, no prescaler	$0.5 T_{CY} + 20$	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
46	T <sub>T1L</sub>	T1CKI/T3CKI Low Time	Synchronous, no prescaler	$0.5 T_{CY} + 5$	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
47	T <sub>T1P</sub>	T1CKI/T3CKI Input Period	Synchronous	Greater of: 20 ns or $(T_{CY} + 40)/N$	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous	83	—	ns	
	FT1	T1CKI Input Frequency Range <sup>(1)</sup>		DC	12	MHz	
48	T <sub>CKE2TMR1</sub>	Delay from External T1CKI Clock Edge to Timer Increment		$2 T_{OSC}$	$7 T_{OSC}$	—	

**Note 1:** The Timer1 oscillator is designed to drive 32.768 kHz crystals. When T1CKI is used as a digital input, frequencies up to 12 MHz are supported.

# PIC18F46J11 FAMILY

**FIGURE 28-9: ENHANCED CAPTURE/COMPARE/PWM TIMINGS**

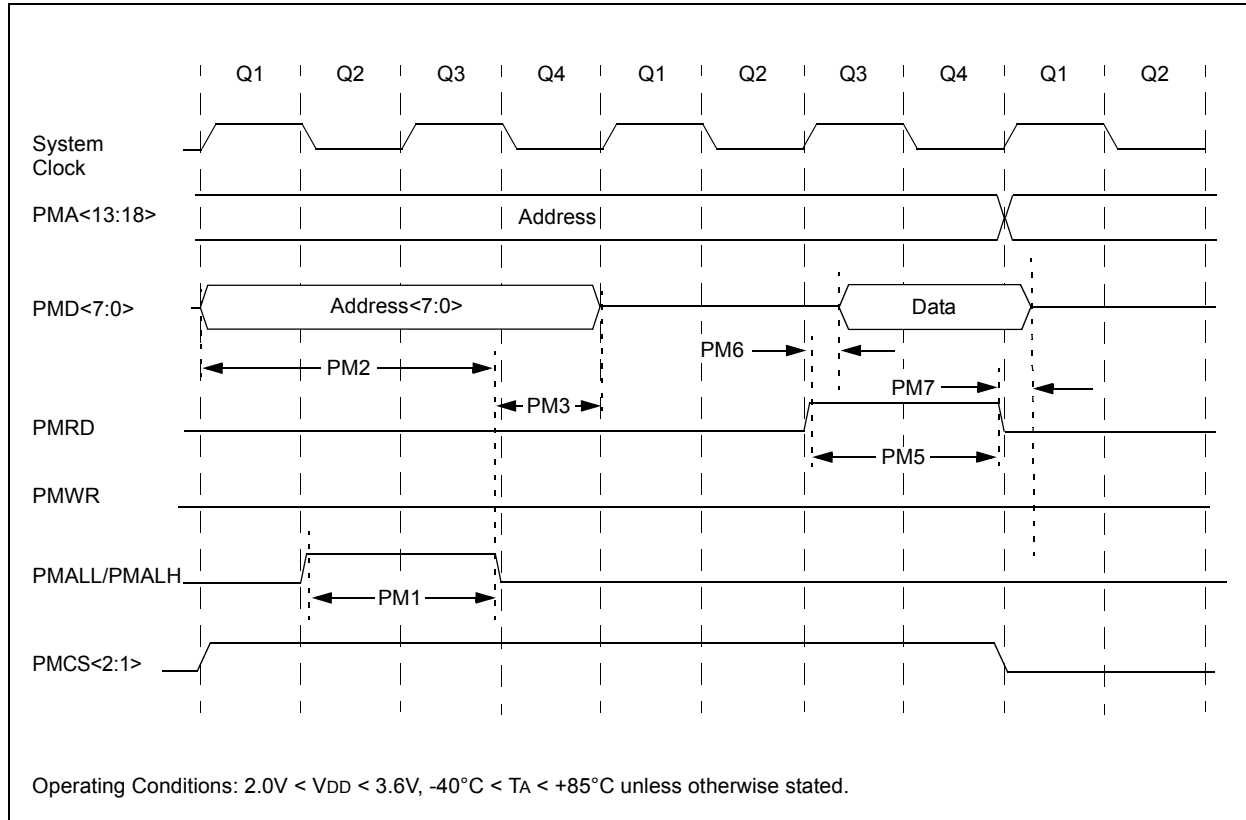


**TABLE 28-14: ENHANCED CAPTURE/COMPARE/PWM REQUIREMENTS**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions	
50	TccL	ECCPx Input Low Time	No prescaler	$0.5 T_{CY} + 20$	—	ns	
		With prescaler	10	—	ns		
51	TccH	ECCPx Input High Time	No prescaler	$0.5 T_{CY} + 20$	—	ns	
		With prescaler	10	—	ns		
52	TccP	ECCPx Input Period	$\frac{3 T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)	
53	TccR	ECCPx Output Fall Time	—	25	ns		
54	TccF	ECCPx Output Rise Time	—	25	ns		

# PIC18F46J11 FAMILY

**FIGURE 28-10: PARALLEL MASTER PORT READ TIMING DIAGRAM**

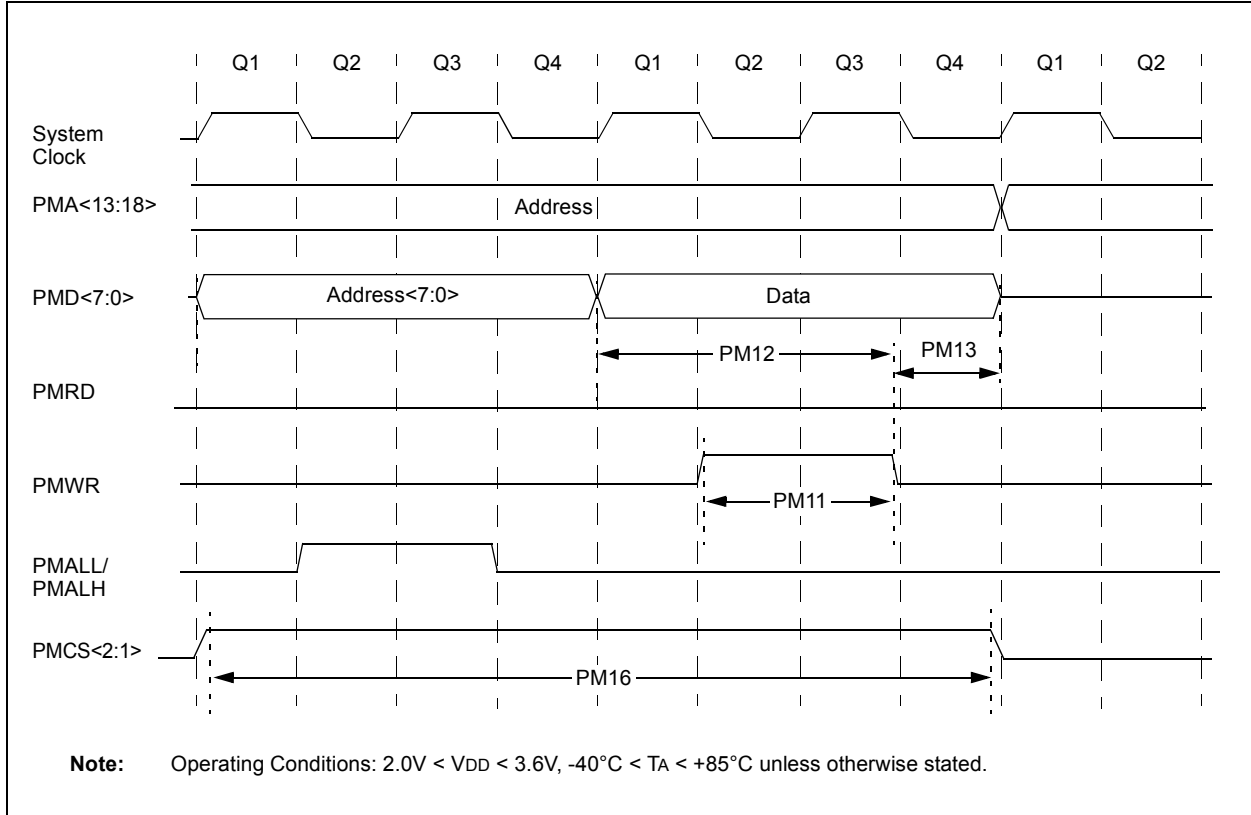


**TABLE 28-15: PARALLEL MASTER PORT READ TIMING REQUIREMENTS**

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
PM1		PMALL/PMALH Pulse Width	—	0.5 T <sub>CY</sub>	—	ns
PM2		Address Out Valid to PMALL/PMALH Invalid (address setup time)	—	0.75 T <sub>CY</sub>	—	ns
PM3		PMALL/PMALH Invalid to Address Out Invalid (address hold time)	—	0.25 T <sub>CY</sub>	—	ns
PM5		PMRD Pulse Width	—	0.5 T <sub>CY</sub>	—	ns
PM6		PMRD or PMENB Active to Data In Valid (data setup time)	—	—	—	ns
PM7		PMRD or PMENB Inactive to Data In Invalid (data hold time)	—	—	—	ns

# PIC18F46J11 FAMILY

**FIGURE 28-11: PARALLEL MASTER PORT WRITE TIMING DIAGRAM**

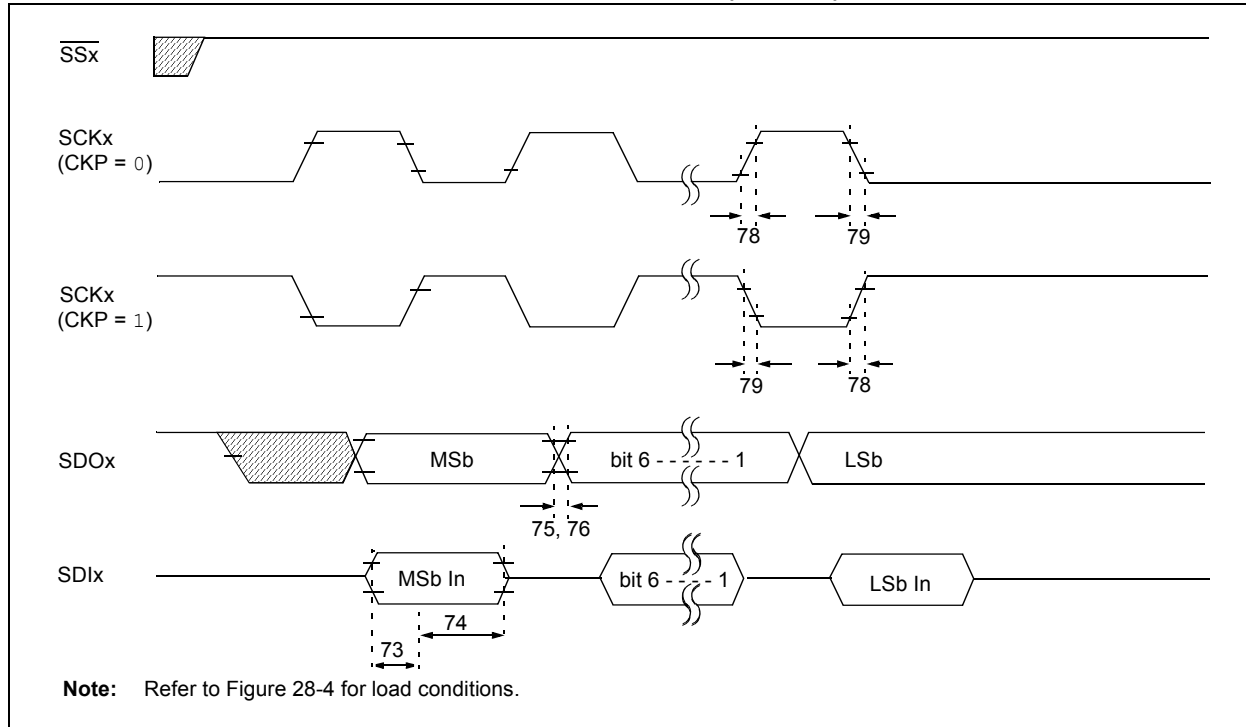


**TABLE 28-16: PARALLEL MASTER PORT WRITE TIMING REQUIREMENTS**

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
PM11		PMWR Pulse Width	—	0.5 T <sub>CY</sub>	—	ns
PM12		Data Out Valid before PMWR or PMENB goes Inactive (data setup time)	—	—	—	ns
PM13		PMWR or PMEMB Invalid to Data Out Invalid (data hold time)	—	—	—	ns
PM16		PMCS Pulse Width	T <sub>CY</sub> - 5	—	—	ns

# PIC18F46J11 FAMILY

**FIGURE 28-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)**



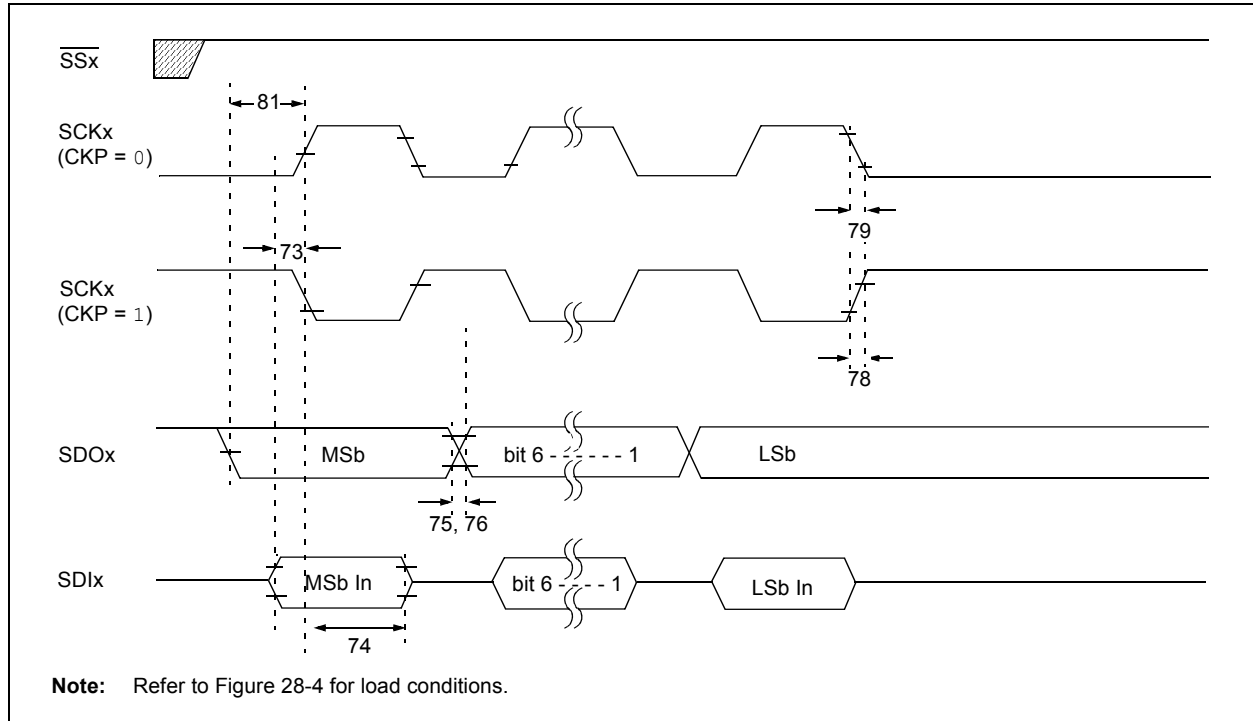
**TABLE 28-17: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TdIV2sCH, TdIV2sCL	Setup Time of SDIx Data Input to SCKx Edge	35 100	— —	ns ns	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V, VDDCORE = 2.15V
73A	TB2B	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 TcY + 40	—	ns	
74	TsCH2dIL, TsCL2dIL	Hold Time of SDIx Data Input to SCKx Edge	30 83	— —	ns ns	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V
75	TDoR	SDOx Data Output Rise Time	—	25	ns	PORTB or PORTC
76	TDoF	SDOx Data Output Fall Time	—	25	ns	PORTB or PORTC
78	TsCR	SCKx Output Rise Time (Master mode)	—	25	ns	PORTB or PORTC
79	TsCF	SCKx Output Fall Time (Master mode)	—	25	ns	PORTB or PORTC



# PIC18F46J11 FAMILY

**FIGURE 28-13: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)**

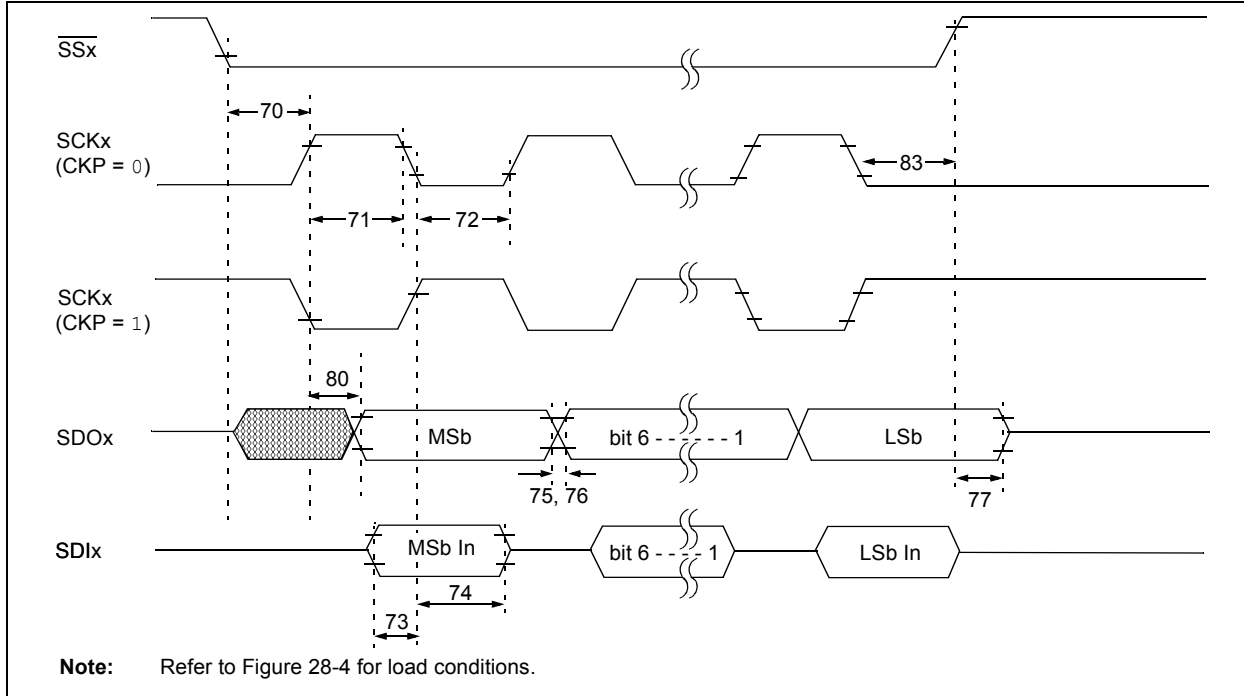


**TABLE 28-18: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	$T_{DiV2sCH}$ , $T_{DiV2sCL}$	Setup Time of SDIx Data Input to SCKx Edge	35 100	—	ns ns	$V_{DD} = 3.3V$ , $V_{DDCORE} = 2.5V$ $V_{DD} = 2.15V$ , $V_{DDCORE} = 2.15V$
74	$T_{SCH2DiL}$ , $T_{SCL2DiL}$	Hold Time of SDIx Data Input to SCKx Edge	30 83	—	ns ns	$V_{DD} = 3.3V$ , $V_{DDCORE} = 2.5V$ $V_{DD} = 2.15V$
75	$T_{DoR}$	SDOx Data Output Rise Time	—	25	ns	PORTB or PORTC
76	$T_{DoF}$	SDOx Data Output Fall Time	—	25	ns	PORTB or PORTC
78	$T_{ScR}$	SCKx Output Rise Time (Master mode)	—	25	ns	PORTB or PORTC
79	$T_{ScF}$	SCKx Output Fall Time (Master mode)	—	25	ns	PORTB or PORTC
81	$T_{DoV2sCH}$ , $T_{DoV2sCL}$	SDOx Data Output Setup to SCKx Edge	$T_{CY}$	—	ns	

# PIC18F46J11 FAMILY

**FIGURE 28-14: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)**



**TABLE 28-19: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)**

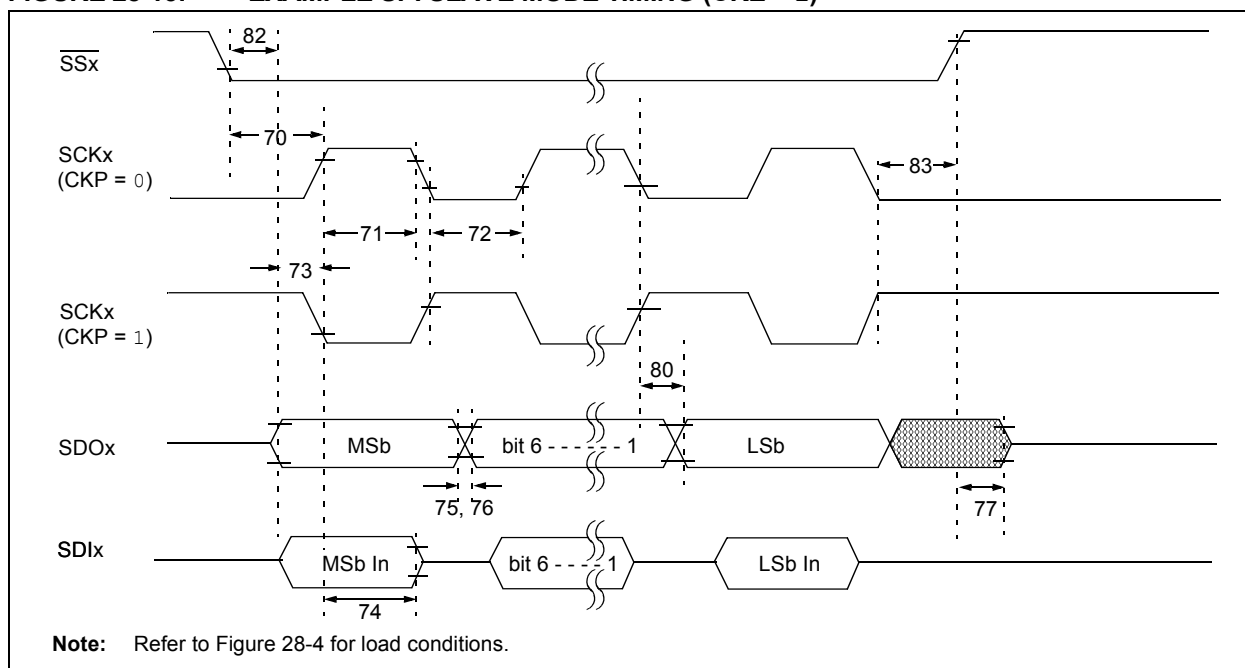
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx} \downarrow$ to SCKx $\downarrow$ or SCKx $\uparrow$ Input	3 Tcy	—	ns	
70A	TssL2WB	$\overline{SSx} \downarrow$ to Write to SSPxBUF	3 Tcy	—	ns	
71	Tsch	SCKx Input High Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns
71A			Single byte	40	—	ns
72	TscL	SCKx Input Low Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns
72A			Single byte	40	—	ns
73	TdIV2scH, TdIV2scL	Setup Time of SDIx Data Input to SCKx Edge	25	—	ns	
73A	Tb2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2	1.5 Tcy + 40	—	ns	(Note 2)
74	Tsch2dIL, TscL2dIL	Hold Time of SDIx Data Input to SCKx Edge	35	—	ns	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V
			100	—	ns	
75	TdoR	SDOx Data Output Rise Time	—	25	ns	PORTB or PORTC
76	TdoF	SDOx Data Output Fall Time	—	25	ns	PORTB or PORTC
77	TssH2boZ	$\overline{SSx} \uparrow$ to SDOx Output High-Impedance	10	70	ns	
80	Tsch2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	50	ns	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V
			100	ns		
83	Tsch2ssH, TscL2ssH	$\overline{SSx} \uparrow$ 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.

# PIC18F46J11 FAMILY

**FIGURE 28-15: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)**



**TABLE 28-20: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)**

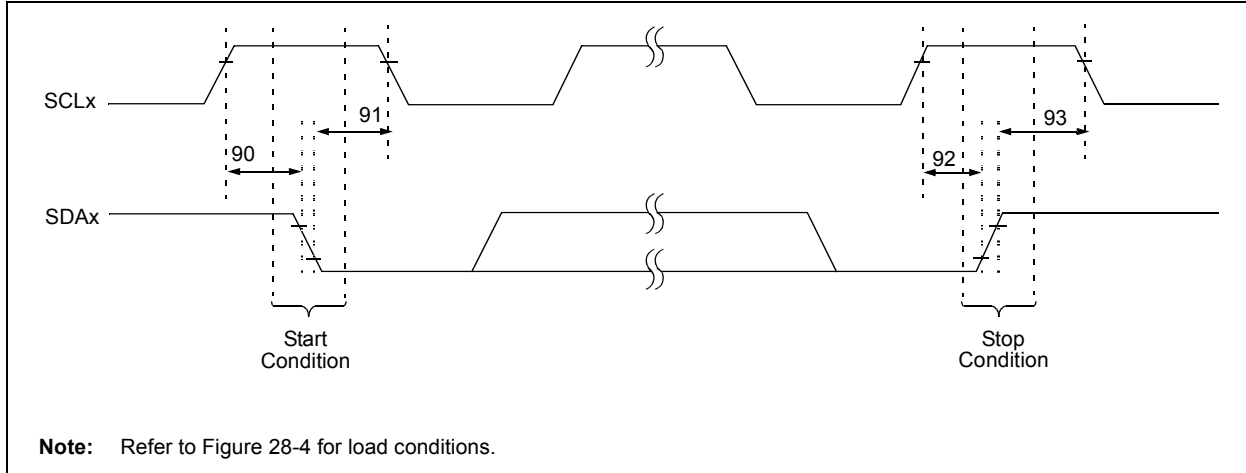
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2sch, TssL2scL	SSx ↓ to SCKx ↓ or SCKx ↑ Input	3 Tcy	—	ns	
70A	TssL2WB	SSx ↓ to Write to SSPxBUF	3 Tcy	—	ns	
71	Tsch	SCKx Input High Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns
71A			Single byte	40	—	ns
72	TscL	SCKx Input Low Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns
72A			Single byte	40	—	ns
73	TdIV2sch, TdIV2scL	Setup Time of SDIx Data Input to SCKx Edge	25	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2	1.5 Tcy + 40	—	ns	(Note 2)
74	Tsch2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	35	—	ns	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V
			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	70	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	VDD = 3.3V, VDDCORE = 2.5V VDD = 2.15V
			—	100	ns	
81	TdoV2sch, TdoV2scL	SDOx Data Output Setup to SCKx Edge	Tcy	—	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.

# PIC18F46J11 FAMILY

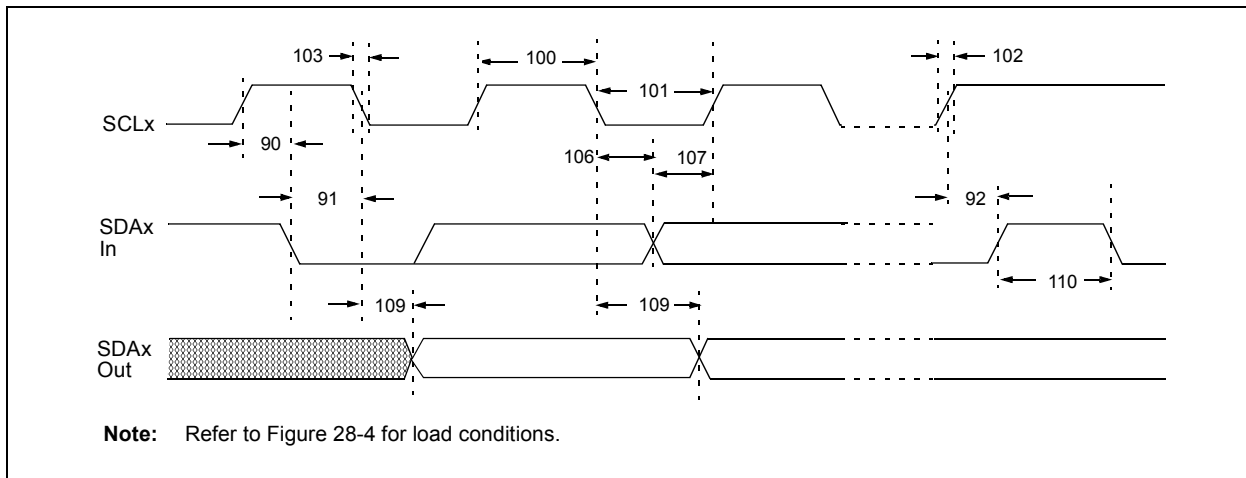
**FIGURE 28-16: I<sup>2</sup>C™ BUS START/STOP BITS TIMING**



**TABLE 28-21: I<sup>2</sup>C™ 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 28-17: I<sup>2</sup>C™ BUS DATA TIMING**



# PIC18F46J11 FAMILY

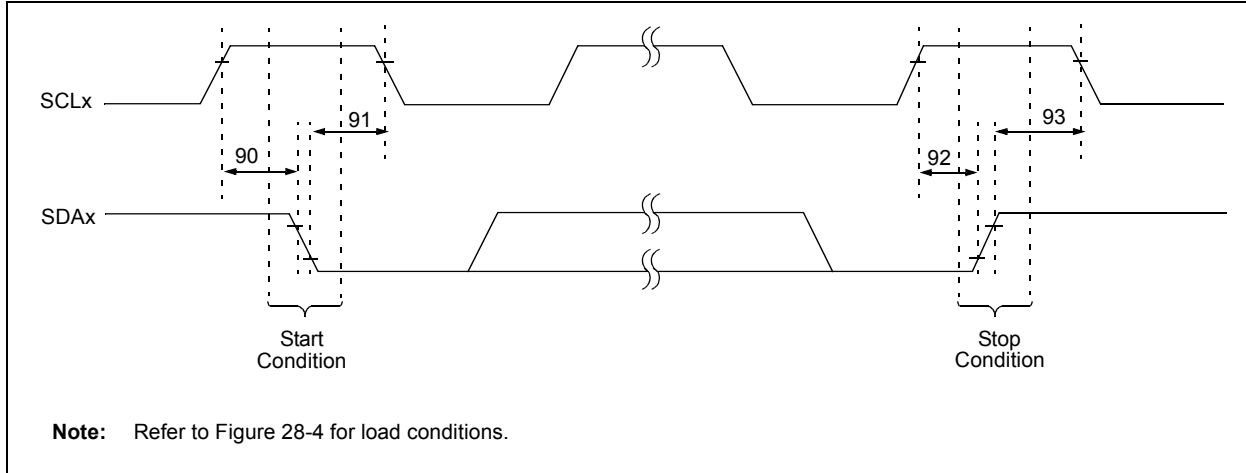
**TABLE 28-22: I<sup>2</sup>C™ BUS DATA REQUIREMENTS (SLAVE MODE)**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	μs
			400 kHz mode	0.6	—	μs
			MSSP modules	1.5 T <sub>CY</sub>	—	
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	μs
			400 kHz mode	1.3	—	μs
			MSSP modules	1.5 T <sub>CY</sub>	—	
102	TR	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns
			400 kHz mode	20 + 0.1 C <sub>B</sub>	300	ns
103	TF	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns
			400 kHz mode	20 + 0.1 C <sub>B</sub>	300	ns
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	μs
			400 kHz mode	0.6	—	μs
91	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	μs
			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
			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
			400 kHz mode	—	—	ns
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	μs
			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 I<sup>2</sup>C™ bus device can be used in a Standard mode I<sup>2</sup>C 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, TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I<sup>2</sup>C bus specification), before the SCLx line is released.

# PIC18F46J11 FAMILY

**FIGURE 28-18: MSSPx I<sup>2</sup>C™ BUS START/STOP BITS TIMING WAVEFORMS**

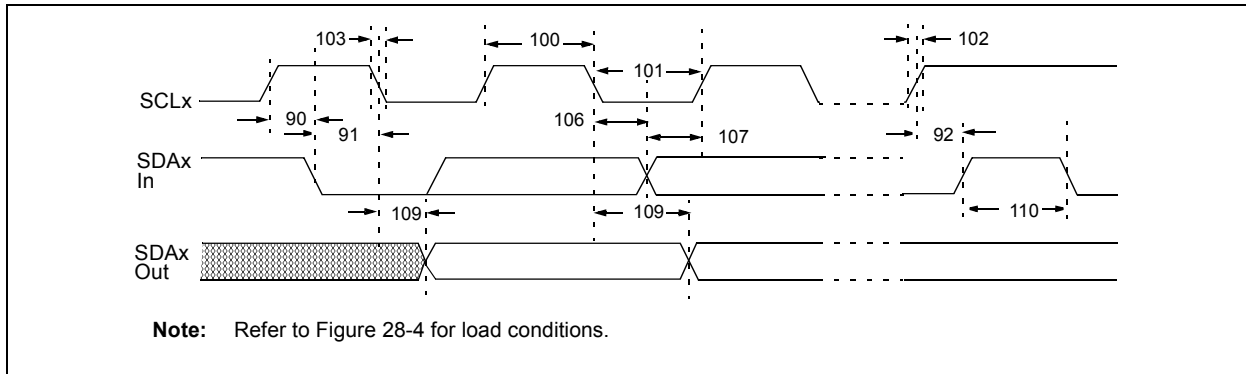


**TABLE 28-23: MSSPx I<sup>2</sup>C™ BUS START/STOP BITS REQUIREMENTS**

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 <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	
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 <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	
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 <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	
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 <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	

**Note 1:** Maximum pin capacitance = 10 pF for all I<sup>2</sup>C™ pins.

**FIGURE 28-19: MSSPx I<sup>2</sup>C™ BUS DATA TIMING**



# PIC18F46J11 FAMILY

**TABLE 28-24: MSSPx I<sup>2</sup>C™ BUS DATA REQUIREMENTS**

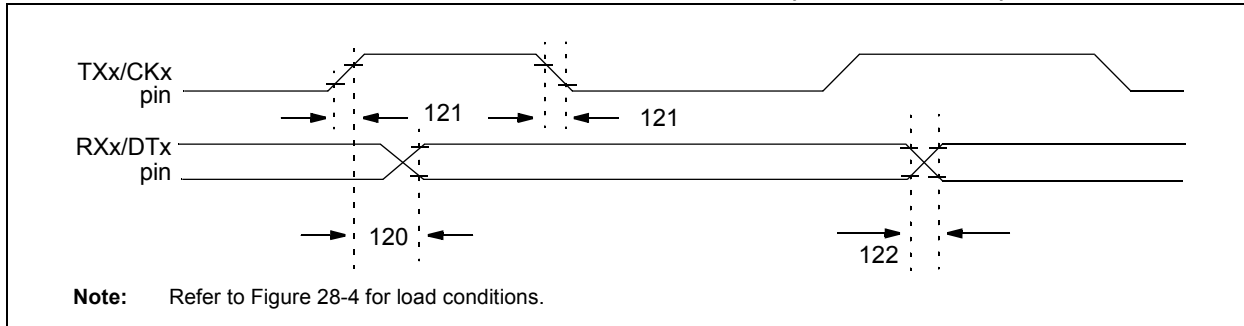
Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms
			1 MHz mode <sup>(1)</sup>	2(Tosc)(BRG + 1)	—	ms
101	TLOW	Clock Low Time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms
			1 MHz mode <sup>(1)</sup>	2(Tosc)(BRG + 1)	—	ms
102	TR	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns
			400 kHz mode	20 + 0.1 C <sub>B</sub>	300	ns
			1 MHz mode <sup>(1)</sup>	—	300	ns
103	TF	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns
			400 kHz mode	20 + 0.1 C <sub>B</sub>	300	ns
			1 MHz mode <sup>(1)</sup>	—	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 <sup>(1)</sup>	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 <sup>(1)</sup>	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 <sup>(1)</sup>	TBD	—	ns
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns
			400 kHz mode	100	—	ns
			1 MHz mode <sup>(1)</sup>	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 <sup>(1)</sup>	2(Tosc)(BRG + 1)	—	ms
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns
			400 kHz mode	—	1000	ns
			1 MHz mode <sup>(1)</sup>	—	—	ns
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	ms
			400 kHz mode	1.3	—	ms
			1 MHz mode <sup>(1)</sup>	TBD	—	ms
D102	CB	Bus Capacitive Loading	—	400	pF	

**Note 1:** Maximum pin capacitance = 10 pF for all I<sup>2</sup>C™ pins.

- 2:** A Fast mode I<sup>2</sup>C bus device can be used in a Standard mode I<sup>2</sup>C 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.

# PIC18F46J11 FAMILY

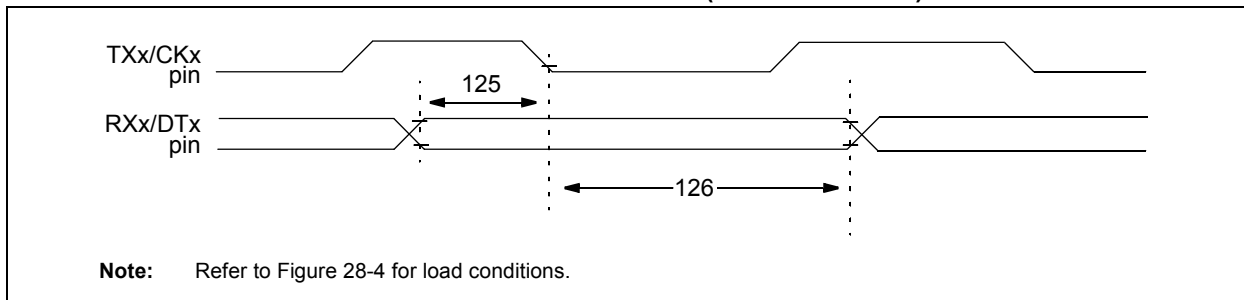
**FIGURE 28-20: EUSARTx SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**



**TABLE 28-25: EUSARTx SYNCHRONOUS TRANSMISSION REQUIREMENTS**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
120	T <sub>CKH2DTV</sub>	Sync XMIT (Master and Slave) Clock High to Data Out Valid	—	40	ns	
121	T <sub>CKRF</sub>	Clock Out Rise Time and Fall Time (Master mode)	—	20	ns	
122	T <sub>DTRF</sub>	Data Out Rise Time and Fall Time	—	20	ns	

**FIGURE 28-21: EUSARTx SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**



**TABLE 28-26: EUSARTx SYNCHRONOUS RECEIVE REQUIREMENTS**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
125	T <sub>DTV2CKL</sub>	Sync RCV (Master and Slave) Data Hold before CKx ↓ (DTx hold time)	10	—	ns	
126	T <sub>CKL2DTL</sub>	Data Hold after CKx ↓ (DTx hold time)	15	—	ns	



# PIC18F46J11 FAMILY

**TABLE 28-27: A/D CONVERTER CHARACTERISTICS: PIC18F46J11 FAMILY (INDUSTRIAL)**

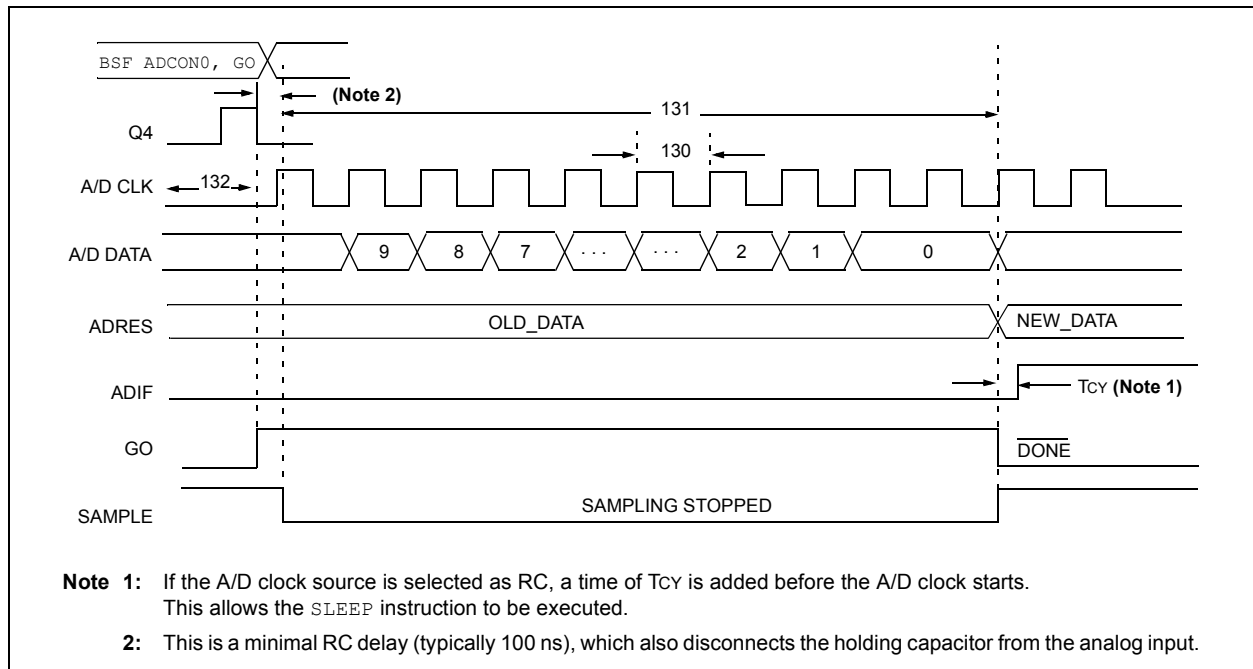
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
A01	NR	Resolution	—	—	10	bit	$\Delta V_{REF} \geq 3.0V$
A03	EIL	Integral Linearity Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A04	EDL	Differential Linearity Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A06	E0FF	Offset Error	—	—	$<\pm 3$	LSb	$\Delta V_{REF} \geq 3.0V$
A07	E0N	Gain Error	—	—	$<\pm 3.5$	LSb	$\Delta V_{REF} \geq 3.0V$
A10		Monotonicity	Guaranteed <sup>(1)</sup>			—	$V_{SS} \leq V_{AIN} \leq V_{REF}$
A20	$\Delta V_{REF}$	Reference Voltage Range (VREFH – VREFL)	2.0 3	— —	— —	V V	$V_{DD} < 3.0V$ $V_{DD} \geq 3.0V$
A21	VREFH	Reference Voltage High	VSS	—	VREFH	V	
A22	VREFL	Reference Voltage Low	$V_{SS} - 0.3V$	—	$V_{DD} - 3.0V$	V	
A25	VAIN	Analog Input Voltage	VREFL	—	VREFH	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	2.5	k $\Omega$	
A50	IREF	VREF Input Current <sup>(2)</sup>	— —	— —	5 150	$\mu A$ $\mu A$	During VAIN acquisition. During A/D conversion cycle.

**Note 1:** The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

**2:** VREFH current is from RA3/AN3/VREF+/C1INB pin or VDD, whichever is selected as the VREFH source.

VREFL current is from RA2/AN2/VREF-/CVREF/C2INB pin or VSS, whichever is selected as the VREFL source.

**FIGURE 28-22: A/D CONVERSION TIMING**



**Note 1:** If the A/D clock source is selected as RC, a time of Tcy is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

**2:** This is a minimal RC delay (typically 100 ns), which also disconnects the holding capacitor from the analog input.

# PIC18F46J11 FAMILY

**TABLE 28-28: A/D CONVERSION REQUIREMENTS**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
130	TAD	A/D Clock Period	0.7	25.0 <sup>(1)</sup>	μs	TOSC based, VREF ≥ 3.0V
131	TcNV	Conversion Time (not including acquisition time) <sup>(2)</sup>	11 —	12 1	TAD μs	A/D RC Mode
132	TACQ	Acquisition Time <sup>(3)</sup>	1.4	—	μs	-40°C to +85°C
135	TswC	Switching Time from Convert → Sample	—	<b>(Note 4)</b>		
137	TDIS	Discharge Time	0.2	—	μs	

**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.

# PIC18F46J11 FAMILY

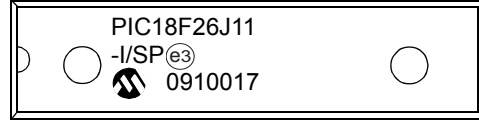
## 29.0 PACKAGING INFORMATION

### 29.1 Package Marking Information

28-Lead SPDIP



Example



28-Lead SSOP



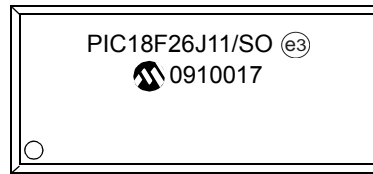
Example



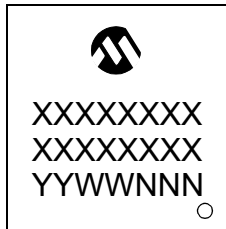
28-Lead SOIC (.300")



Example



28-Lead QFN



Example

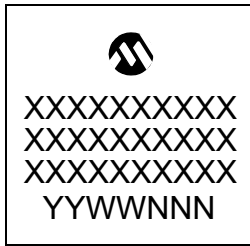


<b>Legend:</b>	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	*	Pb-free JEDEC designator for Matte Tin (Sn)
		This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.
<b>Note:</b>	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.	

# PIC18F46J11 FAMILY

---

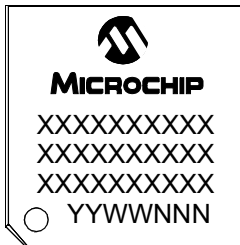
44-Lead QFN



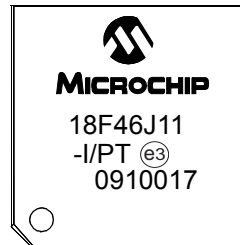
Example



44-Lead TQFP



Example



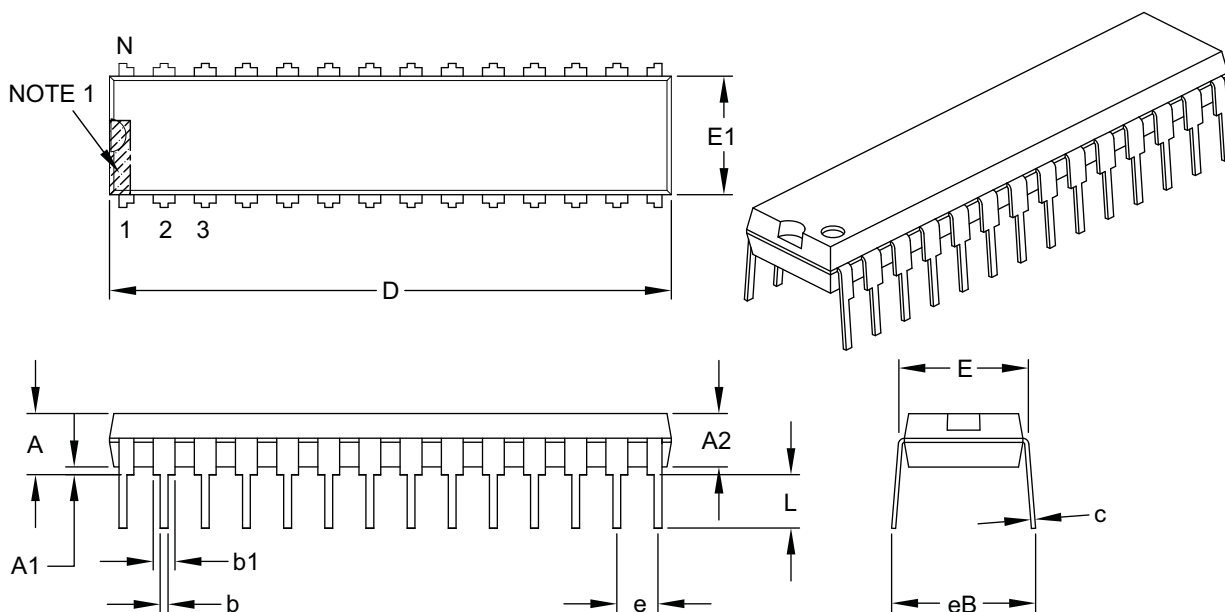
# PIC18F46J11 FAMILY

## 29.2 Package Details

The following sections give the technical details of the packages.

### 28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Dimension Limits	Units	INCHES		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

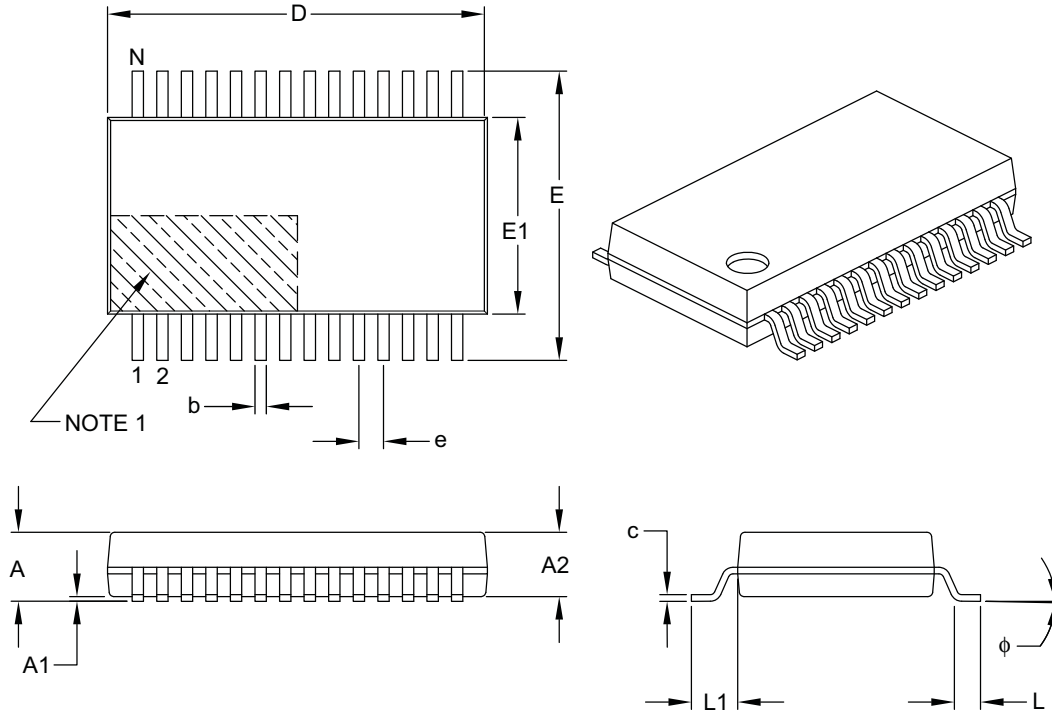
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

# PIC18F46J11 FAMILY

## 28-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Number of Pins	N		28		
Pitch	e		0.65 BSC		
Overall Height	A	–	–		2.00
Molded Package Thickness	A2	1.65	1.75		1.85
Standoff	A1	0.05	–		–
Overall Width	E	7.40	7.80		8.20
Molded Package Width	E1	5.00	5.30		5.60
Overall Length	D	9.90	10.20		10.50
Foot Length	L	0.55	0.75		0.95
Footprint	L1		1.25 REF		
Lead Thickness	c	0.09	–		0.25
Foot Angle	φ	0°	4°		8°
Lead Width	b	0.22	–		0.38

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

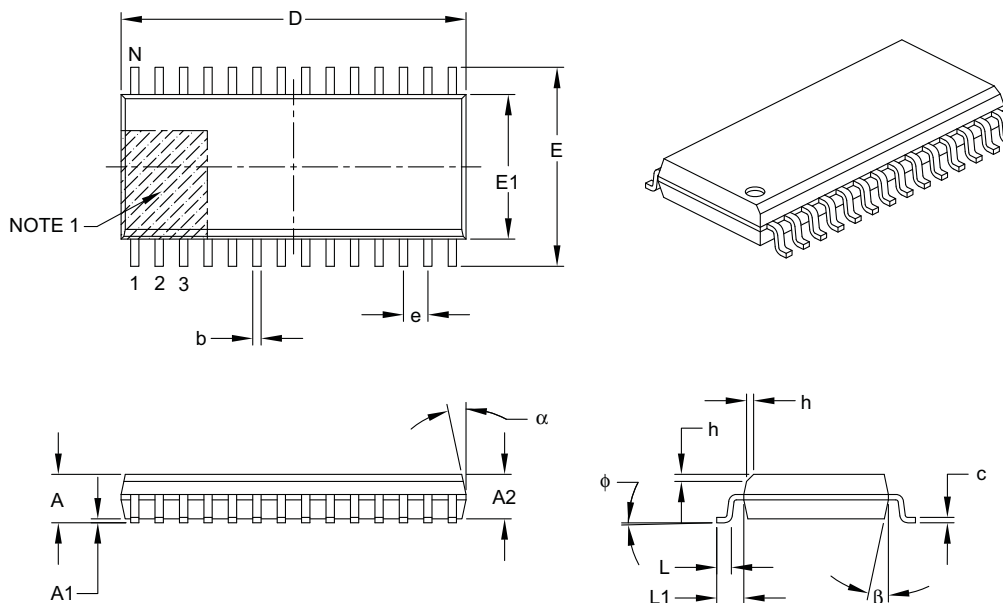
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

# PIC18F46J11 FAMILY

## 28-Lead Plastic Small Outline (SO) – Wide, 7.50 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	1.27 BSC		
Overall Height	A	–	–	2.65
Molded Package Thickness	A2	2.05	–	–
Standoff §	A1	0.10	–	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	17.90 BSC		
Chamfer (optional)	h	0.25	–	0.75
Foot Length	L	0.40	–	1.27
Footprint	L1	1.40 REF		
Foot Angle Top	φ	0°	–	8°
Lead Thickness	c	0.18	–	0.33
Lead Width	b	0.31	–	0.51
Mold Draft Angle Top	α	5°	–	15°
Mold Draft Angle Bottom	β	5°	–	15°

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

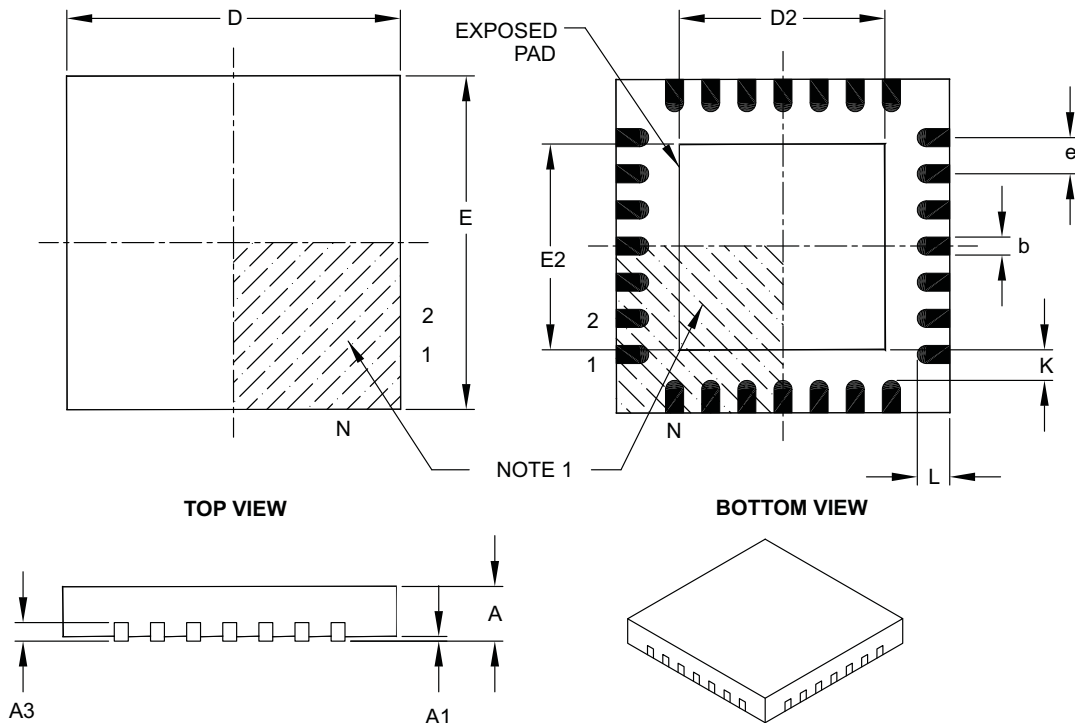
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

# PIC18F46J11 FAMILY

## 28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	6.00 BSC		
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D	6.00 BSC		
Exposed Pad Length	D2	3.65	3.70	4.20
Contact Width	b	0.23	0.30	0.35
Contact Length	L	0.50	0.55	0.70
Contact-to-Exposed Pad	K	0.20	–	–

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

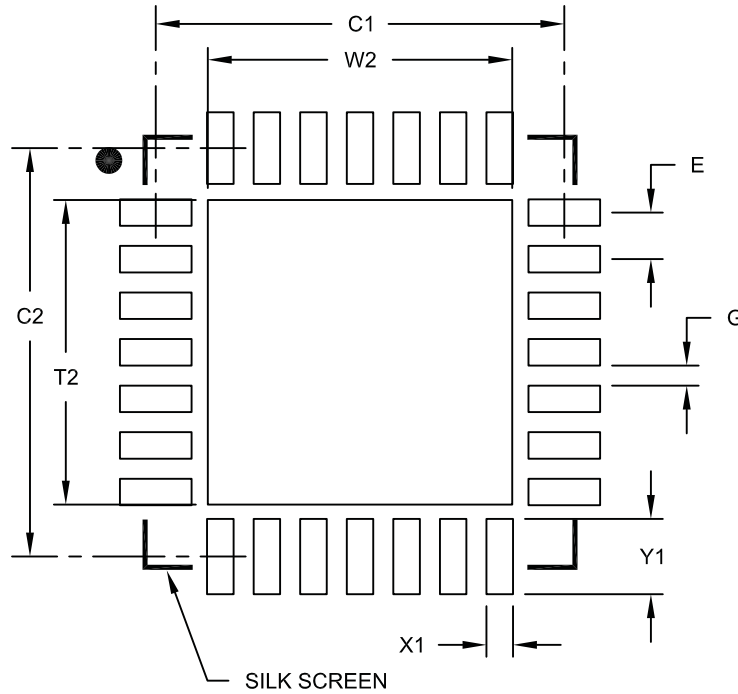
Microchip Technology Drawing C04-105B



# PIC18F46J11 FAMILY

## 28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

**Notes:**

1. Dimensioning and tolerancing per ASME Y14.5M

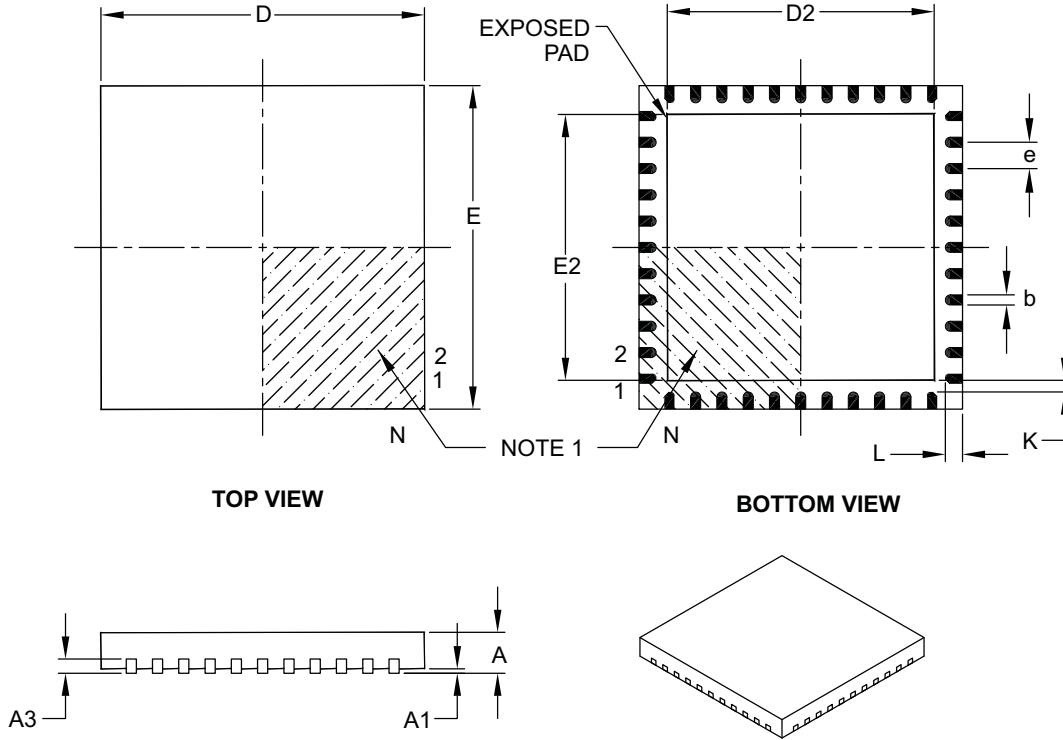
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

# PIC18F46J11 FAMILY

## 44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packageing>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	44		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	8.00 BSC		
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D	8.00 BSC		
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

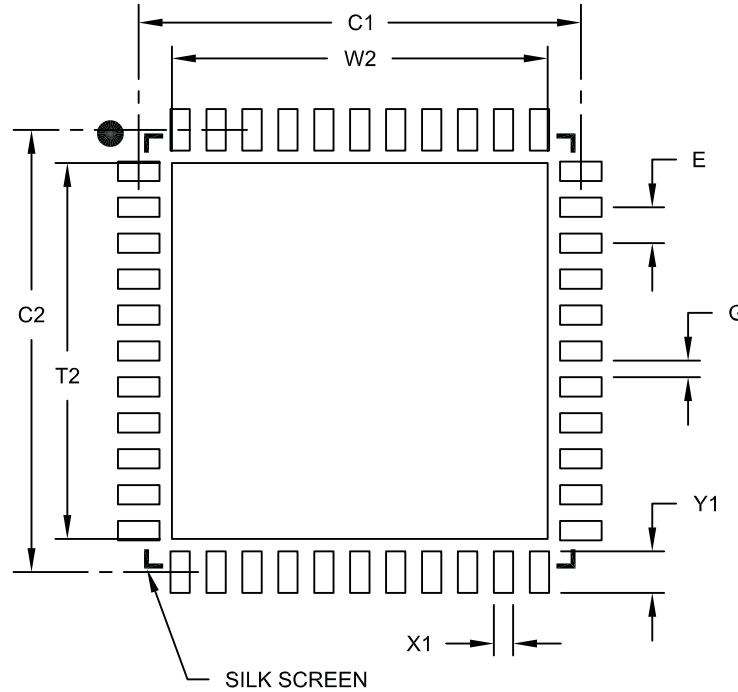
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

# PIC18F46J11 FAMILY

## 44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Optional Center Pad Width	W2			6.80
Optional Center Pad Length	T2			6.80
Contact Pad Spacing	C1		8.00	
Contact Pad Spacing	C2		8.00	
Contact Pad Width (X44)	X1			0.35
Contact Pad Length (X44)	Y1			0.80
Distance Between Pads	G	0.25		

**Notes:**

1. Dimensioning and tolerancing per ASME Y14.5M

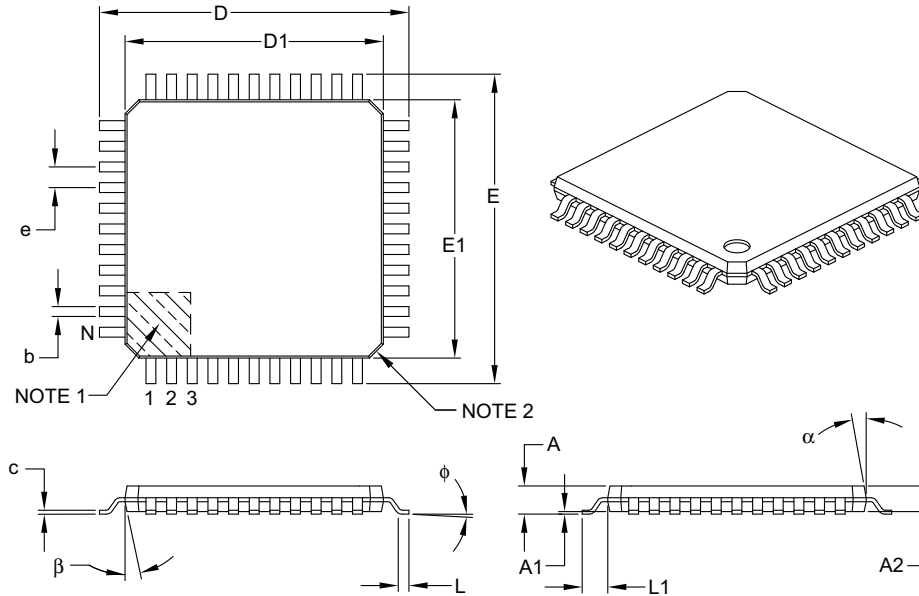
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2103A

# PIC18F46J11 FAMILY

## 44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm [TQFP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Leads	N	44		
Lead Pitch	e	0.80 BSC		
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1	1.00 REF		
Foot Angle	$\phi$	0°	3.5°	7°
Overall Width	E	12.00 BSC		
Overall Length	D	12.00 BSC		
Molded Package Width	E1	10.00 BSC		
Molded Package Length	D1	10.00 BSC		
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	$\alpha$	11°	12°	13°
Mold Draft Angle Bottom	$\beta$	11°	12°	13°

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Chamfers at corners are optional; size may vary.
- Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

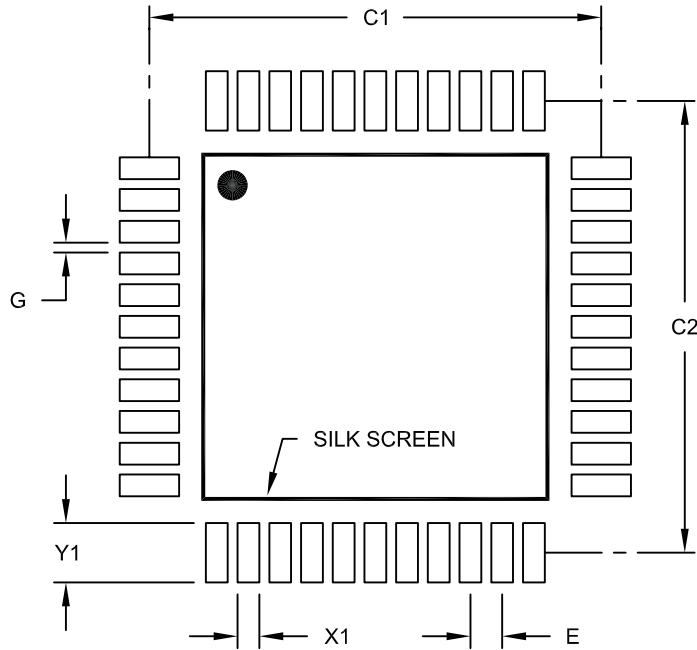
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

# PIC18F46J11 FAMILY

## 44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm [TQFP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.80 BSC		
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X44)	X1			0.55
Contact Pad Length (X44)	Y1			1.50
Distance Between Pads	G	0.25		

**Notes:**

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076A

# PIC18F46J11 FAMILY

---

NOTES:

# PIC18F46J11 FAMILY

## APPENDIX A: REVISION HISTORY

### Revision A (October 2008)

Original data sheet for the PIC18F46J11 family of devices.

### Revision B (February 2009)

Changes to the Electrical Characteristics and minor edits throughout text.

### Revision C (October 2009)

Removed "Preliminary" marking.

## APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1,

**TABLE B-1: DEVICE DIFFERENCES BETWEEN PIC18F46J11 FAMILY MEMBERS**

Features	PIC18F24J11	PIC18F25J11	PIC18F26J11	PIC18F44J11	PIC18F45J11	PIC18F46J11
Program Memory	16K	32K	64K	16K	32K	64K
Program Memory (Instructions)	8,192	16,384	32,768	8,192	16,384	32,768
I/O Ports (Pins)	Ports A, B, C			Ports A, B, C, D, E		
10-Bit ADC Module	10 Input Channels			13 Input Channels		
Packages	28-Pin QFN, SOIC, SSOP and SPDIP (300 mil)			44-Pin QFN and TQFP		

# PIC18F46J11 FAMILY

---

NOTES:



# PIC18F46J11 FAMILY

## INDEX

### A

A/D	345	CTMU	373
A/D Converter Interrupt, Configuring	349	CTMU Current Source Calibration Circuit	376
Acquisition Requirements	350	CTMU Typical Connections and Internal Configuration for Pulse Delay Generation	384
ADCAL Bit	353	CTMU Typical Connections and Internal Configuration for Time Measurement	383
ADRESH Register	348	Demultiplexed Addressing Mode	180
Analog Port Pins, Configuring	351	Device Clock	32
Associated Registers	354	EUSART Transmit	331
Automatic Acquisition Time	351	EUSARTx Receive	333
Calibration	353	Fail-Safe Clock Monitor	404
Configuring the Module	349	Fully Multiplexed Addressing Mode	180
Conversion Clock (TAD)	351	Generic I/O Port Operation	125
Conversion Requirements	498	High/Low-Voltage Detect with External Input	368
Conversion Status (GO/DONE Bit)	348	Interrupt Logic	110
Conversions	352	LCD Control	188
Converter Characteristics	497	Legacy Parallel Slave Port	174
Operation in Power-Managed Modes	353	MSSPx (I <sup>2</sup> C Master Mode)	305
Special Event Trigger (ECCPx)	352	MSSPx (I <sup>2</sup> C Mode)	285
Use of the ECCP2 Trigger	352	MSSPx (SPI Mode)	266
Absolute Maximum Ratings	461	Multiplexed Addressing Application	187
AC (Timing) Characteristics	479	On-Chip Reset Circuit	57
Load Conditions for Device Timing Specifications	480	Parallel EEPROM (Up to 15-Bit Address, 16-Bit Data)	188
Parameter Symbology	479	Parallel EEPROM (Up to 15-Bit Address, 8-Bit Data)	188
Temperature and Voltage Specifications	480	Parallel Master/Slave Connection Addressed Buffer	177
Timing Conditions	480	Parallel Master/Slave Connection Buffered	176
ACKSTAT	310	Partially Multiplexed Addressing Application	187
ACKSTAT Status Flag	310	Partially Multiplexed Addressing Mode	180
ADCAL Bit	353	PIC18F2XJ11 (28-Pin)	14
ADCON0 Register		PIC18F4XJ11 (44-Pin)	15
GO/DONE Bit	348	PMP Module	165
ADDFSR	450	PWM (Enhanced)	249
ADDLW	413	PWM Operation (Simplified)	246
ADDULNK	450	Reads From Flash Program Memory	101
ADDWF	413	RTCC	221
ADDWFC	414	Simplified Steering	262
ADRESL Register	348	Single Comparator	358
Analog-to-Digital Converter. <i>See</i> A/D.		Table Read Operation	97
ANDLW	414	Table Write Operation	98
ANDWF	415	Table Writes to Flash Program Memory	103
Assembler		Timer0 in 16-Bit Mode	192
MPASM Assembler	458	Timer0 in 8-Bit Mode	192
Auto-Wake-up on Sync Break Character	334	Timer1	199
		Timer2	208
		Timer3	212
		Timer4	220
		Using the Open-Drain Output	126
		Watchdog Timer	399
		BN	416
		BNC	417
		BNN	417
		BNOV	418
		BNZ	418
		BOR. <i>See</i> Brown-out Reset.	
		BOV	421
		BRA	419
		Break Character (12-Bit) Transmit and Receive	336
		BRG. <i>See</i> Baud Rate Generator.	
		Brown-out Reset (BOR)	59
		and On-Chip Voltage Regulator	402
		Detecting	59

# PIC18F46J11 FAMILY

Disabling in Sleep Mode .....	59	Effects of a Reset .....	362
BSF .....	419	Enable and Input Selection .....	359
BTFSC .....	420	Enable and Output Selection .....	359
BTFSS .....	420	Interrupts .....	361
BTG .....	421	Operation .....	358
BZ .....	422	Operation During Sleep .....	362
<b>C</b>		Registers .....	355
C Compilers		Response Time .....	358
MPLAB C18 .....	458	Comparator Specifications .....	476
MPLAB C30 .....	458	Comparator Voltage Reference .....	363
Calibration (A/D Converter) .....	353	Accuracy and Error .....	365
CALL .....	422	Associated Registers .....	365
CALLW .....	451	Configuring .....	364
Capture (ECCP Module) .....	243	Connection Considerations .....	365
CCPRxH:CCPRxL Registers .....	243	Effects of a Reset .....	365
ECCP Pin Configuration .....	243	Operation During Sleep .....	365
Prescaler .....	244	Compare (ECCP Module) .....	245
Software Interrupt .....	243	CCPRx Register .....	245
Timer1/Timer3 Mode Selection .....	243	Pin Configuration .....	245
Clock Sources .....	36	Software Interrupt .....	245
Effects of Power-Managed Modes .....	40	Special Event Trigger .....	217, 245
Selecting the 31 kHz Source .....	37	Timer1/Timer3 Mode Selection .....	245
Selection Using OSCCON Register .....	37	Compare (ECCPx Module)	
CLRf .....	423	Special Event Trigger .....	352
CLRWDt .....	423	Computed GOTO .....	75
Code Examples		Configuration Bits .....	389
16 x 16 Signed Multiply Routine .....	108	Configuration Mismatch (CM) Reset .....	60
16 x 16 Unsigned Multiply Routine .....	108	Configuration Register Protection .....	405
512-Byte SPI Master Mode Init and Transfer .....	283	Configuration Registers	
8 x 8 Signed Multiply Routine .....	107	Bits and Device IDs .....	390
8 x 8 Unsigned Multiply Routine .....	107	Mapping Flash Configuration Words .....	390
A/D Calibration Routine .....	353	Core Features	
Calculating Baud Rate Error .....	326	Easy Migration .....	12
Capacitance Calibration Routine .....	380	Expanded Memory .....	11
Capacitive Touch Switch Routine .....	382	Extended Instruction Set .....	12
Changing Between Capture Prescalers .....	244	nanoWatt Technology .....	11
Communicating with the +5V System .....	126	Oscillator Options and Features .....	11
Computed GOTO Using an Offset Value .....	75	CPFSEQ .....	424
Configuring EUSART2 Input and Output Functions .....	148	CPFSGT .....	425
Current Calibration Routine .....	378	CPFSLT .....	425
Erasing Flash Program Memory .....	102	Crystal Oscillator/Ceramic Resonators .....	33
Fast Register Stack .....	75	CTMU	
How to Clear RAM (Bank 1) Using Indirect Addressing .....	90	Associated Registers .....	387
Initializing PORTA .....	129	Calibrating .....	375
Initializing PORTB .....	132	Creating a Delay with .....	384
Initializing PORTC .....	136	Effects of a Reset .....	384
Initializing PORTD .....	139	Initialization .....	375
Initializing PORTE .....	142	Measuring Capacitance with .....	381
Loading the SSP1BUF (SSP1SR) Register .....	269	Measuring Time with .....	383
Reading a Flash Program Memory Word .....	101	Operation .....	374
Saving STATUS, WREG and BSR Registers in RAM .....	124	Operation During Idle Mode .....	384
Setup for CTMU Calibration Routines .....	377	Operation During Sleep Mode .....	384
Single-Word Write to Flash Program Memory .....	105	CTMU Current Source Specifications .....	476
Two-Word Instructions .....	77	Customer Change Notification Service .....	525
Ultra Low-Power Wake-up Initialization .....	56	Customer Notification Service .....	525
Writing to Flash Program Memory .....	104	Customer Support .....	525
Code Protection .....	389	<b>D</b>	
COMF .....	424	Data Addressing Modes .....	90
Comparator .....	355	Comparing Addressing Modes with the Extended In-	
Analog Input Connection Considerations .....	358	struction Set Enabled .....	94
Associated Registers .....	362	Direct .....	90
Configuration .....	359	Indexed Literal Offset .....	93
Control .....	359	BSR .....	95
		Instructions Affected .....	93
		Mapping Access Bank .....	95

# PIC18F46J11 FAMILY

Indirect .....	90	Auto-Wake-up on Sync Break .....	334
Inherent and Literal .....	90	Receiver .....	333
Data Memory .....	78	Setting Up 9-Bit Mode with Address Detect ....	333
Access Bank .....	80	Transmitter .....	331
Extended Instruction Set .....	92	Baud Rate Generator	
General Purpose Registers .....	80	Operation in Power-Managed Mode .....	325
Memory Maps		Baud Rate Generator (BRG) .....	325
Access Bank Special Function Registers .....	81	Associated Registers .....	326
Non-Access Bank Special Function Registers ..	82	Auto-Baud Rate Detect .....	329
PIC18F46J11 Family Devices .....	79	Baud Rates, Asynchronous Modes .....	327
Special Function Registers .....	81	Formulas .....	325
Context Defined SFRs .....	83	High Baud Rate Select (BRGH Bit) .....	325
DAW .....	426	Sampling .....	325
DC Characteristics .....	474	Synchronous Master Mode .....	337
Power-Down and Supply Current .....	464	Associated Registers, Reception .....	340
Supply Voltage .....	463	Associated Registers, Transmission .....	338
DCFSNZ .....	427	Reception .....	339
DECF .....	426	Transmission .....	337
DECFSZ .....	427	Synchronous Slave Mode .....	341
Development Support .....	457	Associated Registers, Reception .....	343
Device Differences .....	511	Associated Registers, Transmission .....	342
Device Overview .....	11	Reception .....	343
Details on Individual Family Members .....	12	Transmission .....	341
Features (28-Pin Devices) .....	13	Extended Instruction Set	
Features (44-Pin Devices) .....	13	ADDFSR .....	450
Other Special Features .....	12	ADDULNK .....	450
Direct Addressing .....	91	CALLW .....	451
<b>E</b>		MOVSF .....	451
Effect on Standard PIC MCU Instructions .....	454	MOVSS .....	452
Electrical Characteristics .....	461	PUSHL .....	452
Absolute Maximum Ratings .....	461	SUBFSR .....	453
Enhanced Capture/Compare/PWM (ECCP) .....	241	SUBULNK .....	453
Associated Registers .....	263	Extended Instructions	
Capture Mode. <i>See</i> Capture.		Considerations when Enabling .....	454
Compare Mode. <i>See</i> Compare.		External Clock Input .....	34
ECCP Mode and Timer Resources .....	243	<b>F</b>	
Enhanced PWM Mode .....	249	Fail-Safe Clock Monitor .....	389, 403
Auto-Restart .....	258	Interrupts in Power-Managed Modes .....	405
Auto-Shutdown .....	257	POR or Wake-up From Sleep .....	405
Direction Change in Full-Bridge Output Mode ..	255	WDT During Oscillator Failure .....	404
Full-Bridge Application .....	253	Fast Register Stack .....	75
Full-Bridge Mode .....	253	Features Overview .....	3
Half-Bridge Application .....	252	Comparative Table .....	4
Half-Bridge Application Examples .....	259	Firmware Instructions .....	407
Half-Bridge Mode .....	252	Flash Program Memory .....	97
Output Relationships (Active-High) .....	250	Associated Registers .....	106
Output Relationships Diagram (Active-Low) ....	251	Control Registers .....	98
Programmable Dead-Band Delay .....	259	EECON1 and EECON2 .....	98
Shoot-Through Current .....	259	TABLAT (Table Latch) .....	100
Start-up Considerations .....	256	TBLPTR (Table Pointer) Register .....	100
Outputs and Configuration .....	243	Erase Sequence .....	102
Enhanced Universal Synchronous Asynchronous Receiver		Erasing .....	102
Transmitter (EUSART). <i>See</i> EUSART.		Operation During Code-Protect .....	106
Equations		Reading .....	101
A/D Acquisition Time .....	350	Table Pointer	
A/D Minimum Charging Time .....	350	Boundaries Based on Operation .....	100
Bytes Transmitted for a Given DMABC .....	281	Table Pointer Boundaries .....	100
Calculating the Minimum Required Acquisition Time ....	350	Table Reads and Table Writes .....	97
Errata .....	9	Write Sequence .....	103
EUSART .....	321	Write Sequence (Word Programming) .....	105
Asynchronous Mode .....	331	Writing .....	103
12-Bit Break Transmit and Receive .....	336	Unexpected Termination .....	106
Associated Registers, Reception .....	334	Write Verify .....	106
Associated Registers, Transmission .....	332	FSCM. <i>See</i> Fail-Safe Clock Monitor.	

# PIC18F46J11 FAMILY

<b>G</b>			
GOTO	428	INCF	428
<b>H</b>		INCFSZ	429
Hardware Multiplier	107	In-Circuit Debugger	406
8 x 8 Multiplication Algorithms	107	In-Circuit Serial Programming (ICSP)	389, 406
Operation	107	Indexed Literal Offset Addressing	
Performance Comparison (table)	107	and Standard PIC18 Instructions	454
High/Low-Voltage Detect	367	Indexed Literal Offset Mode	454
Applications	371	Indirect Addressing	91
Associated Registers	372	INFSNZ	429
Characteristics	478	Initialization Conditions	63–69
Current Consumption	369	Instruction Cycle	76
Effects of a Reset	372	Clocking Scheme	76
Operation	368	Flow/Pipelining	76
During Sleep	372	Instruction Set	407
Setup	369	ADDLW	413
Start-up Time	369	ADDWF	413
Typical Application	371	ADDWF (Indexed Literal Offset Mode)	455
		ADDWFC	414
<b>I</b>		ANDLW	414
I/O Ports	125	ANDWF	415
Open-Drain Outputs	126	BC	415
Pin Capabilities	125	BCF	416
TTL Input Buffer Option	126	BN	416
I <sup>2</sup> C Mode	285	BNC	417
I <sup>2</sup> C Mode (MSSP)		BNN	417
Acknowledge Sequence Timing	313	BNOV	418
Associated Registers	319	BNZ	418
Baud Rate Generator	306	BOV	421
Bus Collision		BRA	419
During a Repeated Start Condition	317	BSF	419
During a Stop Condition	318	BSF (Indexed Literal Offset Mode)	455
Clock Arbitration	308	BTFSC	420
Clock Stretching	300	BTFSS	420
10-Bit Slave Receive Mode (SEN = 1)	300	BTG	421
10-Bit Slave Transmit Mode	300	BZ	422
7-Bit Slave Receive Mode (SEN = 1)	300	CALL	422
7-Bit Slave Transmit Mode	300	CLRf	423
Clock Synchronization and CKP bit	301	CLRWDT	423
Effects of a Reset	314	COMF	424
General Call Address Support	304	CPFSEQ	424
I <sup>2</sup> C Clock Rate w/BRG	307	CPFSGT	425
Master Mode	304	CPFSLT	425
Operation	305	DAW	426
Reception	310	DCFSNZ	427
Repeated Start Condition Timing	309	DECf	426
Start Condition Timing	308	DECFSZ	427
Transmission	310	Extended Instructions	449
Multi-Master Communication, Bus Collision and Arbitration	314	Considerations when Enabling	454
Multi-Master Mode	314	Syntax	449
Operation	290	Use with MPLAB IDE Tools	456
Read/Write Bit Information (R/W Bit)	290, 293	General Format	409
Registers	285	GOTO	428
Serial Clock (SCLx Pin)	293	INCF	428
Slave Mode	290	INCFSZ	429
Addressing	290	INFSNZ	429
Addressing Masking Modes		IORLW	430
5-Bit	291	IORWF	430
7-Bit	292	LFSR	431
Reception	293	MOVF	431
Transmission	293	MOVFF	432
Sleep Operation	314	MOVLB	432
Stop Condition Timing	313	MOVLW	433
		MOVWF	433
		MULLW	434
		MULWF	434

# PIC18F46J11 FAMILY

NEGF .....	435
NOP .....	435
Opcode Field Descriptions .....	408
POP .....	436
PUSH .....	436
RCALL .....	437
RESET .....	437
RETFIE .....	438
RETLW .....	438
RETURN .....	439
RLCF .....	439
RLNCF .....	440
RRCF .....	440
RRNCF .....	441
SETF .....	441
SETF (Indexed Literal Offset Mode) .....	455
SLEEP .....	442
Standard Instructions .....	407
SUBFWB .....	442
SUBLW .....	443
SUBWF .....	443
SUBWFB .....	444
SWAPF .....	444
TBLRD .....	445
TBLWT .....	446
TSTFSZ .....	447
XORLW .....	447
XORWF .....	448
INTCON .....	111
INTCON Registers .....	111
Inter-Integrated Circuit. <i>See</i> I <sup>2</sup> C.	
Internal Oscillator	
Frequency Drift. <i>See</i> INTOSC Frequency Drift.	
Internal Oscillator Block .....	34
Adjustment .....	35
OSCTUNE Register .....	35
Internal RC Oscillator	
Use with WDT .....	399
Internal Voltage Reference Specifications .....	477
Internet Address .....	525
Interrupt Sources .....	389
A/D Conversion Complete .....	349
Capture Complete (ECCP) .....	243
Compare Complete (ECCP) .....	245
Interrupt-on-Change (RB7:RB4) .....	132
TMR0 Overflow .....	193
TMR1 Overflow .....	201
TMR3 Overflow .....	209, 217
TMR4 to PR4 Match .....	220
TMR4 to PR4 Match (PWM) .....	219
Interrupts .....	109
Control Bits .....	109
Control Registers. <i>See</i> INTCON Registers.	
During, Context Saving .....	124
INTx Pin .....	124
PORTB, Interrupt-on-Change .....	124
RCON Register .....	123
TMR0 .....	124
Interrupts, Flag Bits	
Interrupt-on-Change (RB7:RB4) Flag (RBIF Bit) .....	132
INTOSC Frequency Drift .....	35
INTOSC, INTRC. <i>See</i> Internal Oscillator Block.	
IORLW .....	430
IORWF .....	430

## L

LFSR .....	431
Low-Power Modes .....	41
Clock Transitions and Status Indicators .....	42
Deep Sleep Mode .....	48
and RTCC Peripheral .....	50
Brown-out Reset (DSBOR) .....	50
Preparing for .....	49
Registers .....	52
Typical Sequence .....	51
Wake-up Sources .....	50
Watchdog Timer (DSWDT) .....	50
Exiting Idle and Sleep Modes .....	48
By Interrupt .....	48
By Reset .....	48
By WDT Time-out .....	48
Without an Oscillator Start-up Delay .....	48
Idle Modes .....	46
PRI_IDLE .....	46
RC_IDLE .....	48
SEC_IDLE .....	46
Multiple Sleep Commands .....	42
Run Modes .....	42
PRI_RUN .....	42
RC_RUN .....	44
SEC_RUN .....	42
Sleep Mode .....	45
Summary (table) .....	42
Ultra Low-Power Wake-up .....	55

## M

Master Clear ( $\overline{\text{MCLR}}$ ) .....	59
Master Synchronous Serial Port (MSSP). <i>See</i> MSSP.	
Memory Organization .....	71
Data Memory .....	78
Program Memory .....	71
Return Address Stack .....	73
Memory Programming Requirements .....	476
Microchip Internet Web Site .....	525
MOVF .....	431
MOVFF .....	432
MOVLB .....	432
MOVLW .....	433
MOVSF .....	451
MOVSS .....	452
MOVWF .....	433
MPLAB ASM30 Assembler, Linker, Librarian .....	458
MPLAB ICD 2 In-Circuit Debugger .....	459
MPLAB ICE 2000 High-Performance Universal In-Circuit Emulator .....	459
MPLAB Integrated Development Environment Software .....	457
MPLAB PM3 Device Programmer .....	459
MPLAB REAL ICE In-Circuit Emulator System .....	459
MPLINK Object Linker/MPLIB Object Librarian .....	458
MSSP	
ACK Pulse .....	290, 293
I <sup>2</sup> C Mode. <i>See</i> I <sup>2</sup> C Mode.	
Module Overview .....	265
SPI Master/Slave Connection .....	270
TMR4 Output for Clock Shift .....	220
MULLW .....	434
MULWF .....	434

## N

NEGF .....	435
------------	-----

# PIC18F46J11 FAMILY

NOP .....	435	RC5/SDO1/RP16 .....	20, 26
<b>O</b>		RC6/PMA5/TX1/CK1/RP17 .....	27
Oscillator Configuration		RC6/TX1/CK1/RP17 .....	20
Internal Oscillator Block .....	34	RC7/PMA4/RX1/DT1/RP18 .....	27
Oscillator Control .....	31	RC7/RX1/DT1/RP18 .....	20
Oscillator Modes .....	32	RD0/PMD0/SCL2 .....	28
Oscillator Types .....	31	RD1/PMD1/SDA2 .....	28
Oscillator Configurations .....	31	RD2/PMD2/RP19 .....	28
Oscillator Selection .....	389	RD3/PMD3/RP20 .....	28
Oscillator Start-up Timer (OST) .....	40	RD4/PMD4/RP21 .....	28
Oscillator Switching .....	36	RD5/PMD5/RP22 .....	28
Oscillator Transitions .....	37	RD6/PMD6/RP23 .....	28
Oscillator, Timer1 .....	195, 200, 213	RD7/PMD7/RP24 .....	28
Oscillator, Timer3 .....	209	RE0/AN5/PMRD .....	29
<b>P</b>		RE1/AN6/PMWR .....	29
P1A/P1B/P1C/P1D. See Enhanced Capture/Compare/PWM (ECCP). .....	249	RE2/AN7/PMCS .....	29
Packaging		VDD .....	21
Details .....	501	VDD1 .....	29
Marking .....	499	VDD2 .....	29
Parallel Master Port (PMP) .....	165	VDDCORE/VCAP .....	21, 29
Application Examples .....	187	VSS1 .....	21, 29
Associated Registers .....	189	VSS2 .....	21, 29
Data Registers .....	172	Pinout I/O Descriptions	
Master Port Modes .....	179	PIC18F2XJ11 (28-Pin) .....	16
Module Registers .....	166	PIC18F4XJ11 (44-Pin) .....	22
Slave Port Modes .....	174	PLL Frequency Multiplier .....	34
Peripheral Pin Select (PPS) .....	144	POP .....	436
PICSTART Plus Development Programmer .....	460	POR. See Power-on Reset.	
Pin Diagrams .....	5–7	PORTA	
Pin Functions		Additional Pin Functions	
AVDD1 .....	29	Ultra Low-Power Wake-up .....	55
AVDD2 .....	29	Associated Registers .....	131
AVSS1 .....	29	LATA Register .....	129
MCLR .....	16, 22	PORTA Register .....	129
OSC1/CLKI/RA7 .....	16, 22	TRISA Register .....	129
OSC2/CLKO/RA6 .....	16, 22	PORTB	
RA0/AN0/C1INA/ULPWU/PMA6/RP0 .....	23	Associated Registers .....	135
RA0/AN0/C1INA/ULPWU/RP0 .....	17	LATB Register .....	132
RA1/AN1/C2INA/PMA7/RP1 .....	23	PORTB Register .....	132
RA1/AN1/C2INA/RP1 .....	17	RB7:RB4 Interrupt-on-Change Flag (RBIF Bit) .....	132
RA2/AN2/VREF-/CVREF/C2INB .....	17, 23	TRISB Register .....	132
RA3/AN3/VREF+/C1INB .....	17, 23	PORTC	
RA5/AN4/SS1/HLVDIN/RP2 .....	17, 23	Associated Registers .....	138
RA6 .....	17, 23	LATC Register .....	136
RA7 .....	17, 23	PORTC Register .....	136
RB0/AN12/INT0/RP3 .....	18, 24	TRISC Register .....	136
RB1/AN10/PMBE/RTCC/RP4 .....	24	PORTD	
RB1/AN10/RTCC/RP4 .....	18	Associated Registers .....	141
RB2/AN8/CTEDG1/PMA3/REFO/RP5 .....	24	LATD Register .....	139
RB2/AN8/CTEDG1/REFO/RP5 .....	18	PORTD Register .....	139
RB3/AN9/CTEDG2/PMA2/RP6 .....	24	TRISD Register .....	139
RB3/AN9/CTEDG2/RP6 .....	18	PORTE	
RB4/KBI0/RP7 .....	18	Associated Registers .....	143
RB4/PMA1/KBI0/RP7 .....	25	LATE Register .....	142
RB5/KBI1/RP8 .....	18	PORTE Register .....	142
RB5/PMA0/KBI1/RP8 .....	25	TRISE Register .....	142
RB6/KBI2/PGC/RP9 .....	19, 25	Power-Managed Modes	
RB7/KBI3/PGD/RP10 .....	19, 25	and EUSART Operation .....	325
RC0/T1OSO/T1CKI/RP11 .....	20, 26	and PWM Operation .....	263
RC1/T1OSI/RP12 .....	20, 26	and SPI Operation .....	274
RC2/AN11/CTPLS/RP13 .....	20, 26	Clock Sources .....	41
RC3/SCK1/SCL1/RP14 .....	20, 26	Entering .....	41
RC4/SDI1/SDA1/RP15 .....	20, 26	Selecting .....	41
		Power-on Reset (POR) .....	59
		Power-up Delays .....	40

# PIC18F46J11 FAMILY

Power-up Timer (PWRT) .....	40, 60	ALRMCFG (Alarm Configuration) .....	225
Time-out Sequence .....	60	ALRMDAY (Alarm Day Value) .....	230
Prescaler, Timer0 .....	193	ALRMMHR (Alarm Hours Value) .....	231
Prescaler, Timer2 (Timer4) .....	247	ALRMMIN (Alarm Minutes Value) .....	232
PRI_IDLE Mode .....	46	ALRMMNTH (Alarm Month Value) .....	230
PRI_RUN Mode .....	42	ALRMRPT (Alarm Calibration) .....	226
Product Identification System .....	527	ALRMSEC (Alarm Seconds Value) .....	232
Program Counter .....	73	ALRMWD (Alarm Weekday Value) .....	231
PCL, PCH and PCU Registers .....	73	ANCON0 (A/D Port Configuration 2) .....	347
PCLATH and PCLATU Registers .....	73	ANCON1 (A/D Port Configuration 1) .....	347
Program Memory		Associated with Comparator .....	355
ALU Status .....	89	Associated with Watchdog Timer .....	400
Extended Instruction Set .....	92	BAUDCONx (Baud Rate Control) .....	324
Flash Configuration Words .....	72	CCPxCON (Enhanced Capture/Compare/PWM x Control) .....	242
Hard Memory Vectors .....	72	CMSTAT (Comparator Status) .....	357
Instructions .....	77	CMxCON (Comparator Control x) .....	356
Two-Word .....	77	CONFIG1H (Configuration 1 High) .....	392
Interrupt Vector .....	72	CONFIG1L (Configuration 1 Low) .....	391
Look-up Tables .....	75	CONFIG2H (Configuration 2 High) .....	394
Memory Maps .....	71	CONFIG2L (Configuration 2 Low) .....	393
Hard Vectors and Configuration Words .....	72	CONFIG3H (Configuration 3 High) .....	396
Reset Vector .....	72	CONFIG3L (Configuration 3 Low) .....	395
Program Verification and Code Protection .....	405	CONFIG4H (Configuration 4 High) .....	397
Programming, Device Instructions .....	407	CONFIG4L (Configuration 4 Low) .....	396
Pulse Steering .....	260	CTMUCONH (CTMU Control High) .....	385
PUSH .....	436	CTMUCONL (CTMU Control Low) .....	386
PUSH and POP Instructions .....	74	CTMUICON (CTMU Current Control) .....	387
PUSHL .....	452	CVRCON (Comparator Voltage Reference Control) .....	364
PWM (CCP Module)		DAY (Day Value) .....	228
Associated Registers .....	248	DEVID1 (Device ID 1) .....	397
Duty Cycle .....	246	DEVID2 (Device ID 2) .....	398
Example Frequencies/Resolutions .....	247	DMACON1 (DMA Control 1) .....	278
Operation Setup .....	247	DMACON2 (DMA Control 2) .....	279
Period .....	246	DSCONH (Deep Sleep Control High Byte) .....	52
PR2/PR4 Registers .....	246	DSCONL (Deep Sleep Control Low Byte) .....	52
TMR2 (TMR4) to PR2 (PR4) Match .....	246	DSGPR0 (Deep Sleep Persistent General Purpose 0) .....	53
PWM (ECCP Module)		DSGPR1 (Deep Sleep Persistent General Purpose 1) .....	53
Effects of a Reset .....	263	DSWAKEH (Deep Sleep Wake High Byte) .....	54
Operation in Power-Managed Modes .....	263	DSWAKEL (Deep Sleep Wake Low Byte) .....	54
Operation with Fail-Safe Clock Monitor .....	263	ECCPxAS (ECCPx Auto-Shutdown Control) .....	257
Pulse Steering .....	260	ECCPxDEL (Enhanced PWM Control) .....	260
Steering Synchronization .....	262	EECON1 (EEPROM Control 1) .....	99
TMR4 to PR4 Match .....	219	File Summary .....	83
PWM Mode. See Enhanced Capture/Compare/PWM .....	249	HLVDCON (High/Low-Voltage Detect Control) .....	367
<b>Q</b>		HOURS (Hours Value) .....	229
Q Clock .....	247	I <sup>2</sup> C Mode (MSSP) .....	285
<b>R</b>		INTCON (Interrupt Control) .....	111
RAM. See Data Memory.		INTCON2 (Interrupt Control 2) .....	112
RBIF Bit .....	132	INTCON3 (Interrupt Control 3) .....	113
RC_IDLE Mode .....	48	IPR1 (Peripheral Interrupt Priority 1) .....	120
RC_RUN Mode .....	44	IPR2 (Peripheral Interrupt Priority 2) .....	121
RCALL .....	437	IPR3 (Peripheral Interrupt Priority 3) .....	122
RCON Register		MINUTES (Minutes Value) .....	229
Bit Status During Initialization .....	62	MONTH (Month Value) .....	227
Reader Response .....	526	ODCON1 (Peripheral Open-Drain Control 1) .....	127
Real-Time Clock and Calendar (RTCC) .....	221	ODCON2 (Peripheral Open-Drain Control 2) .....	127
Operation .....	233	ODCON3 (Peripheral Open-Drain Control 3) .....	128
Registers .....	222	OSCCON (Oscillator Control) .....	38
Reference Clock Output .....	39	OSCTUNE (Oscillator Tuning) .....	36
Register File .....	80	PADCFG1 (Pad Configuration Control 1) .....	128
Register File Summary .....	83–88	PADCFG1 (Pad Configuration) .....	224
Registers		Parallel Master Port .....	166
ADCON0 (A/D Control 0) .....	345		
ADCON1 (A/D Control 1) .....	346		

# PIC18F46J11 FAMILY

PIE1 (Peripheral Interrupt Enable 1) .....	117	SPI Mode (MSSP) .....	267
PIE2 (Peripheral Interrupt Enable 2) .....	118	SSPxCON1 (MSSPx Control 1, I <sup>2</sup> C Mode) .....	287
PIE3 (Peripheral Interrupt Enable 3) .....	119	SSPxCON1 (MSSPx Control 1, SPI Mode) .....	268
PIR1 (Peripheral Interrupt Request (Flag) 1) .....	114	SSPxCON2 (MSSPx Control 2, I <sup>2</sup> C Master Mode) ..	288
PIR2 (Peripheral Interrupt Request (Flag) 2) .....	115	SSPxCON2 (MSSPx Control 2, I <sup>2</sup> C Slave Mode) ....	289
PIR3 (Peripheral Interrupt Request (Flag) 3) .....	116	SSPxMSK (I <sup>2</sup> C Slave Address Mask) .....	289
PMADDRH (Parallel Port Address High Byte) .....	173	SSPxSTAT (MSSPx Status, I <sup>2</sup> C Mode) .....	286
PMADDRL (Parallel Port Address Low Byte) .....	173	SSPxSTAT (MSSPx Status, SPI Mode) .....	267
PMCONH (Parallel Port Control High Byte) .....	166	STATUS .....	89
PMCONL (Parallel Port Control Low Byte) .....	167	STKPTR (Stack Pointer) .....	74
PMEH (Parallel Port Enable High Byte) .....	170	T0CON (Timer0 Control) .....	191
PMEL (Parallel Port Enable Low Byte) .....	170	T1CON (Timer1 Control) .....	195
PMMODEH (Parallel Port Mode High Byte) .....	168	T1GCON (Timer1 Gate Control) .....	196
PMMODEL (Parallel Port Mode Low Byte) .....	169	T2CON (Timer2 Control) .....	207
PMSTATH (Parallel Port Status High Byte) .....	171	T3CON (Timer3 Control) .....	209
PMSTATL (Parallel Port Status Low Byte) .....	171	T3GCON (Timer3 Gate Control) .....	210
PPSCON (Peripheral Pin Select Input 0) .....	149	T4CON (Timer4 Control) .....	219
PSTRxCON (Pulse Steering Control) .....	261	TCLKCON (Timer Clock Control) .....	197, 211
RCON (Reset Control) .....	58, 123	TXSTAx (Transmit Status and Control) .....	322
RCSTAx (Receive Status and Control) .....	323	WDTCON (Watchdog Timer Control) .....	400
REF0CON (Reference Oscillator Control) .....	39	WKDY (Weekday Value) .....	228
Reserved .....	227	YEAR (Year Value) .....	227
RPINR1 (Peripheral Pin Select Input 1) .....	150	RESET .....	437
RPINR12 (Peripheral Pin Select Input 12) .....	152	Reset .....	57
RPINR13 (Peripheral Pin Select Input 13) .....	152	Brown-out Reset .....	59
RPINR16 (Peripheral Pin Select Input 16) .....	153	Brown-out Reset (BOR) .....	57
RPINR17 (Peripheral Pin Select Input 17) .....	153	Configuration Mismatch (CM) .....	57
RPINR2 (Peripheral Pin Select Input 2) .....	150	Configuration Mismatch Reset .....	60
RPINR21 (Peripheral Pin Select Input 21) .....	153	Deep Sleep .....	57
RPINR22 (Peripheral Pin Select Input 22) .....	154	Fast Register Stack .....	75
RPINR23 (Peripheral Pin Select Input 23) .....	154	MCLR .....	59
RPINR24 (Peripheral Pin Select Input 24) .....	154	MCLR Reset, During Power-Managed Modes .....	57
RPINR3 (Peripheral Pin Select Input 3) .....	150	MCLR Reset, Normal Operation .....	57
RPINR4 (Peripheral Pin Select Input 4) .....	151	Power-on Reset .....	59
RPINR6 (Peripheral Pin Select Input 6) .....	151	Power-on Reset (POR) .....	57
RPINR7 (Peripheral Pin Select Input 7) .....	151	Power-up Timer .....	60
RPINR8 (Peripheral Pin Select Input 8) .....	152	RESET Instruction .....	57
RPOR0 (Peripheral Pin Select Output 0) .....	155	Stack Full .....	57
RPOR1 (Peripheral Pin Select Output 1) .....	155	Stack Underflow .....	57
RPOR10 (Peripheral Pin Select Output 10) .....	158	State of Registers .....	62
RPOR11 (Peripheral Pin Select Output 11) .....	158	Watchdog Timer (WDT) Reset .....	57
RPOR12 (Peripheral Pin Select Output 12) .....	159	Resets .....	389
RPOR13 (Peripheral Pin Select Output 13) .....	159	Brown-out Reset (BOR) .....	389
RPOR14 (Peripheral Pin Select Output 14) .....	159	Oscillator Start-up Timer (OST) .....	389
RPOR15 (Peripheral Pin Select Output 15) .....	160	Power-on Reset (POR) .....	389
RPOR16 (Peripheral Pin Select Output 16) .....	160	Power-up Timer (PWRT) .....	389
RPOR17 (Peripheral Pin Select Output 17) .....	160	RETFIE .....	438
RPOR18 (Peripheral Pin Select Output 18) .....	161	RETLW .....	438
RPOR19 (Peripheral Pin Select Output 19) .....	161	RETURN .....	439
RPOR2 (Peripheral Pin Select Output 2) .....	155	Return Address Stack .....	73
RPOR20 (Peripheral Pin Select Output 20) .....	161	Associated Registers .....	73
RPOR21 (Peripheral Pin Select Output 21) .....	162	Revision History .....	511
RPOR22 (Peripheral Pin Select Output 22) .....	162	RLCF .....	439
RPOR23 (Peripheral Pin Select Output 23) .....	162	RLNCF .....	440
RPOR24 (Peripheral Pin Select Output 24) .....	163	RRCF .....	440
RPOR3 (Peripheral Pin Select Output 3) .....	156	RRNCF .....	441
RPOR4 (Peripheral Pin Select Output 4) .....	156	RTCC	
RPOR5 (Peripheral Pin Select Output 5) .....	156	Alarm .....	237
RPOR6 (Peripheral Pin Select Output 6) .....	157	Configuring .....	237
RPOR7 (Peripheral Pin Select Output 7) .....	157	Interrupt .....	238
RPOR8 (Peripheral Pin Select Output 8) .....	157	Mask Settings .....	237
RPOR9 (Peripheral Pin Select Output 9) .....	158	Alarm Value Registers (ALRMVAL) .....	230
RTCCAL (RTCC Calibration) .....	224	Control Registers .....	223
RTCCFG (RTCC Configuration) .....	223	Low-Power Modes .....	238
SECONDS (Seconds Value) .....	229	Operation .....	



# PIC18F46J11 FAMILY

ALRMVAL Register Mapping .....	236	SUBLW .....	443
Calibration .....	236	SUBULNK .....	453
Clock Source .....	234	SUBWF .....	443
Digit Carry Rules .....	234	SUBWFB .....	444
General Functionality .....	235	SWAPF .....	444
Leap Year .....	235		
Register Mapping .....	235	<b>T</b>	
RTCVAL Register Mapping .....	236	Table Pointer Operations with TBLRD, TBLWT (table) ...	100
Safety Window for Register Reads and Writes .....	235	Table Reads/Table Writes .....	75
Write Lock .....	235	TAD .....	351
Peripheral Module Disable (PMD) Register .....	238	TBLRD .....	445
Register Interface .....	233	TBLWT .....	446
Register Maps .....	239	Timer0 .....	191
Reset .....	238	Associated Registers .....	193
Device .....	238	Operation .....	192
Power-on Reset (POR) .....	238	Overflow Interrupt .....	193
Value Registers (RTCVAL) .....	227	Prescaler .....	193
RTCCEN Bit Write .....	233	Switching Assignment .....	193
		Prescaler Assignment (PSA Bit) .....	193
<b>S</b>		Prescaler Select (T0PS2:T0PS0 Bits) .....	193
SCKx .....	266	Reads and Writes in 16-Bit Mode .....	192
SDIx .....	266	Source Edge Select (T0SE Bit) .....	192
SDOx .....	266	Source Select (T0CS Bit) .....	192
SEC_IDLE Mode .....	46	Timer1 .....	195
SEC_RUN Mode .....	42	16-Bit Read/Write Mode .....	200
Serial Clock, SCKx .....	266	Associated Registers .....	206
Serial Data In (SDIx) .....	266	Clock Source Selection .....	198
Serial Data Out (SDOx) .....	266	Gate .....	202
Serial Peripheral Interface. <i>See</i> SPI Mode.		Interrupt .....	201
SETF .....	441	Operation .....	198
Shoot-Through Current .....	259	Oscillator .....	195, 200
Slave Select (SSx) .....	266	Layout Considerations .....	201
SLEEP .....	442	Resetting, Using the ECCP Special Event Trigger ..	202
Software Simulator (MPLAB SIM) .....	458	TMR1H Register .....	195
Special Event Trigger. <i>See</i> Compare (ECCP Mode).		TMR1L Register .....	195
Special Features of the CPU .....	389	Use as a Clock Source .....	201
SPI Mode (MSSP) .....	266	Timer2 .....	207
Associated Registers .....	275	Associated Registers .....	208
Bus Mode Compatibility .....	274	Interrupt .....	208
Clock Speed, Interactions .....	274	Operation .....	207
Effects of a Reset .....	274	Output .....	208
Enabling SPI I/O .....	270	Timer3 .....	209
Master Mode .....	271	16-Bit Read/Write Mode .....	213
Master/Slave Connection .....	270	Associated Registers .....	217
Operation .....	269	Gate .....	213
Open-Drain Output Option .....	269	Operation .....	212
Operation in Power-Managed Modes .....	274	Oscillator .....	209, 213
Registers .....	267	Overflow Interrupt .....	209, 217
Serial Clock .....	266	Special Event Trigger (ECCP) .....	217
Serial Data In .....	266	TMR3H Register .....	209
Serial Data Out .....	266	TMR3L Register .....	209
Slave Mode .....	272	Timer4 .....	219
Slave Select .....	266	Associated Registers .....	220
Slave Select Synchronization .....	272	Interrupt .....	220
SPI Clock .....	271	MSSP Clock Shift .....	220
SSPxBUF Register .....	271	Operation .....	219
SSPxSR Register .....	271	Output .....	220
Typical Connection .....	270	Postscaler. <i>See</i> Postscaler, Timer4.	
SSPOV .....	310	PR4 Register .....	219
SSPOV Status Flag .....	310	Prescaler. <i>See</i> Prescaler, Timer4.	
SSPxSTAT Register		TMR4 Register .....	219
R/W Bit .....	290, 293	TMR4 to PR4 Match Interrupt .....	219, 220
SSx .....	266	Timing Diagrams	
Stack Full/Underflow Resets .....	75	A/D Conversion .....	497
SUBFSR .....	453	Asynchronous Reception .....	334
SUBWFB .....	442	Asynchronous Transmission .....	332

# PIC18F46J11 FAMILY

Asynchronous Transmission (Back-to-Back) .....	332	PWM Output (Active-High) .....	250
Automatic Baud Rate Calculation .....	330	PWM Output (Active-Low) .....	251
Auto-Wake-up Bit (WUE) During Normal Operation .....	335	Read and Write, 8-Bit Data, Demultiplexed Address .....	182
Auto-Wake-up Bit (WUE) During Sleep .....	335	Read, 16-Bit Data, Demultiplexed Address .....	185
Baud Rate Generator with Clock Arbitration .....	308	Read, 16-Bit Multiplexed Data, Fully Multiplexed 16-Bit Address .....	186
BRG Overflow Sequence .....	330	Read, 16-Bit Multiplexed Data, Partially Multiplexed Address .....	185
BRG Reset Due to SDAX Arbitration During Start Condition .....	316	Read, 8-Bit Data, Fully Multiplexed 16-Bit Address .....	184
Bus Collision During a Repeated Start Condition (Case 1) .....	317	Read, 8-Bit Data, Partially Multiplexed Address .....	182
Bus Collision During a Repeated Start Condition (Case 2) .....	317	Read, 8-Bit Data, Partially Multiplexed Address, Enable Strobe .....	183
Bus Collision During a Start Condition (SCLx = 0) .....	316	Read, 8-Bit Data, Wait States Enabled, Partially Multiplexed Address .....	182
Bus Collision During a Stop Condition (Case 1) .....	318	Repeated Start Condition .....	309
Bus Collision During a Stop Condition (Case 2) .....	318	Reset, Watchdog Timer (WDT), Oscillator Start-up Timer (OST) and Power-up Timer (PWRT) .....	483
Bus Collision During Start Condition (SDAX Only) .....	315	Send Break Character Sequence .....	336
Bus Collision for Transmit and Acknowledge .....	314	Slave Synchronization .....	272
CLKO and I/O .....	482	Slow Rise Time (MCLR Tied to VDD, VDD Rise > TPWRT) .....	61
Clock Synchronization .....	301	SPI Mode (Master Mode) .....	271
Clock/Instruction Cycle .....	76	SPI Mode (Slave Mode, CKE = 0) .....	273
Enhanced Capture/Compare/PWM .....	485	SPI Mode (Slave Mode, CKE = 1) .....	273
EUSARTx Synchronous Receive (Master/Slave) .....	496	Steering Event at Beginning of Instruction (STRSYNC = 1) .....	262
EUSARTx Synchronous Transmission (Master/Slave) .....	496	Steering Event at End of Instruction (STRSYNC = 0) .....	262
Example SPI Master Mode (CKE = 0) .....	488	Synchronous Reception (Master Mode, SREN) .....	339
Example SPI Master Mode (CKE = 1) .....	489	Synchronous Transmission .....	337
Example SPI Slave Mode (CKE = 0) .....	490	Synchronous Transmission (Through TXEN) .....	338
Example SPI Slave Mode (CKE = 1) .....	491	Time-out Sequence on Power-up (MCLR Not Tied to VDD), Case 1 .....	61
External Clock .....	480	Time-out Sequence on Power-up (MCLR Not Tied to VDD), Case 2 .....	61
Fail-Safe Clock Monitor .....	404	Time-out Sequence on Power-up (MCLR Tied to VDD, VDD Rise < TPWRT) .....	60
First Start Bit .....	308	Timer Pulse Generation .....	238
Full-Bridge PWM Output .....	254	Timer0 and Timer1 External Clock .....	484
Half-Bridge PWM Output .....	252, 259	Timer1 Gate Count Enable Mode .....	203
High/Low-Voltage Detect Characteristics .....	478	Timer1 Gate Single Pulse Mode .....	205
High-Voltage Detect (VDIRMAG = 1) .....	371	Timer1 Gate Single Pulse/Toggle Combined Mode .....	206
I <sup>2</sup> C Bus Data .....	492	Timer1 Gate Toggle Mode .....	204
I <sup>2</sup> C Acknowledge Sequence .....	313	Timer3 Gate Count Enable Mode .....	213
I <sup>2</sup> C Bus Start/Stop Bits .....	492	Timer3 Gate Single Pulse Mode .....	215
I <sup>2</sup> C Master Mode (7 or 10-Bit Transmission) .....	311	Timer3 Gate Single Pulse/Toggle Combined Mode .....	216
I <sup>2</sup> C Master Mode (7-Bit Reception) .....	312	Timer3 Gate Toggle Mode .....	214
I <sup>2</sup> C Slave Mode (10-Bit Reception, SEN = 0, ADMSK = 01001) .....	297	Transition for Entry to Idle Mode .....	47
I <sup>2</sup> C Slave Mode (10-Bit Reception, SEN = 0) .....	298	Transition for Entry to SEC_RUN Mode .....	43
I <sup>2</sup> C Slave Mode (10-Bit Reception, SEN = 1) .....	303	Transition for Entry to Sleep Mode .....	45
I <sup>2</sup> C Slave Mode (10-Bit Transmission) .....	299	Transition for Two-Speed Start-up (INTRC to HSPLL) .....	403
I <sup>2</sup> C Slave Mode (7-Bit Reception, SEN = 0, ADMSK = 01011) .....	295	Transition for Wake From Idle to Run Mode .....	47
I <sup>2</sup> C Slave Mode (7-Bit Reception, SEN = 0) .....	294	Transition for Wake From Sleep (HSPLL) .....	45
I <sup>2</sup> C Slave Mode (7-Bit Reception, SEN = 1) .....	302	Transition From RC_RUN Mode to PRI_RUN Mode .....	44
I <sup>2</sup> C Slave Mode (7-Bit Transmission) .....	296	Transition From SEC_RUN Mode to PRI_RUN Mode (HSPLL) .....	43
I <sup>2</sup> C Slave Mode General Call Address Sequence (7 or 10-Bit Addressing Mode) .....	304	Transition to RC_RUN Mode .....	44
I <sup>2</sup> C Stop Condition Receive or Transmit Mode .....	313	Write, 16-Bit Data, Demultiplexed Address .....	185
Low-Voltage Detect (VDIRMAG = 0) .....	370	Write, 16-Bit Multiplexed Data, Fully Multiplexed 16-Bit Address .....	186
MSSPx I <sup>2</sup> C Bus Data .....	494	Write, 16-Bit Multiplexed Data, Partially Multiplexed Address .....	186
MSSPx I <sup>2</sup> C Bus Start/Stop Bits .....	494	Write, 8-Bit Data, Fully Multiplexed 16-Bit Address .....	184
Parallel Master Port Read .....	486	Write, 8-Bit Data, Partially Multiplexed Address .....	183
Parallel Master Port Write .....	487		
Parallel Slave Port Read .....	175, 177		
Parallel Slave Port Write .....	175, 178		
PWM Auto-Shutdown with Auto-Restart Enabled .....	258		
PWM Auto-Shutdown with Firmware Restart .....	258		
PWM Direction Change .....	255		
PWM Direction Change at Near 100% Duty Cycle .....	256		
PWM Output .....	246		

Write, 8-Bit Data, Partially Multiplexed Address, Enable Strobe .....	184
Write, 8-Bit Data, Wait States Enabled, Partially Multiplexed Address .....	183
Timing Diagrams and Specifications	
AC Characteristics	
Internal RC Accuracy .....	481
CLKO and I/O Requirements .....	482
Enhanced Capture/Compare/PWM Requirements ..	485
EUSARTx Synchronous Receive Requirements .....	496
EUSARTx Synchronous Transmission Requirements ...	496
Example SPI Mode Requirements (Master Mode, CKE = 0) .....	488
Example SPI Mode Requirements (Master Mode, CKE = 1) .....	489
Example SPI Mode Requirements (Slave Mode, CKE = 0) .....	490
Example SPI Slave Mode Requirements (CKE = 1) ..	491
External Clock Requirements .....	481
I <sup>2</sup> C Bus Data Requirements (Slave Mode) .....	493
I <sup>2</sup> C Bus Start/Stop Bits Requirements (Slave Mode) .....	492
MSSPx I <sup>2</sup> C Bus Data Requirements .....	495
MSSPx I <sup>2</sup> C Bus Start/Stop Bits Requirements .....	494
Parallel Master Port Read Requirements .....	486
Parallel Master Port Write Requirements .....	487
PLL Clock .....	481
Reset, Watchdog Timer, Oscillator Start-up Timer, Power-up Timer and Brown-out Reset Requirements ..	483
Timer0 and Timer1 External Clock Requirements ...	484
TSTFSZ .....	447
Two-Speed Start-up .....	389, 403
Two-Word Instructions	
Example Cases .....	77
TXSTAx Register	
BRGH Bit .....	325
<b>U</b>	
Ultra Low-Power Wake-up .....	55
<b>V</b>	
Voltage Reference Specifications .....	477
Voltage Regulator (On-Chip) .....	401
Operation in Sleep Mode .....	402
<b>W</b>	
Watchdog Timer (WDT) .....	389, 399
Associated Registers .....	400
Control Register .....	399
During Oscillator Failure .....	404
Programming Considerations .....	399
WCOL .....	308, 309, 310, 313
WCOL Status Flag .....	308, 309, 310, 313
WWW Address .....	525
WWW, On-Line Support .....	9
<b>X</b>	
XORLW .....	447
XORWF .....	448

# PIC18F46J11 FAMILY

---

NOTES:

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at [www.microchip.com](http://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](http://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>**

# PIC18F46J11 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  
From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: PIC18F46J11 Family Literature Number: DS39932C

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?  
\_\_\_\_\_  
\_\_\_\_\_

# PIC18F46J11 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 <sup>(1)</sup>	PIC18F24J11 PIC18F25J11 PIC18F26J11 PIC18F44J11 PIC18F45J11 PIC18F46J11 PIC18LF24J11 PIC18LF25J11 PIC18LF26J11 PIC18LF44J11 PIC18LF45J11 PIC18LF46J11		
Temperature Range	I = -40°C to +85°C (Industrial)		
Package	SP = Skinny PDIP SS = SSOP SO = SOIC ML = QFN PT = TQFP (Thin Quad Flatpack)		
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)		

**Examples:**

a) PIC18F46J11-I/PT 301 = Industrial temp., TQFP package, QTP pattern #301.

b) PIC18F46J11T-I/PT = Tape and reel, Industrial temp., TQFP package.

**Note 1:** F = Standard Voltage Range  
LF = Extended Voltage Range

**2:** T = In tape and reel



---

---

## 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](http://www.microchip.com)

**Atlanta**  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

**Boston**  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Cleveland**  
Independence, OH  
Tel: 216-447-0464  
Fax: 216-447-0643

**Dallas**  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
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

**Santa Clara**  
Santa Clara, CA  
Tel: 408-961-6444  
Fax: 408-961-6445

**Toronto**  
Mississauga, Ontario,  
Canada  
Tel: 905-673-0699  
Fax: 905-673-6509

### ASIA/PACIFIC

**Asia Pacific Office**  
Suites 3707-14, 37th Floor  
Tower 6, The Gateway  
Harbour City, Kowloon  
Hong Kong  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**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: 86-28-8665-5511  
Fax: 86-28-8665-7889

**China - Hong Kong SAR**  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**China - Nanjing**  
Tel: 86-25-8473-2460  
Fax: 86-25-8473-2470

**China - Qingdao**  
Tel: 86-532-8502-7355  
Fax: 86-532-8502-7205

**China - Shanghai**  
Tel: 86-21-5407-5533  
Fax: 86-21-5407-5066

**China - Shenyang**  
Tel: 86-24-2334-2829  
Fax: 86-24-2334-2393

**China - Shenzhen**  
Tel: 86-755-8203-2660  
Fax: 86-755-8203-1760

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xiamen**  
Tel: 86-592-2388138  
Fax: 86-592-2388130

**China - Xian**  
Tel: 86-29-8833-7252  
Fax: 86-29-8833-7256

**China - Zhuhai**  
Tel: 86-756-3210040  
Fax: 86-756-3210049

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444  
Fax: 91-80-3090-4080

**India - New Delhi**  
Tel: 91-11-4160-8631  
Fax: 91-11-4160-8632

**India - Pune**  
Tel: 91-20-2566-1512  
Fax: 91-20-2566-1513

**Japan - Yokohama**  
Tel: 81-45-471- 6166  
Fax: 81-45-471-6122

**Korea - Daegu**  
Tel: 82-53-744-4301  
Fax: 82-53-744-4302

**Korea - Seoul**  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932 or  
82-2-558-5934

**Malaysia - Kuala Lumpur**  
Tel: 60-3-6201-9857  
Fax: 60-3-6201-9859

**Malaysia - Penang**  
Tel: 60-4-227-8870  
Fax: 60-4-227-4068

**Philippines - Manila**  
Tel: 63-2-634-9065  
Fax: 63-2-634-9069

**Singapore**  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
Tel: 886-3-6578-300  
Fax: 886-3-6578-370

**Taiwan - Kaohsiung**  
Tel: 886-7-536-4818  
Fax: 886-7-536-4803

**Taiwan - Taipei**  
Tel: 886-2-2500-6610  
Fax: 886-2-2508-0102

**Thailand - Bangkok**  
Tel: 66-2-694-1351  
Fax: 66-2-694-1350

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Munich**  
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

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**UK - Wokingham**  
Tel: 44-118-921-5869  
Fax: 44-118-921-5820

03/26/09