



# **PIC18F6390/6490/8390/8490**

## **Data Sheet**

64/80-Pin Flash Microcontrollers  
with LCD Driver and nanoWatt Technology

---

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

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

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip 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, Accuron, dsPIC, KEELOQ, KEELOQ logo, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, rPIC and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rLAB, Select Mode, Smart Serial, SmartTel, Total Endurance, UNI/O, 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.

© 2007, 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

# PIC18F6390/6490/8390/8490

## 64/80-Pin Flash Microcontrollers with LCD Driver and nanoWatt Technology

### LCD Driver Module Features:

- Direct Driving of LCD Panel
- Up to 48 Segments: Software Selectable
- Programmable LCD Timing module:
  - Multiple LCD timing sources available
  - Up to 4 commons: Static, 1/2, 1/3 or 1/4 multiplex
  - Static, 1/2 or 1/3 bias configuration
- Can drive LCD Panel while in Sleep mode

### Power-Managed Modes:

- Run: CPU On, Peripherals On
- Idle: CPU Off, Peripherals On
- Sleep: CPU Off, Peripherals Off
- Run mode Currents Down to 14.0  $\mu$ A Typical
- Idle mode Currents Down to 5.8  $\mu$ A Typical
- Sleep Current Down to 0.1  $\mu$ A Typical
- Timer1 Oscillator: 1.8  $\mu$ A, 32 kHz, 2V
- Watchdog Timer: 2.1  $\mu$ A
- Two-Speed Oscillator Start-up

### Flexible Oscillator Structure:

- Four Crystal modes:
  - LP: up to 200 kHz
  - XT: up to 4 MHz
  - HS: up to 40 MHz
  - HSPLL: 4-10 MHz (16-40 MHz internal)
- 4x Phase Lock Loop (available for crystal and internal oscillators)
- Two External RC modes, up to 4 MHz
- Two External Clock modes, up to 40 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies, from 31 kHz to 8 MHz
  - Provides a complete range of clock speeds from 31 kHz to 32 MHz when used with PLL
  - User-tunable to compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shut down of device if primary or secondary clock fails

### Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA
- Four External Interrupts
- Four Input Change Interrupts
- Four 8-Bit/16-Bit Timer/Counter modules
- Real-Time Clock (RTC) Software module:
  - Configurable 24-hour clock, calendar, automatic 100-year or 12800-year, day-of-week calculator
  - Uses Timer1
- Up to 2 Capture/Compare/PWM (CCP) modules
- Master Synchronous Serial Port (MSSP) module supporting 3-Wire SPI (all 4 modes) and I<sup>2</sup>C™ Master and Slave modes
- Addressable USART module:
  - Supports RS-485 and RS-232
- Enhanced Addressable USART module:
  - Supports RS-485, RS-232 and LIN 1.2
  - Auto-wake-up on Start bit
  - Auto-Baud Detect
- 10-Bit, up to 12-Channel Analog-to-Digital (A/D) Converter module:
  - Auto-acquisition capability
  - Conversion available during Sleep
- Dual Analog Comparators with Input Multiplexing

### Special Microcontroller Features:

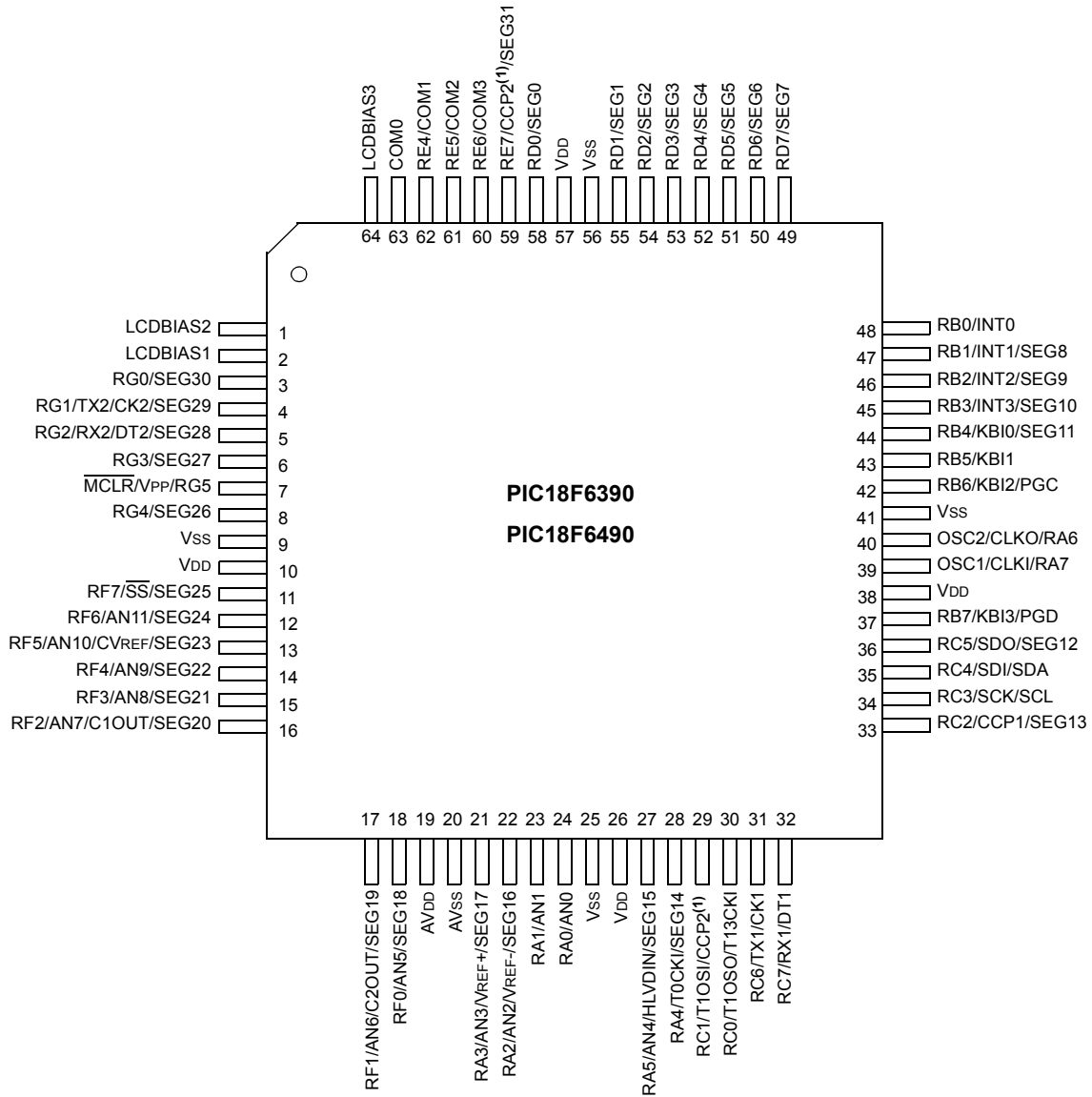
- C Compiler Optimized Architecture:
  - Optional extended instruction set designed to optimize re-entrant code
- 1000 Erase/Write Cycle Flash Program Memory Typical
- Flash Retention: 100 Years Typical
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 132s
  - 2% stability over VDD and temperature
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) via Two Pins
- Wide Operating Voltage Range: 2.0V to 5.5V

Device	Program Memory		Data Memory	I/O	LCD (pixel)	10-Bit A/D (ch)	CCP (PWM)	MSSP		EUSART/AUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)					SPI	Master I <sup>2</sup> C™			
PIC18F6390	8K	4096	768	50	128	12	2	Y	Y	1/1	2	1/3
PIC18F6490	16K	8192	768	50	128	12	2	Y	Y	1/1	2	1/3
PIC18F8390	8K	4096	768	66	192	12	2	Y	Y	1/1	2	1/3
PIC18F8490	16K	8192	768	66	192	12	2	Y	Y	1/1	2	1/3

# PIC18F6390/6490/8390/8490

## Pin Diagrams

### 64-Pin TQFP

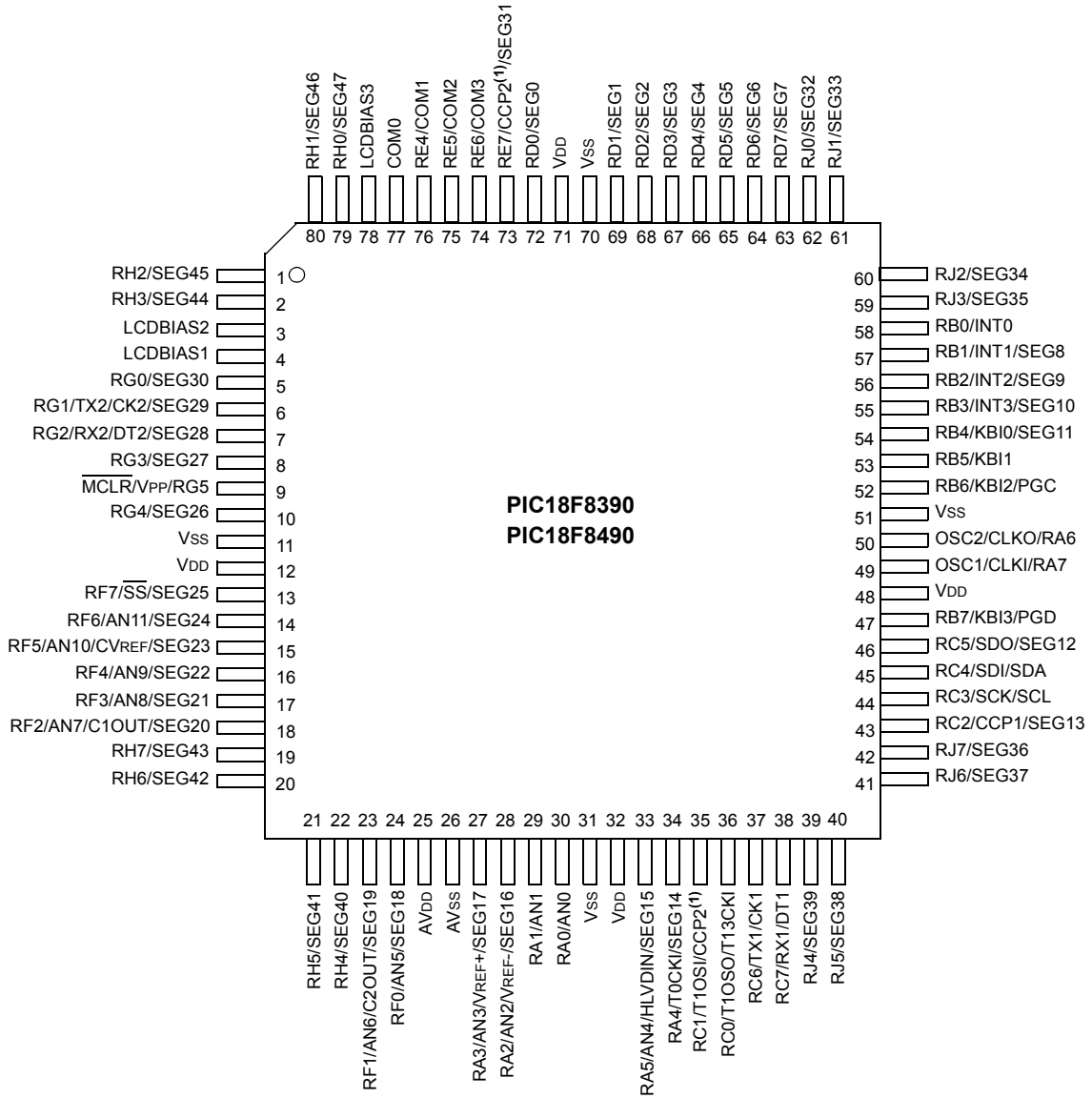


**Note 1:** RE7 is the alternate pin for CCP2 multiplexing.

# PIC18F6390/6490/8390/8490

## Pin Diagrams (Continued)

### 80-Pin TQFP



**Note 1:** RE7 is the alternate pin for CCP2 multiplexing.

# PIC18F6390/6490/8390/8490

---

## Table of Contents

1.0	Device Overview .....	7
2.0	Oscillator Configurations .....	31
3.0	Power-Managed Modes .....	41
4.0	Reset .....	51
5.0	Memory Organization .....	65
6.0	Flash Program Memory .....	87
7.0	8 x 8 Hardware Multiplier .....	91
8.0	Interrupts .....	93
9.0	I/O Ports .....	109
10.0	Timer0 Module .....	131
11.0	Timer1 Module .....	135
12.0	Timer2 Module .....	141
13.0	Timer3 Module .....	143
14.0	Capture/Compare/PWM (CCP) Modules .....	147
15.0	Master Synchronous Serial Port (MSSP) Module .....	157
16.0	Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) .....	197
17.0	Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART) .....	217
18.0	10-Bit Analog-to-Digital Converter (A/D) Module .....	231
19.0	Comparator Module .....	241
20.0	Comparator Voltage Reference Module .....	247
21.0	High/Low-Voltage Detect (HLVD) .....	251
22.0	Liquid Crystal Display (LCD) Driver Module .....	257
23.0	Special Features of the CPU .....	281
24.0	Instruction Set Summary .....	295
25.0	Development Support .....	345
26.0	Electrical Characteristics .....	349
27.0	DC and AC Characteristics Graphs and Tables .....	387
28.0	Packaging Information .....	389
	Appendix A: Revision History .....	395
	Appendix B: Device Differences .....	395
	Appendix C: Conversion Considerations .....	396
	Appendix D: Migration from Baseline to Enhanced Devices .....	396
	Appendix E: migration from Mid-Range to Enhanced Devices .....	397
	Appendix F: Migration from High-End to Enhanced Devices .....	397
	Index .....	399
	The Microchip Web Site .....	409
	Customer Change Notification Service .....	409
	Customer Support .....	409
	Reader Response .....	410
	PIC18F6390/6490/8390/8490 Product Identification System .....	411

## TO OUR VALUED CUSTOMERS

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

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

### Most Current Data Sheet

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

<http://www.microchip.com>

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS3000A is version A of document DS30000).

### Errata

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

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

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

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

### Customer Notification System

Register on our web site at [www.microchip.com/cn](http://www.microchip.com/cn) to receive the most current information on all of our products.

# PIC18F6390/6490/8390/8490

---

NOTES:



# PIC18F6390/6490/8390/8490

## 1.0 DEVICE OVERVIEW

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

- PIC18F6390
- PIC18F8390
- PIC18F6490
- PIC18F8490

This family offers the advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price. In addition to these features, the PIC18F6390/6490/8390/8490 family introduces design enhancements that make these microcontrollers a logical choice for many high-performance, power-sensitive applications.

## 1.1 New Core Features

### 1.1.1 nanoWatt TECHNOLOGY

All of the devices in the PIC18F6390/6490/8390/8490 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal oscillator block, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled, but the peripherals still active. In these states, power consumption can be reduced even further – to as little as 4% of normal operation requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.
- **Lower Consumption in Key Modules:** The power requirements for both Timer1 and the Watchdog Timer have been reduced by up to 80%, with typical values of 1.1  $\mu\text{A}$  and 2.1  $\mu\text{A}$ , respectively.

### 1.1.2 MULTIPLE OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F6390/6490/8390/8490 family offer ten different oscillator options, allowing users a wide range of choices in developing application hardware. These include:

- Four Crystal modes using crystals or ceramic resonators.
- Two External Clock modes offering the option of using two pins (oscillator input and a divide-by-4 clock output) or one pin (oscillator input, with the second pin reassigned as general I/O).
- Two External RC Oscillator modes with the same pin options as the External Clock modes.
- An internal oscillator block which provides an 8 MHz clock ( $\pm 2\%$  accuracy) 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 the two oscillator pins for use as additional general purpose I/O.
- A Phase Lock Loop (PLL) frequency multiplier, available to both the High-Speed Crystal and Internal Oscillator modes, which allows clock speeds of up to 40 MHz. Used with the internal oscillator, the PLL gives users a complete selection of clock speeds from 31 kHz to 32 MHz – all without using an external crystal or clock circuit.

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

- **Fail-Safe Clock Monitor:** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator block, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up:** This option allows the internal oscillator to serve as the clock source from Power-on Reset or wake-up from Sleep mode until the primary clock source is available.

# PIC18F6390/6490/8390/8490

---

## 1.2 Other Special Features

- **Memory Endurance:** The Flash cells for program memory are rated to last for approximately a thousand erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 100 years.
- **Extended Instruction Set:** The PIC18F6390/6490/8390/8490 family introduces an optional extension to the PIC18 instruction set, which adds 8 new instructions and an Indexed Addressing mode. This extension, enabled as a device configuration option, has been specifically designed to optimize re-entrant application code originally developed in high-level languages such as C.
- **Enhanced Addressable USART:** This serial communication module is capable of standard RS-232 operation and provides support for the LIN bus protocol. Other enhancements include Automatic Baud Rate Detection and a 16-bit Baud Rate Generator for improved resolution. When the microcontroller is using the internal oscillator block, the EUSART provides stable operation for applications that talk to the outside world, without using an external crystal (or its accompanying power requirement).
- **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, reduces code overhead.
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing a time-out range from 4 ms to over 10 minutes that is stable across operating voltage and temperature.

## 1.3 Details on Individual Family Members

Devices in the PIC18F6390/6490/8390/8490 family are available in 64-pin (PIC18F6X90) and 80-pin (PIC18F8X90) packages. Block diagrams for the two groups are shown in Figure 1-1 and Figure 1-2, respectively.

The devices are differentiated from each other in three ways:

1. I/O Ports: 7 bidirectional ports on 64-pin devices; 9 bidirectional ports on 80-pin devices.
2. LCD Pixels: 128 (32 SEGs x 4 COMs) pixels can be driven by 64-pin devices; 192 (48 SEGs x 4 COMs) pixels can be driven by 80-pin devices.
3. Flash Program Memory: 8 Kbytes for PIC18FX390 devices; 16 Kbytes for PIC18FX490.

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

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

Like all Microchip PIC18 devices, members of the PIC18F6390/6490/8390/8490 family are available as both standard and low-voltage devices. Standard devices with Flash memory, designated with an "F" in the part number (such as PIC18F6390), accommodate an operating  $V_{DD}$  range of 4.2V to 5.5V. Low-voltage parts, designated by "LF" (such as PIC18LF6490), function over an extended  $V_{DD}$  range of 2.0V to 5.5V.

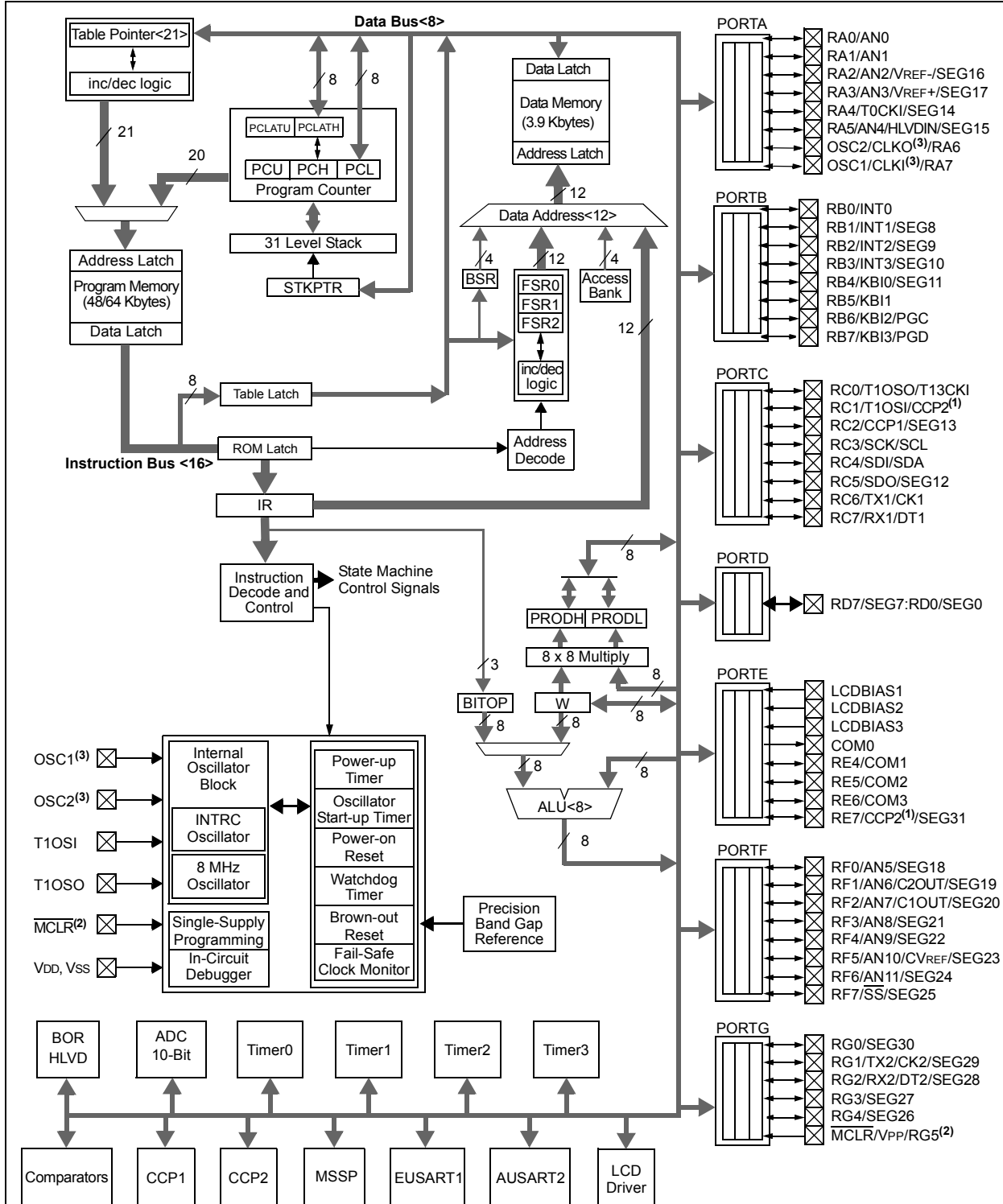
# PIC18F6390/6490/8390/8490

**TABLE 1-1: DEVICE FEATURES**

Features	PIC18F6390	PIC18F6490	PIC18F8390	PIC18F8490
Operating Frequency	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz
Program Memory (Bytes)	8K	16K	8K	16K
Program Memory (Instructions)	4096	8192	4096	8192
Data Memory (Bytes)	768	768	768	768
Interrupt Sources	22	22	22	22
I/O Ports	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
Number of Pixels the LCD Driver can Drive	128 (32 SEGs x 4 COMs)	128 (32 SEGs x 4 COMs)	192 (48 SEGs x 4 COMs)	192 (48 SEGs x 4 COMs)
Timers	4	4	4	4
Capture/Compare/PWM Modules	2	2	2	2
Serial Communications	MSSP, AUSART Enhanced USART	MSSP, AUSART Enhanced USART	MSSP, AUSART Enhanced USART	MSSP, AUSART Enhanced USART
10-Bit Analog-to-Digital Module	12 Input Channels	12 Input Channels	12 Input Channels	12 Input Channels
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set Enabled	75 Instructions; 83 with Extended Instruction Set Enabled	75 Instructions; 83 with Extended Instruction Set Enabled	75 Instructions; 83 with Extended Instruction Set Enabled
Packages	64-Pin TQFP	64-Pin TQFP	80-Pin TQFP	80-Pin TQFP

# PIC18F6390/6490/8390/8490

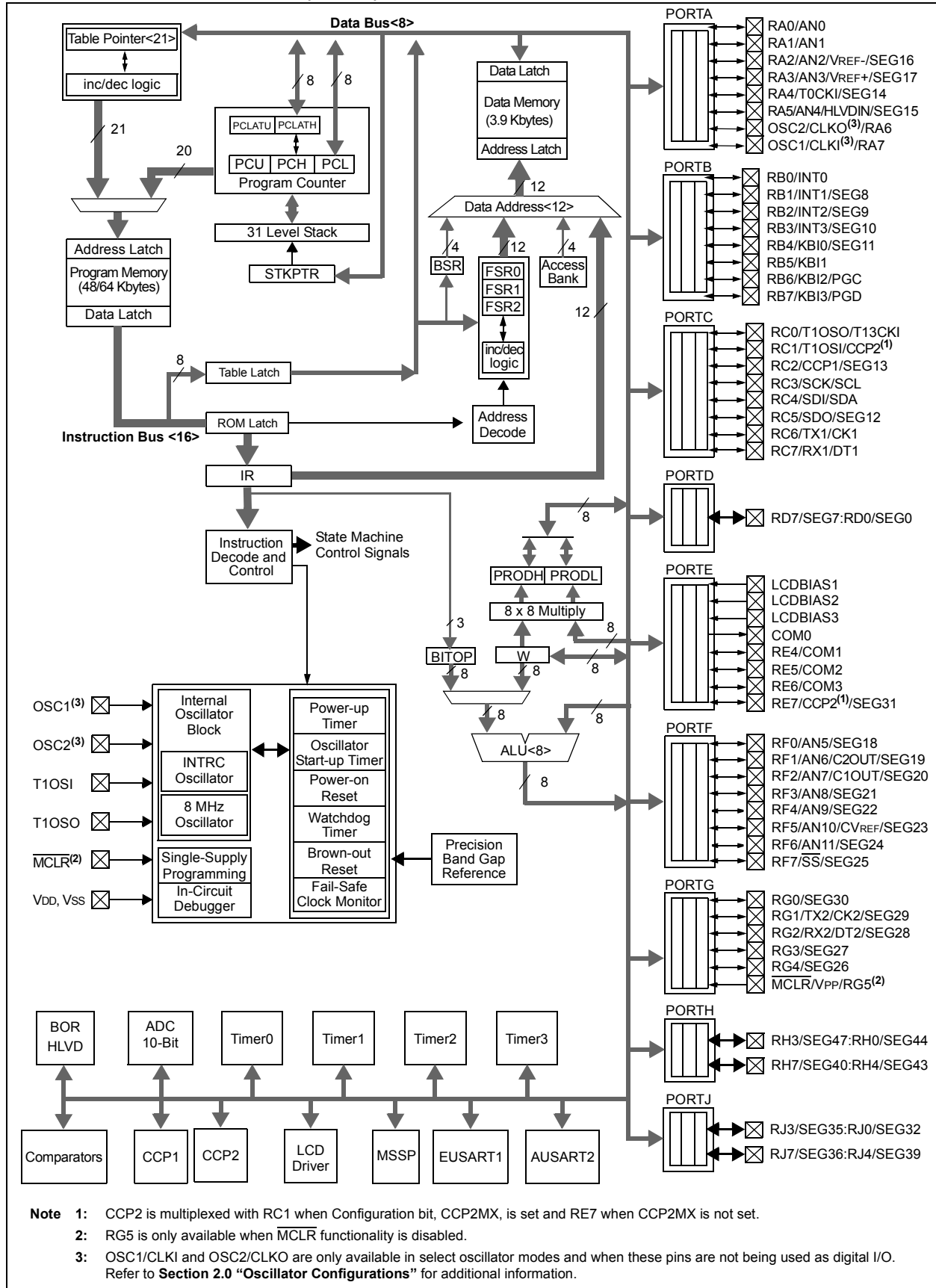
FIGURE 1-1: PIC18F6X90 (64-PIN) BLOCK DIAGRAM



- Note** 1: CCP2 is multiplexed with RC1 when Configuration bit, CCP2MX, is set, or RE7 when CCP2MX is not set.  
 2: RG5 is only available when MCLR functionality is disabled.  
 3: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to Section 2.0 "Oscillator Configurations" for additional information.

# PIC18F6390/6490/8390/8490

FIGURE 1-2: PIC18F8X90 (80-PIN) BLOCK DIAGRAM



# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
MCLR/VPP/RG5 MCLR  VPP RG5	7	I  P I	ST  ST	Master Clear (input) or programming voltage (input). Master Clear (Reset) input. This pin is an active-low Reset to the device. Programming voltage input. Digital input.
OSC1/CLKI/RA7 OSC1  CLKI  RA7	39	I  I I/O	ST  CMOS TTL	Oscillator crystal or external clock input. Oscillator crystal input or external clock source input. ST buffer when configured in RC mode, CMOS otherwise. External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.) General purpose I/O pin.
OSC2/CLKO/RA6 OSC2  CLKO  RA6	40	O  O I/O	—  — TTL	Oscillator crystal or clock output. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. General purpose I/O pin.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RA0/AN0 RA0 AN0	24	I/O I	TTL Analog	PORTA is a bidirectional I/O port.  Digital I/O. Analog input 0.
RA1/AN1 RA1 AN1	23	I/O I	TTL Analog	Digital I/O. Analog input 1.
RA2/AN2/VREF-/SEG16 RA2 AN2 VREF- SEG16	22	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 2. A/D reference voltage (Low) input. SEG16 output for LCD.
RA3/AN3/VREF+/SEG17 RA3 AN3 VREF+ SEG17	21	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 3. A/D reference voltage (High) input. SEG17 output for LCD.
RA4/T0CKI/SEG14 RA4 T0CKI SEG14	28	I/O I O	ST/OD ST Analog	Digital I/O. Open-drain when configured as output. Timer0 external clock input. SEG14 output for LCD.
RA5/AN4/HLVDIN/SEG15 RA5 AN4 HLVDIN SEG15	27	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 4. Low-Voltage Detect input. SEG15 output for LCD.
RA6				See the OSC2/CLKO/RA6 pin.
RA7				See the OSC1/CLKI/RA7 pin.

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

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RB0/INT0 RB0 INT0	48	I/O I	TTL ST	PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs. Digital I/O. External interrupt 0.
RB1/INT1/SEG8 RB1 INT1 SEG8	47	I/O I O	TTL ST Analog	Digital I/O. External interrupt 1. SEG8 output for LCD.
RB2/INT2/SEG9 RB2 INT2 SEG9	46	I/O I O	TTL ST Analog	Digital I/O. External interrupt 2. SEG9 output for LCD.
RB3/INT3/SEG10 RB3 INT3 SEG10	45	I/O I O	TTL ST Analog	Digital I/O. External interrupt 3. SEG10 output for LCD.
RB4/KBI0/SEG11 RB4 KBI0 SEG11	44	I/O I O	TTL TTL Analog	Digital I/O. Interrupt-on-change pin. SEG11 output for LCD.
RB5/KBI1 RB5 KBI1	43	I/O I	TTL TTL	Digital I/O. Interrupt-on-change pin.
RB6/KBI2/PGC RB6 KBI2 PGC	42	I/O I I/O	TTL TTL ST	Digital I/O. Interrupt-on-change pin. In-Circuit Debugger and ICSP™ programming clock pin.
RB7/KBI3/PGD RB7 KBI3 PGD	37	I/O I I/O	TTL TTL ST	Digital I/O. Interrupt-on-change pin. In-Circuit Debugger and ICSP programming data pin.

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

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



# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RC0/T1OSO/T13CKI RC0 T1OSO T13CKI	30	I/O O I	ST — ST	PORTC is a bidirectional I/O port.  Digital I/O. Timer1 oscillator output. Timer1/Timer3 external clock input.
RC1/T1OSI/CCP2 RC1 T1OSI CCP2 <sup>(1)</sup>	29	I/O I I/O	ST CMOS ST	Digital I/O. Timer1 oscillator input. Capture 2 input/Compare 2 output/PWM2 output.
RC2/CCP1/SEG13 RC2 CCP1 SEG13	33	I/O I/O O	ST ST Analog	Digital I/O. Capture 1 input/Compare 1 output/PWM1 output. SEG13 output for LCD.
RC3/SCK/SCL RC3 SCK SCL	34	I/O I/O I/O	ST ST ST	Digital I/O. Synchronous serial clock input/output for SPI mode. Synchronous serial clock input/output for I <sup>2</sup> C™ mode.
RC4/SDI/SDA RC4 SDI SDA	35	I/O I I/O	ST ST ST	Digital I/O. SPI data in. I <sup>2</sup> C data I/O.
RC5/SDO/SEG12 RC5 SDO SEG12	36	I/O O O	ST — Analog	Digital I/O. SPI data out. SEG12 output for LCD.
RC6/TX1/CK1 RC6 TX1 CK1	31	I/O O I/O	ST — ST	Digital I/O. EUSART1 asynchronous transmit. EUSART1 synchronous clock (see related RX1/DT1).
RC7/RX1/DT1 RC7 RX1 DT1	32	I/O I I/O	ST ST ST	Digital I/O. EUSART1 asynchronous receive. EUSART1 synchronous data (see related TX1/CK1).

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RD0/SEG0 RD0 SEG0	58	I/O O	ST Analog	PORTD is a bidirectional I/O port.  Digital I/O. SEG0 output for LCD.
RD1/SEG1 RD1 SEG1	55	I/O O	ST Analog	Digital I/O. SEG1 output for LCD.
RD2/SEG2 RD2 SEG2	54	I/O O	ST Analog	Digital I/O. SEG2 output for LCD.
RD3/SEG3 RD3 SEG3	53	I/O O	ST Analog	Digital I/O. SEG3 output for LCD.
RD4/SEG4 RD4 SEG4	52	I/O O	ST Analog	Digital I/O. SEG4 output for LCD.
RD5/SEG5 RD5 SEG5	51	I/O O	ST Analog	Digital I/O. SEG5 output for LCD.
RD6/SEG6 RD6 SEG6	50	I/O O	ST Analog	Digital I/O. SEG6 output for LCD.
RD7/SEG7 RD7 SEG7	49	I/O O	ST Analog	Digital I/O. SEG7 output for LCD.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
LCDBIAS1 LCDBIAS1	2	I	Analog	PORTC is a bidirectional I/O port. BIAS1 input for LCD.
LCDBIAS2 LCDBIAS2	1	I	Analog	BIAS2 input for LCD.
LCDBIAS3 LCDBIAS3	64	I	Analog	BIAS3 input for LCD.
COM0 COM0	63	O	Analog	COM0 output for LCD.
RE4/COM1 RE4 COM1	62	I/O O	ST Analog	Digital I/O. COM1 output for LCD.
RE5/COM2 RE5 COM2	61	I/O O	ST Analog	Digital I/O. COM2 output for LCD.
RE6/COM3 RE6 COM3	60	I/O O	ST Analog	Digital I/O. COM3 output for LCD.
RE7/CCP2/SEG31 RE7 CCP2 <sup>(2)</sup> SEG31	59	I/O I/O O	ST ST Analog	Digital I/O. Capture 2 input/Compare 2 output/PWM2 output. SEG31 output for LCD.

**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)

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RF0/AN5/SEG18	18	I/O	ST	PORTF is a bidirectional I/O port. Digital I/O. Analog input 5. SEG18 output for LCD.
RF0		I	Analog	
AN5		O	Analog	
SEG18		O	Analog	
RF1/AN6/C2OUT/SEG19	17	I/O	ST	Digital I/O. Analog input 6. Comparator 2 output. SEG19 output for LCD.
RF1		I	Analog	
AN6		O	—	
C2OUT		O	Analog	
SEG19		O	Analog	
RF2/AN7/C1OUT/SEG20	16	I/O	ST	Digital I/O. Analog input 7. Comparator 1 output. SEG20 output for LCD.
RF2		I	Analog	
AN7		O	—	
C1OUT		O	Analog	
SEG20		O	Analog	
RF3/AN8/SEG21	15	I/O	ST	Digital I/O. Analog input 8. SEG21 output for LCD.
RF3		I	Analog	
AN8		O	Analog	
SEG21		O	Analog	
RF4/AN9/SEG22	14	I/O	ST	Digital I/O. Analog input 9. SEG22 output for LCD.
RF4		I	Analog	
AN9		O	Analog	
SEG22		O	Analog	
RF5/AN10/CVREF/SEG23	13	I/O	ST	Digital I/O. Analog input 10. Comparator reference voltage output. SEG23 output for LCD.
RF5		I	Analog	
AN10		O	Analog	
CVREF		O	Analog	
SEG23		O	Analog	
RF6/AN11/SEG24	12	I/O	ST	Digital I/O. Analog input 11. SEG24 output for LCD.
RF6		I	Analog	
AN11		O	Analog	
SEG24		O	Analog	
RF7/ $\overline{SS}$ /SEG25	11	I/O	ST	Digital I/O. SPI slave select input. SEG25 output for LCD.
RF7		I	TTL	
$\overline{SS}$		O	Analog	
SEG25		O	Analog	

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-2: PIC18F6X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
PORTG is a bidirectional I/O port.				
RG0/SEG30 RG0 SEG30	3	I/O O	ST Analog	Digital I/O. SEG30 output for LCD.
RG1/TX2/CK2/SEG29 RG1 TX2 CK2 SEG29	4	I/O O I/O O	ST — ST Analog	Digital I/O. AUSART2 asynchronous transmit. AUSART2 synchronous clock (see related RX2/DT2). SEG29 output for LCD.
RG2/RX2/DT2/SEG28 RG2 RX2 DT2 SEG28	5	I/O I I/O O	ST ST ST Analog	Digital I/O. AUSART2 asynchronous receive. AUSART2 synchronous data (see related TX2/CK2). SEG28 output for LCD.
RG3/SEG27 RG3 SEG27	6	I/O O	ST Analog	Digital I/O. SEG27 output for LCD.
RG4/SEG26 RG4 SEG26	8	I/O O	ST Analog	Digital I/O. SEG26 output for LCD.
RG5				See $\overline{MCLR}/VPP/RG5$ pin.
VSS	9, 25, 41, 56	P	—	Ground reference for logic and I/O pins.
VDD	10, 26, 38, 57	P	—	Positive supply for logic and I/O pins.
AVSS	20	P	—	Ground reference for analog modules.
AVDD	19	P	—	Positive supply for analog modules.

**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)

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
MCLR/VPP/PG5 MCLR	9	I	ST	Master Clear (input) or programming voltage (input). Master Clear (Reset) input. This pin is an active-low Reset to the device.
VPP		P		Programming voltage input.
RG5		I	ST	Digital input.
OSC1/CLKI/RA7 OSC1	49	I	ST	Oscillator crystal or external clock input. Oscillator crystal input or external clock source input. ST buffer when configured in RC mode, CMOS otherwise.
CLKI		I	CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.)
RA7		I/O	TTL	General purpose I/O pin.
OSC2/CLKO/RA6 OSC2	50	O	—	Oscillator crystal or clock output. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6		I/O	TTL	General purpose I/O pin.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RA0/AN0 RA0 AN0	30	I/O I	TTL Analog	PORTA is a bidirectional I/O port.  Digital I/O. Analog input 0.
RA1/AN1 RA1 AN1	29	I/O I	TTL Analog	Digital I/O. Analog input 1.
RA2/AN2/VREF-/SEG16 RA2 AN2 VREF- SEG16	28	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 2. A/D reference voltage (Low) input. SEG16 output for LCD.
RA3/AN3/VREF+/SEG17 RA3 AN3 VREF+ SEG17	27	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 3. A/D reference voltage (High) input. SEG17 output for LCD.
RA4/T0CKI/SEG14 RA4 T0CKI SEG14	34	I/O I O	ST/OD ST Analog	Digital I/O. Open-drain when configured as output. Timer0 external clock input. SEG14 output for LCD.
RA5/AN4/HLVDIN/SEG15 RA5 AN4 HLVDIN SEG15	33	I/O I I O	TTL Analog Analog Analog	Digital I/O. Analog input 4. Low-Voltage Detect input. SEG15 output for LCD.
RA6				See the OSC2/CLKO/RA6 pin.
RA7				See the OSC1/CLKI/RA7 pin.

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

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RB0/INT0 RB0 INT0	58	I/O I	TTL ST	PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.  Digital I/O. External interrupt 0.
RB1/INT1/SEG8 RB1 INT1 SEG8	57	I/O I O	TTL ST Analog	Digital I/O. External interrupt 1. SEG8 output for LCD.
RB2/INT2/SEG9 RB2 INT2 SEG9	56	I/O I O	TTL ST Analog	Digital I/O. External interrupt 2. SEG9 output for LCD.
RB3/INT3/SEG10 RB3 INT3 SEG10	55	I/O I O	TTL ST Analog	Digital I/O. External interrupt 3. SEG10 output for LCD.
RB4/KBI0/SEG11 RB4 KBI0 SEG11	54	I/O I O	TTL TTL Analog	Digital I/O. Interrupt-on-change pin. SEG11 output for LCD.
RB5/KBI1 RB5 KBI1	53	I/O I	TTL TTL	Digital I/O. Interrupt-on-change pin.
RB6/KBI2/PGC RB6 KBI2 PGC	52	I/O I I/O	TTL TTL ST	Digital I/O. Interrupt-on-change pin. In-Circuit Debugger and ICSP™ programming clock pin.
RB7/KBI3/PGD RB7 KBI3 PGD	47	I/O I I/O	TTL TTL ST	Digital I/O. Interrupt-on-change pin. In-Circuit Debugger and ICSP programming data pin.

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

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





# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RD0/SEG0 RD0 SEG0	72	I/O O	ST Analog	PORTD is a bidirectional I/O port.  Digital I/O. SEG0 output for LCD.
RD1/SEG1 RD1 SEG1	69	I/O O	ST Analog	
RD2/SEG2 RD2 SEG2	68	I/O O	ST Analog	
RD3/SEG3 RD3 SEG3	67	I/O O	ST Analog	
RD4/SEG4 RD4 SEG4	66	I/O O	ST Analog	
RD5/SEG5 RD5 SEG5	65	I/O O	ST Analog	
RD6/SEG6 RD6 SEG6	64	I/O O	ST Analog	
RD7/SEG7 RD7 SEG7	63	I/O O	ST Analog	

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTE is a bidirectional I/O port.
LCDBIAS1 LCDBIAS1	4	I	Analog	BIAS1 input for LCD.
LCDBIAS2 LCDBIAS2	3	I	Analog	BIAS2 input for LCD.
LCDBIAS3 LCDBIAS3	78	I	Analog	BIAS3 input for LCD.
COM0 COM0	77	O	Analog	COM0 output for LCD.
RE4/COM1 RE4 COM1	76	I/O O	ST Analog	Digital I/O. COM1 output for LCD.
RE5/COM2 RE5 COM2	75	I/O O	ST Analog	Digital I/O. COM2 output for LCD.
RE6/COM3 RE6 COM3	74	I/O O	ST Analog	Digital I/O. COM3 output for LCD.
RE7/CCP2/SEG31 RE7 CCP2 <sup>(2)</sup> SEG31	73	I/O I/O O	ST ST Analog	Digital I/O. Capture 2 input/Compare 2 output/PWM2 output. SEG31 output for LCD.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RF0/AN5/SEG18	24			PORTF is a bidirectional I/O port.
RF0		I/O	ST	Digital I/O.
AN5		I	Analog	Analog input 5.
SEG18		O	Analog	SEG18 output for LCD.
RF1/AN6/C2OUT/SEG19	23			
RF1		I/O	ST	Digital I/O.
AN6		I	Analog	Analog input 6.
C2OUT		O	—	Comparator 2 output.
SEG19		O	Analog	SEG19 output for LCD.
RF2/AN7/C1OUT/SEG20	18			
RF2		I/O	ST	Digital I/O.
AN7		I	Analog	Analog input 7.
C1OUT		O	—	Comparator 1 output.
SEG20		O	Analog	SEG20 output for LCD.
RF3/AN8/SEG21	17			
RF3		I/O	ST	Digital I/O.
AN8		I	Analog	Analog input 8.
SEG21		O	Analog	SEG21 output for LCD.
RF4/AN9/SEG22	16			
RF4		I/O	ST	Digital I/O.
AN9		I	Analog	Analog input 9.
SEG22		O	Analog	SEG22 output for LCD.
RF5/AN10/CVREF/SEG23	15			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog input 10.
CVREF		O	Analog	Comparator reference voltage output.
SEG23		O	Analog	SEG23 output for LCD.
RF6/AN11/SEG24	14			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog input 11.
SEG24		O	Analog	SEG24 output for LCD.
RF7/ $\overline{SS}$ /SEG25	13			
RF7		I/O	ST	Digital I/O.
$\overline{SS}$		I	TTL	SPI slave select input.
SEG25		O	Analog	SEG25 output for LCD.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RG0/SEG30 RG0 SEG30	5	I/O O	ST Analog	PORTG is a bidirectional I/O port.  Digital I/O. SEG30 output for LCD.
RG1/TX2/CK2/SEG29 RG1 TX2 CK2 SEG29	6	I/O O I/O O	ST — ST Analog	Digital I/O. AUSART2 asynchronous transmit. AUSART2 synchronous clock (see related RX2/DT2). SEG29 output for LCD.
RG2/RX2/DT2/SEG28 RG2 RX2 DT2 SEG28	7	I/O I I/O O	ST ST ST Analog	Digital I/O. AUSART2 asynchronous receive. AUSART2 synchronous data (see related TX2/CK2). SEG28 output for LCD.
RG3/SEG27 RG3 SEG27	8	I/O O	ST Analog	Digital I/O. SEG27 output for LCD.
RG4/SEG26 RG4 SEG26	10	I/O O	ST Analog	Digital I/O. SEG26 output for LCD.
RG5				See $\overline{\text{MCLR}}/\text{VPP}/\text{RG5}$ pin.

**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 V<sub>DD</sub>)

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

# PIC18F6390/6490/8390/8490

TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RH0/SEG47 RH0 SEG47	79	I/O O	ST Analog	PORTH is a bidirectional I/O port. Digital I/O. SEG47 output for LCD.
RH1/SEG46 RH1 SEG46	80	I/O O	ST Analog	Digital I/O. SEG46 output for LCD.
RH2/SEG45 RH2 SEG45	1	I/O O	ST Analog	Digital I/O. SEG45 output for LCD.
RH3/SEG44 RH3 SEG44	2	I/O O	ST Analog	Digital I/O. SEG44 output for LCD.
RH4/SEG40 RH4 SEG40	22	I/O O	ST Analog	Digital I/O. SEG40 output for LCD.
RH5/SEG41 RH5 SEG41	21	I/O O	ST Analog	Digital I/O. SEG41 output for LCD.
RH6/SEG42 RH6 SEG42	20	I/O O	ST Analog	Digital I/O. SEG42 output for LCD.
RH7/SEG43 RH7 SEG43	19	I/O O	ST Analog	Digital I/O. SEG43 output for LCD.

**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)

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

# PIC18F6390/6490/8390/8490

**TABLE 1-3: PIC18F8X90 PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RJ0/SEG32 RJ0 SEG32	62	I/O O	ST Analog	PORTJ is a bidirectional I/O port.  Digital I/O. SEG32 output for LCD.
RJ1/SEG33 RJ1 SEG33	61	I/O O	ST Analog	Digital I/O. SEG33 output for LCD.
RJ2/SEG34 RJ2 SEG34	60	I/O O	ST Analog	Digital I/O. SEG34 output for LCD.
RJ3/SEG35 RJ3 SEG35	59	I/O O	ST Analog	Digital I/O. SEG35 output for LCD.
RJ4/SEG39 RJ4 SEG39	39	I/O O	ST Analog	Digital I/O. SEG39 output for LCD.
RJ5/SEG38 RJ5 SEG38	40	I/O O	ST Analog	Digital I/O SEG38 output for LCD.
RJ6/SEG37 RJ6 SEG37	41	I/O O	ST Analog	Digital I/O. SEG37 output for LCD.
RJ7/SEG36 RJ7 SEG36	42	I/O O	ST Analog	Digital I/O. SEG36 output for LCD.
Vss	11, 31, 51, 70	P	—	Ground reference for logic and I/O pins.
VDD	12, 32, 48, 71	P	—	Positive supply for logic and I/O pins.
AVSS	26	P	—	Ground reference for analog modules.
AVDD	25	P	—	Positive supply for analog modules.

**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)

**Note 1:** Default assignment for CCP2 when Configuration bit, CCP2MX, is set.

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

# PIC18F6390/6490/8390/8490

---

NOTES:



# PIC18F6390/6490/8390/8490

## 2.0 OSCILLATOR CONFIGURATIONS

### 2.1 Oscillator Types

PIC18F6390/6490/8390/8490 devices can be operated in ten different oscillator modes. The user can program the Configuration bits, FOSC3:FOSC0, in Configuration Register 1H to select one of these ten modes:

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

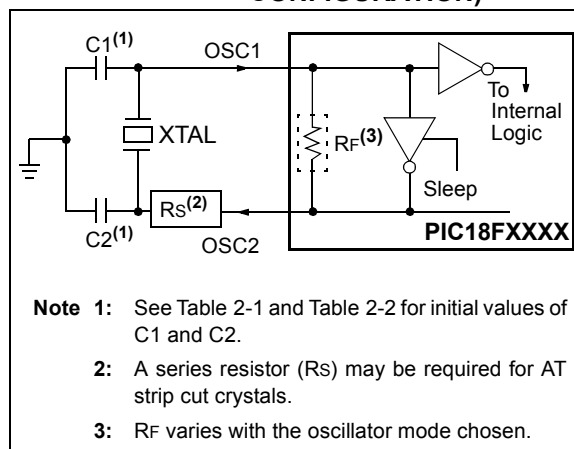
### 2.2 Crystal Oscillator/Ceramic Resonators

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

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

**Note:** Use of a series cut crystal may give a frequency out of the crystal manufacturer's specifications.

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



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

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

**Capacitor values are for design guidance only.**  
 These capacitors were tested with the resonators listed below for basic start-up and operation. **These values are not optimized.**  
 Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.  
 See the notes following Table 2-2 for additional information.

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

# PIC18F6390/6490/8390/8490

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

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

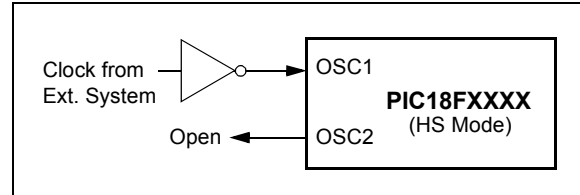
**Capacitor values are for design guidance only.**  
 These capacitors were tested with the crystals listed below for basic start-up and operation. **These values are not optimized.**  
 Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.  
 See the notes following this table for additional information.

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

- Note 1:** Higher capacitance increases the stability of oscillator, but also increases the start-up time.
- When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.
  - Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.
  - Rs may be required to avoid overdriving crystals with low drive level specification.
  - Always verify oscillator performance over the VDD and temperature range that is expected for the application.

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

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

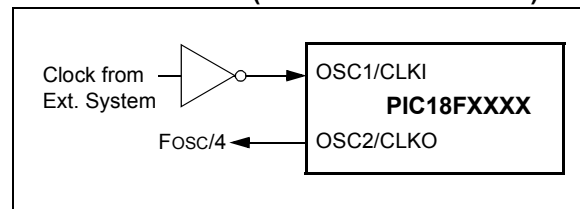


## 2.3 External Clock Input

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

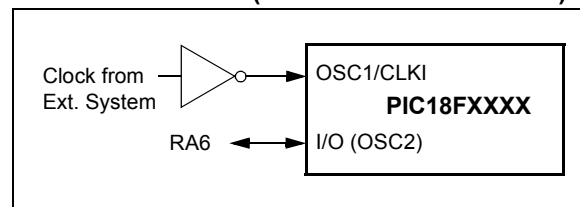
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-3 shows the pin connections for the EC Oscillator mode.

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



The ECIO Oscillator mode functions like the EC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 2-4 shows the pin connections for the ECIO Oscillator mode.

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



## 2.4 RC Oscillator

For timing insensitive applications, the “RC” and “RCIO” device options offer additional cost savings. The actual oscillator frequency is a function of several factors:

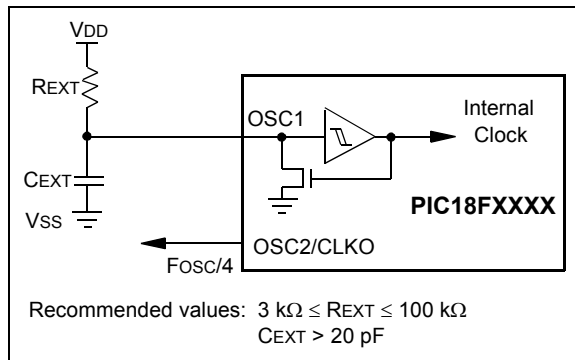
- Supply voltage
- Values of the external resistor (R<sub>EXT</sub>) and capacitor (C<sub>EXT</sub>)
- Operating temperature

Given the same device, operating voltage and temperature and component values, there will also be unit-to-unit frequency variations. These are due to factors such as:

- Normal manufacturing variation
- Difference in lead frame capacitance between package types (especially for low C<sub>EXT</sub> values)
- Variations within the tolerance of limits of R<sub>EXT</sub> and C<sub>EXT</sub>

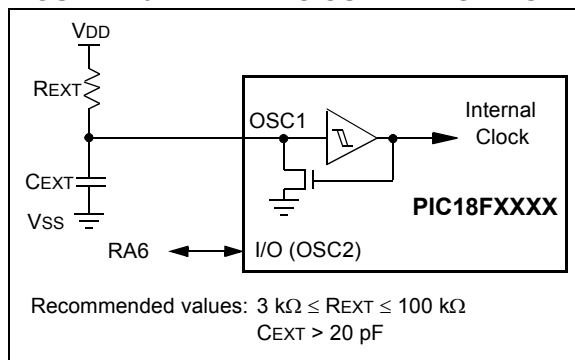
In the RC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-5 shows how the R/C combination is connected.

**FIGURE 2-5: RC OSCILLATOR MODE**



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

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



## 2.5 PLL Frequency Multiplier

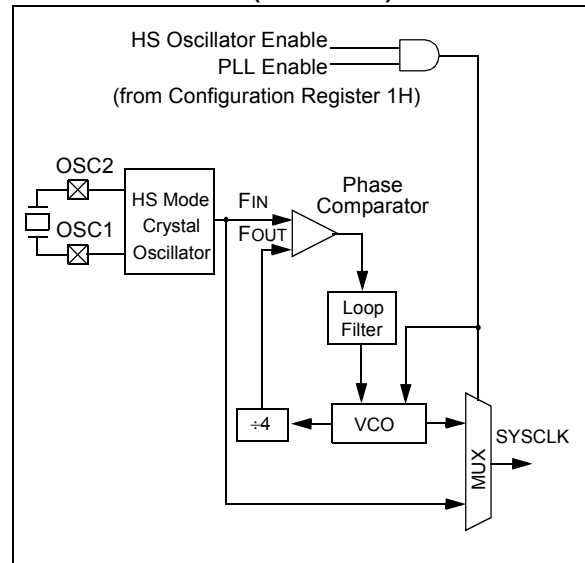
A Phase Locked Loop (PLL) circuit is provided as an option for users who want to use a lower frequency oscillator circuit, or to clock the device up to its highest rated frequency from a crystal oscillator. This may be useful for customers who are concerned with EMI due to high-frequency crystals, or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

### 2.5.1 HSPLL OSCILLATOR MODE

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

The PLL is only available to the crystal oscillator when the FOSC3:FOSC0 Configuration bits are programmed for HSPLL mode (= 0110).

**FIGURE 2-7: PLL BLOCK DIAGRAM (HS MODE)**



### 2.5.2 PLL AND INTOSC

The PLL is also available to the internal oscillator block in selected oscillator modes. In this configuration, the PLL is enabled in software and generates a clock output of up to 32 MHz. The operation of INTOSC with the PLL is described in **Section 2.6.4 “PLL in INTOSC Modes”**.

# PIC18F6390/6490/8390/8490

## 2.6 Internal Oscillator Block

The PIC18F6390/6490/8390/8490 devices include an internal oscillator block, which generates two different clock signals; either can be used as the microcontroller's clock source. This 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 a postscaler, which can provide a range of clock frequencies from 31 kHz to 4 MHz. The INTOSC output is enabled when a clock frequency from 125 kHz to 8 MHz is selected.

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
- LCD with INTRC as its clock source

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

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (Register 2-2).

### 2.6.1 INTIO MODES

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

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

### 2.6.2 INTOSC OUTPUT FREQUENCY

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8.0 MHz.

The 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.

### 2.6.3 OSCTUNE REGISTER

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

When the OSCTUNE register is modified, the INTOSC and INTRC frequencies will begin shifting to the new frequency. The INTRC clock will reach the new frequency within 8 clock cycles (approximately  $8 * 32 \mu\text{s} = 256 \mu\text{s}$ ). The INTOSC clock will stabilize within 1 ms. Code execution continues during this shift. There is no indication that the shift has occurred.

The OSCTUNE register also implements the INTSRC and PLEN bits, which control certain features of the internal oscillator block. 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 greater detail in **Section 2.7.1 "Oscillator Control Register"**.

The PLEN bit controls the operation of the frequency multiplier, PLL, in internal oscillator modes.

### 2.6.4 PLL IN INTOSC MODES

The 4x frequency multiplier can be used with the internal oscillator block to produce faster device clock speeds than are normally possible with an internal oscillator. When enabled, the PLL produces a clock speed of up to 32 MHz.

Unlike HSPLL mode, the PLL is controlled through software. The control bit, PLEN (OSCTUNE<6>), is used to enable or disable its operation.

The PLL is available when the device is configured to use the internal oscillator block as its primary clock source (FOSC3:FOSC0 = 1001 or 1000). Additionally, the PLL will only function when the selected output frequency is either 4 MHz or 8 MHz (OSCCON<6:4> = 111 or 110). If both of these conditions are not met, the PLL is disabled.

The PLEN control bit is only functional in those internal oscillator modes where the PLL is available. In all other modes, it is forced to '0' and is effectively unavailable.

### 2.6.5 INTOSC FREQUENCY DRIFT

The factory calibrates the internal oscillator block output (INTOSC) for 8 MHz. However, this frequency may drift as VDD or temperature changes, which can affect the controller operation in a variety of ways. It is possible to adjust the INTOSC frequency by modifying the value in the OSTUNE 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. Three compensation techniques are discussed in **Section 2.6.5.1 "Compensating with the AUSART"**, **Section 2.6.5.2 "Compensating with the Timers"** and **Section 2.6.5.3 "Compensating with the Timers"**, but other techniques may be used.

# PIC18F6390/6490/8390/8490

## 2.6.5.1 Compensating with the AUSART

An adjustment may be required when the AUSART 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 OSTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low. To compensate, increment OSTUNE to increase the clock frequency.

## 2.6.5.2 Compensating with the Timers

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

Both timers are cleared, but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value

is greater than expected, then the internal oscillator block is running too fast. To adjust for this, decrement the OSTUNE register.

## 2.6.5.3 Compensating with the Timers

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

If the measured time is much greater than the calculated time, then the internal oscillator block is running too fast. To compensate, decrement the OSTUNE register. If the measured time is much less than the calculated time, then the internal oscillator block is running too slow. To compensate, increment the OSTUNE register.

## REGISTER 2-1: OSTUNE: OSCILLATOR TUNING REGISTER

R/W-0	R/W-0 <sup>(1)</sup>	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	PLLEN <sup>(1)</sup>	—	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 PLL for INTOSC Enable bit<sup>(1)</sup>  
          1 = PLL enabled for INTOSC (4 MHz and 8 MHz only)  
          0 = PLL disabled
- bit 5    **Unimplemented:** Read as '0'
- bit 4-0   **TUN4:TUN0:** Frequency Tuning bits  
          01111 = Maximum frequency  
          •          •  
          •          •  
          00001  
          00000 = Center frequency. Oscillator module is running at the calibrated frequency.  
          11111  
          •          •  
          •          •  
          10000 = Minimum frequency

**Note 1:** Available only in certain oscillator configurations; otherwise, this bit is unavailable and read as '0'. See **Section 2.6.4 "PLL in INTOSC Modes"** for details.

# PIC18F6390/6490/8390/8490

## 2.7 Clock Sources and Oscillator Switching

Like previous PIC18 devices, the PIC18F6390/6490/8390/8490 family includes a feature that allows the device clock source to be switched from the main oscillator to an alternate low-frequency clock source. PIC18F6390/6490/8390/8490 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 RC modes, the External Clock modes and the internal oscillator block. The particular mode is defined by the FOSC3:FOSC0 Configuration bits. The details of these modes are covered earlier in this chapter.

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

PIC18F6390/6490/8390/8490 devices offer the Timer1 oscillator as a secondary oscillator. This oscillator, in all power-managed modes, is often the time base for functions such as a Real-Time Clock.

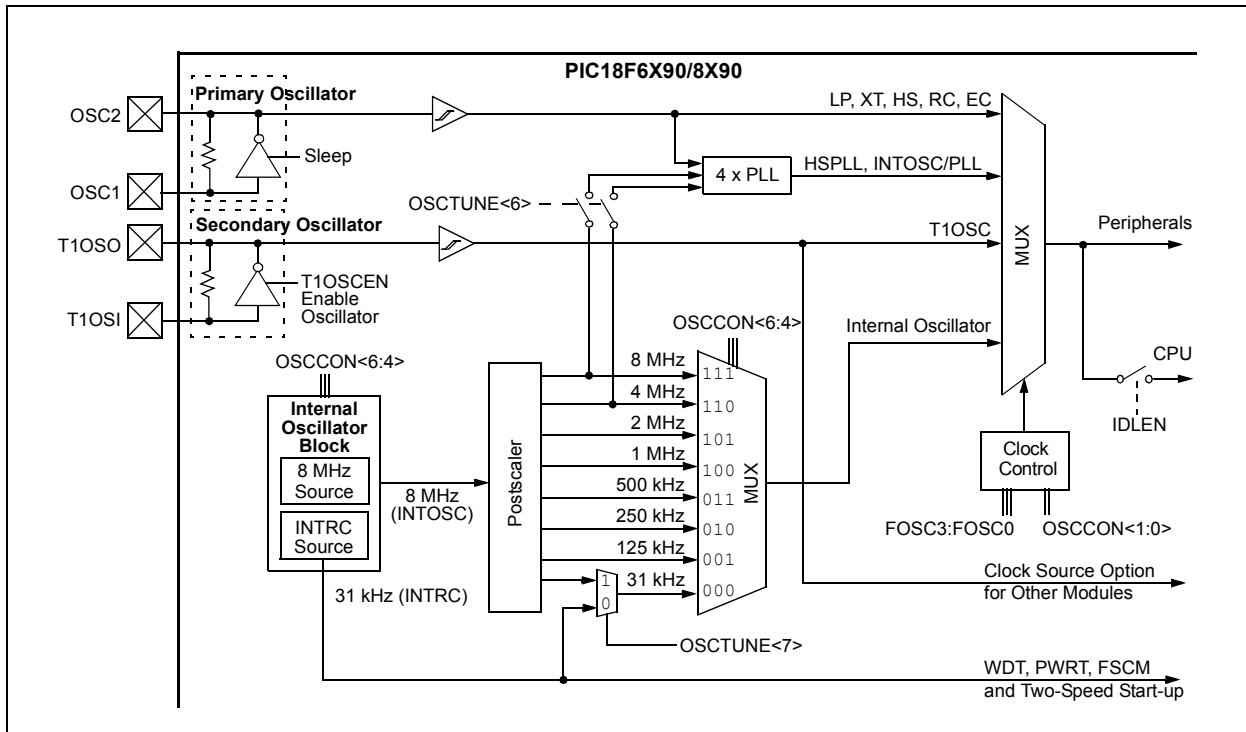
Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T13CKI and RC1/T1OSI pins. Like the LP Oscillator mode circuit, loading capacitors are also connected from each pin to ground.

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

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

The clock sources for the PIC18F6390/6490/8390/8490 devices are shown in Figure 2-8. See **Section 23.0 “Special Features of the CPU”** for Configuration register details.

**FIGURE 2-8: PIC18F6390/6490/8390/8490 CLOCK DIAGRAM**



## 2.7.1 OSCILLATOR CONTROL REGISTER

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

The System Clock Select bits, SCS1:SCS0, select the clock source. The available clock sources are the primary clock (defined by the FOSC:FOSC0 Configuration bits), the secondary clock (Timer1 oscillator) and the internal oscillator block. 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, IRCF2:IRCF0, select the frequency output of the internal oscillator block to drive the device clock. The choices are the INTRC source, the INTOSC source (8 MHz) or one of the frequencies derived from the INTOSC postscaler (31.25 kHz to 4 MHz). If the internal oscillator block is supplying the device clock, changing the states of these bits will have an immediate change on the internal oscillator's output.

When an output frequency of 31 kHz is selected (IRCF2:IRCF0 = 000), users may choose which internal oscillator 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 Watchdog Timer and the Fail-Safe Clock Monitor.

The OSTS, IOFS and T1RUN bits indicate which clock source is currently providing the device clock. The OSTS bit indicates that the Oscillator Start-up Timer has timed out and the primary clock is providing the device clock in primary clock modes. The IOFS bit indicates when the internal oscillator block has stabilized and is providing the device clock in RC 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 three 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 "Power-Managed Modes"**.

**Note 1:** The Timer1 oscillator must be enabled to select the secondary clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON<3>). If the Timer1 oscillator is not enabled, then any attempt to select a secondary clock source when executing a SLEEP instruction will be ignored.

**2:** It is recommended that the Timer1 oscillator be operating and stable before executing the SLEEP instruction, or a very long delay may occur while the Timer1 oscillator starts.

## 2.7.2 OSCILLATOR TRANSITIONS

PIC18F6390/6490/8390/8490 devices contain circuitry to prevent clock "glitches" when switching between clock sources. A short pause in the device clock occurs during the clock switch. The length of this pause is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

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

# PIC18F6390/6490/8390/8490

## REGISTER 2-2: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R/W-1	R/W-0	R/W-0	R <sup>(1)</sup>	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	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 **IRCF2:IRCF0:** Internal Oscillator Frequency Select bits

111 = 8 MHz (INTOSC drives clock directly)

110 = 4 MHz

101 = 2 MHz

100 = 1 MHz<sup>(3)</sup>

011 = 500 kHz

010 = 250 kHz

001 = 125 kHz

000 = 31 kHz (from either INTOSC/256 or INTRC directly)<sup>(2)</sup>

bit 3 **OSTS:** Oscillator Start-up Timer 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 **IOFS:** INTOSC Frequency Stable bit

1 = INTOSC frequency is stable

0 = INTOSC frequency is not stable

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

1x = Internal oscillator block

01 = Timer1 oscillator

00 = Primary oscillator

**Note 1:** Depends on state of the IESO Configuration bit.

**Note 2:** Source selected by the INTSRC bit (OSCTUNE<7>), see **Section 2.6.3 “OSCTUNE Register”**.

**Note 3:** Default output frequency of INTOSC on Reset.



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

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

In secondary clock modes (SEC\_RUN and SEC\_IDLE), the Timer1 oscillator is operating and providing the device clock. The Timer1 oscillator may also run in all power-managed modes if required to clock Timer1 or Timer3.

In 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 23.2 “Watchdog Timer (WDT)”** through **Section 23.4 “Fail-Safe Clock Monitor”** for more information on WDT, Fail-Safe Clock Monitor 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 the Sleep mode is selected, all clock sources are stopped. Since all the transistor switching currents have been stopped, Sleep mode achieves the lowest current consumption of the device (only leakage currents).

Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The INTRC is required to support WDT operation. The Timer1 oscillator may be operating to support a Real-Time Clock. Other features may be operating that do not require a device clock source (i.e., MSSP slave, INTx pins and others). Peripherals that may add significant current consumption are listed in **Section 26.2 “DC Characteristics: Power-Down and Supply Current”**.

## 2.9 Power-up Delays

Power-up delays are controlled by two timers, so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply is stable under normal circumstances and the primary clock is operating and stable. For additional information on power-up delays, see **Section 4.5 “Device Reset Timers”**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on power-up (parameter 33, **Table 26-10**). It is enabled by clearing (= 0) the PWRTEN Configuration bit.

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

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

There is a delay of interval TcSD (parameter 38, **Table 26-10**) following POR while the controller becomes ready to execute instructions. This delay runs concurrently with any other delays. This may be the only delay that occurs when any of the EC, RC or INTIO modes are used as the primary clock source.

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

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

**Note:** See Table 4-2 in **Section 4.0 “Reset”** for time-outs due to Sleep and MCLR Reset.

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 3.0 POWER-MANAGED MODES

PIC18F6390/6490/8390/8490 devices offer a total of seven operating modes for more efficient power management. These modes provide a variety of options for selective power conservation in applications where resources may be limited (i.e., battery-powered devices).

There are three categories of power-managed modes:

- Sleep mode
- Idle modes
- Run modes

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

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

### 3.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires deciding if the CPU is to be clocked or not and selecting a clock source. The IDLEN bit controls CPU clocking, while the SCS1:SCS0 bits select a clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 3-1.

### 3.1.1 CLOCK SOURCES

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

- the primary clock, as defined by the FOSC3:FOSC0 Configuration bits
- the secondary clock (the Timer1 oscillator)
- the internal oscillator block (for RC modes)

### 3.1.2 ENTERING POWER-MANAGED MODES

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

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

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

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

Mode	OSCCON<7,1:0>		Module Clocking		Available Clock and Oscillator Source
	IDLEN <sup>(1)</sup>	SCS1:SCS0	CPU	Peripherals	
Sleep	0	N/A	Off	Off	None – All clocks are disabled
PRI_RUN	N/A	00	Clocked	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC, INTRC <sup>(2)</sup> ; This is the normal, full-power execution mode.
SEC_RUN	N/A	01	Clocked	Clocked	Secondary – Timer1 Oscillator
RC_RUN	N/A	1x	Clocked	Clocked	Internal Oscillator Block <sup>(2)</sup>
PRI_IDLE	1	00	Off	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC
SEC_IDLE	1	01	Off	Clocked	Secondary – Timer1 Oscillator
RC_IDLE	1	1x	Off	Clocked	Internal Oscillator Block <sup>(2)</sup>

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

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

# PIC18F6390/6490/8390/8490

## 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.

Three bits indicate the current clock source and its status. They are:

- OSTS (OSCCON<3>)
- IOFS (OSCCON<2>)
- T1RUN (T1CON<6>)

In general, only one of these bits will be set while in a given power-managed mode. When the OSTS bit is set, the primary clock is providing the device clock. When the IOFS bit is set, the INTOSC output provides a stable, 8 MHz clock source to a divider that actually drives the device clock. When the T1RUN bit is set, the Timer1 oscillator provides the clock. If none of these bits are set, then either the INTRC clock source clocks the device, or the INTOSC source is not yet stable.

If the internal oscillator block is configured as the primary clock source by the FOSC3:FOSC0 Configuration bits, then both the OSTS and IOFS bits may be set when in PRI\_RUN or PRI\_IDLE modes. This indicates that the primary clock (INTOSC output) is generating a stable 8 MHz output. Entering another power-managed RC mode at the same frequency would clear the OSTS bit.

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

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

## 3.1.4 MULTIPLE SLEEP COMMANDS

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

## 3.2 Run Modes

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

### 3.2.1 PRI\_RUN MODE

The PRI\_RUN mode is the normal, full-power execution mode of the microcontroller. This is also the default mode upon a device Reset unless Two-Speed Start-up is enabled (see **Section 23.3 “Two-Speed Start-up”** for details). In this mode, the OSTS bit is set. The IOFS bit may be set if the internal oscillator block is the primary clock source (see **Section 2.7.1 “Oscillator Control Register”**).

### 3.2.2 SEC\_RUN MODE

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

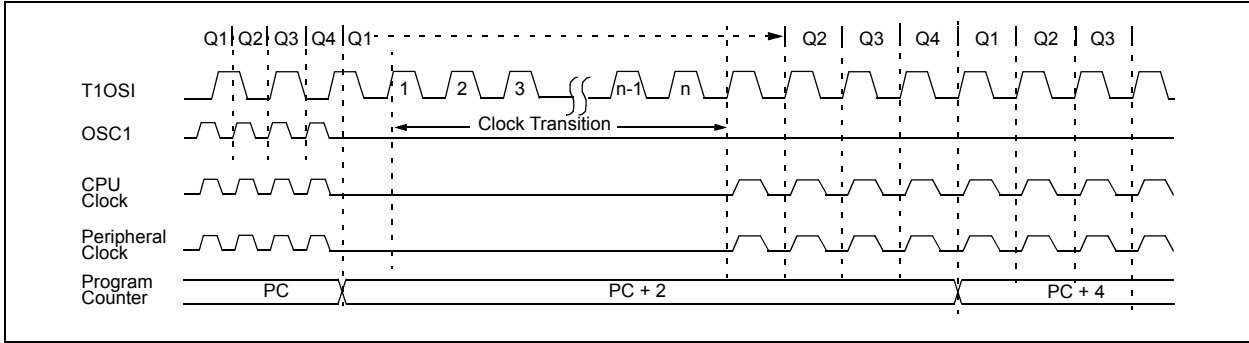
SEC\_RUN mode is entered by setting the SCS1:SCS0 bits to ‘01’. The device clock source is switched to the Timer1 oscillator (see Figure 3-1), the primary oscillator is shut down, the T1RUN bit (T1CON<6>) is set and the OSTS bit is cleared.

**Note:** The Timer1 oscillator should already be running prior to entering SEC\_RUN mode. If the T1OSCN bit is not set when the SCS1:SCS0 bits are set to ‘01’, entry to SEC\_RUN mode will not occur. If the Timer1 oscillator is enabled, but not yet running, 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.

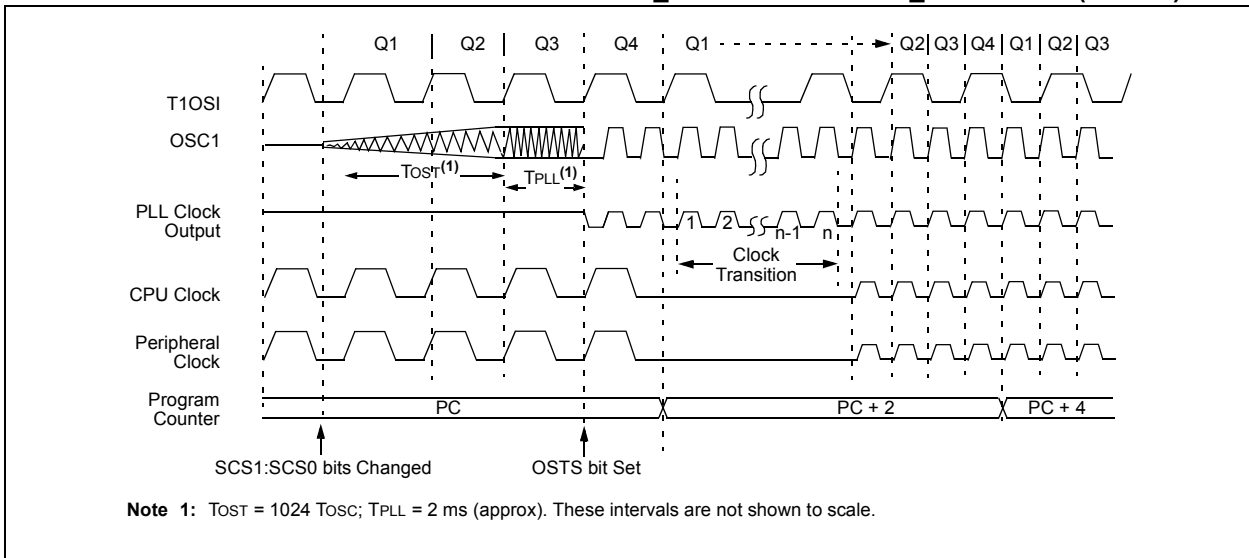
On transitions from SEC\_RUN mode to PRI\_RUN, the peripherals and CPU continue to be clocked from the Timer1 oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 3-2). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock provides the clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

# PIC18F6390/6490/8390/8490

**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)**



# PIC18F6390/6490/8390/8490

## 3.2.3 RC\_RUN MODE

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

If the primary clock source is the internal oscillator block (either INTRC or INTOSC), there are no distinguishable differences between PRI\_RUN and RC\_RUN modes during execution. However, a clock switch delay will occur during entry to and exit from RC\_RUN mode. Therefore, if the primary clock source is the internal oscillator block, the use of RC\_RUN mode is not recommended.

This mode is entered by setting the SCS1 bit to '1'. Although it is ignored, it is recommended that the SCS0 bit also be cleared; this is to maintain software compatibility with future devices. When the clock source is switched to the INTOSC multiplexer (see Figure 3-3), the primary oscillator is shut down and the OSTS bit is cleared. The IRCF bits may be modified at any time to immediately change the clock speed.

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

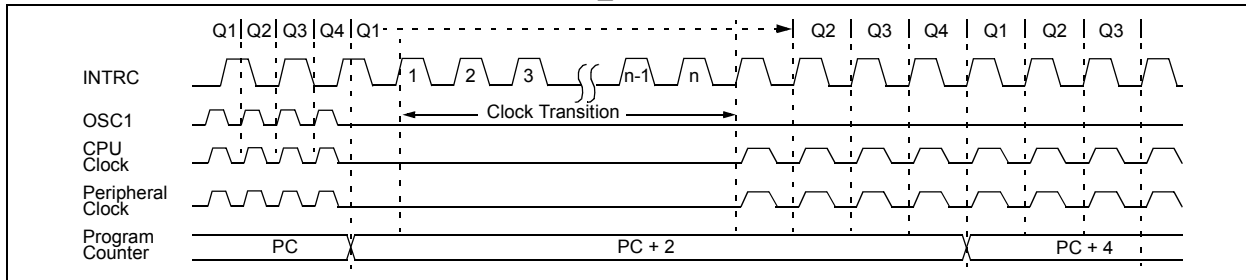
If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source provides the device clocks.

If the IRCF bits are changed from all clear (thus enabling the INTOSC output), or if INTSRC is set, the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the device continue while the INTOSC source stabilizes after an interval of TIOBST.

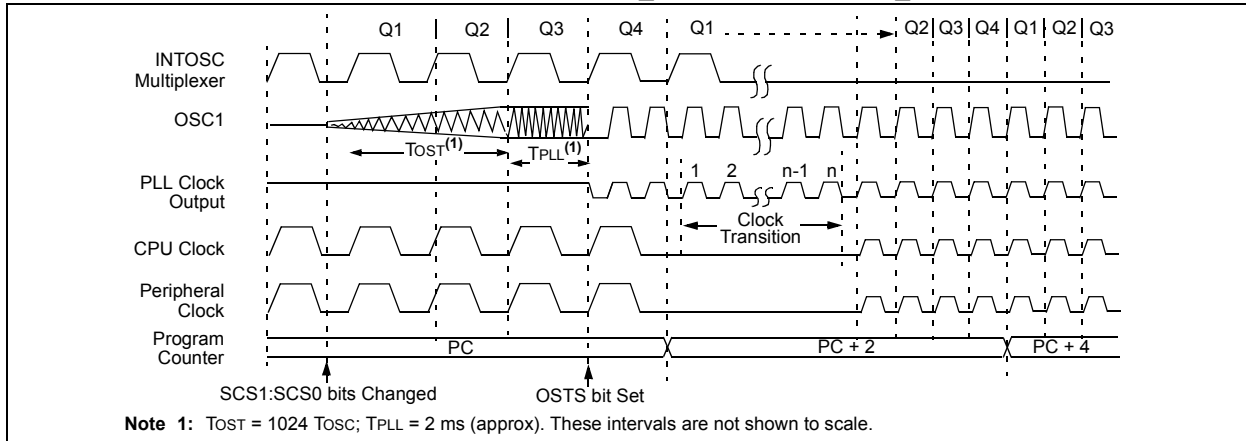
If the IRCF bits were previously at a non-zero value, or if INTSRC was set before setting SCS1 and the INTOSC source was already stable, the IOFS bit will remain set.

On transitions from RC\_RUN mode to PRI\_RUN, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-4). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock provides the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

**FIGURE 3-3: 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 in the PIC18F6390/6490/8390/8490 devices 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 (see 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. 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 primary clock source becomes ready (see Figure 3-6), or it will be clocked from the internal oscillator block if either the Two-Speed Start-up or the Fail-Safe Clock Monitor are enabled (see **Section 23.0 “Special Features of the CPU”**). In either case, the OSTS bit is set when the primary clock provides the device clocks. The IDLEN and SCS bits are not affected by the wake-up.

## 3.4 Idle Modes

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

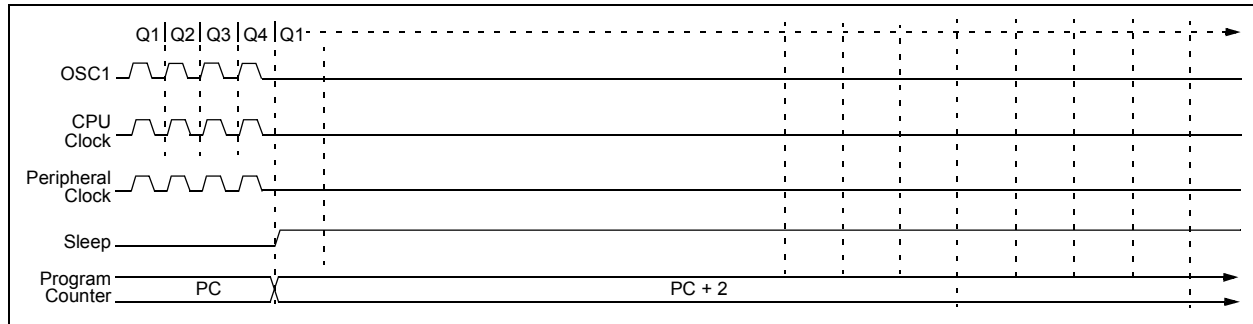
If the IDLEN bit is set to a ‘1’ when a `SLEEP` instruction is executed, the peripherals will be clocked from the clock source selected using the SCS1:SCS0 bits; however, the CPU will not be clocked. The clock source status bits are not affected. Setting IDLEN and executing `SLEEP` provides a quick method of switching from a given Run mode to its corresponding Idle mode.

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

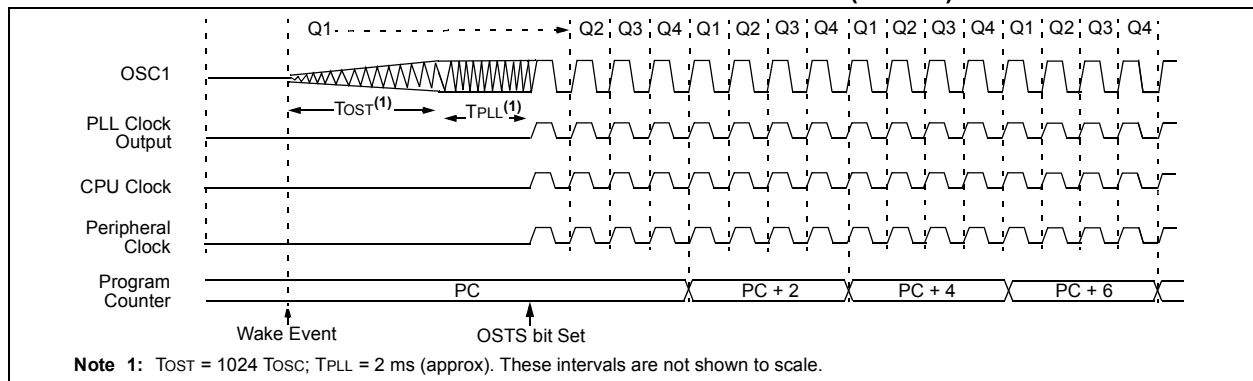
Since the CPU is not executing instructions, the only exits from any of the Idle modes are by interrupt, WDT time-out or a Reset. When a wake event occurs, CPU execution is delayed by an interval of  $T_{CSD}$  (parameter 38, Table 26-10) while it becomes ready to execute code. When the CPU begins executing code, it resumes with the same clock source for the current Idle mode. For example, when waking from RC\_IDLE mode, the internal oscillator block will clock the CPU and peripherals (in other words, RC\_RUN mode). The IDLEN and SCS bits are not affected by the wake-up.

While in any Idle mode or the Sleep mode, a WDT time-out will result in a WDT wake-up to the Run mode currently specified by the SCS1:SCS0 bits.

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



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



# PIC18F6390/6490/8390/8490

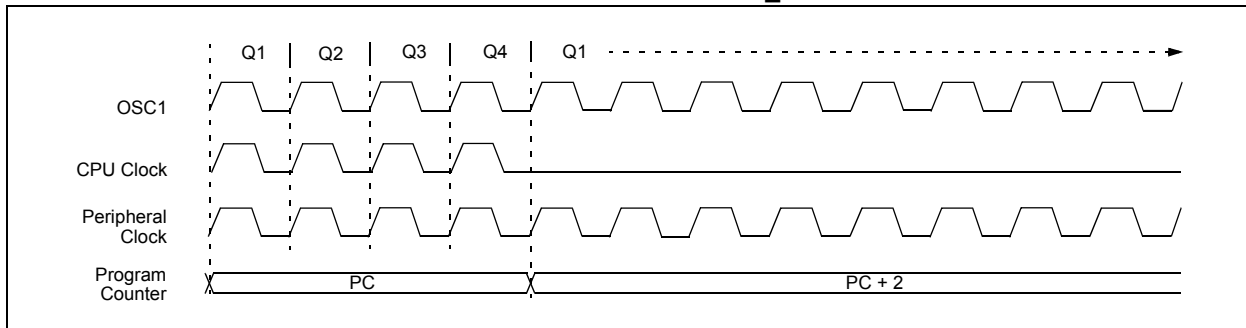
## 3.4.1 PRI\_IDLE MODE

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.

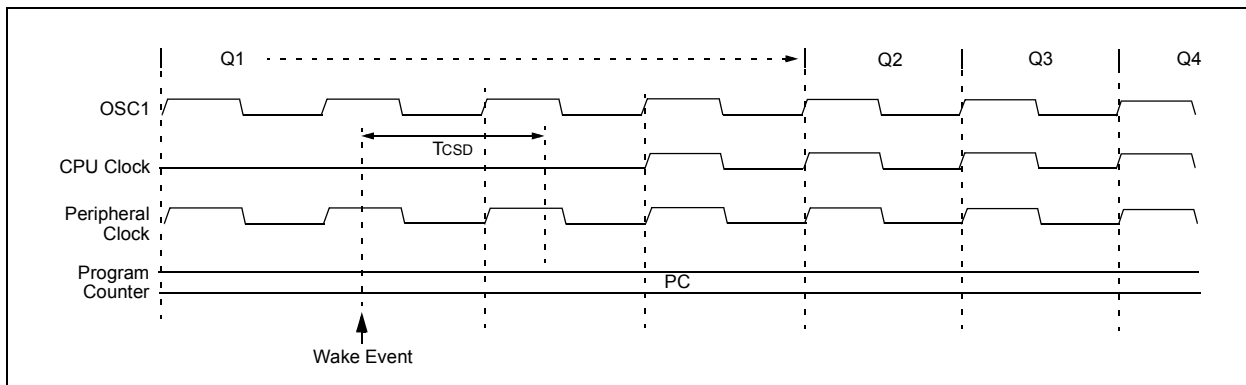
PRI\_IDLE mode is entered from PRI\_RUN mode by setting the IDLEN bit and executing a `SLEEP` instruction. If the device is in another Run mode, set IDLEN first, then clear the SCS bits and execute `SLEEP`. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the FOSC3:FOSC0 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  $T_{CSD}$  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).

**FIGURE 3-7: TRANSITION TIMING FOR ENTRY TO PRI\_IDLE MODE**



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





## 3.4.2 SEC\_IDLE MODE

In SEC\_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered from SEC\_RUN by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, set IDLEN first, then set SCS1:SCS0 to '01' and execute SLEEP. When the clock source is switched to the Timer1 oscillator, the primary oscillator is shut down, the OSTS bit is cleared and the T1RUN bit is set.

When a wake event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After an interval of TcSD following the wake event, the CPU begins executing code being clocked by the Timer1 oscillator. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run (see Figure 3-8).

**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.

## 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 using the INTOSC multiplexer. 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 set the SCS1 bit and execute SLEEP. Although its value is ignored, it is recommended that SCS0 also be cleared; this is to maintain software compatibility with future devices. The INTOSC multiplexer may be used to select a higher clock frequency by modifying the IRCF bits before executing the SLEEP instruction. When the clock source is switched to the INTOSC multiplexer, the primary oscillator is shut down and the OSTS bit is cleared.

If the IRCF bits are set to any non-zero value, or the INTSRC bit is set, the INTOSC output is enabled. The IOFS bit becomes set after the INTOSC output becomes stable after an interval of TIOBST (parameter 39, Table 26-10). Clocks to the peripherals continue while the INTOSC source stabilizes. If the IRCF bits were previously at a non-zero value, or INTSRC was set before the SLEEP instruction was executed and the INTOSC source was already stable, the IOFS bit will remain set. If the IRCF bits and INTSRC are all clear, the INTOSC output will not be enabled; the IOFS bit will remain clear and there will be no indication of the current clock source.

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

# PIC18F6390/6490/8390/8490

---

## 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 (see **Section 3.2 “Run Modes”** through **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 or 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, TcSD, 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 23.2 “Watchdog Timer (WDT)”**).

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

### 3.5.3 EXIT BY RESET

Normally, the device is held in Reset by the Oscillator Start-up Timer (OST) until the primary clock becomes ready. At that time, the OSTS bit is set and the device begins executing code. If the internal oscillator block is the new clock source, the IOFS bit is set instead.

The exit delay time from Reset to the start of code execution depends on both the clock sources before and after the wake-up and the type of oscillator if the new clock source is the primary clock. Exit delays are summarized in Table 3-2.

Code execution can begin before the primary clock becomes ready. If either the Two-Speed Start-up (see **Section 23.3 “Two-Speed Start-up”**) or Fail-Safe Clock Monitor (see **Section 23.4 “Fail-Safe Clock Monitor”**) is enabled, the device may begin execution as soon as the Reset source has cleared. Execution is clocked by the INTOSC multiplexer driven by the internal oscillator block. Execution is clocked by the internal oscillator block until either the primary clock becomes ready, or a power-managed mode is entered before the primary clock becomes ready; the primary clock is then shut down.

### 3.5.4 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 not any of the LP, XT, HS or HSPLL modes.

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 (RC, EC and INTIO Oscillator modes). However, a fixed delay of interval, TcSD, 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.

# PIC18F6390/6490/8390/8490

**TABLE 3-2: EXIT DELAY ON WAKE-UP BY RESET FROM SLEEP MODE OR ANY IDLE MODE (BY CLOCK SOURCES)**

Clock Source before Wake-up	Clock Source after Wake-up	Exit Delay	Clock Ready Status Bit (OSCCON)
Primary Device Clock (PRI_IDLE mode)	LP, XT, HS	T <sub>CSD</sub> <sup>(2)</sup>	OSTS
	HSPLL		—
	EC, RC, INTRC <sup>(1)</sup>		IOFS
	INTOSC <sup>(3)</sup>		
T1OSC or INTRC <sup>(1)</sup>	LP, XT, HS	T <sub>OSt</sub> <sup>(4)</sup>	OSTS
	HSPLL	T <sub>OSt</sub> + t <sub>rc</sub> <sup>(4)</sup>	
	EC, RC, INTRC <sup>(1)</sup>	T <sub>CSD</sub> <sup>(2)</sup>	—
	INTOSC <sup>(3)</sup>	T <sub>IOBST</sub> <sup>(5)</sup>	IOFS
INTOSC <sup>(3)</sup>	LP, XT, HS	T <sub>OSt</sub> <sup>(5)</sup>	OSTS
	HSPLL	T <sub>OSt</sub> + t <sub>rc</sub> <sup>(4)</sup>	
	EC, RC, INTRC <sup>(1)</sup>	T <sub>CSD</sub> <sup>(2)</sup>	—
	INTOSC <sup>(3)</sup>	None	IOFS
None (Sleep mode)	LP, XT, HS	T <sub>OSt</sub> <sup>(4)</sup>	OSTS
	HSPLL	T <sub>OSt</sub> + t <sub>rc</sub> <sup>(4)</sup>	
	EC, RC, INTRC <sup>(1)</sup>	T <sub>CSD</sub> <sup>(2)</sup>	—
	INTOSC <sup>(3)</sup>	T <sub>IOBST</sub> <sup>(5)</sup>	IOFS

**Note 1:** In this instance, refers specifically to the 31 kHz INTRC clock source.

**2:** T<sub>CSD</sub> (parameter 38) is a required delay when waking from Sleep and all Idle modes and runs concurrently with any other required delays (see **Section 3.4 “Idle Modes”**).

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

**4:** T<sub>OSt</sub> is the Oscillator Start-up Timer (parameter 32). t<sub>rc</sub> is the PLL Lock-out Timer (parameter F12); it is also designated as T<sub>PLL</sub>.

**5:** Execution continues during T<sub>IOBST</sub> (parameter 39), the INTOSC stabilization period.

# PIC18F6390/6490/8390/8490

---

NOTES:

## 4.0 RESET

The PIC18F6390/6490/8390/8490 devices differentiate between various kinds of Reset:

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

This section discusses Resets generated by  $\overline{\text{MCLR}}$ , POR and BOR and covers the operation of the various start-up timers. Stack Reset events are covered in **Section 5.1.2.4 “Stack Full and Underflow Resets”**. WDT Resets are covered in **Section 23.2 “Watchdog Timer (WDT)”**.

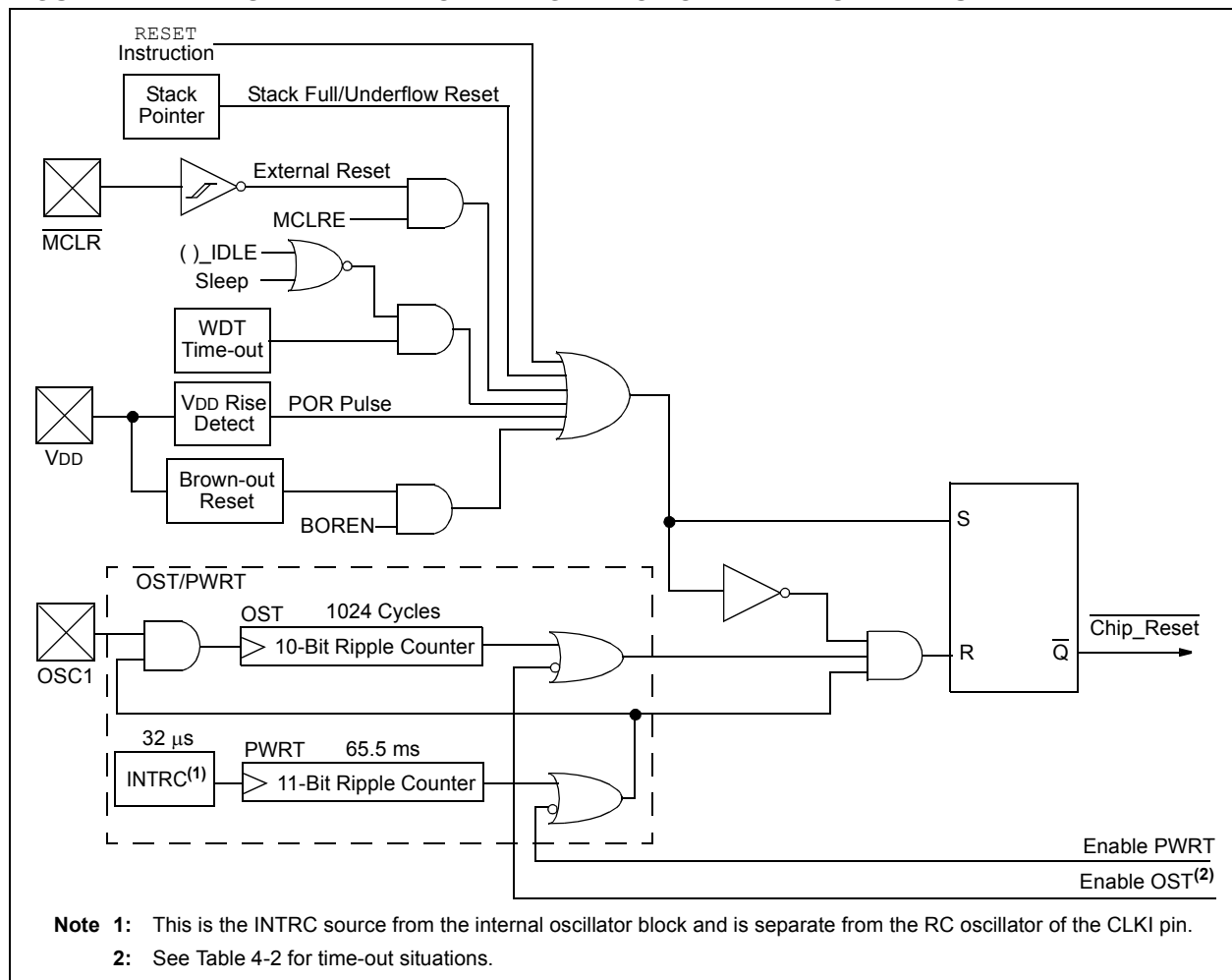
A simplified block diagram of the on-chip Reset circuit is shown in Figure 4-1.

## 4.1 RCON Register

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

The RCON register also has control bits for setting interrupt priority (IPEN) and software control of the BOR (SBOREN). Interrupt priority is discussed in **Section 8.0 “Interrupts”**. BOR is covered in **Section 4.4 “Brown-out Reset (BOR)”**.

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



# PIC18F6390/6490/8390/8490

## REGISTER 4-1: RCON: RESET CONTROL REGISTER

R/W-0	R/W-1 <sup>(1)</sup>	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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            **SBOREN:** BOR Software Enable bit<sup>(1)</sup>  
If BOREN1:BOREN0 = 01:  
 1 = BOR is enabled  
 0 = BOR is disabled  
If BOREN1:BOREN0 = 00, 10 or 11:  
 Bit is disabled and read as '0'.
- bit 5            **Unimplemented:** Read as '0'
- bit 4             **$\overline{RI}$ :** RESET Instruction Flag bit  
 1 = The RESET instruction was not executed (set by firmware only)  
 0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)
- bit 3             **$\overline{TO}$ :** Watchdog Time-out Flag bit  
 1 = Set by power-up, CLRWDT instruction or SLEEP instruction  
 0 = A WDT time-out occurred
- bit 2             **$\overline{PD}$ :** Power-Down Detection Flag bit  
 1 = Set by power-up or by the CLRWDT instruction  
 0 = Set by execution of the SLEEP instruction
- bit 1             **$\overline{POR}$ :** Power-on Reset Status bit  
 1 = A Power-on Reset has not occurred (set by firmware only)  
 0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)
- bit 0             **$\overline{BOR}$ :** Brown-out Reset Status bit  
 1 = A Brown-out Reset has not occurred (set by firmware only)  
 0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

**Note 1:** If SBOREN is enabled, its Reset state is '1'; otherwise, it is '0'.

**Note 1:** It is recommended that the  $\overline{POR}$  bit be set after a Power-on Reset has been detected, so that subsequent Power-on Resets may be detected.

**2:** Brown-out Reset is said to have occurred when  $\overline{BOR}$  is '0' and  $\overline{POR}$  is '1' (assuming that  $\overline{POR}$  was set to '1' by software immediately after a Power-on Reset).

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

The  $\overline{\text{MCLR}}$  pin provides a method for triggering a hard external Reset of the device. A Reset is generated by holding the pin low. PIC18 Extended MCU 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.

In PIC18F6390/6490/8390/8490 devices, the  $\overline{\text{MCLR}}$  input can be disabled with the MCLRE Configuration bit. When  $\overline{\text{MCLR}}$  is disabled, the pin becomes a digital input. See **Section 9.7 “PORTG, TRISG and LATG Registers”** for more information.

## 4.3 Power-on Reset (POR)

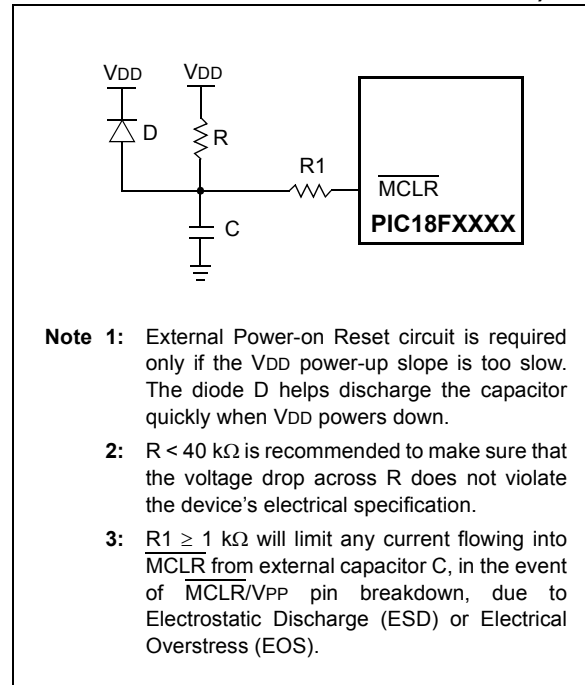
A Power-on Reset pulse is generated on-chip whenever VDD rises above a certain threshold. This allows the device to start in the initialized state when VDD is adequate for operation.

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 Power-on Reset delay. A minimum rise rate for VDD is specified (parameter D004). For a slow rise time, see Figure 4-2.

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

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 Power-on Reset.

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



# PIC18F6390/6490/8390/8490

## 4.4 Brown-out Reset (BOR)

PIC18F6390/6490/8390/8490 devices implement a BOR circuit that provides the user with a number of configuration and power-saving options. The BOR is controlled by the BORV1:BORV0 and BOREN1:BOREN0 Configuration bits. There are a total of four BOR configurations, which are summarized in Table 4-1.

The BOR threshold is set by the BORV1:BORV0 bits. If BOR is enabled (any values of BOREN1:BOREN0 except '00'), any drop of VDD below VBOR (parameter D005) for greater than TBOR (parameter 35) will reset the device. A Reset may or may not occur if VDD falls below VBOR for less than TBOR. The chip will remain in Brown-out Reset until VDD rises above VBOR.

If the Power-up Timer is enabled, it will be invoked after VDD rises above VBOR; it then will keep the chip in Reset for an additional time delay, TPWRT (parameter 33). If VDD drops below VBOR while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be initialized. Once VDD rises above VBOR, the Power-up Timer will execute the additional time delay.

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

### 4.4.1 SOFTWARE ENABLED BOR

When BOREN1:BOREN0 = 01, the BOR can be enabled or disabled by the user in software. This is done with the control bit, SBOREN (RCON<6>). Setting SBOREN enables the BOR to function as previously described. Clearing SBOREN disables the BOR entirely. The SBOREN bit operates only in this mode; otherwise it is read as '0'.

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

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

### 4.4.2 DETECTING BOR

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

### 4.4.3 DISABLING BOR IN SLEEP MODE

When BOREN1:BOREN0 = 10, the BOR remains under hardware control and operates as previously described. Whenever the device enters Sleep mode, however, the BOR is automatically disabled. When the device returns to any other operating mode, BOR is automatically re-enabled.

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

**TABLE 4-1: BOR CONFIGURATIONS**

BOR Configuration		Status of SBOREN (RCON<6>)	BOR Operation
BOREN1	BOREN0		
0	0	Unavailable	BOR is disabled; must be enabled by reprogramming the Configuration bits.
0	1	Available	BOR is enabled in software; operation controlled by SBOREN.
1	0	Unavailable	BOR is enabled in hardware and active during the Run and Idle modes, disabled during Sleep mode.
1	1	Unavailable	BOR is enabled in hardware; must be disabled by reprogramming the Configuration bits.



# PIC18F6390/6490/8390/8490

## 4.5 Device Reset Timers

PIC18F6390/6490/8390/8490 devices incorporate three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

### 4.5.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of PIC18F6390/6490/8390/8490 devices is an 11-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of  $2048 \times 32 \mu\text{s} = 65.6 \text{ ms}$ . While the PWRT is counting, the device is held in Reset.

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

The PWRT is enabled by clearing the  $\overline{\text{PWRTE}}\text{N}$  Configuration bit.

### 4.5.2 OSCILLATOR START-UP TIMER (OST)

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

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

### 4.5.3 PLL LOCK TIME-OUT

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

### 4.5.4 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

1. After the POR pulse has cleared, PWRT time-out is invoked (if enabled).
2. Then, the OST is activated.

The total time-out will vary based on oscillator configuration and the status of the PWRT. Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7 all depict time-out sequences on power-up, with the Power-up Timer enabled and the device operating in HS Oscillator mode. Figures 4-3 through 4-6 also apply to devices operating in XT or LP modes. For devices in RC mode and with the PWRT disabled, on the other hand, there will be no time-out at all.

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

TABLE 4-2: TIME-OUT IN VARIOUS SITUATIONS

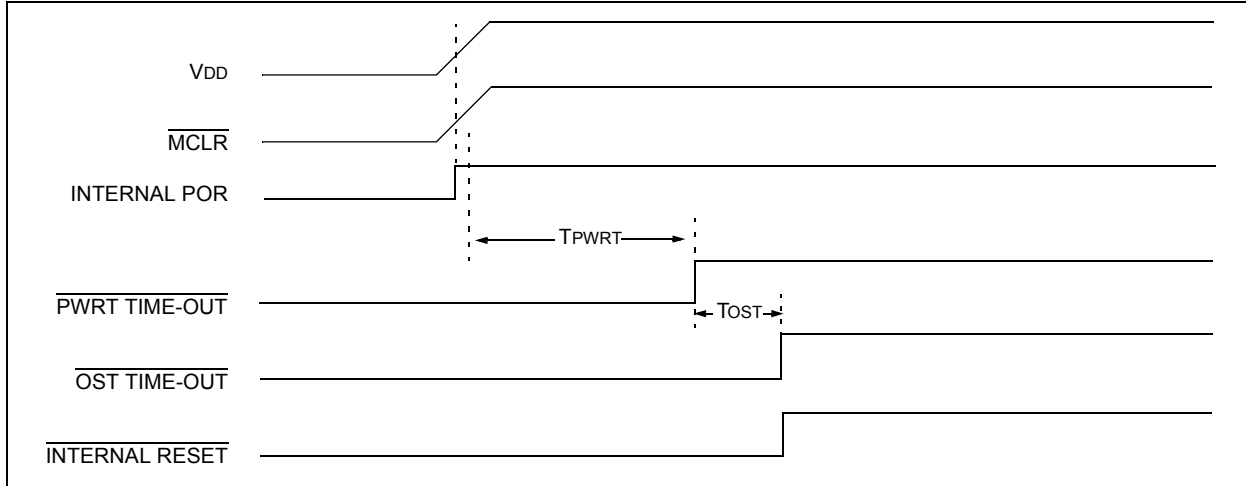
Oscillator Configuration	Power-up <sup>(2)</sup> and Brown-out		Exit from Power-Managed Mode
	$\overline{\text{PWRTE}}\text{N} = 0$	$\overline{\text{PWRTE}}\text{N} = 1$	
HSPLL	$66 \text{ ms}^{(1)} + 1024 \text{ T}_{\text{osc}} + 2 \text{ ms}^{(2)}$	$1024 \text{ T}_{\text{osc}} + 2 \text{ ms}^{(2)}$	$1024 \text{ T}_{\text{osc}} + 2 \text{ ms}^{(2)}$
HS, XT, LP	$66 \text{ ms}^{(1)} + 1024 \text{ T}_{\text{osc}}$	$1024 \text{ T}_{\text{osc}}$	$1024 \text{ T}_{\text{osc}}$
EC, ECIO	$66 \text{ ms}^{(1)}$	—	—
RC, RCIO	$66 \text{ ms}^{(1)}$	—	—
INTIO1, INTIO2	$66 \text{ ms}^{(1)}$	—	—

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

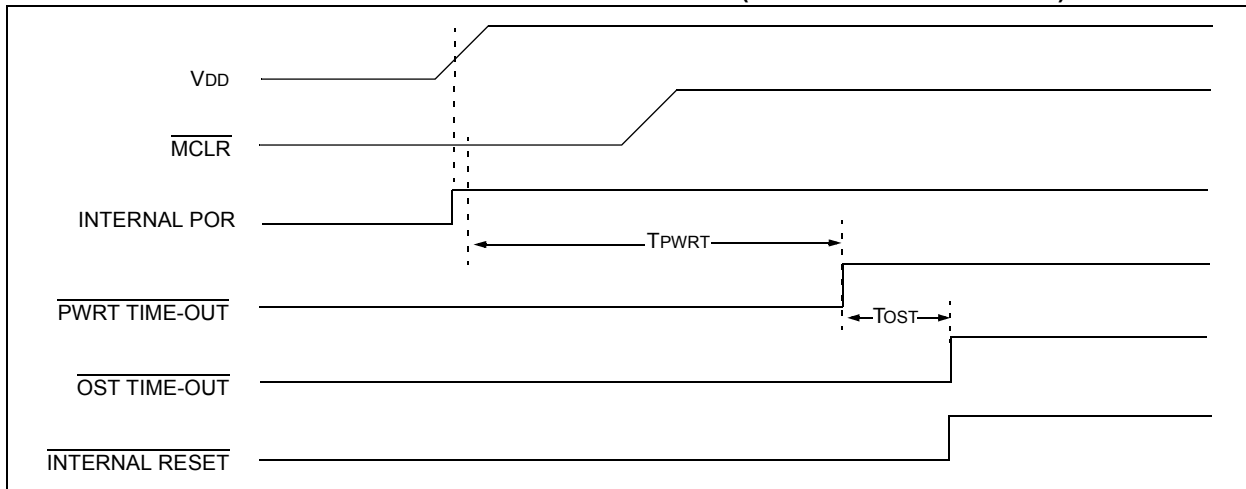
**Note 2:** 2 ms is the nominal time required for the PLL to lock.

# PIC18F6390/6490/8390/8490

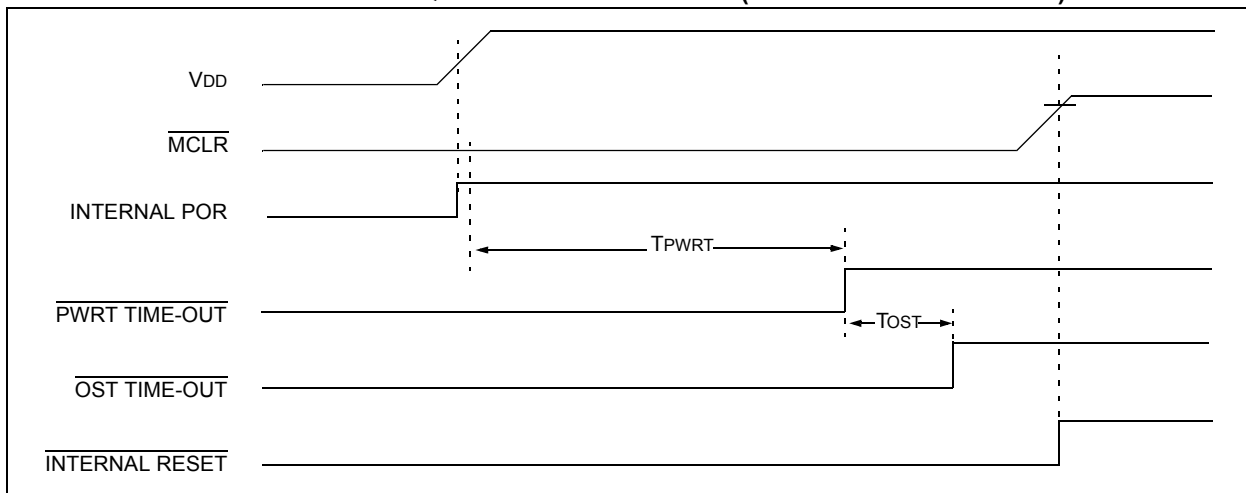
**FIGURE 4-3: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{DD}$ ,  $V_{DD}$  RISE <  $T_{PWRT}$ )**



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

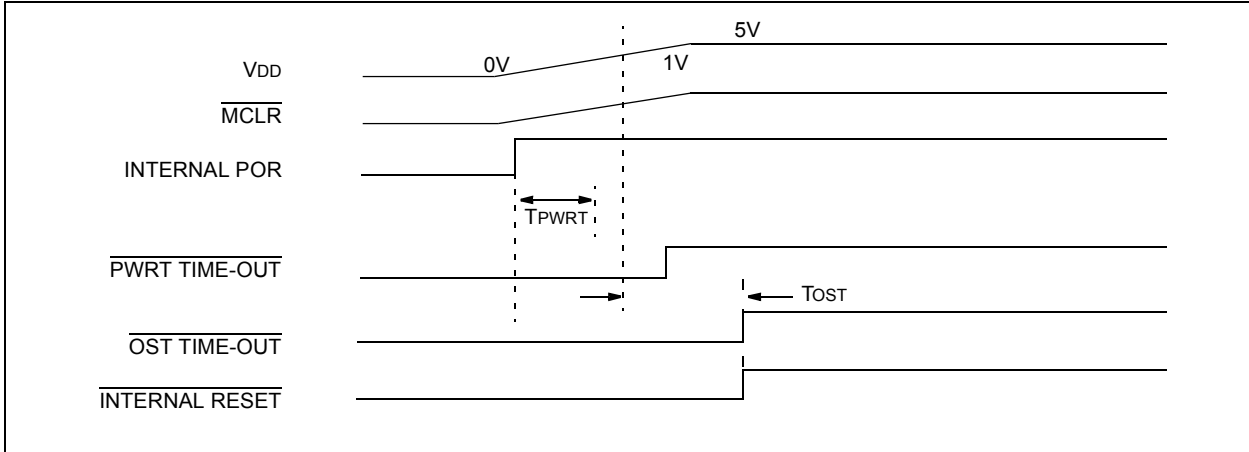


**FIGURE 4-5: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{DD}$ ): CASE 2**

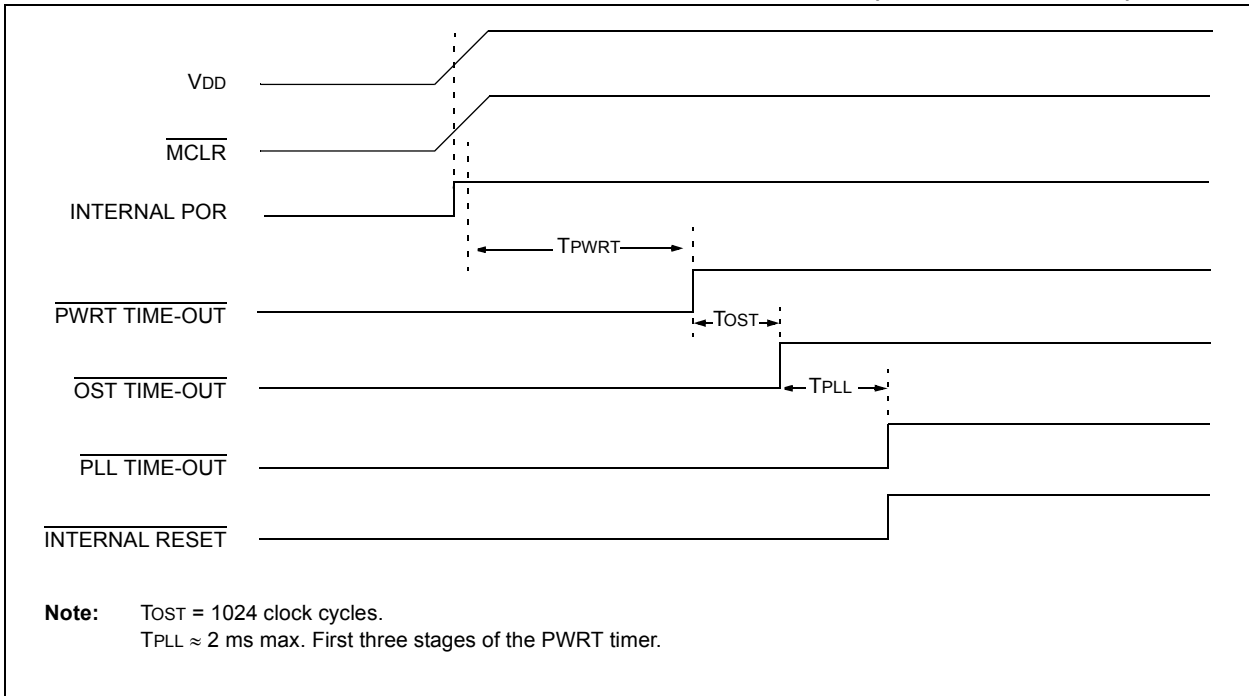


# PIC18F6390/6490/8390/8490

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



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



# PIC18F6390/6490/8390/8490

## 4.6 Reset State of Registers

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

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

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

**TABLE 4-3: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER**

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

**Legend:** u = unchanged

**Note 1:** When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (008h or 0018h).

**2:** Reset state is '1' for POR and unchanged for all other Resets when software BOR is enabled (BOREN1:BOREN0 Configuration bits = 01 and SBOREN = 1); otherwise, the Reset state is '0'.

# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TOSU	6X90	8X90	---0 0000	---0 0000	---0 uuuu <sup>(3)</sup>
TOSH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu <sup>(3)</sup>
TOSL	6X90	8X90	0000 0000	0000 0000	uuuu uuuu <sup>(3)</sup>
STKPTR	6X90	8X90	00-0 0000	00-0 0000	uu-u uuuu <sup>(3)</sup>
PCLATU	6X90	8X90	---0 0000	---0 0000	---u uuuu
PCLATH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PCL	6X90	8X90	0000 0000	0000 0000	PC + 2 <sup>(2)</sup>
TBLPTRU	6X90	8X90	--00 0000	--00 0000	--uu uuuu
TBLPTRH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TABLAT	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PRODH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	6X90	8X90	0000 000x	0000 000u	uuuu uuuu <sup>(1)</sup>
INTCON2	6X90	8X90	1111 1111	1111 1111	uuuu uuuu <sup>(1)</sup>
INTCON3	6X90	8X90	1100 0000	1100 0000	uuuu uuuu <sup>(1)</sup>
INDF0	6X90	8X90	N/A	N/A	N/A
POSTINC0	6X90	8X90	N/A	N/A	N/A
POSTDEC0	6X90	8X90	N/A	N/A	N/A
PREINC0	6X90	8X90	N/A	N/A	N/A
PLUSW0	6X90	8X90	N/A	N/A	N/A
FSR0H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR0L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	6X90	8X90	N/A	N/A	N/A
POSTINC1	6X90	8X90	N/A	N/A	N/A
POSTDEC1	6X90	8X90	N/A	N/A	N/A
PREINC1	6X90	8X90	N/A	N/A	N/A
PLUSW1	6X90	8X90	N/A	N/A	N/A

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 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).
  - 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.
  - See Table 4-3 for Reset value for specific condition.
  - Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
  - These registers are cleared on POR and unchanged on BOR.

# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
FSR1H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	6X90	8X90	---- 0000	---- 0000	---- uuuu
INDF2	6X90	8X90	N/A	N/A	N/A
POSTINC2	6X90	8X90	N/A	N/A	N/A
POSTDEC2	6X90	8X90	N/A	N/A	N/A
PREINC2	6X90	8X90	N/A	N/A	N/A
PLUSW2	6X90	8X90	N/A	N/A	N/A
FSR2H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR2L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	6X90	8X90	---x xxxx	---u uuuu	---u uuuu
TMR0H	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TMR0L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
OSCCON	6X90	8X90	0100 q000	0100 00q0	uuuu uugq
HLVDCON	6X90	8X90	0-00 0101	0-00 0101	u-uu uuuu
WDTCON	6X90	8X90	---- ---0	---- ---0	---- ---u
RCON <sup>(4)</sup>	6X90	8X90	0q-1 11q0	0q-q qqqu	uq-u qqqu
TMR1H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	6X90	8X90	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PR2	6X90	8X90	1111 1111	1111 1111	1111 1111
T2CON	6X90	8X90	-000 0000	-000 0000	-uuu uuuu
SSPBUF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPCON1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPCON2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
- 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
- 4:** See Table 4-3 for Reset value for specific condition.
- 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
- 6:** These registers are cleared on POR and unchanged on BOR.

# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
	6X90	8X90			
ADRESH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	6X90	8X90	--00 0000	--00 0000	--uu uuuu
ADCON1	6X90	8X90	--00 0000	--00 0000	--uu uuuu
ADCON2	6X90	8X90	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	6X90	8X90	--00 0000	--00 0000	--uu uuuu
CCPR2H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	6X90	8X90	--00 0000	--00 0000	--uu uuuu
CVRCON	6X90	8X90	000- 0000	000- 0000	uuu- uuuu
CMCON	6X90	8X90	0000 0111	0000 0111	uuuu uuuu
TMR3H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	6X90	8X90	0000 0000	uuuu uuuu	uuuu uuuu
SPBRG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
RCREG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXREG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXSTA1	6X90	8X90	0000 0010	0000 0010	uuuu uuuu
RCSTA1	6X90	8X90	0000 000x	0000 000x	uuuu uuuu
IPR3	6X90	8X90	-111 ----	-111 ----	-uuu ----
PIR3	6X90	8X90	-000 ----	-000 ----	-uuu ---- <sup>(1)</sup>
PIE3	6X90	8X90	-000 ----	-000 ----	-uuu ----
IPR2	6X90	8X90	11-- 1111	11-- 1111	uu-- uuuu
PIR2	6X90	8X90	00-- 0000	00-- 0000	uu-- uuuu <sup>(1)</sup>
PIE2	6X90	8X90	00-- 0000	00-- 0000	uu-- uuuu
IPR1	6X90	8X90	-111 1111	-111 1111	-uuu uuuu
PIR1	6X90	8X90	-000 0000	-000 0000	-uuu uuuu <sup>(1)</sup>
PIE1	6X90	8X90	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	6X90	8X90	00-0 0000	00-0 0000	uu-u uuuu

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
- 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
- 4:** See Table 4-3 for Reset value for specific condition.
- 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
- 6:** These registers are cleared on POR and unchanged on BOR.

# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TRISJ	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISH	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISG	6X90	8X90	---1 1111	---1 1111	---u uuuu
TRISF	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISE	6X90	8X90	1111 ----	1111 ----	uuuu ----
TRISD	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISC	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISB	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISA <sup>(5)</sup>	6X90	8X90	1111 1111 <sup>(5)</sup>	1111 1111 <sup>(5)</sup>	uuuu uuuu <sup>(5)</sup>
LATJ	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	6X90	8X90	---x xxxx	---u uuuu	---u uuuu
LATF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	6X90	8X90	xxxx ----	uuuu ----	uuuu ----
LATD	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA <sup>(5)</sup>	6X90	8X90	xxxx xxxx <sup>(5)</sup>	uuuu uuuu <sup>(5)</sup>	uuuu uuuu <sup>(5)</sup>
PORTJ	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTG	6X90	8X90	--xx xxxx	--uu uuuu	--uu uuuu
PORTF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE	6X90	8X90	xxxx ----	uuuu ----	uuuu ----
PORTD	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA <sup>(5)</sup>	6X90	8X90	xx0x 0000 <sup>(5)</sup>	uu0u 0000 <sup>(5)</sup>	uuuu uuuu <sup>(5)</sup>
SPBRGH1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	6X90	8X90	01-0 0-00	01-0 0-00	uu-u u-uu

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
- 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
- 4:** See Table 4-3 for Reset value for specific condition.
- 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
- 6:** These registers are cleared on POR and unchanged on BOR.



# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets	Wake-up via WDT or Interrupt
				WDT Reset RESET Instruction Stack Resets	
LCDDATA23	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA22	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA21	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA20	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA19	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA18	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA17	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA16	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA15	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA14	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA13	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA12	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA11	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
SPBRG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
RCREG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXREG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXSTA2	6X90	8X90	0000 -010	0000 -010	uuuu -uuu
RCSTA2	6X90	8X90	0000 000x	0000 000x	uuuu uuuu
LCDDATA10	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA9	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA8	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA7	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA6	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA5	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA4	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA3	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA2	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA1	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu
LCDDATA0	6X90	8X90	xxxx xxxx	0000 0000	uuuu uuuu

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
  - 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
  - 4:** See Table 4-3 for Reset value for specific condition.
  - 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
  - 6:** These registers are cleared on POR and unchanged on BOR.

# PIC18F6390/6490/8390/8490

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
	6X90	8X90			
LCDSE5	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDSE4	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDSE3	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDSE2	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDSE1	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDSE0	6X90	8X90	0000 0000	0000 0000 <sup>(6)</sup>	uuuu uuuu
LCDCON	6X90	8X90	000- 0000	000- 0000	uuu- uuuu
LCDPS	6X90	8X90	0000 0000	0000 0000	uuuu uuuu

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

- Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).
- 2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).
- 3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.
- 4:** See Table 4-3 for Reset value for specific condition.
- 5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.
- 6:** These registers are cleared on POR and unchanged on BOR.

# PIC18F6390/6490/8390/8490

## 5.0 MEMORY ORGANIZATION

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

- Program Memory
- Data RAM

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

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

## 5.1 Program Memory Organization

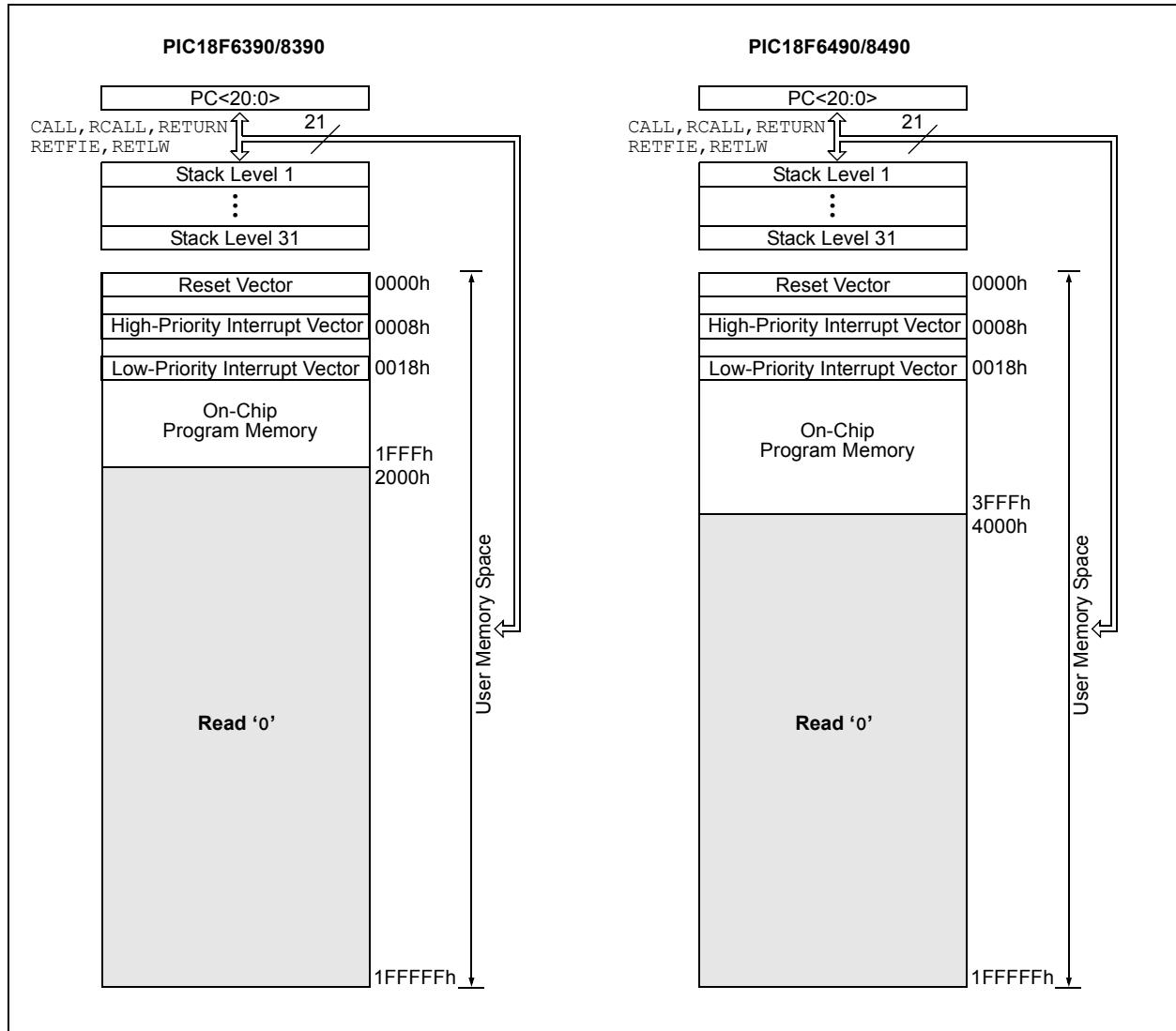
PIC18 microcontrollers implement a 21-bit program counter, which is capable of addressing a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all ‘0’s (a NOP instruction).

The PIC18FX390 have 8 Kbytes of Flash memory and can store up to 4,096 single-word instructions and the PIC18FX490 have 16 Kbytes of Flash memory and can store up to 8,192 single-word instructions.

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

The program memory maps for PIC18F6390/6490/8390/8490 devices are shown in Figure 5-1.

**FIGURE 5-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F6390/6490/8390/8490 DEVICES**



# PIC18F6390/6490/8390/8490

## 5.1.1 PROGRAM COUNTER

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

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

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

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

## 5.1.2 RETURN ADDRESS STACK

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

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer. 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. 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` register are transferred to the PC and then the Stack Pointer is decremented.

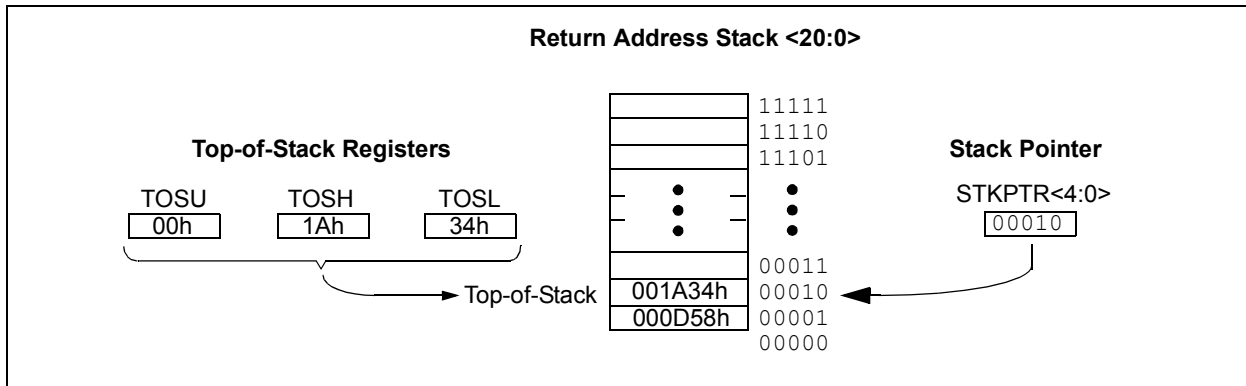
The Stack Pointer is initialized to ‘00000’ after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of ‘00000’; this is only a Reset value. Status bits indicate if the stack is full, has overflowed or has underflowed.

### 5.1.2.1 Top-of-Stack Access

Only the top of the return address stack (TOS) is readable and writable. A set of three registers, `TOSU:TOSH:TOSL`, hold the contents of the stack location pointed to by the lower five bits of the `STKPTR` register (Figure 5-2). This allows users to implement a software stack if necessary. After a `CALL`, `RCALL` or interrupt, the software can read the pushed value by reading the `TOSU:TOSH:TOSL` registers. These values can be placed on a user-defined software stack. At return time, the software can return these values to `TOSU:TOSH:TOSL` and do a return.

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

**FIGURE 5-2: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS**



# PIC18F6390/6490/8390/8490

## 5.1.2.2 Return Stack Pointer (STKPTR)

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

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

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

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

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

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

## 5.1.2.3 PUSH and POP Instructions

Since the Top-of-Stack is readable and writable, the ability to push values onto the stack and pull values off the stack, without disturbing normal program execution, is a desirable feature. The PIC18 instruction set includes two instructions, `PUSH` and `POP`, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

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

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

## REGISTER 5-1: STKPTR: STACK POINTER REGISTER

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

<b>Legend:</b>	C = Clearable only bit
R = Readable bit	W = Writable bit
-n = Value at POR	U = Unimplemented bit, read as '0'
	'1' = Bit is set
	'0' = Bit is cleared
	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    **SP4:SP0:** Stack Pointer Location bits

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

# PIC18F6390/6490/8390/8490

## 5.1.2.4 Stack Full and Underflow Resets

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

## 5.1.3 FAST REGISTER STACK

A Fast Register Stack is provided for the STATUS, WREG and BSR registers, to provide a “fast return” option for interrupts. 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 will 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 and high-priority interrupts are enabled, the stack registers cannot be used reliably to return from low-priority interrupts. If a high-priority interrupt occurs while servicing a low-priority interrupt, the Stack register values stored by the low-priority interrupt will be overwritten. In these cases, users must save the key registers in software during a low-priority interrupt.

If interrupt priority is not used, all interrupts may use the Fast Register Stack for returns from interrupt. If no interrupts are used, the Fast Register Stack can be used to restore the STATUS, WREG and BSR registers at the end of a subroutine call. To use the Fast Register Stack for a subroutine call, a CALL label, FAST instruction must be executed to save the STATUS, WREG and BSR registers to the Fast Register Stack. A RETURN, FAST instruction is then executed to restore these registers from the Fast Register Stack.

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

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

```
CALL SUB1, FAST      ;STATUS, WREG, BSR
                    ;SAVED IN FAST REGISTER
                    ;STACK
    .
    .
SUB1    .
    .
        RETURN FAST  ;RESTORE VALUES SAVED
                    ;IN FAST REGISTER STACK
```

## 5.1.4 LOOK-UP TABLES IN PROGRAM MEMORY

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

- Computed GOTO
- Table Reads

### 5.1.4.1 Computed GOTO

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

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

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

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

### EXAMPLE 5-2: COMPUTED GOTO USING AN OFFSET VALUE

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

### 5.1.4.2 Table Reads

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

Look-up table data may be stored two bytes per program word while programming. The Table Pointer register (TBLPTR) specifies the byte address and the Table Latch register (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”.

# PIC18F6390/6490/8390/8490

## 5.2 PIC18 Instruction Cycle

### 5.2.1 CLOCKING SCHEME

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

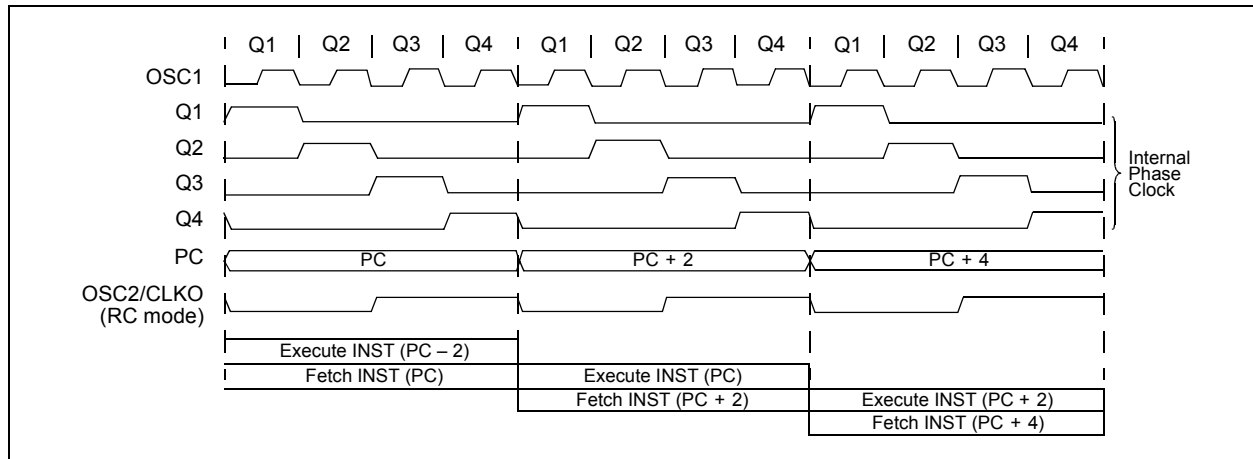
### 5.2.2 INSTRUCTION FLOW/PIPELINING

An "Instruction Cycle" consists of four Q cycles, Q1 through Q4. The instruction fetch and execute are pipelined in such a manner that a fetch takes one instruction cycle, while the decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO), then two cycles are required to complete the instruction (Example 5-3).

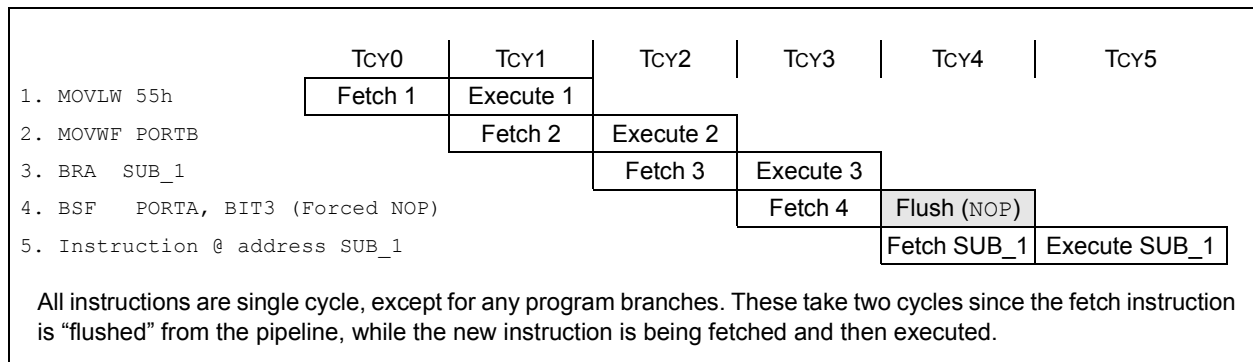
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

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

**FIGURE 5-3: CLOCK/INSTRUCTION CYCLE**



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



# PIC18F6390/6490/8390/8490

## 5.2.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '0' (see Section 5.1.1 "Program Counter").

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

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction #2 in Figure 5-4 shows how the instruction, GOTO 0006h, is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the

number of single-word instructions that the PC will be offset by. Section 24.0 "Instruction Set Summary" provides further details of the instruction set.

## 5.2.4 TWO-WORD INSTRUCTIONS

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

The use of '1111' in the 4 MSBs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence – immediately after the first word – the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 5-4 shows how this works.

**Note:** See Section 5.5 "Program Memory and the Extended Instruction Set" for information on two-word instructions in the extended instruction set.

**FIGURE 5-4: INSTRUCTIONS IN PROGRAM MEMORY**

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

**EXAMPLE 5-4: TWO-WORD INSTRUCTIONS**

CASE 1:			
Object Code	Source Code		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
CASE 2:			
Object Code	Source Code		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code



## 5.3 Data Memory Organization

**Note:** The operation of some aspects of data memory are changed when the PIC18 extended instruction set is enabled. See **Section 5.6 “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; PIC18F6390/6490/8390/8490 devices implement only 4 banks. Figure 5-5 shows the data memory organization for the PIC18F6390/6490/8390/8490 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 (SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to SFRs and the lower portion of GPR Bank 0 without using the BSR. **Section 5.3.2 “Access Bank”** provides a detailed description of the Access RAM.

### 5.3.1 BANK SELECT REGISTER

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

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

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

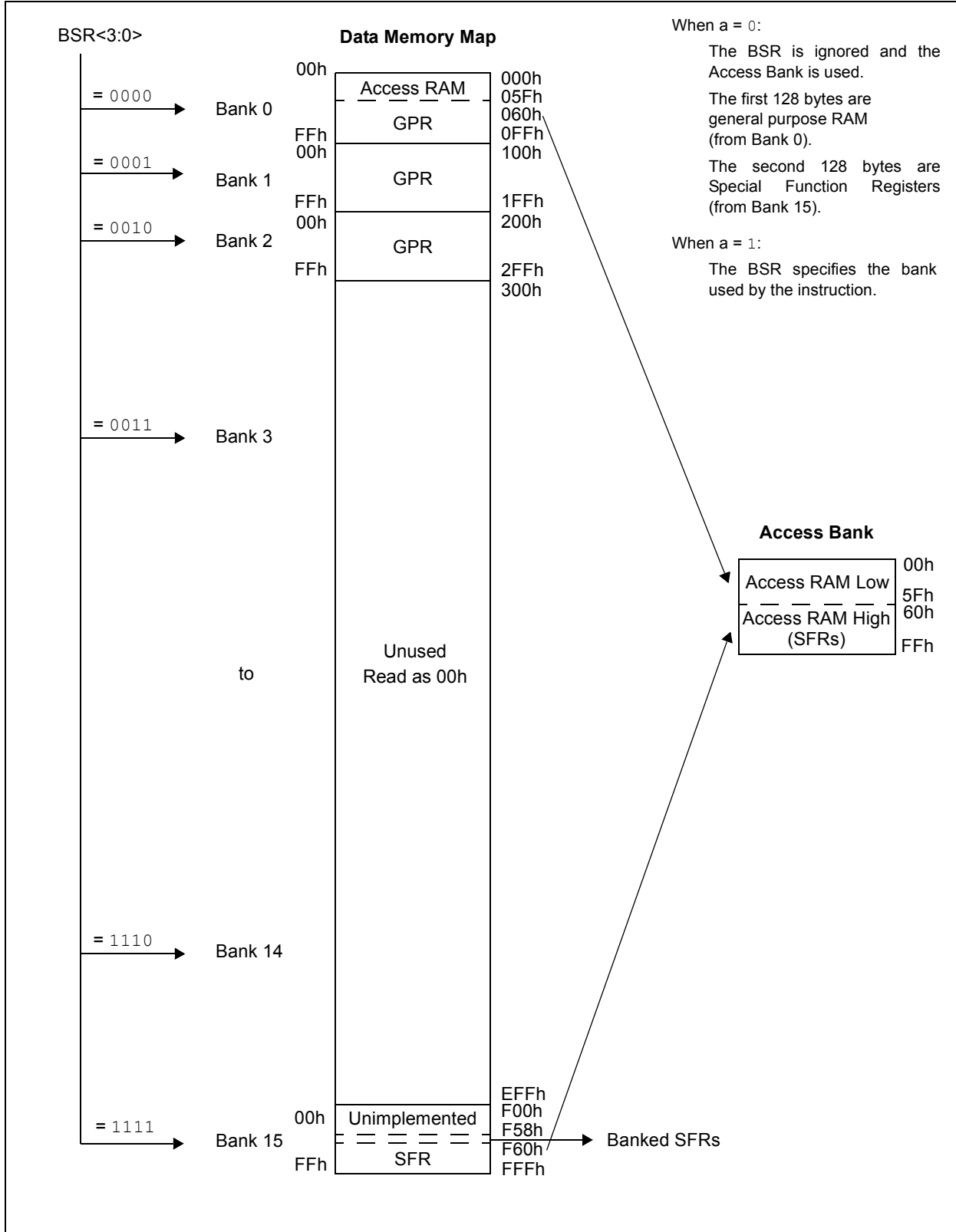
Since up to 16 registers may share the same low-order address, the user must always be careful to ensure that the proper bank is selected before performing a data read or write. For example, writing what should be program data to an 8-bit address of F9h, while the BSR is 0Fh will end up resetting the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

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

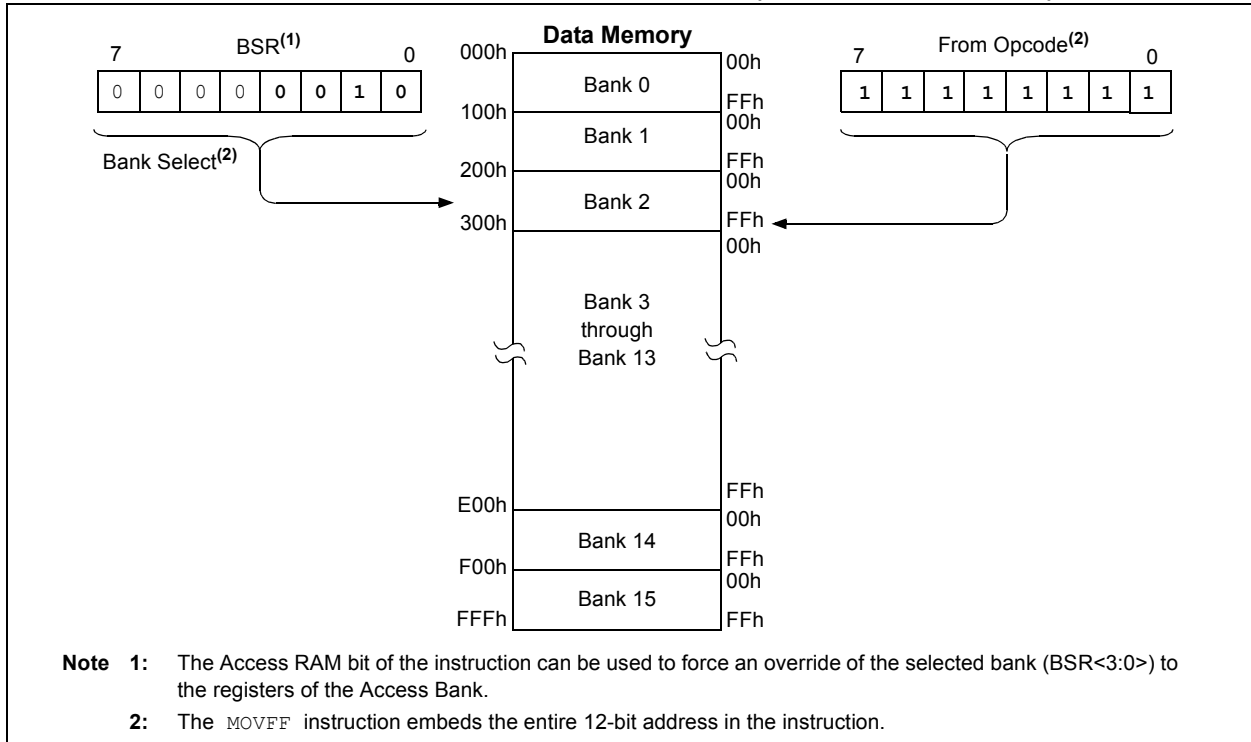
In the core PIC18 instruction set, only the `MOVWF` instruction fully specifies the 12-bit address of the source and target registers. This instruction ignores the BSR completely when it executes. All other instructions include only the low-order address as an operand and must use either the BSR or the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

# PIC18F6390/6490/8390/8490

**FIGURE 5-5: DATA MEMORY MAP FOR PIC18F6390/6490/8390/8490 DEVICES**



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



## 5.3.2 ACCESS BANK

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

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Block 15. The lower half is known as the "Access RAM" and is composed of GPRs. This 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 5-5).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the 'a' parameter in the instruction). When 'a' is equal to '1', the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When 'a' is '0', however, the instruction is forced to use the Access Bank address map; the current value of the BSR is ignored entirely.

Using this "forced" addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 80h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 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 upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

# PIC18F6390/6490/8390/8490

## 5.3.4 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. SFRs start at the top of data memory (FFFh) and extend downward to occupy three-quarters of Bank 15 (from F40h to FFFh). A list of these registers is given in Table 5-1 and Table 5-2.

The SFRs can be classified into two sets: those associated with the “core” device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The Reset and Interrupt registers are described in their respective chapters, while the ALU’s STATUS register is described later in this section. Registers related to the operation of 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.

**TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F6390/6490/8390/8490 DEVICES**

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2 <sup>(1)</sup>	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2 <sup>(1)</sup>	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2 <sup>(1)</sup>	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2 <sup>(1)</sup>	FBCh	CCPR2H	F9Ch	MEMCON <sup>(3)</sup>
FFBh	PCLATU	FDBh	PLUSW2 <sup>(1)</sup>	FBBh	CCPR2L	F9Bh	OSCTUNE
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ <sup>(3)</sup>
FF9h	PCL	FD9h	FSR2L	FB9h	__ <sup>(2)</sup>	F99h	TRISH <sup>(3)</sup>
FF8h	TBLPTRU	FD8h	STATUS	FB8h	__ <sup>(2)</sup>	F98h	TRISG
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	__ <sup>(2)</sup>	F97h	TRISF
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	__ <sup>(2)</sup>	F96h	TRISE
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD
FF4h	PRODH	FD4h	__ <sup>(2)</sup>	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ <sup>(3)</sup>
FF0h	INTCON3	FD0h	RCON	FB0h	__ <sup>(2)</sup>	F90h	LATH <sup>(3)</sup>
FEFh	INDF0 <sup>(1)</sup>	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG
FEEh	POSTINC0 <sup>(1)</sup>	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF
FEDh	POSTDEC0 <sup>(1)</sup>	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE
FECh	PREINC0 <sup>(1)</sup>	FCCh	TMR2	FACH	TXSTA1	F8Ch	LATD
FEBh	PLUSW0 <sup>(1)</sup>	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	__ <sup>(2)</sup>	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	__ <sup>(2)</sup>	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	__ <sup>(2)</sup>	F88h	PORTJ <sup>(3)</sup>
FE7h	INDF1 <sup>(1)</sup>	FC7h	SSPSTAT	FA7h	__ <sup>(2)</sup>	F87h	PORTH <sup>(3)</sup>
FE6h	POSTINC1 <sup>(1)</sup>	FC6h	SSPCON1	FA6h	__ <sup>(2)</sup>	F86h	PORTG
FE5h	POSTDEC1 <sup>(1)</sup>	FC5h	SSPCON2	FA5h	IPR3	F85h	PORTF
FE4h	PREINC1 <sup>(1)</sup>	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE
FE3h	PLUSW1 <sup>(1)</sup>	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

- Note** 1: This is not a physical register.  
 2: Unimplemented registers are read as ‘0’.  
 3: This register is not available on 64-pin devices.  
 4: This register is implemented but unused on 64-pin devices.

# PIC18F6390/6490/8390/8490

**TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F6390/6490/8390/8490 DEVICES (CONTINUED)**

Address	Name	Address	Name	Address	Name	Address	Name
F7Fh	SPBRGH1	F6Fh	SPBRG2	F5Fh	LCDSE5 <sup>(3)</sup>	F4Fh	__ <sup>(2)</sup>
F7Eh	BAUDCON1	F6Eh	RCREG2	F5Eh	LCDSE4 <sup>(3)</sup>	F4Eh	__ <sup>(2)</sup>
F7Dh	__ <sup>(2)</sup>	F6Dh	TXREG2	F5Dh	LCDSE3	F4Dh	__ <sup>(2)</sup>
F7Ch	LCDDATA23 <sup>(4)</sup>	F6Ch	TXSTA2	F5Ch	LCDSE2	F4Ch	__ <sup>(2)</sup>
F7Bh	LCDDATA22 <sup>(4)</sup>	F6Bh	RCSTA2	F5Bh	LCDSE1	F4Bh	__ <sup>(2)</sup>
F7Ah	LCDDATA21	F6Ah	LCDDATA10 <sup>(4)</sup>	F5Ah	LCDSE0	F4Ah	__ <sup>(2)</sup>
F79h	LCDDATA20	F69h	LCDDATA9	F59h	LCDCON	F49h	__ <sup>(2)</sup>
F78h	LCDDATA19	F68h	LCDDATA8	F58h	LCDPS	F48h	__ <sup>(2)</sup>
F77h	LCDDATA18	F67h	LCDDATA7	F57h	__ <sup>(2)</sup>	F47h	__ <sup>(2)</sup>
F76h	LCDDATA17 <sup>(4)</sup>	F66h	LCDDATA6	F56h	__ <sup>(2)</sup>	F46h	__ <sup>(2)</sup>
F75h	LCDDATA16 <sup>(4)</sup>	F65h	LCDDATA5 <sup>(4)</sup>	F55h	__ <sup>(2)</sup>	F45h	__ <sup>(2)</sup>
F74h	LCDDATA15	F64h	LCDDATA4 <sup>(4)</sup>	F54h	__ <sup>(2)</sup>	F44h	__ <sup>(2)</sup>
F73h	LCDDATA14	F63h	LCDDATA3	F53h	__ <sup>(2)</sup>	F43h	__ <sup>(2)</sup>
F72h	LCDDATA13	F62h	LCDDATA2	F52h	__ <sup>(2)</sup>	F42h	__ <sup>(2)</sup>
F71h	LCDDATA12	F61h	LCDDATA1	F51h	__ <sup>(2)</sup>	F41h	__ <sup>(2)</sup>
F70h	LCDDATA11 <sup>(4)</sup>	F60h	LCDDATA0	F50h	__ <sup>(2)</sup>	F40h	__ <sup>(2)</sup>

- Note**
- 1: This is not a physical register.
  - 2: Unimplemented registers are read as '0'.
  - 3: This register is not available on 64-pin devices.
  - 4: This register is implemented but unused on 64-pin devices.

# PIC18F6390/6490/8390/8490

**TABLE 5-2: PIC18F6390/6490/8390/8490 REGISTER FILE SUMMARY**

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	59, 66
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	59, 66
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	59, 66
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					00-0 0000	59, 67
PCLATU	—	—	—	Holding Register for PC<20:16>					---0 0000	59, 66
PCLATH	Holding Register for PC<15:8>								0000 0000	59, 66
PCL	PC Low Byte (PC<7:0>)								0000 0000	59, 66
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	59, 88
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	59, 88
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	59, 88
TABLAT	Program Memory Table Latch								0000 0000	59, 88
PRODH	Product Register High Byte								xxxx xxxx	59, 91
PRODL	Product Register Low Byte								xxxx xxxx	59, 91
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	59, 95
INTCON2	RBPV	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	59, 96
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	59, 97
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	59, 82
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	59, 83
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	59, 83
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	59, 83
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	59, 83
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- xxxx	59, 82
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	59, 82
WREG	Working Register								xxxx xxxx	59
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	59, 82
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	59, 83
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	59, 83
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	59, 83
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	59, 83
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- xxxx	60, 82
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	60, 82
BSR	—	—	—	—	Bank Select Register				---- 0000	60, 71
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	60, 82
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	60, 83
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	60, 83
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	60, 83
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	60, 83
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- xxxx	60, 82
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	60, 82
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	60, 80

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

- Note 1:** The SBOREN bit is only available when the BOREN1:BOREN0 Configuration bits = 01; otherwise it is disabled and reads as '0'. See **Section 4.4 “Brown-out Reset (BOR)”**.
- 2:** These registers and/or bits are not implemented on 64-pin devices; read as '0'.
- 3:** The PLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 2.6.4 “PLL in INTOSC Modes”**.
- 4:** The RG5 bit is only available when Master Clear is disabled (MCLRE Configuration bit = 0); otherwise, RG5 reads as '0'. This bit is read-only.
- 5:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.
- 6:** These registers are implemented but unused in 64-pin devices and may be used as general-purpose data RAM if required.

# PIC18F6390/6490/8390/8490

**TABLE 5-2: PIC18F6390/6490/8390/8490 REGISTER FILE SUMMARY (CONTINUED)**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
TMR0H	Timer0 Register High Byte								0000 0000	60, 132
TMR0L	Timer0 Register Low Byte								xxxx xxxx	60, 132
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	60, 131
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0100 q000	38, 60
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0-00 0101	60, 251
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	60, 288
RCON	IPEN	SBOREN <sup>(1)</sup>	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0q-1 11q0	52, 60, 107
TMR1H	Timer1 Register High Byte								xxxx xxxx	60, 137
TMR1L	Timer1 Register Low Byte								xxxx xxxx	60, 137
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	60, 135
TMR2	Timer2 Register								0000 0000	60, 141
PR2	Timer2 Period Register								1111 1111	60, 141
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	60, 141
SSPBUF	MSSP Receive Buffer/Transmit Register								xxxx xxxx	60, 158, 166
SSPADD	MSSP Address Register in I <sup>2</sup> C™ Slave Mode. MSSP Baud Rate Reload Register in I <sup>2</sup> C Master Mode.								0000 0000	60, 166
SSPSTAT	SMP	CKE	D/ $\overline{A}$	P	S	R/ $\overline{W}$	UA	BF	0000 0000	60, 158, 167
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	60, 159, 168
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	60, 169
ADRESH	A/D Result Register High Byte								xxxx xxxx	61, 240
ADRESL	A/D Result Register Low Byte								xxxx xxxx	61, 240
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/ $\overline{DONE}$	ADON	--00 0000	61, 231
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	61, 232
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	61, 233
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	61, 152, 155
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	61, 152, 155
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	61, 147
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	61, 152, 155
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	61, 152, 155
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	61, 147
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	61, 247
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	61, 241
TMR3H	Timer3 Register High Byte								xxxx xxxx	61, 145
TMR3L	Timer3 Register Low Byte								xxxx xxxx	61, 145
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	61, 143

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

- Note 1:** The SBOREN bit is only available when the BOREN1:BOREN0 Configuration bits = 01; otherwise it is disabled and reads as '0'. See Section 4.4 "Brown-out Reset (BOR)".
- 2:** These registers and/or bits are not implemented on 64-pin devices; read as '0'.
- 3:** The PLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See Section 2.6.4 "PLL in INTOSC Modes".
- 4:** The RG5 bit is only available when Master Clear is disabled (MCLRE Configuration bit = 0); otherwise, RG5 reads as '0'. This bit is read-only.
- 5:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.
- 6:** These registers are implemented but unused in 64-pin devices and may be used as general-purpose data RAM if required.

# PIC18F6390/6490/8390/8490

**TABLE 5-2: PIC18F6390/6490/8390/8490 REGISTER FILE SUMMARY (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:
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								0000 0000	61, 201
RCREG1	EUSART1 Receive Register								0000 0000	61, 208
TXREG1	EUSART1 Transmit Register								0000 0000	61, 206
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	61, 198
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	61, 199
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	-111 ----	61, 106
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	-000 ----	61, 100
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	-000 ----	61, 103
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	11-- 1111	61, 105
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	00-- 0000	61, 99
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	00-- 0000	61, 102
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	-111 1111	61, 104
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	61, 98
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	61, 101
OSCTUNE	INTSRC	PLLEN <sup>(3)</sup>	—	TUN4	TUN3	TUN2	TUN1	TUN0	00-0 0000	35, 61
TRISJ <sup>(2)</sup>	PORTJ Data Direction Register								1111 1111	62, 130
TRISH <sup>(2)</sup>	PORTH Data Direction Register								1111 1111	62, 128
TRISG	—	—	—	PORTG Data Direction Register				---	1111	62, 126
TRISF	PORTF Data Direction Register								1111 1111	62, 124
TRISE	PORTE Data Direction Register				—	—	—	—	1111 ----	62, 121
TRISD	PORTD Data Direction Register								1111 1111	62, 119
TRISC	PORTC Data Direction Register								1111 1111	62, 117
TRISB	PORTB Data Direction Register								1111 1111	62, 114
TRISA	TRISA7 <sup>(5)</sup>	TRISA6 <sup>(5)</sup>	PORTA Data Direction Register					1111 1111	62, 111	
LATJ <sup>(2)</sup>	LATJ Data Output Register								xxxx xxxx	62, 130
LATH <sup>(2)</sup>	LATH Data Output Register								xxxx xxxx	62, 128
LATG	—	—	—	LATG Data Output Register				---x xxxx	62, 126	
LATF	LATF Data Output Register								xxxx xxxx	62, 124
LATE	LATE Data Output Register				—	—	—	—	xxxx ----	62, 121
LATD	LATD Data Output Register								xxxx xxxx	62, 119
LATC	LATC Data Output Register								xxxx xxxx	62, 117
LATB	LATB Data Output Register								xxxx xxxx	62, 114
LATA	LATA7 <sup>(5)</sup>	LATA6 <sup>(5)</sup>	LATA Data Output Register					xxxx xxxx	62, 111	
PORTJ <sup>(2)</sup>	Read PORTJ pins, Write PORTJ Data Latch								xxxx xxxx	62, 130
PORTH <sup>(2)</sup>	Read PORTH pins, Write PORTH Data Latch								xxxx xxxx	62, 128
PORTG	—	—	RG5 <sup>(4)</sup>	Read PORTG pins <4:0>, Write PORTG Data Latch <4:0>					--xx xxxx	62, 126
PORTF	Read PORTF pins, Write PORTF Data Latch								xxxx xxxx	62, 124
PORTE	Read PORTE pins, Write PORTE Data Latch				—	—	—	—	xxxx ----	62, 121
PORTD	Read PORTD pins, Write PORTD Data Latch								xxxx xxxx	62, 119
PORTC	Read PORTC pins, Write PORTC Data Latch								xxxx xxxx	62, 117
PORTB	Read PORTB pins, Write PORTB Data Latch								xxxx xxxx	62, 114
PORTA	RA7 <sup>(5)</sup>	RA6 <sup>(5)</sup>	Read PORTA pins, Write PORTA Data Latch					xx0x 0000	62, 111	

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

- Note 1:** The SBOREN bit is only available when the BOREN1:BOREN0 Configuration bits = 01; otherwise it is disabled and reads as '0'. See **Section 4.4 "Brown-out Reset (BOR)"**.
- 2:** These registers and/or bits are not implemented on 64-pin devices; read as '0'.
- 3:** The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 2.6.4 "PLL in INTOSC Modes"**.
- 4:** The RG5 bit is only available when Master Clear is disabled (MCLRE Configuration bit = 0); otherwise, RG5 reads as '0'. This bit is read-only.
- 5:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.
- 6:** These registers are implemented but unused in 64-pin devices and may be used as general-purpose data RAM if required.



# PIC18F6390/6490/8390/8490

**TABLE 5-2: PIC18F6390/6490/8390/8490 REGISTER FILE SUMMARY (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:
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								0000 0000	62, 201
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	62, 200
LCDDATA23 <sup>(6)</sup>	S47C3	S46C3	S45C3	S44C3	S43C3	S42C3	S41C3	S40C3	xxxx xxxx	63, 261
LCDDATA22 <sup>(6)</sup>	S39C3	S38C3	S37C3	S36C3	S35C3	S34C3	S33C3	S32C3	xxxx xxxx	63, 261
LCDDATA21	S31C3	S30C3	S29C3	S28C3	S27C3	S26C3	S25C3	S24C3	xxxx xxxx	63, 261
LCDDATA20	S23C3	S22C3	S21C3	S20C3	S19C3	S18C3	S17C3	S16C3	xxxx xxxx	63, 261
LCDDATA19	S15C3	S14C3	S13C3	S12C3	S11C3	S10C3	S09C3	S08C3	xxxx xxxx	63, 261
LCDDATA18	S07C3	S06C3	S05C3	S04C3	S03C3	S02C3	S01C3	S00C3	xxxx xxxx	63, 261
LCDDATA17 <sup>(6)</sup>	S47C2	S46C2	S45C2	S44C2	S43C2	S42C2	S41C2	S40C2	xxxx xxxx	63, 261
LCDDATA16 <sup>(6)</sup>	S39C2	S38C2	S37C2	S36C2	S35C2	S34C2	S33C2	S32C2	xxxx xxxx	63, 261
LCDDATA15	S31C2	S30C2	S29C2	S28C2	S27C2	S26C2	S25C2	S24C2	xxxx xxxx	63, 261
LCDDATA14	S23C2	S22C2	S21C2	S20C2	S19C2	S18C2	S17C2	S16C2	xxxx xxxx	63, 261
LCDDATA13	S15C2	S14C2	S13C2	S12C2	S11C2	S10C2	S09C2	S08C2	xxxx xxxx	63, 261
LCDDATA12	S07C2	S06C2	S05C2	S04C2	S03C2	S02C2	S01C2	S00C2	xxxx xxxx	63, 261
LCDDATA11 <sup>(6)</sup>	S47C1	S46C1	S45C1	S44C1	S43C1	S42C1	S41C1	S40C1	xxxx xxxx	63, 261
SPBRG2	AUSART2 Baud Rate Generator Register								0000 0000	63, 220
RCREG2	AUSART2 Receive Register								0000 0000	63, 224
TXREG2	AUSART2 Transmit Register								0000 0000	63, 222
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	63, 218
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	63, 219
LCDDATA10 <sup>(6)</sup>	S39C1	S38C1	S37C1	S36C1	S35C1	S34C1	S33C1	S32C1	xxxx xxxx	63, 261
LCDDATA9	S31C1	S30C1	S29C1	S28C1	S27C1	S26C1	S25C1	S24C1	xxxx xxxx	63, 261
LCDDATA8	S23C1	S22C1	S21C1	S20C1	S19C1	S18C1	S17C1	S16C1	xxxx xxxx	63, 261
LCDDATA7	S15C1	S14C1	S13C1	S12C1	S11C1	S10C1	S09C1	S08C1	xxxx xxxx	63, 261
LCDDATA6	S07C1	S06C1	S05C1	S04C1	S03C1	S02C1	S01C1	S00C1	xxxx xxxx	63, 261
LCDDATA5 <sup>(6)</sup>	S47C0	S46C0	S45C0	S44C0	S43C0	S42C0	S41C0	S40C0	xxxx xxxx	63, 261
LCDDATA4 <sup>(6)</sup>	S39C0	S38C0	S37C0	S36C0	S35C0	S34C0	S33C0	S32C0	xxxx xxxx	63, 261
LCDDATA3	S31C0	S30C0	S29C0	S28C0	S27C0	S26C0	S25C0	S24C0	xxxx xxxx	63, 261
LCDDATA2	S23C0	S22C0	S21C0	S20C0	S19C0	S18C0	S17C0	S16C0	xxxx xxxx	63, 261
LCDDATA1	S15C0	S14C0	S13C0	S12C0	S11C0	S10C0	S09C0	S08C0	xxxx xxxx	63, 261
LCDDATA0	S07C0	S06C0	S05C0	S04C0	S03C0	S02C0	S01C0	S00C0	xxxx xxxx	63, 261
LCDSE5 <sup>(2)</sup>	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	0000 0000	64, 261
LCDSE4 <sup>(2)</sup>	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	0000 0000	64, 260
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	0000 0000	64, 260
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	0000 0000	64, 260
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	0000 0000	64, 260
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	0000 0000	64, 260
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	000- 0000	64, 258
LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0	0000 0000	64, 259

**Legend:** x = unknown, u = unchanged, - = unimplemented,  $\varphi$  = value depends on condition

- Note 1:** The SBOREN bit is only available when the BOREN1:BOREN0 Configuration bits = 01; otherwise it is disabled and reads as '0'. See **Section 4.4 "Brown-out Reset (BOR)"**.
- 2:** These registers and/or bits are not implemented on 64-pin devices; read as '0'.
- 3:** The PLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 2.6.4 "PLL in INTOSC Modes"**.
- 4:** The RG5 bit is only available when Master Clear is disabled (MCLRE Configuration bit = 0); otherwise, RG5 reads as '0'. This bit is read-only.
- 5:** RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.
- 6:** These registers are implemented but unused in 64-pin devices and may be used as general-purpose data RAM if required.

# PIC18F6390/6490/8390/8490

## 5.3.5 STATUS REGISTER

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

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

It is recommended that only `BCF`, `BSF`, `SWAPF`, `MOVFF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions that do not affect Status bits, see the instruction set summaries in Table 24-2 and Table 24-3.

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

### REGISTER 5-2: STATUS REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC <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 4th low-order bit of the result occurred  
0 = No carry-out from the 4th low-order bit of the result

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

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

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

**Note 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.

## 5.4 Data Addressing Modes

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

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

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST Configuration bit = 1). Its operation is discussed in greater detail in **Section 5.6.1 “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 Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 5.3.3 “General**

**Purpose Register File”**), or a location in the Access Bank (**Section 5.3.2 “Access Bank”**) as the data source for the instruction.

The Access RAM bit ‘a’ determines how the address is interpreted. When ‘a’ is ‘1’, the contents of the BSR (**Section 5.3.1 “Bank Select Register”**) 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 op codes. In those cases, the BSR is ignored entirely.

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

### 5.4.3 INDIRECT ADDRESSING

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

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

```

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

# PIC18F6390/6490/8390/8490

## 5.4.3.1 FSR Registers and the INDF Operand

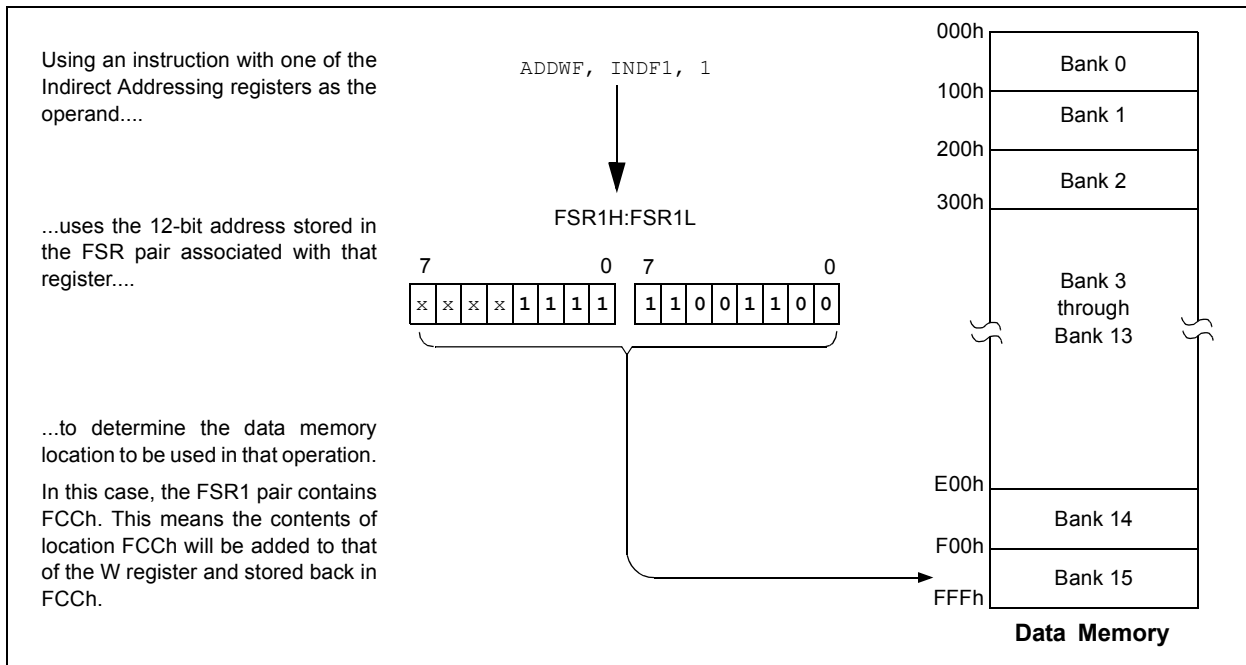
At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. The four upper bits of the FSRnH register are not used, so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of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

Indirect Addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as “virtual” registers: they are

mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction’s target. The INDF operand is just a convenient way of using the pointer.

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

**FIGURE 5-7: INDIRECT ADDRESSING**



## 5.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

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

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

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value offset by 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 **NOP**.

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

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

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

# PIC18F6390/6490/8390/8490

---

## 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`, `MOVWF`, `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 the proper conditions, instructions that use the Access Bank – that is, most bit-oriented and byte-oriented instructions – can invoke a form of Indexed Addressing using an offset specified in the instruction. This special addressing mode is known as Indexed Addressing with Literal Offset, or Indexed Literal Offset mode.

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

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

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

### 5.6.2 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-oriented and bit-oriented instructions, or almost one-half of the standard PIC18 instruction set. Instructions that only use Inherent or Literal Addressing modes are unaffected.

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

Those who desire to use byte-oriented or bit-oriented instructions in the Indexed Literal Offset mode should note the changes to assembler syntax for this mode. This is described in more detail in **Section 24.2.1 “Extended Instruction Syntax”**.

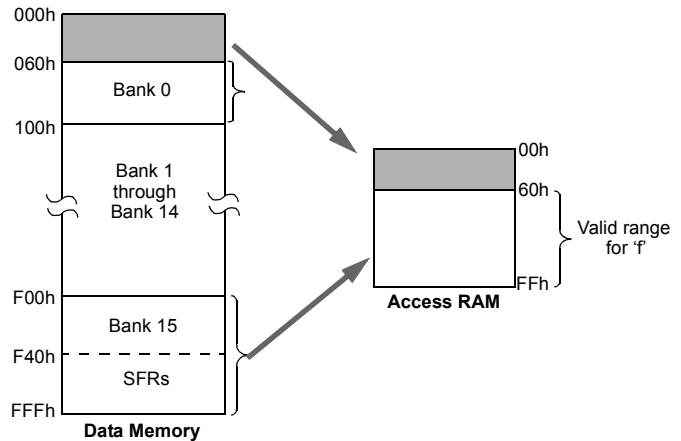
**FIGURE 5-8: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)**

**EXAMPLE INSTRUCTION:** `ADDWF, f, d, a` (Opcode: `0010 01da ffff ffff`)

**When a = 0 and f ≥ 60h:**

The instruction executes in Direct Forced mode. 'f' is interpreted as a location in the Access RAM between 060h and 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.

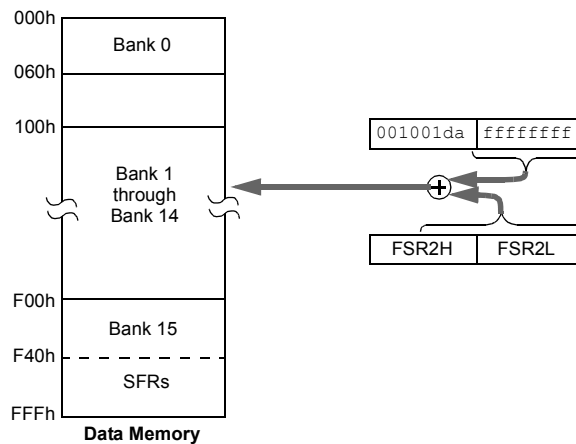


**When a = 0 and f ≤ 5Fh:**

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.

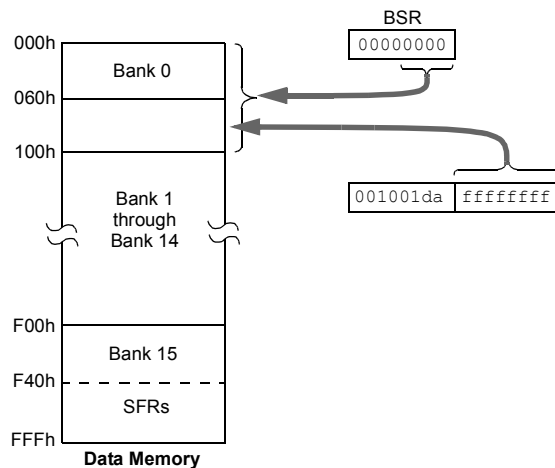
Note that in this mode, the correct syntax is now:

`ADDWF [k], d`  
where 'k' is the same as 'f'.



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

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



# PIC18F6390/6490/8390/8490

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

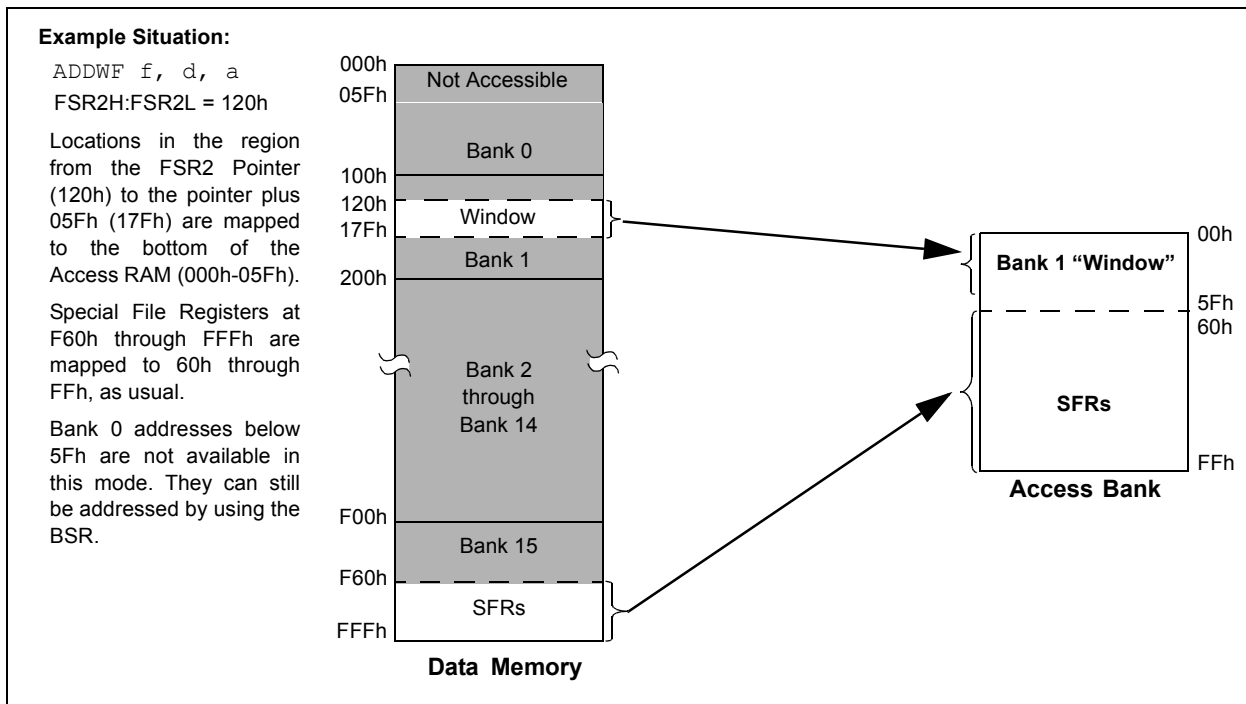
The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined “window” that can be located anywhere in the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

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is ‘1’) will continue to use Direct Addressing as before. Any indirect or indexed 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 5-9: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING**





## 6.0 FLASH PROGRAM MEMORY

In PIC18F6390/6490/8390/8490 devices, the program memory is implemented as read-only Flash memory. It is readable over the entire VDD range during normal operation. A read from program memory is executed on one byte at a time.

### 6.1 Table Reads

For PIC18 devices, there are two operations that allow the processor to move bytes between the program memory space and the data RAM: table read (TBLRD) and table write (TBLWT).

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

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.

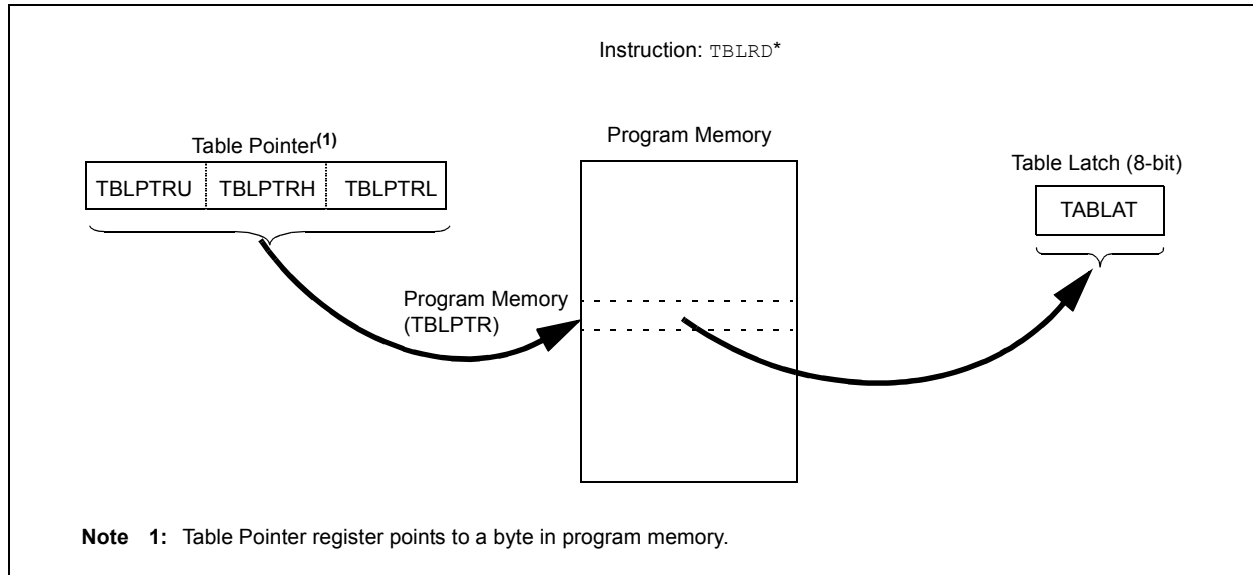
Table reads work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address.

Because the program memory cannot be written to or erased under normal operation, the TBLWT operation is not discussed here.

**Note 1:** Although it cannot be used in PIC18F6390/6490/8390/8490 devices in normal operation, the TBLWT instruction is still implemented in the instruction set. Executing the instruction takes two instruction cycles, but effectively results in a NOP.

**2:** The TBLWT instruction is available only in programming modes and is used during In-Circuit Serial Programming™ (ICSP™).

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



# PIC18F6390/6490/8390/8490

## 6.2 Control Registers

Two control registers are used in conjunction with the `TBLRD` instruction: the `TABLAT` register and the `TBLPTR` register set.

### 6.2.1 TABLE LATCH REGISTER (TABLAT)

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

### 6.2.2 TABLE POINTER REGISTER (TBLPTR)

The Table Pointer register (`TBLPTR`) addresses a byte within the program memory. It is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (`TBLPTRU:TBLPTRH:TBLPTRL`). Only the lower six bits of `TBLPTRU` are used with `TBLPTRH` and `TBLPTRL` to form a 22-bit wide pointer.

The contents of `TBLPTR` indicates a location in program memory space. The low-order 21 bits allow the device to address the full 2 Mbytes of program memory space. The 22nd bit allows access to the configuration space, including the device ID, user ID locations and the Configuration bits.

The `TBLPTR` register set is updated when executing a `TBLRD` in one of four ways, based on the instruction's arguments. These are detailed in Table 6-1. These operations on the `TBLPTR` only affect the low-order 21 bits.

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

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

Example	Operation on Table Pointer
<code>TBLRD*</code>	<code>TBLPTR</code> is not modified
<code>TBLRD*+</code>	<code>TBLPTR</code> is incremented after the read
<code>TBLRD*-</code>	<code>TBLPTR</code> is decremented after the read
<code>TBLRD+*</code>	<code>TBLPTR</code> is incremented before the read

## 6.3 Reading the Flash Program Memory

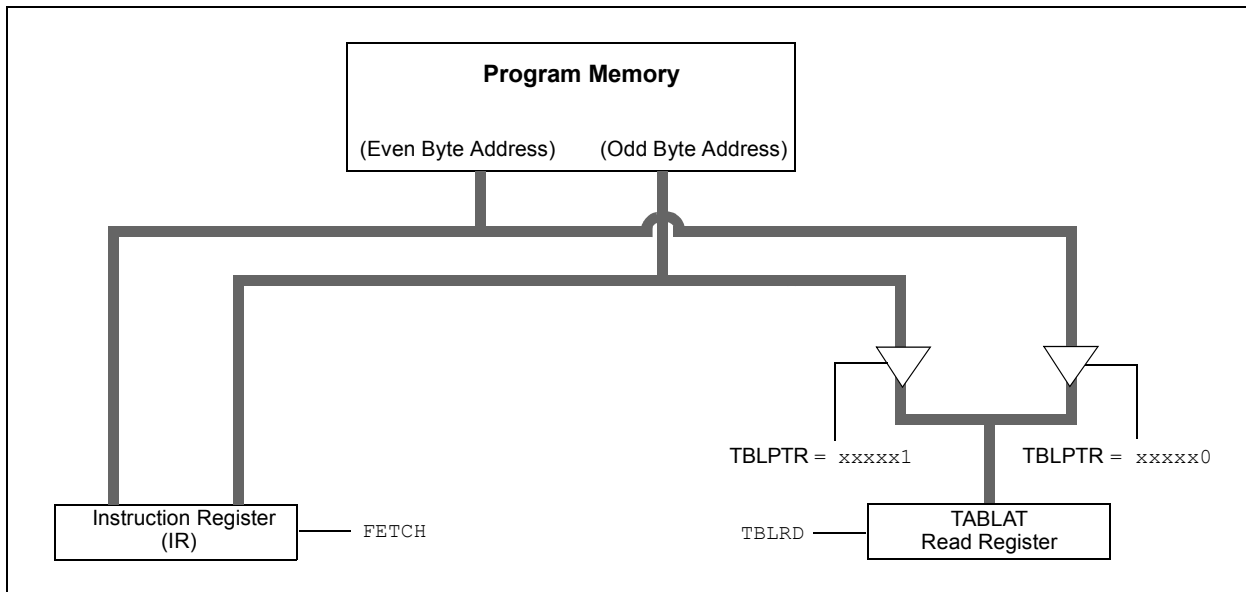
The `TBLRD` instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

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

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 6-2 shows the interface between the internal program memory and the `TABLAT`.

A typical method for reading data from program memory is shown in Example 6-1.

**FIGURE 6-2: READS FROM FLASH PROGRAM MEMORY**



# PIC18F6390/6490/8390/8490

## 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		

**TABLE 6-2: REGISTERS ASSOCIATED WITH READING 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>)					59
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								59
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								59
TABLAT	Program Memory Table Latch								59

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

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 7.0 8 x 8 HARDWARE MULTIPLIER

### 7.1 Introduction

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

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

### 7.2 Operation

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

Example 7-2 shows the sequence to do an 8 x 8 signed multiplication. To account for the signed bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

#### EXAMPLE 7-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

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

#### EXAMPLE 7-2: 8 x 8 SIGNED MULTIPLY ROUTINE

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

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

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

**TABLE 7-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS**

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

# PIC18F6390/6490/8390/8490

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

## EQUATION 7-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

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

## EXAMPLE 7-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```

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

```

Example 7-4 shows the sequence to do a 16 x 16 signed multiply. Equation 7-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the signed 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       ;
;
BTFSS ARG2H, 7       ; ARG2H:ARG2L neg?
BRA SIGN_ARG1        ; no, check ARG1
MOVF ARG1L, W        ;
SUBWF RES2            ;
MOVF ARG1H, W        ;
SUBWFB RES3           ;
;
SIGN_ARG1
BTFSS ARG1H, 7       ; ARG1H:ARG1L neg?
BRA CONT_CODE        ; no, done
MOVF ARG2L, W        ;
SUBWF RES2            ;
MOVF ARG2H, W        ;
SUBWFB RES3           ;
;
CONT_CODE
:

```

## 8.0 INTERRUPTS

The PIC18F6390/6490/8390/8490 devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

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

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

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

In general, interrupt sources have three bits to control their operation. They are:

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

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 0008h or 0018h, depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with 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.

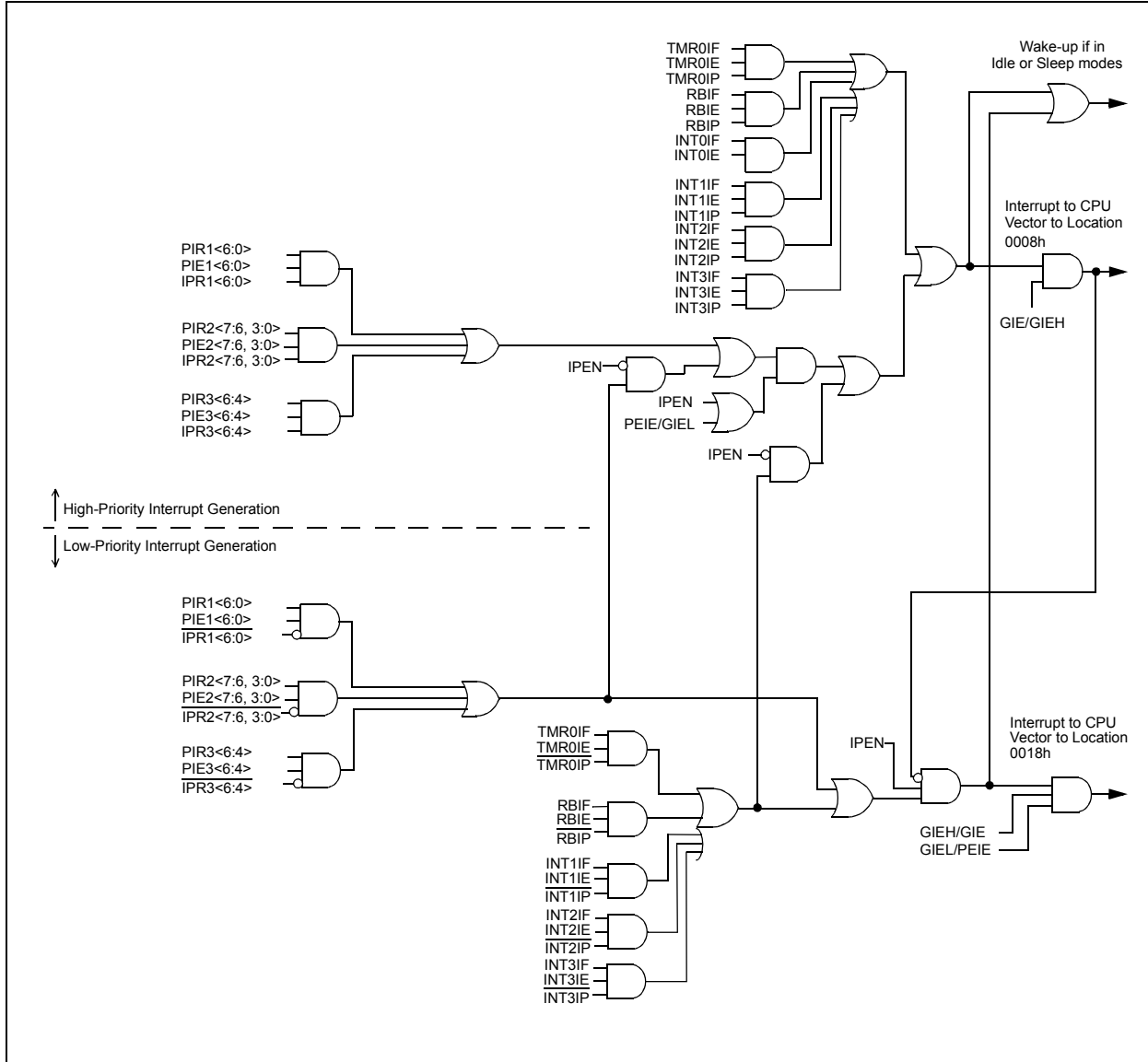
The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

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.

# PIC18F6390/6490/8390/8490

FIGURE 8-1: PIC18F6X90/8X90 INTERRUPT LOGIC





# PIC18F6390/6490/8390/8490

## 8.1 INTCON Registers

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

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

### REGISTER 8-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBFIF <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            **RBFIF:** RB Port Change Interrupt Flag bit<sup>(1)</sup>  
                  1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)  
                  0 = None of the RB7:RB4 pins have changed state

**Note 1:** A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

# PIC18F6390/6490/8390/8490

## REGISTER 8-2: INTCON2: INTERRUPT CONTROL REGISTER 2

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.

# PIC18F6390/6490/8390/8490

## REGISTER 8-3: INTCON3: INTERRUPT CONTROL REGISTER 3

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.

# PIC18F6390/6490/8390/8490

## 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

U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIF	RC1IF	TX1IF	SSPIF	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 **Unimplemented:** Read as '0'

bit 6 **ADIF:** A/D Converter Interrupt Flag bit

1 = An A/D conversion completed (must be cleared in software)  
0 = The A/D conversion is not complete

bit 5 **RC1IF:** EUSART Receive Interrupt Flag bit

1 = The EUSART receive buffer, RCREG1, is full (cleared when RCREG1 is read)  
0 = The EUSART receive buffer is empty

bit 4 **TX1IF:** EUSART Transmit Interrupt Flag bit

1 = The EUSART transmit buffer, TXREG1, is empty (cleared when TXREG1 is written)  
0 = The EUSART transmit buffer is full

bit 3 **SSPIF:** Master Synchronous Serial Port Interrupt Flag bit

1 = The transmission/reception is complete (must be cleared in software)  
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit

#### Capture mode:

1 = A TMR1/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

# PIC18F6390/6490/8390/8490

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

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	CMIF	—	—	BCLIF	HLVDIF	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      **CMIF:** Comparator Interrupt Flag bit  
 1 = Comparator input has changed (must be cleared in software)  
 0 = Comparator input has not changed
- bit 5-4    **Unimplemented:** Read as '0'
- bit 3      **BCLIF:** Bus Collision Interrupt Flag bit  
 1 = A bus collision occurred (must be cleared in software)  
 0 = No bus collision occurred
- bit 2      **HLVDIF:** High/Low-Voltage Detect Interrupt Flag bit  
 1 = A low-voltage condition occurred (must be cleared in software)  
 0 = The device voltage is above the Low-Voltage Detect trip point
- bit 1      **TMR3IF:** TMR3 Overflow Interrupt Flag bit  
 1 = TMR3 register overflowed (must be cleared in software)  
 0 = TMR3 register did not overflow
- bit 0      **CCP2IF:** CCP2 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.

# PIC18F6390/6490/8390/8490

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

U-0	R/W-0	R-0	R/W-0	U-0	U-0	U-0	U-0
—	LCDIF	RC2IF	TX2IF	—	—	—	—
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      **LCDIF:** LCD Interrupt Flag bit (valid when Type-B waveform with Non-Static mode is selected)  
1 = LCD data of all COMs is output (must be cleared in software)  
0 = LCD data of all COMs is not yet output
- bit 5      **RC2IF:** AUSART Receive Interrupt Flag bit  
1 = The AUSART receive buffer, RCREG2, is full (cleared when RCREG2 is read)  
0 = The AUSART receive buffer is empty
- bit 4      **TX2IF:** AUSART Transmit Interrupt Flag bit  
1 = The AUSART transmit buffer, TXREG2, is empty (cleared when TXREG2 is written)  
0 = The AUSART transmit buffer is full
- bit 3-0    **Unimplemented:** Read as '0'

# PIC18F6390/6490/8390/8490

## 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

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIE	RC1IE	TX1IE	SSPIE	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        **Unimplemented:** Read as '0'
- bit 6        **ADIE:** A/D Converter Interrupt Enable bit  
              1 = Enables the A/D interrupt  
              0 = Disables the A/D interrupt
- bit 5        **RC1IE:** EUSART Receive Interrupt Enable bit  
              1 = Enables the EUSART receive interrupt  
              0 = Disables the EUSART receive interrupt
- bit 4        **TX1IE:** EUSART Transmit Interrupt Enable bit  
              1 = Enables the EUSART transmit interrupt  
              0 = Disables the EUSART transmit interrupt
- bit 3        **SSPIE:** Master Synchronous Serial Port Interrupt Enable bit  
              1 = Enables the MSSP interrupt  
              0 = Disables the MSSP interrupt
- bit 2        **CCP1IE:** CCP1 Interrupt Enable bit  
              1 = Enables the CCP1 interrupt  
              0 = Disables the CCP1 interrupt
- bit 1        **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit  
              1 = Enables the TMR2 to PR2 match interrupt  
              0 = Disables the TMR2 to PR2 match interrupt
- bit 0        **TMR1IE:** TMR1 Overflow Interrupt Enable bit  
              1 = Enables the TMR1 overflow interrupt  
              0 = Disables the TMR1 overflow interrupt

# PIC18F6390/6490/8390/8490

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

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	CMIE	—	—	BCLIE	HLVDIE	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      **CMIE:** Comparator Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 5-4    **Unimplemented:** Read as '0'
- bit 3      **BCL1IE:** Bus Collision Interrupt Enable bit  
1 = Enabled  
0 = Disabled
- bit 2      **HLVDIE:** 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:** CCP2 Interrupt Enable bit  
1 = Enabled  
0 = Disabled



# PIC18F6390/6490/8390/8490

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

U-0	R/W-0	R-0	R-0	U-0	U-0	U-0	U-0
—	LCDIE	RC2IE	TX2IE	—	—	—	—
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        **LCDIE:** LCD Interrupt Enable bit (valid when Type-B waveform with Non-Static mode is selected)
  - 1 = Enabled
  - 0 = Disabled
- bit 5        **RC2IE:** AUSART Receive Interrupt Enable bit
  - 1 = Enabled
  - 0 = Disabled
- bit 4        **TX2IE:** AUSART Transmit Interrupt Enable bit
  - 1 = Enabled
  - 0 = Disabled
- bit 3-0      **Unimplemented:** Read as '0'

# PIC18F6390/6490/8390/8490

## 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

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	ADIP	RC1IP	TX1IP	SSPIP	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	<b>Unimplemented:</b> Read as '0'
bit 6	<b>ADIP:</b> A/D Converter Interrupt Priority bit 1 = High priority 0 = Low priority
bit 5	<b>RC1IP:</b> EUSART Receive Interrupt Priority bit 1 = High priority 0 = Low priority
bit 4	<b>TX1IP:</b> EUSART Transmit Interrupt Priority bit 1 = High priority 0 = Low priority
bit 3	<b>SSPIP:</b> Master Synchronous Serial Port Interrupt Priority bit 1 = High priority 0 = Low priority
bit 2	<b>CCP1IP:</b> CCP1 Interrupt Priority bit 1 = High priority 0 = Low priority
bit 1	<b>TMR2IP:</b> TMR2 to PR2 Match Interrupt Priority bit 1 = High priority 0 = Low priority
bit 0	<b>TMR1IP:</b> TMR1 Overflow Interrupt Priority bit 1 = High priority 0 = Low priority

# PIC18F6390/6490/8390/8490

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

R/W-1	R/W-1	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	CMIP	—	—	BCLIP	HLVDIP	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      **CMIP:** Comparator Interrupt Priority bit  
          1 = High priority  
          0 = Low priority
- bit 5-4    **Unimplemented:** Read as '0'
- bit 3      **BCLIP:** Bus Collision Interrupt Priority bit  
          1 = High priority  
          0 = Low priority
- bit 2      **HLVDIP:** 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:** CCP2 Interrupt Priority bit  
          1 = High priority  
          0 = Low priority

# PIC18F6390/6490/8390/8490

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

U-0	R/W-1	R/W-1	R/W-1	U-0	U-0	U-0	U-0
—	LCDIP	RC2IP	TX2IP	—	—	—	—
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 **LCDIP:** LCD Interrupt Priority bit (valid when Type-B waveform with Non-Static mode is selected)

1 = High priority

0 = Low priority

bit 5 **RC2IP:** AUSART Receive Interrupt Priority bit

1 = High priority

0 = Low priority

bit 4 **TX2IP:** AUSART Transmit Interrupt Priority bit

1 = High priority

0 = Low priority

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

# PIC18F6390/6490/8390/8490

## 8.5 RCON Register

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

**REGISTER 8-13: RCON: RESET CONTROL REGISTER**

R/W-0	R/W-1	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
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      **SBOREN:** Software BOR Enable bit  
For details of bit operation and Reset state, see Register 4-1.
- bit 5      **Unimplemented:** Read as '0'
- bit 4       **$\overline{\text{RI}}$ :**  $\overline{\text{RESET}}$  Instruction Flag bit  
For details of bit operation, see Register 4-1.
- bit 3       **$\overline{\text{TO}}$ :** Watchdog Timer Time-out Flag bit  
For details of bit operation, see Register 4-1.
- bit 2       **$\overline{\text{PD}}$ :** Power-Down Detection Flag bit  
For details of bit operation, see Register 4-1.
- bit 1       **$\overline{\text{POR}}$ :** Power-on Reset Status bit  
For details of bit operation, see Register 4-1.
- bit 0       **$\overline{\text{BOR}}$ :** Brown-out Reset Status bit  
For details of bit operation, see Register 4-1.

# PIC18F6390/6490/8390/8490

## 8.6 INTx Pin Interrupts

External interrupts on the RB0/INT0, RB1/INT1, RB2/INT2 and RB3/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 RBx/INTx pin, the corresponding flag bit INTxIF is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxIE. The interrupt flag bit 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 10.0 “Timer0 Module”** for further details on the Timer0 module.

## 8.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

## 8.9 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (see **Section 5.3 “Data Memory Organization”**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine. Depending on the user’s application, other registers may also need to be saved. Example 8-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

### EXAMPLE 8-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF    W_TEMP                ; W_TEMP is in virtual bank
MOVFF    STATUS, STATUS_TEMP    ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP          ; BSR_TEMP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR          ; Restore BSR
MOVFF    W_TEMP, W              ; Restore WREG
MOVFF    STATUS_TEMP, STATUS     ; Restore STATUS
```

## 9.0 I/O PORTS

Depending on the device selected and features enabled, there are up to nine ports available. Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

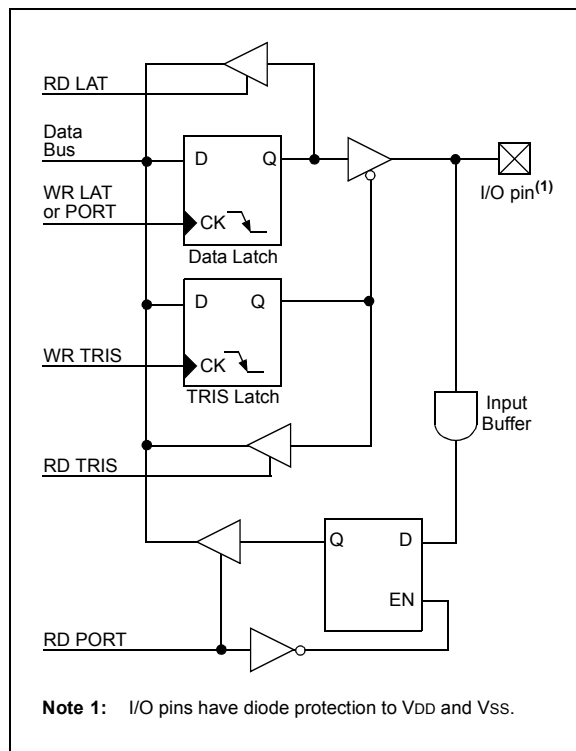
Each port has three registers for its operation. These registers are:

- TRIS register (data direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (output latch)

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

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

**FIGURE 9-1: GENERIC I/O PORT OPERATION**



## 9.1 PORTA, TRISA and LATA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch.

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

The RA4 pin is multiplexed with the Timer0 module clock input and the LCD segment drive to become the RA4/T0CKI/SEG14 pin. Pins RA6 and RA7 are multiplexed with the main oscillator pins; they are enabled as oscillator or I/O pins by the selection of the main oscillator in the Configuration register (see **Section 23.1 “Configuration Bits”** for details). When they are not used as port pins, RA6 and RA7 and their associated TRIS and LAT bits are read as '0'.

The other PORTA pins are multiplexed with analog inputs and the analog VREF+ and VREF- inputs. The operation of pins RA3:RA0 and RA5 as A/D converter inputs is selected by clearing or setting the control bits in the ADCON1 register (A/D Control Register 1).

The RA4/T0CKI/SEG14 pin is a Schmitt Trigger input and an open-drain output. All other PORTA pins have TTL input levels and full CMOS output drivers.

The TRISA register controls the direction of the 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.

RA5:RA2 are also multiplexed with LCD segment drives controlled by bits in the LCDSE1 and LCDSE2 registers. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-1: 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
    
```

# PIC18F6390/6490/8390/8490

**TABLE 9-1: PORTA FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RA0/AN0	RA0	0	O	DIG	LATA<0> data output. Not affected by analog pin setting.
		1	I	TTL	PORTA<0> data input. Reads '0' on POR.
	AN0	1	I	ANA	A/D input channel 0. Default configuration on POR.
RA1/AN1	RA1	0	O	DIG	LATA<1> data output. Not affected by analog pin setting.
		1	I	TTL	PORTA<1> data input. Reads '0' on POR.
	AN1	1	I	ANA	A/D input channel 1. Default configuration on POR.
RA2/AN2/VREF-/SEG16	RA2	0	O	DIG	LATA<2> data output. Not affected by analog pin setting; disabled when LCD segment enabled.
		1	I	TTL	PORTA<2> data input. Reads '0' on POR.
	AN2	1	I	ANA	A/D input channel 2. Default configuration on POR.
	VREF-	1	I	ANA	A/D low reference voltage input.
	SEG16	x	O	ANA	Segment 16 analog output for LCD.
RA3/AN3/VREF+/SEG17	RA3	0	O	DIG	LATA<3> data output. Output is unaffected by analog pin setting; disabled when LCD segment enabled.
		1	I	TTL	PORTA<3> data input. Reads '0' on POR.
	AN3	1	I	ANA	A/D input channel 3. Default configuration on POR.
	VREF+	1	I	ANA	A/D high reference voltage input.
	SEG17	x	O	ANA	Segment 17 analog output for LCD. Disables all other digital outputs.
RA4/T0CKI/SEG14	RA4	0	O	DIG	LATA<4> data output; disabled when LCD segment enabled.
		1	I	ST	PORTA<4> data input.
	T0CKI		I	ST	Timer0 clock input.
	SEG14	x	O	ANA	Segment 14 analog output for LCD.
RA5/AN4/HLVDIN/SEG15	RA5	0	O	DIG	LATA<5> data output. Not affected by analog pin setting; disabled when LCD segment enabled.
		1	I	TTL	PORTA<5> data input. Reads '0' on POR.
	AN4	1	I	ANA	A/D input channel 5. Default configuration on POR.
	HLVDIN	1	I	ANA	High/Low-Voltage Detect external trip point input.
OSC2/CLKO/RA6	OSC2	x	O	ANA	Main oscillator feedback output connection (XT, HS and LP modes).
	CLKO	x	O	DIG	System cycle clock output (FOSC/4) in all oscillator modes except RCIO, INTIO2 and ECIO.
	RA6	0	O	DIG	LATA<6> data output. Enabled in RCIO, INTIO2 and ECIO modes only.
		1	I	TTL	PORTA<6> data input. Enabled in RCIO, INTIO2 and ECIO modes only.
OSC1/CLKI/RA7	OSC1	x	I	ANA	Main oscillator input connection, all modes except INTIO.
	CLKI	x	I	ANA	Main clock input connection, all modes except INTIO.
	RA7	0	O	DIG	LATA<7> data output. Available only in INTIO modes; otherwise reads as '0'.
		1	I	TTL	PORTA<7> data input. Available only in INTIO modes; otherwise reads as '0'.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).



# PIC18F6390/6490/8390/8490

**TABLE 9-2: 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 <sup>(1)</sup>	RA6 <sup>(1)</sup>	RA5	RA4	RA3	RA2	RA1	RA0	62
LATA	LATA7 <sup>(1)</sup>	LATA6 <sup>(1)</sup>	LATA Data Output Register						62
TRISA	TRISA7 <sup>(1)</sup>	TRISA6 <sup>(1)</sup>	PORTA Data Direction Register						62
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64

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

**Note 1:** RA7:RA6 and their associated latch and data direction bits are enabled as I/O pins based on oscillator configuration; otherwise, they are read as '0'.

# PIC18F6390/6490/8390/8490

## 9.2 PORTB, TRISB and LATB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

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

### EXAMPLE 9-2: INITIALIZING PORTB

```
CLRF   PORTB   ; Initialize PORTB by
              ; clearing output
              ; data latches
CLRF   LATB    ; Alternate method
              ; to clear output
              ; data latches
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  $\overline{\text{RBP}}\text{U}$  (INTCON2<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

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

This interrupt can wake the device from power-managed modes. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

- a) Any read or write of PORTB (except with the MOVFF (ANY), PORTB instruction). This will end the mismatch condition.
- b) Clear flag bit, RBIF.

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

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

RB4:RB1 are also multiplexed with LCD segment drives controlled by bits in the LCDSE1 register. I/O port functions are only available when the segments are disabled.

# PIC18F6390/6490/8390/8490

**TABLE 9-3: PORTB FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RB0/INT0	RB0	0	O	DIG	LATB<0> data output.
		1	I	TTL	PORTB<0> data input; programmable weak pull-up.
	INT0	1	I	ST	External interrupt 0 input.
RB1/INT1/SEG8	RB1	0	O	DIG	LATB<1> data output; disabled when LCD segment enabled.
		1	I	TTL	PORTB<1> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	INT1	1	I	ST	External interrupt 1 input.
RB2/INT2/SEG9	RB2	0	O	DIG	LATB<2> data output; disabled when LCD segment enabled.
		1	I	TTL	PORTB<2> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	INT2	1	I	ST	External interrupt 2 input.
RB3/INT3/SEG10	RB3	0	O	DIG	LATB<3> data output; disabled when LCD segment enabled.
		1	I	TTL	PORTB<3> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	INT3	1	I	ST	External interrupt 3 input.
RB4/KBI0/SEG11	RB4	0	O	DIG	LATB<4> data output; disabled when LCD segment enabled.
		1	I	TTL	PORTB<4> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI0	1	I	TTL	Interrupt-on-pin change.
RB5/KBI1	RB5	0	O	DIG	LATB<5> data output; disabled when LCD segment enabled.
		1	I	TTL	PORTB<5> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI1	1	I	TTL	Interrupt-on-pin change.
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> data output; unavailable when ICD or ICSP™ enabled.
		1	I	TTL	PORTB<6> data input; unavailable when ICD or ICSP enabled.
	KBI2	1	I	TTL	Interrupt-on-pin change; unavailable when ICD or ICSP enabled.
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> data output; unavailable when ICD or ICSP enabled.
		1	I	TTL	PORTB<7> data input; unavailable when ICD or ICSP enabled.
	KBI3	1	I	TTL	Interrupt-on-pin change; unavailable when ICD or ICSP enabled.
PGD	x	O	DIG	Serial execution data output for ICSP and ICD operation. <sup>(1)</sup>	
	x	I	ST	Serial execution data input for ICSP and ICD operation. <sup>(1)</sup>	

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**Note 1:** All other pin functions are disabled when ICSP or ICD are enabled.

# PIC18F6390/6490/8390/8490

**TABLE 9-4: 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	62
LATB	LATB Data Output Register								62
TRISB	PORTB Data Direction Register								62
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
INTCON2	RBP $\bar{U}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	59
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	59
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64

**Legend:** Shaded cells are not used by PORTB.

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

PORTC is multiplexed with several peripheral functions (Table 9-5). The pins have Schmitt Trigger input buffers. RC1 is normally configured by Configuration bit, CCP2MX, as the default peripheral pin of the CCP2 module (default/erased state, CCP2MX = 1).

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTC pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

**Note:** On a Power-on Reset, these pins are configured as digital inputs.

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

RC2 and RC5 are also multiplexed with LCD segment drives controlled by bits in the LCDSE1 register. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-3: INITIALIZING PORTC

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

# PIC18F6390/6490/8390/8490

**TABLE 9-5: PORTC FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RC0/T1OSO/ T13CKI/	RC0	0	O	DIG	LATC<0> data output; disabled when Timer1 oscillator is used.
		1	I	ST	PORTC<0> data input; disabled when Timer1 oscillator is used.
	T1OSO	x	O	ANA	Timer1 oscillator output.
	T13CKI	x	I	ST	Timer1/Timer3 clock input.
RC1/T1OSI/ CCP2	RC1	0	O	DIG	LATC<1> data output; disabled when Timer1 oscillator is used.
		1	I	ST	PORTC<1> data input; disabled when Timer1 oscillator is used.
	T1OSI	x	I	ANA	Timer1 oscillator input.
	CCP2 <sup>(1)</sup>	0	O	DIG	CCP2 compare output or PWM output; takes priority over digital I/O data.
1		I	ST	CCP2 capture input.	
RC2/CCP1/ SEG13	RC2	0	O	DIG	LATC<2> data output; disabled when LCD segment enabled.
		1	I	ST	PORTC<2> data input.
	CCP1	0	O	DIG	CCP1 compare output or PWM output; takes priority over digital I/O data.
		1	I	ST	CCP1 capture input.
SEG13	x	O	ANA	Segment 13 analog output for LCD.	
RC3/SCK/SCL	RC3	0	O	DIG	LATC<3> data output.
		1	I	ST	PORTC<3> data input.
	SCK	0	O	DIG	SPI clock output (MSSP module); takes priority over port data.
		1	I	ST	SPI clock input (MSSP module).
	SCL	0	O	DIG	I <sup>2</sup> C™ clock output (MSSP module); takes priority over port data.
		1	I	ST	I <sup>2</sup> C clock input (MSSP module); input type depends on module setting.
RC4/SDI/SDA	RC4	0	O	DIG	LATC<4> data output.
		1	I	ST	PORTC<4> data input.
	SDI	1	I	ST	SPI data input (MSSP module).
	SDA	1	O	DIG	I <sup>2</sup> C data output (MSSP module); takes priority over port data.
		1	I	ST	I <sup>2</sup> C data input (MSSP module); input type depends on module setting.
RC5/SDO/ SEG12	RC5	0	O	DIG	LATC<5> data output; disabled when LCD segment enabled.
		1	I	ST	PORTC<5> data input.
	SDO	0	O	DIG	SPI data output (MSSP module); takes priority over port data.
	SEG12	x	O	ANA	Segment 12 analog output for LCD.
RC6/TX1/CK1	RC6	0	O	DIG	LATC<6> data output.
		1	I	ST	PORTC<6> data input.
	TX1	1	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
		1	O	DIG	Synchronous serial data input (EUSART module). User must configure as an input.
RC7/RX1/DT1	RC7	0	O	DIG	LATC<7> data output.
		1	I	ST	PORTC<7> data input.
	RX1	1	I	ST	Asynchronous serial receive data input (EUSART module).
	DT1	1	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
1		I	ST	Synchronous serial data input (EUSART module). User must configure as an input.	

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**Note 1:** Default assignment for CCP2 (CCP2MX Configuration bit = 1).

# PIC18F6390/6490/8390/8490

**TABLE 9-6: 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	62
LATC	LATC Data Output Register								62
TRISC	PORTC Data Direction Register								62
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64

**Legend:** Shaded cells are not used by PORTC.

# PIC18F6390/6490/8390/8490

## 9.4 PORTD, TRISD and LATD Registers

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 Power-on Reset, these pins are configured as digital inputs.

PORTD is also multiplexed with LCD segment drives controlled by the LCDSE0 register. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-4: 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
```

**TABLE 9-7: PORTD FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RD0/SEG0	RD0	0	O	DIG	LATD<0> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<0> data input.
	SEG0	x	O	ANA	Segment 0 analog output for LCD.
RD1/SEG1	RD1	0	O	DIG	LATD<1> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<1> data input.
	SEG1	x	O	ANA	Segment 1 analog output for LCD.
RD2/SEG2	RD2	0	O	DIG	LATD<2> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<2> data input.
	SEG2	x	O	ANA	Segment 2 analog output for LCD.
RD3/SEG3	RD3	0	O	DIG	LATD<3> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<3> data input.
	SEG3	x	O	ANA	Segment 3 analog output for LCD.
RD4/SEG4	RD4	0	O	DIG	LATD<4> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<4> data input.
	SEG4	x	O	ANA	Segment 4 analog output for LCD module.
RD5/SEG5	RD5	0	O	DIG	LATD<5> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<5> data input.
	SEG5	x	O	ANA	Segment 5 analog output for LCD.
RD6/SEG6	RD6	0	O	DIG	LATD<6> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<6> data input.
	SEG6	x	O	ANA	Segment 6 analog output for LCD.
RD7/SEG7	RD7	0	O	DIG	LATD<7> data output; disabled when LCD segment enabled.
		1	I	ST	PORTD<7> data input.
	SEG7	x	O	ANA	Segment 7 analog output for LCD.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).



# PIC18F6390/6490/8390/8490

**TABLE 9-8: 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	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	62
LATD	LATD Data Output Register								62
TRISD	PORTD Data Direction Register								62
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	64

# PIC18F6390/6490/8390/8490

## 9.5 PORTE, TRISE and LATE Registers

PORTE is a 4-bit wide, bidirectional port. 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).

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.

All pins on PORTE are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

**Note:** On a Power-on Reset, these pins are configured as digital inputs.

Pins RE6:RE4 are multiplexed with three of the LCD common drives. I/O port functions are only available on those PORTE pins, depending on which commons are active. The configuration is determined by the LMUX1:LMUX0 control bits (LCDCON<1:0>). The availability is summarized in Table 9-9.

RE7 is also multiplexed with LCD segment drive (SEG31) controlled by the LCDSE3<7> bit. I/O port function is only available when the segment is disabled.

**Note:** The pins corresponding to RE2:RE0 of other PIC18F parts have the function of LCDBIAS3:LCDBIAS1 and the pin corresponding to RE3 of other PIC18F parts has the function of COM0. These four pins cannot be used as digital I/O.

RE7 also can be configured as the alternate peripheral pin for the CCP2 module. This is done by clearing the CCP2MX Configuration bit.

**TABLE 9-9: PORTE PINS AVAILABLE IN DIFFERENT LCD DRIVE CONFIGURATIONS**

LCDCON <1:0>	Active LCD Commons	PORTE Available for I/O
00	COM0	RE6, RE5, RE4
01	COM0, COM1	RE6, RE5
10	COM0, COM1 and COM2	RE6
11	All (COM0 through COM3)	None

**EXAMPLE 9-5: INITIALIZING PORTE**

```
CLRF   PORTE   ; Initialize PORTE by
              ; clearing output
              ; data latches
CLRF   LATE    ; Alternate method
              ; to clear output
              ; data latches
MOVLW  30h    ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISE   ; Set RE<5:4> as inputs
              ; RE<7:6> as outputs
```

# PIC18F6390/6490/8390/8490

**TABLE 9-10: PORTE FUNCTIONS**

Pad Name	Function	TRIS Setting	I/O	Buffer	Description
RE4/COM1	RE4	0	O	DIG	LATE<4> data output; disabled when LCD common enabled.
		1	I	ST	PORTE<4> data input.
	COM1	x	O	ANA	Common 1 analog output for LCD.
RE5/COM2	RE5	0	O	DIG	LATE<5> data output; disabled when LCD common enabled.
		1	I	ST	PORTE<5> data input.
	COM2	x	O	ANA	Common 2 analog output for LCD.
RE6/COM3	RE6	0	O	DIG	LATE<6> data output; disabled when LCD segment enabled.
		1	I	ST	PORTE<6> data input.
	COM3	x	O	ANA	Common 3 analog output for LCD.
RE7/CCP2/ SEG31	RE7	0	O	DIG	LATE<7> data output; disabled when LCD segment enabled.
		1	I	ST	PORTE<7> data input.
	CCP2 <sup>(1)</sup>	0	O	DIG	CCP2 compare output and CCP2 PWM output; takes priority over port data.
		1	I	ST	CCP2 capture input.
	SEG31	x	O	ANA	Segment 31 analog output for LCD.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**Note 1:** Alternate assignment for CCP2 when the CCP2MX Configuration bit = 0.

**TABLE 9-11: 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	RE7	RE6	RE5	RE4	—	—	—	—	62
LATE	LATE Data Output Register				—	—	—	—	62
TRISE	PORTE Data Direction Register				—	—	—	—	62
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

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

# PIC18F6390/6490/8390/8490

## 9.6 PORTF, LATF and TRISF Registers

PORTF is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISF. Setting a TRISF bit (= 1) will make the corresponding PORTF pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISF bit (= 0) will make the corresponding PORTF pin an output (i.e., put the contents of the output latch on the selected pin).

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

All pins on PORTF are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

PORTF is multiplexed with several analog peripheral functions, including the A/D converter inputs and comparator inputs, outputs and voltage reference.

**Note 1:** On a Power-on Reset, the RF6:RF0 pins are configured as inputs and read as '0'.

**2:** To configure PORTF as digital I/O, turn off comparators and set ADCON1 value.

PORTF is also multiplexed with LCD segment drives controlled by bits in the LCDSE2 and LCDSE3 registers. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-6: INITIALIZING PORTF

```
CLRF   PORTF   ; Initialize PORTF by
              ; clearing output
              ; data latches
CLRF   LATF    ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0x07   ;
MOVWF  CMCON  ; Turn off comparators
MOVLW  0x0F   ;
MOVWF  ADCON1 ; Set PORTF as digital I/O
MOVLW  0xCF   ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISF  ; Set RF3:RF0 as inputs
              ; RF5:RF4 as outputs
              ; RF7:RF6 as inputs
```

# PIC18F6390/6490/8390/8490

**TABLE 9-12: PORTF FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RF0/AN5/ SEG18	RF0	0	O	DIG	LATF<0> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<0> data input. Reads '0' on POR.
	AN5	1	I	ANA	A/D input channel 5. Default configuration on POR.
	SEG18	x	O	ANA	Segment 18 analog output for LCD.
RF1/AN6/ C2OUT/SEG19	RF1	0	O	DIG	LATF<1> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<1> data input. Reads '0' on POR.
	AN6	1	I	ANA	A/D input channel 6. Default configuration on POR.
	C2OUT	0	O	DIG	Comparator 2 output; takes priority over port data.
SEG19	x	O	ANA	Segment 19 analog output for LCD.	
RF2/AN7/ C1OUT/SEG20	RF2	0	O	DIG	LATF<2> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<2> data input. Reads '0' on POR.
	AN7	1	I	ANA	A/D input channel 7. Default configuration on POR.
	C1OUT	0	O	TTL	Comparator 1 output; takes priority over port data.
SEG20	x	O	ANA	Segment 20 analog output for LCD.	
RF3/AN8/ SEG21	RF3	0	O	DIG	LATF<3> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<3> data input. Reads '0' on POR.
	AN8	1	I	ANA	A/D input channel 8 and Comparator C2+ input. Default input configuration on POR; not affected by analog output.
	SEG21	x	O	ANA	Segment 21 analog output for LCD.
RF4/AN9/ SEG22	RF4	0	O	DIG	LATF<4> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<4> data input. Reads '0' on POR.
	AN9	1	I	ANA	A/D input channel 9 and Comparator C2- input. Default input configuration on POR; does not affect digital output.
	SEG22	x	O	ANA	Segment 22 analog output for LCD.
RF5/AN10/ CVREF/SEG23	RF5	0	O	DIG	LATF<5> data output. Output unaffected by analog input; disabled when either LCD segment or CVREF is enabled.
		1	I	ST	PORTF<5> data input. Reads '0' on POR.
	AN10	1	I	ANA	A/D input channel 10 and Comparator C1+ input. Default input configuration on POR.
	CVREF	0	O	ANA	Comparator voltage reference output. Enabling this feature disables digital I/O.
SEG23	x	O	ANA	Segment 23 analog output for LCD.	
RF6/AN11/ SEG24	RF6	0	O	DIG	LATF<6> data output. Output is unaffected by analog input; disabled when LCD segment is enabled.
		1	I	ST	PORTF<6> data input. Reads '0' on POR.
	AN11	1	I	ANA	A/D input channel 11 and Comparator C1- input. Default input configuration on POR; does not affect digital output.
	SEG24	x	O	ANA	Segment 24 analog output for LCD.
RF7/ $\overline{SS}$ /SEG25	RF7	0	O	DIG	LATF<7> data output; disabled when LCD segment is enabled.
		1	I	ST	PORTF<7> data input.
	$\overline{SS}$	1	I	TTL	Slave select input for MSSP module.
	SEG25	x	O	ANA	Segment 25 analog output for LCD.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

# PIC18F6390/6490/8390/8490

**TABLE 9-13: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TRISF	PORTF Data Direction Register								62
PORTF	Read PORTF Data Latch/Write PORTF Data Latch								62
LATF	LATF Data Output Register								62
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

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

## 9.7 PORTG, TRISG and LATG Registers

PORTG is a 6-bit wide, bidirectional port. The corresponding Data Direction register is TRISG. Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISG bit (= 0) will make the corresponding PORTG pin an output (i.e., put the contents of the output latch on the selected pin).

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

PORTG is multiplexed with both USART and LCD functions (Table 9-14). PORTG pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTG pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings. The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register without concern due to peripheral overrides.

PORTG<4:0> are also multiplexed with LCD segment drives controlled by bits in the LCDSE3 register. I/O port functions are only available when the segments are disabled.

The sixth pin of PORTG ( $\overline{\text{MCLR}}/\text{VPP}/\text{RG5}$ ) is an input only pin. Its operation is controlled by the MCLRE Configuration bit. When selected as a port pin (MCLRE = 0), it functions as a digital input only pin; as such, it does not have TRIS or LAT bits associated with its operation. Otherwise, it functions as the device's Master Clear input. In either configuration, RG5 also functions as the programming voltage input during programming.

**Note:** On a Power-on Reset, RG5 is enabled as a digital input only if Master Clear functionality is disabled. All other 5 pins are configured as digital inputs.

### EXAMPLE 9-7: INITIALIZING PORTG

```
CLRF   PORTG   ; Initialize PORTG by
              ; clearing output
              ; data latches
CLRF   LATG    ; Alternate method
              ; to clear output
              ; data latches
MOVLW  0x04    ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISG   ; Set RG1:RG0 as outputs
              ; RG2 as input
              ; RG4:RG3 as inputs
```

# PIC18F6390/6490/8390/8490

**TABLE 9-14: PORTG FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RG0/SEG30	RG0	0	O	DIG	LATG<0> data output; disabled when LCD segment enabled.
		1	I	ST	PORTG<0> data input.
	SEG30	x	O	ANA	Segment 30 analog output for LCD.
RG1/TX2/CK2/SEG29	RG1	0	O	DIG	LATG<1> data output; disabled when LCD segment enabled.
		1	I	ST	PORTG<1> data input.
	TX2	1	O	DIG	Synchronous serial data output (AUSART module); takes priority over port data.
		1	O	DIG	Synchronous serial data input (AUSART module). User must configure as an input.
	CK2	1	O	DIG	Synchronous serial data input (AUSART module). User must configure as an input.
		1	I	ST	Synchronous serial clock input (AUSART module).
SEG29	x	O	ANA	Segment 29 analog output for LCD.	
RG2/RX2/DT2/SEG28	RG2	0	O	DIG	LATG<2> data output; disabled when LCD segment enabled.
		1	I	ST	PORTG<2> data input.
	RX2	1	I	ST	Asynchronous serial receive data input (AUSART module).
	DT2	1	O	DIG	Synchronous serial data output (AUSART module); takes priority over port data.
		1	I	ST	Synchronous serial data input (AUSART module). User must configure as an input.
	SEG28	x	O	ANA	Segment 28 analog output for LCD.
RG3/SEG27	RG3	0	O	DIG	LATG<3> data output; disabled when LCD segment enabled.
		1	I	ST	PORTG<3> data input.
	SEG27	0	O	ANA	Segment 27 analog output for LCD.
RG4/SEG26	RG4	0	O	DIG	LATG<4> data output; disabled when LCD segment enabled.
		1	I	ST	PORTG<4> data input.
	SEG26	x	O	ANA	Segment 26 analog output for LCD.
MCLR/VPP/RG5	MCLR	— <sup>(1)</sup>	I	ST	External Master Clear input; enabled when MCLRE Configuration bit is set.
	VPP	— <sup>(1)</sup>	I	ANA	High-voltage detection; used for ICSP™ mode entry detection. Always available, regardless of pin mode.
	RG5	— <sup>(1)</sup>	I	ST	PORTG<5> data input; enabled when MCLRE Configuration bit is clear.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**Note 1:** RG5 does not have a corresponding TRISG bit.

**TABLE 9-15: SUMMARY OF REGISTERS ASSOCIATED WITH PORTG**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
PORTG	—	—	RG5 <sup>(1)</sup>	Read PORTG pin/Write PORTG Data Latch					62
LATG	—	—	—	LATG Data Output Register					62
TRISG	—	—	—	PORTG Data Direction Register					62
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

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

**Note 1:** RG5 is available as an input only when MCLR is disabled.



## 9.8 PORTH, LATH and TRISH Registers

**Note:** PORTH is available only on 80-pin devices.

PORTH is an 8-bit wide, bidirectional I/O port. The corresponding Data Direction register is TRISH. Setting a TRISH bit (= 1) will make the corresponding PORTH pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISH bit (= 0) will make the corresponding PORTH pin an output (i.e., put the contents of the output latch on the selected pin).

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

All pins on PORTH are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

**Note:** On a Power-on Reset, these pins are configured as digital inputs.

PORTH is also multiplexed with LCD segment drives controlled by the LCDSE5 register. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-8: INITIALIZING PORTH

```
CLRF    PORTH    ; Initialize PORTH by
                ; clearing output
                ; data latches
CLRF    LATH     ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISH    ; Set RH3:RH0 as inputs
                ; RH5:RH4 as outputs
                ; RH7:RH6 as inputs
```

# PIC18F6390/6490/8390/8490

**TABLE 9-16: PORTH FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RH0/SEG47	RH0	0	O	DIG-4	LATH<0> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<0> data input.
	SEG47	x	O	ANA	Segment 47 analog output for LCD.
RH1/SEG46	RH1	0	O	DIG	LATH<1> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<1> data input.
	SEG46	x	O	ANA	Segment 46 analog output for LCD.
RH2/SEG45	RH2	0	O	DIG	LATH<2> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<2> data input.
	SEG45	x	O	ANA	Segment 45 analog output for LCD.
RH3/SEG44	RH3	0	O	DIG	LATH<3> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<3> data input.
	SEG44	x	O	ANA	Segment 44 analog output for LCD.
RH4/SEG40	RH4	0	O	DIG	LATH<4> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<4> data input.
	SEG40	x	O	ANA	Segment 40 analog output for LCD
RH5/SEG41	RH5	0	O	DIG	LATH<5> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<5> data input.
	SEG41	x	O	ANA	Segment 41 analog output for LCD.
RH6/SEG42	RH6	0	O	DIG	LATH<6> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<6> data input.
	SEG42	x	O	ANA	Segment 42 analog output for LCD.
RH7/SEG43	RH7	0	O	DIG	LATH<7> data output; disabled when LCD segment enabled.
		1	I	ST	PORTH<7> data input.
	SEG43	x	O	ANA	Segment 43 analog output for LCD.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**TABLE 9-17: SUMMARY OF REGISTERS ASSOCIATED WITH PORTH**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TRISH	PORTH Data Direction Register								62
PORTH	Read PORTH pin/Write PORTH Data Latch								62
LATH	LATH Data Output Register								62
LCDSE5	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	64

## 9.9 PORTJ, TRISJ and LATJ Registers

**Note:** PORTJ is available only on 80-pin devices.

PORTJ is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISJ. Setting a TRISJ bit (= 1) will make the corresponding PORTJ pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISJ bit (= 0) will make the corresponding PORTJ pin an output (i.e., put the contents of the output latch on the selected pin).

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

All pins on PORTJ are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

**Note:** On a Power-on Reset, these pins are configured as digital inputs.

PORTJ is also multiplexed with LCD segment drives controlled by the LCDSE4 register. I/O port functions are only available when the segments are disabled.

### EXAMPLE 9-9: INITIALIZING PORTJ

```
CLRF   PORTJ   ; Initialize PORTG by
              ; clearing output
              ; data latches
CLRF   LATJ    ; Alternate method
              ; to clear output
              ; data latches
MOVLW 0xCF    ; Value used to
              ; initialize data
              ; direction
MOVWF  TRISJ   ; Set RJ3:RJ0 as inputs
              ; RJ5:RJ4 as output
              ; RJ7:RJ6 as inputs
```

# PIC18F6390/6490/8390/8490

**TABLE 9-18: PORTJ FUNCTIONS**

Pin Name	Function	TRIS Setting	I/O	Buffer	Description
RJ0/SEG32	RJ0	0	O	DIG	LATJ<0> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<0> data input.
	SEG32	x	O	ANA	Segment 32 analog output for LCD.
RJ1/SEG33	RJ1	0	O	DIG	LATJ<1> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<1> data input.
	SEG33	x	O	ANA	Segment 33 analog output for LCD.
RJ2/SEG34	RJ2	0	O	DIG	LATJ<2> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<2> data input.
	SEG34	x	O	ANA	Segment 34 analog output for LCD.
RJ3/SEG35	RJ3	0	O	DIG	LATJ<3> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<3> data input.
	SEG35	x	O	ANA	Segment 35 analog output for LCD.
RJ4/SEG39	RJ4	0	O	DIG	LATJ<4> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<4> data input.
	SEG39	x	O	ANA	Segment 39 analog output for LCD.
RJ5/SEG38	RJ5	0	O	DIG	LATJ<5> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<5> data input.
	SEG38	x	O	ANA	Segment 38 analog output for LCD.
RJ6/SEG37	RJ6	0	O	DIG	LATJ<6> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<6> data input.
	SEG37	x	O	ANA	Segment 37 analog output for LCD.
RJ7/SEG36	RJ7	0	O	DIG	LATJ<7> data output; disabled when LCD segment enabled.
		1	I	ST	PORTJ<7> data input.
	SEG36	x	O	ANA	Segment 36 analog output for LCD.

**Legend:** O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**TABLE 9-19: SUMMARY OF REGISTERS ASSOCIATED WITH PORTJ**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
PORTJ	Read PORTJ pin/Write PORTJ Data Latch								62
LATJ	LATJ Data Output Register								62
TRISJ	PORTJ Data Direction Register								62
LCDSE4	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	64

# PIC18F6390/6490/8390/8490

## 10.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

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

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

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

### REGISTER 10-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 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    **T0PS2:T0PS0:** Timer0 Prescaler Select bits  
111 = 1:256 Prescale value  
110 = 1:128 Prescale value  
101 = 1:64 Prescale value  
100 = 1:32 Prescale value  
011 = 1:16 Prescale value  
010 = 1:8 Prescale value  
001 = 1:4 Prescale value  
000 = 1:2 Prescale value

# PIC18F6390/6490/8390/8490

## 10.1 Timer0 Operation

Timer0 can operate as either a timer or a counter; the mode is selected by clearing the T0CS bit (T0CON<5>). In Timer mode (T0CS = 0), the module increments on every clock by default, unless a different prescaler value is selected (see **Section 10.3 “Prescaler”**). If the TMR0 register is written to, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

The Counter mode is selected by setting the T0CS bit (= 1). In Counter mode, Timer0 increments either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (T0CON<4>); clearing this bit selects the rising edge. Restrictions on the external clock input are discussed below.

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

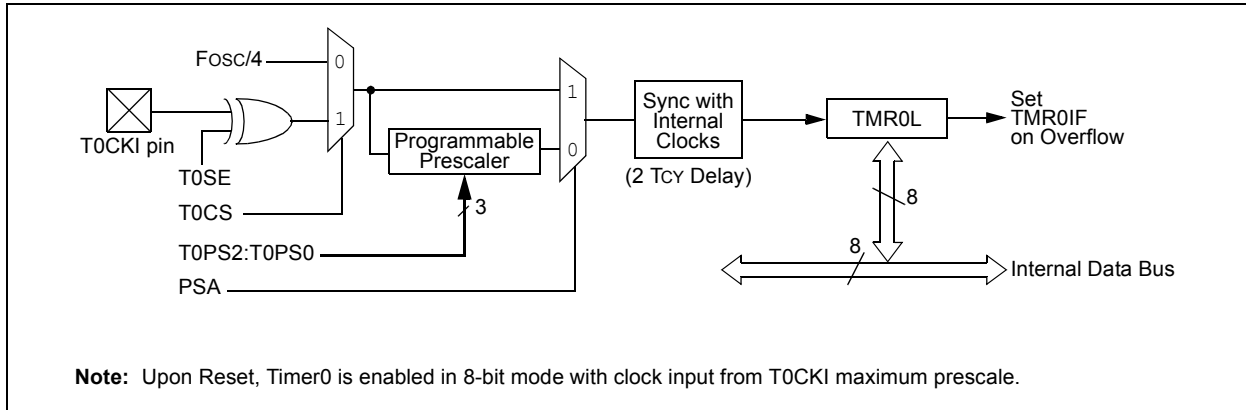
internal phase clock (Tosc). There is a delay between synchronization and the onset of incrementing the timer/counter.

## 10.2 Timer0 Reads and Writes in 16-Bit Mode

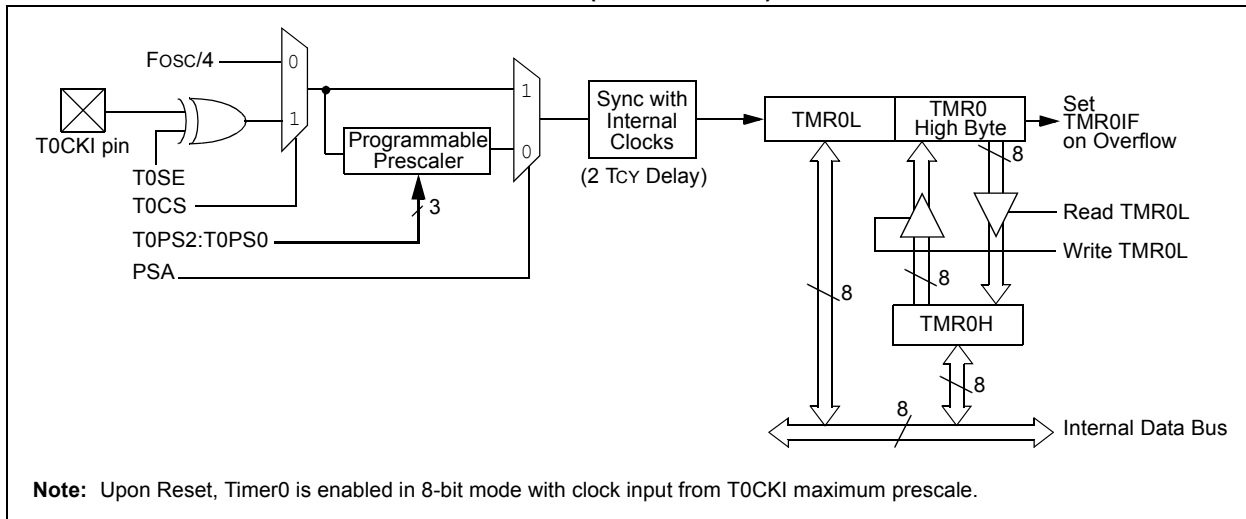
TMR0H is not the actual high byte of Timer0 in 16-bit mode; it is actually a buffered version of the real high byte of Timer0, which is not directly readable nor writable (refer to Figure 10-2). TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0, without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

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



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



# PIC18F6390/6490/8390/8490

## 10.3 Prescaler

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

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

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

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

### 10.3.1 SWITCHING PRESCALER ASSIGNMENT

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

## 10.4 Timer0 Interrupt

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

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

**TABLE 10-1: REGISTERS ASSOCIATED WITH TIMER0**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TMR0L	Timer0 Register Low Byte								60
TMR0H	Timer0 Register High Byte								60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	60
TRISA	PORTA Data Direction Register								62

**Legend:** Shaded cells are not used by Timer0.

# PIC18F6390/6490/8390/8490

---

NOTES:



# PIC18F6390/6490/8390/8490

## 11.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

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

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

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

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

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

### REGISTER 11-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	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      **RD16:** 16-Bit Read/Write Mode Enable bit  
 1 = Enables register read/write of Timer1 in one 16-bit operation  
 0 = Enables register read/write of Timer1 in two 8-bit operations
- bit 6      **T1RUN:** Timer1 System Clock Status bit  
 1 = Device clock is derived from Timer1 oscillator  
 0 = Device clock is derived from another source
- bit 5-4    **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits  
 11 = 1:8 Prescale value  
 10 = 1:4 Prescale value  
 01 = 1:2 Prescale value  
 00 = 1:1 Prescale value
- bit 3      **T1OSCEN:** Timer1 Oscillator Enable bit  
 1 = Timer1 oscillator is enabled  
 0 = Timer1 oscillator is shut off  
 The oscillator inverter and feedback resistor are turned off to eliminate power drain.
- bit 2      **T1SYNC:** Timer1 External Clock Input Synchronization Select bit  
When TMR1CS = 1:  
 1 = Do not synchronize external clock input  
 0 = Synchronize external clock input  
When TMR1CS = 0:  
 This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.
- bit 1      **TMR1CS:** Timer1 Clock Source Select bit  
 1 = External clock from pin RC0/T1OSO/T13CKI (on the rising edge)  
 0 = Internal clock (Fosc/4)
- bit 0      **TMR1ON:** Timer1 On bit  
 1 = Enables Timer1  
 0 = Stops Timer1

# PIC18F6390/6490/8390/8490

## 11.1 Timer1 Operation

Timer1 can operate in one of these modes:

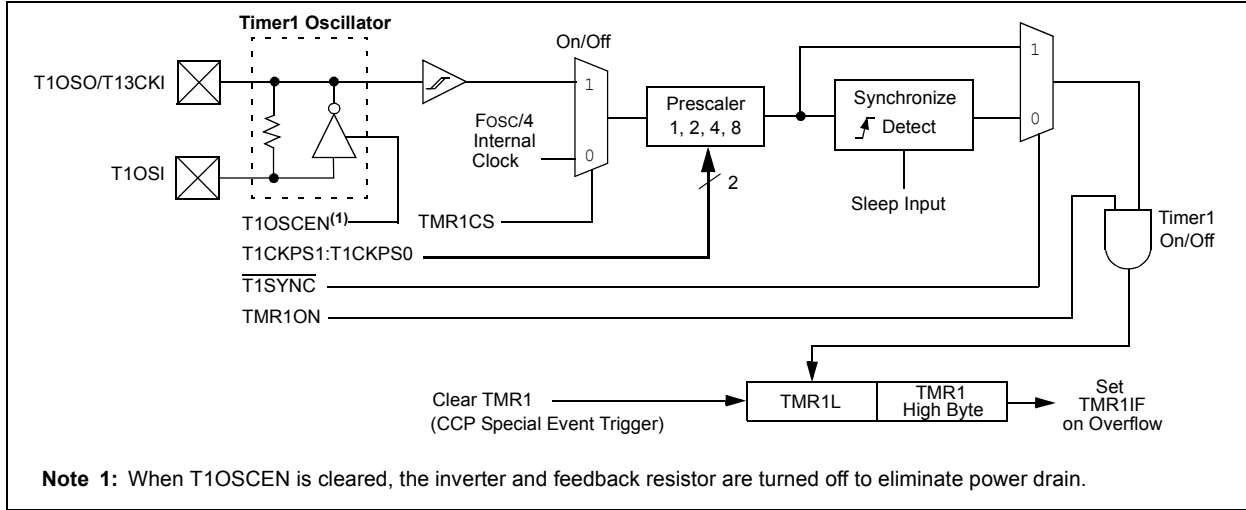
- Timer
- Synchronous Counter
- Asynchronous Counter

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

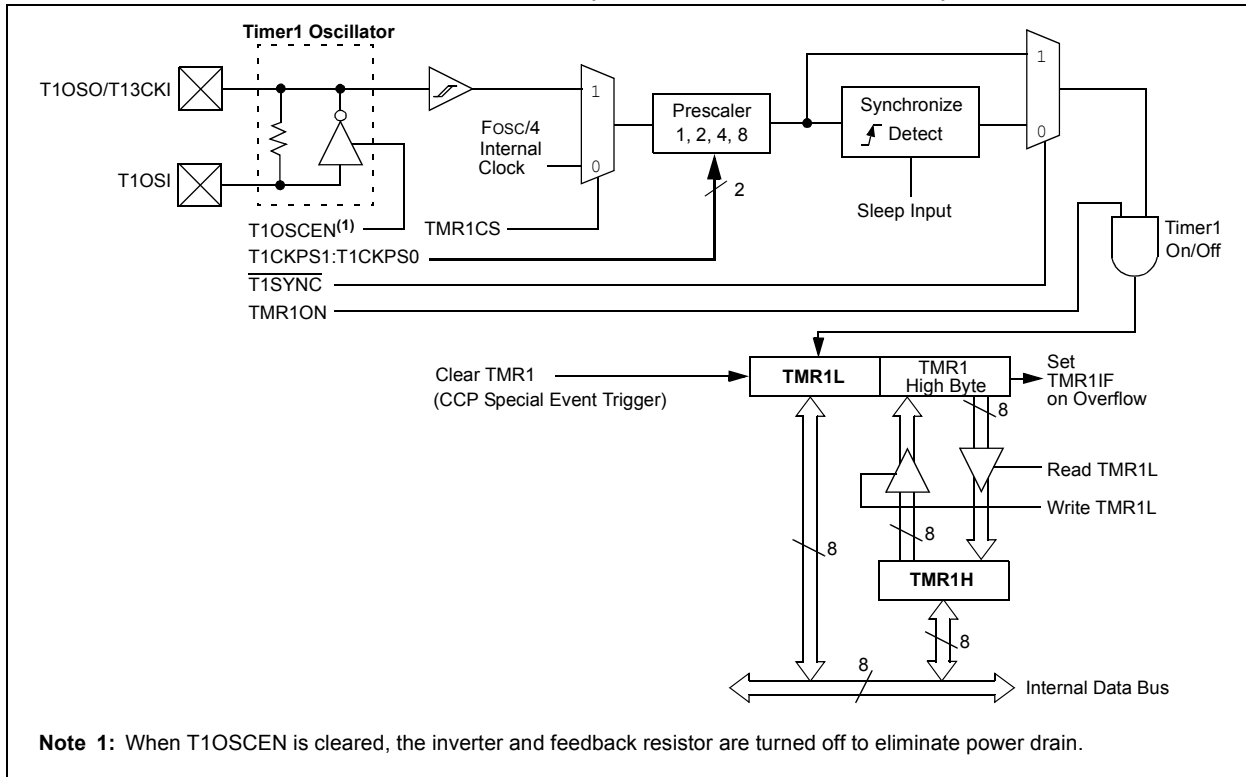
cycle ( $F_{osc}/4$ ). When the bit is set, Timer1 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

When Timer1 is enabled, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

**FIGURE 11-1: TIMER1 BLOCK DIAGRAM**



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



## 11.2 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 11-2). When the RD16 control bit (T1CON<7>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L will load the contents of the high byte of Timer1 into the Timer1 High Byte Buffer register. This provides the user with the ability to accurately read all 16 bits of Timer1 without having to determine whether a read of the high byte, followed by a read of the low byte, has become invalid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. The Timer1 high byte is updated with the contents of TMR1H when a write occurs to TMR1L. This allows a user to write all 16 bits to both the high and low bytes of Timer1 at once.

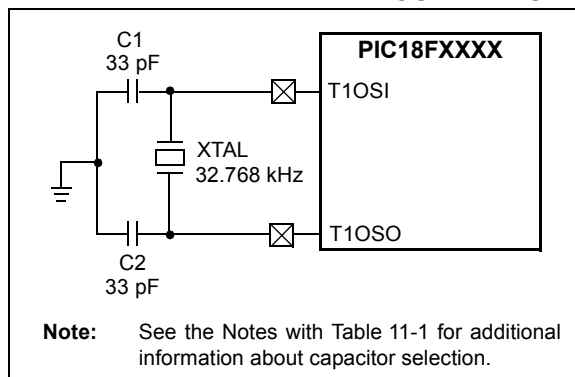
The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

## 11.3 Timer1 Oscillator

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

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

**FIGURE 11-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR**



**TABLE 11-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR<sup>(2,3,4)</sup>**

Osc Type	Freq	C1	C2
LP	32 kHz	27 pF <sup>(1)</sup>	27 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.

### 11.3.1 USING TIMER1 AS A CLOCK SOURCE

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

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

### 11.3.2 LOW-POWER TIMER1 OPTION

The Timer1 oscillator can operate at two distinct levels of power consumption based on device configuration. When the LPT1OSC Configuration bit is set, the Timer1 oscillator operates in a low-power mode. When LPT1OSC is not set, Timer1 operates at a higher power level. Power consumption for a particular mode is relatively constant, regardless of the device's operating mode. The default Timer1 configuration is the higher power mode.

As the Low-Power Timer1 mode tends to be more sensitive to interference, high noise environments may cause some oscillator instability. The low-power option is therefore best suited for low noise applications where power conservation is an important design consideration.

# PIC18F6390/6490/8390/8490

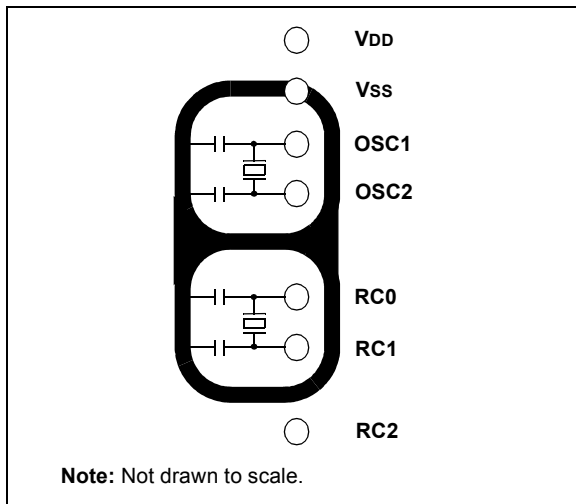
## 11.3.3 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

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

The oscillator circuit, shown in Figure 11-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high-speed circuit must be located near the oscillator (such as the CCP1 pin in Output Compare or PWM mode, or the primary oscillator using the OSC2 pin), a grounded guard ring around the oscillator circuit, as shown in Figure 11-4, may be helpful when used on a single sided PCB or in addition to a ground plane.

**FIGURE 11-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING**



## 11.4 Timer1 Interrupt

The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The Timer1 interrupt, if enabled, is generated on overflow, which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled or disabled by setting or clearing the Timer1 Interrupt Enable bit, TMR1IE (PIE1<0>).

## 11.5 Resetting Timer1 Using the CCP Special Event Trigger

If either of the CCP modules is configured in Compare mode to generate a Special Event Trigger (CCP1M3:CCP1M0 or CCP2M3:CCP2M0 = 1011), this signal will reset Timer1. The trigger from CCP2 will also start an A/D conversion if the A/D module is enabled (see Section 14.3.4 “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 CCRH:CCPL register pair effectively becomes a period register for Timer1.

If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a Special Event Trigger, the write operation will take precedence.

**Note:** The Special Event Triggers from the CCP2 module will not set the TMR1IF interrupt flag bit (PIR1<0>).

## 11.6 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in Section 11.3 “Timer1 Oscillator”, above) gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time base and several lines of application code to calculate the time. When operating in Sleep mode and using a battery or supercapacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCISR`, shown in Example 11-1, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow, triggers the interrupt and calls the routine, which increments the seconds counter by one; additional counters for minutes and hours are incremented as the previous counter overflows.

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

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

# PIC18F6390/6490/8390/8490

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

```

RTCinit
    MOVLW    80h                ; Preload TMR1 register pair
    MOVWF   TMR1H              ; for 1 second overflow
    CLRF    TMR1L
    MOVLW   b'00001111'        ; Configure for external clock,
    MOVWF   T1OSC              ; Asynchronous operation, external oscillator
    CLRF    secs               ; Initialize timekeeping registers
    CLRF    mins
    MOVLW   .12
    MOVWF   hours
    BSF     PIE1, TMR1IE       ; Enable Timer1 interrupt
    RETURN

RTCisr
    BSF     TMR1H, 7           ; Preload for 1 sec overflow
    BCF     PIR1, TMR1IF      ; Clear interrupt flag
    INCF    secs, F           ; Increment seconds
    MOVLW   .59               ; 60 seconds elapsed?
    CPFSGT  secs
    RETURN                       ; No, done
    CLRF    secs              ; Clear seconds
    INCF    mins, F          ; Increment minutes
    MOVLW   .59               ; 60 minutes elapsed?
    CPFSGT  mins
    RETURN                       ; No, done
    CLRF    mins              ; clear minutes
    INCF    hours, F         ; Increment hours
    MOVLW   .23               ; 24 hours elapsed?
    CPFSGT  hours
    RETURN                       ; No, done
    MOVLW   .01               ; Reset hours to 1
    MOVWF   hours
    RETURN                       ; Done
    
```

**TABLE 11-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TMR1L	Timer1 Register Low Byte								60
TMR1H	Timer1 Register High Byte								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 12.0 TIMER2 MODULE

The Timer2 timer module incorporates the following features:

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

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

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

## 12.1 Timer2 Operation

In normal operation, TMR2 is incremented from 00h on each clock ( $F_{osc}/4$ ). A 2-bit counter/prescaler on the clock input gives direct input, divide-by-4 and divide-by-16 prescale options; these are selected by the prescaler control bits, T2CKPS1:T2CKPS0 (T2CON<1:0>). The value of TMR2 is compared to that of the Period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/postscaler (see **Section 12.2 “Timer2 Interrupt”**).

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

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

TMR2 is not cleared when T2CON is written.

### REGISTER 12-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

#### 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    **T2OUTPS3:T2OUTPS0:** Timer2 Output Postscale Select bits  
           0000 = 1:1 Postscale  
           0001 = 1:2 Postscale  
           •  
           •  
           •  
           1111 = 1:16 Postscale

bit 2      **TMR2ON:** Timer2 On bit  
           1 = Timer2 is on  
           0 = Timer2 is off

bit 1-0    **T2CKPS1:T2CKPS0:** Timer2 Clock Prescale Select bits  
           00 = Prescaler is 1  
           01 = Prescaler is 4  
           1x = Prescaler is 16

# PIC18F6390/6490/8390/8490

## 12.2 Timer2 Interrupt

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

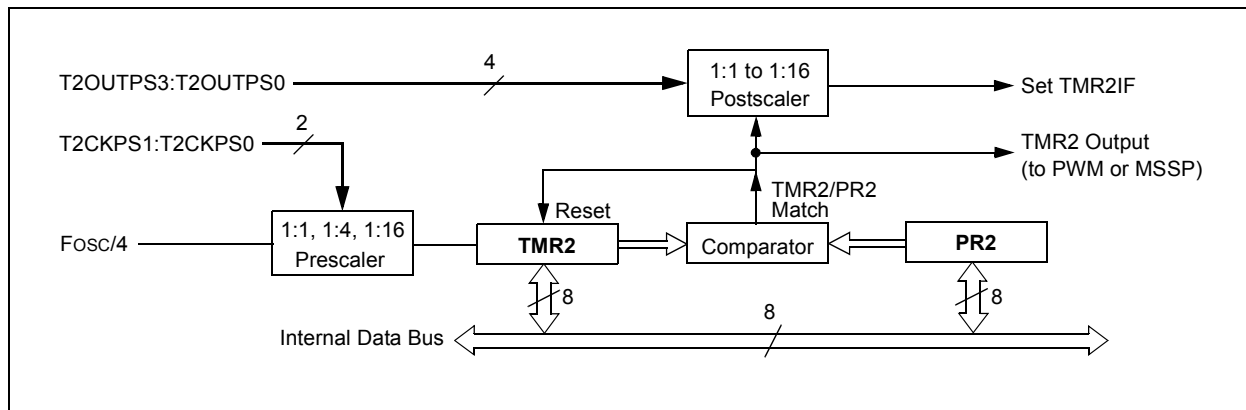
A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, T2OUTPS3:T2OUTPS0 (T2CON<6:3>).

## 12.3 TMR2 Output

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

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

**FIGURE 12-1: TIMER2 BLOCK DIAGRAM**



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

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TMR2	Timer2 Register								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 Period Register								60

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



# PIC18F6390/6490/8390/8490

## 13.0 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 CCP Special Event Trigger

A simplified block diagram of the Timer3 module is shown in Figure 13-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 13-2.

The Timer3 module is controlled through the T3CON register (Register 13-1). It also selects the clock source options for the CCP modules (see **Section 14.1.1 "CCP Modules and Timer Resources"** for more information).

### REGISTER 13-1: T3CON: TIMER3 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON
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      **RD16:** 16-Bit Read/Write Mode Enable bit  
1 = Enables register read/write of Timer3 in one 16-bit operation  
0 = Enables register read/write of Timer3 in two 8-bit operations
- bit 6,3    **T3CCP2:T3CCP1:** Timer3 and Timer1 to CCPx Enable bits  
1x = Timer3 is the capture/compare clock source for the CCPx modules  
01 = Timer3 is the capture/compare clock source for the CCP2 module;  
      Timer1 is the capture/compare clock source for the CCP1 module  
00 = Timer1 is the capture/compare clock source for the CCPx modules
- bit 5-4    **T3CKPS1:T3CKPS0:** Timer3 Input Clock Prescale Select bits  
11 = 1:8 Prescale value  
10 = 1:4 Prescale value  
01 = 1:2 Prescale value  
00 = 1:1 Prescale value
- bit 2      **T3SYNC:** Timer3 External Clock Input Synchronization Control bit  
(Not usable if the device clock comes from Timer1/Timer3.)  
When TMR3CS = 1:  
1 = Do not synchronize external clock input  
0 = Synchronize external clock input  
When TMR3CS = 0:  
This bit is ignored. Timer3 uses the internal clock when TMR3CS = 0.
- bit 1      **TMR3CS:** Timer3 Clock Source Select bit  
1 = External clock input from Timer1 oscillator or T13CKI (on the rising edge after the first falling edge)  
0 = Internal clock (Fosc/4)
- bit 0      **TMR3ON:** Timer3 On bit  
1 = Enables Timer3  
0 = Stops Timer3

# PIC18F6390/6490/8390/8490

## 13.1 Timer3 Operation

Timer3 can operate in one of three modes:

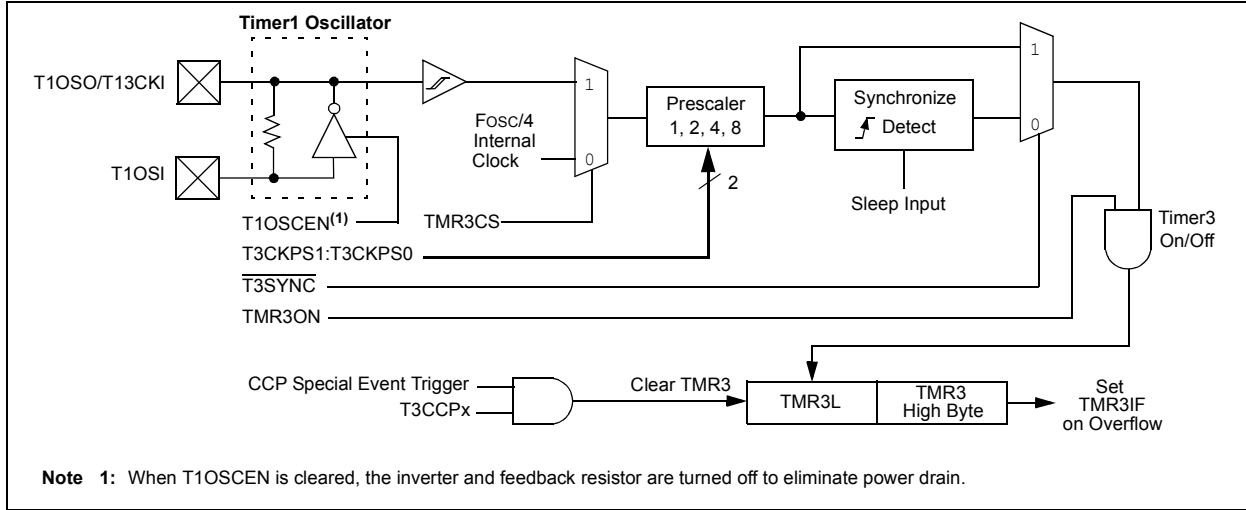
- Timer
- Synchronous counter
- Asynchronous counter

The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>). When TMR3CS is cleared (= 0), Timer3 increments on every internal instruction

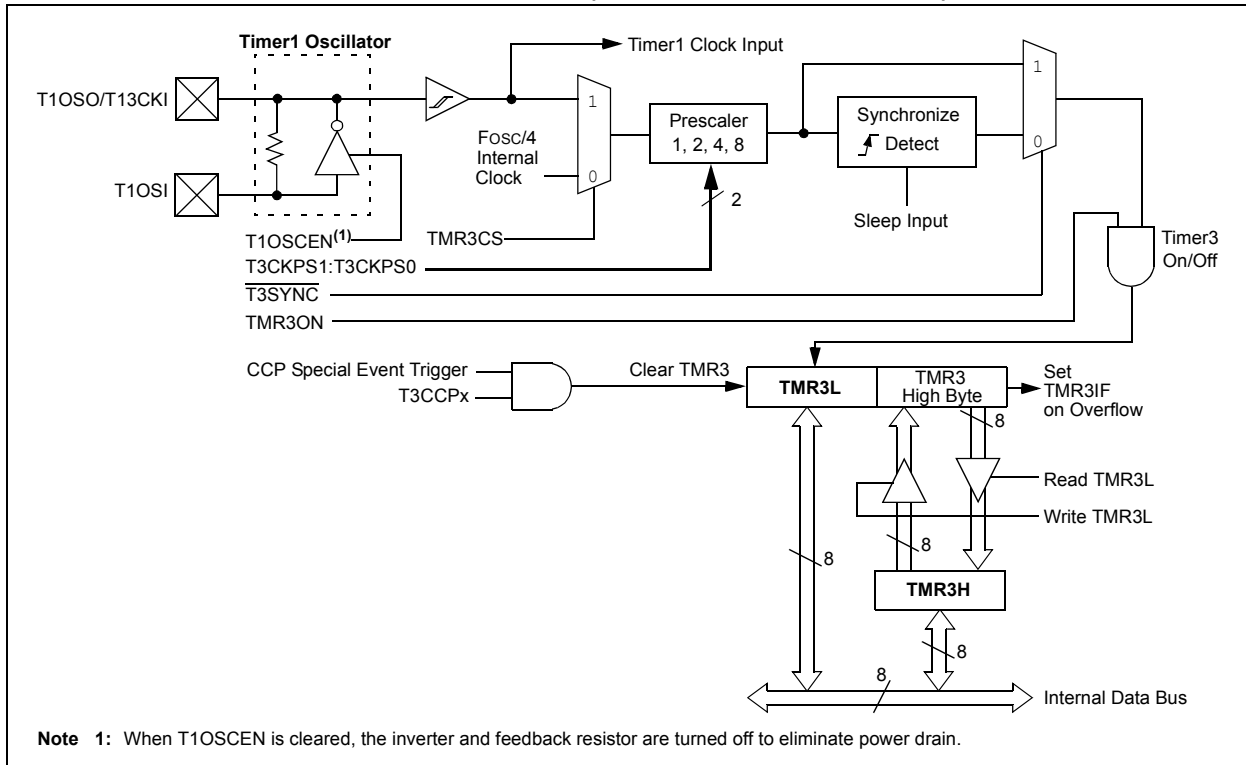
cycle ( $F_{osc}/4$ ). When the bit is set, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

As with Timer1, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs when the Timer1 oscillator is enabled. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

**FIGURE 13-1: TIMER3 BLOCK DIAGRAM**



**FIGURE 13-2: TIMER3 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)**



# PIC18F6390/6490/8390/8490

## 13.2 Timer3 16-Bit Read/Write Mode

Timer3 can be configured for 16-bit reads and writes (see Figure 13-2). When the RD16 control bit (T3CON<7>) 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 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 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.

## 13.3 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 11.0 “Timer1 Module”**.

## 13.4 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>).

## 13.5 Resetting Timer3 Using the CCP Special Event Trigger

If either of the CCP modules is configured in Compare mode to generate a Special Event Trigger (CCP1M3:CCP1M0 or CCP2M3:CCP2M0 = 1011), this signal will reset Timer1. The trigger from CCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 14.3.4 “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 CCPR2H:CCPR2L 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 a CCP module, the write will take precedence.

**Note:** The Special Event Triggers from the CCP2 module will not set the TMR3IF interrupt flag bit (PIR2<1>).

**TABLE 13-1: REGISTERS ASSOCIATED WITH TIMER3 AS A TIMER/COUNTER**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
TMR3L	Timer3 Register Low Byte								61
TMR3H	Timer3 Register High Byte								61
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYN $\bar{C}$	TMR1CS	TMR1ON	60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYN $\bar{C}$	TMR3CS	TMR3ON	61

**Legend:** — = unimplemented, read as ‘0’. Shaded cells are not used by the Timer3 module.

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 14.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F6390/6490/8390/8490 devices have two CCP (Capture/Compare/PWM) modules, designated CCP1 and CCP2. Both modules implement standard capture, compare and Pulse-Width Modulation (PWM) modes.

Each CCP module contains a 16-bit register which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. For the sake of clarity, all CCP module operation in the following sections is described with respect to CCP2, but is equally applicable to CCP1.

**REGISTER 14-1: CCPxCON: CCPx CONTROL REGISTER**

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

**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      **DCxB1:DCxB0:** PWM Duty Cycle bit 1 and bit 0 for CCPx Module

Capture mode:

Unused.

Compare mode:

Unused.

PWM mode:

These bits are the two Least Significant bits (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight Most Significant bits (DCxB9:DCxB2) of the duty cycle are found in CCPRxL.

bit 3-0      **CCPxM3:CCPxM0:** CCPx Module Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)

0001 = Reserved

0010 = Compare mode, toggle output on match (CCPxIF bit is set)

0011 = Reserved

0100 = Capture mode, every falling edge

0101 = Capture mode, every rising edge

0110 = Capture mode, every 4th rising edge

0111 = Capture mode, every 16th rising edge

1000 = Compare mode: initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)

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

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

1011 = Compare mode: trigger special event, reset timer, start A/D conversion on CCPx match (CCPxIF bit is set)<sup>(1)</sup>

11xx = PWM mode

**Note 1:** CCPxM3:CCPxM0 = 1011 will only reset the timer and not start the A/D conversion on the CCPx match.

# PIC18F6390/6490/8390/8490

## 14.1 CCP Module Configuration

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

### 14.1.1 CCP MODULES AND TIMER RESOURCES

The CCP modules utilize Timers 1, 2 or 3, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 is available for modules in PWM mode.

**TABLE 14-1: CCP MODE – TIMER RESOURCE**

CCP Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3
PWM	Timer2

The assignment of a particular timer to a module is determined by the Timer to CCP enable bits in the T3CON register (Register 13-1). Both modules may be active at any given time and may share the same timer resource if they are configured to operate in the same mode (capture/compare or PWM) at the same time. The interactions between the two modules are summarized in Table 14-2.

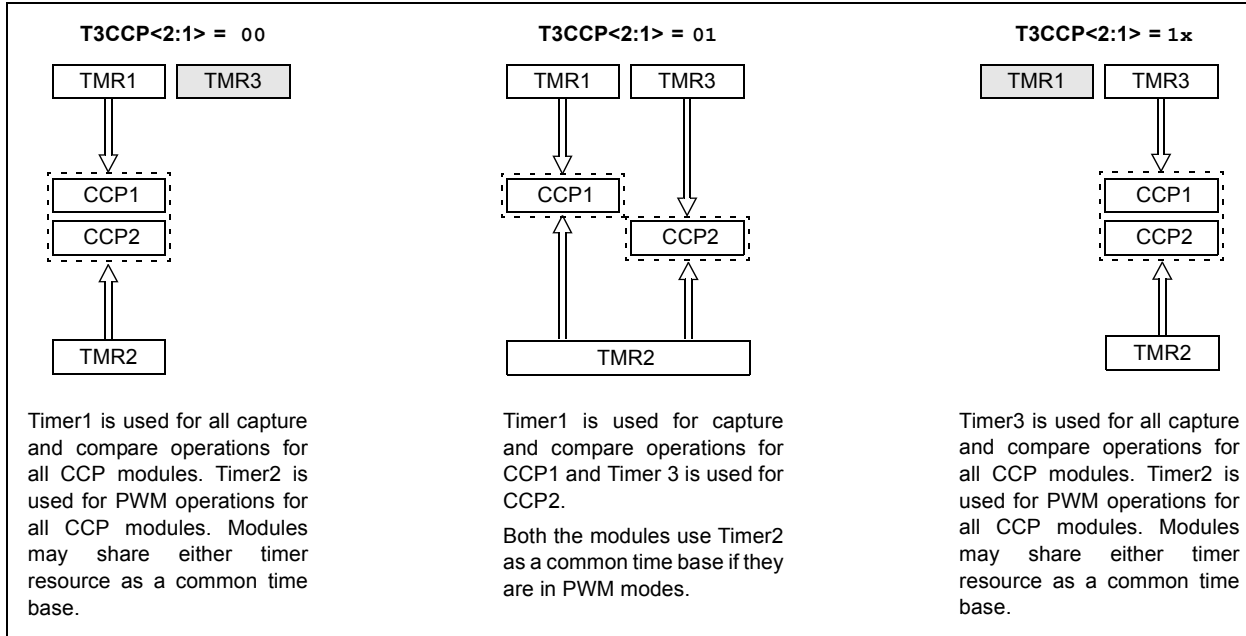
Depending on the configuration selected, up to four timers may be active at once, with modules in the same configuration (capture/compare or PWM) sharing timer resources. The possible configurations are shown in Figure 14-1.

### 14.1.2 CCP2 PIN ASSIGNMENT

The pin assignment for CCP2 (capture input, compare and PWM output) can change based on device configuration. The CCP2MX Configuration bit determines which pin CCP2 is multiplexed to. By default, it is assigned to RC1 (CCP2MX = 1). If the Configuration bit is cleared, CCP2 is multiplexed with RE7.

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

**FIGURE 14-1: CCP AND TIMER INTERCONNECT CONFIGURATIONS**



# PIC18F6390/6490/8390/8490

**TABLE 14-2: INTERACTIONS BETWEEN CCP1 AND CCP2 FOR TIMER RESOURCES**

CCP1 Mode	CCP2 Mode	Interaction
Capture	Capture	Each module can use TMR1 or TMR3 as the time base. The time base can be different for each CCP.
Capture	Compare	CCP2 can be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Automatic A/D conversions on trigger event can also be done. Operation of CCP1 could be affected if it is using the same timer as a time base.
Compare	Capture	CCP1 can be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Operation of CCP2 could be affected if it is using the same timer as a time base.
Compare	Compare	Either module can be configured for the Special Event Trigger to reset the time base. Automatic A/D conversions on CCP2 trigger event can be done. Conflicts may occur if both modules are using the same time base.
Capture	PWM*	None
Compare	PWM*	None
PWM*	Capture	None
PWM*	Compare	None
PWM*	PWM	Both PWMs will have the same frequency and update rate (TMR2 interrupt).

\* Includes standard and Enhanced PWM operation.

# PIC18F6390/6490/8390/8490

## 14.2 Capture Mode

In Capture mode, the CCPR2H:CCPR2L register pair captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on the CCP2 pin (RC1 or RE7, depending on device configuration). An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

The event is selected by the mode select bits, CCP2M3:CCP2M0 (CCP2CON<3:0>). When a capture is made, the interrupt request flag bit, CCP2IF (PIR2<0>), is set; it must be cleared in software. If another capture occurs before the value in register CCPR2 is read, the old captured value is overwritten by the new captured value.

### 14.2.1 CCP PIN CONFIGURATION

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

**Note:** If RC1/CCP2 or RE7/CCP2 is configured as an output, a write to the port can cause a capture condition.

### 14.2.2 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 CCP module is selected in the T3CON register (see Section 14.1.1 “CCP Modules and Timer Resources”).

### 14.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP2IE (PIE2<0>) clear to avoid false interrupts and should clear the flag bit, CCP2IF, following any such change in operating mode.

### 14.2.4 CCP PRESCALER

There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCP2M3:CCP2M0). Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

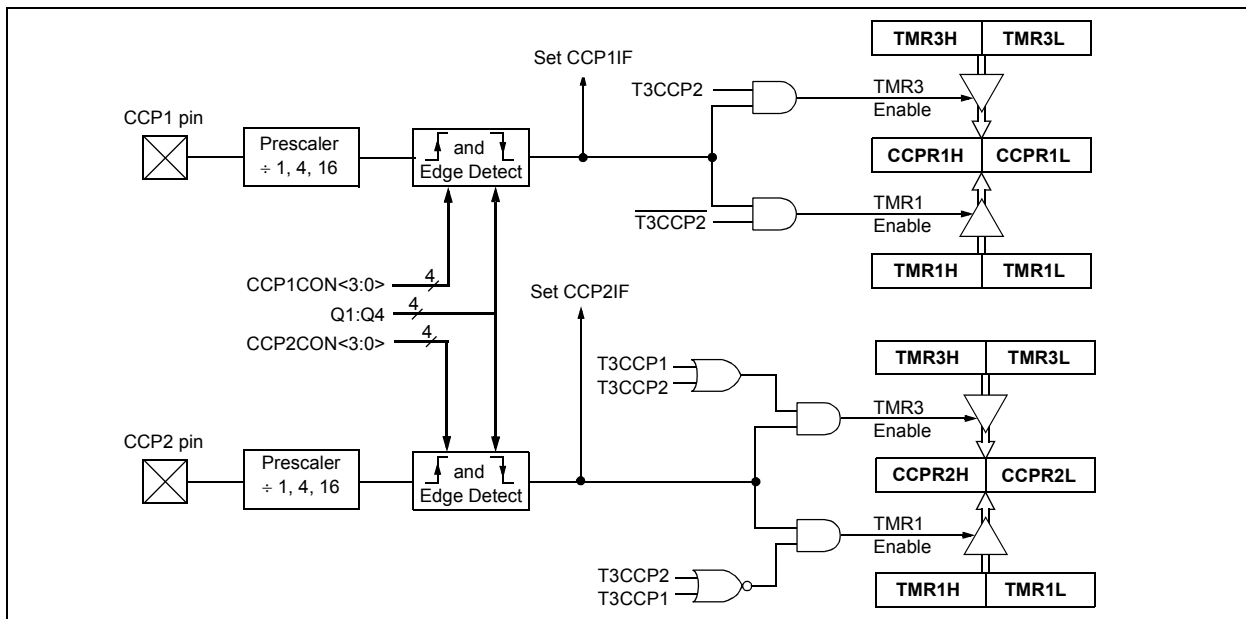
Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared; therefore, the first capture may be from a non-zero prescaler. Example 14-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

#### EXAMPLE 14-1: CHANGING BETWEEN CAPTURE PRESCALERS

```

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

FIGURE 14-2: CAPTURE MODE OPERATION BLOCK DIAGRAM





## 14.3 Compare Mode

In Compare mode, the 16-bit CCPR2 register value is constantly compared against either the TMR1 or TMR3 register pair value. When a match occurs, the CCP2 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 (CCP2M3:CCP2M0). At the same time, the interrupt flag bit, CCP2IF, is set.

### 14.3.1 CCP PIN CONFIGURATION

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

**Note:** Clearing the CCP2CON register will force the RC1 or RE7 compare output latch (depending on device configuration) to the default low level. This is not the PORTC or PORTE I/O data latch.

### 14.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode, or Synchronized Counter mode, if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

### 14.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCP2M3:CCP2M0 = 1010), the CCP2 pin is not affected. Only a CCP interrupt is generated if enabled and the CCP2IE bit is set.

### 14.3.4 SPECIAL EVENT TRIGGER

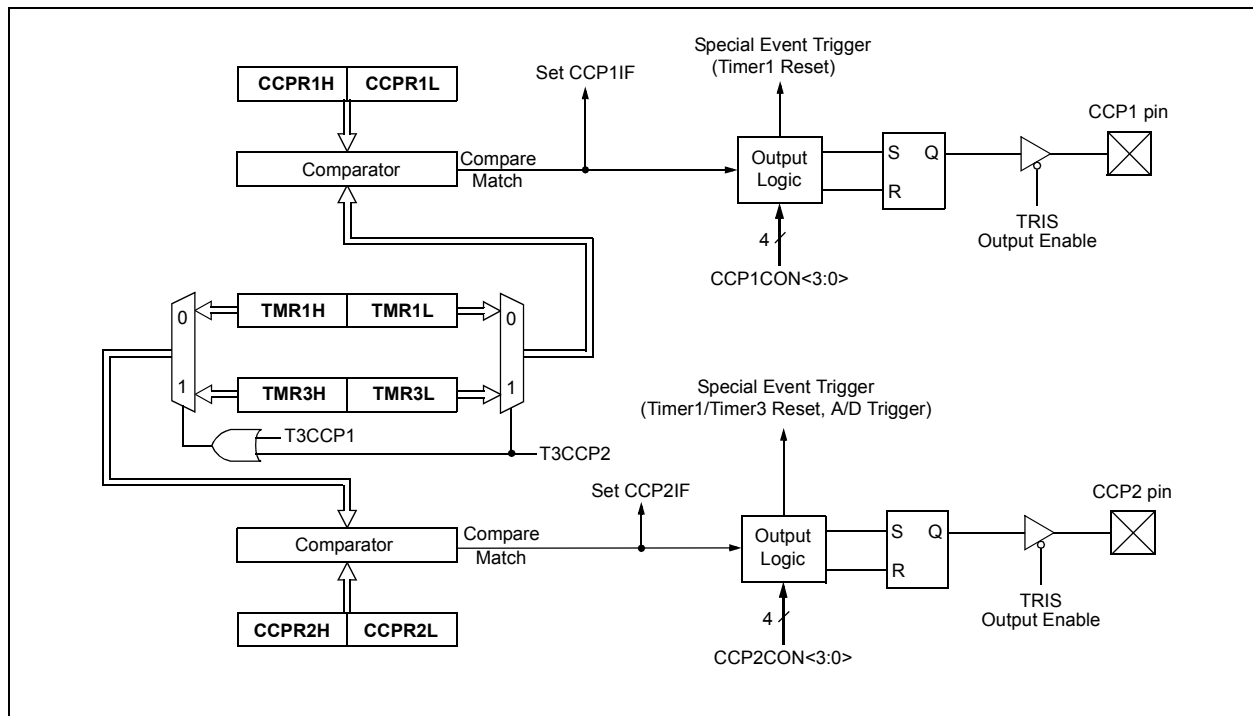
Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCP2M3:CCP2M0 = 1011).

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

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

**Note:** The Special Event Trigger of CCP1 only resets Timer1/Timer3 and cannot start an A/D conversion even when the A/D converter is enabled.

**FIGURE 14-3: COMPARE MODE OPERATION BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

**TABLE 14-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
TRISC	PORTC Data Direction Register								62
TRISE	PORTE Data Direction Register								62
TMR1L	Timer1 Register Low Byte								60
TMR1H	Timer1 Register High Byte								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	60
TMR3H	Timer3 Register High Byte								61
TMR3L	Timer3 Register Low Byte								61
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	61
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								61
CCPR1H	Capture/Compare/PWM Register 1 High Byte								61
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								61
CCPR2H	Capture/Compare/PWM Register 2 High Byte								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by capture/compare, Timer1 or Timer3.

## 14.4 PWM Mode

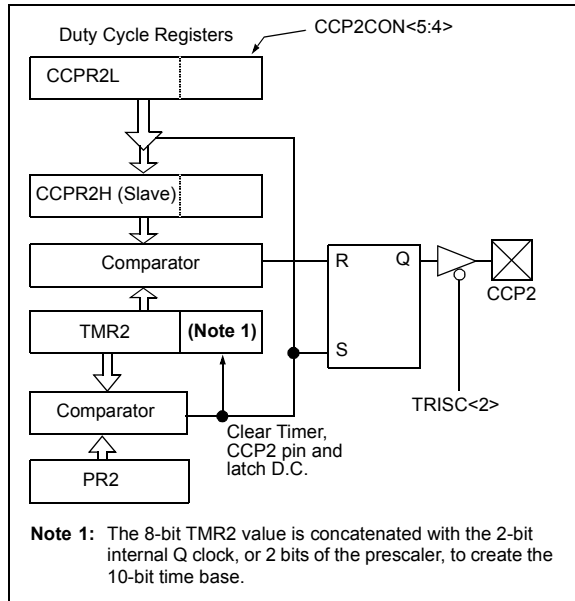
In Pulse-Width Modulation (PWM) mode, the CCP2 pin produces up to a 10-bit resolution PWM output. Since the CCP2 pin is multiplexed with a PORTC or PORTE data latch, the appropriate TRIS bit must be cleared to make the CCP2 pin an output.

**Note:** Clearing the CCP2CON register will force the RC1 or RE7 output latch (depending on device configuration) to the default low level. This is not the PORTC or PORTE I/O data latch.

Figure 14-4 shows a simplified block diagram of the CCP2 module in PWM mode.

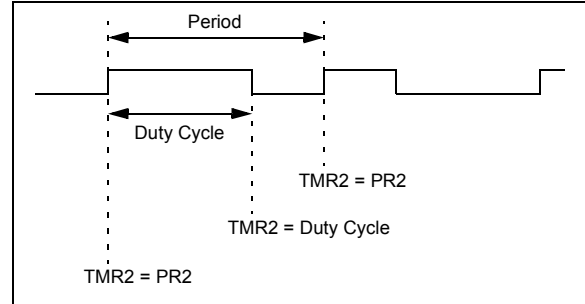
For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 14.4.3 “Setup for PWM Operation”**.

**FIGURE 14-4: SIMPLIFIED PWM BLOCK DIAGRAM**



A PWM output (Figure 14-5) 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 14-5: PWM OUTPUT**



### 14.4.1 PWM PERIOD

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

**EQUATION 14-1:**

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

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

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

- TMR2 is cleared
- The CCP2 pin is set (exception: if PWM duty cycle = 0%, the CCP2 pin will not be set)
- The PWM duty cycle is latched from CCPR2L into CCPR2H

**Note:** The Timer2 postscalers (see **Section 12.0 “Timer2 Module”**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

# PIC18F6390/6490/8390/8490

## 14.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR2L register and to the CCP2CON<5:4> bits. Up to 10-bit resolution is available. The CCPR2L contains the eight MSbs and the CCP2CON<5:4> bits contain the two LSbs. This 10-bit value is represented by CCPR2L:CCP2CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

### EQUATION 14-2:

$$\text{PWM Duty Cycle} = (\text{CCPR2L:CCP2CON<5:4>}) \cdot T_{\text{OSC}} \cdot (\text{TMR2 Prescale Value})$$

CCPR2L and CCP2CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR2H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR2H is a read-only register.

The CCPR2H 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 CCPR2H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP2 pin is cleared.

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

### EQUATION 14-3:

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

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

**TABLE 14-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz**

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	14	12	10	8	7	6.58

# PIC18F6390/6490/8390/8490

## 14.4.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP2 module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR2L register and CCP2CON<5:4> bits.
3. Make the CCP2 pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCP2 module for PWM operation.

**TABLE 14-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC Data Direction Register								62
TRISE	PORTE Data Direction Register								62
TMR2	Timer2 Register								60
PR2	Timer2 Period Register								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								61
CCPR1H	Capture/Compare/PWM Register 1 High Byte								61
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								61
CCPR2H	Capture/Compare/PWM Register 2 High Byte								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61

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

# PIC18F6390/6490/8390/8490

---

NOTES:

## 15.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

### 15.1 Master SSP (MSSP) Module Overview

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

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I<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

### 15.2 Control Registers

The MSSP module has three associated registers. These include a status register (SSPSTAT) and two control registers (SSPCON1 and SSPCON2). The use of these registers and their individual Configuration bits differ significantly depending on whether the MSSP module is operated in SPI or I<sup>2</sup>C mode.

Additional details are provided under the individual sections.

### 15.3 SPI Mode

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

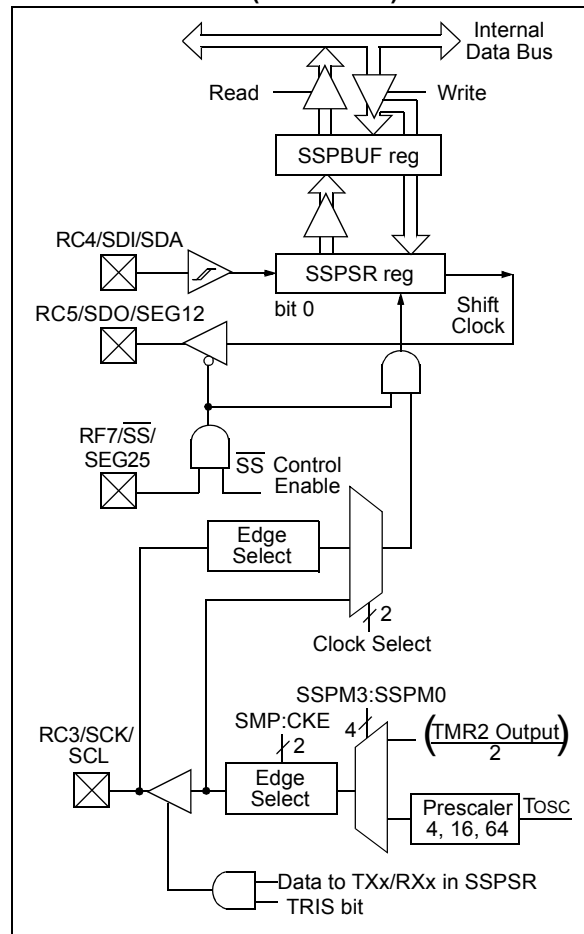
- Serial Data Out (SDO) – RC5/SDO/SEG12
- Serial Data In (SDI) – RC4/SDI/SDA
- Serial Clock (SCK) – RC3/SCK/SCL

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

- Slave Select ( $\overline{SS}$ ) – RF7/ $\overline{SS}$ /SEG25

Figure 15-1 shows the block diagram of the MSSP module when operating in SPI mode.

**FIGURE 15-1: MSSP BLOCK DIAGRAM (SPI MODE)**



# PIC18F6390/6490/8390/8490

## 15.3.1 REGISTERS

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

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper 2 bits of the SSPSTAT are read/write.

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

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

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

### REGISTER 15-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/ $\bar{A}$	P	S	R/ $\bar{W}$	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 Edge Select bit  
When CKP = 0:  
 1 = Data transmitted on rising edge of SCK  
 0 = Data transmitted on falling edge of SCK  
When CKP = 1:  
 1 = Data transmitted on falling edge of SCK  
 0 = Data transmitted on rising edge of SCK

bit 5      **D/ $\bar{A}$ :** 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/ $\bar{W}$ :** 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 (Receive mode only)  
 1 = Receive complete, SSPBUF is full  
 0 = Receive not complete, SSPBUF is empty



# PIC18F6390/6490/8390/8490

**REGISTER 15-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV <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 (Transmit mode only)
  - 1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
  - 0 = No collision
- bit 6      **SSPOV:** Receive Overflow Indicator bit<sup>(1)</sup>

SPI Slave mode:

  - 1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
  - 0 = No overflow
- bit 5      **SSPEN:** Synchronous Serial Port Enable bit<sup>(2)</sup>
  - 1 = Enables serial port and configures SCK, SDO, SDI and  $\overline{SS}$  as serial port pins
  - 0 = Disables serial port and configures these pins as I/O port pins
- bit 4      **CKP:** Clock Polarity Select bit
  - 1 = Idle state for clock is a high level
  - 0 = Idle state for clock is a low level
- bit 3-0    **SSPM3:SSPM0:** Master Synchronous Serial Port Mode Select bits<sup>(3)</sup>
  - 0101 = SPI Slave mode, clock = SCK pin,  $\overline{SS}$  pin control disabled,  $\overline{SS}$  can be used as I/O pin
  - 0100 = SPI Slave mode, clock = SCK pin,  $\overline{SS}$  pin control enabled
  - 0011 = SPI Master mode, clock = TMR2 output/2
  - 0010 = SPI Master mode, clock = Fosc/64
  - 0001 = SPI Master mode, clock = Fosc/16
  - 0000 = SPI Master mode, clock = Fosc/4

- Note 1:** In Master mode, the overflow bit is not set, since each new reception (and transmission) is initiated by writing to the SSPBUF register.
- 2:** When enabled, these pins must be properly configured as inputs or outputs.
- 3:** Bit combinations not specifically listed here are either reserved or implemented in I<sup>2</sup>C™ mode only.

# PIC18F6390/6490/8390/8490

## 15.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

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

The MSSP consists of a Transmit/Receive Shift register (SSPSR) and a Buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then, the Buffer Full detect bit, BF (SSPSTAT<0>), and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before

reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored and the Write Collision Detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. The Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 15-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

### EXAMPLE 15-1: LOADING THE SSPBUF (SSPSR) REGISTER

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

# PIC18F6390/6490/8390/8490

## 15.3.3 ENABLING SPI I/O

To enable the serial port, MSSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPCONx registers and then set the  $\overline{\text{SS}}$  bit. This configures the SDI, SDO, SCK and  $\overline{\text{SS}}$  pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- SDI is automatically controlled by the SPI module
- SDO must have TRISC<5> bit cleared
- SCK (Master mode) must have TRISC<3> bit cleared
- SCK (Slave mode) must have TRISC<3> bit set
- $\overline{\text{SS}}$  must have TRISF<7> bit set

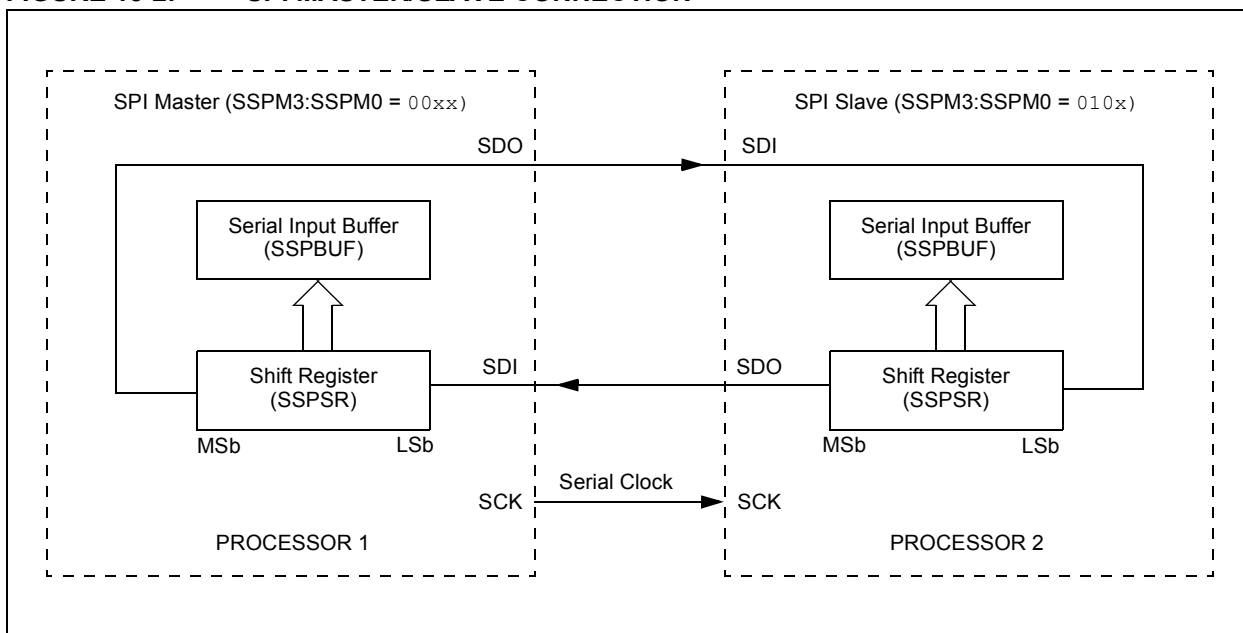
Any serial port function that is not desired may be overridden by programming the corresponding Data Direction (TRIS) register to the opposite value.

## 15.3.4 TYPICAL CONNECTION

Figure 15-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

**FIGURE 15-2: SPI MASTER/SLAVE CONNECTION**



# PIC18F6390/6490/8390/8490

## 15.3.5 MASTER MODE

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

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a "Line Activity Monitor" mode.

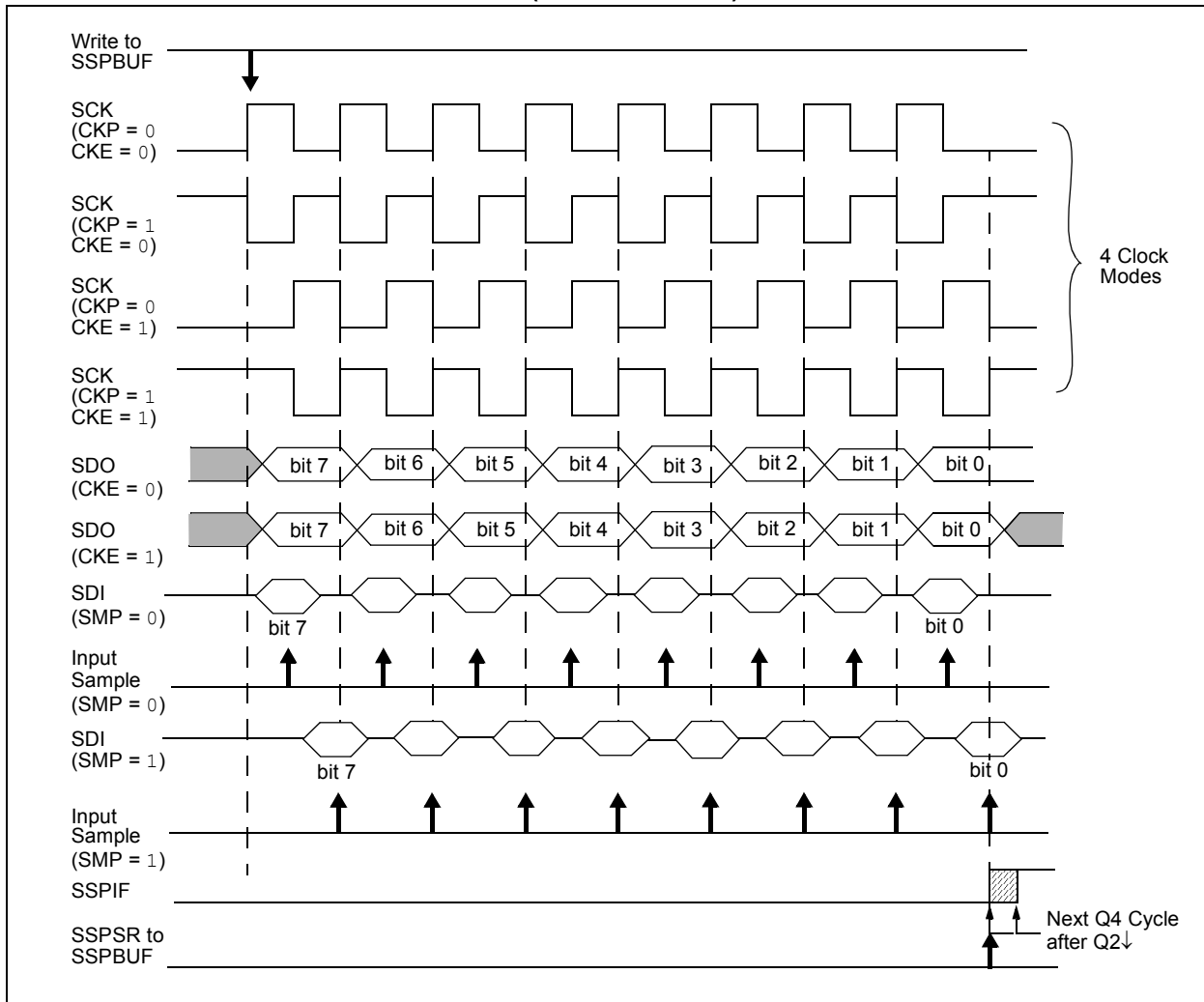
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication, as shown in Figure 15-3, Figure 15-5 and Figure 15-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- $F_{osc}/4$  (or  $T_{CY}$ )
- $F_{osc}/16$  (or  $4 \cdot T_{CY}$ )
- $F_{osc}/64$  (or  $16 \cdot T_{CY}$ )
- $\text{Timer2 output}/2$

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

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

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



## 15.3.6 SLAVE MODE

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

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

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

## 15.3.7 SLAVE SELECT SYNCHRONIZATION

The  $\overline{SS}$  pin allows a Synchronous Slave mode. The SPI must be in Slave mode with  $\overline{SS}$  pin control enabled ( $SSPCON1\langle 3:0 \rangle = 04h$ ). The pin must not be driven low for the  $\overline{SS}$  pin to function as an input. The data latch must be high. When the  $\overline{SS}$  pin is low, transmission and reception are enabled and the SDO pin is driven. When the  $\overline{SS}$  pin goes high, the SDO pin is no longer driven,

even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

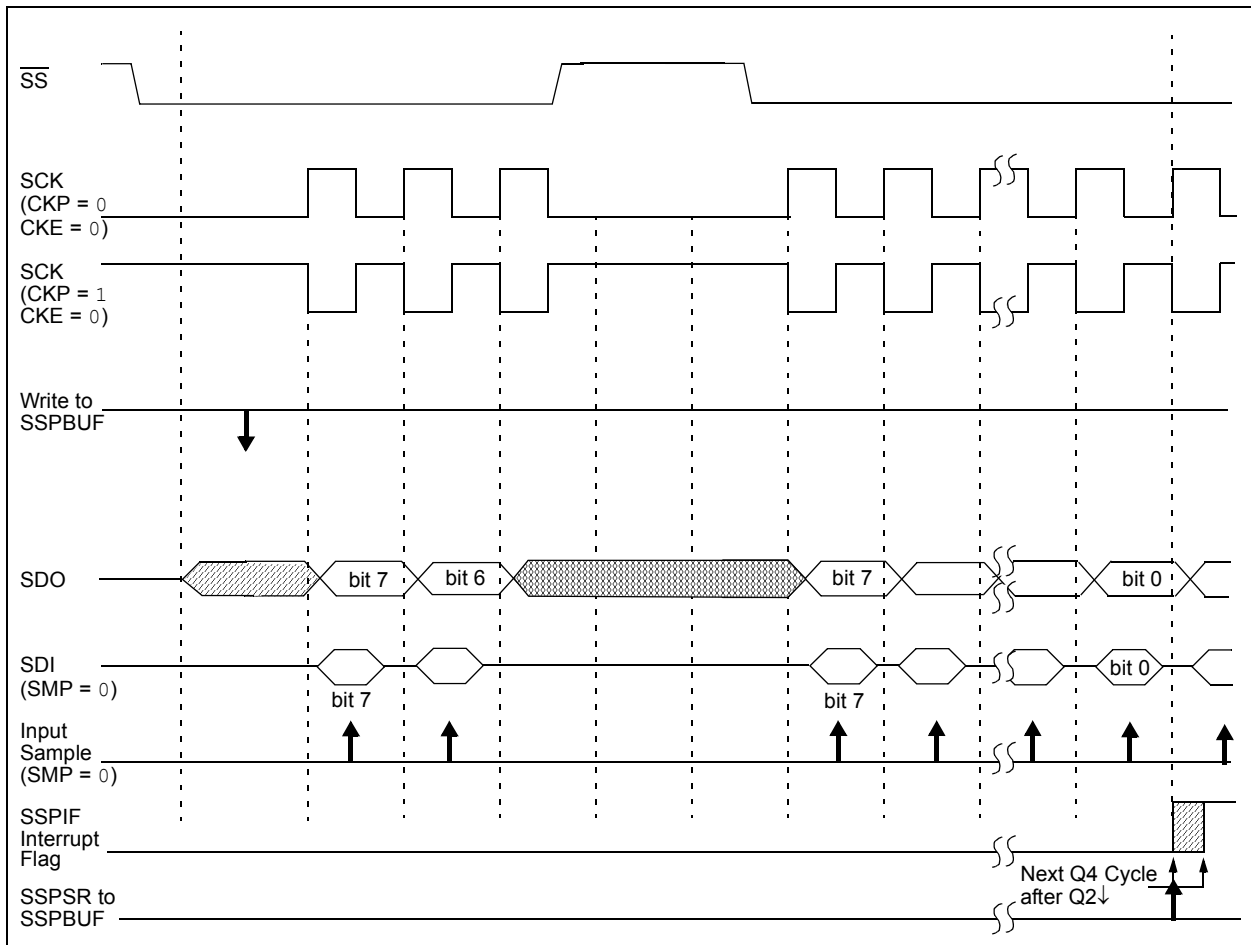
**Note 1:** When the SPI is in Slave mode with  $\overline{SS}$  pin control enabled ( $SSPCON\langle 3:0 \rangle = 0100$ ), the SPI module will reset if the  $\overline{SS}$  pin is set to VDD.

**2:** If the SPI is used in Slave mode with CKE set, then the  $\overline{SS}$  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  $\overline{SS}$  pin to a high level or clearing the SSPEN bit.

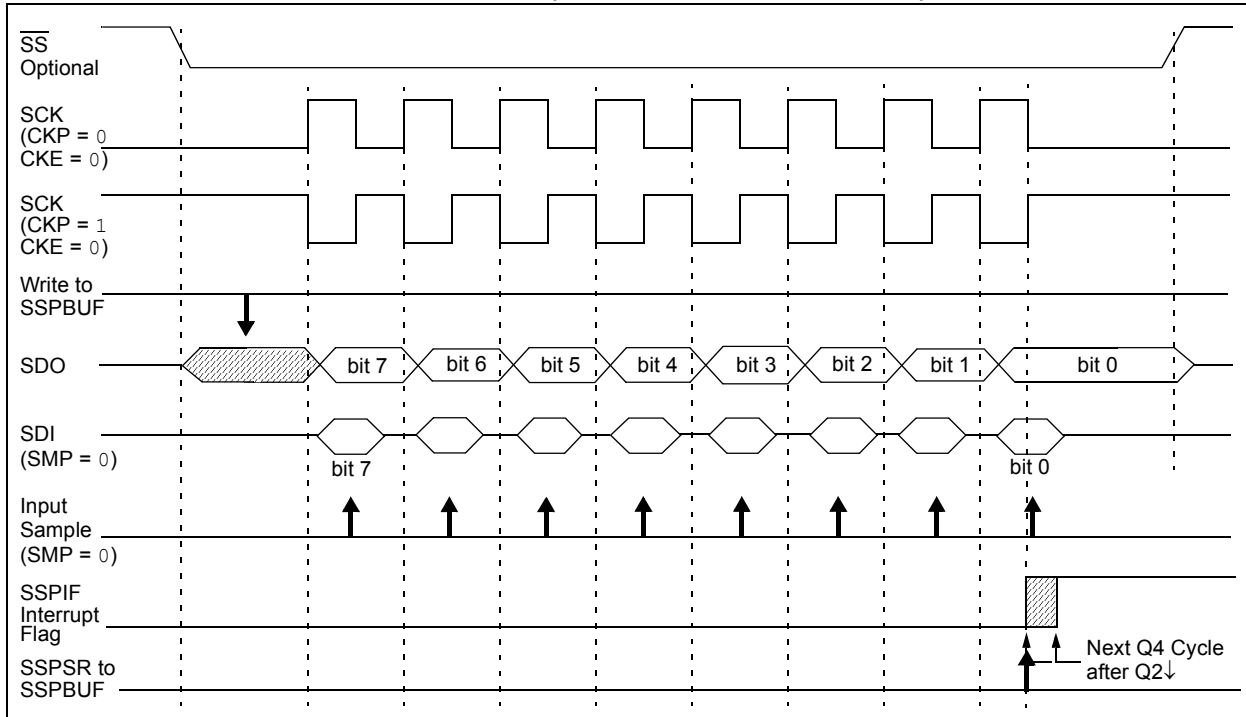
To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver, the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.

**FIGURE 15-4: SLAVE SYNCHRONIZATION WAVEFORM**

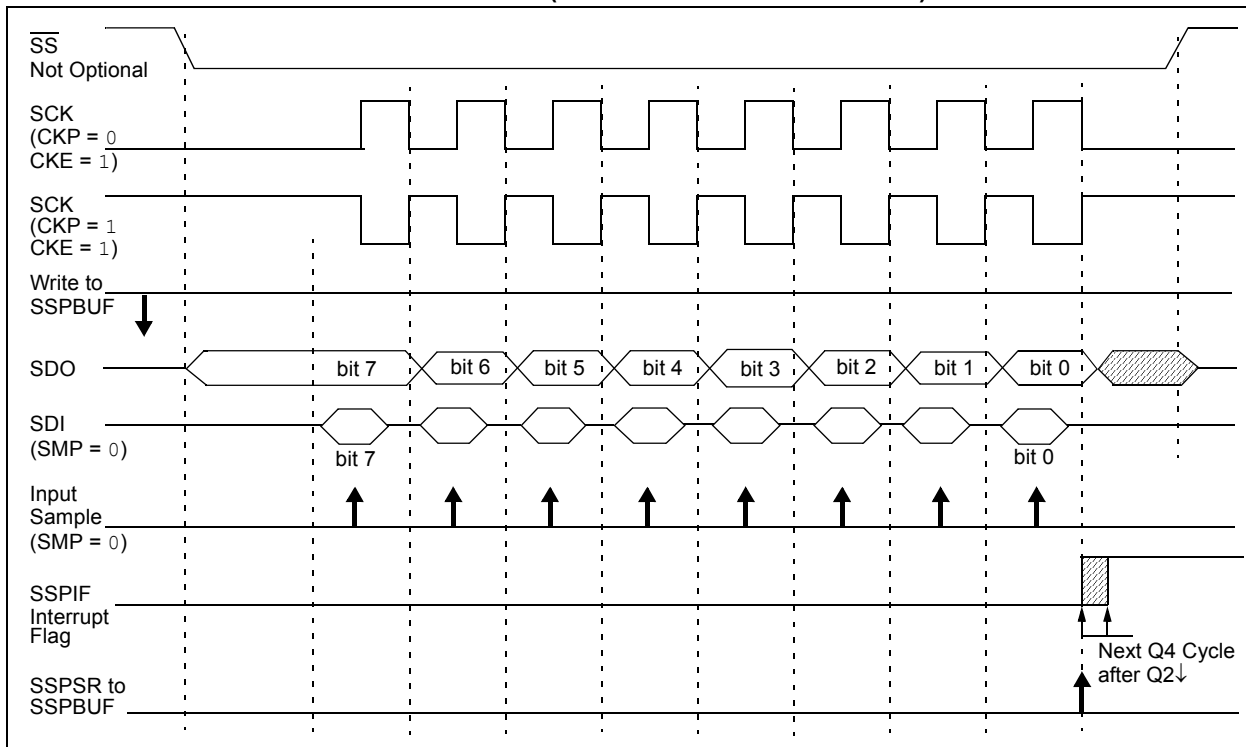


# PIC18F6390/6490/8390/8490

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



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



# PIC18F6390/6490/8390/8490

## 15.3.8 SLEEP OPERATION

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 most power-managed modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTOSC source. See **Section 2.7 “Clock Sources and Oscillator Switching”** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in any power-managed mode and data to be shifted into the SPI Transmit/Receive Shift register. When all 8 bits have been received, the MSSP interrupt flag bit will be set and if enabled, will wake the device.

## 15.3.9 EFFECTS OF A RESET

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

## 15.3.10 BUS MODE COMPATIBILITY

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

**TABLE 15-1: SPI BUS MODES**

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also an SMP bit which controls when the data is sampled.

**TABLE 15-2: REGISTERS ASSOCIATED WITH SPI OPERATION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC Data Direction Register								62
TRISF	PORTF Data Direction Register								62
SSPBUF	MSSP Receive Buffer/Transmit Register								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPSTAT	SMP	CKE	D/Ā	P	S	R/W	UA	BF	60

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

# PIC18F6390/6490/8390/8490

## 15.4 I<sup>2</sup>C Mode

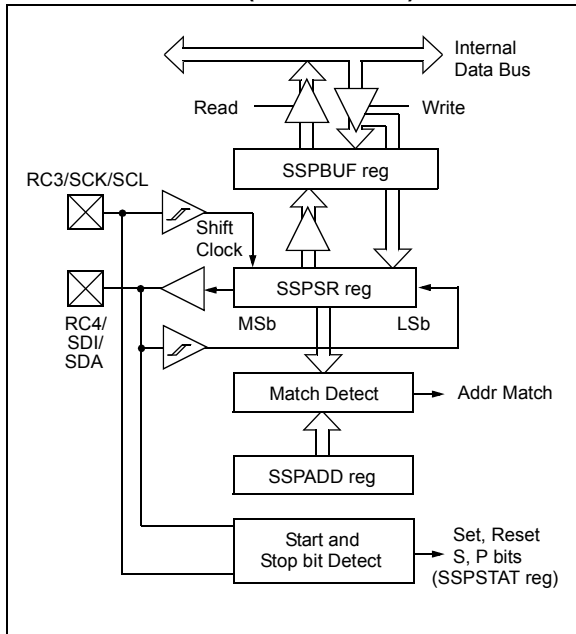
The MSSP module in I<sup>2</sup>C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial clock (SCL) – RC3/SCK/SCL
- Serial data (SDA) – RC4/SDI/SDA

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

**FIGURE 15-7: MSSP BLOCK DIAGRAM (I<sup>2</sup>C™ MODE)**



### 15.4.1 REGISTERS

The MSSP module has six registers for I<sup>2</sup>C operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON1, SSPCON2 and SSPSTAT are the control and status registers in I<sup>2</sup>C mode operation. The SSPCON1 and SSPCON2 registers are readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper 2 bits of the SSPSTAT are read/write.

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

SSPADD register holds the slave device address when the MSSP is configured in I<sup>2</sup>C Slave mode. When the MSSP is configured in Master mode, the lower 7 bits of SSPADD act as the Baud Rate Generator reload value.

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

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



# PIC18F6390/6490/8390/8490

**REGISTER 15-3: SSPSTAT: MSSP STATUS REGISTER (I<sup>2</sup>C™ MODE)**

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
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 SSPADD register  
 0 = Address does not need to be updated
- bit 0      **BF:** Buffer Full Status bit  
In Transmit mode:  
 1 = Receive complete, SSPBUF is full  
 0 = Receive not complete, SSPBUF is empty  
In Receive mode:  
 1 = Data transmit in progress (does not include the ACK and Stop bits), SSPBUF is full  
 0 = Data transmit complete (does not include the ACK and Stop bits), SSPBUF is empty

- Note 1:** This bit is cleared on Reset and when SSPEN is cleared.
- 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 ACK bit.
- 3:** ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Idle mode.

# PIC18F6390/6490/8390/8490

## REGISTER 15-4: SSPCON1: MSSP CONTROL REGISTER 1 (I<sup>2</sup>C™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN <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:** Synchronous Serial Port Enable bit<sup>(1)</sup>  
 1 = Enables the serial port and configures the SDA and SCL pins as the serial port pins  
 0 = Disables serial port and configures these pins as I/O port pins
- bit 4      **CKP:** SCK 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      **SSPM3:SSPM0:** 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)  
 1000 = I<sup>2</sup>C Master mode, clock = Fosc/(4 \* (SSPADD + 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 SDA and SCL pins must be configured as inputs.

**Note 2:** Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

# PIC18F6390/6490/8390/8490

**REGISTER 15-5: SSPCON2: MSSP CONTROL REGISTER 2 (I<sup>2</sup>C™ MODE)**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT <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)  
           1 = Enable interrupt when a general call address (0000h) is received in the SSPSR  
           0 = General call address disabled
- bit 6      **ACKSTAT:** Acknowledge Status bit (Master Transmit mode only)  
           1 = Acknowledge was not received from slave  
           0 = Acknowledge was received from slave
- bit 5      **ACKDT:** Acknowledge Data bit (Master Receive mode only)<sup>(1)</sup>  
           1 = Not Acknowledge  
           0 = Acknowledge
- bit 4      **ACKEN:** Acknowledge Sequence Enable bit (Master Receive mode only)<sup>(2)</sup>  
           1 = Initiate Acknowledge sequence on SDA and SCL pins and transmit ACKDT data bit. Automatically cleared by hardware.  
           0 = Acknowledge sequence Idle
- bit 3      **RCEN:** Receive Enable bit (Master mode only)<sup>(2)</sup>  
           1 = Enables Receive mode for I<sup>2</sup>C  
           0 = Receive Idle
- bit 2      **PEN:** Stop Condition Enable bit (Master mode only)<sup>(2)</sup>  
           1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.  
           0 = Stop condition Idle
- bit 1      **RSEN:** Repeated Start Condition Enable bit (Master mode only)<sup>(2)</sup>  
           1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.  
           0 = Repeated Start condition Idle
- bit 0      **SEN:** Start Condition Enable/Stretch Enable bit<sup>(2)</sup>  
           In Master mode:  
           1 = Initiate Start condition on SDA and SCL pins. Automatically cleared by hardware.  
           0 = Start condition Idle  
           In Slave mode:  
           1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)  
           0 = Clock stretching is disabled

- Note 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 not in the Idle mode, these bits may not be set (no spooling) and the SSPBUF may not be written (or writes to the SSPBUF are disabled).

# PIC18F6390/6490/8390/8490

## 15.4.2 OPERATION

The MSSP module functions are enabled by setting MSSP Enable bit, SSPEN (SSPCON1<5>).

The SSPCON1 register allows control of the I<sup>2</sup>C operation. Four mode selection bits (SSPCON1<3:0>) allow one of the following I<sup>2</sup>C modes to be selected:

- I<sup>2</sup>C Master mode, clock = (Fosc/4) x (SSPADD + 1)
- 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 SCL and SDA pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCL and SDA pins.

## 15.4.3 SLAVE MODE

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

The I<sup>2</sup>C Slave mode hardware will always generate an interrupt on an address match. Through the mode select bits, the user can also choose to interrupt on Start and Stop bits

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

Any combination of the following conditions will cause the MSSP module not to give this  $\overline{ACK}$  pulse:

- The Buffer Full bit, BF (SSPSTAT<0>), was set before the transfer was received.
- The overflow bit, SSPOV (SSPCON1<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but the SSPIF bit (PIR1<3>) is set. The BF bit is cleared by reading the SSPBUF register, while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low time 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.

## 15.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPSR register value is loaded into the SSPBUF register.
2. The Buffer Full bit, BF, is set.
3. An  $\overline{ACK}$  pulse is generated.
4. MSSP Interrupt Flag bit, SSPIF (PIR1<3>), is set (interrupt is generated, if enabled) on the falling edge of the ninth SCL pulse.

In 10-Bit 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 (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '11110 A9 A8 0', where 'A9' and 'A8' are the two MSBs of the address. The sequence of events for 10-Bit Addressing mode is as follows, with steps 7 through 9 for the slave-transmitter:

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

## 15.4.3.2 Reception

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

When the address byte overflow condition exists, then the no Acknowledge ( $\overline{ACK}$ ) pulse is given. An overflow condition is defined as either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON1<6>) is set.

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

If SEN is enabled (SSPCON2<0> = 1), RC3/SCK/SCL will be held low (clock stretch) following each data transfer. The clock must be released by setting bit CKP (SSPCON1<4>). See **Section 15.4.4 “Clock Stretching”** for more detail.

## 15.4.3.3 Transmission

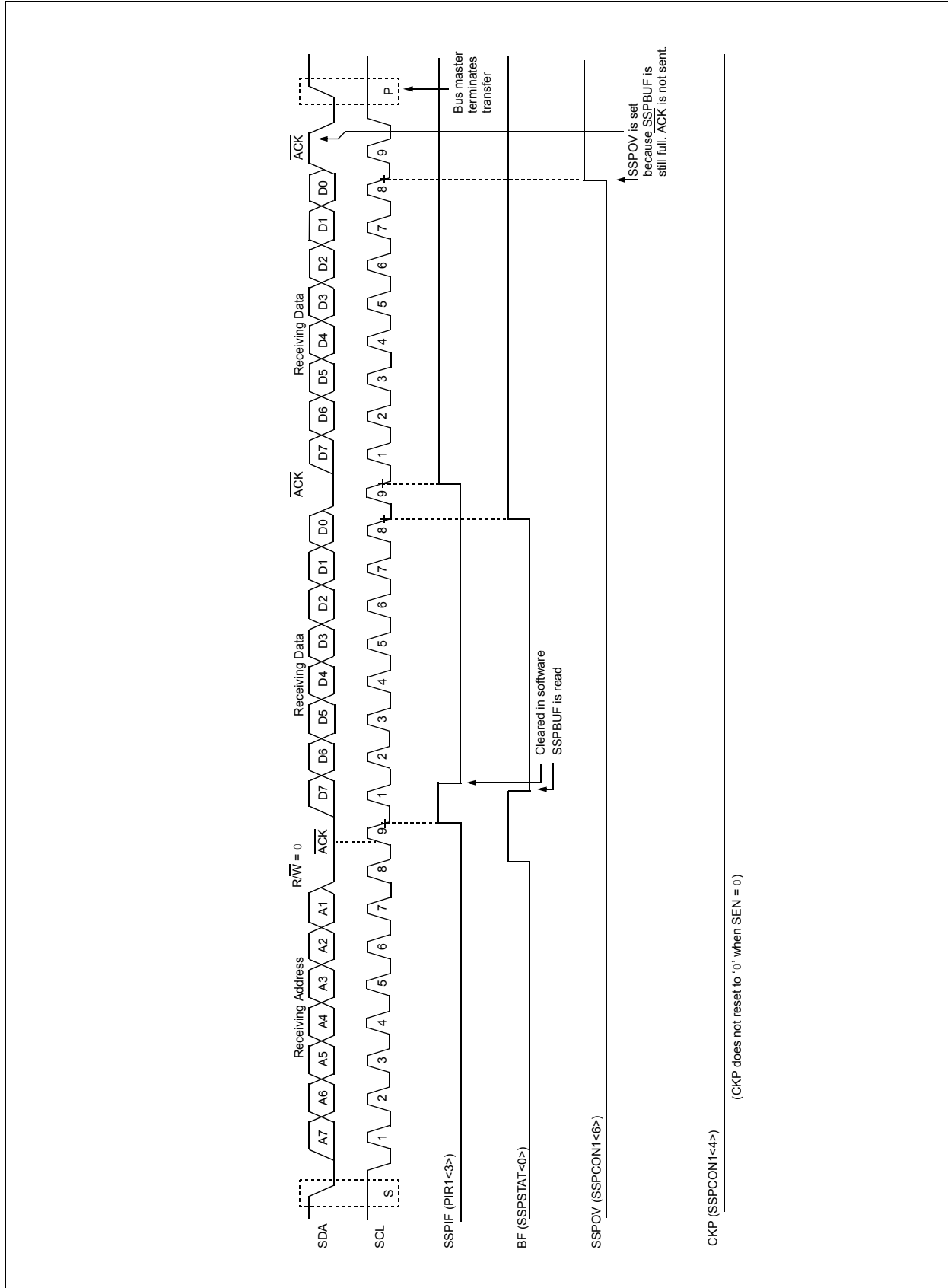
When the  $\overline{R/W}$  bit of the incoming address byte is set and an address match occurs, the  $\overline{R/W}$  bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The  $\overline{ACK}$  pulse will be sent on the ninth bit and pin RC3/SCK/SCL is held low regardless of SEN (see **Section 15.4.4 “Clock Stretching”** for more detail). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then, pin RC3/SCK/SCL should be enabled by setting bit, CKP (SSPCON1<4>). The 8 data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 15-9).

The  $\overline{ACK}$  pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not  $\overline{ACK}$ ), then the data transfer is complete. In this case, when the  $\overline{ACK}$  is latched by the slave, the slave logic is reset and the slave monitors for another occurrence of the Start bit. If the SDA line was low ( $\overline{ACK}$ ), the next transmit data must be loaded into the SSPBUF register. Again, pin RC3/SCK/SCL must be enabled by setting bit CKP.

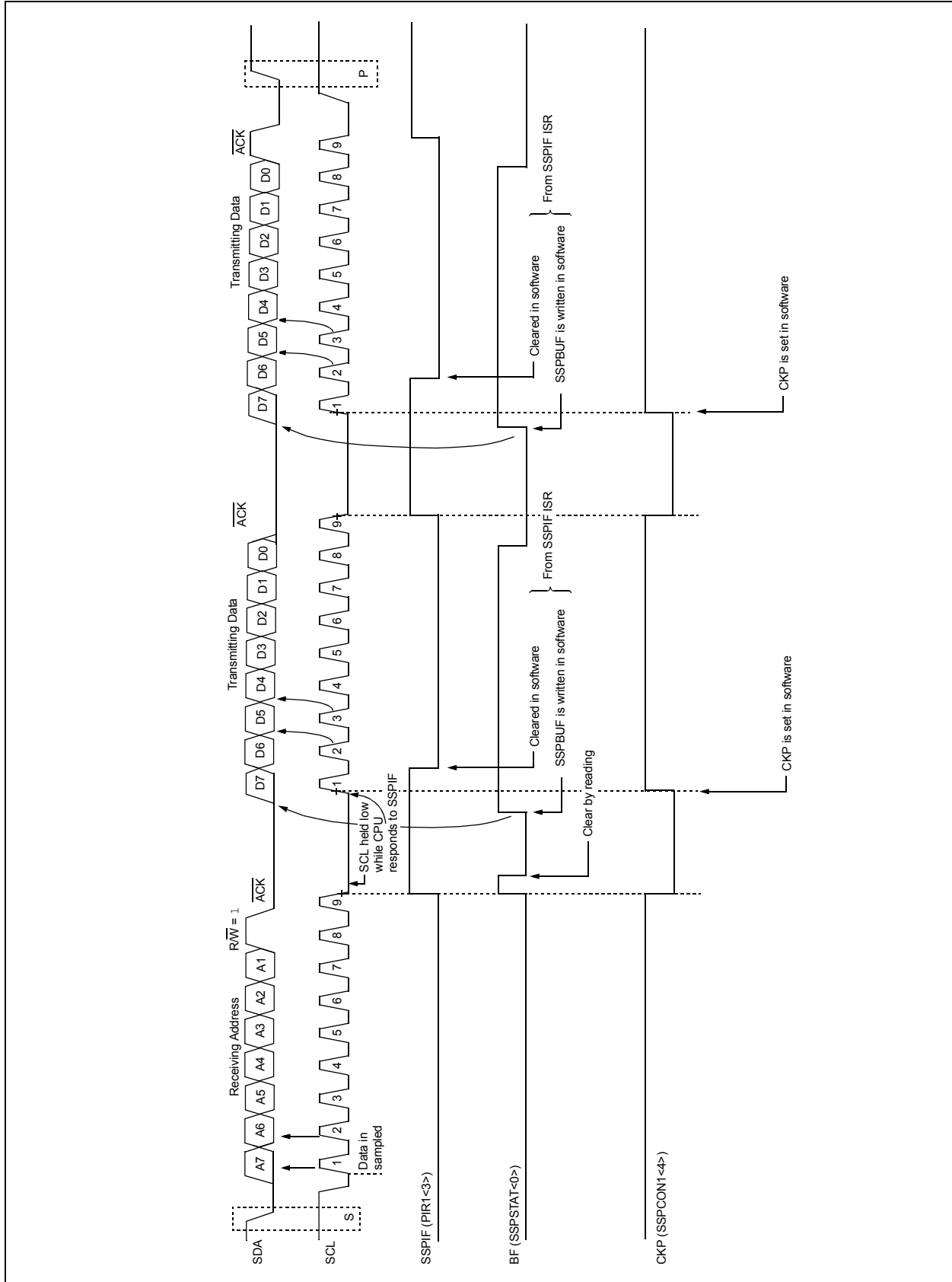
An MSSP interrupt is generated for each data transfer byte. The SSPIF bit must be cleared in software and the SSPSTAT register is used to determine the status of the byte. The SSPIF bit is set on the falling edge of the ninth clock pulse.

# PIC18F6390/6490/8390/8490

FIGURE 15-8: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)

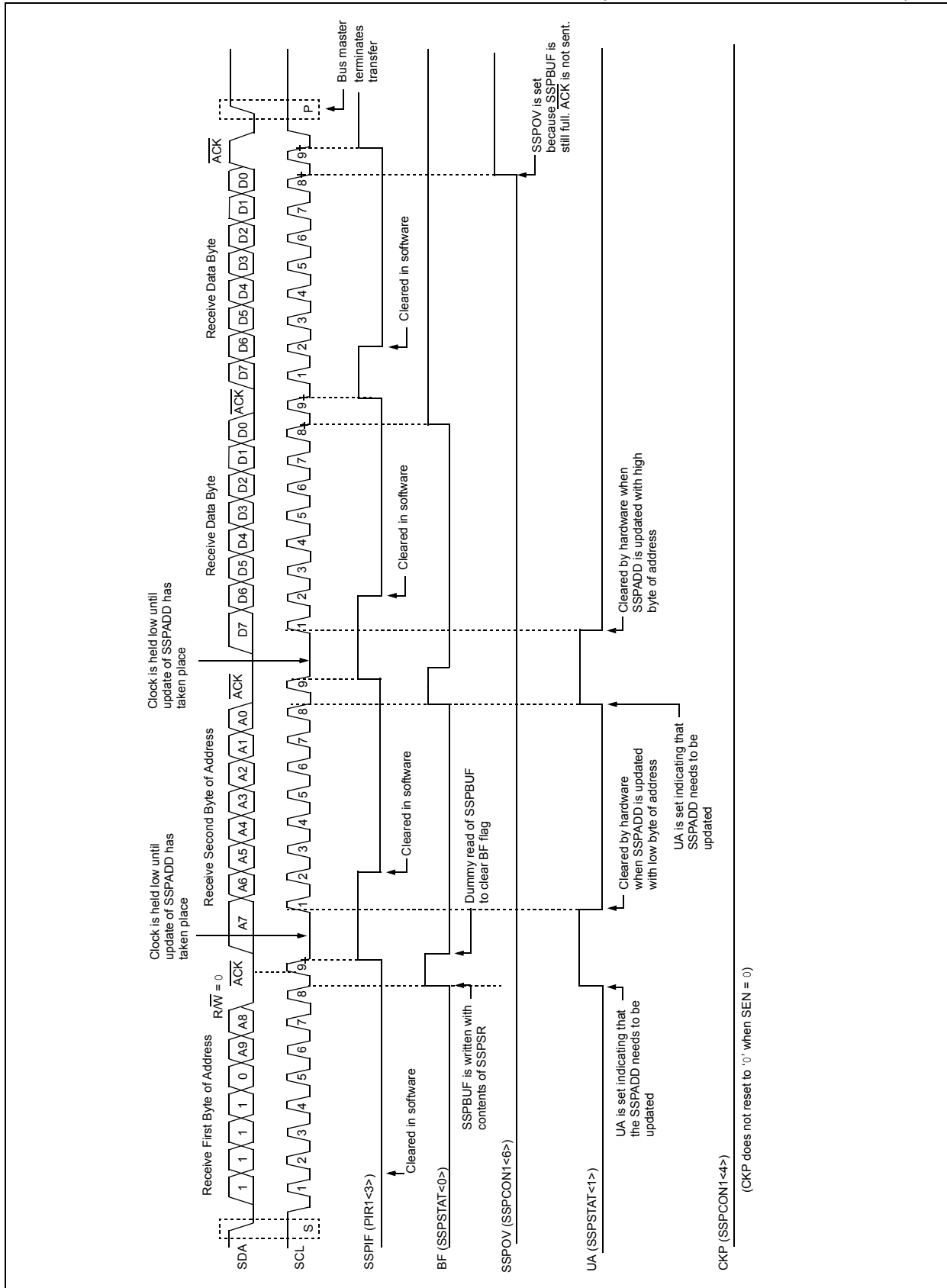


**FIGURE 15-9: I<sup>2</sup>C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)**



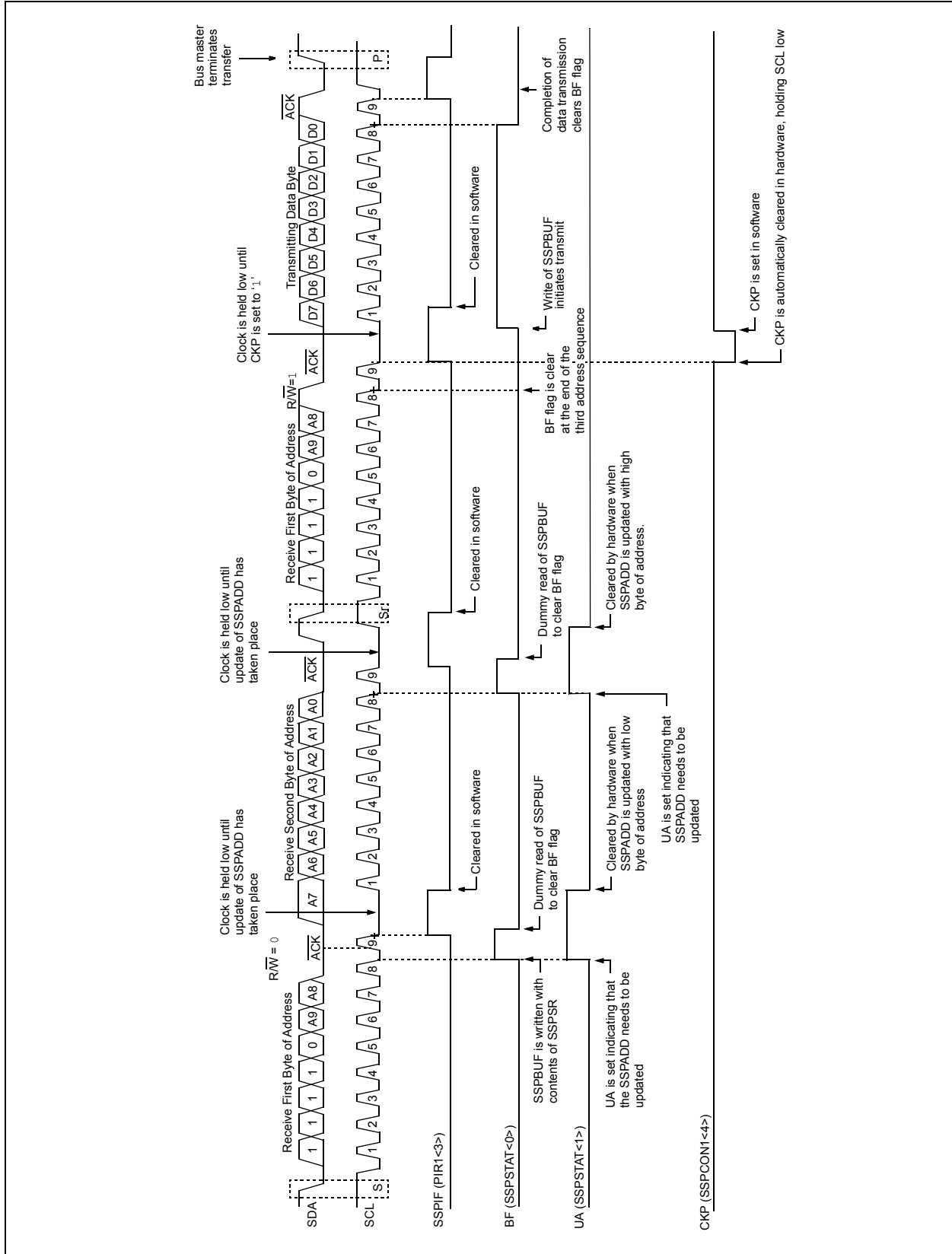
# PIC18F6390/6490/8390/8490

FIGURE 15-10: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)





**FIGURE 15-11: I<sup>2</sup>C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)**



# PIC18F6390/6490/8390/8490

## 15.4.4 CLOCK STRETCHING

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

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

### 15.4.4.1 Clock Stretching for 7-Bit Slave Receive Mode (SEN = 1)

In 7-Bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence, if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 15-13).

**Note 1:** If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.

**2:** The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

### 15.4.4.2 Clock Stretching for 10-Bit Slave Receive Mode (SEN = 1)

In 10-Bit Slave Receive mode during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/W bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

**Note:** If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

### 15.4.4.3 Clock Stretching for 7-Bit Slave Transmit Mode

7-Bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock, if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 15-9).

**Note 1:** If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.

**2:** The CKP bit can be set in software regardless of the state of the BF bit.

### 15.4.4.4 Clock Stretching for 10-Bit Slave Transmit Mode

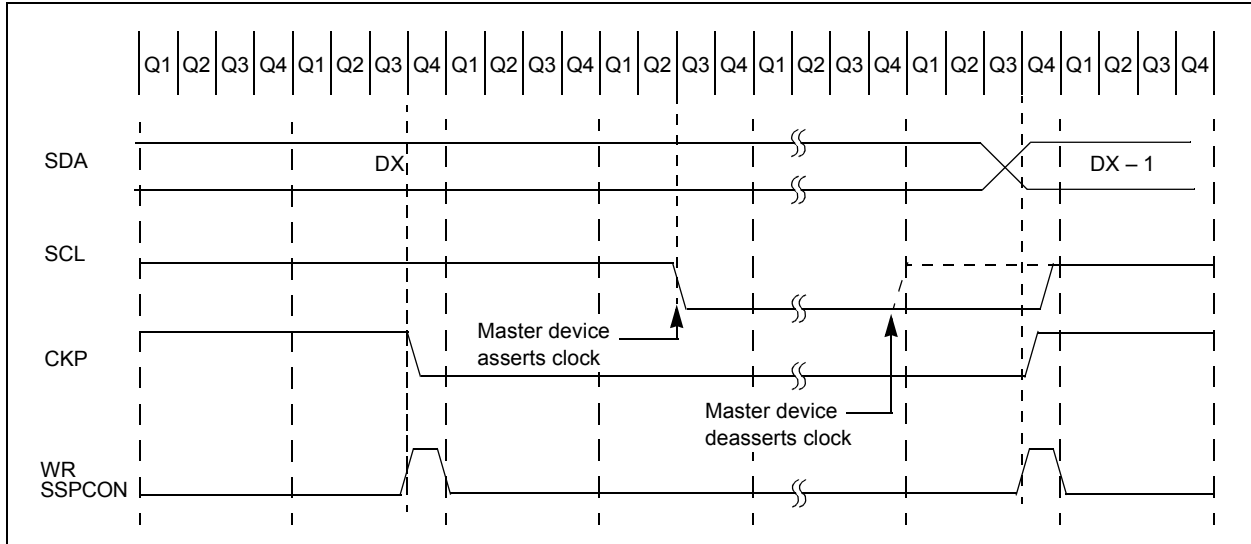
In 10-Bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-Bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high-order bits of the 10-bit address and the R/W bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-Bit Slave Transmit mode (see Figure 15-11).

## 15.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCL output is forced to '0'. However, setting the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external I<sup>2</sup>C master device has

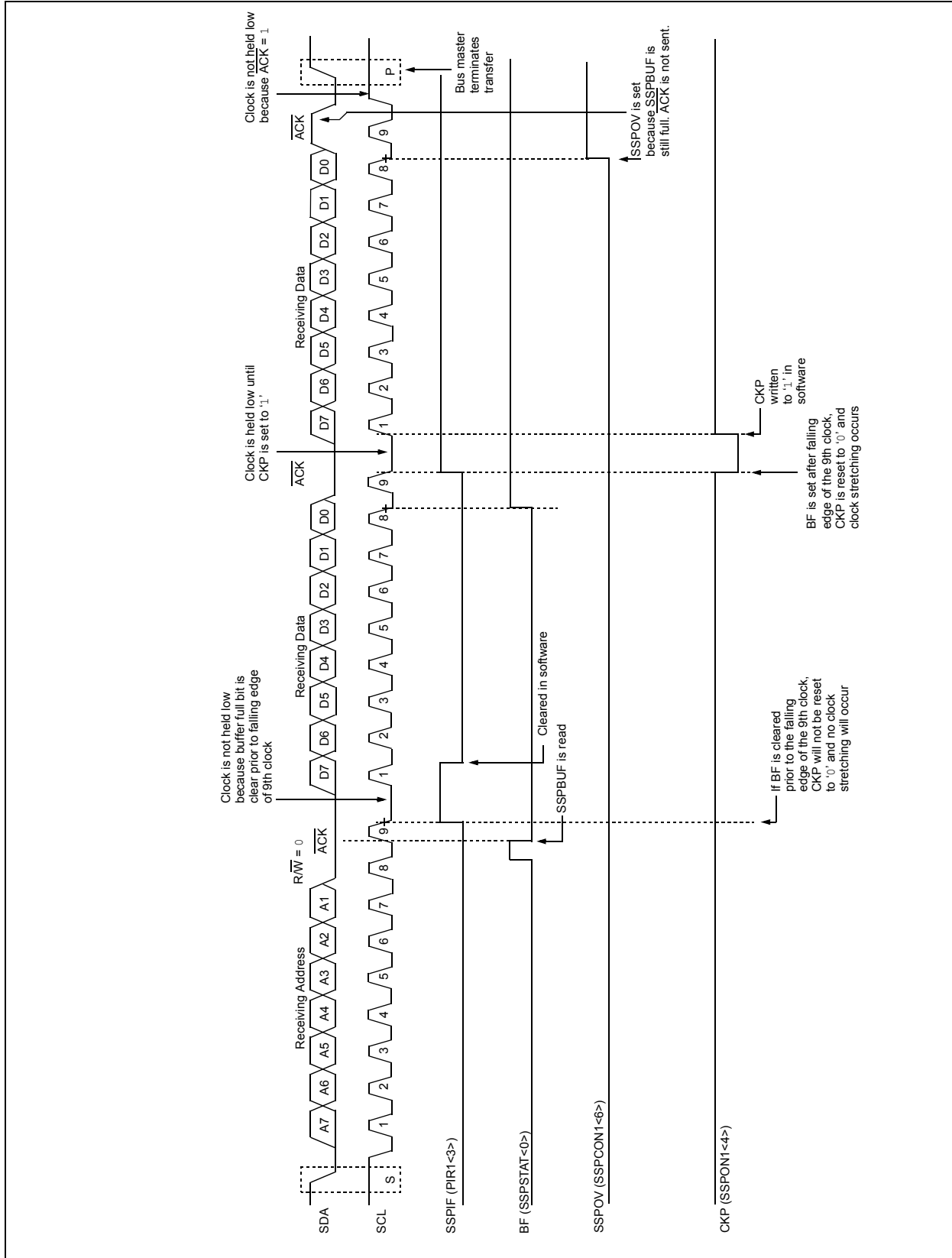
already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the I<sup>2</sup>C bus have deasserted SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 15-12).

**FIGURE 15-12: CLOCK SYNCHRONIZATION TIMING**

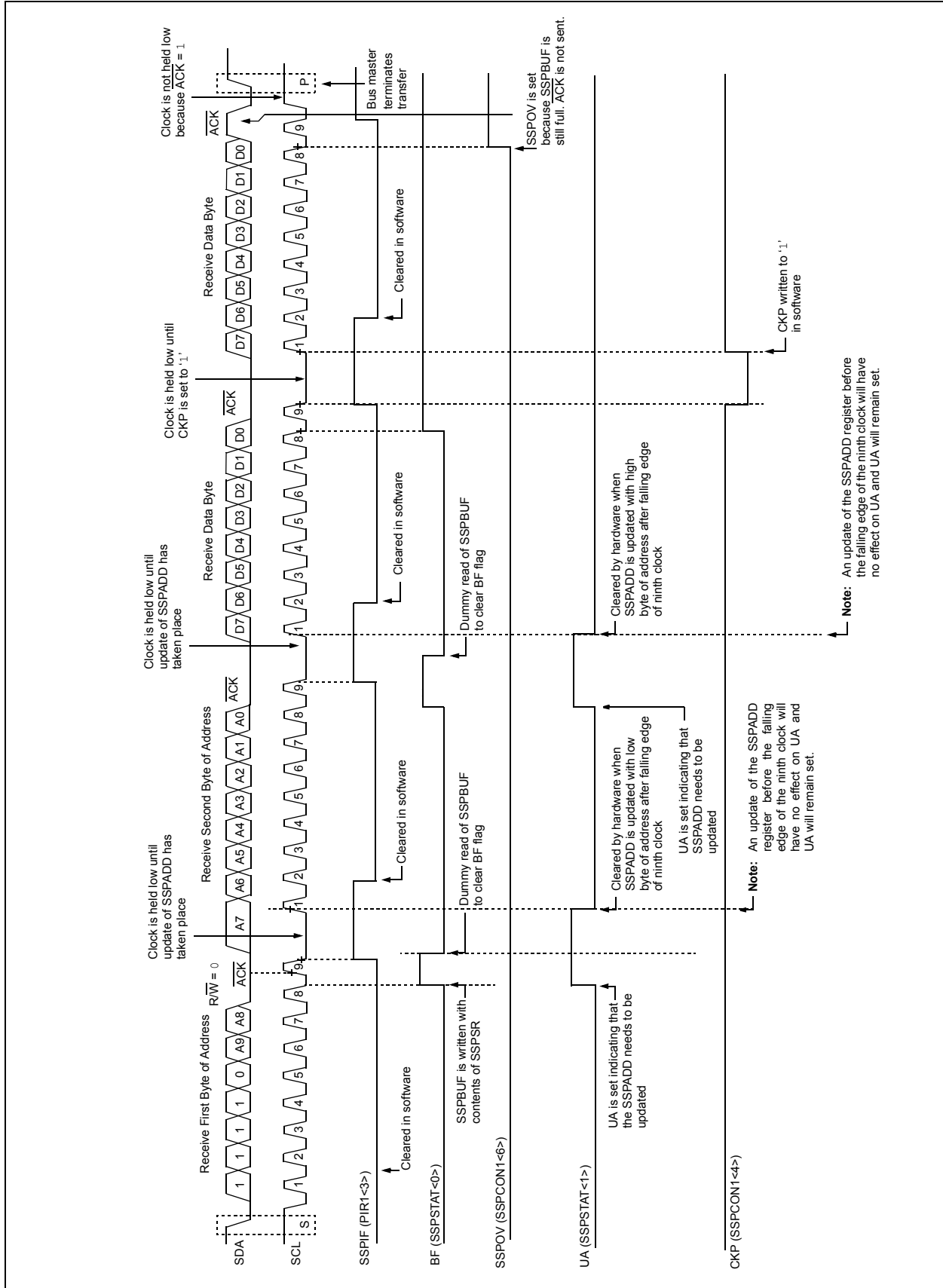


# PIC18F6390/6490/8390/8490

FIGURE 15-13: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)



**FIGURE 15-14: I<sup>2</sup>C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)**



# PIC18F6390/6490/8390/8490

## 15.4.5 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.

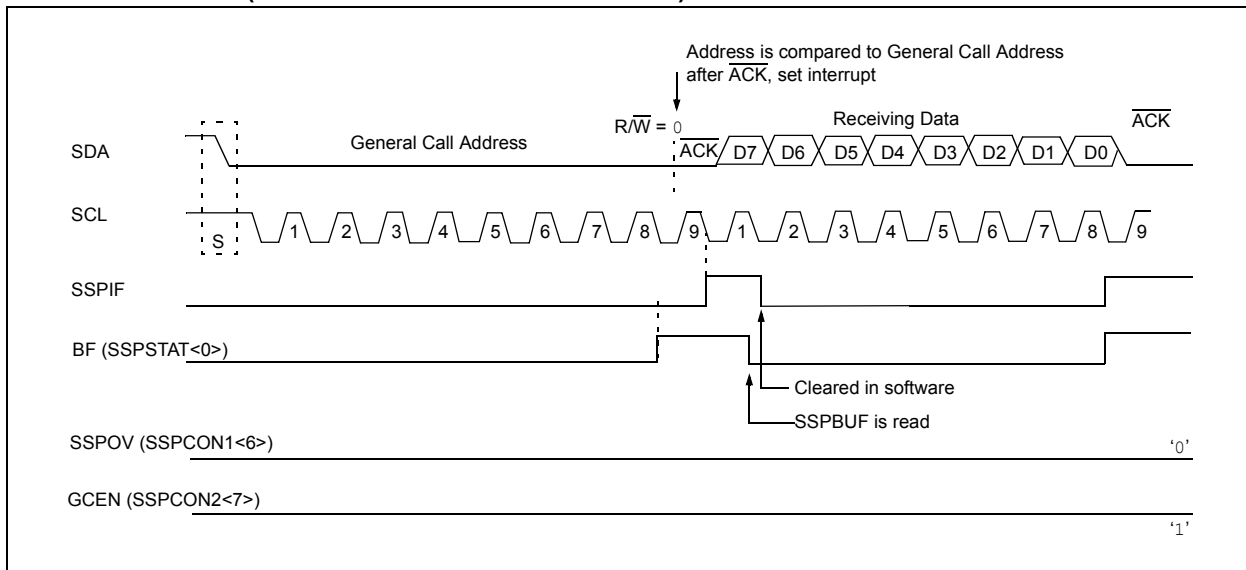
The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit ( $\overline{ACK}$  bit), the SSPIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPBUF. The value can be used to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPADD is required to be updated for the second half of the address to match and the UA bit is set (SSPSTAT<1>). If the general call address is sampled when the GCEN bit is set, while the slave is configured in 10-Bit 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 15-15).

**FIGURE 15-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESSING MODE)**



# PIC18F6390/6490/8390/8490

## 15.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I<sup>2</sup>C bus may be taken when the P bit is set or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I<sup>2</sup>C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

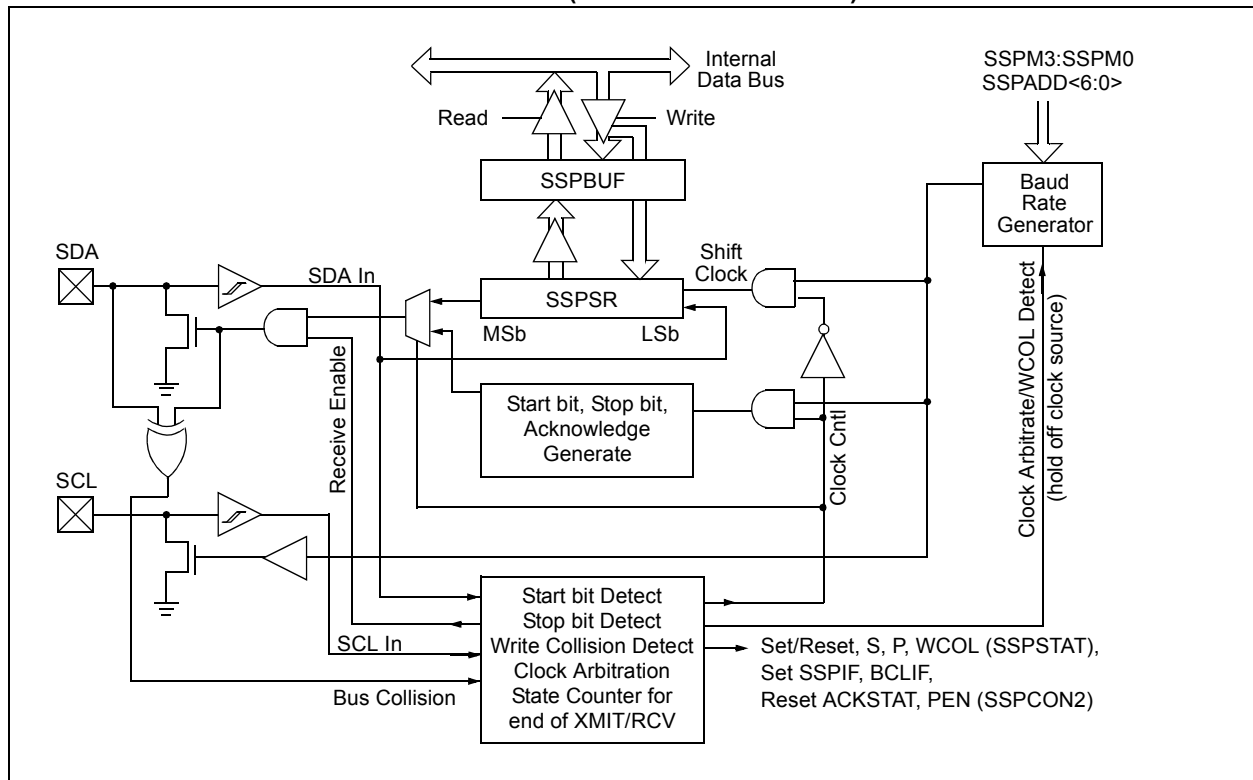
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I<sup>2</sup>C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

**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 SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

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

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmit
- Repeated Start

**FIGURE 15-16: MSSP BLOCK DIAGRAM (I<sup>2</sup>C™ MASTER MODE)**



# PIC18F6390/6490/8390/8490

---

## 15.4.6.1 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 SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/W) bit. In this case, the R/W bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address, followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator used for the SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz or 1 MHz I<sup>2</sup>C operation. See **Section 15.4.7 "Baud Rate"** for more detail.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start Enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. Address is shifted out the SDA pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with 8 bits of data.
8. Data is shifted out the SDA pin until all 8 bits are transmitted.
9. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
11. The user generates a Stop condition by setting the Stop Enable bit, PEN (SSPCON2<2>).
12. Interrupt is generated once the Stop condition is complete.



# PIC18F6390/6490/8390/8490

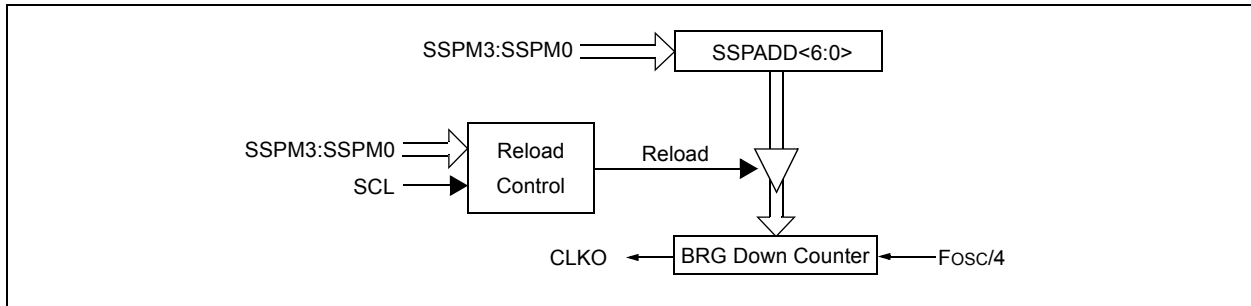
## 15.4.7 BAUD RATE

In I<sup>2</sup>C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPADD register (Figure 15-17). When a write occurs to SSPBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (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 ACK), the internal clock will automatically stop counting and the SCL pin will remain in its last state.

Table 15-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD.

**FIGURE 15-17: BAUD RATE GENERATOR BLOCK DIAGRAM**



**TABLE 15-3: I<sup>2</sup>C™ CLOCK RATE w/BRG**

F <sub>cy</sub>	F <sub>cy</sub> * 2	BRG Value	F <sub>scl</sub> (2 Rollovers of BRG)
10 MHz	20 MHz	19h	400 kHz <sup>(1)</sup>
10 MHz	20 MHz	20h	312.5 kHz
10 MHz	20 MHz	3Fh	100 kHz
4 MHz	8 MHz	0Ah	400 kHz <sup>(1)</sup>
4 MHz	8 MHz	0Dh	308 kHz
4 MHz	8 MHz	28h	100 kHz
1 MHz	2 MHz	03h	333 kHz <sup>(1)</sup>
1 MHz	2 MHz	0Ah	100 kHz
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.

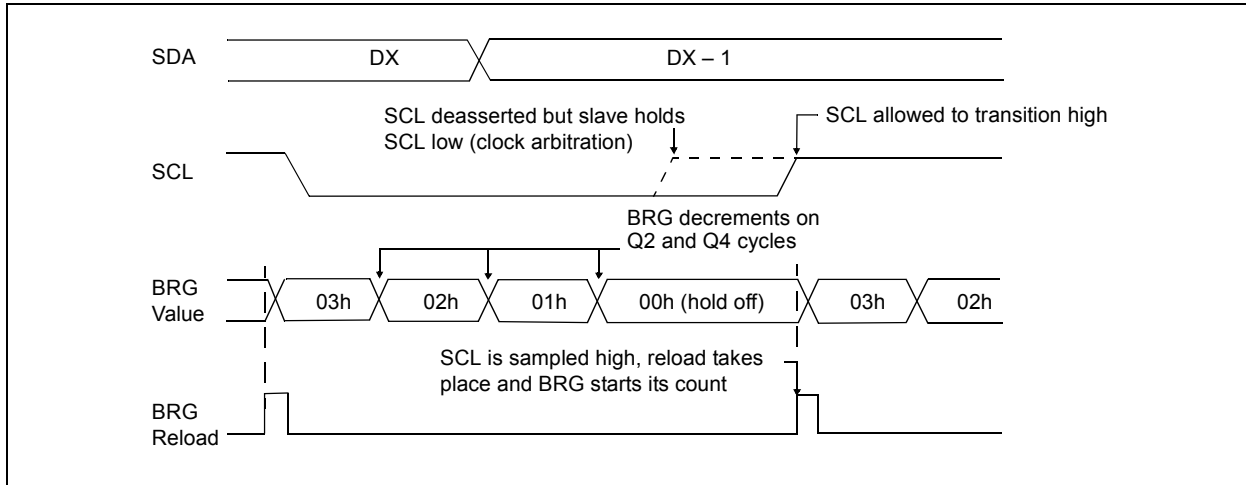
# PIC18F6390/6490/8390/8490

## 15.4.7.1 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the

SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 15-18).

**FIGURE 15-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**



## 15.4.8 I<sup>2</sup>C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Condition Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware, the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

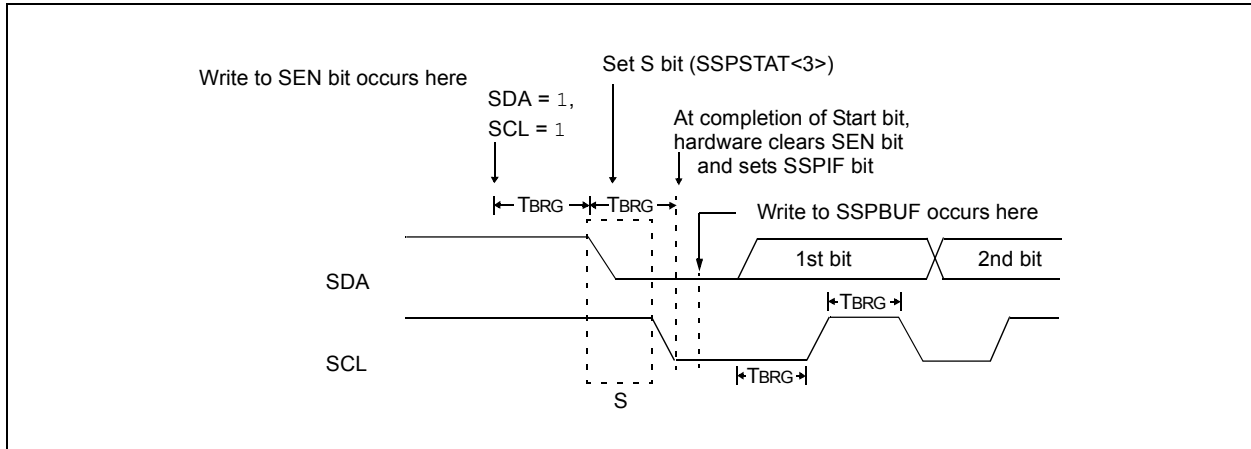
**Note:** If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I<sup>2</sup>C module is reset into its Idle state.

### 15.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

**Note:** Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.

**FIGURE 15-19: FIRST START BIT TIMING**



# PIC18F6390/6490/8390/8490

## 15.4.9 I<sup>2</sup>C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<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 SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPADD<5:0> and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = 0) for one TBRG while SCL is high. Following this, the RSEN bit (SSPCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the Baud Rate Generator has timed out.

**Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.

**2:** A bus collision during the Repeated Start condition occurs if:

- SDA is sampled low when SCL goes from low-to-high.
- SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data '1'.

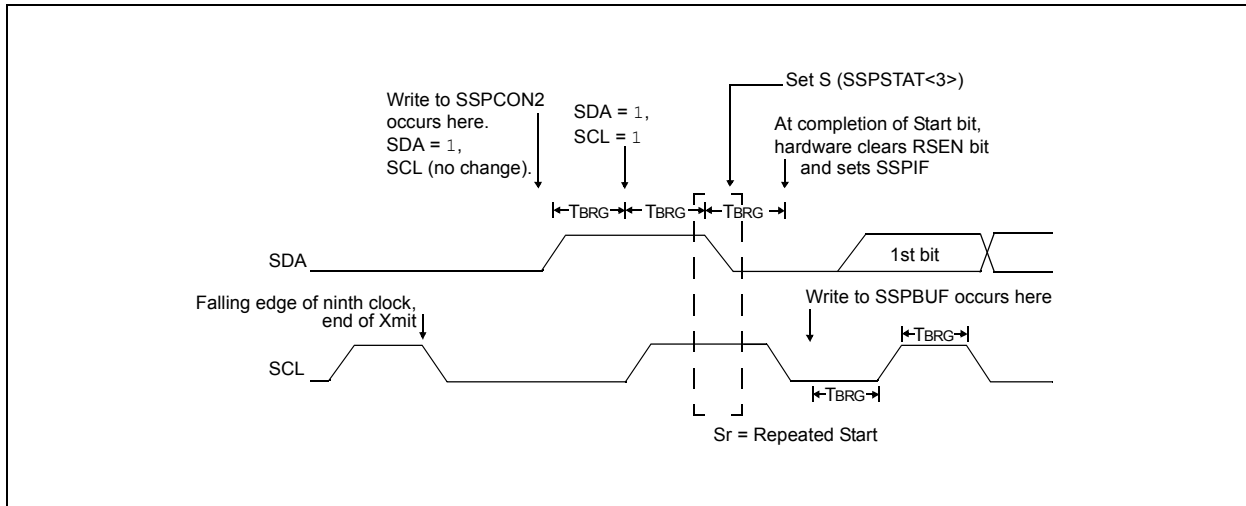
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first 8 bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or 8 bits of data (7-bit mode).

### 15.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

**Note:** Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

**FIGURE 15-20: REPEAT START CONDITION WAVEFORM**



## 15.4.10 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 SSPBUF register. This action will set the Buffer Full flag bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted (see data hold time specification parameter #106). SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high (see data setup time specification parameter #107). When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. This allows the slave device being addressed to respond with an  $\overline{\text{ACK}}$  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 SSPIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPBUF, leaving SCL low and SDA unchanged (Figure 15-21).

After the write to the SSPBUF, each bit of address will be shifted out on the falling edge of SCL until all 7 address bits and the R/W bit are completed. On the falling edge of the eighth clock, the master will deassert the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the  $\overline{\text{ACK}}$  bit is loaded into the ACKSTAT status bit (SSPCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

### 15.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPSTAT<0>) is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

### 15.4.10.2 WCOL Status Flag

If the user writes the SSPBUF when a transmit is already in progress (i.e., SSPSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

WCOL must be cleared in software.

### 15.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPCON2<6>) is cleared when the slave has sent an Acknowledge ( $\overline{\text{ACK}} = 0$ ) and is set when the slave does not Acknowledge ( $\overline{\text{ACK}} = 1$ ). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

## 15.4.11 I<sup>2</sup>C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPCON2<3>).

<b>Note:</b>	The MSSP module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded.
--------------	--

The Baud Rate Generator begins counting and on each rollover, the state of the SCL pin changes (high-to-low/low-to-high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR are loaded into the SSPBUF, the BF flag bit is set, the SSPIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCL low. The MSSP is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>).

### 15.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPBUF from SSPSR. It is cleared when the SSPBUF register is read.

### 15.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

### 15.4.11.3 WCOL Status Flag

If the user writes the SSPBUF when a receive is already in progress (i.e., SSPSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

# PIC18F6390/6490/8390/8490

FIGURE 15-21: I<sup>2</sup>C™ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)

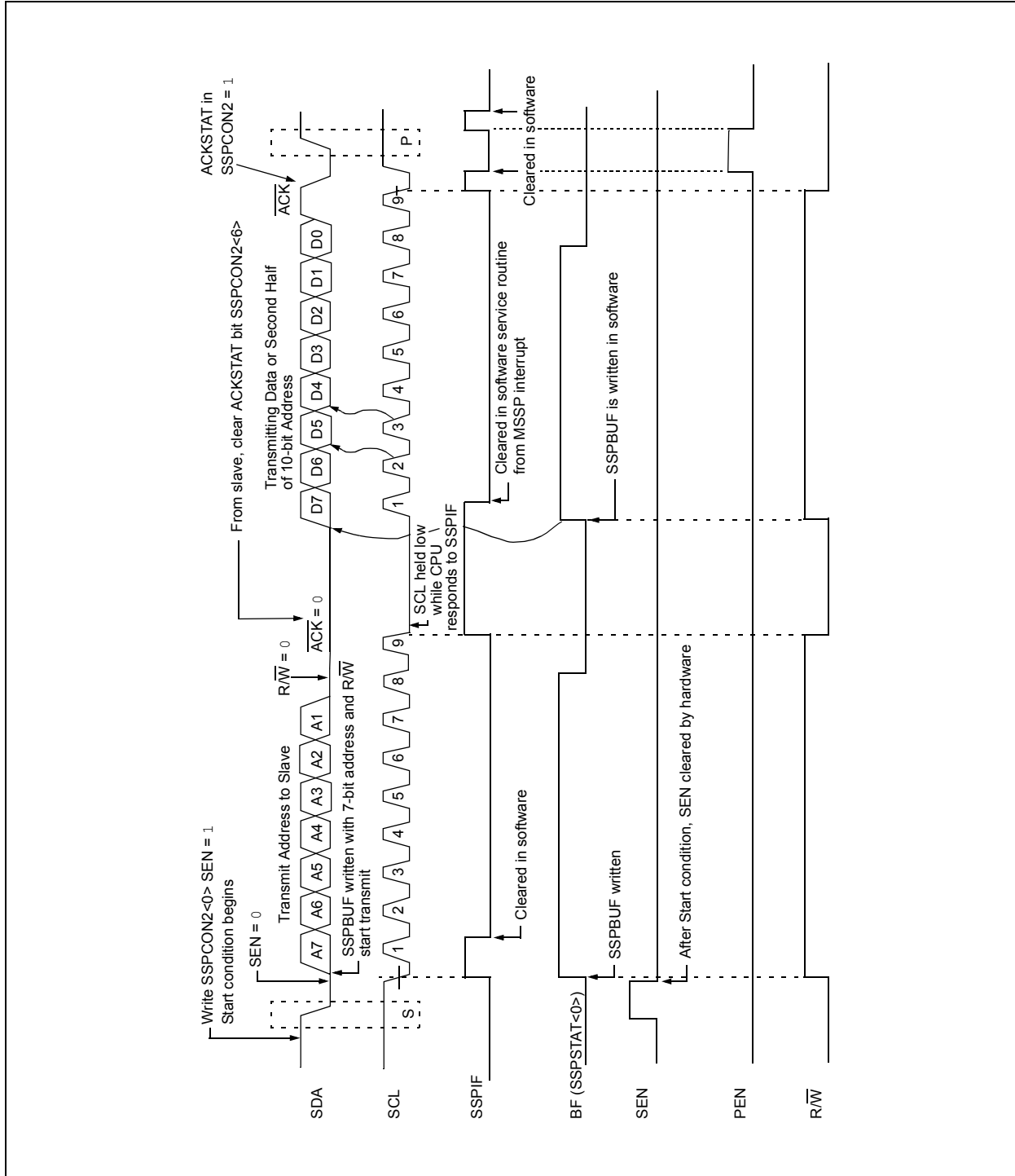
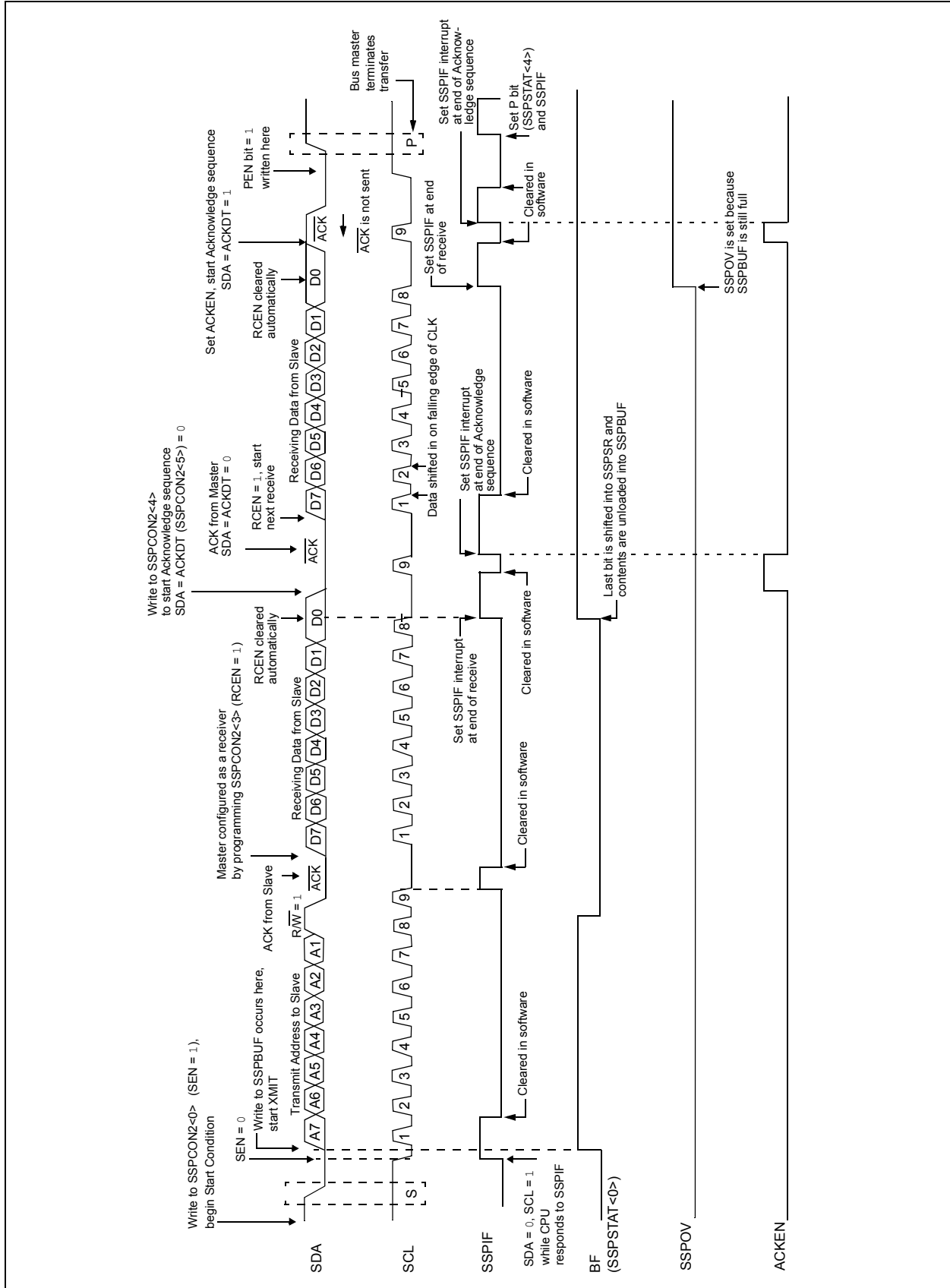


FIGURE 15-22: I<sup>2</sup>C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



# PIC18F6390/6490/8390/8490

## 15.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 15-23).

### 15.4.12.1 WCOL Status Flag

If the user writes the SSPBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

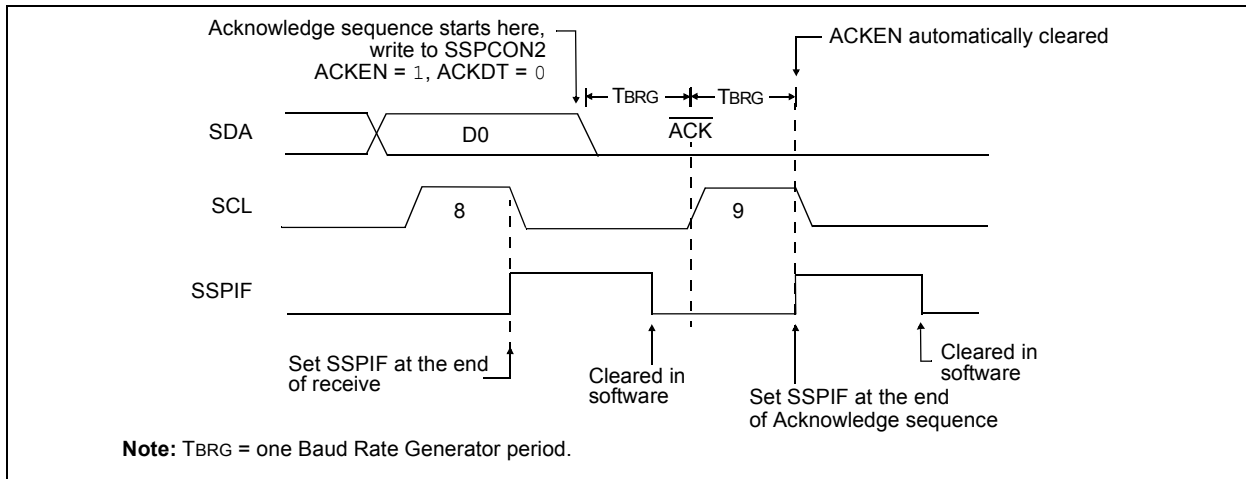
## 15.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 15-24).

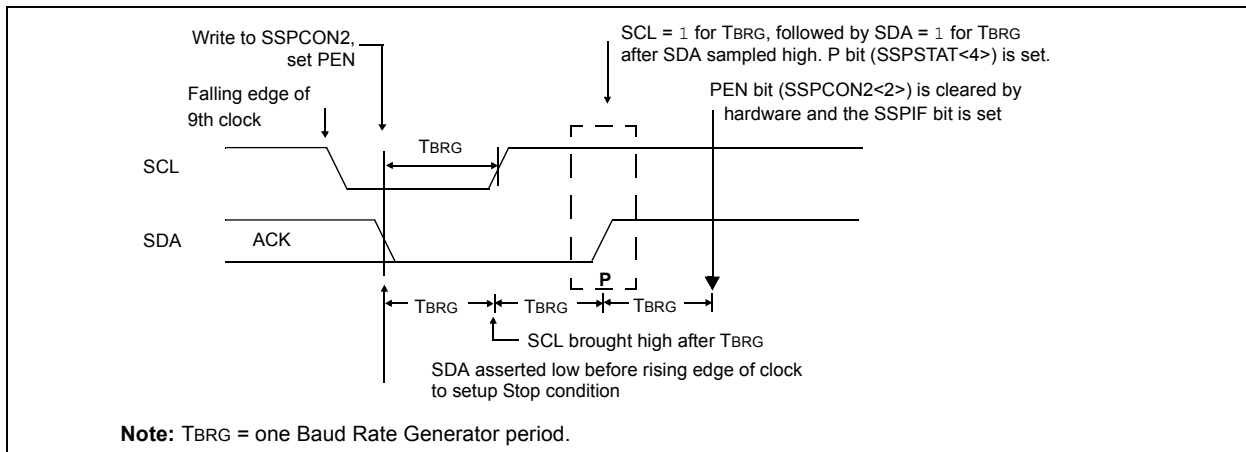
### 15.4.13.1 WCOL Status Flag

If the user writes the SSPBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

**FIGURE 15-23: ACKNOWLEDGE SEQUENCE WAVEFORM**



**FIGURE 15-24: STOP CONDITION RECEIVE OR TRANSMIT MODE**





## 15.4.14 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).

## 15.4.15 EFFECT OF A RESET

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

## 15.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I<sup>2</sup>C bus may be taken when the P bit (SSPSTAT<4>) is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the MSSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed in hardware with the result placed in the BCLIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

## 15.4.17 MULTI-MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA, by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the I<sup>2</sup>C port to its Idle state (Figure 15-25).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I<sup>2</sup>C bus is free, the user can resume communication by asserting a Start condition.

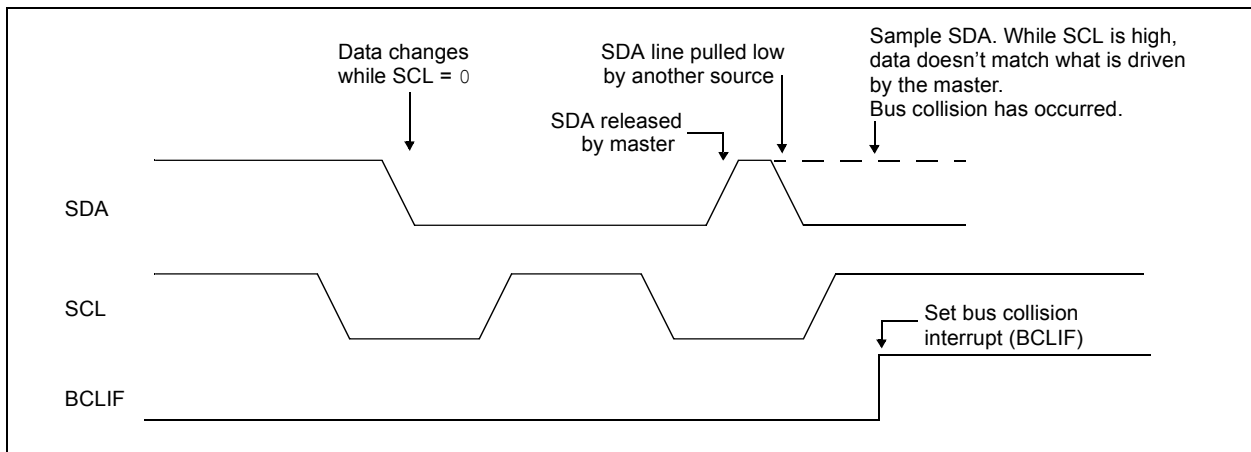
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the 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 SDA and SCL pins. If a Stop condition occurs, the SSPIF bit will be set.

A write to the SSPBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I<sup>2</sup>C bus can be taken when the P bit is set in the SSPSTAT register, or the bus is Idle and the S and P bits are cleared.

**FIGURE 15-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**



# PIC18F6390/6490/8390/8490

## 15.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

- SDA or SCL are sampled low at the beginning of the Start condition (Figure 15-26).
- SCL is sampled low before SDA is asserted low (Figure 15-27).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low, or the SCL pin is already low, then all of the following occur:

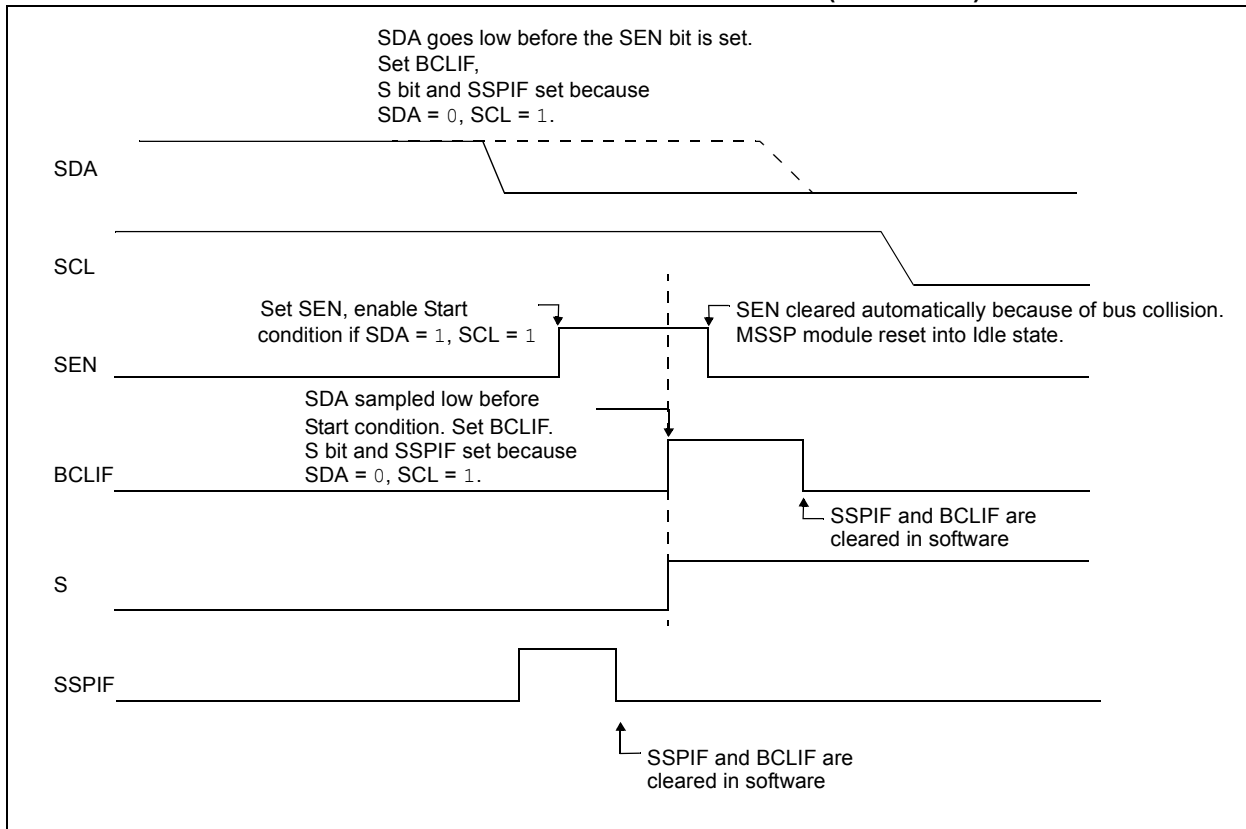
- the Start condition is aborted,
- the BCLIF flag is set and
- the MSSP module is reset to its Idle state (Figure 15-26).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded from SSPADD<6:0> and counts down to '0'. If the SCL pin is sampled low while SDA is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

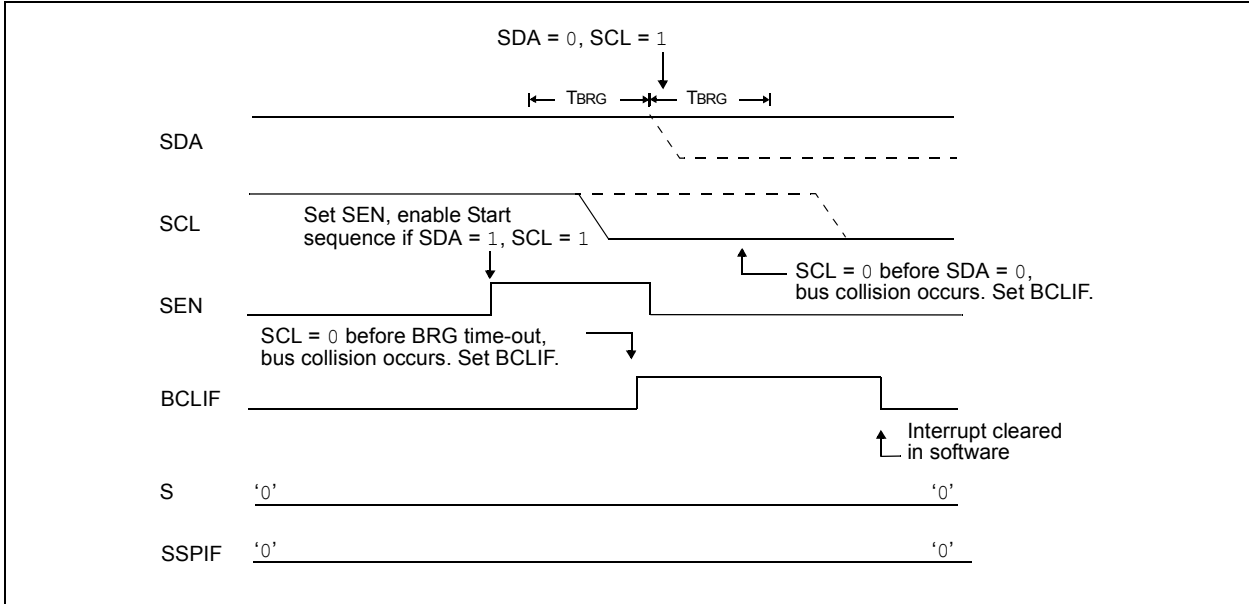
If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 15-28). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to 0 and during this time, if the SCL pins are sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

**Note:** The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

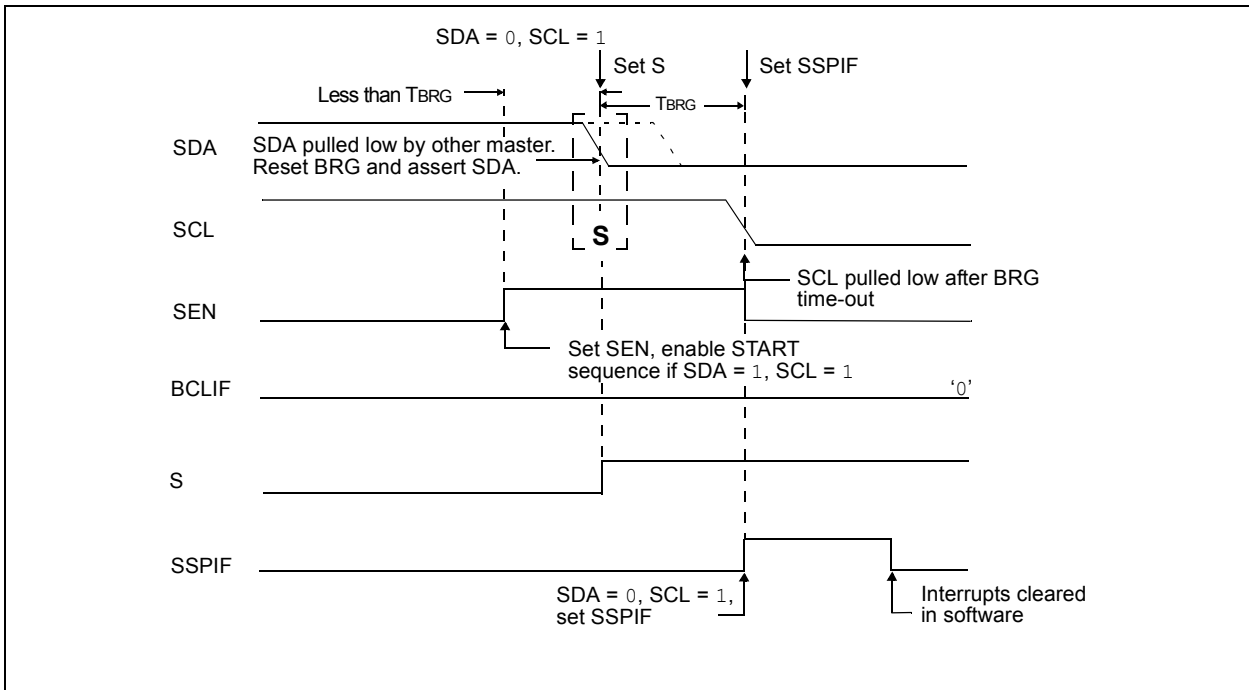
**FIGURE 15-26: BUS COLLISION DURING START CONDITION (SDA ONLY)**



**FIGURE 15-27: BUS COLLISION DURING START CONDITION (SCL = 0)**



**FIGURE 15-28: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION**



# PIC18F6390/6490/8390/8490

## 15.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

- a) A low level is sampled on SDA when SCL goes from low level to high level.
- b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<6:0> and counts down to 0. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 15-29). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

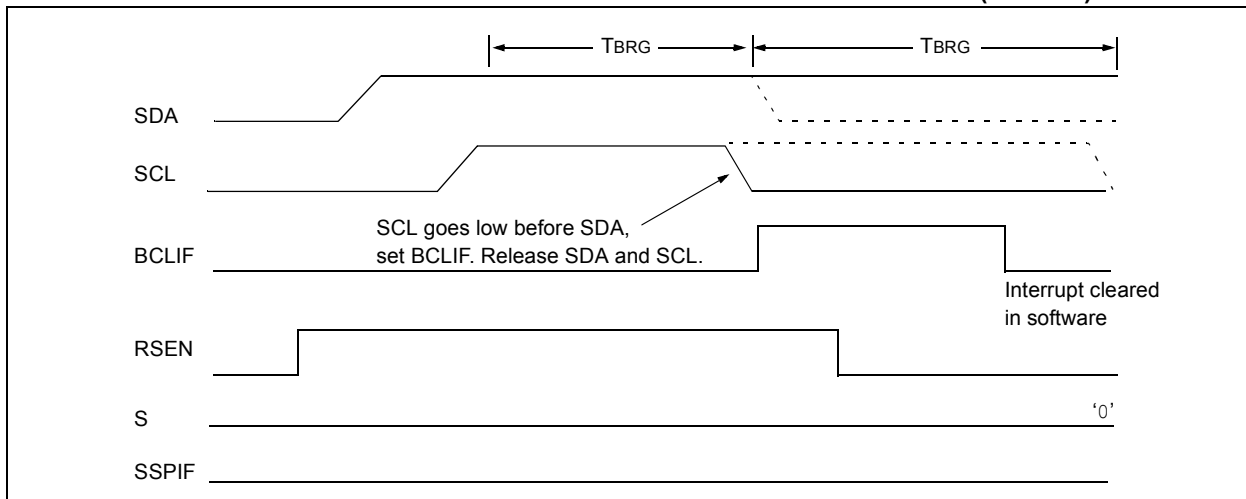
If SCL goes from high-to-low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 15-30).

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

**FIGURE 15-29: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)**



**FIGURE 15-30: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)**



# PIC18F6390/6490/8390/8490

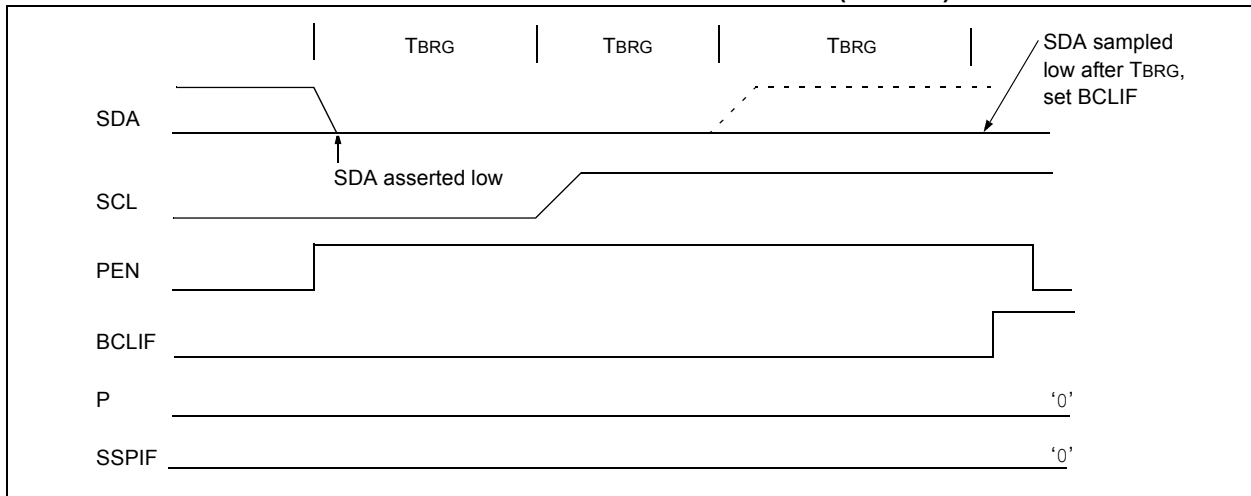
## 15.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

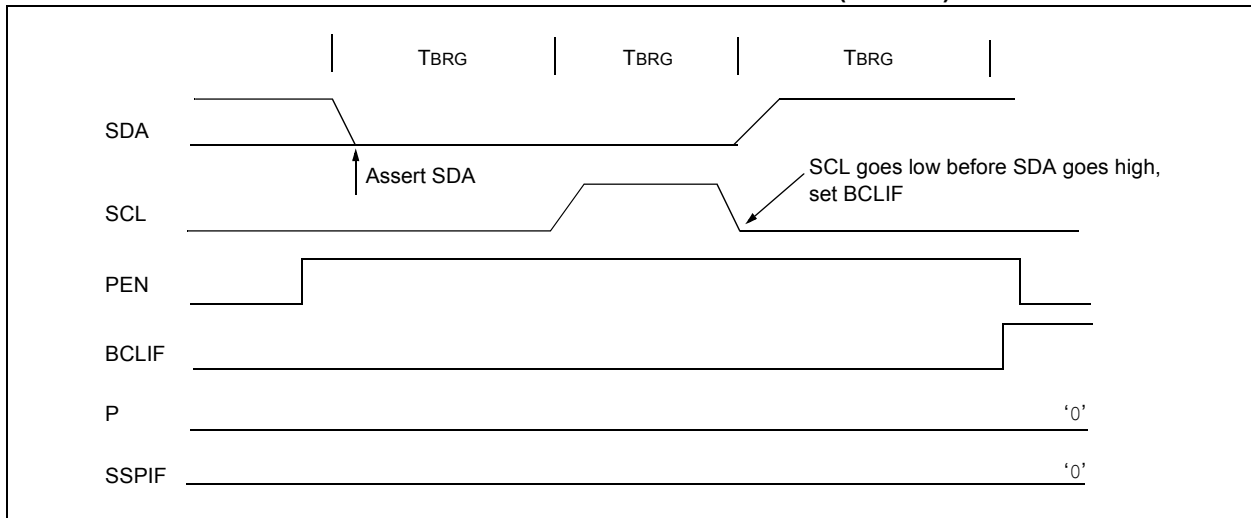
- After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out.
- After the SCL pin is deasserted, SCL is sampled low before SDA goes high.

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD<6:0> and counts down to 0. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 15-31). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 15-32).

**FIGURE 15-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)**



**FIGURE 15-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)**



# PIC18F6390/6490/8390/8490

**TABLE 15-4: 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	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC Data Direction Register								62
SSPBUF	MSSP Receive Buffer/Transmit Register								60
SSPADD	MSSP Address Register in I <sup>2</sup> C Slave Mode. MSSP Baud Rate Reload Register in I <sup>2</sup> C Slave Mode.								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	60
SSPSTAT	SMP	CKE	D $\bar{A}$	P	S	R $\bar{W}$	UA	BF	60

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

## 16.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

PIC18F6390/6490/8390/8490 devices have three serial I/O modules: the MSSP module, discussed in the previous chapter and two Universal Synchronous Asynchronous Receiver Transmitter (USART) modules. (Generically, the USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

There are two distinct implementations of the USART module in these devices: the Enhanced USART (EUSART) discussed here and the Addressable USART discussed in the next chapter. For this device family, USART1 always refers to the EUSART, while USART2 is always the AUSART.

The EUSART and AUSART modules implement the same core features for serial communications; their basic operation is essentially the same. The EUSART module provides additional features, including Automatic Baud Rate Detection (ABD) and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These features make it ideally suited for use in Local Interconnect Network bus (LIN bus) systems.

The EUSART can be configured in the following modes:

- Asynchronous (full-duplex) with:
  - Auto-wake-up on character reception
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half-duplex) with selectable clock polarity
- Synchronous – Slave (half-duplex) with selectable clock polarity

The pins of the EUSART are multiplexed with the functions of PORTC (RC6/TX1/CK1 and RC7/RX1/DT1). In order to configure these pins as an EUSART:

- bit SPEN (RCSTA1<7>) must be set (= 1)
- bit TRISC<7> must be set (= 1)
- bit TRISC<6> must be set (= 1)

**Note:** The USART control will automatically reconfigure the pin from input to output as needed.

The operation of the Enhanced USART module is controlled through three registers:

- Transmit Status and Control Register 1 (TXSTA1)
- Receive Status and Control Register 1 (RCSTA1)
- Baud Rate Control Register 1 (BAUDCON1)

The registers are described in Register 16-1, Register 16-2 and Register 16-3.

# PIC18F6390/6490/8390/8490

## REGISTER 16-1: TXSTA1: EUSART TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN <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:** 9th bit of Transmit Data  
 Can be address/data bit or a parity bit.

**Note 1:** SREN/CREN overrides TXEN in Sync mode.



# PIC18F6390/6490/8390/8490

**REGISTER 16-2: RCSTA1: EUSART RECEIVE STATUS AND CONTROL REGISTER**

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

**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 RX1/DT1 and TX1/CK1 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 RCREG1 register and receiving next valid byte)  
           0 = No framing error
- bit 1      **OERR:** Overrun Error bit  
           1 = Overrun error (can be cleared by clearing bit, CREN)  
           0 = No overrun error
- bit 0      **RX9D:** 9th bit of Received Data  
           This can be address/data bit or a parity bit and must be calculated by user firmware.

# PIC18F6390/6490/8390/8490

## REGISTER 16-3: BAUDCON1: BAUD RATE CONTROL REGISTER 1

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

### 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      **ABDOVF:** Auto-Baud Acquisition Rollover Status bit  
           1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)  
           0 = No BRG rollover has occurred
- bit 6      **RCIDL:** Receive Operation Idle Status bit  
           1 = Receive operation is Idle  
           0 = Receive operation is active
- bit 5      **Unimplemented:** Read as '0'
- bit 4      **SCKP:** Synchronous Clock Polarity Select bit  
           Asynchronous mode:  
           Unused in this mode.  
           Synchronous mode:  
           1 = Idle state for clock (CK1) is a high level  
           0 = Idle state for clock (CK1) is a low level
- bit 3      **BRG16:** 16-Bit Baud Rate Register Enable bit  
           1 = 16-bit Baud Rate Generator – SPBRGH1 and SPBRG1  
           0 = 8-bit Baud Rate Generator – SPBRG1 only (Compatible mode), SPBRGH1 value ignored
- bit 2      **Unimplemented:** Read as '0'
- bit 1      **WUE:** Wake-up Enable bit  
           Asynchronous mode:  
           1 = EUSART will continue to sample the RX1 pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge  
           0 = RX1 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.

# PIC18F6390/6490/8390/8490

## 16.1 EUSART 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 (BAUDCON1<3>) selects 16-bit mode.

The SPBRGH1:SPBRG1 register pair controls the period of a free-running timer. In Asynchronous mode, the BRGH (TXSTA1<2>) and BRG16 (BAUDCON1<3>) bits also control the baud rate. In Synchronous mode, BRGH is ignored. Table 16-1 shows the formula for computation of the baud rate for different EUSART modes that only apply in Master mode (internally generated clock).

Given the desired baud rate and Fosc, the nearest integer value for the SPBRGH1:SPBRG1 registers can be calculated using the formulas in Table 16-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 16-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 16-3. It may be advanta-

geous 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 SPBRGH1:SPBRG1 registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

### 16.1.1 OPERATION IN POWER-MANAGED MODES

The device clock is used to generate the desired baud rate. When one of the power-managed modes is entered, the new clock source may be operating at a different frequency. This may require an adjustment to the value in the SPBRG1 register pair.

### 16.1.2 SAMPLING

The data on the RX1 pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX1 pin.

**TABLE 16-1: BAUD RATE FORMULAS**

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-Bit/Asynchronous	$F_{osc}/[64(n+1)]$
0	0	1	8-Bit/Asynchronous	$F_{osc}/[16(n+1)]$
0	1	0	16-Bit/Asynchronous	
0	1	1	16-Bit/Asynchronous	$F_{osc}/[4(n+1)]$
1	0	x	8-Bit/Synchronous	
1	1	x	16-Bit/Synchronous	

**Legend:** x = Don't care, n = Value of SPBRGH1:SPBRG1 register pair

### EXAMPLE 16-1: CALCULATING BAUD RATE ERROR

For a device with Fosc of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:	
Desired Baud Rate	= $F_{osc}/(64([SPBRGH1:SPBRG1] + 1))$
Solving for SPBRGH1:SPBRG1:	
X	= $((F_{osc}/\text{Desired Baud Rate})/64) - 1$
	= $((16000000/9600)/64) - 1$
	= $[25.042] = 25$
Calculated Baud Rate	= $16000000/(64(25 + 1))$
	= 9615
Error	= $(\text{Calculated Baud Rate} - \text{Desired Baud Rate})/\text{Desired Baud Rate}$
	= $(9615 - 9600)/9600 = 0.16\%$

**TABLE 16-2: REGISTERS ASSOCIATED WITH THE BAUD RATE GENERATOR**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

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

# PIC18F6390/6490/8390/8490

**TABLE 16-3: BAUD RATES FOR ASYNCHRONOUS MODES**

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	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	—	—	—	—	—	—

# PIC18F6390/6490/8390/8490

**TABLE 16-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)**

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	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	2403	-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	—	—	—	—	—	—

# PIC18F6390/6490/8390/8490

## 16.1.3 AUTO-BAUD RATE DETECT

The Enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 16-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RX1 signal, the RX1 signal is timing the BRG. In ABD mode, the internal Baud Rate Generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a Start bit. The Auto-Baud Rate Detect must receive a byte with the value, 55h (ASCII “U”, which is also the LIN bus Sync character), in order to calculate the proper bit rate. The measurement is taken over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRG1 begins counting up, using the preselected clock source on the first rising edge of RX1. After eight bits on the RX1 pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGH1:SPBRG1 register pair. Once the 5th edge is seen (this should correspond to the Stop bit), the ABDEN bit is automatically cleared.

If a rollover of the BRG occurs (an overflow from FFFFh to 0000h), the event is trapped by the ABDOVF status bit (BAUDCON1<7>). It is set in hardware by BRG rollovers and can be set or cleared by the user in software. ABD mode remains active after rollover events and the ABDEN bit remains set (Figure 16-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. Note that the BRG clock will be configured by the BRG16 and BRGH bits. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGH1 register. Refer to Table 16-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSART state machine is held in Idle. The RC1IF interrupt is set once the fifth rising edge on RX1 is detected. The value in the RCREG1 needs to be read to clear the RC1IF interrupt. The contents of RCREG1 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.

**3:** When the auto-baud feature is enabled, the BRG16 bit (BAUDCON<3>) must be set.

**TABLE 16-4: BRG COUNTER CLOCK RATES**

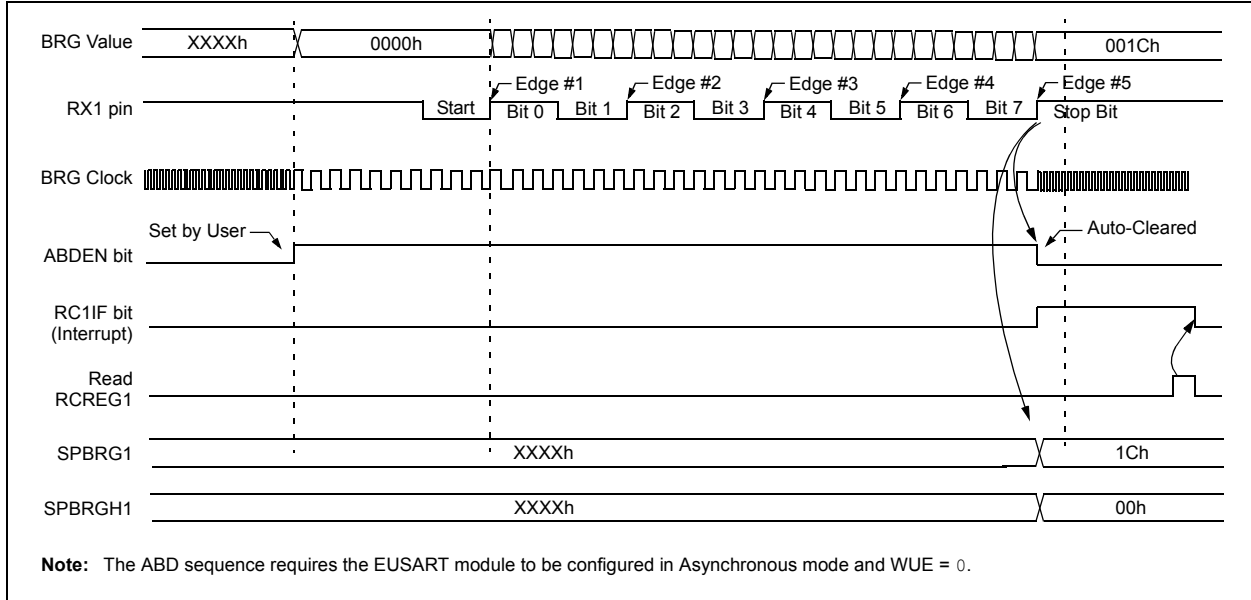
BRG16	BRGH	BRG Counter Clock
0	0	Fosc/512
0	1	Fosc/128
1	0	Fosc/128
1	1	Fosc/32

### 16.1.3.1 ABD and EUSART Transmission

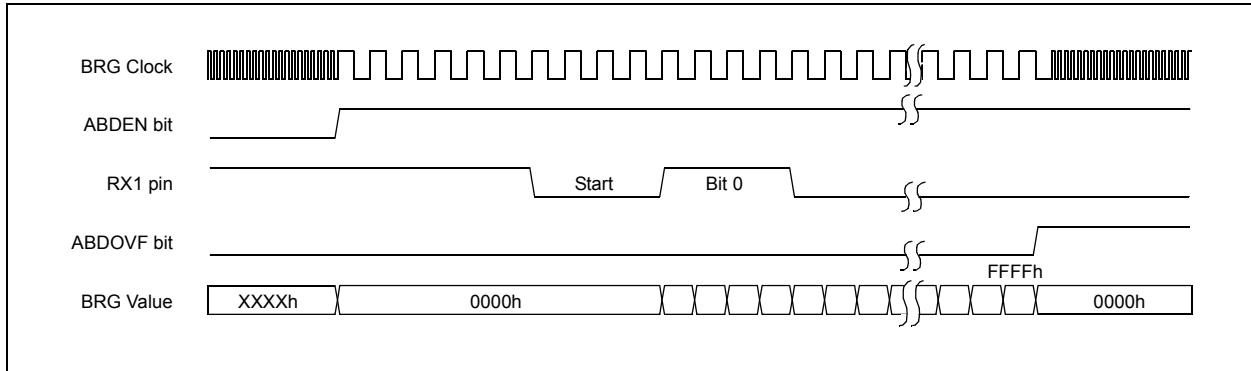
Since the BRG clock is reversed during ABD acquisition, the EUSART transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREG1 cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSART operation.

# PIC18F6390/6490/8390/8490

**FIGURE 16-1: AUTOMATIC BAUD RATE CALCULATION**



**FIGURE 16-2: BRG OVERFLOW SEQUENCE**



# PIC18F6390/6490/8390/8490

## 16.2 EUSART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTA1<4>). In this mode, the EUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated 8-bit/16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent, but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on the BRGH and BRG16 bits (TXSTA1<2> and BAUDCON1<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the EUSART module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto-Wake-up on Sync Break Character
- 12-Bit Break Character Transmit
- Auto-Baud Rate Detection

### 16.2.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 16-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG1. The TXREG1 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 TXREG1 register (if available).

Once the TXREG1 register transfers the data to the TSR register (occurs in one Tcy), the TXREG1 register is empty and the TX1IF flag bit (PIR1<4>) is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX1IE (PIE1<4>). TX1IF will be set regardless of the state of TX1IE; it cannot be cleared in software. TX1IF is also not cleared immediately upon loading TXREG1, but becomes valid in the second instruction cycle following the load instruction. Polling TX1IF immediately following a load of TXREG1 will return invalid results.

While TX1IF indicates the status of the TXREG1 register, another bit, TRMT (TXSTA1<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.

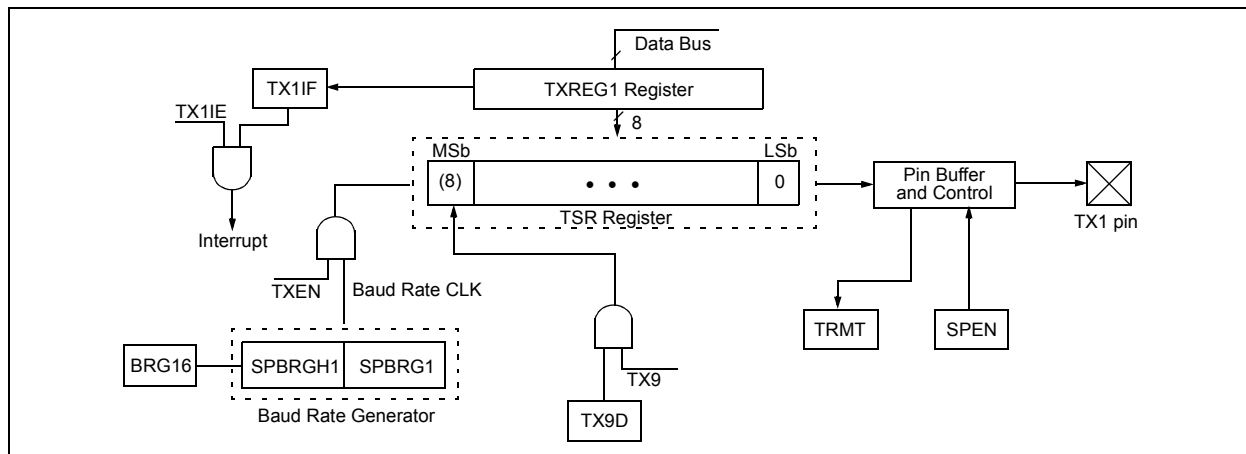
**Note 1:** The TSR register is not mapped in data memory so it is not available to the user.

**2:** Flag bit, TX1IF, is set when enable bit TXEN is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGH1:SPBRG1 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, TX1IE.
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, TX1IF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREG1 register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

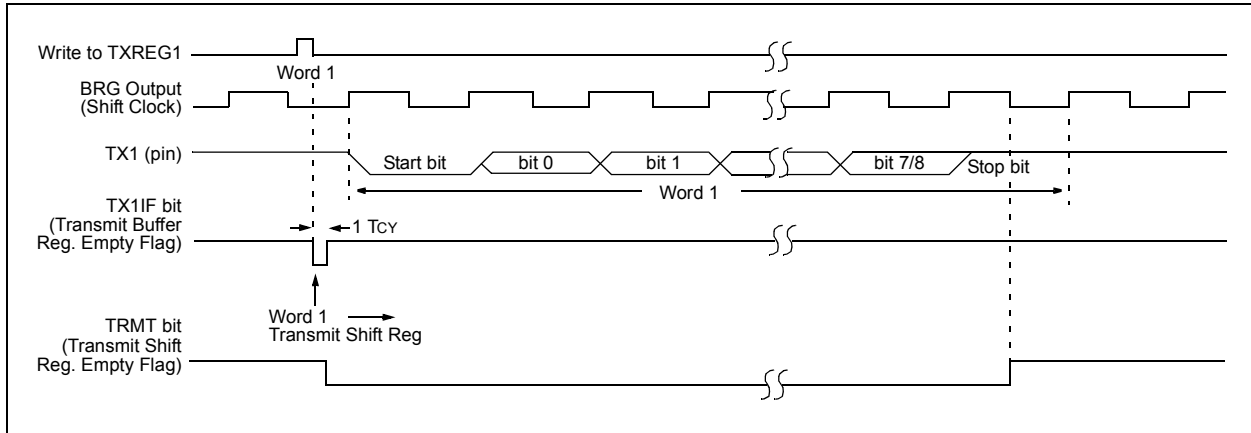
FIGURE 16-3: EUSART TRANSMIT BLOCK DIAGRAM



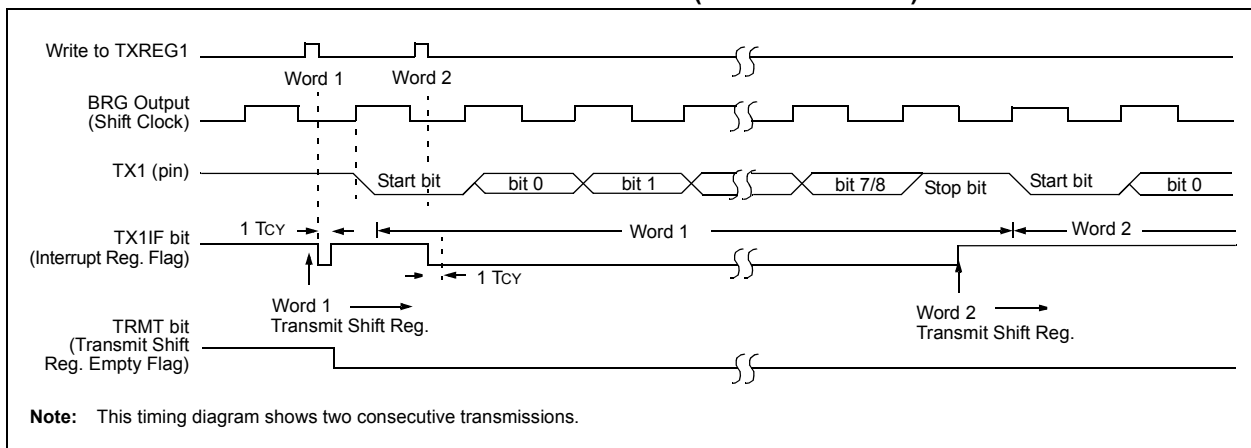


# PIC18F6390/6490/8390/8490

**FIGURE 16-4: ASYNCHRONOUS TRANSMISSION**



**FIGURE 16-5: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)**



**TABLE 16-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 Transmit Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

# PIC18F6390/6490/8390/8490

## 16.2.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 16-6. The data is received on the RX1 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:

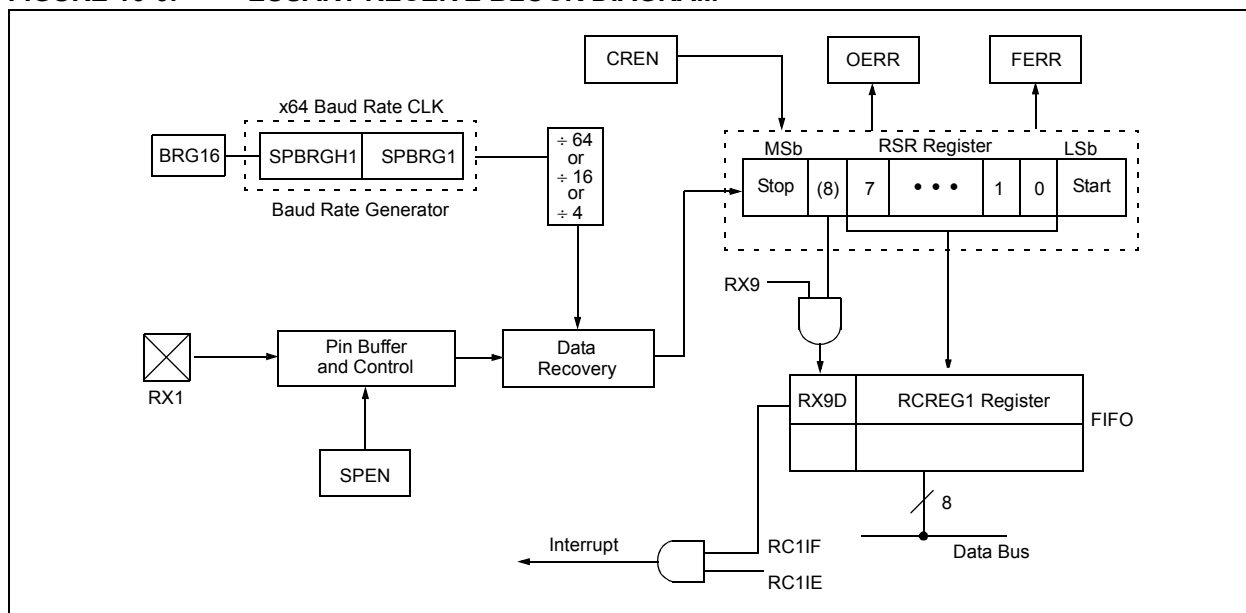
1. Initialize the SPBRGH1:SPBRG1 registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, RC1IE.
4. If 9-bit reception is desired, set bit, RX9.
5. Enable the reception by setting bit, CREN.
6. Flag bit, RC1IF, will be set when reception is complete and an interrupt will be generated if enable bit, RC1IE, was set.
7. Read the RCSTA1 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG1 register.
9. If any error occurred, clear the error by clearing enable bit, CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

## 16.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

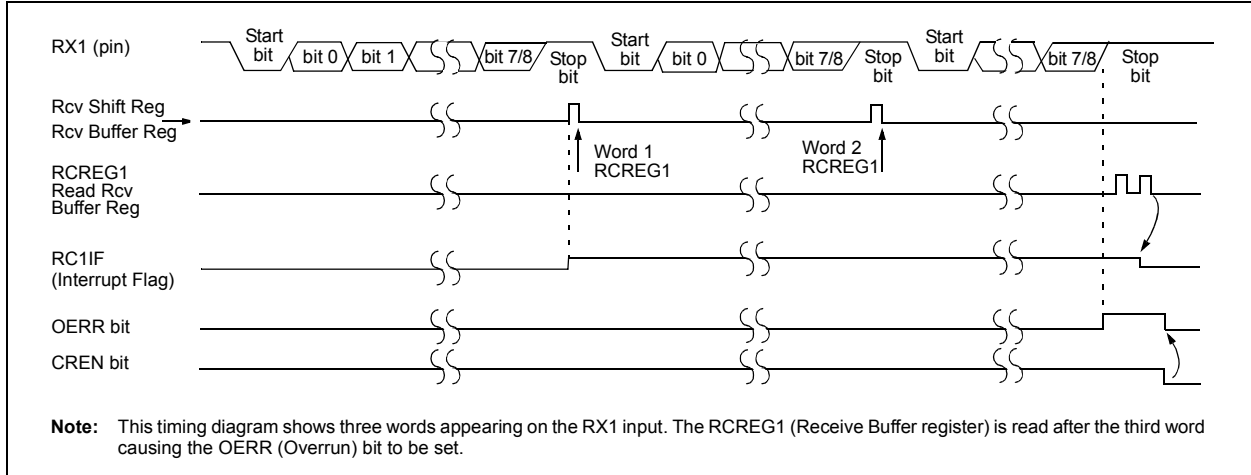
1. Initialize the SPBRGH1:SPBRG1 registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RC1IP 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 RC1IF bit will be set when reception is complete. The interrupt will be Acknowledged if the RC1IE and GIE bits are set.
8. Read the RCSTA1 register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREG1 to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

**FIGURE 16-6: EUSART RECEIVE BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

**FIGURE 16-7: ASYNCHRONOUS RECEPTION**



**TABLE 16-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 Receive Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

# PIC18F6390/6490/8390/8490

## 16.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper byte reception cannot be performed. The auto-wake-up feature allows the controller to wake-up, due to activity on the RX1/DT1 line, while the EUSART is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCON<1>). Once set, the typical receive sequence on RX1/DT1 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 RX1/DT1 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 RC1IF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 16-8) and asynchronously, if the device is in Sleep mode (Figure 16-9). The interrupt condition is cleared by reading the RCREG1 register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RX1 line following the wake-up event. At this point, the EUSART module is in Idle mode and returns to normal operation. This signals to the user that the Sync Break event is over.

### 16.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RX1/DT1, information with any state changes before the Stop bit may signal a false

End-Of-Character (EOC) and cause data or framing errors. Therefore, to work properly, 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., XT or HS mode). The Sync Break (or Wake-up Signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

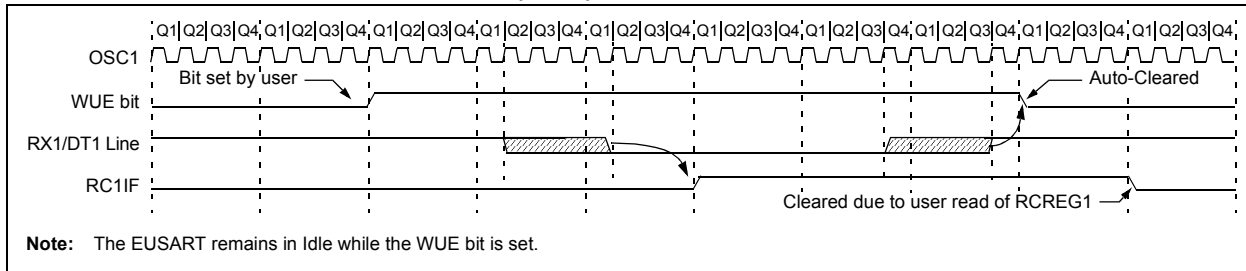
### 16.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RC1IF 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 RC1IF bit. The WUE bit is cleared after this when a rising edge is seen on RX1/DT1. The interrupt condition is then cleared by reading the RCREG1 register. Ordinarily, the data in RCREG1 will be dummy data and should be discarded.

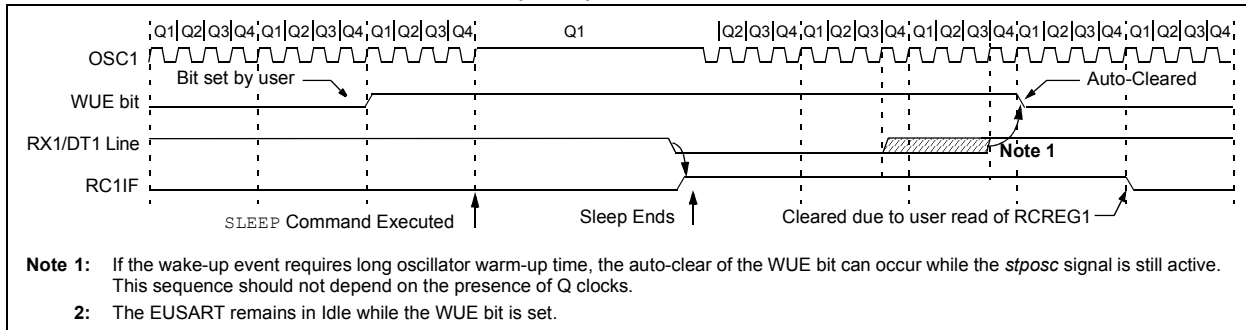
The fact that the WUE bit has been cleared (or is still set), and the RC1IF flag is set, should not be used as an indicator of the integrity of the data in RCREG1. Users should consider implementing a parallel method in firmware to verify received data integrity.

To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

**FIGURE 16-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION**



**FIGURE 16-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP**



## 16.2.5 BREAK CHARACTER SEQUENCE

The Enhanced USART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The Frame Break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift Register is loaded with data. Note that the value of data written to TXREG1 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 TXREG1 for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 16-10 for the timing of the Break character sequence.

### 16.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.

3. Load the TXREG1 with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG1 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 TXREG1 becomes empty, as indicated by the TX1IF, the next data byte can be written to TXREG1.

## 16.2.6 RECEIVING A BREAK CHARACTER

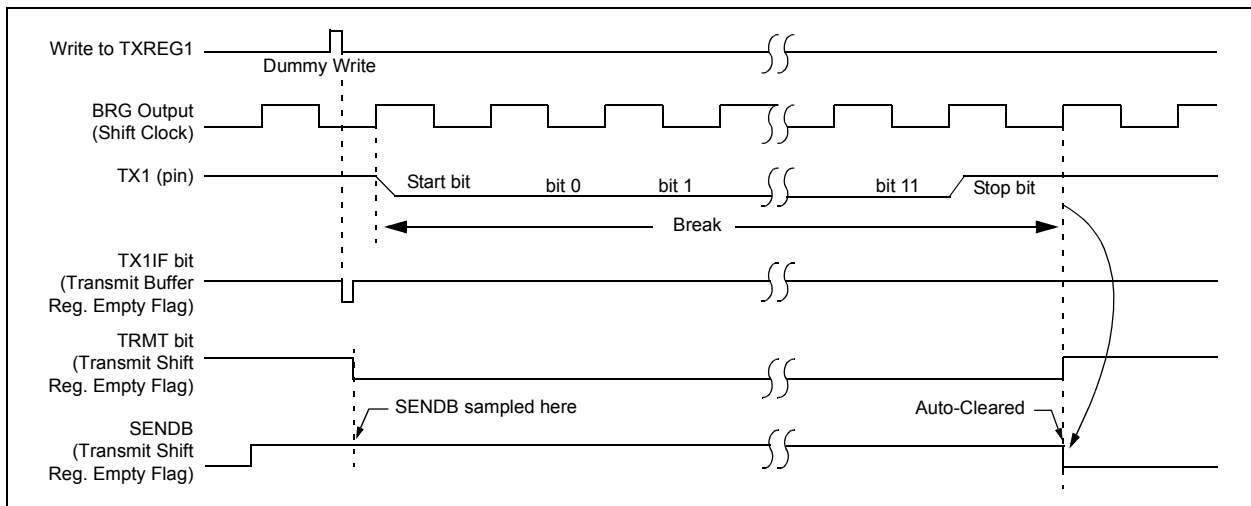
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 16.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RX1/DT1, cause an RC1IF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABD bit once the TX1IF interrupt is observed.

**FIGURE 16-10: SEND BREAK CHARACTER SEQUENCE**



# PIC18F6390/6490/8390/8490

## 16.3 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit, SYNC (TXSTA<4>). In addition, enable bit, SPEN (RCSTA1<7>), is set in order to configure the TX1 and RX1 pins to CK1 (clock) and DT1 (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK1 line. Clock polarity is selected with the SCKP bit (BAUDCON<4>); setting SCKP sets the Idle state on CK1 as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

### 16.3.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 16-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG1. The TXREG1 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 TXREG1 (if available).

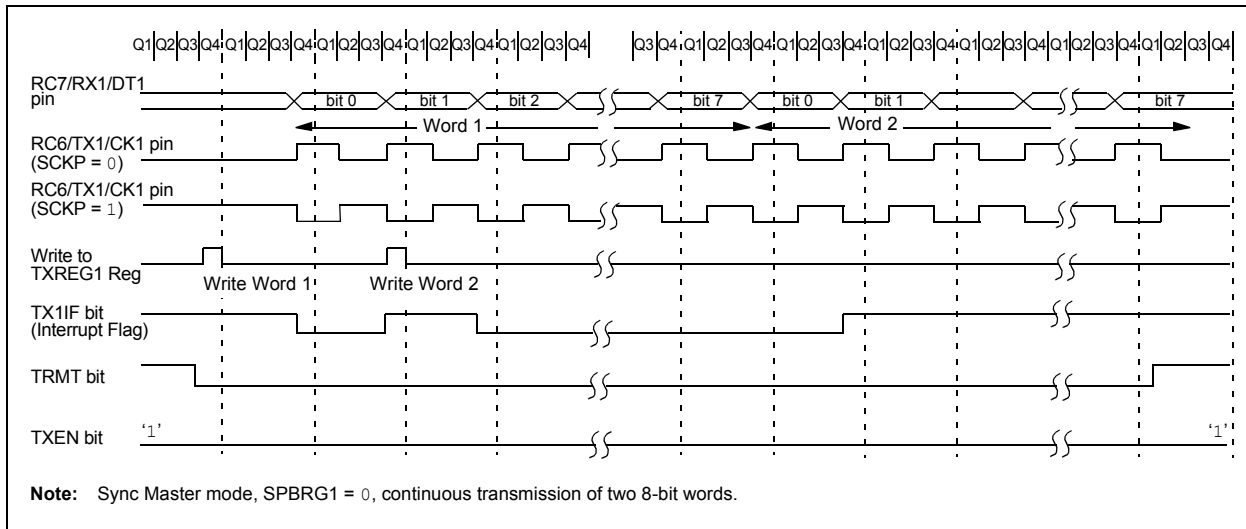
Once the TXREG1 register transfers the data to the TSR register (occurs in one T<sub>CYCLE</sub>), the TXREG1 is empty and the TX1IF flag bit (PIR1<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX1IE (PIE1<4>). TX1IF is set regardless of the state of enable bit, TX1IE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREG1 register.

While flag bit, TX1IF, indicates the status of the TXREG1 register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

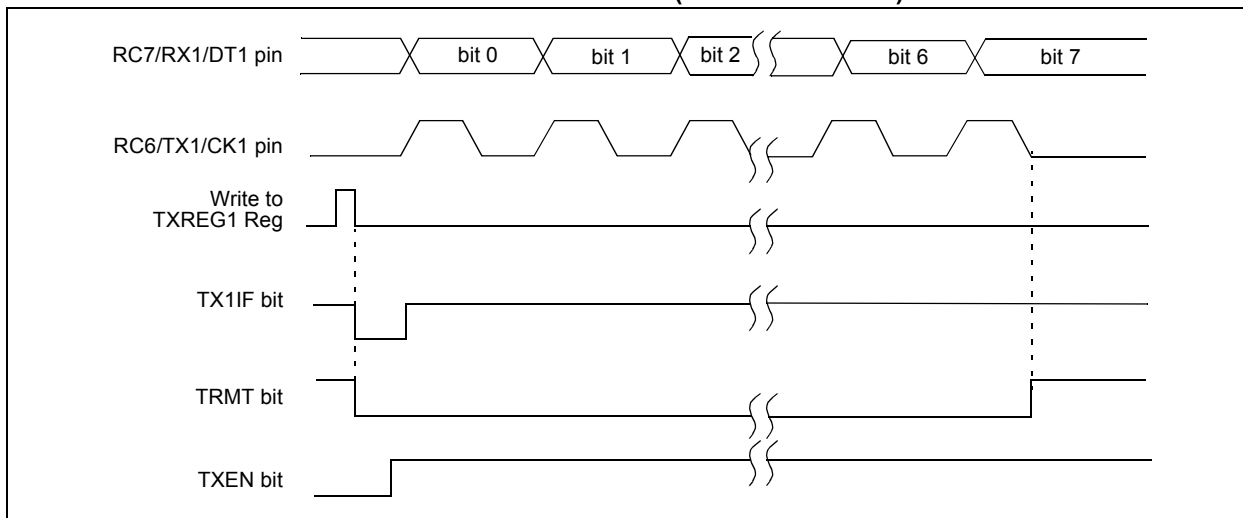
1. Initialize the SPBRGH1:SPBRG1 registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TX1IE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREG1 register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 16-11: SYNCHRONOUS TRANSMISSION**



# PIC18F6390/6490/8390/8490

**FIGURE 16-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



**TABLE 16-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 Transmit Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

# PIC18F6390/6490/8390/8490

## 16.3.2 EUSART SYNCHRONOUS MASTER RECEPTION

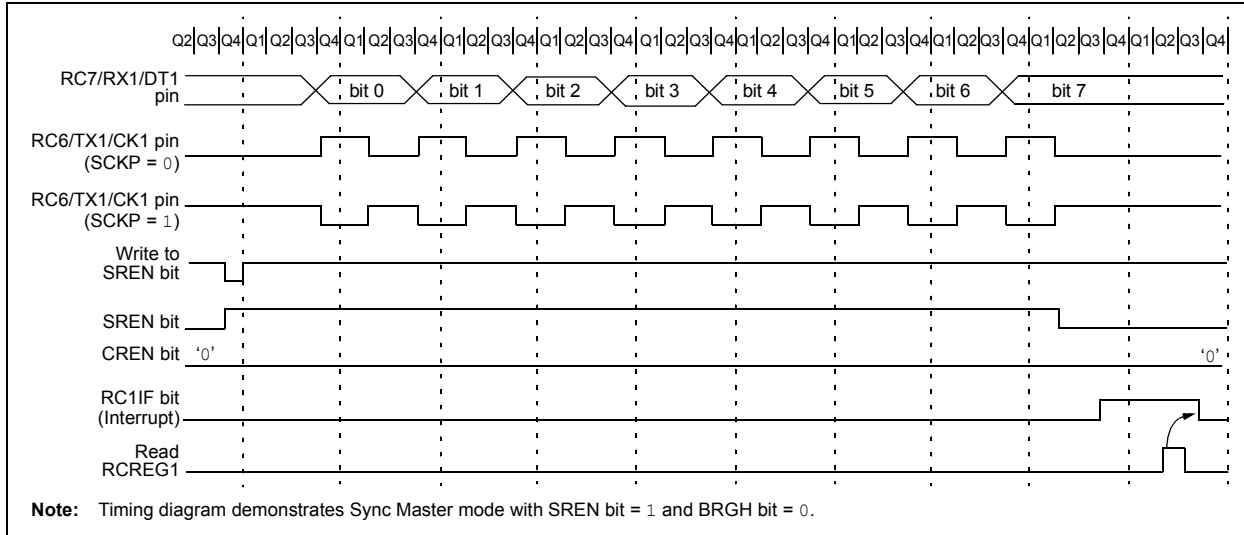
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA1<5>), or the Continuous Receive Enable bit, CREN (RCSTA1<4>). Data is sampled on the RX1 pin on the falling edge of the clock.

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 SPBRGH1:SPBRG1 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, RC1IE.
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, RC1IF, will be set when reception is complete and an interrupt will be generated if the enable bit, RC1IE, was set.
8. Read the RCSTA1 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG1 register.
10. If any error occurred, clear the error by clearing bit, CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 16-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**



**TABLE 16-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 Receive Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.



# PIC18F6390/6490/8390/8490

## 16.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA1<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CK1 pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

### 16.4.1 EUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical except in the case of the Sleep mode.

If two words are written to the TXREG1, and then the SLEEP instruction is executed, the following will occur:

- The first word will immediately transfer to the TSR register and transmit.
- The second word will remain in the TXREG1 register.
- Flag bit, TX1IF, will not be set.
- When the first word has been shifted out of TSR, the TXREG1 register will transfer the second word to the TSR and flag bit, TX1IF, will now be set.
- If enable bit, TX1IE, 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:

- Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
- Clear bits, CREN and SREN.
- If interrupts are desired, set enable bit, TX1IE.
- If 9-bit transmission is desired, set bit, TX9.
- Enable the transmission by setting enable bit, TXEN.
- If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
- Start transmission by loading data to the TXREG1 register.
- If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**TABLE 16-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 Transmit Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SEnDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

# PIC18F6390/6490/8390/8490

## 16.4.2 EUSART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical except in the case of Sleep or any Idle mode and bit, SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this low-power mode. Once the word is received, the RSR register will transfer the data to the RCREG1 register. If the RC1IE 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, RC1IE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RC1IF, will be set when reception is complete. An interrupt will be generated if enable bit, RC1IE, was set.
6. Read the RCSTA1 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG1 register.
8. If any error occurred, clear the error by clearing bit, CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**TABLE 16-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 Receive Register								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								62
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								61

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave reception.

## 17.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (AUSART)

The Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART) module is very similar in function to the Enhanced USART module, discussed in the previous chapter. It is provided as an additional channel for serial communication, with external devices, for those situations that do not require Auto-Baud Detection or LIN bus support.

The AUSART can be configured in the following modes:

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

The pins of the AUSART module are multiplexed with the functions of PORTG (RG1/TX2/CK2/SEG29 and RG2/RX2/DT2/SEG28, respectively). In order to configure these pins as an AUSART:

- SPEN bit (RCSTA2<7>) must be set (= 1)
- TRISG<2> bit must be set (= 1)
- TRISG<1> bit must be cleared (= 0) for Asynchronous and Synchronous Master modes
- TRISG<1> bit must be set (= 1) for Synchronous Slave mode

<b>Note:</b> The AUSART control will automatically reconfigure the pin from input to output as needed.
--

The operation of the Addressable USART module is controlled through two registers, TXSTA2 and RXSTA2. These are detailed in Register 17-1 and Register 17-2 respectively.

# PIC18F6390/6490/8390/8490

## REGISTER 17-1: TXSTA2: AUSART TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN <sup>(1)</sup>	SYNC	—	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:** AUSART Mode Select bit  
 1 = Synchronous mode  
 0 = Asynchronous mode
- bit 3            **Unimplemented:** Read as '0'
- bit 2            **BRGH:** High Baud Rate Select bit  
Asynchronous mode:  
 1 = High speed  
 0 = Low speed  
Synchronous mode:  
 Unused in this mode.
- bit 1            **TRMT:** Transmit Shift Register Status bit  
 1 = TSR empty  
 0 = TSR full
- bit 0            **TX9D:** 9th bit of Transmit Data  
 Can be address/data bit or a parity bit.

**Note 1:** SREN/CREN overrides TXEN in Sync mode.

# PIC18F6390/6490/8390/8490

## REGISTER 17-2: RCSTA2: AUSART RECEIVE STATUS AND CONTROL REGISTER

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

### 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 RX2/DT2 and TX2/CK2 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 RCREG2 register and receiving next valid byte)  
 0 = No framing error
- bit 1      **OERR:** Overrun Error bit  
 1 = Overrun error (can be cleared by clearing bit, CREN)  
 0 = No overrun error
- bit 0      **RX9D:** 9th bit of Received Data  
 This can be address/data bit or a parity bit and must be calculated by user firmware.

# PIC18F6390/6490/8390/8490

## 17.1 AUSART Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit generator that supports both the Asynchronous and Synchronous modes of the AUSART.

The SPBRG2 register controls the period of a free-running timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, BRGH is ignored. Table 17-1 shows the formula for computation of the baud rate for different AUSART modes, which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRG2 register can be calculated using the formulas in Table 17-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 17-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 17-3. It may be advanta-

geous to use the high baud rate (BRGH = 1) to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRG2 register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

### 17.1.1 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 SPBRG2 register.

### 17.1.2 SAMPLING

The data on the RX2 pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX2 pin.

**TABLE 17-1: BAUD RATE FORMULAS**

Configuration Bits		BRG/AUSART Mode	Baud Rate Formula
SYNC	BRGH		
0	0	Asynchronous	$F_{osc}/[64 (n + 1)]$
0	1	Asynchronous	$F_{osc}/[16 (n + 1)]$
1	x	Synchronous	$F_{osc}/[4 (n + 1)]$

**Legend:** x = Don't care, n = Value of SPBRG2 register

### EXAMPLE 17-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, BRGH = 0:	
Desired Baud Rate	= $F_{osc}/(64 ([SPBRG2] + 1))$
Solving for SPBRG2:	
X	= $((F_{osc}/\text{Desired Baud Rate})/64) - 1$
	= $((16000000/9600)/64) - 1$
	= $[25.042] = 25$
Calculated Baud Rate	= $16000000/(64 (25 + 1))$
	= 9615
Error	= $(\text{Calculated Baud Rate} - \text{Desired Baud Rate})/\text{Desired Baud Rate}$
	= $(9615 - 9600)/9600 = 0.16\%$

**TABLE 17-2: REGISTERS ASSOCIATED WITH THE BAUD RATE GENERATOR**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** Shaded cells are not used by the BRG.

# PIC18F6390/6490/8390/8490

**TABLE 17-3: BAUD RATES FOR ASYNCHRONOUS MODES**

BAUD RATE (K)	BRGH = 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)	BRGH = 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)	BRGH = 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	—	—	—	—	—	—	—	—	—	—	—	—
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)	BRGH = 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	-0.16	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	—	—	—	—	—	—

# PIC18F6390/6490/8390/8490

## 17.2 AUSART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTA2<4>). In this mode, the AUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The AUSART transmits and receives the LSb first. The AUSART's transmitter and receiver are functionally independent but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on the BRGH bit (TXSTA2<2>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the AUSART module consists of the following important elements:

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

### 17.2.1 AUSART ASYNCHRONOUS TRANSMITTER

The AUSART transmitter block diagram is shown in Figure 17-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG2. The TXREG2 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 TXREG2 register (if available).

Once the TXREG2 register transfers the data to the TSR register (occurs in one Tcy), the TXREG2 register is empty and the TX2IF flag bit (PIR3<4>) is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX2IE (PIE3<4>). TX2IF will be set regardless of the state of TX2IE; it cannot be cleared in software. TX2IF is also not cleared immediately upon loading TXREG2, but becomes valid in the second instruction cycle following the load instruction. Polling TX2IF immediately following a load of TXREG2 will return invalid results.

While TX2IF indicates the status of the TXREG2 register, another bit, TRMT (TXSTA2<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.

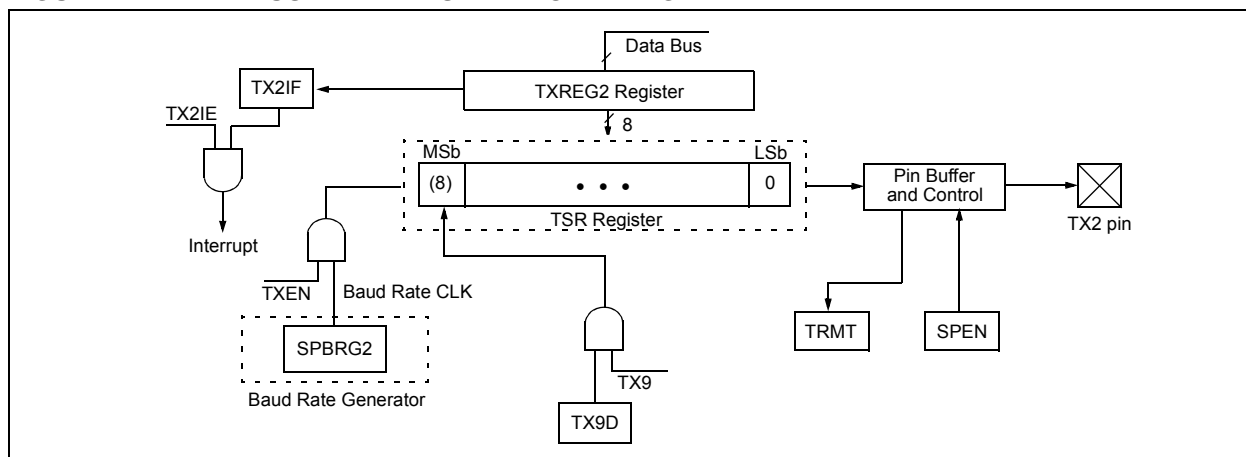
**Note 1:** The TSR register is not mapped in data memory, so it is not available to the user.

**2:** Flag bit, TX2IF, is set when enable bit, TXEN is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRG2 register for the appropriate baud rate. Set or clear the BRGH bit, 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, TX2IE.
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, TX2IF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREG2 register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

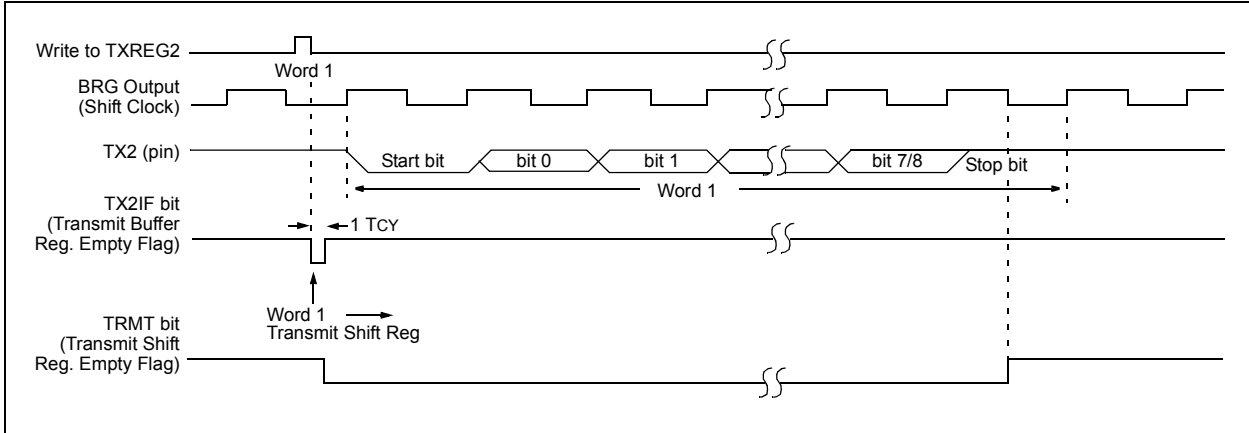
**FIGURE 17-1: AUSART TRANSMIT BLOCK DIAGRAM**



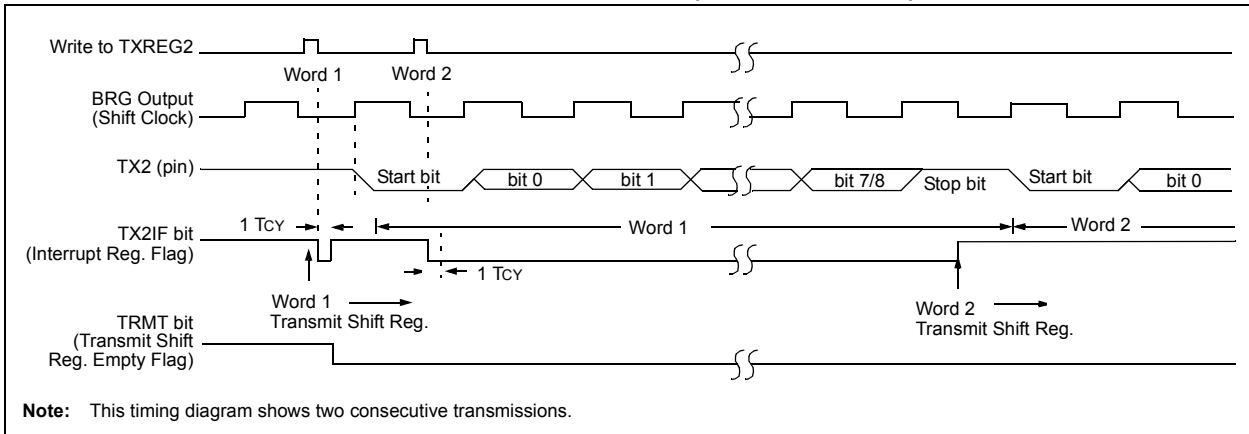


# PIC18F6390/6490/8390/8490

**FIGURE 17-2: ASYNCHRONOUS TRANSMISSION**



**FIGURE 17-3: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)**



**TABLE 17-4: 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	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 Transmit Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

# PIC18F6390/6490/8390/8490

## 17.2.2 AUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 17-4. The data is received on the RX2 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:

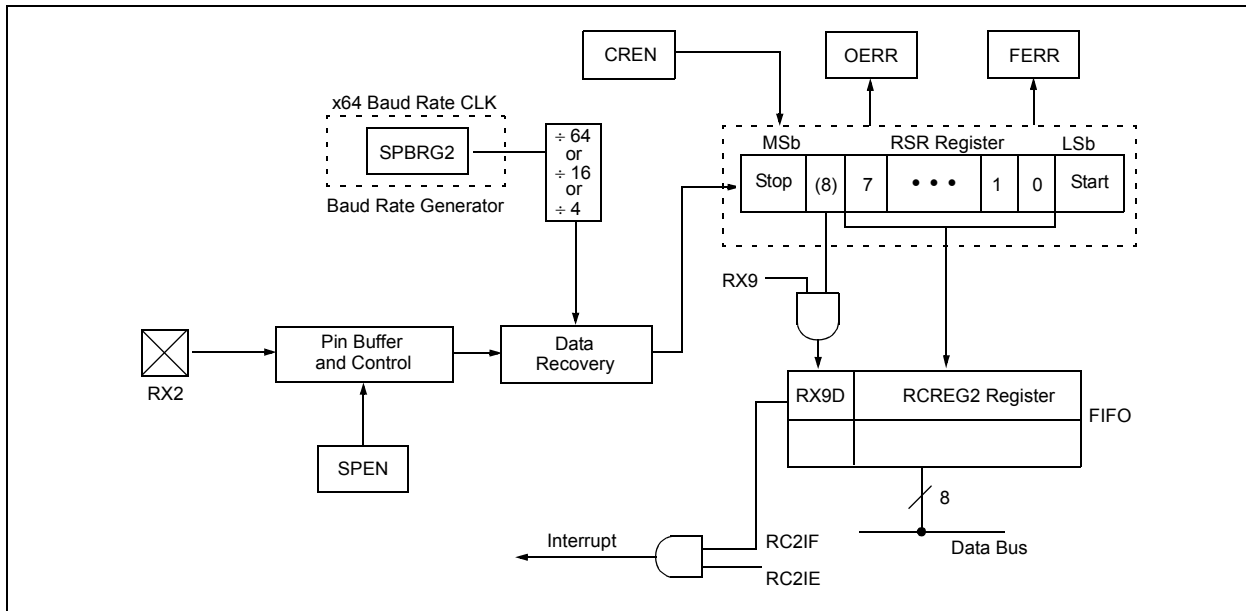
1. Initialize the SPBRG2 register for the appropriate baud rate. Set or clear the BRGH bit, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, RC2IE.
4. If 9-bit reception is desired, set bit, RX9.
5. Enable the reception by setting bit, CREN.
6. Flag bit, RC2IF, will be set when reception is complete and an interrupt will be generated if enable bit, RC2IE, was set.
7. Read the RCSTA2 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG2 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.

## 17.2.3 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:

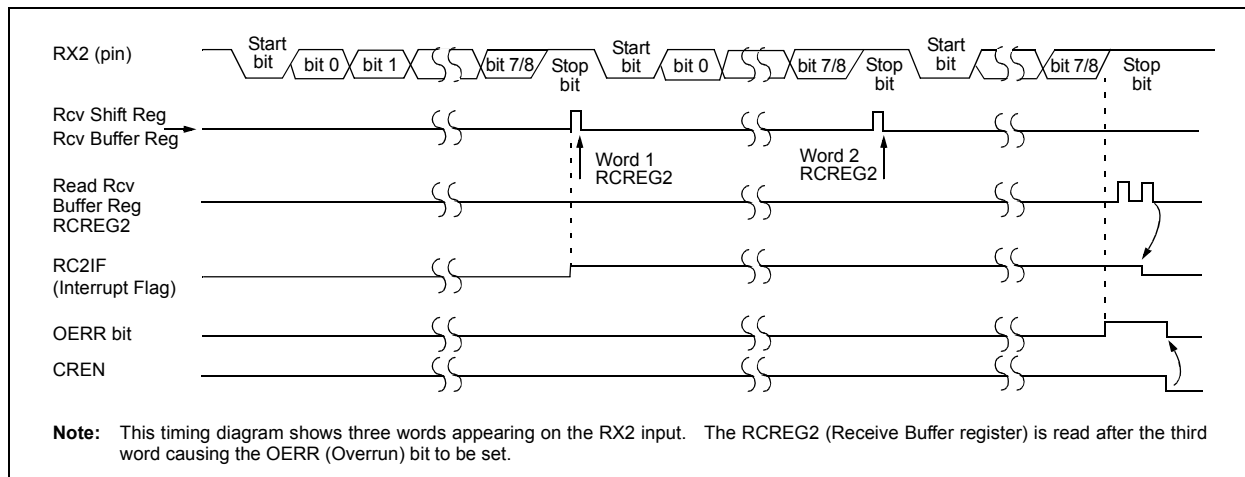
1. Initialize the SPBRG2 register for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RC2IP 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 RC2IF bit will be set when reception is complete. The interrupt will be Acknowledged if the RC2IE and GIE bits are set.
8. Read the RCSTA2 register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREG2 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 17-4: AUSART RECEIVE BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

**FIGURE 17-5: ASYNCHRONOUS RECEPTION**



**TABLE 17-5: 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 Receive Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

# PIC18F6390/6490/8390/8490

## 17.3 AUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA2<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 (TXSTA2<4>). In addition, enable bit, SPEN (RCSTA2<7>), is set in order to configure the TX2 and RX2 pins to CK2 (clock) and DT2 (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK2 line.

### 17.3.1 AUSART SYNCHRONOUS MASTER TRANSMISSION

The AUSART transmitter block diagram is shown in Figure 17-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG2. The TXREG2 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 TXREG2 (if available).

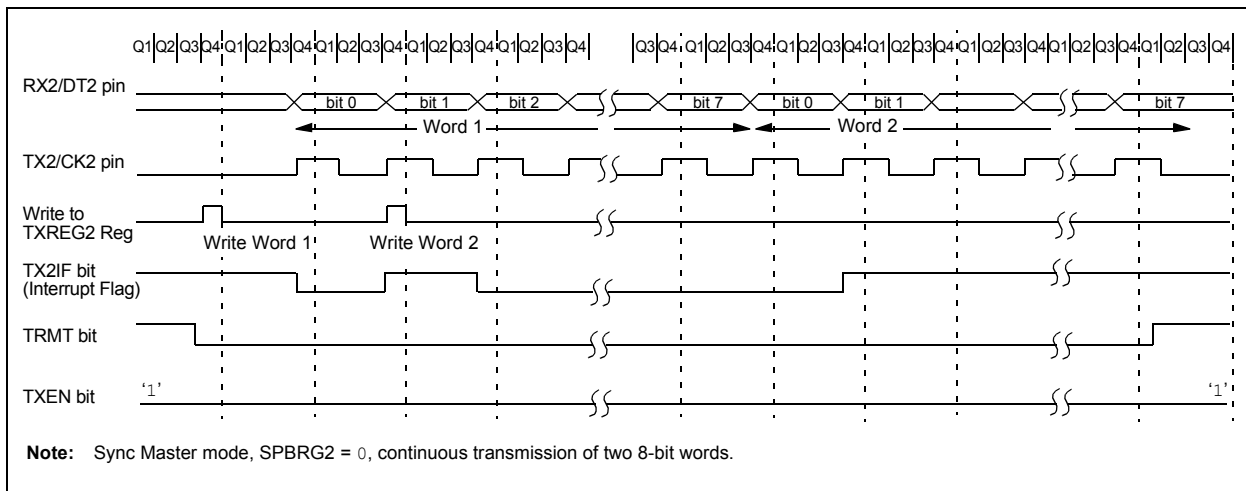
Once the TXREG2 register transfers the data to the TSR register (occurs in one T<sub>CYCLE</sub>), the TXREG2 is empty and the TX2IF flag bit (PIR3<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX2IE (PIE3<4>). TX2IF is set regardless of the state of enable bit, TX2IE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREG2 register.

While flag bit, TX2IF, indicates the status of the TXREG2 register, another bit, TRMT (TXSTA2<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

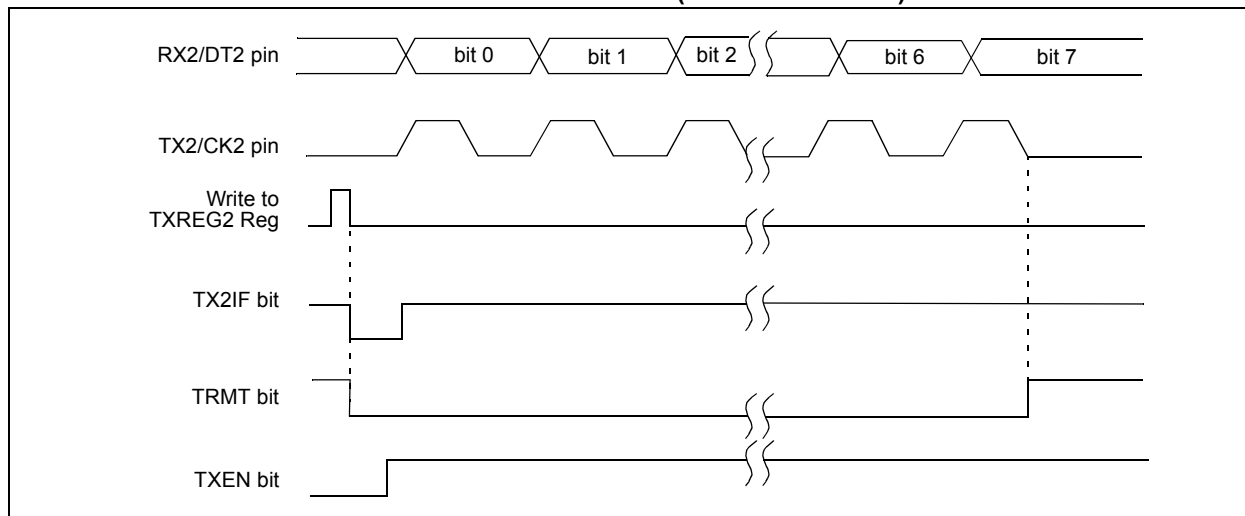
1. Initialize the SPBRG2 register for the appropriate baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TX2IE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREG2 register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 17-6: SYNCHRONOUS TRANSMISSION**



# PIC18F6390/6490/8390/8490

**FIGURE 17-7: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



**TABLE 17-6: 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 Transmit Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

# PIC18F6390/6490/8390/8490

## 17.3.2 AUSART SYNCHRONOUS MASTER RECEPTION

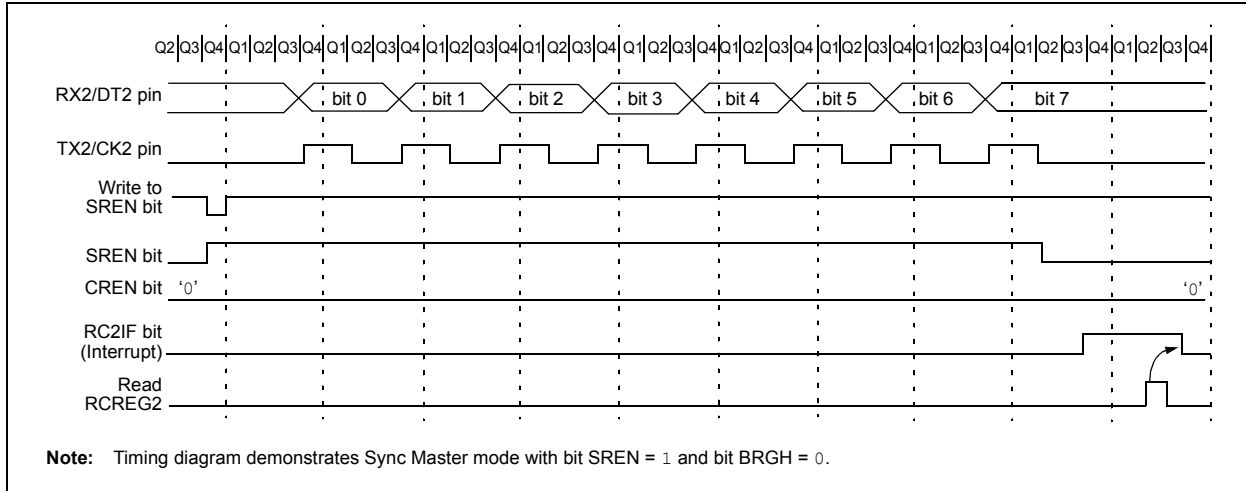
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA2<5>), or the Continuous Receive Enable bit, CREN (RCSTA2<4>). Data is sampled on the RX2 pin on the falling edge of the clock.

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 SPBRG2 register for the appropriate 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, RC2IE.
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, RC2IF, will be set when reception is complete and an interrupt will be generated if the enable bit, RC2IE, was set.
8. Read the RCSTA2 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG2 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 17-8: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**



**TABLE 17-7: 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 Receive Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

# PIC18F6390/6490/8390/8490

## 17.4 AUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA2<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CK2 pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

### 17.4.1 AUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical except in the case of the Sleep mode.

If two words are written to the TXREG2 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 TXREG2 register.
- c) Flag bit, TX2IF, will not be set.
- d) When the first word has been shifted out of TSR, the TXREG2 register will transfer the second word to the TSR and flag bit, TX2IF, will now be set.
- e) If enable bit, TX2IE, 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, TX2IE.
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 TXREG2 register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**TABLE 17-8: 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 Transmit Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

# PIC18F6390/6490/8390/8490

## 17.4.2 AUSART 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 RCREG2 register; if the RC2IE enable bit is set, the interrupt generated will wake the chip from 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, RC2IE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RC2IF, will be set when reception is complete. An interrupt will be generated if enable bit, RC2IE, was set.
6. Read the RCSTA2 register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG2 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 17-9: 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 Receive Register								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 Baud Rate Generator Register								63

**Legend:** — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.



# PIC18F6390/6490/8390/8490

## 18.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has 12 inputs for the PIC18F6X90/8X90 devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 18-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 18-2, configures the functions of the port pins. The ADCON2 register, shown in Register 18-3, configures the A/D clock source, programmed acquisition time and justification.

### REGISTER 18-1: ADCON0: A/D CONTROL REGISTER 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/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 7-6 **Unimplemented:** Read as '0'

bit 5-2 **CHS3:CHS0:** Analog Channel Select bits

0000 = Channel 0 (AN0)

0001 = Channel 1 (AN1)

0010 = Channel 2 (AN2)

0011 = Channel 3 (AN3)

0100 = Channel 4 (AN4)

0101 = Channel 5 (AN5)

0110 = Channel 6 (AN6)

0111 = Channel 7 (AN7)

1000 = Channel 8 (AN8)

1001 = Channel 9 (AN9)

1010 = Channel 10 (AN10)

1011 = Channel 11 (AN11)

1100 = Unimplemented<sup>(1)</sup>

1101 = Unimplemented<sup>(1)</sup>

1110 = Unimplemented<sup>(1)</sup>

1111 = Unimplemented<sup>(1)</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:** Performing a conversion on unimplemented channels will return a floating input measurement.

# PIC18F6390/6490/8390/8490

## REGISTER 18-2: ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-q	R/W-q	R/W-q	R/W-q
—	—	VCFG1	VCFG0	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 7-6 **Unimplemented:** Read as '0'

bit 5 **VCFG1:** Voltage Reference Configuration bit (VREF- source)

1 = VREF- (AN2)

0 = AVSS

bit 4 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)

1 = VREF+ (AN3)

0 = AVDD

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits:

PCFG3: PCFG0	AN11	AN10	AN9	AN8	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A
0011	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input

D = Digital I/O

# PIC18F6390/6490/8390/8490

**REGISTER 18-3: ADCON2: A/D CONTROL REGISTER 2**

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

**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        **Unimplemented:** Read as '0'
- bit 5-3     **ACQT2:ACQT0:** A/D Acquisition Time Select bits
  - 111 = 20 TAD
  - 110 = 16 TAD
  - 101 = 12 TAD
  - 100 = 8 TAD
  - 011 = 6 TAD
  - 010 = 4 TAD
  - 001 = 2 TAD
  - 000 = 0 TAD<sup>(1)</sup>
- bit 2-0     **ADCS2:ADCS0:** A/D Conversion Clock Select bits
  - 111 = FRC (clock derived from A/D RC oscillator)<sup>(1)</sup>
  - 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.

# PIC18F6390/6490/8390/8490

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+/SEG17 and RA2/AN2/VREF-/SEG16 pins.

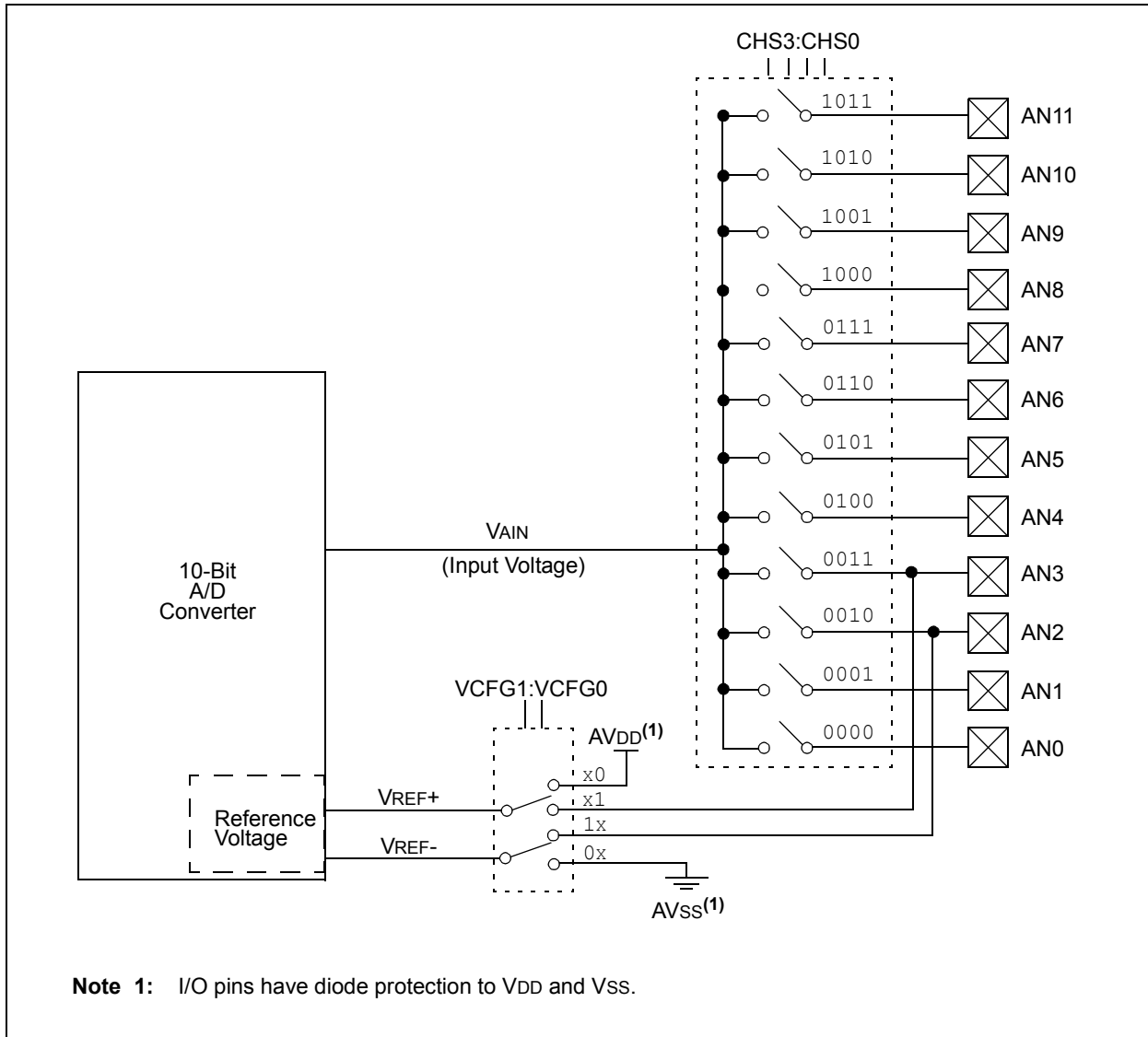
The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

A device Reset forces all registers to their Reset state. This forces the A/D module to be turned off and any conversion in progress is aborted.

Each port pin associated with the A/D converter can be configured as an analog input or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH/ADRESL registers, the GO/DONE bit (ADCON0<1>) is cleared and the A/D Interrupt Flag bit, ADIF, is set. The block diagram of the A/D module is shown in Figure 18-1.

**FIGURE 18-1: A/D BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

The value in the ADRESH:ADRESL registers is not modified for a Power-on Reset. The ADRESH:ADRESL registers will contain unknown data after a Power-on Reset.

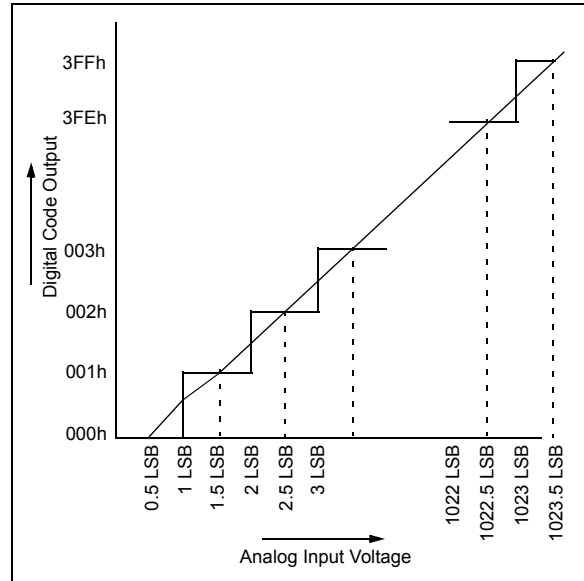
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as an input. To determine acquisition time, see **Section 18.1 “A/D Acquisition Requirements”**. After 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 perform an A/D conversion:

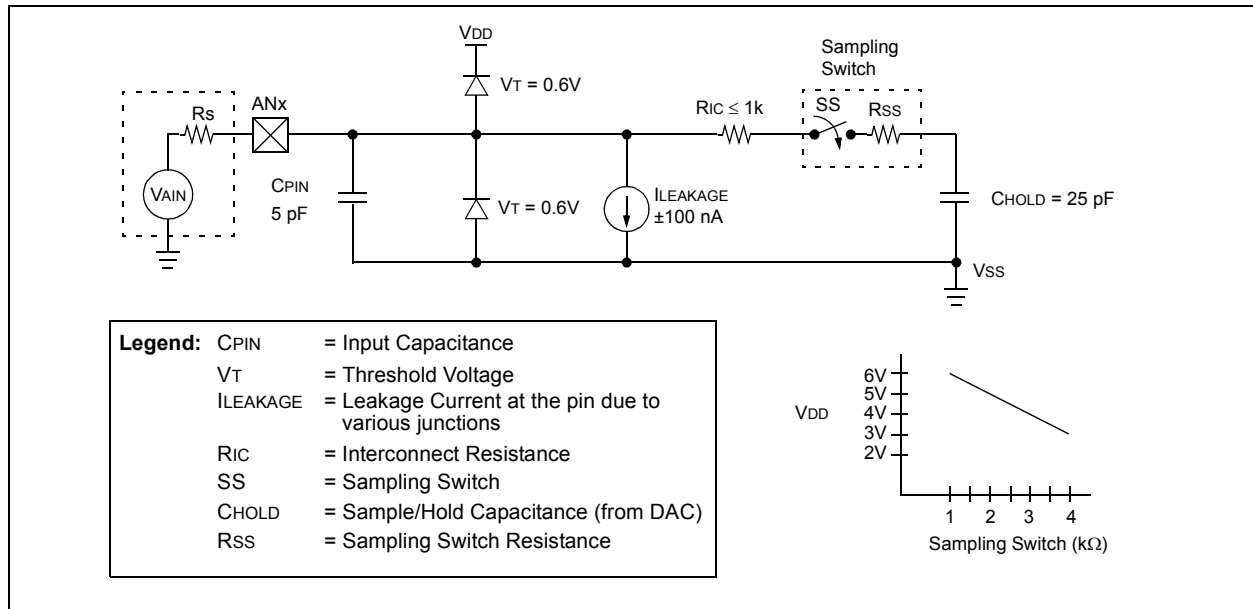
1. Configure the A/D module:
  - Configure analog pins, voltage reference and digital I/O (ADCON1)
  - Select A/D input channel (ADCON0)
  - Select A/D acquisition time (ADCON2)
  - Select A/D conversion clock (ADCON2)
  - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
  - Clear ADIF bit
  - Set ADIE bit
  - Set GIE bit
3. Wait the required acquisition time (if required).
4. Start conversion:
  - Set  $\overline{\text{GO/DONE}}$  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 ADIF bit, if required.
7. For next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is required. The A/D conversion time per bit is defined as TAD. A minimum wait of 3 TAD is required before the next acquisition starts.

**FIGURE 18-2: A/D TRANSFER FUNCTION**



**FIGURE 18-3: ANALOG INPUT MODEL**



# PIC18F6390/6490/8390/8490

## 18.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 18-3. The source impedance (Rs) and the internal sampling switch (Rss) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (Rss) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ.** After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

**Note:** When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 18-1 may be used. This equation assumes that 1/2 LSB error is used (1024 steps for the A/D). The 1/2 LSB error is the maximum error allowed for the A/D to meet its specified resolution.

Example 18-3 shows the calculation of the minimum required acquisition time, TACQ. This calculation is based on the following application system assumptions:

CHOLD	=	25 pF
Rs	=	2.5 kΩ
Conversion Error	≤	1/2 LSB
VDD	=	5V → Rss = 2 kΩ
Temperature	=	85°C (system max.)

### EQUATION 18-1: ACQUISITION TIME

$$\begin{aligned}TACQ &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= TAMP + TC + TCOFF\end{aligned}$$

### EQUATION 18-2: A/D MINIMUM CHARGING TIME

$$\begin{aligned}V_{HOLD} &= (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{-(TC/CHOLD)(RIC + R_{SS} + R_S)}) \\ \text{or} \\ TC &= -(CHOLD)(RIC + R_{SS} + R_S) \ln(1/2048)\end{aligned}$$

### EQUATION 18-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$TACQ = TAMP + TC + TCOFF$$

$$TAMP = 0.2 \mu s$$

$$\begin{aligned}TCOFF &= (\text{Temp} - 25^\circ C)(0.02 \mu s/^\circ C) \\ &= (50^\circ C - 25^\circ C)(0.02 \mu s/^\circ C) \\ &= 1.2 \mu s\end{aligned}$$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, TCOFF = 0 ms.

$$\begin{aligned}TC &= -(CHOLD)(RIC + R_{SS} + R_S) \ln(1/2047) \mu s \\ &= -(25 \text{ pF})(1 \text{ k}\Omega + 2 \text{ k}\Omega + 2.5 \text{ k}\Omega) \ln(0.0004883) \mu s \\ &= 5.03 \mu s\end{aligned}$$

$$\begin{aligned}TACQ &= 0.2 \mu s + 5 \mu s + 1.2 \mu s \\ &= 6.4 \mu s\end{aligned}$$

## 18.2 Selecting and Configuring Automatic Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time the  $\overline{\text{GO/DONE}}$  bit is set.

When the  $\overline{\text{GO/DONE}}$  bit is set, sampling is stopped and a conversion begins. The user is responsible for ensuring the required acquisition time has passed between selecting the desired input channel and setting the  $\overline{\text{GO/DONE}}$  bit. This occurs when the ACQT2:ACQT0 bits (ADCON2<5:3>) remain in their Reset state ('000') and is compatible with devices that do not offer programmable acquisition times.

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When the  $\overline{\text{GO/DONE}}$  bit is set, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and setting the  $\overline{\text{GO/DONE}}$  bit.

In either case, when the conversion is completed, the  $\overline{\text{GO/DONE}}$  bit is cleared, the ADIF flag is set and the A/D begins sampling the currently selected channel again. If an acquisition time is programmed, there is nothing to indicate if the acquisition time has ended or if the conversion has begun.

## 18.3 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as  $T_{AD}$ . The A/D conversion requires 11  $T_{AD}$  per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are seven possible options for  $T_{AD}$ :

- 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 ( $T_{AD}$ ) must be as short as possible, but greater than the minimum  $T_{AD}$  (approximately 2  $\mu\text{s}$ , see parameter 130 for more information).

Table 18-1 shows the resultant  $T_{AD}$  times derived from the device operating frequencies and the A/D clock source selected.

**TABLE 18-1:  $T_{AD}$  vs. DEVICE OPERATING FREQUENCIES**

AD Clock Source ( $T_{AD}$ )		Maximum Device Frequency	
Operation	ADCS2:ADCS0	PIC18F6X90/8X90	PIC18LF6X90/8X90 <sup>(4)</sup>
2 TOSC	000	1.25 MHz	666 kHz
4 TOSC	100	2.50 MHz	1.33 MHz
8 TOSC	001	5.00 MHz	2.66 MHz
16 TOSC	101	10.0 MHz	5.33 MHz
32 TOSC	010	20.0 MHz	10.65 MHz
64 TOSC	110	40.0 MHz	21.33 MHz
RC <sup>(3)</sup>	x11	1.00 MHz <sup>(1)</sup>	1.00 MHz <sup>(2)</sup>

- Note 1:** The RC source has a typical  $T_{AD}$  time of 4  $\mu\text{s}$ .
- Note 2:** The RC source has a typical  $T_{AD}$  time of 6  $\mu\text{s}$ .
- Note 3:** For device frequencies above 1 MHz, the device must be in Sleep for the entire conversion or the A/D accuracy may be out of specification.
- Note 4:** Low-power (PIC18LFXXXX) devices only.

# PIC18F6390/6490/8390/8490

---

## 18.4 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined in part by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT2:ACQT0 and ADCS2:ADCS0 bits in ADCON2 should be updated in accordance with the power-managed mode clock that will be used. After the power-managed mode is entered, 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 FRC clock to be selected. If bits, ACQT2:ACQT0, are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS bits in the OSCCON register must have already been cleared prior to starting the conversion.

## 18.5 Configuring Analog Port Pins

The ADCON1, TRISA and TRISF registers all configure the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS3:CHS0 bits and the TRIS bits.

**Note 1:** When reading the PORT register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert 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.



## 18.6 A/D Conversions

Figure 18-4 shows the operation of the A/D converter after the  $\overline{\text{GO/DONE}}$  bit has been set and the ACQT2:ACQT0 bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins.

Figure 18-5 shows the operation of the A/D converter after the  $\overline{\text{GO/DONE}}$  bit has been set, the ACQT2:ACQT0 bits are set to '010' and a 4 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 ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers).

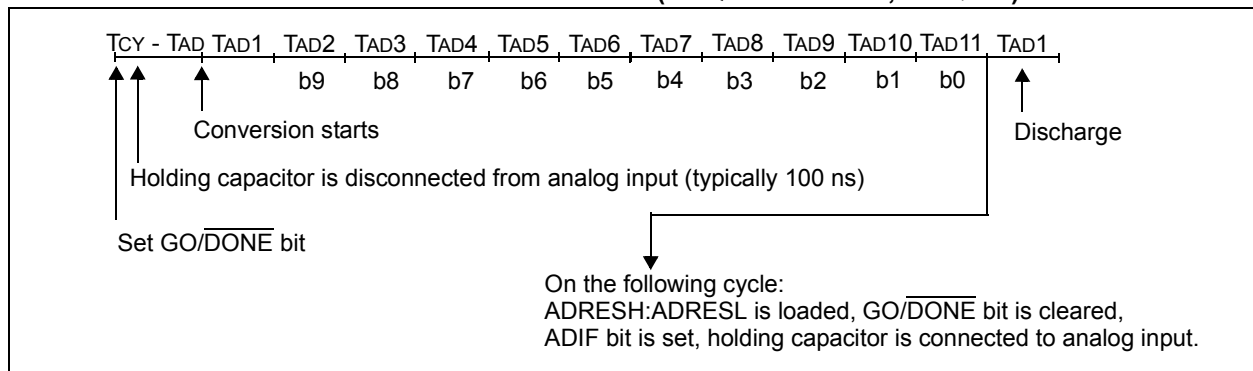
After the A/D conversion is completed or aborted, a 2 TAD wait is required before the next acquisition can be started. After this wait, acquisition on the selected channel is automatically started.

**Note:** The  $\overline{\text{GO/DONE}}$  bit should **NOT** be set in the same instruction that turns on the A/D.

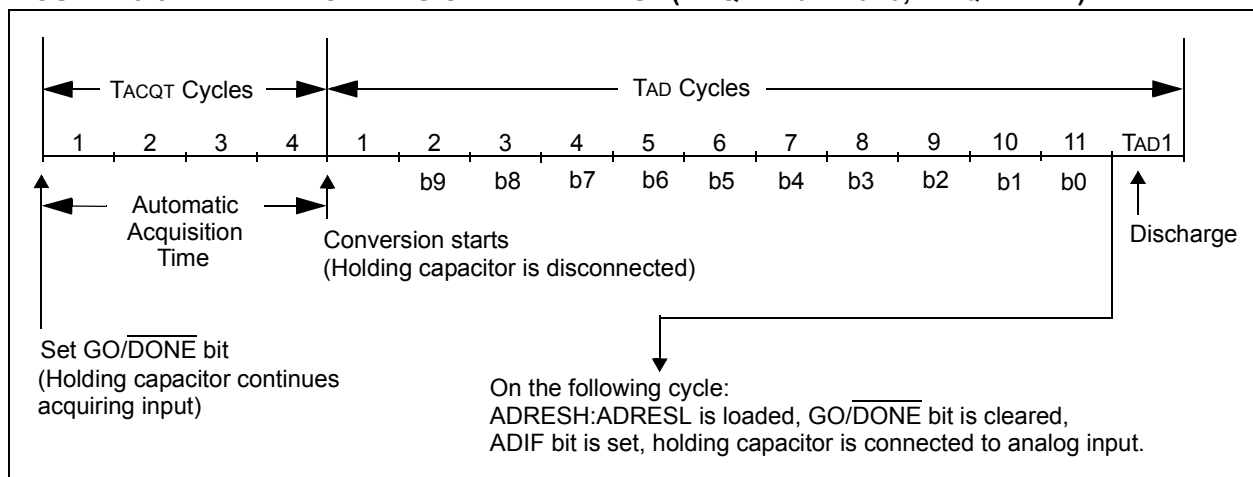
## 18.7 Discharge

The discharge phase is used to initialize the value of the capacitor array. The array is discharged before every sample. This feature helps to optimize the unity-gain amplifier as the circuit always needs to charge the capacitor array, rather than charge/discharge based on previous measure values.

**FIGURE 18-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)**



**FIGURE 18-5: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)**



# PIC18F6390/6490/8390/8490

## 18.8 Use of the CCP2 Trigger

An A/D conversion can be started by the “Special Event Trigger” of the CCP2 module. This requires that the CCP2M3:CCP2M0 bits (CCP2CON<3:0>) be programmed as ‘1011’ and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/DONE bit will be set, starting the A/D acquisition and conversion and the Timer1 (or Timer3) counter will be reset to zero. Timer1 (or Timer3) is reset to automatically repeat the A/D acquisition period with minimal

software overhead (moving ADRESH/ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition period is either timed by the user, or an appropriate TACQ time selected before the “Special Event Trigger” sets the GO/DONE bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), the “Special Event Trigger” will be ignored by the A/D module, but will still reset the Timer1 (or Timer3) counter.

**TABLE 18-2: REGISTERS ASSOCIATED WITH A/D OPERATION**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
ADRESH	A/D Result Register High Byte								61
ADRESL	A/D Result Register Low Byte								61
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	61
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	61
PORTA	RA7 <sup>(1)</sup>	RA6 <sup>(1)</sup>	RA5	RA4	RA3	RA2	RA1	RA0	62
TRISA	TRISA7 <sup>(1)</sup>	TRISA6 <sup>(1)</sup>	PORTA Data Direction Register						62
PORTF	Read PORTF pins, Write LATCH Latch								62
TRISF	PORTF Data Direction Register								62
LATF	LATF Data Output Register								62

**Legend:** — = unimplemented, read as ‘0’. Shaded cells are not used for A/D conversion.

**Note 1:** These pins may be configured as port pins depending on the oscillator mode selected.

# PIC18F6390/6490/8390/8490

## 19.0 COMPARATOR MODULE

The analog comparator module contains two comparators that can be configured in a variety of ways. The inputs can be selected from the analog inputs multiplexed with pins RF3 through RF6, as well as the on-chip voltage reference (see **Section 20.0 “Comparator Voltage Reference Module”**). The digital outputs (normal or inverted) are available at the pin level and can also be read through the control register.

The CMCON register (Register 19-1) selects the comparator input and output configuration. Block diagrams of the various comparator configurations are shown in Figure 19-1.

**REGISTER 19-1: CMCON: COMPARATOR CONTROL REGISTER**

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

**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      **C2OUT:** Comparator 2 Output bit  
             When C2INV = 0:  
             1 = C2 VIN+ > C2 VIN-  
             0 = C2 VIN+ < C2 VIN-  
             When C2INV = 1:  
             1 = C2 VIN+ < C2 VIN-  
             0 = C2 VIN+ > C2 VIN-
- bit 6      **C1OUT:** Comparator 1 Output bit  
             When C1INV = 0:  
             1 = C1 VIN+ > C1 VIN-  
             0 = C1 VIN+ < C1 VIN-  
             When C1INV = 1:  
             1 = C1 VIN+ < C1 VIN-  
             0 = C1 VIN+ > C1 VIN-
- bit 5      **C2INV:** Comparator 2 Output Inversion bit  
             1 = C2 output inverted  
             0 = C2 output not inverted
- bit 4      **C1INV:** Comparator 1 Output Inversion bit  
             1 = C1 Output inverted  
             0 = C1 Output not inverted
- bit 3      **CIS:** Comparator Input Switch bit  
             When CM2:CM0 = 110:  
             1 = C1 VIN- connects to RF5/AN10  
                     C2 VIN- connects to RF3/AN8  
             0 = C1 VIN- connects to RF6/AN11  
                     C2 VIN- connects to RF4/AN9
- bit 2-0    **CM2:CM0:** Comparator Mode bits  
             Figure 19-1 shows the Comparator modes and the CM2:CM0 bit settings.

# PIC18F6390/6490/8390/8490

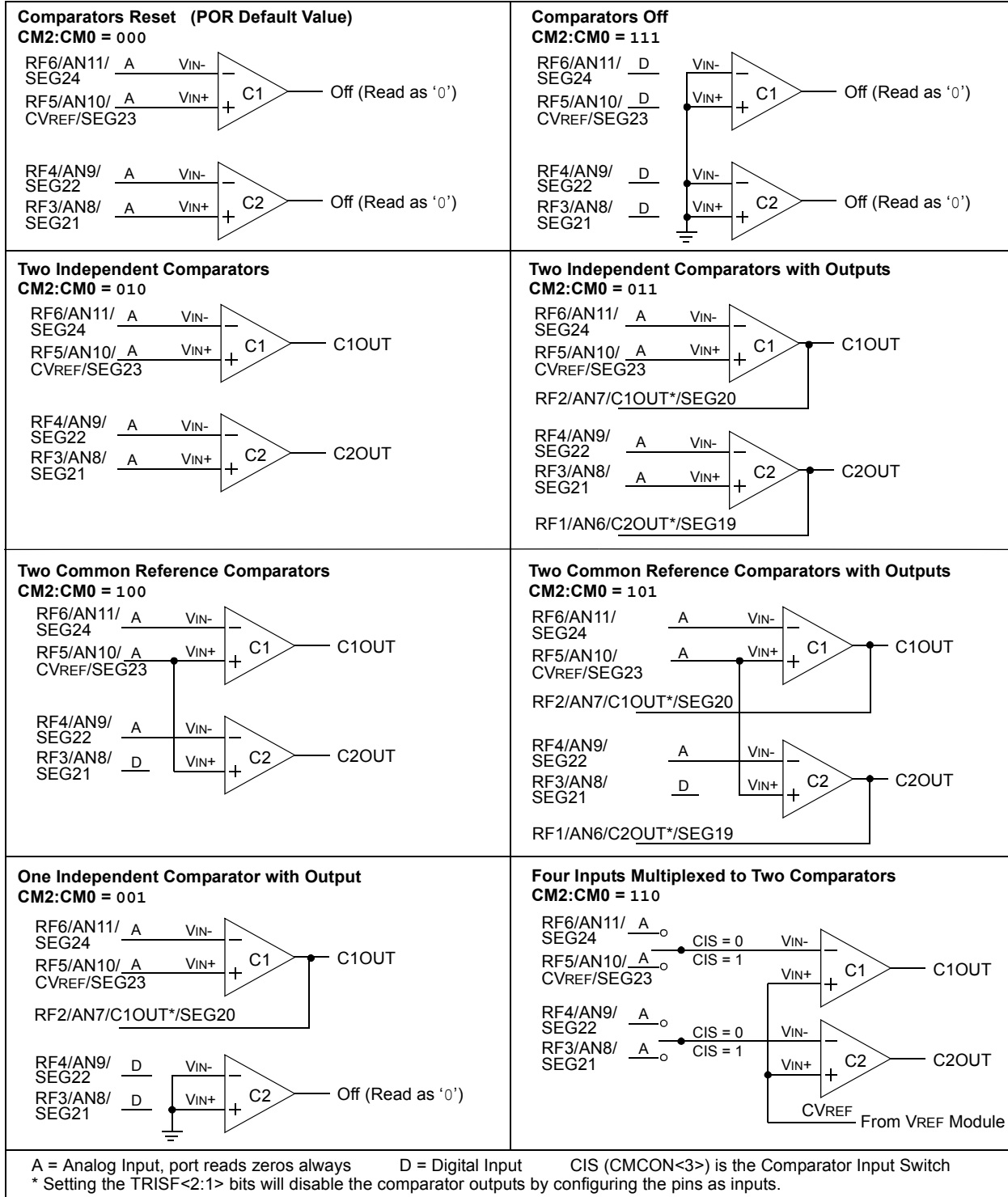
## 19.1 Comparator Configuration

There are eight modes of operation for the comparators, shown in Figure 19-1. Bits, CM2:CM0 of the CMCON register, are used to select these modes. The TRISF register controls the data direction of the comparator pins for each mode. If the Comparator

mode is changed, the comparator output level may not be valid for the specified mode change delay shown in Section 26.0 “Electrical Characteristics”.

**Note:** Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur.

**FIGURE 19-1: COMPARATOR I/O OPERATING MODES**



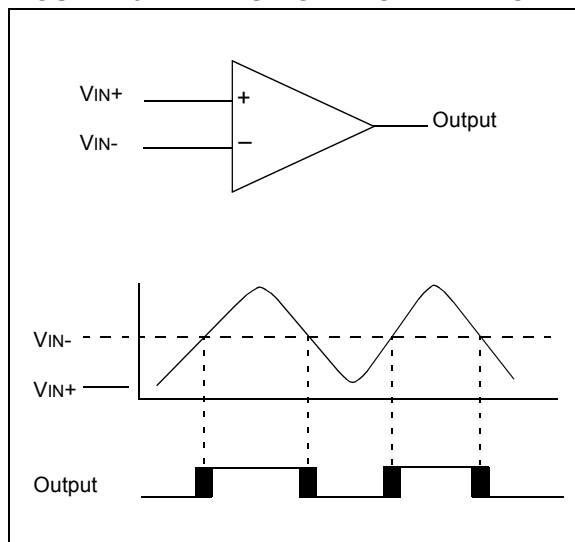
## 19.2 Comparator Operation

A single comparator is shown in Figure 19-2, 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 19-2 represent the uncertainty, due to input offsets and response time.

## 19.3 Comparator Reference

Depending on the comparator operating mode, either an external or internal voltage reference may be used. The analog signal present at  $V_{IN-}$  is compared to the signal at  $V_{IN+}$  and the digital output of the comparator is adjusted accordingly (Figure 19-2).

**FIGURE 19-2: SINGLE COMPARATOR**



### 19.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same, or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between  $V_{SS}$  and  $V_{DD}$  and can be applied to either pin of the comparator(s).

### 19.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference from the comparator voltage reference module. This module is described in more detail in **Section 20.0 “Comparator Voltage Reference Module”**.

The internal reference is only available in the mode where four inputs are multiplexed to two comparators ( $CM2:CM0 = 110$ ). In this mode, the internal voltage reference is applied to the  $V_{IN+}$  pin of both comparators.

## 19.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise, the maximum delay of the comparators should be used (see **Section 26.0 “Electrical Characteristics”**).

## 19.5 Comparator Outputs

The comparator outputs are read through the  $CMCON$  register. These bits are read-only. The comparator outputs may also be directly output to the  $RF2$  and  $RF1$  I/O pins. When enabled, multiplexers in the output path of the  $RF2$  and  $RF1$  pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 19-3 shows the comparator output block diagram.

The  $TRISF$  bits will still function as an output enable/disable for the  $RF2$  and  $RF1$  pins while in this mode.

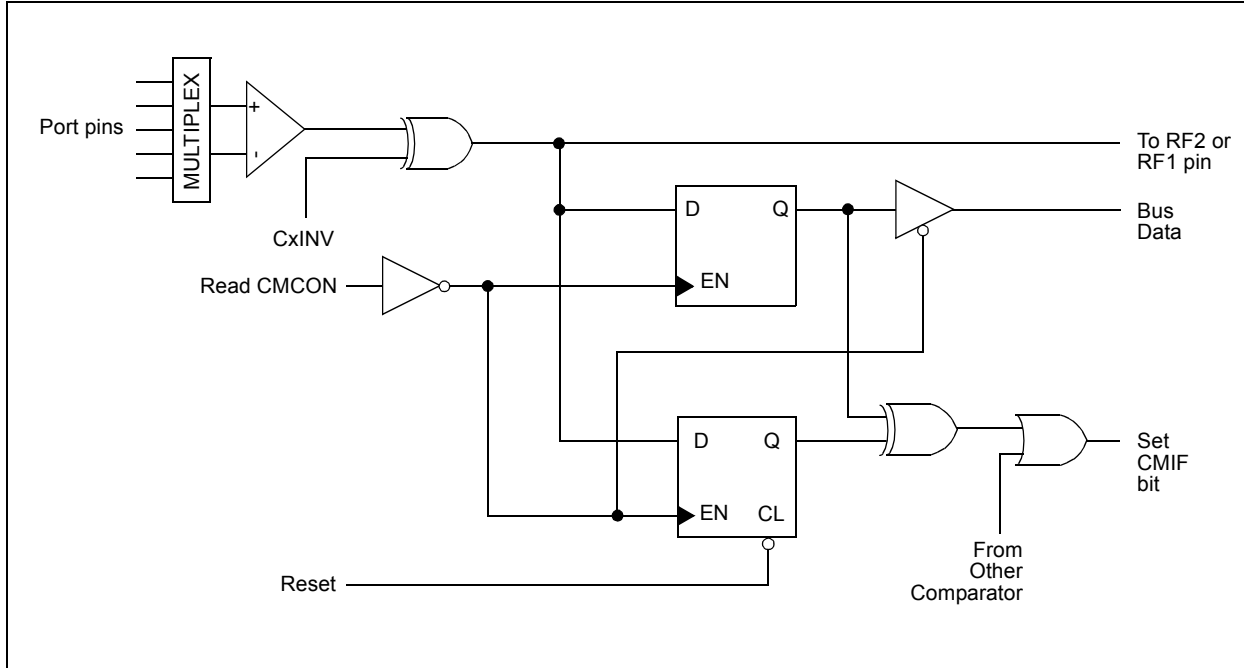
The polarity of the comparator outputs can be changed using the  $C2INV$  and  $C1INV$  bits ( $CMCON<5:4>$ ).

**Note 1:** When reading the  $PORT$  register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert an analog input according to the Schmitt Trigger input specification.

**2:** Analog levels on any pin defined as a digital input may cause the input buffer to consume more current than is specified.

# PIC18F6390/6490/8390/8490

FIGURE 19-3: COMPARATOR OUTPUT BLOCK DIAGRAM



## 19.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR2<6>) is the Comparator Interrupt Flag. The CMIF bit must be reset by clearing it. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

Both the CMIE bit (PIE2<6>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

**Note:** If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR2<6>) interrupt flag may not get set.

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

- Any read or write of CMCON will end the mismatch condition.
- Clear flag bit, CMIF.

A mismatch condition will continue to set flag bit, CMIF. Reading CMCON will end the mismatch condition and allow flag bit, CMIF, to be cleared.

## 19.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional, if enabled. This interrupt will wake-up the device from Sleep mode, when enabled. While the comparator is powered up, higher Sleep currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators (CM2:CM0 = 111) before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

## 19.8 Effects of a Reset

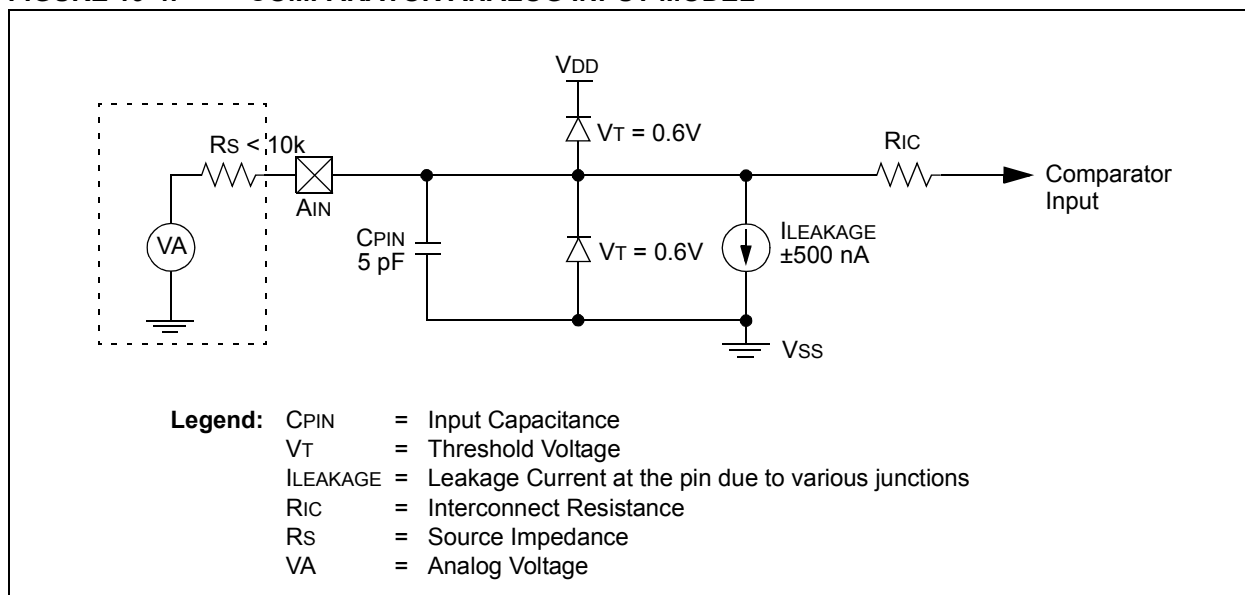
A device Reset forces the CMCON register to its Reset state, causing the comparator module to be in the Comparator Reset mode (CM2:CM0 = 000). This ensures that all potential inputs are analog inputs. Device current is minimized when analog inputs are present at Reset time. The comparators are powered down during the Reset interval.

## 19.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 19-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this

range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

**FIGURE 19-4: COMPARATOR ANALOG INPUT MODEL**



**TABLE 19-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
PORTF	Read PORTF pins, Write LATF Latch								62
LATF	LATF Data Output Register								62
TRISF	PORTF Data Direction Register								62

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the comparator module.

# PIC18F6390/6490/8390/8490

---

NOTES:



## 20.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable reference voltage. Although its primary purpose is to provide a reference for the analog comparators, it may also be used independently of them.

A block diagram of the module is shown in Figure 20-1. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device VDD/VSS, or an external voltage reference.

### 20.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 20-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be

used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF selection bits (CVR3:CVR0), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:

$$CVREF = ((CVR3:CVR0)/24) \times CVRSRC$$

If CVRR = 0:

$$CVREF = (CVDD \times 1/4) + (((CVR3:CVR0)/32) \times CVRSRC)$$

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF- that are multiplexed with RA2 and RA3. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 26-3 in **Section 26.0 "Electrical Characteristics"**).

**REGISTER 20-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE <sup>(1)</sup>	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

**Legend:**

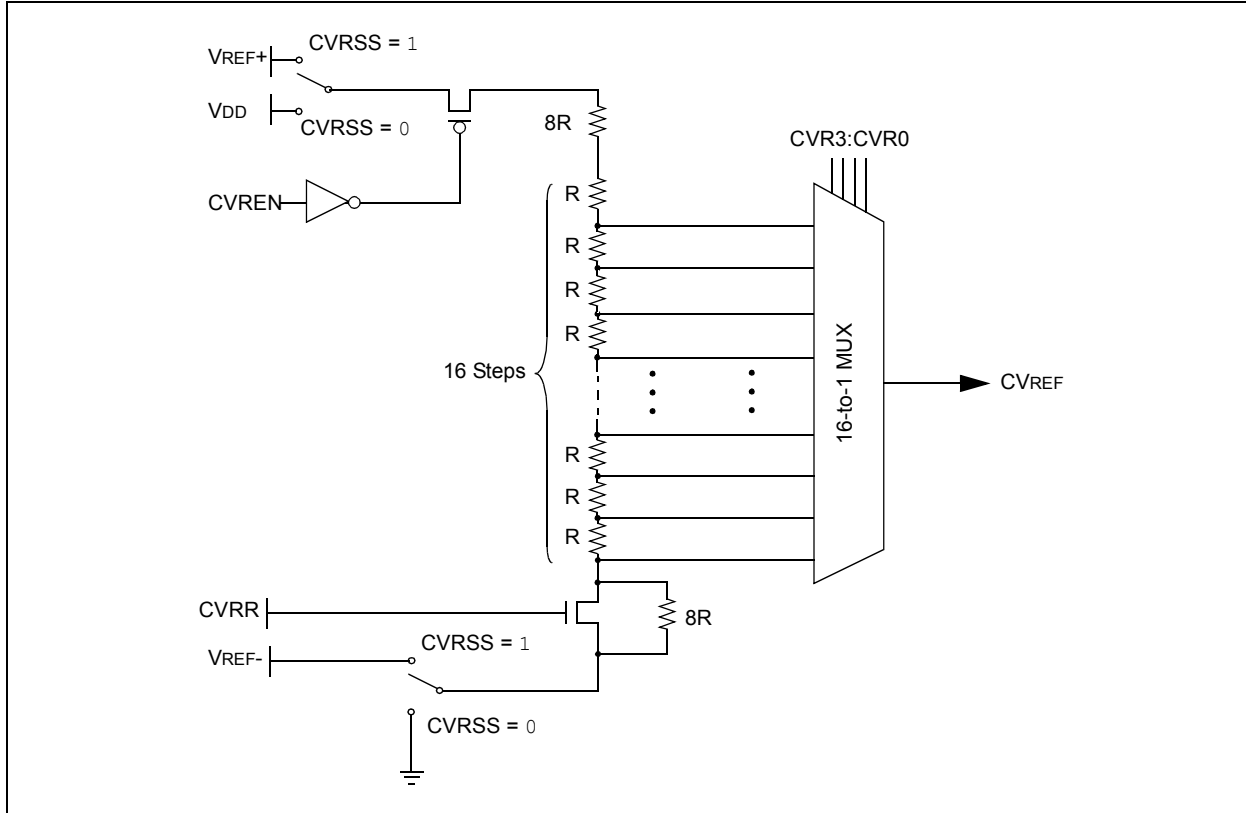
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

- bit 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 RF5/AN10/CVREF/SEG23 pin  
0 = CVREF voltage is disconnected from the RF5/AN10/CVREF/SEG23 pin
- bit 5 **CVRR:** Comparator VREF Range Selection bit  
1 = 0.00 CVRSRC to 0.75 CVRSRC, with CVRSRC/24 step size  
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size
- bit 4 **CVRSS:** Comparator VREF Source Selection bit  
1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)  
0 = Comparator reference source, CVRSRC = VDD – VSS
- bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ (CVR3:CVR0) ≤ 15)  
When CVRR = 1:  
CVREF = ((CVR3:CVR0)/24) • (CVRSRC)  
When CVRR = 0:  
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

**Note 1:** CVROE overrides the TRISF<5> bit setting if enabled for output; RF5 must also be configured as an input by setting TRISF<5> to '1'.

# PIC18F6390/6490/8390/8490

FIGURE 20-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



## 20.2 Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 20-1) 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 26.0 “Electrical Characteristics”**.

## 20.3 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the  $CVRCON$  register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

## 20.4 Effects of a Reset

A device Reset disables the voltage reference by clearing bit,  $CVREN$  ( $CVRCON<7>$ ). This Reset also disconnects the reference from the  $RA2$  pin by clearing bit,  $CVROE$  ( $CVRCON<6>$ ), and selects the high-voltage range by clearing bit,  $CVRR$  ( $CVRCON<5>$ ). The  $CVR$  value select bits are also cleared.

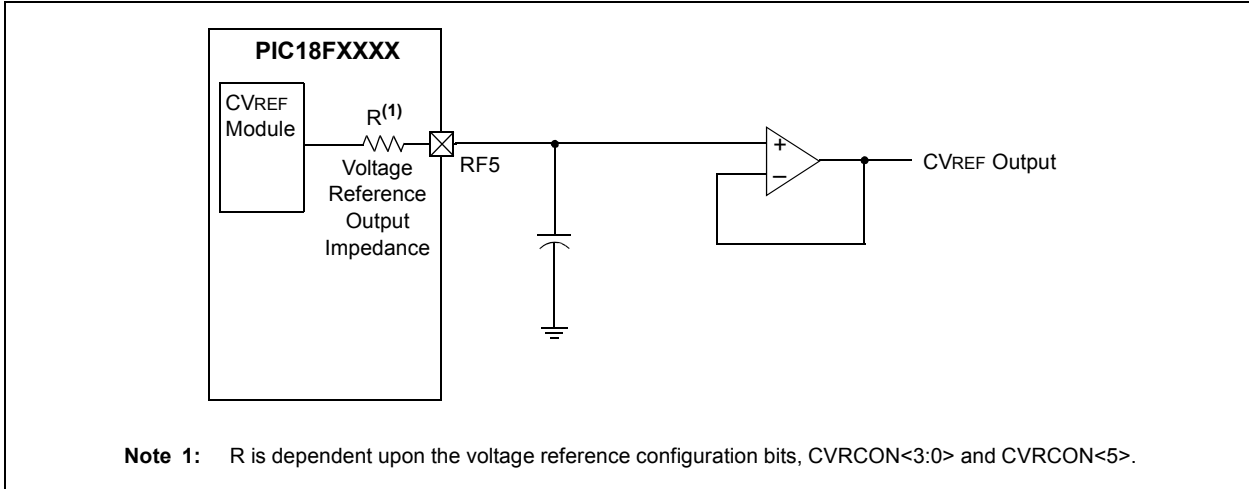
## 20.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the  $RF5$  pin if the  $TRISF<5>$  bit and the  $CVROE$  bit are both set. Enabling the voltage reference output onto the  $RF5$  pin, with an input signal present, will increase current consumption. Connecting  $RF5$  as a digital output with  $CVRSS$  enabled will also increase current consumption.

The  $RF5$  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  $V_{REF}$ . Figure 20-2 shows an example buffering technique.

# PIC18F6390/6490/8390/8490

**FIGURE 20-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



**TABLE 20-1: REGISTERS ASSOCIATED WITH THE 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	61
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
TRISF	PORTF Data Direction Register								62

**Legend:** Shaded cells are not used with the comparator voltage reference.

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 21.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

PIC18F6390/6490/8390/8490 devices have a High/Low-Voltage Detect module (HLVD). 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 device experiences 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 21-1) 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.

The block diagram for the HLVD module is shown in Figure 21-1.

### REGISTER 21-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER

R/W-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	—	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 (HLVDL3:HLVDL0)  
 0 = Event occurs when voltage equals or falls below trip point (HLVDL3:HLVDL0)
- bit 6      **Unimplemented:** Read as '0'
- 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    **HLVDL3:HLVDL0:** Voltage Detection Limit bits<sup>(1)</sup>  
 1111 = External analog input is used (input comes from the HLVDIN pin)  
 1110 = 4.41V-4.87V  
 1101 = 4.11V-4.55V  
 1100 = 3.92V-4.34V  
 1011 = 3.72V-4.12V  
 1010 = 3.53V-3.91V  
 1001 = 3.43V-3.79V  
 1000 = 3.24V-3.58V  
 0111 = 2.95V-3.26V  
 0110 = 2.75V-3.03V  
 0101 = 2.64V-2.92V  
 0100 = 2.43V-2.69V  
 0011 = 2.35V-2.59V  
 0010 = 2.16V-2.38V  
 0001 = 1.96V-2.16V  
 0000 = Reserved

**Note 1:** HLVDL3:HLVDL0 modes that result in a trip point below the valid operating voltage of the device are not tested.

# PIC18F6390/6490/8390/8490

The module is enabled by setting the HLVDEN bit. Each time that the HLVD module is enabled, the circuitry requires some time to stabilize. The IRVST bit is a read-only bit and is used to indicate when the circuit is stable. The module can only generate an interrupt 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.

## 21.1 Operation

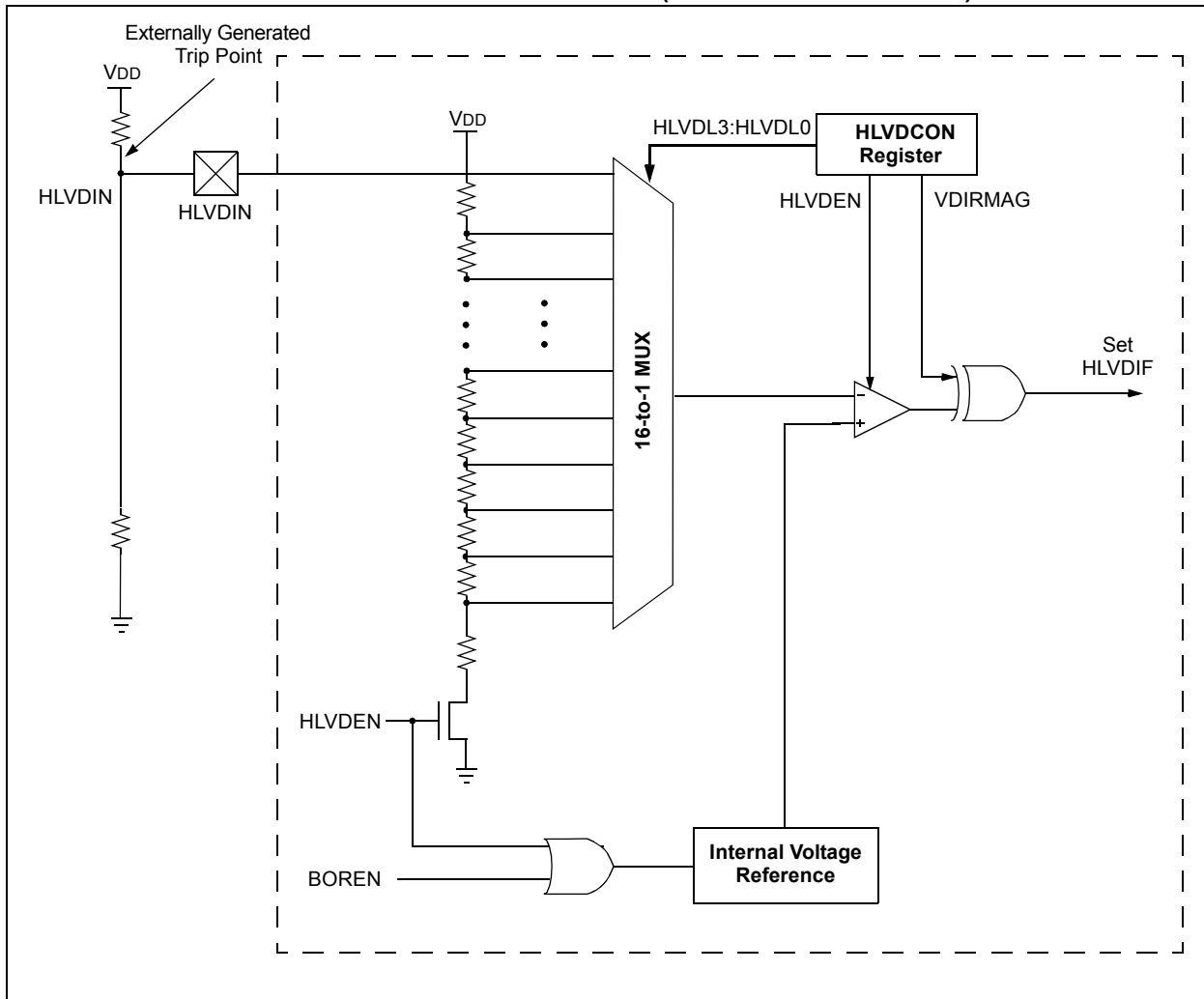
When the HLVD module is enabled, a comparator uses an internally generated reference voltage as the set point. The set point is compared with the trip point where each node in the resistor divider represents a trip point voltage. The “trip point” voltage is the voltage level at which the device detects 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 HLVDIF bit.

The trip point voltage is software programmable to any one of 16 values. The trip point is selected by programming the HLVDL3:HLVDL0 bits (HLVDCON<3:0>).

The HLVD module has an additional feature that allows the user to supply the trip voltage to the module from an external source. This mode is enabled when bits, HLVDL3:HLVDL0, 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 High/Low-Voltage Detect interrupt to occur at any voltage in the valid operating range.

FIGURE 21-1: HLVD MODULE BLOCK DIAGRAM (WITH EXTERNAL INPUT)



## 21.2 HLVD Setup

The following steps are needed to set up the HLVD module:

1. Disable the module by clearing the HLVDEN bit (HLVDCON<4>).
2. Write the value to the HLVDL3:HLVDL0 bits that selects the desired HLVD trip point.
3. Set the VDIRMAG bit to detect high voltage (VDIRMAG = 1) or low voltage (VDIRMAG = 0).
4. Enable the HLVD module by setting the HLVDEN bit.
5. Clear the HLVD interrupt flag (PIR2<2>), which may have been set from a previous interrupt.
6. Enable the HLVD interrupt, if interrupts are desired, by setting the HLVDIE and GIE bits (PIE<2> and INTCON<7>). An interrupt will not be generated until the IRVST bit is set.

## 21.3 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.

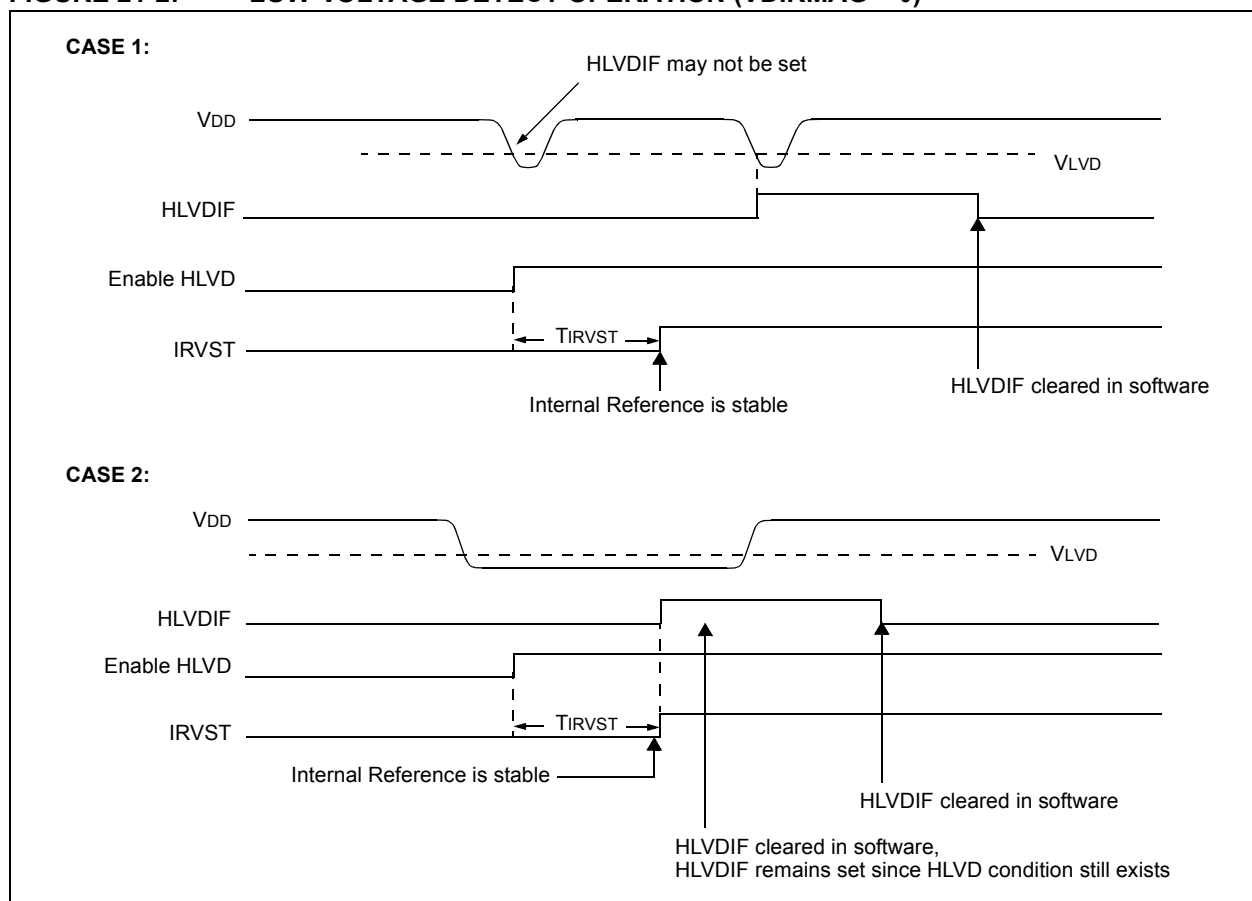
Depending on the application, the HLVD module does not need to be operating constantly. To decrease the current requirements, the HLVD circuitry may only need to be enabled for short periods where the voltage is checked. After doing the check, the HLVD module may be disabled.

## 21.4 HLVD Start-up Time

The internal reference voltage of the HLVD module, specified in electrical specification parameter #D423, may be used by other internal circuitry, such as the programmable Brown-out Reset. 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 26-10).

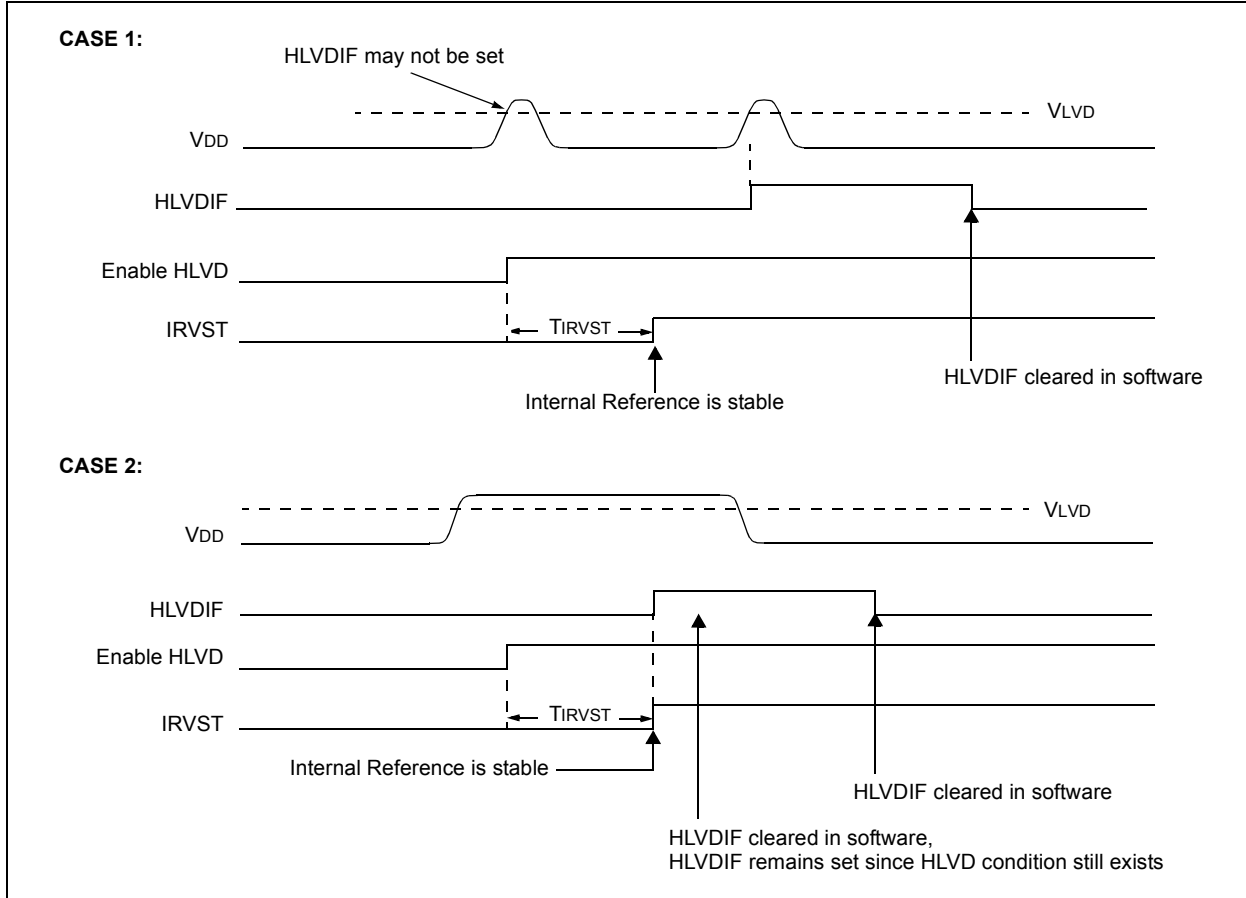
The HLVD interrupt flag is not enabled until T<sub>IRVST</sub> has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval. Refer to Figure 21-2 or Figure 21-3.

**FIGURE 21-2: LOW-VOLTAGE DETECT OPERATION (VDIRMAG = 0)**



# PIC18F6390/6490/8390/8490

**FIGURE 21-3: HIGH-VOLTAGE DETECT OPERATION (VDIRMAG = 1)**

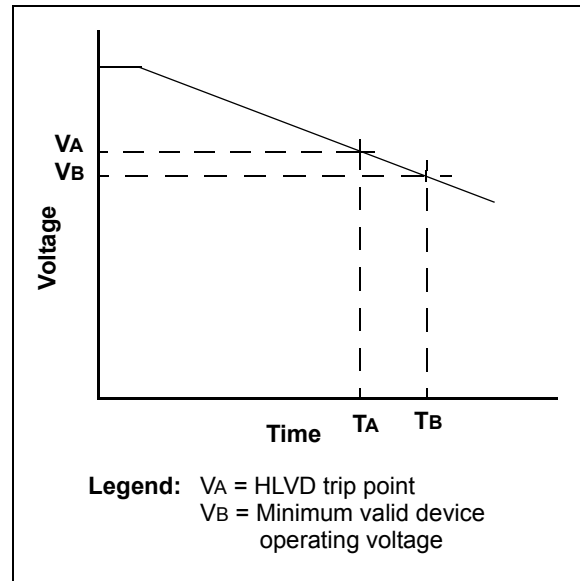


## 21.5 Applications

In many applications, the ability to detect a drop below, or rise above a particular threshold, is desirable. For example, the HLVD module could be periodically enabled to detect USB attach or detach. This assumes the device is powered by a lower voltage source than the Universal Serial Bus when detached. An attach would indicate a high-voltage detect from, for example, 3.3V to 5V (the voltage on USB) and vice versa for a detach. This feature could save a design a few extra components and an attach signal (input pin).

For general battery applications, Figure 21-4 shows a possible voltage curve. Over time, the device voltage decreases. When the device voltage reaches voltage, V<sub>A</sub>, the HLVD logic generates an interrupt at time, T<sub>A</sub>. The interrupt could cause the execution of an ISR, which would allow the application to perform “house-keeping tasks” and perform a controlled shutdown before the device voltage exits the valid operating range at T<sub>B</sub>. The HLVD, thus, would give the application a time window, represented by the difference between T<sub>A</sub> and T<sub>B</sub>, to safely exit.

**FIGURE 21-4: TYPICAL LOW-VOLTAGE DETECT APPLICATION**





# PIC18F6390/6490/8390/8490

## 21.6 Operation During Sleep

When enabled, the HLVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the HLVDIF 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.

## 21.7 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the HLVD module to be turned off.

**TABLE 21-1: 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	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the HLVD module.

# PIC18F6390/6490/8390/8490

---

NOTES:

## 22.0 LIQUID CRYSTAL DISPLAY (LCD) DRIVER MODULE

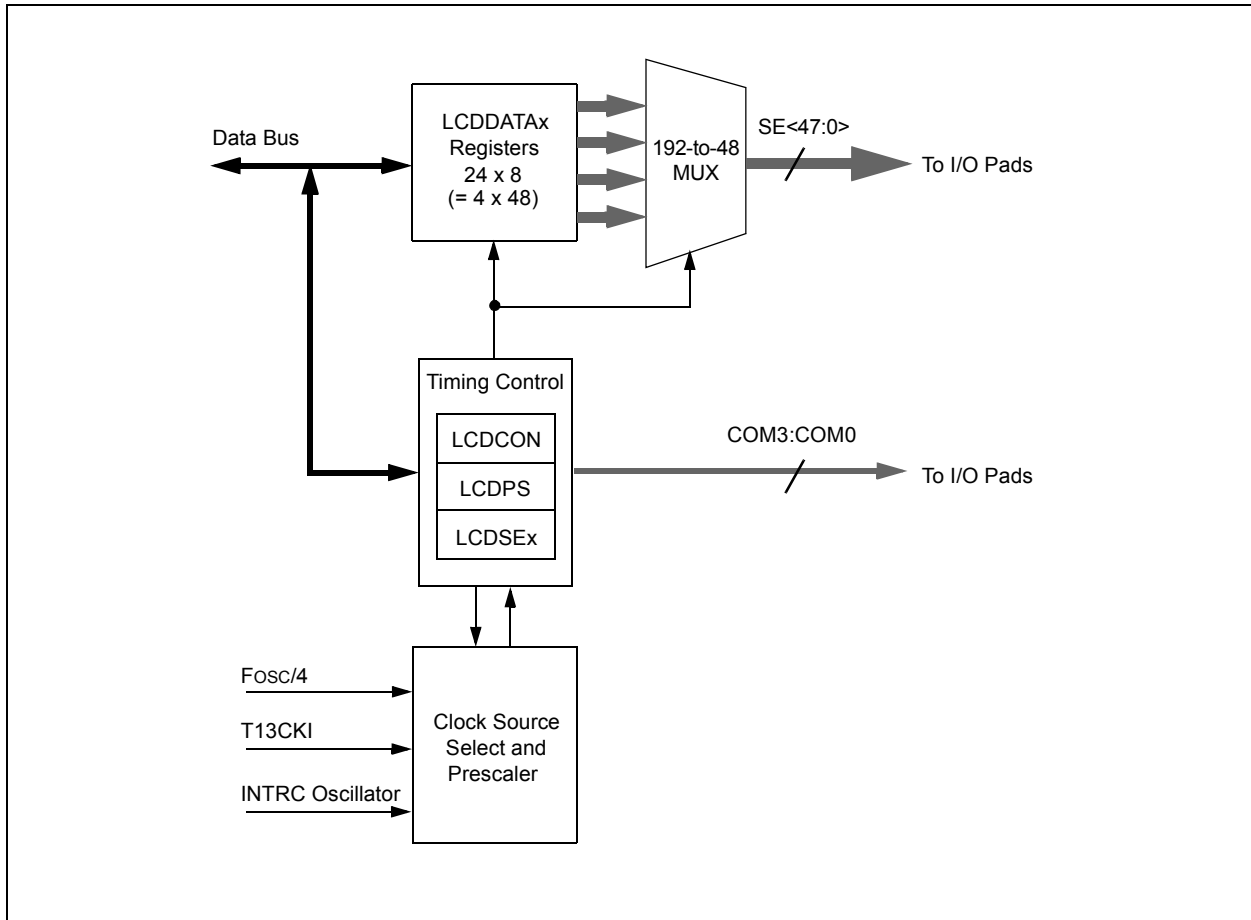
The Liquid Crystal Display (LCD) driver module generates the timing control to drive a static or multiplexed LCD panel. In the 80-pin devices (PIC18F8390/8490), the module drives the panels of up to four commons and up to 48 segments and in the 64-pin devices (PIC18F6390/6490), the module drives the panels of up to four commons and up to 32 segments. It also provides control of the LCD pixel data.

The LCD driver module supports:

- Direct driving of LCD panel
- Three LCD clock sources with selectable prescaler
- Up to four commons:
  - Static
  - 1/2 multiplex
  - 1/3 multiplex
  - 1/4 multiplex
- Up to 48 (in 80-pin devices)/32 (in 64-pin devices) segments
- Static, 1/2 or 1/3 LCD bias

A simplified block diagram of the module is shown in Figure 22-1.

**FIGURE 22-1: LCD DRIVER MODULE BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

## 22.1 LCD Registers

The LCD driver module has 32 registers:

- LCD Control Register (LCDCON)
- LCD Phase Register (LCDPS)
- Six LCD Segment Enable Registers (LCDSE5:LCDSE0)
- 24 LCD Data Registers (LCDDATA23:LCDDATA0)

The LCDCON register, shown in Register 22-1, controls the overall operation of the module. Once the module is configured, the LCDEN (LCDCON<7>) bit is

used to enable or disable the LCD module. The LCD panel can also operate during Sleep by clearing the SLPEN (LCDCON<6>) bit.

The LCDPS register, shown in Register 22-2, configures the LCD clock source prescaler and the type of waveform, Type-A or Type-B. Details on these features are provided in **Section 22.2 “LCD Clock Source Selection”**, **Section 22.3 “LCD Bias Types”** and **Section 22.8 “LCD Waveform Generation”**.

### REGISTER 22-1: LCDCON: LCD CONTROL REGISTER

R/W-0	R/W-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0
bit 7							bit 0

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

- bit 7      **LCDEN:** LCD Driver Enable bit  
 1 = LCD driver module is enabled  
 0 = LCD driver module is disabled
- bit 6      **SLPEN:** LCD Driver Enable in Sleep mode bit  
 1 = LCD driver module is disabled in Sleep mode  
 0 = LCD driver module is enabled in Sleep mode
- bit 5      **WERR:** LCD Write Failed Error bit  
 1 = LCDDATAx register written while LCDPS<WA> = 0 (must be cleared in software)  
 0 = No LCD write error
- bit 4      **Unimplemented:** Read as '0'
- bit 3-2    **CS1:CS0:** Clock Source Select bits  
 00 = (Fosc/4)/8192  
 01 = T13CKI (Timer1)/32  
 1x = INTRC (31.25 kHz)/32
- bit 1-0    **LMUX1:LMUX0:** Commons Select bits

LMUX1:LMUX0	Multiplex	Maximum Number of Pixels (PIC18F6X90)	Maximum Number of Pixels (PIC18F8X90)	Bias
00	Static (COM0)	32	48	Static
01	1/2 (COM1:COM0)	64	96	1/2 or 1/3
10	1/3 (COM2:COM0)	96	144	1/2 or 1/3
11	1/4 (COM3:COM0)	128	192	1/3

# PIC18F6390/6490/8390/8490

## REGISTER 22-2: LCDPS: LCD PHASE REGISTER

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0
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            **WFT:** Waveform Type Select bit  
                   1 = Type-B waveform (phase changes on each frame boundary)  
                   0 = Type-A waveform (phase changes within each common type)
- bit 6            **BIASMD:** Bias Mode Select bit  
                   When LMUX1:LMUX0 = 00:  
                   0 = Static Bias mode (do not set this bit to '1')  
                   When LMUX1:LMUX0 = 01:  
                   1 = 1/2 Bias mode  
                   0 = 1/3 Bias mode  
                   When LMUX1:LMUX0 = 10:  
                   1 = 1/2 Bias mode  
                   0 = 1/3 Bias mode  
                   When LMUX1:LMUX0 = 11:  
                   0 = 1/3 Bias mode (do not set this bit to '1')
- bit 5            **LCDA:** LCD Active Status bit  
                   1 = LCD driver module is active  
                   0 = LCD driver module is inactive
- bit 4            **WA:** LCD Write Allow Status bit  
                   1 = Write into the LCDDATAx registers is allowed  
                   0 = Write into the LCDDATAx registers is not allowed
- bit 3-0        **LP3:LP0:** LCD Prescaler Select bits  
                   1111 = 1:16  
                   1110 = 1:15  
                   1101 = 1:14  
                   1100 = 1:13  
                   1011 = 1:12  
                   1010 = 1:11  
                   1001 = 1:10  
                   1000 = 1:9  
                   0111 = 1:8  
                   0110 = 1:7  
                   0101 = 1:6  
                   0100 = 1:5  
                   0011 = 1:4  
                   0010 = 1:3  
                   0001 = 1:2  
                   0000 = 1:1

# PIC18F6390/6490/8390/8490

The LCDSE5:LCDSE0 registers configure the functions of the port pins. Setting the segment enable bit for a particular segment configures that pin as an LCD driver. There are six LCD Segment Enable registers listed in Table 22-1. The prototype LCDSE<sub>x</sub> register is shown in Register 22-3.

**TABLE 22-1: LCDSE REGISTERS AND ASSOCIATED SEGMENTS**

Register	Segments
LCDSE0	7:0
LCDSE1	15:8
LCDSE2	23:16
LCDSE3	31:24
LCDSE4	39:32
LCDSE5	47:40

**Note:** The LCDSE5:LCDSE4 registers are not implemented in PIC18F6X90 devices.

Once the module is initialized for the LCD panel, the individual bits of the LCDDATA23:LCDDATA0 registers are cleared or set to represent a clear or dark pixel, respectively. Specific sets of LCDDATA registers are used with specific segments and common signals. Each bit represents a unique combination of a specific segment connected to a specific common. Individual LCDDATA bits are named by the convention “SxxCy”, with “xx” as the segment number and “y” as the common number. The relationship is summarized in Table 22-2. The prototype LCDDATA<sub>x</sub> register is shown in Register 22-4.

**Note:** Writing into the registers, LCDDATA4, LCDDATA5, LCDDATA10, LCDDATA11, LCDDATA16, LCDDATA17, LCDDATA22 and LCDDATA23, in PIC18F6X90 devices will not affect the status of any pixel and these registers can be used as General Purpose Registers.

**REGISTER 22-3: LCDSE<sub>x</sub>: LCD SEGMENT<sub>x</sub> ENABLE REGISTER**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SE(n + 7)	SE(n + 6)	SE(n + 5)	SE(n + 4)	SE(n + 3)	SE(n + 2)	SE(n + 1)	SE(n)
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                      **SE(n + 7):SE(n):** Segment Enable bits  
For LCDSE0: n = 0  
For LCDSE1: n = 8  
For LCDSE2: n = 16  
For LCDSE3: n = 24  
For LCDSE4: n = 32  
For LCDSE5: n = 40  
 1 = Segment function of the pin is enabled, digital I/O is disabled  
 0 = I/O function of the pin is enabled

# PIC18F6390/6490/8390/8490

**TABLE 22-2: LCDDATA REGISTERS AND BITS FOR SEGMENT AND COM COMBINATIONS**

Segments	COM Lines			
	0	1	2	3
0 through 7	LCDDATA0	LCDDATA6	LCDDATA12	LCDDATA18
	S00C0:S07C0	S00C1:S07C1	S00C2:S07C2	S00C3:S07C3
8 through 15	LCDDATA1	LCDDATA7	LCDDATA13	LCDDATA19
	S08C0:S15C0	S08C1:S15C1	S08C2:S15C2	S08C3:S15C3
16 through 23	LCDDATA2	LCDDATA8	LCDDATA14	LCDDATA20
	S16C0:S23C0	S16C1:S23C1	S16C2:S23C2	S16C3:S23C3
24 through 31	LCDDATA3	LCDDATA9	LCDDATA15	LCDDATA21
	S24C0:S31C0	S24C1:S31C1	S24C2:S31C2	S24C3:S31C3
32 through 39	LCDDATA4 <sup>(1)</sup>	LCDDATA10 <sup>(1)</sup>	LCDDATA16 <sup>(1)</sup>	LCDDATA22 <sup>(1)</sup>
	S32C0:S39C0	S32C1:S39C1	S32C2:S39C2	S32C3:S39C3
40 through 47	LCDDATA5 <sup>(1)</sup>	LCDDATA11 <sup>(1)</sup>	LCDDATA17 <sup>(1)</sup>	LCDDATA23 <sup>(1)</sup>
	S40C0:S47C0	S40C1:S47C1	S40C2:S47C2	S40C3:S47C3

**Note 1:** These registers are implemented but not used as LCD data registers in 64-pin devices. They may be used as general purpose data memory.

**REGISTER 22-4: LCDDATAx: LCD DATAx REGISTER**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
S(n + 7)Cy	S(n + 6)Cy	S(n + 5)Cy	S(n + 4)Cy	S(n + 3)Cy	S(n + 2)Cy	S(n + 1)Cy	S(n)Cy
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

**S(n + 7)Cy:S(n)Cy:** Pixel On bits

For LCDDATA0 through LCDDATA5: n = (8x), y = 0

For LCDDATA6 through LCDDATA11: n = (8(x - 6)), y = 1

For LCDDATA12 through LCDDATA17: n = (8(x - 12)), y = 2

For LCDDATA18 through LCDDATA23: n = (8(x - 18)), y = 3

1 = Pixel on (dark)

0 = Pixel off (clear)

# PIC18F6390/6490/8390/8490

## 22.2 LCD Clock Source Selection

The LCD driver module has 3 possible clock sources:

- $(F_{osc}/4)/8192$
- T13CKI Clock/32
- INTRC/32

The first clock source is the system clock divided by 8192 ( $(F_{osc}/4)/8192$ ). This divider ratio is chosen to provide about 1 kHz output when the system clock is 8 MHz. The divider is not programmable. Instead, the LCD prescaler bits, LCDPS<3:0>, are used to set the LCD frame clock rate.

The second clock source is the Timer1 oscillator/32. This also gives about 1 kHz when a 32.768 kHz crystal is used with the Timer1 oscillator. To use the Timer1 oscillator as a clock source, the T1OSCEN (T1CON<3>) bit should be set.

The third clock source is a 31.25 kHz internal RC oscillator/32, which provides approximately 1 kHz output.

The second and third clock sources may be used to continue running the LCD while the processor is in Sleep.

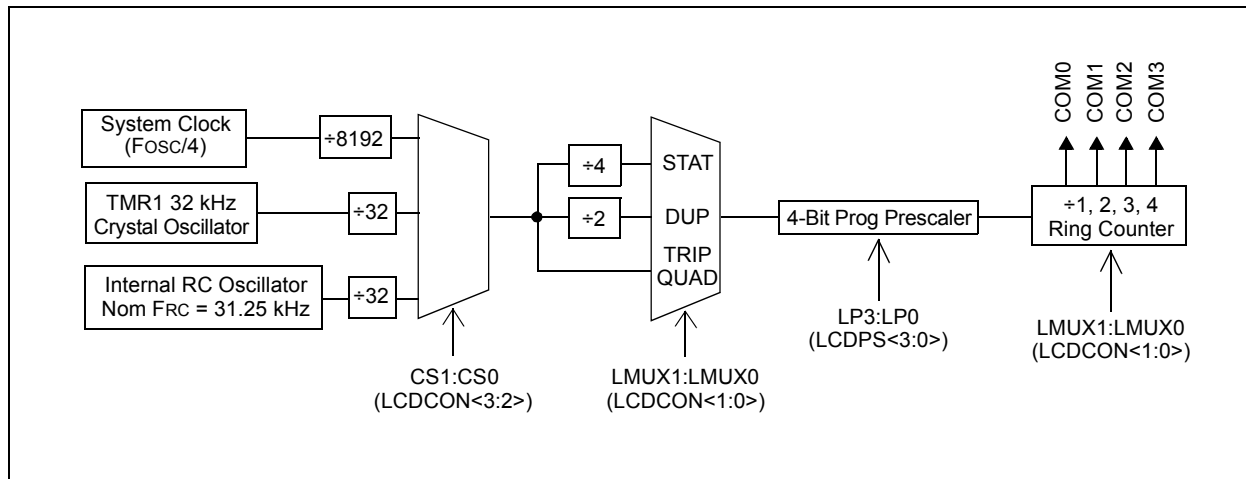
Using the bits, CS1:CS0 (LCDCON<3:2>), any of these clock sources can be selected.

### 22.2.1 LCD PRESCALER

A 16-bit counter is available as a prescaler for the LCD clock. The prescaler is not directly readable or writable; its value is set by the LP3:LP0 bits (LCDPS<3:0>), which determine the prescaler assignment and prescale ratio.

The prescale values from 1:1 through 1:32768 in power-of-2 increments are selectable.

FIGURE 22-2: LCD CLOCK GENERATION





# PIC18F6390/6490/8390/8490

## 22.3 LCD Bias Types

The LCD driver module can be configured into three bias types:

- Static Bias (2 voltage levels: AVSS and AVDD)
- 1/2 Bias (3 voltage levels: AVSS, 1/2 AVDD and AVDD)
- 1/3 Bias (4 voltage levels: AVSS, 1/3 AVDD, 2/3 AVDD and AVDD)

This module uses an external resistor ladder to generate the LCD bias voltages.

The external resistor ladder should be connected to the Bias 1 pin, Bias 2 pin, Bias 3 pin and VSS. The Bias 3 pin should also be connected to AVDD.

Figure 22-3 shows the proper way to connect the resistor ladder to the Bias pins.

## 22.4 LCD Multiplex Types

The LCD driver module can be configured into four multiplex types:

- Static (only COM0 used)
- 1/2 multiplex (COM0 and COM1 are used)
- 1/3 multiplex (COM0, COM1 and COM2 are used)
- 1/4 multiplex (all COM0, COM1, COM2 and COM3 are used)

The LMUX1:LMUX0 setting decides the function of the PORTE<6:4> bits (see Table 22-3 for details).

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. If the pin is a COM drive, then the TRIS setting of that pin is overridden.

**Note:** On a Power-on Reset, the LMUX1:LMUX0 bits are '00'.

**TABLE 22-3: PORTE<6:4> FUNCTION**

LMUX1:LMUX0	PORTE<6>	PORTE<5>	PORTE<4>
00	Digital I/O	Digital I/O	Digital I/O
01	Digital I/O	Digital I/O	COM1 Driver
10	Digital I/O	COM2 Driver	COM1 Driver
11	COM3 Driver	COM2 Driver	COM1 Driver

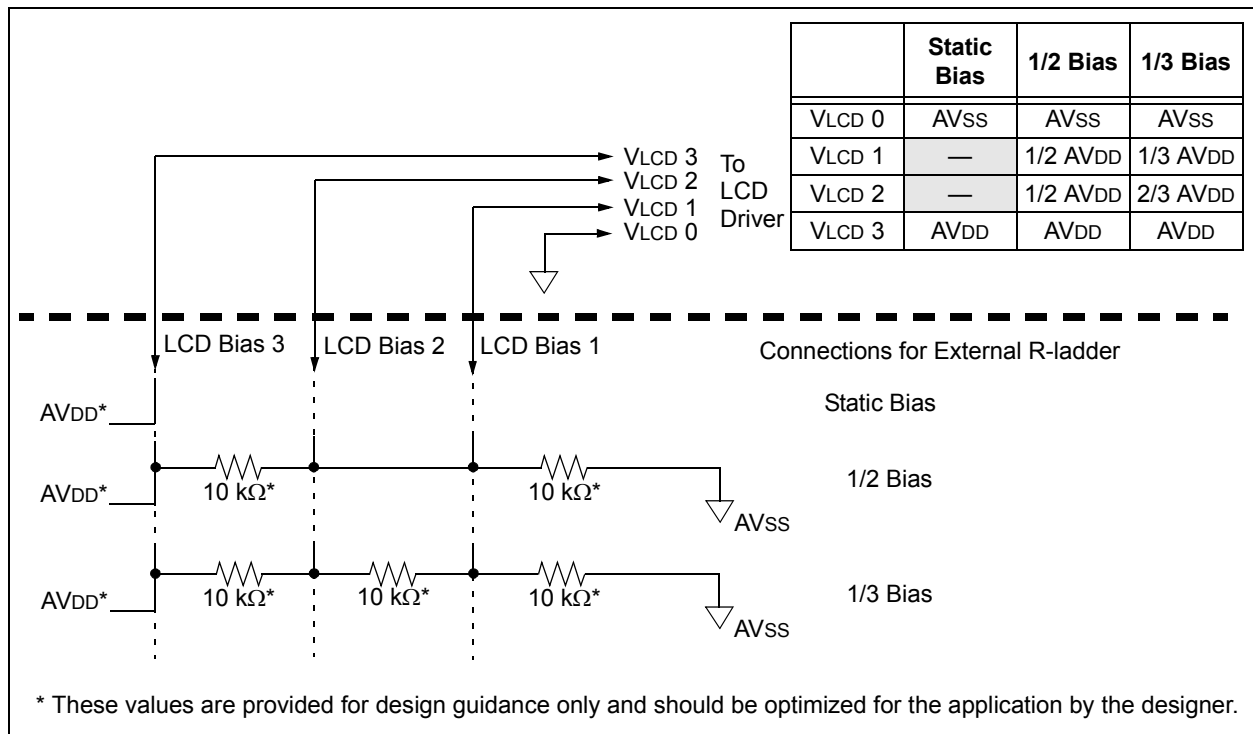
## 22.5 Segment Enables

The LCDSEx registers are used to select the pin function for each segment pin. The selection allows each pin to operate as either an LCD segment driver or a digital only pin. To configure the pin as a segment pin, the corresponding bits in the LCDSEx registers must be set to '1'.

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. Any bit set in the LCDSEx registers overrides any bit settings in the corresponding TRIS register.

**Note:** On a Power-on Reset, these pins are configured as digital I/O.

**FIGURE 22-3: LCD BIAS RESISTOR LADDER CONNECTION DIAGRAM**



# PIC18F6390/6490/8390/8490

## 22.6 Pixel Control

The LCDDATAx registers contain bits which define the state of each pixel. Each bit defines one unique pixel.

Table 22-2 shows the correlation of each bit in the LCDDATAx registers to the respective common and segment signals.

Any LCD pixel location not being used for display can be used as general purpose RAM.

## 22.7 LCD Frame Frequency

The rate at which the COM and SEG outputs changes is called the LCD frame frequency

**TABLE 22-4: FRAME FREQUENCY FORMULAS**

Multiplex	Frame Frequency =
Static	$\text{Clock Source}/(4 \times 1 \times (\text{LP3:LP0} + 1))$
1/2	$\text{Clock Source}/(2 \times 2 \times (\text{LP3:LP0} + 1))$
1/3	$\text{Clock Source}/(1 \times 3 \times (\text{LP3:LP0} + 1))$
1/4	$\text{Clock Source}/(1 \times 4 \times (\text{LP3:LP0} + 1))$

**Note:** Clock source is  $(F_{\text{osc}}/4)/8192$ ,  
Timer1 Osc/32 or INTRC/32.

**TABLE 22-5: APPROXIMATE FRAME FREQUENCY (IN Hz) USING  $F_{\text{osc}}$  @ 32 MHz, TIMER1 @ 32.768 kHz OR INTRC OSCILLATOR**

LP3:LP0	Static	1/2	1/3	1/4
1	125	125	167	125
2	83	83	111	83
3	62	62	83	62
4	50	50	67	50
5	42	42	56	42
6	36	36	48	36
7	31	31	42	31

## 22.8 LCD Waveform Generation

LCD waveform generation is based on the philosophy that the net AC voltage across the dark pixel should be maximized and the net AC voltage across the clear pixel should be minimized. The net DC voltage across any pixel should be zero.

The COM signal represents the time slice for each common, while the SEG contains the pixel data.

The pixel signal (COM-SEG) will have no DC component and it can take only one of the two rms values. The higher rms value will create a dark pixel and a lower rms value will create a clear pixel.

As the number of commons increases, the delta between the two rms values decreases. The delta represents the maximum contrast that the display can have.

The LCDs can be driven by two types of waveform: Type-A and Type-B. In Type-A waveform, the phase changes within each common type, whereas in Type-B waveform, the phase changes on each frame boundary. Thus, Type-A waveform maintains 0 V<sub>DC</sub> over a single frame, whereas Type-B waveform takes two frames.

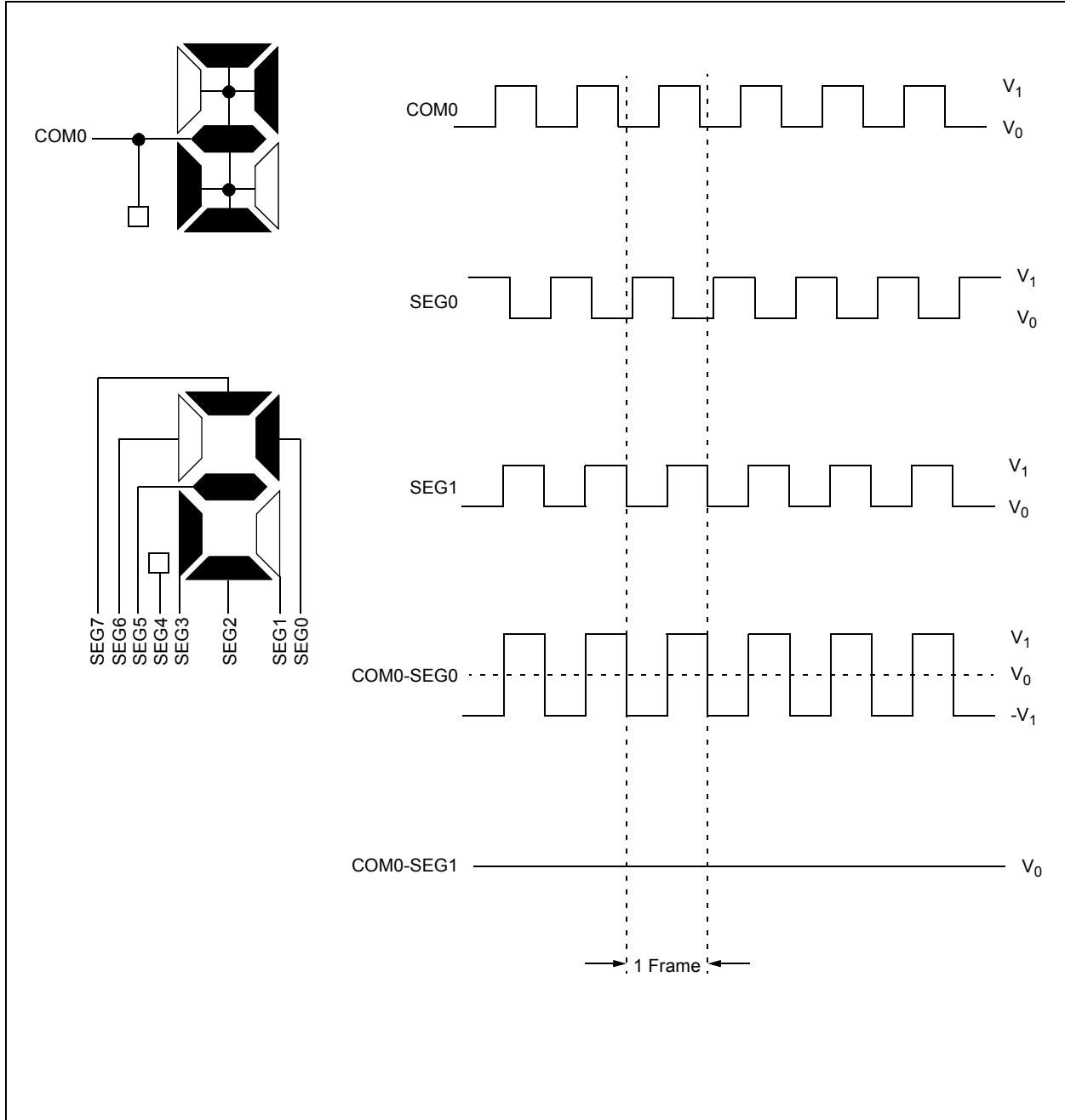
**Note 1:** If Sleep has to be executed with LCD Sleep enabled (LCDCON<SLPEN> is '1'), then care must be taken to execute Sleep only when V<sub>DC</sub> on all the pixels is '0'.

**2:** When the LCD clock source is  $(F_{\text{osc}}/4)/8192$ , if Sleep is executed irrespective of the LCDCON<SLPEN> setting, the LCD goes into Sleep. Thus, take care to see that V<sub>DC</sub> on all pixels is '0' when Sleep is executed.

Figure 22-4 through Figure 22-14 provide waveforms for static, half-multiplex, one-third-multiplex and quarter-multiplex drives for Type-A and Type-B waveforms.

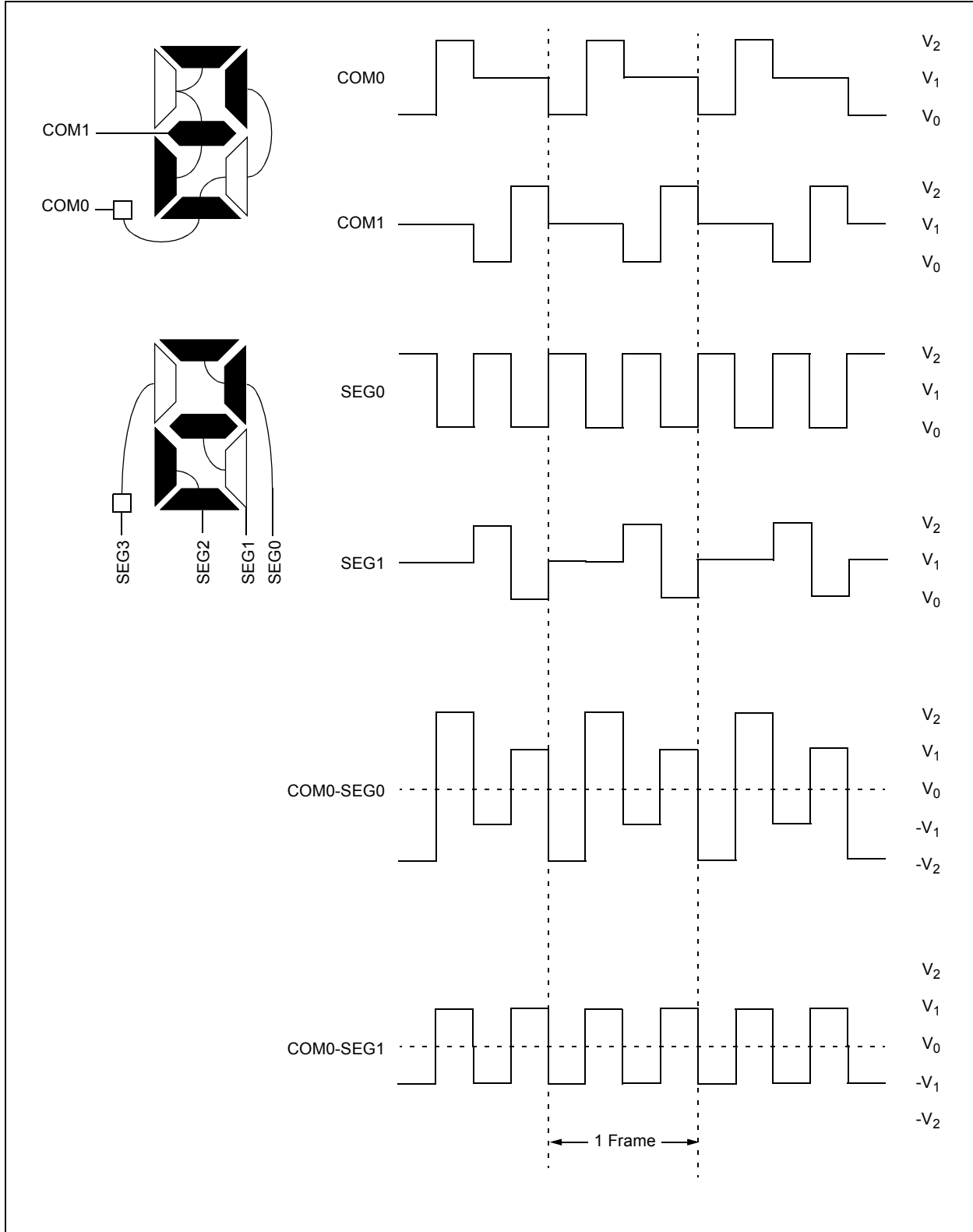
# PIC18F6390/6490/8390/8490

FIGURE 22-4: TYPE-A/TYPE-B WAVEFORMS IN STATIC DRIVE



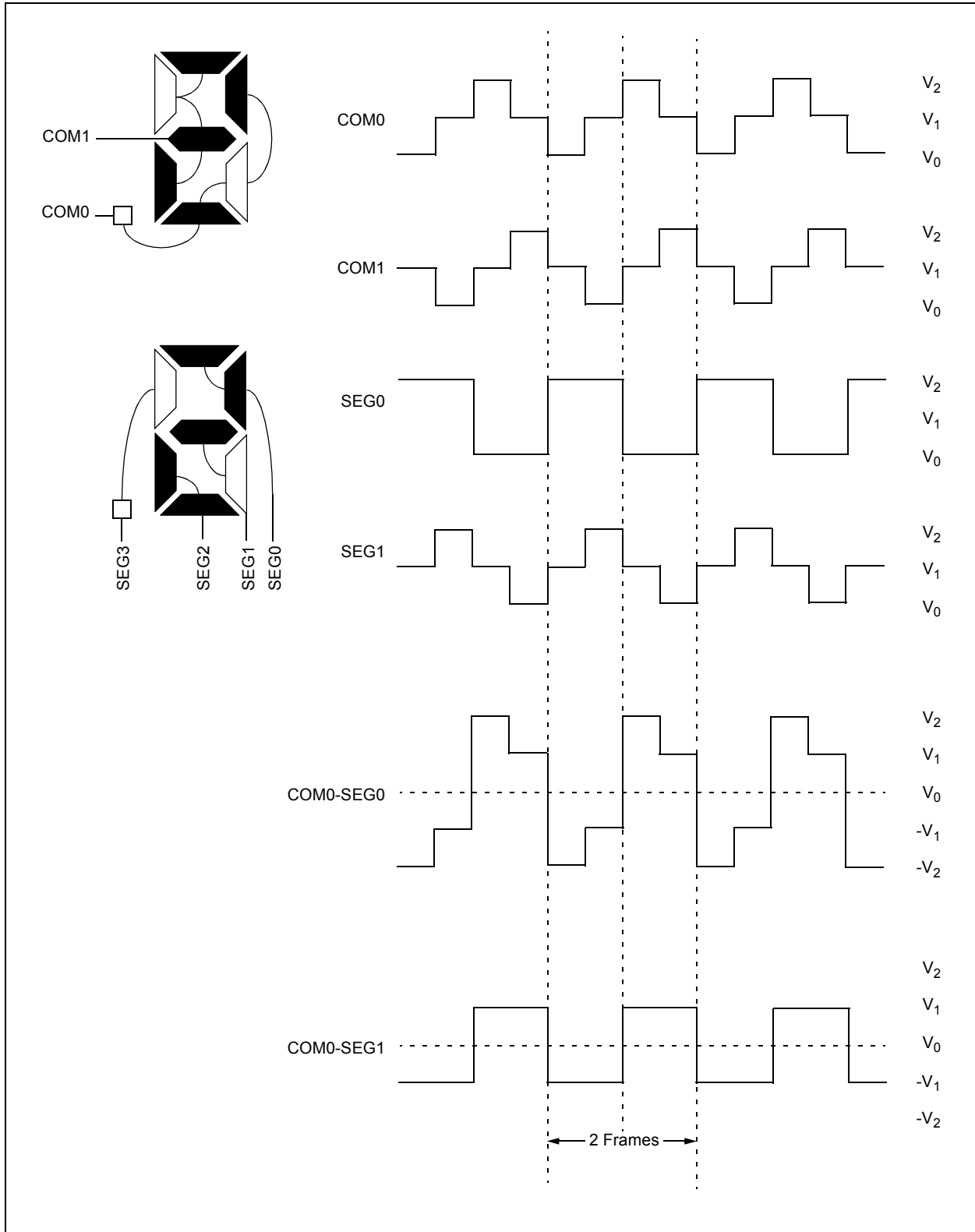
# PIC18F6390/6490/8390/8490

FIGURE 22-5: TYPE-A WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE



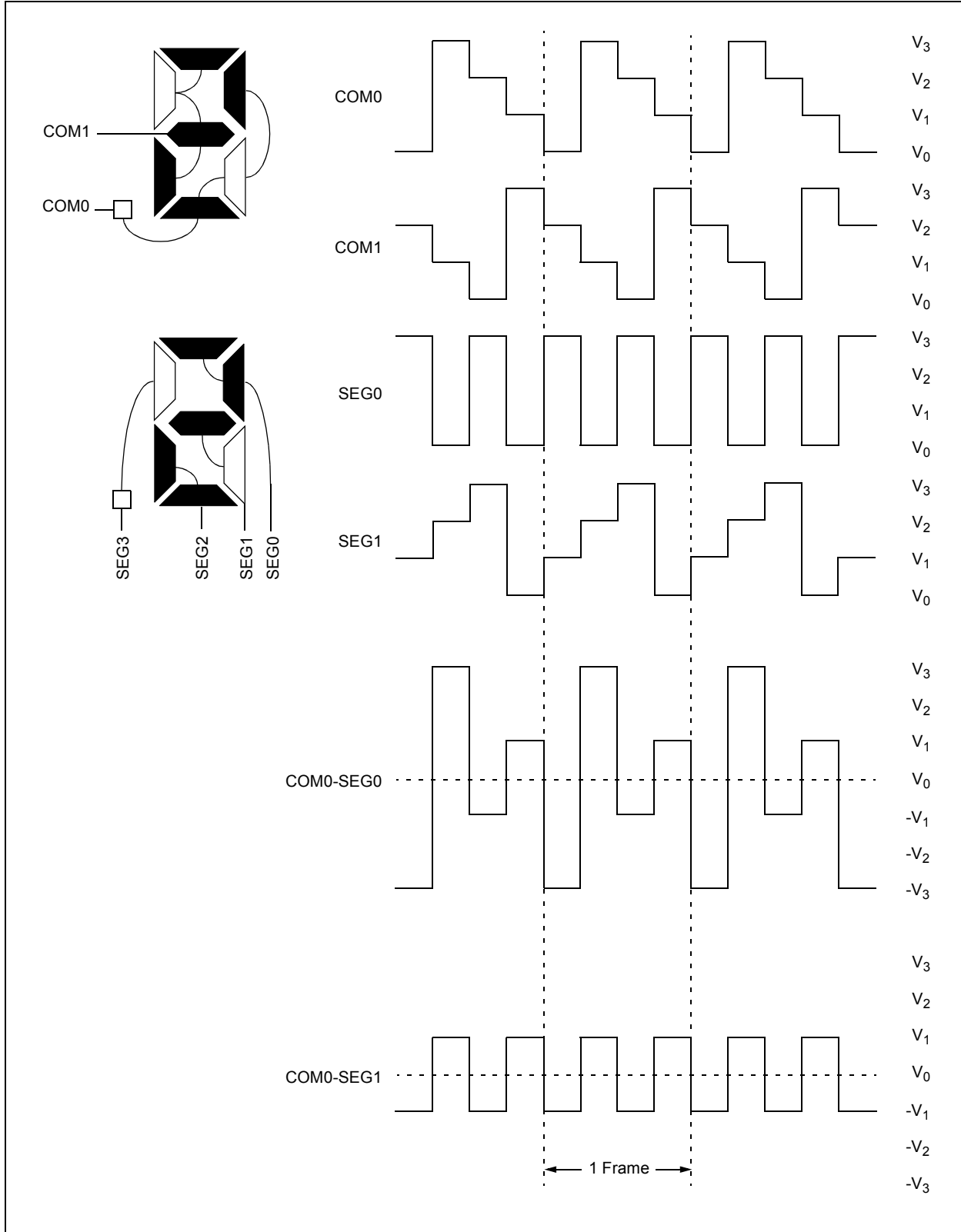
# PIC18F6390/6490/8390/8490

FIGURE 22-6: TYPE-B WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE

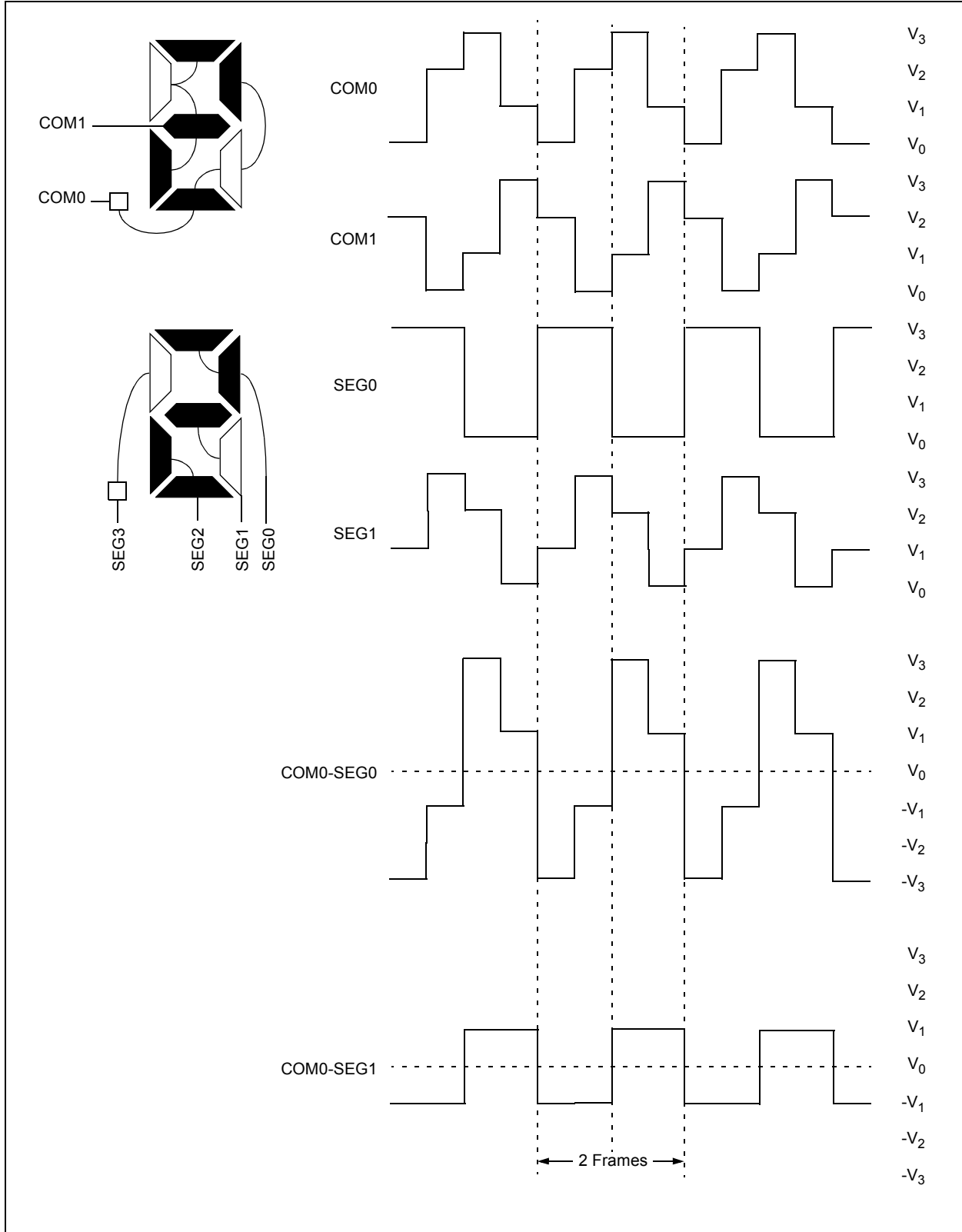


# PIC18F6390/6490/8390/8490

FIGURE 22-7: TYPE-A WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE

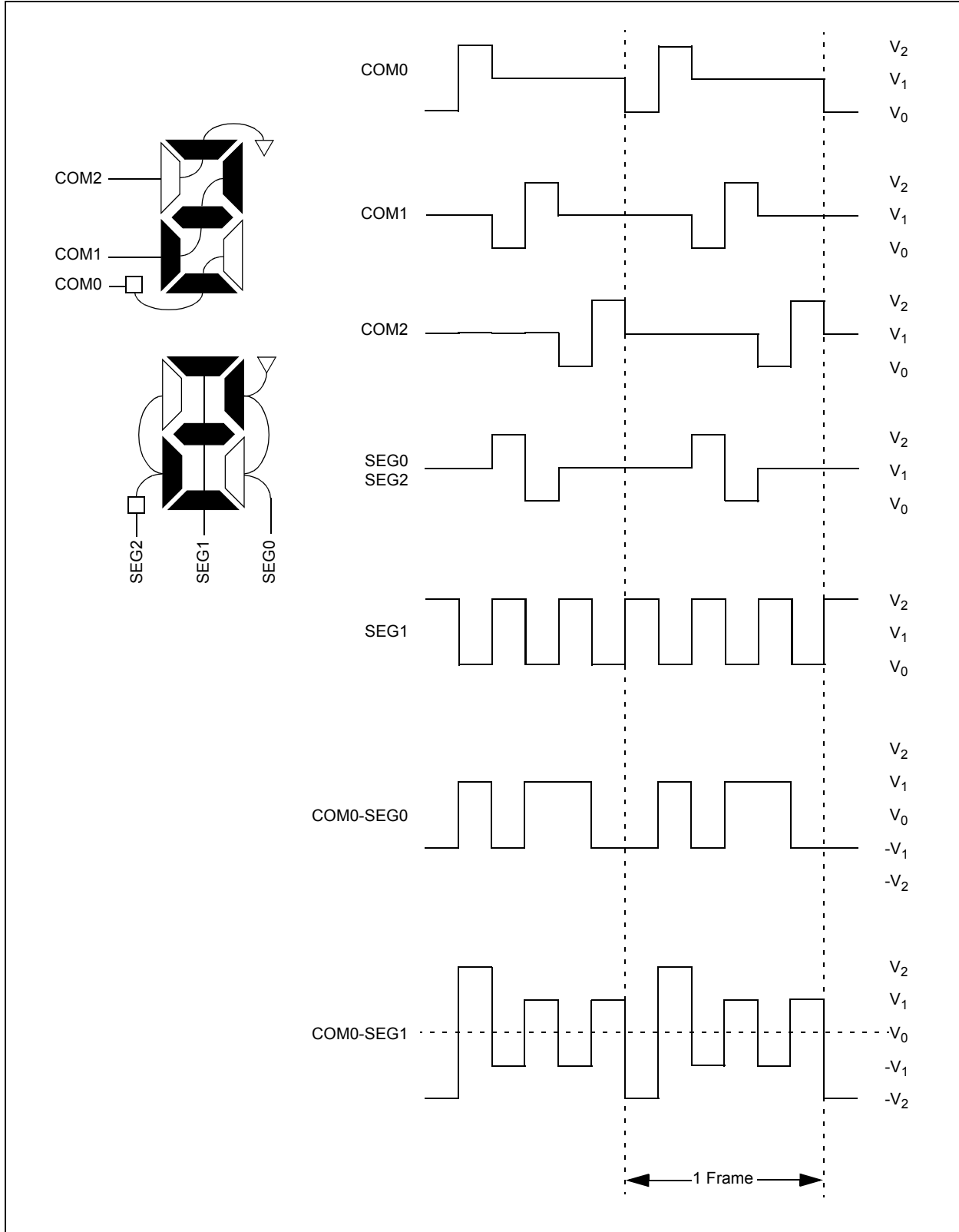


**FIGURE 22-8: TYPE-B WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE**



# PIC18F6390/6490/8390/8490

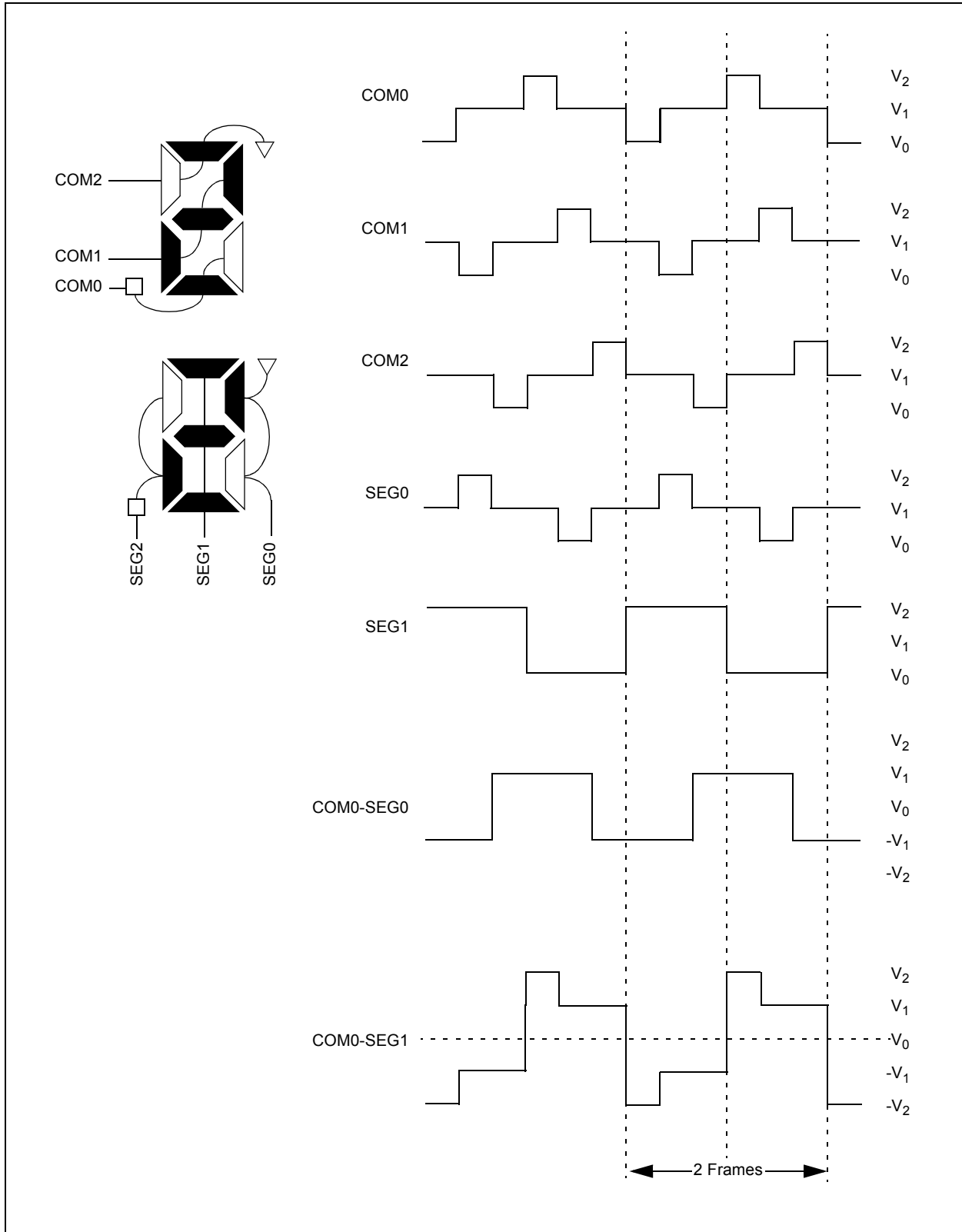
FIGURE 22-9: TYPE-A WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE





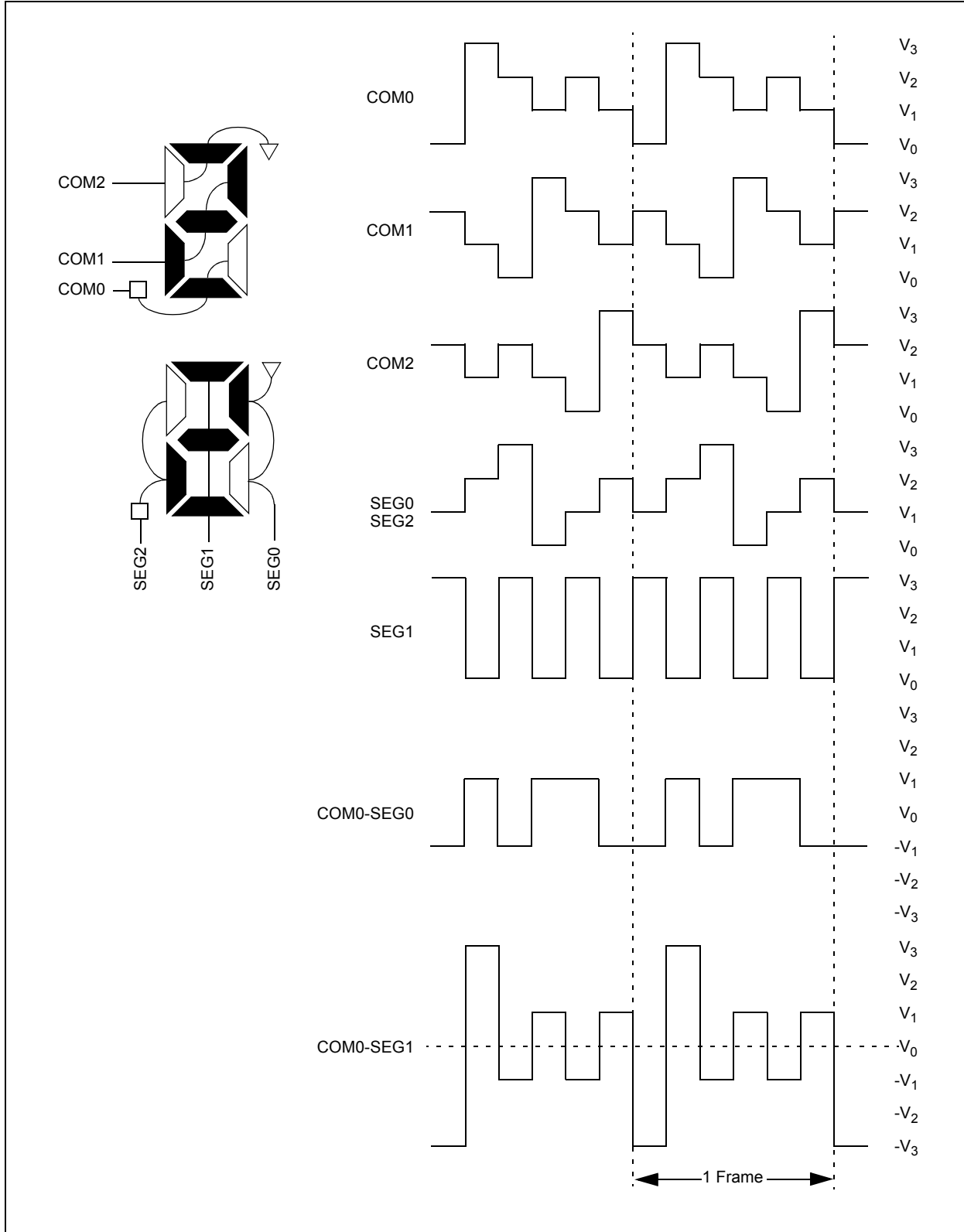
# PIC18F6390/6490/8390/8490

FIGURE 22-10: TYPE-B WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE



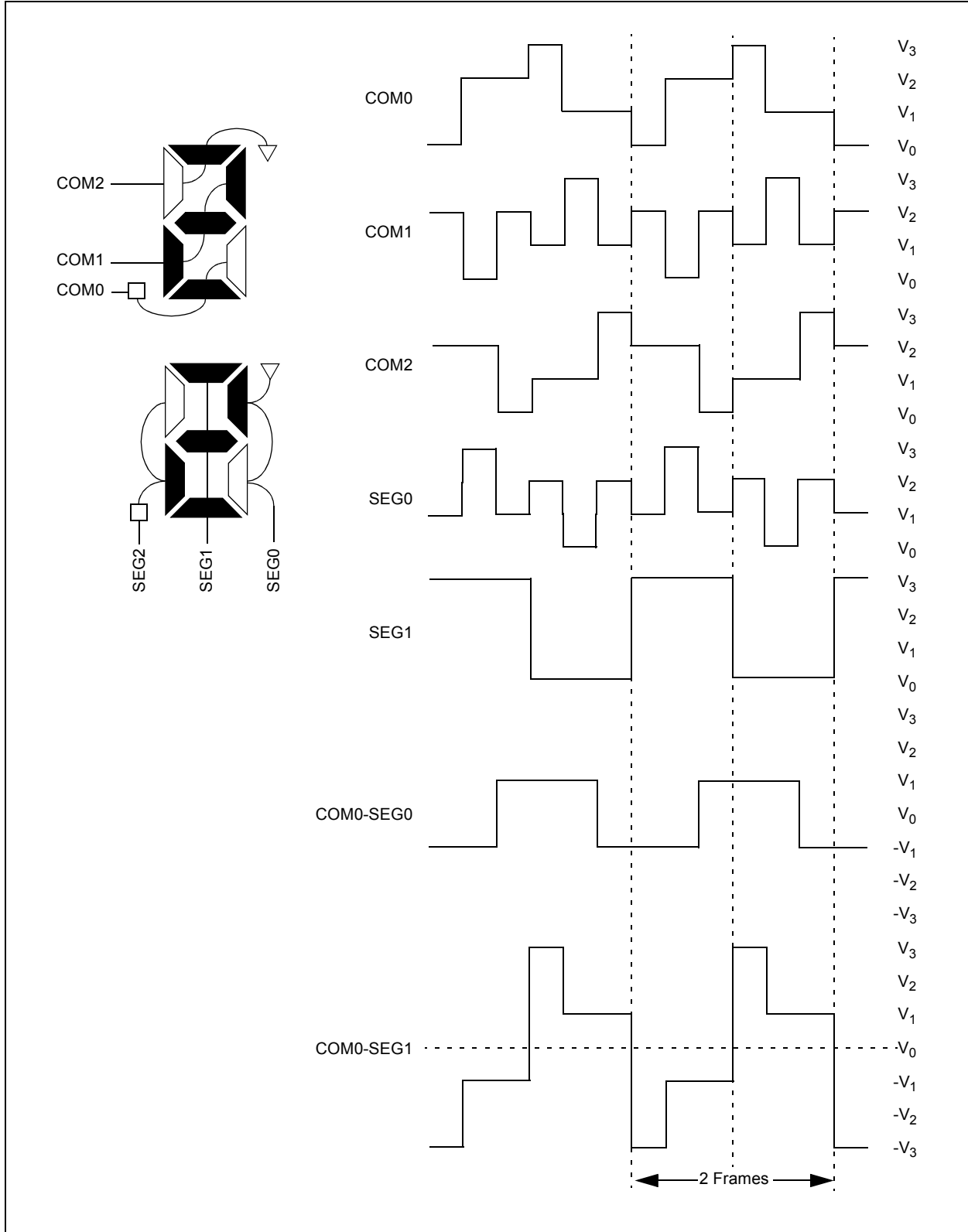
# PIC18F6390/6490/8390/8490

FIGURE 22-11: TYPE-A WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE



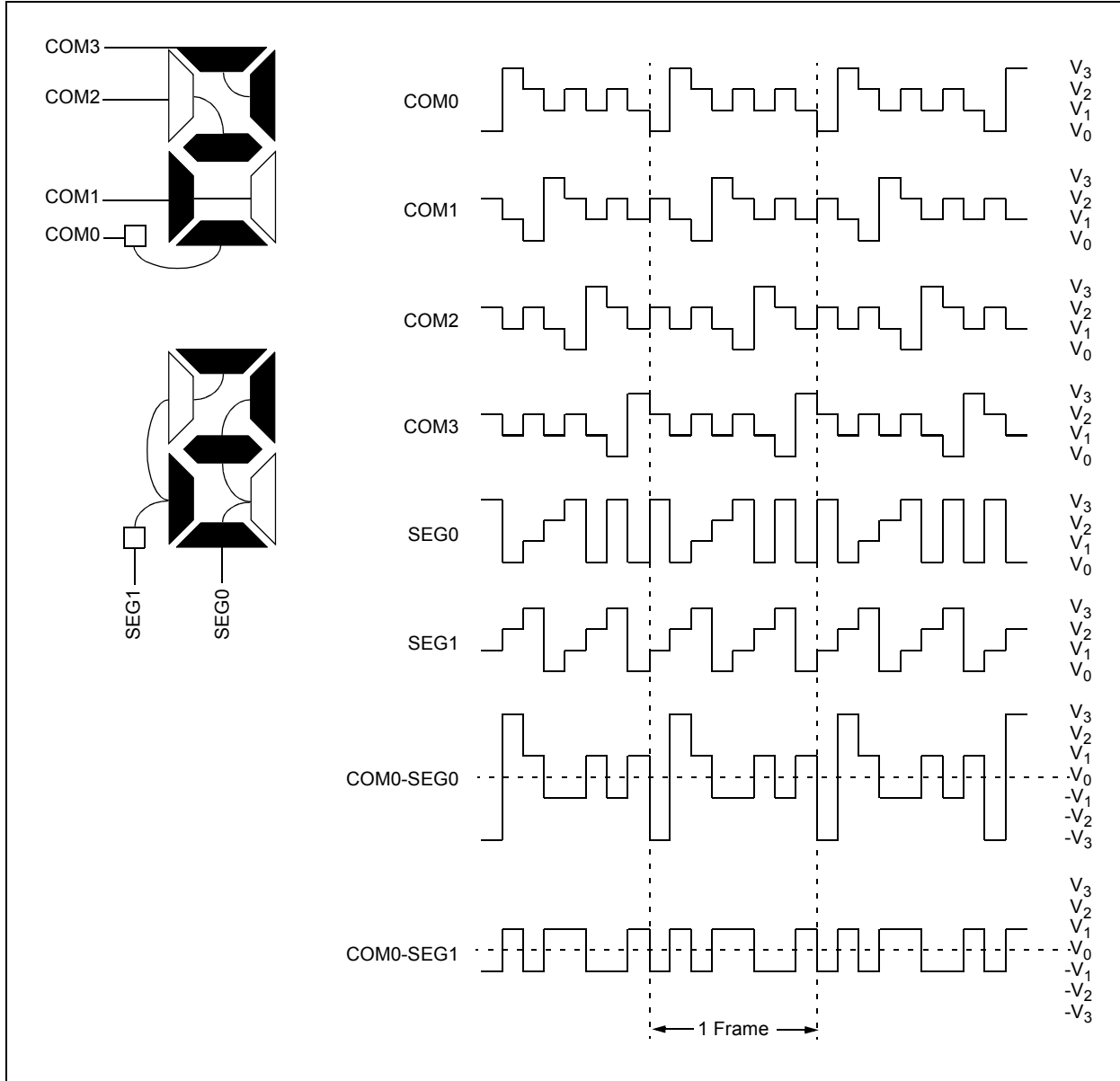
# PIC18F6390/6490/8390/8490

FIGURE 22-12: TYPE-B WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE



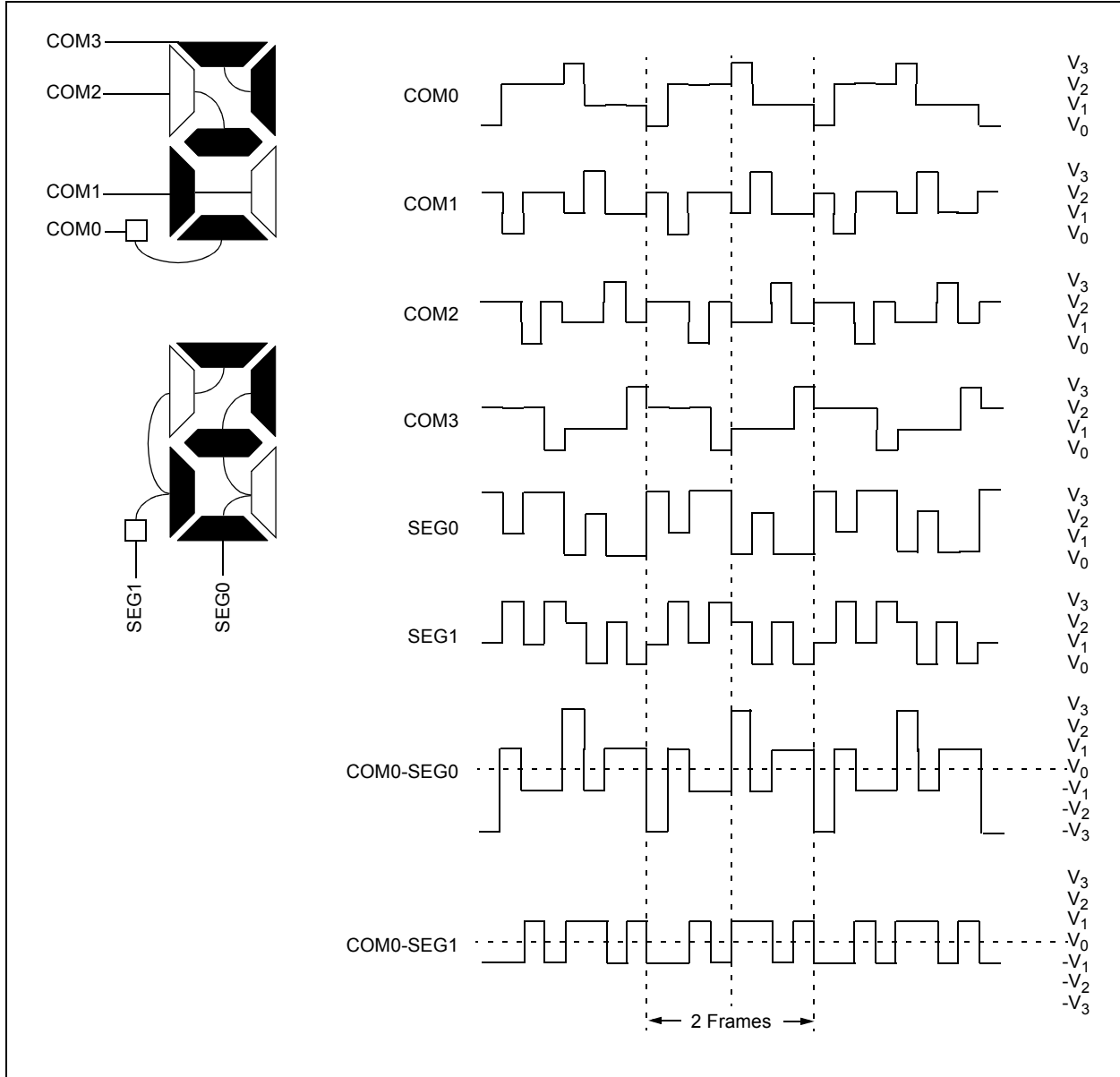
# PIC18F6390/6490/8390/8490

FIGURE 22-13: TYPE-A WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE



# PIC18F6390/6490/8390/8490

**FIGURE 22-14: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE**



# PIC18F6390/6490/8390/8490

## 22.9 LCD Interrupts

The LCD timing generation provides an interrupt that defines the LCD frame timing. This interrupt can be used to coordinate the writing of the pixel data with the start of a new frame. Writing pixel data at the frame boundary allows a visually crisp transition of the image. This interrupt can also be used to synchronize external events to the LCD. For example, the interface to an external segment driver can be synchronized for segment data update to the LCD frame.

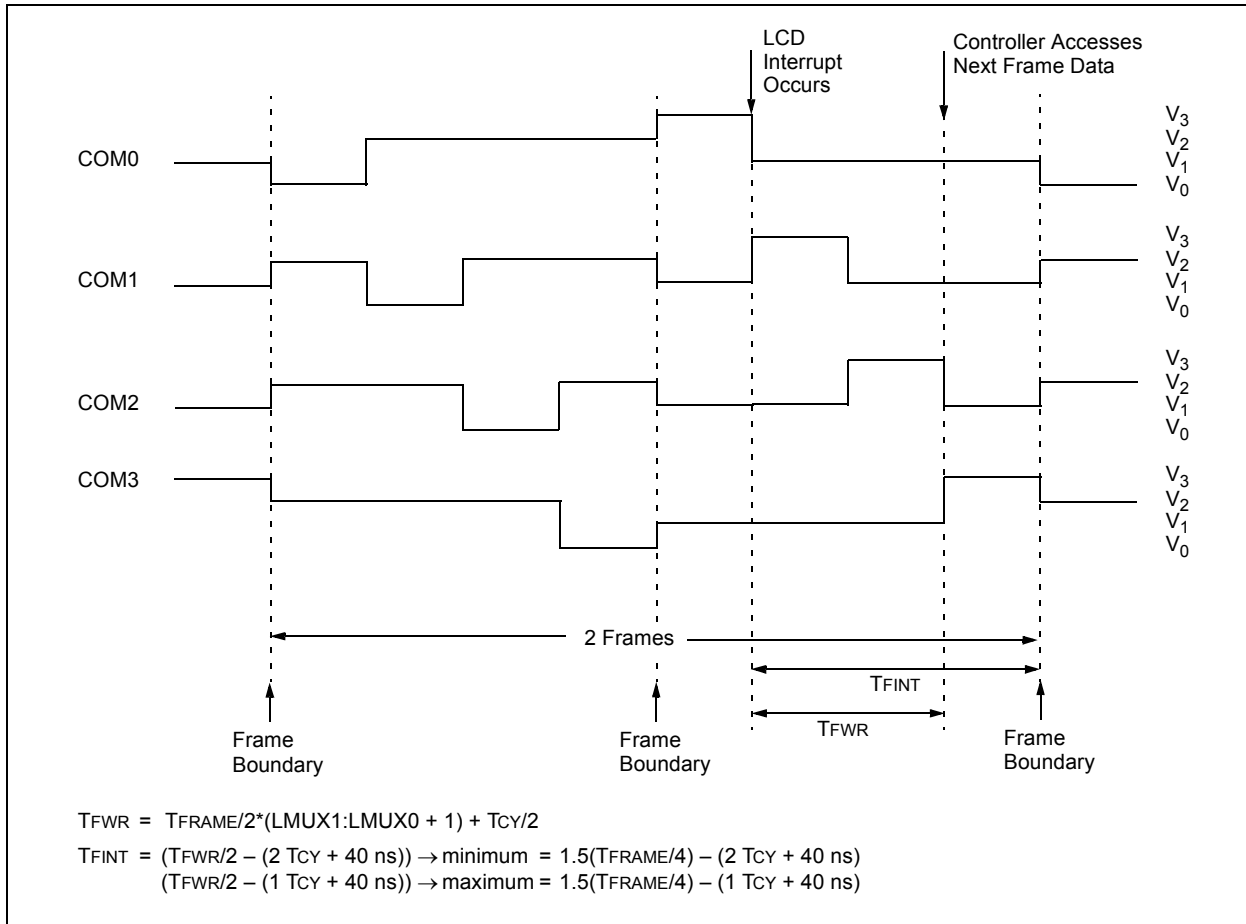
A new frame is defined to begin at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (TFINT), as shown in Figure 22-15. The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins to access data after the interrupt (TFWR). New data must be written within TFWR, as this is when the LCD controller will begin to access the data for the next frame.

When the LCD driver is running with Type-B waveforms and the LMUX1:LMUX0 bits are not equal to '00', there are some additional issues that must be addressed. Since the DC voltage on the pixel takes two frames to maintain zero volts, the pixel data must not change between subsequent frames. If the pixel data were allowed to change, the waveform for the odd frames would not necessarily be the complement of the waveform generated in the even frames and a DC component would be introduced into the panel. Therefore, when using Type-B waveforms, the user must synchronize the LCD pixel updates to occur within a subframe after the frame interrupt.

To correctly sequence writing while in Type-B, the interrupt will only occur on complete phase intervals. If the user attempts to write when the write is disabled, the WERR (LCDCON<5>) bit is set.

**Note:** The interrupt is not generated when the Type-A waveform is selected and when the Type-B with no multiplex (static) is selected.

**FIGURE 22-15: EXAMPLE WAVEFORMS AND INTERRUPT TIMING IN QUARTER-DUTY CYCLE DRIVE**



## 22.10 Operation During Sleep

The LCD module can operate during Sleep. The selection is controlled by bit, SLPEN (LCDCON<6>). Setting the SLPEN bit allows the LCD module to go to Sleep. Clearing the SLPEN bit allows the module to continue to operate during Sleep.

If a SLEEP instruction is executed and SLPEN = 1, the LCD module will cease all functions and go into a very low-current consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 22-16 shows this operation.

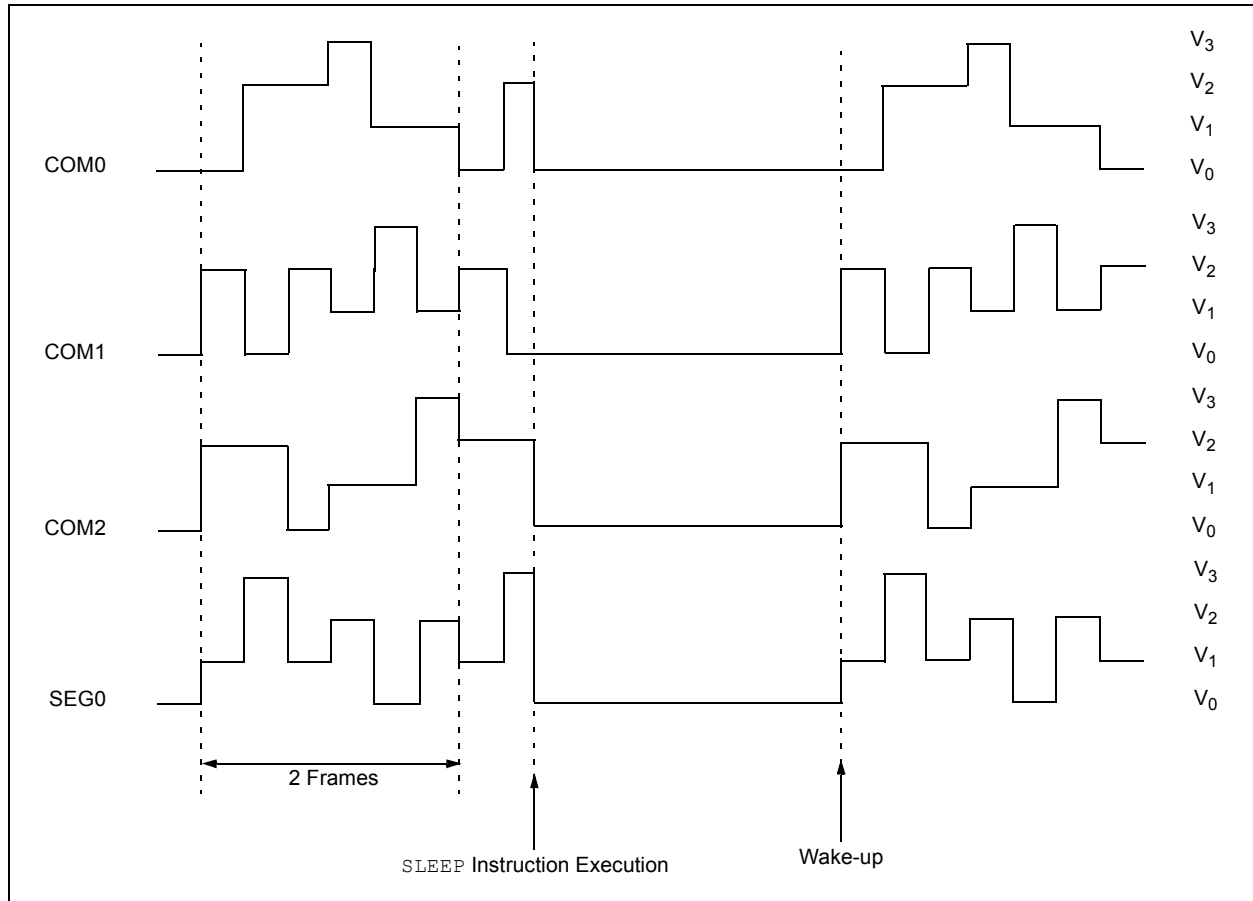
To ensure that no DC component is introduced on the panel, the SLEEP instruction should be executed immediately after a LCD frame boundary. The LCD interrupt can be used to determine the frame boundary. See Section 22.9 “LCD Interrupts” for the formulas to calculate the delay.

If a SLEEP instruction is executed and SLPEN = 0, the module will continue to display the current contents of the LCDDATA registers. To allow the module to continue operation while in Sleep, the clock source must be either the internal RC oscillator or Timer1 external oscillator. While in Sleep, the LCD data cannot be changed. The LCD module current consumption will not decrease in this mode, however, the overall consumption of the device will be lower due to shut down of the core and other peripheral functions.

If the system clock is selected and the module is programmed to not Sleep, the module will ignore the SLPEN bit and stop operation immediately. The minimum LCD voltage will then be driven onto the segments and commons.

**Note:** The internal RC oscillator or external Timer1 oscillator must be used to operate the LCD module during Sleep.

**FIGURE 22-16: SLEEP ENTRY/EXIT WHEN SLPEN = 1 OR CS1:CS0 = 00**



# PIC18F6390/6490/8390/8490

---

## 22.11 Configuring the LCD Module

The following is the sequence of steps to configure the LCD module.

1. Select the frame clock prescale using bits, LP3:LP0 (LCDPS<3:0>).
2. Configure the appropriate pins to function as segment drivers using the LCDSEx registers.
3. Configure the LCD module for the following using the LCDCON register:
  - Multiplex and Bias mode, LMUX1:LMUX0 bits
  - Timing source, CS1:CS0 bits
  - Sleep mode, SLPEN bit
4. Write initial values to Pixel Data registers, LCDDATA0 through LCDDATA23.
5. Clear LCD Interrupt Flag, LCDIF (PIR3<6>), and if desired, enable the interrupt by setting bit, LCDIE (PIE3<6>).
6. Enable the LCD module by setting bit, LCDEN (LCDCON<7>).



# PIC18F6390/6490/8390/8490

**TABLE 22-6: REGISTERS ASSOCIATED WITH LCD 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	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
LCDDATA23 <sup>(1)</sup>	S47C3	S46C3	S45C3	S44C3	S43C3	S42C3	S41C3	S40C3	63
LCDDATA22 <sup>(1)</sup>	S39C3	S38C3	S37C3	S36C3	S35C3	S34C3	S33C3	S32C3	63
LCDDATA21	S31C3	S30C3	S29C3	S28C3	S27C3	S26C3	S25C3	S24C3	63
LCDDATA20	S23C3	S22C3	S21C3	S20C3	S19C3	S18C3	S17C3	S16C3	63
LCDDATA19	S15C3	S14C3	S13C3	S12C3	S11C3	S10C3	S09C3	S08C3	63
LCDDATA18	S07C3	S06C3	S05C3	S04C3	S03C3	S02C3	S01C3	S00C3	63
LCDDATA17 <sup>(1)</sup>	S47C2	S46C2	S45C2	S44C2	S43C2	S42C2	S41C2	S40C2	63
LCDDATA16 <sup>(1)</sup>	S39C2	S38C2	S37C2	S36C2	S35C2	S34C2	S33C2	S32C2	63
LCDDATA15	S31C2	S30C2	S29C2	S28C2	S27C2	S26C2	S25C2	S24C2	63
LCDDATA14	S23C2	S22C2	S21C2	S20C2	S19C2	S18C2	S17C2	S16C2	63
LCDDATA13	S15C2	S14C2	S13C2	S12C2	S11C2	S10C2	S09C2	S08C2	63
LCDDATA12	S07C2	S06C2	S05C2	S04C2	S03C2	S02C2	S01C2	S00C2	63
LCDDATA11 <sup>(1)</sup>	S47C1	S46C1	S45C1	S44C1	S43C1	S42C1	S41C1	S40C1	63
LCDDATA10 <sup>(1)</sup>	S39C1	S38C1	S37C1	S36C1	S35C1	S34C1	S33C1	S32C1	63
LCDDATA9	S31C1	S30C1	S29C1	S28C1	S27C1	S26C1	S25C1	S24C1	63
LCDDATA8	S23C1	S22C1	S21C1	S20C1	S19C1	S18C1	S17C1	S16C1	63
LCDDATA7	S15C1	S14C1	S13C1	S12C1	S11C1	S10C1	S09C1	S08C1	63
LCDDATA6	S07C1	S06C1	S05C1	S04C1	S03C1	S02C1	S01C1	S00C1	63
LCDDATA5 <sup>(1)</sup>	S47C0	S46C0	S45C0	S44C0	S43C0	S42C0	S41C0	S40C0	63
LCDDATA4 <sup>(1)</sup>	S39C0	S38C0	S37C0	S36C0	S35C0	S34C0	S33C0	S32C0	63
LCDDATA3	S31C0	S30C0	S29C0	S28C0	S27C0	S26C0	S25C0	S24C0	63
LCDDATA2	S23C0	S22C0	S21C0	S20C0	S19C0	S18C0	S17C0	S16C0	63
LCDDATA1	S15C0	S14C0	S13C0	S12C0	S11C0	S10C0	S09C0	S08C0	63
LCDDATA0	S07C0	S06C0	S05C0	S04C0	S03C0	S02C0	S01C0	S00C0	63
LCDSE5 <sup>(2)</sup>	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	64
LCDSE4 <sup>(2)</sup>	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	64
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	64
LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0	64

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

**Note 1:** These registers are implemented but unused on 64-pin devices and may be used as general purpose data RAM.

**2:** These registers are unimplemented on 64-pin devices.

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## 23.0 SPECIAL FEATURES OF THE CPU

PIC18F6390/6490/8390/8490 devices include several features intended to maximize reliability and minimize cost through elimination of external components. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- 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, PIC18F6390/6490/8390/8490 devices have a Watchdog Timer, which is either permanently enabled via the Configuration bits, or software controlled (if configured as disabled).

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up, while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

### 23.1 Configuration Bits

The Configuration bits can be programmed (read as ‘0’), or left unprogrammed (read as ‘1’), to select various device configurations. These bits are mapped starting at program memory location 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFh), which can only be accessed using table reads.

**TABLE 23-1: CONFIGURATION BITS AND DEVICE IDs**

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	00-- 0111
300002h	CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	PWRTEN	---1 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	—	CCP2MX	1--- -0-1
300006h	CONFIG4L	DEBUG	XINST	—	—	—	—	—	STVREN	10-- ---1
300008h	CONFIG5L	—	—	—	—	—	—	—	CP	---- ---1
3FFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx <sup>(1)</sup>
3FFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 xxxx <sup>(1)</sup>

**Legend:** x = unknown, u = unchanged, - = unimplemented, q = value depends on condition.  
Shaded cells are unimplemented, read as ‘0’.

**Note 1:** See Register 23-7 for DEVID values. DEVID registers are read-only and cannot be programmed by the user.

# PIC18F6390/6490/8390/8490

## REGISTER 23-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

### Legend:

R = Readable bit                      P = Programmable bit                      U = Unimplemented bit, read as '0'  
 -n = Value when device is unprogrammed                      u = Unchanged from programmed state

- bit 7            **IESO:** Internal/External Oscillator Switchover bit  
                   1 = Oscillator Switchover mode enabled  
                   0 = Oscillator Switchover mode disabled
- bit 6            **FCMEN:** Fail-Safe Clock Monitor Enable bit  
                   1 = Fail-Safe Clock Monitor enabled  
                   0 = Fail-Safe Clock Monitor disabled
- bit 5-4        **Unimplemented:** Read as '0'
- bit 3-0        **FOSC3:FOSC0:** Oscillator Selection bits  
                   11xx = External RC oscillator, CLKO function on RA6  
                   101x = External RC oscillator, CLKO function on RA6  
                   1001 = Internal oscillator block, CLKO function on RA6, port function on RA7  
                   1000 = Internal oscillator block, port function on RA6 and RA7  
                   0111 = External RC oscillator, port function on RA6  
                   0110 = HS oscillator, PLL enabled (clock frequency = 4 x FOSC1)  
                   0101 = EC oscillator, port function on RA6  
                   0100 = EC oscillator, CLKO function on RA6  
                   0011 = External RC oscillator, CLKO function on RA6  
                   0010 = HS oscillator  
                   0001 = XT oscillator  
                   0000 = LP oscillator

# PIC18F6390/6490/8390/8490

## REGISTER 23-2: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	BORV1	BORV0	BOREN1 <sup>(1)</sup>	BOREN0 <sup>(1)</sup>	$\overline{\text{PWRTEN}}^{\text{(1)}}$
bit 7							bit 0

### Legend:

R = Readable bit

P = Programmable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

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

bit 4-3 **BORV1:BORV0:** Brown-out Reset Voltage bits

11 = VBOR set to 2.1V

10 = VBOR set to 2.8V

01 = VBOR set to 4.3V

00 = VBOR set to 4.6V

bit 2-1 **BOREN1:BOREN0** Brown-out Reset Enable bits<sup>(1)</sup>

11 = Brown-out Reset enabled in hardware only (SBOREN is disabled)

10 = Brown-out Reset enabled in hardware only and disabled in Sleep mode (SBOREN is disabled)

01 = Brown-out Reset enabled and controlled by software (SBOREN is enabled)

00 = Brown-out Reset disabled in hardware and software

bit 0 **PWRTEN:** Power-up Timer Enable bit<sup>(1)</sup>

1 = PWRT disabled

0 = PWRT enabled

**Note 1:** The Power-up Timer is decoupled from Brown-out Reset, allowing these features to be independently controlled.

# PIC18F6390/6490/8390/8490

## REGISTER 23-3: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

### Legend:

R = Readable bit                      P = Programmable bit                      U = Unimplemented bit, read as '0'  
 -n = Value when device is unprogrammed                      u = Unchanged from programmed state

- bit 7-5            **Unimplemented:** Read as '0'
- bit 4-1            **WDTPS3:WDTPS0:** Watchdog Timer Postscale Select bits  
 1111 = 1:32,768  
 1110 = 1:16,384  
 1101 = 1:8,192  
 1100 = 1:4,096  
 1011 = 1:2,048  
 1010 = 1:1,024  
 1001 = 1:512  
 1000 = 1:256  
 0111 = 1:128  
 0110 = 1:64  
 0101 = 1:32  
 0100 = 1:16  
 0011 = 1:8  
 0010 = 1:4  
 0001 = 1:2  
 0000 = 1:1
- bit 0            **WDTEN:** Watchdog Timer Enable bit  
 1 = WDT enabled  
 0 = WDT disabled (control is placed on the SWDTEN bit)

## REGISTER 23-4: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

R/P-1	U-0	U-0	U-0	U-0	R/P-0	U-0	R/P-1
MCLRE	—	—	—	—	LPT1OSC	—	CCP2MX
bit 7							bit 0

### Legend:

R = Readable bit                      P = Programmable bit                      U = Unimplemented bit, read as '0'  
 -n = Value when device is unprogrammed                      u = Unchanged from programmed state

- bit 7            **MCLRE:**  $\overline{\text{MCLR}}$  Pin Enable bit  
 1 =  $\overline{\text{MCLR}}$  pin enabled; RG5 input pin disabled  
 0 = RG5 input pin enabled;  $\overline{\text{MCLR}}$  disabled
- bit 6-3            **Unimplemented:** Read as '0'
- bit 2            **LPT1OSC:** Low-Power Timer 1 Oscillator Enable bit  
 1 = Timer1 configured for low-power operation  
 0 = Timer1 configured for higher power operation
- bit 1            **Unimplemented:** Read as '0'
- bit 0            **CCP2MX:** CCP2 MUX bit  
 1 = CCP2 input/output is multiplexed with RC1  
 0 = CCP2 input/output is multiplexed with RE7

# PIC18F6390/6490/8390/8490

## REGISTER 23-5: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/P-1	R/P-0	U-0	U-0	U-0	U-0	U-0	R/P-1
$\overline{\text{DEBUG}}$	XINST	—	—	—	—	—	STVREN
bit 7							bit 0

### Legend:

R = Readable bit                      P = Programmable bit                      U = Unimplemented bit, read as '0'  
 -n = Value when device is unprogrammed                      u = Unchanged from programmed state

- bit 7                      **DEBUG:** Background Debugger Enable bit  
                                     1 = Background debugger disabled, RB6 and RB7 configured as general purpose I/O pins  
                                     0 = Background debugger enabled, RB6 and RB7 are dedicated to In-Circuit Debug
- bit 6                      **XINST:** Extended Instruction Set Enable bit  
                                     1 = Instruction set extension and Indexed Addressing mode enabled  
                                     0 = Instruction set extension and Indexed Addressing mode disabled (Legacy mode)
- bit 5-1                      **Unimplemented:** Read as '0'
- bit 0                      **STVREN:** Stack Full/Underflow Reset Enable bit  
                                     1 = Stack full/underflow will cause Reset  
                                     0 = Stack full/underflow will not cause Reset

## REGISTER 23-6: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/C-1
—	—	—	—	—	—	—	CP
bit 7							bit 0

### Legend:

R = Readable bit                      C = Clearable bit                      U = Unimplemented bit, read as '0'  
 -n = Value when device is unprogrammed                      u = Unchanged from programmed state

- bit 7-1                      **Unimplemented:** Read as '0'
- bit 0                      **CP:** Code Protection bit  
                                     1 = Program memory block (000000-003FFFh) not code-protected  
                                     0 = Program memory block (000000-003FFFh) code-protected

# PIC18F6390/6490/8390/8490

## REGISTER 23-7: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F6390/6490/8390/8490 DEVICES

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

### Legend:

R = Readable bit

P = Programmable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

bit 7-5 **DEV2:DEV0:** Device ID bits

100 = PIC18F8390/8490

101 = PIC18F6390/6490

bit 4-0 **REV4:REV0:** Revision ID bits

These bits are used to indicate the device revision.

## REGISTER 23-8: DEVID2: DEVICE ID REGISTER 2 FOR PIC18F6390/6490/8390/8490 DEVICES

R	R	R	R	R	R	R	R
DEV10 <sup>(1)</sup>	DEV9 <sup>(1)</sup>	DEV8 <sup>(1)</sup>	DEV7 <sup>(1)</sup>	DEV6 <sup>(1)</sup>	DEV5 <sup>(1)</sup>	DEV4 <sup>(1)</sup>	DEV3 <sup>(1)</sup>
bit 7							bit 0

### Legend:

R = Readable bit

P = Programmable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

bit 7-0 **DEV10:DEV3:** Device ID bits<sup>(1)</sup>

These bits are used with the DEV2:DEV0 bits in the Device ID Register 1 to identify the part number.

0000 0110 = PIC18F6490/8490 devices

0000 1011 = PIC18F6390/8390 devices

**Note 1:** These values for DEV10:DEV3 may be shared with other devices. The specific device is always identified by using the entire DEV10:DEV0 bit sequence.



## 23.2 Watchdog Timer (WDT)

For PIC18F6390/6490/8390/8490 devices, the WDT is driven by the INTRC source. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the INTRC oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexer, controlled by bits in Configuration Register 2H. Available periods range from 4 ms to 134.2 seconds (2.24 minutes). The WDT and postscaler are cleared when any of the following events occur: a `SLEEP` or `CLRWDT` instruction is executed, the IRCF bits (`OSCCON<6:4>`) are changed, or a clock failure has occurred.

**Note 1:** The `CLRWDT` and `SLEEP` instructions clear the WDT and postscaler counts when executed.

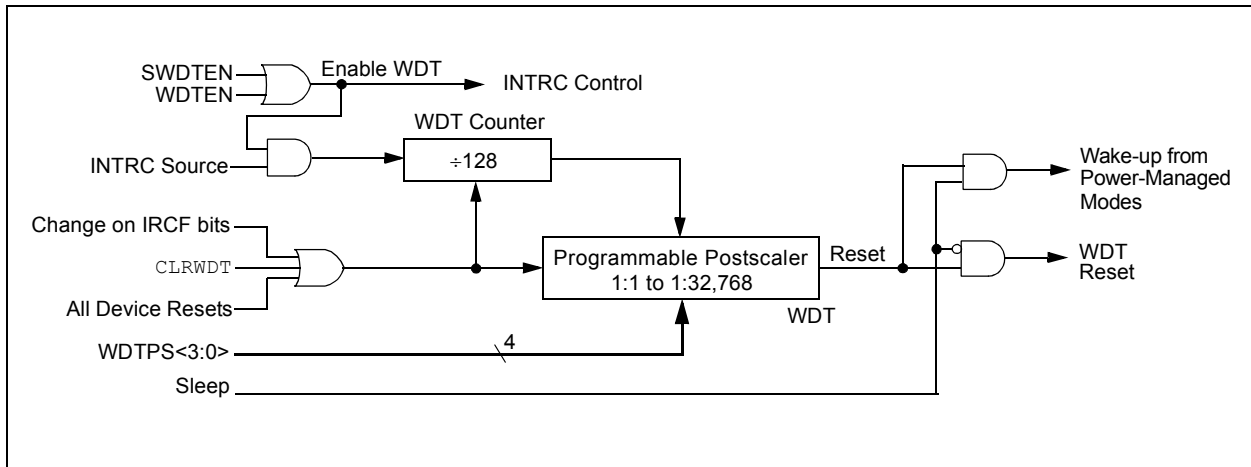
**2:** Changing the setting of the IRCF bits (`OSCCON<6:4>`) clears the WDT and postscaler counts.

**3:** When a `CLRWDT` instruction is executed, the postscaler count will be cleared.

### 23.2.1 CONTROL REGISTER

Register 23-9 shows the WDTCON register. This is a readable and writable register, which contains a control bit that allows software to override the WDT enable Configuration bit, but only if the Configuration bit has disabled the WDT.

**FIGURE 23-1: WDT BLOCK DIAGRAM**



# PIC18F6390/6490/8390/8490

## REGISTER 23-9: WDTCON: WATCHDOG TIMER CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	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 7-1                      **Unimplemented:** Read as '0'  
 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, WDTEEN, is enabled.

**TABLE 23-2: SUMMARY OF WATCHDOG TIMER REGISTERS**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
WDTCON	—	—	—	—	—	—	—	SWDTEN	60

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

## 23.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is LP, XT, HS or HSPLL (Crystal-Based modes). Other sources do not require a OST start-up delay; for these, Two-Speed Start-up 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.

Because the OSCCON register is cleared on Reset events, the INTOSC (or postscaler) clock source is not initially available after a Reset event; the INTRC clock is used directly at its base frequency. To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF2:IRCF0, immediately after

Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF2:IRCF0 bits prior to entering Sleep mode.

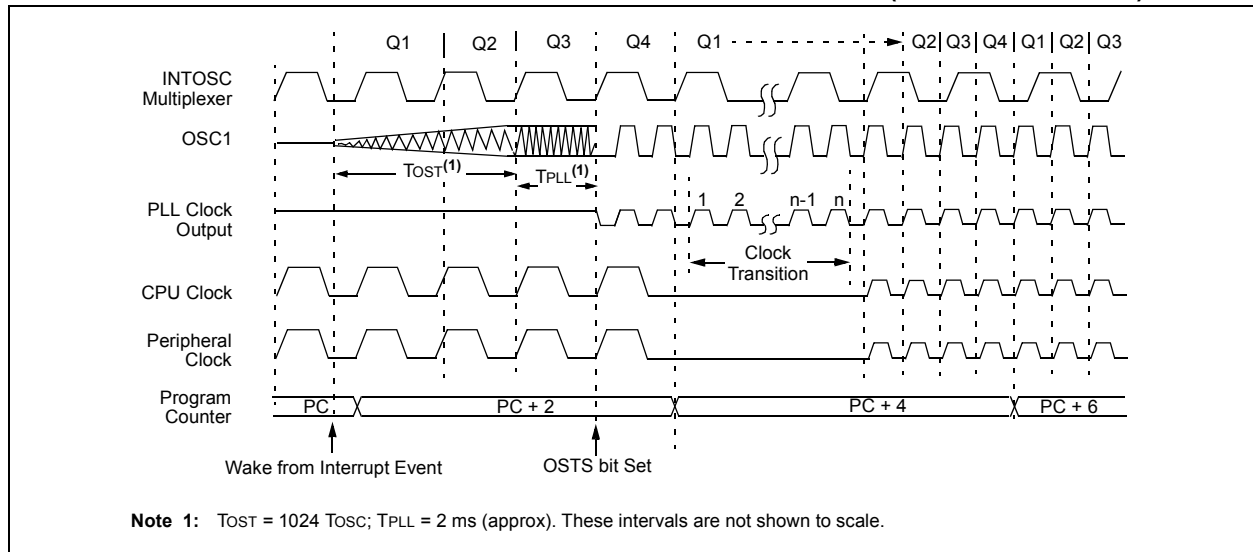
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

### 23.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial `SLEEP` instructions (refer to **Section 3.1.2 “Entering Power-Managed Modes”**). In practice, this means that user code can change the SCS1:SCS0 bit settings or issue `SLEEP` instructions before the OST times out. This would allow an application to briefly wake-up, perform routine “housekeeping” tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the OSTS bit (`OSCCON<3>`). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

**FIGURE 23-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)**



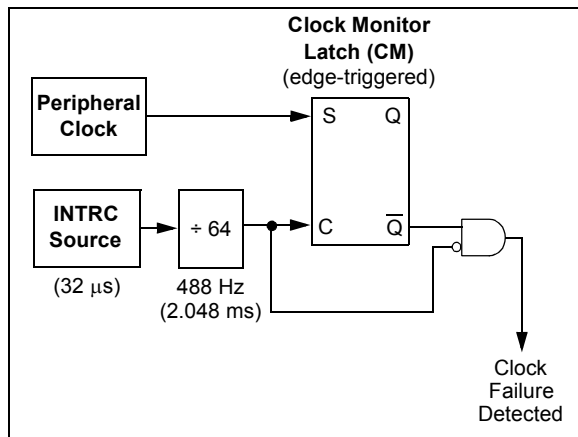
# PIC18F6390/6490/8390/8490

## 23.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the 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 23-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the device clock source, but cleared on the rising edge of the sample clock.

**FIGURE 23-3: FSCM BLOCK DIAGRAM**



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 23-4). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit OSCFIF (PIR2<7>);
- the device clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the Fail-Safe condition); and
- the WDT is reset.

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shut-down. See **Section 3.1.2 “Entering Power-Managed Modes”** and **Section 23.3.1 “Special Considerations for Using Two-Speed Start-up”** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF2:IRCF0, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF2:IRCF0 bits prior to entering Sleep mode.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

### 23.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

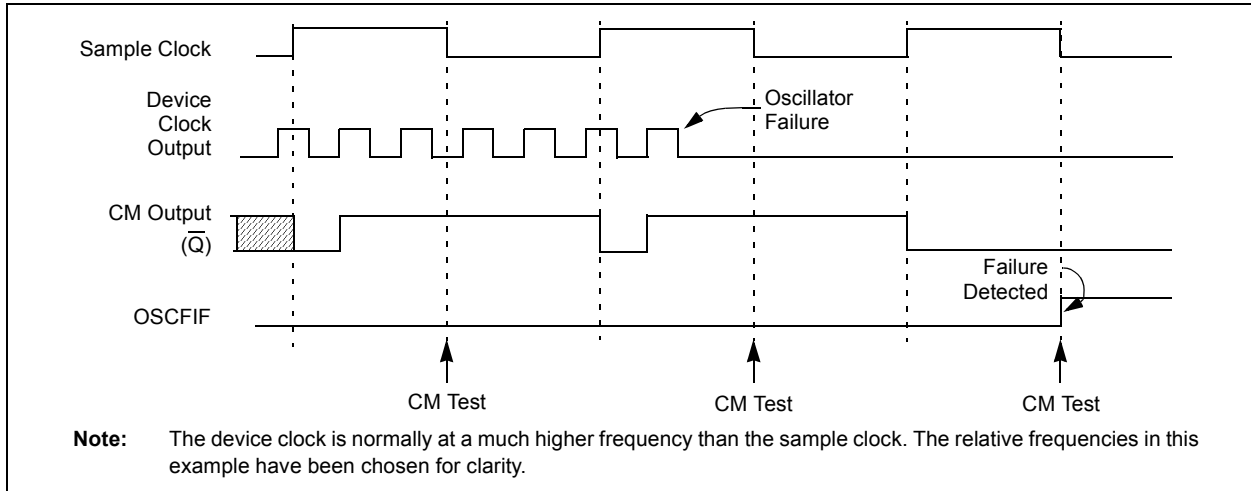
As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF2:IRCF0 bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing them to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

### 23.4.2 EXITING FAIL-SAFE OPERATION

The Fail-Safe condition is terminated by either a device Reset or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as the OST or PLL timer). The INTOSC multiplexer 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 Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

**FIGURE 23-4: FSCM TIMING DIAGRAM**



### 23.4.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. Fail-Safe Clock Monitoring of the power-managed clock source resumes in the power-managed mode.

If an oscillator failure occurs during power-managed operation, the subsequent events depend on whether or not the Oscillator Failure Interrupt Flag is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTOSC multiplexer. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, the device will not exit the power-managed mode on oscillator failure. Instead, the device will continue to operate as before, but clocked by the INTOSC multiplexer. While in Idle mode, subsequent interrupts will cause the CPU to begin executing instructions while being clocked by the INTOSC multiplexer.

### 23.4.4 POR OR WAKE FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary device clock is EC, RC or INTRC modes, monitoring can begin immediately following these events.

For oscillator modes involving a crystal or resonator (HS, HSPLL, LP or XT), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the 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 from Sleep, will also prevent the detection of the oscillator's failure to start at all following these events. This can be avoided by monitoring the OSTs bit and using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged.

As noted in **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration and enter an alternate power-managed mode while waiting for the primary clock to become stable. When the new power-managed mode is selected, the primary clock is disabled.

# PIC18F6390/6490/8390/8490

## 23.5 Program Verification and Code Protection

The overall structure of the code protection on the PIC18F6390/6490/8390/8490 Flash devices differs from previous PIC18 devices.

For all devices in the PIC18F6X90/8X90 family, the user program memory is made of a single block. Figure 23-5 shows the program memory organization for individual devices. Code protection for this block is controlled by a single bit, CP (CONFIG5L<0>). The CP bit inhibits external reads from and writes to the entire program memory space. It has no direct effect in normal execution mode.

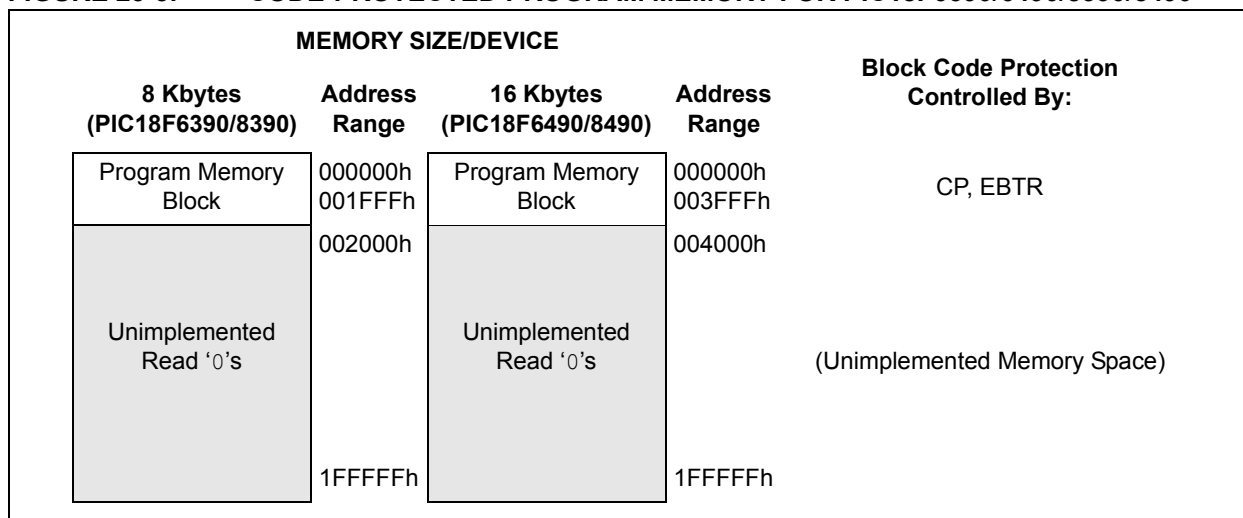
### 23.5.1 READING PROGRAM MEMORY AND OTHER LOCATIONS

The program memory may be read to any location using the table read instructions. The Device ID and the Configuration registers may be read with the table read instructions.

### 23.5.2 CONFIGURATION REGISTER PROTECTION

The Configuration registers can only be written via ICSP using an external programmer. No separate protection bit is associated with them.

**FIGURE 23-5: CODE-PROTECTED PROGRAM MEMORY FOR PIC18F6390/6490/8390/8490**



**TABLE 23-3: SUMMARY OF CODE PROTECTION REGISTERS**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h   CONFIG5L	—	—	—	—	—	—	—	CP

**Legend:** Shaded cells are unimplemented.

# PIC18F6390/6490/8390/8490

## 23.6 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are readable during normal execution through the `TBLRD` instruction. During program/verify, these locations are readable and writable. The ID locations can be read when the device is code-protected.

## 23.7 In-Circuit Serial Programming

PIC18F6390/6490/8390/8490 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.

## 23.8 In-Circuit Debugger

When the `DEBUG` Configuration bit is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB<sup>®</sup> IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 23-4 shows which resources are required by the background debugger.

**TABLE 23-4: DEBUGGER RESOURCES**

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	512 bytes
Data Memory:	10 bytes

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to `MCLR/VPP`, `VDD`, `VSS`, `RB7` and `RB6`. This will interface to the In-Circuit Debugger module available from Microchip or one of the third party development tool companies.

# PIC18F6390/6490/8390/8490

---

NOTES:



## 24.0 INSTRUCTION SET SUMMARY

PIC18FXX90 devices incorporate the standard set of seventy-five PIC18 core instructions, as well as an extended set of 8 new instructions for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

### 24.1 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 24-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 24-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The destination of the result (specified by 'd')
3. The accessed memory (specified by 'a')

The file register designator, 'f', specifies which file register is to be used by the instruction. The destination designator, 'd', specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the WREG register. If 'd' is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The bit in the file register (specified by 'b')
3. The accessed memory (specified by 'a')

The bit field designator, 'b', selects the number of the bit affected by the operation, while the file register designator, 'f', represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by 'k')
- The desired FSR register to load the literal value into (specified by 'f')
- No operand required (specified by '—')

The **control** instructions may use some of the following operands:

- A program memory address (specified by 'n')
- The mode of the `CALL` or `RETURN` instructions (specified by 's')
- The mode of the table read and table write instructions (specified by 'm')
- No operand required (specified by '—')

All instructions are a single word, except for 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 `NOOP`.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a `NOOP`.

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 24-1 shows the general formats that the instructions can have. All examples use the convention 'nnh' to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 24-2, lists the standard instructions recognized by the Microchip MPASM<sup>™</sup> Assembler.

**Section 24.1.1 "Standard Instruction Set"** provides a description of each instruction.

# PIC18F6390/6490/8390/8490

**TABLE 24-1: 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. Only used with table read and table write instructions:
*	No change to register (such as TBLPTR with table reads and writes)
++	Post-Increment register (such as TBLPTR with table reads and writes)
--	Post-Decrement register (such as TBLPTR with table reads and writes)
++	Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions or the direct address for Call/Branch and Return instructions.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.
$\overline{PD}$	Power-Down bit.
PRODH	Product of Multiply High Byte.
PRODL	Product of Multiply Low Byte.
s	Fast Call/Return mode select bit s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
TBLPTR	21-bit Table Pointer (points to a Program Memory location).
TABLAT	8-bit Table Latch.
TO	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 an indexed address.
(text)	The contents of text.
[expr]<n>	Specifies bit n of the register indicated by the pointer expr.
→	Assigned to.
< >	Register bit field.
∈	In the set of.
<i>italics</i>	User-defined term (font is Courier New).

# PIC18F6390/6490/8390/8490

**FIGURE 24-1: GENERAL FORMAT FOR INSTRUCTIONS**

Byte-oriented file register operations	Example Instruction																																																
<table border="1" 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">OPCODE</td> <td style="text-align: center;">d</td> <td style="text-align: center;">a</td> <td colspan="2">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 border="1" 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">OPCODE</td> <td colspan="2">f (Source FILE #)</td> </tr> </table> <table border="1" 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">1111</td> <td colspan="2">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 border="1" 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">OPCODE</td> <td style="text-align: center;">b (BIT #)</td> <td style="text-align: center;">a</td> <td colspan="3">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 border="1" 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">OPCODE</td> <td colspan="2">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 border="1" 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">OPCODE</td> <td colspan="2">n&lt;7:0&gt; (literal)</td> </tr> </table> <table border="1" 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">1111</td> <td colspan="2">n&lt;19:8&gt; (literal)</td> </tr> </table> <p>n = 20-bit immediate value</p> <table border="1" 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">OPCODE</td> <td style="text-align: center;">S</td> <td>n&lt;7:0&gt; (literal)</td> </tr> </table> <table border="1" 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">1111</td> <td colspan="2">n&lt;19:8&gt; (literal)</td> </tr> </table> <p>S = Fast bit</p> <table border="1" 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">OPCODE</td> <td colspan="2">n&lt;10:0&gt; (literal)</td> </tr> </table> <table border="1" 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">OPCODE</td> <td colspan="2">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	12	11	0	1111		n<19:8> (literal)		15	11	10	0	OPCODE		n<10:0> (literal)		15	8	7	0	OPCODE		n<7:0> (literal)		GOTO Label  CALL MYFUNC  BRA MYFUNC  BC MYFUNC
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	12	11	0																																														
1111		n<19:8> (literal)																																															
15	11	10	0																																														
OPCODE		n<10:0> (literal)																																															
15	8	7	0																																														
OPCODE		n<7:0> (literal)																																															

# PIC18F6390/6490/8390/8490

TABLE 24-2: PIC18FXXX 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 1st word f <sub>d</sub> (destination) 2nd word	2	1100	ffff	ffff	ffff	None	
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	1, 2
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	1, 2
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	1, 2
SUBFWB	f, d, a	Subtract f from WREG with Borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	Subtract WREG from f with Borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap Nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, Skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	

**Note 1:** When a PORT register is modified as a function of itself (e.g., MOVF PORTB, 1, 0), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

- If this instruction is executed on the TMR0 register (and where applicable, 'd' = 1), the prescaler will be cleared if assigned.
- 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.
- 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.
- If the table write starts the write cycle to internal memory, the write will continue until terminated.

# PIC18F6390/6490/8390/8490

**TABLE 24-2: PIC18FXXXX 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, d, 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 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$ , $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C	
GOTO	n	Go to Address 1st word	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	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., `MOVWF 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 an input and is driven low by an external device, the data will be written back with a '0'.

- If this instruction is executed on the TMR0 register (and where applicable, 'd' = 1), the prescaler will be cleared if assigned.
- 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`.
- 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.
- If the table write starts the write cycle to internal memory, the write will continue until terminated.

# PIC18F6390/6490/8390/8490

**TABLE 24-2: PIC18FXXXX 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	5
TBLWT*+		Table Write with Post-Increment		0000	0000	0000	1101	None	5
TBLWT*-		Table Write with Post-Decrement		0000	0000	0000	1110	None	5
TBLWT+*		Table Write with Pre-Increment		0000	0000	0000	1111	None	5

- 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 an input and is driven low by an external device, the data will be written back with a '0'.
- If this instruction is executed on the TMR0 register (and where applicable, 'd' = 1), the prescaler will be cleared if assigned.
  - 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`.
  - 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.
  - If the table write starts the write cycle to internal memory, the write will continue until terminated.

# PIC18F6390/6490/8390/8490

## 24.1.1 STANDARD INSTRUCTION SET

### ADDLW ADD Literal to W

Syntax:	ADDLW k			
Operands:	$0 \leq k \leq 255$			
Operation:	$(W) + k \rightarrow W$			
Status Affected:	N, OV, C, DC, Z			
Encoding:	0000	1111	kkkk	kkkk
Description:	The contents of W are added to the 8-bit literal 'k' and the result is placed in W.			
Words:	1			
Cycles:	1			
Q Cycle Activity:	Q1	Q2	Q3	Q4
	Decode	Read literal 'k'	Process Data	Write to W

**Example:** ADDLW 15h

Before Instruction  
W = 10h  
After Instruction  
W = 25h

### ADDWF ADD W to f

Syntax:	ADDWF f {,d {,a}}			
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$			
Operation:	$(W) + (f) \rightarrow \text{dest}$			
Status Affected:	N, OV, C, DC, Z			
Encoding:	0010	01da	ffff	ffff
Description:	Add W to register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank. 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"</b> for details.			
Words:	1			
Cycles:	1			
Q Cycle Activity:	Q1	Q2	Q3	Q4
	Decode	Read register 'f'	Process Data	Write to destination

**Example:** ADDWF REG, 0, 0

Before Instruction  
W = 17h  
REG = 0C2h  
After Instruction  
W = 0D9h  
REG = 0C2h

**Note:** All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

# PIC18F6390/6490/8390/8490

## ADDWFC      ADD W and Carry bit to f

**Syntax:**            ADDWFC    f {,d {,a}}

**Operands:**         $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

**Operation:**         $(W) + (f) + (C) \rightarrow \text{dest}$

**Status Affected:**    N,OV, C, DC, Z

**Encoding:**

0010	00da	ffff	ffff
------	------	------	------

**Description:**     Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.  
 If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "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:**         $(W) .AND. k \rightarrow W$

**Status Affected:**    N, Z

**Encoding:**

0000	1011	kkkk	kkkk
------	------	------	------

**Description:**     The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

**Words:**            1

**Cycles:**            1

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

**Example:**            ANDLW    05Fh

Before Instruction  
     W        = A3h

After Instruction  
     W        = 03h



# PIC18F6390/6490/8390/8490

## 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.  
 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** ANDWF REG, 0, 0

Before Instruction  
 W = 17h  
 REG = C2h  
 After Instruction  
 W = 02h  
 REG = C2h

## BC Branch if Carry

**Syntax:** BC n

**Operands:**  $-128 \leq n \leq 127$

**Operation:** if Carry bit is '1',  
 $(PC) + 2 + 2n \rightarrow PC$

**Status Affected:** None

**Encoding:**

1110	0010	nnnn	nnnn
------	------	------	------

**Description:** If the Carry bit is '1', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.

**Words:** 1

**Cycles:** 1(2)

**Q Cycle Activity:**

**If Jump:**

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

**If No Jump:**

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:** HERE BC 5

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Carry = 1;  
 PC = address (HERE + 12)  
 If Carry = 0;  
 PC = address (HERE + 2)

# PIC18F6390/6490/8390/8490

## BCF

### Bit Clear f

Syntax: BCF f, b {,a}

Operands:  $0 \leq f \leq 255$   
 $0 \leq b \leq 7$   
 $a \in [0,1]$

Operation:  $0 \rightarrow f \leftarrow b$

Status Affected: None

Encoding: 

1001	bbba	ffff	ffff
------	------	------	------

Description: Bit 'b' in register 'f' is cleared.  
 If 'a' is '0', the Access Bank is selected.  
 If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

**Example:** BCF FLAG\_REG, 7, 0

Before Instruction  
 FLAG\_REG = C7h  
 After Instruction  
 FLAG\_REG = 47h

## BN

### Branch if Negative

Syntax: BN n

Operands:  $-128 \leq n \leq 127$

Operation: if Negative bit is '1',  
 $(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding: 

1110	0110	nnnn	nnnn
------	------	------	------

Description: If the Negative bit is '1', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:** HERE BN Jump

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Negative = 1;  
 PC = address (Jump)  
 If Negative = 0;  
 PC = address (HERE + 2)

# PIC18F6390/6490/8390/8490

## BNC Branch if Not Carry

**Syntax:** BNC n

**Operands:**  $-128 \leq n \leq 127$

**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:**  $-128 \leq n \leq 127$

**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)

# PIC18F6390/6490/8390/8490

## BNOV Branch if Not Overflow

Syntax: BNOV n  
 Operands:  $-128 \leq n \leq 127$   
 Operation: if Overflow bit is '0',  
 $(PC) + 2 + 2n \rightarrow PC$   
 Status Affected: None  
 Encoding: 

1110	0101	nnnn	nnnn
------	------	------	------

  
 Description: If the Overflow bit is '0', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.  
 Words: 1  
 Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:                    HERE                    BNOV Jump

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Overflow = 0;  
 PC = address (Jump)  
 If Overflow = 1;  
 PC = address (HERE + 2)

## BNZ Branch if Not Zero

Syntax: BNZ n  
 Operands:  $-128 \leq n \leq 127$   
 Operation: if Zero bit is '0',  
 $(PC) + 2 + 2n \rightarrow PC$   
 Status Affected: None  
 Encoding: 

1110	0001	nnnn	nnnn
------	------	------	------

  
 Description: If the Zero bit is '0', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.  
 Words: 1  
 Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:                    HERE                    BNZ Jump

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Zero = 0;  
 PC = address (Jump)  
 If Zero = 1;  
 PC = address (HERE + 2)

# PIC18F6390/6490/8390/8490

## 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.  
 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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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 '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.

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 24.2.3 "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 '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.

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 24.2.3 "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)

# PIC18F6390/6490/8390/8490

## 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:**

0111	bbba	ffff	ffff
------	------	------	------

**Description:** Bit 'b' in data memory location 'f' is inverted.  
 If 'a' is '0', the Access Bank is selected.  
 If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

**Example:** BTG PORTC, 4, 0

Before Instruction:  
 PORTC = 0111 0101 [75h]  
 After Instruction:  
 PORTC = 0110 0101 [65h]

## BOV Branch if Overflow

**Syntax:** BOV n

**Operands:**  $-128 \leq n \leq 127$

**Operation:** if Overflow bit is '1',  
 $(PC) + 2 + 2n \rightarrow PC$

**Status Affected:** None

**Encoding:**

1110	0100	nnnn	nnnn
------	------	------	------

**Description:** If the Overflow bit is '1', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.

**Words:** 1

**Cycles:** 1(2)

**Q Cycle Activity:**

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:** HERE BOV Jump

Before Instruction  
 PC = address (HERE)  
 After Instruction  
 If Overflow = 1;  
 PC = address (Jump)  
 If Overflow = 0;  
 PC = address (HERE + 2)

# PIC18F6390/6490/8390/8490

## BZ Branch if Zero

**Syntax:** BZ n

**Operands:**  $-128 \leq n \leq 127$

**Operation:** if Zero bit is '1',  
 $(PC) + 2 + 2n \rightarrow PC$

**Status Affected:** None

**Encoding:**

1110	0000	nnnn	nnnn
------	------	------	------

**Description:** If the Zero bit is '1', then the program will branch.  
 The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be  $PC + 2 + 2n$ . This instruction is then a two-cycle instruction.

**Words:** 1

**Cycles:** 1(2)

**Q Cycle Activity:**

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:**                    HERE                    BZ    Jump

Before Instruction  
 PC = address (HERE)

After Instruction  
 If Zero = 1;  
 PC = address (Jump)  
 If Zero = 0;  
 PC = address (HERE + 2)

## CALL Subroutine Call

**Syntax:** CALL k {,s}

**Operands:**  $0 \leq k \leq 1048575$   
 $s \in [0,1]$

**Operation:**  $(PC) + 4 \rightarrow TOS$ ,  
 $k \rightarrow PC<20:1>$ ;  
 if  $s = 1$ ,  
 $(W) \rightarrow WS$ ,  
 $(STATUS) \rightarrow STATUSS$ ,  
 $(BSR) \rightarrow BSRS$

**Status Affected:** None

**Encoding:**

1110	110s	$k_7kkk$	$kkkk_0$
1111	$k_{19}kkk$	$kkkk$	$kkkk_8$

**Description:** Subroutine call of entire 2-Mbyte memory range. First, return address  $(PC + 4)$  is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into  $PC<20:1>$ . CALL is a two-cycle instruction.

**Words:** 2

**Cycles:** 2

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>, Push PC to stack	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



# PIC18F6390/6490/8390/8490

## CLRF Clear f

**Syntax:** CLRF f{,a}

**Operands:**  $0 \leq f \leq 255$   
 $a \in [0,1]$

**Operation:**  $000h \rightarrow f$ ,  
 $1 \rightarrow Z$

**Status Affected:** Z

**Encoding:**

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.  
 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 24.2.3 "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:** CLRF FLAG\_REG, 1

Before Instruction  
 FLAG\_REG = 5Ah  
 After Instruction  
 FLAG\_REG = 00h

## CLRWDT Clear Watchdog Timer

**Syntax:** CLRWDT

**Operands:** None

**Operation:**  $000h \rightarrow$  WDT,  
 $000h \rightarrow$  WDT postscaler,  
 $1 \rightarrow \overline{TO}$ ,  
 $1 \rightarrow \overline{PD}$

**Status Affected:**  $\overline{TO}$ ,  $\overline{PD}$

**Encoding:**

0000	0000	0000	0100
------	------	------	------

**Description:** CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits,  $\overline{TO}$  and  $\overline{PD}$ , are set.

**Words:** 1

**Cycles:** 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

**Example:** CLRWDT

Before Instruction  
 WDT Counter = ?  
 After Instruction  
 WDT Counter = 00h  
 WDT Postscaler = 0  
 $\overline{TO}$  = 1  
 $\overline{PD}$  = 1

# PIC18F6390/6490/8390/8490

## 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 '1', the result is stored in W. If 'd' is '0', 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.  
 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 24.2.3 "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.  
 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 24.2.3 "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)

# PIC18F6390/6490/8390/8490

## 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. 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 24.2.3 "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.

**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)

# PIC18F6390/6490/8390/8490

## DAW Decimal Adjust W Register

**Syntax:** DAW

**Operands:** None

**Operation:** If [W<3:0> >9] or [DC = 1] then, (W<3:0>) + 6 → W<3:0>;  
else,  
(W<3:0>) → W<3:0>;

If [W<7:4> >9] or [C = 1] then,  
(W<7:4>) + 6 → W<7:4>,  
C = 1;  
else,  
(W<7:4>) → 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 → 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.

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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## DECFSZ Decrement f, Skip if 0

**Syntax:** DECFSZ f {,d {,a}}

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

**Operation:**  $(f) - 1 \rightarrow \text{dest}$ ,  
skip if result = 0

**Status Affected:** None

**Encoding:**

0010	11da	ffff	ffff
------	------	------	------

**Description:** The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).  
If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.  
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1(2)  
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

**Example:**           HERE       DECFSZ   CNT, 1, 1  
  GOTO     LOOP  
  CONTINUE

Before Instruction  
PC = Address (HERE)  
After Instruction  
CNT = CNT - 1  
If CNT = 0;  
PC = Address (CONTINUE)  
If CNT  $\neq$  0;  
PC = Address (HERE + 2)

## DCFSNZ Decrement f, Skip if not 0

**Syntax:** DCFSNZ f {,d {,a}}

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

**Operation:**  $(f) - 1 \rightarrow \text{dest}$ ,  
skip if result  $\neq$  0

**Status Affected:** None

**Encoding:**

0100	11da	ffff	ffff
------	------	------	------

**Description:** The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).  
If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.  
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1(2)  
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

**Example:**           HERE       DCFSNZ   TEMP, 1, 0  
  ZERO     :  
  NZERO    :

Before Instruction  
TEMP = ?  
After Instruction  
TEMP = TEMP - 1,  
If TEMP = 0;  
PC = Address (ZERO)  
If TEMP  $\neq$  0;  
PC = Address (NZERO)

# PIC18F6390/6490/8390/8490

## GOTO Unconditional Branch

Syntax: GOTO k

Operands:  $0 \leq k \leq 1048575$

Operation:  $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word ( $k<7:0>$ )

1110	1111	$k_7$ kkk	$kkkk_0$
1111	$k_{19}$ kkk	kkkk	$kkkk_8$

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.

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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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. 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1(2)  
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

**Example:**           HERE    INCFSZ   CNT, 1, 0  
                      NZERO   :  
                      ZERO     :

Before Instruction  
PC = Address (HERE)

After Instruction  
CNT = CNT + 1  
If CNT = 0;  
PC = Address (ZERO)  
If CNT  $\neq$  0;  
PC = Address (NZERO)

## INFSNZ Increment f, Skip if not 0

**Syntax:** INFSNZ f{,d{,a}}

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

**Operation:**  $(f) + 1 \rightarrow \text{dest}$ ,  
skip if result  $\neq 0$

**Status Affected:** None

**Encoding:**

0100	10da	ffff	ffff
------	------	------	------

**Description:** The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank. 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

**Words:** 1

**Cycles:** 1(2)  
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

**Example:**           HERE    INFSNZ   REG, 1, 0  
                      ZERO   :  
                      NZERO   :

Before Instruction  
PC = Address (HERE)

After Instruction  
REG = REG + 1  
If REG  $\neq$  0;  
PC = Address (NZERO)  
If REG = 0;  
PC = Address (ZERO)

# PIC18F6390/6490/8390/8490

## IORLW Inclusive OR Literal with W

Syntax: IORLW k

Operands:  $0 \leq k \leq 255$

Operation: (W) .OR. k  $\rightarrow$  W

Status Affected: N, Z

Encoding: 

0000	1001	kkkk	kkkk
------	------	------	------

Description: The contents of W are ORed with the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

**Example:** IORLW 35h

Before Instruction  
W = 9Ah

After Instruction  
W = BFh

## IORWF Inclusive OR W with f

Syntax: IORWF f {,d {,a}}

Operands:  $0 \leq f \leq 255$   
d  $\in$  [0,1]  
a  $\in$  [0,1]

Operation: (W) .OR. (f)  $\rightarrow$  dest

Status Affected: N, Z

Encoding: 

0001	00da	ffff	ffff
------	------	------	------

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** IORWF RESULT, 0, 1

Before Instruction  
RESULT = 13h  
W = 91h

After Instruction  
RESULT = 13h  
W = 93h



# PIC18F6390/6490/8390/8490

## 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 <sub>11</sub> kkk
1111	0000	k <sub>7</sub> kkk	kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
	Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
	Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

**Example:** LFSR 2, 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.

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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## MOVFF Move f to f

Syntax: MOVFF  $f_s, f_d$

Operands:  $0 \leq f_s \leq 4095$   
 $0 \leq f_d \leq 4095$

Operation:  $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	ffff <sub>s</sub>
2nd word (destin.)	1111	ffff	ffff	ffff <sub>d</sub>

Description: The contents of source register 'f<sub>s</sub>' are moved to destination register 'f<sub>d</sub>'. Location of source 'f<sub>s</sub>' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination 'f<sub>d</sub>' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

	Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation	
Decode	No operation No dummy read	No operation	Write register 'f' (dest)	

**Example:** MOVFF REG1, REG2

Before Instruction  
 REG1 = 33h  
 REG2 = 11h

After Instruction  
 REG1 = 33h  
 REG2 = 33h

## MOVLB Move Literal to Low Nibble in BSR

Syntax: MOVLW k

Operands:  $0 \leq k \leq 255$

Operation:  $k \rightarrow \text{BSR}$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk
------	------	------	------

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0', regardless of the value of k<sub>7:k<sub>4</sub></sub>.

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

# PIC18F6390/6490/8390/8490

## 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.

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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## MULLW Multiply Literal with W

Syntax: MULLW k

Operands:  $0 \leq k \leq 255$

Operation:  $(W) \times k \rightarrow \text{PRODH:PRODL}$

Status Affected: None

Encoding: 

0000	1101	kkkk	kkkk
------	------	------	------

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in the PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged.  
None of the Status flags are affected.  
Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH:PRODL

Example: MULLW 0C4h

Before Instruction

W = E2h  
PRODH = ?  
PRODL = ?

After Instruction

W = E2h  
PRODH = ADh  
PRODL = 08h

## MULWF Multiply W with f

Syntax: MULWF f{,a}

Operands:  $0 \leq f \leq 255$   
 $a \in [0,1]$

Operation:  $(W) \times (f) \rightarrow \text{PRODH:PRODL}$

Status Affected: None

Encoding: 

0000	001a	ffff	ffff
------	------	------	------

Description: An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged.  
None of the Status flags are affected.  
Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.  
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">0110</td> <td style="padding: 2px;">110a</td> <td style="padding: 2px;">ffff</td> <td style="padding: 2px;">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.</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 24.2.3 "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;"> <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" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0000</td> <td style="padding: 2px;">0000</td> </tr> <tr> <td style="padding: 2px;">1111</td> <td style="padding: 2px;">xxxx</td> <td style="padding: 2px;">xxxx</td> <td style="padding: 2px;">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" style="display: inline-table; border-collapse: collapse; text-align: center;"> <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.

# PIC18F6390/6490/8390/8490

## POP Pop Top of Return Stack

Syntax: POP  
 Operands: None  
 Operation: (TOS) → bit bucket  
 Status Affected: None  
 Encoding: 

0000	0000	0000	0110
------	------	------	------

  
 Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack.  
 This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.  
 Words: 1  
 Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

**Example:**

	POP		
	GOTO	NEW	
Before Instruction			
TOS	=	0031A2h	
Stack (1 level down)	=	014332h	
After Instruction			
TOS	=	014332h	
PC	=	NEW	

## PUSH Push Top of Return Stack

Syntax: PUSH  
 Operands: None  
 Operation: (PC + 2) → TOS  
 Status Affected: None  
 Encoding: 

0000	0000	0000	0101
------	------	------	------

  
 Description: The PC + 2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack.  
 This instruction allows implementing a software stack by modifying TOS and then pushing it onto the return stack.  
 Words: 1  
 Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC + 2 onto return stack	No operation	No operation

**Example:**

	PUSH		
Before Instruction			
TOS	=	345Ah	
PC	=	0124h	
After Instruction			
PC	=	0126h	
TOS	=	0126h	
Stack (1 level down)	=	345Ah	

# PIC18F6390/6490/8390/8490

## RCALL

### Relative Call

**Syntax:** RCALL n

**Operands:**  $-1024 \leq n \leq 1023$

**Operation:** (PC) + 2 → TOS,  
(PC) + 2 + 2n → PC

**Status Affected:** None

**Encoding:**

1101	1nnn	nnnn	nnnn
------	------	------	------

**Description:** Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

**Words:** 1

**Cycles:** 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

**Example:**           HERE       RCALL Jump

Before Instruction

PC = Address (HERE)

After Instruction

PC = Address (Jump)

TOS = Address (HERE + 2)

## RESET

### Reset

**Syntax:** RESET

**Operands:** None

**Operation:** Reset all registers and flags that are affected by a MCLR Reset.

**Status Affected:** All

**Encoding:**

0000	0000	1111	1111
------	------	------	------

**Description:** This instruction provides a way to execute a MCLR Reset in software.

**Words:** 1

**Cycles:** 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start Reset	No operation	No operation

**Example:**           RESET

After Instruction

Registers = Reset Value

Flags\* = Reset Value

# PIC18F6390/6490/8390/8490

## RETFIE Return from Interrupt

**Syntax:** RETFIE {s}

**Operands:** s ∈ [0,1]

**Operation:** (TOS) → PC,  
1 → GIE/GIEH or PEIE/GIEL;  
if s = 1,  
(WS) → W,  
(STATUS) → STATUS,  
(BSRS) → BSR,  
PCLATU, PCLATH are unchanged

**Status Affected:** GIE/GIEH, PEIE/GIEL.

**Encoding:**

0000	0000	0001	000s
------	------	------	------

**Description:** Return from Interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low-priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUS and 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	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         = BSR
STATUS      = STATUS
GIE/GIEH, PEIE/GIEL = 1
    
```

## RETLW Return Literal to W

**Syntax:** RETLW k

**Operands:** 0 ≤ k ≤ 255

**Operation:** k → W,  
(TOS) → PC,  
PCLATU, PCLATH are unchanged

**Status Affected:** None

**Encoding:**

0000	1100	kkkk	kkkk
------	------	------	------

**Description:** W is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

**Words:** 1

**Cycles:** 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	POP PC from stack, Write to W
No operation	No operation	No operation	No operation

**Example:**

```

CALL TABLE ; W contains table
              ; offset value
              ; W now has
              ; table value
:
TABLE
  ADDWF PCL ; W = offset
  RETLW k0  ; Begin table
  RETLW k1  ;
:
:
  RETLW kn  ; End of table
Before Instruction
  W        = 07h
After Instruction
  W        = value of kn
    
```



# PIC18F6390/6490/8390/8490

## RETURN Return from Subroutine

**Syntax:** RETURN {s}

**Operands:**  $s \in [0,1]$

**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 BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

**Words:** 1

**Cycles:** 2

**Q Cycle Activity:**

	Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack	
No operation	No operation	No operation	No operation	No operation

**Example:** RETURN

After Interrupt  
PC = TOS

## RLCF Rotate Left f through Carry

**Syntax:** RLCF f{,d{,a}}

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

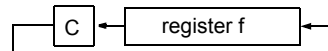
**Operation:** (f<n>) → dest<n + 1>,  
(f<7>) → C,  
(C) → dest<0>

**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.  
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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## RLNCF Rotate Left f (No Carry)

Syntax: RLNCF f {,d {,a}}

Operands:  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

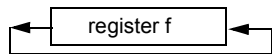
Operation:  $(f<n>) \rightarrow \text{dest}<n + 1>$ ,  
 $(f<7>) \rightarrow \text{dest}<0>$

Status Affected: N, Z

Encoding: 

0100	01da	ffff	ffff
------	------	------	------

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default).  
 If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "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:  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

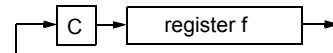
Operation:  $(f<n>) \rightarrow \text{dest}<n - 1>$ ,  
 $(f<0>) \rightarrow C$ ,  
 $(C) \rightarrow \text{dest}<7>$

Status Affected: C, N, Z

Encoding: 

0011	00da	ffff	ffff
------	------	------	------

Description: The contents of register 'f' are rotated one bit to the right through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).  
 If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## RRNCF Rotate Right f (No Carry)

**Syntax:** RRNCF f {,d {,a}}

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

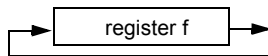
**Operation:**  $(f<n>) \rightarrow \text{dest}<n - 1>$ ,  
 $(f<0>) \rightarrow \text{dest}<7>$

**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.  
 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 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



**Words:** 1  
**Cycles:** 1

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example 1:** RRNCF REG, 1, 0

Before Instruction  
 REG = 1101 0111  
 After Instruction  
 REG = 1110 1011

**Example 2:** RRNCF REG, 0, 0

Before Instruction  
 W = ?  
 REG = 1101 0111  
 After Instruction  
 W = 1110 1011  
 REG = 1101 0111

## SETF Set f

**Syntax:** SETF f {,a}

**Operands:**  $0 \leq f \leq 255$   
 $a \in [0,1]$

**Operation:** FFh  $\rightarrow$  f

**Status Affected:** None

**Encoding:**

0110	100a	ffff	ffff
------	------	------	------

**Description:** The contents of the specified register are set to FFh.  
 If 'a' is '0', the Access Bank is selected.  
 If 'a' is '1', the BSR is used to select the GPR bank.  
 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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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. 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.  
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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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.  
 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 24.2.3 "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

# PIC18F6390/6490/8390/8490

**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.  
 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 24.2.3 "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.  
 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 24.2.3 "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

# PIC18F6390/6490/8390/8490

## 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 (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 = 0AAh  
TBLPTR = 01A357h  
MEMORY(01A357h) = 12h  
MEMORY(01A358h) = 34h

After Instruction  
TABLAT = 34h  
TBLPTR = 01A358h

# PIC18F6390/6490/8390/8490

## 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 the TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 6.0 “Flash Program Memory”** for additional details on programming Flash memory.)

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-MByte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR<0> = 0: Least Significant Byte of Program Memory Word

TBLPTR<0> = 1: Most Significant Byte of Program Memory Word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

**Words:** 1

**Cycles:** 2

**Q Cycle Activity:**

	Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)	No operation

## 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

**Note:** The table write (TBLWT) instructions are not available in user mode in PIC18F6X90/8X90 devices, as these devices are standard Flash parts without an external bus interface.



# PIC18F6390/6490/8390/8490

**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.  
 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 24.2.3 "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 ≠ 00h,  
 PC = Address (NZERO)

**XORLW      Exclusive OR Literal with W**

Syntax:      XORLW k

Operands:     $0 \leq k \leq 255$

Operation:    (W) .XOR. k → 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

# PIC18F6390/6490/8390/8490

## 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.

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 24.2.3 "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

## 24.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, PIC18FXX90 devices also provide an optional extension to the core CPU functionality. The added features include eight additional instructions that augment Indirect and Indexed Addressing operations and the implementation of Indexed Literal Offset Addressing mode for many of the standard PIC18 instructions.

The additional features are disabled by default. To enable them, users must set the XINST Configuration bit.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers or use them for Indexed Addressing. Two of the instructions, ADDFSR and SUBFSR, each have an additional special instantiation for using FSR2. These versions (ADDULNK and SUBULNK) allow for automatic return after execution.

The extended instructions are specifically implemented to optimize re-entrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- dynamic allocation and deallocation of software stack space when entering and leaving subroutines
- Function Pointer invocation
- Software Stack Pointer manipulation
- manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 24-3. Detailed descriptions are provided in **Section 24.2.2 “Extended Instruction Set”**. The opcode field descriptions in Table 24-1 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.

### 24.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the File Select Registers and some offset to specify a source or destination register. When an argument for an instruction serves as part of Indexed Addressing, it is enclosed in square brackets (“[]”). This is done to indicate that the argument is used as an index or offset. MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 24.2.3.1 “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 24-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET**

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected
			MSb		LSb		
ADDFSR f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
ADDULNK k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
CALLW	Call Subroutine using WREG	2	0000	0000	0001	0100	None
MOVSF z <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
MOVSS 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
PUSHL k	Store Literal at FSR2, Decrement FSR2	1	1110	1010	kkkk	kkkk	None
SUBFSR f, k	Subtract Literal from FSR	1	1110	1001	ffkk	kkkk	None
SUBULNK k	Subtract Literal from FSR2 and Return	2	1110	1001	11kk	kkkk	None

# PIC18F6390/6490/8390/8490

## 24.2.2 EXTENDED INSTRUCTION SET

### ADDFSR Add Literal to FSR

Syntax: ADDFSR f, k  
 Operands:  $0 \leq k \leq 63$   
 $f \in [0, 1, 2]$   
 Operation:  $FSR(f) + k \rightarrow FSR(f)$   
 Status Affected: None  
 Encoding: 

1110	1000	ffkk	kkkk
------	------	------	------

  
 Description: The 6-bit literal 'k' is added to the contents of the FSR specified by 'f'.  
 Words: 1  
 Cycles: 1  
 Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR

**Example:** ADDFSR 2, 23h

Before Instruction  
 FSR2 = 03FFh  
 After Instruction  
 FSR2 = 0422h

### ADDULNK Add Literal to FSR2 and Return

Syntax: ADDULNK k  
 Operands:  $0 \leq k \leq 63$   
 Operation:  $FSR2 + k \rightarrow FSR2$ ,  
 PC = (TOS)  
 Status Affected: None  
 Encoding: 

1110	1000	11kk	kkkk
------	------	------	------

  
 Description: The 6-bit literal 'k' is added to the contents of FSR2. A RETURN is then executed by loading the PC with the TOS.  
 The instruction takes two cycles to execute; a NOP is performed during the second cycle.  
 This may be thought of as a special case of the ADDFSR instruction, where  $f = 3$  (binary '11'); it operates only on FSR2.  
 Words: 1  
 Cycles: 2  
 Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR
No Operation	No Operation	No Operation	No Operation

**Example:** ADDULNK 23h

Before Instruction  
 FSR2 = 03FFh  
 PC = 0100h  
 After Instruction  
 FSR2 = 0422h  
 PC = (TOS)

**Note:** All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction syntax then becomes: {label} instruction argument(s).

# PIC18F6390/6490/8390/8490

## 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

# PIC18F6390/6490/8390/8490

## 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:

1111	1010	kkkk	kkkk
------	------	------	------

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation.

This instruction allows users to push values onto a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process data	Write to destination

**Example:** PUSHL 08h

Before Instruction

FSR2H:FSR2L = 01ECh  
 Memory (01ECh) = 00h

After Instruction

FSR2H:FSR2L = 01EBh  
 Memory (01ECh) = 08h

# PIC18F6390/6490/8390/8490

## **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)

# PIC18F6390/6490/8390/8490

---

## 24.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

**Note:** Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing mode (**Section 5.6.1 “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 24.2.3.1 “Extended Instruction Syntax with Standard PIC18 Commands”**).

Although the Indexed Literal Offset Addressing mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset Addressing mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

### 24.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument, ‘f’, in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value, ‘k’. As already noted, this occurs only when ‘f’ is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets (“[ ]”). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within brackets, will generate an error in the MPASM™ Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be ‘0’. This is in contrast to standard operation (extended instruction set disabled) when ‘a’ is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument, ‘d’, functions as before.

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.

## 24.2.4 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 PIC18FXX90, 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.



# PIC18F6390/6490/8390/8490

## ADDWF ADD W to Indexed (Indexed Literal Offset mode)

Syntax: ADDWF [k] {,d}

Operands:  $0 \leq k \leq 95$   
 $d \in \{0,1\}$   
 $a = 0$

Operation:  $(W) + ((FSR2) + k) \rightarrow dest$

Status Affected: N, OV, C, DC, Z

Encoding: 

0010	01d0	kkkk	kkkk
------	------	------	------

Description: The contents of W are added to the contents of the register indicated by FSR2, offset by the value 'k'.  
 If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process Data	Write to destination

**Example:** ADDWF [OFST], 0

Before Instruction	W	=	17h
	OFST	=	2Ch
	FSR2	=	0A00h
	Contents of 0A2Ch	=	20h
After Instruction	W	=	37h
	Contents of 0A2Ch	=	20h

## BSF Bit Set Indexed (Indexed Literal Offset mode)

Syntax: BSF [k], b

Operands:  $0 \leq f \leq 95$   
 $0 \leq b \leq 7$   
 $a = 0$

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

# PIC18F6390/6490/8390/8490

---

## 24.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set of the PIC18FXX90 family of devices. This includes the MPLAB C18 C compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default Configuration bits for that device. The default setting for the XINST Configuration bit is '0', disabling the extended instruction set and Indexed Literal Offset Addressing mode. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option, or dialog box within the environment, that allows the user to configure the language tool and its settings for the project
- A command line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

## 25.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK™ Object Linker/  
MPLIB™ Object Librarian
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PICSTART® Plus Development Programmer
  - MPLAB PM3 Device Programmer
  - PICKit™ 2 Development Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

## 25.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® 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.

# PIC18FXX90

---

## 25.2 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

## 25.3 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.

## 25.4 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

## 25.5 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

## 25.6 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.

## 25.7 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.

## 25.8 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.

## 25.9 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.

## 25.10 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.

# PIC18FXX90

---

## 25.11 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.

## 25.12 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.

## 25.13 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.

# PIC18F6390/6490/8390/8490

## 26.0 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 pin with respect to VSS (except VDD and $\overline{\text{MCLR}}$ ) .....	-0.3V to (VDD + 0.3V)
Voltage on VDD with respect to VSS .....	-0.3V to +7.5V
Voltage on $\overline{\text{MCLR}}$ with respect to VSS ( <b>Note 2</b> ) .....	0V to +13.25V
Total power dissipation ( <b>Note 1</b> ) .....	1.0W
Maximum current out of VSS pin .....	300 mA
Maximum current into VDD pin .....	250 mA
Input clamp current, I <sub>IK</sub> (V <sub>I</sub> < 0 or V <sub>I</sub> > VDD) .....	±20 mA
Output clamp current, I <sub>OK</sub> (V <sub>O</sub> < 0 or V <sub>O</sub> > VDD) .....	±20 mA
Maximum output current sunk by any I/O pin .....	25 mA
Maximum output current sourced by any I/O pin .....	25 mA
Maximum current sunk by all ports .....	200 mA
Maximum current sourced by all ports .....	200 mA

**Note 1:** Power dissipation is calculated as follows:

$$P_{\text{dis}} = V_{\text{DD}} \times \{I_{\text{DD}} - \sum I_{\text{OH}}\} + \sum \{(V_{\text{DD}} - V_{\text{OH}}) \times I_{\text{OH}}\} + \sum (V_{\text{OL}} \times I_{\text{OL}})$$

- 2:** Voltage spikes below VSS at the  $\overline{\text{MCLR}}/\text{VPP}/\text{RG5}$  pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a “low” level to the  $\overline{\text{MCLR}}/\text{VPP}/\text{RG5}$  pin, rather than pulling this pin directly to VSS.

† **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.

# PIC18F6390/6490/8390/8490

FIGURE 26-1: PIC18F6390/6490/8390/8490 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)

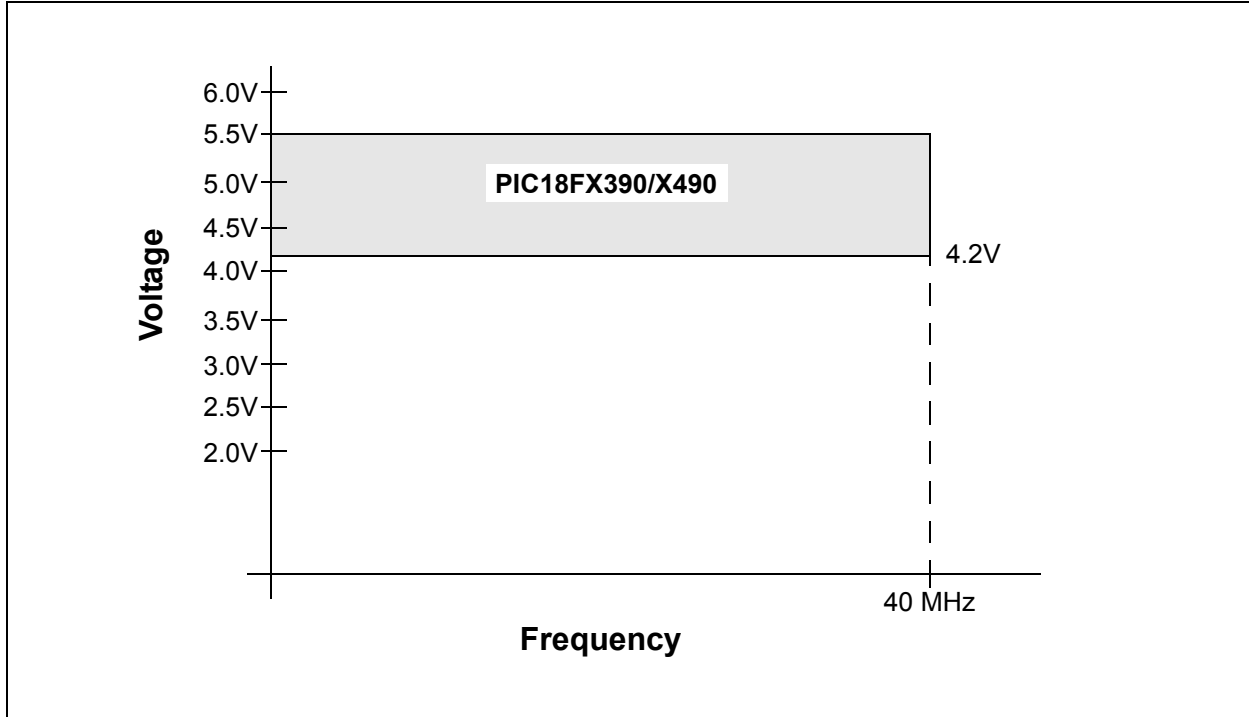
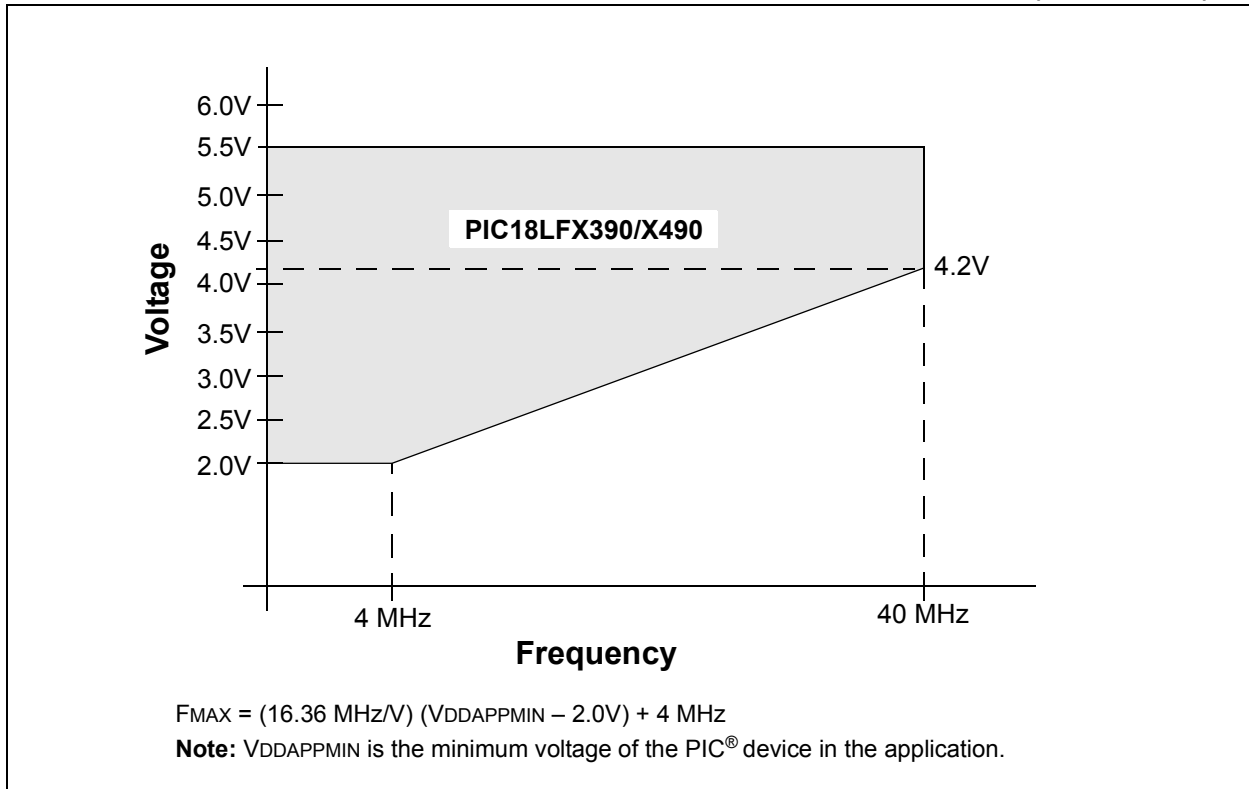


FIGURE 26-2: PIC18LF6390/6490/8390/8490 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)





# PIC18F6390/6490/8390/8490

## 26.1 DC Characteristics: Supply Voltage PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
D001	VDD	<b>Supply Voltage</b>					
		PIC18LF6390/6490/8390/8490	2.0	—	5.5	V	HS, XT, RC and LP Oscillator modes
		PIC18F6390/6490/8390/8490	4.2	—	5.5	V	
D002	VDR	<b>RAM Data Retention Voltage<sup>(1)</sup></b>	1.5	—	—	V	
D003	VPOR	<b>VDD Start Voltage</b> to Ensure Internal Power-on Reset Signal	—	—	0.7	V	See section on Power-on Reset for details
D004	SVDD	<b>VDD Rise Rate</b> to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	See section on Power-on Reset for details
D005	VBOR	<b>Brown-out Reset Voltage</b>					
		PIC18LF6390/6490/8390/8490					
		BORV1:BORV0 = 11	2.00	2.05	2.16	V	
D005		BORV1:BORV0 = 10	2.65	2.79	2.93	V	
		All devices					
		BORV1:BORV0 = 01	4.11	4.33	4.55	V	
		BORV1:BORV0 = 00	4.36	4.59	4.82	V	

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial			
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial			
Param No.	Device	Typ	Max	Units	Conditions
<b>Power-Down Current (<math>I_{PD}</math>)<sup>(1)</sup></b>					
PIC18LF6390/6490/8390/8490		0.1	1	$\mu\text{A}$	$-40^{\circ}\text{C}$
		0.1	1	$\mu\text{A}$	$+25^{\circ}\text{C}$
		0.2	5	$\mu\text{A}$	$+85^{\circ}\text{C}$
PIC18LF6390/6490/8390/8490		0.1	2	$\mu\text{A}$	$-40^{\circ}\text{C}$
		0.1	2	$\mu\text{A}$	$+25^{\circ}\text{C}$
		0.3	8	$\mu\text{A}$	$+85^{\circ}\text{C}$
All devices		0.1	2.0	$\mu\text{A}$	$-40^{\circ}\text{C}$
		0.1	2.0	$\mu\text{A}$	$+25^{\circ}\text{C}$
		0.4	15	$\mu\text{A}$	$+85^{\circ}\text{C}$

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all  $I_{PD}$  measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		12	26	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	FOSC = 31 kHz (RC_RUN mode, INTRC source)
		12	24	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		12	23	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		32	50	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		27	48	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		22	46	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		84	134	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 5.0V	
		82	128	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		72	128	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		.26	.8	mA	$-40^{\circ}\text{C}$	VDD = 2.0V	FOSC = 1 MHz (RC_RUN mode, INTOSC source)
		.26	.8	mA	$+25^{\circ}\text{C}$		
		.26	.8	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		.48	1.04	mA	$-40^{\circ}\text{C}$	VDD = 3.0V	
		.44	.96	mA	$+25^{\circ}\text{C}$		
		.48	.88	mA	$+85^{\circ}\text{C}$		
All devices		.88	1.84	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		.88	1.76	mA	$+25^{\circ}\text{C}$		
		.8	1.68	mA	$+85^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		0.6	1.7	mA	$-40^{\circ}\text{C}$	VDD = 2.0V	Fosc = 4 MHz (RC_RUN mode, INTOSC source)
		0.6	1.6	mA	$+25^{\circ}\text{C}$		
		0.6	1.5	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		1.0	2.4	mA	$-40^{\circ}\text{C}$	VDD = 3.0V	
		1.0	2.4	mA	$+25^{\circ}\text{C}$		
		1.0	2.4	mA	$+85^{\circ}\text{C}$		
All devices		2.0	4.2	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		2.0	4	mA	$+25^{\circ}\text{C}$		
		2.0	3.8	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		2.3	6.4	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	Fosc = 31 kHz (RC_IDLE mode, INTRC source)
		2.5	6.4	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		2.9	8.8	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		3.6	8.8	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		3.8	8.8	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		4.6	12	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		7.4	16	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 5.0V	
		7.8	16	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		9.1	29	$\mu\text{A}$	$+85^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		132	400	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	Fosc = 1 MHz (RC_IDLE mode, INTOSC source)
		140	400	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		152	400	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		200	600	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		216	600	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		252	600	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		0.40	1	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		0.42	1	mA	$+25^{\circ}\text{C}$		
		0.44	1	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		272	700	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	
		280	700	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		288	700	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		0.416	1	mA	$-40^{\circ}\text{C}$	VDD = 3.0V	
		0.432	1	mA	$+25^{\circ}\text{C}$		
		0.464	1	mA	$+85^{\circ}\text{C}$		
All devices		.8	1.6	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		.9	1.6	mA	$+25^{\circ}\text{C}$		
		.9	1.6	mA	$+85^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		250	500	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	Fosc = 1 MHz (PRI_RUN, EC oscillator)
		260	500	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		250	500	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		550	650	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		480	650	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		460	650	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		1.2	1.6	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		1.1	1.5	mA	$+25^{\circ}\text{C}$		
		1.0	1.4	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		0.72	2.0	mA	$-40^{\circ}\text{C}$	VDD = 2.0V	
		0.74	2.0	mA	$+25^{\circ}\text{C}$		
		0.74	2.0	mA	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		1.3	3.0	mA	$-40^{\circ}\text{C}$	VDD = 3.0V	
		1.3	3.0	mA	$+25^{\circ}\text{C}$		
		1.3	3.0	mA	$+85^{\circ}\text{C}$		
All devices		2.7	6.0	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		2.6	6.0	mA	$+25^{\circ}\text{C}$		
		2.5	6.0	mA	$+85^{\circ}\text{C}$		
All devices		15	35	mA	$-40^{\circ}\text{C}$	VDD = 4.2V	
		16	35	mA	$+25^{\circ}\text{C}$		
		16	35	mA	$+85^{\circ}\text{C}$		
All devices		21	40	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		21	40	mA	$+25^{\circ}\text{C}$		
		21	40	mA	$+85^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial				
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial				
Param No.	Device	Typ	Max	Units	Conditions	
<b>Supply Current (IDD)<sup>(2)</sup></b>						
	All devices	7.5	16	mA	$-40^{\circ}\text{C}$	VDD = 4.2V Fosc = 4 MHz, 16 MHz internal <b>(PRI_RUN HS+PLL)</b>
		7.4	15	mA	$+25^{\circ}\text{C}$	
		7.3	14	mA	$+85^{\circ}\text{C}$	
	All devices	10	21	mA	$-40^{\circ}\text{C}$	VDD = 5.0V Fosc = 4 MHz, 16 MHz internal <b>(PRI_RUN HS+PLL)</b>
		10	20	mA	$+25^{\circ}\text{C}$	
		9.7	19	mA	$+85^{\circ}\text{C}$	
	All devices	17	35	mA	$-40^{\circ}\text{C}$	VDD = 4.2V Fosc = 10 MHz, 40 MHz internal <b>(PRI_RUN HS+PLL)</b>
		17	35	mA	$+25^{\circ}\text{C}$	
		17	35	mA	$+85^{\circ}\text{C}$	
	All devices	23	40	mA	$-40^{\circ}\text{C}$	VDD = 5.0V Fosc = 10 MHz, 40 MHz internal <b>(PRI_RUN HS+PLL)</b>
		23	40	mA	$+25^{\circ}\text{C}$	
		23	40	mA	$+85^{\circ}\text{C}$	

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

$\overline{\text{OSC1}}$  = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

$\overline{\text{MCLR}}$  = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		59	117	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	Fosc = 1 MHz (PRI_IDLE mode, EC oscillator)
		59	108	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		63	104	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		108	243	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		108	225	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		117	216	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		270	432	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 5.0V	
		216	405	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		270	387	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		234	428	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	
		230	405	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		243	387	$\mu\text{A}$	$+85^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		378	810	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		387	765	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		405	729	$\mu\text{A}$	$+85^{\circ}\text{C}$		
All devices		.8	1.35	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		.8	1.26	mA	$+25^{\circ}\text{C}$		
		.8	1.17	mA	$+85^{\circ}\text{C}$		
All devices		5.4	14.4	mA	$-40^{\circ}\text{C}$	VDD = 4.2V	
		5.6	14.4	mA	$+25^{\circ}\text{C}$		
		5.9	14.4	mA	$+85^{\circ}\text{C}$		
All devices		7.3	16.2	mA	$-40^{\circ}\text{C}$	VDD = 5.0V	
		8.2	16.2	mA	$+25^{\circ}\text{C}$		
		7.5	16.2	mA	$+85^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.



# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
<b>Supply Current (IDD)<sup>(2)</sup></b>							
PIC18LF6390/6490/8390/8490		13	40	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	FOSC = 32 kHz ( <b>SEC_RUN</b> mode, Timer1 as clock) <sup>(4)</sup>
		14	40	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		16	40	$\mu\text{A}$	$+80^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		34	70	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		31	70	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		28	70	$\mu\text{A}$	$+80^{\circ}\text{C}$		
All devices		72	150	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 5.0V	
		65	150	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		59	150	$\mu\text{A}$	$+80^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		5.5	15	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 2.0V	FOSC = 32 kHz ( <b>SEC_IDLE</b> mode, Timer1 as clock) <sup>(4)</sup>
		5.8	15	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		6.1	18	$\mu\text{A}$	$+80^{\circ}\text{C}$		
PIC18LF6390/6490/8390/8490		8.2	30	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 3.0V	
		8.6	30	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		8.8	35	$\mu\text{A}$	$+80^{\circ}\text{C}$		
All devices		13	80	$\mu\text{A}$	$-40^{\circ}\text{C}$	VDD = 5.0V	
		13	80	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		13	85	$\mu\text{A}$	$+80^{\circ}\text{C}$		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

MCLR = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial					
Param No.	Device	Typ	Max	Units	Conditions		
D022 ( $\Delta I_{WDT}$ )	Watchdog Timer	Module Differential Currents ( $\Delta I_{WDT}$ , $\Delta I_{BOR}$ , $\Delta I_{LVD}$ , $\Delta I_{LCD}$ , $\Delta I_{OSCB}$ , $\Delta I_{AD}$ )					
		1.7	4	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 2.0V	
		2.1	4	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		2.6	5	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		2.2	6	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 3.0V	
		2.4	6	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		2.8	7	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		2.9	10	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 5.0V	
		3.1	10	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		3.3	13	$\mu\text{A}$	$+85^{\circ}\text{C}$		
D022A ( $\Delta I_{BOR}$ )	Brown-out Reset	17	50	$\mu\text{A}$	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$	V <sub>DD</sub> = 3.0V	
		42	60	$\mu\text{A}$	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$	V <sub>DD</sub> = 5.0V	
D022B ( $\Delta I_{LVD}$ )	High/Low-Voltage Detect	14	38	$\mu\text{A}$	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$	V <sub>DD</sub> = 2.0V	
		18	40	$\mu\text{A}$	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$	V <sub>DD</sub> = 3.0V	
		21	45	$\mu\text{A}$	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$	V <sub>DD</sub> = 5.0V	
D024 ( $\Delta I_{LCD}$ )	LCD Module	1.5	3	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 2.0V	LCD on INTRC clock, LCD segments enabled.
		1.5	3	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		1.7	4	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		2.2	5	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 3.0V	
		2.5	5	$\mu\text{A}$	$+25^{\circ}\text{C}$		
		2.7	6	$\mu\text{A}$	$+85^{\circ}\text{C}$		
		6.1	10	$\mu\text{A}$	$-40^{\circ}\text{C}$	V <sub>DD</sub> = 5.0V	
		6.5	10	$\mu\text{A}$	$+25^{\circ}\text{C}$		
7.2	10	$\mu\text{A}$	$+85^{\circ}\text{C}$				

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to 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.

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> or V<sub>SS</sub>;

MCLR = V<sub>DD</sub>; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.2 DC Characteristics: Power-Down and Supply Current PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (Industrial) (Continued)

PIC18LF6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial						
PIC18F6390/6490/8390/8490 (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial						
Param No.	Device	Typ	Max	Units	Conditions			
D025 ( $\Delta\text{IOSCB}$ )	Timer1 Oscillator	Module Differential Currents ( $\Delta\text{IWDT}$ , $\Delta\text{IBOR}$ , $\Delta\text{ILVD}$ , $\Delta\text{ILCD}$ , $\Delta\text{IOSCB}$ , $\Delta\text{IAD}$ )						
		1.0	3.5	$\mu\text{A}$	$-10^{\circ}\text{C}$	VDD = 2.0V	32 kHz on Timer1 <sup>(4)</sup>	
		1.1	3.5	$\mu\text{A}$	$+25^{\circ}\text{C}$			
		1.1	4.5	$\mu\text{A}$	$+70^{\circ}\text{C}$			
			1.2	4.5	$\mu\text{A}$	$-10^{\circ}\text{C}$	VDD = 3.0V	32 kHz on Timer1 <sup>(4)</sup>
			1.3	4.5	$\mu\text{A}$	$+25^{\circ}\text{C}$		
			1.2	5.5	$\mu\text{A}$	$+70^{\circ}\text{C}$		
			1.8	6.0	$\mu\text{A}$	$-10^{\circ}\text{C}$	VDD = 5.0V	32 kHz on Timer1 <sup>(4)</sup>
			1.9	6.0	$\mu\text{A}$	$+25^{\circ}\text{C}$		
			1.9	7.0	$\mu\text{A}$	$+70^{\circ}\text{C}$		
D026 ( $\Delta\text{IAD}$ )	A/D Converter	1.0	3.0	$\mu\text{A}$	—	VDD = 2.0V	A/D on, not converting	
		1.0	4.0	$\mu\text{A}$	—	VDD = 3.0V		
		1.0	8.0	$\mu\text{A}$	—	VDD = 5.0V		

**Legend:** Shading of rows is to assist in readability of the table.

**Note 1:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

**2:** The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

$\text{OSC1}$  = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD or VSS;

$\text{MCLR}$  = VDD; WDT enabled/disabled as specified.

**3:** Low-power Timer1 oscillator selected.

**4:** BOR and HLVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

# PIC18F6390/6490/8390/8490

## 26.3 DC Characteristics: PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (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 D032A D033	V <sub>IL</sub>	<b>Input Low Voltage</b> I/O Ports: with TTL Buffer  with Schmitt Trigger Buffer RC3 and RC4  $\overline{\text{MCLR}}$ OSC1 and T1OSI  OSC1	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> 0.3 V <sub>DD</sub> 0.2 V <sub>DD</sub> 0.3 V <sub>DD</sub> 0.2 V <sub>DD</sub>	V V V V V V V	V <sub>DD</sub> < 4.5V 4.5V ≤ V <sub>DD</sub> ≤ 5.5V  LP, XT, HS, HSPLL modes <sup>(1)</sup> EC mode <sup>(1)</sup>
D040 D040A D041 D042 D042A D043	V <sub>IH</sub>	<b>Input High Voltage</b> I/O Ports: with TTL Buffer  with Schmitt Trigger Buffer RC3 and RC4  $\overline{\text{MCLR}}$ OSC1 and T1OSI  OSC1	0.25 V <sub>DD</sub> + 0.8V 2.0 0.8 V <sub>DD</sub> 0.7 V <sub>DD</sub> 0.8 V <sub>DD</sub> 0.7 V <sub>DD</sub> 0.8 V <sub>DD</sub>	V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub> V <sub>DD</sub>	V V V V V V V	V <sub>DD</sub> < 4.5V 4.5V ≤ V <sub>DD</sub> ≤ 5.5V  LP, XT, HS, HSPLL modes <sup>(1)</sup> EC mode <sup>(1)</sup>
D060 D061 D063	I <sub>IL</sub>	<b>Input Leakage Current<sup>(2,3)</sup></b> I/O Ports  $\overline{\text{MCLR}}$ OSC1	— — —	±1 ±5 ±5	μA μA μA	V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> , Pin at hi-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 Weak Pull-up Current	50	400	μA	V <sub>DD</sub> = 5V, V <sub>PIN</sub> = V <sub>SS</sub>

- Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC<sup>®</sup> device be driven with an external clock while in RC mode.
- 2:** 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.
- 3:** Negative current is defined as current sourced by the pin.
- 4:** Parameter is characterized but not tested.

# PIC18F6390/6490/8390/8490

## 26.3 DC Characteristics: PIC18F6390/6490/8390/8490 (Industrial) PIC18LF6390/6490/8390/8490 (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	—	0.6	V	$I_{OL} = 8.5 \text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D083		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	—	0.6	V	$I_{OL} = 1.6 \text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D090	VOH	<b>Output High Voltage<sup>(3)</sup></b> I/O Ports	$V_{DD} - 0.7$	—	V	$I_{OH} = -3.0 \text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D092		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	$V_{DD} - 0.7$	—	V	$I_{OH} = -1.3 \text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D150	VOD	<b>Open-Drain High Voltage</b>	—	8.5	V	RA4 pin
<b>Capacitive Loading Specs on Output Pins</b>						
D100 <sup>(4)</sup>	COSC2	OSC2 pin	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1
D101	CIO	All I/O pins and OSC2 (in RC mode)	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	I <sup>2</sup> C™ Specification

- Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.
- 2:** 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.
- 3:** Negative current is defined as current sourced by the pin.
- 4:** Parameter is characterized but not tested.

# PIC18F6390/6490/8390/8490

**TABLE 26-1: 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
<b>Program Flash Memory</b>							
D110	VPP	Voltage on $\overline{\text{MCLR}}/\text{VPP}$ pin	10.0	—	12.0	V	
D113	IDDP	Supply Current during Programming	—	—	1	mA	
D130	EP	Cell Endurance	—	1K	—	E/W	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D131	VPR	VDD for Read	V <sub>MIN</sub>	—	5.5	V	V <sub>MIN</sub> = Minimum operating voltage
D132	VIE	VDD for Block Erase	2.75	—	5.5	V	Using ICSP™ port
D132A	VIW	VDD for Externally Timed Erase or Write	2.75	—	5.5	V	Using ICSP port
D132B	VPEW	VDD for Self-Timed Write	V <sub>MIN</sub>	—	5.5	V	V <sub>MIN</sub> = Minimum operating voltage
D133	TIE	ICSP Block Erase Cycle Time	—	4	—	ms	V <sub>DD</sub> > 4.5V
D133A	TIW	ICSP Erase or Write Cycle Time (externally timed)	2	—	—	ms	V <sub>DD</sub> > 4.5V
D133A	TIW	Self-Timed Write Cycle Time	—	2	—	ms	
D134	TRETD	Characteristic Retention	40	100	—	Year	Provided no other specifications are violated

† Data in “Typ” column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

# PIC18F6390/6490/8390/8490

**TABLE 26-2: COMPARATOR SPECIFICATIONS**

Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +85°C, unless otherwise stated.							
Param No.	Sym	Characteristics	Min	Typ	Max	Units	Comments
D300	VIOFF	Input Offset Voltage	—	±5.0	±10	mV	
D301	VICM	Input Common Mode Voltage*	0	—	VDD – 1.5	V	
D302	CMRR	Common Mode Rejection Ratio*	55	—	—	dB	
300	TRESP	Response Time*(1)	—	150	400	ns	PIC18FXXXX
300A			—	150	600	ns	PIC18LFXXXX, VDD = 2.0V
301	TMC2OV	Comparator Mode Change to Output Valid*	—	—	10	µs	

\* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at (VDD – 1.5)/2, while the other input transitions from VSS to VDD.

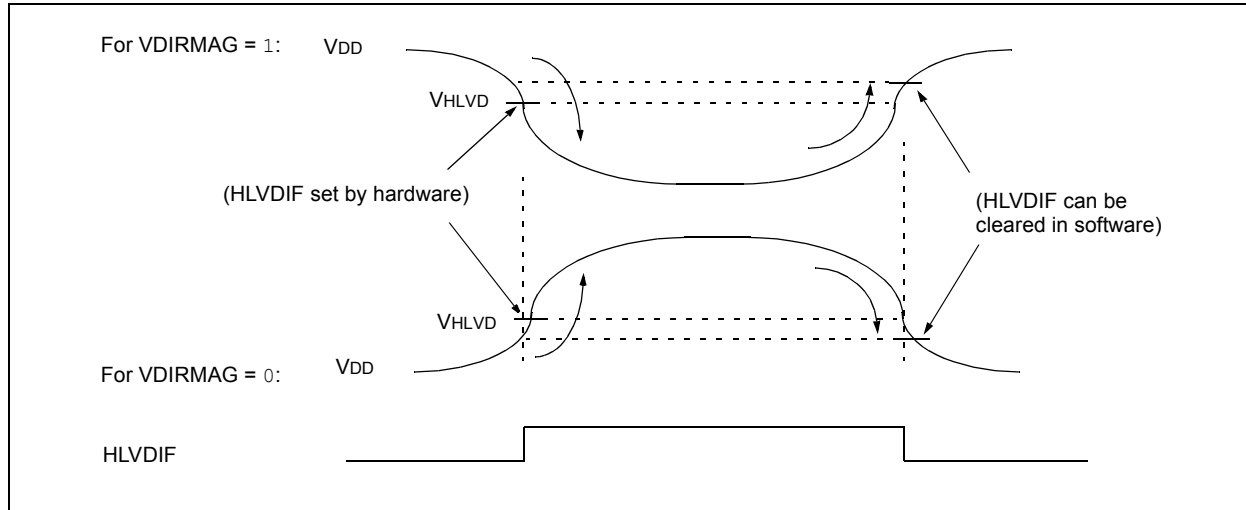
**TABLE 26-3: VOLTAGE REFERENCE SPECIFICATIONS**

Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +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(1)	—	—	10	µs	

**Note 1:** Settling time measured while CVRR = 1 and CVR3:CVR0 transitions from '0000' to '1111'.

# PIC18F6390/6490/8390/8490

**FIGURE 26-3: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS**



**TABLE 26-4: 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.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
D420		HLVD Voltage on VDD Transition High-to-Low	HLVDL<3:0> = 0000	2.06	2.17	2.28	V	
			HLVDL<3:0> = 0001	2.12	2.23	2.34	V	
			HLVDL<3:0> = 0010	2.24	2.36	2.48	V	
			HLVDL<3:0> = 0011	2.32	2.44	2.56	V	
			HLVDL<3:0> = 0100	2.47	2.60	2.73	V	
			HLVDL<3:0> = 0101	2.65	2.79	2.93	V	
			HLVDL<3:0> = 0110	2.74	2.89	3.04	V	
			HLVDL<3:0> = 0111	2.96	3.12	3.28	V	
			HLVDL<3:0> = 1000	3.22	3.39	3.56	V	
			HLVDL<3:0> = 1001	3.37	3.55	3.73	V	
			HLVDL<3:0> = 1010	3.52	3.71	3.90	V	
			HLVDL<3:0> = 1011	3.70	3.90	4.10	V	
			HLVDL<3:0> = 1100	3.90	4.11	4.32	V	
			HLVDL<3:0> = 1101	4.11	4.33	4.55	V	
HLVDL<3:0> = 1110	4.36	4.59	4.82	V				
D423	VBG	Band Gap Reference Voltage Value	HLVDL<3:0> = 1111	—	1.2	—	V	HLVD input external.

† Production tested at  $T_{AMB} = 25^{\circ}\text{C}$ . Specifications over temperature limits ensured by characterization.



# PIC18F6390/6490/8390/8490

## 26.4 AC (Timing) Characteristics

### 26.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I<sup>2</sup>C specifications only)
4. Ts (I<sup>2</sup>C specifications only)

<p>T</p> <p>F      Frequency</p>	<p>T      Time</p>
----------------------------------	--------------------

Lowercase letters (pp) and their meanings:

<p>pp</p> <p>cc      CCP1</p> <p>ck      CLKO</p> <p>cs      <math>\overline{CS}</math></p> <p>di      SDI</p> <p>do      SDO</p> <p>dt      Data in</p> <p>io      I/O port</p> <p>mc      <math>\overline{MCLR}</math></p>	<p>osc      OSC1</p> <p>rd      <math>\overline{RD}</math></p> <p>rw      <math>\overline{RD}</math> or <math>\overline{WR}</math></p> <p>sc      SCK</p> <p>ss      <math>\overline{SS}</math></p> <p>t0      T0CKI</p> <p>t1      T13CKI</p> <p>wr      <math>\overline{WR}</math></p>
--	--

Uppercase letters and their meanings:

<p>S</p> <p>F      Fall</p> <p>H      High</p> <p>I      Invalid (High-impedance)</p> <p>L      Low</p> <p>I<sup>2</sup>C only</p> <p>AA      output access</p> <p>BUF      Bus free</p>	<p>P      Period</p> <p>R      Rise</p> <p>V      Valid</p> <p>Z      High-impedance</p> <p>High    High</p> <p>Low     Low</p>
--	---

TCC:ST (I<sup>2</sup>C specifications only)

<p>CC</p> <p>HD      Hold</p> <p>ST</p> <p>DAT     DATA input hold</p> <p>STA     Start condition</p>	<p>SU      Setup</p> <p>STO     Stop condition</p>
---	--

# PIC18F6390/6490/8390/8490

## 26.4.2 TIMING CONDITIONS

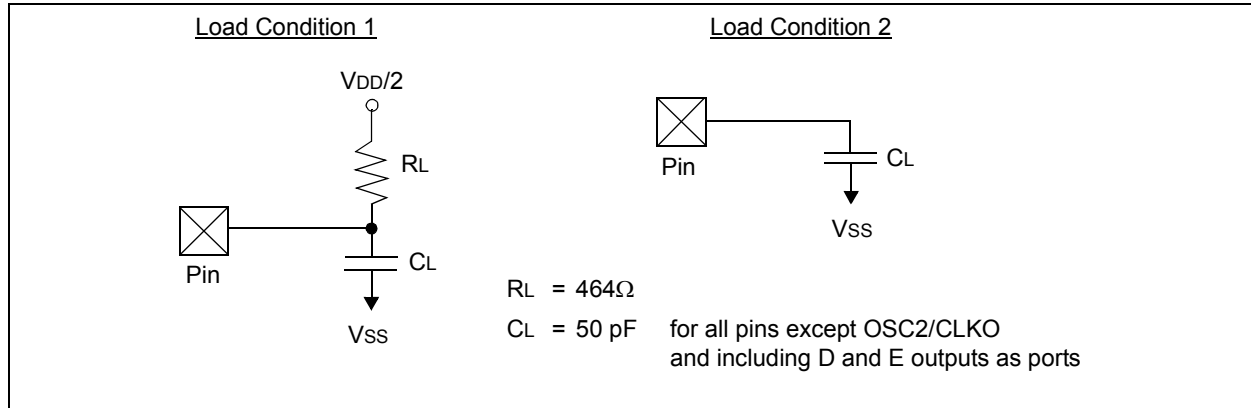
The temperature and voltages specified in Table 26-5 apply to all timing specifications unless otherwise noted. Figure 26-4 specifies the load conditions for the timing specifications.

**Note:** Because of space limitations, the generic terms “PIC18FXXXX” and “PIC18LFXXXX” are used throughout this section to refer to the PIC18F6390/6490/8390/8490 and PIC18LF6390/6490/8390/8490 families of devices specifically and only those devices.

**TABLE 26-5: 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 DC spec <b>Section 26.1</b> and <b>Section 26.3</b> .
	LF parts operate for industrial temperatures only.

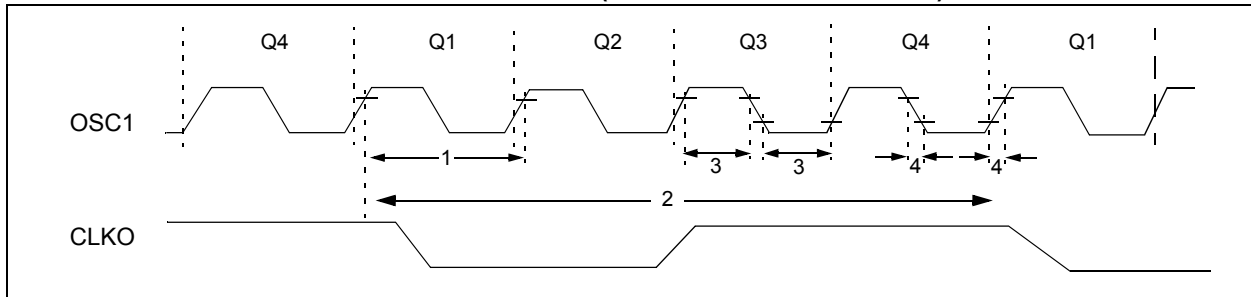
**FIGURE 26-4: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**



# PIC18F6390/6490/8390/8490

## 26.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

**FIGURE 26-5: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)**



**TABLE 26-6: EXTERNAL CLOCK TIMING REQUIREMENTS**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	Fosc	External CLKI Frequency <sup>(1)</sup>	DC	1	MHz	XT, RC Oscillator mode
			DC	20	MHz	HS Oscillator mode
			DC	31.25	kHz	LP Oscillator mode
		Oscillator Frequency <sup>(1)</sup>	DC	4	MHz	RC Oscillator mode
			0.1	4	MHz	XT Oscillator mode
			4	20	MHz	HS Oscillator mode
			5	200	kHz	LP Oscillator mode
1	Tosc	External CLKI Period <sup>(1)</sup>	1000	—	ns	XT, RC Oscillator mode
			50	—	ns	HS Oscillator mode
		Oscillator Period <sup>(1)</sup>	32	—	μs	LP Oscillator mode
			250	—	ns	RC Oscillator mode
			250	1	μs	XT Oscillator mode
			100	250	ns	HS Oscillator mode
			50	250	ns	HS Oscillator mode
			5	—	μs	LP Oscillator mode
2	Tcy	Instruction Cycle Time <sup>(1)</sup>	100	—	ns	Tcy = 4/Fosc
3	TosL, TosH	External Clock in (OSC1) High or Low Time	30	—	ns	XT Oscillator mode
			2.5	—	μs	LP Oscillator mode
			10	—	ns	HS Oscillator mode
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	20	ns	XT Oscillator mode
			—	50	ns	LP Oscillator mode
			—	7.5	ns	HS 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.

# PIC18F6390/6490/8390/8490

**TABLE 26-7: PLL CLOCK TIMING SPECIFICATIONS (V<sub>DD</sub> = 4.2V TO 5.5V)**

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
F10	FOSC	Oscillator Frequency Range	4	—	10	MHz	HS mode only
F11	FSYS	On-Chip VCO System Frequency	16	—	40	MHz	HS mode only
F12	t <sub>rc</sub>	PLL Start-up Time (Lock Time)	—	—	2	ms	
F13	ΔCLK	CLKO Stability (Jitter)	-2	—	+2	%	

† Data in “Typ” column is at 5V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

**TABLE 26-8: AC CHARACTERISTICS: INTERNAL RC ACCURACY  
PIC18LF6390/6490/8390/8490 (INDUSTRIAL)  
PIC18F6390/6490/8390/8490 (INDUSTRIAL)**

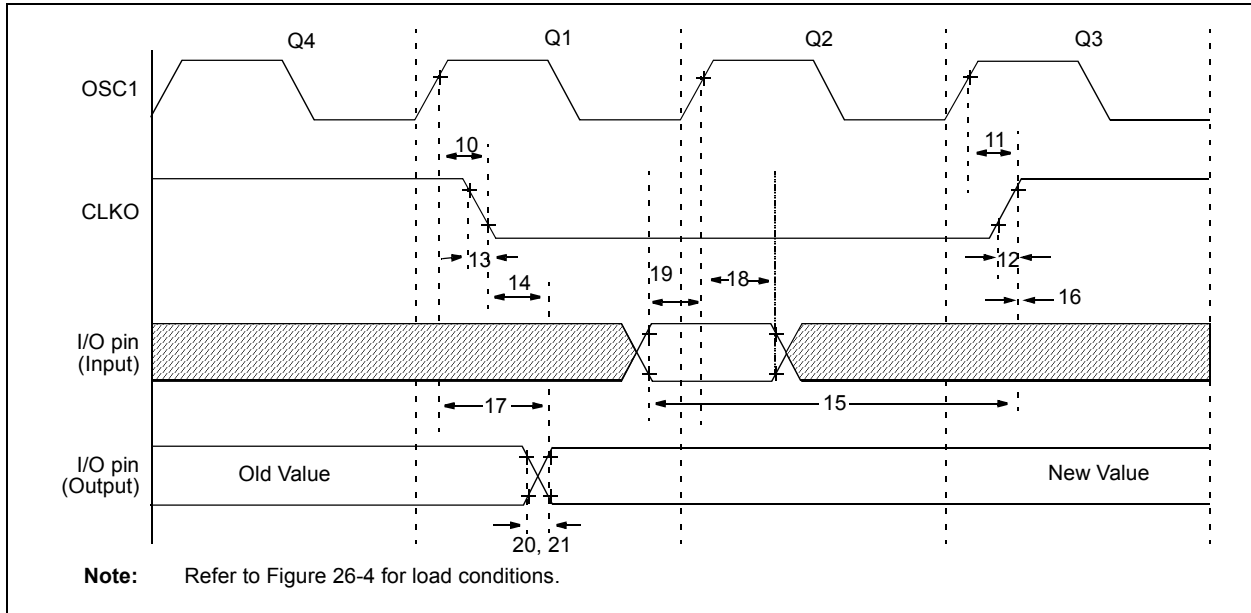
<b>PIC18LF6390/6490/8390/8490</b> (Industrial)		<b>Standard Operating Conditions (unless otherwise stated)</b> Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
<b>PIC18F6390/6490/8390/8490</b> (Industrial)		<b>Standard Operating Conditions (unless otherwise stated)</b> Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	Device	Min	Typ	Max	Units	Conditions	
<b>INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz<sup>(1)</sup></b>							
	PIC18LF6390/6490/8390/8490	-2	+/-1	2	%	+25°C	V <sub>DD</sub> = 2.7-3.3V
		-5	—	5	%	-10°C to +85°C	V <sub>DD</sub> = 2.7-3.3V
		-10	+/-1	10	%	-40°C to +85°C	V <sub>DD</sub> = 2.7-3.3V
	PIC18F6390/6490/8390/8490	-2	+/-1	2	%	+25°C	V <sub>DD</sub> = 4.5-5.5V
		-5	—	5	%	-10°C to +85°C	V <sub>DD</sub> = 4.5-5.5V
		-10	+/-1	10	%	-40°C to +85°C	V <sub>DD</sub> = 4.5-5.5V
<b>INTRC Accuracy @ Freq = 31 kHz<sup>(2)</sup></b>							
	<b>PIC18LF6390/6490/8390/8490</b>	26.562	—	35.938	kHz	-40°C to +85°C	V <sub>DD</sub> = 2.7-3.3V
	<b>PIC18F6390/6490/8390/8490</b>	26.562	—	35.938	kHz	-40°C to +85°C	V <sub>DD</sub> = 4.5-5.5V

**Legend:** Shading of rows is to assist in readability of the table.

- Note 1:** Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.  
**Note 2:** INTRC frequency after calibration.

# PIC18F6390/6490/8390/8490

**FIGURE 26-6: CLKO AND I/O TIMING**



**TABLE 26-9: CLKO AND I/O TIMING REQUIREMENTS**

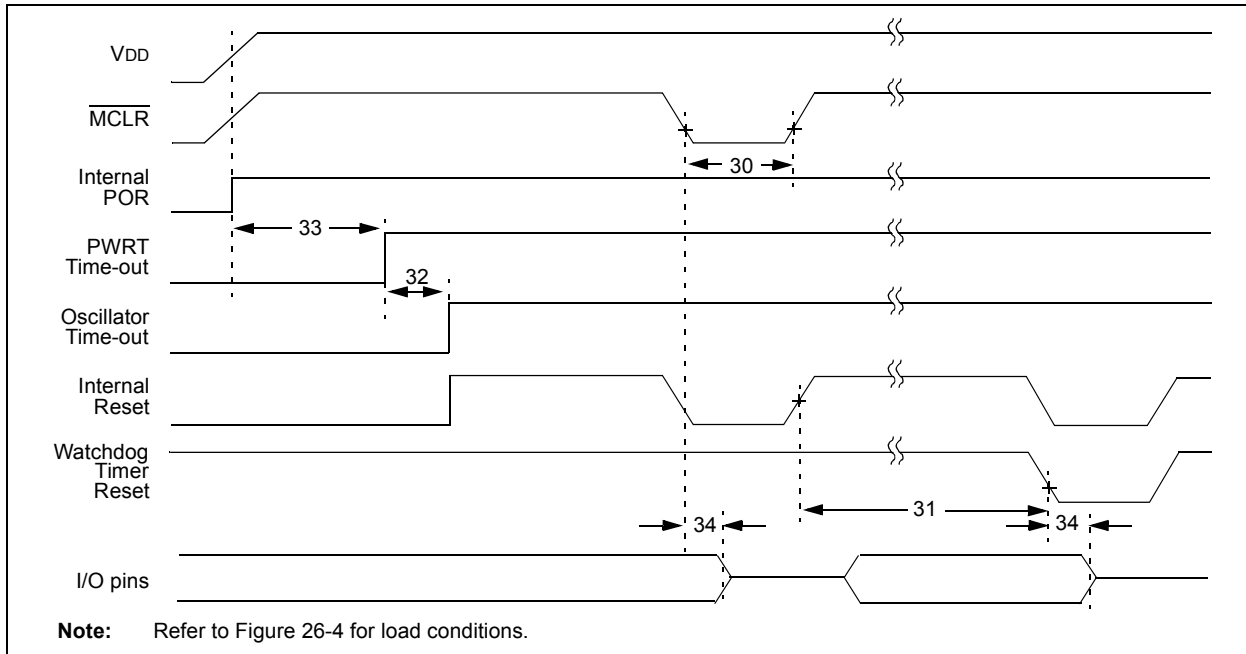
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions	
10	TosH2ckL	OSC1 ↑ to CLKO ↓	—	75	200	ns	(Note 1)	
11	TosH2ckH	OSC1 ↑ to CLKO ↑	—	75	200	ns	(Note 1)	
12	TckR	CLKO Rise Time	—	35	100	ns	(Note 1)	
13	TckF	CLKO Fall Time	—	35	100	ns	(Note 1)	
14	TckL2ioV	CLKO ↓ to Port Out Valid	—	—	0.5 T <sub>CY</sub> + 20	ns	(Note 1)	
15	TioV2ckH	Port In Valid before CLKO ↑	0.25 T <sub>CY</sub> + 25	—	—	ns	(Note 1)	
16	TckH2ioI	Port In Hold after CLKO ↑	0	—	—	ns	(Note 1)	
17	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port Out Valid	—	50	150	ns		
18	TosH2ioI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	PIC18FXXXX	100	—	—	ns	
18A			PIC18LFXXXX	200	—	—	ns	V <sub>DD</sub> = 2.0V
19	TioV2osH	Port Input Valid to OSC1 ↑ (I/O in setup time)	0	—	—	ns		
20	TioR	Port Output Rise Time	PIC18FXXXX	—	10	25	ns	
20A			PIC18LFXXXX	—	—	60	ns	V <sub>DD</sub> = 2.0V
21	TioF	Port Output Fall Time	PIC18FXXXX	—	10	25	ns	
21A			PIC18LFXXXX	—	—	60	ns	V <sub>DD</sub> = 2.0V
22†	TINP	INTx pin High or Low Time	T <sub>CY</sub>	—	—	ns		
23†	TRBP	RB7:RB4 Change INTx High or Low Time	T <sub>CY</sub>	—	—	ns		

† These parameters are asynchronous events not related to any internal clock edges.

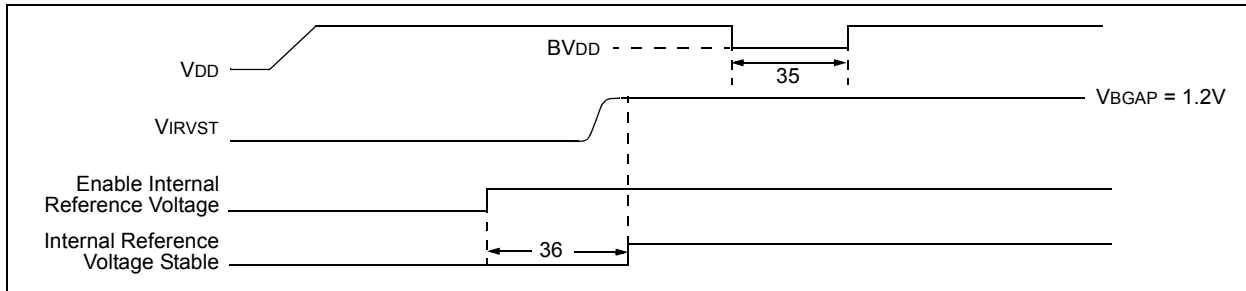
**Note 1:** Measurements are taken in RC mode, where CLKO output is 4 x T<sub>osc</sub>.

# PIC18F6390/6490/8390/8490

**FIGURE 26-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**FIGURE 26-8: BROWN-OUT RESET TIMING**

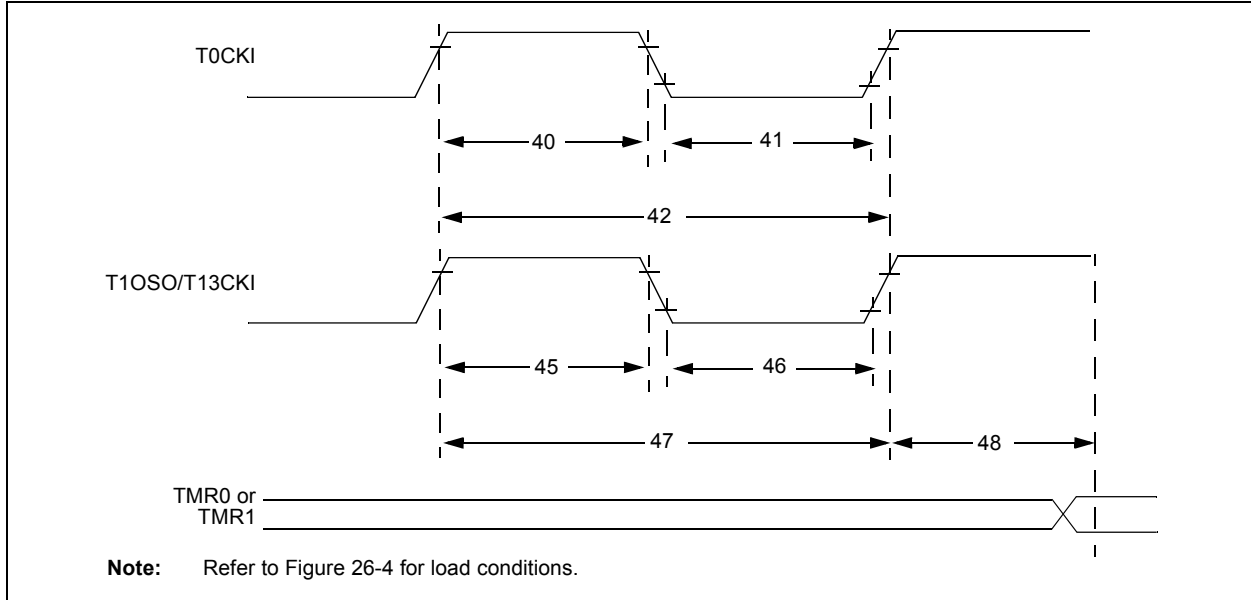


**TABLE 26-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS**

Param. No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	T <sub>MCL</sub>	MCLR Pulse Width (low)	2	—	—	μs	
31	T <sub>WDT</sub>	Watchdog Timer Time-out Period (No postscaler)	3.4	4.0	4.6	ms	
32	T <sub>OSt</sub>	Oscillator Start-up Timer Period	1024 T <sub>osc</sub>	—	1024 T <sub>osc</sub>	—	T <sub>osc</sub> = OSC1 period
33	T <sub>PWRT</sub>	Power-up Timer Period	55.5	65.5	75	ms	
34	T <sub>IOZ</sub>	I/O High-Impedance from MCLR Low or Watchdog Timer Reset	—	2	—	μs	
35	T <sub>BOR</sub>	Brown-out Reset Pulse Width	200	—	—	μs	V <sub>DD</sub> ≤ V <sub>BDD</sub> (see D005)
36	T <sub>IRVST</sub>	Time for Internal Reference Voltage to become Stable	—	20	50	μs	
37	T <sub>LVd</sub>	Low-Voltage Detect Pulse Width	200	—	—	μs	V <sub>DD</sub> ≤ V <sub>LVD</sub>
38	T <sub>CSD</sub>	CPU Start-up Time	—	10	—	μs	
39	T <sub>IOBST</sub>	Time for INTRC Block to Stabilize	—	1	—	ms	

# PIC18F6390/6490/8390/8490

**FIGURE 26-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**

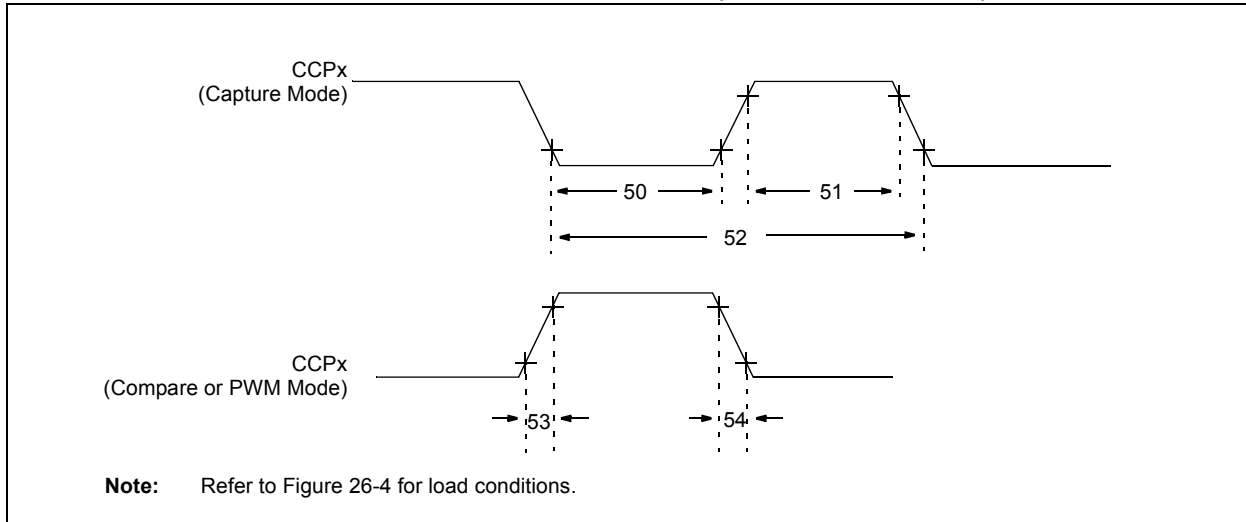


**TABLE 26-11: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions	
40	TT0H	T0CKI High Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	ns		
			With Prescaler	10	—	ns		
41	TT0L	T0CKI Low Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	ns		
			With Prescaler	10	—	ns		
42	TT0P	T0CKI Period	No Prescaler	$T_{CY} + 10$	—	ns	N = prescale value (1, 2, 4, ..., 256)	
			With Prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$	—	ns		
45	TT1H	T13CKI High Time	Synchronous, No Prescaler	$0.5 T_{CY} + 20$	—	ns	V <sub>DD</sub> = 2.0V	
			Synchronous, with Prescaler	PIC18FXXXX	10	—		ns
				PIC18LFXXXX	25	—		ns
			Asynchronous	PIC18FXXXX	30	—		ns
PIC18LFXXXX	50	—		ns				
46	TT1L	T13CKI Low Time	Synchronous, No Prescaler	$0.5 T_{CY} + 5$	—	ns	V <sub>DD</sub> = 2.0V	
			Synchronous, with Prescaler	PIC18FXXXX	10	—		ns
				PIC18LFXXXX	25	—		ns
			Asynchronous	PIC18FXXXX	30	—		ns
PIC18LFXXXX	50	—		ns				
47	TT1P	T13CKI Input Period	Synchronous	Greater of: 20 ns or $(T_{CY} + 40)/N$	—	ns	N = prescale value (1, 2, 4, 8)	
			Asynchronous	60	—	ns		
	FT1	T13CKI Oscillator Input Frequency Range		DC	50	kHz		
48	TCKE2TMR1	Delay from External T13CKI Clock Edge to Timer Increment		$2 T_{OSC}$	$7 T_{OSC}$	—		

# PIC18F6390/6490/8390/8490

**FIGURE 26-10: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)**



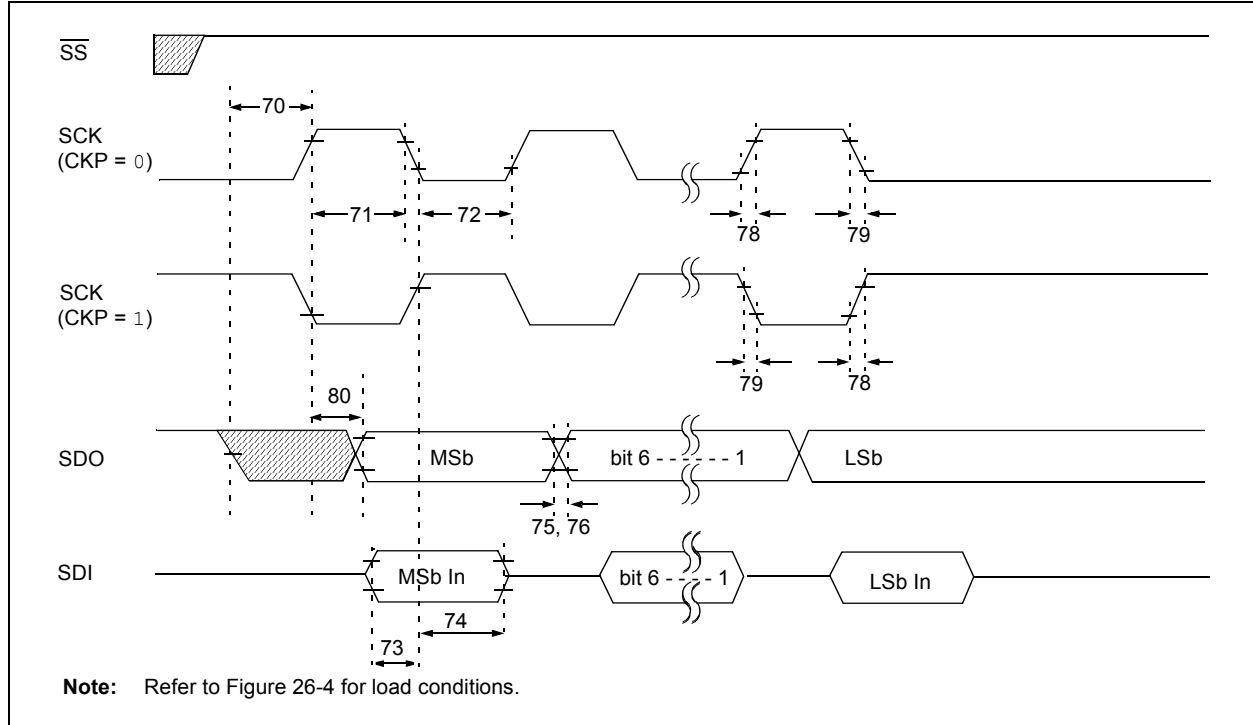
**TABLE 26-12: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)**

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
50	TccL	CCPx Input Low Time	No Prescaler	$0.5 T_{CY} + 20$	—	ns	VDD = 2.0V
			With Prescaler	PIC18FXXXXX	10	—	
			PIC18LFXXXXX	20	—	ns	
51	TccH	CCPx Input High Time	No Prescaler	$0.5 T_{CY} + 20$	—	ns	VDD = 2.0V
			With Prescaler	PIC18FXXXXX	10	—	
			PIC18LFXXXXX	20	—	ns	
52	TccP	CCPx Input Period		$\frac{3 T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time	PIC18FXXXXX	—	25	ns	VDD = 2.0V
			PIC18LFXXXXX	—	45	ns	
54	TccF	CCPx Output Fall Time	PIC18FXXXXX	—	25	ns	VDD = 2.0V
			PIC18LFXXXXX	—	45	ns	



# PIC18F6390/6490/8390/8490

**FIGURE 26-11: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)**



**TABLE 26-13: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)**

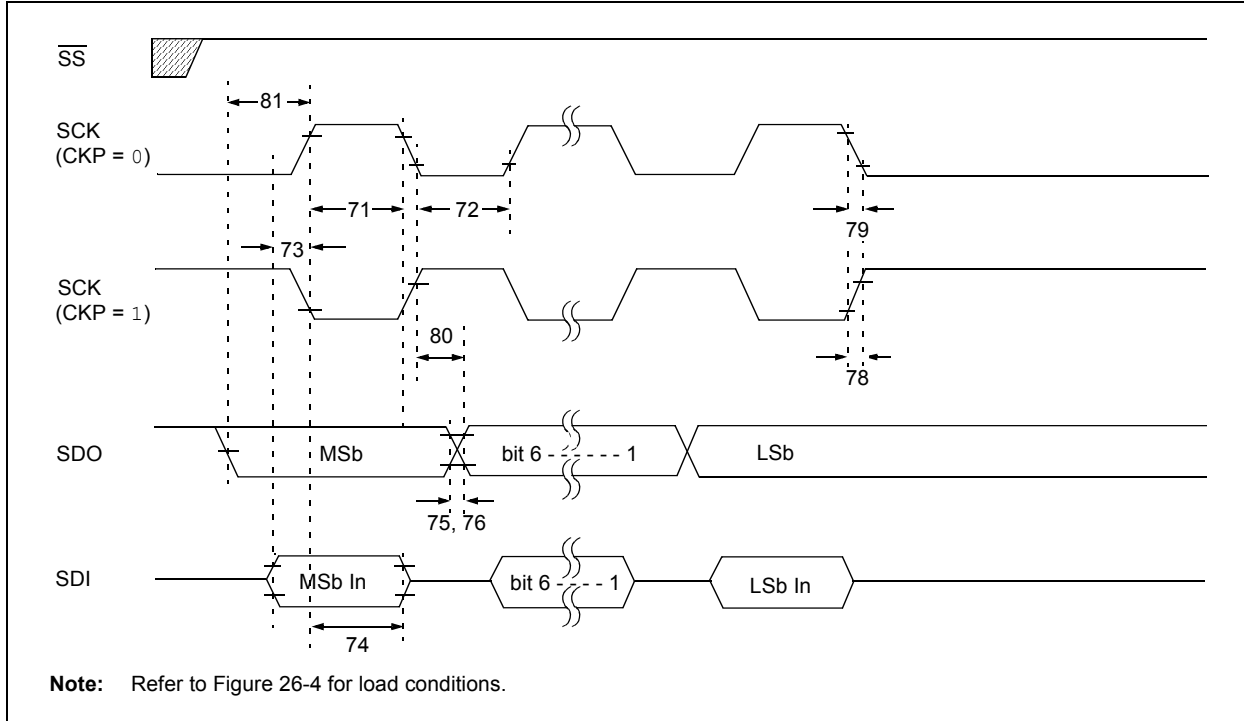
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2sCH, TssL2sCL	$\overline{SS} \downarrow$ to SCK $\downarrow$ or SCK $\uparrow$ Input	T <sub>cy</sub>	—	ns	
71	Tsch	SCK Input High Time (Slave mode)	Continuous	1.25 T <sub>cy</sub> + 30	—	ns
71A			Single Byte	40	—	ns
72	TscL	SCK Input Low Time (Slave mode)	Continuous	1.25 T <sub>cy</sub> + 30	—	ns
72A			Single Byte	40	—	ns
73	TdiV2sCH, TdiV2sCL	Setup Time of SDI Data Input to SCK Edge	100	—	ns	
73A	Tb2B	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 T <sub>cy</sub> + 40	—	ns	(Note 2)
74	Tsch2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge	100	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXXXX	—	25	ns
			PIC18LFXXXX	—	45	ns
76	TdoF	SDO Data Output Fall Time	—	25	ns	
78	TscR	SCK Output Rise Time (Master mode)	PIC18FXXXX	—	25	ns
			PIC18LFXXXX	—	45	ns
79	TscF	SCK Output Fall Time (Master mode)	—	25	ns	
80	Tsch2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXXXX	—	50	ns
			PIC18LFXXXX	—	100	ns

**Note 1:** Requires the use of Parameter #73A.

**Note 2:** Only if Parameter #71A and #72A are used.

# PIC18F6390/6490/8390/8490

**FIGURE 26-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)**



**TABLE 26-14: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)**

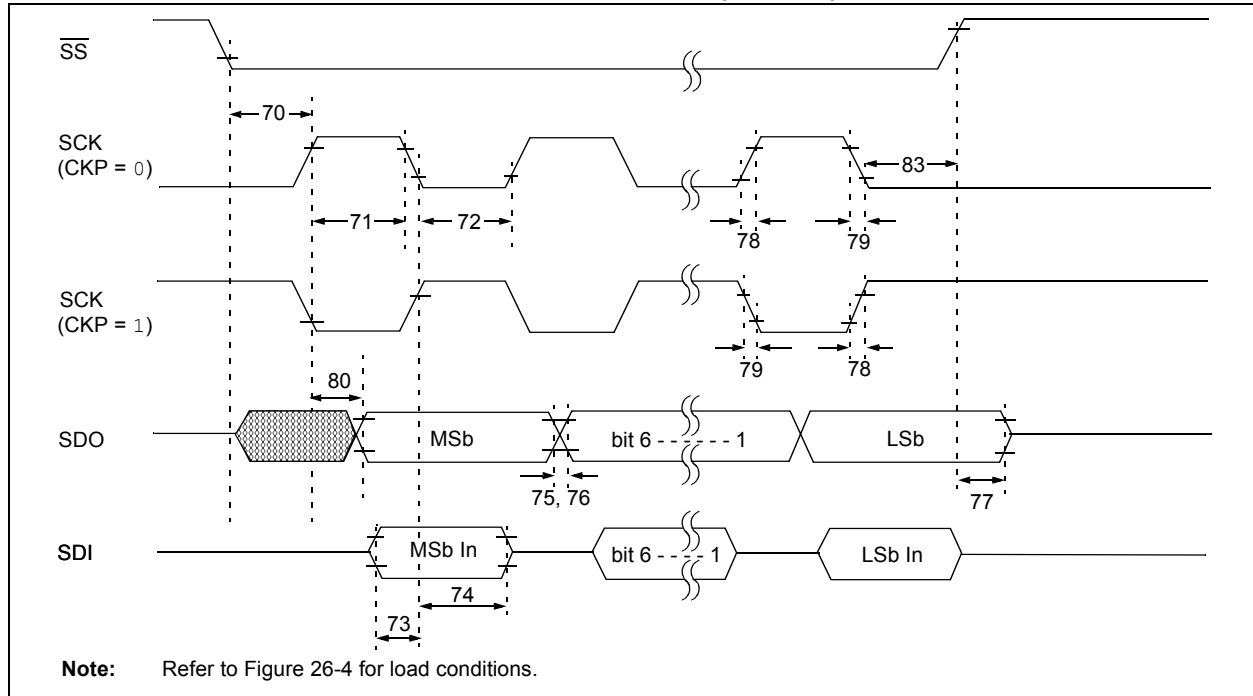
Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
71	Tsch	SCK Input High Time	1.25 Tcy + 30	—	ns	
71A		(Slave mode)				
		Continuous	40	—	ns	(Note 1)
		Single Byte	40	—	ns	(Note 1)
72	Tscl	SCK Input Low Time	1.25 Tcy + 30	—	ns	
72A		(Slave mode)				
		Continuous	40	—	ns	(Note 1)
		Single Byte	40	—	ns	(Note 1)
73	TdIV2sch, TdIV2scl	Setup Time of SDI Data Input to SCK Edge	100	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 Tcy + 40	—	ns	(Note 2)
74	Tsch2dIL, Tscl2dIL	Hold Time of SDI Data Input to SCK Edge	100	—	ns	
75	TdoR	SDO Data Output Rise Time				
		PIC18FXXXX	—	25	ns	
		PIC18LFXXXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO Data Output Fall Time	—	25	ns	
78	TscR	SCK Output Rise Time				
		(Master mode)				
		PIC18FXXXX	—	25	ns	
		PIC18LFXXXX	—	45	ns	VDD = 2.0V
79	TscF	SCK Output Fall Time (Master mode)	—	25	ns	
80	Tsch2doV, Tscl2doV	SDO Data Output Valid after SCK Edge				
		PIC18FXXXX	—	50	ns	
		PIC18LFXXXX	—	100	ns	VDD = 2.0V
81	TdoV2sch, TdoV2scl	SDO Data Output Setup to SCK Edge	Tcy	—	ns	

**Note 1:** Requires the use of Parameter #73A.

**Note 2:** Only if Parameter #71A and #72A are used.

# PIC18F6390/6490/8390/8490

**FIGURE 26-13: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)**



**TABLE 26-15: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)**

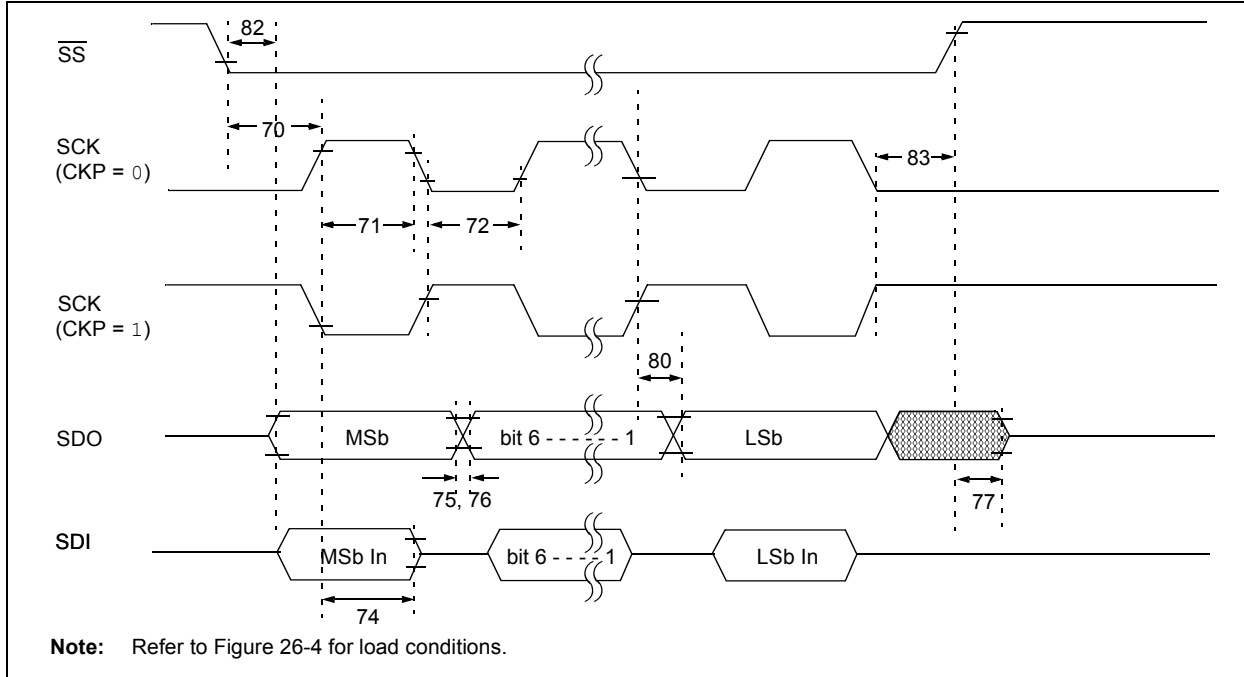
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2sCH, TssL2sCL	$\overline{SS} \downarrow$ to SCK $\downarrow$ or SCK $\uparrow$ Input	T <sub>CY</sub>	—	ns	
71	Tsch	SCK Input High Time (Slave mode)	Continuous Single Byte	1.25 T <sub>CY</sub> + 30 40	— —	ns ns (Note 1)
72	Tscl	SCK Input Low Time (Slave mode)	Continuous Single Byte	1.25 T <sub>CY</sub> + 30 40	— —	ns ns (Note 1)
73	TdIV2sCH, TdIV2sCL	Setup Time of SDI Data Input to SCK Edge	100	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2	1.5 T <sub>CY</sub> + 40	—	ns	(Note 2)
74	Tsch2dIL, Tscl2dIL	Hold Time of SDI Data Input to SCK Edge	100	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXXXX PIC18LFXXXX	— —	25 45	ns ns V <sub>DD</sub> = 2.0V
76	TdoF	SDO Data Output Fall Time	—	25	ns	
77	TssH2doZ	$\overline{SS} \uparrow$ to SDO Output High-Impedance	10	50	ns	
78	TscR	SCK Output Rise Time (Master mode)	PIC18FXXXX PIC18LFXXXX	— —	25 45	ns ns V <sub>DD</sub> = 2.0V
79	TscF	SCK Output Fall Time (Master mode)	—	25	ns	
80	Tsch2doV, Tscl2doV	SDO Data Output Valid after SCK Edge	PIC18FXXXX PIC18LFXXXX	— —	50 100	ns ns V <sub>DD</sub> = 2.0V
83	Tsch2ssH, Tscl2ssH	$\overline{SS} \uparrow$ after SCK Edge	1.5 T <sub>CY</sub> + 40	—	ns	

**Note 1:** Requires the use of Parameter #73A.

**2:** Only if Parameter #71A and #72A are used.

# PIC18F6390/6490/8390/8490

**FIGURE 26-14: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)**



**TABLE 26-16: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)**

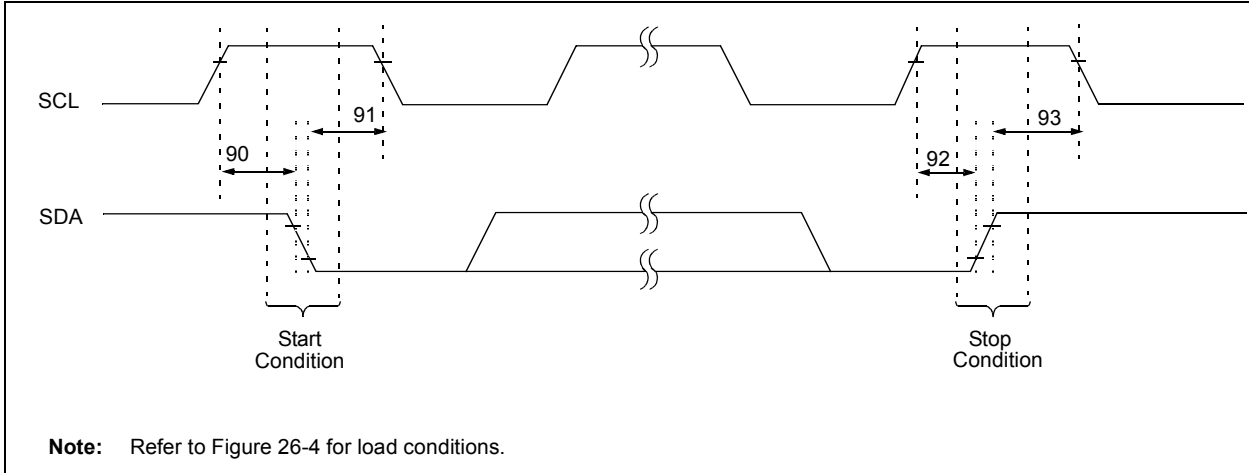
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
70	TssL2sch, TssL2scl	$\overline{SS} \downarrow$ to SCK $\downarrow$ or SCK $\uparrow$ Input	T <sub>cy</sub>	—	ns	
71	Tsch	SCK Input High Time (Slave mode)	Continuous	1.25 T <sub>cy</sub> + 30	—	ns
71A			Single Byte	40	—	ns
72	Tschl	SCK Input Low Time (Slave mode)	Continuous	1.25 T <sub>cy</sub> + 30	—	ns
72A			Single Byte	40	—	ns
73A	Tb2b	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2	1.5 T <sub>cy</sub> + 40	—	ns	(Note 2)
74	Tsch2diL, Tschl2diL	Hold Time of SDI Data Input to SCK Edge	100	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXXXX	—	25	ns
76			PIC18LFXXXX	—	45	ns
76	TdoF	SDO Data Output Fall Time	—	25	ns	
77	TssH2doZ	$\overline{SS} \uparrow$ to SDO Output High-Impedance	10	50	ns	
78	TscR	SCK Output Rise Time (Master mode)	PIC18FXXXX	—	25	ns
79			PIC18LFXXXX	—	45	ns
79	TscF	SCK Output Fall Time (Master mode)	—	25	ns	
80	Tsch2doV, Tschl2doV	SDO Data Output Valid after SCK Edge	PIC18FXXXX	—	50	ns
82			PIC18LFXXXX	—	100	ns
82	TssL2doV	SDO Data Output Valid after $\overline{SS} \downarrow$ Edge	PIC18FXXXX	—	50	ns
83			PIC18LFXXXX	—	100	ns
83	Tsch2ssh, Tschl2ssh	$\overline{SS} \uparrow$ after SCK Edge	1.5 T <sub>cy</sub> + 40	—	ns	

**Note 1:** Requires the use of Parameter #73A.

**Note 2:** Only if Parameter #71A and #72A are used.

# PIC18F6390/6490/8390/8490

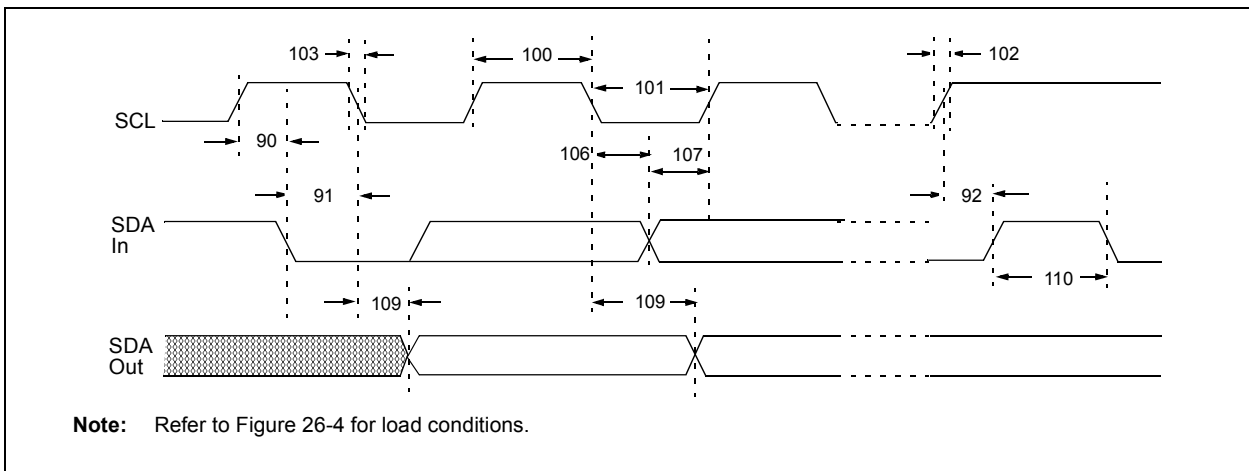
**FIGURE 26-15: I<sup>2</sup>C™ BUS START/STOP BITS TIMING**



**TABLE 26-17: 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 26-16: I<sup>2</sup>C™ BUS DATA TIMING**



# PIC18F6390/6490/8390/8490

**TABLE 26-18: I<sup>2</sup>C™ BUS DATA REQUIREMENTS (SLAVE MODE)**

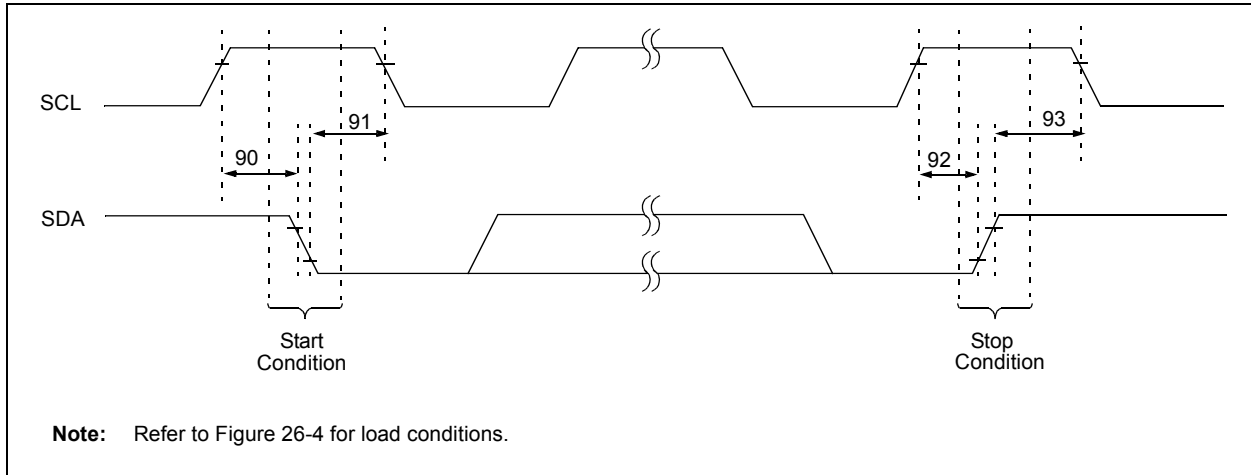
Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions	
100	THIGH	Clock High Time	100 kHz mode	4.0	—	μs	PIC18FXXXX must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	μs	PIC18FXXXX must operate at a minimum of 10 MHz
			MSSP module	1.5 Tcy	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	μs	PIC18FXXXX must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	μs	PIC18FXXXX must operate at a minimum of 10 MHz
			MSSP module	1.5 Tcy	—		
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 Cb	300	ns	Cb is specified to be from 10 to 400 pF
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 Cb	300	ns	Cb is specified to be from 10 to 400 pF
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	μs	
91	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	μs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	μs	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	μs	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	<b>(Note 2)</b>
			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	<b>(Note 1)</b>
			400 kHz mode	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
D102	CB	Bus Capacitive Loading	—	400	pF		

**Note 1:** As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

**2:** A Fast mode I<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 SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line, TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I<sup>2</sup>C bus specification), before the SCL line is released.

# PIC18F6390/6490/8390/8490

**FIGURE 26-17: MASTER SSP I<sup>2</sup>C™ BUS START/STOP BITS TIMING WAVEFORMS**

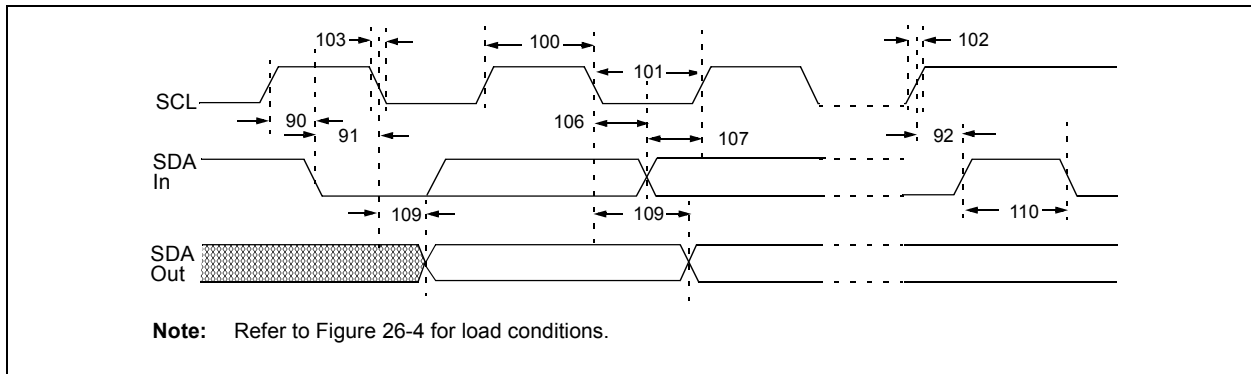


**TABLE 26-19: MASTER SSP I<sup>2</sup>C™ BUS START/STOP BITS REQUIREMENTS**

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 26-18: MASTER SSP I<sup>2</sup>C™ BUS DATA TIMING**



# PIC18F6390/6490/8390/8490

**TABLE 26-20: MASTER SSP I<sup>2</sup>C™ BUS DATA REQUIREMENTS**

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			400 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			1 MHz mode <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	ms	
101	TLOW	Clock Low Time	100 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			400 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			1 MHz mode <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	ms	
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_b$	300	ns	
			1 MHz mode <sup>(1)</sup>	—	300	ns	
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_b$	300	ns	
			1 MHz mode <sup>(1)</sup>	—	100	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	Only relevant for Repeated Start condition
			400 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			1 MHz mode <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	ms	
91	THD:STA	Start Condition Hold Time	100 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	After this period, the first clock pulse is generated
			400 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			1 MHz mode <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	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>	—	—	ns	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	<b>(Note 2)</b>
			400 kHz mode	100	—	ns	
			1 MHz mode <sup>(1)</sup>	—	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			400 kHz mode	$2(T_{osc})(BRG + 1)$	—	ms	
			1 MHz mode <sup>(1)</sup>	$2(T_{osc})(BRG + 1)$	—	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	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	ms	
			1 MHz mode <sup>(1)</sup>	—	—	ms	
D102	Cb	Bus Capacitive Loading		—	400	pF	

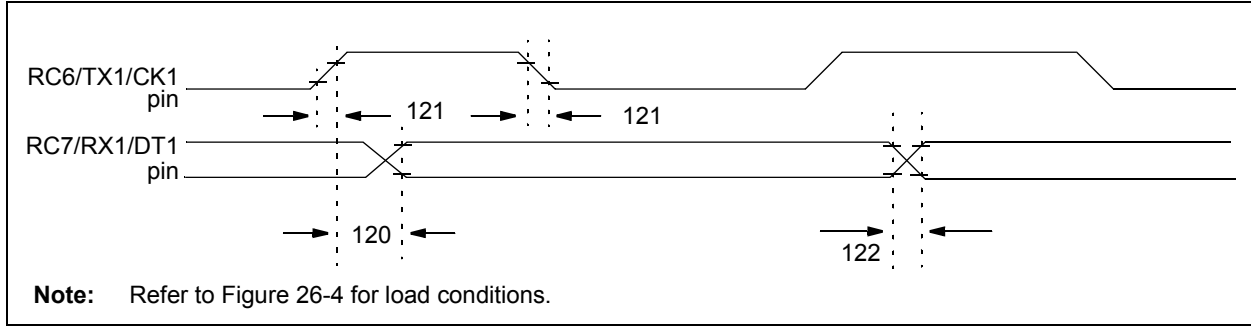
**Note 1:** Maximum pin capacitance = 10 pF for all I<sup>2</sup>C™ pins.

- 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 SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line, parameter #102 + parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode,) before the SCL line is released.



# PIC18F6390/6490/8390/8490

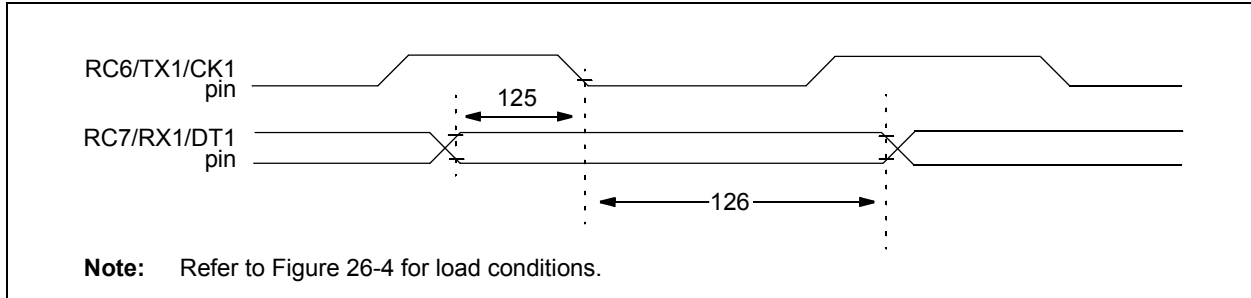
**FIGURE 26-19: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**



**TABLE 26-21: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
120	TckH2DTV	SYNC XMIT (MASTER and SLAVE) Clock High to Data Out Valid	PIC18FXXXX	—	40	ns
			PIC18LFXXXX	—	100	ns
121	TCKRF	Clock Out Rise Time and Fall Time (Master mode)	PIC18FXXXX	—	20	ns
			PIC18LFXXXX	—	50	ns
122	TDTRF	Data Out Rise Time and Fall Time	PIC18FXXXX	—	20	ns
			PIC18LFXXXX	—	50	ns

**FIGURE 26-20: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**



**TABLE 26-22: USART SYNCHRONOUS RECEIVE REQUIREMENTS**

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
125	TdtV2ckL	SYNC RCV (MASTER and SLAVE) Data Hold before CKx ↓ (DTx hold time)	10	—	ns	
126	TckL2DTL	Data Hold after CKx ↓ (DTx hold time)	15	—	ns	

# PIC18F6390/6490/8390/8490

**TABLE 26-23: A/D CONVERTER CHARACTERISTICS: PIC18F6390/6490/8390/8490 (INDUSTRIAL)  
PIC18LF6390/6490/8390/8490 (INDUSTRIAL)**

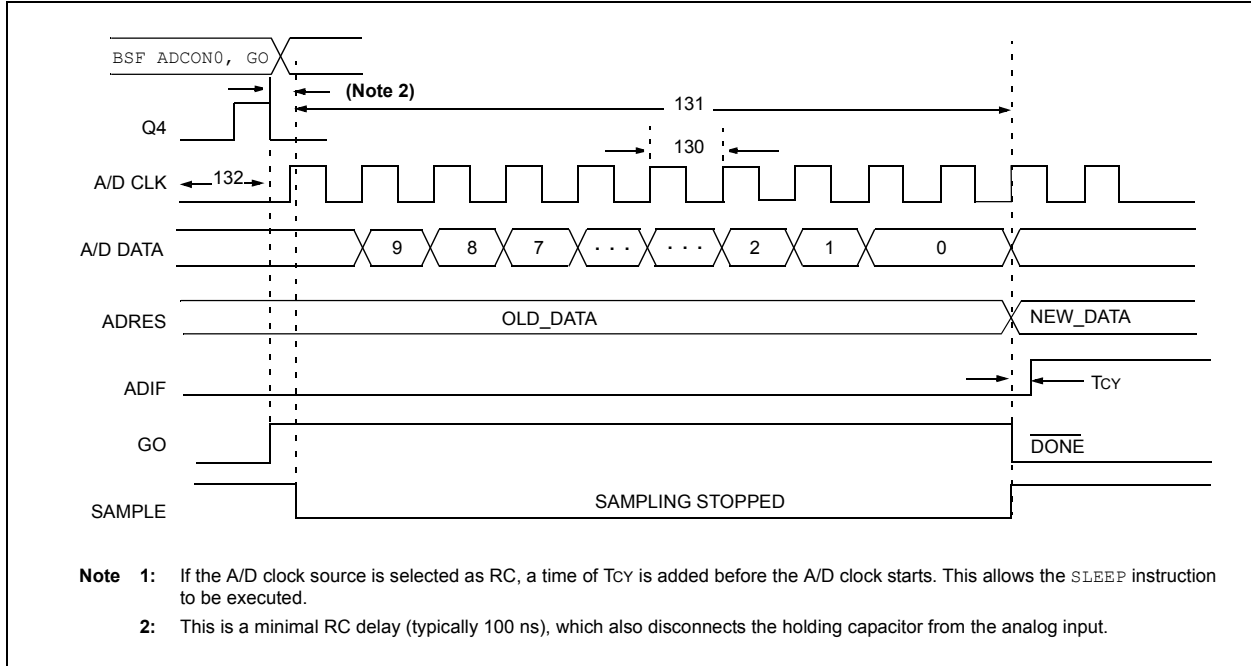
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
A01	NR	Resolution	—	—	10	bit	$\Delta V_{REF} \geq 3.0V$
A03	EIL	Integral Linearity Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A04	EDL	Differential Linearity Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A06	E <sub>OFF</sub>	Offset Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A07	E <sub>GN</sub>	Gain Error	—	—	$<\pm 1$	LSb	$\Delta V_{REF} \geq 3.0V$
A10	—	Monotonicity	Guaranteed <sup>(1)</sup>			—	
A20	$\Delta V_{REF}$	Reference Voltage Range (V <sub>REFH</sub> – V <sub>REFL</sub> )	3	—	AV <sub>DD</sub> – AV <sub>SS</sub>	V	For 10-bit resolution
A21	V <sub>REFH</sub>	Reference Voltage High	AV <sub>SS</sub> + 3.0V	—	AV <sub>DD</sub> + 0.3V	V	For 10-bit resolution
A22	V <sub>REFL</sub>	Reference Voltage Low	AV <sub>SS</sub> – 0.3V	—	AV <sub>DD</sub> – 3.0V	V	For 10-bit resolution
A25	V <sub>AIN</sub>	Analog Input Voltage	V <sub>REFL</sub>	—	V <sub>REFH</sub>	V	
A30	Z <sub>AIN</sub>	Recommended Impedance of Analog Voltage Source	—	—	2.5	k $\Omega$	
A50	I <sub>REF</sub>	V <sub>REF</sub> Input Current ( <b>Note 2</b> )	— —	— —	$\pm 5$ $\pm 150$	$\mu A$ $\mu A$	During V <sub>AIN</sub> 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:** V<sub>REFH</sub> current is from RA3/AN3/V<sub>REF+</sub>/SEG17 pin or AV<sub>DD</sub>, whichever is selected as the V<sub>REFH</sub> source.  
V<sub>REFL</sub> current is from RA2/AN2/V<sub>REF-</sub>/SEG16 pin or AV<sub>SS</sub>, whichever is selected as the V<sub>REFL</sub> source.

# PIC18F6390/6490/8390/8490

**FIGURE 26-21: A/D CONVERSION TIMING**



**TABLE 26-24: A/D CONVERSION REQUIREMENTS**

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions	
130	TAD	A/D Clock Period	PIC18FXXXX	0.7	25.0 <sup>(1)</sup>	$\mu$ S	TOSC based, VREF $\geq$ 3.0V
			PIC18LFXXXX	1.4	25.0 <sup>(1)</sup>	$\mu$ S	VDD = 2.0V; TOSC based, VREF full range
		PIC18FXXXX	TBD	1	$\mu$ S	A/D RC mode	
		PIC18LFXXXX	TBD	3	$\mu$ S	VDD = 2.0V; A/D RC mode	
131	T <sub>CV</sub>	Conversion Time (not including acquisition time) <b>(Note 2)</b>	11	12	TAD		
132	T <sub>ACQ</sub>	Acquisition Time <b>(Note 3)</b>	1.4	—	$\mu$ S	-40°C to +85°C	
			TBD	—	$\mu$ S	0°C $\leq$ to $\leq$ +85°C	
135	T <sub>SWC</sub>	Switching Time from Convert $\rightarrow$ Sample	—	<b>(Note 4)</b>			
TBD	T <sub>DIS</sub>	Discharge Time	0.2	—	$\mu$ S		

**Legend:** TBD = To Be Determined

- Note 1:** The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.
- Note 2:** ADRES register may be read on the following  $T_{CY}$  cycle.
- Note 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 ( $R_S$ ) on the input channels is 50 $\Omega$ .
- Note 4:** On the following cycle of the device clock.

# PIC18F6390/6490/8390/8490

---

NOTES:

## 27.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Graphs and tables are not available at this time.

# PIC18F6390/6490/8390/8490

---

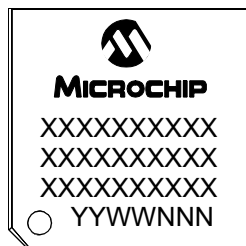
NOTES:

# PIC18F6390/6490/8390/8490

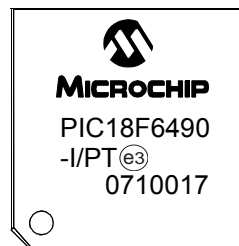
## 28.0 PACKAGING INFORMATION

### 28.1 Package Marking Information

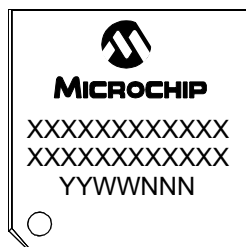
64-Lead TQFP



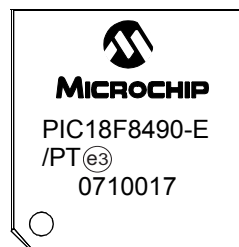
Example



80-Lead TQFP



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
	<sup>(e3)</sup>	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ( <sup>(e3)</sup> ) can be found on the outer packaging for this package.

**Note:** In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

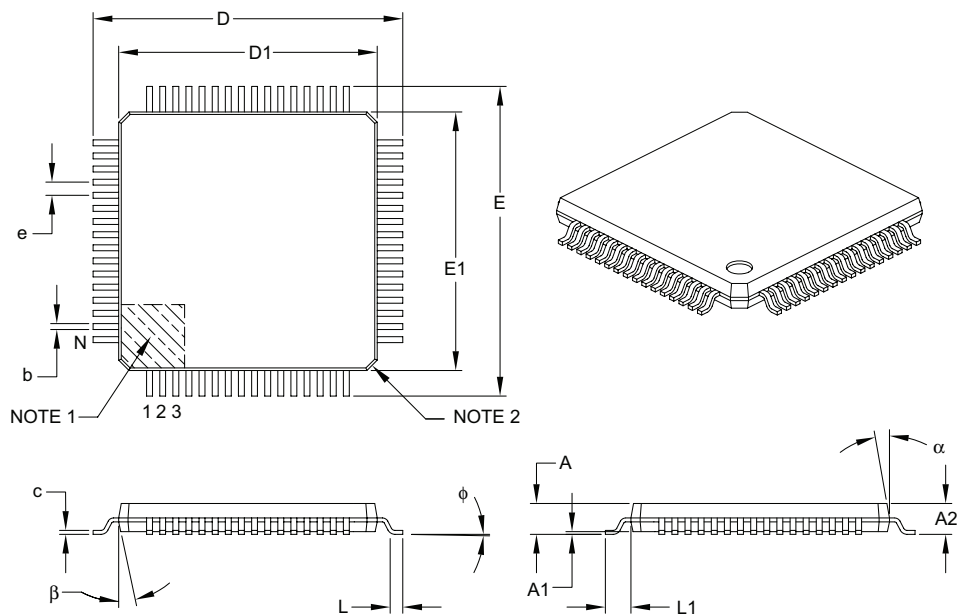
# PIC18F6390/6490/8390/8490

## 28.2 Package Details

The following sections give the technical details of the packages.

### 64-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	64		
Lead Pitch	e	0.50 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	0.17	0.22	0.27
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.

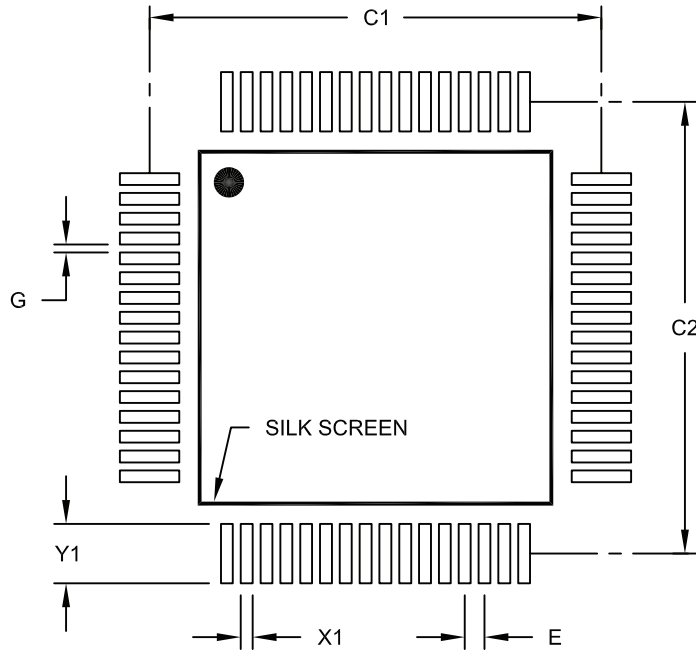
Microchip Technology Drawing C04-085B



# PIC18F6390/6490/8390/8490

## 64-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

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.50 BSC		
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			1.50
Distance Between Pads	G	0.20		

**Notes:**

1. Dimensioning and tolerancing per ASME Y14.5M

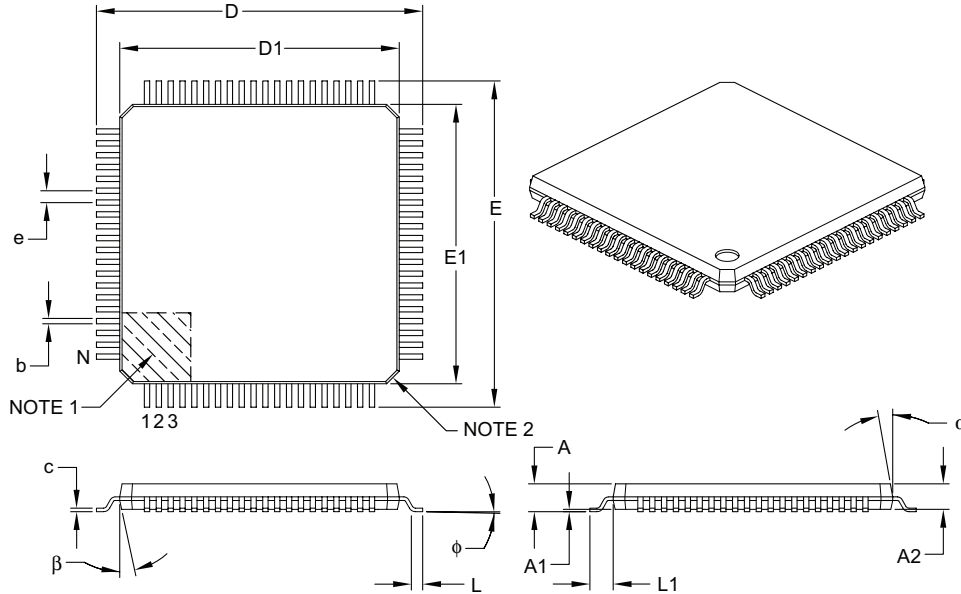
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2085A

# PIC18F6390/6490/8390/8490

## 80-Lead Plastic Thin Quad Flatpack (PT) – 12x12x1 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	80		
Lead Pitch	e	0.50 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	14.00 BSC		
Overall Length	D	14.00 BSC		
Molded Package Width	E1	12.00 BSC		
Molded Package Length	D1	12.00 BSC		
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
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.

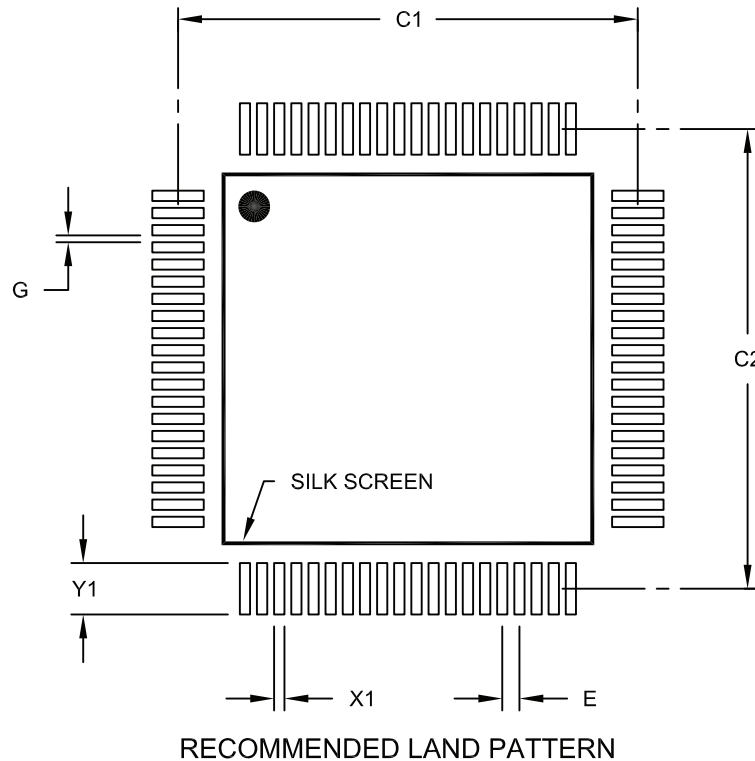
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-092B

# PIC18F6390/6490/8390/8490

## 80-Lead Plastic Thin Quad Flatpack (PT) – 12x12x1 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>



		Units	MILLIMETERS		
		Dimension Limits	MIN	NOM	MAX
Contact Pitch	E		0.50 BSC		
Contact Pad Spacing	C1			13.40	
Contact Pad Spacing	C2			13.40	
Contact Pad Width (X80)	X1				0.30
Contact Pad Length (X80)	Y1				1.50
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. C04-2092A

# PIC18F6390/6490/8390/8490

---

NOTES:

# PIC18F6390/6490/8390/8490

## APPENDIX A: REVISION HISTORY

### Revision A (July 2004)

Original data sheet for PIC18F6390/6490/8390/8490 devices.

### Revision B (August 2004)

Updated preliminary “electrical characteristics” data.

### Revision C (November 2007)

Revised I<sup>2</sup>C™ Slave Mode Timing figure. Updated DC Power-Down and Supply Current table and package drawings.

## APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

**TABLE B-1: DEVICE DIFFERENCES**

Features	PIC18F6390	PIC18F6490	PIC18F8390	PIC18F8490
Number of Pixels the LCD Driver can Drive	128 (4 x 32)	128 (4 x 32)	192 (4 x 48)	192 (4 x 48)
I/O Ports	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
Flash Program Memory	8 Kbytes	16 Kbytes	8 Kbytes	16 Kbytes
Packages	64-Pin TQFP	64-Pin TQFP	80-Pin TQFP	80-Pin TQFP

# PIC18F6390/6490/8390/8490

---

## APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

**Not Applicable**

## APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a Baseline device (i.e., PIC16C5X) to an Enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

**Not Currently Available**

## **APPENDIX E: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES**

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in AN716, "Migrating Designs from PIC16C74A/74B to PIC18C442." The changes discussed, while device specific, are generally applicable to all mid-range to enhanced device migrations.

This Application Note is available as Literature Number DS00716.

## **APPENDIX F: MIGRATION FROM HIGH-END TO ENHANCED DEVICES**

A detailed discussion of the migration pathway and differences between the high-end MCU devices (i.e., PIC17CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in AN726, "PIC17CXXX to PIC18CXXX Migration." This Application Note is available as Literature Number DS00726.

# PIC18F6390/6490/8390/8490

---

NOTES:



# PIC18F6390/6490/8390/8490

## INDEX

### A

A/D	231
A/D Converter Interrupt, Configuring	235
Acquisition Requirements	236
ADCON0 Register	231
ADCON1 Register	231
ADCON2 Register	231
ADRESH Register	231, 234
ADRESL Register	231
Analog Port Pins, Configuring	238
Associated Registers	240
Automatic Acquisition Time	237
Configuring the Module	235
Conversion Clock (TAD)	237
Conversion Status (GO/DONE Bit)	234
Conversions	239
Converter Characteristics	384
Discharge	239
Operation in Power-Managed Modes	238
Special Event Trigger (CCP)	240
Use of the CCP2 Trigger	240
Absolute Maximum Ratings	349
AC (Timing) Characteristics	367
Load Conditions for Device Timing	
Specifications	368
Parameter Symbology	367
Temperature and Voltage Specifications	368
Timing Conditions	368
Access Bank	73
Mapping with Indexed Literal Offset Mode	86
ACKSTAT	187
ACKSTAT Status Flag	187
ADCON0 Register	231
GO/DONE Bit	234
ADCON1 Register	231
ADCON2 Register	231
ADDFSR	338
ADDLW	301
Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART). See AUSART.	
ADDULNK	338
ADDWF	301
ADDWFC	302
ADRESH Register	231
ADRESL Register	231, 234
Analog-to-Digital Converter. See A/D.	
ANDLW	302
ANDWF	303
Assembler	
MPASM Assembler	346
AUSART	
Asynchronous Mode	222
Associated Registers, Receive	225
Associated Registers, Transmit	223
Receiver	224
Setting up 9-Bit Mode with	
Address Detect	224
Transmitter	222

Baud Rate Generator (BRG)	220
Associated Registers	220
Baud Rate Error, Calculating	220
Baud Rates, Asynchronous Modes	221
High Baud Rate Select (BRGH Bit)	220
Operation in Power-Managed Modes	220
Sampling	220
Synchronous Master Mode	226
Associated Registers, Receive	228
Associated Registers, Transmit	227
Reception	228
Transmission	226
Synchronous Slave Mode	229
Associated Registers, Receive	230
Associated Registers, Transmit	229
Reception	230
Transmission	229
Auto-Wake-up on Sync Break Character	210

### B

Bank Select Register (BSR)	71
Baud Rate Generator	183
BC	303
BCF	304
BF	187
BF Status Flag	187
Block Diagrams	
A/D	234
Analog Input Model	235
AUSART Receive	224
AUSART Transmit	222
Baud Rate Generator	183
Capture Mode Operation	150
Comparator Analog Input Model	245
Comparator I/O Operating Modes	242
Comparator Output	244
Comparator Voltage Reference	248
Compare Mode Operation	151
Device Clock	36
EUSART Receive	208
EUSART Transmit	206
External Power-on Reset Circuit (Slow VDD Power-up)	53
Fail-Safe Clock Monitor	290
Generic I/O Port Operation	109
HLVD Module (with External Input)	252
Interrupt Logic	94
LCD Clock Generation	262
LCD Driver Module	257
LCD Resistor Ladder Connection	263
MSSP (I <sup>2</sup> C Master Mode)	181
MSSP (I <sup>2</sup> C Mode)	166
MSSP (SPI Mode)	157
On-Chip Reset Circuit	51
PLL (HS Mode)	33
PWM Operation (Simplified)	153
Reads from Flash Program Memory	88
Single Comparator	243
Table Read Operation	87
Timer0 in 16-Bit Mode	132

# PIC18F6390/6490/8390/8490

Timer0 in 8-Bit Mode .....	132	Initializing PORTA .....	109
Timer1 .....	136	Initializing PORTB .....	112
Timer1 (16-Bit Read/Write Mode) .....	136	Initializing PORTC .....	115
Timer2 .....	142	Initializing PORTD .....	118
Timer3 .....	144	Initializing PORTE .....	120
Timer3 (16-Bit Read/Write Mode) .....	144	Initializing PORTF .....	122
Voltage Reference Output Buffer Example .....	249	Initializing PORTG .....	125
Watchdog Timer .....	287	Initializing PORTH .....	127
BN .....	304	Initializing PORTJ .....	129
BNC .....	305	Loading the SSPBUF (SSPSR) Register .....	160
BNN .....	305	Reading a Flash Program Memory Word .....	89
BNOV .....	306	Saving STATUS, WREG and BSR	
BNZ .....	306	Registers in RAM .....	108
BOR. See Brown-out Reset.		Code Protection .....	281
BOV .....	309	COMF .....	312
BRA .....	307	Comparator .....	241
Break Character (12-Bit) Transmit and Receive .....	211	Analog Input Connection Considerations .....	245
BRG. See Baud Rate Generator.		Associated Registers .....	245
Brown-out Reset (BOR) .....	54, 281	Configuration .....	242
Disabling in Sleep Mode .....	54	Effects of a Reset .....	244
BSF .....	307	Interrupts .....	244
BSR .....	86	Operation .....	243
BTFSC .....	308	Operation During Sleep .....	244
BTFSS .....	308	Outputs .....	243
BTG .....	309	Reference .....	243
BZ .....	310	External Signal .....	243
		Internal Signal .....	243
<b>C</b>		Response Time .....	243
C Compilers		Comparator Specifications .....	365
MPLAB C18 .....	346	Comparator Voltage Reference .....	247
MPLAB C30 .....	346	Accuracy and Error .....	248
CALL .....	310	Associated Registers .....	249
CALLW .....	339	Configuring .....	247
Capture (CCP Module) .....	150	Connection Considerations .....	248
Associated Registers .....	152	Effects of a Reset .....	248
CCP Pin Configuration .....	150	Operation During Sleep .....	248
CCPR2H:CCPR2L Registers .....	150	Compare (CCP Module) .....	151
Software Interrupt .....	150	Associated Registers .....	152
Timer1/Timer3 Mode Selection .....	150	CCP Pin Configuration .....	151
Capture/Compare/PWM (CCP) .....	147	CCPR2 Register .....	151
Capture Mode. See Capture.		Software Interrupt .....	151
CCP Mode and Timer Resources .....	148	Special Event Trigger .....	145, 151, 240
CCPRxH Register .....	148	Timer1/Timer3 Mode Selection .....	151
CCPRxL Register .....	148	Computed GOTO .....	68
Compare Mode. See Compare.		Configuration Bits .....	281
Interaction of CCP1 and CCP2 for		Configuration Register Protection .....	292
Timer Resources .....	149	Context Saving During Interrupts .....	108
Interconnect Configurations .....	148	Conversion Considerations .....	396
Module Configuration .....	148	CPFSEQ .....	312
Clock Sources .....	36	CPFSGT .....	313
Selecting the 31 kHz Source .....	37	CPFSLT .....	313
Selection Using OSCCON Register .....	37	Crystal Oscillator/Ceramic Resonator .....	31
CLRf .....	311	Customer Change Notification Service .....	409
CLRWDT .....	311	Customer Notification Service .....	409
Code Examples		Customer Support .....	409
16 x 16 Signed Multiply Routine .....	92		
16 x 16 Unsigned Multiply Routine .....	92	<b>D</b>	
8 x 8 Signed Multiply Routine .....	91	Data Addressing Modes .....	81
8 x 8 Unsigned Multiply Routine .....	91	Comparing Addressing Modes with the	
Changing Between Capture Prescalers .....	150	Extended Instruction Set Enabled .....	85
Computed GOTO Using an Offset Value .....	68	Direct .....	81
Fast Register Stack .....	68	Indexed Literal Offset .....	84
How to Clear RAM (Bank 1) Using		Indirect .....	81
Indirect Addressing .....	81	Inherent and Literal .....	81
Implementing a Real-Time Clock Using			
a Timer1 Interrupt Service .....	139		

# PIC18F6390/6490/8390/8490

Data Memory .....	71	Extended Instruction Set	
Access Bank .....	73	ADDFSR .....	338
and the Extended Instruction Set .....	84	ADDLNLK .....	338
Bank Select Register (BSR) .....	71	CALLW .....	339
General Purpose Registers .....	73	MOVSF .....	339
Map for PIC18F6X90/8X90 Devices .....	72	MOVSS .....	340
Special Function Registers .....	74	PUSHL .....	340
DAW .....	314	SUBFSR .....	341
DC and AC Characteristics		SUBLNLK .....	341
Graphs and Tables .....	387	External Clock Input .....	32
DC Characteristics .....	362	<b>F</b>	
Power-Down and Supply Current .....	352	Fail-Safe Clock Monitor .....	281, 290
Supply Voltage .....	351	Interrupts in Power-Managed Modes .....	291
DCFSNZ .....	315	POR or Wake from Sleep .....	291
DECF .....	314	WDT During Oscillator Failure .....	290
DECFSZ .....	315	Fast Register Stack .....	68
Development Support .....	345	Firmware Instructions .....	295
Device Differences .....	395	Flash Program Memory .....	87
Device Overview .....	7	Associated Registers .....	89
Features (table) .....	9	Control Registers .....	88
New Core Features .....	7	TABLAT (Table Latch) Register .....	88
Special Features .....	8	TBLPTR (Table Pointer) Register .....	88
Direct Addressing .....	82	Reading .....	88
<b>E</b>		Table Reads .....	87
Effect on Standard Instructions .....	84	FSCM. See Fail-Safe Clock Monitor.	
Effect on Standard PIC MCU Instructions .....	342	<b>G</b>	
Electrical Characteristics .....	349	GOTO .....	316
Enhanced Universal Synchronous Asynchronous		<b>H</b>	
Receiver Transmitter (EUSART). See EUSART.		Hardware Multiplier .....	91
Equations		Introduction .....	91
A/D Acquisition Time .....	236	Operation .....	91
A/D Minimum Charging Time .....	236	Performance Comparison .....	91
Calculating the Minimum Required		High/Low-Voltage Detect .....	251
Acquisition Time .....	236	Applications .....	254
Errata .....	5	Typical Low-Voltage Detect (diagram) .....	254
EUSART		Associated Registers .....	255
Asynchronous Mode .....	206	Characteristics .....	366
12-Bit Break Transmit and Receive .....	211	Current Consumption .....	253
Associated Registers, Receive .....	209	Effects of a Reset .....	255
Associated Registers, Transmit .....	207	Operation .....	252
Auto-Wake-up on Sync Break .....	210	During Sleep .....	255
Receiver .....	208	Setup .....	253
Setting up 9-Bit Mode with		Start-up Time .....	253
Address Detect .....	208	HLVD. See High/Low-Voltage Detect. ....	251
Transmitter .....	206	<b>I</b>	
Baud Rate Generator (BRG) .....	201	I/O Ports .....	109
Associated Registers .....	201	I <sup>2</sup> C Mode (MSSP)	
Auto-Baud Rate Detect .....	204	Acknowledge Sequence Timing .....	190
Baud Rate Error, Calculating .....	201	Associated Registers .....	196
Baud Rates, Asynchronous Modes .....	202	Baud Rate Generator .....	183
High Baud Rate Select (BRGH Bit) .....	201	Bus Collision	
Operation in Power-Managed Modes .....	201	During a Repeated Start Condition .....	194
Sampling .....	201	During a Start Condition .....	192
Synchronous Master Mode .....	212	During a Stop Condition .....	195
Associated Registers, Receive .....	214	Clock Arbitration .....	184
Associated Registers, Transmit .....	213	Clock Stretching .....	176
Reception .....	214	10-Bit Slave Receive Mode (SEN = 1) .....	176
Transmission .....	212	10-Bit Slave Transmit Mode .....	176
Synchronous Slave Mode .....	215	7-Bit Slave Receive Mode (SEN = 1) .....	176
Associated Registers, Receive .....	216	7-Bit Slave Transmit Mode .....	176
Associated Registers, Transmit .....	215		
Reception .....	216		
Transmission .....	215		

# PIC18F6390/6490/8390/8490

Effect of a Reset .....	191	DAW .....	314
General Call Address Support .....	180	DCFSNZ .....	315
I <sup>2</sup> C Clock Rate w/BRG .....	183	DECFSZ .....	314
Master Mode .....	181	DECFSZ .....	315
Operation .....	182	Extended Instructions .....	337
Reception .....	187	and Using MPLAB IDE Tools .....	344
Repeated Start Condition Timing .....	186	Considerations when Enabling .....	342
Start Condition .....	185	Syntax .....	337
Transmission .....	187	General Format .....	297
Transmit Sequence .....	182	GOTO .....	316
Multi-Master Communication, Bus Collision		INCF .....	316
and Arbitration .....	191	INCFSZ .....	317
Multi-Master Mode .....	191	INFSNZ .....	317
Operation .....	170	IORLW .....	318
Read/Write Bit Information (R/W Bit) .....	170, 171	IORWF .....	318
Registers .....	166	LFSR .....	319
Serial Clock (RC3/SCK/SCL) .....	171	MOVF .....	319
Slave Mode .....	170	MOVFF .....	320
Addressing .....	170	MOVLB .....	320
Reception .....	171	MOVLW .....	321
Sleep Operation .....	191	MOVWF .....	321
Stop Condition Timing .....	190	MULLW .....	322
Transmission .....	171	MULWF .....	322
ID Locations .....	281, 293	NEGF .....	323
INCF .....	316	NOP .....	323
INCFSZ .....	317	POP .....	324
In-Circuit Debugger .....	293	PUSH .....	324
In-Circuit Serial Programming (ICSP) .....	281, 293	RCALL .....	325
Indexed Literal Offset Addressing Mode .....	342	RESET .....	325
and Standard PIC18 Instructions .....	342	RETFIE .....	326
Indexed Literal Offset Mode .....	84, 86	RETLW .....	326
Indirect Addressing .....	82	RETURN .....	327
INFSNZ .....	317	RLCF .....	327
Initialization Conditions for all Registers .....	59–64	RLNCF .....	328
Instruction Cycle .....	69	RRCF .....	328
Clocking Scheme .....	69	RRNCF .....	329
Instruction Flow/Pipelining .....	69	SETF .....	329
Instruction Set .....	295	SETF (Indexed Literal Offset mode) .....	343
ADDLW .....	301	SLEEP .....	330
ADDWF .....	301	Standard Instructions .....	295
ADDWF (Indexed Literal Offset mode) .....	343	SUBFWB .....	330
ADDWFC .....	302	SUBLW .....	331
ANDLW .....	302	SUBWF .....	331
ANDWF .....	303	SUBWFB .....	332
BC .....	303	SWAPF .....	332
BCF .....	304	TBLRD .....	333
BN .....	304	TBLWT .....	334
BNC .....	305	TSTFSZ .....	335
BNN .....	305	XORLW .....	335
BNOV .....	306	XORWF .....	336
BNZ .....	306	Summary Table .....	298
BOV .....	309	INTCON Register	
BRA .....	307	RBIF Bit .....	112
BSF .....	307	INTCON Registers .....	95
BSF (Indexed Literal Offset mode) .....	343	Inter-Integrated Circuit. <i>See</i> I <sup>2</sup> C.	
BTFSC .....	308	Internal Oscillator Block .....	34
BTFSS .....	308	Adjustment .....	34
BTG .....	309	INTIO Modes .....	34
BZ .....	310	INTOSC Output Frequency .....	34
CALL .....	310	OSCTUNE Register .....	34
CLRf .....	311	Internal RC Oscillator	
CLRWDT .....	311	Use with WDT .....	287
COMF .....	312	Internet Address .....	409
CPFSEQ .....	312		
CPFSGT .....	313		
CPFSLT .....	313		

# PIC18F6390/6490/8390/8490

Interrupt Sources .....	281	MOVWF .....	321
A/D Conversion Complete .....	235	MPLAB ASM30 Assembler, Linker, Librarian .....	346
Capture Complete (CCP) .....	150	MPLAB ICD 2 In-Circuit Debugger .....	347
Compare Complete (CCP) .....	151	MPLAB ICE 2000 High-Performance Universal In-Circuit Emulator .....	347
Interrupt-on-Change (RB7:RB4) .....	112	MPLAB Integrated Development Environment Software .....	345
INTx Pin .....	108	MPLAB PM3 Device Programmer .....	347
PORTB, Interrupt-on-Change .....	108	MPLAB REAL ICE In-Circuit Emulator System .....	347
TMR0 .....	108	MPLINK Object Linker/MPLIB Object Librarian .....	346
TMR0 Overflow .....	133	MSSP ACK Pulse .....	170, 171
TMR1 Overflow .....	135	Control Registers (general) .....	157
TMR2 to PR2 Match (PWM) .....	153	I <sup>2</sup> C Mode. See I <sup>2</sup> C Mode.	
TMR3 Overflow .....	143, 145	Module Overview .....	157
Interrupts .....	93	SPI Master/Slave Connection .....	161
Interrupts, Flag Bits		SPI Mode. See SPI Mode.	
Interrupt-on-Change (RB7:RB4) Flag (RBIF Bit) .....	112	SSPBUF .....	162
INTOSC Frequency Drift .....	34	SSPSR .....	162
INTOSC, INTRC. See Internal Oscillator Block.		MULLW .....	322
IORLW .....	318	MULWF .....	322
IORWF .....	318		
IPR Registers .....	104	<b>N</b>	
		NEGf .....	323
<b>L</b>		NOP .....	323
LCD		<b>O</b>	
Associated Registers .....	279	Opcode Field Descriptions .....	296
Bias Types .....	263	Oscillator Configuration .....	31
Clock Source Selection .....	262	EC .....	31
Configuring the Module .....	278	ECIO .....	31
Frame Frequency .....	264	HS .....	31
Interrupts .....	276	HSPLL .....	31
LCDCON Register .....	258	Internal Oscillator Block .....	34
LCDDATA Register .....	258	INTIO1 .....	31
LCDPS Register .....	258	INTIO2 .....	31
LCDSE Register .....	258	LP .....	31
Multiplex Types .....	263	RC .....	31
Operation During Sleep .....	277	RCIO .....	31
Pixel Control .....	264	XT .....	31
Prescaler .....	262	Oscillator Selection .....	281
Segment Enables .....	263	Oscillator Start-up Timer (OST) .....	39, 55, 281
Waveform Generation .....	264	Oscillator Switching .....	36
LCDCON Register .....	258	Oscillator Transitions .....	37
LCDDATA Register .....	258	Oscillator, Timer1 .....	135, 145
LCDPS Register .....	258	Oscillator, Timer3 .....	143
LP3:LP0 Bits .....	262		
LCDSE Register .....	258	<b>P</b>	
LFSR .....	319	Packaging .....	389
Liquid Crystal Display (LCD) Driver .....	257	Details .....	390
Look-up Tables .....	68	Marking .....	389
		PICSTART Plus Development Programmer .....	348
<b>M</b>		PIE Registers .....	101
Master Clear ( $\overline{\text{MCLR}}$ ) .....	53	Pin Functions	
Master Synchronous Serial Port (MSSP). See MSSP.		AVDD .....	29
Memory Organization .....	65	AVDD .....	19
Data Memory .....	71	AVss .....	29
Program Memory .....	65	AVss .....	19
Memory Programming Requirements .....	364	COM0 .....	17, 25
Microchip Internet Web Site .....	409	LCDBIAS1 .....	17, 25
Migration from Baseline to Enhanced Devices .....	396	LCDBIAS2 .....	17, 25
Migration from High-End to Enhanced Devices .....	397	LCDBIAS3 .....	17, 25
Migration from Mid-Range to Enhanced Devices .....	397	$\overline{\text{MCLR}}/\text{VPP}/\text{RG5}$ .....	12, 20
MOVf .....	319	OSC1/CLKI/RA7 .....	12, 20
MOVFF .....	320	OSC2/CLKO/RA6 .....	12, 20
MOVLB .....	320	RA0/AN0 .....	13, 21
MOVLW .....	321		
MOVsf .....	339		
MOVSS .....	340		

# PIC18F6390/6490/8390/8490

RA1/AN1 .....	13, 21	Vss .....	29
RA2/AN2/VREF-/SEG16 .....	13, 21	Vss .....	19
RA3/AN3/VREF+/SEG17 .....	13, 21	Pinout I/O Descriptions	
RA4/T0CKI/SEG14 .....	13, 21	PIC18F6X90 .....	12
RA5/AN4/HLVDIN/SEG15 .....	13, 21	PIC18F8X90 .....	20
RB0/INT0 .....	14, 22	PIR Registers .....	98
RB1/INT1/SEG8 .....	14, 22	PLL .....	33
RB2/INT2/SEG9 .....	14, 22	HSPLL Oscillator Mode .....	33
RB3/INT3/SEG10 .....	14, 22	Use with INTOSC .....	33, 34
RB4/KBI0/SEG11 .....	14, 22	PLL Lock Time-out .....	55
RB5/KBI1 .....	14, 22	POP .....	324
RB6/KBI2/PGC .....	14, 22	POR. See Power-on Reset.	
RB7/KBI3/PGD .....	14, 22	PORTA	
RC0/T1OSO/T13CKI .....	15, 23	Associated Registers .....	111
RC1/T1OSI/CCP2 .....	15, 23	LATA Register .....	109
RC2/CCP1/SEG13 .....	15, 23	PORTA Register .....	109
RC3/SCK/SCL .....	15, 23	TRISA Register .....	109
RC4/SDI/SDA .....	15, 23	PORTB	
RC5/SDO/SEG12 .....	15, 23	Associated Registers .....	114
RC6/TX1/CK1 .....	15, 23	LATB Register .....	112
RC7/RX1/DT1 .....	15, 23	PORTB Register .....	112
RD0/SEG0 .....	16, 24	RB7:RB4 Interrupt-on-Change Flag	
RD1/SEG1 .....	16, 24	(RBIF Bit) .....	112
RD2/SEG2 .....	16, 24	TRISB Register .....	112
RD3/SEG3 .....	16, 24	PORTC	
RD4/SEG4 .....	16, 24	Associated Registers .....	117
RD5/SEG5 .....	16, 24	LATC Register .....	115
RD6/SEG6 .....	16, 24	PORTC Register .....	115
RD7/SEG7 .....	16, 24	RC3/SCK/SCL Pin .....	171
RE4/COM1 .....	17, 25	TRISC Register .....	115
RE5/COM2 .....	17, 25	PORTD	
RE6/COM3 .....	17, 25	Associated Registers .....	119
RE7/CCP2/SEG31 .....	17, 25	LATD Register .....	118
RF0/AN5/SEG18 .....	18, 26	PORTD Register .....	118
RF1/AN6/C2OUT/SEG19 .....	18, 26	TRISD Register .....	118
RF2/AN7/C1OUT/SEG20 .....	18, 26	PORTE	
RF3/AN8/SEG21 .....	18, 26	Associated Registers .....	121
RF4/AN9/SEG22 .....	18, 26	LATE Register .....	120
RF5/AN10/CVREF/SEG23 .....	18, 26	PORTE Register .....	120
RF6/AN11/SEG24 .....	18, 26	TRISE Register .....	120
RF7/SS/SEG25 .....	18, 26	PORTF	
RG0/SEG30 .....	19, 27	Associated Registers .....	124
RG1/TX2/CK2/SEG29 .....	19, 27	LATF Register .....	122
RG2/RX2/DT2/SEG28 .....	19, 27	PORTF Register .....	122
RG3/SEG27 .....	19, 27	TRISF Register .....	122
RG4/SEG26 .....	19, 27	PORTG	
RG5 .....	19, 27	Associated Registers .....	126
RH0/SEG47 .....	28	LATG Register .....	125
RH1/SEG46 .....	28	PORTG Register .....	125
RH2/SEG45 .....	28	TRISG Register .....	125
RH3/SEG44 .....	28	PORTH	
RH4/SEG40 .....	28	Associated Registers .....	128
RH5/SEG41 .....	28	LATH Register .....	127
RH6/SEG42 .....	28	PORTH Register .....	127
RH7/SEG43 .....	28	TRISH Register .....	127
RJ0/SEG32 .....	29	PORTJ	
RJ1/SEG33 .....	29	Associated Registers .....	130
RJ2/SEG34 .....	29	LATJ Register .....	129
RJ3/SEG35 .....	29	PORTJ Register .....	129
RJ4/SEG39 .....	29	TRISJ Register .....	129
RJ5/SEG38 .....	29	Postscaler, WDT	
RJ6/SEG37 .....	29	Assignment (PSA Bit) .....	133
RJ7/SEG36 .....	29	Rate Select (TOPS2:TOPS0 Bits) .....	133
VDD .....	29	Switching Between Timer0 and WDT .....	133
VDD .....	19		

# PIC18F6390/6490/8390/8490

Power-Managed Modes .....	41	Reader Response .....	410
and Multiple Sleep Commands .....	42	Reading Program Memory and Other Locations .....	292
Effects on Clock Sources .....	39	Register File .....	73
Entering .....	41	Register File Summary .....	76–79
Exiting Idle and Sleep Modes .....		Registers .....	
by Reset .....	48	ADCON0 (A/D Control 0) .....	231
by WDT Time-out .....	48	ADCON1 (A/D Control 1) .....	232
Without a Start-up Delay .....	48	ADCON2 (A/D Control 2) .....	233
Exiting Idle or Sleep Modes .....	48	BAUDCON1 (Baud Rate Control 1) .....	200
by Interrupt .....	48	CCPxCON (CCPx Control) .....	147
Idle Modes .....	45	CMCON (Comparator Control) .....	241
PRI_IDLE .....	46	CONFIG1H (Configuration 1 High) .....	282
Run Modes .....	42	CONFIG2H (Configuration 2 High) .....	284
PRI_RUN .....	42	CONFIG2L (Configuration 2 Low) .....	283
RC_RUN .....	44	CONFIG3H (Configuration 3 High) .....	284
SEC_RUN .....	42	CONFIG4L (Configuration 4 Low) .....	285
Selecting .....	41	CONFIG5L (Configuration 5 Low) .....	285
Sleep Mode .....	45	CVRCON (Comparator Voltage	
Summary (table) .....	41	Reference Control) .....	247
Power-on Reset (POR) .....	53, 281	DEVID1 (Device ID 1) .....	286
Oscillator Start-up Timer (OST) .....	55	DEVID2 (Device ID 2) .....	286
Power-up Timer (PWRT) .....	55, 281	HLVDCON (High/Low-Voltage	
Time-out Sequence .....	55	Detect Control) .....	251
Power-up Delays .....	39	INTCON (Interrupt Control) .....	95
Power-up Timer (PWRT) .....	39, 55	INTCON2 (Interrupt Control 2) .....	96
Prescaler, Capture .....	150	INTCON3 (Interrupt Control 3) .....	97
Prescaler, Timer0 .....	133	IPR1 (Peripheral Interrupt Priority 1) .....	104
Assignment (PSA Bit) .....	133	IPR2 (Peripheral Interrupt Priority 2) .....	105
Rate Select (T0PS2:T0PS0 Bits) .....	133	IPR3 (Peripheral Interrupt Priority 3) .....	106
Switching Between Timer0 and WDT .....	133	LCDCON (LCD Control) .....	258
Prescaler, Timer2 .....	154	LCDDATAx (LCD Datax) .....	261
Program Counter .....	66	LCDPS (LCD Phase) .....	259
PCL, PCH and PCU Registers .....	66	LCDSEx (LCD Segmentx Enable) .....	260
PCLATH and PCLATU Registers .....	66	OSCCON (Oscillator Control) .....	38
Program Memory .....		OSCTUNE (Oscillator Tuning) .....	35
and Extended Instruction Set .....	84	PIE1 (Peripheral Interrupt Enable 1) .....	101
Instructions .....	70	PIE2 (Peripheral Interrupt Enable 2) .....	102
Two-Word .....	70	PIE3 (Peripheral Interrupt Enable 3) .....	103
Interrupt Vector .....	65	PIR1 (Peripheral Interrupt Request (Flag) 1) .....	98
Map and Stack (diagram) .....	65	PIR2 (Peripheral Interrupt Request (Flag) 2) .....	99
Reset Vector .....	65	PIR3 (Peripheral Interrupt Request (Flag) 3) .....	100
Program Verification and Code Protection .....	292	RCON (Reset Control) .....	52, 107
Associated Registers .....	292	RCSTA1 (EUSART Receive	
Programming, Device Instructions .....	295	Status and Control) .....	199
Pulse-Width Modulation. See PWM (CCP Module).		RCSTA2 (AUSART Receive	
PUSH .....	324	Status and Control) .....	219
PUSH and POP Instructions .....	67	SSPCON1 (MSSP Control 1, I <sup>2</sup> C Mode) .....	168
PUSHL .....	340	SSPCON1 (MSSP Control 1, SPI Mode) .....	159
PWM (CCP Module) .....		SSPCON2 (MSSP Control 2,	
Associated Registers .....	155	I <sup>2</sup> C Master Mode) .....	169
Duty Cycle .....	154	SSPSTAT (MSSP Status, I <sup>2</sup> C Mode) .....	167
Example Frequencies/Resolutions .....	154	SSPSTAT (MSSP Status, SPI Mode) .....	158
Period .....	153	STATUS .....	80
Setup for PWM Operation .....	155	STKPTR (Stack Pointer) .....	67
TMR2 to PR2 Match .....	153	T0CON (Timer0 Control) .....	131
<b>Q</b> .....		T1CON (Timer1 Control) .....	135
Q Clock .....	154	T2CON (Timer2 Control) .....	141
<b>R</b> .....		T3CON (Timer3 Control) .....	143
RAM. See Data Memory.		TXSTA1 (EUSART Transmit	
RC Oscillator .....	33	Status and Control) .....	198
RCIO Oscillator Mode .....	33	TXSTA2 (AUSART Transmit	
RCALL .....	325	Status and Control) .....	218
RCON Register .....		WDTCON (Watchdog Timer Control) .....	288
Bit Status During Initialization .....	58	RESET .....	325

# PIC18F6390/6490/8390/8490

Reset .....	51	SUBULNK .....	341
MCLR Reset, during Power-Managed Modes .....	51	SUBWFB .....	332
MCLR Reset, Normal Operation .....	51	SWAPF .....	332
Power-on Reset (POR) .....	51		
Programmable Brown-out Reset (BOR) .....	51	<b>T</b>	
Stack Full Reset .....	51	T0CON Register	
Stack Underflow Reset .....	51	PSA Bit .....	133
Watchdog Timer (WDT) Reset .....	51	T0CS Bit .....	132
Resets .....	281	T0PS2:T0PS0 Bits .....	133
RETFIE .....	326	T0SE Bit .....	132
RETLW .....	326	Table Pointer Operations (table) .....	88
RETURN .....	327	Table Reads .....	68
Return Address Stack .....	66	TBLRD .....	333
Return Stack Pointer (STKPTR) .....	67	TBLWT .....	334
Revision History .....	395	Time-out in Various Situations (table) .....	55
RLCF .....	327	Timer0 .....	131
RLNCF .....	328	16-Bit Mode Timer Reads and Writes .....	132
RRCF .....	328	Associated Registers .....	133
RRNCF .....	329	Clock Source Edge Select (T0SE Bit) .....	132
		Clock Source Select (T0CS Bit) .....	132
<b>S</b>		Operation .....	132
SCK .....	157	Overflow Interrupt .....	133
SDI .....	157	Prescaler. <i>See</i> Prescaler, Timer0.	
SDO .....	157	Timer1 .....	135
Serial Clock, SCK .....	157	16-Bit Read/Write Mode .....	137
Serial Data In (SDI) .....	157	Associated Registers .....	139
Serial Data Out (SDO) .....	157	Interrupt .....	138
Serial Peripheral Interface. <i>See</i> SPI Mode.		Operation .....	136
SETF .....	329	Oscillator .....	135, 137
Slave Select (SS) .....	157	Layout Considerations .....	138
SLEEP .....	330	Overflow Interrupt .....	135
Sleep		Resetting, Using a Special Event Trigger	
OSC1 and OSC2 Pin States .....	39	Output (CCP) .....	138
Software Enabled BOR .....	54	TMR1H Register .....	135
Software Simulator (MPLAB SIM) .....	346	TMR1L Register .....	135
Special Event Trigger. <i>See</i> Compare (CCP Module).		Use as a Real-Time Clock .....	138
Special Features of the CPU .....	281	Timer2 .....	141
Special Function Registers .....	74	Associated Registers .....	142
Map .....	74–75	Interrupt .....	142
SPI Mode (MSSP)		Operation .....	141
Associated Registers .....	165	Output .....	142
Bus Mode Compatibility .....	165	PR2 Register .....	153
Effects of a Reset .....	165	TMR2 to PR2 Match Interrupt .....	153
Enabling SPI I/O .....	161	Timer3 .....	143
Master Mode .....	162	16-Bit Read/Write Mode .....	145
Master/Slave Connection .....	161	Associated Registers .....	145
Operation .....	160	Operation .....	144
Serial Clock .....	157	Oscillator .....	143, 145
Serial Data In .....	157	Overflow Interrupt .....	143, 145
Serial Data Out .....	157	Special Event Trigger (CCP) .....	145
Slave Mode .....	163	TMR3H Register .....	143
Slave Select .....	157	TMR3L Register .....	143
Slave Select Synchronization .....	163	Timing Diagrams	
Sleep Operation .....	165	A/D Conversion .....	385
SPI Clock .....	162	Acknowledge Sequence .....	190
Typical Connection .....	161	Asynchronous Reception .....	209, 225
SS .....	157	Asynchronous Transmission .....	207, 223
SSPOV .....	187	Asynchronous Transmission	
SSPOV Status Flag .....	187	(Back-to-Back) .....	207, 223
SSPSTAT Register		Automatic Baud Rate Calculation .....	205
R/W Bit .....	170, 171	Auto-Wake-up Bit (WUE) During	
Stack Full/Underflow Resets .....	68	Normal Operation .....	210
STATUS Register .....	80	Auto-Wake-up Bit (WUE) During Sleep .....	210
SUBFSR .....	341	Baud Rate Generator with Clock Arbitration .....	184
SUBFWB .....	330	BRG Overflow Sequence .....	205
SUBLW .....	331		



# PIC18F6390/6490/8390/8490

BRG Reset Due to SDA Arbitration		
During Start Condition	193	
Brown-out Reset (BOR)	372	
Bus Collision During a Repeated		
Start Condition (Case 1)	194	
Bus Collision During a Repeated		
Start Condition (Case 2)	194	
Bus Collision During a Start		
Condition (SCL = 0)	193	
Bus Collision During a Start		
Condition (SDA Only)	192	
Bus Collision During a Stop		
Condition (Case 1)	195	
Bus Collision During a Stop		
Condition (Case 2)	195	
Bus Collision for Transmit and Acknowledge	191	
Capture/Compare/PWM (All CCP Modules)	374	
CLKO and I/O	371	
Clock Synchronization	177	
Clock/Instruction Cycle	69	
Example SPI Master Mode (CKE = 0)	375	
Example SPI Master Mode (CKE = 1)	376	
Example SPI Slave Mode (CKE = 0)	377	
Example SPI Slave Mode (CKE = 1)	378	
External Clock (All Modes Except PLL)	369	
Fail-Safe Clock Monitor	291	
High/Low-Voltage Detect Characteristics	366	
High-Voltage Detect Operation (VDIRMAG = 1)	254	
I <sup>2</sup> C Bus Data	379	
I <sup>2</sup> C Bus Start/Stop Bits	379	
I <sup>2</sup> C Master Mode (7 or 10-Bit Transmission)	188	
I <sup>2</sup> C Master Mode (7-Bit Reception)	189	
I <sup>2</sup> C Master Mode First Start Bit	185	
I <sup>2</sup> C Slave Mode (10-Bit Reception, SEN = 0)	174	
I <sup>2</sup> C Slave Mode (10-Bit Reception, SEN = 1)	179	
I <sup>2</sup> C Slave Mode (10-Bit Transmission)	175	
I <sup>2</sup> C Slave Mode (7-Bit Reception, SEN = 0)	172	
I <sup>2</sup> C Slave Mode (7-Bit Reception, SEN = 1)	178	
I <sup>2</sup> C Slave Mode (7-Bit Transmission)	173	
I <sup>2</sup> C Slave Mode General Call Address		
Sequence (7 or 10-Bit Addressing Mode)	180	
I <sup>2</sup> C Stop Condition Receive or Transmit Mode	190	
LCD Interrupt Timing in Quarter-Duty Cycle Drive	276	
LCD Sleep Entry/Exit When SLPEN = 1 or		
CS1:CS0 = 00	277	
Low-Voltage Detect Operation (VDIRMAG = 0)	253	
Master SSP I <sup>2</sup> C Bus Data	381	
Master SSP I <sup>2</sup> C Bus Start/Stop Bits	381	
PWM Output	153	
Repeat Start Condition	186	
Reset, Watchdog Timer (WDT), Oscillator Start-up		
Timer (OST) and Power-up Timer (PWRT)	372	
Send Break Character Sequence	211	
Slave Synchronization	163	
Slow Rise Time (MCLR Tied to VDD,		
VDD Rise > TPWRT)	57	
SPI Mode (Master Mode)	162	
SPI Mode (Slave Mode, CKE = 0)	164	
SPI Mode (Slave Mode, CKE = 1)	164	
Synchronous Reception		
(Master Mode, SREN)	214, 228	
Synchronous Transmission	212, 226	
Synchronous Transmission		
(Through TXEN)	213, 227	
Time-out Sequence on POR w/PLL Enabled		
(MCLR Tied to VDD)	57	
Time-out Sequence on Power-up		
MCLR Not Tied to VDD, Case 1	56	
MCLR Not Tied to VDD, Case 2	56	
MCLR Tied to VDD, VDD Rise < TPWRT	56	
Timer0 and Timer1 External Clock	373	
Transition for Entry to PRI_IDLE Mode	46	
Transition for Entry to SEC_RUN Mode	43	
Transition for Entry to Sleep Mode	45	
Transition for Two-Speed Start-up		
(INTOSC to HSPLL)	289	
Transition for Wake from Idle to Run Mode	46	
Transition for Wake from Sleep (HSPLL)	45	
Transition from RC_RUN Mode to		
PRI_RUN Mode	44	
Transition from SEC_RUN Mode to		
PRI_RUN Mode (HSPLL)	43	
Transition to RC_RUN Mode	44	
Type-A in 1/2 MUX, 1/2 Bias Drive	266	
Type-A in 1/2 MUX, 1/3 Bias Drive	268	
Type-A in 1/3 MUX, 1/2 Bias Drive	270	
Type-A in 1/3 MUX, 1/3 Bias Drive	272	
Type-A in 1/4 MUX, 1/3 Bias Drive	274	
Type-A/Type-B in Static Drive	265	
Type-B in 1/2 MUX, 1/2 Bias Drive	267	
Type-B in 1/2 MUX, 1/3 Bias Drive	269	
Type-B in 1/3 MUX, 1/2 Bias Drive	271	
Type-B in 1/3 MUX, 1/3 Bias Drive	273	
Type-B in 1/4 MUX, 1/3 Bias Drive	275	
USART Synchronous Receive (Master/Slave)	383	
USART Synchronous Transmission		
(Master/Slave)	383	
Timing Diagrams and Specifications		
A/D Conversion Requirements	385	
AC Characteristics - Internal RC Accuracy	370	
Capture/Compare/PWM Requirements		
(All CCP Modules)	374	
CLKO and I/O Requirements	371	
Example SPI Mode Requirements		
(Master Mode, CKE = 0)	375	
Example SPI Mode Requirements		
(Master Mode, CKE = 1)	376	
Example SPI Mode Requirements		
(Slave Mode, CKE = 0)	377	
Example SPI Slave Mode Requirements		
(CKE = 1)	378	
External Clock Requirements	369	
I <sup>2</sup> C Bus Data Requirements (Slave Mode)	380	
I <sup>2</sup> C Bus Start/Stop Bits Requirements		
(Slave Mode)	379	
Master SSP I <sup>2</sup> C Bus Data Requirements	382	
Master SSP I <sup>2</sup> C Bus Start/Stop Bits		
Requirements	381	
PLL Clock	370	
Reset, Watchdog Timer, Oscillator Start-up		
Timer, Power-up Timer and Brown-out		
Reset Requirements	372	
Timer0 and Timer1 External		
Clock Requirements	373	
USART Synchronous Receive		
Requirements	383	
USART Synchronous Transmission		
Requirements	383	

# PIC18F6390/6490/8390/8490

---

Top-of-Stack Access .....	66
TSTFSZ .....	335
Two-Speed Start-up .....	281, 289
Two-Word Instructions	
Example Cases .....	70
TXSTA1 Register	
BRGH Bit .....	201
TXSTA2 Register	
BRGH Bit .....	220

## V

Voltage Reference Specifications .....	365
--	-----

## W

Watchdog Timer (WDT) .....	281, 287
Associated Registers .....	288
Control Register .....	287
During Oscillator Failure .....	290
Programming Considerations .....	287
WCOL .....	185, 186, 187, 190
WCOL Status Flag .....	185, 186, 187, 190
WWW Address .....	409
WWW, On-Line Support .....	5

## X

XORLW .....	335
XORWF .....	336

## 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>**

# PIC18FXX90

---

---

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager  
RE: Reader Response  
Total Pages Sent \_\_\_\_\_

From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: PIC18F6390/6490/8390/8490 Literature Number: DS39629C

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?

---

---

# PIC18F6390/6490/8390/8490

## PIC18F6390/6490/8390/8490 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), (2)</sup>	PIC18F6390/6490/8390/8490, PIC18F6390/6490/8390/8490T; VDD range 4.2V to 5.5V PIC18LF6390/6490/8390/8490, PIC18LF6390/6490/8390/8490T; VDD range 2.0V to 5.5V		
Temperature Range	I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)		
Package	PT = TQFP (Thin Quad Flatpack)		
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)		

**Examples:**

- a) PIC18LF6490-I/PT 301 = Industrial temp., TQFP package, Extended VDD limits, QTP pattern #301.
- b) PIC18F8490-I/PT = Industrial temp., TQFP package, normal VDD limits.
- c) PIC18F8490-E/PT = Extended temp., TQFP package, normal VDD limits.

**Note 1:** F = Standard Voltage Range  
 LF = Wide 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

#### 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 - Fuzhou**  
Tel: 86-591-8750-3506  
Fax: 86-591-8750-3521

**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 - Shunde**  
Tel: 86-757-2839-5507  
Fax: 86-757-2839-5571

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xian**  
Tel: 86-29-8833-7252  
Fax: 86-29-8833-7256

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-4182-8400  
Fax: 91-80-4182-8422

**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: 886-3-572-9526  
Fax: 886-3-572-6459

**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

10/05/07

Компания «Океан Электроники» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Поставка оригинальных импортных электронных компонентов напрямую с производств Америки, Европы и Азии, а так же с крупнейших складов мира;
- Широкая линейка поставок активных и пассивных импортных электронных компонентов (более 30 млн. наименований);
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Помощь Конструкторского Отдела и консультации квалифицированных инженеров;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Поставка электронных компонентов под контролем ВП;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- При необходимости вся продукция военного и аэрокосмического назначения проходит испытания и сертификацию в лаборатории (по согласованию с заказчиком);
- Поставка специализированных компонентов военного и аэрокосмического уровня качества (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Actel, Aeroflex, Peregrine, VPT, Syfer, Eurofarad, Texas Instruments, MS Kennedy, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits, General Dynamics и др.);

Компания «Океан Электроники» является официальным дистрибьютором и эксклюзивным представителем в России одного из крупнейших производителей разъемов военного и аэрокосмического назначения «JONHON», а так же официальным дистрибьютором и эксклюзивным представителем в России производителя высокотехнологичных и надежных решений для передачи СВЧ сигналов «FORSTAR».



## JONHON

«JONHON» (основан в 1970 г.)

Разъемы специального, военного и аэрокосмического назначения:

(Применяются в военной, авиационной, аэрокосмической, морской, железнодорожной, горно- и нефтедобывающей отраслях промышленности)

«FORSTAR» (основан в 1998 г.)

ВЧ соединители, коаксиальные кабели, кабельные сборки и микроволновые компоненты:

(Применяются в телекоммуникациях гражданского и специального назначения, в средствах связи, РЛС, а так же военной, авиационной и аэрокосмической отраслях промышленности).



Телефон: 8 (812) 309-75-97 (многоканальный)

Факс: 8 (812) 320-03-32

Электронная почта: [ocean@oceanchips.ru](mailto:ocean@oceanchips.ru)

Web: <http://oceanchips.ru/>

Адрес: 198099, г. Санкт-Петербург, ул. Калинина, д. 2, корп. 4, лит. А