PIC18F85J11 Family Data Sheet

64/80-Pin, High-Performance Microcontrollers with nanoWatt Technology

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

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

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

Trademarks

ISBN: 978-1-60932-015-7

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.

PIC18F85J11 FAMILY

64/80-Pin, High-Performance Microcontrollers with nanoWatt Technology

Low-Power Features:

- Power-Managed modes: Run, Idle, Sleep
- Run Current Down to 7 μA, Typical
- Idle Current Down to 2.5 μA, Typical
- Sleep Current Down to 100 nA, Typical
- Fast INTOSC Start-up from Sleep, 1 μs Typical
- Two-Speed Oscillator Start-up Reduces Crystal Stabilization Wait Time

Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA (PORTB and PORTC)
- Up to Four External 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 using Timer1
- Two Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 6.25 ns (TCY/16)
  - Compare is 16-bit, max. resolution 100 ns (TCY)
  - PWM output: PWM resolution is up to 10-bit
- Master Synchronous Serial Port (MSSP) module with Two Modes of Operation:
  - 3-wire/4-wire SPI (supports all 4 SPI modes)
  - I^2C™ Master and Slave mode
- One Addressable USART module
- One Enhanced Addressable USART module:
  - Supports LIN/J2602
  - Auto-wake-up on Start bit and Break character
  - Auto-Baud Detect (ABD)
- 10-Bit, up to 12-Channel A/D Converter:
  - Auto-acquisition
  - Conversion available during Sleep
- Two Analog Comparators
- Programmable Reference Voltage for Comparators

External Memory Bus (PIC18F8XJ11 only):

- Address Capability of up to 2 Mbytes
- 8-Bit or 16-Bit Interface
- 12-Bit, 16-Bit and 20-Bit Addressing modes

Flexible Oscillator Structure:

- Two Crystal modes, 4-25 MHz
- Two External Clock modes, up to 40 MHz
- 4x Phase Lock Loop (PLL)
- Internal Oscillator Block:
  - 8 user-selectable frequencies from 31.25 kHz to 8 MHz
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if peripheral clock fails

Special Microcontroller Features:

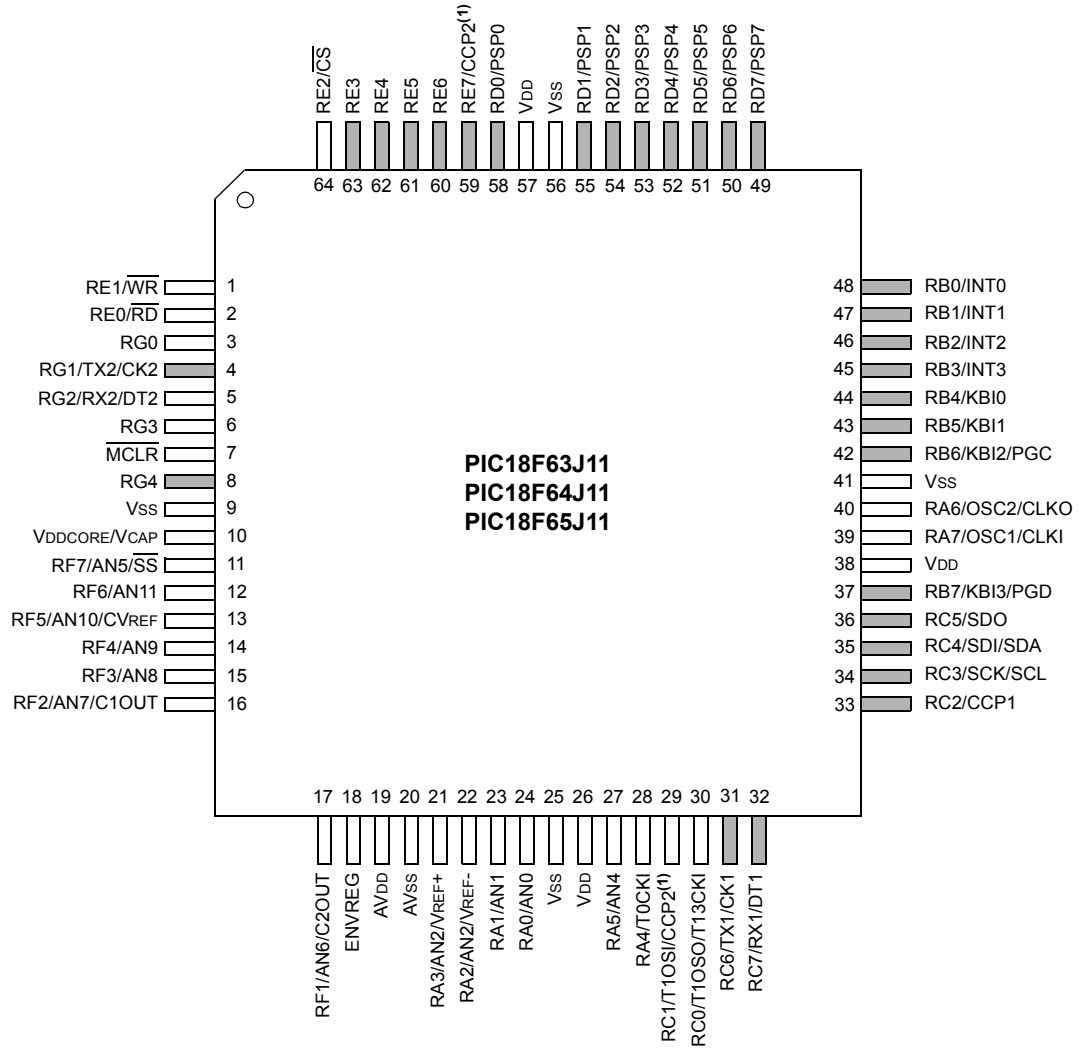
- 1,000 Erase/Write Cycle Flash Program Memory Typical
- Flash Retention 20 Years Minimum
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 131s
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug via two pins
- Operating Voltage Range: 2.0V to 3.6V
- 5.5V Tolerant Input (digital pins only)
- Selectable Open-Drain Configuration for Serial Communication and CCP pins for Driving Outputs up to 5V
- On-Chip 2.5V Regulator

Device	Program Memory		SRAM Data Memory (bytes)	I/O	Timers 8/16-Bit	CCP	MSSP		EUSART/ AUSART	10-Bit A/D (ch)	Comparators	BOR/LVD	External Bus	PSP
	Flash (bytes)	# Single-Word Instructions					SPI	Master I^2C™						
PIC18F63J11	8K	4096	1K	52	1/3	2	Y	Y	1/1	12	2	Y	N	Y
PIC18F64J11	16K	8192	1K	52	1/3	2	Y	Y	1/1	12	2	Y	N	Y
PIC18F65J11	32K	16384	2K	52	1/3	2	Y	Y	1/1	12	2	Y	N	Y
PIC18F83J11	8K	4096	1K	68	1/3	2	Y	Y	1/1	12	2	Y	Y	Y
PIC18F84J11	16K	8192	1K	68	1/3	2	Y	Y	1/1	12	2	Y	Y	Y
PIC18F85J11	32K	16384	2K	68	1/3	2	Y	Y	1/1	12	2	Y	Y	Y

Pin Diagrams

64-Pin TQFP

Pins are up to 5.5V tolerant

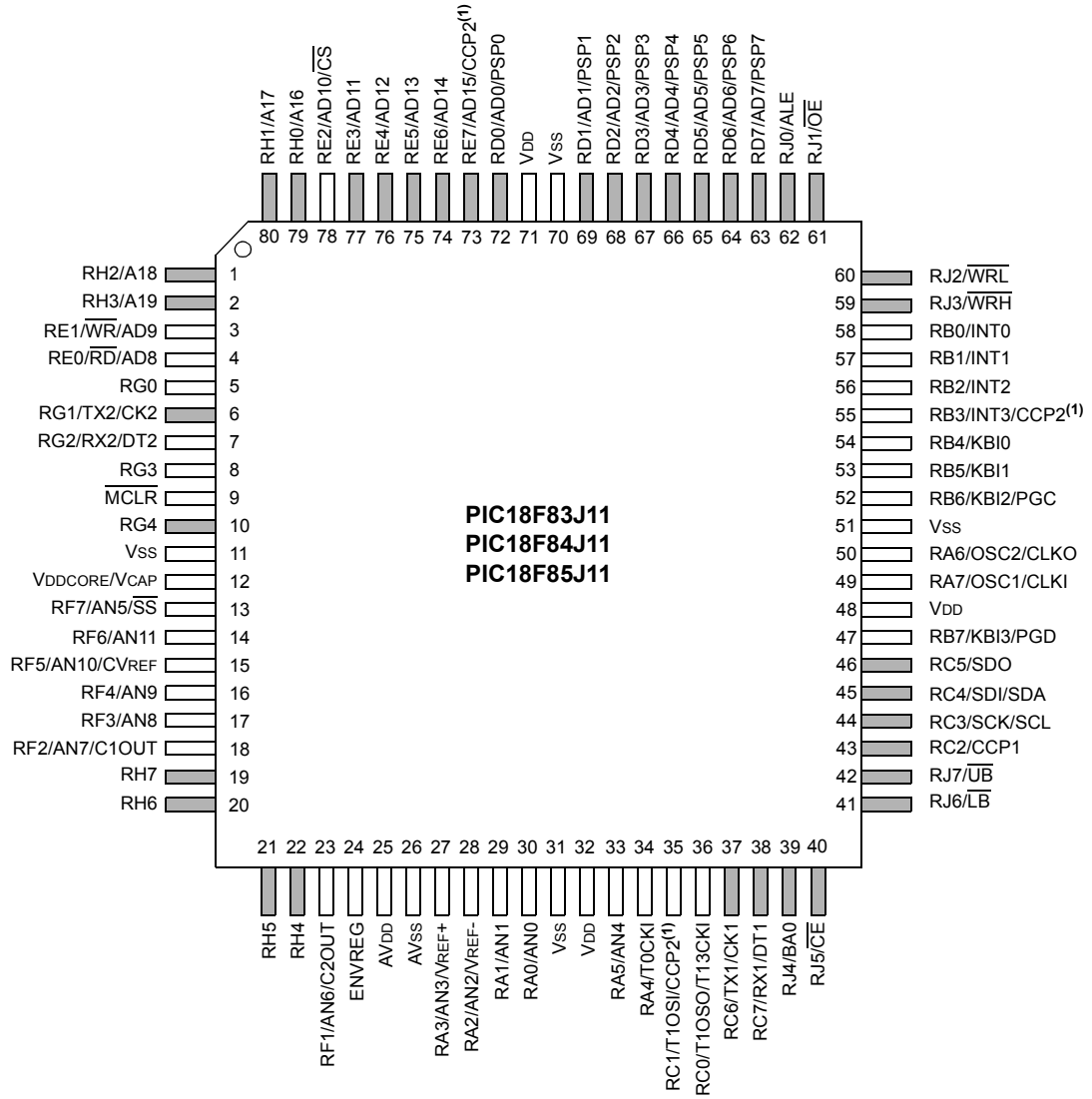


Note 1: The CCP2 pin placement depends on the CCP2MX Configuration bit setting.

Pin Diagrams (Continued)

80-Pin TQFP

Pins are up to 5.5V tolerant



Note 1: The CCP2 pin placement depends on the settings of the CCP2MX and EMB<1:0> Configuration bits.

Table of Contents

TO OUR VALUED CUSTOMERS

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

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

Most Current Data Sheet

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

http://www.microchip.com

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

Errata

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

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

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

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

Customer Notification System

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

NOTES:

1.0 DEVICE OVERVIEW

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

- PIC18F63J11
- PIC18F64J11
- PIC18F65J11
- PIC18F83J11
- PIC18F84J11
- PIC18F85J11

This family combines the traditional advantages of all PIC18 microcontrollers – namely, high computational performance and a rich feature set – while maintaining an extremely competitive price point. These features make the PIC18F85J11 family a logical choice for many high-performance applications where price is a primary consideration.

1.1 Core Features

1.1.1 nanoWatt TECHNOLOGY

All of the devices in the PIC18F85J11 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 RC oscillator, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled but the peripherals still active. In these states, power consumption can be reduced even further, to as little as 4% of normal 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.

1.1.2 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F85J11 family offer six different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes using crystals or ceramic resonators.
- Two External Clock modes offering the option of a divide-by-4 clock output.
- A Phase Lock Loop (PLL) frequency multiplier, available to the External Oscillator modes, which allows clock speeds of up to 40 MHz.
- An internal oscillator block which provides an 8 MHz clock (±2% accuracy) and an INTRC source (approximately 31 kHz, stable over temperature and V_{DD}), 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.

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, 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.

1.1.3 MEMORY OPTIONS

The PIC18F85J11 family provides a range of program memory options, from 8 Kbytes to 32 Kbytes of code space. The Flash cells for program memory are rated to last up to 1000 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 20 years.

The PIC18F85J11 family also provides plenty of room for dynamic application data, with up to 2048 bytes of data RAM.

1.1.4 EXTENDED INSTRUCTION SET

The PIC18F85J11 family implements the optional extension to the PIC18 instruction set, adding 8 new instructions and an Indexed Addressing mode. Enabled as a device configuration option, the extension has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as 'C'.

1.1.5 EXTERNAL MEMORY BUS

In the event that 32 Kbytes of memory are inadequate for an application, the 80-pin members of the PIC18F85J11 family also implement an external memory bus. This allows the controller's internal program counter to address a memory space of up to 2 Mbytes, permitting a level of data access that few 8-bit devices can claim. This allows additional memory options, including:

- Using combinations of on-chip and external memory up to the 2-Mbyte limit
- Using external Flash memory for reprogrammable application code or large data tables
- Using external RAM devices for storing large amounts of variable data

1.1.6 EASY MIGRATION

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

The consistent pinout scheme used throughout the entire family also aids in migrating to the next larger device. This is true when moving between the 64-pin members, between the 80-pin members or even jumping from 64-pin to 80-pin devices.

The PIC18F85J11 family is also largely pin compatible with other PIC18 general purpose families, such as the PIC18F8720 and PIC18F8722. This allows a new dimension to the evolution of applications, allowing developers to select different price points within Microchip's PIC18 portfolio, while maintaining a similar feature set.

1.2 Other Special Features

- **Communications:** The PIC18F85J11 family incorporates a range of serial communication peripherals, including an Addressable USART, a separate Enhanced USART that supports LIN Specification LIN/J2602, and one Master SSP (MSSP) module capable of both SPI and I^2C™ (Master and Slave) modes of operation.
- **CCP Modules:** All devices in the family incorporate two Capture/Compare/PWM (CCP) modules. Up to four different time bases may be used to perform several different operations at once.
- **10-Bit A/D Converter:** This module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period and thus, reducing code overhead.
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing an extended time-out range that is stable across operating voltage and temperature. See **Section 26.0 "Electrical Characteristics"** for time-out periods.

1.3 Details on Individual Family Members

Devices in the PIC18F85J11 family are available in 64-pin and 80-pin packages. Block diagrams for the two groups are shown in Figure 1-1 and Figure 1-2.

The devices are differentiated from each other in four ways:

1. Flash program memory (three sizes, ranging from 8 Kbytes for PIC18FX3J11 devices to 32 Kbytes for PIC18FX5J11 devices).
2. Data RAM (1024 bytes for PIC18FX3J11 and PIC18FX4J11 devices, 2048 bytes for PIC18FX5J11 devices).
3. I/O ports (7 bidirectional ports on 64-pin devices, 9 bidirectional ports on 80-pin devices).
4. External Memory Bus (implemented in 80-pin devices only).

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

The pinouts for all devices are listed in Table 1-3 and Table 1-4.

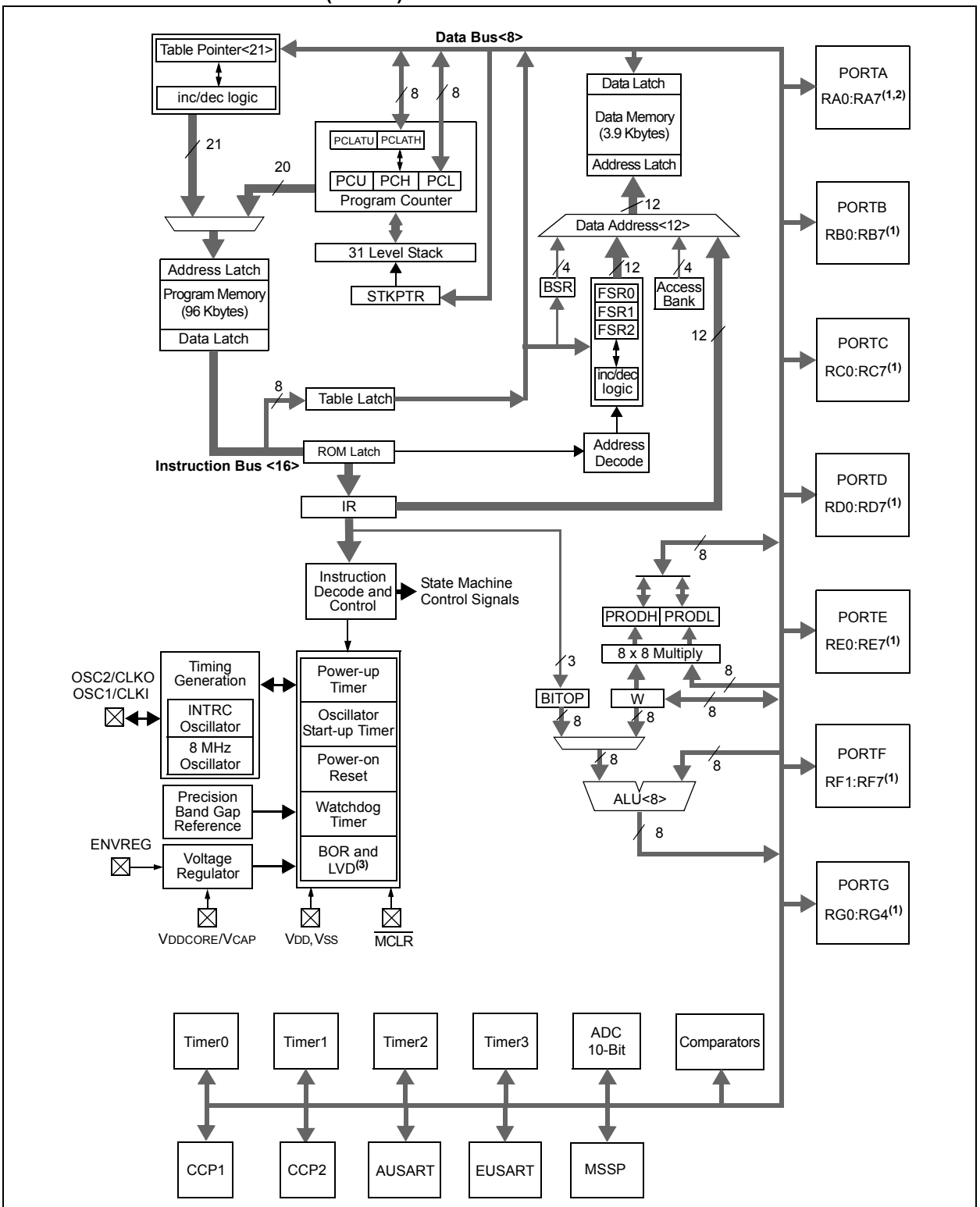
TABLE 1-1: DEVICE FEATURES FOR THE PIC18F6XJ11 FAMILY (64-PIN DEVICES)

Features	PIC18F63J11	PIC18F64J11	PIC18F65J11
Operating Frequency	DC – 40 MHz		
Program Memory (Bytes)	8K	16K	32K
Program Memory (Instructions)	4096	8192	16384
Data Memory (Bytes)	1024	1024	2048
Interrupt Sources	27		
I/O Ports	Ports A, B, C, D, E, F, G		
Timers	4		
Capture/Compare/PWM Modules	2		
Serial Communications	MSSP, Addressable USART, Enhanced USART		
Parallel Communications (PSP)	Yes		
External Memory Bus	No		
10-Bit Analog-to-Digital Module	12 Input Channels		
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	64-Pin TQFP		

TABLE 1-2: DEVICE FEATURES FOR THE PIC18F8XJ11 FAMILY (80-PIN DEVICES)

Features	PIC18F83J11	PIC18F84J11	PIC18F85J11
Operating Frequency	DC – 40 MHz		
Program Memory (Bytes)	8K	16K	32K
Program Memory (Instructions)	4096	8192	16384
Data Memory (Bytes)	1024	1024	2048
Interrupt Sources	27		
I/O Ports	Ports A, B, C, D, E, F, G, H, J		
Timers	4		
Capture/Compare/PWM Modules	2		
Serial Communications	MSSP, Addressable USART, Enhanced USART		
Parallel Communications (PSP)	Yes		
External Memory Bus	Yes		
10-Bit Analog-to-Digital Module	12 Input Channels		
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	80-Pin TQFP		

FIGURE 1-1: PIC18F6XJ11 (64-PIN) BLOCK DIAGRAM

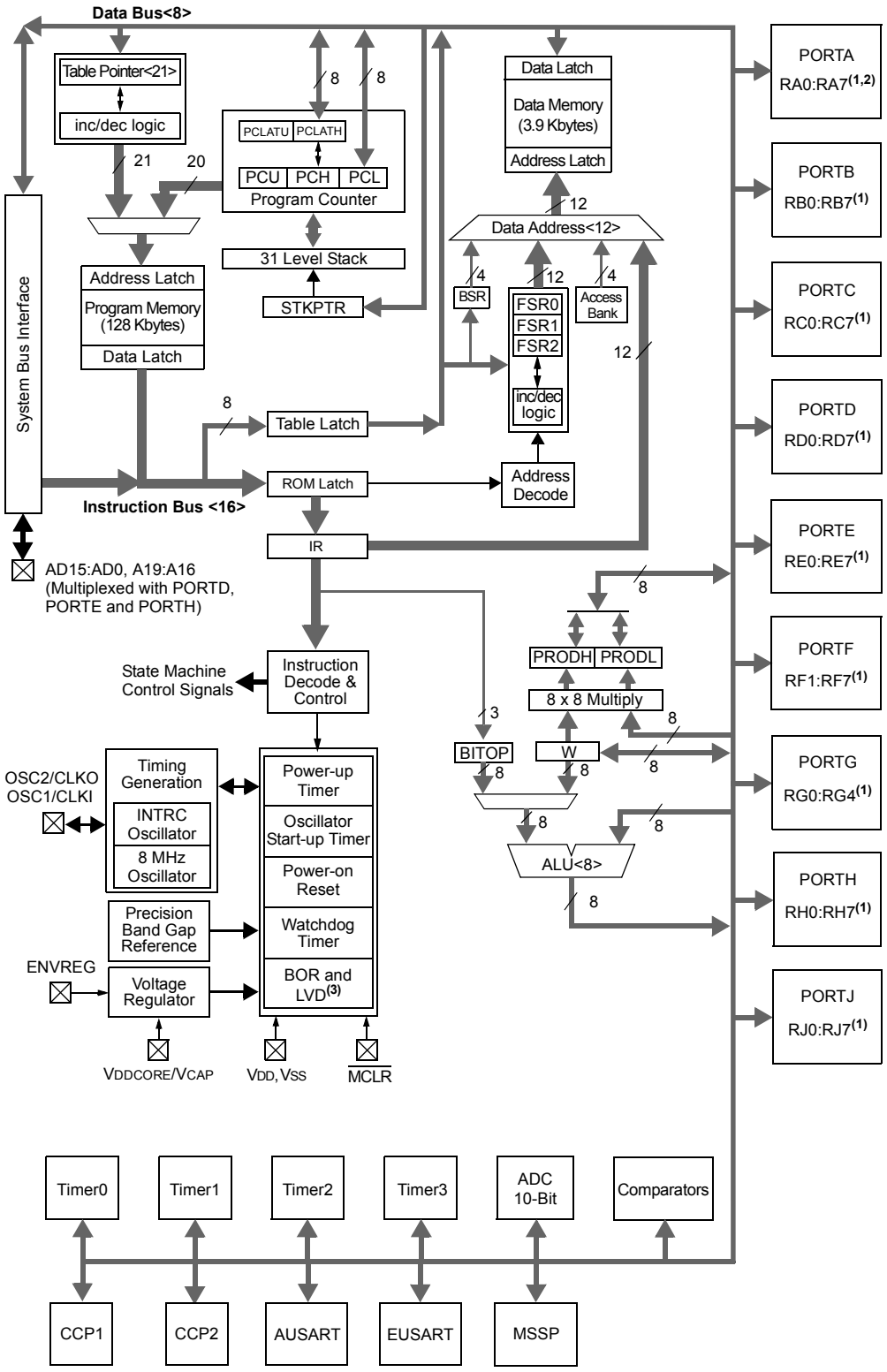


Note 1: See Table 1-3 for I/O port pin descriptions.

2: RA6 and RA7 are only available as digital I/O in select oscillator modes. See **Section 3.0 "Oscillator Configurations"** for more information

3: Brown-out Reset and Low-Voltage Detect functions are provided when the on-board voltage regulator is enabled.

FIGURE 1-2: PIC18F8XJ11 (80-PIN) BLOCK DIAGRAM



Note 1: See Table 1-3 for I/O port pin descriptions.

2: RA6 and RA7 are only available as digital I/O in select oscillator modes. See **Section 3.0 "Oscillator Configurations"** for more information

3: Brown-out Reset and Low-Voltage Detect functions are provided when the on-board voltage regulator is enabled.

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
$\overline{MCLR}$	7	I	ST	Master Clear (input) or programming voltage (input). This pin is an active-low Reset to the device.
RA7/OSC1/CLKI	39			Oscillator crystal or external clock input.
RA7		I/O	TTL	General purpose I/O pin.
OSC1		I	CMOS	Oscillator crystal input.
CLKI		I	CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.)
RA6/OSC2/CLKO	40			Oscillator crystal or clock output.
RA6		I/O	TTL	General purpose I/O pin.
OSC2		O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In EC modes, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
				PORTA is a bidirectional I/O port.
RA0/AN0	24			
RA0		I/O	TTL	Digital I/O.
AN0		I	Analog	Analog Input 0.
RA1/AN1	23			
RA1		I/O	TTL	Digital I/O.
AN1		I	Analog	Analog Input 1.
RA2/AN2/VREF-	22			
RA2		I/O	TTL	Digital I/O.
AN2		I	Analog	Analog Input 2.
VREF-		I	Analog	A/D reference voltage (low) input.
RA3/AN3/VREF+	21			
RA3		I/O	TTL	Digital I/O.
AN3		I	Analog	Analog Input 3.
VREF+		I	Analog	A/D reference voltage (high) input.
RA4/T0CKI	28			
RA4		I/O	ST/OD	Digital I/O. Open-drain when configured as output.
T0CKI		I	ST	Timer0 external clock input.
RA5/AN4	27			
RA5		I/O	TTL	Digital I/O.
AN4		I	Analog	Analog Input 4.
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); I^2C™ = I^2C/SMBus

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

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

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); I^2C™ = I^2C/SMBus

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

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

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTD is a bidirectional I/O port.
RD0/PSP0	58			
RD0		I/O	ST	Digital I/O.
PSP0		I/O	TTL	Parallel Slave Port data.
RD1/PSP1	55			
RD1		I/O	ST	Digital I/O.
PSP1		I/O	TTL	Parallel Slave Port data.
RD2/PSP2	54			
RD2		I/O	ST	Digital I/O.
PSP2		I/O	TTL	Parallel Slave Port data.
RD3/PSP3	53			
RD3		I/O	ST	Digital I/O.
PSP3		I/O	TTL	Parallel Slave Port data.
RD4/PSP4	52			
RD4		I/O	ST	Digital I/O.
PSP4		I/O	TTL	Parallel Slave Port data.
RD5/PSP5	51			
RD5		I/O	ST	Digital I/O.
PSP5		I/O	TTL	Parallel Slave Port data.
RD6/PSP6	50			
RD6		I/O	ST	Digital I/O.
PSP6		I/O	TTL	Parallel Slave Port data.
RD7/PSP7	49			
RD7		I/O	ST	Digital I/O.
PSP7		I/O	TTL	Parallel Slave Port data.

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_{DD}); I^2C™ = I^2C/SMBus

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTE is a bidirectional I/O port.
RE0/$\overline{RD}$	2			
RE0		I/O	ST	Digital I/O.
$\overline{RD}$		I	TTL	Read control for Parallel Slave Port.
RE1/$\overline{WR}$	1			
RE1		I/O	ST	Digital I/O.
$\overline{WR}$		I	TTL	Write control for Parallel Slave Port.
RE2/$\overline{CS}$	64			
RE2		I/O	ST	Digital I/O.
$\overline{CS}$		I	TTL	Chip select control for Parallel Slave Port.
RE3	63	I/O	ST	Digital I/O.
RE4	62	I/O	ST	Digital I/O.
RE5	61	I/O	ST	Digital I/O.
RE6	60	I/O	ST	Digital I/O.
RE7/CCP2	59			
RE7		I/O	ST	Digital I/O.
CCP2**(2)**		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.

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_{DD}); I^2C™ = I^2C/SMBus

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTF is a bidirectional I/O port.
RF1/AN6/C2OUT	17			
RF1		I/O	ST	Digital I/O.
AN6		I	Analog	Analog Input 6.
C2OUT		O	—	Comparator 2 output.
RF2/AN7/C1OUT	16			
RF2		I/O	ST	Digital I/O.
AN7		I	Analog	Analog Input 7.
C1OUT		O	—	Comparator 1 output.
RF3/AN8	15			
RF3		I/O	ST	Digital I/O.
AN8		I	Analog	Analog Input 8.
RF4/AN9	14			
RF4		I/O	ST	Digital I/O.
AN9		I	Analog	Analog Input 9.
RF5/AN10/CV$_{REF}$	13			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog Input 10.
CV$_{REF}$		O	Analog	Comparator reference voltage output.
RF6/AN11	12			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog Input 11.
RF7/AN5/$\overline{SS}$	11			
RF7		I/O	ST	Digital I/O.
AN5		O	Analog	Analog Input 5.
$\overline{SS}$		I	TTL	SPI slave select input.

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$_{DD}$); I^{2}C™ = I^{2}C/SMBus

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

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

TABLE 1-3: PIC18F6XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTG is a bidirectional I/O port.
RG0	3	I/O	ST	Digital I/O.
RG1/TX2/CK2	4			
RG1		I/O	ST	Digital I/O.
TX2		O	—	AUSART asynchronous transmit.
CK2		I/O	ST	AUSART synchronous clock (see related RX2/DT2).
RG2/RX2/DT2	5			
RG2		I/O	ST	Digital I/O.
RX2		I	ST	AUSART asynchronous receive.
DT2		I/O	ST	AUSART synchronous data (see related TX2/CK2).
RG3	6	I/O	ST	Digital I/O.
RG4	8	I/O	ST	Digital I/O.
VSS	9, 25, 41, 56	P	—	Ground reference for logic and I/O pins.
VDD	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.
ENVREG	18	I	ST	Enable for on-chip voltage regulator.
VDDCORE/VCAP	10			Core logic power or external filter capacitor connection.
VDDCORE		P	—	Positive supply for microcontroller core logic (regulator disabled).
VCAP		P	—	External filter capacitor connection (regulator enabled).

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to VDD); I^2C™ = I^2C/SMBus

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

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

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
$\overline{MCLR}$	9	I	ST	Master Clear (input) or programming voltage (input). This pin is an active-low Reset to the device.
RA7/OSC1/CLKI	49			Oscillator crystal or external clock input.
RA7		I/O	TTL	General purpose I/O pin.
OSC1		I	CMOS	Oscillator crystal input.
CLKI		I	CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.)
RA6/OSC2/CLKO	50			Oscillator crystal or clock output.
RA6		I/O	TTL	General purpose I/O pin.
OSC2		O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In EC modes, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
				PORTA is a bidirectional I/O port.
RA0/AN0	30			
RA0		I/O	TTL	Digital I/O.
AN0		I	Analog	Analog Input 0.
RA1/AN1	29			
RA1		I/O	TTL	Digital I/O.
AN1		I	Analog	Analog Input 1.
RA2/AN2/V_{REF-}	28			
RA2		I/O	TTL	Digital I/O.
AN2		I	Analog	Analog Input 2.
V_{REF-}		I	Analog	A/D reference voltage (low) input.
RA3/AN3/V_{REF+}	27			
RA3		I/O	TTL	Digital I/O.
AN3		I	Analog	Analog Input 3.
V_{REF+}		I	Analog	A/D reference voltage (high) input.
RA4/T0CKI	34			
RA4		I/O	ST/OD	Digital I/O. Open-drain when configured as output.
T0CKI		I	ST	Timer0 external clock input.
RA5/AN4	33			
RA5		I/O	TTL	Digital I/O.
AN4		I	Analog	Analog Input 4.
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 V_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0	58			
RB0		I/O	TTL	Digital I/O.
INT0		I	ST	External Interrupt 0.
RB1/INT1	57			
RB1		I/O	TTL	Digital I/O.
INT1		I	ST	External Interrupt 1.
RB2/INT2	56			
RB2		I/O	TTL	Digital I/O.
INT2		I	ST	External Interrupt 2.
RB3/INT3/CCP2	55			
RB3		I/O	TTL	Digital I/O.
INT3		I	ST	External Interrupt 3.
CCP2(1)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
RB4/KBI0	54			
RB4		I/O	TTL	Digital I/O.
KBI0		I	TTL	Interrupt-on-change pin.
RB5/KBI1	53			
RB5		I/O	TTL	Digital I/O.
KBI1		I	TTL	Interrupt-on-change pin.
RB6/KBI2/PGC	52			
RB6		I/O	TTL	Digital I/O.
KBI2		I	TTL	Interrupt-on-change pin.
PGC		I/O	ST	In-Circuit Debugger and ICSP™ programming clock pin.
RB7/KBI3/PGD	47			
RB7		I/O	TTL	Digital I/O.
KBI3		I	TTL	Interrupt-on-change pin.
PGD		I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTC is a bidirectional I/O port.
RC0/T1OSO/T13CKI	36			
RC0		I/O	ST	Digital I/O.
T1OSO		O	—	Timer1 oscillator output.
T13CKI		I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI/CCP2	35			
RC1		I/O	ST	Digital I/O.
T1OSI		I	CMOS	Timer1 oscillator input.
CCP2(2)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
RC2/CCP1	43			
RC2		I/O	ST	Digital I/O.
CCP1		I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
RC3/SCK/SCL	44			
RC3		I/O	ST	Digital I/O.
SCK		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL		I/O	I^2C	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI/SDA	45			
RC4		I/O	ST	Digital I/O.
SDI		I	ST	SPI data in.
SDA		I/O	I^2C	I^2C data I/O.
RC5/SDO	46			
RC5		I/O	ST	Digital I/O.
SDO		O	—	SPI data out.
RC6/TX1/CK1	37			
RC6		I/O	ST	Digital I/O.
TX1		O	—	EUSART asynchronous transmit.
CK1		I/O	ST	EUSART synchronous clock (see related RX1/DT1).
RC7/RX1/DT1	38			
RC7		I/O	ST	Digital I/O.
RX1		I	ST	EUSART asynchronous receive.
DT1		I/O	ST	EUSART synchronous data (see related TX1/CK1).

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C™ = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTD is a bidirectional I/O port.
RD0/AD0/PSP0	72			
RD0		I/O	ST	Digital I/O.
AD0		I/O	TTL	External memory address/data 0.
PSP0		I/O	TTL	Parallel Slave Port data.
RD1/AD1/PSP1	69			
RD1		I/O	ST	Digital I/O.
AD1		I/O	TTL	External memory address/data 1.
PSP1		I/O	TTL	Parallel Slave Port data.
RD2/AD2/PSP2	68			
RD2		I/O	ST	Digital I/O.
AD2		I/O	TTL	External memory address/data 2.
PSP2		I/O	TTL	Parallel Slave Port data.
RD3/AD3/PSP3	67			
RD3		I/O	ST	Digital I/O.
AD3		I/O	TTL	External memory address/data 3.
PSP3		I/O	TTL	Parallel Slave Port data.
RD4/AD4/PSP4	66			
RD4		I/O	ST	Digital I/O.
AD4		I/O	TTL	External memory address/data 4.
PSP4		I/O	TTL	Parallel Slave Port data.
RD5/AD5/PSP5	65			
RD5		I/O	ST	Digital I/O.
AD5		I/O	TTL	External memory address/data 5.
PSP5		I/O	TTL	Parallel Slave Port data.
RD6/AD6/PSP6	64			
RD6		I/O	ST	Digital I/O.
AD6		I/O	TTL	External memory address/data 6.
PSP6		I/O	TTL	Parallel Slave Port data.
RD7/AD7/PSP7	63			
RD7		I/O	ST	Digital I/O.
AD7		I/O	TTL	External memory address/data 7.
PSP7		I/O	TTL	Parallel Slave Port data.

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_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTE is a bidirectional I/O port.
RE0/$\overline{RD}$/AD8	4			
RE0		I/O	ST	Digital I/O.
$\overline{RD}$		I	TTL	Read control for Parallel Slave Port.
AD8		I/O	TTL	External memory address/data 8.
RE1/$\overline{WR}$/AD9	3			
RE1		I/O	ST	Digital I/O.
$\overline{WR}$		I	TTL	Write control for Parallel Slave Port.
AD9		I/O	TTL	External memory address/data 9.
RE2/AD10/$\overline{CS}$	78			
RE2		I/O	ST	Digital I/O.
AD10		I/O	TTL	External memory address/data 10.
$\overline{CS}$		I	TTL	Chip select control for Parallel Slave Port.
RE3/AD11	77			
RE3		I/O	ST	Digital I/O.
AD11		I/O	TTL	External memory address/data 11.
RE4/AD12	76			
RE4		I/O	ST	Digital I/O.
AD12		I/O	TTL	External memory address/data 12.
RE5/AD13	75			
RE5		I/O	ST	Digital I/O.
AD13		I/O	TTL	External memory address/data 13.
RE6/AD14	74			
RE6		I/O	ST	Digital I/O.
AD14		I/O	TTL	External memory address/data 14.
RE7/AD15/CCP2	73			
RE7		I/O	ST	Digital I/O.
AD15		I/O	TTL	External memory address/data 15.
CCP2**(3)**		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.

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_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTF is a bidirectional I/O port.
RF1/AN6/C2OUT	23			
RF1		I/O	ST	Digital I/O.
AN6		I	Analog	Analog Input 6.
C2OUT		O	—	Comparator 2 output.
RF2/AN7/C1OUT	18			
RF2		I/O	ST	Digital I/O.
AN7		I	Analog	Analog Input 7.
C1OUT		O	—	Comparator 1 output.
RF3/AN8	17			
RF3		I/O	ST	Digital I/O.
AN8		I	Analog	Analog Input 8.
RF4/AN9	16			
RF4		I/O	ST	Digital I/O.
AN9		I	Analog	Analog Input 9.
RF5/AN10/CV$_{REF}$	15			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog Input 10.
CV$_{REF}$		O	Analog	Comparator reference voltage output.
RF6/AN11	14			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog Input 11.
RF7/AN5/$\overline{SS}$	13			
RF7		I/O	ST	Digital I/O.
AN5		O	Analog	Analog Input 5.
$\overline{SS}$		I	TTL	SPI slave select input.

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$_{DD}$); I^{2}C™ = I^{2}C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTG is a bidirectional I/O port.
RG0	5	I/O	ST	Digital I/O.
RG1/TX2/CK2	6			
RG1		I/O	ST	Digital I/O.
TX2		O	—	AUSART asynchronous transmit.
CK2		I/O	ST	AUSART synchronous clock (see related RX2/DT2).
RG2/RX2/DT2	7			
RG2		I/O	ST	Digital I/O.
RX2		I	ST	AUSART asynchronous receive.
DT2		I/O	ST	AUSART synchronous data (see related TX2/CK2).
RG3	8	I/O	ST	Digital I/O.
RG4	10	I/O	ST	Digital I/O.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTH is a bidirectional I/O port.
RH0/A16	79			
RH0		I/O	ST	Digital I/O.
A16		I/O	TTL	External memory address/data 16.
RH1/A17	80			
RH1		I/O	ST	Digital I/O.
A17		I/O	TTL	External memory address/data 17.
RH2/A18	1			
RH2		I/O	ST	Digital I/O.
A18		I/O	TTL	External memory address/data 18.
RH3/A19	2			
RH3		I/O	ST	Digital I/O.
A19		I/O	TTL	External memory address/data 19.
RH4	22	I/O	ST	Digital I/O.
RH5	21	I/O	ST	Digital I/O.
RH6	20	I/O	ST	Digital I/O.
RH7	19	I/O	ST	Digital I/O.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

TABLE 1-4: PIC18F8XJ11 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTJ is a bidirectional I/O port.
RJ0/ALE	62			
RJ0		I/O	ST	Digital I/O.
ALE		O	—	External memory address latch enable.
RJ1/$\overline{OE}$	61			
RJ1		I/O	ST	Digital I/O.
$\overline{OE}$		O	—	External memory output enable.
RJ2/$\overline{WRL}$	60			
RJ2		I/O	ST	Digital I/O.
$\overline{WRL}$		O	—	External memory write low control.
RJ3/$\overline{WRH}$	59			
RJ3		I/O	ST	Digital I/O.
$\overline{WRH}$		O	—	External memory write high control.
RJ4/BA0	39			
RJ4		I/O	ST	Digital I/O.
BA0		O	—	External memory byte address 0 control.
RJ5/$\overline{CE}$	40			
RJ5		I/O	ST	Digital I/O
$\overline{CE}$		O	—	External memory chip enable control.
RJ6/$\overline{LB}$	41			
RJ6		I/O	ST	Digital I/O.
$\overline{LB}$		O	—	External memory low byte control.
RJ7/$\overline{UB}$	42			
RJ7		I/O	ST	Digital I/O.
$\overline{UB}$		O	—	External memory high byte control.
VSS	11, 31, 51, 70	P	—	Ground reference for logic and I/O pins.
VDD	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.
ENVREG	24	I	ST	Enable for on-chip voltage regulator.
VDDCORE/VCAP	12			Core logic power or external filter capacitor connection.
VDDCORE		P	—	Positive supply for microcontroller core logic (regulator disabled).
VCAP		P	—	External filter capacitor connection (regulator enabled).

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to VDD); I^2C™ = I^2C/SMBus

Note 1: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared (80-pin devices, Extended Microcontroller mode only).

2: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

3: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

NOTES:

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18FJ MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18F85J11 family family of 8-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins (see **Section 2.2 "Power Supply Pins"**)
- All AVDD and AVSS pins, regardless of whether or not the analog device features are used (see **Section 2.2 "Power Supply Pins"**)
- $\overline{\text{MCLR}}$ pin (see **Section 2.3 "Master Clear (MCLR) Pin"**)
- ENVREG (if implemented) and VCAP/VDDCORE pins (see **Section 2.4 "Voltage Regulator Pins (ENVREG and VCAP/VDDCORE)"**)

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

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

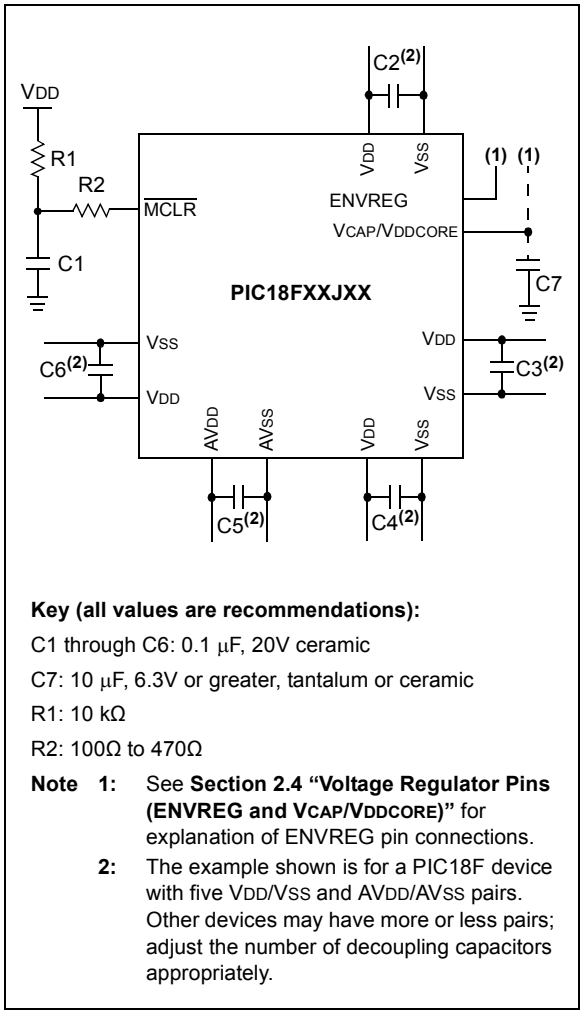
Additionally, the following pins may be required:

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

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

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

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS



Key (all values are recommendations):

C1 through C6: 0.1 μF, 20V ceramic

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

R1: 10 kΩ

R2: 100Ω to 470Ω

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

2: The example shown is for a PIC18F device with five VDD/VSS and AVDD/AVSS pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.

2.2 Power Supply Pins

2.2.1 DECOUPLING CAPACITORS

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

Consider the following criteria when using decoupling capacitors:

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

2.2.2 TANK CAPACITORS

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

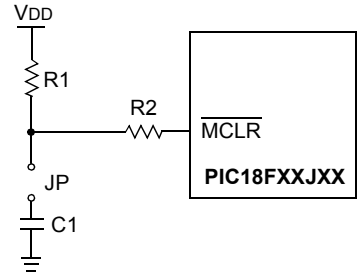
2.3 Master Clear ($\overline{\text{MCLR}}$) Pin

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

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

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

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



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

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

2.4 Voltage Regulator Pins (ENVREG and VCAP/VDDCORE)

The on-chip voltage regulator enable pin, ENVREG, must always be connected directly to either a supply voltage or to ground. Tying ENVREG to VDD enables the regulator, while tying it to ground disables the regulator. Refer to **Section 23.3 "On-Chip Voltage Regulator"** for details on connecting and using the on-chip regulator.

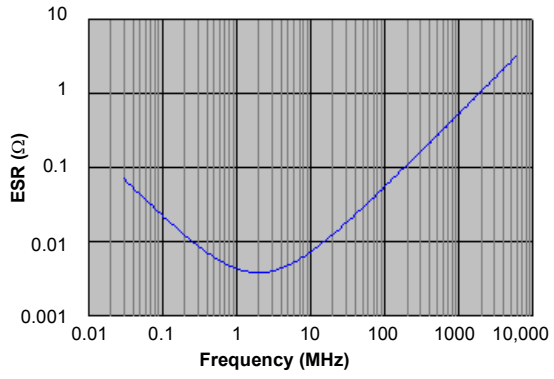
When the regulator is enabled, a low-ESR (< 5Ω) capacitor is required on the VCAP/VDDCORE pin to stabilize the voltage regulator output voltage. The VCAP/VDDCORE pin must not be connected to VDD and must use a capacitor of 10 μF connected to ground. The type can be ceramic or tantalum. A suitable example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

It is recommended that the trace length not exceed 0.25 inch (6 mm). Refer to **Section 26.0 "Electrical Characteristics"** for additional information.

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

Note that the "LF" versions of some low pin count PIC18FJ parts (e.g., the PIC18LF45J10) do not have the ENVREG pin. These devices are provided with the voltage regulator permanently disabled; they must always be provided with a supply voltage on the VDDCORE pin.

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



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

2.5 ICSP Pins

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

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

For device emulation, ensure that the "Communication Channel Select" (i.e., PGCx/PGDx pins) programmed into the device matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

For more information on available Microchip development tools connection requirements, refer to **Section 24.0 "Development Support"**.

2.6 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 3.0 "Oscillator Configurations"** for details).

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

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

Layout suggestions are shown in Figure 2-4. In-line packages may be handled with a single-sided layout that completely encompasses the oscillator pins. With fine-pitch packages, it is not always possible to completely surround the pins and components. A suitable solution is to tie the broken guard sections to a mirrored ground layer. In all cases, the guard trace(s) must be returned to ground.

In planning the application's routing and I/O assignments, ensure that adjacent port pins and other signals in close proximity to the oscillator are benign (i.e., free of high frequencies, short rise and fall times, and other similar noise).

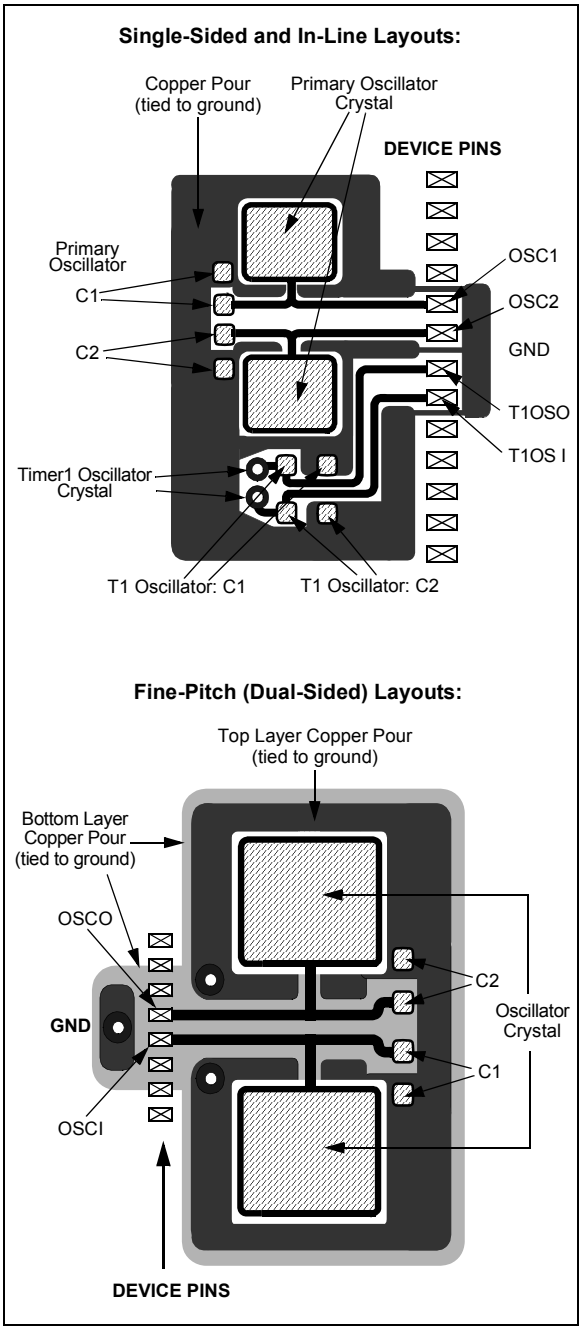
For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

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

2.7 Unused I/Os

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

FIGURE 2-4: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



3.0 OSCILLATOR CONFIGURATIONS

3.1 Oscillator Types

The PIC18F85J11 family of devices can be operated in six different oscillator modes:

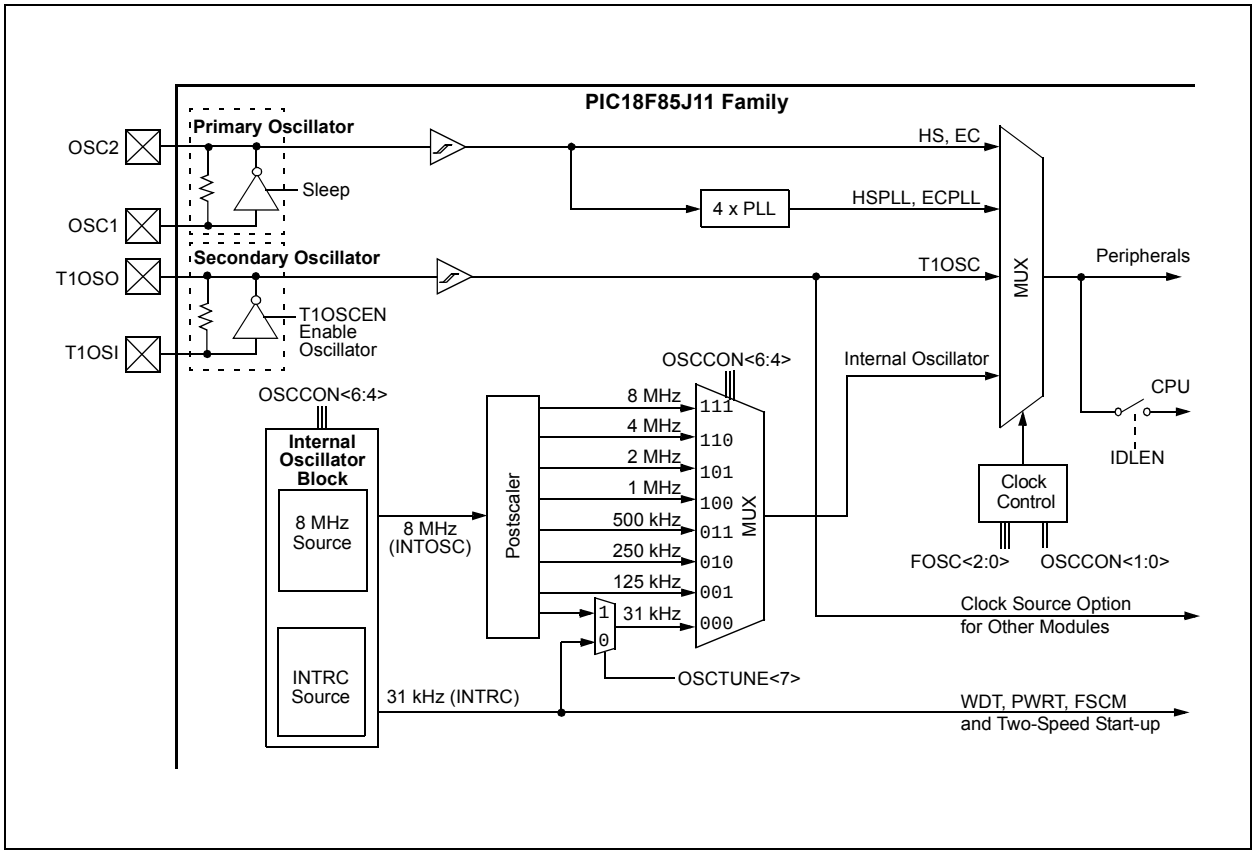
1. HS High-Speed Crystal/Resonator
2. HSPLL High-Speed Crystal/Resonator with Software PLL Control
3. EC External Clock with FOSC/4 Output
4. ECPLL External Clock with Software PLL Control
5. INTOSC Internal Fast RC (8 MHz) Oscillator
6. INTRC Internal 31 kHz Oscillator

Five of these are selected by the user by programming the FOSC<2:0> Configuration bits. The sixth mode (INTRC) may be invoked under software control; it can also be configured as the default mode on device Resets.

In addition, PIC18F85J11 family devices can switch between different clock sources, either under software control or automatically under certain conditions. This allows for additional power savings by managing device clock speed in real time without resetting the application.

The clock sources for the PIC18F85J11 family of devices are shown in Figure 3-1.

FIGURE 3-1: PIC18F85J11 FAMILY CLOCK DIAGRAM



3.2 Control Registers

The OSCCON register (Register 3-1) controls the main aspects of the device clock's operation. It selects the oscillator type to be used, which of the power-managed modes to invoke and the output frequency of the INTOSC source. It also provides status on the oscillators.

The OSCTUNE register (Register 3-2) controls the tuning and operation of the internal oscillator block. It also implements the PLLEN bits which control the operation of the Phase Locked Loop (PLL) in Internal Oscillator modes (see **Section 3.4.3 "PLL Frequency Multiplier"**).

REGISTER 3-1: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R/W-1	R/W-0	R/W-0	R(1)	R-0	R/W-0	R/W-0
IDLEN	IRCF2(2)	IRCF1(2)	IRCF0(2)	OSTS	IOFS	SCS1(4)	SCS0(4)
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 an Idle mode when a SLEEP instruction is executed
0 = Device enters Sleep mode when a SLEEP instruction is executed

bit 6-4 **IRCF<2:0>:** INTOSC Source Frequency Select bits(2)
111 = 8 MHz (INTOSC drives clock directly)
110 = 4 MHz
101 = 2 MHz
100 = 1 MHz (default)
011 = 500 kHz
010 = 250 kHz
001 = 125 kHz
000 = 31 kHz (from either INTOSC/256 or INTRC)(3)

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

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

bit 1-0 **SCS<1:0>:** System Clock Select bits(4)
11 = Internal oscillator block
10 = Primary oscillator
01 = Timer1 oscillator
When FOSC2 = 1:
00 = Primary oscillator
When FOSC2 = 0:
00 = Internal oscillator

Note 1: Reset state depends on the state of the IESO Configuration bit.
2: Modifying these bits will cause an immediate clock frequency switch if the internal oscillator is providing the device clocks.
3: Source selected by the INTSRC bit (OSCTUNE<7>), see text.
4: Modifying these bits will cause an immediate clock source switch.

REGISTER 3-2: OSCTUNE: OSCILLATOR TUNING 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
INTSRC	PLLEN(1)	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
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 **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 from INTRC 31 kHz oscillator

bit 6 **PLLEN:** Frequency Multiplier PLL Enable bit(1)
1 = PLL is enabled
0 = PLL is disabled

bit 5-0 **TUN<5:0>:** Fast RC Oscillator (INTOSC) Frequency Tuning bits
011111 = Maximum frequency
•
•
000001
000000 = Center frequency. Fast RC oscillator is running at the calibrated frequency.
111111
•
•
100000 = Minimum frequency

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

3.3 Clock Sources and Oscillator Switching

Essentially, PIC18F85J11 family devices have three independent clock sources:

- Primary oscillators
- Secondary oscillators
- Internal oscillator

The **primary oscillators** can be thought of as the main device oscillators. These are any external oscillators connected to the OSC1 and OSC2 pins, and include the External Crystal and Resonator modes and the External Clock modes. In some circumstances, the internal oscillator block may be considered a primary oscillator. The particular mode is defined by the FOSC Configuration bits. The details of these modes are covered in **Section 3.4 "External Oscillator Modes"**.

The **secondary oscillators** are external clock sources that are not connected to the OSC1 or OSC2 pins. These sources may continue to operate even after the controller is placed in a power-managed mode. PIC18F85J11 family devices offer the Timer1 oscillator as a secondary oscillator source. This oscillator, in all power-managed modes, is often the time base for functions such as a Real-Time Clock. The Timer1 oscillator is discussed in greater detail in **Section 13.3 "Timer1 Oscillator"**

In addition to being a primary clock source in some circumstances, the **internal oscillator** is available as a power-managed mode clock source. The INTRC source is also used as the clock source for several special features, such as the WDT and Fail-Safe Clock Monitor. The internal oscillator block is discussed in more detail in **Section 3.5 "Internal Oscillator Block"**.

The PIC18F85J11 family includes features that allow the device clock source to be switched from the main oscillator, chosen by device configuration, to one of the alternate clock sources. When an alternate clock source is enabled, various power-managed operating modes are available.

3.3.1 CLOCK SOURCE SELECTION

The System Clock Select bits, SCS<1:0> (OSCCON<1:0>), select the clock source. The available clock sources are the primary clock, defined by the FOSC<1:0> Configuration bits, the secondary clock (Timer1 oscillator) and the internal oscillator. The clock source changes after one or more of the bits are written to, following a brief clock transition interval.

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

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

The use of the flag and control bits in the OSCCON register is discussed in more detail in **Section 4.0 "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.

3.3.1.1 System Clock Selection and the FOSC2 Configuration Bit

The SCS bits are cleared on all forms of Reset. In the device's default configuration, this means the primary oscillator defined by FOSC<1:0> (that is, one of the HS or EC modes) is used as the primary clock source on device Resets.

The default clock configuration on Reset can be changed with the FOSC2 Configuration bit. This bit determines whether the external or internal oscillator will be the default clock source on subsequent device Resets. By extension, it also has the effect of determining the clock source selected when SCS<1:0> are in their Reset state (= 00). When FOSC2 = 1 (default), the oscillator source defined by FOSC<1:0> is selected whenever SCS<1:0> = 00. When FOSC2 = 0, the internal oscillator block is selected whenever SCS<2:1> = 00.

In those cases when the internal oscillator block is the default clock on Reset, the Fast RC oscillator (INTOSC) will be used as the device clock source. It will initially start at 1 MHz; the postscaler selection that corresponds to the Reset value of the IRCF<2:0> bits ('100').

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

Note that either the primary clock or the internal oscillator will have two bit setting options for the possible values of SCS<1:0>, at any given time, depending on the setting of FOSC2.

3.3.2 OSCILLATOR TRANSITIONS

PIC18F85J11 family devices contain circuitry to prevent clock "glitches" when switching between clock sources. A short pause in the device clock occurs during the clock switch. The length of this pause is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

Clock transitions are discussed in greater detail in **Section 4.1.2 "Entering Power-Managed Modes"**.

3.4 External Oscillator Modes

3.4.1 CRYSTAL OSCILLATOR/CERAMIC RESONATORS (HS MODES)

In HS or HSPLL Oscillator modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. Figure 3-2 shows the pin connections.

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

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

TABLE 3-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Typical Capacitor Values Used:			
Mode	**Freq.**	**OSC1**	**OSC2**
HS	8.0 MHz	27 pF	27 pF
	16.0 MHz	22 pF	22 pF

Capacitor values are for design guidance only.

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. Refer to the following application notes for oscillator-specific information:

- *AN588, "PIC® Microcontroller Oscillator Design Guide"*
- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices"*
- *AN849, "Basic PIC® Oscillator Design"*
- *AN943, "Practical PIC® Oscillator Analysis and Design"*
- *AN949, "Making Your Oscillator Work"*

See the notes following Table 3-2 for additional information.

TABLE 3-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq.	Typical Capacitor Values Tested:	
		C1	**C2**
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

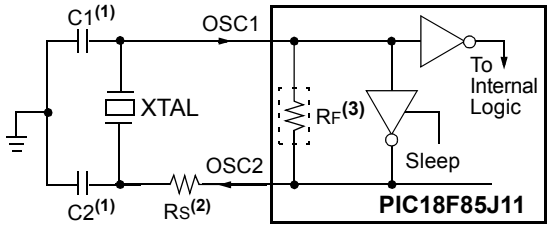
Capacitor values are for design guidance only.

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.

Refer to the Microchip application notes cited in Table 3-1 for oscillator-specific information. Also see the notes following this table for additional information.

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

FIGURE 3-2: CRYSTAL/CERAMIC RESONATOR OPERATION (HS OR HSPLL CONFIGURATION)



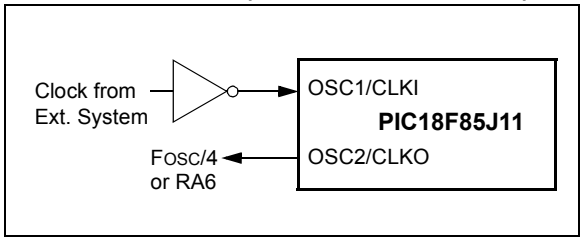
Note 1:	See Table 3-1 and Table 3-2 for initial values of C1 and C2.
2:	A series resistor (RS) may be required for AT strip cut crystals.
3:	RF varies with the oscillator mode chosen.

3.4.2 EXTERNAL CLOCK INPUT (EC MODES)

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

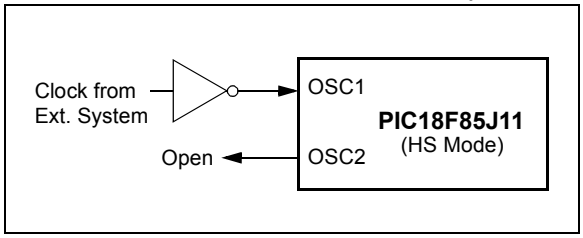
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 3-3 shows the pin connections for the EC Oscillator mode.

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



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

FIGURE 3-4: EXTERNAL CLOCK INPUT OPERATION (HS OSC CONFIGURATION)

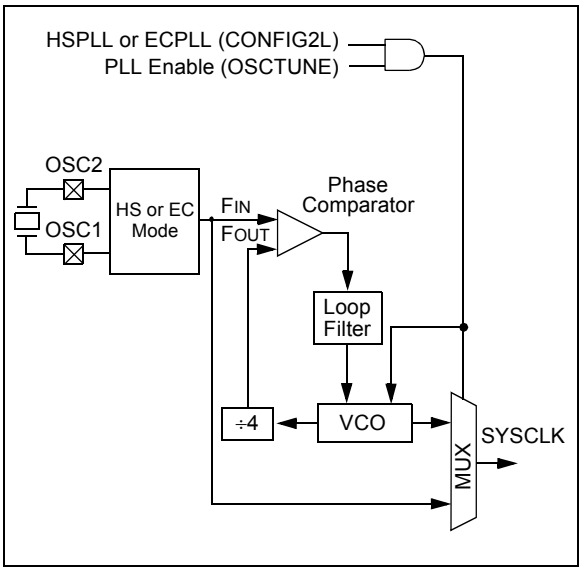


3.4.3 PLL FREQUENCY MULTIPLIER

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

The HSPLL and ECPLL modes provide the ability to selectively run the device at 4 times the external oscillating source to produce frequencies up to 40 MHz. The PLL is enabled by programming the FOSC<2:0> Configuration bits (CONFIG2L<2:0>) to either '110' (for ECPLL) or '100' (for HSPLL). In addition, the PLLEN bit (OSCTUNE<6>) must also be set. Clearing PLLEN disables the PLL, regardless of the chosen oscillator configuration. It also allows additional flexibility for controlling the application's clock speed in software.

FIGURE 3-5: PLL BLOCK DIAGRAM



3.5 Internal Oscillator Block

The PIC18F85J11 family of devices includes 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 an external oscillator circuit on the OSC1 and/or OSC2 pins.

The main output is the Fast RC oscillator, or INTOSC, 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. INTOSC is enabled when a clock frequency from 125 kHz to 8 MHz is selected. The INTOSC output can also be enabled when 31 kHz is selected, depending on the INTSRC bit (OSCTUNE<7>).

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

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

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

The clock source frequency (INTOSC direct, INTOSC with postscaler or INTRC direct) is selected by configuring the IRCF bits of the OSCCON register. The default frequency on device Resets is 1 MHz.

3.5.1 OSC1 AND OSC2 PIN CONFIGURATION

Whenever the internal oscillator is configured as the default clock source (FOSC2 = 0), the OSC1 and OSC2 pins are reconfigured automatically as port pins, RA6 and RA7. In this mode, they function as general digital I/O. All oscillator functions on the pins are disabled.

3.5.2 INTERNAL OSCILLATOR OUTPUT FREQUENCY AND TUNING

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8 MHz. It can be adjusted in the user's application by writing to TUN<5:0> (OSCTUNE<5:0>) in the OSCTUNE register (Register 3-2).

When the OSCTUNE register is modified, the INTOSC frequency will begin shifting to the new frequency. The oscillator will stabilize within 1 ms. Code execution continues during this shift. There is no indication that the shift has occurred.

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 or vice versa. The frequency of INTRC is not affected by OSCTUNE.

3.5.3 INTOSC FREQUENCY DRIFT

The INTOSC frequency may drift as V_{DD} or temperature changes, and can affect the controller operation in a variety of ways. It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This will have no effect on the INTRC clock source frequency.

Tuning INTOSC 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 shown here.

3.5.3.1 Compensating with the EUSART

An adjustment may be required when the EUSART begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the device clock frequency is too high. To adjust for this, decrement the value in OSCTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low. To compensate, increment OSCTUNE to increase the clock frequency.

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

3.5.3.3 Compensating with the CCP Module in Capture Mode

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

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

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

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

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

In RC_RUN and RC_IDLE modes, the internal oscillator provides the device clock source. The 31 kHz INTRC output can be used directly to provide the clock and may be enabled to support various special features, regardless of the power-managed mode (see **Section 23.2 "Watchdog Timer (WDT)"** through **Section 23.5 "Fail-Safe Clock Monitor"** for more information on WDT, Fail-Safe Clock Monitor and Two-Speed Start-up).

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

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

3.7 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 5.6 "Power-up Timer (PWRT)"**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on power-up (parameter 33, Table 26-12). It is always enabled.

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

There is a delay of interval TCSD (parameter 38, Table 26-12), following POR, while the controller becomes ready to execute instructions.

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

Oscillator Mode	OSC1 Pin	OSC2 Pin
EC, ECPLL	Floating, pulled by external clock	At logic low (clock/4 output)
HS, HSPLL	Feedback inverter disabled at quiescent voltage level	Feedback inverter disabled at quiescent voltage level
INTOSC	I/O pin, RA6, direction controlled by TRISA<6>	I/O pin, RA7, direction controlled by TRISA<7>

Note: See Table 5-2 in **Section 5.0 "Reset"** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

4.0 POWER-MANAGED MODES

The PIC18F85J11 family devices provide the ability to manage power consumption by simply managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. For the sake of managing power in an application, there are three primary modes of operation:

- Run mode
- Idle mode
- Sleep mode

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

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

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires two decisions: if the CPU is to be clocked or not and which clock source is to be used. The IDLEN bit (OSCCON<7>) controls CPU clocking, while the SCS<1:0> bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 4-1.

4.1.1 CLOCK SOURCES

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

- the primary clock, as defined by the FOSC<2:0> Configuration bits
- the secondary clock (Timer1 oscillator)
- the internal oscillator

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one power-managed mode to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source and determine which Run or Idle mode is to be used. Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch may also be subject to clock transition delays. These are discussed in **Section 4.1.3 "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 4-1: POWER-MANAGED MODES

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

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

4.1.3 CLOCK TRANSITIONS AND STATUS INDICATORS

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

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

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

4.1.4 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.

4.2 Run Modes

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

4.2.1 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 23.4 “Two-Speed Start-up”** for details). In this mode, the OSTS bit is set (see **Section 3.2 “Control Registers”**).

4.2.2 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 SCS<1:0> bits to ‘01’. The device clock source is switched to the Timer1 oscillator (see Figure 4-1), the primary oscillator is shut down, the T1RUN bit (T1CON<6>) is set and the OSTS bit is cleared.

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

On transitions from SEC_RUN mode to PRI_RUN mode, the peripherals and CPU continue to be clocked from the Timer1 oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 4-2). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

FIGURE 4-1: TRANSITION TIMING FOR ENTRY TO SEC_RUN MODE

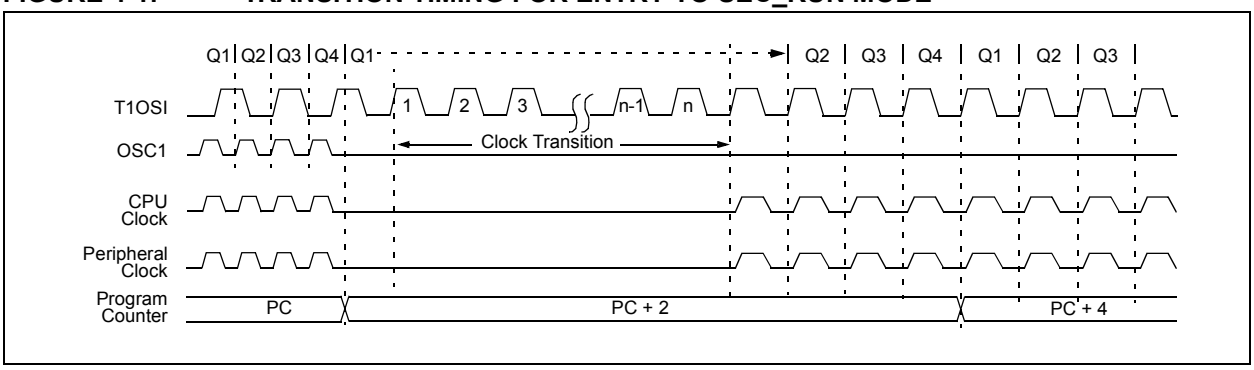
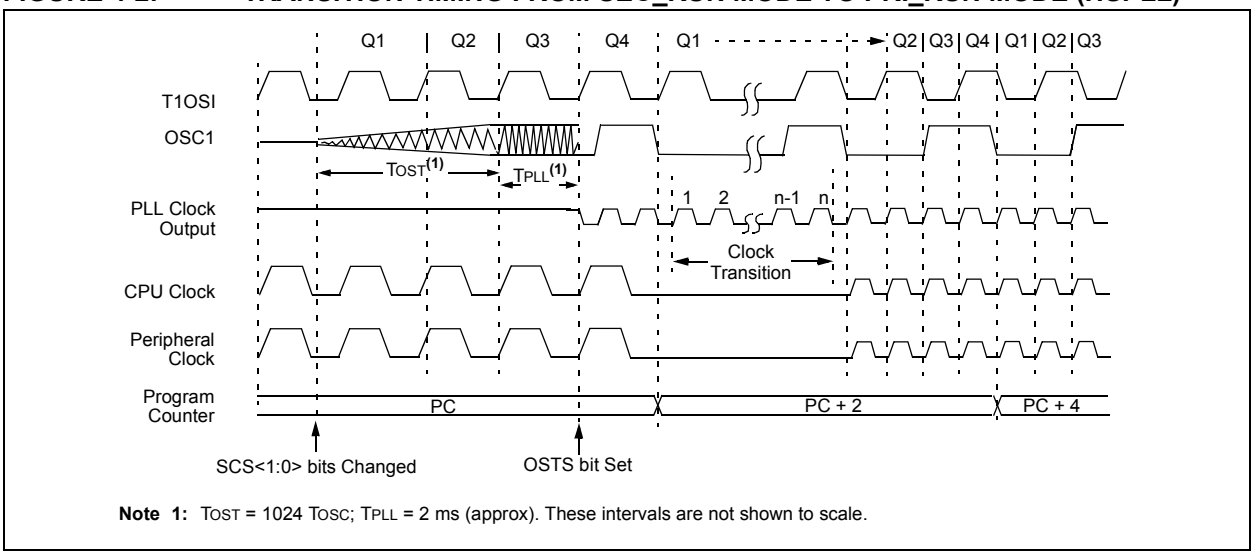


FIGURE 4-2: TRANSITION TIMING FROM SEC_RUN MODE TO PRI_RUN MODE (HSPLL)



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

4.2.3 RC_RUN MODE

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

This mode is entered by setting SCS bits to '11'. When the clock source is switched to the INTRC (see Figure 4-3), the primary oscillator is shut down and the OSTS bit is cleared.

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

FIGURE 4-3: TRANSITION TIMING TO RC_RUN MODE

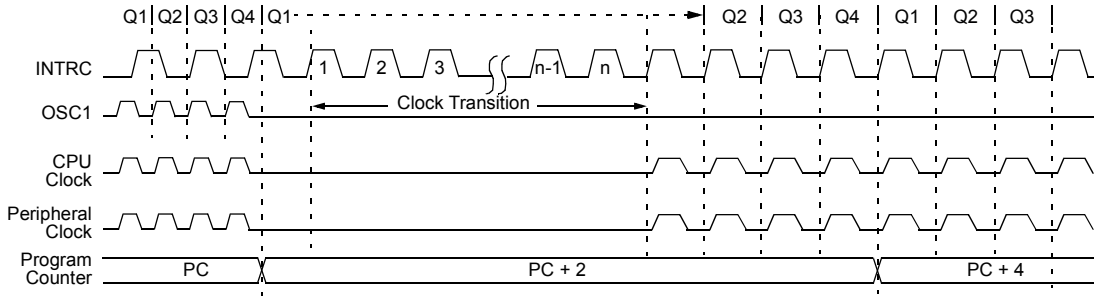
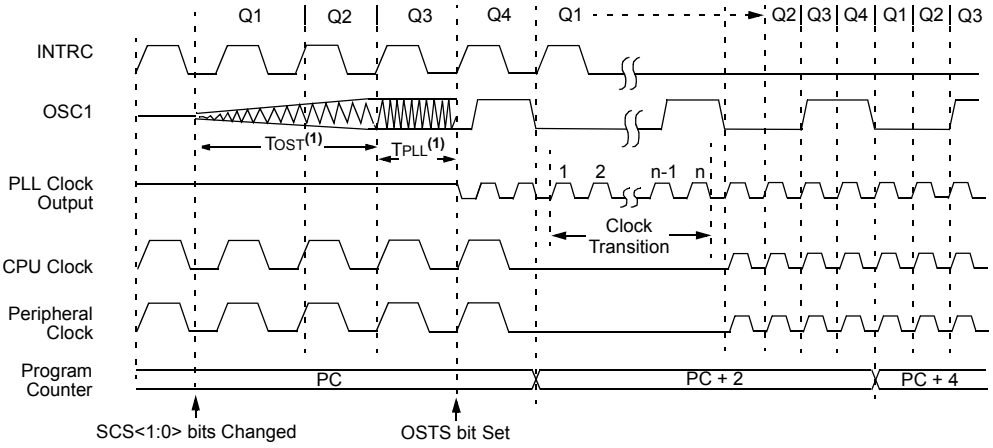


FIGURE 4-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



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

4.3 Sleep Mode

The power-managed Sleep mode is identical to the legacy Sleep mode offered in all other PIC micro-controllers. It is entered by clearing the IDLEN bit (the default state on device Reset) and executing the SLEEP instruction. This shuts down the selected oscillator (Figure 4-5). 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 clock source selected by the SCS<1:0> bits becomes ready (see Figure 4-6), or it will be clocked from the internal oscillator if either the Two-Speed Start-up or the Fail-Safe Clock Monitor is enabled (see **Section 23.0 "Special Features of the CPU"**). In either case, the OSTS bit is set when the primary clock is providing the device clocks. The IDLEN and SCS bits are not affected by the wake-up.

4.4 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 SCS<1:0> bits; however, the CPU will not be clocked. The clock source status bits are not affected. Setting IDLEN and executing a SLEEP instruction provides a quick method of switching from a given Run mode to its corresponding Idle mode.

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

Since the CPU is not executing instructions, the only exits from any of the Idle modes are by interrupt, WDT time-out or a Reset. When a wake event occurs, CPU execution is delayed by an interval of TCSD (parameter 38, Table 26-12) 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 SCS<1:0> bits.

FIGURE 4-5: TRANSITION TIMING FOR ENTRY TO SLEEP MODE

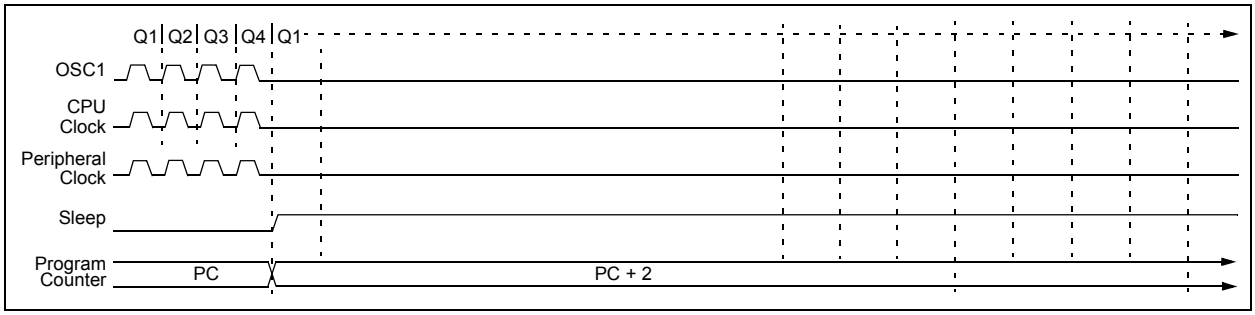
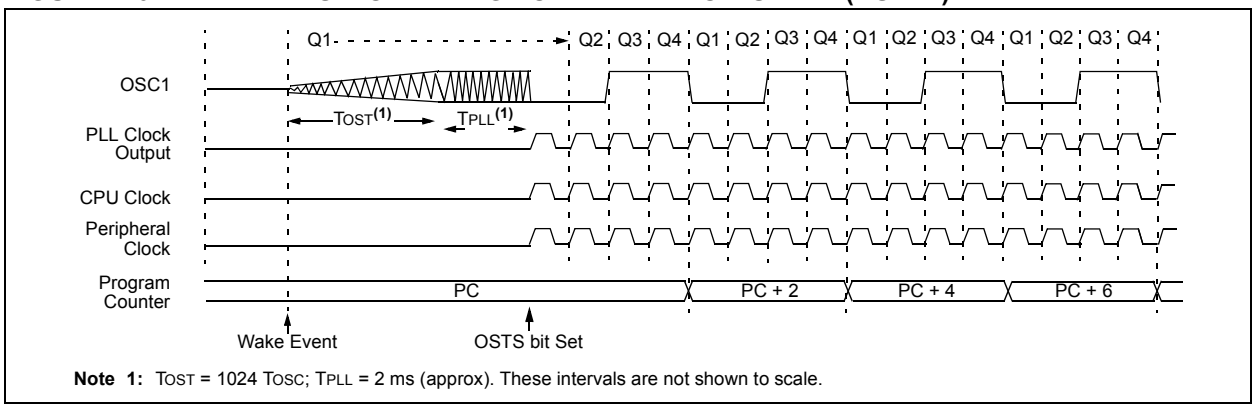


FIGURE 4-6: TRANSITION TIMING FOR WAKE FROM SLEEP (HSPLL)



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

4.4.1 PRI_IDLE MODE

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

PRI_IDLE mode is entered from PRI_RUN mode by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, set IDLEN first, then set the SCS bits to ‘10’ and execute SLEEP. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the FOSC<1:0> Configuration bits. The OSTS bit remains set (see Figure 4-7).

When a wake event occurs, the CPU is clocked from the primary clock source. A delay of interval TCSD is required between the wake event and when code execution starts. This is required to allow the CPU to become ready to execute instructions. After the wake-up, the OSTS bit remains set. The IDLEN and SCS bits are not affected by the wake-up (see Figure 4-8).

4.4.2 SEC_IDLE MODE

In SEC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered from SEC_RUN by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, set IDLEN first, then set SCS<1:0> to ‘01’ and execute SLEEP. When the clock source is switched to the Timer1 oscillator, the primary oscillator is shut down, the OSTS bit is cleared and the T1RUN bit is set.

When a wake event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After an interval of TCSD following the wake event, the CPU begins executing code being clocked by the Timer1 oscillator. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run (see Figure 4-8).

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.

FIGURE 4-7: TRANSITION TIMING FOR ENTRY TO IDLE MODE

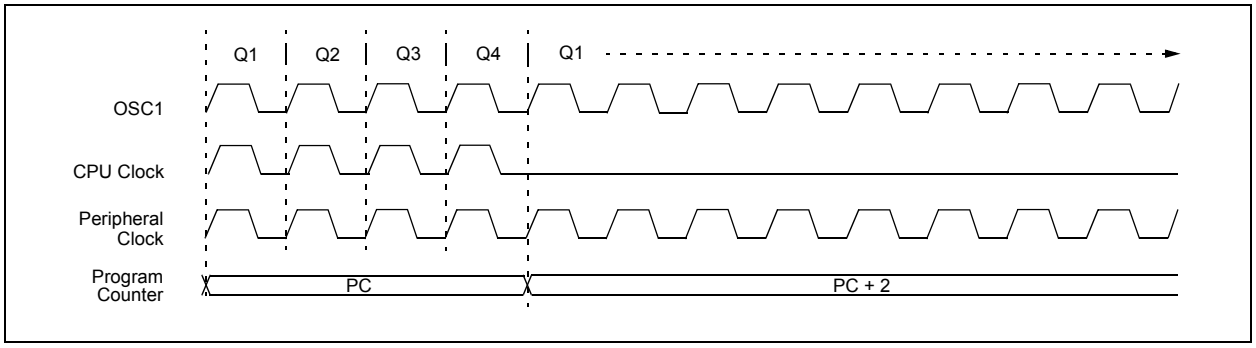
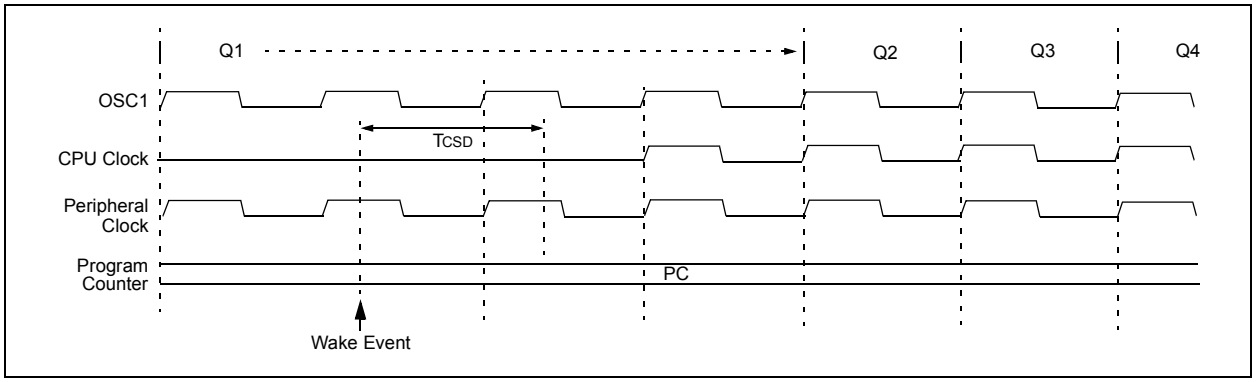


FIGURE 4-8: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE



4.4.3 RC_IDLE MODE

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

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

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

4.5 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 mode sections (see **Section 4.2 "Run Modes", Section 4.3 "Sleep Mode"** and **Section 4.4 "Idle Modes"**).

4.5.1 EXIT BY INTERRUPT

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

On all exits from Idle or Sleep modes by interrupt, code execution branches to the interrupt vector if the GIE/GIEH bit (INTCON<7>) is set. Otherwise, code execution continues or resumes without branching (see **Section 10.0 "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.

4.5.2 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 4.2 "Run Modes"** and **Section 4.3 "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 Watchdog Timer and postscaler are cleared by one of the following events:

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

4.5.3 EXIT BY RESET

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

4.5.4 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 either the EC or ECPLL mode.

In these instances, the primary clock source either does not require an oscillator start-up delay, since it is already running (PRI_IDLE), or normally does not require an oscillator start-up delay (EC). However, a fixed delay of interval 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.

NOTES:

5.0 RESET

The PIC18F85J11 family of devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset during normal operation
c) $\overline{\text{MCLR}}$ Reset during power-managed modes
d) Watchdog Timer (WDT) Reset (during execution)
e) Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) 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 6.1.6.4 "Stack Full and Underflow Resets"**. WDT Resets are covered in **Section 23.2 "Watchdog Timer (WDT)"**.

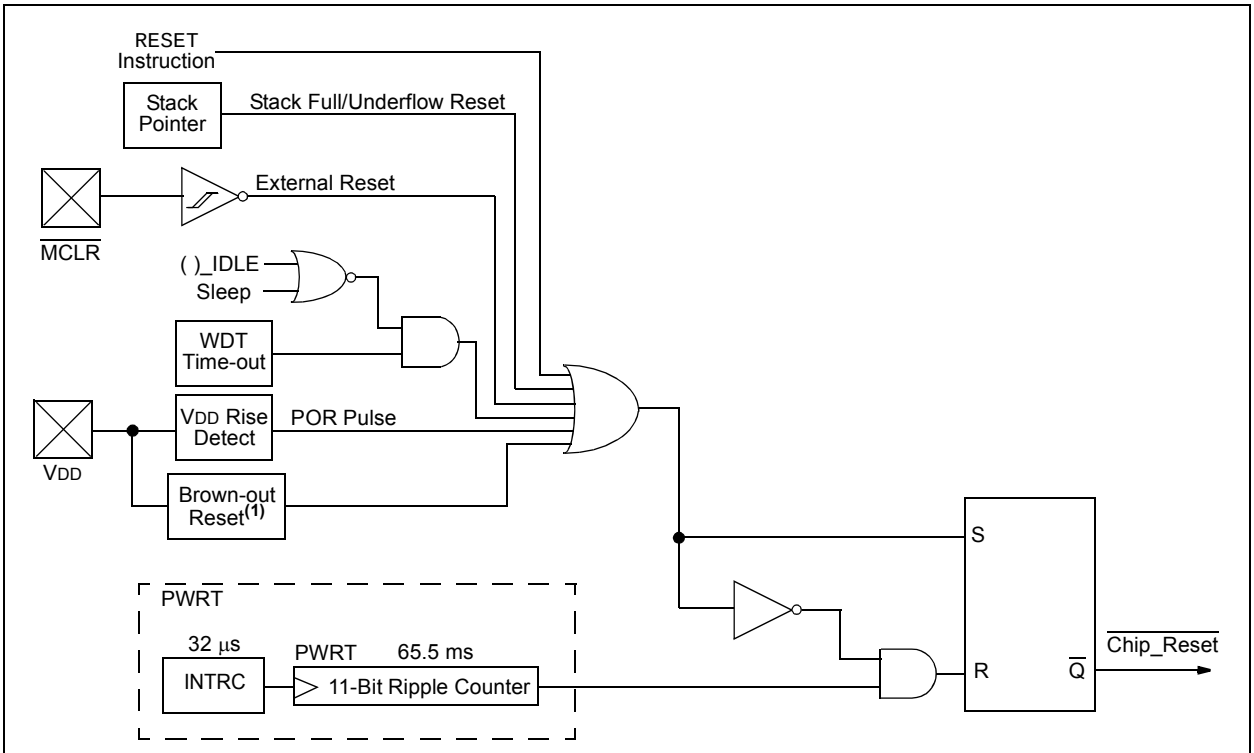
A simplified block diagram of the on-chip Reset circuit is shown in Figure 5-1.

5.1 RCON Register

Device Reset events are tracked through the RCON register (Register 5-1). 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 5.7 "Reset State of Registers"**.

The RCON register also has a control bit for setting interrupt priority (IPEN). Interrupt priority is discussed in **Section 10.0 "Interrupts"**.

FIGURE 5-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: The ENVREG pin must be tied high to enable Brown-out Reset. The Brown-out Reset is provided by the on-chip voltage regulator when there is insufficient source voltage to maintain regulation.

REGISTER 5-1: RCON: RESET CONTROL REGISTER

R/W-0	U-0	R/W-1	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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 (PIC16XXXX Compatibility mode)

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

bit 5 **$\overline{CM}$:** Configuration Mismatch Flag bit
1 = A Configuration Mismatch Reset has not occurred
0 = A Configuration Mismatch Reset has occurred. Must be set in software once the Reset occurs.

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

2: If the on-chip voltage regulator is disabled, $\overline{BOR}$ remains '0' at all times. See **Section 5.4.1 "Detecting BOR"** for more information.

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

5.2 Master Clear ($\overline{MCLR}$)

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

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

5.3 Power-on Reset (POR)

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

To take advantage of the POR circuitry, tie the $\overline{MCLR}$ pin through a resistor (1 kΩ to 10 kΩ) to VDD. This will eliminate external RC components usually needed to create a Power-on Reset delay. A minimum rise rate for VDD is specified (parameter D004). For a slow rise time, see Figure 5-2.

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.

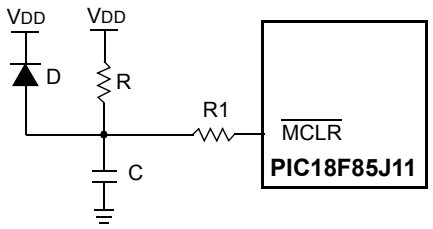
Power-on Reset events are captured by the $\overline{POR}$ bit (RCON<1>). The state of the bit is set to '0' whenever a Power-on Reset occurs; it does not change for any other Reset event. $\overline{POR}$ is not reset to '1' by any hardware event. To capture multiple events, the user manually resets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

5.4 Brown-out Reset (BOR)

The PIC18F85J11 family of devices incorporates a simple BOR function when the internal regulator is enabled (ENVREG pin is tied to VDD). The voltage regulator will trigger a Brown-out Reset when output of the regulator to the device core approaches the voltage at which the device is unable to run at full speed. The BOR circuit also keeps the device in Reset as VDD rises, until the regulator's output level is sufficient for full-speed operation.

Once a BOR has occurred, the Power-up Timer will keep the chip in Reset for TPWRT (parameter 33). If VDD drops below the threshold for full-speed operation 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 to the point where regulator output is sufficient, the Power-up Timer will execute the additional time delay.

FIGURE 5-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



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

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

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

5.4.1 DETECTING BOR

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

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

5.5 Configuration Mismatch (CM)

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

In PIC18FXXJXX Flash devices, the device Configuration registers (located in the configuration memory space) are continuously monitored during operation by comparing their values to complimentary Shadow registers.

If a mismatch is detected between the two sets of registers, a CM Reset automatically occurs. These events are captured by the $\overline{CM}$ bit (RCON<5>) being set to '0'. This bit does not change for any other Reset event.

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

5.6 Power-up Timer (PWRT)

PIC18F85J11 family devices incorporate an on-chip Power-up Timer (PWRT) to help regulate the Power-on Reset process. The PWRT is always enabled. The main function is to ensure that the device voltage is stable before code is executed.

The Power-up Timer (PWRT) of the PIC18F85J11 family devices is an 11-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of 2048 x 32 µs = 65.6 ms. While the PWRT is counting, the device is held in Reset.

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

5.6.1 TIME-OUT SEQUENCE

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

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

FIGURE 5-3: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO VDD, VDD RISE < TPWRT)

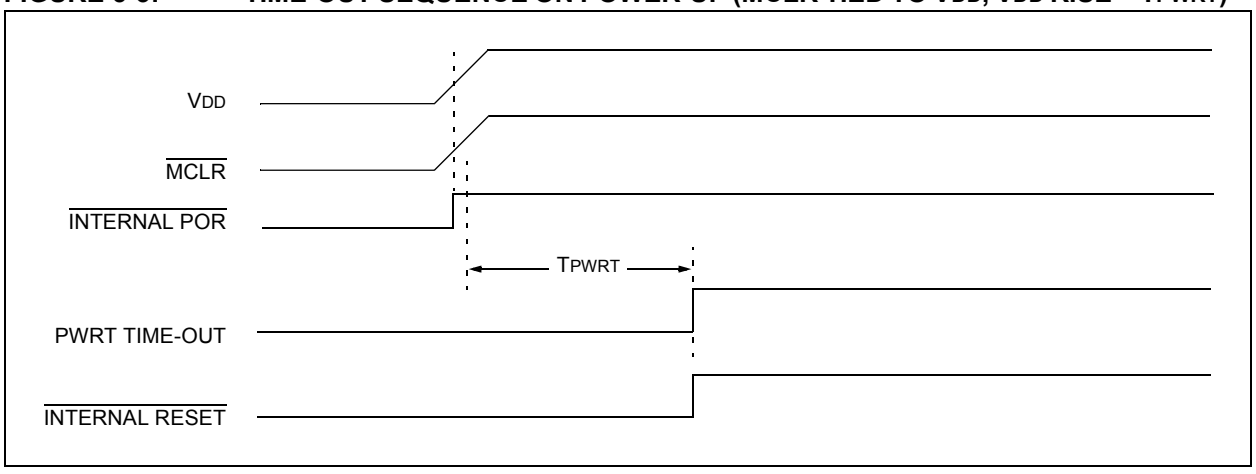


FIGURE 5-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1

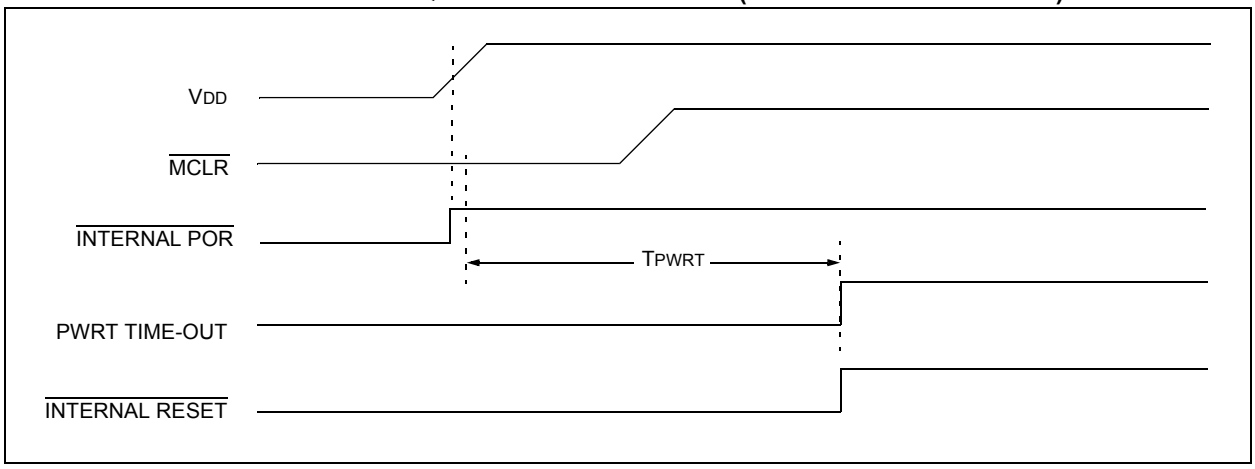


FIGURE 5-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

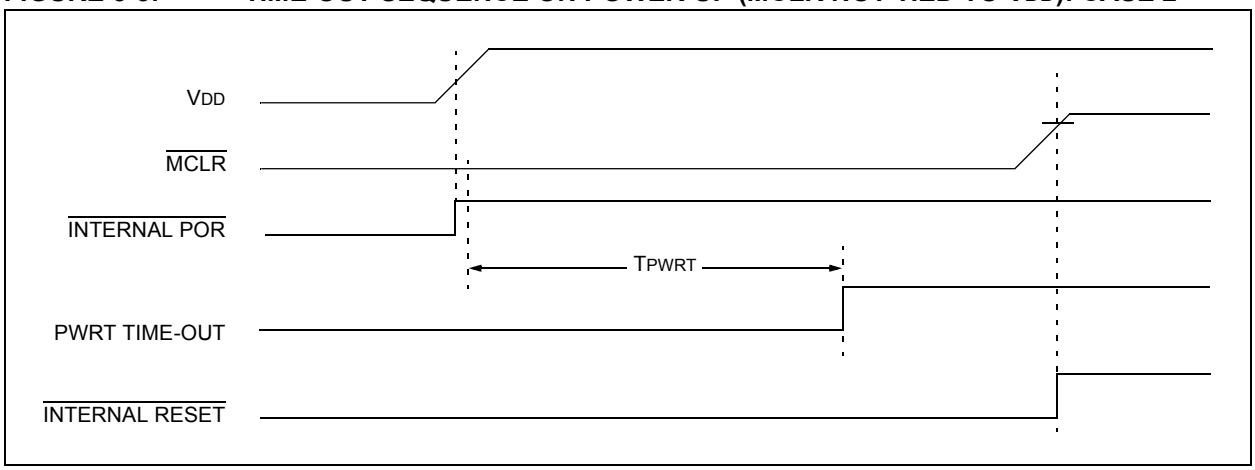
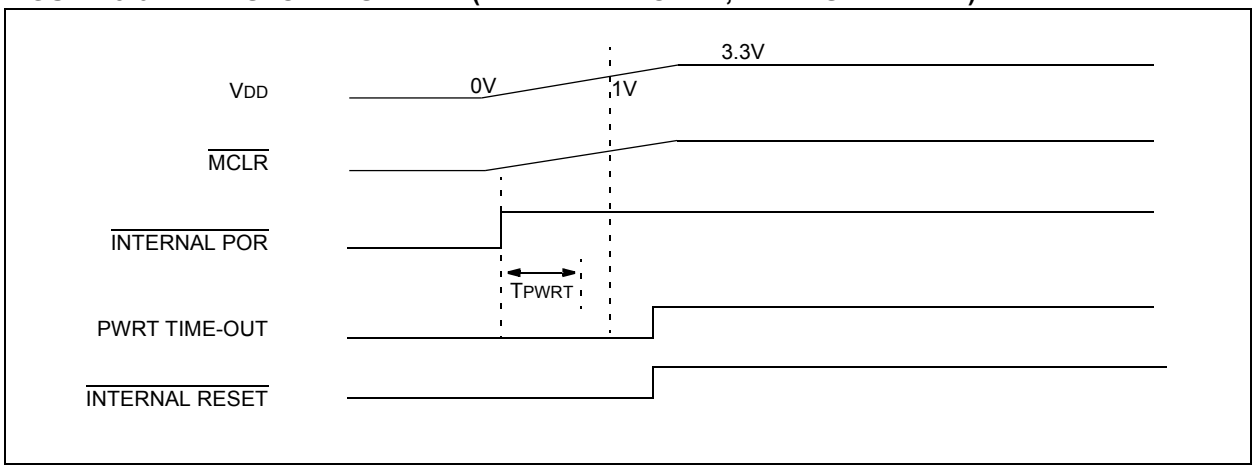


FIGURE 5-6: SLOW RISE TIME (MCLR TIED TO VDD, VDD RISE > TPWRT)



5.7 Reset State of Registers

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

Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. Status bits from the RCON register, $\overline{CM}$, $\overline{RI}$, $\overline{TO}$, $\overline{PD}$, $\overline{POR}$ and $\overline{BOR}$, are set or cleared differently in different Reset situations, as indicated in Table 5-1. These bits are used in software to determine the nature of the Reset.

Table 5-2 describes the Reset states for all of the Special Function Registers. These are categorized by Power-on and Brown-out Resets, Master Clear and WDT Resets and WDT wake-ups.

TABLE 5-1: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER

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

Legend: u = unchanged

Note 1: When the wake-up is due to an interrupt and the GIEH or GIEL bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
TOSU	PIC18F6XJ11	PIC18F8XJ11	---0 0000	---0 0000	---0 uuuu(1)
TOSH	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F6XJ11	PIC18F8XJ11	uu-0 0000	00-0 0000	uu-u uuuu(1)
PCLATU	PIC18F6XJ11	PIC18F8XJ11	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6XJ11	PIC18F8XJ11	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6XJ11	PIC18F8XJ11	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F6XJ11	PIC18F8XJ11	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTINC0	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTDEC0	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PREINC0	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PLUSW0	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
FSR0H	PIC18F6XJ11	PIC18F8XJ11	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTINC1	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTDEC1	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PREINC1	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PLUSW1	PIC18F6XJ11	PIC18F8XJ11	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: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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

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

4: See Table 5-1 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 as '0'.

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
FSR1H	PIC18F6XJ11	PIC18F8XJ11	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6XJ11	PIC18F8XJ11	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTINC2	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
POSTDEC2	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PREINC2	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
PLUSW2	PIC18F6XJ11	PIC18F8XJ11	N/A	N/A	N/A
FSR2H	PIC18F6XJ11	PIC18F8XJ11	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F6XJ11	PIC18F8XJ11	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6XJ11	PIC18F8XJ11	0100 q000	0100 q000	uuuu quuu
WDTCON	PIC18F6XJ11	PIC18F8XJ11	0--- ---0	0--- ---0	u--- ---u
RCON(4)	PIC18F6XJ11	PIC18F8XJ11	0-11 11q0	0-uq qquu	u-uu qquu
TMR1H	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6XJ11	PIC18F8XJ11	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6XJ11	PIC18F8XJ11	-000 0000	-000 0000	-uuu uuuu
SSPBUF	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
SSPCON1	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
SSPCON2	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu

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

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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

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

4: See Table 5-1 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 as '0'.

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
ADRESH	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6XJ11	PIC18F8XJ11	0-00 0000	0-00 0000	u-uu uuuu
ADCON1	PIC18F6XJ11	PIC18F8XJ11	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6XJ11	PIC18F8XJ11	0-00 0000	0-00 0000	u-uu uuuu
CVRCON	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6XJ11	PIC18F8XJ11	0000 0111	0000 0111	uuuu uuuu
TMR3H	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6XJ11	PIC18F8XJ11	0000 0000	uuuu uuuu	uuuu uuuu
PSPCON	PIC18F6XJ11	PIC18F8XJ11	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TXSTA1	PIC18F6XJ11	PIC18F8XJ11	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F6XJ11	PIC18F8XJ11	0000 000x	0000 000x	uuuu uuuu
EECON2	PIC18F6XJ11	PIC18F8XJ11	---- ----	---- ----	---- ----
EECON1	PIC18F6XJ11	PIC18F8XJ11	---0 x00-	---0 u00-	---0 u00-
IPR3	PIC18F6XJ11	PIC18F8XJ11	--00 -11-	--00 -11-	--uu -uu-
PIR3	PIC18F6XJ11	PIC18F8XJ11	--00 -00-	--00 -00-	--uu -00-(3)
PIE3	PIC18F6XJ11	PIC18F8XJ11	--00 -00-	--00 -00-	--uu -00-
IPR2	PIC18F6XJ11	PIC18F8XJ11	11-- 111-	11-- 111-	uu-- uuu-
PIR2	PIC18F6XJ11	PIC18F8XJ11	00-- 000-	00-- 000-	uu-- uuu-(3)
PIE2	PIC18F6XJ11	PIC18F8XJ11	00-- 000-	00-- 000-	uu-- uuu-
IPR1	PIC18F6XJ11	PIC18F8XJ11	1111 1-11	1111 1-11	uuuu u-uu
PIR1	PIC18F6XJ11	PIC18F8XJ11	0000 0-00	0000 0-00	uuuu u-uu(3)
PIE1	PIC18F6XJ11	PIC18F8XJ11	0000 0-00	0000 0-00	uuuu u-uu
MEMCON	PIC18F6XJ11	PIC18F8XJ11	0-00 --00	0-00 --00	u-uu --uu
OSCTUNE	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu

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

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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

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

4: See Table 5-1 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 as '0'.

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
TRISJ	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6XJ11	PIC18F8XJ11	0001 1111	0001 1111	uuuu uuuu
TRISF	PIC18F6XJ11	PIC18F8XJ11	1111 111-	1111 111-	uuuu uuu-
TRISE	PIC18F6XJ11	PIC18F8XJ11	1111 1-11	1111 1-11	uuuu u-uu
TRISD	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6XJ11	PIC18F8XJ11	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	PIC18F6XJ11	PIC18F8XJ11	1111 1111(5)	1111 1111(5)	uuuu uuuu(5)
LATJ	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6XJ11	PIC18F8XJ11	00-x xxxx	00-u uuuu	uu-u uuuu
LATF	PIC18F6XJ11	PIC18F8XJ11	xxxx xxx-	uuuu uuu-	uuuu uuu-
LATE	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5)	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx(5)	uuuu uuuu(5)	uuuu uuuu(5)
PORTJ	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTG	PIC18F6XJ11	PIC18F8XJ11	000x xxxx	000u uuuu	000u uuuu
PORTF	PIC18F6XJ11	PIC18F8XJ11	xxxx xxx-	uuuu uuu-	uuuu uuu-
PORTE	PIC18F6XJ11	PIC18F8XJ11	xxxx x-xx	uuuu u-uu	uuuu u-uu
PORTD	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5)	PIC18F6XJ11	PIC18F8XJ11	xx0x 0000(5)	uu0u 0000(5)	uuuu uuuu(5)
SPBRGH1	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6XJ11	PIC18F8XJ11	0100 0-00	0100 0-00	uuuu u-uu
CCPR1H	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6XJ11	PIC18F8XJ11	--00 0000	--00 0000	--uu uuuu

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

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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

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

4: See Table 5-1 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 as '0'.

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
CCPR2H	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6XJ11	PIC18F8XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6XJ11	PIC18F8XJ11	--00 0000	--00 0000	--uu uuuu
SPBRG2	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XJ11	PIC18F8XJ11	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6XJ11	PIC18F8XJ11	0000 -010	0000 -010	uuuu -uuu
RCSTA2	PIC18F6XJ11	PIC18F8XJ11	0000 000x	0000 000x	uuuu uuuu

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

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

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

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

4: See Table 5-1 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 as '0'.

NOTES:

6.0 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.

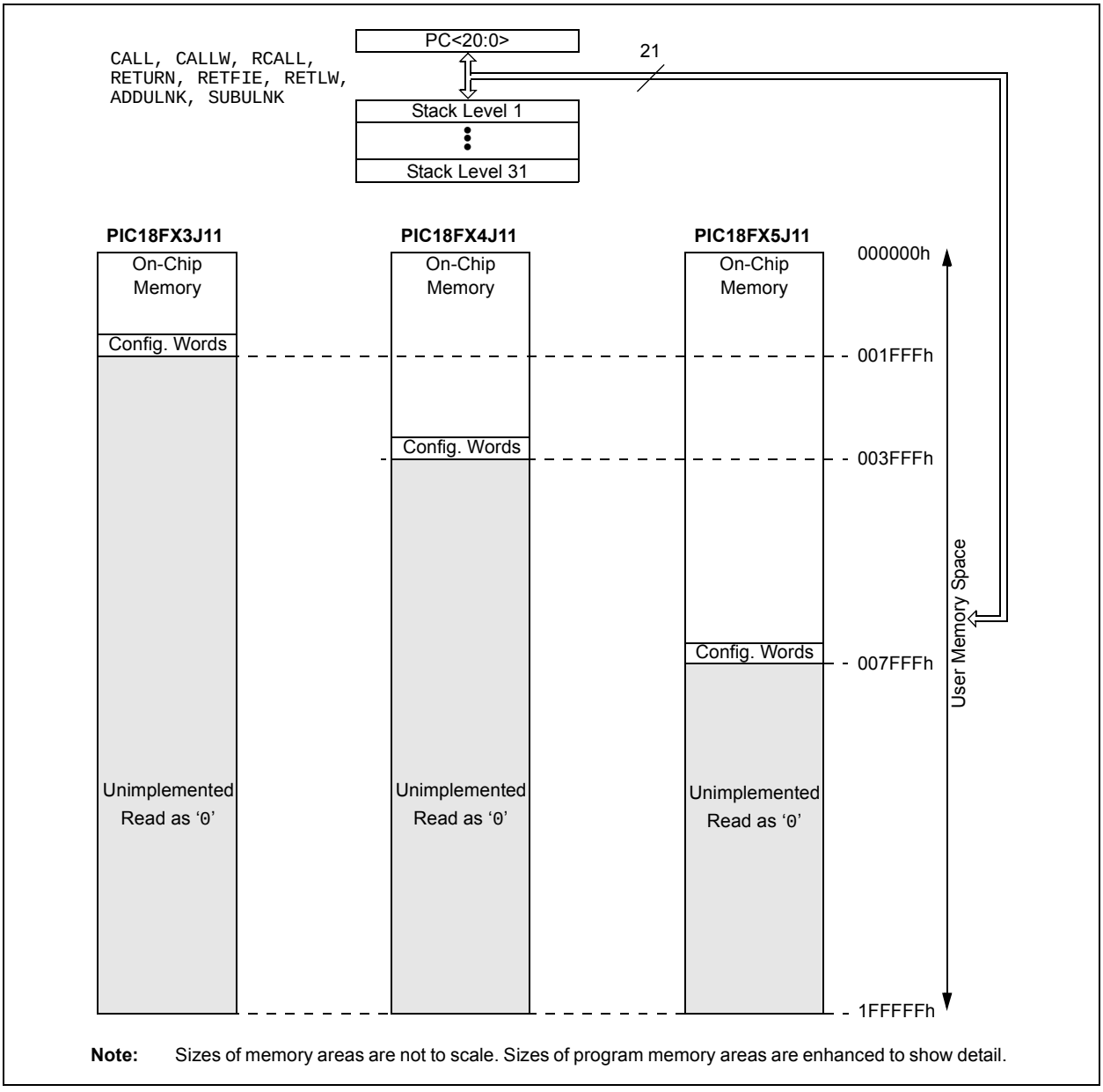
Additional detailed information on the operation of the Flash program memory is provided in **Section 7.0 "Flash Program Memory"**.

6.1 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 entire PIC18F85J11 family offers a range of on-chip Flash program memory sizes, from 8 Kbytes (up to 4,096 single-word instructions) to 32 Kbytes (32,768 single-word instructions). The program memory maps for individual family members are shown in Figure 6-1.

FIGURE 6-1: MEMORY MAPS FOR PIC18F85J11 FAMILY DEVICES



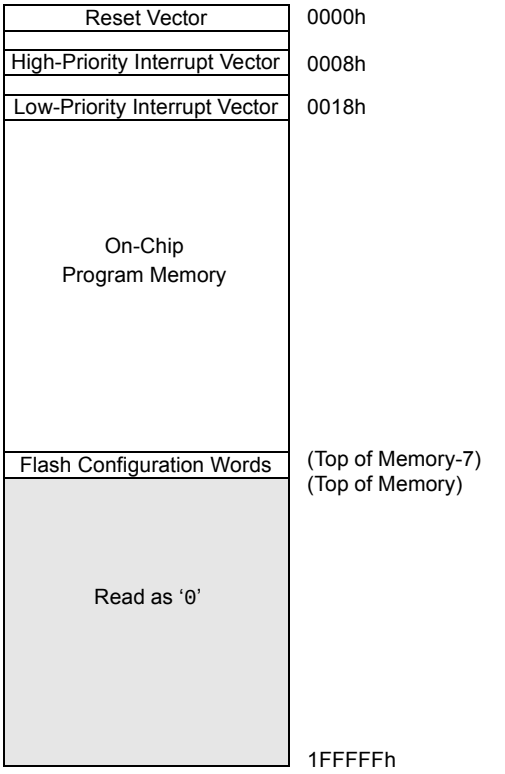
Note: Sizes of memory areas are not to scale. Sizes of program memory areas are enhanced to show detail.

6.1.1 HARD MEMORY VECTORS

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

PIC18 devices also have two interrupt vector addresses for the handling of high-priority and low-priority interrupts. The high-priority interrupt vector is located at 0008h and the low-priority interrupt vector is at 0018h. Their locations in relation to the program memory map are shown in Figure 6-2.

FIGURE 6-2: HARD VECTOR AND CONFIGURATION WORD LOCATIONS FOR PIC18F85J11 FAMILY FAMILY DEVICES



Legend: (Top of Memory) represents upper boundary of on-chip program memory space (see Figure 6-1 for device-specific values). Shaded area represents unimplemented memory. Areas are not shown to scale.

6.1.2 FLASH CONFIGURATION WORDS

Because PIC18F85J11 family devices do not have persistent configuration memory, the top four words of on-chip program memory are reserved for configuration information. On Reset, the configuration information is copied into the Configuration registers.

The Configuration Words are stored in their program memory location in numerical order, starting with the lower byte of CONFIG1 at the lowest address and ending with the upper byte of CONFIG4. For these devices, only Configuration Words, CONFIG1 through CONFIG3, are used; CONFIG4 is reserved. The actual addresses of the Flash Configuration Word for devices in the PIC18F85J11 family are shown in Table 6-1. Their location in the memory map is shown with the other memory vectors in Figure 6-2.

Additional details on the device Configuration Words are provided in **Section 23.1 "Configuration Bits"**.

TABLE 6-1: FLASH CONFIGURATION WORD FOR PIC18F85J11 FAMILY DEVICES

Device	Program Memory (Kbytes)	Configuration Word Addresses
PIC18F63J11	8	1FF8h to 1FFFh
PIC18F83J11		
PIC18F64J11	16	3FF8h to 3FFFh
PIC18F84J11		
PIC18F65J11	32	7FF8h to 7FFFh
PIC18F85J11		

6.1.3 PIC18F8XJ11 PROGRAM MEMORY MODES

The 80-pin devices in this family can address up to a total of 2 Mbytes of program memory. This is achieved through the external memory bus. There are two distinct operating modes available to the controllers:

- Microcontroller (MC)
- Extended Microcontroller (EMC)

The Program Memory mode is determined by setting the EMB Configuration bits (CONFIG3L<5:4>), as shown in Register 6-1. (See also **Section 23.1 "Configuration Bits"** for additional details on the device Configuration bits.)

The Program Memory modes operate as follows:

- The **Microcontroller Mode** accesses only on-chip Flash memory. Attempts to read above the top of on-chip memory causes a read of all '0's (a NOP instruction).

  The Microcontroller mode is also the only operating mode available to 64-pin devices.

- The **Extended Microcontroller Mode** allows access to both internal and external program memories as a single block. The device can access its entire on-chip program memory; above this, the device accesses external program memory up to the 2-Mbyte program space limit. Execution automatically switches between the two memories as required.

The setting of the EMB Configuration bits also controls the address bus width of the external memory bus. This is covered in more detail in **Section 8.0 "External Memory Bus"**.

In all modes, the microcontroller has complete access to data RAM.

Figure 6-3 compares the memory maps of the different Program Memory modes. The differences between on-chip and external memory access limitations are more fully explained in Table 6-2.

REGISTER 6-1: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)(1)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT	BW	EMB1	EMB0	EASHFT	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7 **WAIT:** External Bus Wait Enable bit
1 = Wait selections from WAIT<1:0> (MEMCON<5:4>) are unavailable and the device will not wait
0 = Wait is programmed by WAIT<1:0> (MEMCON<5:4>)

bit 6 **BW:** Data Bus Width Select bit
1 = 16-Bit External Bus mode
0 = 8-Bit External Bus mode

bit 5:4 **EMB<1:0>:** External Memory Bus Configuration bits
00 = Extended Microcontroller mode – 20-Bit Address mode
01 = Extended Microcontroller mode – 16-Bit Address mode
10 = Extended Microcontroller mode – 12-Bit Address mode
11 = Microcontroller mode – external bus disabled

bit 3 **EASHFT:** External Address Bus Shift Enable bit
1 = Address shifting enabled – external address bus is shifted to start at 000000h
0 = Address shifting disabled – external address bus reflects the PC value

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

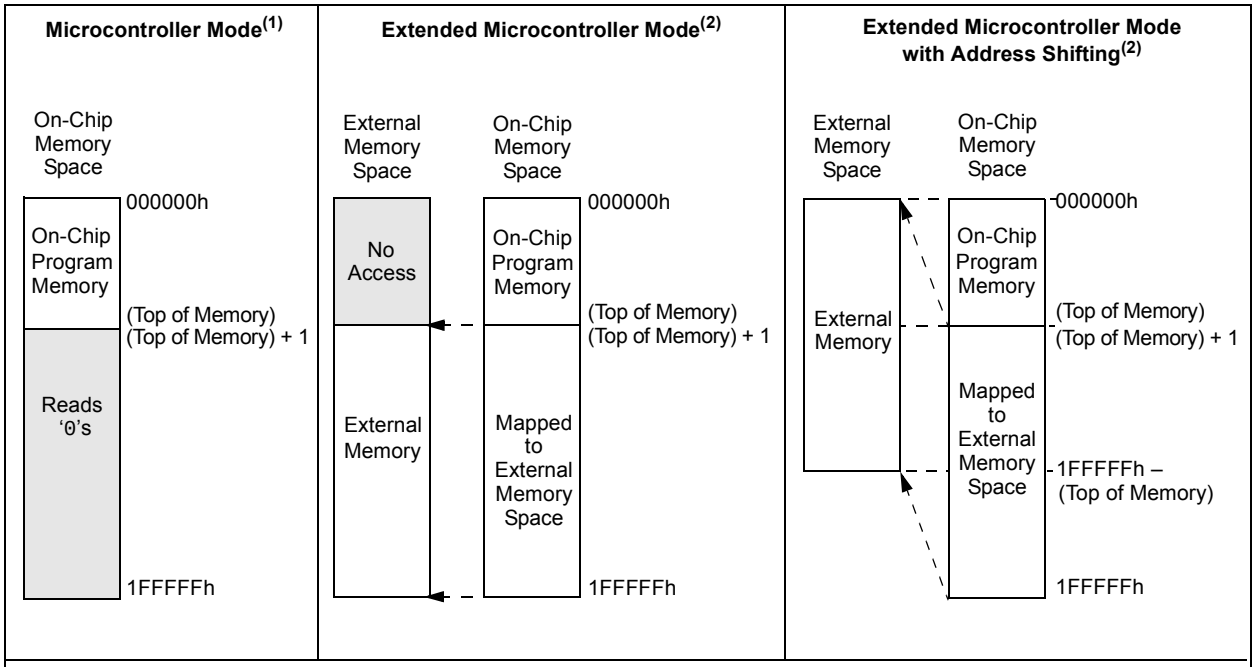
Note 1: CONFIG3L and its associated bits are implemented only in 80-pin devices.

6.1.4 EXTENDED MICROCONTROLLER MODE AND ADDRESS SHIFTING

By default, devices in Extended Microcontroller mode directly present the program counter value on the external address bus for those addresses in the range of the external memory space. In practical terms, this means addresses in the external memory device below the top of on-chip memory are unavailable.

To avoid this, the Extended Microcontroller mode implements an address shifting option to enable automatic address translation. In this mode, addresses presented on the external bus are shifted down by the size of the on-chip program memory and are remapped to start at 0000h. This allows the complete use of the external memory device's memory space.

FIGURE 6-3: MEMORY MAPS FOR PIC18F85J11 FAMILY PROGRAM MEMORY MODES



Legend: (Top of Memory) represents upper boundary of on-chip program memory space (see Figure 6-1 for device-specific values). Shaded areas represent unimplemented, or inaccessible areas, depending on the mode.

Note 1: This mode is the only available mode on 64-pin devices and the default mode on 80-pin devices.

2: These modes are only available on 80-pin devices.

TABLE 6-2: MEMORY ACCESS FOR PIC18F8XJ11 PROGRAM MEMORY MODES

Operating Mode	Internal Program Memory			External Program Memory		
	Execution From	Table Read From	Table Write To	Execution From	Table Read From	Table Write To
Microcontroller	Yes	Yes	Yes	No Access	No Access	No Access
Extended Microcontroller	Yes	Yes	Yes	Yes	Yes	Yes

6.1.5 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 6.1.8.1 "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.

6.1.6 RETURN ADDRESS STACK

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

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer, STKPTR. The stack space is not part of either program or data space. The Stack Pointer is readable and writable and the address on the top of the stack is readable and writable through the Top-of-Stack Special Function Registers. Data can also be pushed to, or popped from the stack, using these registers.

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

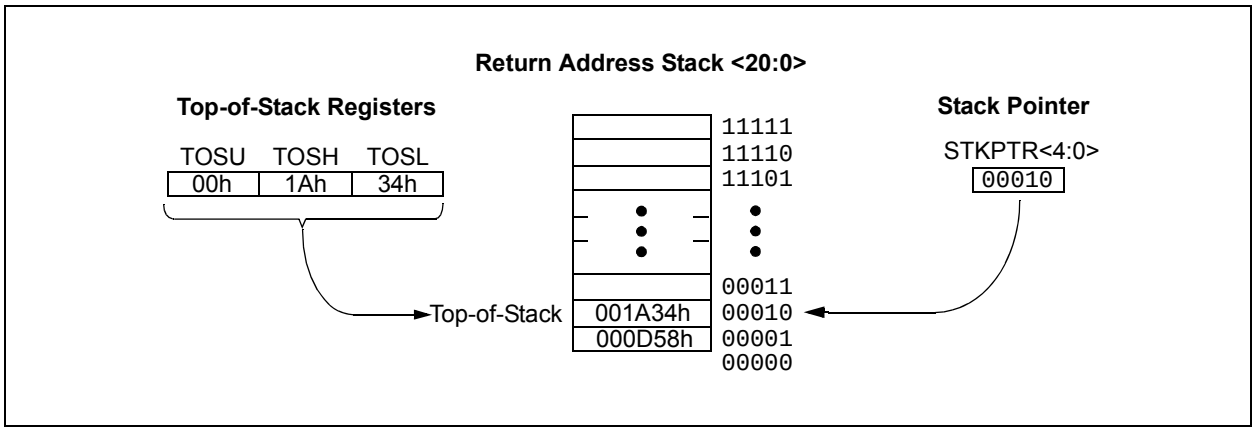
The Stack Pointer is initialized to '00000' after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of '00000'; this is only a Reset value. Status bits indicate if the stack is full, has overflowed or has underflowed.

6.1.6.1 Top-of-Stack Access

Only the top of the return address stack (TOS) is readable and writable. A set of three registers, TOSU:TOSH:TOSL, holds the contents of the stack location pointed to by the STKPTR register (Figure 6-4). This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt (and ADDULNK and SUBULNK instructions if the extended instruction set is enabled), the software can read the pushed value by reading the TOSU:TOSH:TOSL registers. These values can be placed on a user-defined software stack. At return time, the software can return these values to TOSU:TOSH:TOSL and do a return.

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

FIGURE 6-4: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



6.1.6.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 6-2) 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 the STKPTR will remain at 31.

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

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

6.1.6.3 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 6-2: STKPTR: STACK POINTER REGISTER

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

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

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

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

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

bit 4-0 **SP<4:0>:** Stack Pointer Location bits

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

6.1.6.4 Stack Full and Underflow Resets

Device Resets on stack overflow and stack underflow conditions are enabled by setting the STVREN bit in Configuration Register 1L. When STVREN is set, a full or underflow condition 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.

6.1.7 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 6-1 shows a source code example that uses the Fast Register Stack during a subroutine call and return.

EXAMPLE 6-1: 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
```

6.1.8 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

6.1.8.1 Computed GOTO

A computed `GOTO` is accomplished by adding an offset to the program counter. An example is shown in Example 6-2.

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 6-2: COMPUTED `GOTO` USING AN OFFSET VALUE

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

6.1.8.2 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 (TBLPTR) specifies the byte address and the Table Latch (TABLAT) contains the data that is read from the program memory. Data is transferred from program memory one byte at a time.

Table read operation is discussed further in **Section 7.1 “Table Reads and Table Writes”**.

6.2 PIC18 Instruction Cycle

6.2.1 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 6-5.

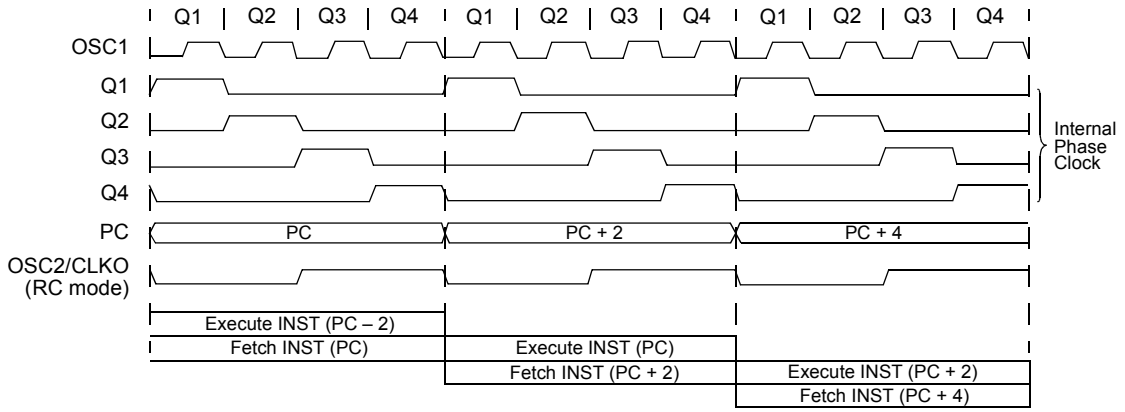
6.2.2 INSTRUCTION FLOW/PIPELINING

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 6-3).

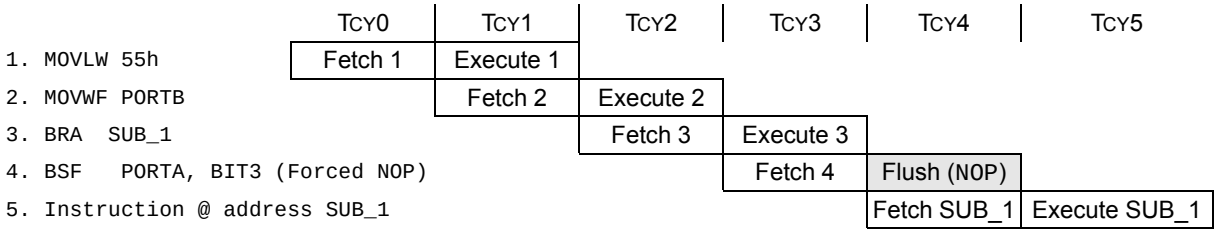
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

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 6-5: CLOCK/INSTRUCTION CYCLE



EXAMPLE 6-3: INSTRUCTION PIPELINE FLOW



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

6.2.3 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 6.1.5 "Program Counter"**).

Figure 6-6 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 6-6 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 25.0 "Instruction Set Summary"** provides further details of the instruction set.

FIGURE 6-6: INSTRUCTIONS IN PROGRAM MEMORY

Program Memory Byte Locations →

			LSB = 1	LSB = 0	Word Address ↓
					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

6.2.4 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 6-4 shows how this works.

Note:	See **Section 6.5 "Program Memory and the Extended Instruction Set"** for information on two-word instructions in the extended instruction set.

EXAMPLE 6-4: 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

6.3 Data Memory Organization

Note: The operation of some aspects of data memory are changed when the PIC18 extended instruction set is enabled. See **Section 6.6 "Data Memory and the Extended Instruction Set"** for more information.

The data memory in PIC18 devices is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4096 bytes of data memory. The memory space is divided into as many as 16 banks that contain 256 bytes each. The PIC18FX3J11/X4J11 devices, with up to 16 Kbytes of program memory, implement 4 complete banks for a total of 1024 bytes. PIC18FX5J11 devices, with 32 Kbytes of program memory, implement 8 complete banks for a total of 2048 bytes. Figure 6-7 and Figure 6-8 show the data memory organization for the devices.

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

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

To ensure that commonly used registers (select SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to select SFRs and the lower portion of GPR Bank 0 without using the BSR. **Section 6.3.2 "Access Bank"** provides a detailed description of the Access RAM.

6.3.1 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 (BSR<3:0>). The upper four bits are unused; they will always read '0' and cannot be written to. The BSR can be loaded directly by using the MOVLB instruction.

The value of the BSR indicates the bank in data memory. The 8 bits in the instruction show the location in the bank and can be thought of as an offset from the bank's lower boundary. The relationship between the BSR's value and the bank division in data memory is shown in Figure 6-9.

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 6-7 indicates which banks are implemented.

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

FIGURE 6-7: DATA MEMORY MAP FOR PIC18FX3J11/X4J11 DEVICES

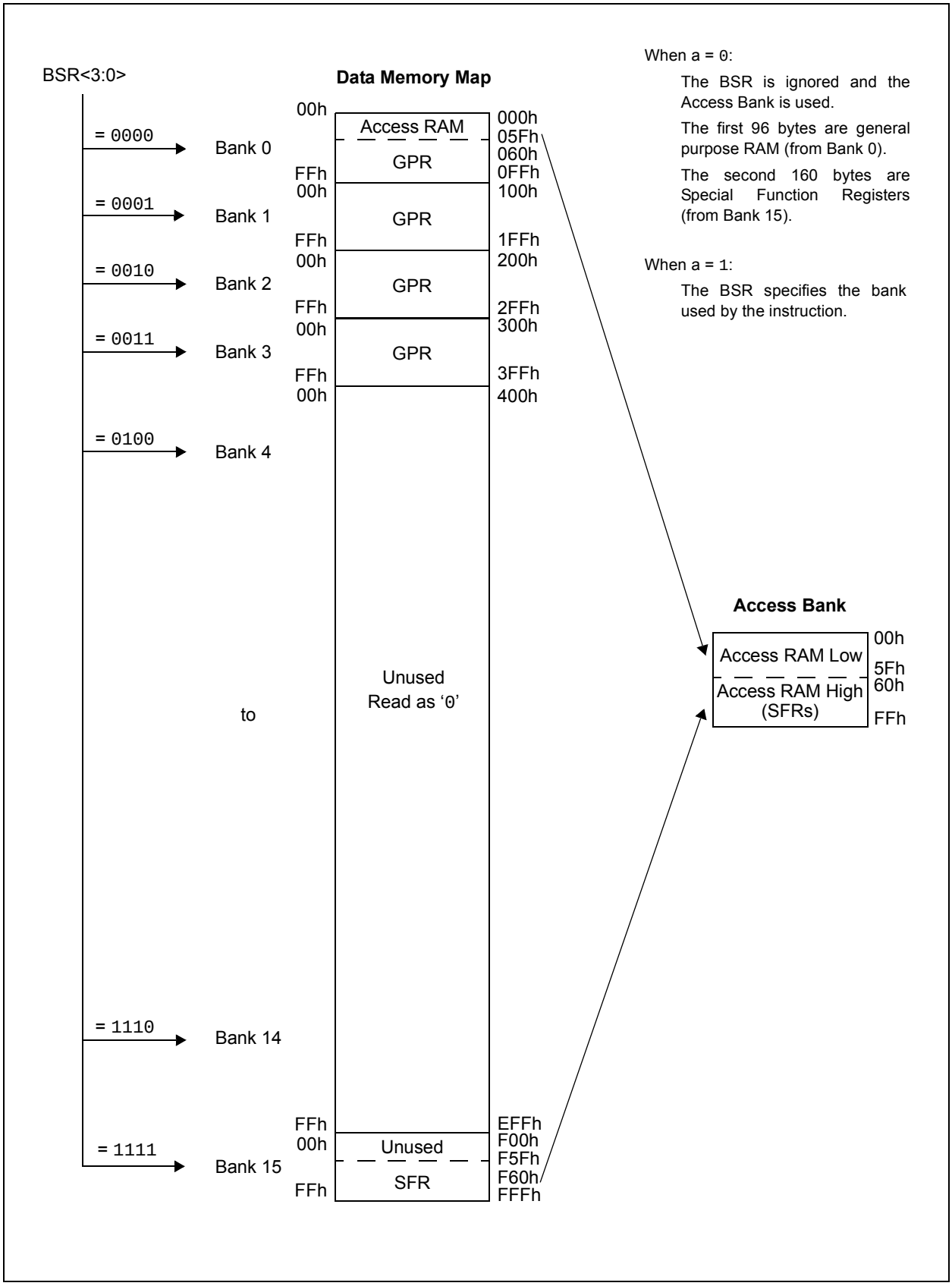


FIGURE 6-8: DATA MEMORY MAP FOR PIC18FX5J11 DEVICES

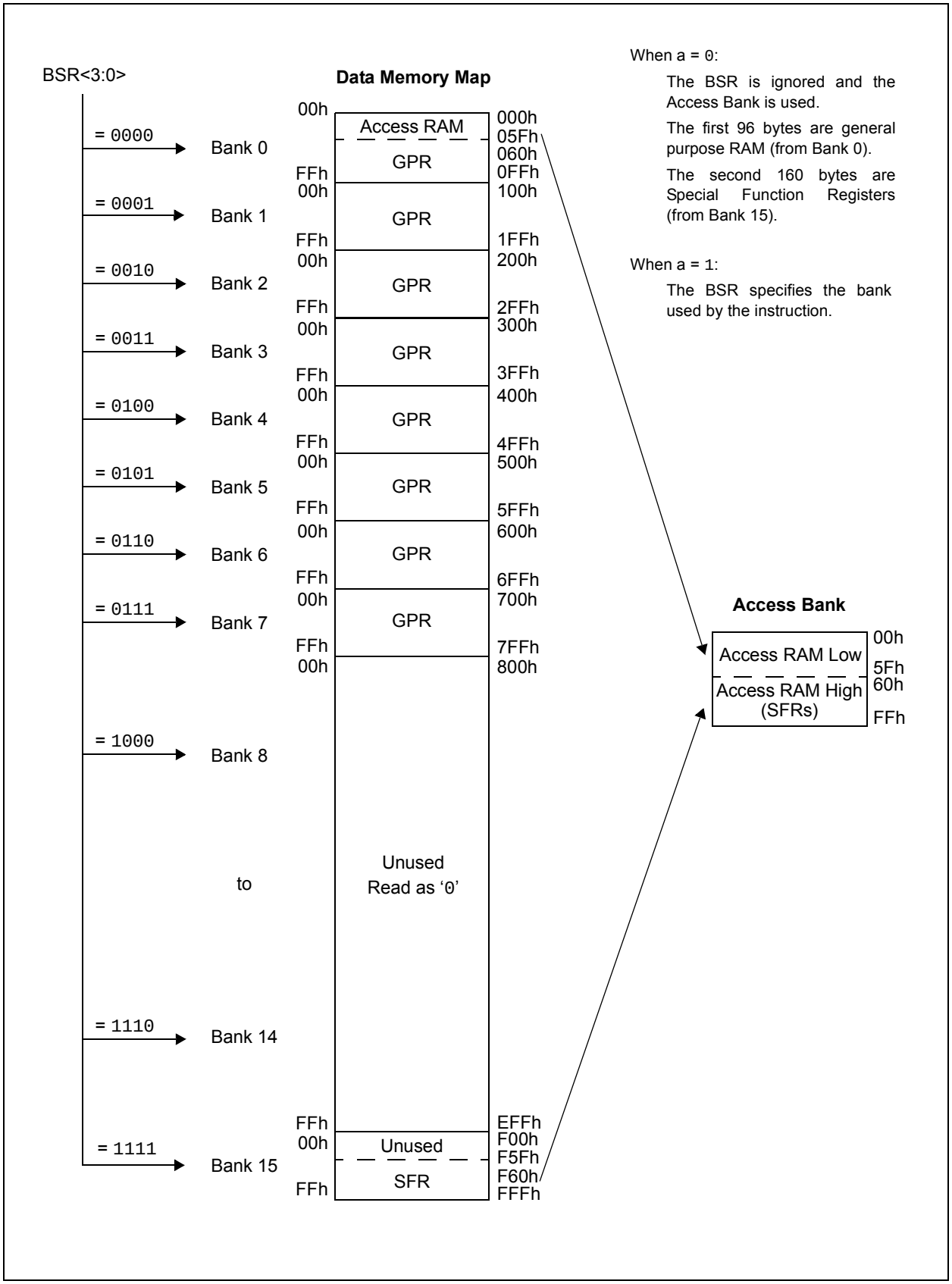
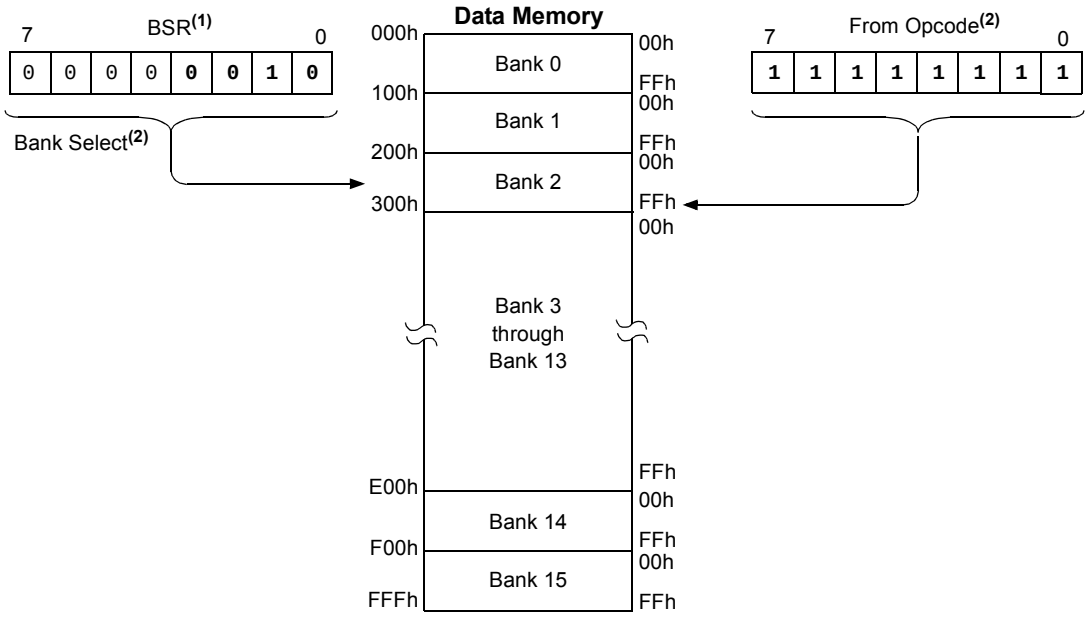


FIGURE 6-9: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

2: The MOVFF instruction embeds the entire 12-bit address in the instruction.

6.3.2 ACCESS BANK

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

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Bank 15. The lower half is known as the “Access RAM” and is composed of GPRs. The upper half is where the device’s SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an 8-bit address (Figure 6-7).

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

Using this “forced” addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 60h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 60h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables. Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST Configuration bit = 1). This is discussed in more detail in **Section 6.6.3 “Mapping the Access Bank in Indexed Literal Offset Mode”**.

6.3.3 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.

6.3.4 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 more than the top half of Bank 15 (F60h to FFFh). A list of these registers is given in Table 6-3 and Table 6-4.

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 6-3: SPECIAL FUNCTION REGISTER MAP FOR PIC18F85J11 FAMILY DEVICES

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	—(2)	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	—(2)	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	—(2)	F9Dh	PIE1	F7Dh	—(2)
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	—(2)	F9Ch	MEMCON(3)	F7Ch	—(2)
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	—(2)	F9Bh	OSCTUNE	F7Bh	—(2)
FFAh	PCLATH	FDAh	FSR2H	FBAh	—(2)	F9Ah	TRISJ(3)	F7Ah	—(2)
FF9h	PCL	FD9h	FSR2L	FB9h	—(2)	F99h	TRISH(3)	F79h	—(2)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—(2)	F98h	TRISG	F78h	—(2)
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—(2)	F97h	TRISF	F77h	—(2)
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—(2)	F96h	TRISE	F76h	—(2)
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	—(2)
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC	F74h	—(2)
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	—(2)
FF2h	INTCON	FD2h	—(2)	FB2h	TMR3L	F92h	TRISA	F72h	—(2)
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(3)	F71h	—(2)
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(3)	F70h	—(2)
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	—(2)
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	—(2)
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	—(2)
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	—(2)
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	—(2)
FEAh	FSR0H	FCAh	T2CON	FAAh	—(2)	F8Ah	LATB	F6Ah	CCPR1H
FE9h	FSR0L	FC9h	SSPBUF	FA9h	—(2)	F89h	LATA	F69h	CCPR1L
FE8h	WREG	FC8h	SSPADD	FA8h	—(2)	F88h	PORTJ(3)	F68h	CCP1CON
FE7h	INDF1(1)	FC7h	SSPSTAT	FA7h	EECON2	F87h	PORTH(3)	F67h	CCPR2H
FE6h	POSTINC1(1)	FC6h	SSPCON1	FA6h	EECON1	F86h	PORTG	F66h	CCPR2L
FE5h	POSTDEC1(1)	FC5h	SSPCON2	FA5h	IPR3	F85h	PORTF	F65h	CCP2CON
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	SPBRG2
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	RCREG2
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	TXREG2
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	TXSTA2
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	RCSTA2

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.

TABLE 6-4: PIC18F85J11 FAMILY 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	57, 67
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	57, 67
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	57, 67
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					uu-0 0000	57, 68
PCLATU	—	—	bit 21(1)	Holding Register for PC<20:16>					---0 0000	57, 67
PCLATH	Holding Register for PC<15:8>								0000 0000	57, 67
PCL	PC Low Byte (PC<7:0>)								0000 0000	57, 67
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	57, 92
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	57, 92
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	57, 92
TABLAT	Program Memory Table Latch								0000 0000	57, 92
PRODH	Product Register High Byte								xxxx xxxx	57, 111
PRODL	Product Register Low Byte								xxxx xxxx	57, 111
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	57, 115
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	57, 116
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	57, 117
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	57, 83
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	57, 84
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	57, 84
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	57, 84
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	57, 84
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- xxxx	57, 83
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	57, 83
WREG	Working Register								xxxx xxxx	57
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	57, 83
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	57, 84
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	57, 84
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	57, 84
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	57, 84
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- xxxx	58, 83
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	58, 83
BSR	—	—	—	—	Bank Select Register				---- 0000	58, 72
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	58, 83
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	58, 84
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	58, 84
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	58, 84
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	58, 84
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- xxxx	58, 83
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	58, 83
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	58, 81

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved, do not modify

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

2: These registers and/or bits are available only on 80-pin devices; otherwise, they are unimplemented and read as '0'. Reset states shown are for 80-pin devices.

3: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode. See **Section 17.4.3.2 "Address Masking"** for details.

4: The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 3.4.3 "PLL Frequency Multiplier"** for details.

5: RA6/RA7 and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as '0'.

TABLE 6-4: PIC18F85J11 FAMILY 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	58, 155
TMR0L	Timer0 Register Low Byte								xxxx xxxx	58, 155
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	58, 153
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0100 q000	36, 58
WDTCON	REGSLP	—	—	—	—	—	—	SWDTEN	0--- ---0	58, 287
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0-11 11q0	52, 58
TMR1H	Timer1 Register High Byte								xxxx xxxx	58, 161
TMR1L	Timer1 Register Low Byte								xxxx xxxx	58, 161
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	58, 157
TMR2	Timer2 Register								0000 0000	58, 164
PR2	Timer2 Period Register								1111 1111	58, 164
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	58, 163
SSPBUF	MSSP Receive Buffer/Transmit Register								xxxx xxxx	58, 187, 222
SSPADD	MSSP Address Register (I^2C™ Slave mode). MSSP Baud Rate Reload Register (I^2C Master mode).								0000 0000	58, 222
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	58, 180, 189
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	58, 181, 190
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	58, 191, 192
	GCEN	ACKSTAT	ADMSK5(3)	ADMSK4(3)	ADMSK3(3)	ADMSK2(3)	ADMSK1(3)	SEN		
ADRESH	A/D Result Register High Byte								xxxx xxxx	59, 267
ADRESL	A/D Result Register Low Byte								xxxx xxxx	59, 267
ADCON0	ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	0-00 0000	59, 259
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	59, 260
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	59, 261
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	59, 275
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	59, 269
TMR3H	Timer3 Register High Byte								xxxx xxxx	59, 167
TMR3L	Timer3 Register Low Byte								xxxx xxxx	59, 167
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	59, 165
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	59, 165
SPBRG1	EUSART Baud Rate Generator Register								0000 0000	59, 228
RCREG1	EUSART Receive Register								0000 0000	59, 236
TXREG1	EUSART Transmit Register								0000 0000	59, 234
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	59, 224
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	59, 225
EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	59, 90
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	---0 x00-	59, 91

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved, do not modify

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

2: These registers and/or bits are available only on 80-pin devices; otherwise, they are unimplemented and read as '0'. Reset states shown are for 80-pin devices.

3: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode. See **Section 17.4.3.2 "Address Masking"** for details.

4: The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 3.4.3 "PLL Frequency Multiplier"** for details.

5: RA6/RA7 and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as '0'.

TABLE 6-4: PIC18F85J11 FAMILY 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
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	--00 -11-	59, 126
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	--00 -00-	59, 120
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	--00 -00-	59, 123
IPR2	OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—	11-- 111-	59, 125
PIR2	OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—	00-- 000-	59, 119
PIE2	OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—	00-- 000-	59, 122
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	1111 1-11	59, 124
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	0000 0-00	59, 118
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	0000 0-00	59, 121
MEMCON(2)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	59, 100
OSCTUNE	INTSRC	PLLEN(4)	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000 0000	37, 59
TRISJ(2)	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	1111 1111	60, 149
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	1111 1111	60, 147
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	0001 1111	60, 146
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	1111 111-	60, 144
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	—	TRISE1	TRISE0	1111 1-11	60, 142
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111	60, 139
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	60, 136
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	60, 133
TRISA	TRISA7(5)	TRISA6(5)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	60, 131
LATJ(2)	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	xxxx xxxx	60, 149
LATH(2)	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	xxxx xxxx	60, 147
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	00-x xxxx	60, 146
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	xxxx xxx-	60, 144
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx xxxx	60, 142
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx	60, 139
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	60, 136
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	60, 133
LATA	LATA7(5)	LATA6(5)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx xxxx	60, 131
PORTJ(2)	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	xxxx xxxx	60, 149
PORTH(2)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	xxxx xxxx	60, 147
PORTG	RDPU	REPU	RJPU(2)	RG4	RG3	RG2	RG1	RG0	000x xxxx	60, 146
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	xxxx xxx-	60, 144
PORTE	RE7	RE6	RE5	RE4	RE3	—	RE1	RE0	xxxx x-xx	60, 142
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	60, 139
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	60, 136
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	60, 133
PORTA	RA7(5)	RA6(5)	RA5	RA4	RA3	RA2	RA1	RA0	xx0x 0000	60, 131

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved, do not modify

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

2: These registers and/or bits are available only on 80-pin devices; otherwise, they are unimplemented and read as '0'. Reset states shown are for 80-pin devices.

3: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode. See **Section 17.4.3.2 "Address Masking"** for details.

4: The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 3.4.3 "PLL Frequency Multiplier"** for details.

5: RA6/RA7 and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as '0'.

TABLE 6-4: PIC18F85J11 FAMILY 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	EUSART Baud Rate Generator High Byte								0000 0000	60, 228
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	01-0 0-00	60, 226
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	60, 170
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	60, 170
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	60, 169
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	61, 170
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	61, 170
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	61, 169
SPBRG2	AUSART Baud Rate Generator Register								0000 0000	61, 248
RCREG2	AUSART Receive Register								0000 0000	61, 253
TXREG2	AUSART Transmit Register								0000 0000	61, 251
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	61, 246
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	61, 247

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition, r = reserved, do not modify

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

2: These registers and/or bits are available only on 80-pin devices; otherwise, they are unimplemented and read as '0'. Reset states shown are for 80-pin devices.

3: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode. See **Section 17.4.3.2 "Address Masking"** for details.

4: The PLLEN bit is only available in specific oscillator configurations; otherwise, it is disabled and reads as '0'. See **Section 3.4.3 "PLL Frequency Multiplier"** for details.

5: RA6/RA7 and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as '0'.

6.3.5 STATUS REGISTER

The STATUS register, shown in Register 6-3, contains the arithmetic status of the ALU. The STATUS register can be the operand for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, then the write to these five bits is disabled.

These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the STATUS register as destination may be different than intended. For example, `CLRF STATUS` will set the Z bit but leave the other bits unchanged. The STATUS register then reads back as '`000u u1uu`'. It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF`, `MOVFF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions not affecting any Status bits, see the instruction set summaries in Table 25-2 and Table 25-3.

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

REGISTER 6-3: 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(1)	C(2)
bit 7							bit 0

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

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

bit 4 **N:** Negative bit
This bit is used for signed arithmetic (2's complement). It indicates whether the result was negative (ALU MSB = 1).
1 = Result was negative
0 = Result was positive

bit 3 **OV:** Overflow bit
This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude which causes the sign bit (bit 7) to change state.
1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{\text{Borrow}}$ bit(1)
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/$\overline{\text{Borrow}}$ bit(2)
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 $\overline{\text{borrow}}$, 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 $\overline{\text{borrow}}$, 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.

6.4 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 6.6 "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 6.6.1 "Indexed Addressing with Literal Offset"**.

6.4.1 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.

6.4.2 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 6.3.3 "General Purpose Register File"**), or a location in the Access Bank (**Section 6.3.2 "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 6.3.1 "Bank Select Register"**) are used with the address to determine the complete 12-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

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

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

6.4.3 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 6-5. It also enables users to perform Indexed Addressing and other Stack Pointer operations for program memory in data memory.

EXAMPLE 6-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

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

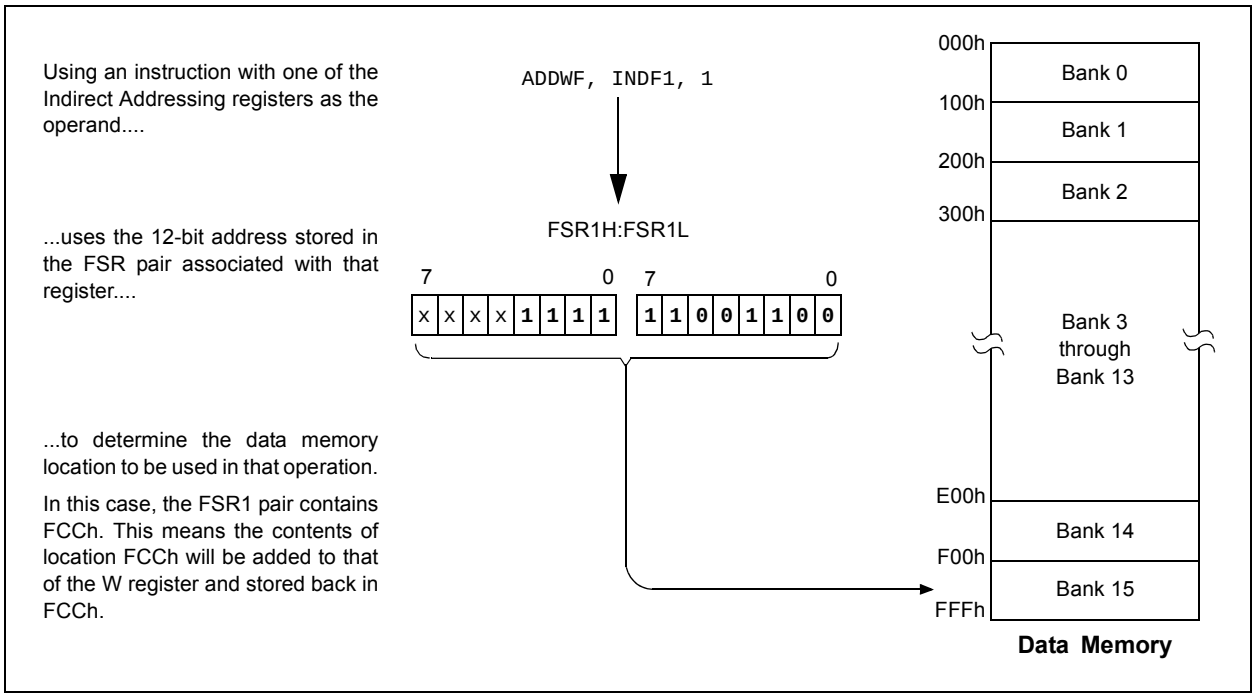
6.4.3.1 FSR Registers and the INDF Operand

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 6-10: INDIRECT ADDRESSING



6.4.3.2 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.

6.4.3.3 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 contain 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.

6.5 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, MOVSF, MOVSS and SUBFSR. These instructions are executed as described in **Section 6.2.4 “Two-Word Instructions”**.

6.6 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.

6.6.1 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.

6.6.2 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 do not use the Access Bank (Access RAM bit is ‘1’) or include a file address of 60h or above. Instructions meeting these criteria will continue to execute as before. A comparison of the different possible addressing modes when the extended instruction set is enabled is shown in Figure 6-11.

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 25.2.1 “Extended Instruction Syntax”**.

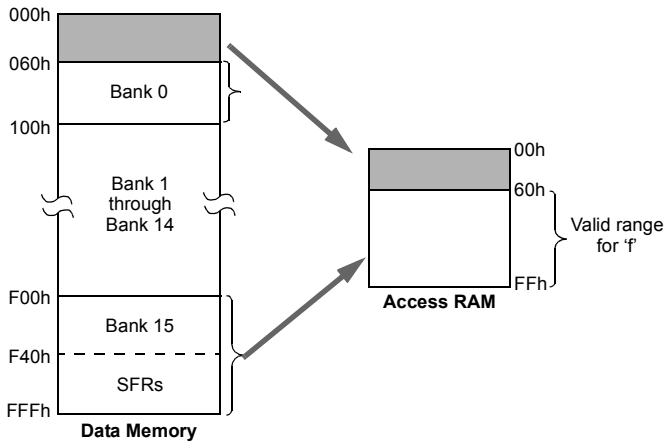
FIGURE 6-11: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)

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.

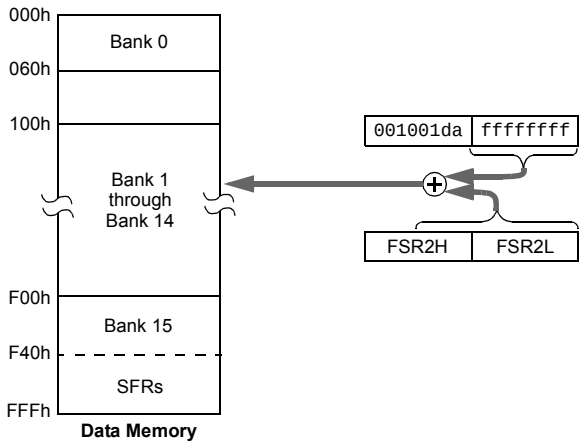
Locations below 060h are not available in this addressing mode.



When a = `0` and f ≤ 5Fh:

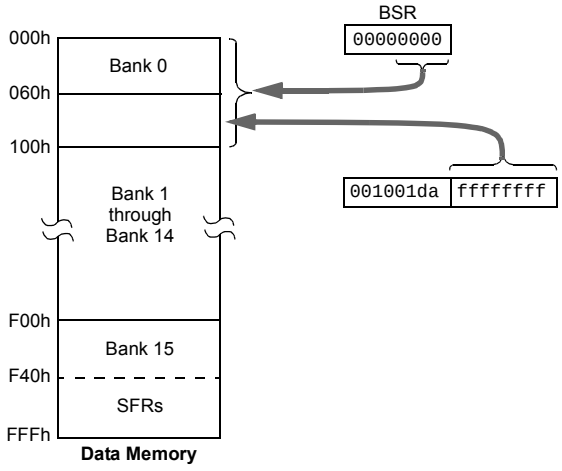
The instruction executes in Indexed Literal Offset mode. 'f' is interpreted as an offset to the address value in FSR2. The two are added together to obtain the address of the target register for the instruction. The address can be anywhere in the data memory space.

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.



6.6.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

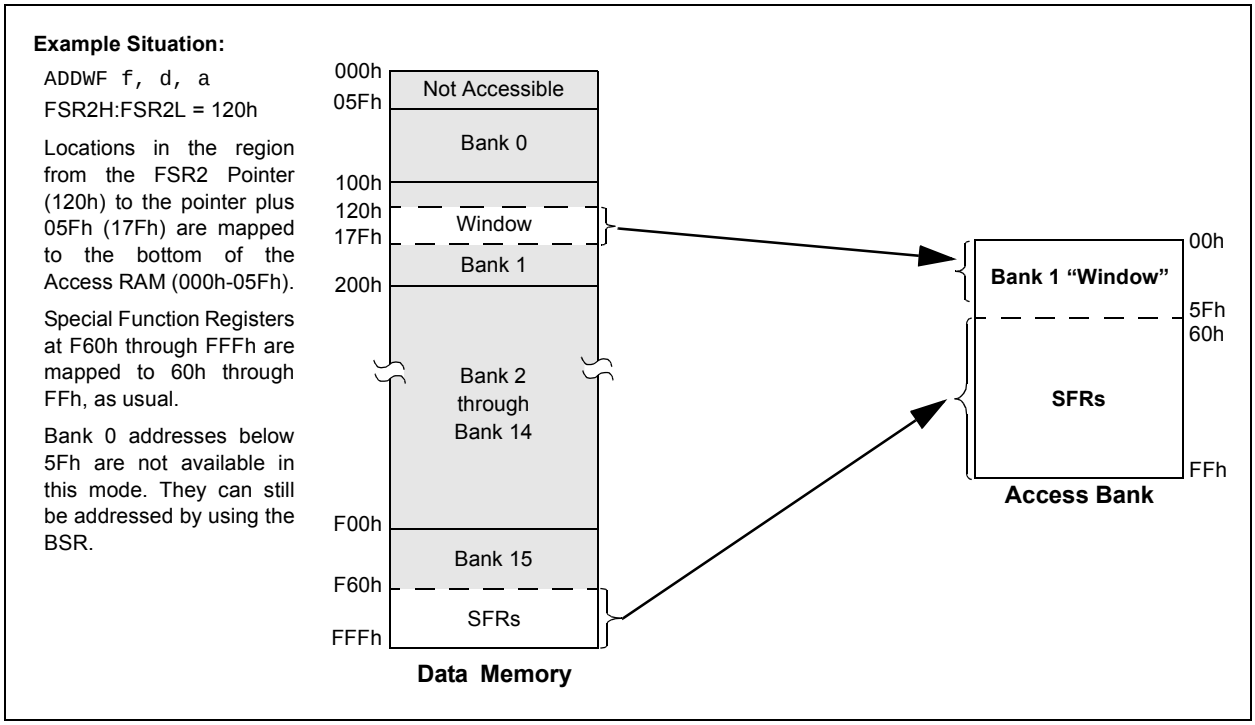
The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined "window" that can be located anywhere in the 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 6.3.2 "Access Bank"**). An example of Access Bank remapping in this addressing mode is shown in Figure 6-12.

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is '1') will continue to use Direct Addressing as before. Any Indirect or Indexed Addressing operation that explicitly uses any of the indirect file operands (including FSR2) will continue to operate as standard Indirect Addressing. Any instruction that uses the Access Bank, but includes a register address of greater than 05Fh, will use Direct Addressing and the normal Access Bank map.

6.6.4 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 6-12: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING



NOTES:

7.0 FLASH PROGRAM MEMORY

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

A read from program memory is executed on one byte at a time. A write to program memory is executed on blocks of 64 bytes at a time. Program memory is erased in blocks of 1024 bytes at a time. A bulk erase operation may not be issued from user code.

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

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

7.1 Table Reads and Table Writes

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

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

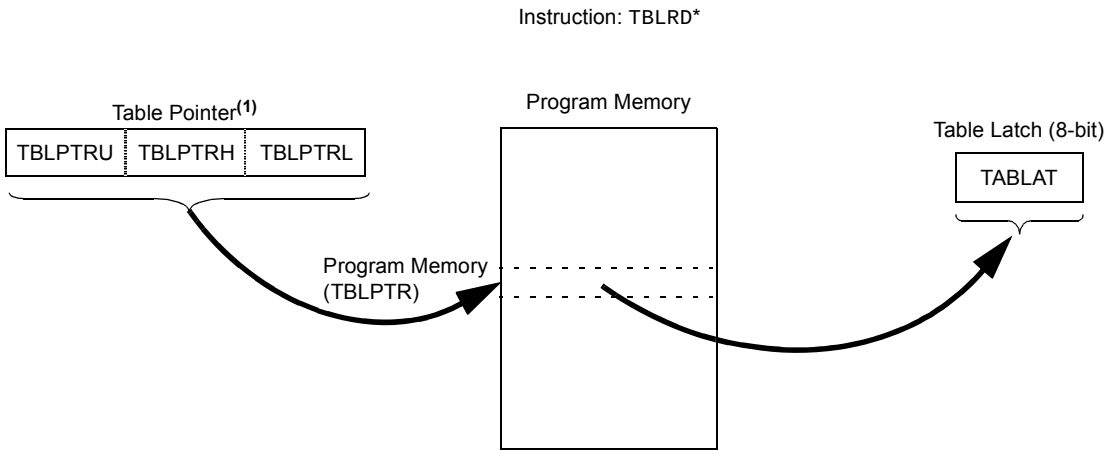
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into the data RAM space. Figure 7-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 7.5 "Writing to Flash Program Memory"**. Figure 7-2 shows the operation of a table write with program memory and data RAM.

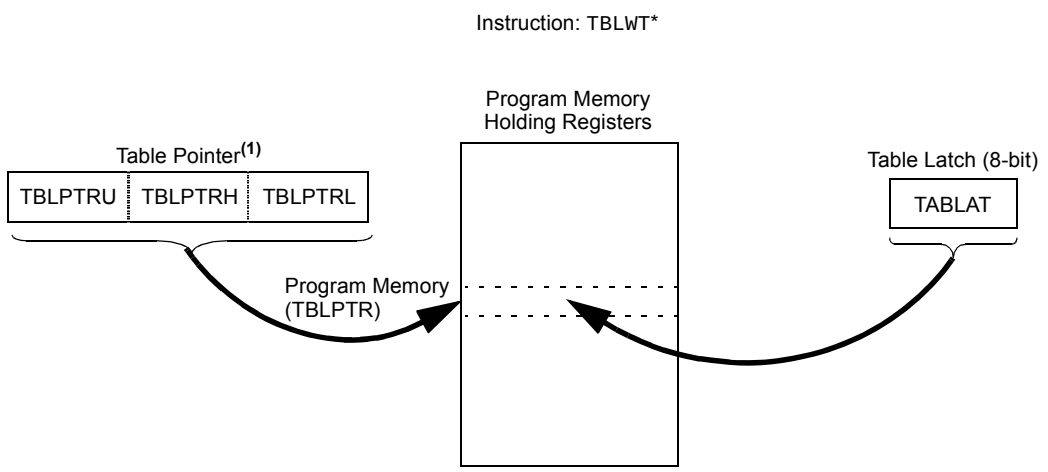
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions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

FIGURE 7-1: TABLE READ OPERATION



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

FIGURE 7-2: TABLE WRITE OPERATION



Note 1: The Table Pointer actually points to one of 64 holding registers, the address of which is determined by TBLPTRL<5:0>. The process for physically writing data to the program memory array is discussed in **Section 7.5 "Writing to Flash Program Memory"**.

7.2 Control Registers

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

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

7.2.1 EECON1 AND EECON2 REGISTERS

The EECON1 register (Register 7-1) is the control register for memory accesses. The EECON2 register is not a physical register; it is used exclusively in the memory write and erase sequences. Reading EECON2 will read all '0's.

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

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

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

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

REGISTER 7-1: EECON1: EEPROM CONTROL REGISTER 1

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

Legend:	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable bit	S = Settable bit (cannot be cleared in software)	
-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 **FREE:** Flash Erase Enable bit
- 1 = Erase the program memory block addressed by TBLPTR on the next WR command (cleared by completion of an erase operation)
- 0 = Perform write only

bit 3 **WRERR:** Flash Program Error Flag bit
- 1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation, or an improper write attempt)
- 0 = The write operation completed

bit 2 **WREN:** Flash Program Write Enable bit
- 1 = Allows write cycles to Flash program memory
- 0 = Inhibits write cycles to Flash program memory

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

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

7.2.2 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.

7.2.3 TABLE POINTER REGISTER (TBLPTR)

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

The Table Pointer register, TBLPTR, is used by the TBLRD and TBLWT instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations are shown in Table 7-1. These operations on the TBLPTR only affect the low-order 21 bits.

7.2.4 TABLE POINTER BOUNDARIES

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

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

When a TBLWT is executed, the seven LSbs of the Table Pointer register (TBLPTR<6:0>) determine which of the 64 program memory holding registers is written to. When the timed write to program memory begins (via the WR bit), the 12 MSbs of the TBLPTR (TBLPTR<21:10>) determine which program memory block of 1024 bytes is written to. For more detail, see **Section 7.5 “Writing to Flash Program Memory”**.

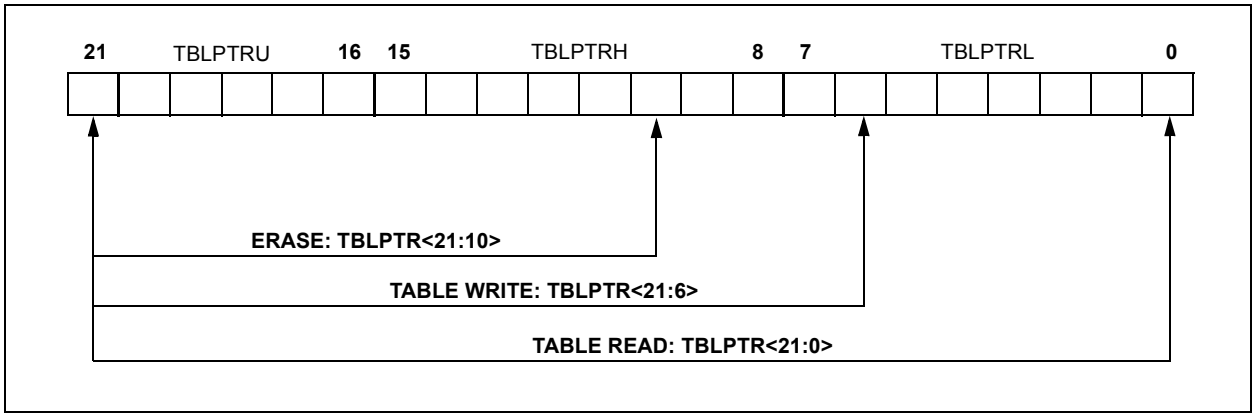
When an erase of program memory is executed, the 12 MSbs of the Table Pointer register point to the 1024-byte block that will be erased. The Least Significant bits are ignored.

Figure 7-3 describes the relevant boundaries of the TBLPTR based on Flash program memory operations.

TABLE 7-1: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

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

FIGURE 7-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



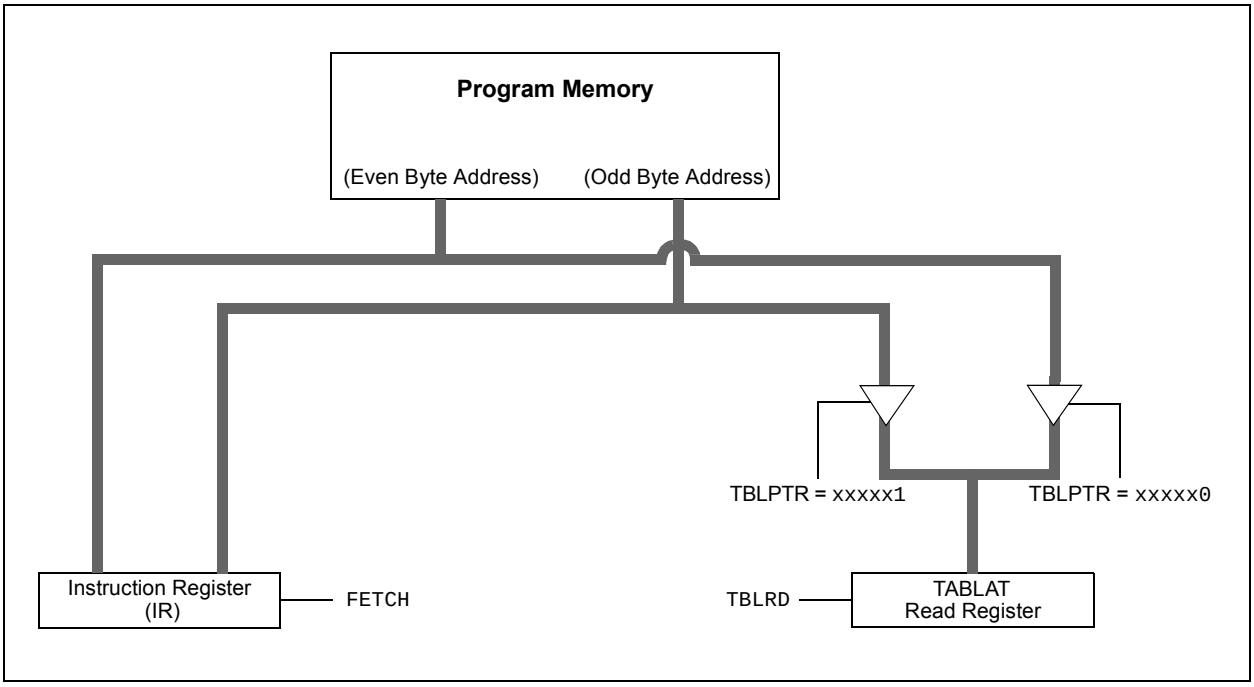
7.3 Reading the Flash Program Memory

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 7-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 7-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 7-1: 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
            MOVF      WORD_ODD
```

7.4 Erasing Flash Program Memory

The minimum erase block is 512 words or 1024 bytes. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the micro-controller itself, a block of 1024 bytes of program memory is erased. The Most Significant 12 bits of the TBLPTR<21:10> point to the block being erased; TBLPTR<9:0> are ignored.

The EECON1 register commands the erase operation. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation. For protection, the write initiate sequence for EECON2 must be used.

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

7.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

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

1. Load the Table Pointer register with the address of the block being erased.
2. Set the WREN and FREE bits (EECON1<2,4>) to enable the erase operation.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit; this will begin the erase cycle.
7. The CPU will stall for the duration of the erase for TIE (see parameter D133B).
8. Re-enable interrupts.

EXAMPLE 7-2: ERASING A FLASH PROGRAM MEMORY BLOCK

```
                    MOVLW   CODE_ADDR_UPPER       ; load TBLPTR with the base
                    MOVWF   TBLPTRU               ; address of the memory block
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
        ERASE_BLOCK
                    BSF     EECON1, WREN          ; enable write to memory
                    BSF     EECON1, FREE          ; enable Erase operation
                    BCF     INTCON, GIE           ; disable interrupts
    Required        MOVLW   55h
    Sequence        MOVWF   EECON2                ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2                ; write 0AAh
                    BSF     EECON1, WR            ; start erase (CPU stall)
                    BSF     INTCON, GIE           ; re-enable interrupts
```

7.5 Writing to Flash Program Memory

The minimum programming block is 32 words or 64 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 64 holding registers used by the table writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction may need to be executed 64 times for each programming operation. All of the table write operations will essentially be short writes because only the holding registers are written. At the end of updating the 64 holding registers, the EECON1 register must be written to in order to start the programming operation with a long write.

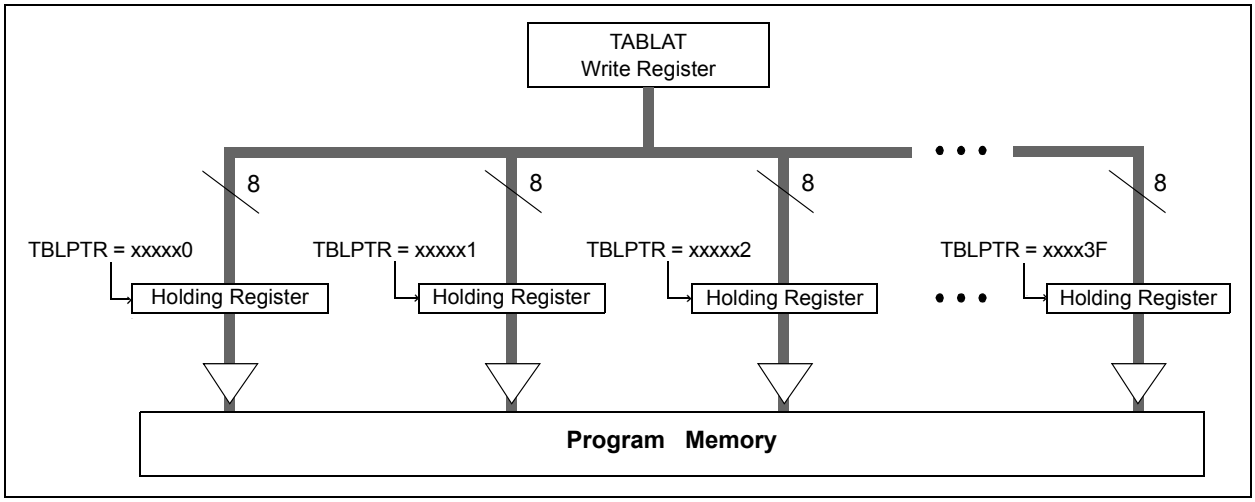
The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

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

Note 1: Unlike previous PIC® MCUs, members of the PIC18F85J11 family do not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence.

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

FIGURE 7-5: TABLE WRITES TO FLASH PROGRAM MEMORY



7.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

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

1. Read 1024 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer register with address being erased.
4. Execute the block erase procedure.
5. Load the Table Pointer register with the address of the first byte being written, minus 1.
6. Write the 64 bytes into the holding registers with auto-increment.
7. Set the WREN bit (EECON1<2>) to enable byte writes.
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write 0AAh to EECON2.
11. Set the WR bit; this will begin the write cycle.
12. The CPU will stall for the duration of the write for TIW (see parameter D133A).
13. Re-enable interrupts.
14. Repeat steps 6 through 13 until all 1024 bytes are written to program memory.
15. Verify the memory (table read).

An example of the required code is shown in Example 7-3.

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

EXAMPLE 7-3: WRITING TO FLASH PROGRAM MEMORY

```
                    MOVLW   CODE_ADDR_UPPER       ; Load TBLPTR with the base address
                    MOVWF   TBLPTRU               ; of the memory block, minus 1
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
ERASE_BLOCK
                    BSF     EECON1, WREN          ; enable write to memory
                    BSF     EECON1, FREE          ; enable Erase operation
                    BCF     INTCON, GIE           ; disable interrupts
                    MOVLW   55h
                    MOVWF   EECON2                ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2                ; write 0AAh
                    BSF     EECON1, WR            ; start erase (CPU stall)
                    BSF     INTCON, GIE           ; re-enable interrupts
                    MOVLW   D'16'
                    MOVWF   WRITE_COUNTER         ; Need to write 16 blocks of 64 to write
                                                  ; one erase block of 1024
RESTART_BUFFER
                    MOVLW   D'64'
                    MOVWF   COUNTER
                    MOVLW   BUFFER_ADDR_HIGH      ; point to buffer
                    MOVWF   FSR0H
                    MOVLW   BUFFER_ADDR_LOW
                    MOVWF   FSR0L
FILL_BUFFER
                    ...                           ; read the new data from I2C, SPI,
                                                  ; PSP, USART, etc.
WRITE_BUFFER
                    MOVLW   D’64                  ; number of bytes in holding register
                    MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                    MOVFF   POSTINC0, WREG        ; get low byte of buffer data
                    MOVWF   TABLAT                ; present data to table latch
                    TBLWT+*                       ; write data, perform a short write
                                                  ; to internal TBLWT holding register.
                    DECFSZ  COUNTER               ; loop until buffers are full
                    BRA     WRITE_BYTE_TO_HREGS
PROGRAM_MEMORY
                    BSF     EECON1, WREN          ; enable write to memory
                    BCF     INTCON, GIE           ; disable interrupts
                    MOVLW   55h
   Required         MOVWF   EECON2                ; write 55h
   Sequence         MOVLW   0AAh
                    MOVWF   EECON2                ; write 0AAh
                    BSF     EECON1, WR            ; start program (CPU stall)
                    BSF     INTCON, GIE           ; re-enable interrupts
                    BCF     EECON1, WREN          ; disable write to memory

                    DECFSZ  WRITE_COUNTER         ; done with one write cycle
                    BRA     RESTART_BUFFER        ; if not done replacing the erase block
```

7.5.2 WRITE VERIFY

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

7.5.3 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. If the write operation is interrupted by a $\overline{\text{MCLR}}$ Reset or a WDT Time-out Reset during normal operation, the user can check the WRERR bit and rewrite the location(s) as needed.

7.6 Flash Program Operation During Code Protection

See **Section 23.6 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

TABLE 7-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					57
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								57
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								57
TABLAT	Program Memory Table Latch								57
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
EECON2	EEPROM Control Register 2 (not a physical register)								59
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	59

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

NOTES:

8.0 EXTERNAL MEMORY BUS

Note: The external memory bus is not implemented on 64-pin devices.

The external memory bus allows the device to access external memory devices (such as Flash, EPROM, SRAM, etc.) as program or data memory. It supports both 8 and 16-Bit Data Width modes, and three address widths of up to 20 bits.

The bus is implemented with 28 pins, multiplexed across four I/O ports. Three ports (PORTD, PORTE and PORTH) are multiplexed with the address/data bus for a total of 20 available lines, while PORTJ is multiplexed with the bus control signals.

A list of the pins and their functions is provided in Table 8-1.

TABLE 8-1: PIC18F85J11 FAMILY EXTERNAL BUS – I/O PORT FUNCTIONS

Name	Port	Bit	External Memory Bus Function
RD0/AD0	PORTD	0	Address bit 0 or Data bit 0
RD1/AD1	PORTD	1	Address bit 1 or Data bit 1
RD2/AD2	PORTD	2	Address bit 2 or Data bit 2
RD3/AD3	PORTD	3	Address bit 3 or Data bit 3
RD4/AD4	PORTD	4	Address bit 4 or Data bit 4
RD5/AD5	PORTD	5	Address bit 5 or Data bit 5
RD6/AD6	PORTD	6	Address bit 6 or Data bit 6
RD7/AD7	PORTD	7	Address bit 7 or Data bit 7
RE0/AD8	PORTE	0	Address bit 8 or Data bit 8
RE1/AD9	PORTE	1	Address bit 9 or Data bit 9
RE2/AD10	PORTE	2	Address bit 10 or Data bit 10
RE3/AD11	PORTE	3	Address bit 11 or Data bit 11
RE4/AD12	PORTE	4	Address bit 12 or Data bit 12
RE5/AD13	PORTE	5	Address bit 13 or Data bit 13
RE6/AD14	PORTE	6	Address bit 14 or Data bit 14
RE7/AD15	PORTE	7	Address bit 15 or Data bit 15
RH0/A16	PORTH	0	Address bit 16
RH1/A17	PORTH	1	Address bit 17
RH2/A18	PORTH	2	Address bit 18
RH3/A19	PORTH	3	Address bit 19
RJ0/ALE	PORTJ	0	Address Latch Enable (ALE) Control pin
RJ1/$\overline{OE}$	PORTJ	1	Output Enable ($\overline{OE}$) Control pin
RJ2/$\overline{WRL}$	PORTJ	2	Write Low ($\overline{WRL}$) Control pin
RJ3/$\overline{WRH}$	PORTJ	3	Write High ($\overline{WRH}$) Control pin
RJ4/BA0	PORTJ	4	Byte Address bit 0 (BA0)
RJ5/$\overline{CE}$	PORTJ	5	Chip Enable ($\overline{CE}$) Control pin
RJ6/$\overline{LB}$	PORTJ	6	Lower Byte Enable ($\overline{LB}$) Control pin
RJ7/$\overline{UB}$	PORTJ	7	Upper Byte Enable ($\overline{UB}$) Control pin

Note: For the sake of clarity, only I/O port and external bus assignments are shown here. One or more additional multiplexed features may be available on some pins.

8.1 External Memory Bus Control

The operation of the interface is controlled by the MEMCON register (Register 8-1). This register is available in all Program Memory modes except Microcontroller mode. In this mode, the register is disabled and cannot be written to.

The EBDIS bit (MEMCON<7>) controls the operation of the bus and related port functions. Clearing EBDIS enables the interface and disables the I/O functions of the ports, as well as any other functions multiplexed to those pins. Setting the bit enables the I/O ports and other functions, but allows the interface to override everything else on the pins when an external memory operation is required. By default, the external bus is always enabled and disables all other I/O.

The operation of the EBDIS bit is also influenced by the Program Memory mode being used. This is discussed in more detail in **Section 8.5 "Program Memory Modes and the External Memory Bus"**.

The WAIT bits allow for the addition of Wait states to external memory operations. The use of these bits is discussed in **Section 8.3 "Wait States"**.

The WM bits select the particular operating mode used when the bus is operating in 16-Bit Data Width mode. These are discussed in more detail in **Section 8.6 "16-Bit Data Width Modes"**. These bits have no effect when an 8-Bit Data Width mode is selected.

REGISTER 8-1: MEMCON: EXTERNAL MEMORY BUS CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit 15							bit 8

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 **EBDIS**: External Bus Disable bit
- 1 = External bus is enabled when microcontroller accesses external memory; otherwise, all external bus drivers are mapped as I/O ports
- 0 = External bus is always enabled, I/O ports are disabled

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

bit 5-4 **WAIT<1:0>:** Table Reads and Writes Bus Cycle Wait Count bits
- 11 = Table reads and writes will wait 0 TCY
- 10 = Table reads and writes will wait 1 TCY
- 01 = Table reads and writes will wait 2 TCY
- 00 = Table reads and writes will wait 3 TCY

bit 3-2 **Unimplemented**: Read as '0'

bit 1-0 **WM<1:0>:** TBLWT Operation with 16-Bit Data Bus Width Select bits
- 1x = Word Write mode: TABLAT0 and TABLAT1 word output; $\overline{WRH}$ active when TABLAT1 written
- 01 = Byte Select mode: TABLAT data copied on both MSB and LSB; $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate
- 00 = Byte Write mode: TABLAT data copied on both MSB and LSB; $\overline{WRH}$ or $\overline{WRL}$ will activate

8.2 Address and Data Width

The PIC18F85J11 family of devices can be independently configured for different address and data widths on the same memory bus. Both address and data width are set by Configuration bits in the CONFIG3L register. As Configuration bits, this means that these options can only be configured by programming the device and are not controllable in software.

The BW bit selects an 8-bit or 16-bit data bus width. Setting this bit (default) selects a data width of 16 bits.

The EMB<1:0> bits determine both the Program Memory mode and the address bus width. The available options are 20-bit, 16-bit and 12-bit, as well as the default Microcontroller mode (external bus disabled). Selecting a 16-bit or 12-bit width makes a corresponding number of high-order lines available for I/O functions; these pins are no longer affected by the setting of the EBDIS bit. For example, selecting a 16-Bit Address mode (EMB<1:0> = 01) disables A<19:16> and allows PORTH<3:0> to function without interruptions from the bus. Using the smaller address widths allows users to tailor the memory bus to the size of the external memory space for a particular design while freeing up pins for dedicated I/O operation.

Because the EMB bits have the effect of disabling pins for memory bus operations, it is important to always select an address width at least equal to the data width. If a 12-bit address width is used with a 16-bit data width, the upper four bits of data will not be available on the bus.

All combinations of address and data widths require multiplexing of address and data information on the same lines. The address and data multiplexing, as well as I/O ports made available by the use of smaller address widths, are summarized in Table 8-2.

8.2.1 ADDRESS SHIFTING ON THE EXTERNAL BUS

By default, the address presented on the external bus is the value of the PC. In practical terms, this means that addresses in the external memory device below the top of on-chip memory are unavailable to the microcontroller. To access these physical locations, the glue logic between the microcontroller and the external memory must somehow translate addresses.

To simplify the interface, the external bus offers an extension of Extended Microcontroller mode that automatically performs address shifting. This feature is controlled by the EASHFT Configuration bit. Setting this bit offsets addresses on the bus by the size of the microcontroller's on-chip program memory and sets the bottom address at 0000h. This allows the device to use the entire range of physical addresses of the external memory.

8.2.2 21-BIT ADDRESSING

As an extension of 20-bit address width operation, the external memory bus can also fully address a 2-Mbyte memory space. This is done by using the Bus Address bit 0 (BA0) control line as the Least Significant bit of the address. The $\overline{UB}$ and $\overline{LB}$ control signals may also be used with certain memory devices to select the upper and lower bytes within a 16-bit wide data word.

This addressing mode is available in both 8-bit and certain 16-Bit Data Width modes. Additional details are provided in **Section 8.6.3 "16-Bit Byte Select Mode"** and **Section 8.7 "8-Bit Data Width Mode"**.

TABLE 8-2: ADDRESS AND DATA LINES FOR DIFFERENT ADDRESS AND DATA WIDTHS

Data Width	Address Width	Multiplexed Data and Address Lines (and Corresponding Ports)	Address-Only Lines (and Corresponding Ports)	Ports Available for I/O
8-bit	12-bit	AD<7:0> (PORTD<7:0>)	AD<11:8> (PORTE<3:0>)	PORTE<7:4>, All of PORTH
	16-bit		AD<15:8> (PORTE<7:0>)	All of PORTH
	20-bit		A<19:16>, AD<15:8> (PORTH<3:0>, PORTE<7:0>)	—
16-bit	16-bit	AD<15:0> (PORTD<7:0>, PORTE<7:0>)	—	All of PORTH
	20-bit		A<19:16> (PORTH<3:0>)	—

8.3 Wait States

While it may be assumed that external memory devices will operate at the microcontroller clock rate, this is often not the case. In fact, many devices require longer times to write or retrieve data than the time allowed by the execution of table read or table write operations.

To compensate for this, the external memory bus can be configured to add a fixed delay to each table operation using the bus. Wait states are enabled by setting the WAIT Configuration bit. When enabled, the amount of delay is set by the WAIT<1:0> bits (MEMCON<5:4>). The delay is based on multiples of microcontroller instruction cycle time and are added following the instruction cycle when the table operation is executed. The range is from no delay to 3 TCY (default value).

8.4 Port Pin Weak Pull-ups

With the exception of the upper address lines, A<19:16>, the pins associated with the external memory bus are equipped with weak pull-ups. The pull-ups are controlled by the upper three bits of the PORTG register. They are named RDPU, REPU and RJPU and control pull-ups on PORTD, PORTE and PORTJ, respectively. Setting one of these bits enables the corresponding pull-ups for that port. All pull-ups are disabled by default on all device Resets.

8.5 Program Memory Modes and the External Memory Bus

The PIC18F85J11 family of devices are capable of operating in one of two Program Memory modes, using combinations of on-chip and external program memory. The functions of the multiplexed port pins depend on the Program Memory mode selected, as well as the setting of the EBDIS bit.

In **Microcontroller Mode,** the bus is not active and the pins have their port functions only. Writes to the MEMCOM register are not permitted. The Reset value of EBDIS (‘0’) is ignored and EMB pins behave as I/O ports.

In **Extended Microcontroller Mode,** the external program memory bus shares I/O port functions on the pins. When the device is fetching or doing table read/table write operations on the external program memory space, the pins will have the external bus function.

If the device is fetching and accessing internal program memory locations only, the EBDIS control bit will change the pins from external memory to I/O port functions. When EBDIS = 0, the pins function as the external bus. When EBDIS = 1, the pins function as I/O ports.

If the device fetches or accesses external memory while EBDIS = 1, the pins will switch to the external bus. If the EBDIS bit is set by a program executing from external memory, the action of setting the bit will be delayed until the program branches into the internal memory. At that time, the pins will change from external bus to I/O ports.

If the device is executing out of internal memory when EBDIS = 0, the memory bus address/data and control pins will not be active. They will go to a state where the active address/data pins are tri-state; the $\overline{CE}$, $\overline{OE}$, $\overline{WRH}$, $\overline{WRL}$, $\overline{UB}$ and $\overline{LB}$ signals are ‘1’ and ALE and BA0 are ‘0’. Note that only those pins associated with the current address width are forced to tri-state; the other pins continue to function as I/O. In the case of 16-bit address width, for example, only AD<15:0> (PORTD and PORTE) are affected; A<19:16> (PORTH<3:0>) continue to function as I/O.

In all External Memory modes, the bus takes priority over any other peripherals that may share pins with it. This includes the Parallel Slave Port and serial communication modules which would otherwise take priority over the I/O port.

8.6 16-Bit Data Width Modes

In 16-Bit Data Width mode, the external memory interface can be connected to external memories in three different configurations:

- 16-Bit Byte Write
- 16-Bit Word Write
- 16-Bit Byte Select

The configuration to be used is determined by the WM<1:0> bits in the MEMCON register (MEMCON<1:0>). These three different configurations allow the designer maximum flexibility in using both 8-bit and 16-bit devices with 16-bit data.

For all 16-Bit Data Width modes, the Address Latch Enable (ALE) pin indicates that the address bits, AD<15:0>, are available on the external memory interface bus. Following the address latch, the Output Enable signal ($\overline{OE}$) will enable both bytes of program memory at once to form a 16-bit instruction word. The Chip Enable signal ($\overline{CE}$) is active at any time that the microcontroller accesses external memory, whether reading or writing. It is inactive (asserted high) whenever the device is in Sleep mode.

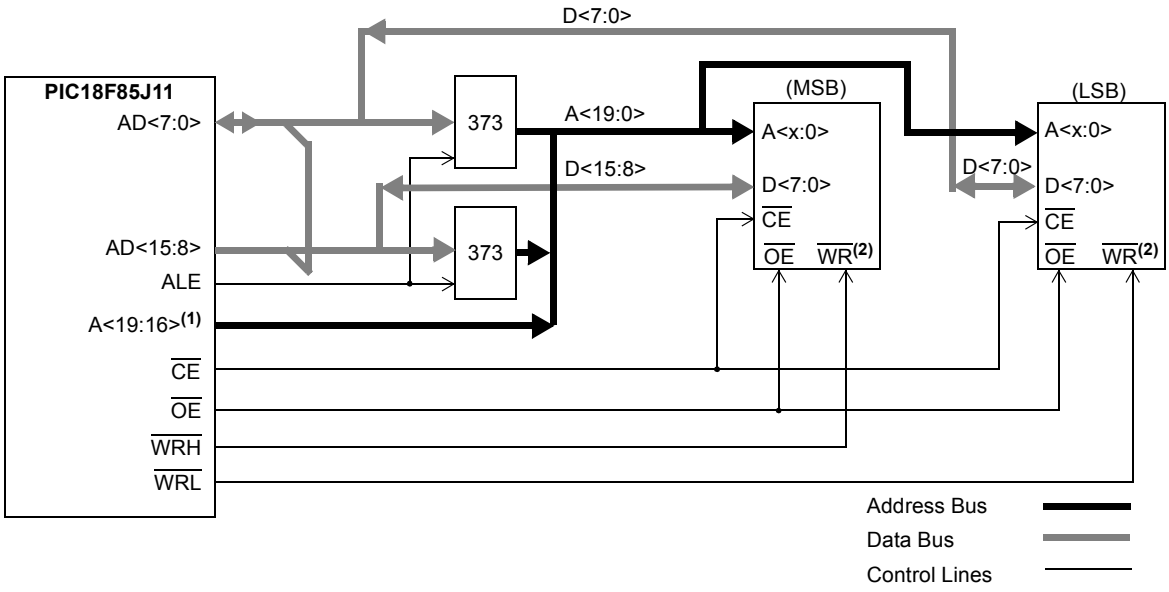
In Byte Select mode, JEDEC standard Flash memories will require BA0 for the byte address line and one I/O line to select between Byte and Word mode. The other 16-Bit Data Width modes do not need BA0. JEDEC standard static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte selection.

8.6.1 16-BIT BYTE WRITE MODE

Figure 8-1 shows an example of 16-Bit Byte Write mode for PIC18F85J11 family devices. This mode is used for two separate 8-bit memories connected for 16-bit operation. This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate $\overline{WRH}$ or $\overline{WRL}$ control line is strobed on the LSb of the TBLPTR.

FIGURE 8-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: The upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

8.6.2 16-BIT WORD WRITE MODE

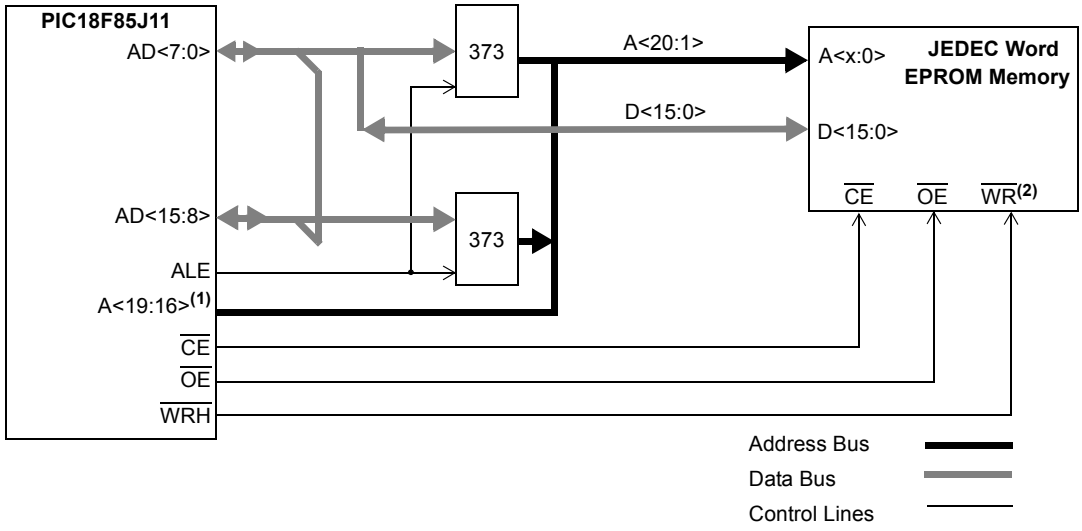
Figure 8-2 shows an example of 16-Bit Word Write mode for PIC18F85J11 family devices. This mode is used for word-wide memories which include some of the EPROM and Flash-type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory and table writes to any type of word-wide external memories. This method makes a distinction between TBLWT cycles to even or odd addresses.

During a TBLWT cycle to an even address (TBLPTR<0> = 0), the TABLAT data is transferred to a holding latch and the external address data bus is tri-stated for the data portion of the bus cycle. No write signals are activated.

During a TBLWT cycle to an odd address (TBLPTR<0> = 1), the TABLAT data is presented on the upper byte of the AD<15:0> bus. The contents of the holding latch are presented on the lower byte of the AD<15:0> bus.

The $\overline{WRH}$ signal is strobed for each write cycle; the $\overline{WRL}$ pin is unused. The signal on the BA0 pin indicates the LSb of the TBLPTR, but it is left unconnected. Instead, the $\overline{UB}$ and $\overline{LB}$ signals are active to select both bytes. The obvious limitation to this method is that the table write must be done in pairs on a specific word boundary to correctly write a word location.

FIGURE 8-2: 16-BIT WORD WRITE MODE EXAMPLE



Note 1: The upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

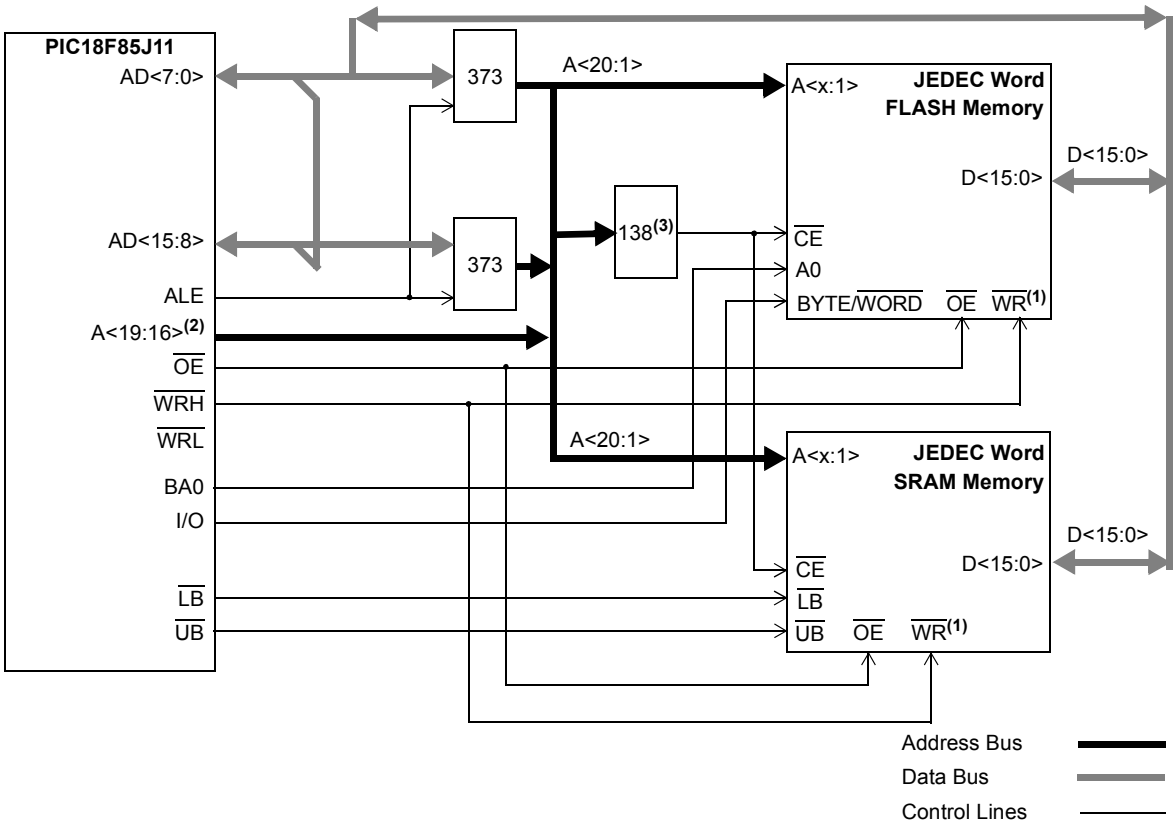
8.6.3 16-BIT BYTE SELECT MODE

Figure 8-3 shows an example of 16-Bit Byte Select mode. This mode allows table write operations to word-wide external memories with byte selection capability. This generally includes both word-wide Flash and SRAM devices.

During a TBLWT cycle, the TABLAT data is presented on the upper and lower byte of the AD<15:0> bus. The $\overline{WRH}$ signal is strobed for each write cycle; the $\overline{WRL}$ pin is not used. The BA0 or $\overline{UB}$/$\overline{LB}$ signals are used to select the byte to be written, based on the Least Significant bit of the TBLPTR register.

Flash and SRAM devices use different control signal combinations to implement Byte Select mode. JEDEC standard Flash memories require that a controller I/O port pin be connected to the memory's BYTE/$\overline{WORD}$ pin to provide the select signal. They also use the BA0 signal from the controller as a byte address. JEDEC standard static RAM memories, on the other hand, use the $\overline{UB}$ or $\overline{LB}$ signals to select the byte.

FIGURE 8-3: 16-BIT BYTE SELECT MODE EXAMPLE



Note 1: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

2: The upper order address lines are used only for 20-bit address width.

3: Demultiplexing is only required when multiple memory devices are accessed.

8.6.4 16-BIT MODE TIMING

The presentation of control signals on the external memory bus is different for the various operating modes. Typical signal timing diagrams are shown in Figure 8-4 and Figure 8-5.

FIGURE 8-4: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)

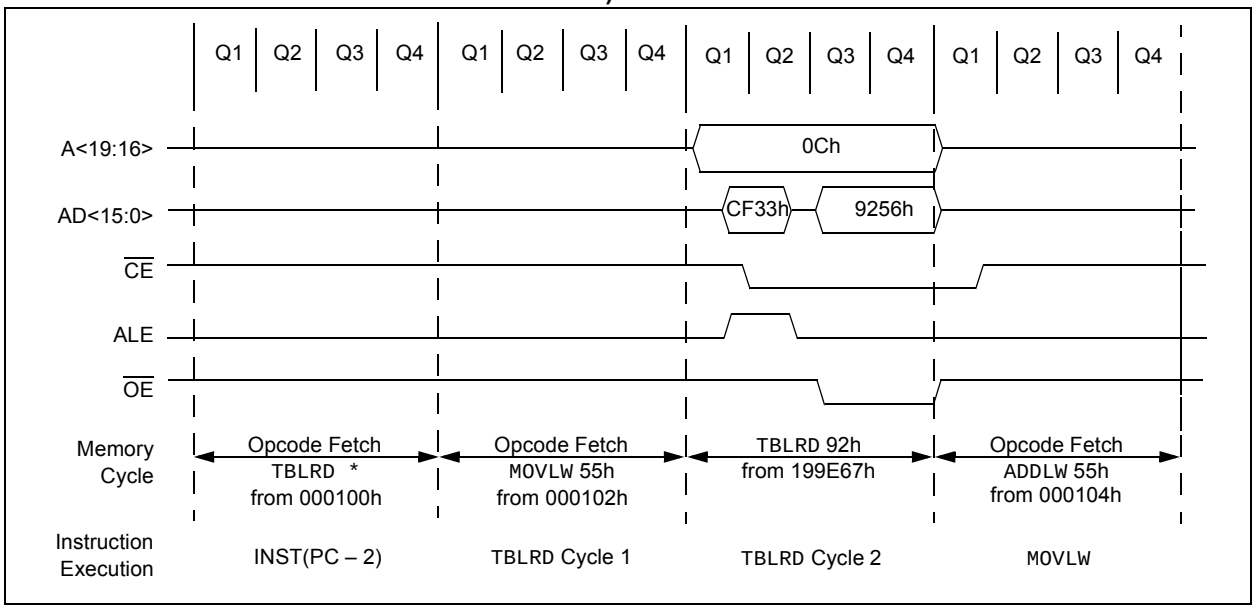
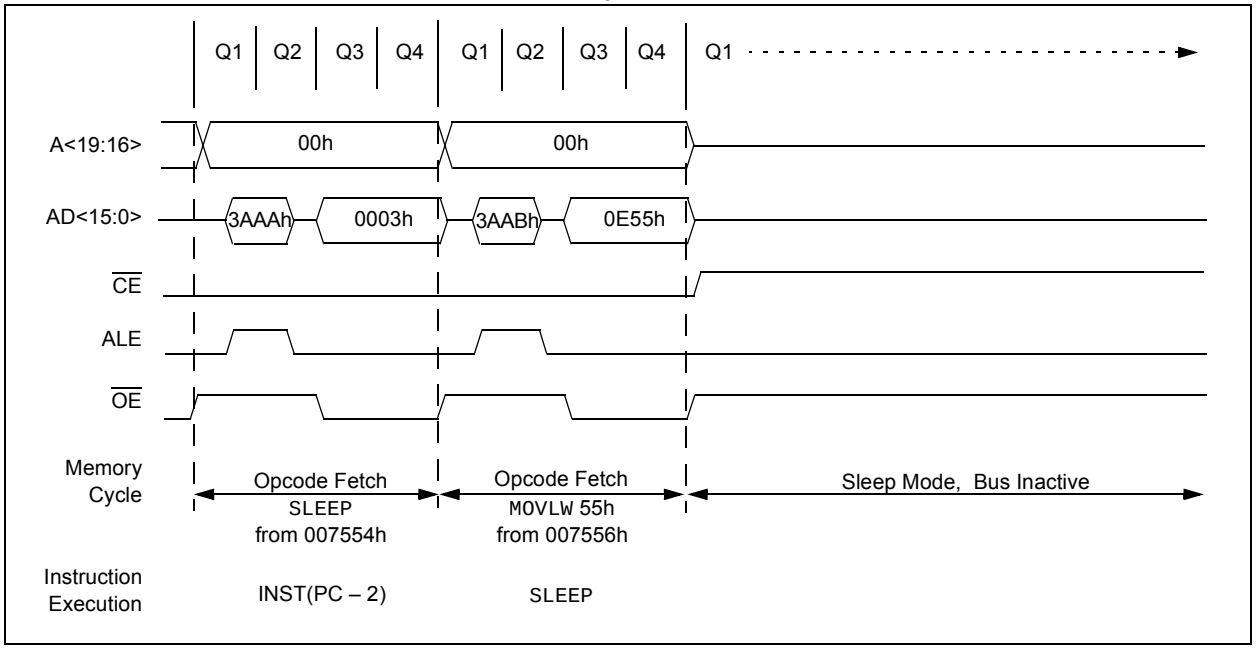


FIGURE 8-5: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



8.7 8-Bit Data Width Mode

In 8-Bit Data Width mode, the external memory bus operates only in Multiplexed mode; that is, data shares the 8 Least Significant bits of the address bus.

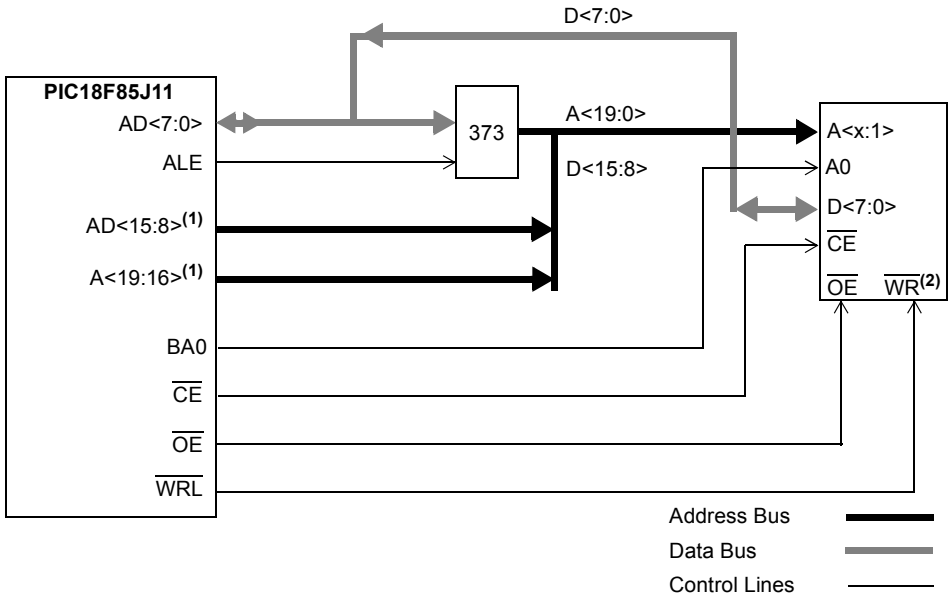
Figure 8-6 shows an example of 8-Bit Multiplexed mode for 80-pin devices. This mode is used for a single 8-bit memory connected for 16-bit operation. The instructions will be fetched as two 8-bit bytes on a shared data/address bus. The two bytes are sequentially fetched within one instruction cycle (TCY). Therefore, the designer must choose external memory devices according to timing calculations based on 1/2 TCY (2 times the instruction rate). For proper memory speed selection, glue logic propagation delay times must be considered, along with setup and hold times.

The Address Latch Enable (ALE) pin indicates that the address bits, AD<15:0>, are available on the external memory interface bus. The Output Enable signal ($\overline{OE}$) will enable one byte of program memory for a portion of the instruction cycle, then BA0 will change and the second byte will be enabled to form the 16-bit instruction word. The Least Significant bit of the address, BA0, must be connected to the memory devices in this mode. The Chip Enable signal ($\overline{CE}$) is active at any time that the microcontroller accesses external memory, whether reading or writing. It is inactive (asserted high) whenever the device is in Sleep mode.

This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate level of the BA0 control line is strobed on the LSb of the TBLPTR.

FIGURE 8-6: 8-BIT MULTIPLEXED MODE EXAMPLE



Note 1: The upper order address bits are only used 20-bit address width. The upper AD byte is used for all address widths except 8-bit.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

8.7.1 8-BIT MODE TIMING

The presentation of control signals on the external memory bus is different for the various operating modes. Typical signal timing diagrams are shown in Figure 8-7 and Figure 8-8.

FIGURE 8-7: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)

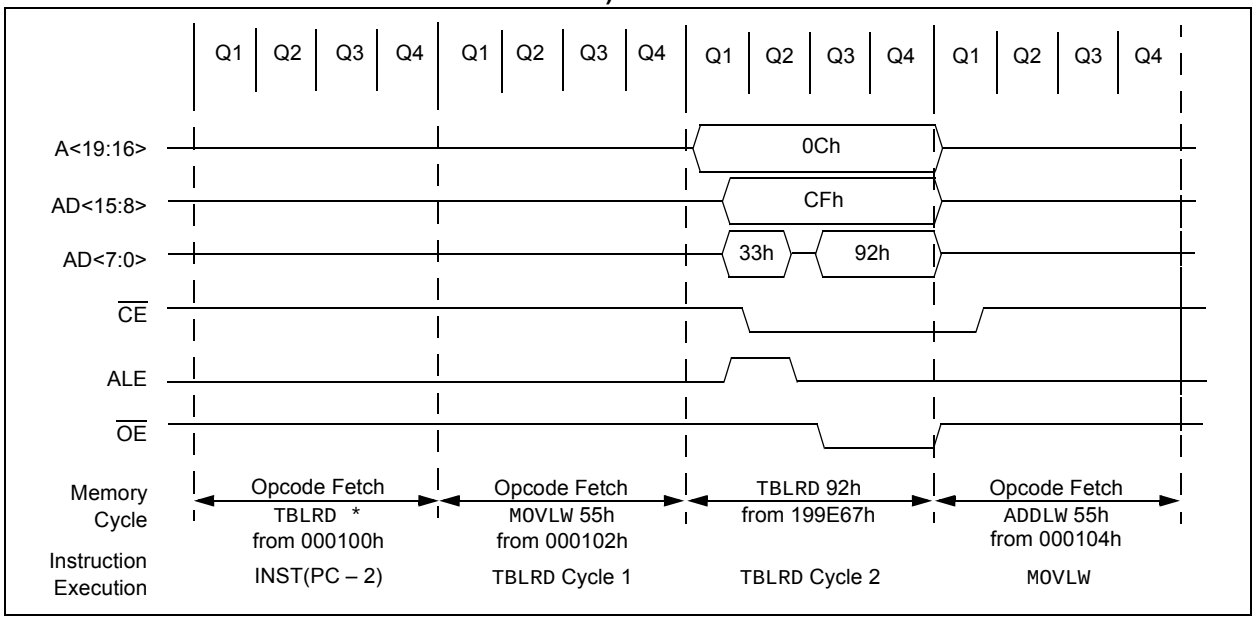
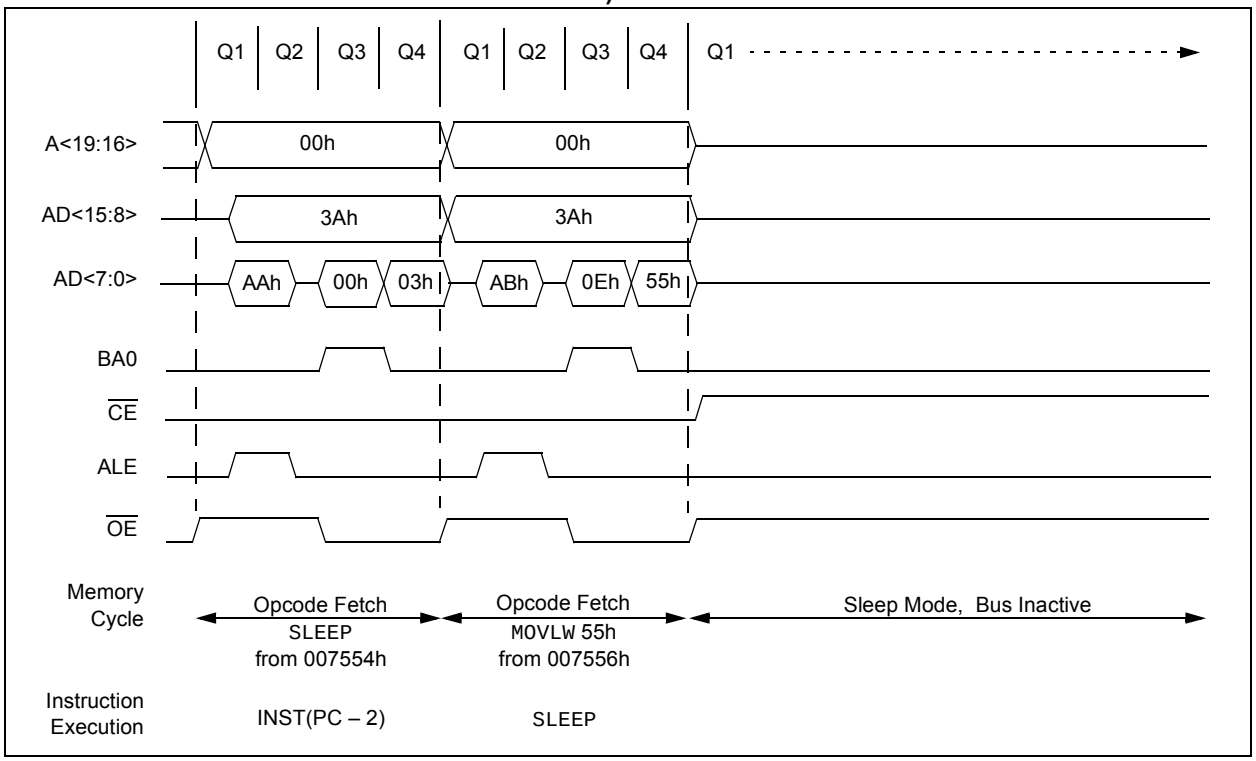


FIGURE 8-8: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



8.8 Operation in Power-Managed Modes

In alternate, power-managed Run modes, the external bus continues to operate normally. If a clock source with a lower speed is selected, bus operations will run at that speed. In these cases, excessive access times for the external memory may result if Wait states have been enabled and added to external memory operations. If operations in a lower power Run mode are anticipated, users should provide in their applications for adjusting memory access times at the lower clock speeds.

In Sleep and Idle modes, the microcontroller core does not need to access data; bus operations are suspended. The state of the external bus is frozen, with the address/data pins and most of the control pins holding at the same state they were in when the mode was invoked. The only potential changes are the $\overline{CE}$, $\overline{LB}$ and $\overline{UB}$ pins, which are held at logic high.

NOTES:

9.0 8 x 8 HARDWARE MULTIPLIER

9.1 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 9-1.

9.2 Operation

Example 9-1 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 9-2 shows the sequence to do an 8 x 8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 9-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

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

EXAMPLE 9-2: 8 x 8 SIGNED MULTIPLY ROUTINE

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

TABLE 9-1: 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

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

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

EXAMPLE 9-3: 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 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

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

EXAMPLE 9-4: 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
   :
```

10.0 INTERRUPTS

Members of the PIC18F85J11 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

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

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

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

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

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

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

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

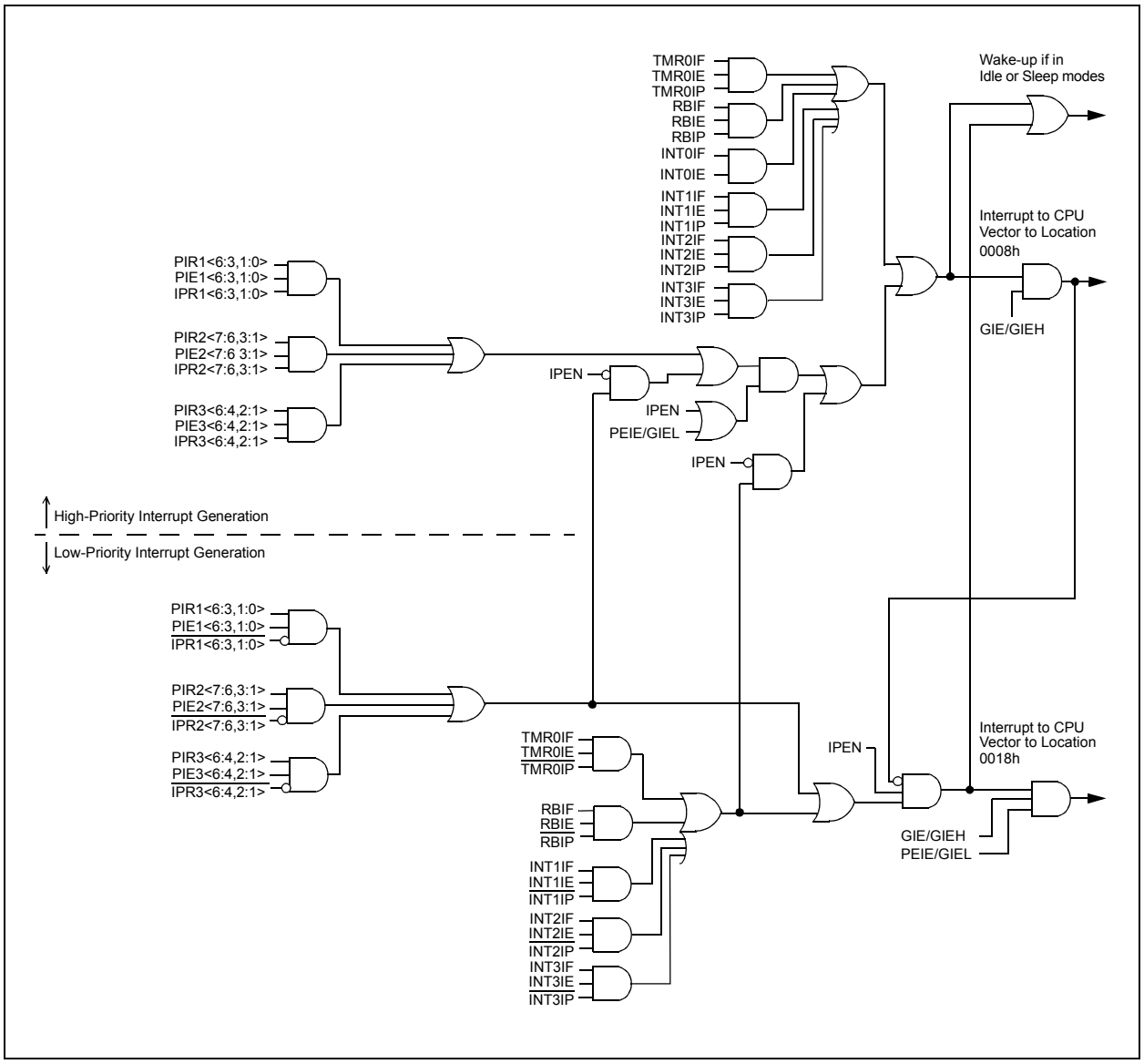
The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine (ISR), the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before 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.

FIGURE 10-1: PIC18F85J11 FAMILY INTERRUPT LOGIC



10.1 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 10-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF(1)
bit 7							bit 0

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

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit(1)
1 = At least one of the RB<7:4> pins changed state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

Note 1: A mismatch condition will continue to set this bit. Reading PORTB, and then waiting one additional instruction cycle, will end the mismatch condition and allow the bit to be cleared.

REGISTER 10-2: 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.

REGISTER 10-3: 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.

10.2 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 10-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1

R/W-0	R/W-0	R-0	R-0	R/W-0	U-0	R/W-0	R/W-0
PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF
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 **PSPIF:** Parallel Slave Port Read/Write Interrupt Flag bit
1 = A read or a write operation has taken place (must be cleared in software)
0 = No read or write has occurred

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 **Unimplemented:** Read as ‘0’

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

REGISTER 10-5: 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	U-0
OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—
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 is 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 **LVDIF:** Low-Voltage Detect Interrupt Flag bit
1 = A low-voltage condition occurred (must be cleared in software)
0 = The device voltage is above the regulator's low-voltage 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 **Unimplemented:** Read as '0'

REGISTER 10-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

U-0	U-0	R-0	R-0	U-0	R/W-0	R/W-0	U-0
—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—
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 **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 **Unimplemented:** Read as '0'

bit 2 **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.

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

10.3 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 10-7: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	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 **PSPIE:** Parallel Slave Port Read/Write Interrupt Enable bit
1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

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

bit 5 **RC1IE:** 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 **Unimplemented:** Read as '0'

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

REGISTER 10-8: 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	U-0
OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—
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 **BCLIE:** Bus Collision Interrupt Enable bit
1 = Enabled
0 = Disabled

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

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

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

REGISTER 10-9: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

U-0	U-0	R-0	R-0	U-0	R/W-0	R/W-0	U-0
—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—
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 **RC2IE:** AUSART Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TX2IE:** AUSART Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 2 **CCP2IE:** CCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

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

10.4 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 10-10: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1
PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	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 **PSPIP:** Parallel Slave Port Read/Write Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 5 **RC1IP:** EUSART Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX1IP:** EUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSPIP:** Master Synchronous Serial Port Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-11: 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	U-0
OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—
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 **LVDIP:** Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR3IP:** TMR3 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

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

REGISTER 10-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

U-0	U-0	R-1	R-1	U-0	R/W-1	R/W-1	U-0
—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—
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 **RC2IP:** AUSART Receive Priority Flag bit
1 = High priority
0 = Low priority

bit 4 **TX2IP:** AUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **Unimplemented:** Read as ‘0’

10.5 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 10-13: RCON: RESET CONTROL REGISTER

R/W-0	U-0	R/W-1	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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 = Enables priority levels on interrupts
0 = Disables priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **$\overline{CM}$:** Configuration Mismatch Flag bit
For details of bit operation, see Register 5-1.

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
For details of bit operation, see Register 5-1.

bit 3 **$\overline{TO}$:** Watchdog Timer Time-out Flag bit
For details of bit operation, see Register 5-1.

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
For details of bit operation, see Register 5-1.

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
For details of bit operation, see Register 5-1.

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
For details of bit operation, see Register 5-1.

10.6 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. Flag bit, INTxIF, must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt.

All external interrupts (INT0, INT1, INT2 and INT3) can wake-up the processor from the power-managed modes if bit INTxIE was set prior to going into the power-managed modes. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1, INT2 and INT3 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>), INT2IP (INTCON3<7>) and INT3IP (INTCON2<1>). There is no priority bit associated with INT0. It is always a high-priority interrupt source.

10.7 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 12.0 "Timer0 Module"** for further details on the Timer0 module.

10.8 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>).

10.9 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 6.3 "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 10-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 10-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

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

11.0 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 memory mapped registers for its operation:

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

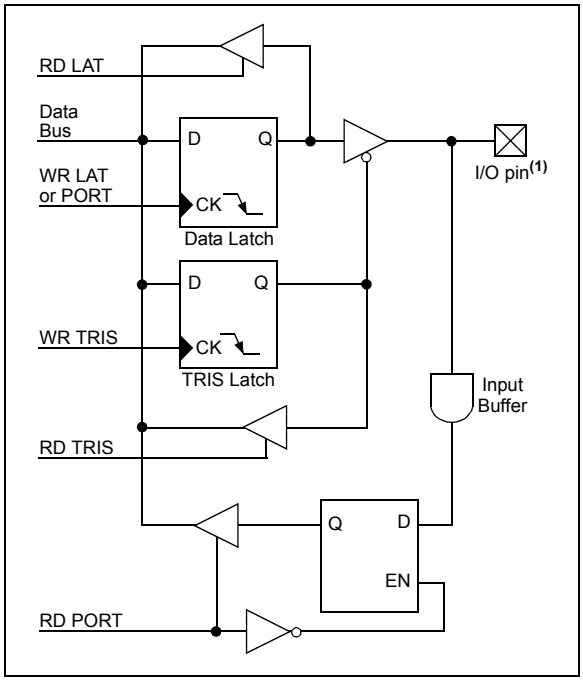
Reading the PORT register reads the current status of the pins, whereas writing to the PORT register writes to the Output Latch (LAT) register.

Setting a TRIS bit (= 1) makes the corresponding PORT pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRIS bit (= 0) makes the corresponding PORT pin an output (i.e., put the contents of the corresponding LAT bit on the selected pin).

The Output Latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving. Read-modify-write operations on the LAT register read and write the latched output value for the PORT register.

A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



11.1 I/O Port Pin Capabilities

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

11.1.1 INPUT PINS AND VOLTAGE CONSIDERATIONS

The voltage tolerance of pins used as device inputs is dependent on the pin's input function. Most pins that are used as digital only inputs are able to handle DC voltages up to 5.5V, a level typical for digital logic circuits. The digital pins that cannot exceed VDD are RE0, RE1, RE2, RG0, RG2 and RG3.

In contrast, pins that also have analog input functions of any kind can only tolerate voltages up to VDD. Voltage excursions beyond VDD on these pins should be avoided.

Table 11-1 summarizes the input voltage capabilities. Refer to **Section 26.0 "Electrical Characteristics"** for more details.

TABLE 11-1: INPUT VOLTAGE TOLERANCE

Port or Pin	Tolerated Input	Description
PORTA<7:5>	VDD	Only VDD input levels tolerated.
PORTA<3:0>		
PORTC<1:0>		
PORTE<2:0>		
PORTF<7:1>		
PORTG<3:2,0>		
PORTA<4>	5.5V	Tolerates input levels above VDD, useful for most standard logic.
PORTB<7:0>		
PORTC<7:2>		
PORTD<7:0>		
PORTE<7:3>		
PORTG<4,1>		
PORTH<7:0>(1)		
PORTJ<7:0>(1)		

Note 1: Not available on 64-pin devices.

11.1.2 PIN OUTPUT DRIVE

When used as digital I/O, the output pin drive strengths vary for groups of pins intended to meet the needs for a variety of applications. In general, there are three classes of output pins in terms of drive capability.

PORTB and PORTC, as well as PORTA<7:6>, are designed to drive higher current loads, such as LEDs. PORTD, PORTE and PORTJ are capable of driving digital circuits associated with external memory devices. They can also drive LEDs, but only those with smaller current requirements. PORTF, PORTG and PORTH, along with PORTA<5:0>, have the lowest drive level, but are capable of driving normal digital circuit loads with a high input impedance.

Table 11-2 summarizes the output capabilities of the ports. Refer to the **"Absolute Maximum Ratings"** in **Section 26.0 "Electrical Characteristics"** for more details.

TABLE 11-2: OUTPUT DRIVE LEVELS FOR VARIOUS PORTS

Low	Medium	High
PORTA<5:0>	PORTD	PORTA<7:6>
PORTF	PORTE	PORTB
PORTG	PORTJ(1)	PORTC
PORTH(1)		

Note 1: Not available on 64-pin devices.

11.1.3 PULL-UP CONFIGURATION

Four of the I/O ports (PORTB, PORTD, PORTE and PORTJ) implement configurable weak pull-ups on all pins. These are internal pull-ups that allow floating digital input signals to be pulled to a consistent level without the use of external resistors.

The pull-ups are enabled with a single bit for each of the ports: $\overline{\text{RBPU}}$ (INTCON2<7>) for PORTB, and RDPU, REPU and RJPU (PORTG<7:5>) for the other ports.

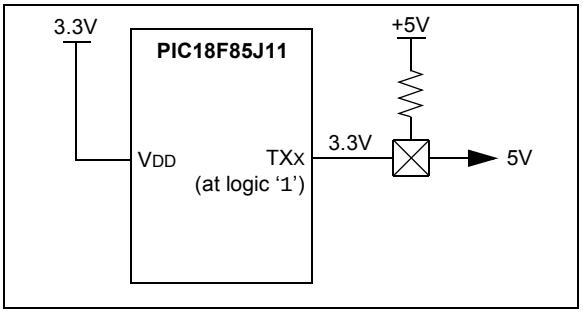
11.1.4 OPEN-DRAIN OUTPUTS

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

The open-drain option is implemented on port pins specifically associated with the data and clock outputs of the USARTs, the MSSP module (in SPI mode) and the CCP modules. The option is selectively enabled by setting the open-drain control bit for the corresponding module in TRISG and LATG. Their configuration is discussed in more detail in the sections for PORTC, PORTE and PORTG.

When the open-drain option is required, the output pin must also be tied through an external pull-up resistor provided by the user to a higher voltage level, up to 5V (Figure 11-2). When a digital logic high signal is output, it is pulled up to the higher voltage level.

FIGURE 11-2: USING THE OPEN-DRAIN OUTPUT (USARTs SHOWN AS EXAMPLES



11.2 PORTA, TRISA and LATA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISA and LATA.

RA4/T0CKI is a Schmitt Trigger input. All other PORTA pins have TTL input levels and full CMOS output drivers.

The RA4 pin is multiplexed with the Timer0 clock input. RA5 and RA<3:0> are multiplexed with analog inputs for the A/D Converter.

The operation of the analog inputs as A/D Converter inputs is selected by clearing or setting the PCFG<3:0> control bits in the ADCON1 register. The corresponding TRISA bits control the direction of these 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.

Note:	RA5 and RA<3:0> are configured as analog inputs on any Reset and are read as '0'. RA4 is configured as a digital input.

RA6/OSC2/CLKO and RA7/OSC1/CLKI normally serve as the external circuit connections for the external (primary) oscillator circuit (HS Oscillator modes), or the external clock input and output (EC Oscillator modes). In these cases, RA6 and RA7 are not available as digital I/O and their corresponding TRIS and LAT bits are read as '0'. When the device is configured to use INTOSC or INTRC as the default oscillator mode (FOSC2 Configuration bit is '0'), RA6 and RA7 are automatically configured as digital I/O; the oscillator and clock in/clock out functions are disabled.

EXAMPLE 11-1: INITIALIZING PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output latches
CLRF    LATA    ; Alternate method to
                ; clear output data latches
MOVLW   07h     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   0BFh    ; Value used to initialize
                ; data direction
MOVWF   TRISA   ; Set RA<7, 5:0> as inputs,
                ; RA<6> as output
```

TABLE 11-3: PORTA FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RA0/AN0	RA0	0	O	DIG	LATA<0> data output; not affected by analog input.
		1	I	TTL	PORTA<0> data input; disabled when analog input enabled.
	AN0	1	I	ANA	A/D Input Channel 0. Default input configuration on POR; does not affect digital output.
RA1/AN1	RA1	0	O	DIG	LATA<1> data output; not affected by analog input.
		1	I	TTL	PORTA<1> data input; disabled when analog input enabled.
	AN1	1	I	ANA	A/D Input Channel 1. Default input configuration on POR; does not affect digital output.
RA2/AN2/VREF-	RA2	0	O	DIG	LATA<2> data output; not affected by analog input.
		1	I	TTL	PORTA<2> data input. Disabled when analog functions enabled.
	AN2	1	I	ANA	A/D Input Channel 2. Default input configuration on POR.
	VREF-	1	I	ANA	A/D and comparator low reference voltage input.
RA3/AN3/ VREF+	RA3	0	O	DIG	LATA<3> data output; not affected by analog input.
		1	I	TTL	PORTA<3> data input; disabled when analog input enabled.
	AN3	1	I	ANA	A/D Input Channel 3. Default input configuration on POR.
	VREF+	1	I	ANA	A/D and comparator high reference voltage input.
RA4/T0CKI	RA4	0	O	DIG	LATA<4> data output.
		1	I	ST	PORTA<4> data input; default configuration on POR.
	T0CKI	x	I	ST	Timer0 clock input.
RA5/AN4	RA5	0	O	DIG	LATA<5> data output; not affected by analog input.
		1	I	TTL	PORTA<5> data input; disabled when analog input enabled.
	AN4	1	I	ANA	A/D Input Channel 4. Default configuration on POR.
RA6/OSC2/ CLKO	RA6	0	O	DIG	LATA<6> data output; disabled when FOSC2 Configuration bit is set.
		1	I	TTL	PORTA<6> data input; disabled when FOSC2 Configuration bit is set.
	OSC2	x	O	ANA	Main oscillator feedback output connection (HS and HSPLL modes).
	CLKO	x	O	DIG	System cycle clock output, FOSC/4 (EC and ECPLL modes).
RA7/OSC1/ CLKI	RA7	0	O	DIG	LATA<7> data output; disabled when FOSC2 Configuration bit is set.
		1	I	TTL	PORTA<7> data input; disabled when FOSC2 Configuration bit is set.
	OSC1	x	I	ANA	Main oscillator input connection (HS and HSPLL modes).
	CLKI	x	I	ANA	Main external clock source input (EC and ECPLL modes).

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 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTA	RA7(1)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	60
LATA	LATA7(1)	LATA6(1)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	60
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	60
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	59

Legend: — = Unimplemented, read as '0', x = Don't care. Shaded cells are not used by PORTA.

Note 1: These bits are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read as 'x'.

11.3 PORTB, TRISB and LATB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISB and LATB. All pins on PORTB are digital only and tolerate voltages up to 5.5V.

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{\text{RBPU}}$ (INTCON2<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of the PORTB pins (RB<7:4>) have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB<7:4> pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB<7:4>) are compared with the old value latched on the last read of PORTB. The “mismatch” outputs of RB<7:4> are ORed together to generate the RB Port Change Interrupt with Flag bit, RBIF (INTCON<0>).

This interrupt can wake the device from 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) Wait one instruction cycle (such as executing a NOP instruction).
c) 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 after one TCY delay.

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

EXAMPLE 11-2: 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
```

TABLE 11-5: PORTB FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RB0/INT0	RB0	0	O	DIG	LATB<0> data output.
		1	I	TTL	PORTB<0> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	INT0	1	I	ST	External Interrupt 0 input.
RB1/INT1	RB1	0	O	DIG	LATB<1> data output.
		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	RB2	0	O	DIG	LATB<2> data output.
		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/ CCP2	RB3	0	O	DIG	LATB<3> data output.
		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.
	CCP2(1)	0	O	DIG	CCP2 compare output and CCP2 PWM output; takes priority over port data.
		1	I	ST	CCP2 capture input.
RB4/KBI0	RB4	0	O	DIG	LATB<4> data output.
		1	I	TTL	PORTB<4> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI0		I	TTL	Interrupt-on-pin change.
RB5/KBI1	RB5	0	O	DIG	LATB<5> data output.
		1	I	TTL	PORTB<5> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI1		I	TTL	Interrupt-on-pin change.
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> data output.
		1	I	TTL	PORTB<6> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI2	1	I	TTL	Interrupt-on-pin change.
	PGC	x	I	ST	Serial execution (ICSP™) clock input for ICSP and ICD operation.(2)
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> data output.
		1	I	TTL	PORTB<7> data input; weak pull-up when $\overline{\text{RBPU}}$ bit is cleared.
	KBI3	1	I	TTL	Interrupt-on-pin change.
	PGD	x	O	DIG	Serial execution data output for ICSP and ICD operation.(2)
		x	I	ST	Serial execution data input for ICSP and ICD operation.(2)

Legend: O = Output, I = Input, 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 is cleared (Extended Microcontroller mode, 80-pin devices only). Default assignment is RC1.

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

TABLE 11-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	60
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	60
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	57
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	57

Legend: Shaded cells are not used by PORTB.

11.4 PORTC, TRISC and LATC Registers

PORTC is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISC and LATC. Only PORTC pins, RC2 through RC7, are digital only pins and can tolerate input voltages up to 5.5V.

PORTC is multiplexed with CCP, MSSP and EUSART peripheral functions (Table 11-7). The pins have Schmitt Trigger input buffers. The pins for CCP, SPI and EUSART are also configurable for open-drain output whenever these functions are active. Open-drain configuration is selected by setting the SPIOD, CCPxOD and U1OD control bits (TRISG<7:5> and LATG<6>, respectively).

RC1 is normally configured as the default peripheral pin for the CCP2 module. Assignment of CCP2 is controlled by Configuration bit, CCP2MX (default 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:	These pins are configured as digital inputs on any device Reset.

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

EXAMPLE 11-3: INITIALIZING PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; 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
```

TABLE 11-7: PORTC FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RC0/T1OSO/ T13CKI	RC0	0	O	DIG	LATC<0> data output.
		1	I	ST	PORTC<0> data input.
	T1OSO	x	O	ANA	Timer1 oscillator output; enabled when Timer1 oscillator enabled. Disables digital I/O.
	T13CKI	1	I	ST	Timer1/Timer3 counter input.
RC1/T1OSI/ CCP2	RC1	0	O	DIG	LATC<1> data output.
		1	I	ST	PORTC<1> data input.
	T1OSI	x	I	ANA	Timer1 oscillator input; enabled when Timer1 oscillator enabled. Disables digital I/O.
	CCP2(1)	0	O	DIG	CCP2 compare output and CCP2 PWM output; takes priority over port data.
		1	I	ST	CCP2 capture input.
RC2/CCP1	RC2	0	O	DIG	LATC<2> data output.
		1	I	ST	PORTC<2> data input.
	CCP1	0	O	DIG	CCP1 compare output and CCP1 PWM output; takes priority over port data.
		1	I	ST	CCP1 capture input.
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^2C™ clock output (MSSP module); takes priority over port data.
		1	I	I^2C	I^2C 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^2C data output (MSSP module); takes priority over port data.
		1	I	I^2C	I^2C data input (MSSP module); input type depends on module setting.
RC5/SDO	RC5	0	O	DIG	LATC<5> data output.
		1	I	ST	PORTC<5> data input.
	SDO	0	O	DIG	SPI data output (MSSP module); takes priority over port data.
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.
	CK1	1	O	DIG	Synchronous serial data input (EUSART module). User must configure as an input.
		1	I	ST	Synchronous serial clock input (EUSART module).
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, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

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

TABLE 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	60
LATC	LATC7	LATBC6	LATC5	LATCB4	LATC3	LATC2	LATC1	LATC0	60
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60

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

11.5 PORTD, TRISD and LATD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISD and LATD. All pins on PORTD are digital only and tolerate voltages up to 5.5V.

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

Note:	These pins are configured as digital inputs on any device Reset.

Each of the PORTD pins has a weak internal pull-up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RDPU (PORTG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on all device Resets.

On 80-pin devices, PORTD is multiplexed with the system bus as part of the external memory interface. I/O port and other functions are only available when the interface is disabled, by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTD is the low-order byte of the multiplexed address/data bus (AD<7:0>). The TRISD bits are also overridden.

PORTD can also be configured to function as an 8-bit wide, parallel microprocessor port by setting the PSPMODE control bit (PSPCON<4>). In this mode, parallel port data takes priority over other digital I/O (but not the external memory interface). When the parallel port is active, the input buffers are TTL. For more information, refer to **Section 11.11 "Parallel Slave Port"**.

EXAMPLE 11-4: 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 11-9: PORTD FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD0/AD0/PSP0	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input.
	AD0(2)	x	O	DIG	External memory interface, address/data bit 0 output.(1)
		x	I	TTL	External memory interface, data bit 0 input.(1)
	PSP0		O	DIG	PSP read output data (LATD<0>); takes priority over port data.
			I	TTL	PSP write data input.
RD1/AD1/PSP1	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input.
	AD1(2)	x	O	DIG	External memory interface, address/data bit 1 output.(1)
		x	I	TTL	External memory interface, data bit 1 input.(1)
	PSP1	x	O	DIG	PSP read output data (LATD<1>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD2/AD2/PSP2	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input.
	AD2(2)	x	O	DIG	External memory interface, address/data bit 2 output.(1)
		x	I	TTL	External memory interface, data bit 2 input.(1)
	PSP2	x	O	DIG	PSP read output data (LATD<2>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD3/AD3/PSP3	RD3	0	O	DIG	LATD<3> data output.
		1	I	ST	PORTD<3> data input.
	AD3(2)	x	O	DIG	External memory interface, address/data bit 3 output.(1)
		x	I	TTL	External memory interface, data bit 3 input.(1)
	PSP3	x	O	DIG	PSP read output data (LATD<3>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD4/AD4/PSP4	RD4	0	O	DIG	LATD<4> data output.
		1	I	ST	PORTD<4> data input.
	AD4(2)	x	O	DIG	External memory interface, address/data bit 4 output.(1)
		x	I	TTL	External memory interface, data bit 4 input.(1)
	PSP4	x	O	DIG	PSP read output data (LATD<4>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD5/AD5/PSP5	RD5	0	O	DIG	LATD<5> data output.
		1	I	ST	PORTD<5> data input.
	AD5(2)	x	O	DIG	External memory interface, address/data bit 5 output.(1)
		x	I	TTL	External memory interface, data bit 5 input.(1)
	PSP5	x	O	DIG	PSP read output data (LATD<5>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD6/AD6/PSP6	RD6	0	O	DIG	LATD<6> data output.
		1	I	ST	PORTD<6> data input.
	AD6(2)	x	O	DIG-3	External memory interface, address/data bit 6 output.(1)
		x	I	TTL	External memory interface, data bit 6 input.(1)
	PSP6	x	O	DIG	PSP read output data (LATD<6>); takes priority over port data.
		x	I	TTL	PSP write data input.

Legend: O = Output, I = Input, 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: External memory interface I/O takes priority over all other digital and PSP I/O.

2: Available on 80-pin devices only.

TABLE 11-9: PORTD FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD7/AD7/PSP7	RD7	0	O	DIG	LATD<7> data output.
		1	I	ST	PORTD<7> data input.
	AD7(2)	x	O	DIG	External memory interface, address/data bit 7 output.(1)
		x	I	TTL	External memory interface, data bit 7 input.(1)
	PSP7	x	O	DIG	PSP read output data (LATD<7>); takes priority over port data.
		x	I	TTL	PSP write data input.

Legend: O = Output, I = Input, 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: External memory interface I/O takes priority over all other digital and PSP I/O.

2: Available on 80-pin devices only.

TABLE 11-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	60
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	60
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	60
PORTG	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	60

Legend: Shaded cells are not used by PORTD.

Note 1: Unimplemented on 64-pin devices, read as '0'.

11.6 PORTE, TRISE and LATE Registers

PORTE is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISE and LATE. All pins on PORTE are digital only and tolerate voltages up to 5.5V.

All pins on PORTE are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output. The RE7 pin is also configurable for open-drain output when CCP2 is active on this pin. Open-drain configuration is selected by setting the CCP2OD control bit (TRISG<6>)

Note:	These pins are configured as digital inputs on any device Reset.

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

On 80-pin devices, PORTE is multiplexed with the system bus as part of the external memory interface. I/O port and other functions are only available when the interface is disabled by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTE is the high-order byte of the multiplexed address/data bus (AD<15:8>). The TRISE bits are also overridden.

When the Parallel Slave Port is active on PORTD, three of the PORTE pins (RE0, RE1 and RE2) are configured as digital control inputs for the port. The control functions are summarized in Table 11-11. The reconfiguration occurs automatically when the PSPMODE control bit (PSPCON<4>) is set. Users must still make certain the corresponding TRISE bits are set to configure these pins as digital inputs.

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

EXAMPLE 11-5: INITIALIZING PORTE

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   03h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISE   ; Set RE<1:0> as inputs
                ; RE<7:2> as outputs
```

TABLE 11-11: PORTE FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RE0/$\overline{RD}$/AD8	RE0	0	O	DIG	LATE<0> data output.
		1	I	ST	PORTE<0> data input.
	$\overline{RD}$	1	I	TTL	Parallel Slave Port read enable control input.
	AD8(1)	x	O	DIG	External memory interface, address/data bit 8 output.(2)
		x	I	TTL	External memory interface, data bit 8 input.(2)
RE1/$\overline{WR}$/AD9	RE1	0	O	DIG	LATE<1> data output.
		1	I	ST	PORTE<1> data input.
	$\overline{WR}$	1	I	TTL	Parallel Slave Port write enable control input.
	AD9(1)	x	O	DIG	External memory interface, address/data bit 9 output.(2)
		x	I	TTL	External memory interface, data bit 9 input.(2)
RE2/AD10/$\overline{CS}$	RE2	0	O	DIG	LATE<2> data output.
		1	I	ST	PORTE<2> data input.
	AD10(1)	x	O	DIG	External memory interface, address/data bit 10 output.(2)
		x	I	TTL	External memory interface, data bit 10 input.(2)
	$\overline{CS}$	1	I	TTL	Parallel Slave Port chip select control input.
RE3/AD11	RE3	0	O	DIG	LATE<3> data output.
		1	I	ST	PORTE<3> data input.
	AD11(1)	x	O	DIG	External memory interface, address/data bit 11 output.(2)
		x	I	TTL	External memory interface, data bit 11 input.(2)
RE4/AD12	RE4	0	O	DIG	LATE<4> data output.
		1	I	ST	PORTE<4> data input.
	AD12(1)	x	O	DIG	External memory interface, address/data bit 12 output.(2)
		x	I	TTL	External memory interface, data bit 12 input.(2)
RE5/AD13	RE5	0	O	DIG	LATE<5> data output.
		1	I	ST	PORTE<5> data input.
	AD13(1)	x	O	DIG	External memory interface, address/data bit 13 output.(2)
		x	I	TTL	External memory interface, data bit 13 input.(2)
RE6/AD14	RE6	0	O	DIG	LATE<6> data output.
		1	I	ST	PORTE<6> data input.
	AD14(1)	x	O	DIG	External memory interface, address/data bit 14 output.(2)
		x	I	TTL	External memory interface, data bit 14 input.(2)
RE7/AD15/ CCP2	RE7	0	O	DIG	LATE<7> data output.
		1	I	ST	PORTE<7> data input.
	AD15(1)	x	O	DIG	External memory interface, address/data bit 15 output.(2)
		x	I	TTL	External memory interface, data bit 15 input.(2)
	CCP2(3)	0	O	DIG	CCP2 compare output and CCP2 PWM output; takes priority over port data.
		1	I	ST	CCP2 capture input.

Legend: O = Output, I = Input, 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: Available on 80-pin devices only.

2: External memory interface I/O takes priority over all other digital and PSP I/O.

3: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared (all devices in Microcontroller mode).

TABLE 11-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	60
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	60
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	60
PORTG	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60

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

Note 1: Unimplemented on 64-pin devices, read as '0'.

11.7 PORTF, LATF and TRISF Registers

PORTF is a 7-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISF and LATF. 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 analog peripheral functions. Pins RF1 through RF6 may be used as comparator inputs or outputs by setting the appropriate bits in the CMCON register. To use RF<6:3> as digital inputs, it is also necessary to turn off the comparators.

Note 1: On device Resets, pins, RF<6:1>, are configured as analog inputs and are read as '0'.

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

EXAMPLE 11-6: INITIALIZING PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ;
MOVWF   CMCON   ; Turn off comparators
MOVLW   0Fh;
MOVWF   ADCON1  ; Set PORTF as digital I/O
MOVLW   0CEh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF1 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

TABLE 11-13: PORTF FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RF1/AN6/ C2OUT	RF1	0	O	DIG	LATF<1> data output; not affected by analog input.
		1	I	ST	PORTF<1> data input; disabled when analog input enabled.
	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.
RF2/AN7/ C1OUT	RF2	0	O	DIG	LATF<2> data output; not affected by analog input.
		1	I	ST	PORTF<2> data input; disabled when analog input enabled.
	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.
RF3/AN8	RF3	0	O	DIG	LATF<3> data output; not affected by analog input.
		1	I	ST	PORTF<3> data input; disabled when analog input enabled.
	AN8	1	I	ANA	A/D Input Channel 8 and Comparator C2+ input. Default input configuration on POR; not affected by analog output.
RF4/AN9	RF4	0	O	DIG	LATF<4> data output; not affected by analog input.
		1	I	ST	PORTF<4> data input; disabled when analog input enabled.
	AN9	1	I	ANA	A/D Input Channel 9 and Comparator C2- input. Default input configuration on POR; does not affect digital output.
RF5/AN10/ CVREF	RF5	0	O	DIG	LATF<5> data output; not affected by analog input. Disabled when CVREF output enabled.
		1	I	ST	PORTF<5> data input; disabled when analog input enabled. Disabled when CVREF output enabled.
	AN10	1	I	ANA	A/D Input Channel 10 and Comparator C1+ input. Default input configuration on POR.
	CVREF	x	O	ANA	Comparator voltage reference output. Enabling this feature disables digital I/O.
RF6/AN11	RF6	0	O	DIG	LATF<6> data output; not affected by analog input.
		1	I	ST	PORTF<6> data input; disabled when analog input enabled.
	AN11	1	I	ANA	A/D Input Channel 11 and Comparator C1- input. Default input configuration on POR; does not affect digital output.
RF7/AN5/$\overline{SS}$	RF7	0	O	DIG	LATF<7> data output.
		1	I	ST	PORTF<7> data input.
	AN5	1	I	ANA	A/D Input Channel 5. Default configuration on POR.
	$\overline{SS}$	1	I	TTL	Slave select input for MSSP module.

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 11-14: 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
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	60
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	59
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	59
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	59

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

11.8 PORTG, TRISG and LATG Registers

PORTG is a 5-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISG and LATG. All pins on PORTG are digital only and tolerate voltages up to 5.5V.

When operating as I/O, all PORTG pins have Schmitt Trigger input buffers. Pins, RG1 and RG2, are multiplexed with the AUSART module. The RG1 pin is also configurable for open-drain output when the AUSART is active. Open-drain configuration is selected by setting the U2OD control bit (LATG<7>).

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.

Although the port itself is only five bits wide, the PORTG<7:5> bits are still implemented to control the weak pull-ups on the I/O ports associated with PORTD, PORTE and PORTJ. Clearing these bits enables the respective port pull-ups. All pull-ups are disabled by default on all device Resets.

Most of the corresponding TRISG and LATG bits are implemented as open-drain control bits for CCP1, CCP2 and SPI (TRISG<7:5>), and the USARTs (LATG<7:6>). Setting these bits configures the output pin for the corresponding peripheral for open-drain operation. LATG<5> is not implemented.

EXAMPLE 11-7: INITIALIZING PORTG

```
CLRF    PORTG   ; Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATG    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   04h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISG   ; Set RG1:RG0 as outputs
                ; RG2 as input
                ; RG4:RG3 as inputs
```

TABLE 11-15: PORTG FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RG0	RG0	0	O	DIG	LATG<0> data output.
		1	I	ST	PORTG<0> data input.
RG1/TX2/CK2	R21	0	O	DIG	LATG<1> data output.
		1	I	ST	PORTG<1> data input.
	TX2	1	O	DIG	Synchronous serial data output (AUSART2 module); takes priority over port data.
	CK2	1	O	DIG	Synchronous serial data input (AUSART2 module). User must configure as an input.
		1	I	ST	Synchronous serial clock input (AUSART2 module).
RG2/RX2/DT2	RG2	0	O	DIG	LATG<2> data output.
		1	I	ST	PORTG<2> data input.
	RX2	1	I	ST	Asynchronous serial receive data input (AUSART2 module).
	DT2	1	O	DIG	Synchronous serial data output (AUSART2 module); takes priority over port data.
		1	I	ST	Synchronous serial data input (AUSART2 module). User must configure as an input.
RG3	RG3	0	O	DIG	LATG<3> data output.
		1	I	ST	PORTG<3> data input.
RG4	RG4	0	O	DIG	LATG<4> data output.
		1	I	ST	PORTG<4> data input.

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

TABLE 11-16: 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	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	60
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60

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

Note 1: Unimplemented on 64-pin devices, read as '0'.

11.9 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 and Output Latch registers are TRISH and LATH. All pins are digital only and tolerate voltages up to 5.5V.

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

When the external memory interface is enabled, four of the PORTH pins function as the high-order address lines for the interface. The address output from the interface takes priority over other digital I/O. The corresponding TRISH bits are also overridden.

EXAMPLE 11-8: INITIALIZING PORTH

```
CLRF    PORTH       ; Initialize PORTH by
                    ; clearing output
                    ; data latches
CLRF    LATH        ; Alternate method
                    ; to clear output
                    ; data latches
MOVLW   0Fh         ; Configure PORTH as
MOVWF   ADCON1      ; digital I/O
MOVLW   0CFh        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISH       ; Set RH3:RH0 as inputs
                    ; RH5:RH4 as outputs
                    ; RH7:RH6 as inputs
```

TABLE 11-17: PORTH FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RH0/A16	RH0	0	O	DIG	LATH<0> data output.
		1	I	ST	PORTH<0> data input.
	A16	x	O	DIG	External memory interface, Address Line 16. Takes priority over port data.
RH1/A17	RH1	0	O	DIG	LATH<1> data output.
		1	I	ST	PORTH<1> data input.
	A17	x	O	DIG	External memory interface, Address Line 17. Takes priority over port data.
RH2/A18	RH2	0	O	DIG	LATH<2> data output.
		1	I	ST	PORTH<2> data input.
	A18	x	O	DIG	External memory interface, Address Line 18. Takes priority over port data.
RH3/A19	RH3	0	O	DIG	LATH<3> data output.
		1	I	ST	PORTH<3> data input.
	A19	x	O	DIG	External memory interface, Address Line 19. Takes priority over port data.
RH4	RH4	0	O	DIG	LATH<4> data output.
		1	I	ST	PORTH<4> data input.
RH5	RH5	0	O	DIG	LATH<5> data output.
		1	I	ST	PORTH<5> data input.
RH6	RH6	0	O	DIG	LATH<6> data output.
		1	I	ST	PORTH<6> data input.
RH7	RH7	0	O	DIG	LATH<7> data output.
		1	I	ST	PORTH<7> data input.

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

TABLE 11-18: 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
PORTH	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	60
LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	60
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	60

11.10 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 and Output Latch registers are TRISJ and LATJ. All pins on PORTJ are digital only and tolerate voltages up to 5.5V.

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

Note:	These pins are configured as digital inputs on any device Reset.

Each of the PORTJ pins has a weak internal pull-up. The pull-ups are provided to keep the inputs at a known state for the external memory interface while powering up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RJPU (PORTG<5>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

When the external memory interface is enabled, all of the PORTJ pins function as control outputs for the interface. This occurs automatically when the interface is enabled by clearing the EBDIS control bit (MEMCON<7>). The TRISJ bits are also overridden.

EXAMPLE 11-9: INITIALIZING PORTJ

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

TABLE 11-19: PORTJ FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RJ0/ALE	RJ0	0	O	DIG	LATJ<0> data output.
		1	I	ST	PORTJ<0> data input.
	ALE	x	O	DIG	External memory interface address latch enable control output; takes priority over digital I/O.
RJ1/$\overline{OE}$	RJ1	0	O	DIG	LATJ<1> data output.
		1	I	ST	PORTJ<1> data input.
	$\overline{OE}$	x	O	DIG	External memory interface output enable control output; takes priority over digital I/O.
RJ2/$\overline{WRL}$	RJ2	0	O	DIG	LATJ<2> data output.
		1	I	ST	PORTJ<2> data input.
	$\overline{WRL}$	x	O	DIG	External memory bus write low byte control; takes priority over digital I/O.
RJ3/$\overline{WRH}$	RJ3	0	O	DIG	LATJ<3> data output.
		1	I	ST	PORTJ<3> data input.
	$\overline{WRH}$	x	O	DIG	External memory interface write high byte control output; takes priority over digital I/O.
RJ4/BA0	RJ4	0	O	DIG	LATJ<4> data output.
		1	I	ST	PORTJ<4> data input.
	BA0	x	O	DIG	External memory interface byte address 0 control output; takes priority over digital I/O.
RJ5/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> data output.
		1	I	ST	PORTJ<5> data input.
	$\overline{CE}$	x	O	DIG	External memory interface chip enable control output; takes priority over digital I/O.
RJ6/$\overline{LB}$	RJ6	0	O	DIG	LATJ<6> data output.
		1	I	ST	PORTJ<6> data input.
	$\overline{LB}$	x	O	DIG	External memory interface lower byte enable control output; takes priority over digital I/O.
RJ7/$\overline{UB}$	RJ7	0	O	DIG	LATJ<7> data output.
		1	I	ST	PORTJ<7> data input.
	$\overline{UB}$	x	O	DIG	External memory interface upper byte enable control output; takes priority over digital I/O.

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

TABLE 11-20: 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	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	60
LATJ	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	60
TRISJ	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	60
PORTG	RDPU	REPU	RJPU	RG4	RG3	RG2	RG1	RG0	60

Legend: Shaded cells are not used by PORTJ.

11.11 Parallel Slave Port

PORTD can also function as an 8-bit wide Parallel Slave Port, or microprocessor port, when control bit PSPMODE (PSPCON<4>) is set. It is asynchronously readable and writable by the external world through $\overline{RD}$ control input pin, RE0/$\overline{RD}$, and $\overline{WR}$ control input pin, RE1/$\overline{WR}$.

Note:	For 80-pin devices, the Parallel Slave Port is available only in Microcontroller mode.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an 8-bit latch. Setting bit PSPMODE enables port pin, RE0/$\overline{RD}$, to be the $\overline{RD}$ input, RE1/$\overline{WR}$ to be the $\overline{WR}$ input and RE2/$\overline{CS}$ to be the $\overline{CS}$ (Chip Select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (set).

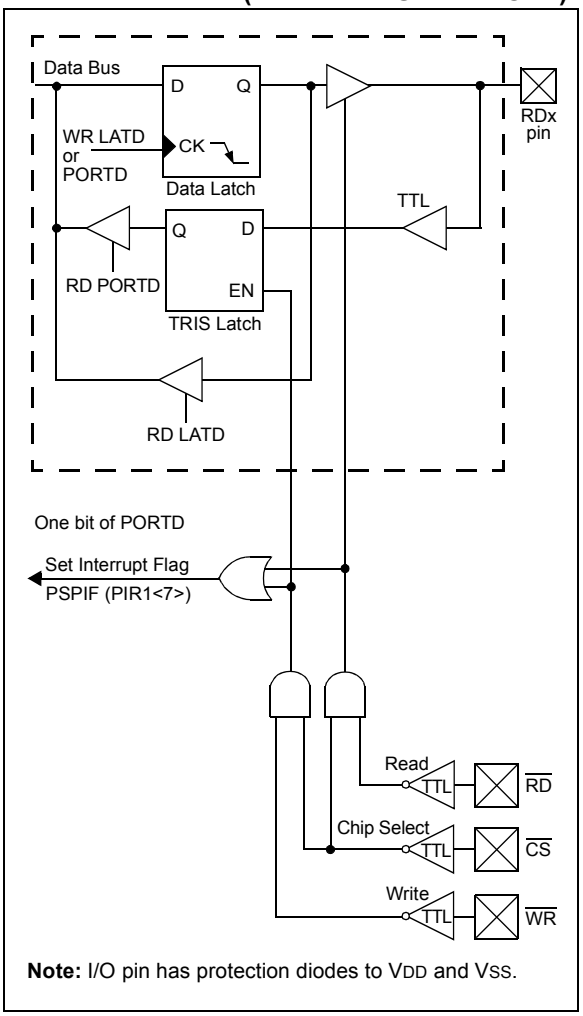
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits are both set when the write ends.

A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low. The data in PORTD is read out and the OBF bit is set. If the user writes new data to PORTD to set OBF, the data is immediately read out; however, the OBF bit is not set.

When either the $\overline{CS}$ or $\overline{RD}$ lines are detected high, the PORTD pins return to the input state and the PSPIF bit is set. User applications should wait for PSPIF to be set before servicing the PSP. When this happens, the IBF and OBF bits can be polled and the appropriate action taken.

The timing for the control signals in Write and Read modes is shown in Figure 11-4 and Figure 11-5, respectively.

FIGURE 11-3: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: I/O pin has protection diodes to VDD and VSS.

REGISTER 11-1: PSPCON: PARALLEL SLAVE PORT CONTROL REGISTER

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IBF	OBF	IBOV	PSPMODE	—	—	—	—
bit 7							bit 0

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

bit 7 **IBF:** Input Buffer Full Status bit
1 = A word has been received and is waiting to be read by the CPU
0 = No word has been received

bit 6 **OBF:** Output Buffer Full Status bit
1 = The output buffer still holds a previously written word
0 = The output buffer has been read

bit 5 **IBOV:** Input Buffer Overflow Detect bit
1 = A write occurred when a previously input word has not been read (must be cleared in software)
0 = No overflow occurred

bit 4 **PSPMODE:** Parallel Slave Port Mode Select bit
1 = Parallel Slave Port mode
0 = General Purpose I/O mode

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

FIGURE 11-4: PARALLEL SLAVE PORT WRITE WAVEFORMS

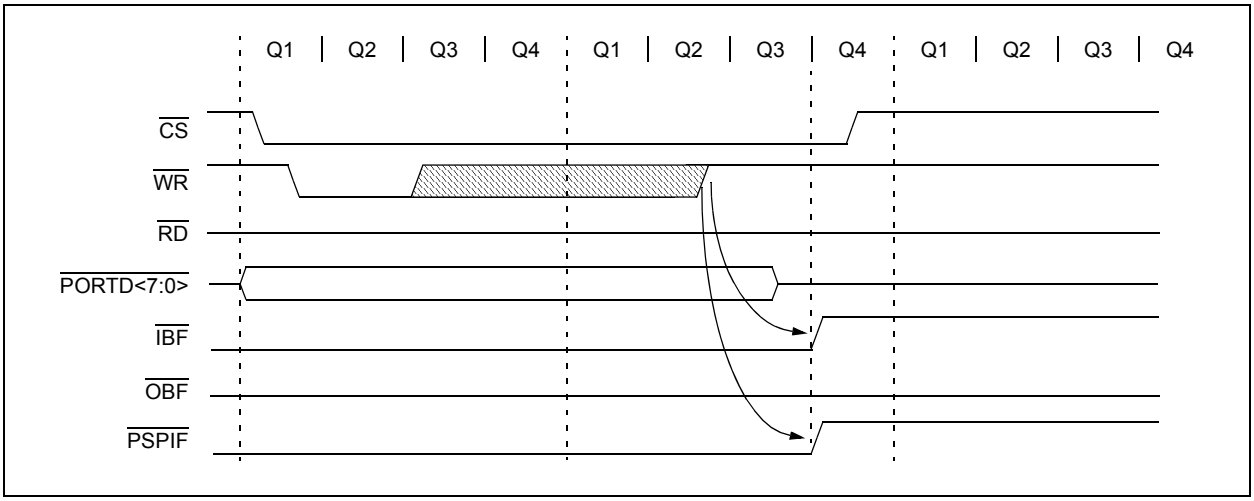


FIGURE 11-5: PARALLEL SLAVE PORT READ WAVEFORMS

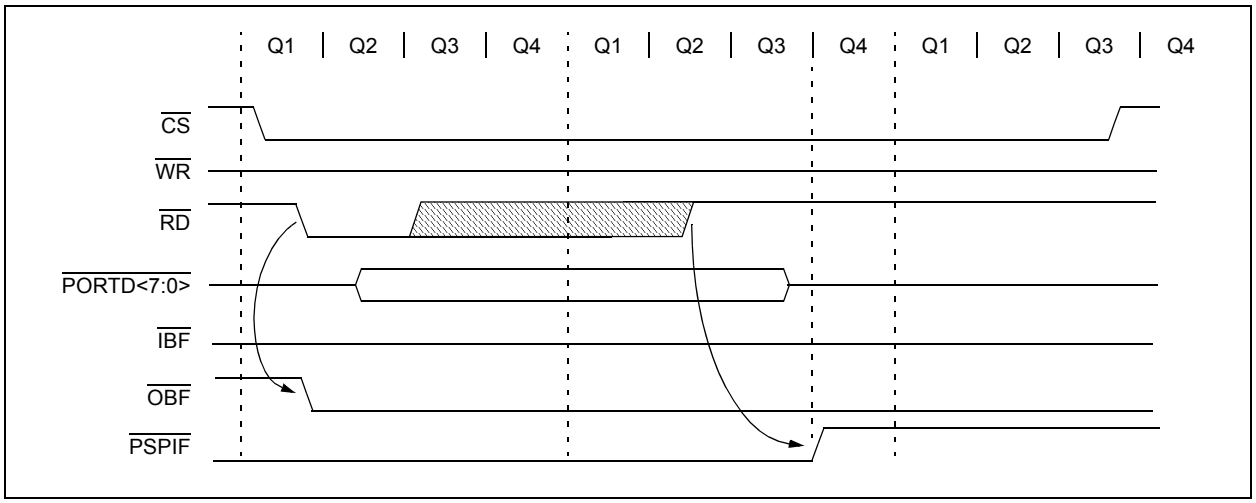


TABLE 11-21: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	60
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	60
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	60
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	60
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	60
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	60
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	59
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	—	TMR2IP	TMR1IP	59

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

12.0 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 12-1) 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 12-1. Figure 12-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 12-1: 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 input edge
0 = Internal clock (FOSC/4)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is not assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS<2:0>**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

12.1 Timer0 Operation

Timer0 can operate as either a timer or a counter. The mode is selected with the T0CS bit (T0CON<5>). In Timer mode (T0CS = 0), the module increments on every clock by default unless a different prescaler value is selected (see **Section 12.3 "Prescaler"**). If the TMR0 register is written to, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

The Counter mode is selected by setting the T0CS bit (= 1). In this mode, Timer0 increments either on every rising or falling edge of pin 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.

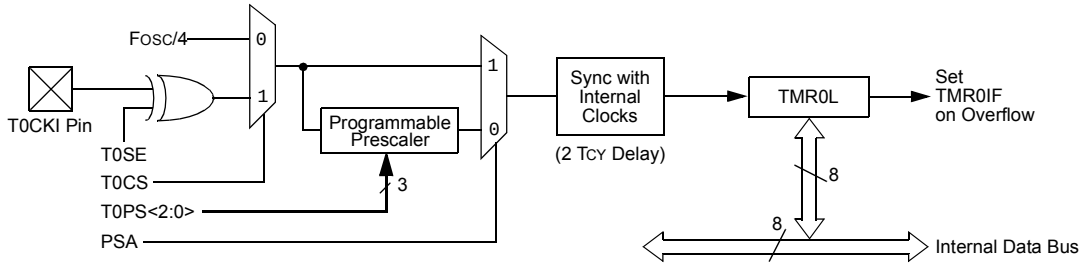
An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the internal phase clock (TOSC). There is a delay between synchronization and the onset of incrementing the timer/counter.

12.2 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 12-2). 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.

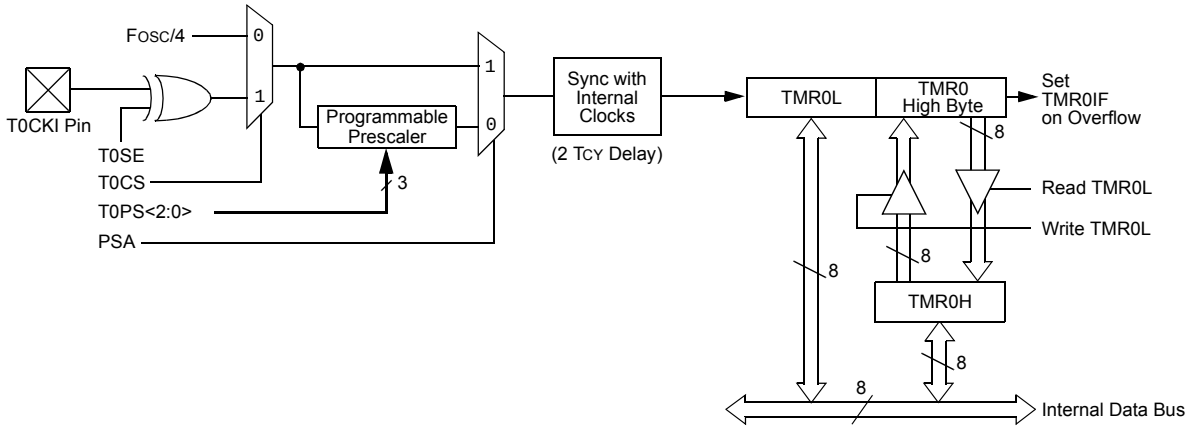
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

FIGURE 12-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 12-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

12.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable. Its value is set by the PSA and T0PS<2:0> bits (T0CON<3:0>) which determine the prescaler assignment and prescale ratio.

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

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

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

12.3.1 SWITCHING PRESCALER ASSIGNMENT

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

12.4 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 12-1: 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								58
TMR0H	Timer0 Register High Byte								58
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	58
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	60

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

Note 1: RA<7:6> and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as ‘0’.

NOTES:

13.0 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 CCPx Special Event Trigger
- Device clock status flag (T1RUN)

A simplified block diagram of the Timer1 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 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 13-1). 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 13-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

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 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
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 the RC0/T1OSO/T13CKI pin (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

13.1 Timer1 Operation

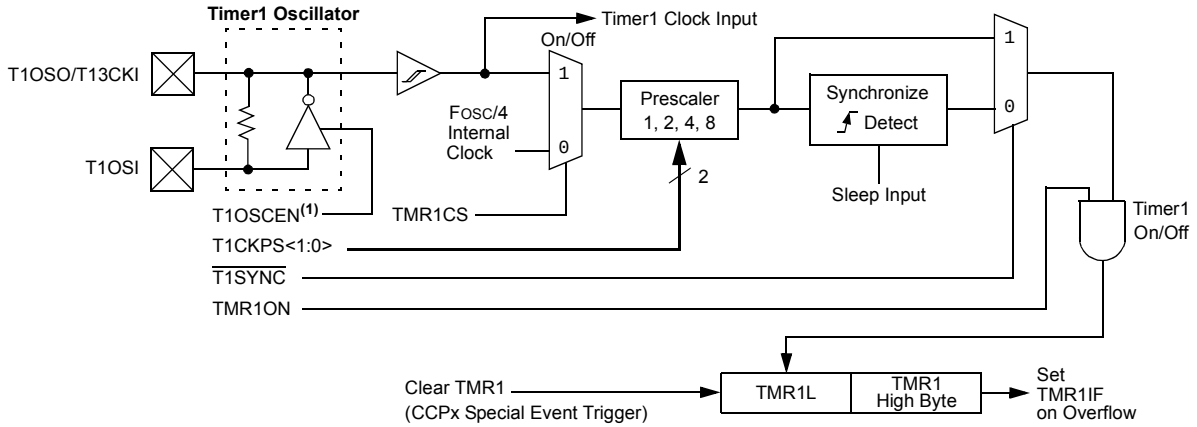
Timer1 can operate in one of these modes:

- 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 (FOSC/4). When the bit is set, Timer1 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

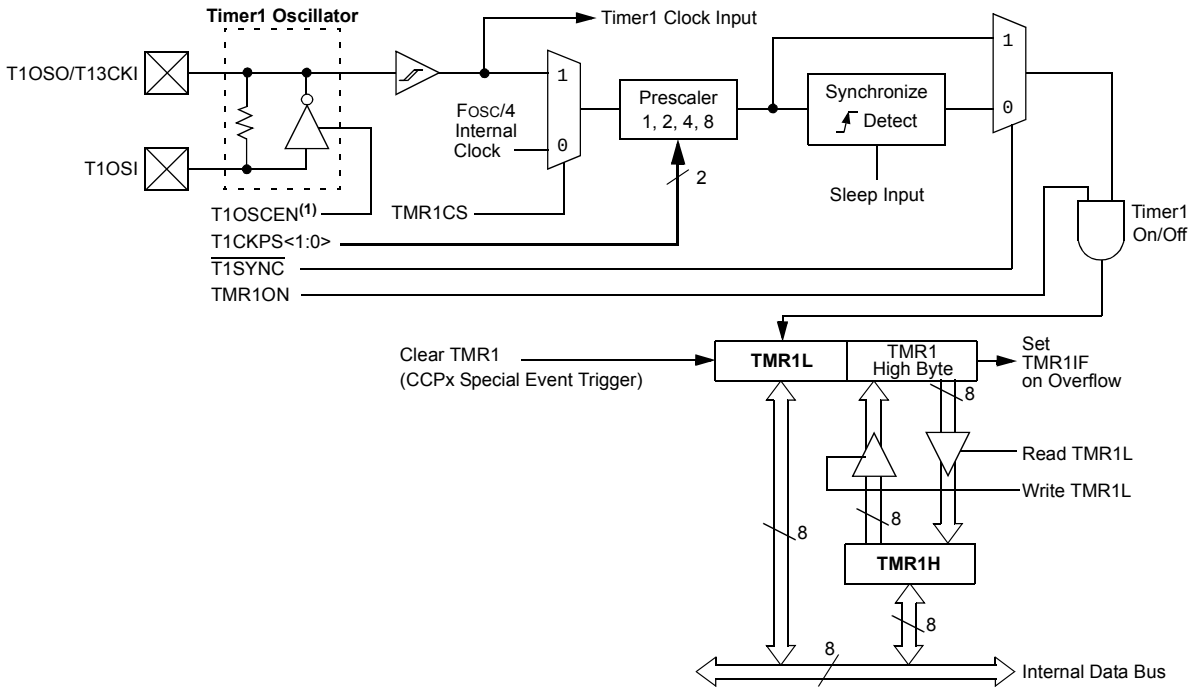
When Timer1 is enabled, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 13-1: TIMER1 BLOCK DIAGRAM (8-BIT MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 13-2: TIMER1 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

13.2 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 13-2). 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.

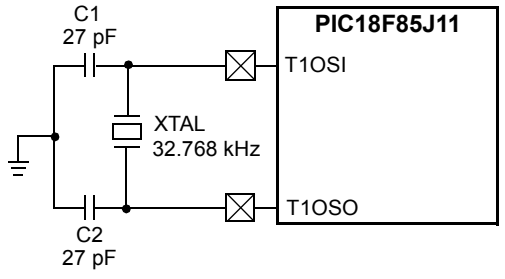
The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

13.3 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 13-3. Table 13-1 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 13-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the Notes with Table 13-1 for additional information about capacitor selection.

TABLE 13-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR(2,3,4)

Oscillator Type	Freq.	C1	C2
LP	32.768 kHz	27 pF(1)	27 pF(1)

Note 1: Microchip suggests these values as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator but also increases the start-up time.

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

4: Capacitor values are for design guidance only.

13.3.1 USING TIMER1 AS A CLOCK SOURCE

The Timer1 oscillator is also available as a clock source in power-managed modes. By setting the System Clock Select bits, SCS<1:0> (OSCCON<1:0>), to '01', the device switches to SEC_RUN mode; both the CPU and peripherals are clocked from the Timer1 oscillator. If the IDLEN bit (OSCCON<7>) is cleared and a SLEEP instruction is executed, the device enters SEC_IDLE mode. Additional details are available in **Section 4.0 "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 Fail-Safe 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.

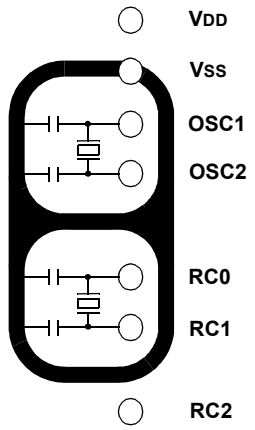
13.3.2 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

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

The oscillator circuit, shown in Figure 13-3, 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 13-4, may be helpful when used on a single-sided PCB or in addition to a ground plane.

FIGURE 13-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING



Note: Not drawn to scale.

13.4 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>).

13.5 Resetting Timer1 Using the CCPx Special Event Trigger

If CCP1 or CCP2 is configured to use Timer1 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3. The trigger from CCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 16.3.4 "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 CCPRxH:CCPRxL 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 CCPx module will not set the TMR1IF interrupt flag bit (PIR1<0>).

13.6 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in **Section 13.3 "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 13-1, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow triggers the interrupt and calls the routine which increments the seconds counter by one. Additional counters for minutes and hours are incremented as the previous counter overflows.

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

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

EXAMPLE 13-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```
RTCinit
            MOVLW   80h             ; Preload TMR1 register pair
            MOVWF   TMR1H           ; for 1 second overflow
            CLRF    TMR1L
            MOVLW   b'00001111'     ; Configure for external clock,
            MOVWF   T1CON           ; Asynchronous operation, external oscillator
            CLRF    secs            ; Initialize timekeeping registers
            CLRF    mins            ;
            MOVLW   .12
            MOVWF   hours
            BSF     PIE1, TMR1IE    ; Enable Timer1 interrupt
            RETURN
RTCisr
            BSF     TMR1H, 7        ; Preload for 1 sec overflow
            BCF     PIR1, TMR1IF    ; Clear interrupt flag
            INCF    secs, F         ; Increment seconds
            MOVLW   .59             ; 60 seconds elapsed?
            CPFSGT  secs
            RETURN                  ; No, done
            CLRF    secs            ; Clear seconds
            INCF    mins, F         ; Increment minutes
            MOVLW   .59             ; 60 minutes elapsed?
            CPFSGT  mins
            RETURN                  ; No, done
            CLRF    mins            ; clear minutes
            INCF    hours, F        ; Increment hours
            MOVLW   .23             ; 24 hours elapsed?
            CPFSGT  hours
            RETURN                  ; No, done
            CLRF    hours           ; Reset hours
            RETURN                  ; Done
```

TABLE 13-2: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
TMR1L	Timer1 Register Low Byte								58
TMR1H	Timer1 Register High Byte								58
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	58

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

NOTES:

14.0 TIMER2 MODULE

The Timer2 module incorporates the following features:

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

The module is controlled through the T2CON register (Register 14-1) 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 14-1.

14.1 Timer2 Operation

In normal operation, TMR2 is incremented from 00h on each clock (FOSC/4). A 4-bit counter/prescaler on the clock input gives direct input, divide-by-4 and divide-by-16 prescale options. These are selected by the prescaler control bits, T2CKPS<1:0> (T2CON<1:0>). The value of TMR2 is compared to that of the Period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/postscaler (see **Section 14.2 "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, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 14-1: 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 **T2OUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

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

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

14.2 Timer2 Interrupt

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

A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, T2OUTPS<3:0> (T2CON<6:3>).

14.3 Timer2 Output

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

Timer2 can be optionally used as the shift clock source for the MSSP module operating in SPI mode. Additional information is provided in **Section 17.0 "Master Synchronous Serial Port (MSSP) Module"**.

FIGURE 14-1: TIMER2 BLOCK DIAGRAM

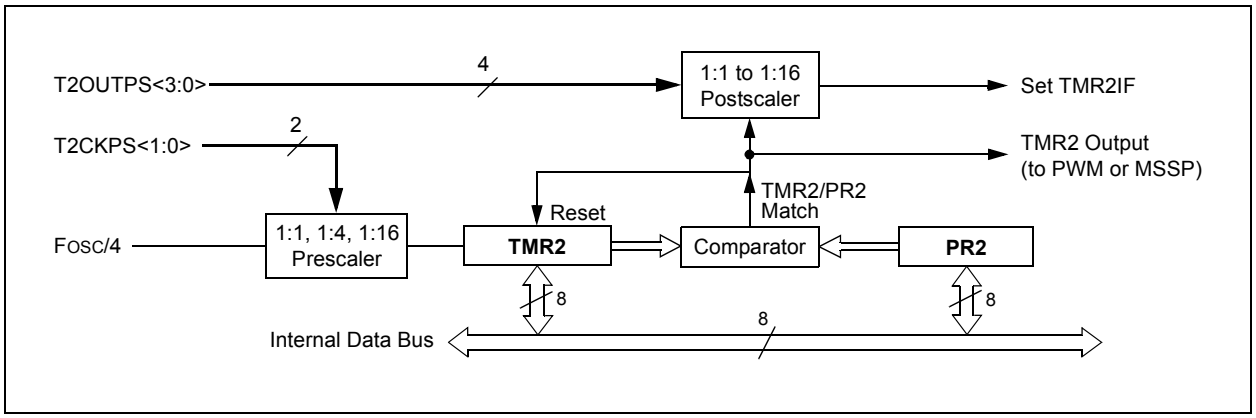


TABLE 14-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
TMR2	Timer2 Register								58
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	58
PR2	Timer2 Period Register								58

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

15.0 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 CCPx Special Event Trigger

A simplified block diagram of the Timer3 module is shown in Figure 15-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 15-2.

The Timer3 module is controlled through the T3CON register (Register 15-1). It also selects the clock source options for the CCP modules. See **Section 16.2.2 "Timer1/Timer3 Mode Selection"** for more information.

REGISTER 15-1: T3CON: TIMER3 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

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 Timer3 in one 16-bit operation
0 = Enables register read/write of Timer3 in two 8-bit operations

bit 6,3 **T3CCP<2:1>:** Timer3 and Timer1 to CCPx Enable bits
1x = Timer3 is the capture/compare clock source for the CCP modules
01 = Timer3 is the capture/compare clock source for CCP2;
Timer1 is the capture/compare clock source for CCP1
00 = Timer1 is the capture/compare clock source for the CCP modules

bit 5-4 **T3CKPS<1:0>**: Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 2 **$\overline{\text{T3SYNC}}$:** 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

15.1 Timer3 Operation

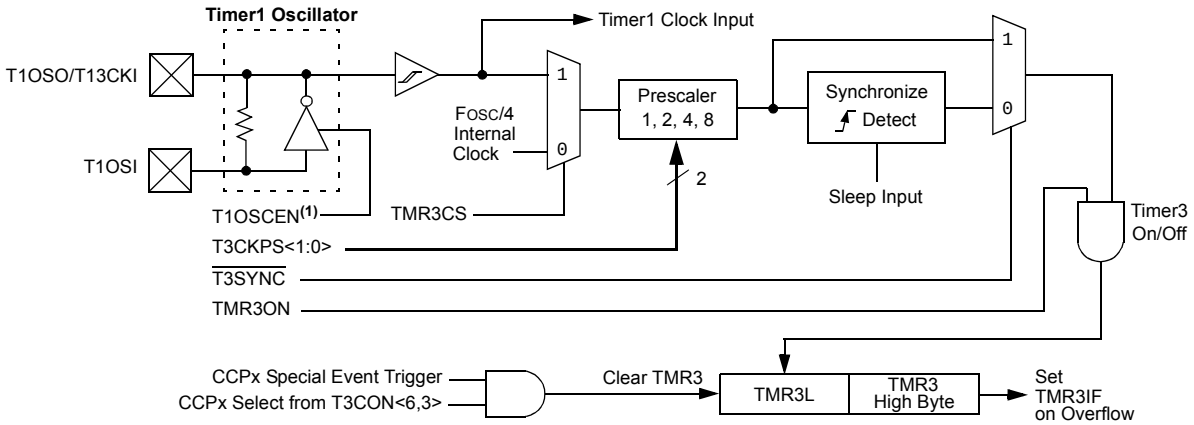
Timer3 can operate in one of three modes:

- 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 (FOSC/4). When the bit is set, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

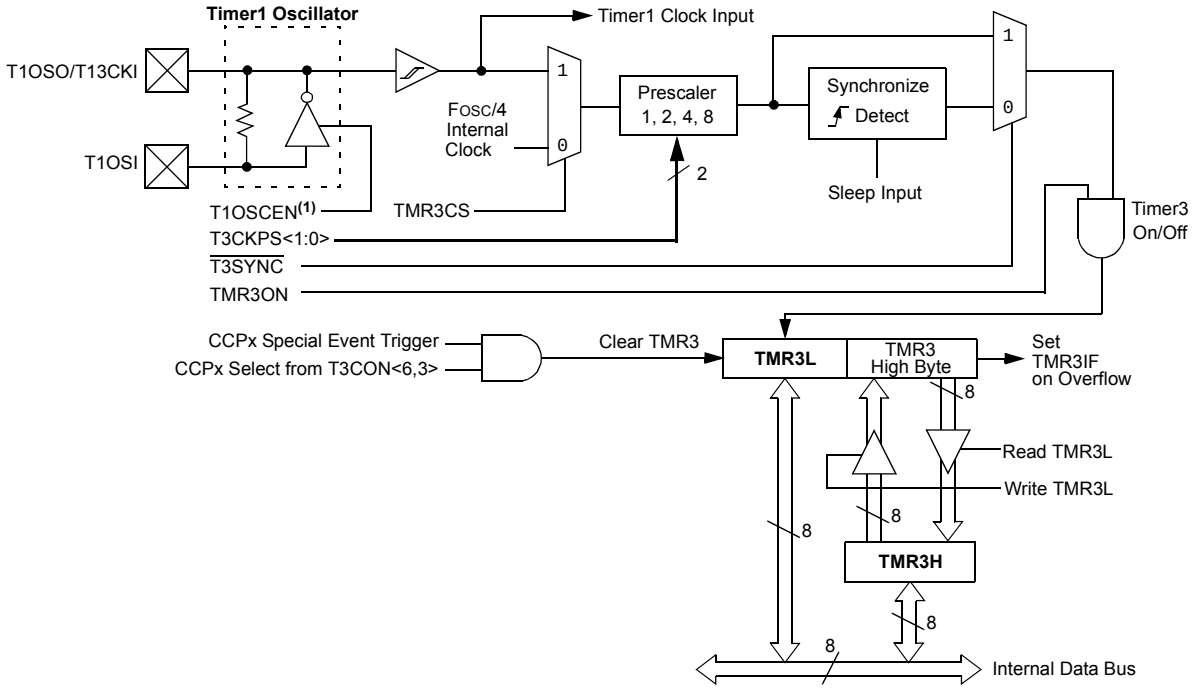
As with Timer1, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs when the Timer1 oscillator is enabled. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 15-1: TIMER3 BLOCK DIAGRAM (8-BIT MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 15-2: TIMER3 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

15.2 Timer3 16-Bit Read/Write Mode

Timer3 can be configured for 16-bit reads and writes (see Figure 15-2). 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.

15.3 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 13.0 "Timer1 Module"**.

15.4 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>).

15.5 Resetting Timer3 Using the CCPx Special Event Trigger

If CCP1 or CCP2 is configured to use Timer3 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3. The trigger from CCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 16.3.4 "Special Event Trigger"** for more information).

The module must be configured as either a timer or synchronous counter to take advantage of this feature. When used this way, the CCPRxH:CCPRxL register pair effectively becomes a Period register for Timer3.

If Timer3 is running in Asynchronous Counter mode, the Reset operation may not work.

In the event that a write to Timer3 coincides with a Special Event Trigger from a CCPx module, the write will take precedence.

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

TABLE 15-1: 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	57
PIR2	OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—	59
PIE2	OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—	59
IPR2	OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—	59
TMR3L	Timer3 Register Low Byte								59
TMR3H	Timer3 Register High Byte								59
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	58
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	59

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

NOTES:

16.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F85J11 family 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 16-1: CCPxCON: CCPx CONTROL REGISTER (CCP1, CCP2 MODULES)

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 **DCxB<1:0>**: 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 (DCx9:DCx2) of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>**: 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: Special Event Trigger; reset timer; start A/D conversion on CCPx match (CCPxIF bit is set)(1)
11xx = PWM mode

Note 1: CCPxM<3:0> = 1011 will only reset the timer and not start an A/D conversion on CCPx match.

16.1 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.

16.1.1 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 mode, while Timer2 is available for modules in PWM mode.

TABLE 16-1: CCPx MODE – TIMER RESOURCE

CCPx 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 CCPx enable bits in the T3CON register (Register 15-1). 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 16-2.

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 16-1.

16.1.2 OPEN-DRAIN OUTPUT OPTION

When operating in Output mode (i.e., in Compare or PWM modes), the drivers for the CCPx pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor and allows the output to communicate with external circuits without the need for additional level shifters.

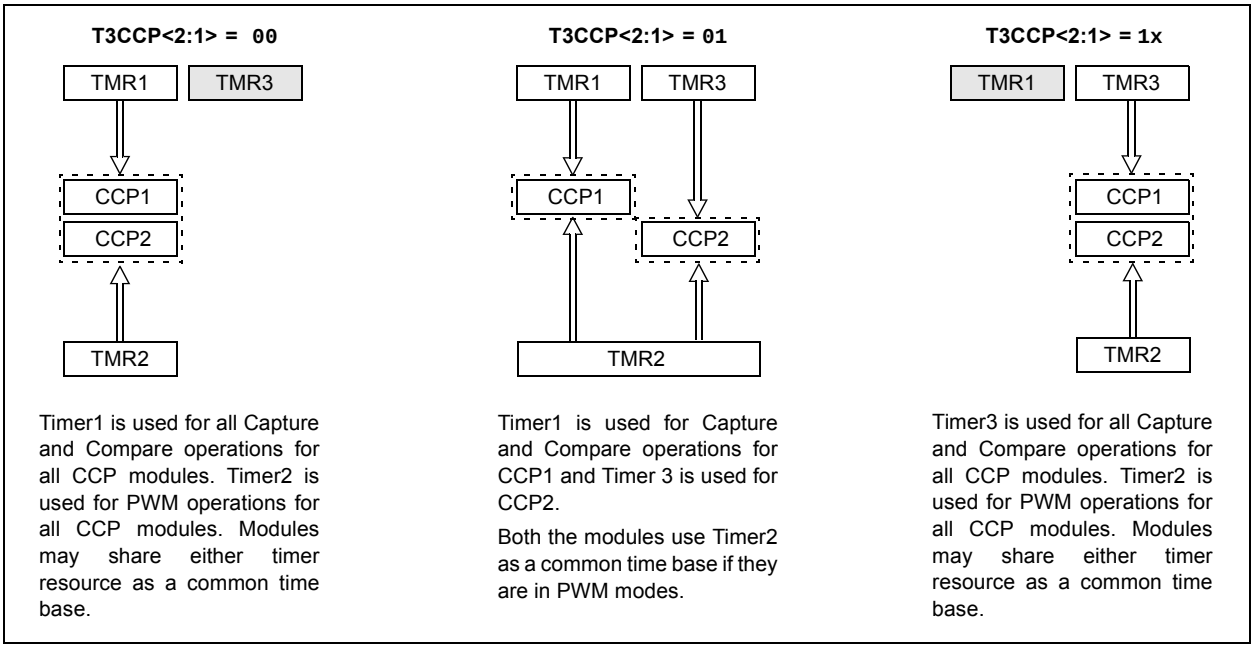
The open-drain output option is controlled by the CCP2OD and CCP1OD bits (TRISG<6:5>). Setting the appropriate bit configures the pin for the corresponding module for open-drain operation.

16.1.3 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 16-1: CCPx AND TIMER INTERCONNECT CONFIGURATIONS



Timer1 is used for all Capture and Compare operations for all CCP modules. Timer2 is used for PWM operations for all CCP modules. Modules may share either timer resource as a common time base.

Timer1 is used for Capture and Compare operations for CCP1 and Timer 3 is used for CCP2.

Both the modules use Timer2 as a common time base if they are in PWM modes.

Timer3 is used for all Capture and Compare operations for all CCP modules. Timer2 is used for PWM operations for all CCP modules. Modules may share either timer resource as a common time base.

TABLE 16-2: 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 module.
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).

16.2 Capture Mode

In Capture mode, the CCPR2H:CCPR2L register pair captures the 16-bit value of the TMR1 or TMR3 register when an event occurs on the CCP2 pin (RB3, 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, CCP2M<3:0> (CCP2CON<3:0>). When a capture is made, the interrupt request flag bit, CCP2IF (PIR3<2>), 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.

16.2.1 CCPx PIN CONFIGURATION

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

Note:	If RB3/INT3/CCP2, RC1/T1OSI/CCP2 or RE7/CCP2 is configured as an output, a write to the port can cause a capture condition.

16.2.2 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 16.1.1 "CCP Modules and Timer Resources"**).

16.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCP2IE bit (PIE3<2>) clear to avoid false interrupts and should clear the flag bit, CCP2IF, following any such change in operating mode.

16.2.4 CCP PRESCALER

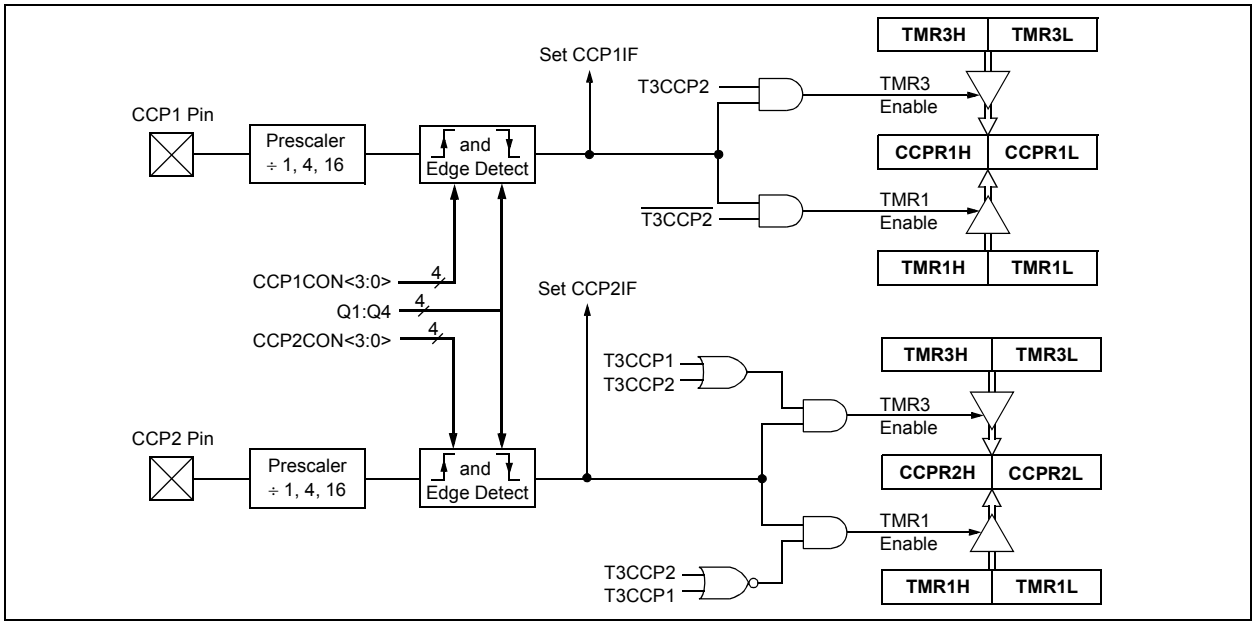
There are four prescaler settings in Capture mode. They are specified as part of the operating mode selected by the mode select bits (CCP2M<3:0>). Whenever the CCP2 module is turned off, or the CCP2 module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear 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 16-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

EXAMPLE 16-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF   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 16-2: CAPTURE MODE OPERATION BLOCK DIAGRAM



16.3 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)
- remains 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 (CCP2M<3:0>). At the same time, the interrupt flag bit, CCP2IF, is set.

16.3.1 CCPx PIN CONFIGURATION

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

Note:	Clearing the CCP2CON register will force the RB3, RC1 or RE7 compare output latch (depending on device configuration) to the default low level. This is not the PORTB, PORTC or PORTE I/O data latch.

16.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode, or Synchronized Counter mode, if the CCPx module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

16.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCP2M<3:0> = 1010), the CCP2 pin is not affected. Only a CCP2 interrupt is generated, if enabled, and the CCP2IE bit is set.

16.3.4 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 (CCP2M<3:0> = 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 16-3: COMPARE MODE OPERATION BLOCK DIAGRAM

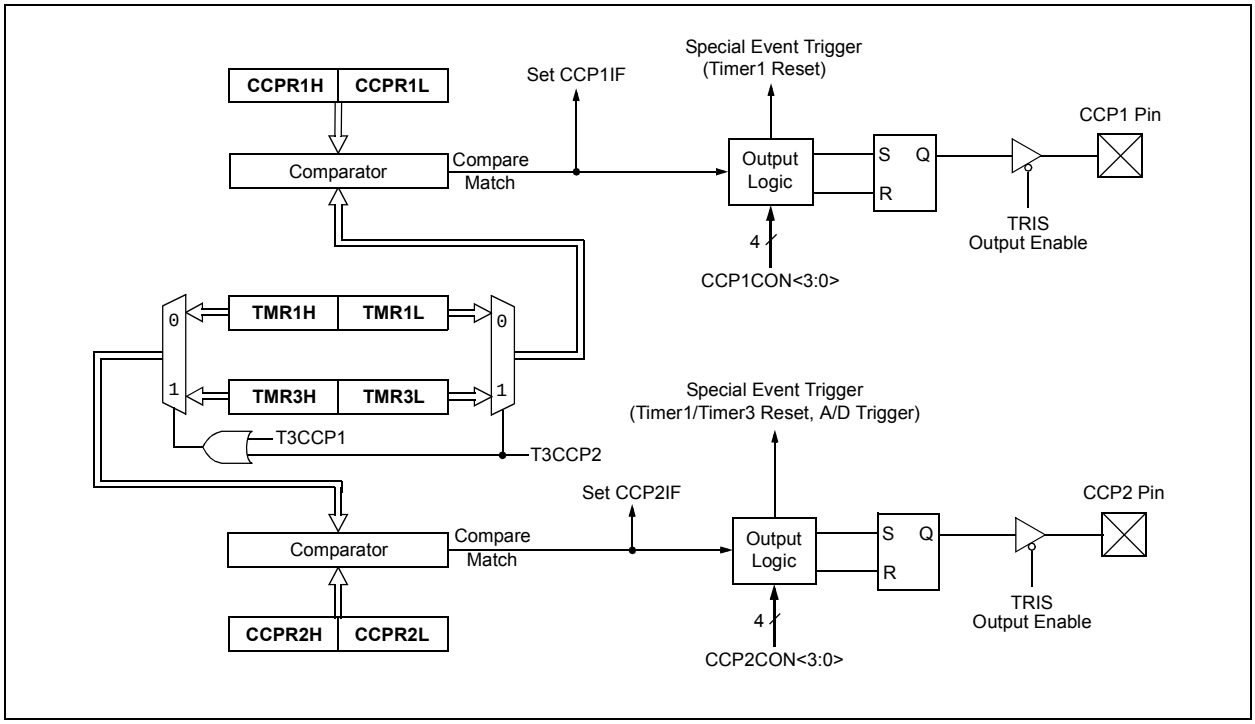


TABLE 16-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

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	57
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	58
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—	59
PIE2	OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—	59
IPR2	OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—	59
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	—	TRISE1	TRISE0	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60
TMR1L	Timer1 Register Low Byte								58
TMR1H	Timer1 Register High Byte								58
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	58
TMR3H	Timer3 Register High Byte								59
TMR3L	Timer3 Register Low Byte								59
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	59
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								60
CCPR1H	Capture/Compare/PWM Register 1 High Byte								60
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	60
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.

16.4 PWM Mode

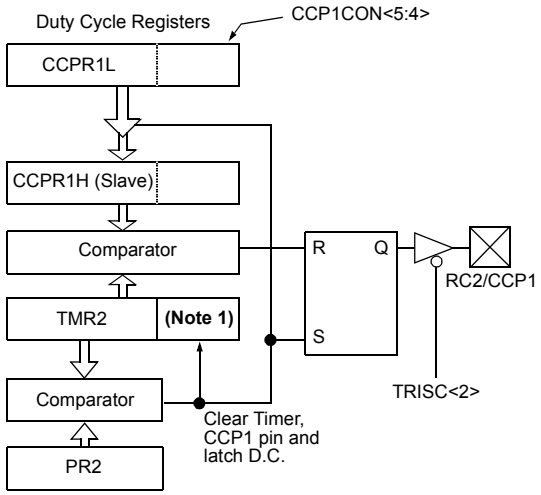
In Pulse-Width Modulation (PWM) mode, the CCP2 pin produces up to a 10-bit resolution PWM output. Since the CCP2 pin is multiplexed with a PORTB, PORTC or PORTE data latch, the appropriate TRIS bit must be cleared to make the CCP2 pin an output.

Note:	Clearing the CCP2CON register will force the RB3, RC1 or RE7 output latch (depending on device configuration) to the default low level. This is not the PORTB, PORTC or PORTE I/O data latch.

Figure 16-4 shows a simplified block diagram of the CCP1 module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 16.4.3 "Setup for PWM Operation"**.

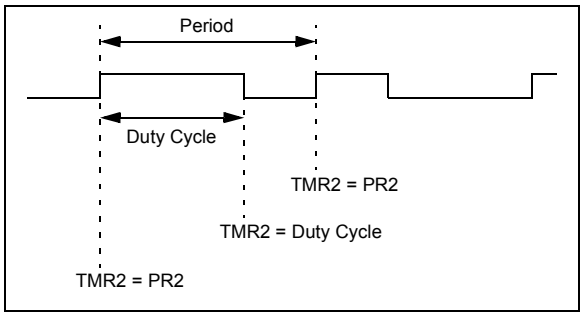
FIGURE 16-4: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: The 8-bit TMR2 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

A PWM output (Figure 16-5) 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 16-5: PWM OUTPUT



16.4.1 PWM PERIOD

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

EQUATION 16-1:

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

PWM frequency is defined as 1/[PWM period].

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

- TMR2 is cleared
- The 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 14.0 "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.

16.4.2 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> contains 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 16-2:

$$\text{PWM Duty Cycle} = (\text{CCPR2L:CCP2CON<5:4>}) \bullet T_{OSC} \bullet (\text{TMR2 Prescale Value})$$

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 16-3:

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

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

TABLE 16-4: 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

16.4.3 SETUP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 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 16-5: 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	57
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	58
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	—	TRISE1	TRISE0	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60
TMR2	Timer2 Register								58
PR2	Timer2 Period Register								58
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	58
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								60
CCPR1H	Capture/Compare/PWM Register 1 High Byte								60
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	60
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.

NOTES:

17.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

17.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D Converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I^2C™)
  - Full Master mode
  - Slave mode (with general address call)

The I^2C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

17.2 Control Registers

The MSSP module has three associated control registers. These include a status register (SSPSTAT) and two control registers (SSPCON1 and SSPCON2). The use of these registers and their individual bits differ significantly depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections.

17.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO) – RC5/SDO
- Serial Data In (SDI) – RC4/SDI/SDA
- Serial Clock (SCK) – RC3/SCK/SCL

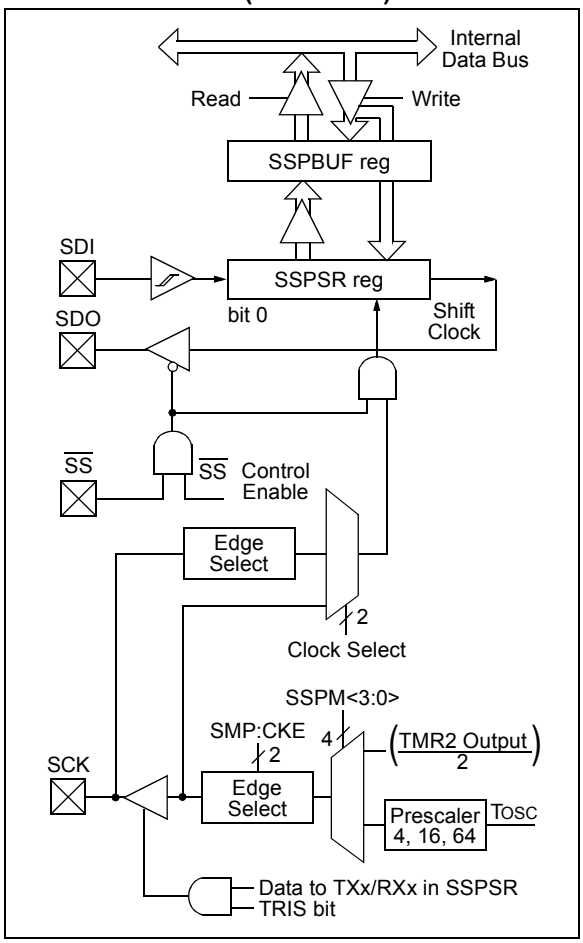
Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) – RF7/AN5/$\overline{SS}$

Figure 17-1 shows the block diagram of the MSSP module when operating in SPI mode.

Note:	Disabling the MSSP module by clearing the SSPEN (SSPCON1<5>) bit may not reset the module. It is recommended to clear the SSPSTAT, SSPCON1 and SSPCON2 registers and select the mode prior to setting the SSPEN bit to enable the MSSP module.

FIGURE 17-1: MSSP BLOCK DIAGRAM (SPI MODE)



17.3.1 REGISTERS

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

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer 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 two bits of the SSPSTAT are read/write.

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

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

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

REGISTER 17-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R0	R-0
SMP	CKE(1)	D/$\overline{A}$	P	S	R/$\overline{W}$	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 is sampled at the end of data output time
0 = Input data is sampled at the middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Select bit(1)
1 = Transmit occurs on transition from active to Idle clock state
0 = Transmit occurs on transition from Idle to active clock state

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit
Used in I²C™ mode only.

bit 4 **P:** Stop bit
Used in I²C mode only. This bit is cleared when the MSSP module is disabled and SSPEN is cleared.

bit 3 **S:** Start bit
Used in I²C mode only.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit
Used in I²C mode only.

bit 1 **UA:** Update Address bit
Used in I²C mode only.

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
1 = Receive is complete, SSPBUF is full
0 = Receive is not complete, SSPBUF is empty

Note 1: The polarity of the clock state is set by the CKP bit (SSPCON1<4>).

REGISTER 17-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
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(1)
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 the overflow (must be cleared in software).
0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(2)
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 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(3)
0101 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control is disabled, $\overline{SS}$ can be used as an I/O pin
0100 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control is 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, this pin must be properly configured as an input or output.
3: Bit combinations not specifically listed here are either reserved or implemented in I^2C™ mode only.

17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

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

The MSSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then, the Buffer Full detect bit, BF (SSPSTAT<0>), and the MSSP 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 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the SSPSTAT register indicates the various status conditions.

Note:	To avoid lost data in Master mode, a read of the SSPBUF must be performed to clear the Buffer Full (BF) detect bit (SSPSTAT<0>) between each transmission.

EXAMPLE 17-1: LOADING THE SSPBUF (SSPSR) REGISTER

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

17.3.3 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 SSPCON registers and then set the SSPEN bit. This configures the SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed 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
- $\overline{SS}$ 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.

17.3.4 OPEN-DRAIN OUTPUT OPTION

The drivers for the SDO output and SCK clock pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, and allows the output to communicate with external circuits without the need for additional level shifters.

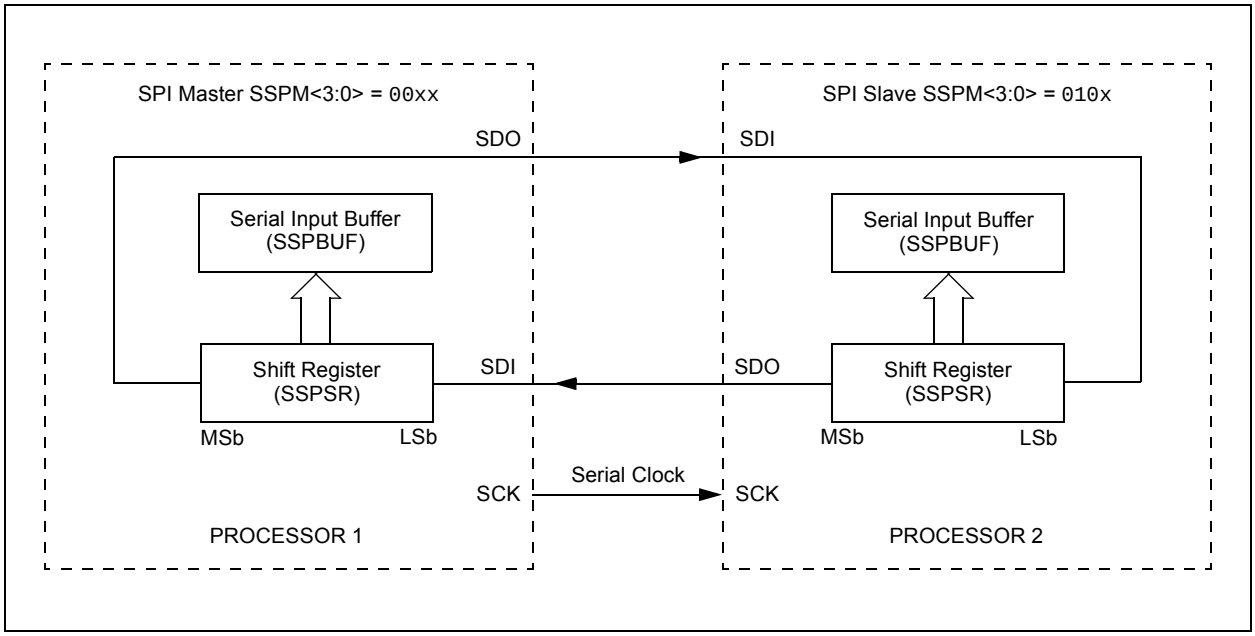
The open-drain output option is controlled by the SPIOD bit (TRISG<7>). Setting the bit configures both pins for open-drain operation.

17.3.5 TYPICAL CONNECTION

Figure 17-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

FIGURE 17-2: SPI MASTER/SLAVE CONNECTION



17.3.6 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 17-2) will broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

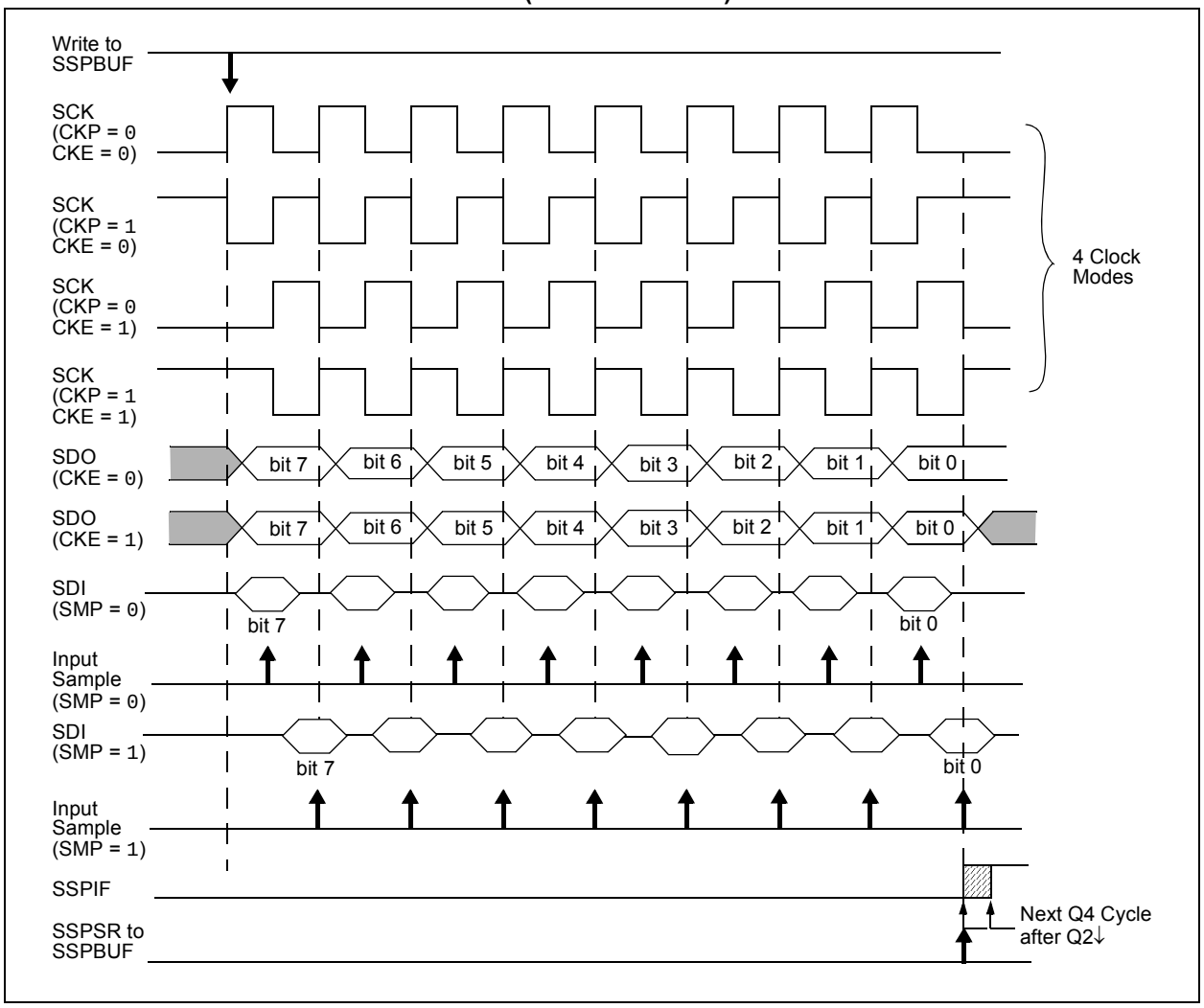
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This, then, would give waveforms for SPI communication as shown in Figure 17-3, Figure 17-5 and Figure 17-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

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

Figure 17-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

FIGURE 17-3: SPI MODE WAVEFORM (MASTER MODE)



17.3.7 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.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCK pin. The Idle state is determined by the CKP bit (SSPCON1<4>).

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.

17.3.8 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<3:0> = 04h). 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 (SSPCON1<3:0> = 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 17-4: SLAVE SYNCHRONIZATION WAVEFORM

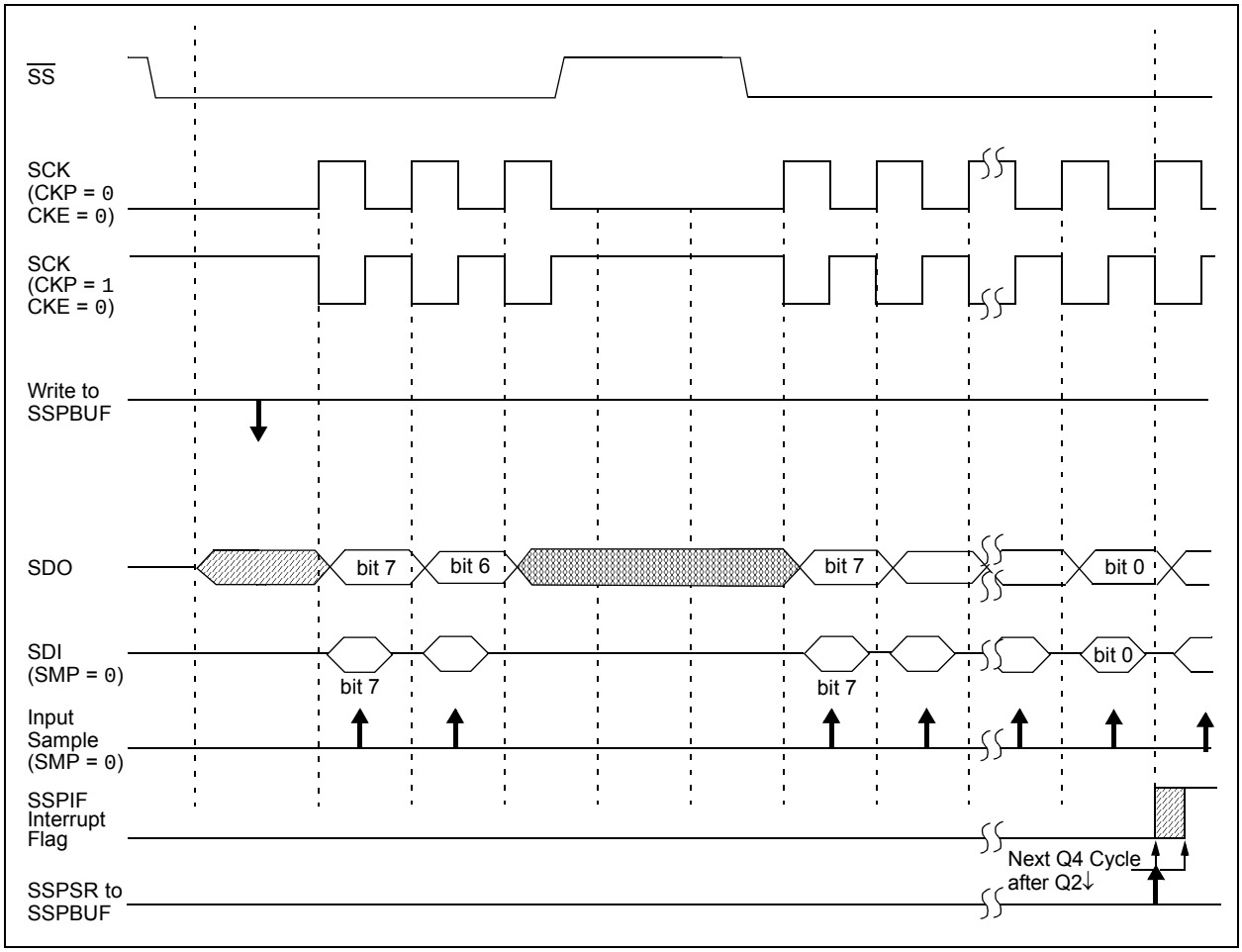


FIGURE 17-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

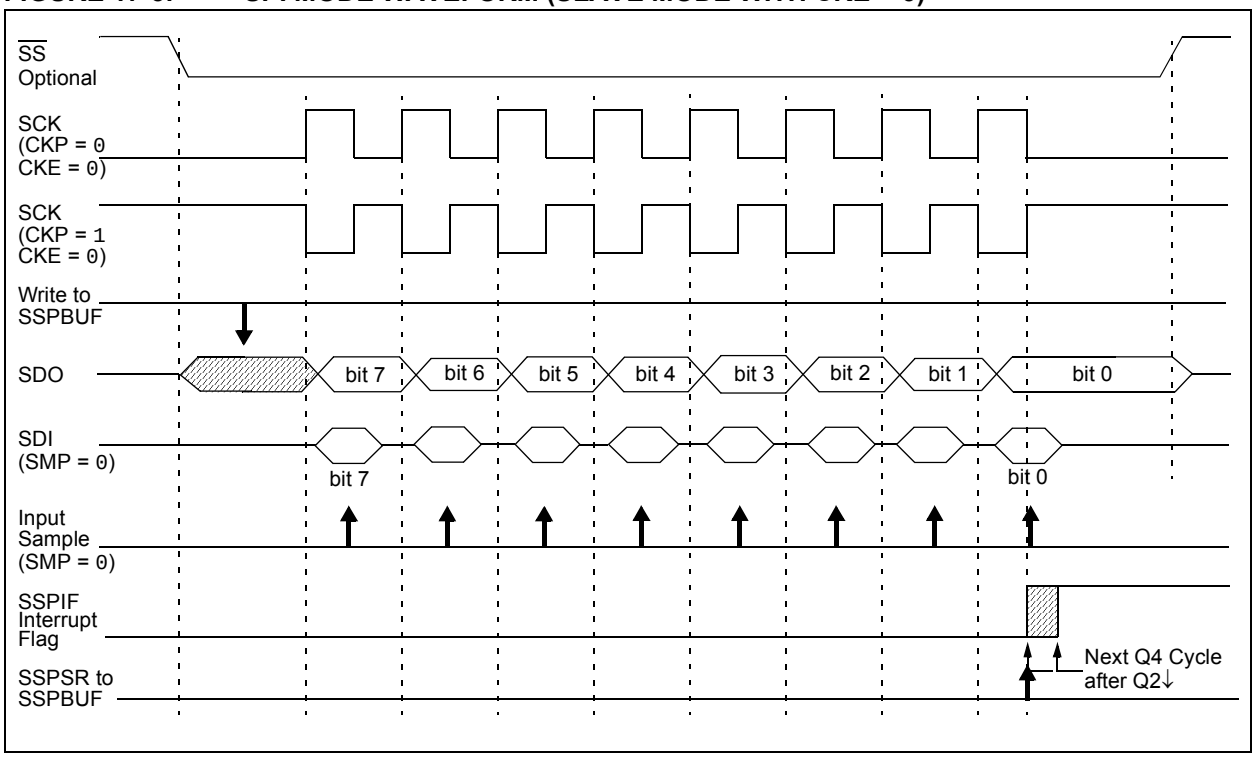
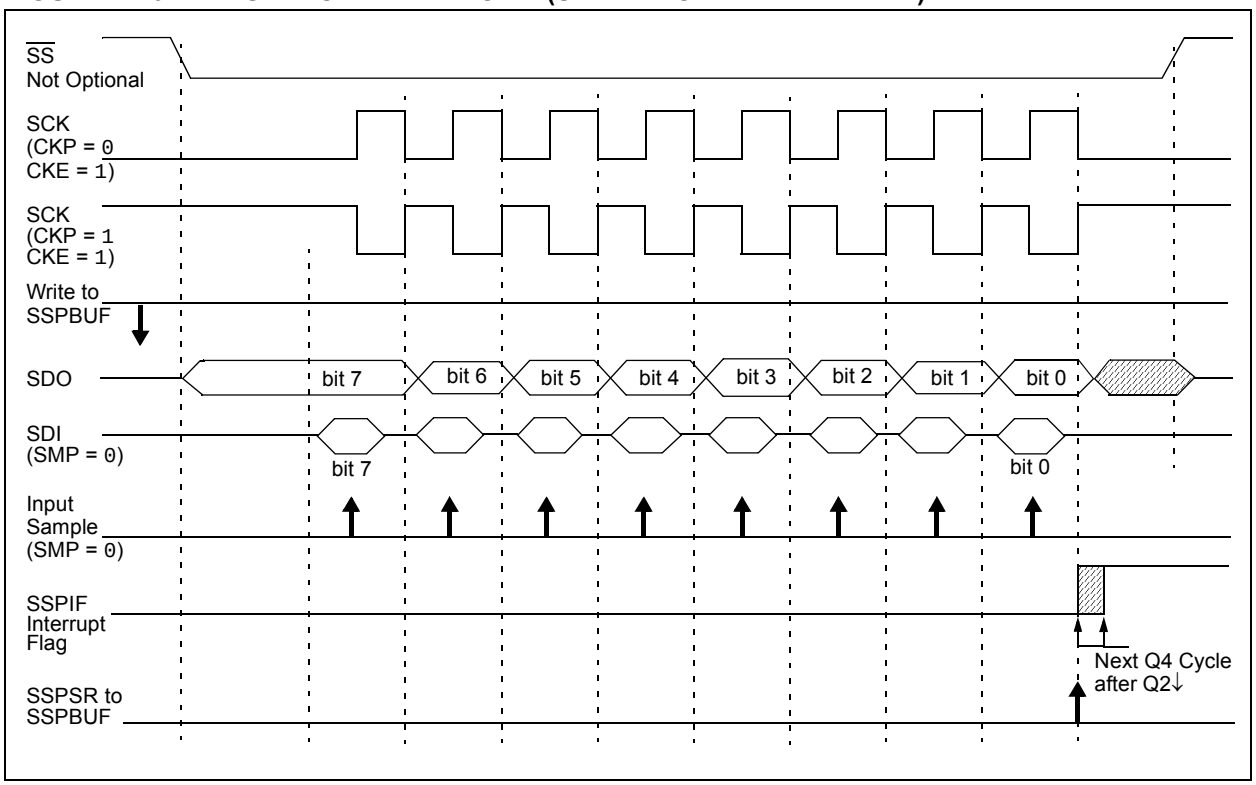


FIGURE 17-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



17.3.9 OPERATION IN POWER-MANAGED MODES

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

In Idle modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTRC source. See **Section 3.3 "Clock Sources and Oscillator Switching"** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the devices 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.

17.3.10 EFFECTS OF A RESET

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

17.3.11 BUS MODE COMPATIBILITY

Table 17-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 17-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also an SMP bit which controls when the data is sampled.

TABLE 17-2: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60
SSPBUF	MSSP Receive Buffer/Transmit Register								58
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	58
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	58

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the MSSP module in SPI mode.

17.4 I²C Mode

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

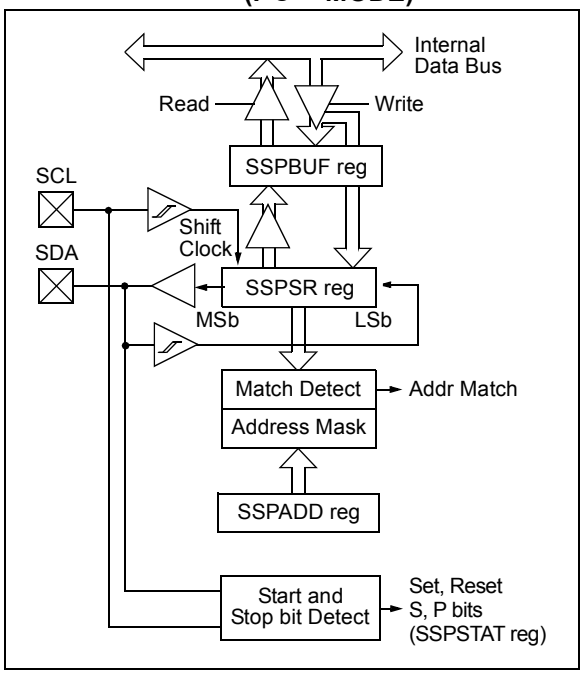
Two pins are used for data transfer:

- Serial Clock (SCL) – RC3/SCK/SCL
- Serial Data (SDA) – RC4/SDI/SDA

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

Note:	Disabling the MSSP module by clearing the SSPEN (SSPCON1<5>) bit may not reset the module. It is recommended to clear the SSPSTAT, SSPCON1 and SSPCON2 registers and select the mode prior to setting the SSPEN bit to enable the MSSP module.

FIGURE 17-7: MSSP BLOCK DIAGRAM (I²C™ MODE)



17.4.1 REGISTERS

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

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON1, SSPCON2 and SSPSTAT are the control and status registers in I²C mode operation. The SSPCON1 and SSPCON2 registers are readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

Many of the bits in SSPCON2 assume different functions, depending on whether the module is operating in Master or Slave mode; SSPCON2<5:2> bits also assume different names in Slave mode. The different aspects of SSPCON2 are shown in Register 17-5 (for Master mode) and Register 17-6 (Slave mode).

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 I²C Slave mode. When the MSSP is configured in Master mode, the lower seven bits of SSPADD act as the Baud Rate Generator reload value.

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

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

REGISTER 17-3: SSPSTAT: MSSP STATUS REGISTER (I²C™ MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$	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:** Slew Rate Control bit
In Master or Slave mode:
1 = Slew rate control is disabled for Standard Speed mode (100 kHz and 1 MHz)
0 = Slew rate control is 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/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit
In Master mode:
Reserved.
In Slave mode:
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit(1)
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last

bit 3 **S:** Start bit(1)
1 = Indicates that a Start bit has been detected last
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit (I²C mode only)
In Slave mode:(2)
1 = Read
0 = Write
In Master mode:(3)
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 = SSPBUF is full
0 = SSPBUF is empty
In Receive mode:
1 = SSPBUF is full (does not include the $\overline{\text{ACK}}$ and Stop bits)
0 = SSPBUF is empty (does not include the $\overline{\text{ACK}}$ and Stop bits)

Note 1: This bit is cleared on Reset and when SSPEN is cleared.
2: This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not $\overline{\text{ACK}}$ bit.
3: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Active mode.

REGISTER 17-4: SSPCON1: MSSP CONTROL REGISTER 1 (I²C™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN(1)	CKP	SSPM3	SSPM2	SSPM1	SSPM0
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 SSPBUF register was attempted while the I²C conditions were not valid for a transmission to be started (must be cleared in software)
0 = No collision

In Slave Transmit mode:
1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

In Receive mode (Master or Slave modes):
This is a "don't care" bit.

bit 6 **SSPOV:** Receive Overflow Indicator bit

In Receive mode:
1 = A byte is received while the SSPBUF register is still holding the previous byte (must be cleared in software)
0 = No overflow

In Transmit mode:
This is a "don't care" bit in Transmit mode.

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(1)

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 = Release clock
0 = Holds clock low (clock stretch); used to ensure data setup time

In Master mode:
Unused in this mode.

bit 3-0 **SSPM<3:0>:** Synchronous Serial Port Mode Select bits

1111 = I²C Slave mode, 10-bit address with Start and Stop bit interrupts enabled
1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled
1011 = I²C Firmware Controlled Master mode (slave Idle)
1000 = I²C Master mode, clock = FOSC/(4 * (SSPADD + 1))
0111 = I²C Slave mode, 10-bit address
0110 = I²C Slave mode, 7-bit address
Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

Note 1: When enabled, the SDA and SCL pins must be configured as inputs.

REGISTER 17-5: SSPCON2: MSSP CONTROL REGISTER 2 (I^2C™ MASTER 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(1)	ACKEN(2)	RCEN(2)	PEN(2)	RSEN(2)	SEN(2)
bit 7							bit 0

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

bit 7 **GCEN:** General Call Enable bit
Unused in Master mode.

bit 6 **ACKSTAT:** Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave

bit 5 **ACKDT:** Acknowledge Data bit (Master Receive mode only)(1)
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit(2)
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 Receive mode only)(2)
1 = Enables Receive mode for I^2C
0 = Receive is Idle

bit 2 **PEN:** Stop Condition Enable bit(2)
1 = Initiate Stop condition on SDA and SCL pins; automatically cleared by hardware.
0 = Stop condition is Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit(2)
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 bit(2)
1 = Initiate Start condition on SDA and SCL pins; automatically cleared by hardware.
0 = Start condition is Idle

Note 1: Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.
2: If the I^2C module is active, these bits may not be set (no spooling) and the SSPBUF may not be written (or writes to the SSPBUF are disabled).

REGISTER 17-6: SSPCON2: MSSP CONTROL REGISTER 2 (I²C™ SLAVE 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	ADMSK5	ADMSK4	ADMSK3	ADMSK2	ADMSK1	SEN(1)
bit 7							bit 0

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

bit 7 **GCEN:** General Call Enable bit
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
Unused in Slave mode.

bit 5-2 **ADMSK<5:2>:** Slave Address Mask Select bits
1 = Masking of corresponding bits of SSPADD enabled
0 = Masking of corresponding bits of SSPADD disabled

bit 1 **ADMSK1:** Slave Address Least Significant bit(s) Mask Select bit
In 7-Bit Address mode:
1 = Masking of SSPADD<1> only is enabled
0 = Masking of SSPADD<1> only is disabled
In 10-Bit Address mode:
1 = Masking of SSPADD<1:0> is enabled
0 = Masking of SSPADD<1:0> is disabled

bit 0 **SEN:** Stretch Enable bit(1)
1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
0 = Clock stretching is disabled

Note 1: If the I²C module is active, this bit may not be set (no spooling) and the SSPBUF may not be written (or writes to the SSPBUF are disabled).

17.4.2 OPERATION

The MSSP module functions are enabled by setting the MSSP Enable bit, SSPEN (SSPCON1<5>).

The SSPCON1 register allows control of the I^2C operation. Four mode selection bits (SSPCON1<3:0>) allow one of the following I^2C modes to be selected:

- I^2C Master mode, clock = (FOSC/4) x (SSPADD + 1)
- I^2C Slave mode (7-bit address)
- I^2C Slave mode (10-bit address)
- I^2C Slave mode (7-bit address) with Start and Stop bit interrupts enabled
- I^2C Slave mode (10-bit address) with Start and Stop bit interrupts enabled
- I^2C Firmware Controlled Master mode, slave is Idle

Selection of any I^2C mode, with the SSPEN bit set, forces the SCL and SDA pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC or TRISD bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCL and SDA pins.

17.4.3 SLAVE MODE

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

The I^2C Slave mode hardware will always generate an interrupt on an exact address match. In addition, address masking will also allow the hardware to generate an interrupt for more than one address (up to 31 in 7-bit addressing and up to 63 in 10-bit addressing). Through the mode select bits, the user can also choose to interrupt on Start and Stop bits.

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) 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 MSSP 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 is set. The BF bit is cleared by reading the SSPBUF register, while the SSPOV bit is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification, as well as the requirement of the MSSP module, are shown in timing parameter 100 and parameter 101.

17.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPSR register value is loaded into the SSPBUF register.
2. The Buffer Full bit, BF, is set.
3. An $\overline{ACK}$ pulse is generated.
4. The MSSP Interrupt Flag bit, SSPIF, is set (and the 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. The R/$\overline{W}$ bit (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 is as follows, with steps 7 through 9 for the slave-transmitter:

1. Receive the 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 (SSPIF, BF and UA bits are set).
5. Update the SSPADD register with the first (high) byte of address. If match releases the SCL line, this will clear the UA bit.
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 BF bit) and clear flag bit, SSPIF.

17.4.3.2 Address Masking

Masking an address bit causes that bit to become a "don't care". When one address bit is masked, two addresses will be Acknowledged and cause an interrupt. It is possible to mask more than one address bit at a time, which makes it possible to Acknowledge up to 31 addresses in 7-Bit Addressing mode and up to 63 addresses in 10-Bit Addressing mode (see Example 17-2).

The I^2C Slave behaves the same way, whether address masking is used or not. However, when address masking is used, the I^2C slave can Acknowledge multiple addresses and cause interrupts. When this occurs, it is necessary to determine which address caused the interrupt by checking SSPBUF.

In 7-Bit Addressing mode, Address Mask bits, ADMSK<5:1> (SSPCON2<5:1>), mask the corresponding address bits in the SSPADD register. For any ADMSK bits that are set (ADMSK<n> = 1), the corresponding address bit is ignored (SSPADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

In 10-Bit Addressing mode, ADMSK<5:2> bits mask the corresponding address bits in the SSPADD register. In addition, ADMSK1 simultaneously masks the two LSbs of the address (SSPADD<1:0>). For any ADMSK bits that are active (ADMSK<n> = 1), the corresponding address bit is ignored (SSPADD<n> = x). Also note that although in 10-Bit Addressing mode, the upper address bits reuse part of the SSPADD register bits, the address mask bits do not interact with those bits. They only affect the lower address bits.

Note 1:	ADMSK1 masks the two Least Significant bits of the address.
2:	The two Most Significant bits of the address are not affected by address masking.

EXAMPLE 17-2: ADDRESS MASKING EXAMPLES

7-Bit Addressing:

SSPADD<7:1> = A0h (1010000) (SSPADD<0> is assumed to be '0')

ADMSK<5:1> = 00111

Addresses Acknowledged: A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

10-Bit Addressing:

SSPADD<7:0> = A0h (10100000) (the two MSbs of the address are ignored in this example, since they are not affected by masking)

ADMSK<5:1> = 00111

Addresses Acknowledged: A0h, A1h, A2h, A3h, A4h, A5h, A6h, A7h, A8h, A9h, AAh, ABh, ACh, ADh, AEh, AFh

17.4.3.3 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register and the SDA line is held low ($\overline{ACK}$).

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit, BF (SSPSTAT<0>), is set or bit, SSPOV (SSPCON1<6>), is set.

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

If SEN is enabled (SSPCON2<0> = 1), SCK/SCL will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPCON1<4>). See **Section 17.4.4 "Clock Stretching"** for more details.

17.4.3.4 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RC3 is held low, regardless of SEN (see **Section 17.4.4 "Clock Stretching"** for more details). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then, pin RC3 should be enabled by setting bit, CKP (SSPCON1<4>). The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 17-10).

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 must be enabled by setting bit, CKP.

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

FIGURE 17-8: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESSING)

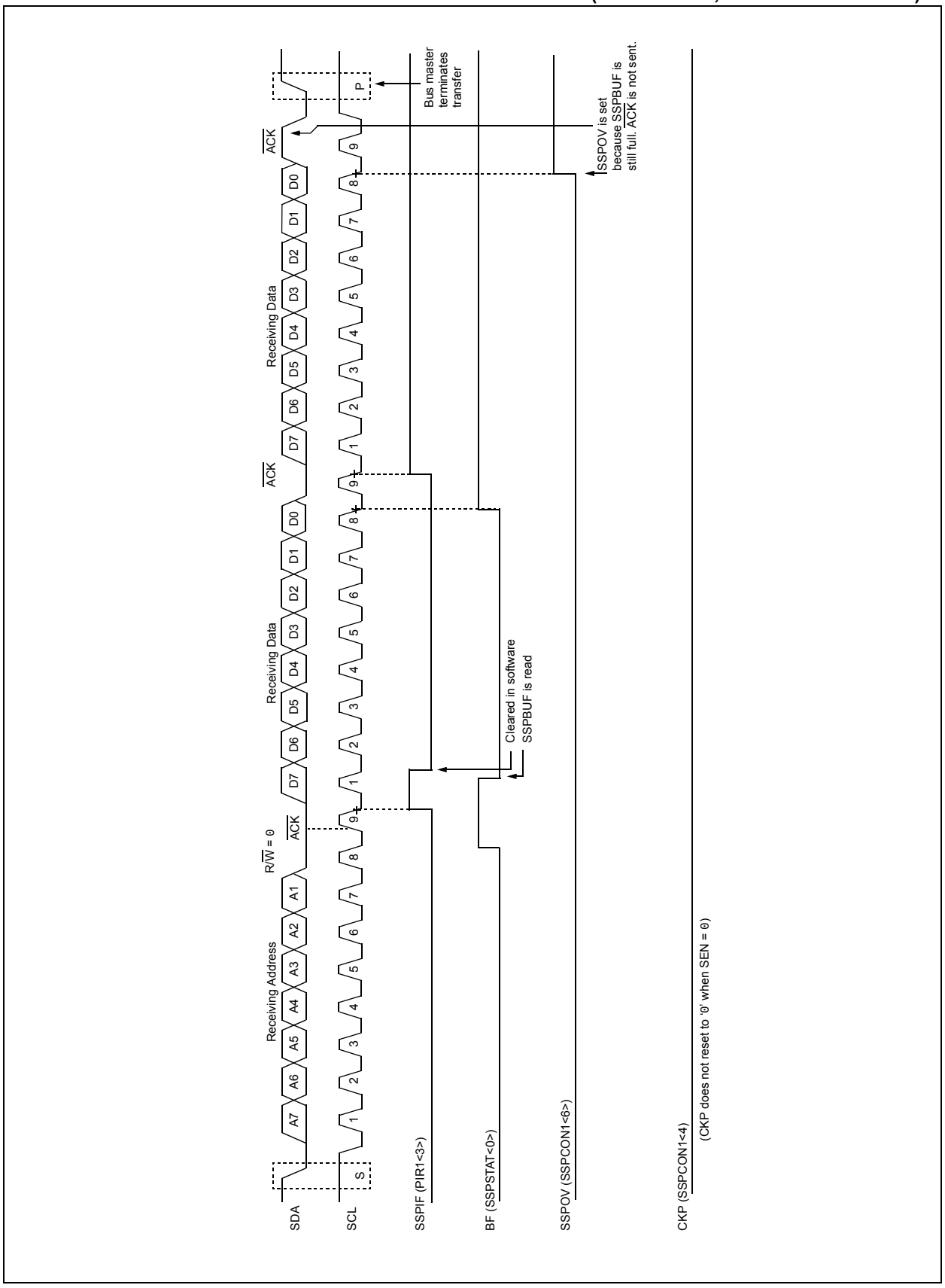
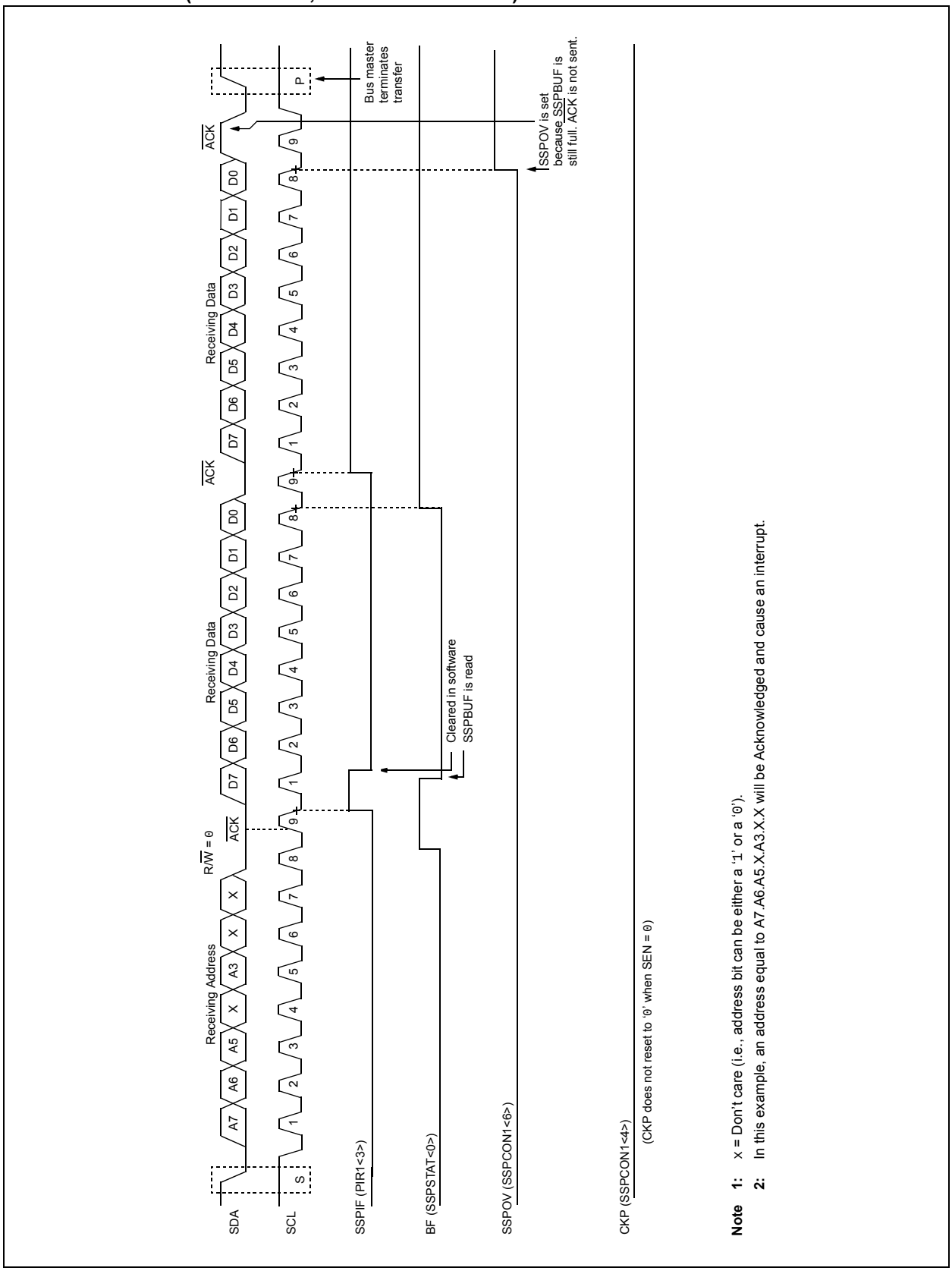


FIGURE 17-9: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01011 (RECEPTION, 7-BIT ADDRESSING)



Note 1: x = Don't care (i.e., address bit can be either a '1' or a '0').

2: In this example, an address equal to A7.A6.A5.X.A3.X.X will be Acknowledged and cause an interrupt.

FIGURE 17-10: I²C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESSING)

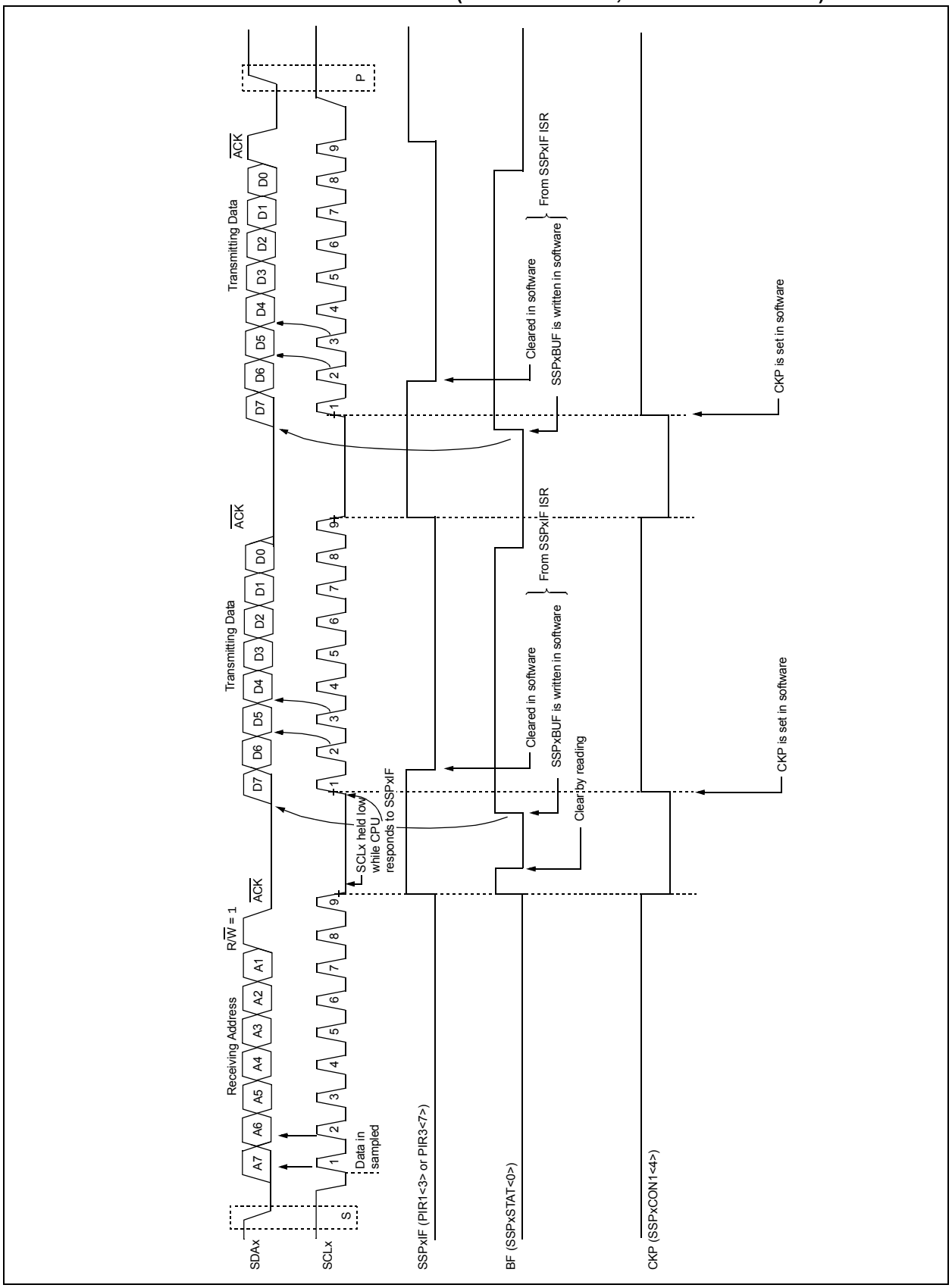


FIGURE 17-11: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESSING)

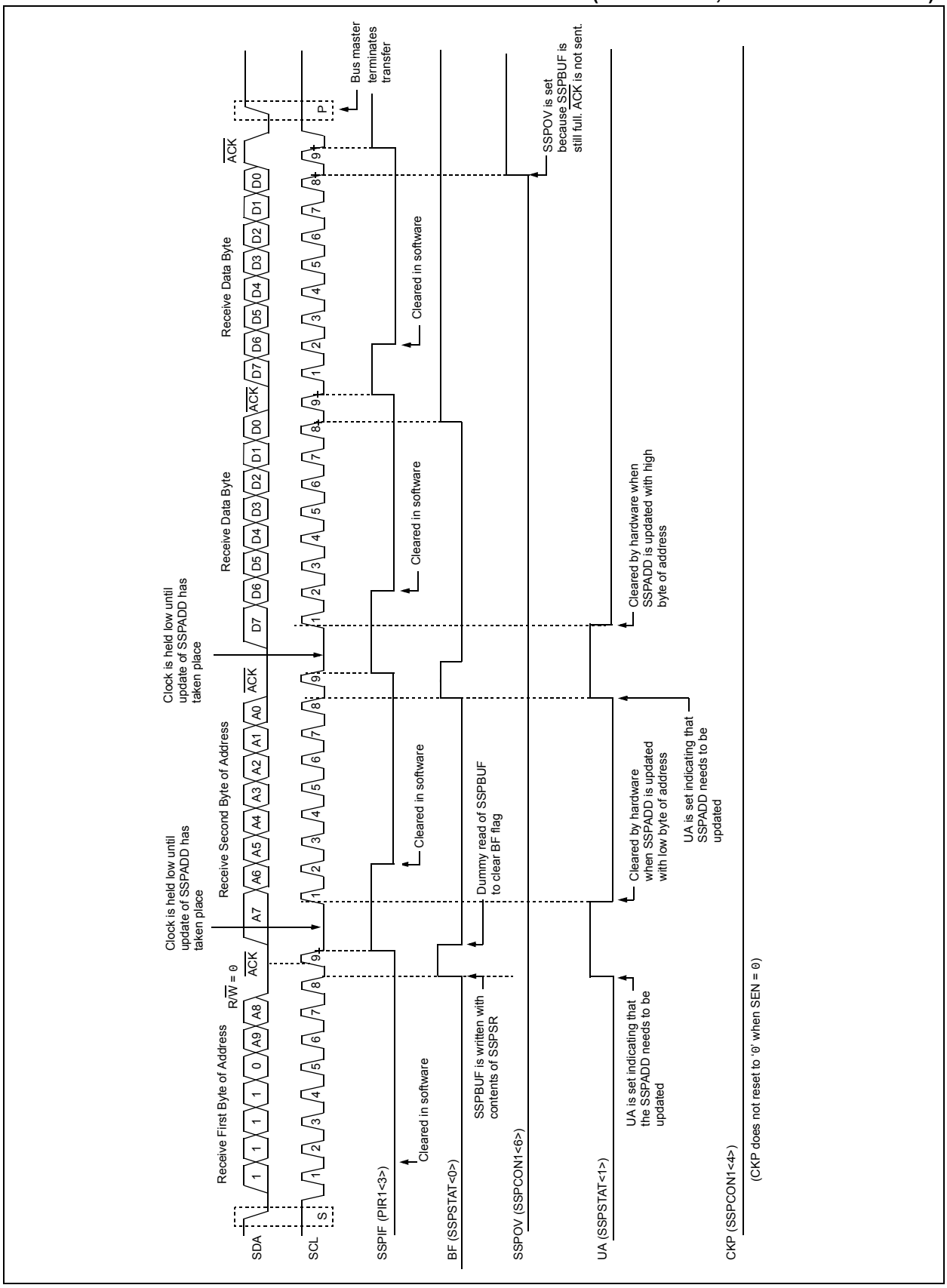
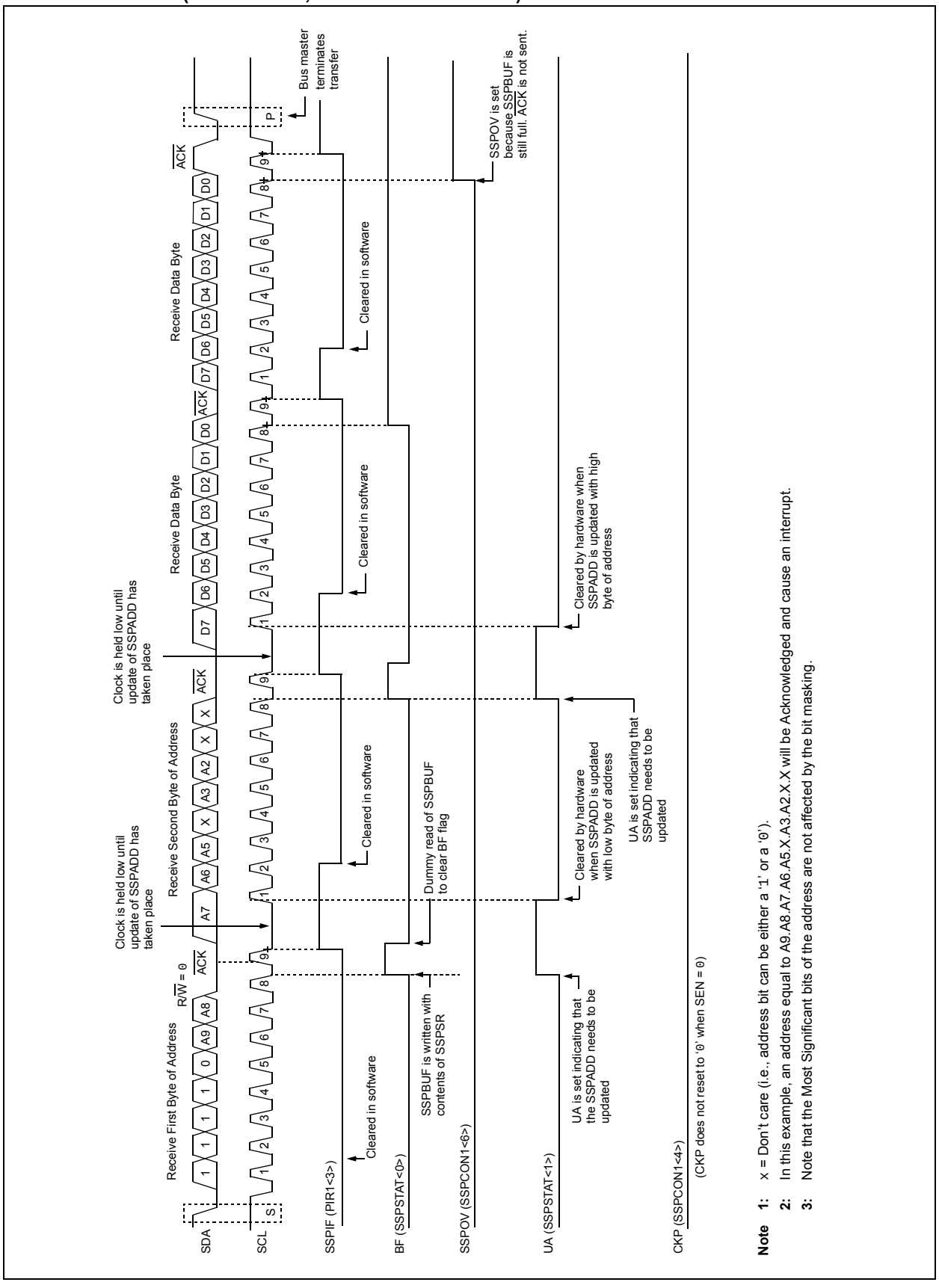


FIGURE 17-12: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01001 (RECEPTION, 10-BIT ADDRESSING)

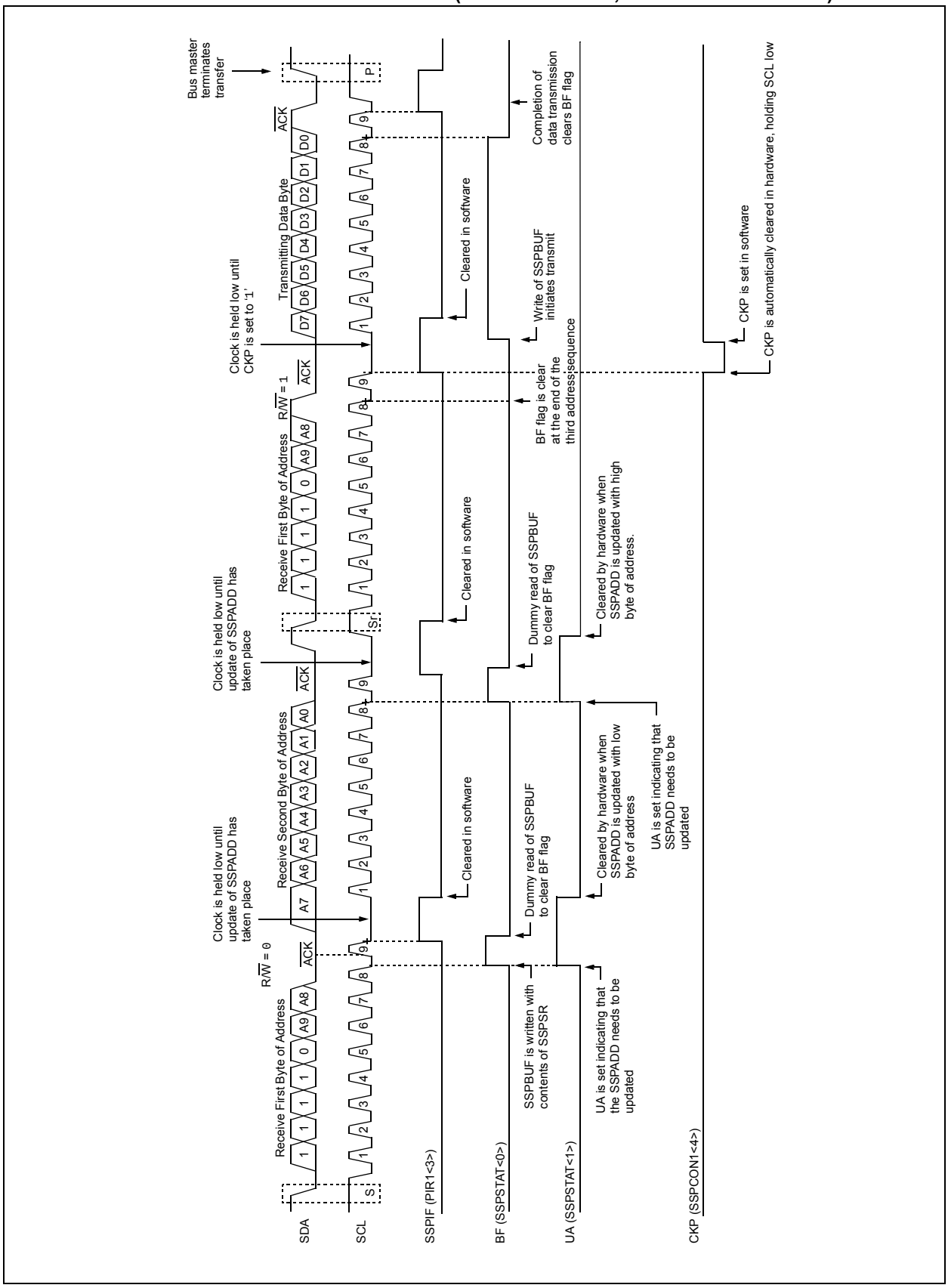


Note 1: x = Don't care (i.e., address bit can be either a '1' or a '0').

2: In this example, an address equal to A9.A8.A7.A6.A5.X.A3.A2.X.X will be Acknowledged and cause an interrupt.

3: Note that the Most Significant bits of the address are not affected by the bit masking.

FIGURE 17-13: I²C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESSING)



17.4.4 CLOCK STRETCHING

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

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

17.4.4.1 Clock Stretching for 7-Bit Slave Receive Mode (SEN = 1)

In 7-Bit Slave Receive mode when the BF bit is set, on the falling edge of the ninth clock at the end of the ACK sequence, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP bit being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-15).

Note 1: If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

17.4.4.2 Clock Stretching for 10-Bit Slave Receive Mode (SEN = 1)

In 10-Bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but the CKP bit is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

Note: If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

17.4.4.3 Clock Stretching for 7-Bit Slave Transmit Mode

The 7-Bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-10).

Note 1: If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software regardless of the state of the BF bit.

17.4.4.4 Clock Stretching for 10-Bit Slave Transmit Mode

In 10-Bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-Bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high-order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-Bit Slave Transmit mode (see Figure 17-13).

17.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCL output is forced to '0'. However, clearing 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^2C master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the I^2C bus have deasserted SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 17-14).

FIGURE 17-14: CLOCK SYNCHRONIZATION TIMING

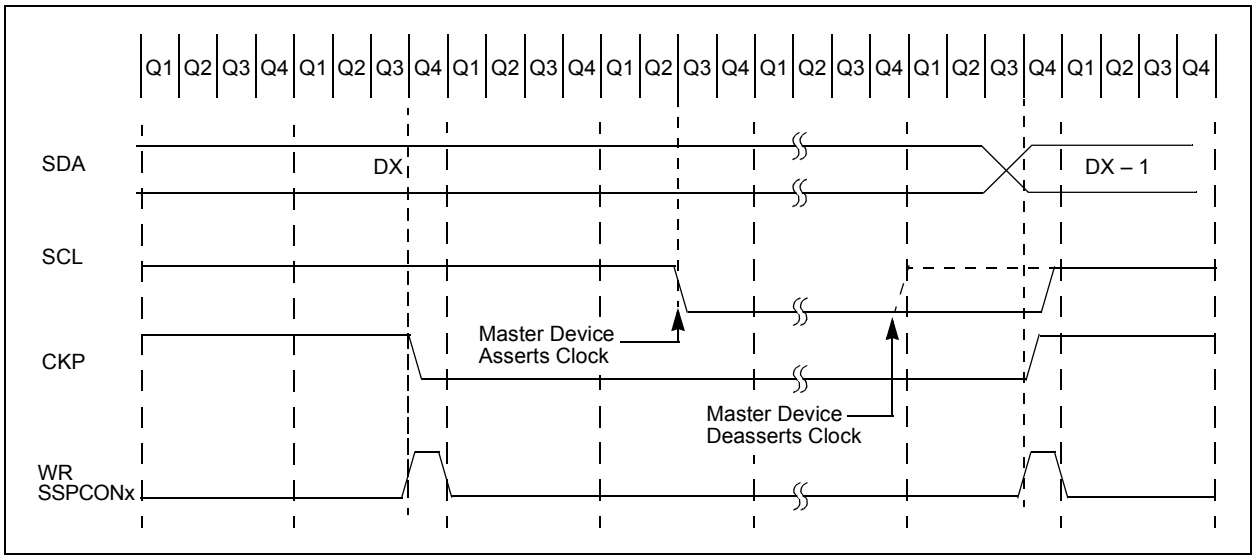


FIGURE 17-15: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESSING)

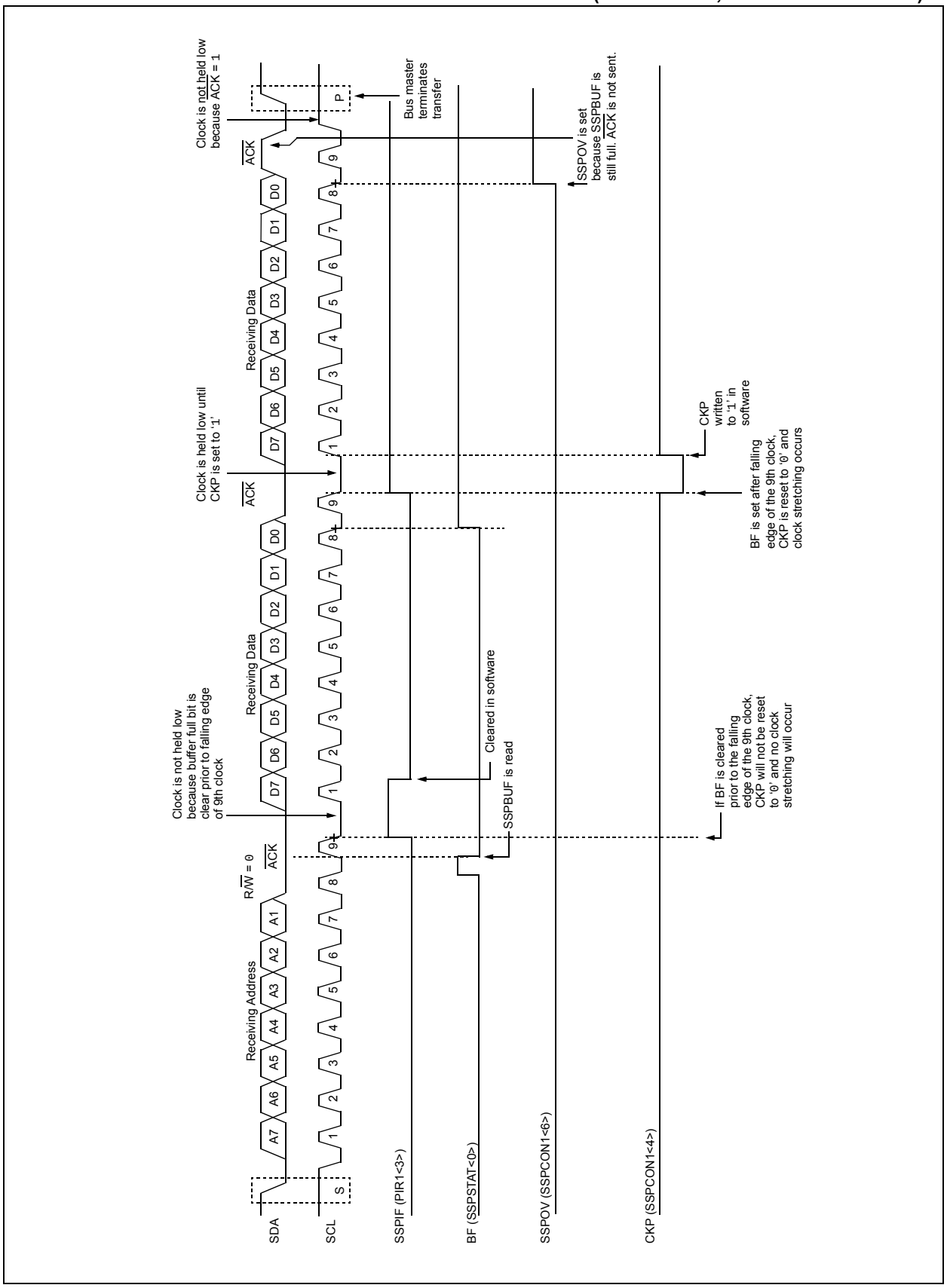
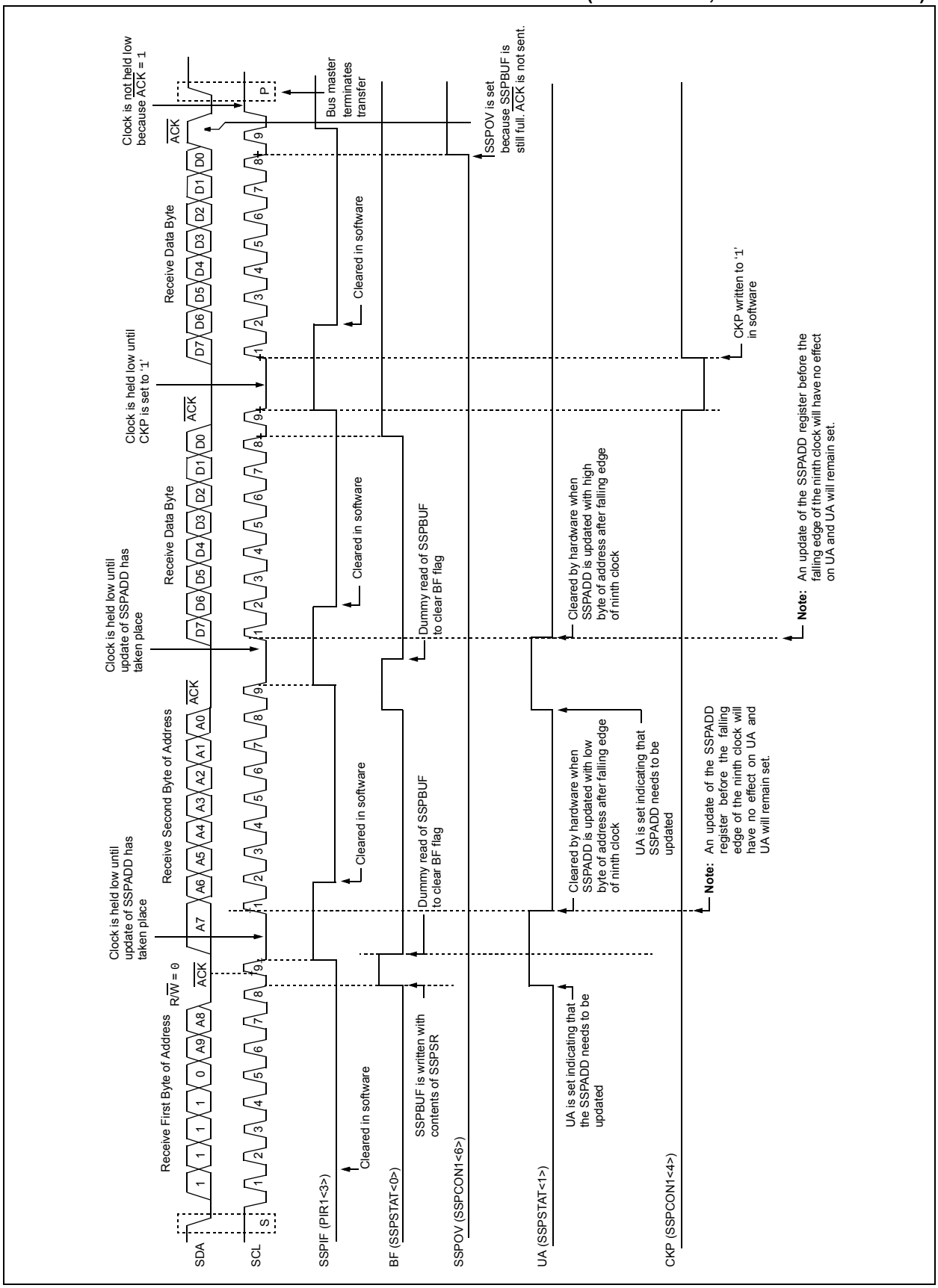


FIGURE 17-16: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESSING)



17.4.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I^2C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I^2C protocol. It consists of all '0's with R/$\overline{W}$ = 0.

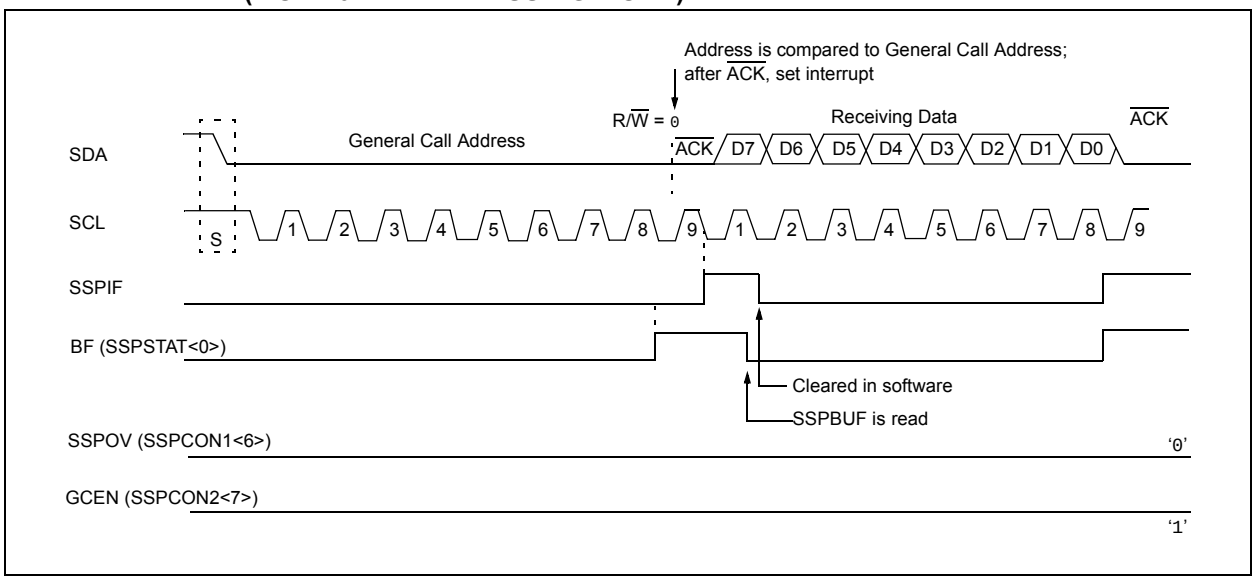
The general call address is recognized when the General Call Enable bit, GCEN, is enabled (SSPCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eighth bit), and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPBUF. The value can be used to determine if the address was device-specific or a general call address.

In 10-Bit Addressing 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 17-17).

FIGURE 17-17: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESSING MODE)



17.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I^2C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

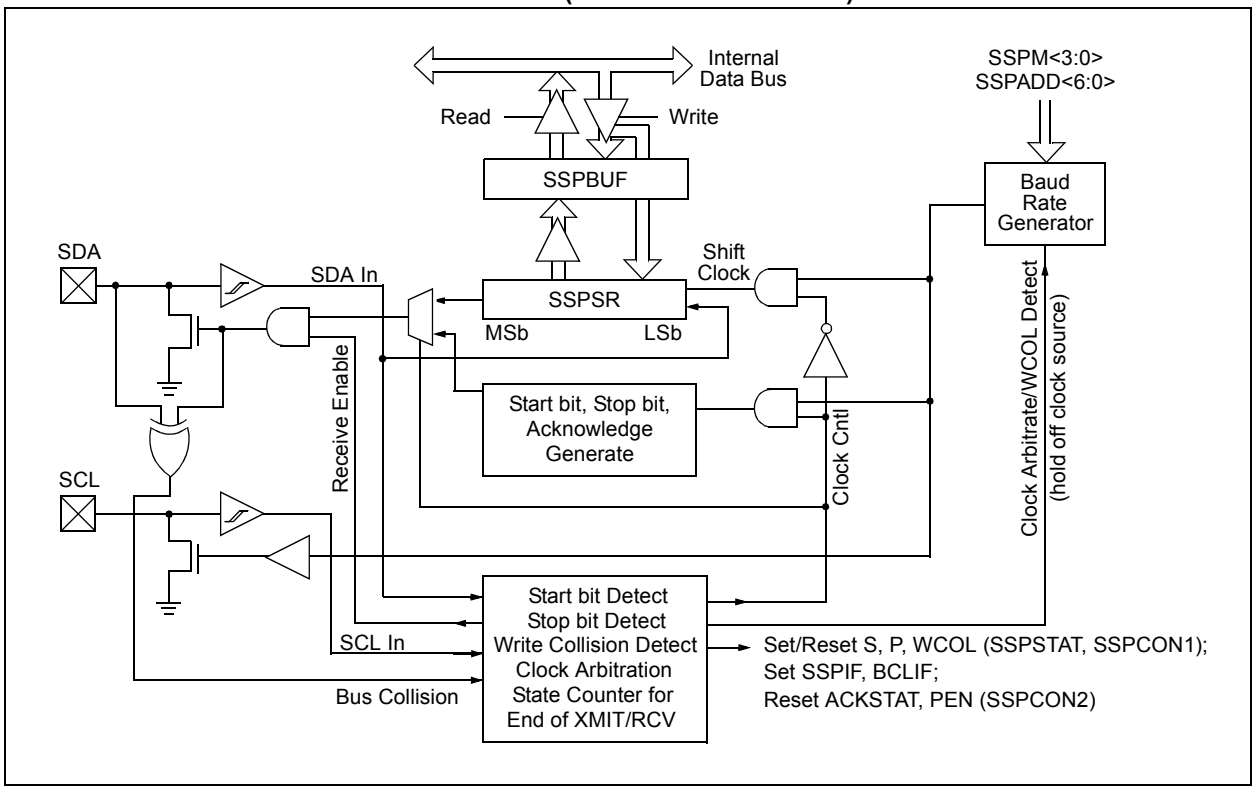
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

Note: The MSSP module, when configured in I^2C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

The following events will cause the MSSP Interrupt Flag bit, SSPIF, to be set (and MSSP interrupt, if enabled):

- Start Condition
- Stop Condition
- Data Transfer Byte Transmitted/Received
- Acknowledge Transmit
- Repeated Start

FIGURE 17-18: MSSP BLOCK DIAGRAM (I^2C™ MASTER MODE)



17.4.6.1 I²C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator used for the SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 17.4.7 "Baud Rate"** for more detail.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start Enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. The address is shifted out of the SDA pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with eight bits of data.
8. Data is shifted out of 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. The interrupt is generated once the Stop condition is complete.

17.4.7 BAUD RATE

In I^2C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPADD register (Figure 17-19). When a write occurs to SSPBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (TCY) on the Q2 and Q4 clocks. In I^2C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by $\overline{ACK}$), the internal clock will automatically stop counting and the SCL pin will remain in its last state.

Table 17-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD. The SSPADD BRG value of 0x00 is not supported.

17.4.7.1 Baud Rate Generation in Power-Managed Modes

When the device is operating in one of the power-managed modes, the clock source to the BRG may change frequency, or even stop, depending on the mode and clock source selected. Switching to a Run or Idle mode from either the secondary clock or internal oscillator is likely to change the clock rate to the BRG. In Sleep mode, the BRG will not be clocked at all.

FIGURE 17-19: BAUD RATE GENERATOR BLOCK DIAGRAM

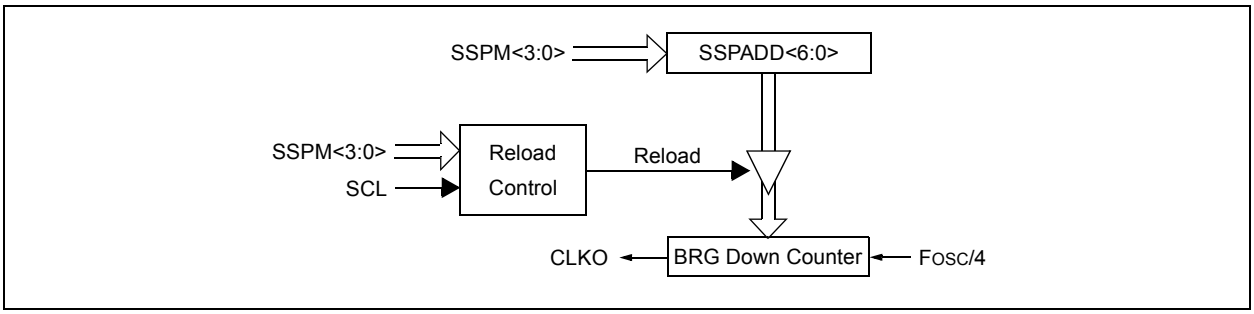


TABLE 17-3: I^2C™ CLOCK RATE w/BRG

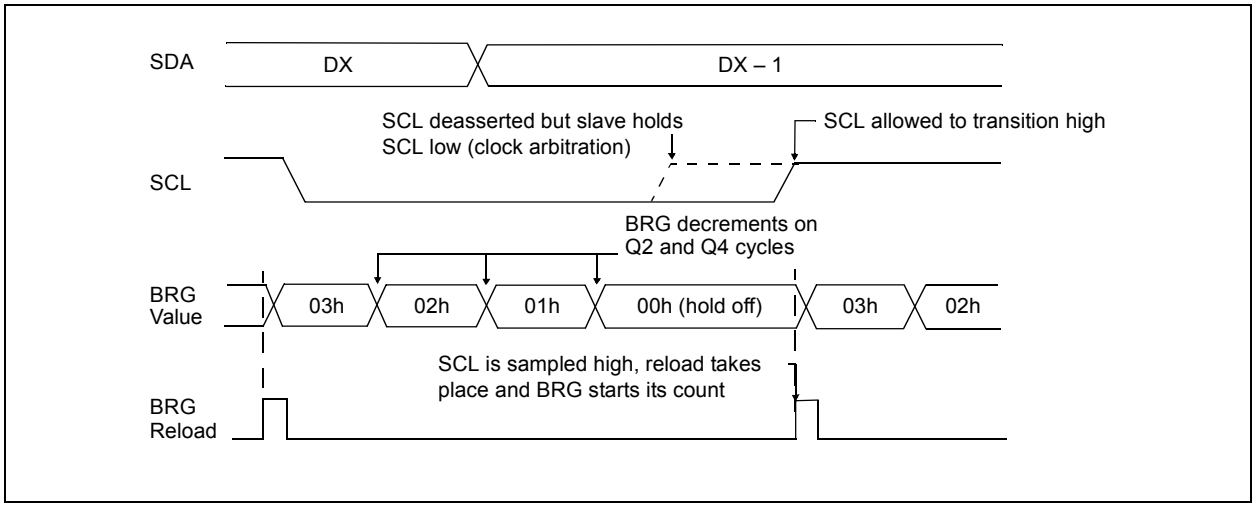
FCY	FCY * 2	BRG Value	FSCL (2 Rollovers of BRG)
10 MHz	20 MHz	18h	400 kHz(1)
10 MHz	20 MHz	1Fh	312.5 kHz
10 MHz	20 MHz	63h	100 kHz
4 MHz	8 MHz	09h	400 kHz(1)
4 MHz	8 MHz	0Ch	308 kHz
4 MHz	8 MHz	27h	100 kHz
1 MHz	2 MHz	02h	333 kHz(1)
1 MHz	2 MHz	09h	100 kHz

Note 1: The I^2C interface does not conform to the 400 kHz I^2C 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.

17.4.7.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 17-20).

FIGURE 17-20: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



17.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware. The Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

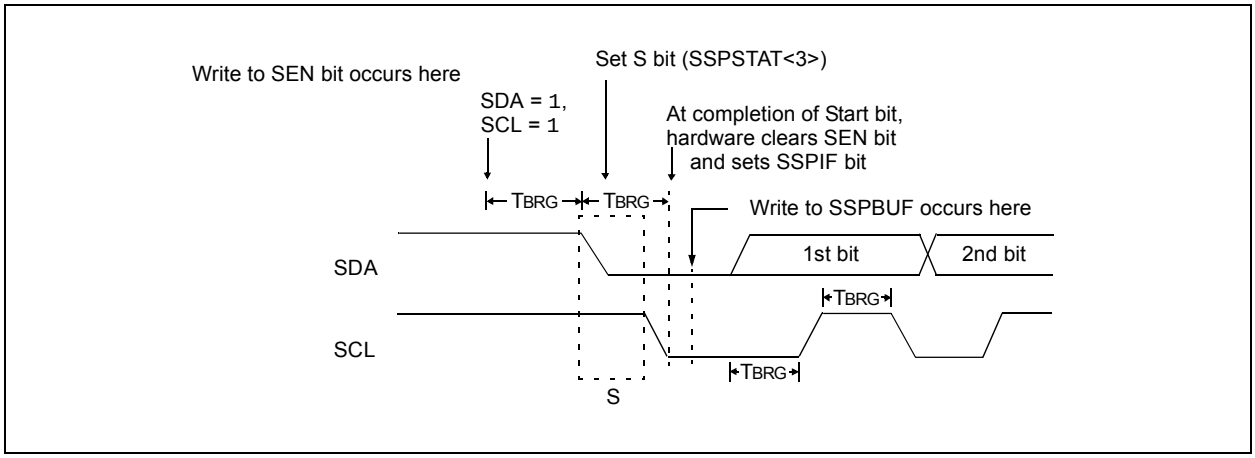
Note:	If, at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs. The Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

17.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.

FIGURE 17-21: FIRST START BIT TIMING



17.4.9 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPADD<6:0> 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'.

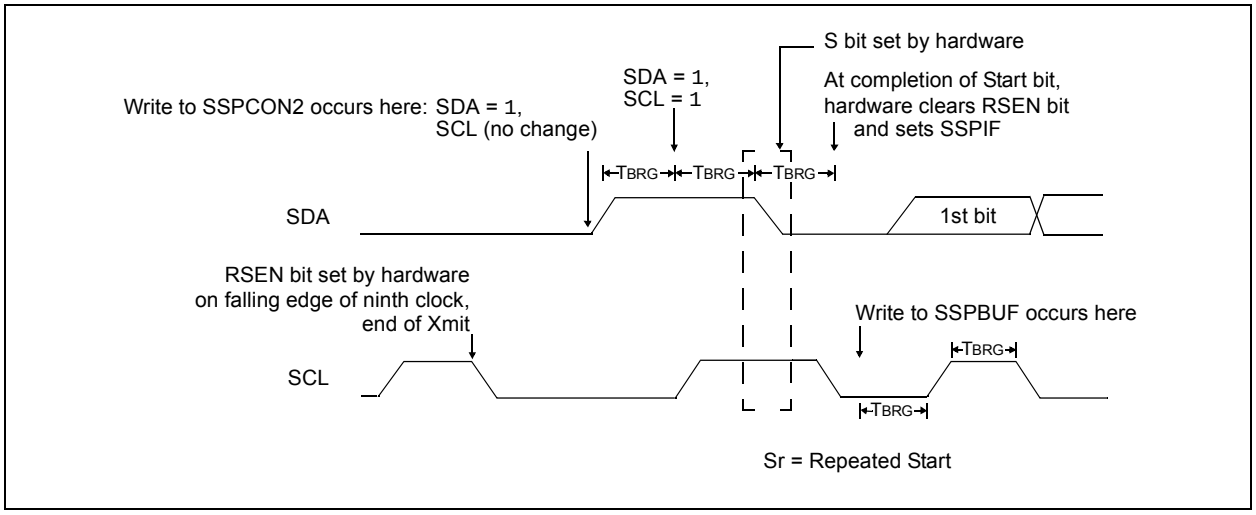
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-Bit Addressing mode or the default first address in 10-Bit Addressing mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-Bit Addressing mode) or eight bits of data (7-Bit Addressing mode).

17.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

FIGURE 17-22: REPEATED START CONDITION WAVEFORM



17.4.10 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPBUF register. This action will set the Buffer Full bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted (see data hold time specification parameter 106). SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high (see data setup time specification parameter 107). When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. This allows the slave device being addressed to respond with an $\overline{ACK}$ bit during the ninth bit time if an address match occurred, or if data was received properly. The status of $\overline{ACK}$ is written into the ACKDT bit on the falling edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared; if not, the bit is set. After the ninth clock, the SSPIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPBUF, leaving SCL low and SDA unchanged (Figure 17-23).

After the write to the SSPBUF, each bit of the address will be shifted out on the falling edge of SCL until all seven address bits and the R/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the master will deassert the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the $\overline{ACK}$ bit is loaded into the ACKSTAT bit (SSPCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPIF bit is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

17.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPSTAT<0>) is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

17.4.10.2 WCOL Status Flag

If the user writes to the SSPBUF when a transmit is already in progress (i.e., SSPSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur) 2 TCY after the SSPBUF write. If SSPBUF is rewritten within 2 TCY, the WCOL bit is set and SSPBUF is updated. This may result in a corrupted transfer.

The user should verify that the WCOL is clear after each write to SSPBUF to ensure the transfer is correct. In all cases, WCOL must be cleared in software.

17.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPCON2<6>) is cleared when the slave has sent an Acknowledge ($\overline{ACK}$ = 0) and is set when the slave does not Acknowledge ($\overline{ACK}$ = 1). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

17.4.11 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPCON2<3>).

Note:	The MSSP module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting, and on each rollover, the state of the SCL pin changes (high-to-low/low-to-high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR are loaded into the SSPBUF, the BF flag bit is set, the SSPIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCL low. The MSSP is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>).

17.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPBUF from SSPSR. It is cleared when the SSPBUF register is read.

17.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

17.4.11.3 WCOL Status Flag

If the user writes the SSPBUF when a receive is already in progress (i.e., SSPSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 17-23: I²C™ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESSING)

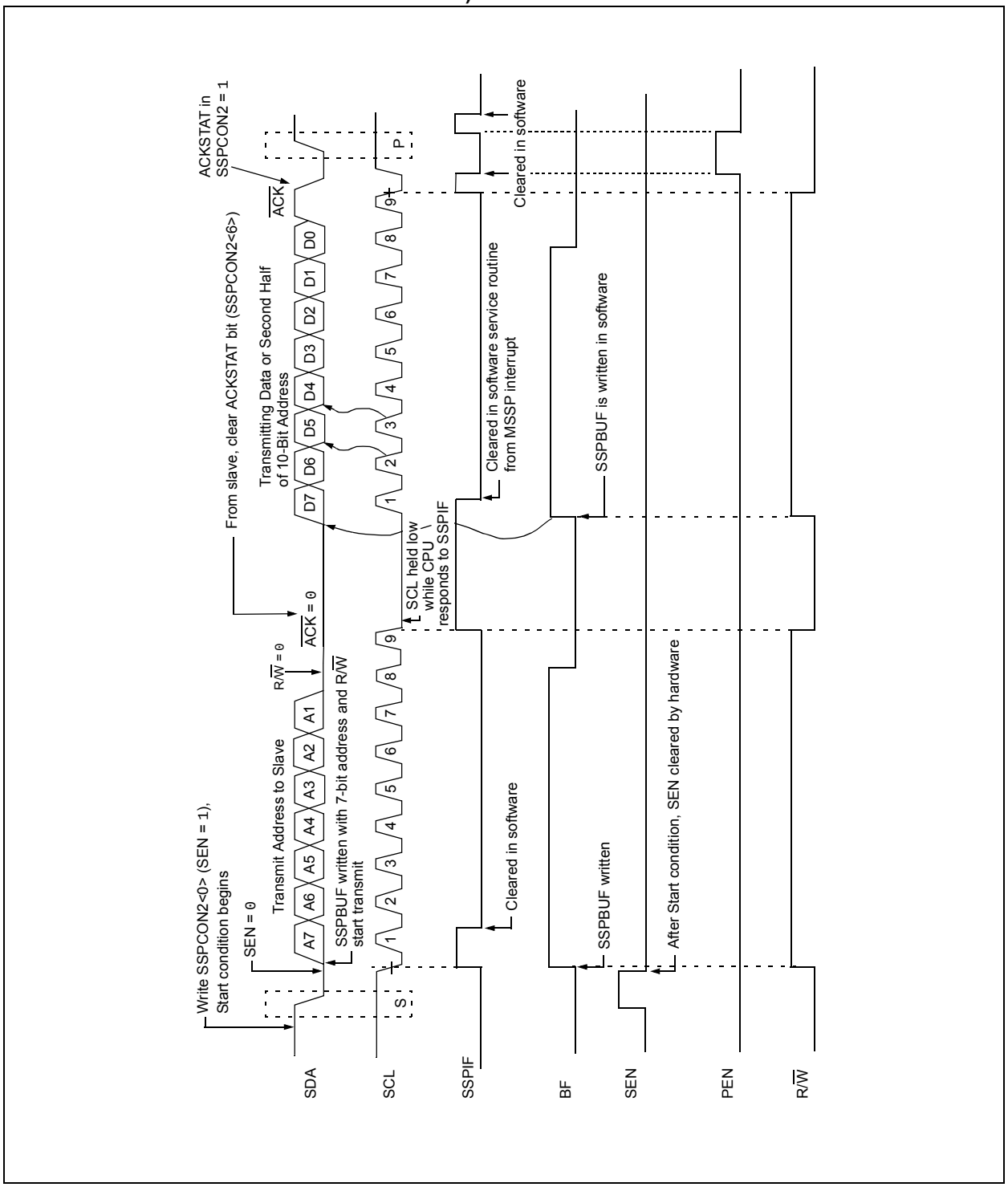
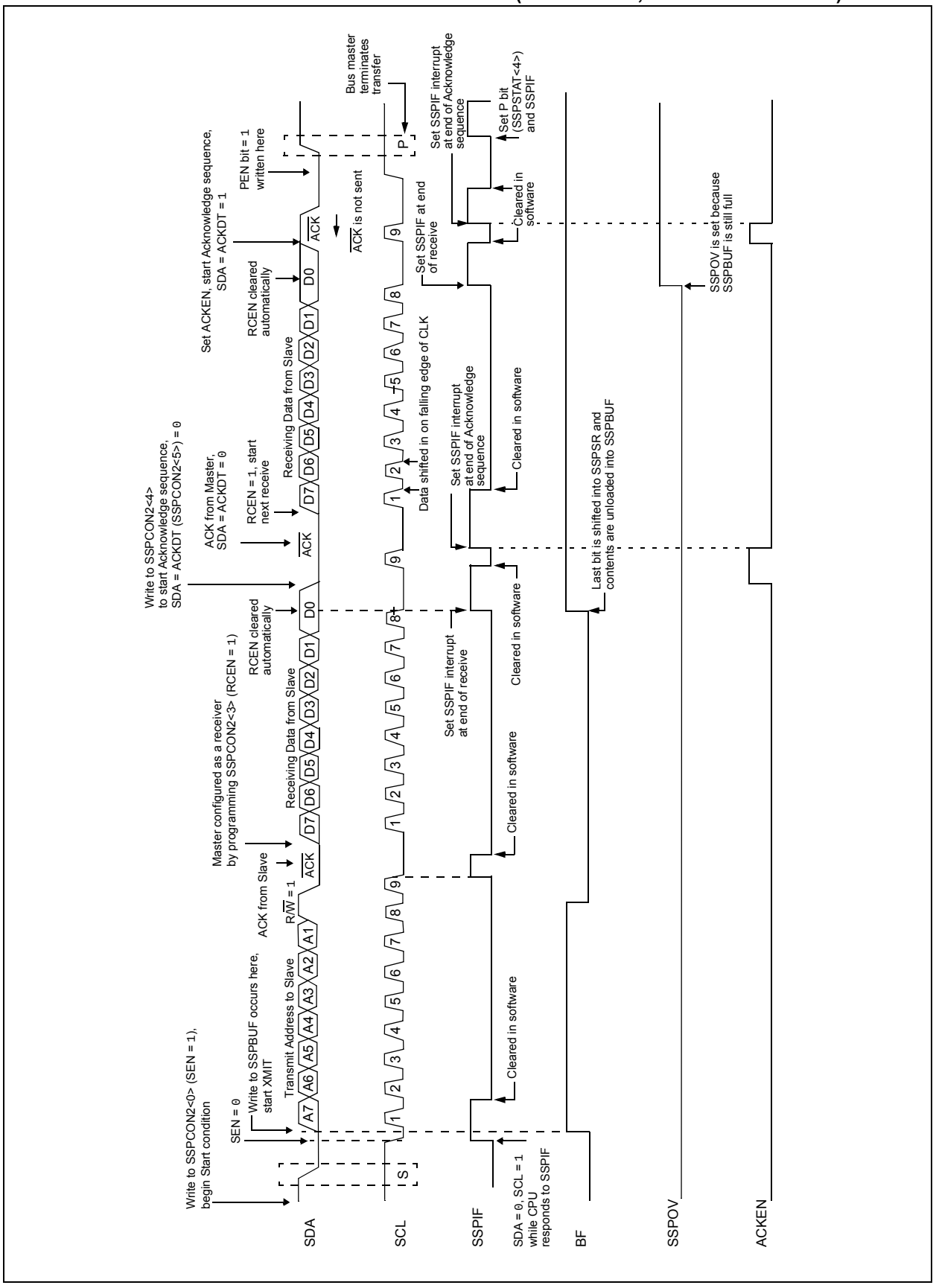


FIGURE 17-24: I²C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESSING)



17.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 17-25).

17.4.12.1 WCOL Status Flag

If the user writes the SSPBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

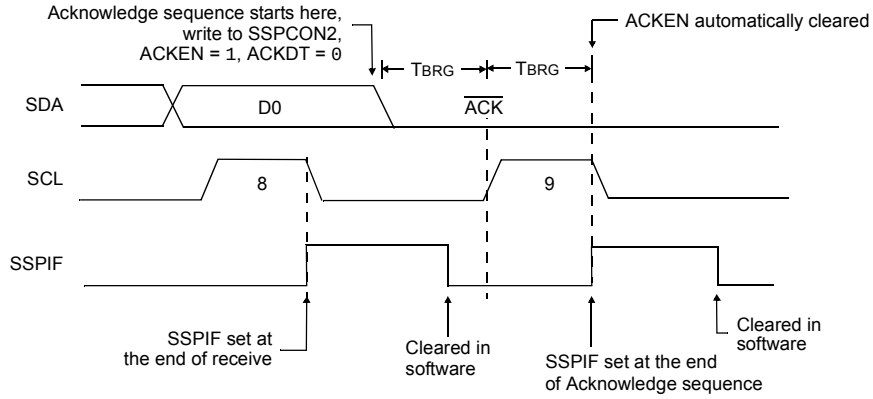
17.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to 0. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 17-26).

17.4.13.1 WCOL Status Flag

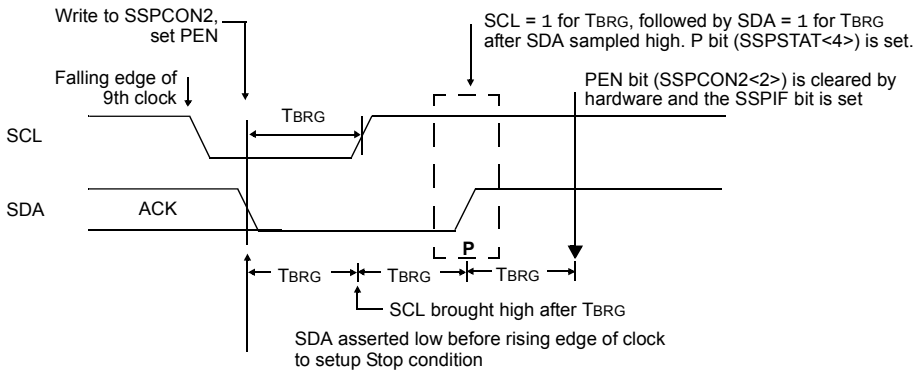
If the user writes the SSPBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 17-25: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 17-26: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one Baud Rate Generator period.

17.4.14 SLEEP OPERATION

While in Sleep mode, the I^2C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

17.4.15 EFFECTS OF A RESET

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

17.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I^2C 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

17.4.17 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA by letting SDA float high, and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the I^2C port to its Idle state (Figure 17-27).

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^2C bus is free, the user can resume communication by asserting a Start condition.

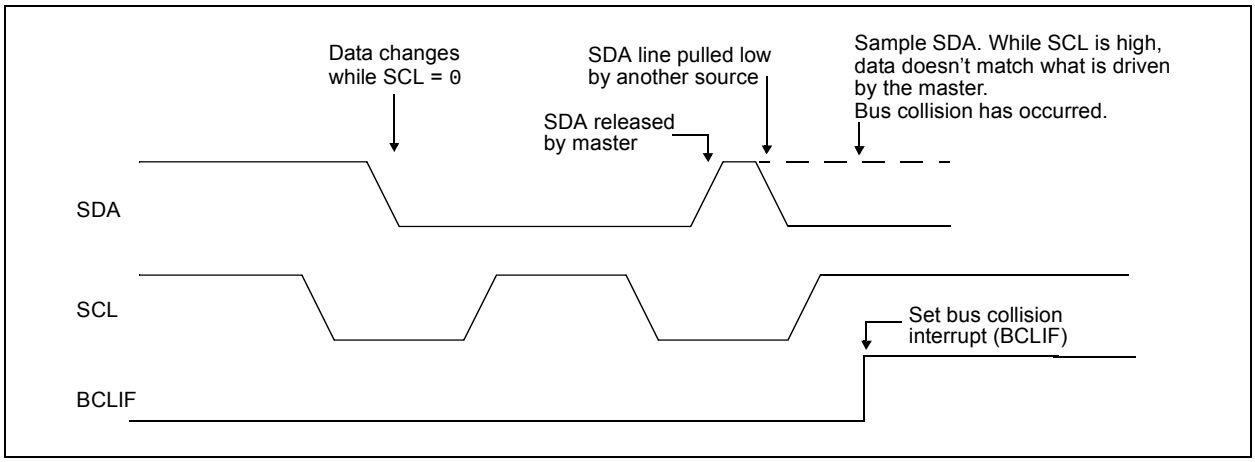
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I^2C 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^2C bus can be taken when the P bit is set in the SSPSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 17-27: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



17.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDA or SCL is sampled low at the beginning of the Start condition (Figure 17-28).

b) SCL is sampled low before SDA is asserted low (Figure 17-29).

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 17-28).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded from SSPADD<6:0> and counts down to 0. If the SCL pin is sampled low while SDA is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 17-30). 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. If the SCL pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

Note:	The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 17-28: BUS COLLISION DURING START CONDITION (SDA ONLY)

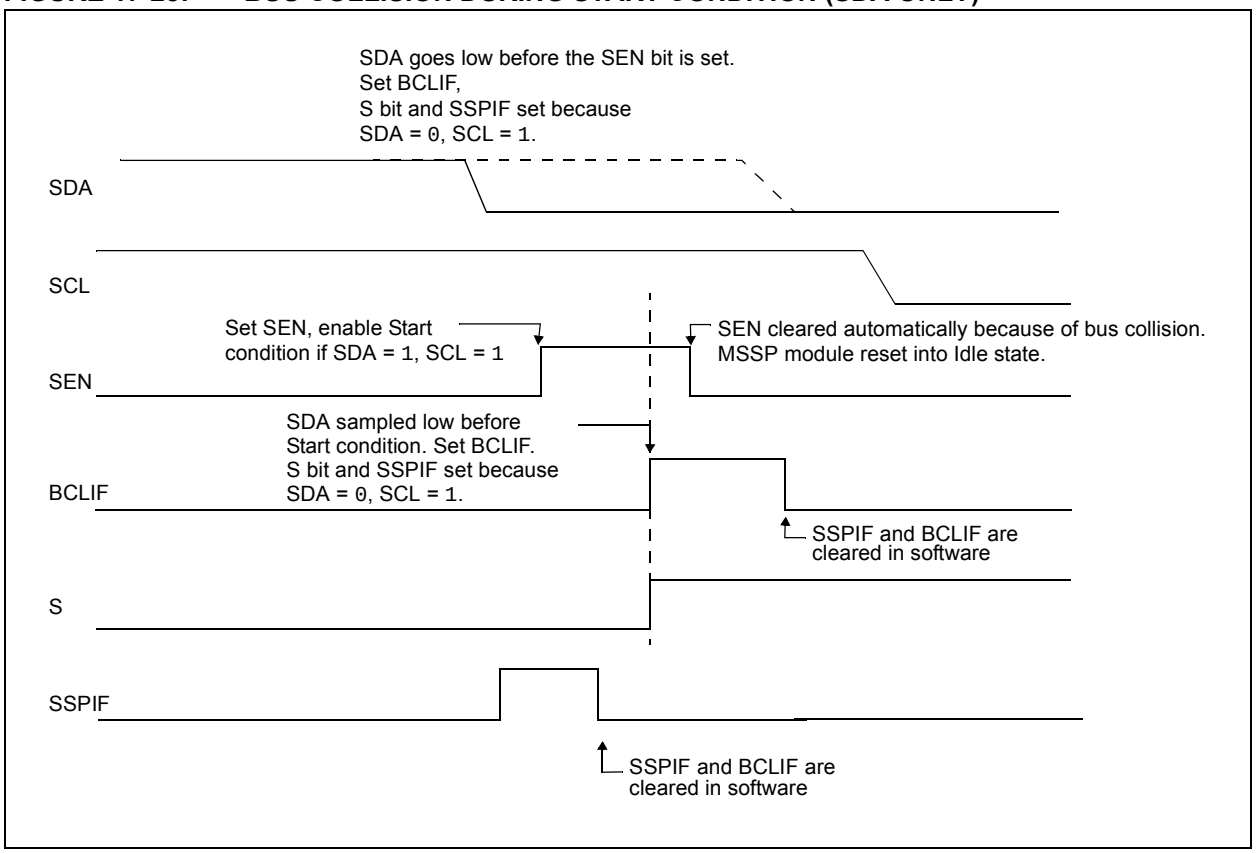


FIGURE 17-29: BUS COLLISION DURING START CONDITION (SCL = 0)

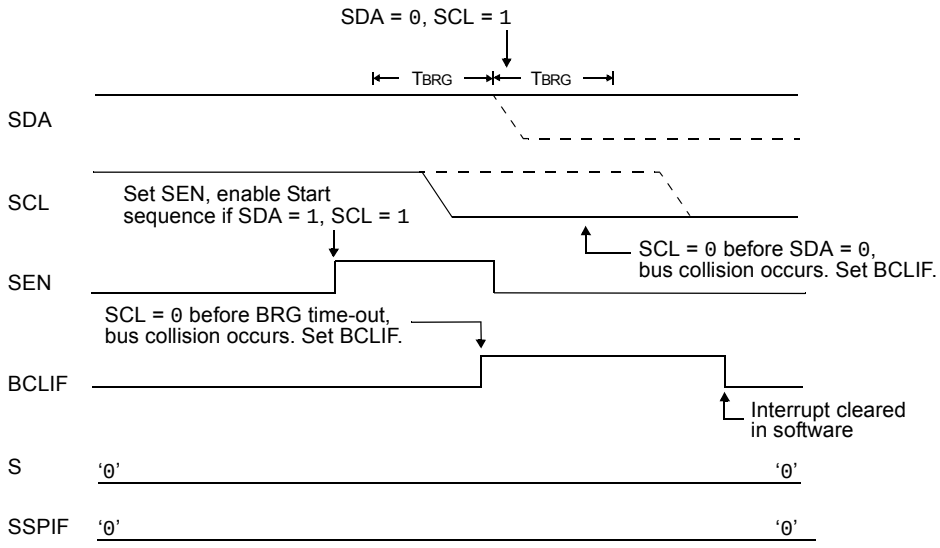
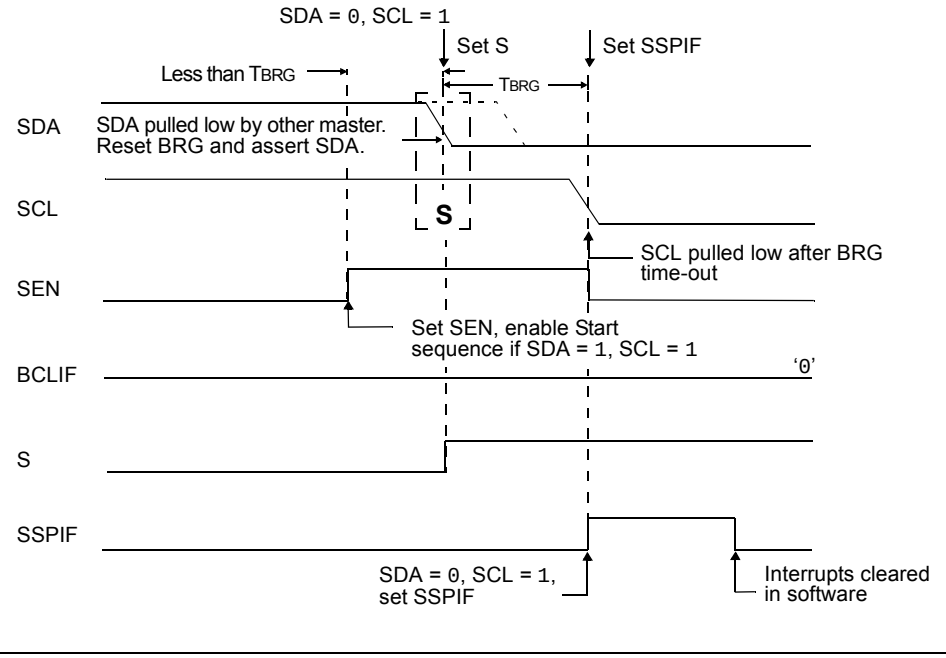


FIGURE 17-30: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION



17.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from a low level to a 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', see Figure 17-31). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high-to-low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 17-32).

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

FIGURE 17-31: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

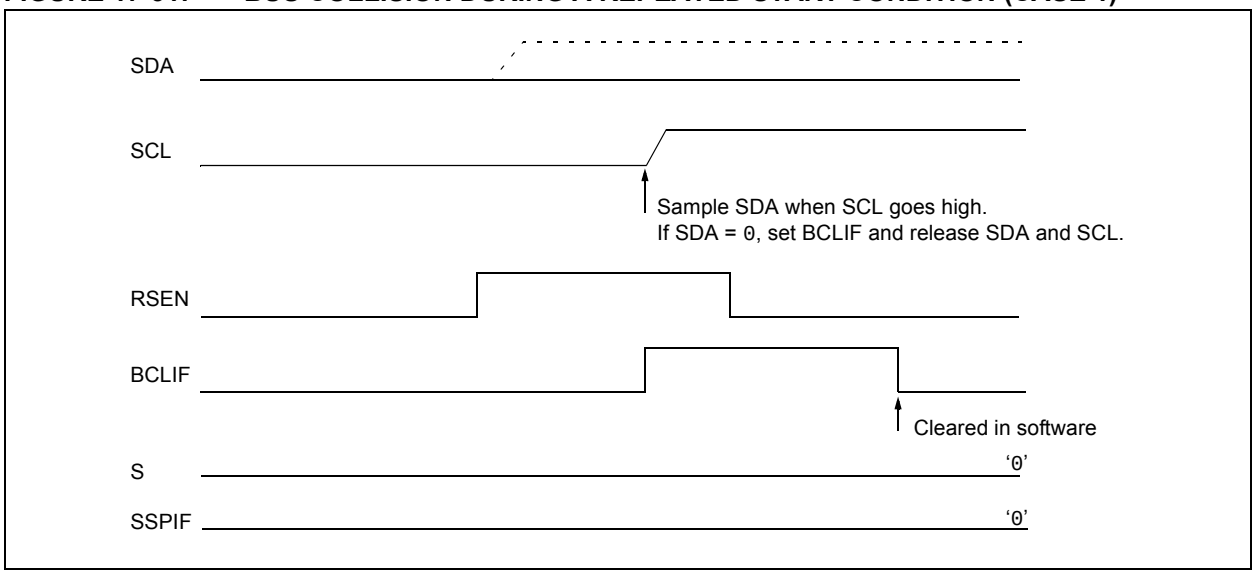
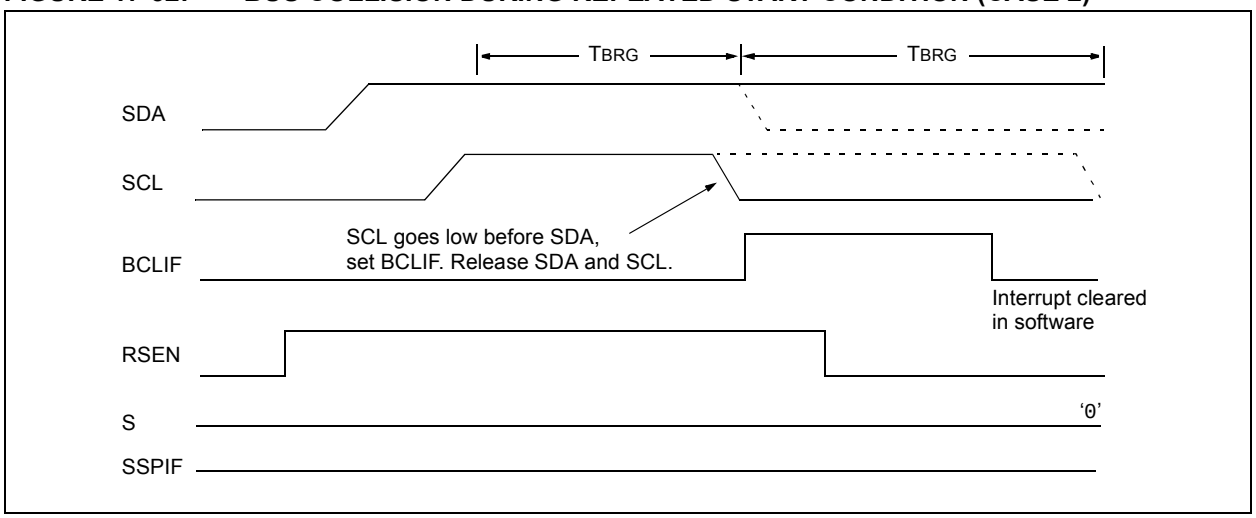


FIGURE 17-32: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



17.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out.

b) After the SCL pin is deasserted, SCL is sampled low before SDA goes high.

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD<6:0> and counts down to 0. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 17-33). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 17-34).

FIGURE 17-33: BUS COLLISION DURING A STOP CONDITION (CASE 1)

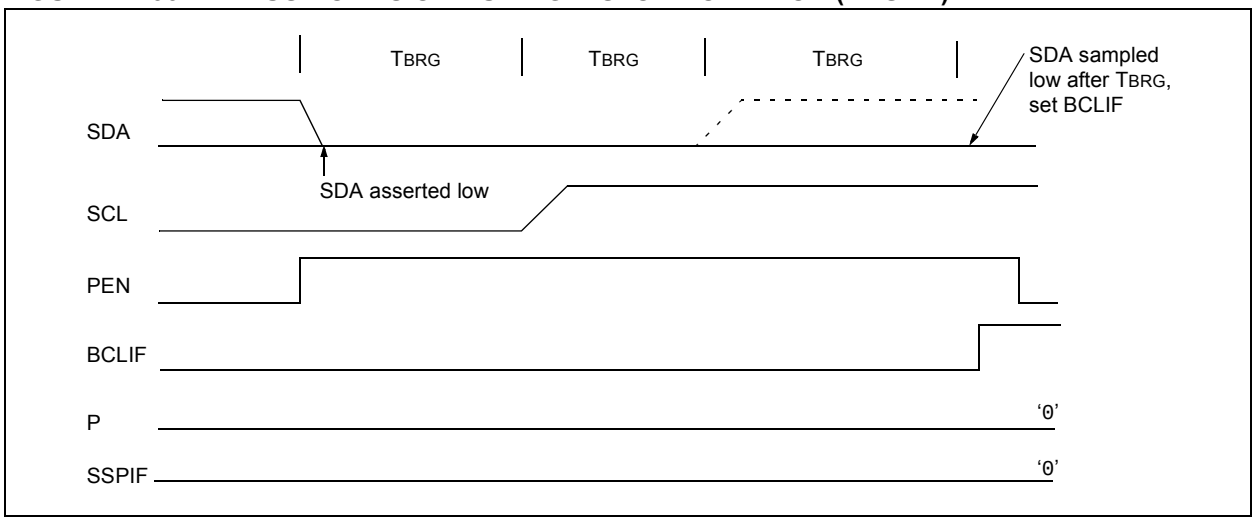


FIGURE 17-34: BUS COLLISION DURING A STOP CONDITION (CASE 2)

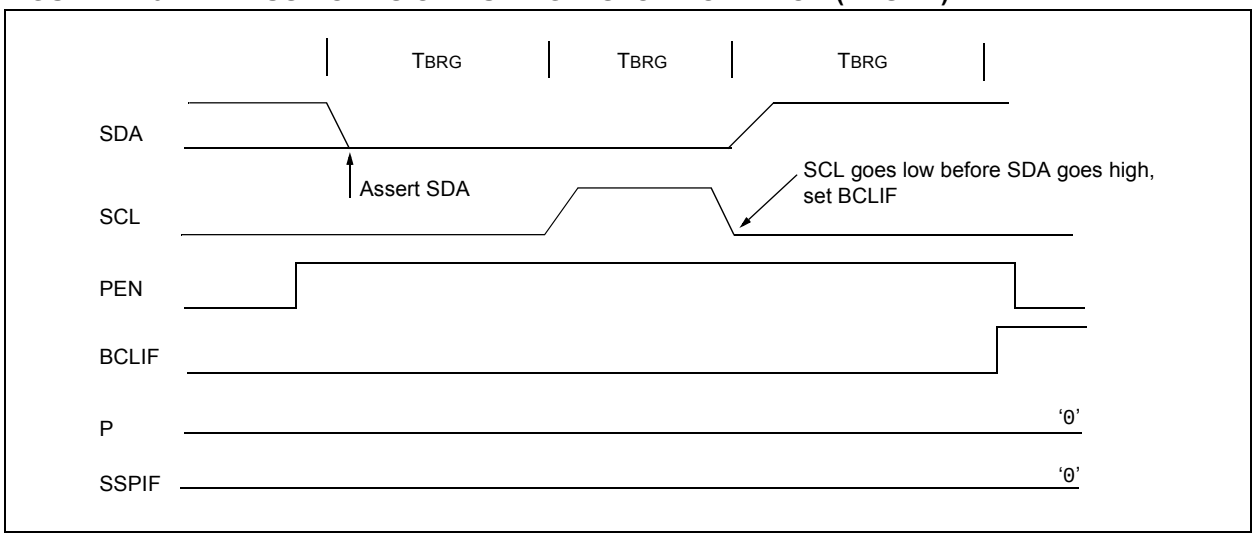


TABLE 17-4: REGISTERS ASSOCIATED WITH I²C™ OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—	59
PIE2	OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—	59
IPR2	OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—	59
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
SSPBUF	MSSP Receive Buffer/Transmit Register								58
SSPADD	MSSP Address Register (I²C™ Slave mode), MSSP Baud Rate Reload Register (I²C Master mode)								58
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	58
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	58
	GCEN	ACKSTAT	ADMSK5(1)	ADMSK4(1)	ADMSK3(1)	ADMSK2(1)	ADMSK1(1)	SEN	
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	58

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the MSSP module in I²C™ mode.

Note 1: Alternate bit definitions for use in I²C Slave mode operations only.

18.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

PIC18F85J11 family 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 EUSART can be configured as a full-duplex, asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex, synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

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 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/J2602 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 a EUSART:

- SPEN bit (RCSTA1<7>) must be set (= 1)
- TRISC<7> bit must be set (= 1)
- TRISC<6> bit must be set (= 1)

Note:	The EUSART control will automatically reconfigure the pin from input to output as needed.

The driver for the TX1 output pin can also be optionally configured as an open-drain output. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, and allows the output to communicate with external circuits without the need for additional level shifters.

The open-drain output option is controlled by the U1OD bit (LATG<6>). Setting the bit configures the pin for open-drain operation.

18.1 Control Registers

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 18-1, Register 18-2 and Register 18-3.

REGISTER 18-1: 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(1)	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(1)

1 = Transmit is enabled
0 = Transmit is disabled

bit 4 **SYNC:** AUSART 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 is empty
0 = TSR is 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.

REGISTER 18-2: 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 is enabled
0 = Serial port is 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 cleared by reading the RCREG1 register and receiving the 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 an address/data bit or a parity bit and must be calculated by user firmware.

REGISTER 18-3: BAUDCON1: BAUD RATE CONTROL REGISTER 1

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

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

bit 7 **ABDOVF**: Auto-Baud Acquisition Rollover Status bit
1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
0 = No BRG rollover has occurred

bit 6 **RCIDL** Receive Operation Idle Status bit
1 = Receive operation is Idle
0 = Receive operation is active

bit 5 **RXDTP:** Received Data Polarity Select bit (Asynchronous mode only)
Asynchronous mode:
1 = Receive data (RXx) is inverted (active-low)
0 = Receive data (RXx) is not inverted (active-high)

bit 4 **TXCKP:** Clock and Data Polarity Select bit
Asynchronous mode:
1 = Idle state for transmit (TXx) is a low level
0 = Idle state for transmit (TXx) is a high level
Synchronous mode:
1 = Idle state for clock (CKx) is a high level
0 = Idle state for clock (CKx) is a low level

bit 3 **BRG16:** 16-Bit Baud Rate Register Enable bit
1 = 16-bit Baud Rate Generator – 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 the following rising edge
0 = RX1 pin is 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.

18.2 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, BRGH (TXSTA1<2>) and BRG16 (BAUDCON1<3>) bits also control the baud rate. In Synchronous mode, BRGH is ignored. Table 18-1 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 18-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 18-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 18-2. It may be advantageous to use the high baud rate (BRGH = 1) or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the 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. When operated in the Synchronous mode, SPBRGH:SPBRG values of 0000h and 0001h are not supported. In the Asynchronous mode, all BRG values may be used.

18.2.1 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.

18.2.2 SAMPLING

The data on the RX1 pin is sampled three times by a majority detect circuit to determine if a high or low level is present at the RX1 pin.

TABLE 18-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-Bit/Asynchronous	FOSC/[64 (n + 1)]
0	0	1	8-Bit/Asynchronous	FOSC/[16 (n + 1)]
0	1	0	16-Bit/Asynchronous	
0	1	1	16-Bit/Asynchronous	FOSC/[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 18-1: 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 = FOSC/(64 ([SPBRGH1:SPBRG1] + 1))

Solving for SPBRGH1:SPBRG1:

X = ((FOSC/Desired Baud Rate)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25

Calculated Baud Rate = 16000000/(64 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
= (9615 – 9600)/9600 = 0.16%

TABLE 18-2: 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	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59

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

TABLE 18-3: 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	—	—	—	—	—	—

TABLE 18-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)	Actual Rate (K)	% Error	SPBRG Value (decimal)
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

18.2.3 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 18-1) 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/J2602 bus Sync character), in order to calculate the proper bit rate. The measurement is taken over both a low and high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the 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 18-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. Note that the BRG clock can be configured by the BRG16 and BRGH bits. The BRG16 bit must be set to use both SPBRG1 and SPBRGH1 as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGH1 register. Refer to Table 18-4 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.

TABLE 18-4: BRG COUNTER CLOCK RATES

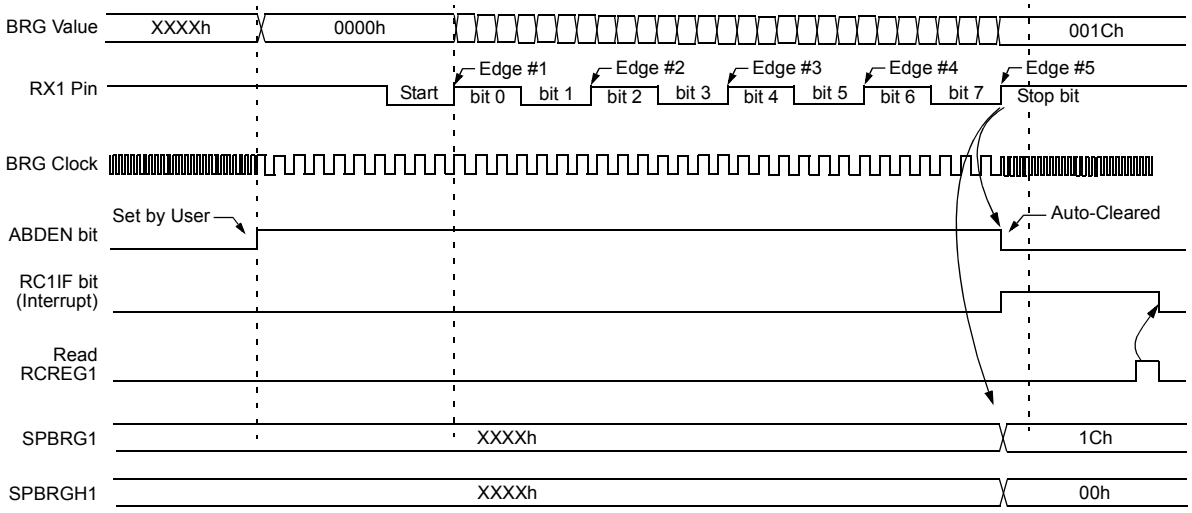
BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

Note: During the ABD sequence, SPBRG1 and SPBRGH1 are both used as a 16-bit counter, independent of the BRG16 setting.

18.2.3.1 ABD and EUSART Transmission

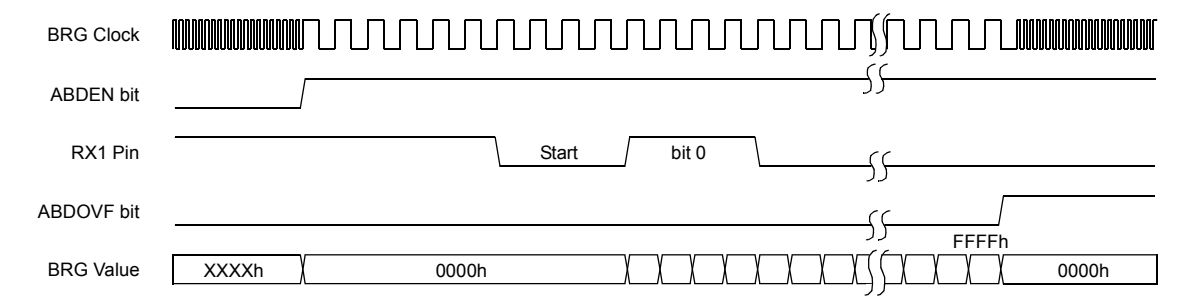
Since the BRG clock is reversed during ABD acquisition, the EUSART transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREG1 cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSART operation.

FIGURE 18-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the EUSART module to be configured in Asynchronous mode and WUE = 0.

FIGURE 18-2: BRG OVERFLOW SEQUENCE



18.3 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

18.3.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 18-3. 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 an 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 18-3: EUSART TRANSMIT BLOCK DIAGRAM

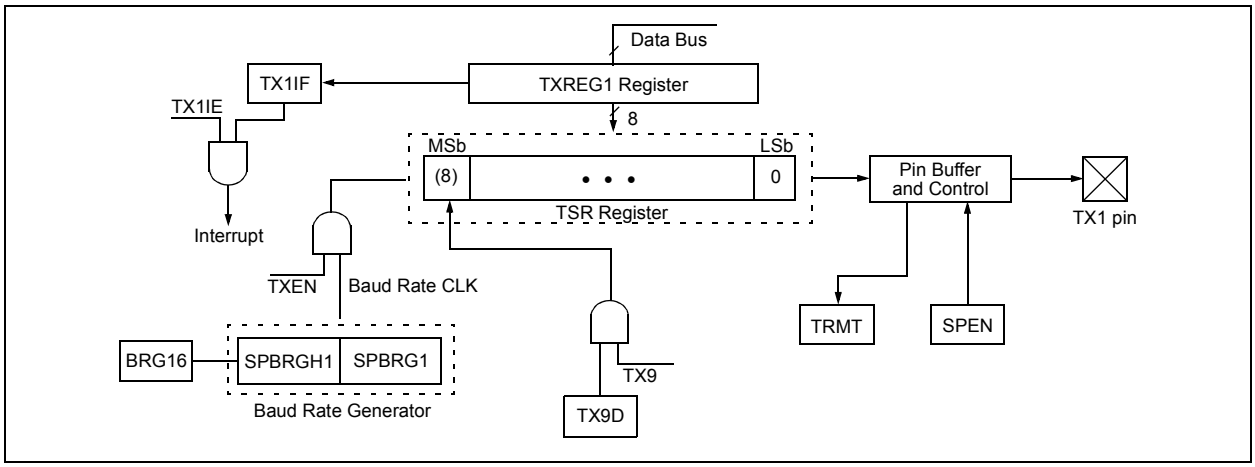


FIGURE 18-4: ASYNCHRONOUS TRANSMISSION

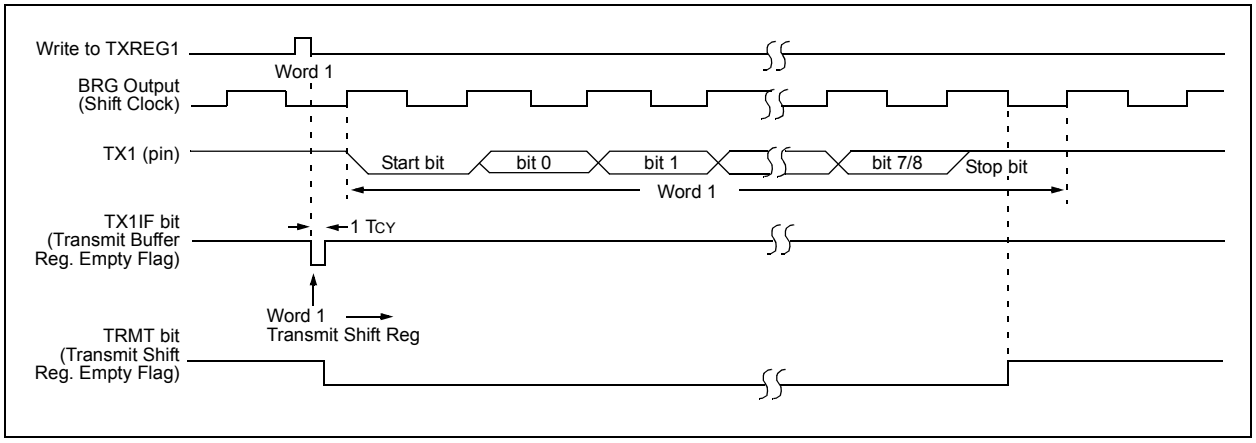
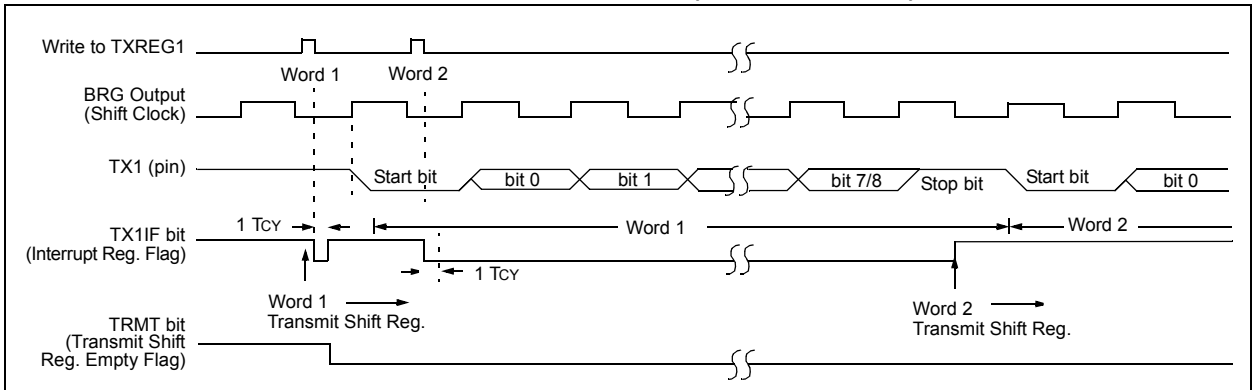


FIGURE 18-5: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 18-5: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
TXREG1	EUSART Transmit Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

18.3.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 18-6. 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.

18.3.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the 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 18-6: EUSART RECEIVE BLOCK DIAGRAM

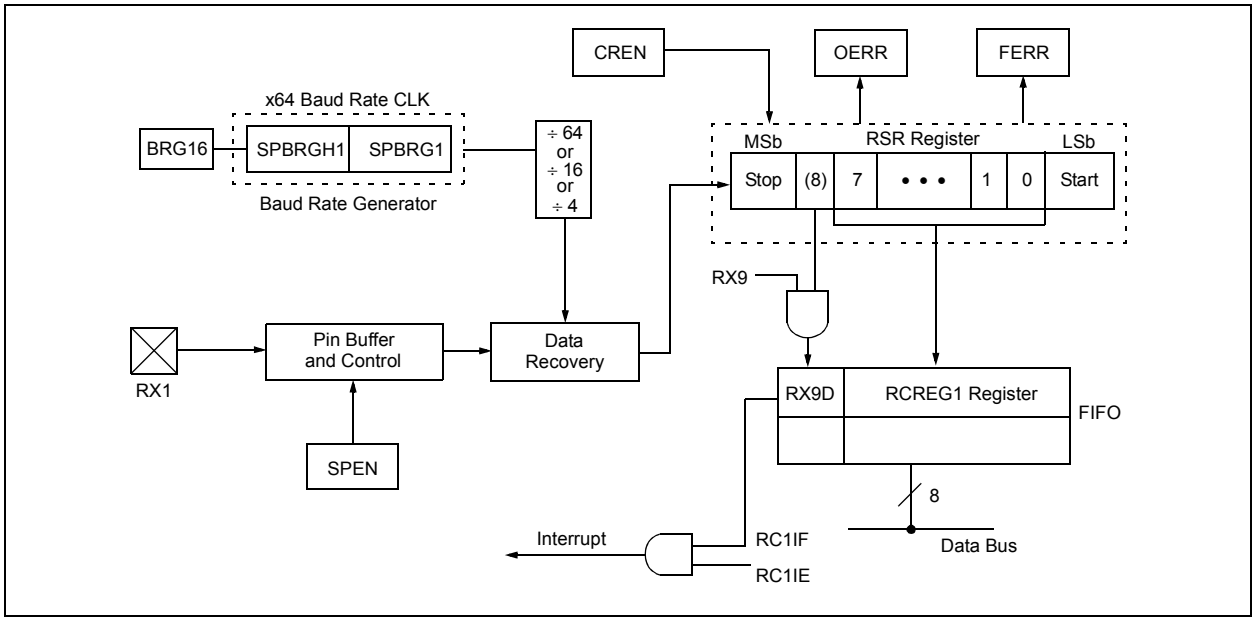
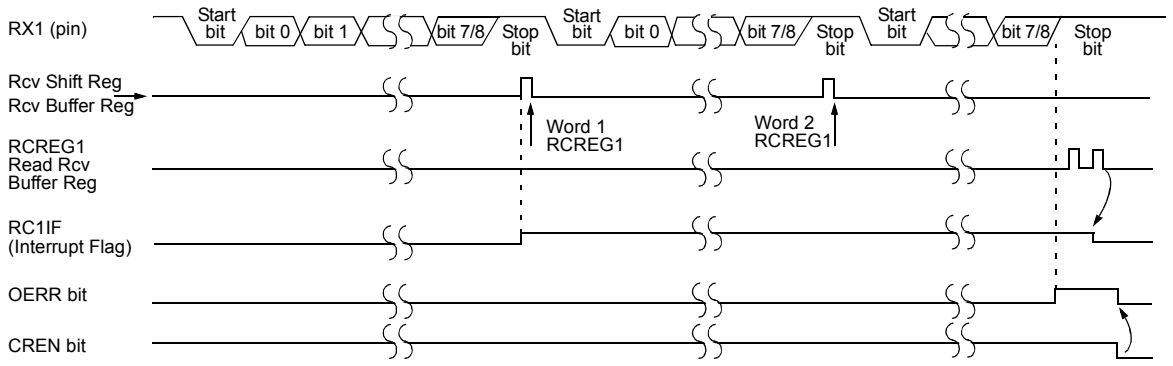


FIGURE 18-7: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX1 input. The RCREG1 (EUSART Receive register) is read after the third word, causing the OERR (Overrun Error) bit to be set.

TABLE 18-6: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
RCREG1	EUSART Receive Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

18.3.4 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/J2602 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 18-8), and asynchronously, if the device is in Sleep mode (Figure 18-9). 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.

18.3.4.1 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 bits) for standard RS-232 devices or 000h (12 bits) for LIN/J2602 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.

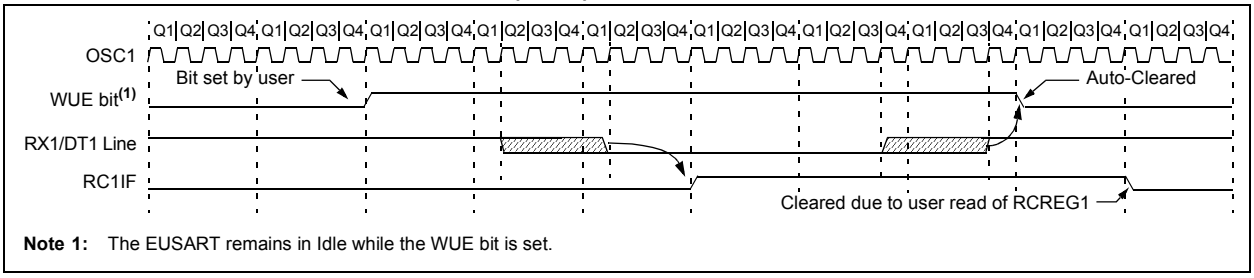
18.3.4.2 Special Considerations Using the WUE Bit

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.

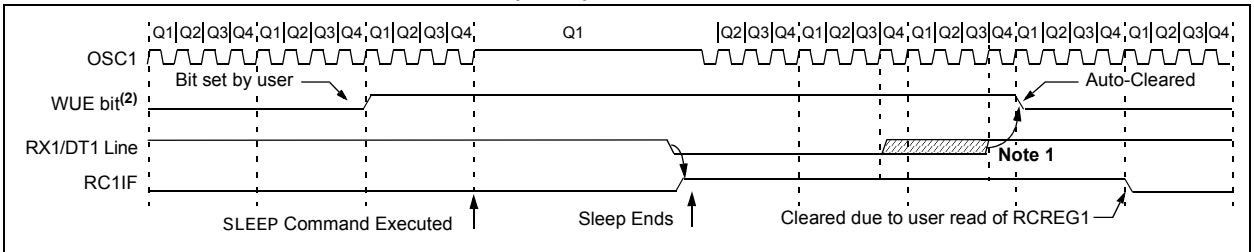
To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 18-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION



Note 1: The EUSART remains in Idle while the WUE bit is set.

FIGURE 18-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the auto-clear of the WUE bit can occur while the *stposc* signal is still active. This sequence should not depend on the presence of Q clocks.

2: The EUSART remains in Idle while the WUE bit is set.

18.3.5 BREAK CHARACTER SEQUENCE

The Enhanced USART module has the capability of sending the special Break character sequences that are required by the LIN/J2602 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 (TXSTA1<3> and TXSTA1<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/J2602 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 18-10 for the timing of the Break character sequence.

18.3.5.1 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/J2602 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 bit, the next data byte can be written to TXREG1.

18.3.6 RECEIVING A BREAK CHARACTER

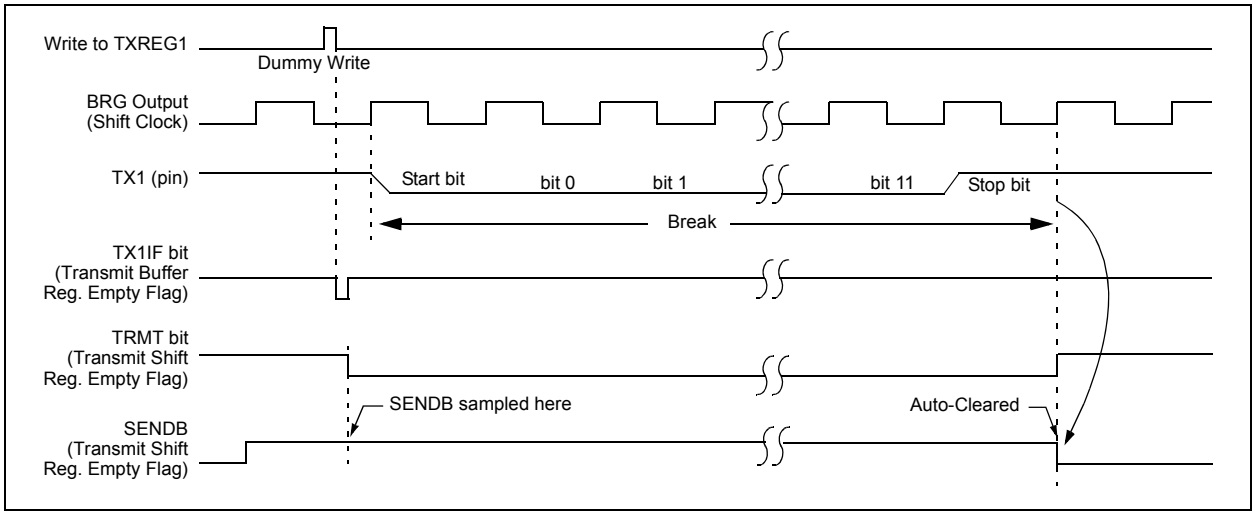
The Enhanced USART module can receive a Break character in two ways.

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 18.3.4 "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 ABDEN bit once the TX1IF interrupt is observed.

FIGURE 18-10: SEND BREAK CHARACTER SEQUENCE



18.4 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA1<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 (TXSTA1<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 TXCKP bit (BAUDCON1<4>). Setting TXCKP 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.

18.4.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 18-3. 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).

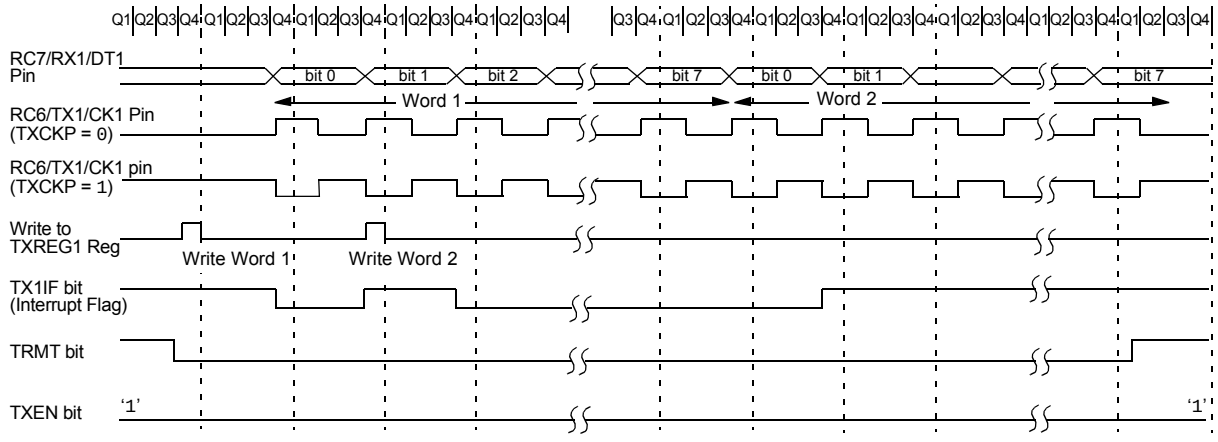
Once the TXREG1 register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG1 is empty and the TX1IF flag bit (PIR1<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX1IE (PIE1<4>). TX1IF is set regardless of the state of enable 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 (TXSTA1<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 18-11: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG1 = 0; continuous transmission of two 8-bit words.

FIGURE 18-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

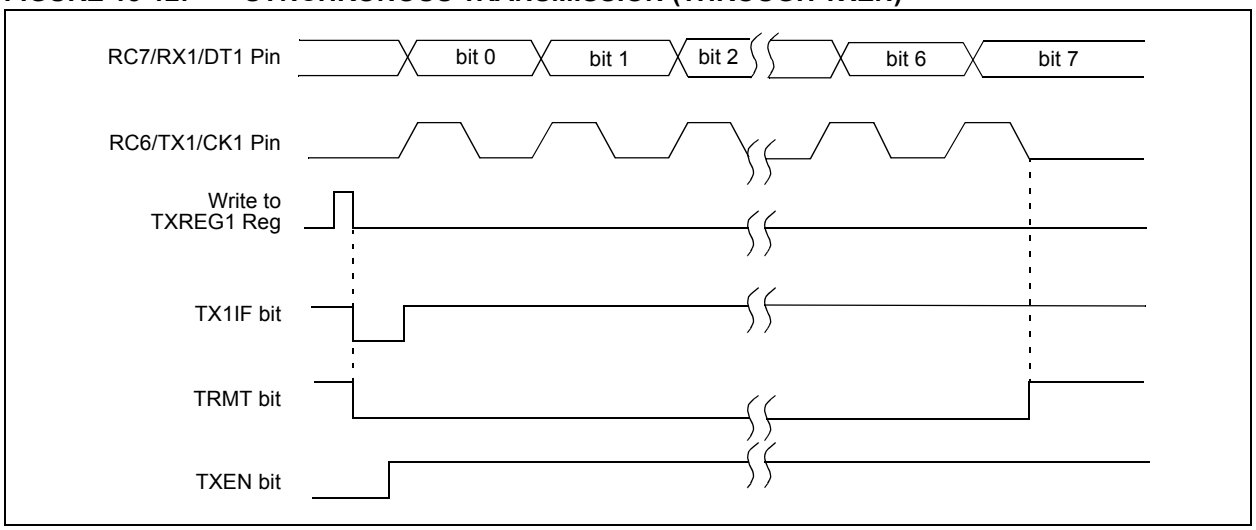


TABLE 18-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
TXREG1	EUSART Transmit Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

18.4.2 EUSART SYNCHRONOUS MASTER RECEPTION

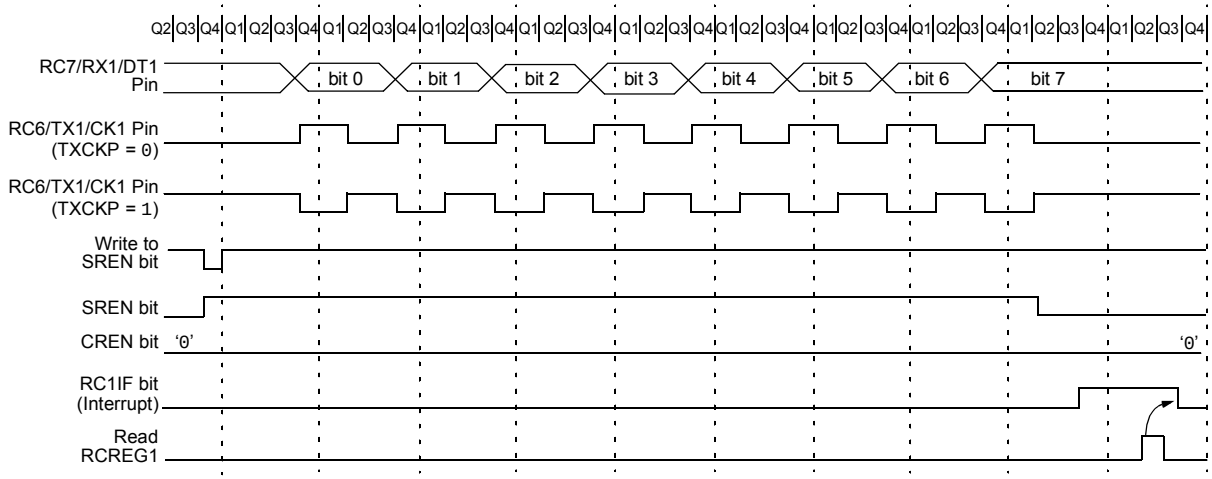
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA1<5>), or the Continuous Receive Enable bit, CREN (RCSTA1<4>). Data is sampled on the RX1 pin on the falling edge of the clock.

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 18-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

TABLE 18-8: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
RCREG1	EUSART Receive Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

18.5 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the 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.

18.5.1 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:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREG1 register.
c) Flag bit, TX1IF, will not be set.
d) 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.
e) 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:

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, TX1IE.
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 TXREG1 register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 18-9: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
TXREG1	EUSART Transmit Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

18.5.2 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 18-10: 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	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
RCREG1	EUSART Receive Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	60
SPBRGH1	EUSART Baud Rate Generator Register High Byte								60
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

NOTES:

19.0 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/J2602 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 and RG2/RX2/DT2, 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

Note:	The AUSART control will automatically reconfigure the pin from input to output as needed.

The driver for the TX2 output pin can also be optionally configured as an open-drain output. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, and allows the output to communicate with external circuits without the need for additional level shifters.

The open-drain output option is controlled by the U2OD bit (LATG<7>). Setting this bit configures the pin for open-drain operation.

19.1 Control Registers

The operation of the Addressable USART module is controlled through two registers, TXSTA2 and RCSTA2. These are detailed in Register 19-1 and Register 19-2, respectively.

REGISTER 19-1: 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(1)	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(1)
1 = Transmit is enabled
0 = Transmit is 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 is empty
0 = TSR is full

bit 0 **TX9D:** 9th Bit of Transmit Data
Can be address/data bit or parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode.

REGISTER 19-2: 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 is enabled (configures RX2/DT2 and TX2/CK2 (TXEN = 1) pins as serial port pins)
0 = Serial port is 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 a parity bit
Asynchronous mode 9-Bit (RX9 = 0):
Don't care.

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be cleared by reading the RCREGx register and receiving the next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit
1 = Overrun error (can be cleared by clearing the CREN bit)
0 = No overrun error

bit 0 **RX9D:** 9th Bit of Received Data
This can be an address/data bit or a parity bit and must be calculated by user firmware.

19.2 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 19-1 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 19-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 19-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 19-2. It may be advantageous 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.

19.2.1 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.

19.2.2 SAMPLING

The data on the RX2 pin is sampled three times by a majority detect circuit to determine if a high or low level is present at the RX2 pin.

TABLE 19-1: BAUD RATE FORMULAS

Configuration Bits		BRG/AUSART Mode	Baud Rate Formula
SYNC	BRGH		
0	0	Asynchronous	FOSC/[64 (n + 1)]
0	1	Asynchronous	FOSC/[16 (n + 1)]
1	x	Synchronous	FOSC/[4 (n + 1)]

Legend: x = Don't care, n = Value of SPBRG2 register

EXAMPLE 19-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, BRGH = 0:

Desired Baud Rate = FOSC/(64 ([SPBRG2] + 1))

Solving for SPBRG2:

X = ((FOSC/Desired Baud Rate)/64) – 1

= ((16000000/9600)/64) – 1

= [25.042] = 25

Calculated Baud Rate = 16000000/(64 (25 + 1))

= 9615

Error = (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate

= (9615 – 9600)/9600 = 0.16%

TABLE 19-2: 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	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61

Legend: — = unimplemented locations read as '0'. Shaded cells are not used by the BRG.

TABLE 19-3: 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	—	—	—	—	—	—

19.3 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

19.3.1 AUSART ASYNCHRONOUS TRANSMITTER

The AUSART transmitter block diagram is shown in Figure 19-1. 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 the SYNC bit and setting the SPEN bit.
3. If interrupts are desired, set enable bit, TX2IE.
4. If 9-bit transmission is desired, set transmit bit, TX9; can be used as an 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 19-1: AUSART TRANSMIT BLOCK DIAGRAM

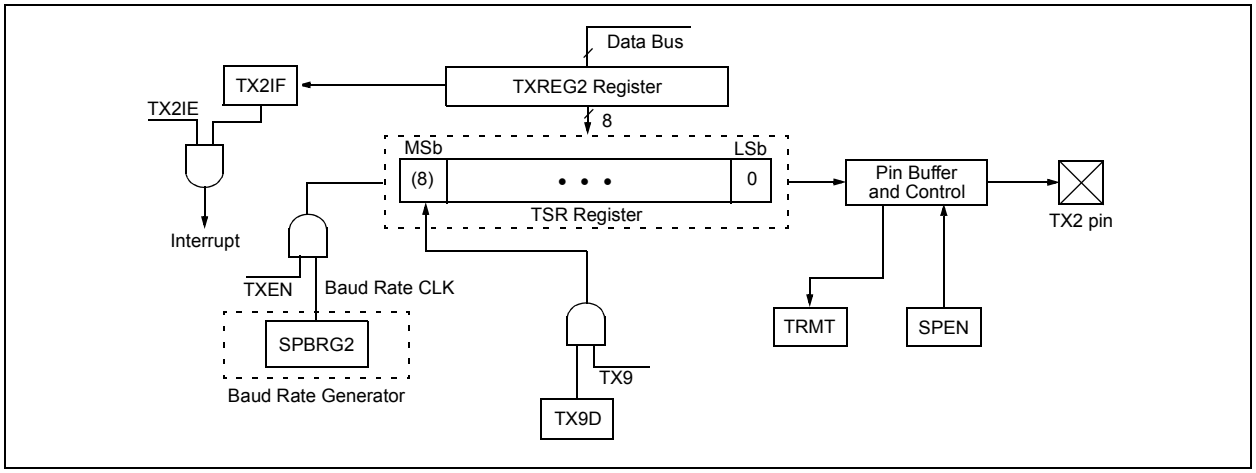


FIGURE 19-2: ASYNCHRONOUS TRANSMISSION

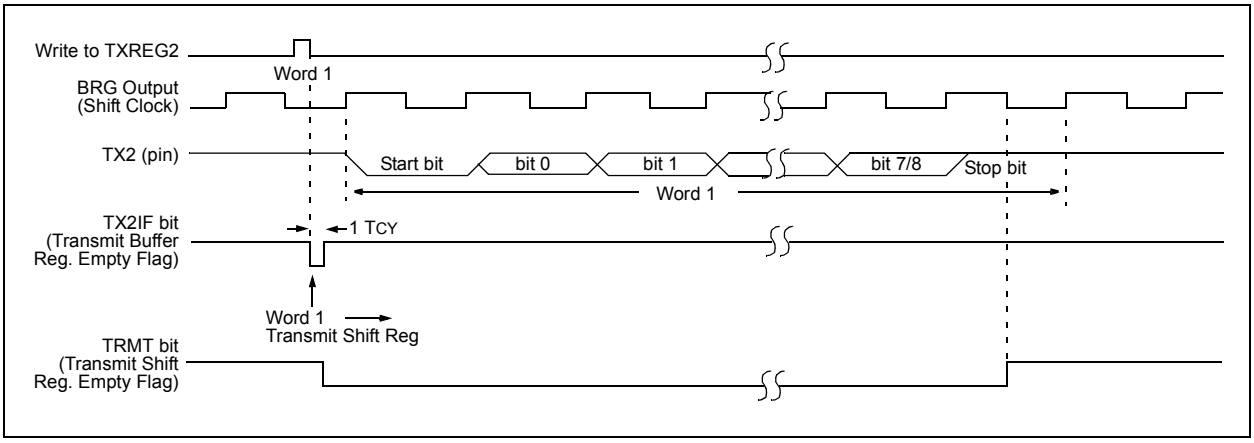
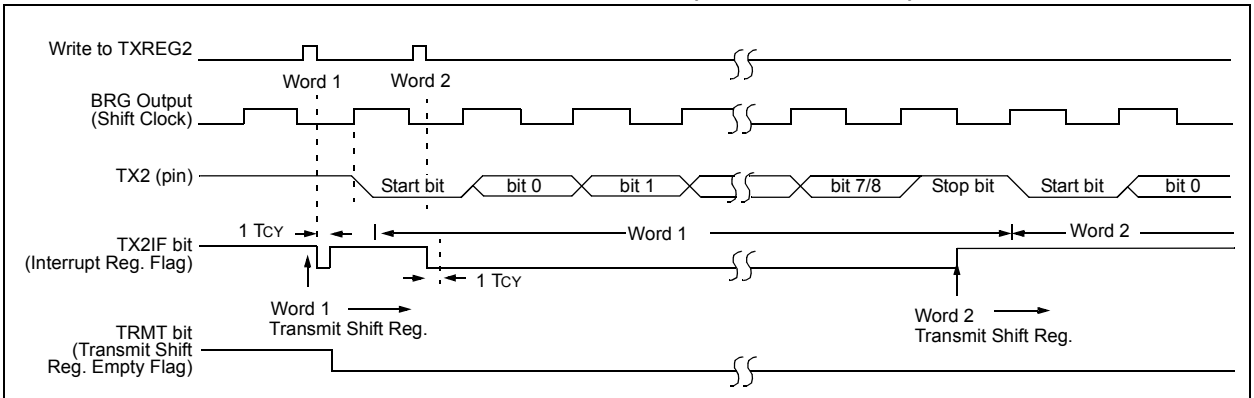


FIGURE 19-3: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 19-4: 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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG2	AUSART Transmit Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

19.3.2 AUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 19-4. 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.

19.3.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the 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 19-4: AUSART RECEIVE BLOCK DIAGRAM

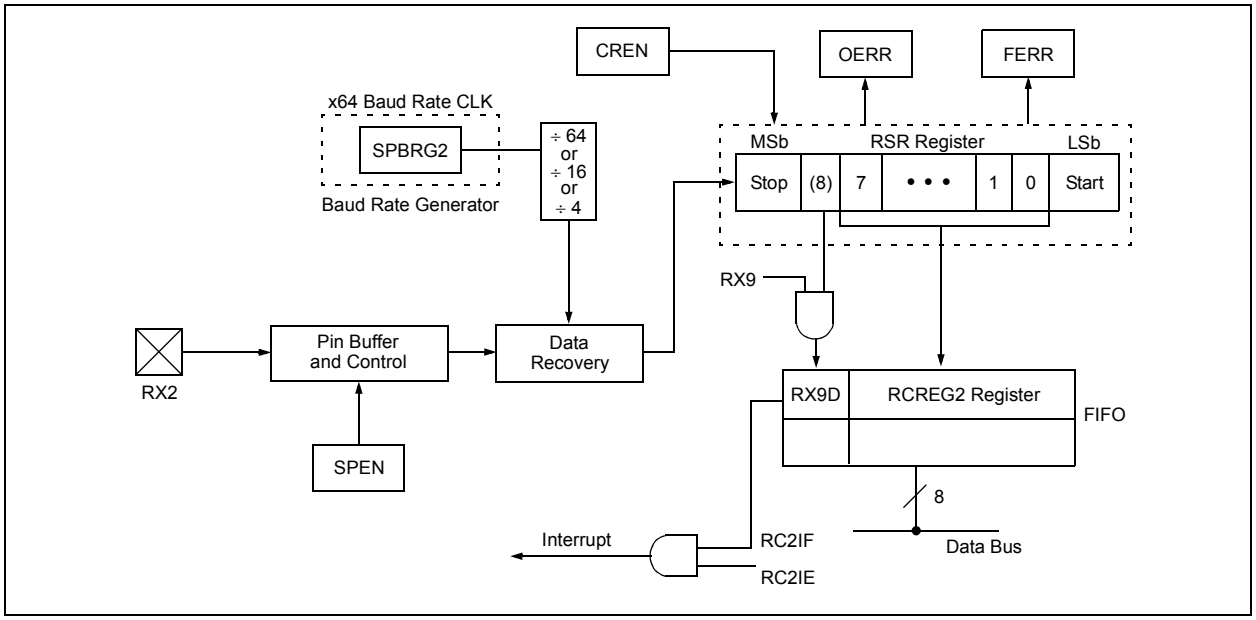
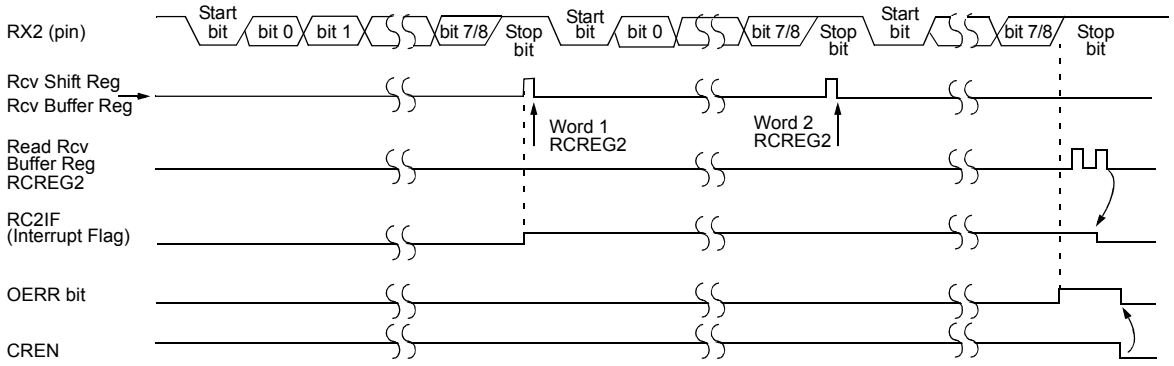


FIGURE 19-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX2 input. The RCREG2 (AUSART Receive register) is read after the third word, causing the OERR (Overrun Error) bit to be set.

TABLE 19-5: 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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG2	AUSART Receive Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

19.4 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.

19.4.1 AUSART SYNCHRONOUS MASTER TRANSMISSION

The AUSART transmitter block diagram is shown in Figure 19-1. 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).

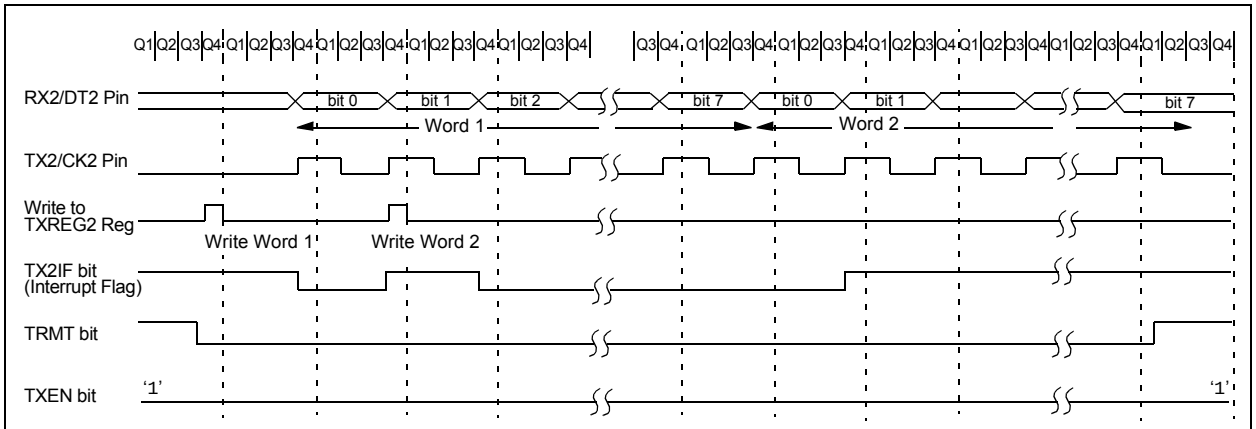
Once the TXREG2 register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG2 is empty and the TX2IF flag bit (PIR3<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TX2IE (PIE3<4>). TX2IF is set regardless of the state of enable 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 19-6: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG2 = 0; continuous transmission of two 8-bit words.

FIGURE 19-7: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

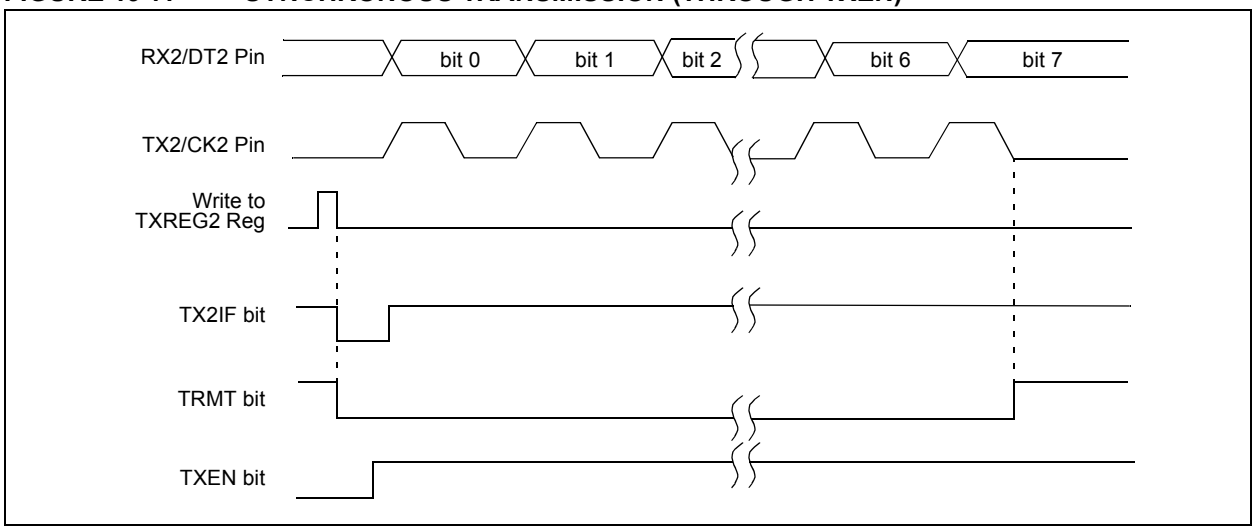


TABLE 19-6: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG2	AUSART Transmit Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

19.4.2 AUSART SYNCHRONOUS MASTER RECEPTION

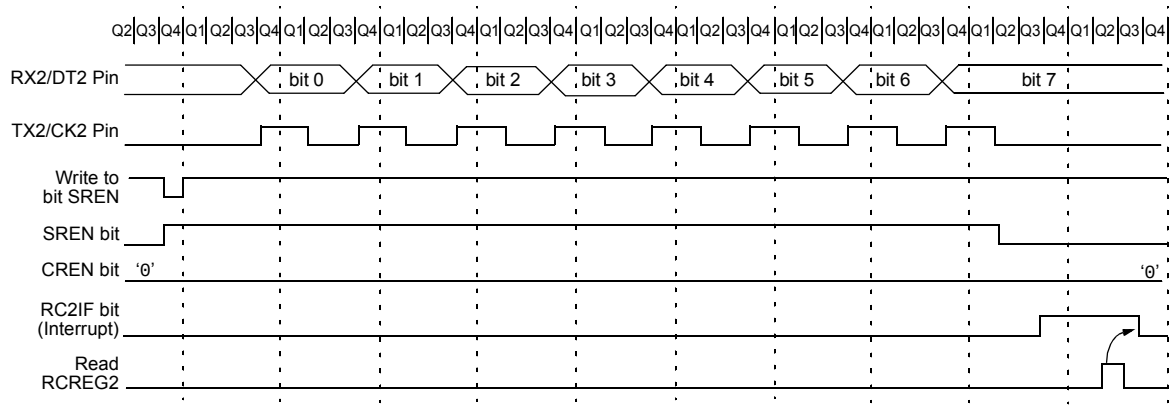
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA2<5>), or the Continuous Receive Enable bit, CREN (RCSTA2<4>). Data is sampled on the RX2 pin on the falling edge of the clock.

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 19-8: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

TABLE 19-7: 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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG2	AUSART Receive Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

19.5 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.

19.5.1 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 19-8: 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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG2	AUSART Transmit Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

19.5.2 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 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, 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 19-9: 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	57
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG2	AUSART Receive Register								61
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	61
SPBRG2	AUSART Baud Rate Generator Register								61

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

20.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has 12 inputs for all PIC18F85J11 family 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 Register High Byte (ADRESH)
- A/D Result Register Low Byte (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 20-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 20-2, configures the functions of the port pins. The ADCON2 register, shown in Register 20-3, configures the A/D clock source, programmed acquisition time and justification.

REGISTER 20-1: ADCON0: A/D CONTROL REGISTER 0

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	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 **ADCAL:** A/D Calibration bit
1 = Calibration is performed on the next A/D conversion
0 = Normal A/D Converter operation (no calibration is performed)

bit 6 **Unimplemented:** Read as ‘0’

bit 5-2 **CHS<3:0>:** Analog Channel Select bits
0000 = Channel 00 (AN0)
0001 = Channel 01 (AN1)
0010 = Channel 02 (AN2)
0011 = Channel 03 (AN3)
0100 = Channel 04 (AN4)
0101 = Channel 05 (AN5)
0110 = Channel 06 (AN6)
0111 = Channel 07 (AN7)
1000 = Channel 08 (AN8)
1001 = Channel 09 (AN9)
1010 = Channel 10 (AN10)
1011 = Channel 11 (AN11)
11xx = Unused

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion is in progress
0 = A/D is Idle

bit 0 **ADON:** A/D On bit
1 = A/D Converter module is enabled
0 = A/D Converter module is disabled

REGISTER 20-2: ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	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 (V_{REF-} source)
1 = V_{REF-} (AN2)
0 = AV_{SS}

bit 4 **VCFG0:** Voltage Reference Configuration bit (V_{REF+} source)
1 = V_{REF+} (AN3)
0 = AV_{DD}

bit 3-0 **PCFG<3:0>:** A/D Port Configuration Control bits:

PCFG<3:0>	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

REGISTER 20-3: 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 **ACQT<2:0>:** A/D Acquisition Time Select bits
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits
111 = FRC (clock derived from A/D RC oscillator)(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock derived from A/D RC oscillator)(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

Note 1: If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

The analog reference voltage is software selectable to either the device's positive and negative supply voltage (AV_{DD} and AV_{SS}), or the voltage level on the RA3/AN3/V_{REF}+ and RA2/AN2/V_{REF}- pins.

The A/D Converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

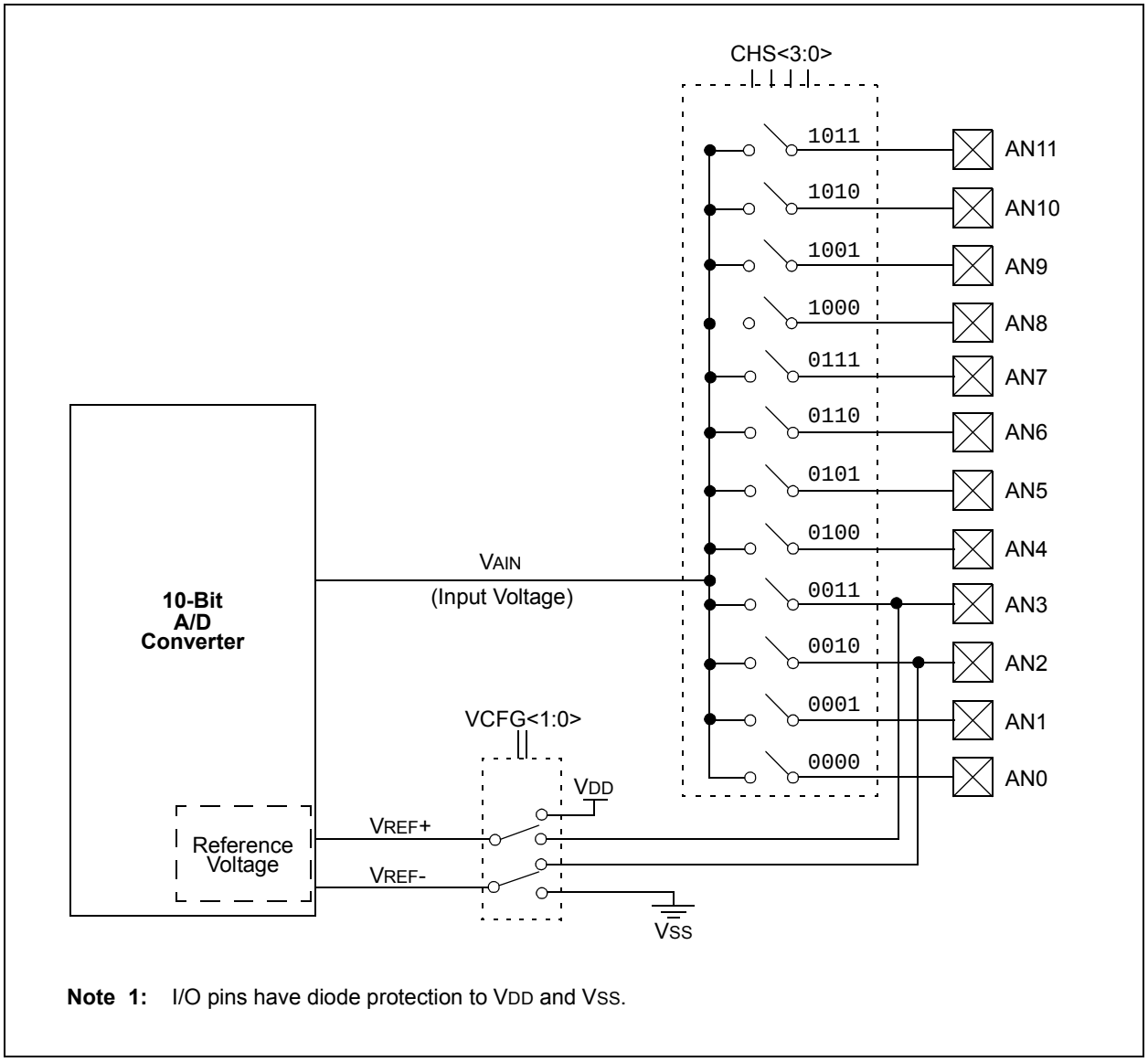
The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

Each port pin associated with the A/D Converter can be configured as an analog input or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH:ADRESL register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<1>) is cleared and the A/D Interrupt Flag bit, ADIF, is set.

A device Reset forces all registers to their Reset state. This forces the A/D module to be turned off and any conversion in progress is aborted. The value in the ADRESH:ADRESL register pair is not modified for a Power-on Reset. These registers will contain unknown data after a Power-on Reset.

The block diagram of the A/D module is shown in Figure 20-1.

FIGURE 20-1: A/D BLOCK DIAGRAM(1)



Note 1: I/O pins have diode protection to V_{DD} and V_{SS}.

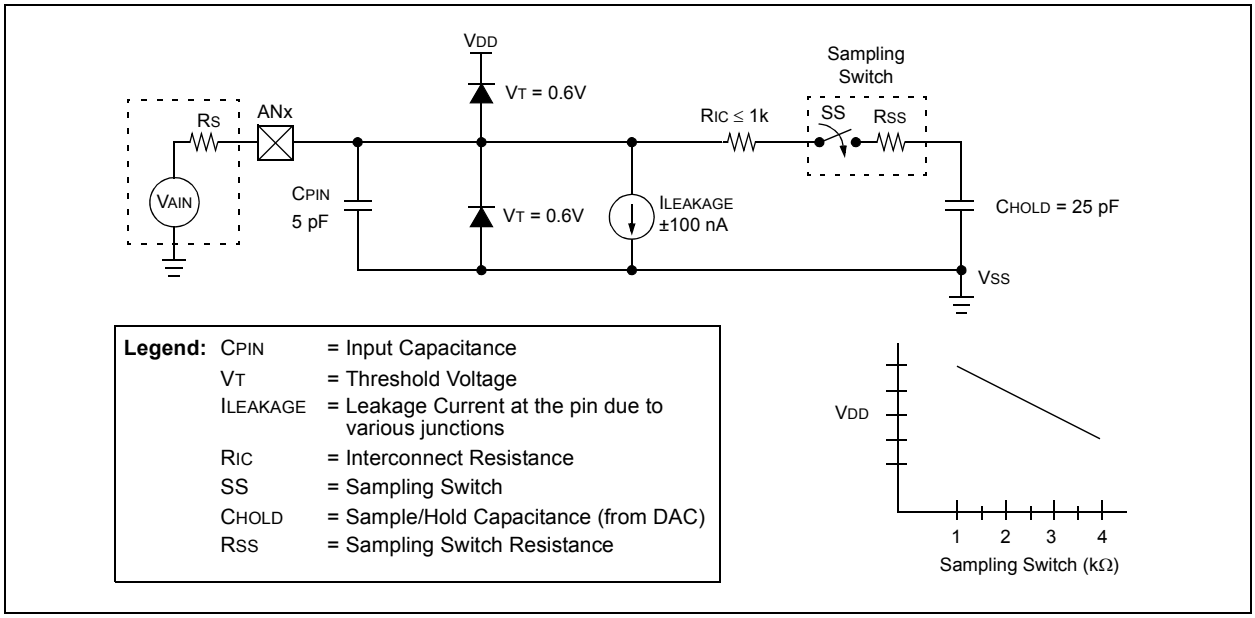
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs. To determine acquisition time, see **Section 20.1 "A/D Acquisition Requirements"**. After this acquisition time has elapsed, the A/D conversion can be started. An acquisition time can be programmed to occur between setting the GO/$\overline{\text{DONE}}$ bit and the actual start of the conversion.

The following steps should be followed to do an A/D conversion:

1. Configure the A/D module:
   - Configure analog pins, voltage reference and digital I/O (ADCON1)
   - Select the A/D input channel (ADCON0)
   - Select the A/D acquisition time (ADCON2)
   - Select the A/D conversion clock (ADCON2)
   - Turn on the A/D module (ADCON0)
2. Configure the A/D interrupt (if desired):
   - Clear the ADIF bit
   - Set the ADIE bit
   - Set the GIE bit
3. Wait the required acquisition time (if required).
4. Start the conversion:
   - Set the GO/$\overline{\text{DONE}}$ bit (ADCON0<1>)
5. Wait for the A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR
   - Waiting for the A/D interrupt
6. Read the A/D Result registers (ADRESH:ADRESL); clear the ADIF bit, if required.
7. For the next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2 TAD is required before the next acquisition starts.

FIGURE 20-2: ANALOG INPUT MODEL



20.1 A/D Acquisition Requirements

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 20-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 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 20-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Equation 20-3 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	=	3V → Rss = 2 kΩ
Temperature	=	85°C (system max.)

EQUATION 20-1: A/D ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 20-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)$$

EQUATION 20-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$
$$T_{AMP} = 0.2\ \mu s$$
$$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$$
$$(85°C - 25°C)(0.02\ \mu s/°C)$$
$$1.2\ \mu s$$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, $T_{COFF} = 0$ ms.

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)\ \mu s$$
$$-(25\ pF)\ (1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega) \ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1\ \mu s + 1.2\ \mu s$$
$$2.4\ \mu s$$

20.2 Selecting and Configuring Automatic Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time the GO/DONE bit is set.

When the GO/DONE bit is set, sampling is stopped and a conversion begins. The user is responsible for ensuring the required acquisition time has passed between selecting the desired input channel and setting the GO/DONE bit. This occurs when the ACQT<2:0> bits (ADCON2<5:3>) remain in their Reset state ('000') and is compatible with devices that do not offer programmable acquisition times.

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When the GO/DONE bit is set, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and setting the GO/DONE bit.

In either case, when the conversion is completed, the GO/DONE bit is cleared, the ADIF flag is set and the A/D begins sampling the currently selected channel again. If an acquisition time is programmed, there is nothing to indicate if the acquisition time has ended or if the conversion has begun.

20.3 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 11 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable.

There are seven possible options for TAD:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal RC Oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be as short as possible but greater than the minimum TAD (see parameter 130 in Table 26-26 for more information).

Table 20-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 20-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency
Operation	ADCS<2:0>	
2 TOSC	000	2.86 MHz
4 TOSC	100	5.71 MHz
8 TOSC	001	11.43 MHz
16 TOSC	101	22.86 MHz
32 TOSC	010	40.0 MHz
64 TOSC	110	40.0 MHz
RC(1)	x11	1.00 MHz(2)

Note 1: The RC source has a typical TAD time of 4 μs.

2: For device frequencies above 1 MHz, the device must be in Sleep mode for the entire conversion or the A/D accuracy may be out of specification.

20.4 Configuring Analog Port Pins

The ADCON1, TRISA, TRISF and TRISH registers control the operation of the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS<3:0> bits and the TRIS bits.

Note 1: When reading the PORT register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will be accurately converted.

2: Analog levels on any pin defined as a digital input may cause the digital input buffer to consume current out of the device's specification limits.

20.5 A/D Conversions

Figure 20-3 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set and the ACQT<2:0> bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins.

Figure 20-4 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set, the ACQT<2:0> bits have been set to '010' and a 4 TAD acquisition time has been selected before the conversion starts.

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D Result register pair will NOT be updated with the partially completed A/D conversion sample. This means that the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers).

After the A/D conversion is completed or aborted, a 2 TAD Wait is required before the next acquisition can be started. After this Wait, acquisition on the selected channel is automatically started.

Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

20.6 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 CCP2M<3:0> bits (CCP2CON<3:0>) be programmed as '1011' and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit will be set, starting the A/D acquisition and conversion, and the Timer1 (or Timer3) counter will be reset to zero. Timer1 (or Timer3) is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving ADRESH/ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition period is either timed by the user, or an appropriate TACQ time is selected before the Special Event Trigger sets the GO/$\overline{\text{DONE}}$ bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), the Special Event Trigger will be ignored by the A/D module but will still reset the Timer1 (or Timer3) counter.

FIGURE 20-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)

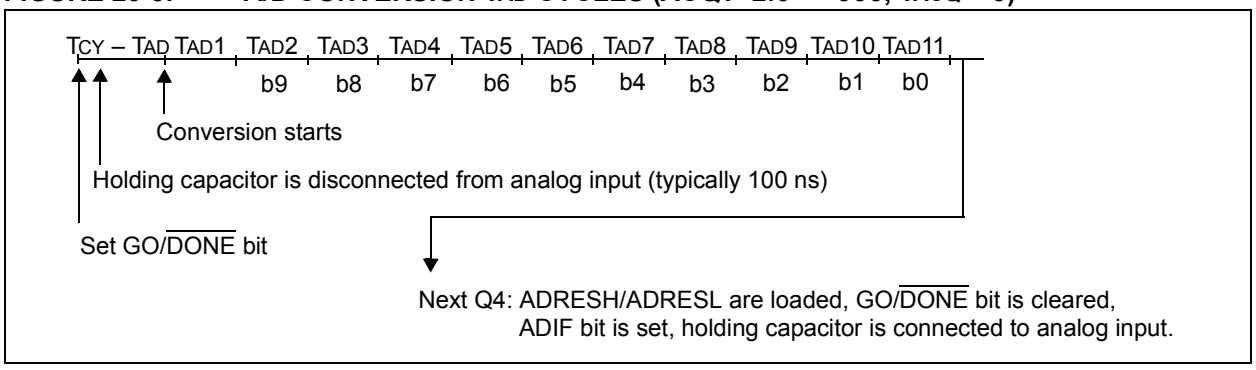
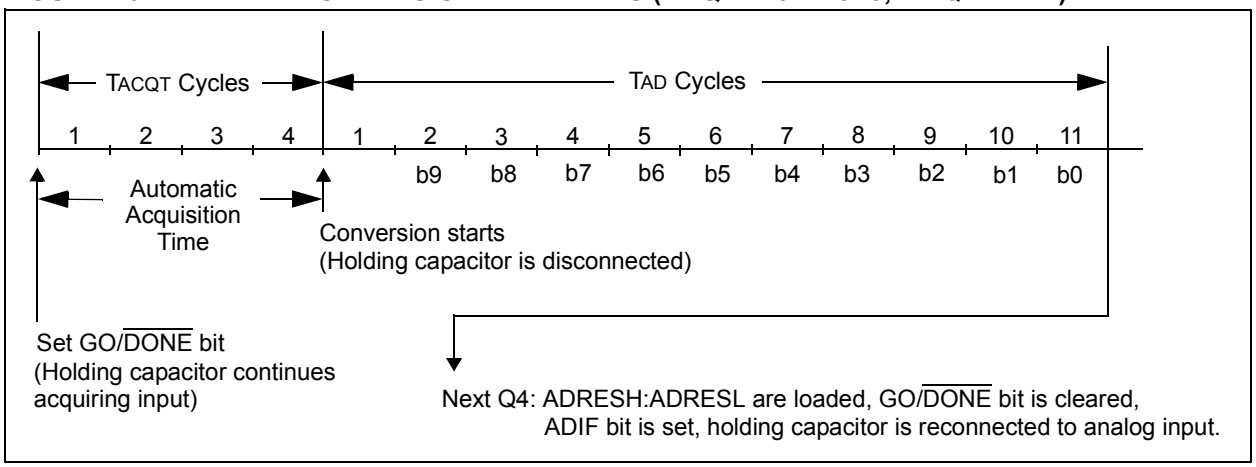


FIGURE 20-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)



20.7 A/D Converter Calibration

The A/D Converter in the PIC18F85J11 family of devices includes a self-calibration feature which compensates for any offset generated within the module. The calibration process is automated and is initiated by setting the ADCAL bit (ADCON0<7>). The next time the GO/$\overline{\text{DONE}}$ bit is set, the module will perform a "dummy" conversion (that is, with reading none of the input channels) and store the resulting value internally to compensate for offset. Thus, subsequent offsets will be compensated.

The calibration process assumes that the device is in a relatively steady-state operating condition. If A/D calibration is used, it should be performed after each device Reset or if there are other major changes in operating conditions.

20.8 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined in part by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT<2:0> and ADCS<2:0> bits in ADCON2 should be updated in accordance with the power-managed mode clock that will be used. After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power-managed Idle mode during the conversion.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in the Sleep mode requires the A/D RC clock to be selected. If bits, ACQT<2:0>, are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS<1:0> bits in the OSCCON register must have already been cleared prior to starting the conversion.

TABLE 20-2: SUMMARY OF A/D REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	59
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	59
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	59
PIR3	—	—	RC2IF	TX2IF	—	CCP2IF	CCP1IF	—	59
PIE3	—	—	RC2IE	TX2IE	—	CCP2IE	CCP1IE	—	59
IPR3	—	—	RC2IP	TX2IP	—	CCP2IP	CCP1IP	—	59
ADRESH	A/D Result Register High Byte								59
ADRESL	A/D Result Register Low Byte								59
ADCON0	ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	59
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	59
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	59
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	60
PORTA	RA7(1)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	60
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	60
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	60
TRISF	TRISF5	TRISF4	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: RA6/RA7 and their associated latch and direction bits are configured as port pins only when the internal oscillator is selected as the default clock source (FOSC2 Configuration bit = 0); otherwise, they are disabled and these bits read as '0'.

NOTES:

21.0 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 RF1 through RF6, as well as the on-chip voltage reference (see **Section 22.0 "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 21-1) selects the comparator input and output configuration. Block diagrams of the various comparator configurations are shown in Figure 21-1.

REGISTER 21-1: CMCON: COMPARATOR MODULE 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 is inverted
0 = C2 output is not inverted

bit 4 **C1INV**: Comparator 1 Output Inversion bit
1 = C1 output is inverted
0 = C1 output is not inverted

bit 3 **CIS**: Comparator Input Switch bit
When CM<2:0> = 110:
1 = C1 VIN- connects to RF5/AN10/CVREF,
C2 VIN- connects to RF3/AN8
0 = C1 VIN- connects to RF6/AN11,
C2 VIN- connects to RF4/AN9

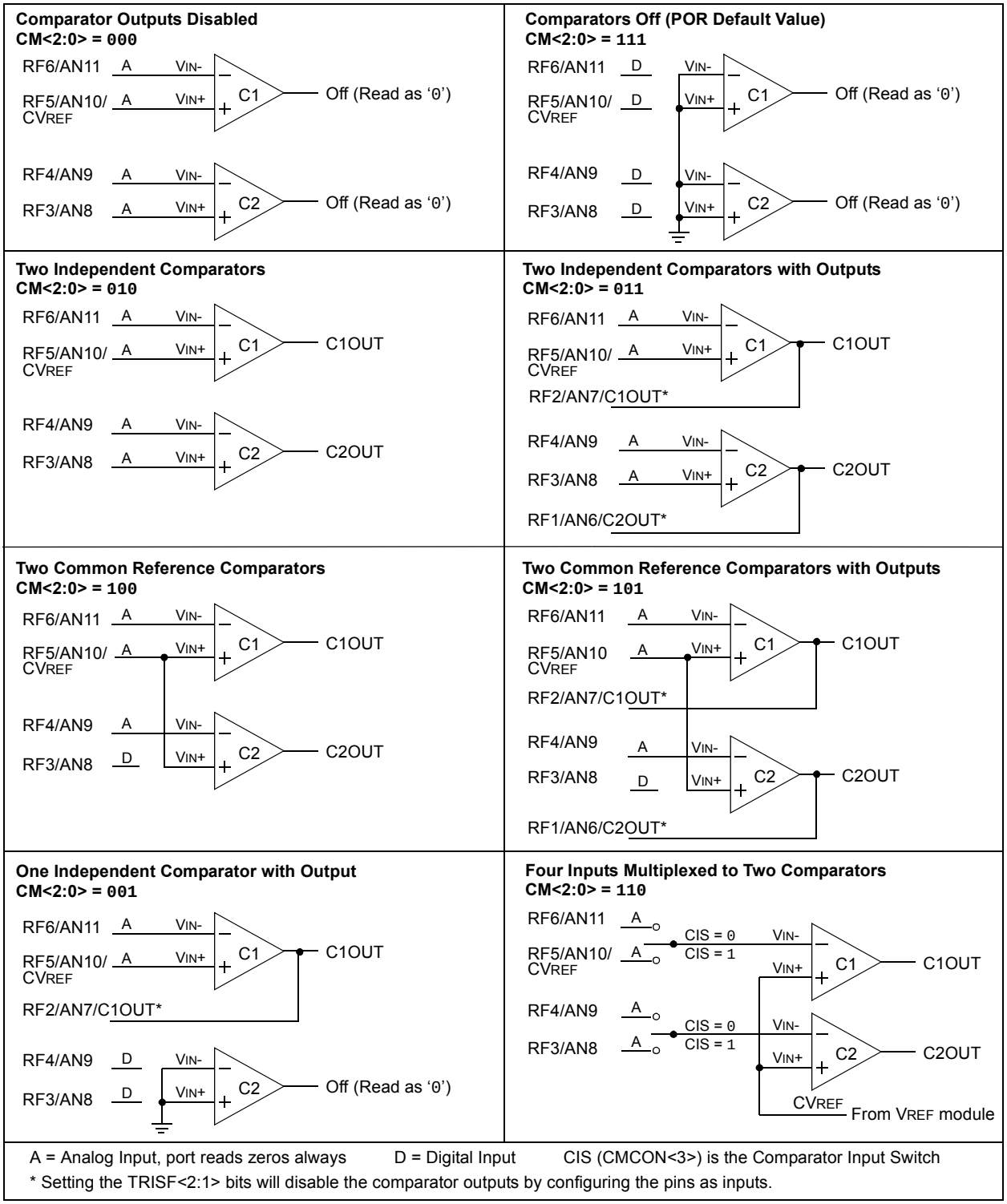
bit 2-0 **CM<2:0>**: Comparator Mode bits
Figure 21-1 shows the Comparator modes and the CM<2:0> bit settings.

21.1 Comparator Configuration

There are eight modes of operation for the comparators, shown in Figure 21-1. Bits, CM<2:0> 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 21-1: COMPARATOR I/O OPERATING MODES



A = Analog Input, port reads zeros always D = Digital Input CIS (CMCON<3>) is the Comparator Input Switch

* Setting the TRISF<2:1> bits will disable the comparator outputs by configuring the pins as inputs.

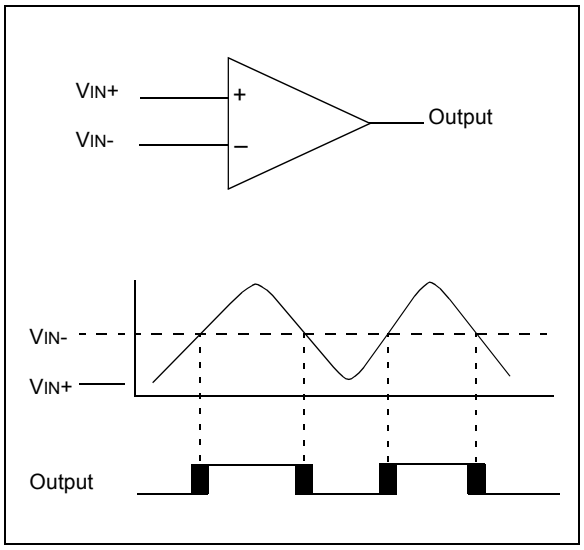
21.2 Comparator Operation

A single comparator is shown in Figure 21-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 21-2 represent the uncertainty due to input offsets and response time.

21.3 Comparator Reference

Depending on the comparator operating mode, either an external or internal voltage reference may be used. The analog signal present at VIN- is compared to the signal at VIN+ and the digital output of the comparator is adjusted accordingly (Figure 21-2).

FIGURE 21-2: SINGLE COMPARATOR



21.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD and can be applied to either pin of the comparator(s).

21.3.2 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 22.0 "Comparator Voltage Reference Module"**.

The internal reference is only available in the mode where four inputs are multiplexed to two comparators (CM<2:0> = 110). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

21.4 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"**).

21.5 Comparator Outputs

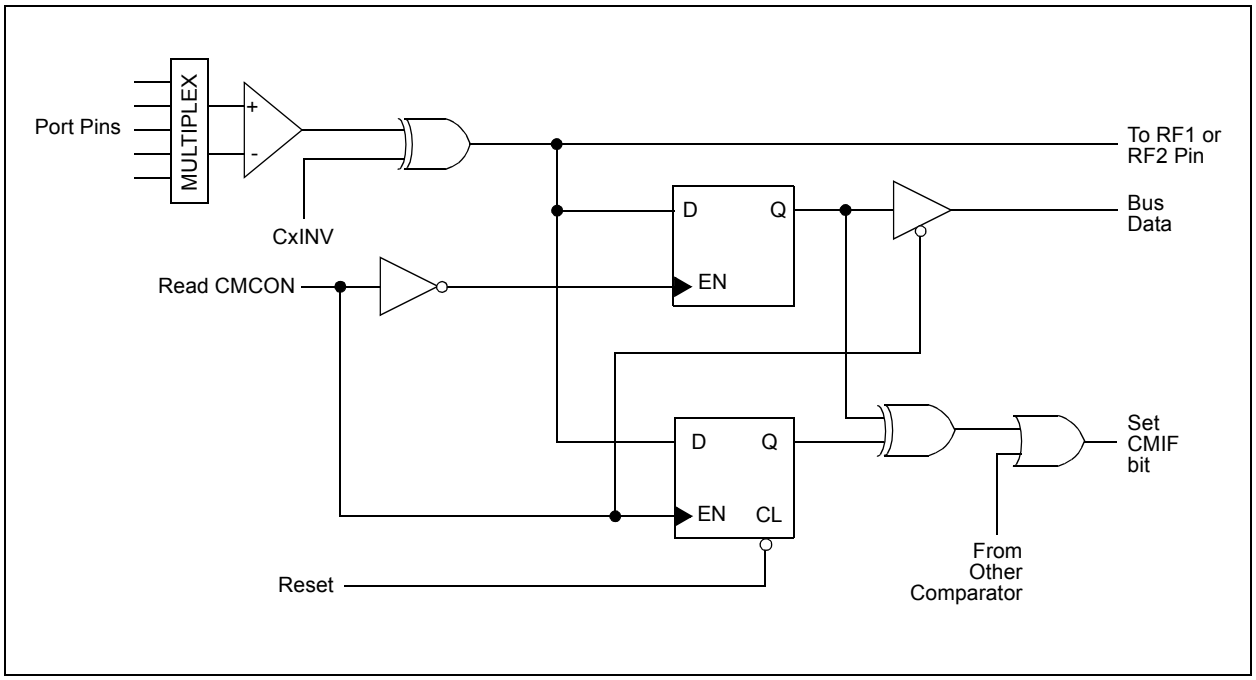
The comparator outputs are read through the CMCON register. These bits are read-only. The comparator outputs may also be directly output to the RF1 and RF2 I/O pins. When enabled, multiplexors in the output path of the RF1 and RF2 pins will switch and the output of each pin will be the unsynchronized 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 21-3 shows the comparator output block diagram.

The TRISF bits will still function as output enables/disables for the RF1 and RF2 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 '0'. Pins configured as digital inputs will convert an analog input according to the Schmitt Trigger input specification.
2:	Analog levels on any pin defined as a digital input may cause the input buffer to consume more current than is specified.

FIGURE 21-3: COMPARATOR OUTPUT BLOCK DIAGRAM



21.6 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:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit, CMIF.

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

21.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from Sleep mode when enabled. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators (CM<2:0> = 111) before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

21.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator modules to be turned off (CM<2:0> = 111). However, the input pins (RF3 through RF6) are configured as analog inputs by default on device Reset. The I/O configuration for these pins is determined by the setting of the PCFG<3:0> bits (ADCON1<3:0>). Therefore, device current is minimized when analog inputs are present at Reset time.

21.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 21-4. 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 21-4: COMPARATOR ANALOG INPUT MODEL

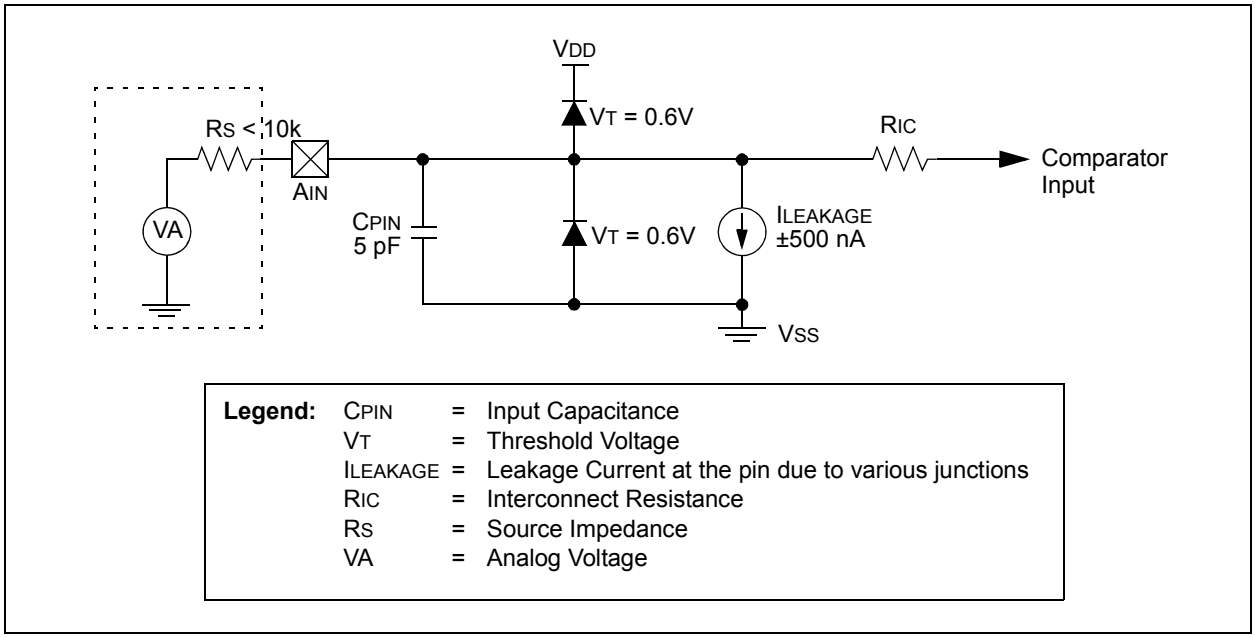


TABLE 21-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR2	OSCFIF	CMIF	—	—	BCLIF	LVDIF	TMR3IF	—	59
PIE2	OSCFIE	CMIE	—	—	BCLIE	LVDIE	TMR3IE	—	59
IPR2	OSCFIP	CMIP	—	—	BCLIP	LVDIP	TMR3IP	—	59
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	59
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	59
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	60
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the comparator module.

NOTES:

22.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable reference voltage. Although its primary purpose is to provide a reference for the analog comparators, it may also be used independently of them.

A block diagram of the module is shown in Figure 22-1. 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.

22.1 Configuring the Comparator Voltage Reference

The comparator voltage reference module is controlled through the CVRCON register (Register 22-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels.

The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR<3:0>), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = ((CVR<3:0>)/24) x (CVRSRC)

If CVRR = 0:
CVREF = (CVRSRC/4) + ((CVR<3:0>)/32) x (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 22-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

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 is powered on
0 = CVREF circuit is powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RF5/AN10/CVREF pin
0 = CVREF voltage is disconnected from the RF5/AN10/CVREF pin

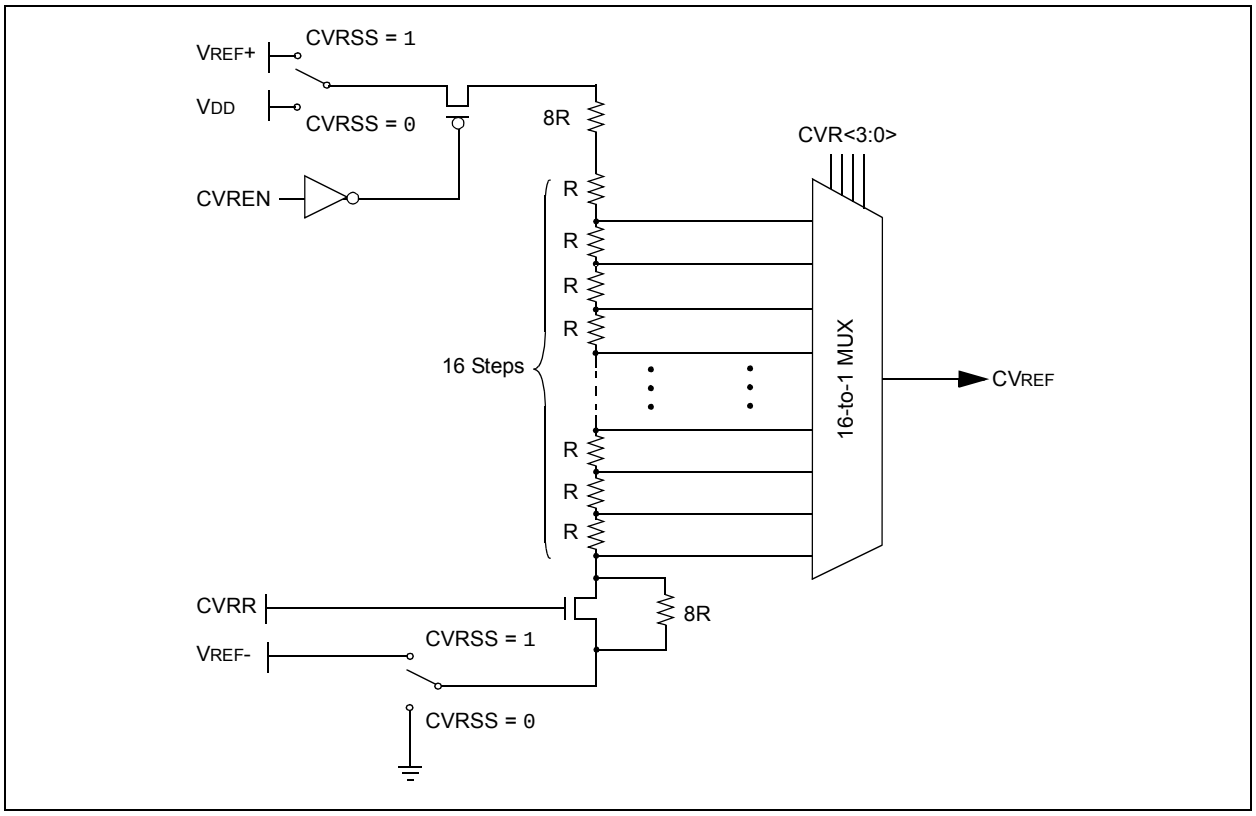
bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)

bit 4 **CVRSS**: Comparator VREF Source Selection bit
1 = Comparator reference source, CVRSRC = (VREF+) – (VREF-)
0 = Comparator reference source, CVRSRC = VDD – VSS

bit 3-0 **CVR<3:0>:** Comparator VREF Value Selection bits ($0 \le$ (CVR<3:0>) ≤ 15)
When CVRR = 1:
CVREF = ((CVR<3:0>)/24) • (CVRSRC)
When CVRR = 0:
CVREF = (CVRSRC/4) + ((CVR<3:0>)/32) • (CVRSRC)

Note 1: CVROE overrides the TRISF<5> bit setting.

FIGURE 22-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



22.2 Comparator Voltage Reference Accuracy/Error

The full range of comparator 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 22-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in **Section 26.0 "Electrical Characteristics"**.

22.3 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 comparator voltage reference should be disabled.

22.4 Effects of a Reset

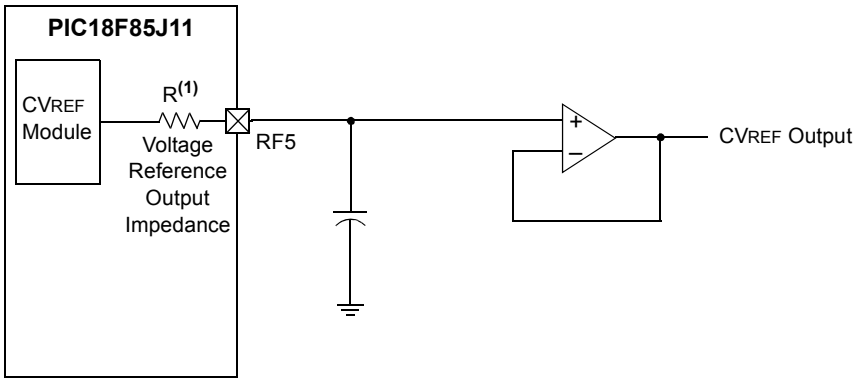
A device Reset disables the comparator voltage reference by clearing bit, CVREN (CVRCON<7>). This Reset also disconnects the reference from the RA2 pin by clearing bit, CVROE (CVRCON<6>) and selects the high-voltage range by clearing bit, CVRR (CVRCON<5>). The CVR<3:0> value select bits are also cleared.

22.5 Connection Considerations

The comparator voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RF5 pin if the CVROE bit is set. Enabling the voltage reference output onto RA2 when it is configured as a digital input will increase current consumption. Connecting 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 comparator voltage reference output for external connections to VREF. Figure 22-2 shows an example buffering technique.

FIGURE 22-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the Comparator Voltage Reference bits, CVRCON<5,3:0>.

TABLE 22-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	59
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	59
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used with the comparator voltage reference.

NOTES:

23.0 SPECIAL FEATURES OF THE CPU

PIC18F85J11 family devices include several features intended to maximize reliability and minimize cost through elimination of external components. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- In-Circuit Serial Programming

The oscillator can be configured for the application depending on frequency, power, accuracy and cost. All of the options are discussed in detail in **Section 3.0 "Oscillator Configurations"**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, the PIC18F85J11 family of devices have a configurable Watchdog Timer which is controlled in software.

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

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. A complete list is shown in Table 23-2. A detailed explanation of the various bit functions is provided in Register 23-1 through Register 23-6.

23.1.1 CONSIDERATIONS FOR CONFIGURING THE PIC18F85J11 FAMILY DEVICES

Unlike some previous PIC18 microcontrollers, devices of the PIC18F85J11 family do not use persistent memory registers to store configuration information. The Configuration registers, CONFIG1L through CONFIG4H, are implemented as volatile memory.

Immediately after power-up, or after a device Reset, the microcontroller hardware automatically loads the CONFIG1L through CONFIG4L registers with configuration data stored in nonvolatile Flash program memory. The last four words of Flash program memory, known as the Flash Configuration Words (FCW), are used to store the configuration data. Table 23-1 provides the Flash program memory, which will be loaded into the corresponding Configuration register.

When creating applications for these devices, users should always specifically allocate the location of the Flash Configuration Word for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The volatile memory cells used for the Configuration bits always reset to '1' on Power-on Resets. For all other types of Reset events, the previously programmed values are maintained and used without reloading from program memory.

The four Most Significant bits of CONFIG1H, CONFIG2H and CONFIG3H in program memory should also be '1111'. This makes these Configuration Words appear to be NOP instructions in the remote event that their locations are ever executed by accident. Since Configuration bits are not implemented in the corresponding locations, writing '1's to these locations has no effect on device operation.

To prevent inadvertent configuration changes during code execution, all programmable Configuration bits are write-once. After a bit is initially programmed during a power cycle, it cannot be written to again. Changing a device configuration requires that power to the device be cycled.

TABLE 23-1: MAPPING OF THE FLASH CONFIGURATION WORDS TO THE CONFIGURATION REGISTERS

Configuration Byte	Code Space Address	Configuration Register Address
CONFIG1L	XXXF8h	300000h
CONFIG1H	XXXF9h	300001h
CONFIG2L	XXXFAh	300002h
CONFIG2H	XXXFBh	300003h
CONFIG3L	XXXFCh	300004h
CONFIG3H	XXXFDh	300005h

TABLE 23-2: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value(1)
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN	111- ---1
300001h	CONFIG1H	—(2)	—(2)	—(2)	—(2)	—(3)	CP0	—	—	---- 01--
300002h	CONFIG2L	IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0	11-- -111
300003h	CONFIG2H	—(2)	—(2)	—(2)	—(2)	WDTPS3	WDTPS2	WDTPS1	WDTPS0	---- 1111
300004h	CONFIG3L(4)	WAIT	BW	EMB1	EMB0	EASHFT	—	—	—	1111 1---
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	—	—	—	CCP2MX	---- ---1
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(5)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 10x1(5)

Legend: x = unknown, — = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: Values reflect the unprogrammed state as received from the factory and following Power-on Resets. In all other Reset states, the configuration bytes maintain their previously programmed states.

2: The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

3: This bit should always be maintained as '0'.

4: CONFIG3L is implemented in 80-pin devices only.

5: See Register 23-7 and Register 23-8 for DEVID values. These registers are read-only and cannot be programmed by the user.

REGISTER 23-1: CONFIG1L: CONFIGURATION REGISTER 1 LOW (BYTE ADDRESS 300000h)

R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0	U-0	R/WO-1
$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7 **$\overline{\text{DEBUG}}$:** Background Debugger Enable bit
1 = Background debugger is disabled; RB6 and RB7 are configured as general purpose I/O pins
0 = Background debugger is 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 are enabled
0 = Instruction set extension and Indexed Addressing mode are disabled (Legacy mode)

bit 5 **STVREN**: Stack Overflow/Underflow Reset Enable bit
1 = Reset on stack overflow/underflow is enabled
0 = Reset on stack overflow/underflow is disabled

bit 4-1 **Unimplemented:** Read as '0'

bit 0 **WDTEN:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled (control is placed on SWDTEN bit)

REGISTER 23-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-0	U-0	U-0	U-0	U-0	R/WO-1	U-0	U-0
—(1)	—(1)	—(1)	—(1)	—(2)	CP0	—	—
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7-4 **Unimplemented:** Read as '1'(1)

bit 3 **Unimplemented:** Read as '0'(2)

bit 2 **CP0:** Code Protection bit
1 = Program memory is not code-protected
0 = Program memory is code-protected

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

Note 1: The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

2: This bit should always be maintained as '0'.

REGISTER 23-3: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)

R/WO-1	R/WO-1	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1
IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7 **IESO:** Two-Speed Start-up (Internal/External Oscillator Switchover) Control bit
1 = Two-Speed Start-up is enabled
0 = Two-Speed Start-up is disabled

bit 6 **FCMEN:** Fail-Safe Clock Monitor Enable bit
1 = Fail-Safe Clock Monitor is enabled
0 = Fail-Safe Clock Monitor is disabled

bit 5-3 **Unimplemented:** Read as '0'

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits
111 = OSC1/OSC2 as primary; EC oscillator with CLKO function and software controlled PLL (ECPLL)
110 = OSC1/OSC2 as primary; EC oscillator with CLKO function (EC)
101 = OSC1/OSC2 as primary; HS oscillator with software controlled PLL (HSPLL)
100 = OSC1/OSC2 as primary; HS oscillator (HS)
011 = INTOSC with CLKO as primary; port function on RA7; EC oscillator with CLKO function and software controlled PLL (ECPLL)
010 = INTOSC with CLKO as primary; port function on RA7; EC oscillator with CLKO function
001 = INTOSC as primary with port function on RA<7:6>; HS oscillator with software controlled PLL (HSPLL)
000 = INTOSC as primary with port function on RA<7:6>; HS oscillator (HS)

REGISTER 23-4: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

U-0	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—(1)	—(1)	—(1)	—(1)	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7-4 **Unimplemented:** Read as '1'(1)

bit 3-0 **WDTPS<3:0>:** Watchdog Timer Postscale Select bits

1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

Note 1: The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

REGISTER 23-5: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)(1)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT	BW	EMB1	EMB0	EASHFT	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7 **WAIT:** External Bus Wait Enable bit
1 = Wait selections from the WAIT<1:0> bits (MEMCON<5:4>) are unavailable and the device will not wait
0 = Wait programmed by the WAIT<1:0> bits (MEMCON<5:4>)

bit 6 **BW:** Data Bus Width Select bit
1 = 16-Bit External Bus mode
0 = 8-Bit External Bus mode

bit 5-4 **EMB<1:0>:** External Memory Bus Configuration bits
00 = Extended Microcontroller mode – 20-Bit Addressing mode
01 = Extended Microcontroller mode – 16-Bit Addressing mode
10 = Extended Microcontroller mode – 12-Bit Addressing mode
11 = Microcontroller mode – external bus disabled

bit 3 **EASHFT:** External Address Bus Shift Enable bit
1 = Address shifting enabled – external address bus is shifted to start at 000000h
0 = Address shifting disabled – external address bus reflects the PC value

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

Note 1: CONFIG3L and its associated bits are implemented only in 80-pin devices.

REGISTER 23-6: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/WO-1
—(1)	—(1)	—(1)	—(1)	—	—	—	CCP2MX
bit 7							bit 0

Legend:		
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	'1' = Bit is set	'0' = Bit is cleared

bit 7-1 **Unimplemented:** Read as '1'(1)

bit 0 **CCP2MX:** CCP2 MUX bit
1 = CCP2 is multiplexed with RC1
0 = CCP2 is multiplexed with RE7 in Microcontroller mode (all devices) or with RB3 in Extended Microcontroller mode (80-pin devices only)

Note 1: The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

REGISTER 23-7: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F85J11 FAMILY DEVICES

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

Legend:
R = Read-only bit

bit 7-5 **DEV<2:0>:** Device ID bits
111 = PIC18F85J11
101 = PIC18F84J11
100 = PIC18F83J11
011 = PIC18F65J11
001 = PIC18F64J11
000 = PIC18F63J11

bit 4-0 **REV<4:0>:** Revision ID bits
These bits are used to indicate the device revision.

REGISTER 23-8: DEVID2: DEVICE ID REGISTER 2 FOR PIC18F85J11 FAMILY DEVICES

R	R	R	R	R	R	R	R
DEV10(1)	DEV9(1)	DEV8(1)	DEV7(1)	DEV6(1)	DEV5(1)	DEV4(1)	DEV3(1)
bit 7							bit 0

Legend:
R = Read-only bit

bit 7-0 **DEV<10:3>:** Device ID bits(1)
These bits are used with the DEV<2:0> bits in the Device ID Register 1 to identify the part number.
0011 1001 = PIC18F6XJ11/8XJ11 devices

Note 1: The values for DEV<10:3> may be shared with other device families. The specific device is always identified by using the entire DEV<10:0> bit sequence.

23.2 Watchdog Timer (WDT)

For PIC18F85J11 family devices, the WDT is driven by the INTRC oscillator. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the INTRC oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexor, controlled by the WDTPS bits in Configuration Register 2H. Available periods range from 4 ms to 131.072 seconds (2.18 minutes). The WDT and postscaler are cleared whenever a SLEEP or CLRWDT instruction is executed, or a clock failure (primary or Timer1 oscillator) has occurred.

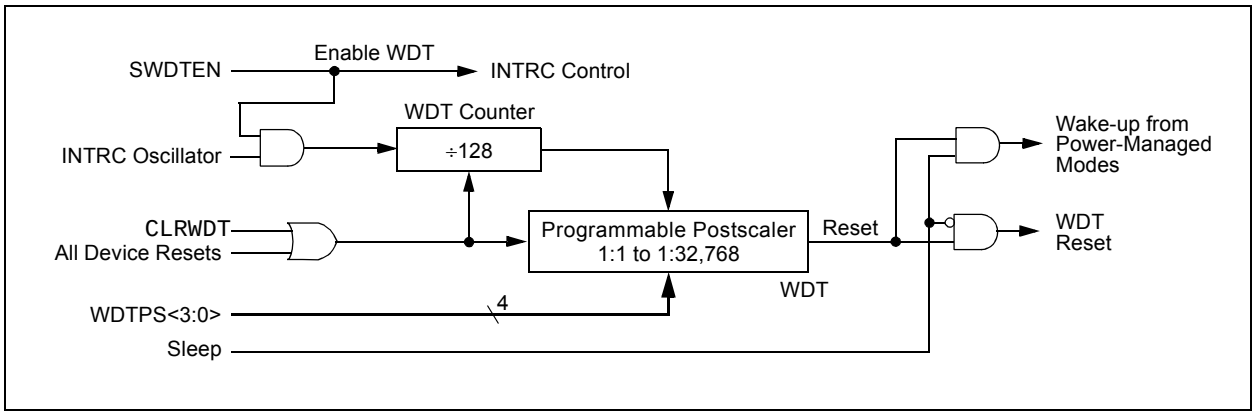
Note 1: The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.

2: When a CLRWDT instruction is executed, the postscaler count will be cleared.

23.2.1 CONTROL REGISTER

The WDTCON register (Register 23-9) is a readable and writable register. The SWDTEN bit enables or disables WDT operation. This allows software to override the WDTEN Configuration bit and enable the WDT only if it has been disabled by the Configuration bit.

FIGURE 23-1: WDT BLOCK DIAGRAM



REGISTER 23-9: WDTCON: WATCHDOG TIMER CONTROL REGISTER

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
REGSLP(1)	—	—	—	—	—	—	SWDTEN(2)
bit 7							bit 0

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

bit 7 **REGSLP:** Voltage Regulator Low-Power Operation Enable bit
1 = On-chip regulator enters low-power operation when device enters Sleep mode
0 = On-chip regulator continues to operate normally in Sleep mode

bit 6-1 **Unimplemented**: Read as '0'

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit(1)
1 = Watchdog Timer is on
0 = Watchdog Timer is off

Note 1: The REGSLP bit is automatically cleared when a Low-Voltage Detect condition occurs.
2: This bit has no effect if the Configuration bit, WDTEN, is enabled.

TABLE 23-3: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	58
WDTCON	REGSLP	—	—	—	—	—	—	SWDTEN	58

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

23.3 On-Chip Voltage Regulator

All of the PIC18F85J11 family devices power their core digital logic at a nominal 2.5V. For designs that are required to operate at a higher typical voltage, such as 3.3V, all devices in the PIC18F85J11 family incorporate an on-chip regulator that allows the device to run its core logic from VDD.

The regulator is controlled by the ENVREG pin. Tying VDD to the pin enables the regulator, which in turn, provides power to the core from the other VDD pins. When the regulator is enabled, a low-ESR filter capacitor must be connected to the VDDCORE/VCAP pin (Figure 23-2). This helps to maintain the stability of the regulator. The recommended value for the filter capacitor is provided in **Section 26.3 "DC Characteristics: PIC18F85J11 Family (Industrial)"**.

If ENVREG is tied to VSS, the regulator is disabled. In this case, separate power for the core logic at a nominal 2.5V must be supplied to the device on the VDDCORE/VCAP pin to run the I/O pins at higher voltage levels, typically 3.3V. Alternatively, the VDDCORE/VCAP and VDD pins can be tied together to operate at a lower nominal voltage. Refer to Figure 23-2 for possible configurations.

23.3.1 VOLTAGE REGULATION AND LOW-VOLTAGE DETECTION

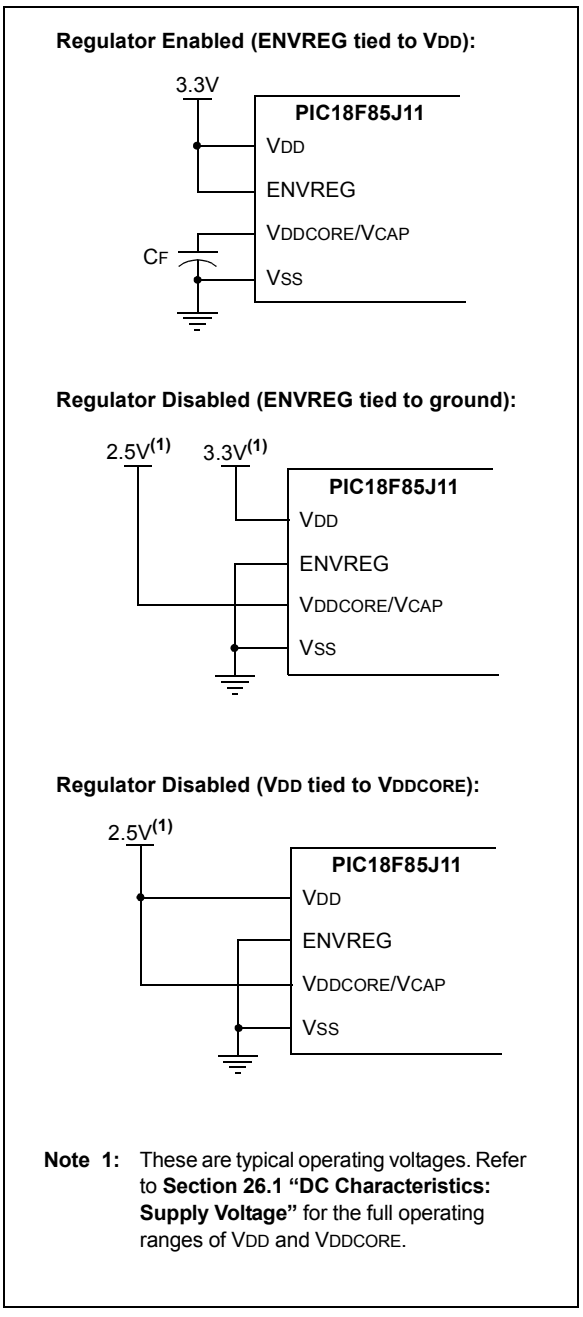
When it is enabled, the on-chip regulator provides a constant voltage of 2.5V nominal to the digital core logic. The regulator can provide this level from a VDD of about 2.5V, all the way up to the device's VDDMAX. It does not have the capability to boost VDD levels below 2.5V.

In order to prevent "brown-out" conditions, the regulator enters Tracking mode when the voltage drops too low for the regulator. In Tracking mode, the regulator output follows VDD, with a typical voltage drop of 100 mV.

The on-chip regulator includes a simple Low-Voltage Detect (LVD) circuit. If VDD drops too low to maintain approximately 2.45V on VDDCORE, the circuit sets the Low-Voltage Detect Interrupt Flag, LVDIF (PIR2<2>), and clears the REGSLP (WDTCON<7>) bit if it was set.

This can be used to generate an interrupt and put the application into a low-power operational mode or to trigger an orderly shutdown. Low-Voltage Detection is only available when the regulator is enabled.

FIGURE 23-2: CONNECTIONS FOR THE ON-CHIP REGULATOR



Note 1: These are typical operating voltages. Refer to **Section 26.1 "DC Characteristics: Supply Voltage"** for the full operating ranges of VDD and VDDCORE.

23.3.2 ON-CHIP REGULATOR AND BOR

When the on-chip regulator is enabled, PIC18F85J11 family devices also have a simple Brown-out Reset capability. If the voltage supplied to the regulator falls to a level that is inadequate to maintain a regulated output for full-speed operation, the regulator Reset circuitry will generate a Brown-out Reset. This event is captured by the $\overline{BOR}$ flag bit (RCON<0>).

The operation of the Brown-out Reset is described in more detail in **Section 5.4 "Brown-out Reset (BOR)"** and **Section 5.4.1 "Detecting BOR"**.

23.3.3 POWER-UP REQUIREMENTS

The on-chip regulator is designed to meet the power-up requirements for the device. If the application does not use the regulator, then strict power-up conditions must be adhered to. While powering up, V_{DDCORE} must never exceed V_{DD} by 0.3 volts.

23.3.4 OPERATION IN SLEEP MODE

When enabled, the on-chip regulator always consumes a small incremental amount of current over I_{DD}. This includes when the device is in Sleep mode, even though the core digital logic does not require power. To provide additional savings in applications where power resources are critical, the regulator can be configured to automatically disable itself whenever the device goes into Sleep mode. This feature is controlled by the REGSLP bit (WDTCON<7>). Setting this bit disables the regulator in Sleep mode and reduces its current consumption to a minimum.

Substantial Sleep-mode power savings can be obtained by setting the REGSLP bit, but this will increase device wake-up time to ensure the regulator has enough time to stabilize.

The REGSLP bit is cleared automatically by hardware when a Low-Voltage Detect condition occurs. The REGSLP bit can be set again in software, which would keep the voltage regulator in Low-Power mode. This is not recommended, however, if any write operations to the Flash will be performed.

23.4 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period, from oscillator start-up to code execution, by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is HS or HSPLL (Crystal-Based) modes. Since the EC and ECPLL modes do not require an Oscillator Start-up Timer delay, Two-Speed Start-up should be disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer, after a Power-on Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

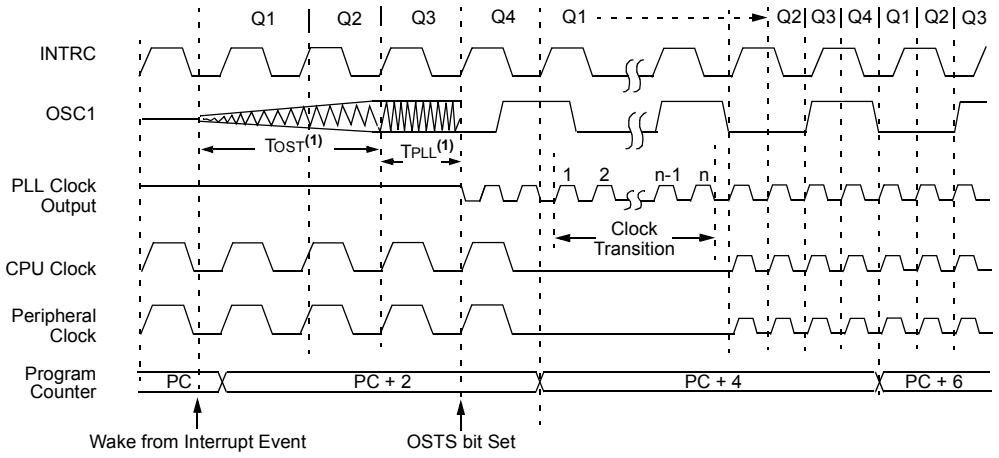
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

23.4.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial SLEEP instructions (refer to **Section 4.1.4 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS<1:0> bit settings or issue SLEEP instructions before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

FIGURE 23-3: TIMING TRANSITION FOR TWO-SPEED START-UP (INTRC TO HSPLL)



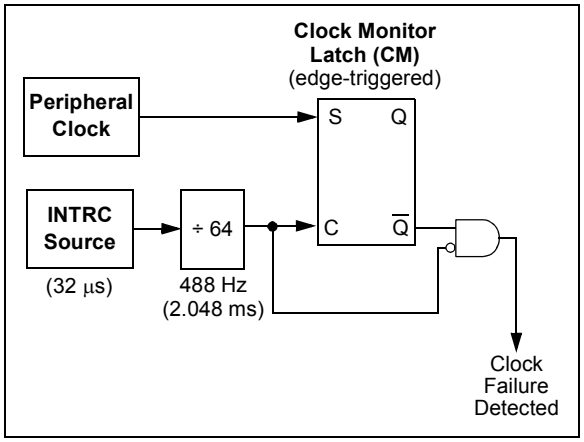
Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

23.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provides a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 23-4) is accomplished by creating a sample clock signal which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the device clock source but cleared on the rising edge of the sample clock.

FIGURE 23-4: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 23-5). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>);
- the device clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition); and
- the WDT is reset.

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing-sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 4.1.4 "Multiple Sleep Commands"** and **Section 23.4.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

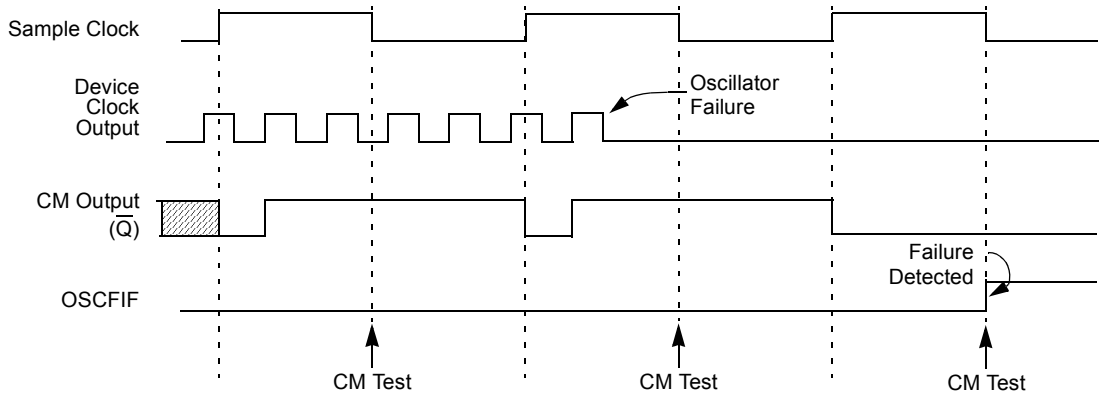
23.5.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTRC clock when a clock failure is detected. This may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

If the interrupt is disabled, subsequent interrupts while in Idle mode will cause the CPU to begin executing instructions while being clocked by the INTRC source.

FIGURE 23-5: FSCM TIMING DIAGRAM



Note: The device clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

23.5.2 EXITING FAIL-SAFE OPERATION

The fail-safe condition is terminated by either a device Reset or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 2H (with any required start-up delays that are required for the oscillator mode, such as OST or PLL timer). The INTRC oscillator provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The 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 multiplexor. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

23.5.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexor selects the clock source selected by the OSCCON register. Fail-Safe 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 is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTRC multiplexor. An automatic transition back to the failed clock source will not occur.

23.5.4 POR OR WAKE-UP FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary device clock is either EC or INTRC mode, monitoring can begin immediately following these events.

For HS or HSPLL modes, the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FSCM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the device clock and functions until the primary clock is stable (the OST and PLL timers have timed out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

Note: The same logic that prevents false oscillator failure interrupts on POR, or wake 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 23.4.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration and enter an alternate power-managed mode while waiting for the primary clock to become stable. When the new power-managed mode is selected, the primary clock is disabled.

23.6 Program Verification and Code Protection

For all devices in the PIC18F85J11 family of devices, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by one Configuration bit, CP0. This bit inhibits external reads and writes to the program memory space. It has no direct effect in normal execution mode.

23.6.1 CONFIGURATION REGISTER PROTECTION

The Configuration registers are protected against untoward changes or reads in two ways. The primary protection is the write-once feature of the Configuration bits which prevents reconfiguration once the bit has been programmed during a power cycle. To safeguard against unpredictable events, Configuration bit changes resulting from individual cell level disruptions (such as ESD events) will cause a parity error and trigger a device Reset.

The data for the Configuration registers is derived from the Flash Configuration Words in program memory. When the CP0 bit is set, the source data for device configuration is also protected as a consequence.

23.7 In-Circuit Serial Programming

PIC18F85J11 family microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

23.8 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ Configuration bit is programmed to '0', the in-circuit debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 23-4 shows which resources are required by the background debugger.

TABLE 23-4: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	512 bytes
Data Memory:	10 bytes

NOTES:

24.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

24.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-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)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either C or assembly)
- One-touch compile or assemble, and download to emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (C or assembly)
  - Mixed C and assembly
  - 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.

24.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

24.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

24.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB 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

24.5 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

24.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB 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 device instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

24.7 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 C Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

24.8 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 emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

24.9 MPLAB ICD 3 In-Circuit Debugger System

MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost effective high-speed hardware debugger/programmer for Microchip Flash Digital Signal Controller (DSC) and microcontroller (MCU) devices. It debugs and programs PIC® Flash microcontrollers and dsPIC® DSCs with the powerful, yet easy-to-use graphical user interface of MPLAB Integrated Development Environment (IDE).

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

24.10 PICkit 3 In-Circuit Debugger/ Programmer and PICkit 3 Debug Express

The MPLAB PICkit 3 allows debugging and programming of PIC® and dsPIC® Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB Integrated Development Environment (IDE). The MPLAB PICkit 3 is connected to the design engineer's PC using a full speed USB interface and can be connected to the target via an Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the reset line to implement in-circuit debugging and In-Circuit Serial Programming™.

The PICkit 3 Debug Express include the PICkit 3, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

24.11 PICkit 2 Development Programmer/Debugger and PICkit 2 Debug Express

The PICkit™ 2 Development Programmer/Debugger is a low-cost development tool with an easy to use interface for programming and debugging Microchip's Flash families of microcontrollers. The full featured Windows® programming interface supports baseline (PIC10F, PIC12F5xx, PIC16F5xx), midrange (PIC12F6xx, PIC16F), PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 families of 8-bit, 16-bit, and 32-bit microcontrollers, and many Microchip Serial EEPROM products. With Microchip's powerful MPLAB Integrated Development Environment (IDE) the PICkit™ 2 enables in-circuit debugging on most PIC® microcontrollers. In-Circuit-Debugging runs, halts and single steps the program while the PIC microcontroller is embedded in the application. When halted at a breakpoint, the file registers can be examined and modified.

The PICkit 2 Debug Express include the PICkit 2, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

24.12 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an MMC card for file storage and data applications.

24.13 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

25.0 INSTRUCTION SET SUMMARY

The PIC18F85J11 family of devices incorporate the standard set of 75 PIC18 core instructions, as well as an extended set of 8 new instructions for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

25.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC® 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 25-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 25-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The File Select Register (specified by ‘f’).
2. The destination of the result (specified by ‘d’).
3. The accessed memory (specified by ‘a’).

The File Select Register designator, ‘f’, specifies which File Select 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 Select Register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The File Select Register (specified by ‘f’).
2. The bit in the File Select 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 Select 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 Select Register (specified by ‘k’).
- The desired FSR register to load the literal value into (specified by ‘f’).
- No operand required (specified by ‘—’).

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’).
- The mode of the CALL or RETURN instructions (specified by ‘s’).
- The mode of the table read and table write instructions (specified by ‘m’).
- No operand required (specified by ‘—’).

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 25-1 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 25-2, lists the standard instructions recognized by the Microchip MPASM™ Assembler.

Section 25.1.1 “Standard Instruction Set” provides a description of each instruction.

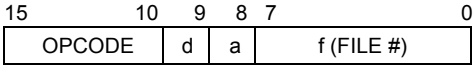
TABLE 25-1: OPCODE FIELD DESCRIPTIONS

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 Select Register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
C, DC, Z, OV, N	ALU STATUS bits: **C**arry, **D**igit **C**arry, **Z**ero, **Ov**erflow, **N**egative.
d	Destination select bit: d = 0: store result in WREG d = 1: store result in File Select Register f
dest	Destination: either the WREG register or the specified register file location.
f	8-bit register file address (00h to FFh) or 2-bit FSR designator (0h to 3h).
f_s	12-bit register file address (000h to FFFh). This is the source address.
f_d	12-bit register file address (000h to FFFh). This is the destination address.
GIE	Global Interrupt Enable bit.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value).
label	Label name.
mm * *+ *- +*	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: No change to register (such as TBLPTR with table reads and writes) Post-Increment register (such as TBLPTR with table reads and writes) Post-Decrement register (such as TBLPTR with table reads and writes) Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions or the direct address for Call/Branch and Return instructions.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.
$\overline{PD}$	Power-Down bit.
PRODH	Product of Multiply High Byte.
PRODL	Product of Multiply Low Byte.
s	Fast Call/Return mode select bit: s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
TBLPTR	21-bit Table Pointer (points to a program memory location).
TABLAT	8-bit Table Latch.
$\overline{TO}$	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_s	7-bit offset value for Indirect Addressing of register files (source).
z_d	7-bit offset value for Indirect Addressing of register files (destination).
{ }	Optional argument.
[text]	Indicates an Indexed Address.
(text)	The contents of text.
[expr]<n>	Specifies bit n of the register indicated by the pointer, expr.
→	Assigned to.
< >	Register bit field.
∈	In the set of.
italics	User-defined term (font is Courier New).

FIGURE 25-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-Oriented File Select Register operations

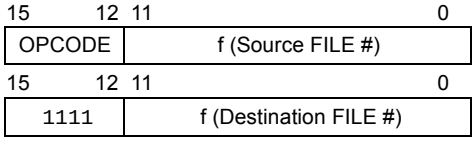
Example Instruction



ADDWF MYREG, W, B

d = 0 for result destination to be WREG register
d = 1 for result destination to be File Select Register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit File Select Register address

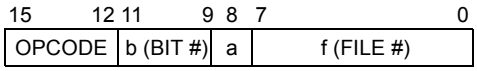
Byte to Byte move operations (2-word)



MOVFF MYREG1, MYREG2

f = 12-bit File Select Register address

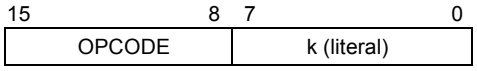
Bit-Oriented File Select Register operations



BSF MYREG, bit, B

b = 3-bit position of bit in File Select Register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit File Select Register address

Literal operations

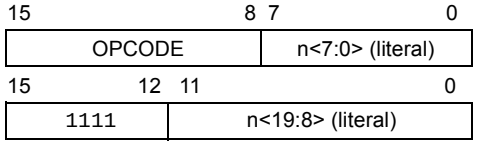


MOVLW 7Fh

k = 8-bit immediate value

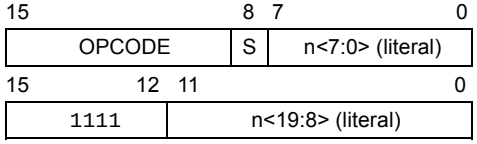
Control operations

CALL, GOTO and Branch operations



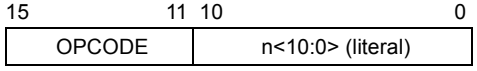
GOTO Label

n = 20-bit immediate value

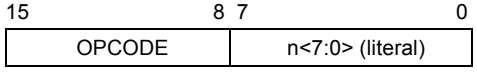


CALL MYFUNC

S = Fast bit



BRA MYFUNC



BC MYFUNC

TABLE 25-2: PIC18F85J11 FAMILY INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-bit Instruction Word				Status Affected	Notes
				MSb			LSb		
BYTE-ORIENTED OPERATIONS									
ADDWF	f, d, a	Add WREG and f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	4
CPFSGT	f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	4
CPFSLT	f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	None	
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	1, 2
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	1, 2
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	1, 2
SUBFWB	f, d, a	Subtract f from WREG with Borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	Subtract WREG from f with Borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap Nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, Skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 25-2: PIC18F85J11 FAMILY INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-bit Instruction Word				Status Affected	Notes
				MSb			LSb		
BIT-ORIENTED OPERATIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, b, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2
CONTROL OPERATIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	1 (2)	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	2	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call Subroutine 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C	
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	4
POP	—	Pop Top of Return Stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software Device Reset	1	0000	0000	1111	1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 25-2: PIC18F85J11 FAMILY INSTRUCTION SET (CONTINUED)

Mnemonic, Operands	Description	Cycles	16-bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS					
ADDLW k	Add Literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW k	AND Literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW k	Inclusive OR Literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR f, k	Move Literal (12-bit) 2nd word to FSR(f) 1st word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB k	Move Literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW k	Move Literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW k	Multiply Literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW k	Subtract WREG from Literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW k	Exclusive OR Literal with WREG	1	0000 1010 kkkk kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS					
TBLRD*	Table Read	2	0000 0000 0000 1000	None	
TBLRD*+	Table Read with Post-Increment		0000 0000 0000 1001	None	
TBLRD*-	Table Read with Post-Decrement		0000 0000 0000 1010	None	
TBLRD+*	Table Read with Pre-Increment		0000 0000 0000 1011	None	
TBLWT*	Table Write	2	0000 0000 0000 1100	None	
TBLWT*+	Table Write with Post-Increment		0000 0000 0000 1101	None	
TBLWT*-	Table Write with Post-Decrement		0000 0000 0000 1110	None	
TBLWT+*	Table Write with Pre-Increment		0000 0000 0000 1111	None	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

25.1.1 STANDARD INSTRUCTION SET

ADDLW **ADD Literal to W**

Syntax: ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal ‘k’ and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: `ADDLW 15h`

Before Instruction
W = 10h
After Instruction
W = 25h

ADDWF **ADD W to f**

Syntax: ADDWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: $(W) + (f) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	01da	ffff	ffff

Description: Add W to register ‘f’. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

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 $f \le 95$ (5Fh). See **Section 25.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `ADDWF REG, 0, 0`

Before Instruction
W = 17h
REG = 0C2h
After Instruction
W = 0D9h
REG = 0C2h

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

ADDWFC **ADD W and Carry bit to f**

Syntax: ADDWFC f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (W) + (f) + (C) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	00da	ffff	ffff

Description: Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

0000	1011	kkkk	kkkk

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `ANDLW 05Fh`

Before Instruction
W = A3h
After Instruction
W = 03h

ANDWF **AND W with f**

Syntax: ANDWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (W) .AND. (f) → dest

Status Affected: N, Z

Encoding:

0001	01da	ffff	ffff

Description: The contents of W are 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'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $-128 \le n \le 127$

Operation: if Carry bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0010	nnnn	nnnn

Description: If the Carry bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE    BC   5
```
Before Instruction
PC = address (HERE)
After Instruction
If Carry = 1;
PC = address (HERE + 12)
If Carry = 0;
PC = address (HERE + 2)

BCF **Bit Clear f**

Syntax: BCF f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0, 1]$

Operation: $0 \rightarrow$ f<b>

Status Affected: None

Encoding:

1001	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is cleared.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $-128 \le n \le 127$

Operation: if Negative bit is '1',
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0110	nnnn	nnnn

Description: If the Negative bit is '1', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE    BN   Jump
```

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 1;
PC = address (Jump)
If Negative = 0;
PC = address (HERE + 2)

BNC **Branch if Not Carry**

Syntax: BNC n

Operands: -128 ≤ n ≤ 127

Operation: if Carry bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the Carry bit is '0', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNC Jump

```
Before Instruction
    PC          =   address (HERE)
After Instruction
    If Carry    =   0;
        PC      =   address (Jump)
    If Carry    =   1;
        PC      =   address (HERE + 2)
```

BNN **Branch if Not Negative**

Syntax: BNN n

Operands: -128 ≤ n ≤ 127

Operation: if Negative bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the Negative bit is '0', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNN Jump

```
Before Instruction
    PC          =   address (HERE)
After Instruction
    If Negative =   0;
        PC      =   address (Jump)
    If Negative =   1;
        PC      =   address (HERE + 2)
```

BNOV **Branch if Not Overflow**

Syntax: BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if Overflow bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0101	nnnn	nnnn

Description: If the Overflow bit is '0', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNOV Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 0;
PC = address (Jump)
If Overflow = 1;
PC = address (HERE + 2)

BNZ **Branch if Not Zero**

Syntax: BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0001	nnnn	nnnn

Description: If the Zero bit is '0', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNZ Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Zero = 0;
PC = address (Jump)
If Zero = 1;
PC = address (HERE + 2)

BRA **Unconditional Branch**

Syntax: BRA n

Operands: $-1024 \le n \le 1023$

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: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0, 1]$

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0, 1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 ≤ 95 (5Fh). See **Section 25.2.3 "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: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0, 1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 ≤ 95 (5Fh). See **Section 25.2.3 "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)

BTG **Bit Toggle f**

Syntax: BTG f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0, 1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit 'b' in data memory location 'f' is inverted.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $-128 \le n \le 127$

Operation: if Overflow bit is '1',
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the Overflow bit is '1', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE    BOV  Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 1;
PC = address (Jump)
If Overflow = 0;
PC = address (HERE + 2)

BZ **Branch if Zero**

Syntax: BZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the Zero bit is '1', then the program will branch.

The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BZ Jump`

Before Instruction
PC = address (HERE)
After Instruction
If Zero = 1;
PC = address (Jump)
If Zero = 0;
PC = address (HERE + 2)

CALL **Subroutine Call**

Syntax: CALL k {,s}

Operands: 0 ≤ k ≤ 1048575
s ∈ [0, 1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>;
if s = 1,
(W) → WS,
(STATUS) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC+ 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs. Then, the 20-bit value, 'k', is loaded into PC<20:1>. CALL is a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `HERE CALL THERE,1`

Before Instruction
PC = address (HERE)
After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF **Clear f**

Syntax: CLRF f {,a}

Operands: $0 \le f \le 255$
$a \in [0, 1]$

Operation: 000h → f,
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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 → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits, $\overline{TO}$ and $\overline{PD}$, are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: CLRWDT

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 00h
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: COMF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: $\bar{f} \rightarrow$ dest

Status Affected: N, Z

Encoding:

0001	11da	ffff	ffff

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $0 \le f \le 255$
$a \in [0, 1]$

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	001a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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 ≠ W;
PC = Address (NEQUAL)

CPFSGT **Compare f with W, Skip if f > W**

Syntax: CPFSGT f {,a}

Operands: $0 \le f \le 255$
$a \in [0, 1]$

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.

If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $0 \le f \le 255$
$a \in [0, 1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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)
```

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 ≤ f ≤ 255
d ∈ [0, 1]
a ∈ [0, 1]

Operation: (f) – 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0000	01da	ffff	ffff

Description: Decrement register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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

DECFSZ **Decrement f, Skip if 0**

Syntax: DECFSZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) – 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0010	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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 ≠ 0;
PC = Address (HERE + 2)

DCFSNZ **Decrement f, Skip if Not 0**

Syntax: DCFSNZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) – 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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 ≠ 0;
PC = Address (NZERO)

GOTO **Unconditional Branch**

Syntax: GOTO k

Operands: $0 \le k \le 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	1111	k_7kkk	$kkkk_0$
2nd word(k<19:8>)	1111	$k_{19}kkk$	kkkk	$kkkk_8$

Description: GOTO allows an unconditional branch anywhere within entire 2-Mbyte memory range. The 20-bit value, 'k', is loaded into PC<20:1>. GOTO is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: GOTO THERE

After Instruction
PC = Address (THERE)

INCF **Increment f**

Syntax: INCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) + 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0010	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

INCFSZ **Increment f, Skip if 0**

Syntax: INCFSZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) + 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0011	11da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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 ≠ 0;
PC = Address (NZERO)

INFSNZ **Increment f, Skip if Not 0**

Syntax: INFSNZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) + 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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 ≠ 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

IORLW **Inclusive OR Literal with W**

Syntax: IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

0000	1001	kkkk	kkkk

Description: The contents of W are 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: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

Description: The 12-bit literal 'k' is loaded into the File Select Register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction
FSR2H = 03h
FSR2L = ABh

MOVF **Move f**

Syntax: MOVF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'. 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

MOVFF **Move f to f**

Syntax: MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register, 'f_s', are moved to destination register, 'f_d'. Location of source, 'f_s', can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination, 'f_d', can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 33h
REG2 = 11h
After Instruction
REG1 = 33h
REG2 = 33h

MOVLB **Move Literal to Low Nibble in BSR**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0' regardless of the value of k_7:k_4.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example: MOVLB 5

Before Instruction
BSR Register = 02h
After Instruction
BSR Register = 05h

MOVLW **Move Literal to W**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

0000	1110	kkkk	kkkk

Description: The eight-bit literal 'k' is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: MOVLW 5Ah

After Instruction
W = 5Ah

MOVWF **Move W to f**

Syntax: MOVWF f {,a}

Operands: $0 \le f \le 255$
$a \in [0, 1]$

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

0110	111a	ffff	ffff

Description: Move data from W to register 'f'. Location 'f' can be anywhere in the 256-byte bank.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

MULLW **Multiply Literal with W**

Syntax: MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in 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: $0 \le f \le 255$
$a \in [0, 1]$

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location, 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged.

None of the Status flags are affected.

Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

NEGF **Negate f**

Syntax: NEGF f {,a}

Operands: $0 \le f \le 255$
$a \in [0, 1]$

Operation: $(\bar{f}) + 1 \rightarrow f$

Status Affected: N, OV, C, DC, Z

Encoding:

0110	110a	ffff	ffff

Description: Location 'f' is negated using two's complement. The result is placed in the data memory location 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 25.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example:

```
NEGF    REG, 1
```

Before Instruction
REG = 0011 1010 [3Ah]
After Instruction
REG = 1100 0110 [C6h]

NOP **No Operation**

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

POP **Pop Top of Return Stack**

Syntax: 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

RCALL **Relative Call**

Syntax: RCALL n

Operands: $-1024 \leq n \leq 1023$

Operation: $(PC) + 2 \rightarrow TOS$,
$(PC) + 2 + 2n \rightarrow PC$

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 $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start Reset	No operation	No operation

Example: RESET

After Instruction
Registers = Reset Value
Flags* = Reset Value

RETFIE **Return from Interrupt**

Syntax: RETFIE {s}

Operands: s ∈ [0, 1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL;
if s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low-priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	POP PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: RETFIE 1

After Interrupt
PC = TOS
W = WS
BSR = BSRS
STATUS = STATUSS
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

RETURN **Return from Subroutine**

Syntax: RETURN {s}

Operands: s ∈ [0, 1]

Operation: (TOS) → PC;
if s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	0000	0001	001s

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If 's'= 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: RETURN

After Instruction:
PC = TOS

RLCF **Rotate Left f through Carry**

Syntax: RLCF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0, 1]
a ∈ [0, 1]

Operation: (f<n>) → dest<n + 1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

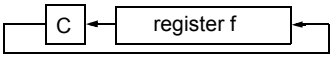
Encoding:

0011	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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

RLNCF **Rotate Left f (No Carry)**

Syntax: RLNCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

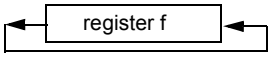
Encoding:

0100	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

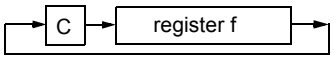
Encoding:

0011	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

RRNCF **Rotate Right f (No Carry)**

Syntax: RRNCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f<n>) → dest<n – 1>,
(f<0>) → dest<7>

Status Affected: N, Z

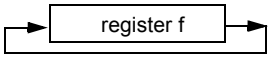
Encoding:

0100	00da	ffff	ffff

Description: The contents of register ‘f’ are rotated one bit to the right. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is placed back in register ‘f’.

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 ≤ 95 (5Fh). See **Section 25.2.3 “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: $0 \le f \le 255$
$a \in [0, 1]$

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh.

If ‘a’ is ‘0’, the Access Bank 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 ≤ 95 (5Fh). See **Section 25.2.3 “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
```

SLEEP **Enter Sleep Mode**

Syntax: SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description: The Power-Down status bit ($\overline{PD}$) is cleared. The Time-out status bit ($\overline{TO}$) is set. The Watchdog Timer and its postscaler are cleared.

The processor is put into Sleep mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?
After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB **Subtract f from W with Borrow**

Syntax: SUBFWB f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0, 1]
a ∈ [0, 1]

Operation: (W) – (f) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description: Subtract register 'f' and Carry flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored in register '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 ≤ 95 (5Fh). See **Section 25.2.3 "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

SUBLW **Subtract W from Literal**

Syntax: SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1:
```
SUBLW   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: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: $(f) - (W) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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
```

SUBWFB **Subtract W from f with Borrow**

Syntax: SUBWFB f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f) – (W) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	10da	ffff	ffff

Description: Subtract W and the Carry flag (borrow) from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (f<3:0>) → dest<7:4>,
(f<7:4>) → dest<3:0>

Status Affected: None

Encoding:

0011	10da	ffff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in register 'f'.

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 ≤ 95 (5Fh). See **Section 25.2.3 "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
```

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 (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 = AAh
TBLPTR = 01A357h
MEMORY(01A357h) = 12h
MEMORY(01A358h) = 34h
After Instruction
TABLAT = 34h
TBLPTR = 01A358h

TBLWT **Table Write**

Syntax: TBLWT (*; *+; *-; +*)

Operands: None

Operation: if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR – No Change
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) + 1 → TBLPTR
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) – 1 → TBLPTR
if TBLWT+*,
(TBLPTR) + 1 → TBLPTR;
(TABLAT) → Holding Register

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 6.0 "Memory Organization"** for additional details on programming Flash memory.)

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR<0> = 0: Least Significant Byte of program memory word

TBLPTR<0> = 1: Most Significant Byte of program memory word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

TBLWT **Table Write (Continued)**

Example 1: TBLWT *+;

Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
HOLDING REGISTER (00A356h) = FFh
After Instructions (table write completion)
TABLAT = 55h
TBLPTR = 00A357h
HOLDING REGISTER (00A356h) = 55h

Example 2: TBLWT +*;

Before Instruction
TABLAT = 34h
TBLPTR = 01389Ah
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = FFh
After Instruction (table write completion)
TABLAT = 34h
TBLPTR = 01389Bh
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = 34h

TSTFSZ **Test f, Skip if 0**

Syntax: TSTFSZ f {,a}

Operands: $0 \le f \le 255$
$a \in [0, 1]$

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction.

If 'a' is '0', the Access Bank 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 $f \le 95$ (5Fh). See **Section 25.2.3 "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: $0 \le k \le 255$

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

XORWF	Exclusive OR W with f

Syntax: XORWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0, 1]$
$a \in [0, 1]$

Operation: (W) .XOR. (f) → dest

Status Affected: N, Z

Encoding:

0001	10da	ffff	ffff

Description: Exclusive OR the contents of W with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register 'f'.

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 $f \le 95$ (5Fh). See **Section 25.2.3 "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

25.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, the PIC18F85J11 family of devices also provides an optional extension to the core CPU functionality. The added features include eight additional instructions that augment Indirect and Indexed Addressing operations and the implementation of Indexed Literal Offset Addressing for many of the standard PIC18 instructions.

The additional features of the extended instruction set are enabled by default on unprogrammed devices. Users must properly set or clear the XINST Configuration bit during programming to enable or disable these features.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers, 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 25-3. Detailed descriptions are provided in **Section 25.2.2 "Extended Instruction Set"**. The opcode field descriptions in Table 25-1 (page 300) 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 the 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.

25.2.1 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. The MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 25.2.3.1 "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 25-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDFSR	f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
ADDULNK	k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
CALLW		Call Subroutine using WREG	2	0000	0000	0001	0100	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word	2	1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word		1111	xxxx	xzzz	zzzz	
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

25.2.2 EXTENDED INSTRUCTION SET

ADDFSR **Add Literal to FSR**

Syntax: ADDFSR f, k

Operands: $0 \le k \le 63$
$f \in [0, 1, 2]$

Operation: FSR(f) + k → 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: $0 \le k \le 63$

Operation: FSR2 + k → FSR2,
(TOS) → PC

Status Affected: None

Encoding:	1110	1000	11kk	kkkk

Description: The 6-bit literal 'k' is added to the contents of FSR2. A RETURN is then executed by loading the PC with the TOS.

The instruction takes two cycles to execute; a NOP is performed during the second cycle.

This may be thought of as a special case of the ADDFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR
No Operation	No Operation	No Operation	No Operation

Example: `ADDULNK 23h`

Before Instruction
FSR2 = 03FFh
PC = 0100h
After Instruction
FSR2 = 0422h
PC = (TOS)

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

CALLW **Subroutine Call Using WREG**

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

0000	0000	0001	0100

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched.

Unlike CALL, there is no option to update W, STATUS or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	Push PC to stack	No operation
No operation	No operation	No operation	No operation

Example: HERE CALLW

Before Instruction
PC = address (HERE)
PCLATH = 10h
PCLATU = 00h
W = 06h
After Instruction
PC = 001006h
TOS = address (HERE + 2)
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF **Move Indexed to f**

Syntax: MOVSF [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 4095$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (source)	1110	1011	0zzz	$zzzz_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register, 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset, 'z_s', in the first word to the value of FSR2. The address of the destination register is specified by the 12-bit literal, 'f_d', in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh).

The MOVSF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVSF [05h], REG2

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 33h

MOVSS **Move Indexed to Indexed**

Syntax: MOVSS [z_s], [z_d]

Operands: $0 \le z_s \le 127$
$0 \le z_d \le 127$

Operation: ((FSR2) + z_s) → ((FSR2) + z_d)

Status Affected: None

Encoding:

1st word (source)	1110	1011	1zzz	$zzzz_s$
2nd word (dest.)	1111	xxxx	xzzz	$zzzz_d$

Description: The contents of the source register are moved to the destination register. The addresses of the source and destination registers are determined by adding the 7-bit literal offsets, 'z_s' or 'z_d', respectively, to the value of FSR2. Both registers can be located anywhere in the 4096-byte data memory space (000h to FFFh).

The MOVSS instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h. If the resultant destination address points to an Indirect Addressing register, the instruction will execute as a NOP.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	Determine dest addr	Determine dest addr	Write to dest reg

Example: MOVSS [05h], [06h]

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 11h

After Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 33h

PUSHL **Store Literal at FSR2, Decrement FSR2**

Syntax: PUSHL k

Operands: $0 \le k \le 255$

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

SUBFSR **Subtract Literal from FSR**

Syntax: SUBFSR f, k

Operands: $0 \le k \le 63$

$f \in [0, 1, 2]$

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: $0 \le k \le 63$

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)

25.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note:	Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing (**Section 6.6.1 "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 Select 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 25.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

25.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the File Select Register argument, 'f', in the standard, byte-oriented and bit-oriented commands is replaced with the literal offset value, 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within the brackets, will generate an error in the MPASM Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled) when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument, 'd', functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

25.2.4 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 PIC18F85J11 family, it is very important to consider the type of code. A large, re-entrant application that is written in C and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

ADDWF **ADD W to Indexed (Indexed Literal Offset mode)**

Syntax: ADDWF [k] {,d}

Operands: $0 \le k \le 95$
$d \in [0, 1]$

Operation: (W) + ((FSR2) + k) → 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'.

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: $0 \le f \le 95$
$0 \le b \le 7$

Operation: 1 → ((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: $0 \le k \le 95$

Operation: FFh → ((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

25.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set for the PIC18F85J11 family. This includes the MPLAB C18 C Compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default Configuration bits for that device. The default setting for the XINST Configuration bit is '1', enabling the extended instruction set and Indexed Literal Offset Addressing. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option or dialog box within the environment that allows the user to configure the language tool and its settings for the project
- A command line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

26.0 ELECTRICAL CHARACTERISTICS

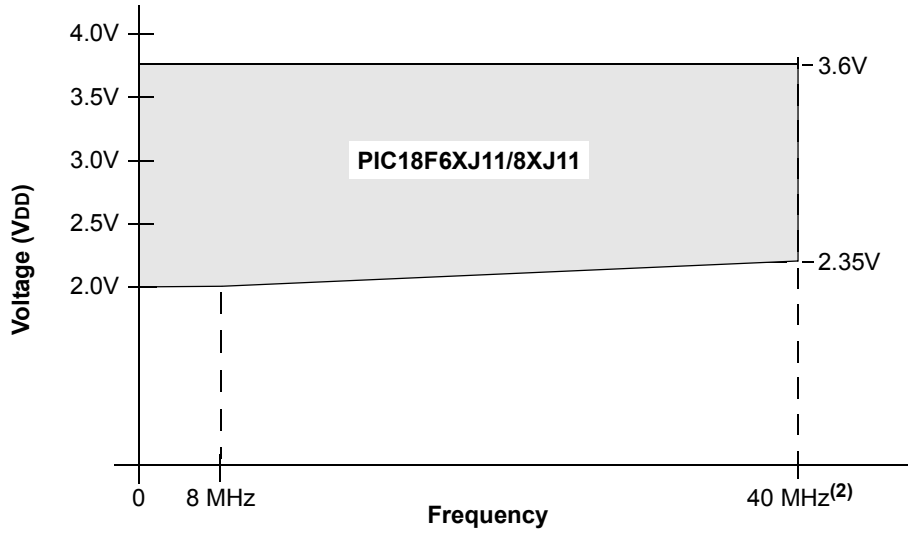
Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -40°C to +100°C
Storage temperature ..... -65°C to +150°C
Voltage on any digital only I/O pin or $\overline{MCLR}$ with respect to V_{SS} (except V_{DD}) ..... -0.3V to 5.6V
Voltage on any combined digital and analog pin with respect to V_{SS} (except V_{DD} and $\overline{MCLR}$)..... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{DDCORE} with respect to V_{SS} ..... -0.3V to 2.75V
Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to 3.6V
Total power dissipation **(Note 1)** 1.0W
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin ..... 250 mA
Maximum output current sunk by PORTA<7:6> and any PORTB and PORTC I/O pins..... 25 mA
Maximum output current sunk by any PORTD, PORTE and PORTJ I/O pins ..... 8 mA
Maximum output current sunk by PORTA<5:0> and any PORTF, PORTG and PORTH I/O pins ..... 2 mA
Maximum output current sourced by PORTA<7:6> and any PORTB and PORTC I/O pins ..... 25 mA
Maximum output current sourced by any PORTD, PORTE and PORTJ I/O pins..... 8 mA
Maximum output current sourced by PORTA<5:0> and any PORTF, PORTG and PORTH I/O pins ..... 2 mA
Maximum current sunk by all ports combined..... 200 mA

Note 1: Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

† **NOTICE:** Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

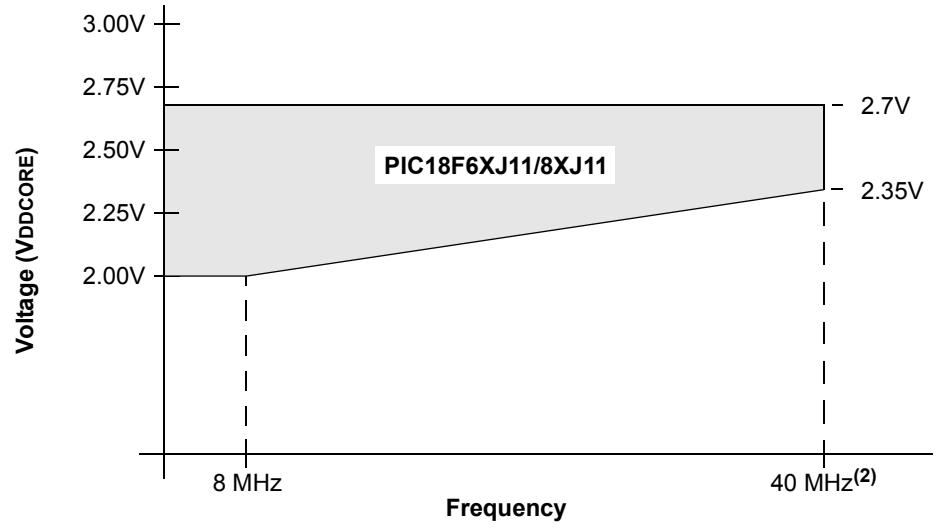
FIGURE 26-1: PIC18F85J11 FAMILY VOLTAGE-FREQUENCY GRAPH, REGULATOR ENABLED (INDUSTRIAL)(1)



Note 1: When the on-chip regulator is enabled, its BOR circuit will automatically trigger a device Reset before VDD reaches a level at which full-speed operation is not possible.

2: FMAX = 25 MHz in 8-bit External Memory mode.

FIGURE 26-2: PIC18F85J11 FAMILY VOLTAGE-FREQUENCY GRAPH, REGULATOR DISABLED (INDUSTRIAL)(1,3)



Note 1: For frequencies between 4 MHz and 40 MHz, FMAX = (51.42 MHz/V) * (VDDCORE – 2V) + 4 MHz.

2: FMAX = 25 MHz in 8-bit External Memory mode.

3: When the on-chip voltage regulator is disabled, VDD and VDDCORE must be maintained so that VDDCORE ≤ VDD ≤ 3.6V.

26.1 DC Characteristics: Supply Voltage PIC18F85J11 Family (Industrial)

PIC18F85J11 Family (Industrial)			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**	VDDCORE 2.0	— —	3.6 3.6	V V	ENVREG tied to VSS ENVREG tied to VDD
D001B	VDDCORE	**External Supply for Microcontroller Core**	2.0	—	2.70	V	ENVREG tied to VSS
D001C	AVDD	**Analog Supply Voltage**	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	**Analog Ground Potential**	VSS – 0.3	—	VSS + 0.3	V	
D002	VDR	**RAM Data Retention Voltage(1)**	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to Ensure Internal Power-on Reset Signal	—	—	0.7	V	See **Section 5.3 "Power-on Reset (POR)"** for details
D004	SVDD	**VDD Rise Rate** to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	See **Section 5.3 "Power-on Reset (POR)"** for details
D005	VBOR	**Brown-out Reset Voltage**	—	1.9	—	V	

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**	
	Power-Down Current (IPD)(1)					
	All devices	0.2	0.9	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V (**Sleep** mode)(4)
		0.1	0.9	µA	+25°C	
		0.3	3	µA	+60°C	
		2.4	5	µA	+85°C	
	All devices	0.5	0.9	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V (**Sleep** mode)(4)
		0.1	0.9	µA	+25°C	
		0.4	3	µA	+60°C	
		2.7	5	µA	+85°C	
	All devices	2.7	6	µA	-40°C	VDD = 3.3V (**Sleep** mode)(5)
		3.5	6	µA	+25°C	
		4.1	8	µA	+60°C	
		6.7	12	µA	+85°C	

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	6.5	16	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 31 kHz (**RC_RUN** mode, internal oscillator source)
		7	16	µA	+25°C		
		9.5	20	µA	+85°C		
	All devices	10	18	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		10.5	18	µA	+25°C		
		12.5	24	µA	+85°C		
	All devices	41	100	µA	-40°C	VDD = 3.3V(5)	
		52	100	µA	+25°C		
		71	110	µA	+85°C		
	All devices	359	750	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**INTOSC_RUN** mode, internal oscillator source)
		387	750	µA	+25°C		
		407	840	µA	+85°C		
	All devices	438	850	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		470	850	µA	+25°C		
		491	910	µA	+85°C		
	All devices	486	900	µA	-40°C	VDD = 3.3V(5)	
		526	900	µA	+25°C		
		564	990	µA	+85°C		
	All devices	0.76	1.45	mA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 4 MHz (**INTOSC_RUN** mode, internal oscillator source)
		0.84	1.45	mA	+25°C		
		0.9	1.6	mA	+85°C		
	All devices	1.1	1.63	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		1.18	1.63	mA	+25°C		
		1.24	1.75	mA	+85°C		
	All devices	1.25	1.86	mA	-40°C	VDD = 3.3V(5)	
		1.29	1.86	mA	+25°C		
		1.37	1.94	mA	+85°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	2.4	8	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 31 kHz (**RC_IDLE** mode, internal oscillator source)
		2.5	8	µA	+25°C		
		4.8	12	µA	+85°C		
	All devices	3.2	9	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		3.2	9	µA	+25°C		
		6	14	µA	+85°C		
	All devices	62	82	µA	-40°C	VDD = 3.3V(5)	
		42	82	µA	+25°C		
		59	97	µA	+85°C		
	All devices	251	570	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**INTOSC_IDLE** mode, internal oscillator source)
		264	570	µA	+25°C		
		272	590	µA	+85°C		
	All devices	284	610	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		284	610	µA	+25°C		
		293	650	µA	+85°C		
	All devices	295	710	µA	-40°C	VDD = 3.3V(5)	
		323	710	µA	+25°C		
		392	790	µA	+85°C		
	All devices	368	760	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 4 MHz (**INTOSC_IDLE** mode, internal oscillator source)
		362	760	µA	+25°C		
		370	800	µA	+85°C		
	All devices	400	850	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		410	850	µA	+25°C		
		418	900	µA	+85°C		
	All devices	460	950	µA	-40°C	VDD = 3.3V(5)	
		462	950	µA	+25°C		
		486	1,000	µA	+85°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	165	490	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**PRI_RUN** mode, EC oscillator)
		180	490	µA	+25°C		
		200	490	µA	+85°C		
	All devices	256	670	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		260	670	µA	+25°C		
		280	670	µA	+85°C		
	All devices	460	850	µA	-40°C	VDD = 3.3V(5)	
		456	850	µA	+25°C		
		482	850	µA	+85°C		
	All devices	0.63	2.2	mA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 4 MHz (**PRI_RUN** mode, EC oscillator)
		0.68	2.2	mA	+25°C		
		0.74	2.2	mA	+85°C		
	All devices	0.91	2.5	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		1.04	2.5	mA	+25°C		
		1.04	2.5	mA	+85°C		
	All devices	1.32	3.0	mA	-40°C	VDD = 3.3V(5)	
		1.32	3.0	mA	+25°C		
		1.41	3.0	mA	+85°C		
	All devices	7.47	14	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 40 MHz (**PRI_RUN** mode, EC oscillator)
		5.81	14	mA	+25°C		
		6.32	13	mA	+85°C		
	All devices	8.84	18	mA	-40°C	VDD = 3.3V(5)	
		8.66	18	mA	+25°C		
		7.97	16	mA	+85°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	2.8	3.8	mA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 4 MHz, 16 MHz internal (**PRI_RUN** mode, HSPLL oscillator)
		3.02	3.8	mA	+25°C		
		3.01	4.5	mA	+85°C		
	All devices	4.5	5.4	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 4 MHz, 16 MHz internal (**PRI_RUN** mode, HSPLL oscillator)
		4.8	5.6	mA	+25°C		
		4.54	5.6	mA	+85°C		
	All devices	5.72	6.7	mA	-40°C	VDD = 3.3V(5)	FOSC = 4 MHz, 16 MHz internal (**PRI_RUN** mode, HSPLL oscillator)
		5.55	6.5	mA	+25°C		
		5.3	6.5	mA	+85°C		
	All devices	7.4	8.5	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 10 MHz, 40 MHz internal (**PRI_RUN** mode, HSPLL oscillator)
		7.23	8.5	mA	+25°C		
		6.55	7.5	mA	+85°C		
	All devices	9.74	11.6	mA	-40°C	VDD = 3.3V(5)	FOSC = 10 MHz, 40 MHz internal (**PRI_RUN** mode, HSPLL oscillator)
		9.43	11.6	mA	+25°C		
		8.89	10.5	mA	+85°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	50	120	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator)
		51	120	µA	+25°C		
		54	130	µA	+85°C		
	All devices	223	480	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		134	300	µA	+25°C		
		110	270	µA	+85°C		
	All devices	307	550	µA	-40°C	VDD = 3.3V(5)	
		254	500	µA	+25°C		
		194	460	µA	+85°C		
	All devices	307	850	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator)
		200	850	µA	+25°C		
		202	800	µA	+85°C		
	All devices	483	950	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		318	950	µA	+25°C		
		343	900	µA	+85°C		
	All devices	0.52	1.3	mA	-40°C	VDD = 3.3V(5)	
		0.47	1.2	mA	+25°C		
		0.47	1.2	mA	+85°C		
	All devices	2.38	8	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator)
		2.04	8	mA	+25°C		
		2.52	9	mA	+85°C		
	All devices	3.02	10	mA	-40°C	VDD = 3.3V(5)	
		2.99	10	mA	+25°C		
		4.23	11	mA	+85°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2)						
	All devices	10.5	22	µA	-10°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 32 kHz(3) (**SEC_RUN** mode, Timer1 as clock)
		13.4	28	µA	+25°C		
		17.6	40	µA	+70°C		
	All devices	13.2	30	µA	-10°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		16.2	35	µA	+25°C		
		20.7	50	µA	+70°C		
	All devices	39	120	µA	-10°C	VDD = 3.3V(5)	
		58	150	µA	+25°C		
		75	190	µA	+70°C		
	All devices	5.7	15	µA	-10°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 32 kHz(3) (**SEC_IDLE** mode, Timer1 as clock)
		8.9	20	µA	+25°C		
		12.8	26	µA	+70°C		
	All devices	6.6	17	µA	-10°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		9.7	24	µA	+25°C		
		13.7	30	µA	+70°C		
	All devices	39	115	µA	-10°C	VDD = 3.3V(5)	
		52.8	145	µA	+25°C		
		72.7	185	µA	+70°C		

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.2 DC Characteristics: Power-Down and Supply Current PIC18F85J11 Family (Industrial) (Continued)

PIC18F85J11 Family (Industrial)		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Module Differential Currents (ΔIWDT, ΔIOSCB, ΔIAD)						
D022 (ΔIWDT)	**Watchdog Timer**	1.6	4	μA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	
		1.7	4	μA	+25°C		
		1.6	4	μA	+85°C		
		2.5	5	μA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		2.5	5	μA	+25°C		
		2.3	5	μA	+85°C		
		3.8	6	μA	-40°C	VDD = 3.3V(5)	
		2.6	6	μA	+25°C		
		2.4	6	μA	+85°C		
D025 (ΔIOSCB)	**Timer1 Oscillator**	6.6	12.5	μA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	32 kHz on Timer1(3)
		7.9	12.5	μA	+25°C		
		11.5	18	μA	+85°C		
		7.2	12.5	μA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	32 kHz on Timer1(3)
		8.1	12.5	μA	+25°C		
		11.9	18.5	μA	+85°C		
		7	12.5	μA	-40°C	VDD = 3.3V(5)	32 kHz on Timer1(3)
		9	12.5	μA	+25°C		
		11	18.5	μA	+85°C		
D026 (ΔIAD)	**A/D Converter**	1	1.5	μA	-40°C to +85°C	VDD = 2.0V, VDDCORE = 2.0V(4)	A/D on, not converting
		1	1.5	μA	-40°C to +85°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		1	1.5	μA	-40°C to	VDD = 3.3V(5)	

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

4: Voltage regulator disabled (ENVREG tied to VSS).

5: Voltage regulator enabled (ENVREG tied to VDD).

26.3 DC Characteristics: PIC18F85J11 Family (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		All I/O Ports:				
D030		with TTL Buffer	VSS	0.15 VDD	V	VDD < 3.3V
D030A			—	0.8	V	3.3V ≤ VDD ≤ 3.6V
D031		with Schmitt Trigger Buffer	VSS	0.2 VDD	V	
D031A		RC3 and RC4	VSS	0.3 VDD	V	I²C™ enabled
D031B			VSS	0.8	V	SMBus enabled
D032		$\overline{MCLR}$	VSS	0.2 VDD	V	
D033		OSC1	VSS	0.3 VDD	V	HS, HSPLL modes
D033A		OSC1	VSS	0.2 VDD	V	EC, ECPLL modes(1)
D034		T13CKI	VSS	0.3	V	
	VIH	**Input High Voltage**				
		I/O Ports with non 5.5V Tolerance:(2)				
D040		with TTL Buffer	0.25 VDD + 0.8V	VDD	V	VDD < 3.3V
D040A			2.0	VDD	V	3.3V ≤ VDD ≤ 3.6V
D041		with Schmitt Trigger Buffer	0.8 VDD	VDD	V	
D041A		RC3 and RC4	0.7 VDD	VDD	V	I²C enabled
D041B			2.1	VDD	V	SMBus enabled, VDD > 3.3V
		I/O Ports with 5.5V Tolerance:(2)				
Dxxx		with TTL Buffer	0.25 VDD + 0.8V	5.5	V	VDD < 3.3V
DxxxA			2.0	5.5	V	3.3V ≤ VDD ≤ 3.6V
Dxxx		with Schmitt Trigger Buffer	0.8 VDD	5.5	V	
D042		$\overline{MCLR}$	0.8 VDD	VDD	V	
D043		OSC1	0.7 VDD	VDD	V	HS, HSPLL modes
D043A		OSC1	0.8 VDD	VDD	V	EC, ECPLL modes
D044		T13CKI	1.6	VDD	V	
	IIL	**Input Leakage Current(1)**				
D060		I/O Ports with non 5.5V tolerance:(2)	—	200	nA	VSS ≤ VPIN ≤ VDD, pin at high-impedance
		I/O Ports with 5.5V tolerance:(2)	—	200	nA	VSS ≤ VPIN ≤ 5.5V, pin at high-impedance
D061		$\overline{MCLR}$	—	±1	µA	VSS ≤ VPIN ≤ VDD
D063		OSC1	—	±1	µA	VSS ≤ VPIN ≤ VDD
	IPU	**Weak Pull-up Current**				
D070	IPURB	PORTB Weak Pull-up Current	30	400	µA	VDD = 3.3V, VPIN = VSS

Note 1: Negative current is defined as current sourced by the pin.

2: Refer to Table 11-1 for the pins that have corresponding tolerance limits.

26.3 DC Characteristics: PIC18F85J11 Family (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**				
D080		I/O Ports:				
		PORTA, PORTF, PORTG, PORTH	—	0.4	V	IOL = 3.4 mA, VDD = 3.3V, -40°C to +85°C
		PORTD, PORTE, PORTJ	—	0.4	V	IOL = 3.4 mA, VDD = 3.3V, -40°C to +85°C
		PORTB, PORTC	—	0.4	V	IOL = 8.5 mA, VDD = 3.3V, -40°C to +85°C
D083		OSC2/CLKO (EC, ECPLL modes)	—	0.4	V	IOL = 1.6 mA, VDD = 3.3V, -40°C to +85°C
	VOH	**Output High Voltage(1)**				
D090		I/O Ports:			V	
		PORTA, PORTF, PORTG, PORTH	2.4	—	V	IOH = -2 mA, VDD = 3.3V, -40°C to +85°C
		PORTD, PORTE, PORTJ	2.4	—	V	IOH = -2 mA, VDD = 3.3V, -40°C to +85°C
		PORTB, PORTC	2.4	—	V	IOH = -6 mA, VDD = 3.3V, -40°C to +85°C
D092		OSC2/CLKO (INTOSC, EC, ECPLL modes)	2.4	—	V	IOH = -1 mA, VDD = 3.3V, -40°C to +85°C
		Capacitive Loading Specs on Output Pins				
D100	COSC2	OSC2 Pin	—	15	pF	In HS mode when external clock is used to drive OSC1
D101	CIO	All I/O Pins and OSC2	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	I^2C™ Specification

Note 1: Negative current is defined as current sourced by the pin.

2: Refer to Table 11-1 for the pins that have corresponding tolerance limits.

TABLE 26-1: MEMORY PROGRAMMING REQUIREMENTS

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Program Flash Memory					
D130	EP	Cell Endurance	100	1k	—	E/W	-40°C to +85°C
D131	VPR	VDD for Read	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D132	VPEW	Voltage for Self-Timed Erase or Write					
		VDD	2.35	—	3.6	V	ENVREG tied to VDD
		VDDCORE	2.25	—	2.7	V	ENVREG tied to VSS
D133A	TIW	Self-Timed Write Cycle Time	—	2.8	—	ms	
D133B	TIE	Self-Timed Block Erase Cycle Time	—	33	—	ms	
D134	TRETD	Characteristic Retention	20	—	—	Year	Provided no other specifications are violated
D135	IDDP	Supply Current during Programming	—	3	7	mA	
D1xxx	TWE	Writes per Erase Cycle	—	—	1		Per one physical word address

† Data in “Typ” column is at 3.3V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 26-2: COMPARATOR SPECIFICATIONS

Operating Conditions: 3.0V ≤ VDD ≤ 3.6V, -40°C ≤ TA ≤ +85°C (unless otherwise stated)							
Param No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D300	VIOFF	Input Offset Voltage	—	±5.0	±25	mV	
D301	VICM	Input Common Mode Voltage	0	—	AVDD – 1.5	V	
D302	CMRR	Common Mode Rejection Ratio	55	—	—	dB	
D303	TRESP	Response Time(1)	—	150	400	ns	
D304	TMC2OV	Comparator Mode Change to Output Valid	—	—	10	μs	
D305	VIRV	Internal Reference Voltage	—	1.2	—	V	

Note 1: Response time measured with one comparator input at (AVDD – 1.5)/2, while the other input transitions from VSS to VDD.

TABLE 26-3: VOLTAGE REFERENCE SPECIFICATIONS

Operating Conditions: 3.0V ≤ VDD ≤ 3.6V, -40°C ≤ TA ≤ +85°C (unless otherwise stated)							
Param No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D310	VRES	Resolution	VDD/24	—	VDD/32	LSb	
D311	VRAA	Absolute Accuracy	—	—	1/2	LSb	
D312	VRUR	Unit Resistor Value (R)	—	2k	—	Ω	
D313	TSET	Settling Time(1)	—	—	10	μs	

Note 1: Settling time measured while CVRR = 1 and CVR<3:0> bits transition from '0000' to '1111'.

TABLE 26-4: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

Operating Conditions: -40°C ≤ TA ≤ +85°C (unless otherwise stated)							
Param No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
	VRGOUT	Regulator Output Voltage	—	2.5	—	V	
	CEFC	External Filter Capacitor Value	4.7	10	—	μF	Capacitor must be low series resistance (<5 Ohms)

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^2C specifications only)
4. Ts (I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T13CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-impedance)	V	Valid
L	Low	Z	High-impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I^2C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	Stop condition
STA	Start condition		

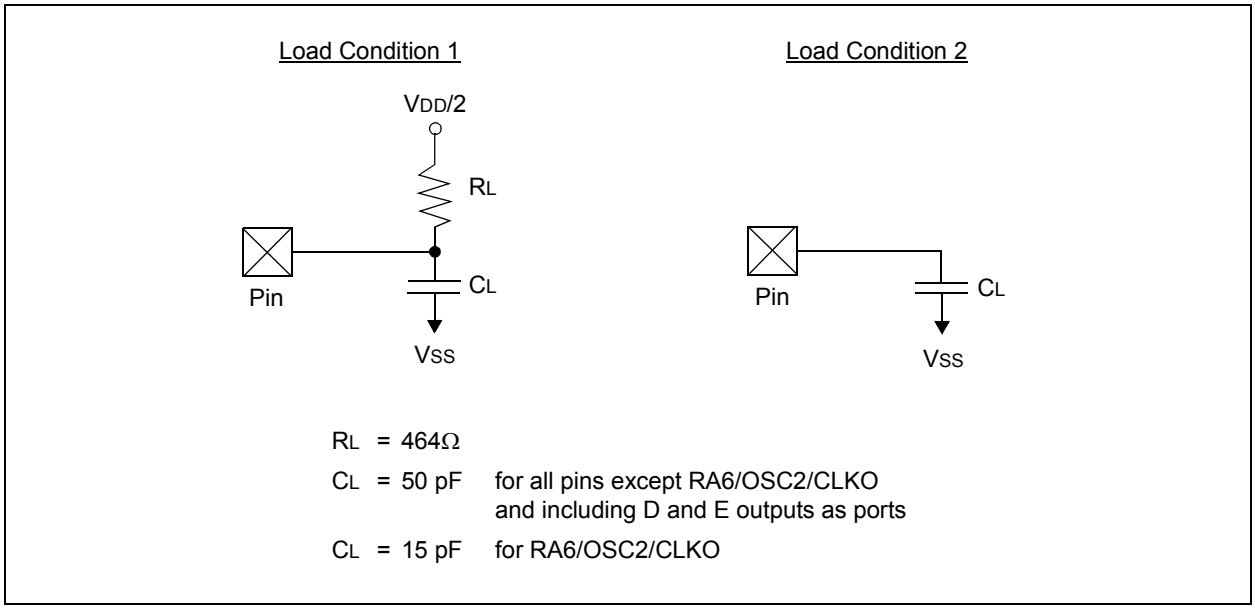
26.4.2 TIMING CONDITIONS

The temperature and voltages specified in Table 26-5 apply to all timing specifications unless otherwise noted. Figure 26-3 specifies the load conditions for the timing specifications.

TABLE 26-5: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

AC CHARACTERISTICS	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial Operating voltage VDD range as described in **Section 26.1** and **Section 26.3**.

FIGURE 26-3: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



26.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 26-4: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)

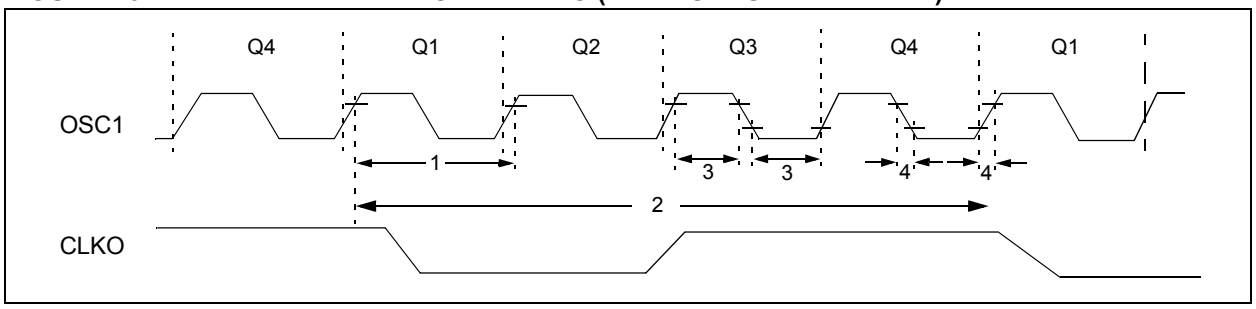


TABLE 26-6: EXTERNAL CLOCK TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	FOSC	External CLKI Frequency(1)	DC	40	MHz	ECPLL Oscillator mode
		Oscillator Frequency(1)	DC	40	MHz	HSPLL Oscillator mode
1	TOSC	External CLKI Period(1)	25	—	ns	EC Oscillator mode
		Oscillator Period(1)	25	250	ns	HS Oscillator mode
2	TCY	Instruction Cycle Time(1)	100	—	ns	TCY = 4/FOSC, Industrial
3	TOSL, TOSH	External Clock in (OSC1) High or Low Time	10	—	ns	EC Oscillator mode
4	TOSR, TOSF	External Clock in (OSC1) Rise or Fall Time	—	7.5	ns	EC Oscillator mode

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

TABLE 26-7: PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 2.15V TO 3.6V)

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
F10	FOSC	Oscillator Frequency Range	4	—	10	MHz	HS mode only
F11	FSYS	On-Chip VCO System Frequency	16	—	40	MHz	HS mode only
F12	t_{rc}	PLL Start-up Time (Lock Time)	—	—	2	ms	
F13	ΔCLK	CLKO Stability (Jitter)	-2	—	+2	%	

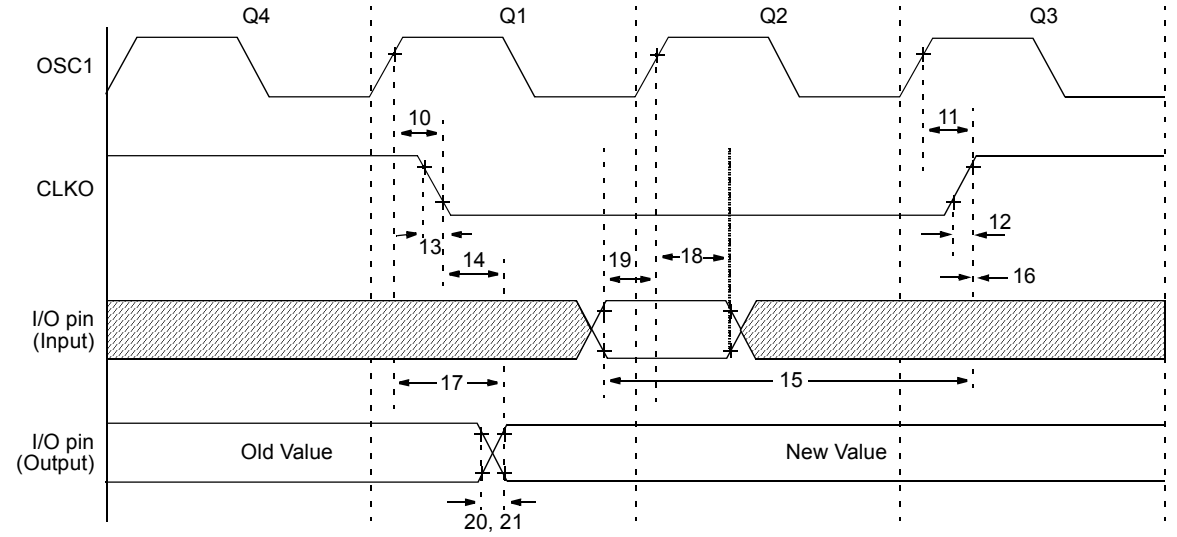
† Data in "Typ" column is at 3.3V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 26-8: INTERNAL RC ACCURACY (INTOSC AND INTRC SOURCES)

PIC18F85J11 Family (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	**Device**	**Min**	**Typ**	**Max**	**Units**	**Conditions**	
	INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz, 31 kHz(1)						
	All devices	-2	+/-1	2	%	+25°C	VDD = 2.7-3.3V
		-5	—	5	%	-10°C to +85°C	VDD = 2.0-3.3V
		-10	+/-1	10	%	-40°C to +85°C	VDD = 2.0-3.3V
	INTRC Accuracy @ Freq = 31 kHz(1)						
	All devices	26.562	—	35.938	kHz	-40°C to +85°C	VDD = 2.0-3.3V

Note 1: The accuracy specification of the 31 kHz clock is determined by which source is providing it at a given time. When INTSRC (OSCTUNE<7>) is '1', use the INTOSC accuracy specification. When INTSRC is '0', use the INTRC accuracy specification.

FIGURE 26-5: CLKO AND I/O TIMING



Note: Refer to Figure 26-3 for load conditions.

TABLE 26-9: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
10	TosH2ckL	OSC1 ↑ to CLKO ↓	—	75	200	ns	**(Note 1)**
11	TosH2ckH	OSC1 ↑ to CLKO ↑	—	75	200	ns	**(Note 1)**
12	TckR	CLKO Rise Time	—	15	30	ns	**(Note 1)**
13	TckF	CLKO Fall Time	—	15	30	ns	**(Note 1)**
14	TckL2ioV	CLKO ↓ to Port Out Valid	—	—	0.5 TCY + 20	ns	
15	TioV2ckH	Port In Valid Before CLKO ↑	0.25 TCY + 25	—	—	ns	
16	TckH2ioI	Port In Hold After CLKO ↑	0	—	—	ns	
17	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port Out Valid	—	50	150	ns	
18	TosH2ioI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	100	—	—	ns	
19	TioV2osH	Port Input Valid to OSC1 ↑ (I/O in setup time)	0	—	—	ns	
20	TioR	Port Output Rise Time	—	—	6	ns	
21	TioF	Port Output Fall Time	—	—	5	ns	
22†	TINP	INTx Pin High or Low Time	TCY	—	—	ns	
23†	TRBP	RB<7:4> Change INTx High or Low Time	TCY	—	—	ns	

† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in EC mode, where CLKO output is 4 x TOSC.

FIGURE 26-6: PROGRAM MEMORY FETCH TIMING DIAGRAM (8-BIT)

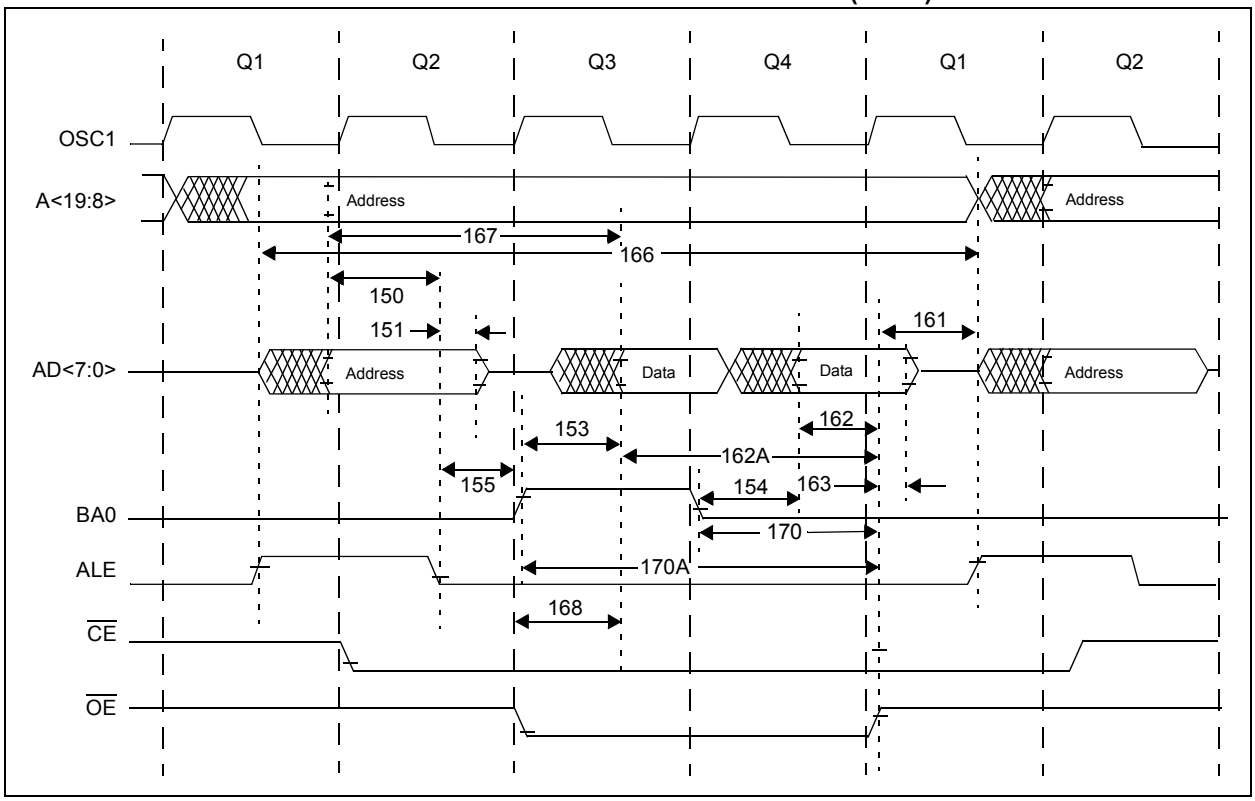
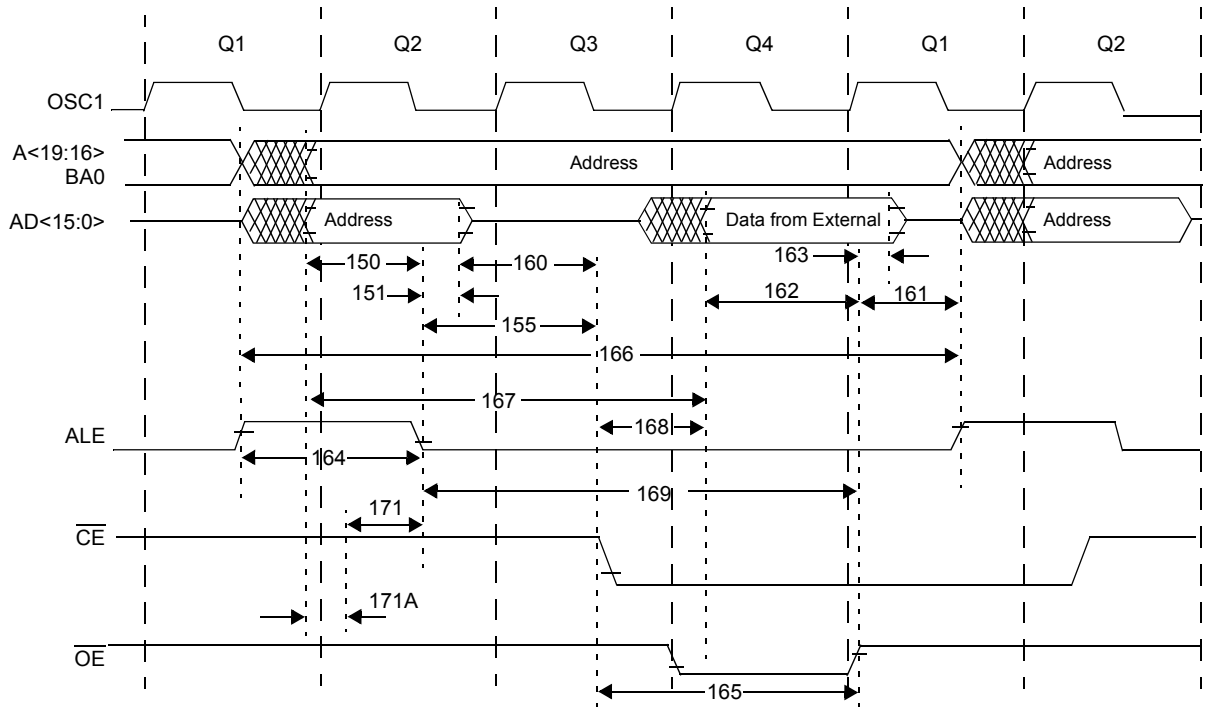


TABLE 26-7: PROGRAM MEMORY FETCH TIMING REQUIREMENTS (8-BIT)

Param No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
153	BA01	BA0 ↑ to Most Significant Data Valid	0.125 TCY	—	—	ns
154	BA02	BA0 ↓ to Least Significant Data Valid	0.125 TCY	—	—	ns
155	TalL2oeL	ALE ↓ to $\overline{OE}$ ↓	0.125 TCY	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ to A/D Driven	0.125 TCY – 5	—	—	ns
162	TadV2oeH	Least Significant Data Valid Before $\overline{OE}$ ↑ (data setup time	20	—	—	ns
162A	TadV2oeH	Most Significant Data Valid Before $\overline{OE}$ ↑ (data setup time)	0.25 TCY + 20	—	—	ns
163	ToeH2adI	$\overline{OE}$ ↑ to Data in Invalid (data Hold Time)	0	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
167	TACC	Address Valid to Data Valid	0.5 TCY – 10	—	—	ns
168	Toe	$\overline{OE}$ ↓ to Data Valid	—	—	0.125 TCY + 5	ns
170	TubH2oeH	BA0 = 0 Valid Before $\overline{OE}$ ↑	0.25 TCY	—	—	ns
170A	TubL2oeH	BA0 = 1 Valid Before $\overline{OE}$ ↑	0.5 TCY	—	—	ns

FIGURE 26-8: PROGRAM MEMORY READ TIMING DIAGRAM

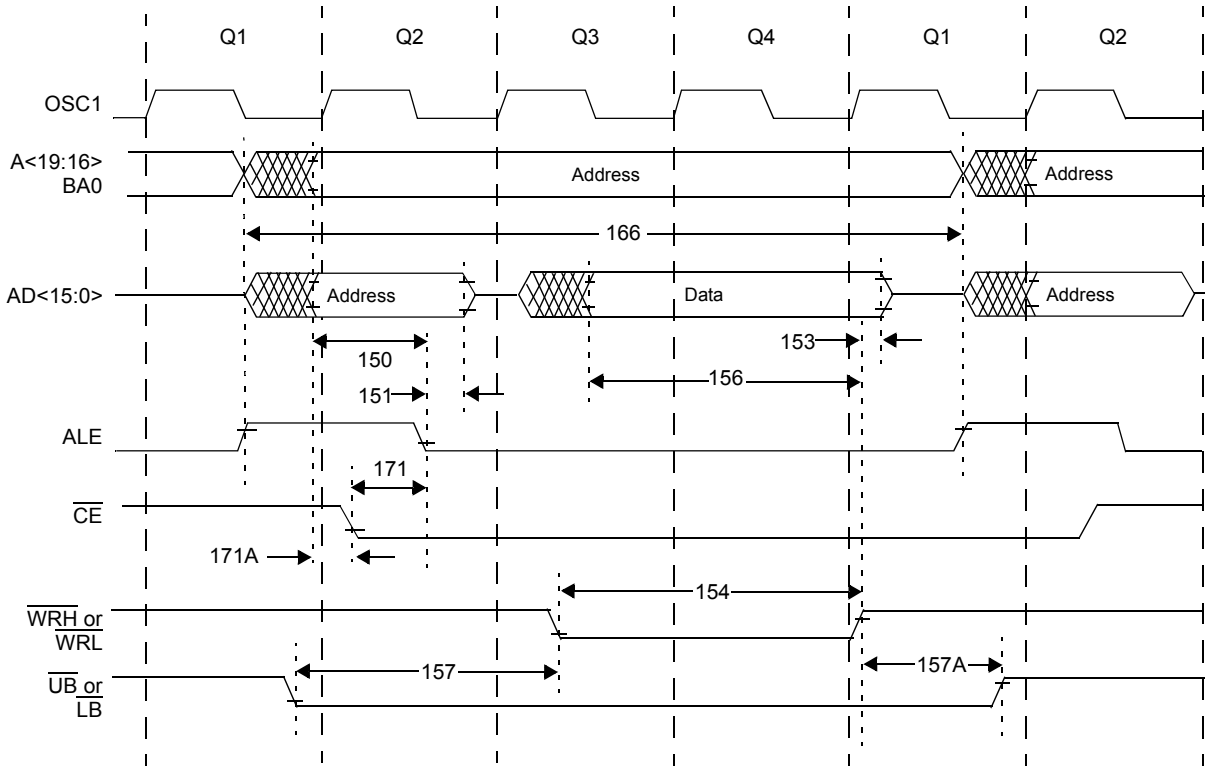


Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C unless otherwise stated.

TABLE 26-10: CLKO AND I/O TIMING REQUIREMENTS

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adI	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
155	TalL2oeL	ALE ↓ to $\overline{OE}$ ↓	10	0.125 TCY	—	ns
160	TadZ2oeL	AD High-Z to $\overline{OE}$ ↓ (bus release to $\overline{OE}$)	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ to AD Driven	0.125 TCY – 5	—	—	ns
162	TadV2oeH	LS Data Valid before $\overline{OE}$ ↑ (data setup time)	20	—	—	ns
163	ToeH2adI	$\overline{OE}$ ↑ to Data In Invalid (data hold time)	0	—	—	ns
164	TalH2alL	ALE Pulse Width	—	0.25 TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ Pulse Width	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
167	Tacc	Address Valid to Data Valid	0.75 TCY – 25	—	—	ns
168	Toe	$\overline{OE}$ ↓ to Data Valid		—	0.5 TCY – 25	ns
169	TalL2oeH	ALE ↓ to $\overline{OE}$ ↑	0.625 TCY – 10	—	0.625 TCY + 10	ns
171	TalH2csL	Chip Enable Active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD Valid to Chip Enable Active	—	—	10	ns

FIGURE 26-9: PROGRAM MEMORY WRITE TIMING DIAGRAM

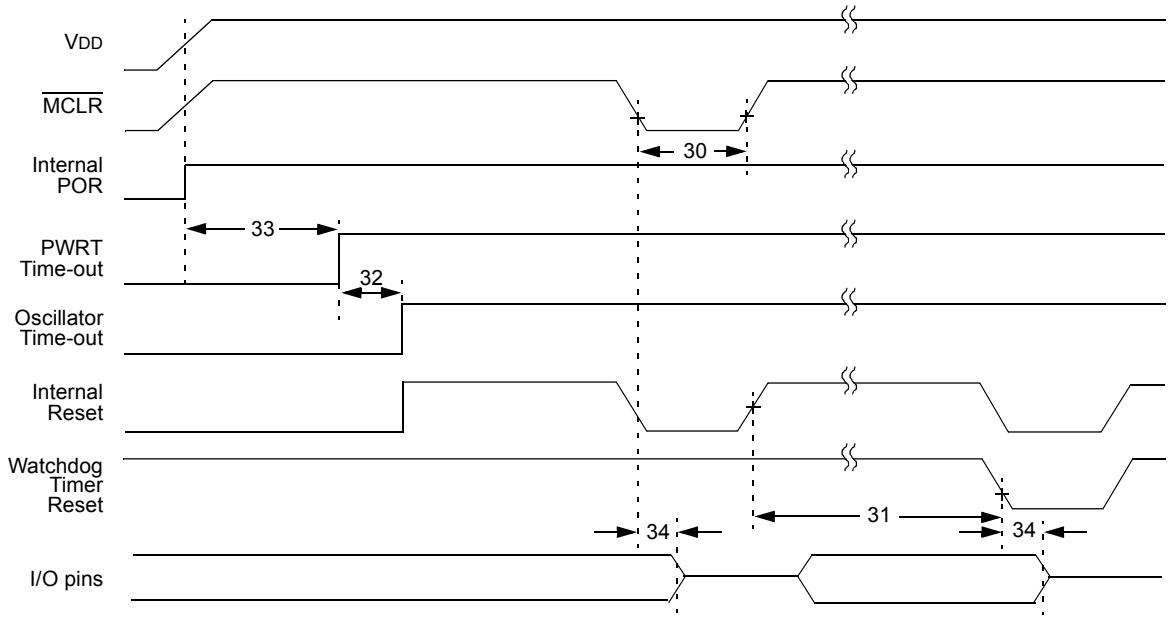


Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C unless otherwise stated.

TABLE 26-11: PROGRAM MEMORY WRITE TIMING REQUIREMENTS

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
153	TwrH2adl	$\overline{WRn}$ ↑ to Data Out Invalid (data hold time)	5	—	—	ns
154	TwrL	$\overline{WRn}$ Pulse Width	0.5 TCY – 5	0.5 TCY	—	ns
156	TadV2wrH	Data Valid before $\overline{WRn}$ ↑ (data setup time)	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	Byte Select Valid before $\overline{WRn}$ ↓ (byte select setup time)	0.25 TCY	—	—	ns
157A	TwrH2bsl	$\overline{WRn}$ ↑ to Byte Select Invalid (byte select hold time)	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
171	TalH2csL	Chip Enable Active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD Valid to Chip Enable Active	—	—	10	ns

FIGURE 26-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



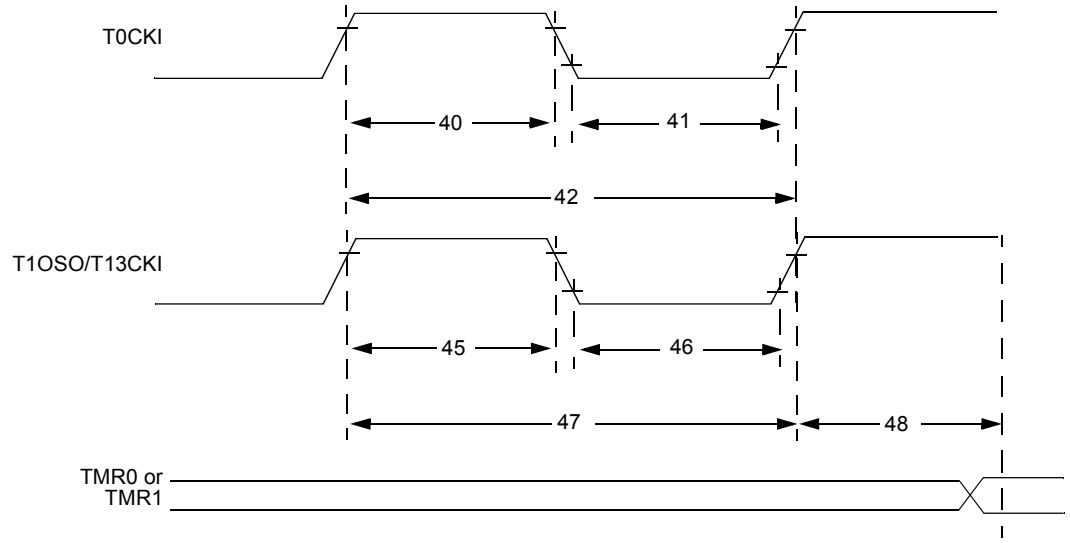
Note: Refer to Figure 26-3 for load conditions.

TABLE 26-12: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2 TCY	10 TCY	—		**(Note 1)**
31	TWDT	Watchdog Timer Time-out Period (no postscaler)	3.4	4.0	4.6	ms	
32	TOST	Oscillator Start-up Timer Period	1024 TOSC	—	1024 TOSC		TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	45.8	65.5	85.2	ms	
34	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	2	—	μs	
38	TCSD	CPU Start-up Time	—	10	—	μs	
			—	200	—	μs	Voltage regulator enabled and put to Sleep
39	TIOBST	Time for INTOSC to Stabilize	—	1	—	μs	

Note 1: To ensure device Reset, $\overline{\text{MCLR}}$ must be low for at least 2 TCY or 400 μs, which ever is lower.

FIGURE 26-11: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

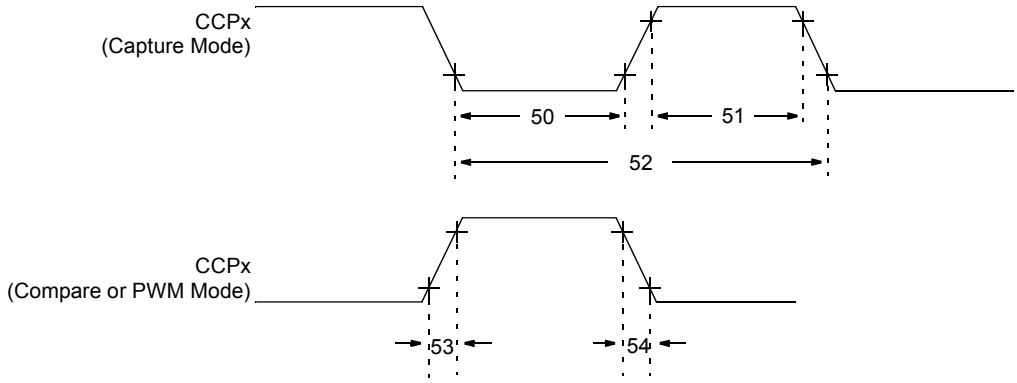


Note: Refer to Figure 26-3 for load conditions.

TABLE 26-13: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
40	TT0H	T0CKI High Pulse Width	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
41	TT0L	T0CKI Low Pulse Width	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
42	TT0P	T0CKI Period	No prescaler	TCY + 10	—	ns	
			With prescaler	Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4,..., 256)
45	TT1H	T13CKI High Time	Synchronous, no prescaler	0.5 TCY + 20	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
46	TT1L	T13CKI Low Time	Synchronous, no prescaler	0.5 TCY + 5	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
47	TT1P	T13CKI Input Period	Synchronous	Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous	60	—	ns	
	FT1	T13CKI Oscillator Input Frequency Range		DC	50	kHz	
48	TCKE2TMRI	Delay from External T13CKI Clock Edge to Timer Increment		2 TOSC	7 TOSC	—	

FIGURE 26-12: CAPTURE/COMPARE/PWM TIMINGS (CCP1, CCP2 MODULES)

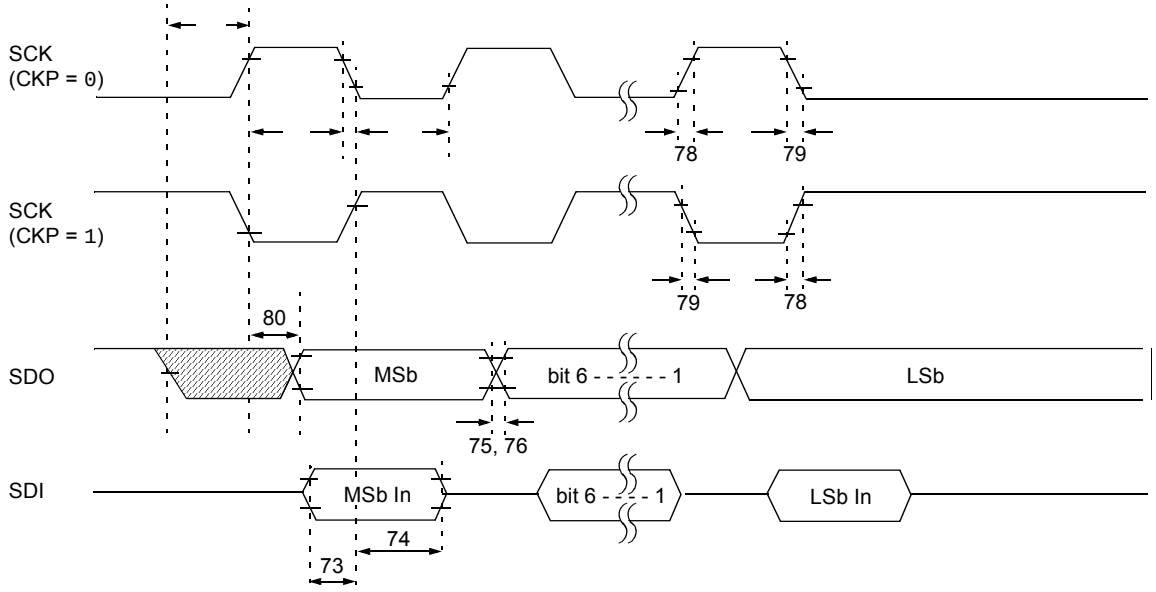


Note: Refer to Figure 26-3 for load conditions.

TABLE 26-14: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP1, CCP2 MODULES)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
50	TccL	CCPx Input Low Time	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
51	TccH	CCPx Input High Time	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
52	TccP	CCPx Input Period		$\frac{3\ T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time		—	25	ns	
54	TccF	CCPx Output Fall Time		—	25	ns	

FIGURE 26-13: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)



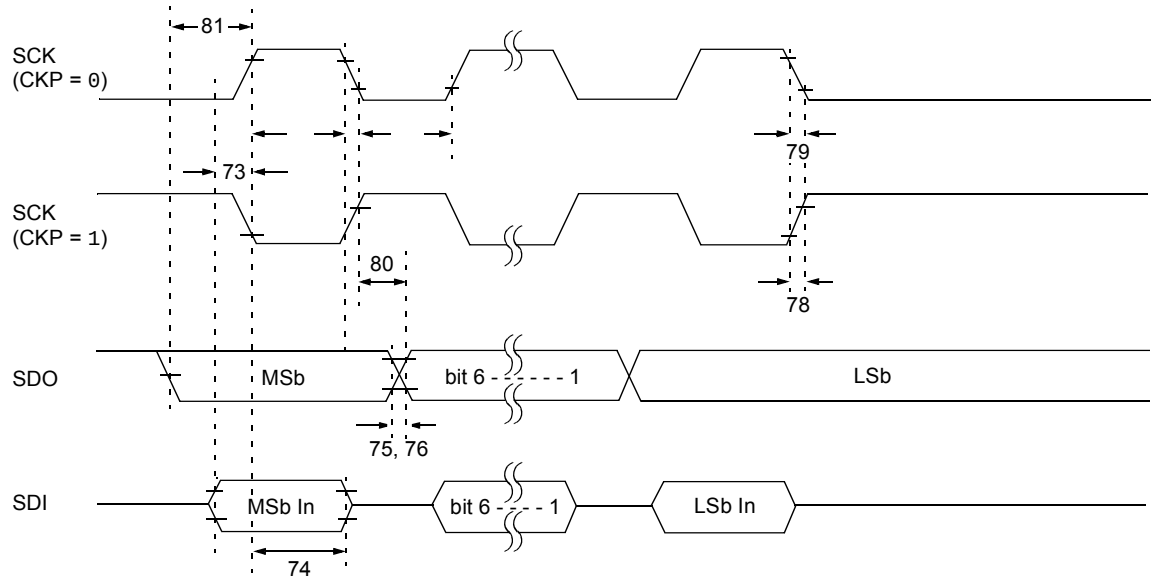
Note: Refer to Figure 26-3 for load conditions.

TABLE 26-15: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDI Data Input to SCK Edge	20	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 TCY + 40	—	ns	**(Note 1)**
74	TSCH2DIL, TSCL2DIL	Hold Time of SDI Data Input to SCK Edge	40	—	ns	
75	TDOR	SDO Data Output Rise Time	—	25	ns	
76	TDOF	SDO Data Output Fall Time	—	25	ns	
78	TSCR	SCK Output Rise Time (Master mode)	—	25	ns	
79	TSCF	SCK Output Fall Time (Master mode)	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDO Data Output Valid after SCK Edge	—	50	ns	

Note 1: Only if Parameter #71A and #72A are used.

FIGURE 26-14: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)



Note: Refer to Figure 26-3 for load conditions.

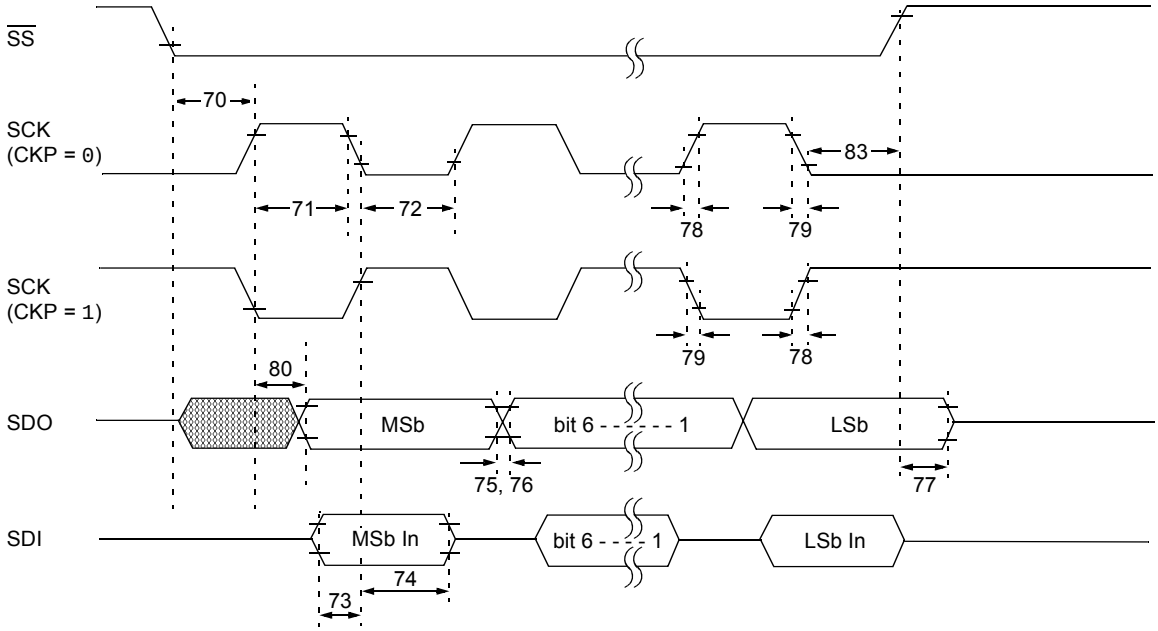
TABLE 26-16: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDI Data Input to SCK Edge	20	—	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	40	—	ns	
75	TDOR	SDO Data Output Rise Time	—	25	ns	
76	TDOF	SDO Data Output Fall Time	—	25	ns	
78	TSCR	SCK Output Rise Time (Master mode)	—	25	ns	
79	TSCF	SCK Output Fall Time (Master mode)	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDO Data Output Valid after SCK Edge	—	50	ns	
81	TDOV2SCH, TDOV2SCL	SDO Data Output Setup to SCK Edge	TCY	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 26-15: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 26-3 for load conditions.

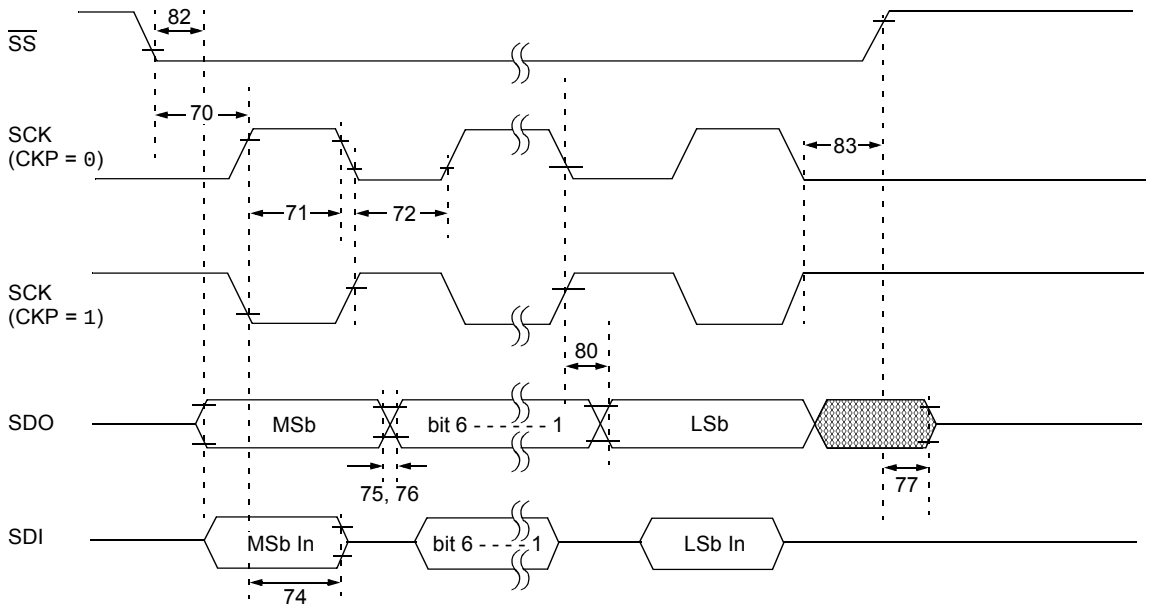
TABLE 26-17: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		3 Tcy	—	ns	
70A	TssL2WB	$\overline{SS}$ to Write to SSPBUF		3 Tcy	—	ns	
71	TscH	SCK Input High Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
71A			Single byte	40	—	ns	**(Note 1)**
72	TscL	SCK Input Low Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
72A			Single byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 Tcy + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time		—	25	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO Output High-Impedance		10	50	ns	
78	TscR	SCK Output Rise Time (Master mode)		—	25	ns	
79	TscF	SCK Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid After SCK Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ After SCK Edge		1.5 Tcy + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 26-16: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)



Note: Refer to Figure 26-3 for load conditions.

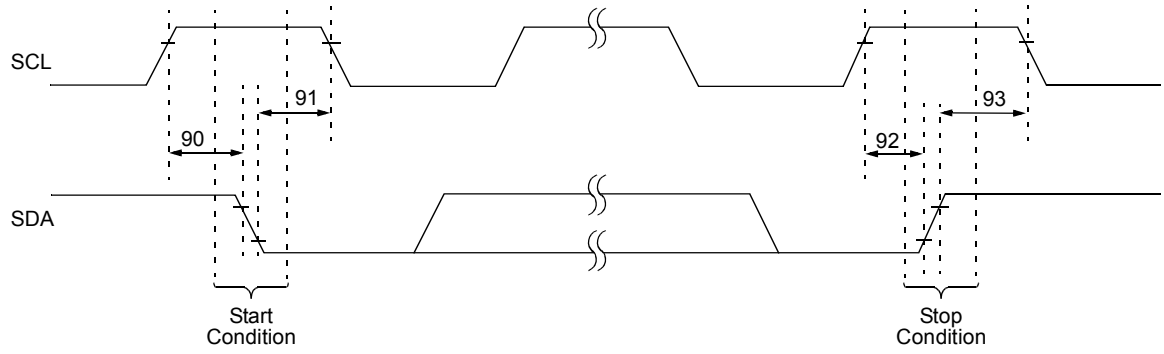
TABLE 26-18: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		3 TCY	—	ns	
70A	TssL2WB	$\overline{SS}$ to Write to SSPBUF		3 TCY	—	ns	
71	TscH	SCK Input High Time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single byte	40	—	ns	**(Note 1)**
72	TscL	SCK Input Low Time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single byte	40	—	ns	**(Note 1)**
73A	TB2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time		—	25	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO Output High-Impedance		10	50	ns	
78	TscR	SCK Output Rise Time (Master mode)		—	25	ns	
79	TscF	SCK Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid After SCK Edge		—	50	ns	
82	TssL2doV	SDO Data Output Valid After $\overline{SS}$ ↓ Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ After SCK Edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 26-17: I²C™ BUS START/STOP BITS TIMING

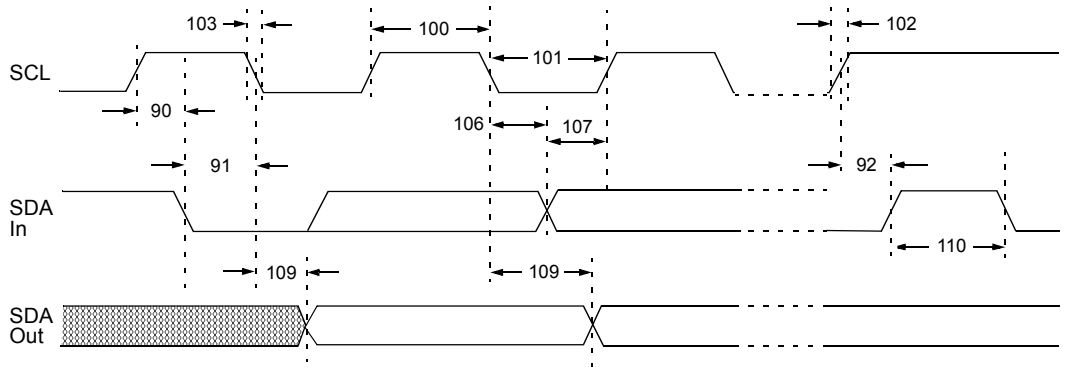


Note: Refer to Figure 26-3 for load conditions.

TABLE 26-19: I²C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4700	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4700	—	ns	
			400 kHz mode	600	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	ns	
			400 kHz mode	600	—		

FIGURE 26-18: I²C™ BUS DATA TIMING



Note: Refer to Figure 26-3 for load conditions.

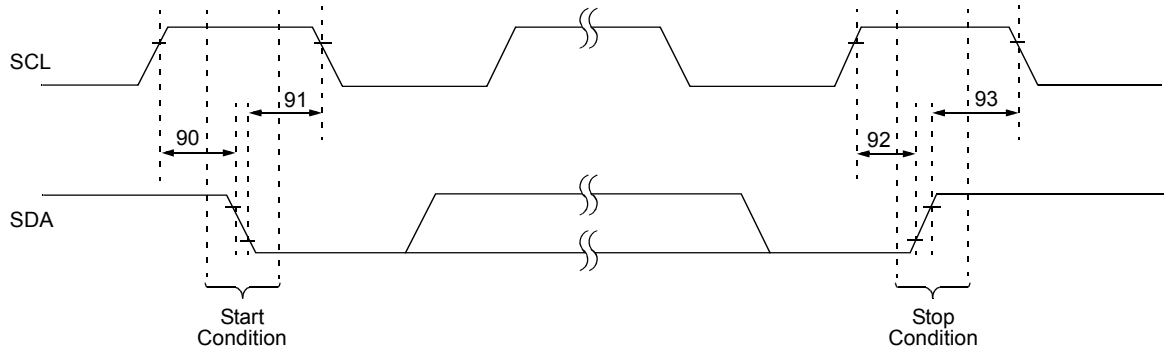
TABLE 26-20: I²C™ BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	µs	
			400 kHz mode	0.6	—	µs	
			MSSP module	1.5 TCY	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	µs	
			400 kHz mode	1.3	—	µs	
			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	**(Note 2)**
			400 kHz mode	100	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	µs	
			400 kHz mode	0.6	—	µs	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	µs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I²C™ bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the 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 I²C bus specification), before the SCL line is released.

FIGURE 26-19: MSSP I²C™ BUS START/STOP BITS TIMING WAVEFORMS



Note: Refer to Figure 26-3 for load conditions.

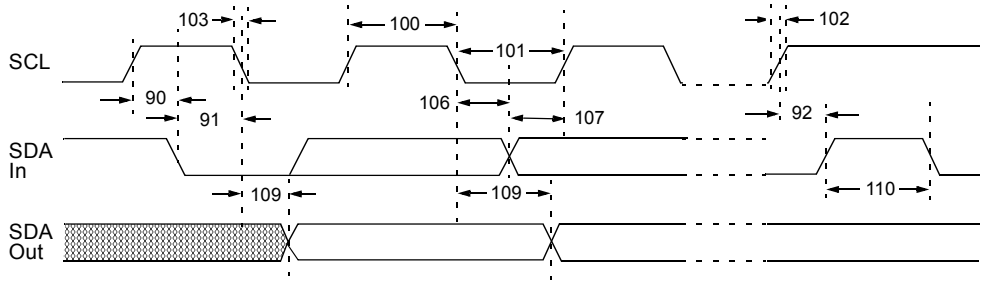
TABLE 26-21: MSSP I²C™ BUS START/STOP BITS REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—		

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

2: A minimum 16 MHz FOSC is required for 1 MHz I²C.

FIGURE 26-20: MSSP I²C™ BUS DATA TIMING



Note: Refer to Figure 26-3 for load conditions.

TABLE 26-22: MSSP I²C™ BUS DATA REQUIREMENTS

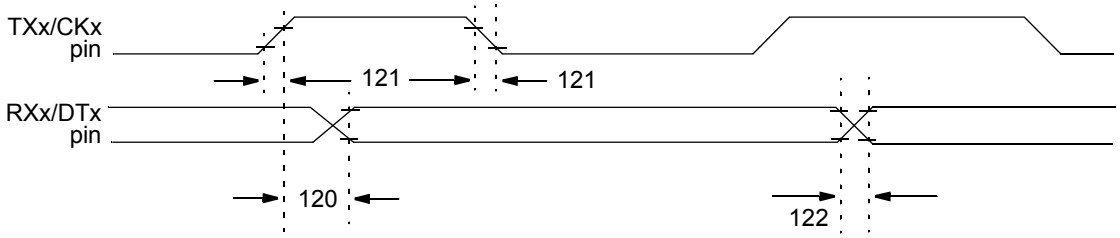
Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—	—	
101	TLOW	Clock Low Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—	—	
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1,2)	—	300	ns	
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1,2)	—	100	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—	—	
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—	—	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	μs	
			1 MHz mode(1,2)	—	—	ns	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 3)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1,2)	—	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1,2)	2(TOSC)(BRG + 1)	—	—	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1,2)	—	—	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	
			1 MHz mode(1,2)	—	—	μs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

2: A minimum 16 MHz FOSC is required for 1 MHz I²C.

3: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the 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.

FIGURE 26-21: EUSART/AUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

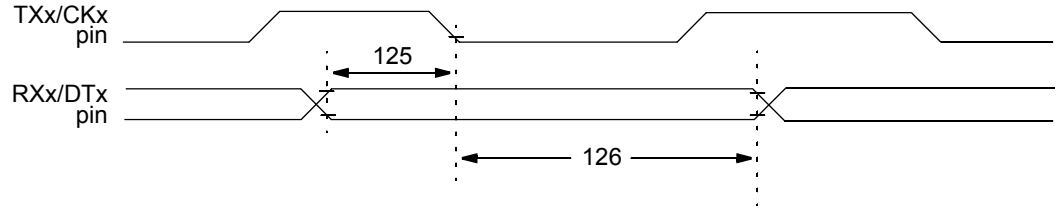


Note: Refer to Figure 26-3 for load conditions.

TABLE 26-23: EUSART/AUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
120	TCKH2DTV	SYNC XMIT (MASTER and SLAVE) Clock High to Data Out Valid	—	40	ns	
121	TCKRF	Clock Out Rise Time and Fall Time (Master mode)	—	20	ns	
122	TDTRF	Data Out Rise Time and Fall Time	—	20	ns	

FIGURE 26-22: EUSART/AUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 26-3 for load conditions.

TABLE 26-24: EUSART/AUSART 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	

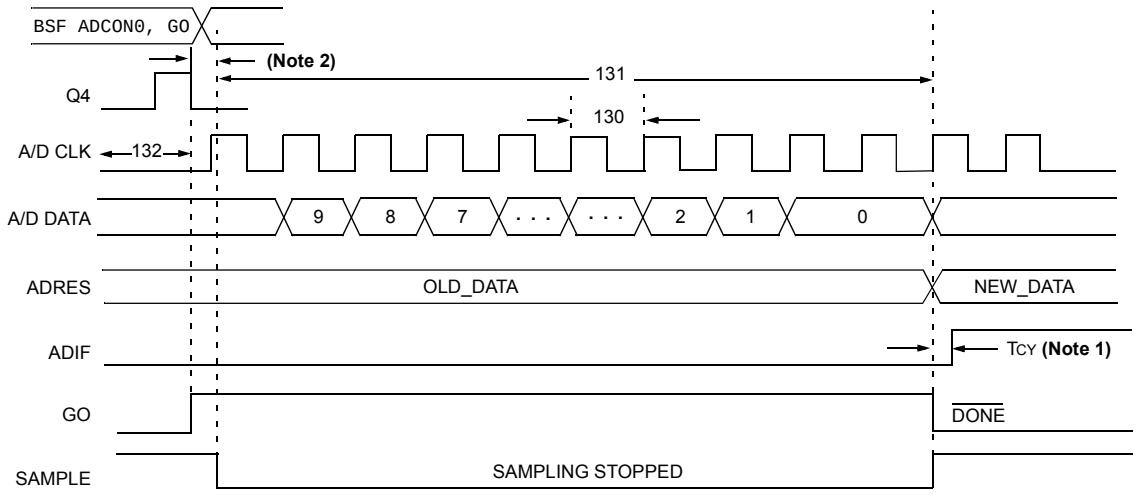
TABLE 26-25: A/D CONVERTER CHARACTERISTICS: PIC18F85J11 FAMILY (INDUSTRIAL)

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
A01	NR	Resolution	—	—	10	bits	
A03	EIL	Integral Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A04	EDL	Differential Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A06	EOFF	Offset Error	—	—	<±3	LSb	ΔVREF ≥ 3.0V
A07	EGN	Gain Error	—	—	<±3	LSb	ΔVREF ≥ 3.0V
A10	—	Monotonicity	Guaranteed(1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	Reference Voltage Range (VREFH – VREFL)	2.0 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
A21	VREFH	Reference Voltage High	VSS + ΔVREF	—	VDD	V	
A22	VREFL	Reference Voltage Low	VSS – 0.3V	—	VDD – 3.0V	V	
A25	VAIN	Analog Input Voltage	VREFL	—	VREFH	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	2.5	kΩ	
A50	IREF	VREF Input Current(2)	— —	— —	5 150	μA μA	During VAIN acquisition During A/D conversion cycle

Note 1: The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

2: VREFH current is from the RA3/AN3/VREF+ pin or VDD, whichever is selected as the VREFH source. VREFL current is from the RA2/AN2/VREF- pin or VSS, whichever is selected as the VREFL source.

FIGURE 26-23: A/D CONVERSION TIMING



Note 1: If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

2: This is a minimal RC delay (typically 100 ns), which also disconnects the holding capacitor from the analog input.

TABLE 26-26: A/D CONVERSION REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
130	TAD	A/D Clock Period	0.7	25.0(1)	µs	TOSC based, VREF ≥ 3.0V
			—	1	µs	A/D RC mode
131	TCNV	Conversion Time (not including acquisition time)(2)	11	12	TAD	
132	TACQ	Acquisition Time(3)	1.4	— —	µs	-40°C to +85°C
135	TSWC	Switching Time from Convert → Sample	—	**(Note 4)**		
137	TDIS	Discharge Time	0.2	—	µs	

Note 1: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

2: ADRES registers may be read on the following TCY cycle.

3: The time for the holding capacitor to acquire the "New" input voltage when the voltage changes full scale after the conversion (VDD to VSS or VSS to VDD). The source impedance (RS) on the input channels is 50Ω.

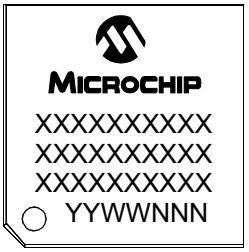
4: On the following cycle of the device clock.

NOTES:

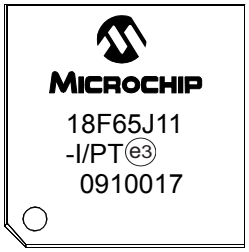
27.0 PACKAGING INFORMATION

27.1 Package Marking Information

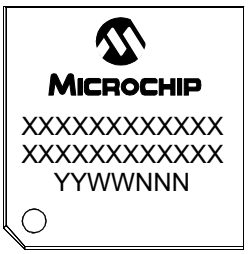
64-Lead TQFP



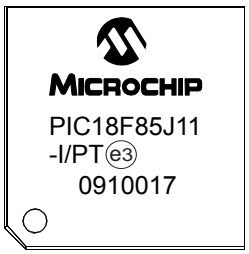
Example



80-Lead TQFP



Example



Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package.

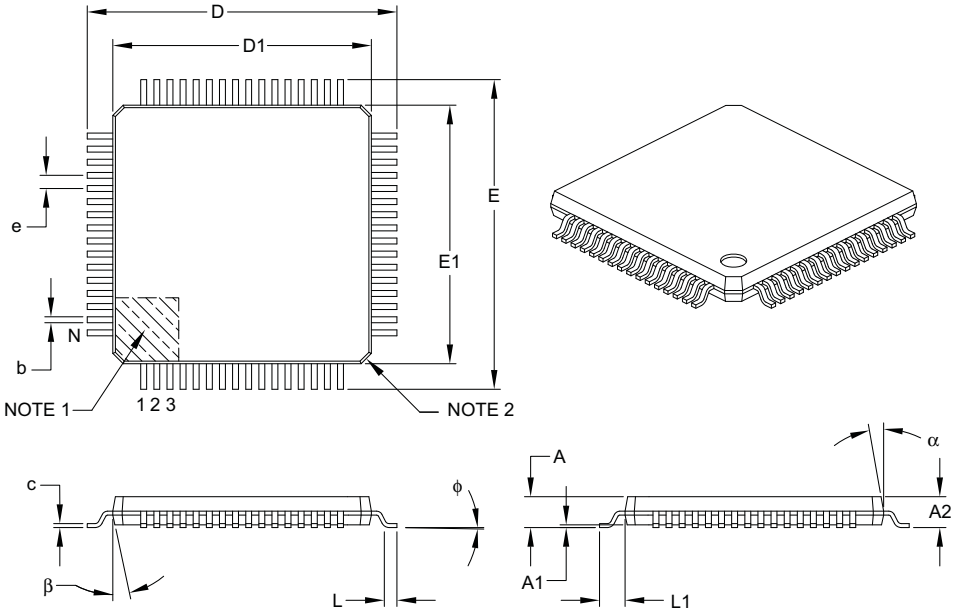
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

27.2 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



Units		MILLIMETERS		
Dimension Limits		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	φ	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	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

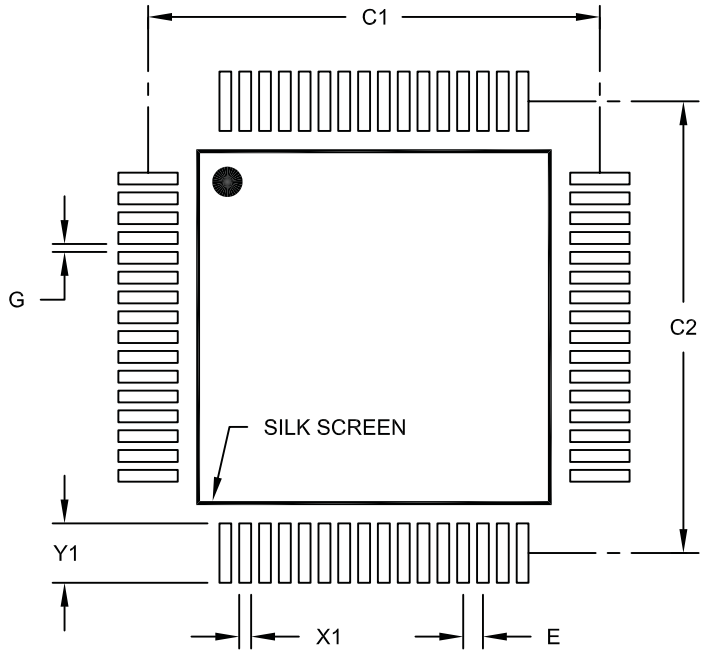
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085B

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

Units		MILLIMETERS		
Dimension Limits		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:

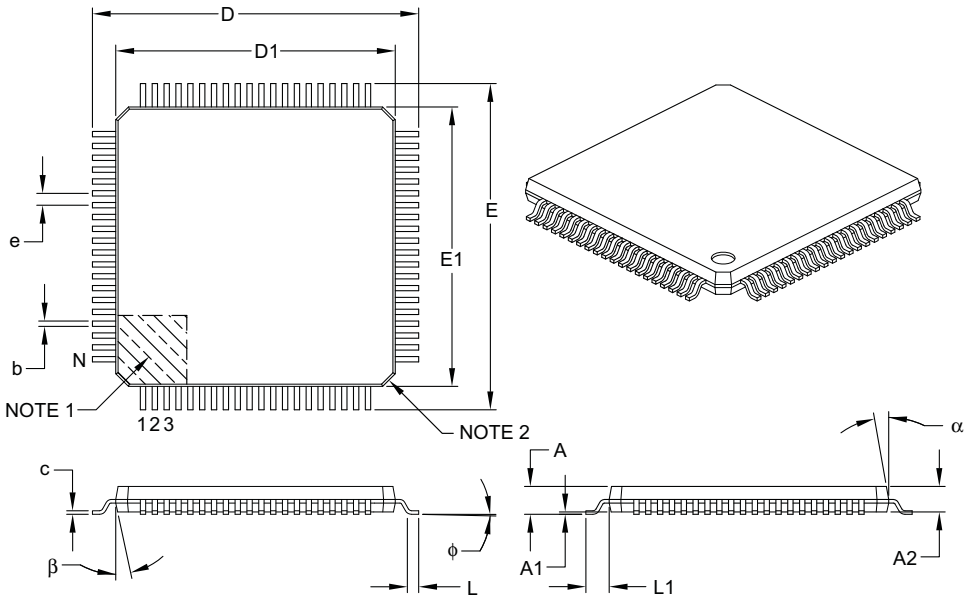
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2085A

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
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	ϕ	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	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

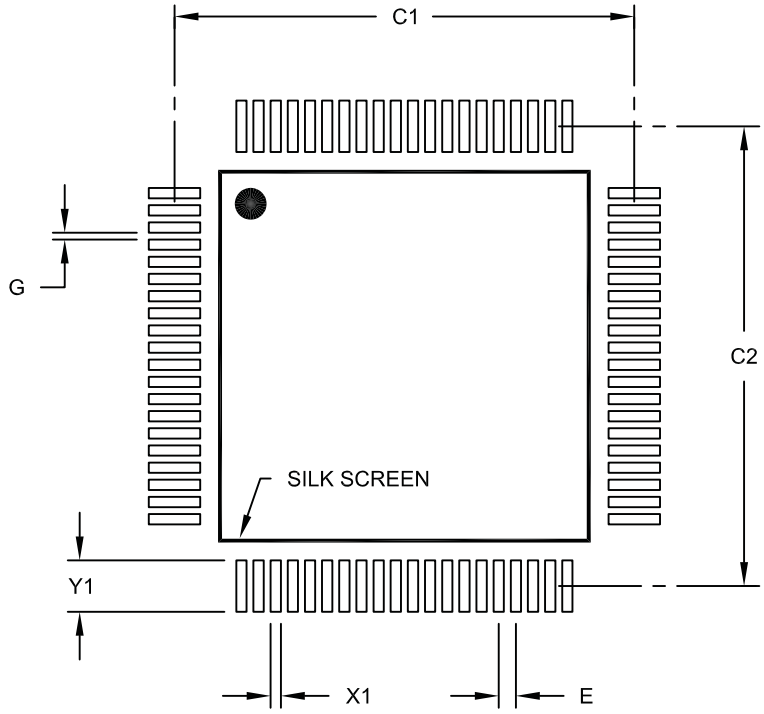
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-092B

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



RECOMMENDED LAND PATTERN

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

NOTES:

APPENDIX A: REVISION HISTORY

Revision A (October 2006)

Original data sheet for PIC18F85J11 family devices.

Revision B (March 2007)

Updated power-down and supply current electrical characteristics and package detail drawings.

Revision C (April 2007)

Updated electrical characteristics.

Revision D (February 2010)

Updated electrical characteristics and package detail illustrations. Minor text edits throughout document.

APPENDIX B: MIGRATION BETWEEN HIGH-END DEVICE FAMILIES

Devices in the PIC18F85J11 and PIC18F8722 families are very similar in their functions and feature sets. However, there are some potentially important differences which should be considered when migrating an application across device families to achieve a new design goal. These are summarized in Table B-1. The areas of difference which could have a major impact on migration are discussed in greater detail later in this section.

TABLE B-1: NOTABLE DIFFERENCES BETWEEN PIC18F85J11 AND PIC18F8722 FAMILIES

Characteristic	PIC18F85J11 Family	PIC18F8722 Family
Operating Frequency	40 MHz @ 2.35V	40 MHz @ 4.2V
Supply Voltage	2.0V-3.6V, dual voltage requirement	2.0V-5.5V
Operating Current	Low	Lower
Program Memory Endurance	1,000 write/erase cycles (typical)	100,000 write/erase cycles (typical)
I/O Sink/Source at 25 mA	PORTB and PORTC only	All ports
Input Voltage Tolerance on I/O pins	5.5V on digital only pins	V_{DD} on all I/O pins
I/O	68 (RF0 is not available)	70
Pull-ups	PORTB, PORTD, PORTE and PORTJ	PORTB
Oscillator Options	Limited options (EC, HS, PLL, flexible INTRC)	More options (EC, HS, XT, LP, RC, PLL, flexible INTRC)
Program Memory Retention	20 years (minimum)	40 years (minimum)
Self-Writes to Program Memory	Available	Available
Programming Time (normalized)	156 µs/byte (10 ms/64-byte block)	15.6 µs/byte (1 ms/64)
Programming Entry	Low voltage, key sequence	V_{PP} and LVP
Code Protection	Single block, all or nothing	Multiple code protection blocks
Configuration Words	Stored in last 4 words of program memory space	Stored in configuration space, starting at 300000h
Power-up Timer	Always on	Configurable
Data EEPROM	Use self-programming	Available
BOR	Simple BOR with voltage regulator	Programmable BOR
LVD	Simple LVD with voltage regulator	Available
A/D Channels	12	16
A/D Calibration	Required	Not required
Microprocessor mode (EMB)	Self-calibration feature	Available
External Memory Addressing	Address shifting available	Address shifting not available
In-Circuit Emulation	Not available	Available

B.1 Power Requirement Differences

The most significant difference between the PIC18F85J11 and PIC18F8722 device families is the power requirements. PIC18F85J11 devices are designed on a smaller process; this results in lower maximum voltage and higher leakage current.

The operating voltage range for PIC18F85J11 devices is 2.0V to 3.6V. In addition, these devices have split power requirements: one for the core logic and one for the I/O. One of the VDD pins is separated for the core logic supply, VDDCORE. This pin has specific voltage and capacitor requirements as described in **Section 26.0 "Electrical Characteristics"**.

The current specifications for PIC18F85J11 devices are yet to be determined.

B.2 Pin Differences

There are several differences in the pinout between the PIC18F85J11 and the PIC18F8722 families:

- Input voltage tolerance
- Output current capabilities
- Available I/O

Pins on the PIC18F85J11 that have digital only input capability will tolerate voltages up to 5.5V and are thus tolerant to voltages above VDD. Table 11-1 in **Section 11.1 "I/O Port Pin Capabilities"** contains the complete list.

In addition to input differences, there are output differences as well. PIC18F85J11 devices have three classes of pin output current capability: high, medium and low. Not all I/O pins can source or sink equal levels of current. Only PORTB and PORTC support the 25 mA source/sink capability that is supported by all output pins on the PIC18F8722. Table 11-2 in **Section 11.1 "I/O Port Pin Capabilities"** contains the complete list of output capabilities.

There are additional differences in how some pin functions are implemented on PIC18F85J11 devices. First, the MCLR pin is dedicated only to MCLR and cannot be configured as an input (RG5). Finally, RF0 does not exist on PIC18F85J11 devices.

All of these pin differences (including power pin differences) should be accounted for when making a conversion between PIC18F8722 and PIC18F85J11 devices.

B.3 Oscillator Differences

PIC18F8722 and PIC18F85J11 family devices share a similar range of oscillator options. The major difference is that PIC18F85J11 family devices support a smaller number of primary (external) oscillator options, namely HS and EC Oscillator modes.

While both device families have an internal PLL that can be used with the primary oscillators, the PLL for the PIC18F85J11 family is not enabled as a device configuration option. Instead, it must be enabled in software.

The clocking differences should be considered when making a conversion between the PIC18F8722 and PIC18F85J11 device families.

B.4 Peripherals

Peripherals must also be considered when making a conversion between the PIC18F85J11 and the PIC18F8722:

- **External Memory Bus:** The External Memory Bus (EMB) on the PIC18F85J11 does not support Microcontroller mode; however, it does support external address offset.
- **A/D Converter:** There are only 12 channels on PIC18F85J11 devices. The converters for these devices also require a calibration step prior to normal operation.
- **Data EEPROM:** PIC18F85J11 devices do not have this module.
- **BOR:** PIC18F85J11 devices do not have a programmable BOR. Simple Brown-out Reset capability is provided through the use of the internal voltage regulator.
- **LVD:** PIC18F85J11 devices do not have a separate programmable LVD module. Simple, Low-Voltage Detection capability with a configurable interrupt is provided through the use of the internal voltage regulator.

INDEX

A

B

D

E

F

G

H

I

L

M

V

W

X

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://support.microchip.com

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC18F85J11 Family Literature Number: DS39774D

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?

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	X	/XX	XXX
Device	Temperature Range	Package	Pattern

Device	PIC18F63J11/64J11/65J11[(1)], PIC18F83J11/84J11/85J11[(1)], PIC18F63J11/64J11/65J11T[(2)], PIC18F83J11/84J11/85J11T[(2)]
Temperature Range	I = -40°C to +85°C (Industrial)
Package	PT = TQFP (Thin Quad Flatpack)
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)

Examples:

a) PIC18F85J11-I/PT 301 = Industrial temp., TQFP package, QTP pattern #301.

b) PIC18F63J11T-I/PT = Tape and reel, Industrial temp., TQFP package.

Note 1: F = Standard 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

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/05/10

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:
PIC18F63J11-I/PT PIC18F63J11T-I/PT PIC18F64J11-I/PT PIC18F64J11T-I/PT PIC18F65J11-I/PT PIC18F65J11T-I/PT PIC18F83J11-I/PT PIC18F83J11T-I/PT PIC18F84J11-I/PT PIC18F84J11T-I/PT PIC18F85J11-I/PT PIC18F85J11T-I/PT

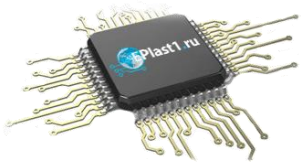
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Оперативные поставки широкого спектра электронных компонентов отечественного и импортного производства напрямую от производителей и с крупнейших мировых складов;
- Поставка более 17-ти миллионов наименований электронных компонентов;
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- Лицензия ФСБ на осуществление работ с использованием сведений, составляющих государственную тайну;
- Поставка специализированных компонентов (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Aeroflex, Peregrine, Syfer, Eurofarad, Texas Instrument, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits,General Dynamics и др.);

Помимо этого, одним из направлений компании «ЭлектроПласт» является направление «Источники питания». Мы предлагаем Вам помощь Конструкторского отдела:

- Подбор оптимального решения, техническое обоснование при выборе компонента;
- Подбор аналогов;
- Консультации по применению компонента;
- Поставка образцов и прототипов;
- Техническая поддержка проекта;
- Защита от снятия компонента с производства.



Как с нами связаться

Телефон: 8 (812) 309 58 32 (многоканальный)
Факс: 8 (812) 320-02-42
Электронная почта: org@eplast1.ru
Адрес: 198099, г. Санкт-Петербург, ул. Калинина, дом 2, корпус 4, литера А.