PIC18F97J60 Family Data Sheet

64/80/100-Pin, High-Performance, 1-Mbit Flash Microcontrollers with Ethernet

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: 978-1-61341-069-1

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.

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

PIC18F97J60 FAMILY

64/80/100-Pin High-Performance, 1-Mbit Flash Microcontrollers with Ethernet

Ethernet Features:

- IEEE 802.3™ Compatible Ethernet Controller
- Fully Compatible with 10/100/1000Base-T Networks
- Integrated MAC and 10Base-T PHY
- 8-Kbyte Transmit/Receive Packet Buffer SRAM
- Supports One 10Base-T Port
- Programmable Automatic Retransmit on Collision
- Programmable Padding and CRC Generation
- Programmable Automatic Rejection of Erroneous Packets
- Activity Outputs for 2 LED Indicators
- Buffer:
  - Configurable transmit/receive buffer size
  - Hardware-managed circular receive FIFO
  - Byte-wide random and sequential access
  - Internal DMA for fast memory copying
  - Hardware assisted checksum calculation for various protocols
- MAC:
  - Support for Unicast, Multicast and Broadcast packets
  - Programmable Pattern Match of up to 64 bytes within packet at user-defined offset
  - Programmable wake-up on multiple packet formats
- PHY:
  - Wave shaping output filter

Flexible Oscillator Structure:

- Selectable System Clock derived from Single 25 MHz External Source:
  - 2.778 to 41.667 MHz
- Internal 31 kHz Oscillator
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if oscillator stops
- Two-Speed Oscillator Start-up

External Memory Bus (100-pin devices only):

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

Peripheral Highlights:

- High-Current Sink/Source: 25 mA/25 mA on PORTB and PORTC
- Five Timer modules (Timer0 to Timer4)
- Four External Interrupt pins
- Two Capture/Compare/PWM (CCP) modules
- Three Enhanced Capture/Compare/PWM (ECCP) modules:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
- Up to Two Master Synchronous Serial Port (MSSP) modules supporting SPI (all 4 modes) and I^2C™ Master and Slave modes
- Up to Two Enhanced USART modules:
  - Supports RS-485, RS-232 and LIN/J2602
  - Auto-wake-up on Start bit
  - Auto-Baud Detect (ABD)
- 10-Bit, Up to 16-Channel Analog-to-Digital Converter module (A/D):
  - Auto-acquisition capability
  - Conversion available during Sleep
- Dual Analog Comparators with Input Multiplexing
- Parallel Slave Port (PSP) module (100-pin devices only)

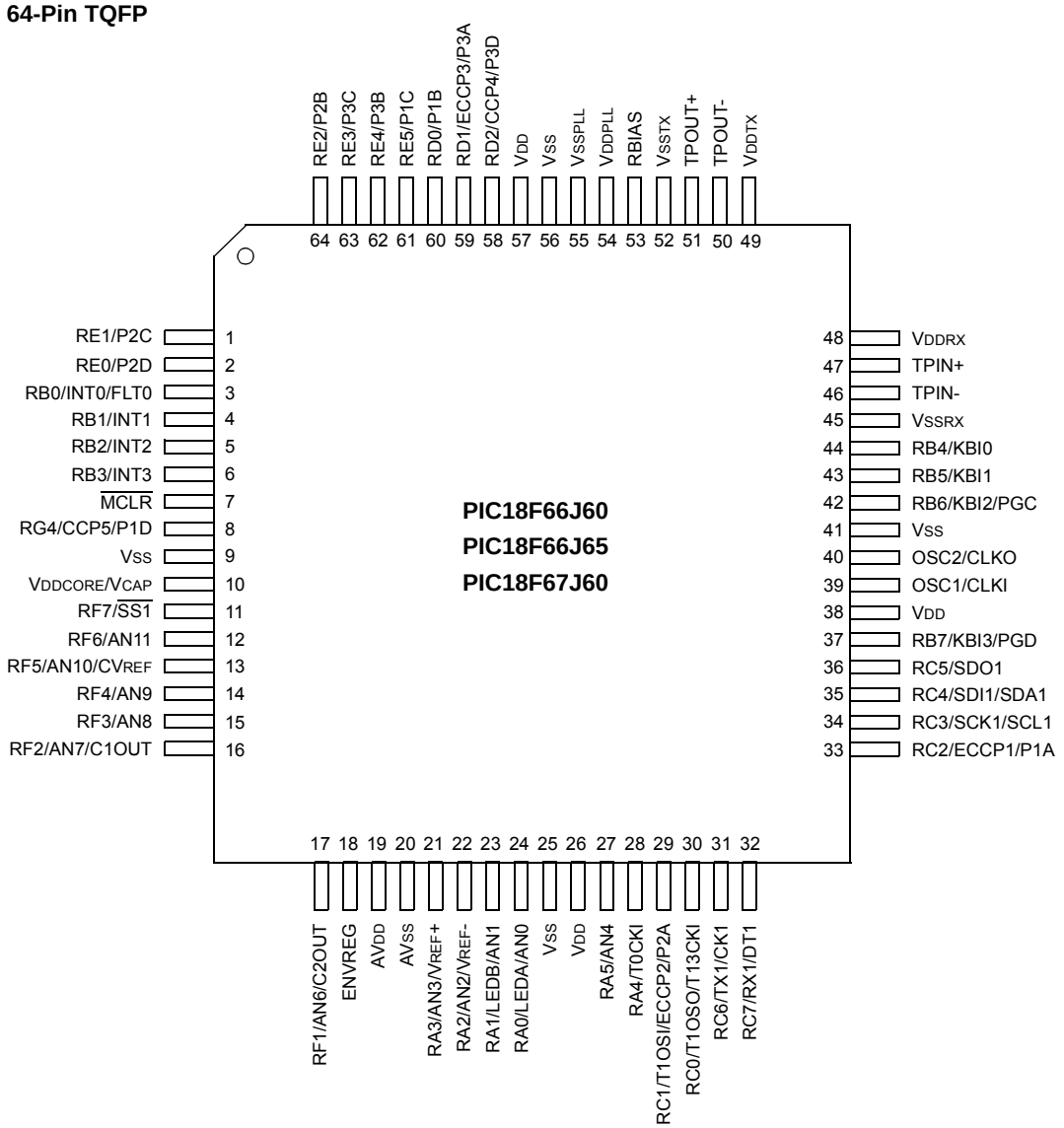
Special Microcontroller Features:

- 5.5V Tolerant Inputs (digital-only pins)
- Low-Power, High-Speed CMOS Flash Technology:
  - Self-reprogrammable under software control
- C compiler Optimized Architecture for Reentrant Code
- Power Management Features:
  - Run: CPU on, peripherals on
  - Idle: CPU off, peripherals on
  - Sleep: CPU off, peripherals off
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 134s
- Single-Supply 3.3V In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) with 3 Breakpoints via Two Pins
- Operating Voltage Range of 2.35V to 3.6V (3.1V to 3.6V using Ethernet module)
- On-Chip 2.5V Regulator

Device	Flash Program Memory (bytes)	SRAM Data Memory (bytes)	Ethernet TX/RX Buffer (bytes)	I/O	10-Bit A/D (ch)	CCP/ ECCP	MSSP			EUSART	Comparators	Timers 8/16-Bit	PSP	External Memory Bus
								SPI	Master I^2C™					
PIC18F66J60	64K	3808	8192	39	11	2/3	1	Y	Y	1	2	2/3	N	N
PIC18F66J65	96K	3808	8192	39	11	2/3	1	Y	Y	1	2	2/3	N	N
PIC18F67J60	128K	3808	8192	39	11	2/3	1	Y	Y	1	2	2/3	N	N
PIC18F86J60	64K	3808	8192	55	15	2/3	1	Y	Y	2	2	2/3	N	N
PIC18F86J65	96K	3808	8192	55	15	2/3	1	Y	Y	2	2	2/3	N	N
PIC18F87J60	128K	3808	8192	55	15	2/3	1	Y	Y	2	2	2/3	N	N
PIC18F96J60	64K	3808	8192	70	16	2/3	2	Y	Y	2	2	2/3	Y	Y
PIC18F96J65	96K	3808	8192	70	16	2/3	2	Y	Y	2	2	2/3	Y	Y
PIC18F97J60	128K	3808	8192	70	16	2/3	2	Y	Y	2	2	2/3	Y	Y

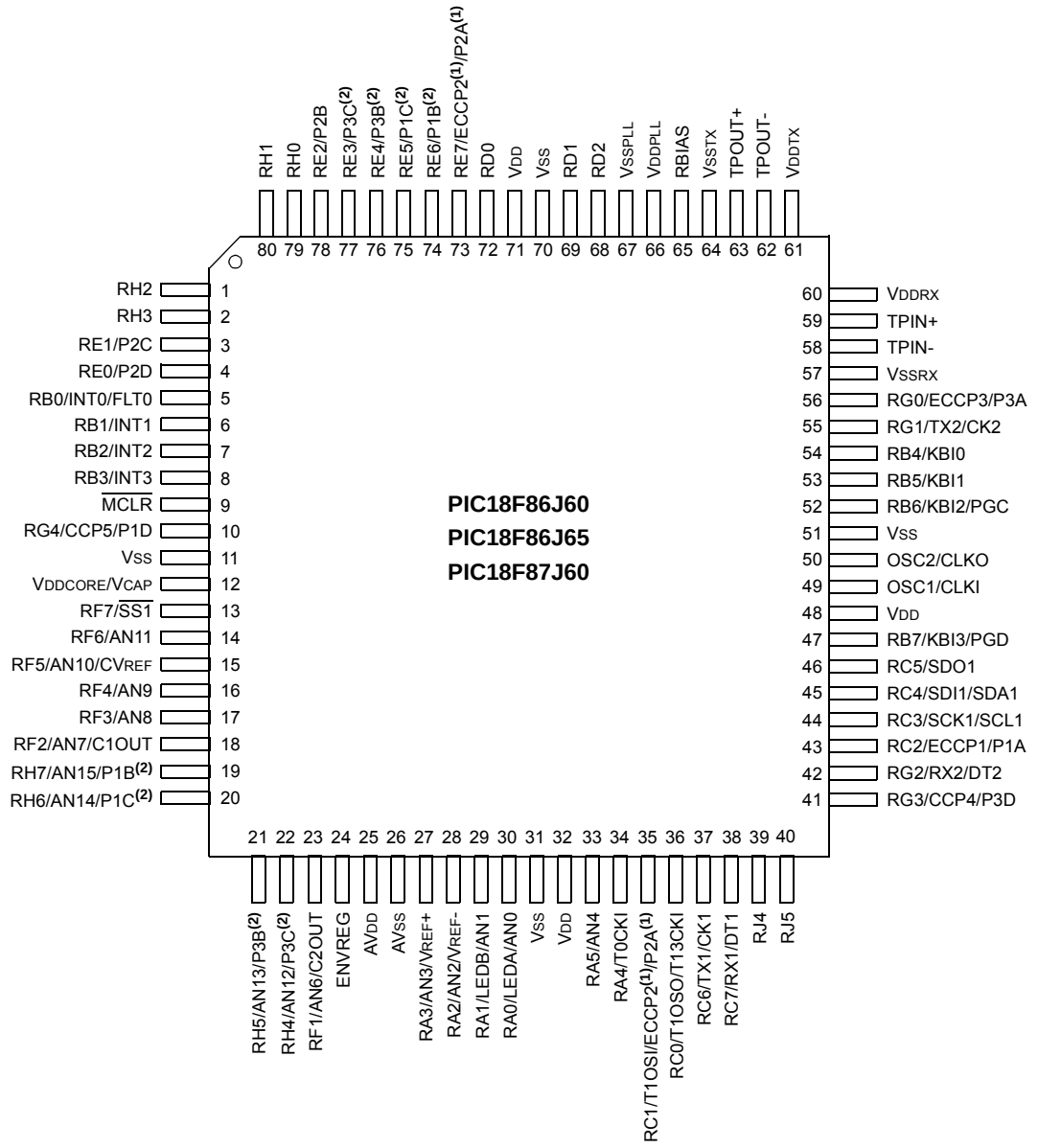
Pin Diagrams

64-Pin TQFP



Pin Diagrams (Continued)

80-Pin TQFP

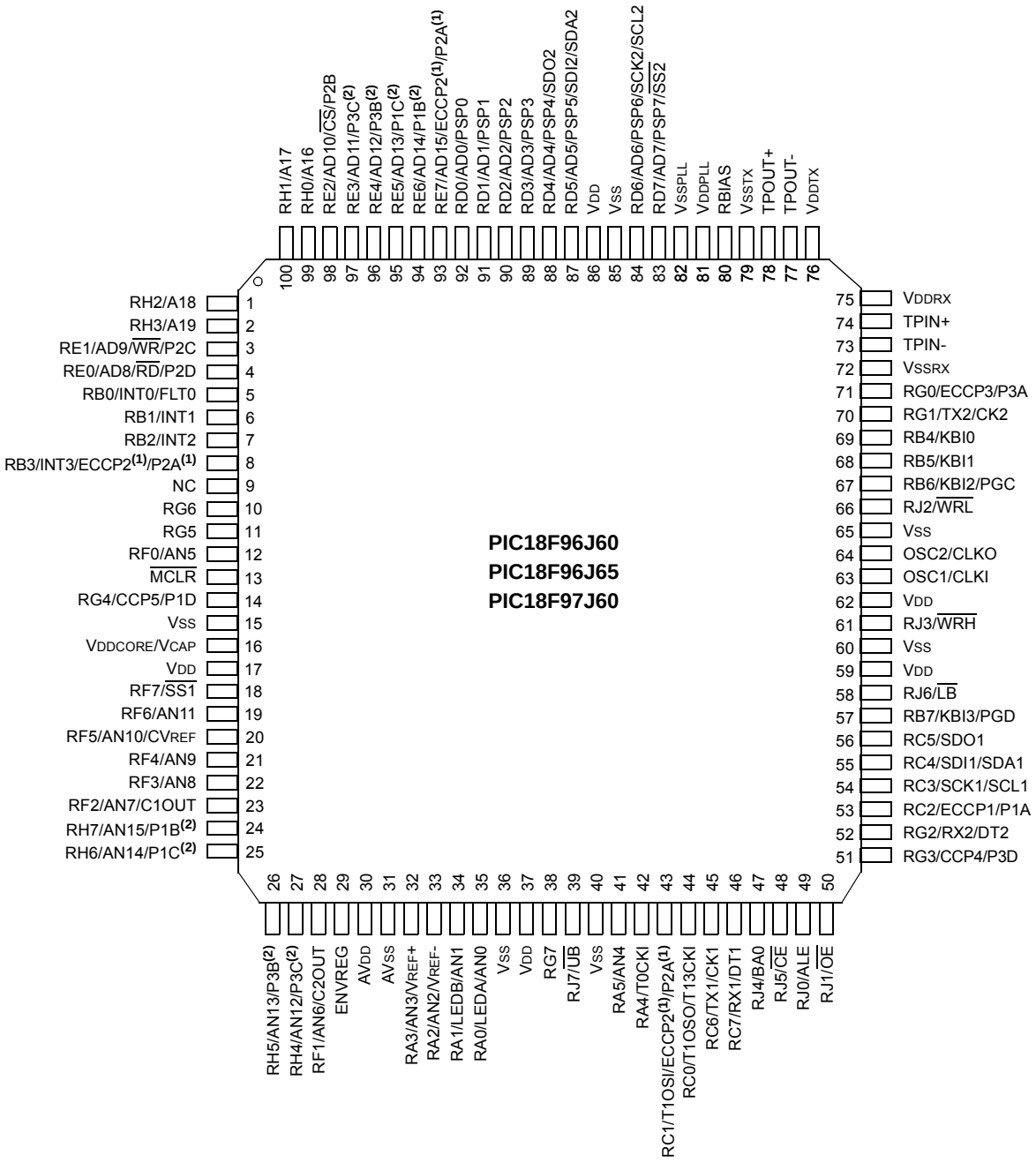


Note 1: The ECCP2/P2A pin placement depends on the CCP2MX Configuration bit setting.

2: P1B, P1C, P3B and P3C pin placement depends on the ECCPMX Configuration bit setting.

Pin Diagrams (Continued)

100-Pin TQFP



Note 1: The ECCP2/P2A pin placement depends on the CCP2MX Configuration bit and Processor mode settings.

2: P1B, P1C, P3B and P3C pin placement depends on the ECCPMX Configuration bit setting.

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:

- PIC18F66J60
- PIC18F66J65
- PIC18F67J60
- PIC18F86J60
- PIC18F86J65
- PIC18F87J60
- PIC18F96J60
- PIC18F96J65
- PIC18F97J60

This family introduces a new line of low-voltage devices with the foremost traditional advantage of all PIC18 microcontrollers – namely, high computational performance and a rich feature set at an extremely competitive price point. These features make the PIC18F97J60 family a logical choice for many high-performance applications where cost is a primary consideration.

1.1 Core Features

1.1.1 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F97J60 family offer five different oscillator options, allowing users a range of choices in developing application hardware. These options 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 41.667 MHz.
- An internal RC oscillator with a fixed 31 kHz output which provides an extremely low-power option for timing-insensitive applications.

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.

1.1.2 EXPANDED MEMORY

The PIC18F97J60 family provides ample room for application code, from 64 Kbytes to 128 Kbytes of code space. The Flash cells for program memory are rated to last 100 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 20 years.

The PIC18F97J60 family also provides plenty of room for dynamic application data with 3808 bytes of data RAM.

1.1.3 EXTERNAL MEMORY BUS

In the unlikely event that 128 Kbytes of memory are inadequate for an application, the 100-pin members of the PIC18F97J60 family also implement an External Memory Bus (EMB). 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

1.1.4 EXTENDED INSTRUCTION SET

The PIC18F97J60 family implements the optional extension to the PIC18 instruction set, adding eight new instructions and an Indexed Addressing mode. Enabled as a device configuration option, the extension has been specifically designed to optimize reentrant application code originally developed in high-level languages, such as C.

1.1.5 EASY MIGRATION

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

1.2 Other Special Features

- **Communications:** The PIC18F97J60 family incorporates a range of serial communication peripherals, including up to two independent Enhanced USARTs and up to two Master SSP modules, capable of both SPI and I^2C™ (Master and Slave) modes of operation. In addition, one of the general purpose I/O ports can be reconfigured as an 8-bit Parallel Slave Port for direct processor-to-processor communications.
- **CCP Modules:** All devices in the family incorporate two Capture/Compare/PWM (CCP) modules and three Enhanced CCP (ECCP) modules to maximize flexibility in control applications. Up to four different time bases may be used to perform several different operations at once. Each of the three ECCP modules offers up to four PWM outputs, allowing for a total of twelve PWMs. The ECCP modules also offer many beneficial features, including polarity selection, programmable dead time, auto-shutdown and restart and Half-Bridge and Full-Bridge Output modes.
- **10-Bit A/D Converter:** This module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period and thus, reducing code overhead.
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing an extended time-out range. See **Section 28.0 "Electrical Characteristics"** for time-out periods.

1.3 Details on Individual Family Members

Devices in the PIC18F97J60 family are available in 64-pin, 80-pin and 100-pin packages. Block diagrams for the three groups are shown in Figure 1-1, Figure 1-2 and Figure 1-3.

The devices are differentiated from each other in four ways:

1. Flash program memory (three sizes, ranging from 64 Kbytes for PIC18FX6J60 devices to 128 Kbytes for PIC18FX7J60 devices).
2. A/D channels (eleven for 64-pin devices, fifteen for 80-pin pin devices and sixteen for 100-pin devices).
3. Serial communication modules (one EUSART module and one MSSP module on 64-pin devices, two EUSART modules and one MSSP module on 80-pin devices and two EUSART modules and two MSSP modules on 100-pin devices).
4. I/O pins (39 on 64-pin devices, 55 on 80-pin devices and 70 on 100-pin devices).

All other features for devices in this family are identical. These are summarized in Table 1-1, Table 1-2 and Table 1-3.

The pinouts for all devices are listed in Table 1-4, Table 1-5 and Table 1-6.

TABLE 1-1: DEVICE FEATURES FOR THE PIC18F97J60 FAMILY (64-PIN DEVICES)

Features	PIC18F66J60	PIC18F66J65	PIC18F67J60
Operating Frequency	DC – 41.667 MHz	DC – 41.667 MHz	DC – 41.667 MHz
Program Memory (Bytes)	64K	96K	128K
Program Memory (Instructions)	32764	49148	65532
Data Memory (Bytes)	3808		
Interrupt Sources	26		
I/O Ports	Ports A, B, C, D, E, F, G		
I/O Pins	39		
Timers	5		
Capture/Compare/PWM Modules	2		
Enhanced Capture/Compare/PWM Modules	3		
Serial Communications	MSSP (1), Enhanced USART (1)		
Ethernet Communications (10Base-T)	Yes		
Parallel Slave Port Communications (PSP)	No		
External Memory Bus	No		
10-Bit Analog-to-Digital Module	11 Input Channels		
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, $\overline{MCLR}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	64-Pin TQFP		

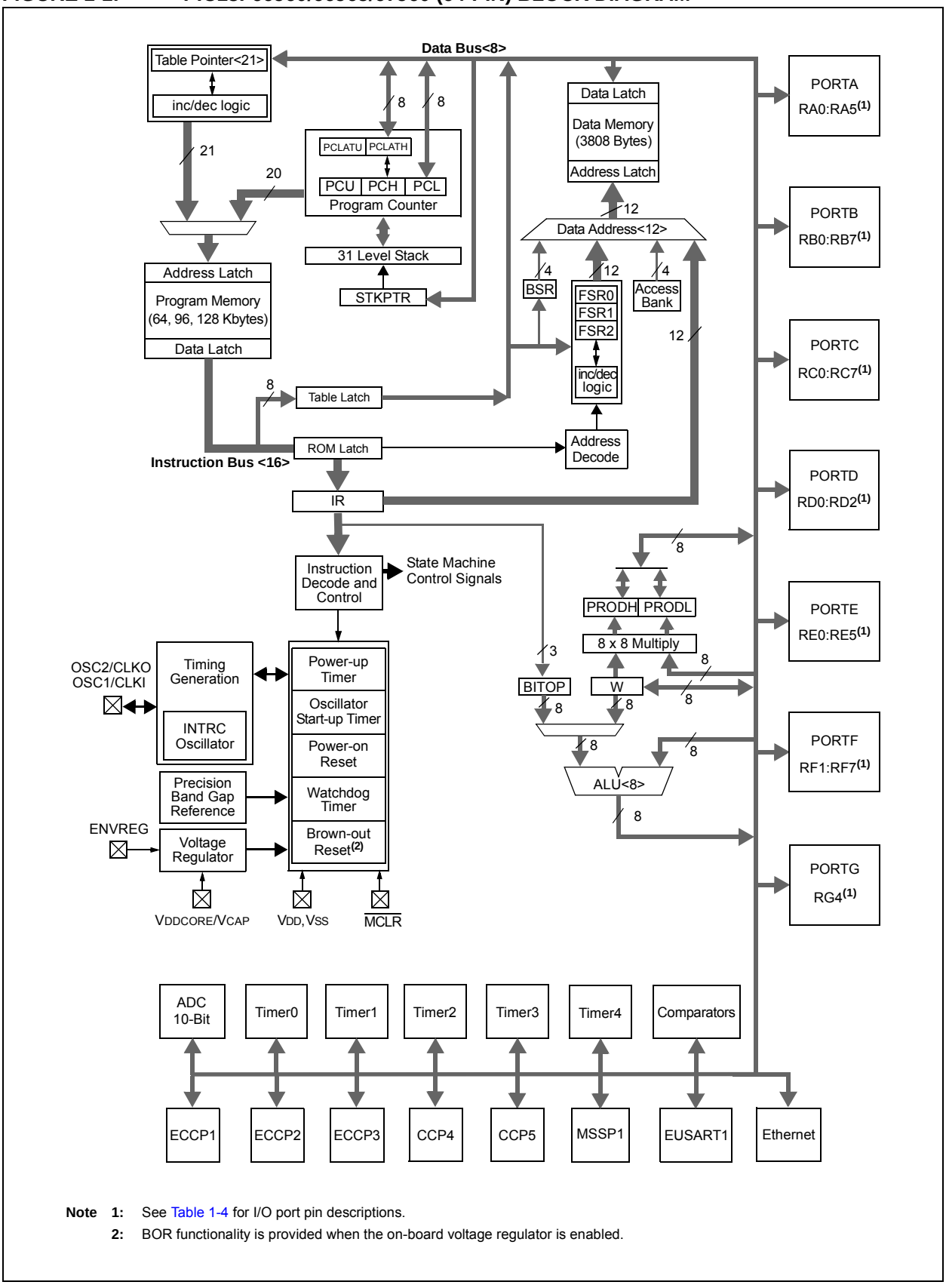
TABLE 1-2: DEVICE FEATURES FOR THE PIC18F97J60 FAMILY (80-PIN DEVICES)

Features	PIC18F86J60	PIC18F86J65	PIC18F87J60
Operating Frequency	DC – 41.667 MHz	DC – 41.667 MHz	DC – 41.667 MHz
Program Memory (Bytes)	64K	96K	128K
Program Memory (Instructions)	32764	49148	65532
Data Memory (Bytes)	3808		
Interrupt Sources	27		
I/O Ports	Ports A, B, C, D, E, F, G, H, J		
I/O Pins	55		
Timers	5		
Capture/Compare/PWM Modules	2		
Enhanced Capture/Compare/PWM Modules	3		
Serial Communications	MSSP (1), Enhanced USART (2)		
Ethernet Communications (10Base-T)	Yes		
Parallel Slave Port Communications (PSP)	No		
External Memory Bus	No		
10-Bit Analog-to-Digital Module	15 Input Channels		
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, $\overline{MCLR}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	80-Pin TQFP		

TABLE 1-3: DEVICE FEATURES FOR THE PIC18F97J60 FAMILY (100-PIN DEVICES)

Features	PIC18F96J60	PIC18F96J65	PIC18F97J60
Operating Frequency	DC – 41.667 MHz	DC – 41.667 MHz	DC – 41.667 MHz
Program Memory (Bytes)	64K	96K	128K
Program Memory (Instructions)	32764	49148	65532
Data Memory (Bytes)	3808		
Interrupt Sources	29		
I/O Ports	Ports A, B, C, D, E, F, G, H, J		
I/O Pins	70		
Timers	5		
Capture/Compare/PWM Modules	2		
Enhanced Capture/Compare/PWM Modules	3		
Serial Communications	MSSP (2), Enhanced USART (2)		
Ethernet Communications (10Base-T)	Yes		
Parallel Slave Port Communications (PSP)	Yes		
External Memory Bus	Yes		
10-Bit Analog-to-Digital Module	16 Input Channels		
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, $\overline{MCLR}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	100-Pin TQFP		

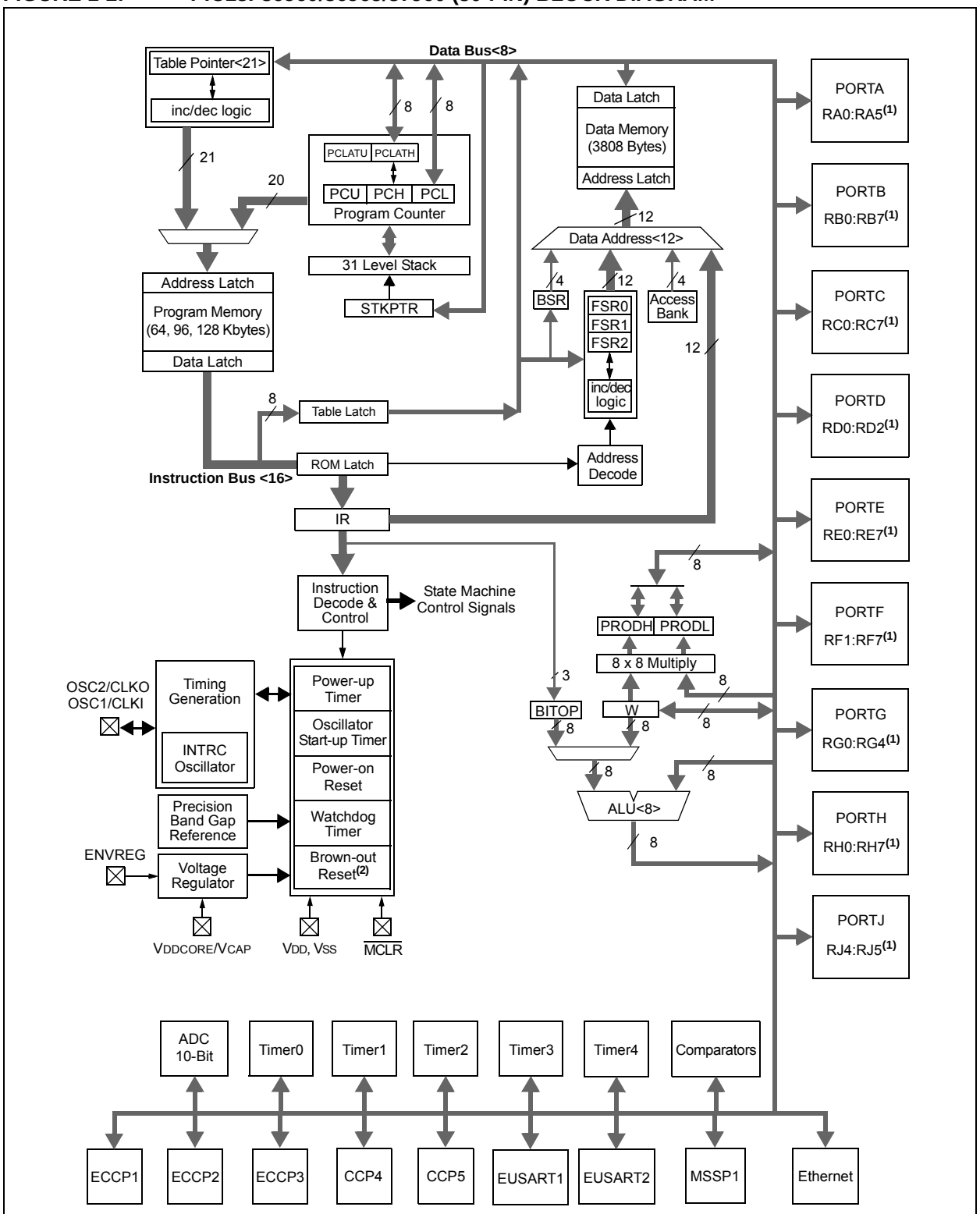
FIGURE 1-1: PIC18F66J60/66J65/67J60 (64-PIN) BLOCK DIAGRAM



Note 1: See Table 1-4 for I/O port pin descriptions.

2: BOR functionality is provided when the on-board voltage regulator is enabled.

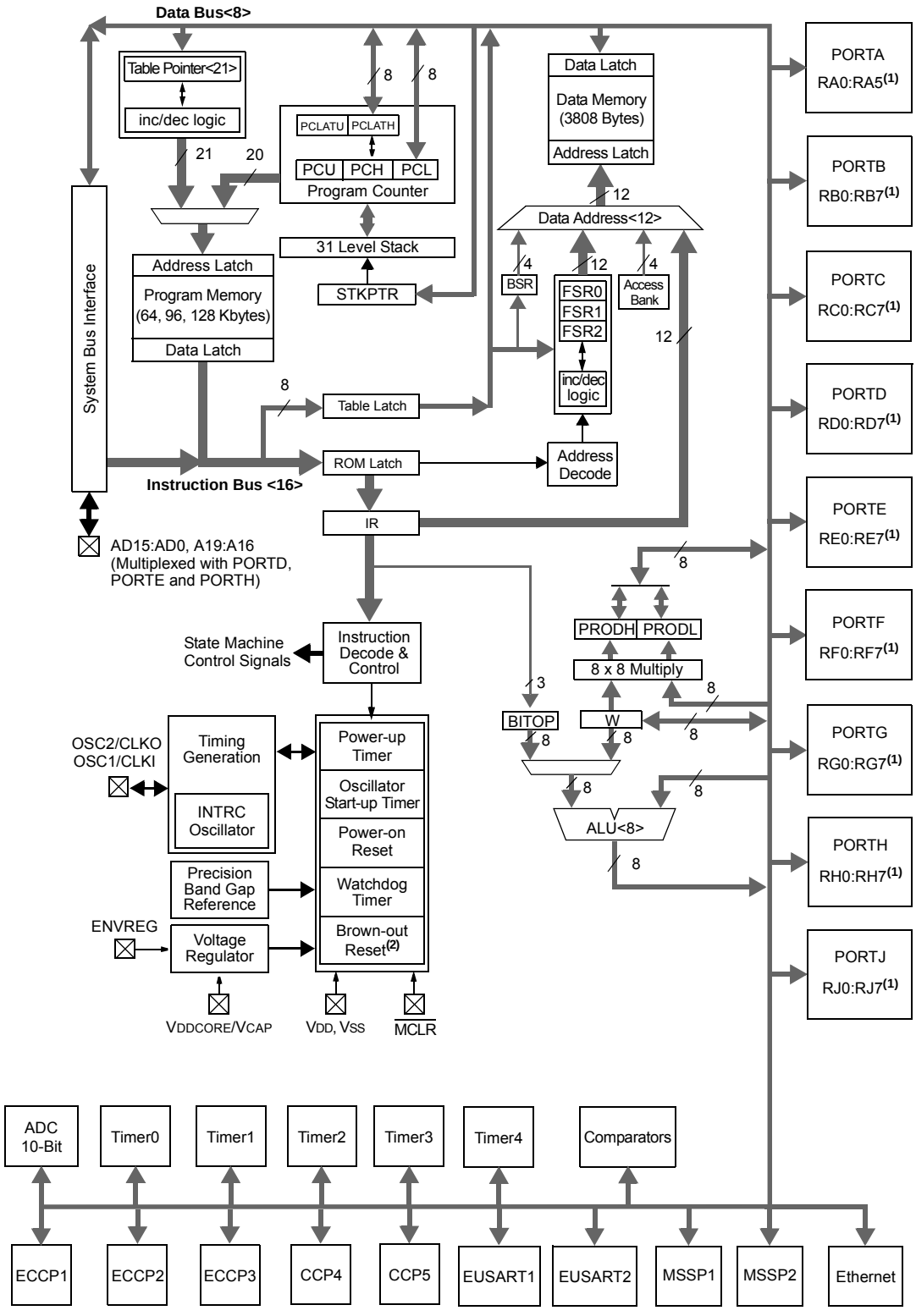
FIGURE 1-2: PIC18F86J60/86J65/87J60 (80-PIN) BLOCK DIAGRAM



Note 1: See Table 1-5 for I/O port pin descriptions.

2: BOR functionality is provided when the on-board voltage regulator is enabled.

FIGURE 1-3: PIC18F96J60/96J65/97J60 (100-PIN) BLOCK DIAGRAM



Note 1: See Table 1-6 for I/O port pin descriptions.

2: BOR functionality is provided when the on-board voltage regulator is enabled.

TABLE 1-4: PIC18F66J60/66J65/67J60 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
$\overline{MCLR}$	7	I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OSC1/CLKI	39			Oscillator crystal or external clock input.
OSC1		I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in internal RC mode; CMOS otherwise.
CLKI		I	CMOS	External clock source input. Always associated with pin function, OSC1. (See related OSC2/CLKO pin.)
OSC2/CLKO	40			Oscillator crystal or clock output.
OSC2		O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In Internal RC mode, 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/LEDA/AN0	24			
RA0		I/O	TTL	Digital I/O.
LEDA		O	—	Ethernet LEDA indicator output.
AN0		I	Analog	Analog Input 0.
RA1/LEDB/AN1	23			
RA1		I/O	TTL	Digital I/O.
LEDB		O	—	Ethernet LEDB indicator output.
AN1		I	Analog	Analog Input 1.
RA2/AN2/V_{REF-}	22			
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+}	21			
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	28			
RA4		I/O	ST	Digital I/O.
T0CKI		I	ST	Timer0 external clock input.
RA5/AN4	27			
RA5		I/O	TTL	Digital I/O.
AN4		I	Analog	Analog Input 4.

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

TABLE 1-4: PIC18F66J60/66J65/67J60 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/FLT0	3			
RB0		I/O	TTL	Digital I/O.
INT0		I	ST	External Interrupt 0.
FLT0		I	ST	Enhanced PWM Fault input (ECCP modules); enabled in software.
RB1/INT1	4			
RB1		I/O	TTL	Digital I/O.
INT1		I	ST	External Interrupt 1.
RB2/INT2	5			
RB2		I/O	TTL	Digital I/O.
INT2		I	ST	External Interrupt 2.
RB3/INT3	6			
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})

TABLE 1-4: PIC18F66J60/66J65/67J60 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/ECCP2/P2A	29			
RC1		I/O	ST	Digital I/O.
T1OSI		I	CMOS	Timer1 oscillator input.
ECCP2		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A		O	—	ECCP2 PWM Output A.
RC2/ECCP1/P1A	33			
RC2		I/O	ST	Digital I/O.
ECCP1		I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
P1A		O	—	ECCP1 PWM Output A.
RC3/SCK1/SCL1	34			
RC3		I/O	ST	Digital I/O.
SCK1		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL1		I/O	ST	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI1/SDA1	35			
RC4		I/O	ST	Digital I/O.
SDI1		I	ST	SPI data in.
SDA1		I/O	ST	I^2C data I/O.
RC5/SDO1	36			
RC5		I/O	ST	Digital I/O.
SDO1		O	—	SPI data out.
RC6/TX1/CK1	31			
RC6		I/O	ST	Digital I/O.
TX1		O	—	EUSART1 asynchronous transmit.
CK1		I/O	ST	EUSART1 synchronous clock (see related RX1/DT1 pin).
RC7/RX1/DT1	32			
RC7		I/O	ST	Digital I/O.
RX1		I	ST	EUSART1 asynchronous receive.
DT1		I/O	ST	EUSART1 synchronous data (see related TX1/CK1 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)

TABLE 1-4: PIC18F66J60/66J65/67J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTD is a bidirectional I/O port.
RD0/P1B	60			
RD0		I/O	ST	Digital I/O.
P1B		O	—	ECCP1 PWM Output B.
RD1/ECCP3/P3A	59			
RD1		I/O	ST	Digital I/O.
ECCP3		I/O	ST	Capture 3 input/Compare 3 output/PWM3 output.
P3A		O	—	ECCP3 PWM Output A.
RD2/CCP4/P3D	58			
RD2		I/O	ST	Digital I/O.
CCP4		I/O	ST	Capture 4 input/Compare 4 output/PWM4 output.
P3D		O	—	CCP4 PWM Output D.

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})

TABLE 1-4: PIC18F66J60/66J65/67J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTE is a bidirectional I/O port.
RE0/P2D	2			
RE0		I/O	ST	Digital I/O.
P2D		O	—	ECCP2 PWM Output D.
RE1/P2C	1			
RE1		I/O	ST	Digital I/O.
P2C		O	—	ECCP2 PWM Output C.
RE2/P2B	64			
RE2		I/O	ST	Digital I/O.
P2B		O	—	ECCP2 PWM Output B.
RE3/P3C	63			
RE3		I/O	ST	Digital I/O.
P3C		O	—	ECCP3 PWM Output C.
RE4/P3B	62			
RE4		I/O	ST	Digital I/O.
P3B		O	—	ECCP3 PWM Output B.
RE5/P1C	61			
RE5		I/O	ST	Digital I/O.
P1C		O	—	ECCP1 PWM Output C.

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})

TABLE 1-4: PIC18F66J60/66J65/67J60 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/CVREF	13			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog Input 10.
CVREF		O	—	Comparator reference voltage output.
RF6/AN11	12			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog Input 11.
RF7/$\overline{SS1}$	11			
RF7		I/O	ST	Digital I/O.
$\overline{SS1}$		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 VDD)

TABLE 1-4: PIC18F66J60/66J65/67J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
RG4/CCP5/P1D	8			PORTG is a bidirectional I/O port.
RG4		I/O	ST	Digital I/O.
CCP5		I/O	ST	Capture 5 input/Compare 5 output/PWM5 output.
P1D		O	—	ECCP1 PWM Output D.
VSS	9, 25, 41, 56	P	—	Ground reference for logic and I/O pins.
VDD	26, 38, 57	P	—	Positive supply for peripheral digital 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).
VSSPLL	55	P	—	Ground reference for Ethernet PHY PLL.
VDDPLL	54	P	—	Positive 3.3V supply for Ethernet PHY PLL.
VSSTX	52	P	—	Ground reference for Ethernet PHY transmit subsystem.
VDDTX	49	P	—	Positive 3.3V supply for Ethernet PHY transmit subsystem.
VSSRX	45	P	—	Ground reference for Ethernet PHY receive subsystem.
VDDRX	48	P	—	Positive 3.3V supply for Ethernet PHY receive subsystem.
RBIAS	53	I	Analog	Bias current for Ethernet PHY. Must be tied to VSS via a resistor; see **Section 19.0 "Ethernet Module"** for specification.
TPOUT+	51	O	—	Ethernet differential signal output.
TPOUT-	50	O	—	Ethernet differential signal output.
TPIN+	47	I	Analog	Ethernet differential signal input.
TPIN-	46	I	Analog	Ethernet differential signal input.

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

TABLE 1-5: PIC18F86J60/86J65/87J60 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
$\overline{MCLR}$	9	I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OSC1/CLKI	49			Oscillator crystal or external clock input.
OSC1		I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in internal RC mode; CMOS otherwise.
CLKI		I	CMOS	External clock source input. Always associated with pin function, OSC1. (See related OSC2/CLKO pin.)
OSC2/CLKO	50			Oscillator crystal or clock output.
OSC2		O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In Internal RC mode, 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/LEDA/AN0	30			
RA0		I/O	TTL	Digital I/O.
LEDA		O	—	Ethernet LEDA indicator output.
AN0		I	Analog	Analog Input 0.
RA1/LEDB/AN1	29			
RA1		I/O	TTL	Digital I/O.
LEDB		O	—	Ethernet LEDB indicator output.
AN1		I	Analog	Analog Input 1.
RA2/AN2/VREF-	28			
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+	27			
RA3		I/O	TTL	Digital I/O.
AN3		I	Analog	Analog Input 3.
VREF+		I	Analog	A/D reference voltage (high) input.
RA4/T0CKI	34			
RA4		I/O	ST	Digital I/O.
T0CKI		I	ST	Timer0 external clock input.
RA5/AN4	33			
RA5		I/O	TTL	Digital I/O.
AN4		I	Analog	Analog Input 4.

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

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 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/FLT0	5			
RB0		I/O	TTL	Digital I/O.
INT0		I	ST	External Interrupt 0.
FLT0		I	ST	Enhanced PWM Fault input (ECCP modules); enabled in software.
RB1/INT1	6			
RB1		I/O	TTL	Digital I/O.
INT1		I	ST	External Interrupt 1.
RB2/INT2	7			
RB2		I/O	TTL	Digital I/O.
INT2		I	ST	External Interrupt 2.
RB3/INT3	8			
RB3		I/O	TTL	Digital I/O.
INT3		I	ST	External Interrupt 3.
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})

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 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/ECCP2/P2A	35			
RC1		I/O	ST	Digital I/O.
T1OSI		I	CMOS	Timer1 oscillator input.
ECCP2(1)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A(1)		O	—	ECCP2 PWM Output A.
RC2/ECCP1/P1A	43			
RC2		I/O	ST	Digital I/O.
ECCP1		I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
P1A		O	—	ECCP1 PWM Output A.
RC3/SCK1/SCL1	44			
RC3		I/O	ST	Digital I/O.
SCK1		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL1		I/O	ST	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI1/SDA1	45			
RC4		I/O	ST	Digital I/O.
SDI1		I	ST	SPI data in.
SDA1		I/O	ST	I^2C data I/O.
RC5/SDO1	46			
RC5		I/O	ST	Digital I/O.
SDO1		O	—	SPI data out.
RC6/TX1/CK1	37			
RC6		I/O	ST	Digital I/O.
TX1		O	—	EUSART1 asynchronous transmit.
CK1		I/O	ST	EUSART1 synchronous clock (see related RX1/DT1 pin).
RC7/RX1/DT1	38			
RC7		I/O	ST	Digital I/O.
RX1		I	ST	EUSART1 asynchronous receive.
DT1		I/O	ST	EUSART1 synchronous data (see related TX1/CK1 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})

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTD is a bidirectional I/O port.
RD0	72	I/O	ST	Digital I/O.
RD1	69	I/O	ST	Digital I/O.
RD2	68	I/O	ST	Digital I/O.
				PORTE is a bidirectional I/O port.
RE0/P2D	4			
RE0		I/O	ST	Digital I/O.
P2D		O	—	ECCP2 PWM Output D.
RE1/P2C	3			
RE1		I/O	ST	Digital I/O.
P2C		O	—	ECCP2 PWM Output C.
RE2/P2B	78			
RE2		I/O	ST	Digital I/O.
P2B		O	—	ECCP2 PWM Output B.
RE3/P3C	77			
RE3		I/O	ST	Digital I/O.
P3C(2)		O	—	ECCP3 PWM Output C.
RE4/P3B	76			
RE4		I/O	ST	Digital I/O.
P3B(2)		O	—	ECCP3 PWM Output B.
RE5/P1C	75			
RE5		I/O	ST	Digital I/O.
P1C(2)		O	—	ECCP1 PWM Output C.
RE6/P1B	74			
RE6		I/O	ST	Digital I/O.
P1B(2)		O	—	ECCP1 PWM Output B.
RE7/ECCP2/P2A	73			
RE7		I/O	ST	Digital I/O.
ECCP2(3)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A(3)		O	—	ECCP2 PWM Output A.

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

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 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/CVREF	15			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog Input 10.
CVREF		O	—	Comparator reference voltage output.
RF6/AN11	14			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog Input 11.
RF7/$\overline{SS1}$	13			
RF7		I/O	ST	Digital I/O.
$\overline{SS1}$		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 VDD)

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTG is a bidirectional I/O port.
RG0/ECCP3/P3A	56			
RG0		I/O	ST	Digital I/O.
ECCP3		I/O	ST	Capture 3 input/Compare 3 output/PWM3 output.
P3A		O	—	ECCP3 PWM Output A.
RG1/TX2/CK2	55			
RG1		I/O	ST	Digital I/O.
TX2		O	—	EUSART2 asynchronous transmit.
CK2		I/O	ST	EUSART2 synchronous clock (see related RX2/DT2 pin).
RG2/RX2/DT2	42			
RG2		I/O	ST	Digital I/O.
RX2		I	ST	EUSART2 asynchronous receive.
DT2		I/O	ST	EUSART2 synchronous data (see related TX2/CK2 pin).
RG3/CCP4/P3D	41			
RG3		I/O	ST	Digital I/O.
CCP4		I/O	ST	Capture 4 input/Compare 4 output/PWM4 output.
P3D		O	—	ECCP3 PWM Output D.
RG4/CCP5/P1D	10			
RG4		I/O	ST	Digital I/O.
CCP5		I/O	ST	Capture 5 input/Compare 5 output/PWM5 output.
P1D		O	—	ECCP1 PWM Output D.

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})

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTH is a bidirectional I/O port.
RH0	79	I/O	ST	Digital I/O.
RH1	80	I/O	ST	Digital I/O.
RH2	1	I/O	ST	Digital I/O.
RH3	2	I/O	ST	Digital I/O.
RH4/AN12/P3C	22			
RH4		I/O	ST	Digital I/O.
AN12		I	Analog	Analog Input 12.
P3C(4)		O	—	ECCP3 PWM Output C.
RH5/AN13/P3B	21			
RH5		I/O	ST	Digital I/O.
AN13		I	Analog	Analog Input 13.
P3B(4)		O	—	ECCP3 PWM Output B.
RH6/AN14/P1C	20			
RH6		I/O	ST	Digital I/O.
AN14		I	Analog	Analog Input 14.
P1C(4)		O	—	ECCP1 PWM Output C.
RH7/AN15/P1B	19			
RH7		I/O	ST	Digital I/O.
AN15		I	Analog	Analog Input 15.
P1B(4)		O	—	ECCP1 PWM Output B.

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})

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-5: PIC18F86J60/86J65/87J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTJ is a bidirectional I/O port.
RJ4	39	I/O	ST	Digital I/O.
RJ5	40	I/O	ST	Digital I/O
VSS	11, 31, 51, 70	P	—	Ground reference for logic and I/O pins.
VDD	32, 48, 71	P	—	Positive supply for peripheral digital 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).
VSSPLL	67	P	—	Ground reference for Ethernet PHY PLL.
VDDPLL	66	P	—	Positive 3.3V supply for Ethernet PHY PLL.
VSSTX	64	P	—	Ground reference for Ethernet PHY transmit subsystem.
VDDTX	61	P	—	Positive 3.3V supply for Ethernet PHY transmit subsystem.
VSSRX	57	P	—	Ground reference for Ethernet PHY receive subsystem.
VDDRX	60	P	—	Positive 3.3V supply for Ethernet PHY receive subsystem.
RBIAS	65	I	Analog	Bias current for Ethernet PHY. Must be tied to VSS via a resistor; see **Section 19.0 "Ethernet Module"** for specification.
TPOUT+	63	O	—	Ethernet differential signal output.
TPOUT-	62	O	—	Ethernet differential signal output.
TPIN+	59	I	Analog	Ethernet differential signal input.
TPIN-	58	I	Analog	Ethernet differential signal input.

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

Note 1: Default assignment for ECCP2/P2A when CCP2MX Configuration bit is set.

2: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

3: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared.

4: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
$\overline{MCLR}$	13	I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OSC1/CLKI	63			Oscillator crystal or external clock input.
OSC1		I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in internal RC mode; CMOS otherwise.
CLKI		I	CMOS	External clock source input. Always associated with pin function, OSC1. (See related OSC2/CLKO pin.)
OSC2/CLKO	64			Oscillator crystal or clock output.
OSC2		O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO		O	—	In Internal RC mode, 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/LEDA/AN0	35			
RA0		I/O	TTL	Digital I/O.
LEDA		O	—	Ethernet LEDA indicator output.
AN0		I	Analog	Analog Input 0.
RA1/LEDB/AN1	34			
RA1		I/O	TTL	Digital I/O.
LEDB		O	—	Ethernet LEDB indicator output.
AN1		I	Analog	Analog Input 1.
RA2/AN2/V_{REF-}	33			
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+}	32			
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	42			
RA4		I/O	ST	Digital I/O.
T0CKI		I	ST	Timer0 external clock input.
RA5/AN4	41			
RA5		I/O	TTL	Digital I/O.
AN4		I	Analog	Analog Input 4.

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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 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/FLT0	5			
RB0		I/O	TTL	Digital I/O.
INT0		I	ST	External Interrupt 0.
FLT0		I	ST	Enhanced PWM Fault input (ECCP modules); enabled in software.
RB1/INT1	6			
RB1		I/O	TTL	Digital I/O.
INT1		I	ST	External Interrupt 1.
RB2/INT2	7			
RB2		I/O	TTL	Digital I/O.
INT2		I	ST	External Interrupt 2.
RB3/INT3/ECCP2/P2A	8			
RB3		I/O	TTL	Digital I/O.
INT3		I	ST	External Interrupt 3.
ECCP2(1)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A(1)		O	—	ECCP2 PWM Output A.
RB4/KBI0	69			
RB4		I/O	TTL	Digital I/O.
KBI0		I	TTL	Interrupt-on-change pin.
RB5/KBI1	68			
RB5		I/O	TTL	Digital I/O.
KBI1		I	TTL	Interrupt-on-change pin.
RB6/KBI2/PGC	67			
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	57			
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 VDD)

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 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	44			
RC0		I/O	ST	Digital I/O.
T1OSO		O	—	Timer1 oscillator output.
T13CKI		I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI/ECCP2/P2A	43			
RC1		I/O	ST	Digital I/O.
T1OSI		I	CMOS	Timer1 oscillator input.
ECCP2(2)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A(2)		O	—	ECCP2 PWM Output A.
RC2/ECCP1/P1A	53			
RC2		I/O	ST	Digital I/O.
ECCP1		I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
P1A		O	—	ECCP1 PWM Output A.
RC3/SCK1/SCL1	54			
RC3		I/O	ST	Digital I/O.
SCK1		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL1		I/O	ST	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI1/SDA1	55			
RC4		I/O	ST	Digital I/O.
SDI1		I	ST	SPI data in.
SDA1		I/O	ST	I^2C data I/O.
RC5/SDO1	56			
RC5		I/O	ST	Digital I/O.
SDO1		O	—	SPI data out.
RC6/TX1/CK1	45			
RC6		I/O	ST	Digital I/O.
TX1		O	—	EUSART1 asynchronous transmit.
CK1		I/O	ST	EUSART1 synchronous clock (see related RX1/DT1 pin).
RC7/RX1/DT1	46			
RC7		I/O	ST	Digital I/O.
RX1		I	ST	EUSART1 asynchronous receive.
DT1		I/O	ST	EUSART1 synchronous data (see related TX1/CK1 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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 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	92			
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	91			
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	90			
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	89			
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/SDO2	88			
RD4		I/O	ST	Digital I/O.
AD4		I/O	TTL	External Memory Address/Data 4.
PSP4		I/O	TTL	Parallel Slave Port data.
SDO2		O	—	SPI data out.
RD5/AD5/PSP5/ SDI2/SDA2	87			
RD5		I/O	ST	Digital I/O.
AD5		I/O	TTL	External Memory Address/Data 5.
PSP5		I/O	TTL	Parallel Slave Port data.
SDI2		I	ST	SPI data in.
SDA2		I/O	ST	I^2C™ data I/O.
RD6/AD6/PSP6/ SCK2/SCL2	84			
RD6		I/O	ST	Digital I/O.
AD6		I/O	TTL	External Memory Address/Data 6.
PSP6		I/O	TTL	Parallel Slave Port data.
SCK2		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL2		I/O	ST	Synchronous serial clock input/output for I^2C™ mode.
RD7/AD7/PSP7/$\overline{SS2}$	83			
RD7		I/O	ST	Digital I/O.
AD7		I/O	TTL	External Memory Address/Data 7.
PSP7		I/O	TTL	Parallel Slave Port data.
$\overline{SS2}$		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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTE is a bidirectional I/O port.
RE0/AD8/$\overline{RD}$/P2D	4			
RE0		I/O	ST	Digital I/O.
AD8		I/O	TTL	External Memory Address/Data 8.
$\overline{RD}$		I	TTL	Read control for Parallel Slave Port.
P2D		O	—	ECCP2 PWM Output D.
RE1/AD9/$\overline{WR}$/P2C	3			
RE1		I/O	ST	Digital I/O.
AD9		I/O	TTL	External Memory Address/Data 9.
$\overline{WR}$		I	TTL	Write control for Parallel Slave Port.
P2C		O	—	ECCP2 PWM Output C.
RE2/AD10/$\overline{CS}$/P2B	98			
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.
P2B		O	—	ECCP2 PWM Output B.
RE3/AD11/P3C	97			
RE3		I/O	ST	Digital I/O.
AD11		I/O	TTL	External Memory Address/Data 11.
P3C(3)		O	—	ECCP3 PWM Output C.
RE4/AD12/P3B	96			
RE4		I/O	ST	Digital I/O.
AD12		I/O	TTL	External Memory Address/Data 12.
P3B(3)		O	—	ECCP3 PWM Output B.
RE5/AD13/P1C	95			
RE5		I/O	ST	Digital I/O.
AD13		I/O	TTL	External Memory Address/Data 13.
P1C(3)		O	—	ECCP1 PWM Output C.
RE6/AD14/P1B	94			
RE6		I/O	ST	Digital I/O.
AD14		I/O	TTL	External Memory Address/Data 14.
P1B(3)		O	—	ECCP1 PWM Output B.
RE7/AD15/ECCP2/P2A	93			
RE7		I/O	ST	Digital I/O.
AD15		I/O	TTL	External Memory Address/Data 15.
ECCP2(4)		I/O	ST	Capture 2 input/Compare 2 output/PWM2 output.
P2A(4)		O	—	ECCP2 PWM Output A.

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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTF is a bidirectional I/O port.
RF0/AN5	12			
RF0		I/O	ST	Digital I/O.
AN5		I	Analog	Analog Input 5.
RF1/AN6/C2OUT	28			
RF1		I/O	ST	Digital I/O.
AN6		I	Analog	Analog Input 6.
C2OUT		O	—	Comparator 2 output.
RF2/AN7/C1OUT	23			
RF2		I/O	ST	Digital I/O.
AN7		I	Analog	Analog Input 7.
C1OUT		O	—	Comparator 1 output.
RF3/AN8	22			
RF3		I/O	ST	Digital I/O.
AN8		I	Analog	Analog Input 8.
RF4/AN9	21			
RF4		I/O	ST	Digital I/O.
AN9		I	Analog	Analog Input 9.
RF5/AN10/CVREF	20			
RF5		I/O	ST	Digital I/O.
AN10		I	Analog	Analog Input 10.
CVREF		O	—	Comparator reference voltage output.
RF6/AN11	19			
RF6		I/O	ST	Digital I/O.
AN11		I	Analog	Analog Input 11.
RF7/$\overline{SS1}$	18			
RF7		I/O	ST	Digital I/O.
$\overline{SS1}$		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 VDD)

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTG is a bidirectional I/O port.
RG0/ECCP3/P3A	71			
RG0		I/O	ST	Digital I/O.
ECCP3		I/O	ST	Capture 3 input/Compare 3 output/PWM3 output.
P3A		O	—	ECCP3 PWM Output A.
RG1/TX2/CK2	70			
RG1		I/O	ST	Digital I/O.
TX2		O	—	EUSART2 asynchronous transmit.
CK2		I/O	ST	EUSART2 synchronous clock (see related RX2/DT2 pin).
RG2/RX2/DT2	52			
RG2		I/O	ST	Digital I/O.
RX2		I	ST	EUSART2 asynchronous receive.
DT2		I/O	ST	EUSART2 synchronous data (see related TX2/CK2 pin).
RG3/CCP4/P3D	51			
RG3		I/O	ST	Digital I/O.
CCP4		I/O	ST	Capture 4 input/Compare 4 output/PWM4 output.
P3D		O	—	ECCP3 PWM Output D.
RG4/CCP5/P1D	14			
RG4		I/O	ST	Digital I/O.
CCP5		I/O	ST	Capture 5 input/Compare 5 output/PWM5 output.
P1D		O	—	ECCP1 PWM Output D.
RG5	11	I/O	ST	Digital I/O.
RG6	10	I/O	ST	Digital I/O.
RG7	38	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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTH is a bidirectional I/O port.
RH0/A16	99			
RH0		I/O	ST	Digital I/O.
A16		O	—	External Memory Address 16.
RH1/A17	100			
RH1		I/O	ST	Digital I/O.
A17		O	—	External Memory Address 17.
RH2/A18	1			
RH2		I/O	ST	Digital I/O.
A18		O	—	External Memory Address 18.
RH3/A19	2			
RH3		I/O	ST	Digital I/O.
A19		O	—	External Memory Address 19.
RH4/AN12/P3C	27			
RH4		I/O	ST	Digital I/O.
AN12		I	Analog	Analog Input 12.
P3C(5)		O	—	ECCP3 PWM Output C.
RH5/AN13/P3B	26			
RH5		I/O	ST	Digital I/O.
AN13		I	Analog	Analog Input 13.
P3B(5)		O	—	ECCP3 PWM Output B.
RH6/AN14/P1C	25			
RH6		I/O	ST	Digital I/O.
AN14		I	Analog	Analog Input 14.
P1C(5)		O	—	ECCP1 PWM Output C.
RH7/AN15/P1B	24			
RH7		I/O	ST	Digital I/O.
AN15		I	Analog	Analog Input 15.
P1B(5)		O	—	ECCP1 PWM Output B.

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

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTJ is a bidirectional I/O port.
RJ0/ALE	49			
RJ0		I/O	ST	Digital I/O.
ALE		O	—	External memory address latch enable.
RJ1/$\overline{OE}$	50			
RJ1		I/O	ST	Digital I/O.
$\overline{OE}$		O	—	External memory output enable.
RJ2/$\overline{WRL}$	66			
RJ2		I/O	ST	Digital I/O.
$\overline{WRL}$		O	—	External memory write low control.
RJ3/$\overline{WRH}$	61			
RJ3		I/O	ST	Digital I/O.
$\overline{WRH}$		O	—	External memory write high control.
RJ4/BA0	47			
RJ4		I/O	ST	Digital I/O.
BA0		O	—	External Memory Byte Address 0 control.
RJ5/$\overline{CE}$	48			
RJ5		I/O	ST	Digital I/O
$\overline{CE}$		O	—	External memory chip enable control.
RJ6/$\overline{LB}$	58			
RJ6		I/O	ST	Digital I/O.
$\overline{LB}$		O	—	External memory low byte control.
RJ7/$\overline{UB}$	39			
RJ7		I/O	ST	Digital I/O.
$\overline{UB}$		O	—	External memory high byte control.

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})

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

TABLE 1-6: PIC18F96J60/96J65/97J60 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
NC	9	—	—	No connect.
VSS	15, 36, 40, 60, 65, 85	P	—	Ground reference for logic and I/O pins.
VDD	17, 37, 59, 62, 86	P	—	Positive supply for peripheral digital logic and I/O pins.
AVSS	31	P	—	Ground reference for analog modules.
AVDD	30	P	—	Positive supply for analog modules.
ENVREG	29	I	ST	Enable for on-chip voltage regulator.
VDDCORE/VCAP	16			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).
VSSPLL	82	P	—	Ground reference for Ethernet PHY PLL.
VDDPLL	81	P	—	Positive 3.3V supply for Ethernet PHY PLL.
VSSTX	79	P	—	Ground reference for Ethernet PHY transmit subsystem.
VDDTX	76	P	—	Positive 3.3V supply for Ethernet PHY transmit subsystem.
VSSRX	72	P	—	Ground reference for Ethernet PHY receive subsystem.
VDDRX	75	P	—	Positive 3.3V supply for Ethernet PHY receive subsystem.
RBIAS	80	I	Analog	Bias current for Ethernet PHY. Must be tied to VSS via a resistor; see **Section 19.0 "Ethernet Module"** for specification.
TPOUT+	78	O	—	Ethernet differential signal output.
TPOUT-	77	O	—	Ethernet differential signal output.
TPIN+	74	I	Analog	Ethernet differential signal input.
TPIN-	73	I	Analog	Ethernet differential signal 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 VDD)

Note 1: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Extended Microcontroller mode).

2: Default assignment for ECCP2/P2A for all devices in all operating modes (CCP2MX Configuration bit is set).

3: Default assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is set).

4: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (Microcontroller mode).

5: Alternate assignments for P1B/P1C/P3B/P3C (ECCPMX Configuration bit is cleared).

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18FJ MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18F97J60 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"**)

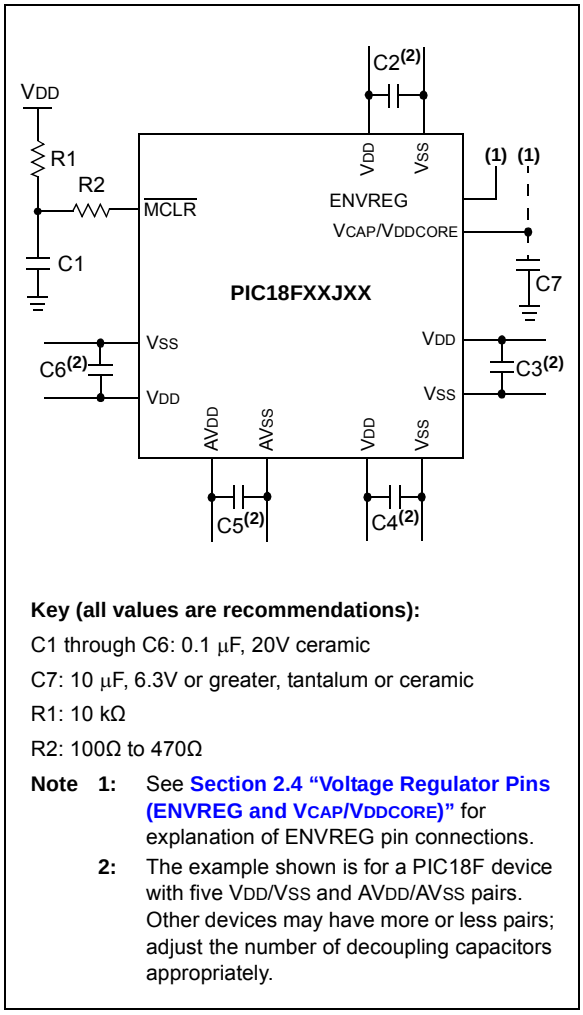
Additionally, the following pins may be required:

- 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.

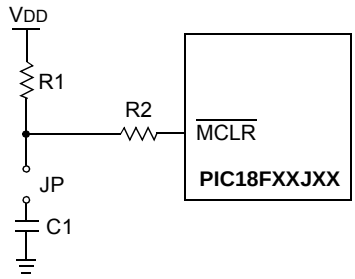
2.3 Master Clear ($\overline{\text{MCLR}}$) Pin

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 25.3 "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. Suitable examples of capacitors are shown in Table 2-1. Capacitors with equivalent specifications can be used.

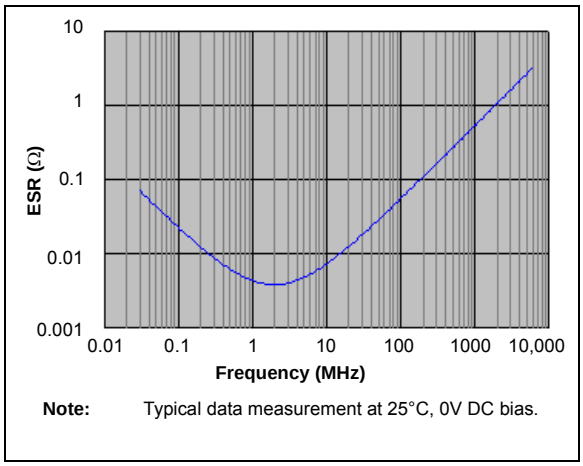
Designers may use Figure 2-3 to evaluate ESR equivalence of candidate devices.

It is recommended that the trace length not exceed 0.25 inch (6 mm). Refer to **28.0 "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 **28.0 "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: Typical data measurement at 25°C, 0V DC bias.

TABLE 2-1: SUITABLE CAPACITOR EQUIVALENTS

Make	Part #	Nominal Capacitance	Base Tolerance	Rated Voltage	Temp. Range
TDK	C3216X7R1C106K	10 µF	±10%	16V	-55 to 125ºC
TDK	C3216X5R1C106K	10 µF	±10%	16V	-55 to 85ºC
Panasonic	ECJ-3YX1C106K	10 µF	±10%	16V	-55 to 125ºC
Panasonic	ECJ-4YB1C106K	10 µF	±10%	16V	-55 to 85ºC
Murata	GRM32DR71C106KA01L	10 µF	±10%	16V	-55 to 125ºC
Murata	GRM31CR61C106KC31L	10 µF	±10%	16V	-55 to 85ºC

2.4.1 CONSIDERATIONS FOR CERAMIC CAPACITORS

In recent years, large value, low-voltage, surface-mount ceramic capacitors have become very cost effective in sizes up to a few tens of microfarad. The low-ESR, small physical size and other properties make ceramic capacitors very attractive in many types of applications.

Ceramic capacitors are suitable for use with the VDDCORE voltage regulator of this microcontroller. However, some care is needed in selecting the capacitor to ensure that it maintains sufficient capacitance over the intended operating range of the application.

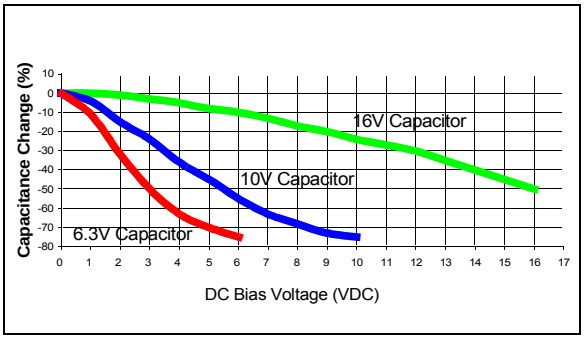
Typical low-cost, 10 µF ceramic capacitors are available in X5R, X7R and Y5V dielectric ratings (other types are also available, but are less common). The initial tolerance specifications for these types of capacitors are often specified as ±10% to ±20% (X5R and X7R), or -20%/+80% (Y5V). However, the effective capacitance that these capacitors provide in an application circuit will also vary based on additional factors, such as the applied DC bias voltage and the temperature. The total in-circuit tolerance is, therefore, much wider than the initial tolerance specification.

The X5R and X7R capacitors typically exhibit satisfactory temperature stability (ex: ±15% over a wide temperature range, but consult the manufacturer's data sheets for exact specifications). However, Y5V capacitors typically have extreme temperature tolerance specifications of +22%/-82%. Due to the extreme temperature tolerance, a 10 µF nominal rated Y5V type capacitor may not deliver enough total capacitance to meet minimum VDDCORE voltage regulator stability and transient response requirements. Therefore, Y5V capacitors are not recommended for use with the VDDCORE regulator if the application must operate over a wide temperature range.

In addition to temperature tolerance, the effective capacitance of large value ceramic capacitors can vary substantially, based on the amount of DC voltage applied to the capacitor. This effect can be very significant, but is often overlooked or is not always documented.

A typical DC bias voltage vs. capacitance graph for X7R type and Y5V type capacitors is shown in Figure 2-4.

FIGURE 2-4: DC BIAS VOLTAGE vs. CAPACITANCE CHARACTERISTICS



When selecting a ceramic capacitor to be used with the VDDCORE voltage regulator, it is suggested to select a high-voltage rating, so that the operating voltage is a small percentage of the maximum rated capacitor voltage. For example, choose a ceramic capacitor rated at 16V for the 2.5V VDDCORE voltage. Suggested capacitors are shown in Table 2-1.

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 27.0 “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 2-5. 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).

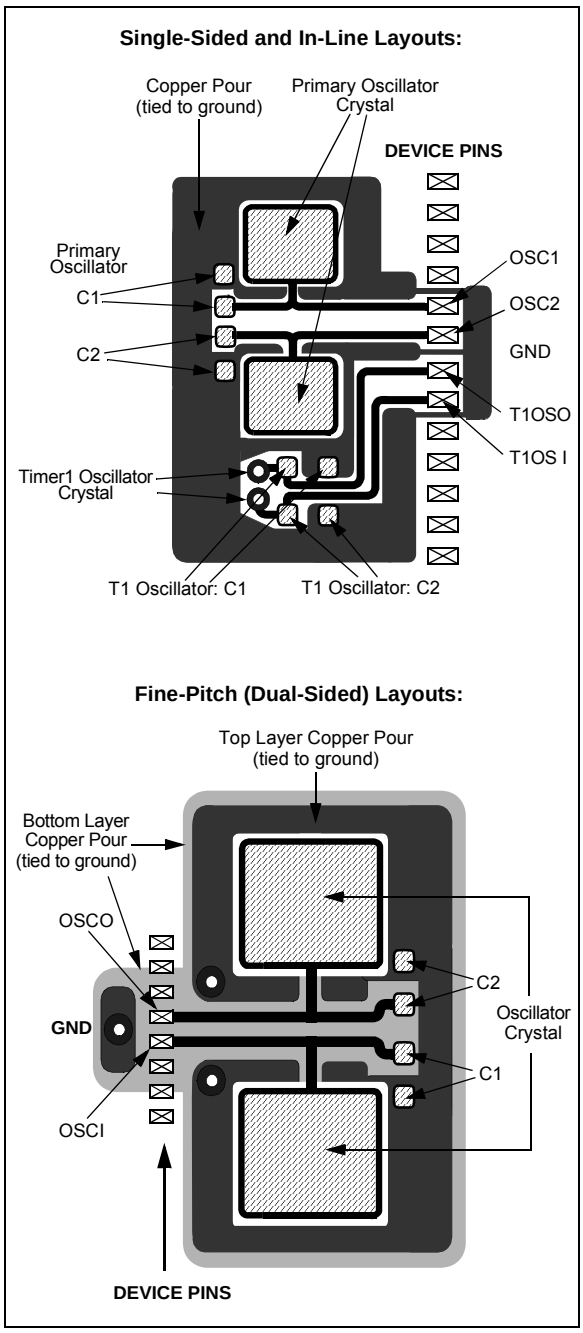
For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

- *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 2-5: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



NOTES:

3.0 OSCILLATOR CONFIGURATIONS

3.1 Overview

Devices in the PIC18F97J60 family incorporate an oscillator and microcontroller clock system that differs from standard PIC18FXXJXX devices. The addition of the Ethernet module, with its requirement for a stable 25 MHz clock source, makes it necessary to provide a primary oscillator that can provide this frequency as well as a range of different microcontroller clock speeds. An overview of the oscillator structure is shown in Figure 3-1.

Other oscillator features used in PIC18FXXJXX enhanced microcontrollers, such as the internal RC oscillator and clock switching, remain the same. They are discussed later in this chapter.

3.2 Oscillator Types

The PIC18F97J60 family of devices can be operated in five different oscillator modes:

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

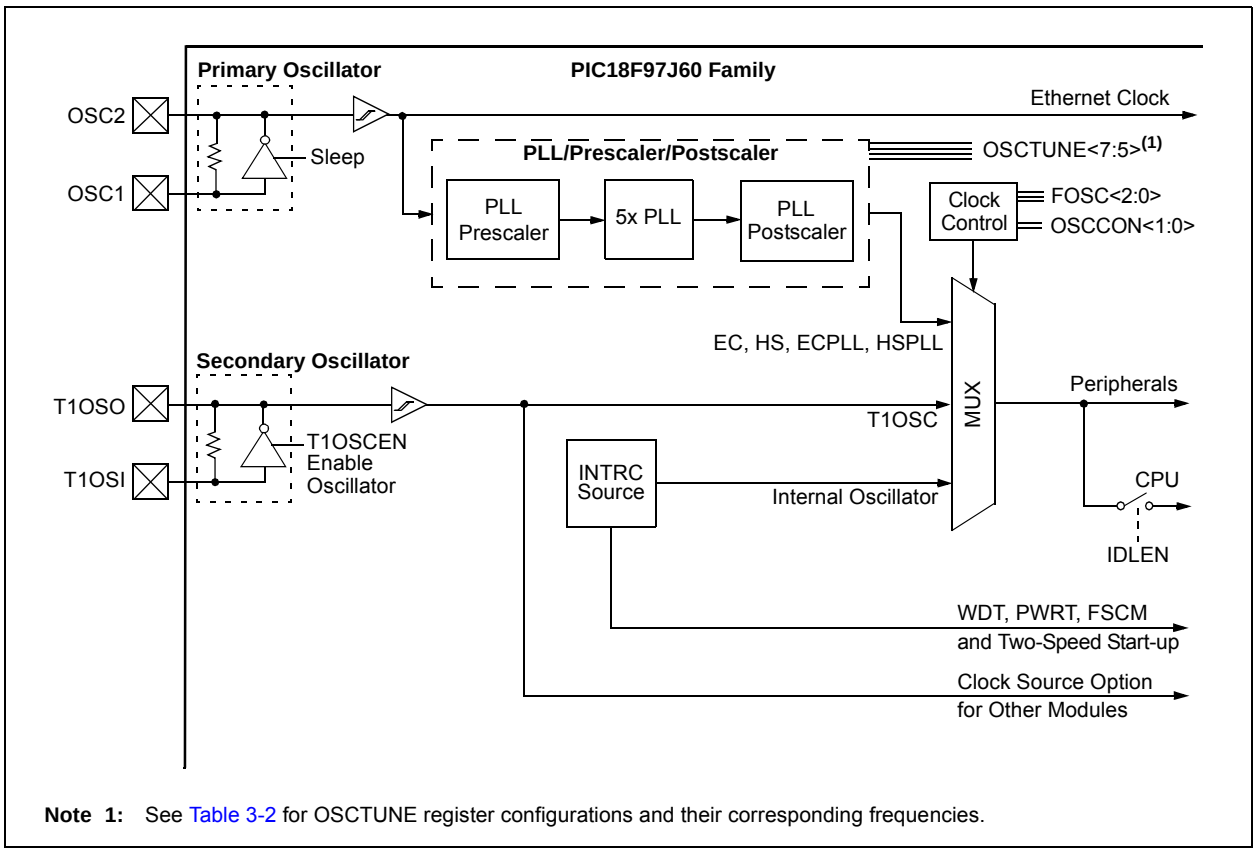
3.2.1 OSCILLATOR CONTROL

The oscillator mode is selected by programming the FOSC<2:0> Configuration bits. FOSC<1:0> bits select the default primary oscillator modes, while FOSC2 selects when INTRC may be invoked.

The OSCCON register (Register 3-2) selects the Active Clock mode. It is primarily used in controlling clock switching in power-managed modes. Its use is discussed in **Section 3.7.1 "Oscillator Control Register"**.

The OSCTUNE register (Register 3-1) is used to select the system clock frequency from the primary oscillator source by selecting combinations of prescaler/postscaler settings and enabling the PLL. Its use is described in **Section 3.6.1 "PLL Block"**.

FIGURE 3-1: PIC18F97J60 FAMILY CLOCK DIAGRAM



Note 1: See Table 3-2 for OSCTUNE register configurations and their corresponding frequencies.

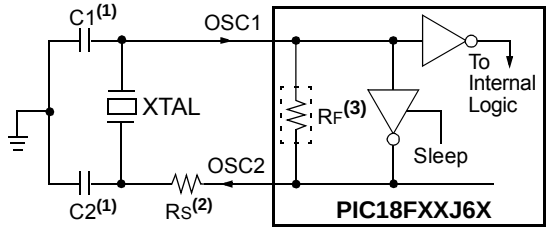
3.3 Crystal Oscillator/Ceramic Resonators (HS Modes)

In HS or HSPLL Oscillator modes, a crystal 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 crystal that is rated for parallel resonant operation.

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

FIGURE 3-2: CRYSTAL OSCILLATOR OPERATION (HS OR HSPLL CONFIGURATION)



Note 1: See Table 3-1 for initial values of C1 and C2.

2: A series resistor (RS) may be required for crystals with a low drive specification.

3: RF varies with the oscillator mode chosen.

TABLE 3-1: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq.	Typical Capacitor Values Tested:	
		C1	C2
HS	25 MHz	33 pF	33 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 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 crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

3: Rs may be required to avoid overdriving crystals with low drive level specifications.

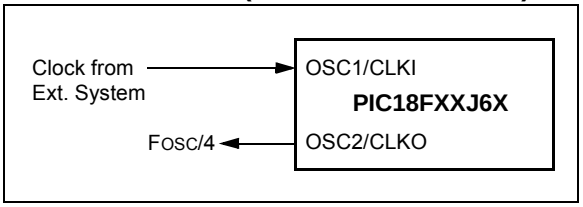
4: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

3.4 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.

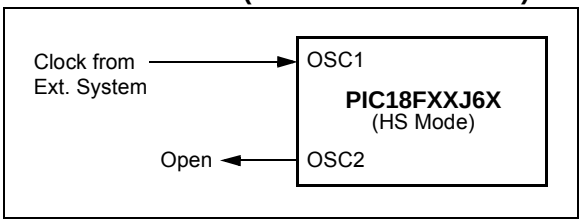
In the EC Oscillator mode, the oscillator frequency, divided by 4, is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 3-3 shows the pin connections for the EC Oscillator mode.

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 OSC2 pin is left open. Current consumption in this configuration will be somewhat higher than EC mode, as the internal oscillator's feedback circuitry will be enabled (in EC mode, the feedback circuit is disabled).

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



3.5 Internal Oscillator Block

The PIC18F97J60 family of devices includes an internal oscillator source (INTRC) which provides a nominal 31 kHz output. The INTRC is enabled on device power-up and clocks the device during its configuration cycle until it enters operating mode. INTRC is also enabled if it is selected as the device clock source or if any of the following are enabled:

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

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

The INTRC can also be optionally configured as the default clock source on device start-up by setting the FOSC2 Configuration bit. This is discussed in **Section 3.7.1 "Oscillator Control Register"**.

3.6 Ethernet Operation and the Microcontroller Clock

Although devices of the PIC18F97J60 family can accept a wide range of crystals and external oscillator inputs, they must always have a 25 MHz clock source when used for Ethernet applications. No provision is made for internally generating the required Ethernet clock from a primary oscillator source of a different frequency. A frequency tolerance is specified, likely excluding the use of ceramic resonators. See **Section 28.0 "Electrical Characteristics"**, Table 28-6, Parameter 5, for more details.

3.6.1 PLL BLOCK

To accommodate a range of applications and microcontroller clock speeds, a separate PLL block is incorporated into the clock system. It consists of three components:

- A configurable prescaler (1:2 or 1:3)
- A 5x PLL frequency multiplier
- A configurable postscaler (1:1, 1:2, or 1:3)

The operation of the PLL block's components is controlled by the OSCTUNE register (Register 3-1). The use of the PLL block's prescaler and postscaler, with or without the PLL itself, provides a range of system clock frequencies to choose from, including the unaltered 25 MHz of the primary oscillator. The full range of possible oscillator configurations compatible with Ethernet operation is shown in Table 3-2.

REGISTER 3-1: OSCTUNE: PLL BLOCK CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
PPST1	PLLEN(1)	PPST0	PPRE	—	—	—	—
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 **PPST1:** PLL Postscaler Configuration bit
1 = Divide-by-2
0 = Divide-by-3

bit 6 **PLLEN:** 5x Frequency Multiplier PLL Enable bit(1)
1 = PLL is enabled
0 = PLL is disabled

bit 5 **PPST0:** PLL Postscaler Enable bit
1 = Postscaler is enabled
0 = Postscaler is disabled

bit 4 **PPRE:** PLL Prescaler Configuration bit
1 = Divide-by-2
0 = Divide-by-3

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

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

TABLE 3-2: DEVICE CLOCK SPEEDS FOR VARIOUS PLL BLOCK CONFIGURATIONS

5x PLL	PLL Prescaler	PLL Postscaler	PLL Block Configuration (OSCTUNE<7:4>)	Clock Frequency (MHz)
Enabled	÷2	Disabled	x101	**(Note 1)**
		÷2	1111	31.2500
		÷3	0111	20.8333
	÷3	Disabled	x100	41.6667
		÷2	1110	20.8333
		÷3	0110	13.8889
Disabled	Disabled(2)	Disabled	x00x	25 (Default)
	÷2	÷2	1011	6.2500
		÷3	0011	4.1667
	÷3	÷2	1010	4.1667
		÷3	0010	2.7778

Legend: x = Don't care

Note 1: Reserved configuration; represents a clock frequency beyond the microcontroller's operating range.

2: The prescaler is automatically disabled when the PLL and postscaler are both disabled.

3.7 Clock Sources and Oscillator Switching

The PIC18F97J60 family of devices includes a feature that allows the device clock source to be switched from the main oscillator to an alternate clock source. These devices also offer two alternate clock sources. When an alternate clock source is enabled, the various power-managed operating modes are available.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator block

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

The **secondary oscillators** are those external sources not connected to the OSC1 or OSC2 pins. These sources may continue to operate even after the controller is placed in a power-managed mode. The PIC18F97J60 family of devices offers the Timer1 oscillator as a secondary oscillator. In all power-managed modes, this oscillator is often the time base for functions such as a Real-Time Clock (RTC).

Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T13CKI and RC1/T1OSI pins. Loading capacitors are also connected from each pin to ground. The Timer1 oscillator is discussed in greater detail in **Section 13.3 "Timer1 Oscillator"**.

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

The clock sources for the PIC18F97J60 family devices are shown in Figure 3-1. See **Section 25.0 "Special Features of the CPU"** for Configuration register details.

3.7.1 OSCILLATOR CONTROL REGISTER

The OSCCON register (Register 3-2) controls several aspects of the device clock's operation, both in full-power operation and in power-managed modes.

The System Clock Select bits, SCS<1:0>, select the clock source. The available clock sources are the primary clock (defined by the FOSC<2:0> Configuration bits), the secondary clock (Timer1 oscillator) and the internal oscillator. The clock source changes after one or more of the bits are changed, 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 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 bit is set, the INTRC source 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 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.

REGISTER 3-2: OSCCON: OSCILLATOR CONTROL REGISTER

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

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

bit 7 **IDLEN:** Idle Enable bit
- 1 = Device enters Idle mode on SLEEP instruction
- 0 = Device enters Sleep mode on SLEEP instruction

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

bit 3 **OSTS:** Oscillator Status bit(1)
- 1 = Device is running from oscillator source defined when SCS<1:0> = 00
- 0 = Device is running from oscillator source defined when SCS<1:0> = 01, 10 or 11

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

bit 1-0 **SCS<1:0>:** System Clock Select bits
- 11 = Internal oscillator
- 10 = Primary oscillator
- 01 = Timer1 oscillator

When FOSC2 = 1:
- 00 = Primary oscillator

When FOSC2 = 0:
- 00 = Internal oscillator

Note 1: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

3.7.1.1 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 HC or EC modes), is used as the primary clock source on device Resets.

The default clock configuration on Reset can be changed with the FOSC2 Configuration bit. This bit affects the clock source selection setting when SCS<1:0> = 00. When FOSC2 = 1 (default), the oscillator source defined by FOSC<1:0> is selected whenever SCS<1:0> = 00. When FOSC2 = 0, the INTRC oscillator is selected whenever SCS<1:0> = 00. Because the SCS bits are cleared on Reset, the FOSC2 setting also changes the default oscillator mode on Reset.

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

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

3.7.2 OSCILLATOR TRANSITIONS

PIC18F97J60 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"**.

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

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

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

In 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 25.2 "Watchdog Timer (WDT)"** through **Section 25.5 "Fail-Safe Clock Monitor"** for more information on WDT, Fail-Safe Clock Monitor and Two-Speed Start-up).

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

Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The INTRC is required to support WDT operation. The Timer1 oscillator may be operating to support a Real-Time Clock. Other features may be operating that do not require a device clock source (i.e., MSSP slave, PSP, INTx pins and others). Peripherals that may add significant current consumption are listed in **Section 28.2 "DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial)"**

3.9 Power-up Delays

Power-up delays are controlled by two timers, so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply is stable under normal circumstances, and the primary clock is operating and stable. For additional information on power-up delays, see **Section 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 28-12); 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 28-12), 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 is disabled at quiescent voltage level	Feedback inverter is disabled at quiescent voltage level

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 PIC18F97J60 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® MCU devices. One is the clock switching feature, offered in other PIC18 devices, allowing the controller to use the Timer1 oscillator in place of the primary oscillator. Also included is the Sleep mode, offered by all PIC MCU devices, 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<7,1:0>		Module Clocking		Available Clock and Oscillator Source
	IDLEN(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 25.4 “Two-Speed Start-up”** for details). In this mode, the OSTS bit is set. (see **Section 3.7.1 “Oscillator Control Register”**).

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, 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

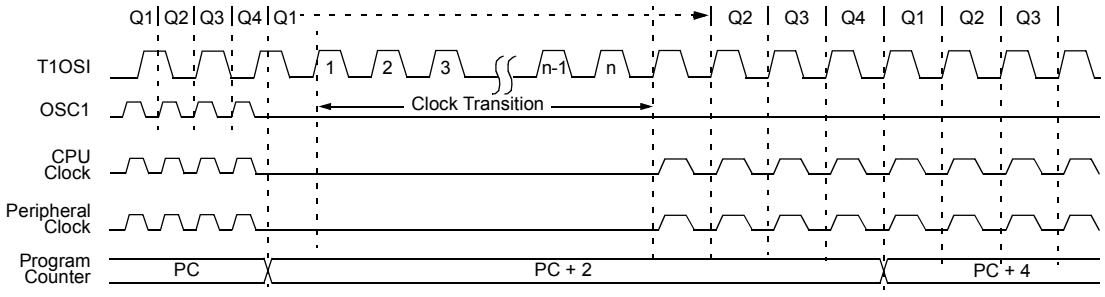
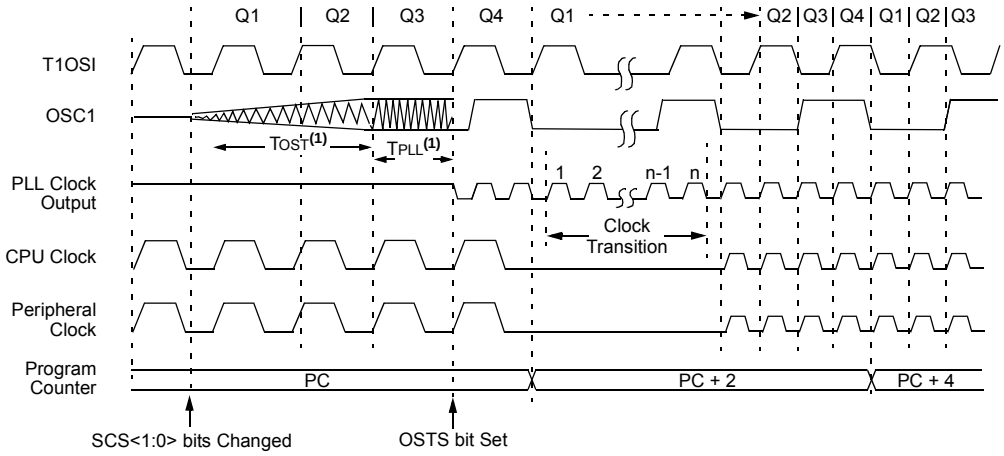


FIGURE 4-2: TRANSITION TIMING FROM SEC_RUN MODE TO PRI_RUN MODE (HSPLL)



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<1:0> 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 Fail-Safe Clock Monitor is enabled.

FIGURE 4-3: TRANSITION TIMING TO RC_RUN MODE

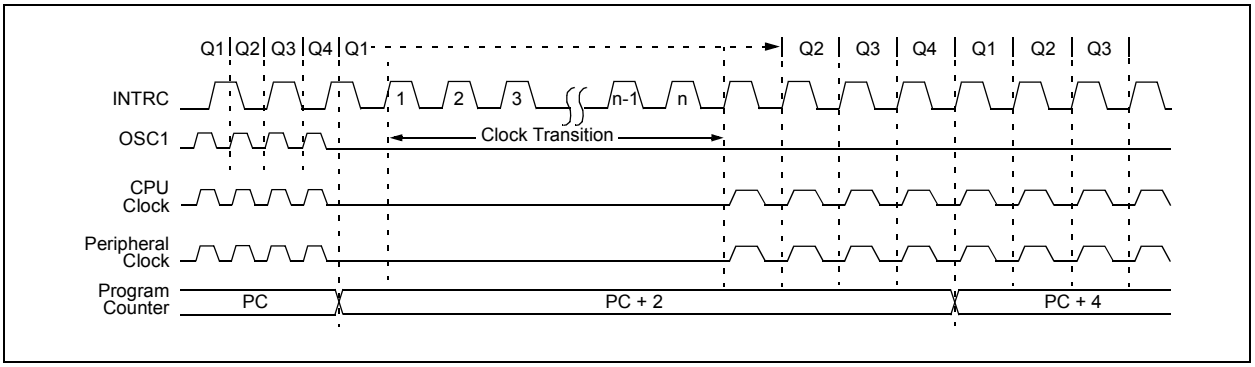
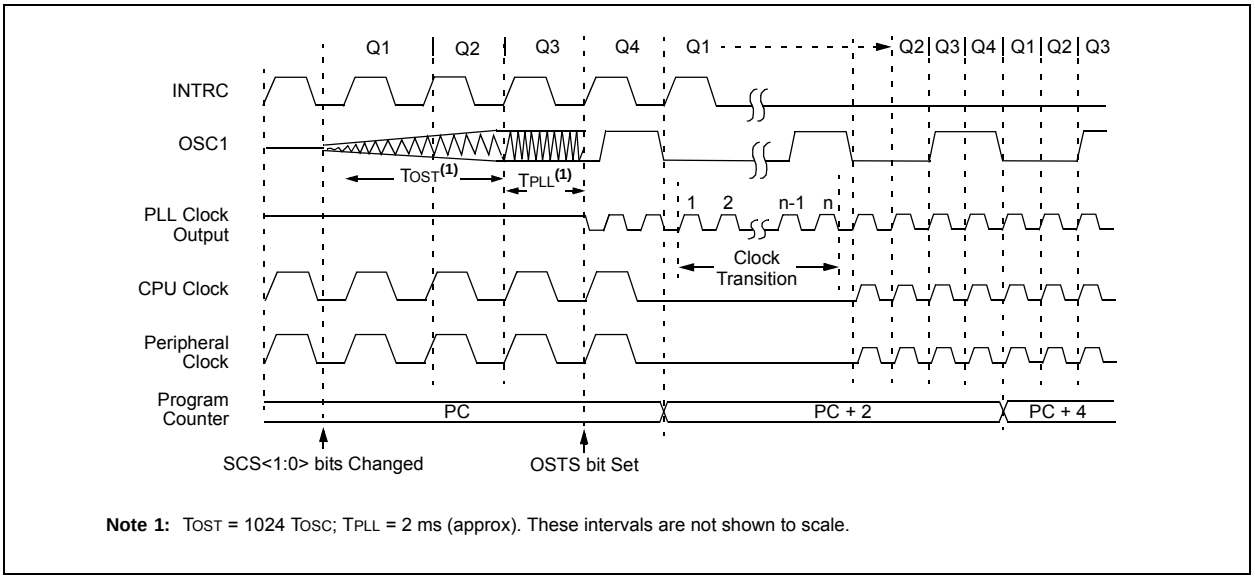


FIGURE 4-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



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 MCU devices. It is entered by clearing the IDLEN bit (the default state on device Reset) and executing the SLEEP instruction. This shuts down the selected oscillator (Figure 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 25.0 "Special Features of the CPU"**). In either case, the OSTS bit is set when the primary clock is providing the device clocks. The IDLEN and SCS bits are not affected by the wake-up.

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

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

Since the CPU is not executing instructions, the only exits from any of the Idle modes are by interrupt, WDT time-out or a Reset. When a wake event occurs, CPU execution is delayed by an interval of TCSD (Parameter 38, Table 28-12) 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 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

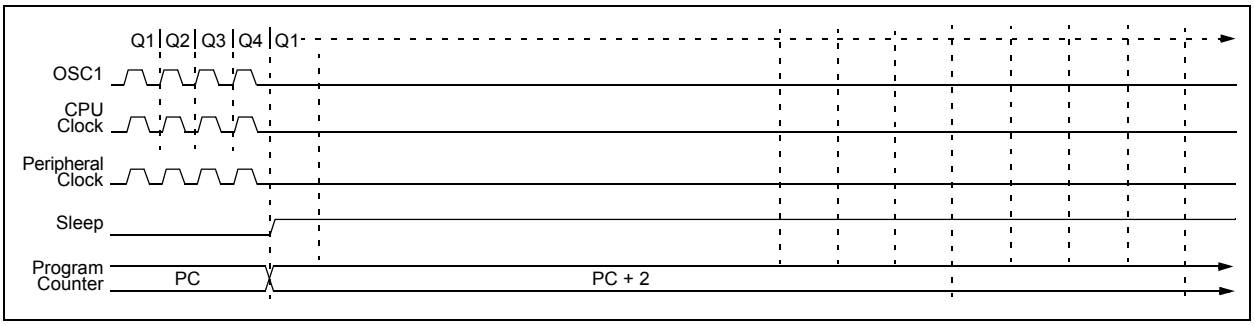
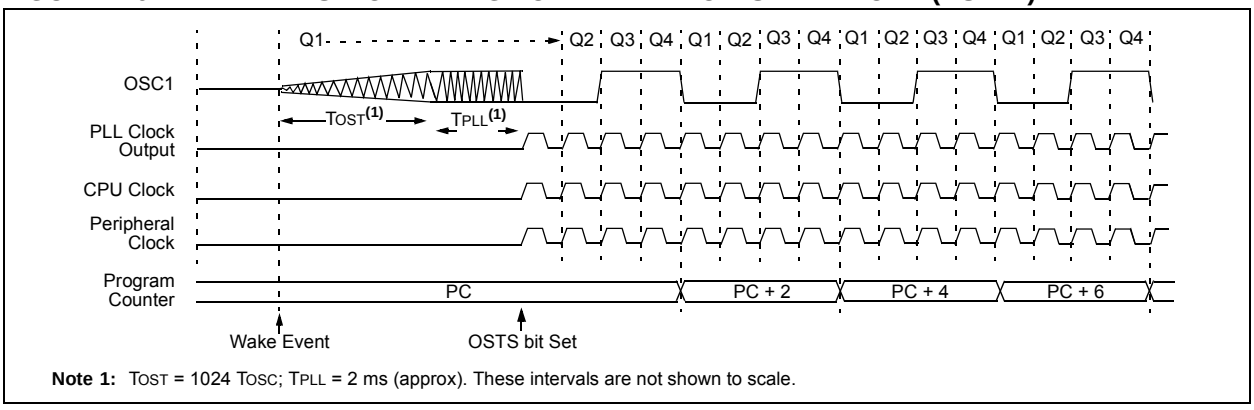


FIGURE 4-6: TRANSITION TIMING FOR WAKE FROM SLEEP MODE (HSPLL)



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<1:0> 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

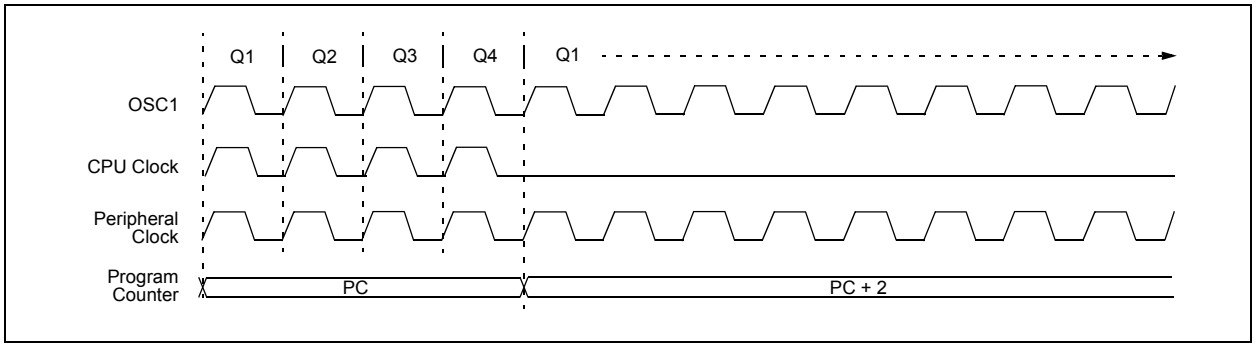
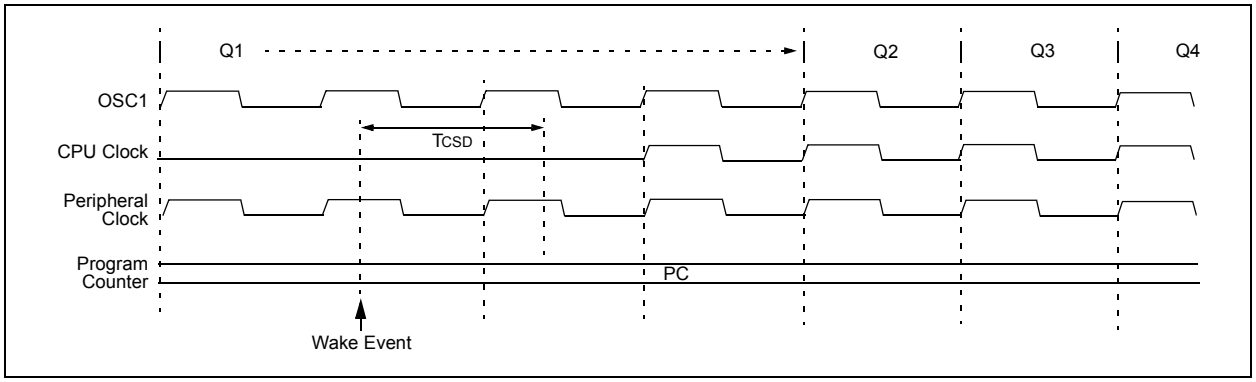


FIGURE 4-8: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE



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

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 modes 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 the 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 25.2 "Watchdog Timer (WDT)"**).

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

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

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 TIMER 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
- 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 the 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 PIC18F97J60 family of devices differentiates between various kinds of Reset:

a) MCLR Reset during normal operation
b) MCLR Reset during power-managed modes
c) Power-on Reset (POR)
d) Brown-out Reset (BOR)
e) Configuration Mismatch (CM)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset
i) Watchdog Timer (WDT) Reset during execution

This section discusses Resets generated by hard events (MCLR), power events (POR and BOR) and Configuration Mismatches (CM). It also 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 25.2 "Watchdog Timer (WDT)"**.

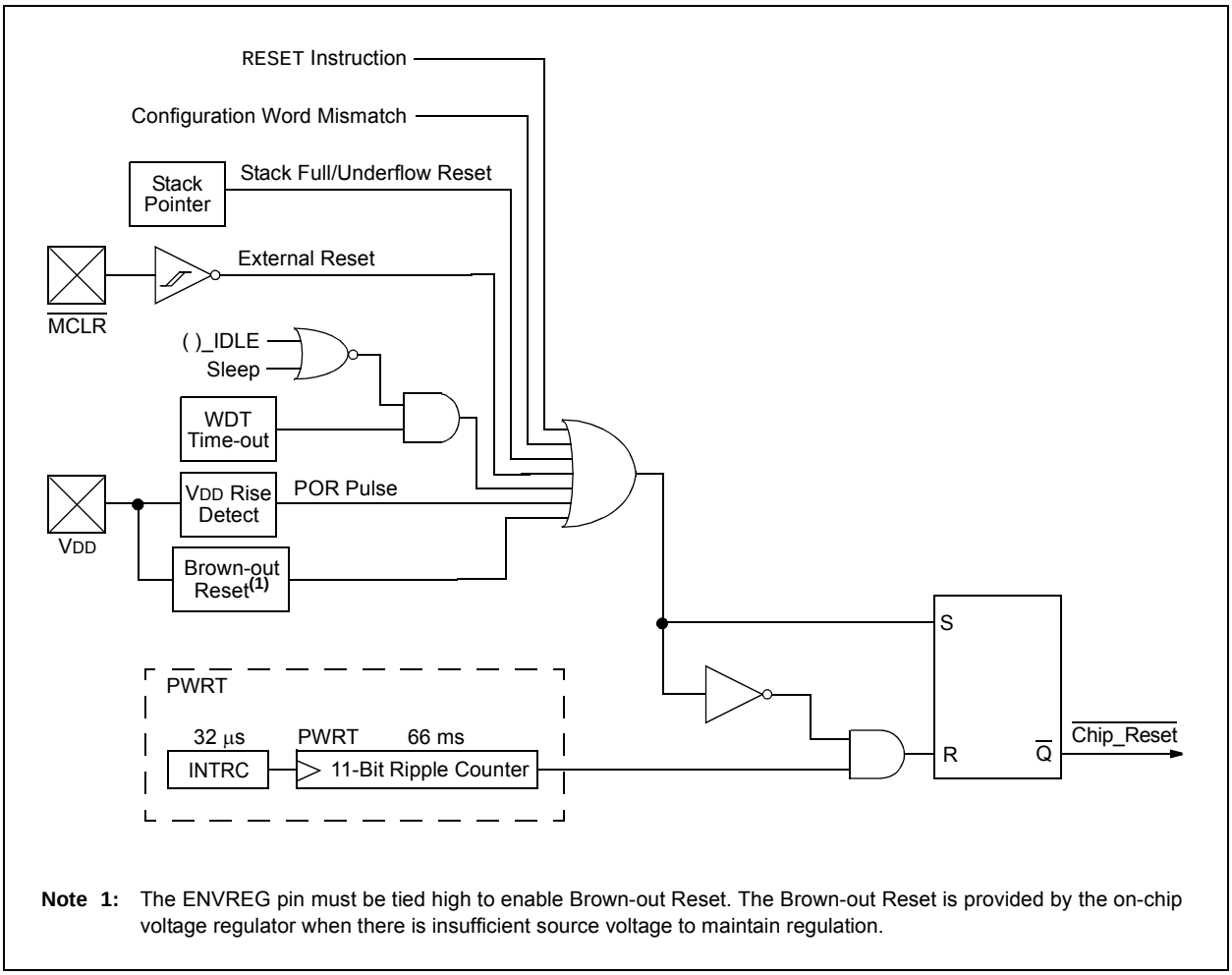
A simplified block diagram of the on-chip Reset circuit is shown in Figure 5-1.

5.1 RCON Register

Device Reset events are tracked through the RCON register (Register 5-1). The lower six 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 (PIC16CXXX 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 after a Configuration Mismatch 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 Timer 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.

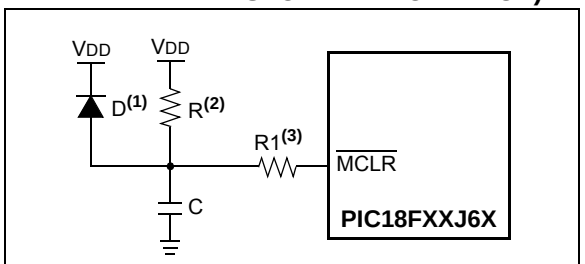
POR events are captured by the $\overline{POR}$ bit (RCON<1>). The state of the bit is set to '0' whenever a Power-on Reset occurs; it does not change for any other Reset event. $\overline{POR}$ is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to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

5.4 Brown-out Reset (BOR)

The PIC18F97J60 family of devices incorporates a simple BOR function when the internal regulator is enabled (ENVREG pin is tied to VDD). Any drop of VDD below VBOR (Parameter D005), for greater than time, TBOR (Parameter 35), will reset the device. A Reset may or may not occur if VDD falls below VBOR for less than TBOR. The chip will remain in Brown-out Reset until VDD rises above VBOR.

Once a BOR has occurred, the Power-up Timer will keep the chip in Reset for TPWRT (Parameter 33). If VDD drops below VBOR while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be initialized. Once VDD rises above VBOR, the Power-up Timer will execute the additional time delay.

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

A CM Reset behaves similarly to a Master Clear Reset, RESET instruction, WDT time-out or Stack Event Reset. 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)

PIC18F97J60 family of devices incorporates 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 PIC18F97J60 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 = 66 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

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.

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 PIC18FXXJ6X device operating in parallel.

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

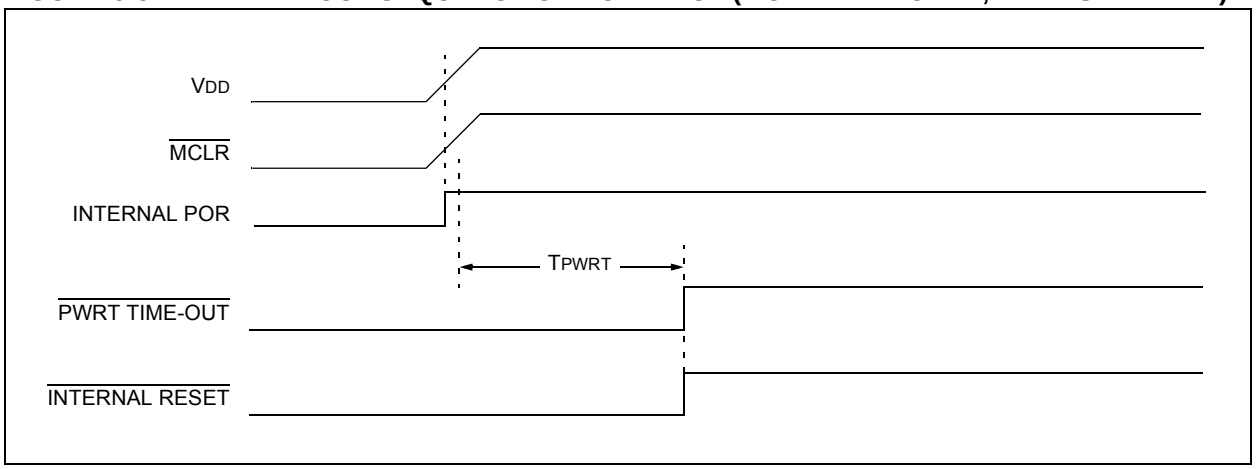


FIGURE 5-4: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO VDD): CASE 1

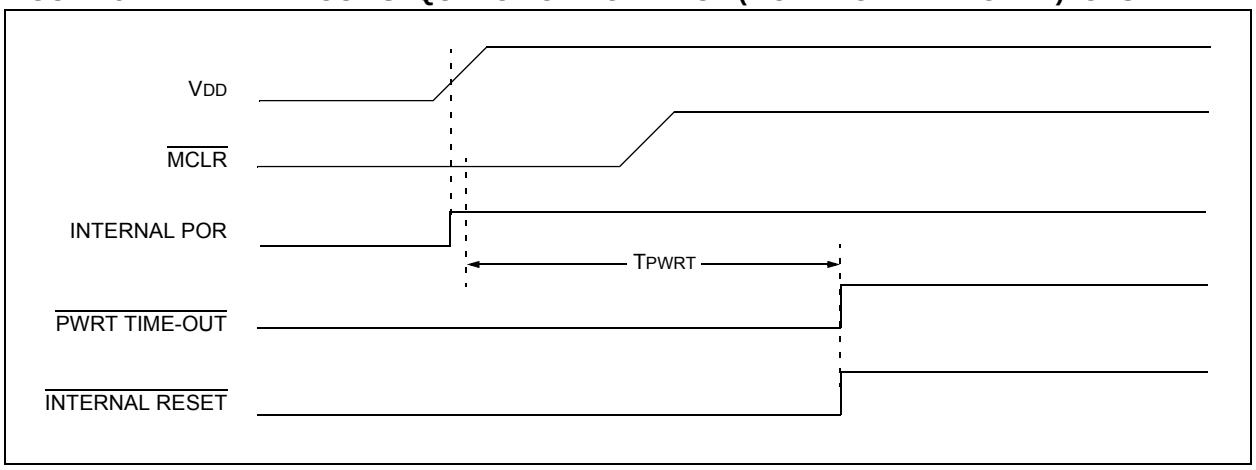


FIGURE 5-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

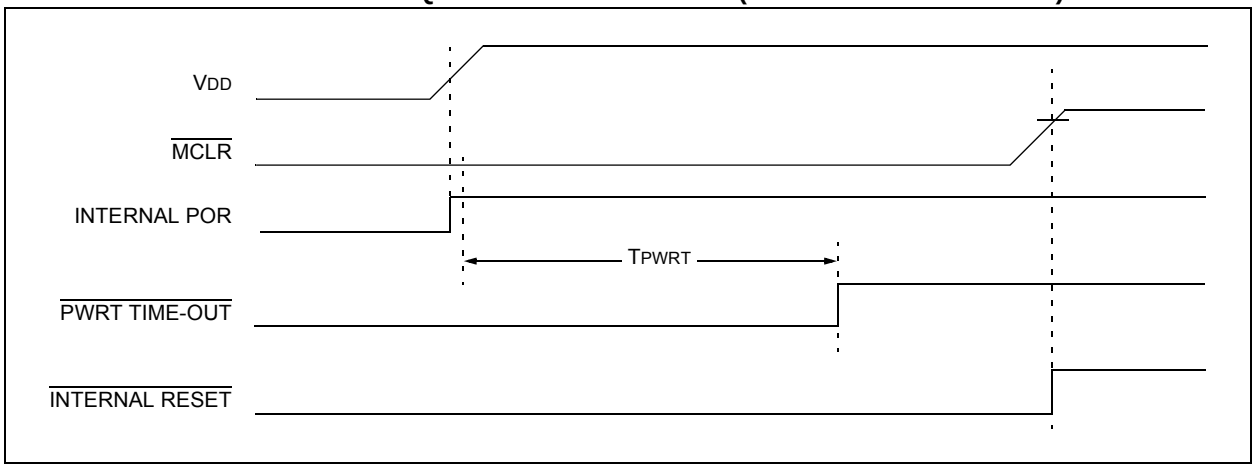
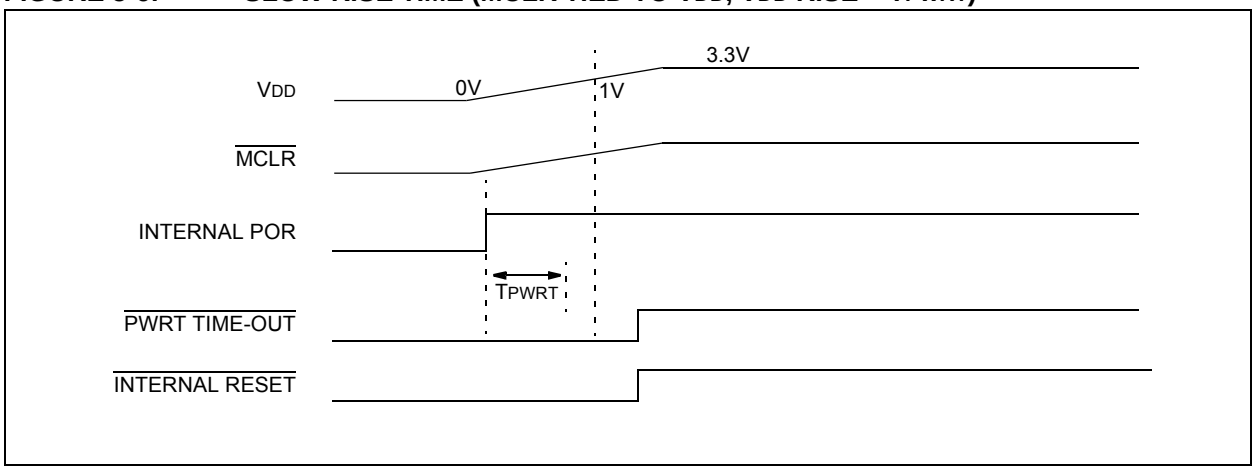


FIGURE 5-6: SLOW RISE TIME (MCLR TIED TO VDD, VDD RISE > TPWRT)



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
Configuration Mismatch Reset	0000h	0	u	u	u	u	u	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
$\overline{MCLR}$ 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 full power or power-managed Run modes	0000h	u	u	0	u	u	u	u	u
WDT time-out during power-managed Idle or Sleep modes	PC + 2	u	u	0	0	u	u	u	u
Interrupt exit from power-managed modes	PC + 2	u	u	u	0	u	u	u	u

Legend: u = unchanged

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

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
TOSU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---0 uuuu(1)
TOSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR0H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTINC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
POSTDEC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PREINC2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
PLUSW2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	N/A	N/A	N/A
FSR2H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
STATUS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0--- q-00	0--- q-00	u--- q-uu
ECON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 00--	0000 00--	uuuu uu--
WDTCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- ---0	---- ---0	---- ---u
RCON(4)	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-q1 1100	0-uq qquu	u-uu qquu
TMR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 0000	0-00 0000	u-uu uuuu
ADCON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CCPR3H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP1AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0111	0000 0111	uuuu uuuu
TMR3H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
T3CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	uuuu uuuu	uuuu uuuu
PSPCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXSTA1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000x	uuuu uuuu
EECON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- ----	---- ----	---- ----
EECON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 x00-	---0 x00-	---u uuu-
IPR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
PIR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(3)
PIE3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1-11	1111 1-11	uuuu u-uu
PIR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0-00	0000 0-00	uuuu u-uu(3)
PIE2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0-00	0000 0-00	uuuu u-uu
IPR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu(3)
PIE1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-00 --00	0-00 --00	u-uu --uu
OSCTUNE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 ----	0000 ----	uuuu ----
TRISJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 ----	--11 ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 ----	---1 ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 1111	---1 1111	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 111-	1111 111-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 1111	--11 1111	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -111	---- -111	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
TRISA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--11 1111	--11 1111	--uu uuuu
LATJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx ----	--uu ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
LATG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x ----	---u ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxx-	uuuu uuu-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx xxxx	--uu uuuu	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -xxx	---- -uuu	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	00xx xxxx	00uu uuuu	uuuu uuuu
PORTJ	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx ----	--uu ----	--uu ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x ----	---u ----	---u ----
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---x xxxx	---u uuuu	---u uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	111x xxxx	111u uuuu	uuuu uuuu
PORTF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	x000 000-	x000 000-	uuuu uuu-
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	x000 000-	x000 000-	uuuu uuu-
PORTE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--xx xxxx	--uu uuuu	--uu uuuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -xxx	---- -uuu	---- -uuu
	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0-0x 0000	0-0u 0000	u-uu uuuu
SPBRGH1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0100 0-00	0100 0-00	uuuu u-uu
SPBRGH2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
BAUDCON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0100 0-00	0100 0-00	uuuu u-uu
ERDPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 1010	---0 1010	---u uuuu
ERDPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 0101	1111 0101	uuuu uuuu
ECCP1DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TMR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
PR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	1111 1111
T4CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
CCPR4H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR4L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
CCP4CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR5L	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP5CON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	--00 0000	--00 0000	--uu uuuu
SPBRG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0010	0000 0010	uuuu uuuu
RCSTA2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 000x	0000 000x	uuuu uuuu
ECCP3AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP2AS	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2BUF	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP2ADD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2STAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2CON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
SSP2CON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDATA	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	xxxx xxxx	uuuu uuuu	uuuu uuuu
EIR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0-00	-000 0-00	-uuu u-uu
ECON2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	100- ----	100- ----	uuu- ----
ESTAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-0-0 -000	-0-0 -000	-u-u -uuu
EIE	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0-00	-000 0-00	-uuu u-uu
EDMACSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMACSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMADSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMADSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMANDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMANDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EDMASTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EDMASTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXWRPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ERXWRPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXRDPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0101	---0 0101	---u uuuu
ERXRDPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1010	1111 1010	uuuu uuuu
ERXNDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---1 1111	---1 1111	---u uuuu
ERXNDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1111	1111 1111	uuuu uuuu
ERXSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0101	---0 0101	---u uuuu
ERXSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1111 1010	1111 1010	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
ETXNDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ETXNDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ETXSTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
ETXSTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EWRPTH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EWRPTL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPKTCNT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
ERXFCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	1010 0001	1010 0001	uuuu uuuu
EPMOH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EPMOL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMCSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMCSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM7	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EPMM0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT7	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EHT0	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIRDH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIRDL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIWRH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIWRL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MIREGADR	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
MICMD	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- --00	---- --00	---- --uu
MAMXFLH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0110	0000 0110	uuuu uuuu
MAMXFLL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	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.

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

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset, RESET Instruction, Stack Resets, CM Reset	Wake-up via WDT or Interrupt
MAIPGH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MAIPGL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MABBIPG	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 0000	-000 0000	-uuu uuuu
MACON4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	-000 --00	-000 --00	-uuu --uu
MACON3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MACON1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---0 0000	---0 0000	---u uuuu
EPAUSH	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0001 0000	0001 0000	000u uuuu
EPAUSL	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
EFLOCON	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- -000	---- -000	---- -uuu
MISTAT	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	---- 0000	---- 0000	---- uuuu
MAADR2	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR1	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR4	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR3	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR6	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	0000 0000	0000 0000	uuuu uuuu
MAADR5	PIC18F6XJ6X	PIC18F8XJ6X	PIC18F9XJ6X	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.

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.

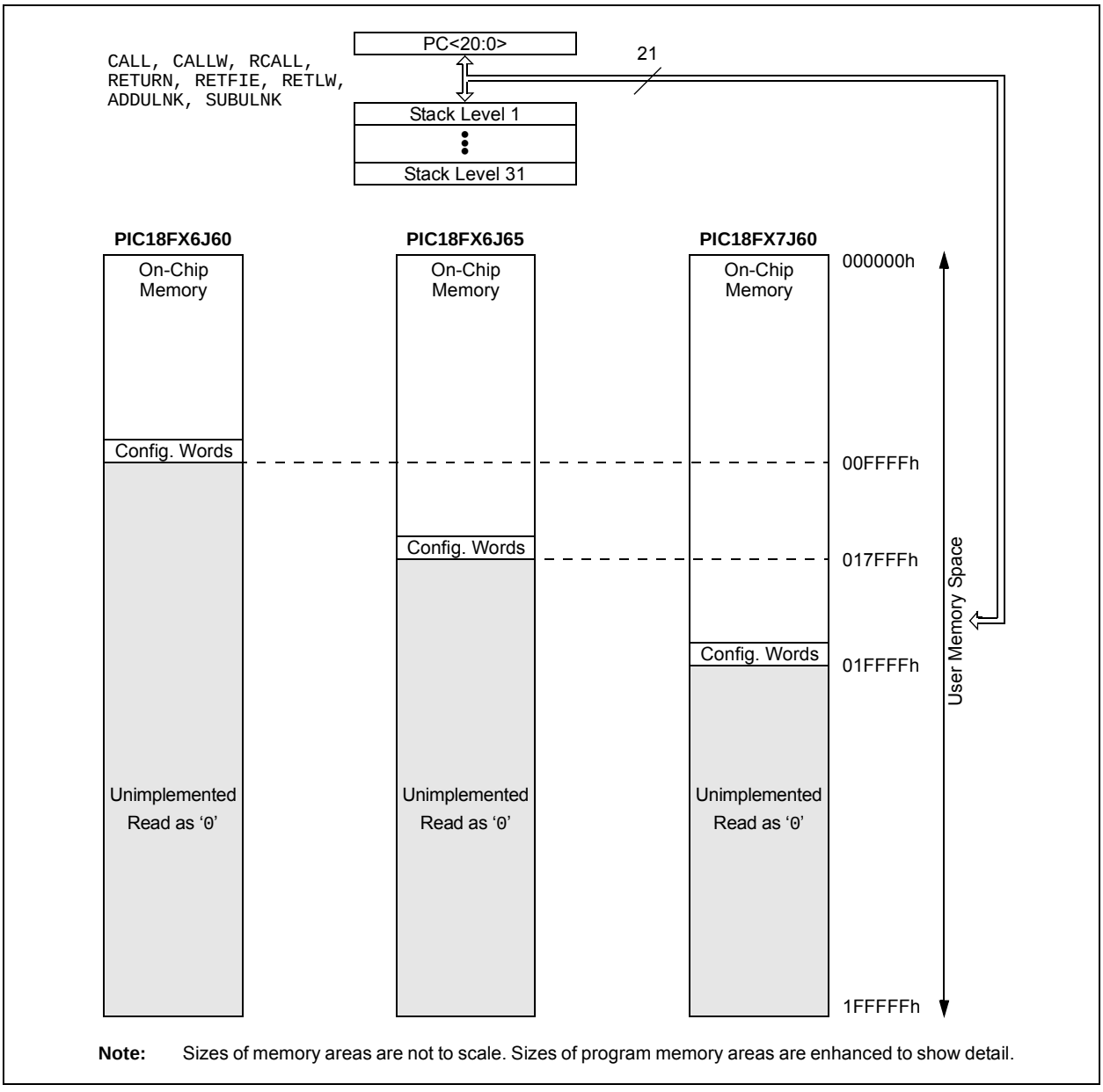
Additional detailed information on the operation of the Flash program memory is provided in **Section 7.0 "Flash Program Memory"**.

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 PIC18F97J60 family offers three sizes of on-chip Flash program memory, from 64 Kbytes (up to 32,764 single-word instructions) to 128 Kbytes (65,532 single-word instructions). The program memory maps for individual family members are shown in Figure 6-1.

FIGURE 6-1: MEMORY MAPS FOR PIC18F97J60 FAMILY DEVICES



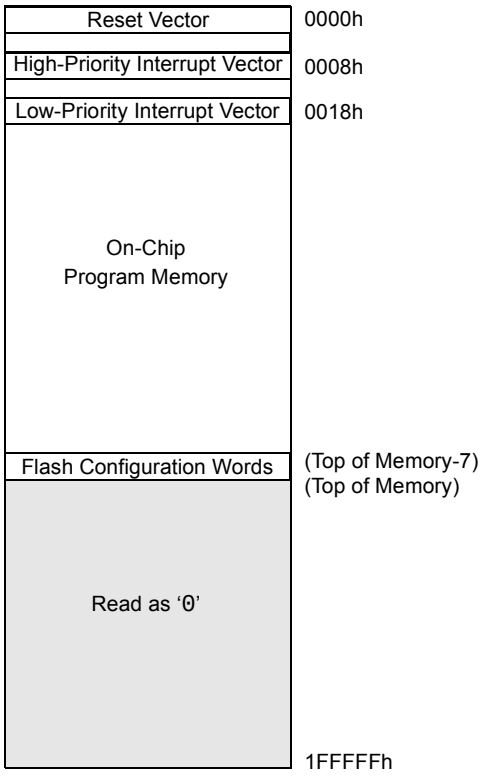
Note: Sizes of memory areas are not to scale. Sizes of program memory areas are enhanced to show detail.

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 PIC18F97J60 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 the PIC18F97J60 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 Words for devices in the PIC18F97J60 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 25.1 "Configuration Bits"**.

TABLE 6-1: FLASH CONFIGURATION WORDS FOR PIC18F97J60 FAMILY DEVICES

Device	Program Memory (Kbytes)	Configuration Word Addresses
PIC18F66J60	64	FFF8h to FFFFh
PIC18F86J60		
PIC18F96J60		
PIC18F66J65	96	17FF8h to 17FFFh
PIC18F86J65		
PIC18F96J65		
PIC18F67J60	128	1FFF8h to 1FFFFh
PIC18F87J60		
PIC18F97J60		

6.1.3 PIC18F9XJ60/9XJ65 PROGRAM MEMORY MODES

The 100-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. (Also see **Section 25.1 "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 and 80-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

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT(1)	BW(1)	EMB1(1)	EMB0(1)	EASHFT(1)	—	—	—
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)
1 = Wait states for operations on external memory bus are disabled
0 = Wait states for operations on external memory bus are enabled and selected by MEMCON<5:4>

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

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

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

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

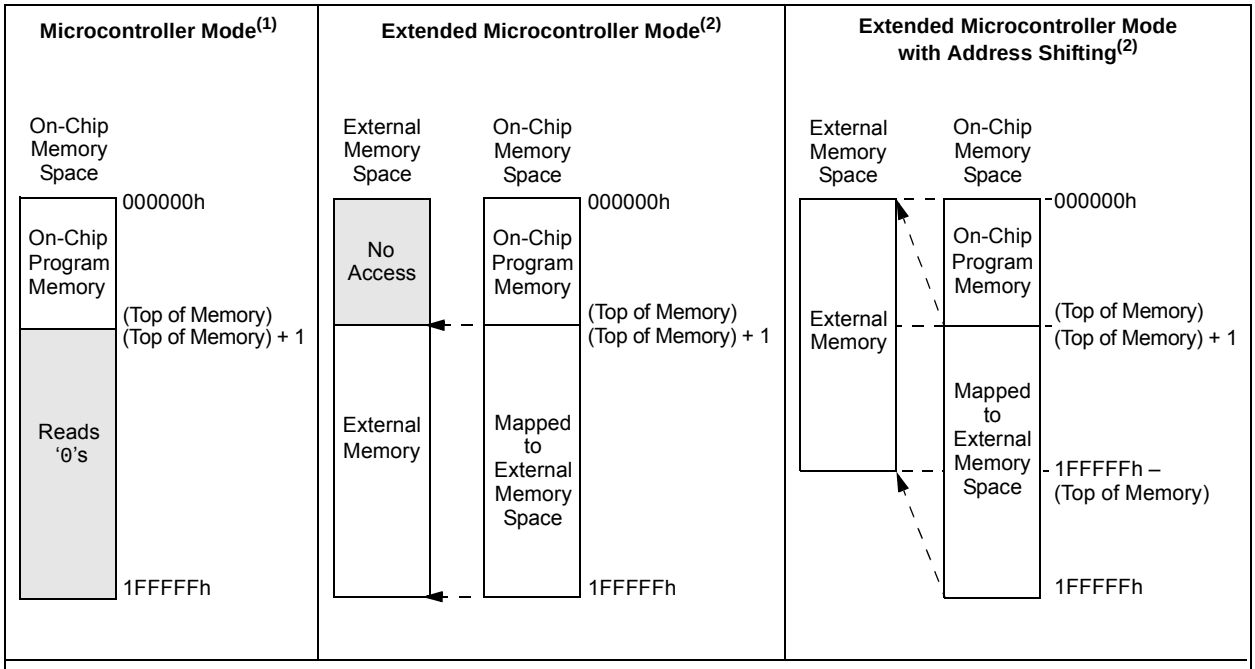
Note 1: Implemented on 100-pin devices only.

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 PIC18F97J60 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 and 80-pin devices and the default on 100-pin devices.

2: These modes are only available in 100-pin devices.

TABLE 6-2: MEMORY ACCESS FOR PIC18F9XJ60/9XJ65 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 to the 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.

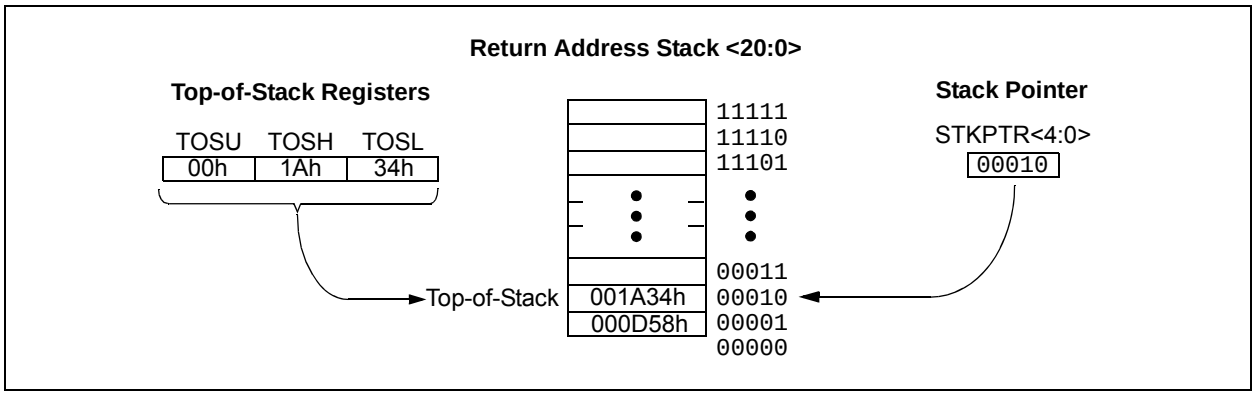
The Stack Pointer is initialized to '00000' after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of '00000'; this is only a Reset value. Status bits indicate if the stack is full, has overflowed or has underflowed.

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 bit. 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 25.1 "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 returns a value of zero to the PC, and sets the STKUNF bit, while the Stack Pointer remains at zero. The STKUNF bit will remain set until cleared by software or until a POR occurs.

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

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 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 bit is cleared by user software or a Power-on Reset.

6.1.7 FAST REGISTER STACK

A Fast Register Stack (FSR) 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.

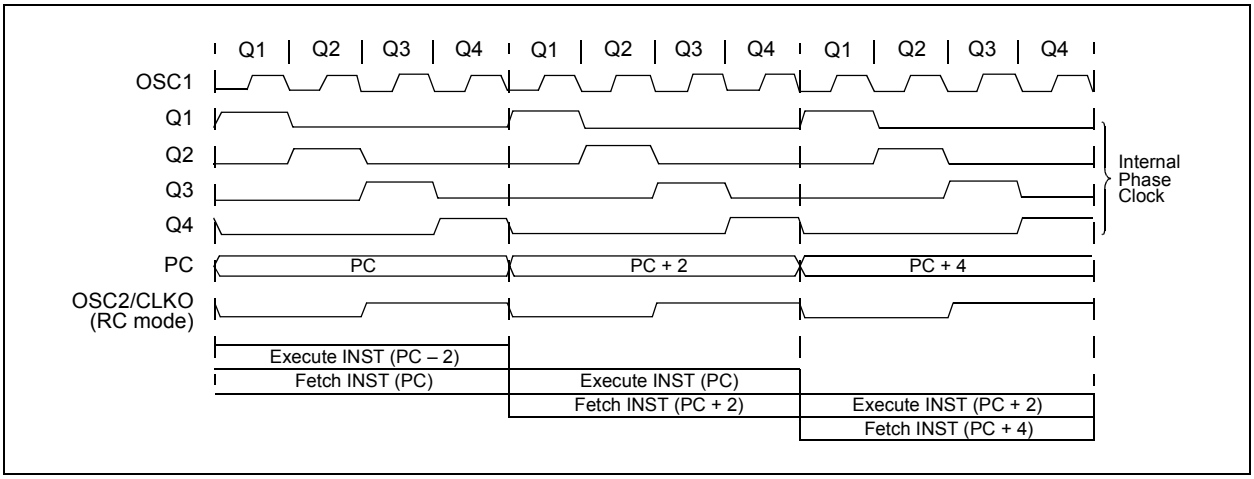
6.2.2 INSTRUCTION FLOW/PIPELINING

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).

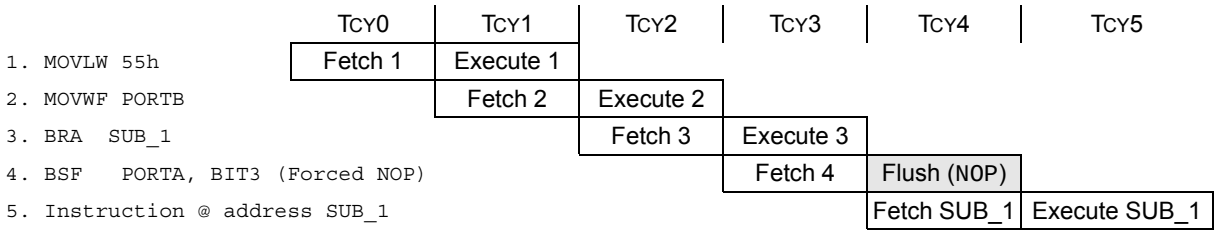
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

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 (LSB) of an instruction word is always stored in a program memory location with an even address (LSb = 0). To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSb will always read '0' (see **Section 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 26.0 "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 (MSbs); the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence, immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped, for some reason, and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 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 is 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 addressable memory. The memory space is divided into 16 banks that contain 256 bytes each. All of the PIC18F97J60 family devices implement all available banks and provide 3808 bytes of data memory available to the user. Figure 6-7 shows 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 (most 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 the majority of 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 (LSbs). Only the four lower bits of the BSR are implemented (BSR3:BSR0). The upper four bits are unused; they will always read '0' and cannot be written to. The BSR can be loaded directly by using the MOVLB instruction.

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

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 PIC18F97J60 FAMILY DEVICES

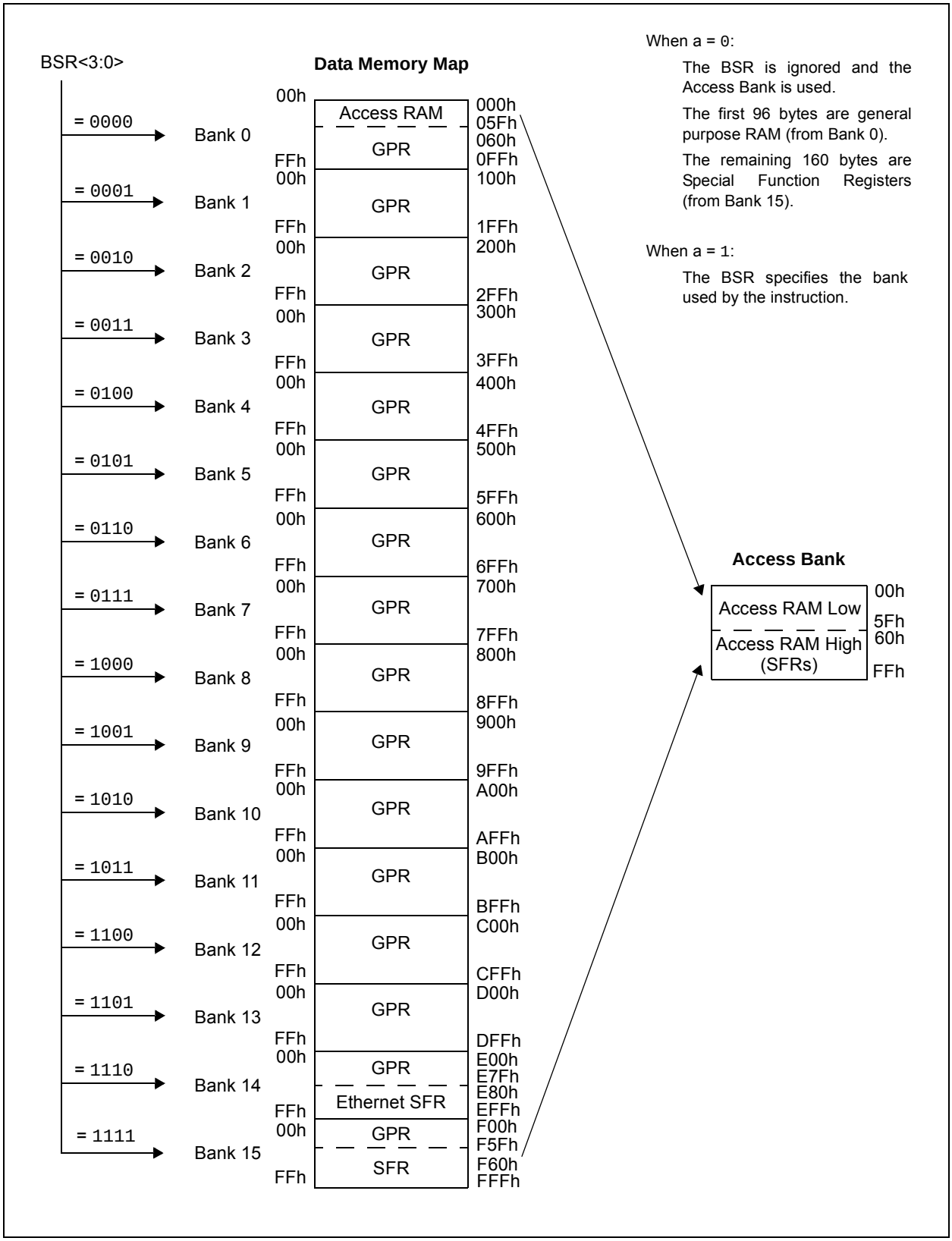
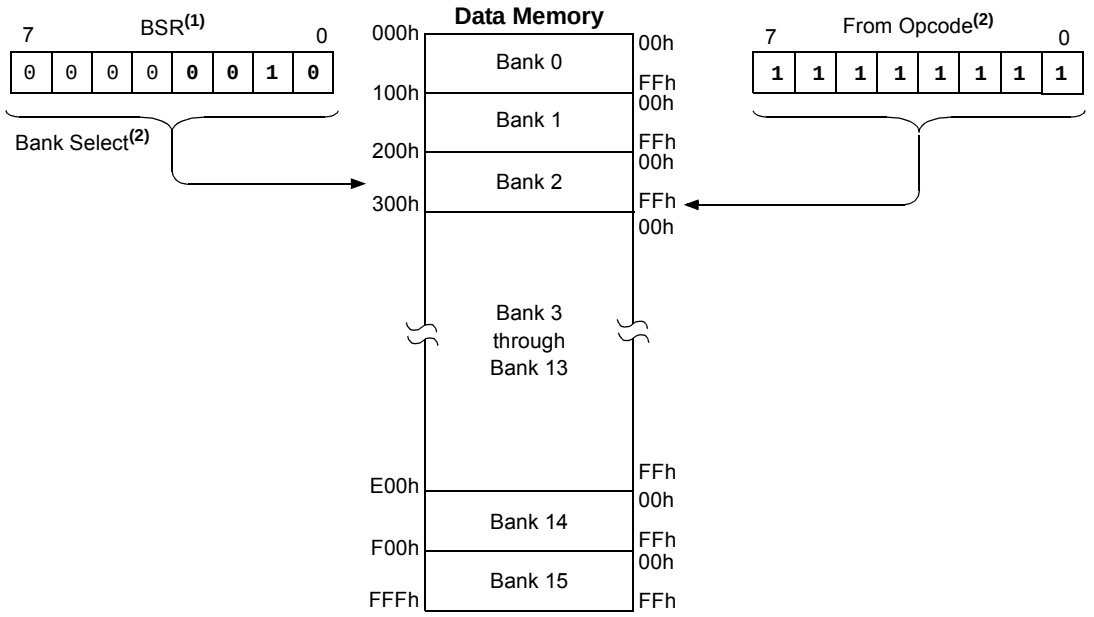


FIGURE 6-8: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR3:BSR0) 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 block is known as the “Access RAM” and is composed of GPRs. The upper block 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.

The main group of 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). These 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. A list of SFRs is given in Table 6-3; a full description is provided in Table 6-5.

TABLE 6-3: SPECIAL FUNCTION REGISTER MAP FOR PIC18F97J60 FAMILY DEVICES

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	SPBRGH2
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	MEMCON(4)	F7Ch	BAUDCON2
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	ERDPTH
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ(3)	F7Ah	ERDPTL
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH(3)	F79h	ECCP1DEL
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG	F78h	TMR4
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF	F77h	PR4
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE	F76h	T4CON
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR4H
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC	F74h	CCPR4L
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCP4CON
FF2h	INTCON	FD2h	ECON1	FB2h	TMR3L	F92h	TRISA	F72h	CCPR5H
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(3)	F71h	CCPR5L
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(3)	F70h	CCP5CON
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	SPBRG2
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	RCREG2
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TXREG2
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	TXSTA2
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	RCSTA2
FEAh	FSR0H	FCAh	T2CON	FAAh	—(2)	F8Ah	LATB	F6Ah	ECCP3AS
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	—(2)	F89h	LATA	F69h	ECCP3DEL
FE8h	WREG	FC8h	SSP1ADD	FA8h	—(2)	F88h	PORTJ(3)	F68h	ECCP2AS
FE7h	INDF1(1)	FC7h	SSP1STAT	FA7h	EECON2(1)	F87h	PORTH(3)	F67h	ECCP2DEL
FE6h	POSTINC1(1)	FC6h	SSP1CON1	FA6h	EECON1	F86h	PORTG	F66h	SSP2BUF
FE5h	POSTDEC1(1)	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	SSP2ADD
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	SSP2STAT
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	SSP2CON1
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SSP2CON2
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	EDATA
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	EIR

Note 1: This is not a physical register.
2: Unimplemented registers are read as '0'.
3: This register is not available in 64-pin devices.
4: This register is not available in 64 and 80-pin devices.

6.3.5 ETHERNET SFRs

In addition to the standard SFR set in Bank 15, members of the PIC18F97J60 family have a second set of SFRs. This group, associated exclusively with the Ethernet module, occupies the top half of Bank 14 (E80h to EFFh).

Note: To improve performance, frequently accessed Ethernet registers are located in the standard SFR bank (F60h through FFFh).

A complete list of Ethernet SFRs is given in Table 6-4. All SFRs are fully described in Table 6-5.

TABLE 6-4: ETHERNET SFR MAP FOR PIC18F97J60 FAMILY DEVICES

Address	Name	Address	Name	Address	Name	Address	Name
EFFh	—(1)	EDFh	—(1)	EBFh	—(1)	E9Fh	—(1)
EFEh	ECON2	EDEh	—(1)	EBEh	—(1)	E9Eh	—(1)
EFDh	ESTAT	EDDh	—(1)	EBDh	—(1)	E9Dh	—(1)
EFCh	—(1)	EDCh	—(1)	EBCh	—(1)	E9Ch	—(1)
EFBh	EIE	EDBh	—(1)	EBBh	—(1)	E9Bh	—(1)
EFAh	—(1)	EDAh	—(1)	EBAh	—(1)	E9Ah	—(1)
EF9h	—(2)	ED9h	EPKTCNT	EB9h	MIRDH	E99h	EPAUSH
EF8h	—(2)	ED8h	ERXFCON	EB8h	MIRDL	E98h	EPAUSL
EF7h	EDMACSH	ED7h	—(1)	EB7h	MIWRH	E97h	EFLOCON
EF6h	EDMACSL	ED6h	—(1)	EB6h	MIWRL	E96h	—(2)
EF5h	EDMADSTH	ED5h	EPMOH	EB5h	—(1)	E95h	—(2)
EF4h	EDMADSTL	ED4h	EPMOL	EB4h	MIREGADR	E94h	—(2)
EF3h	EDMANDH	ED3h	—(2)	EB3h	—(2)	E93h	—(2)
EF2h	EDMANDL	ED2h	—(2)	EB2h	MICMD	E92h	—(2)
EF1h	EDMASTH	ED1h	EPMCSH	EB1h	—(1)	E91h	—(2)
EF0h	EDMASTL	ED0h	EPMCSL	EB0h	—(1)	E90h	—(2)
EEFh	ERXWRPTH	ECFh	EPMM7	EAFh	—(2)	E8Fh	—(2)
EEEh	ERXWRPTL	ECEh	EPMM6	EAEh	—(1)	E8Eh	—(2)
EEDh	ERXRDPTH	ECDh	EPMM5	EADh	—(1)	E8Dh	—(2)
EECh	ERXRDPTL	ECCh	EPMM4	EACh	—(1)	E8Ch	—(2)
EEBh	ERXNDH	ECBh	EPMM3	EABh	MAMXFLH	E8Bh	—(2)
EEAh	ERXNDL	ECAh	EPMM2	EAAh	MAMXFLL	E8Ah	MISTAT
EE9h	ERXSTH	EC9h	EPMM1	EA9h	—(1)	E89h	—(1)
EE8h	ERXSTL	EC8h	EPMM0	EA8h	—(1)	E88h	—(1)
EE7h	ETXNDH	EC7h	EHT7	EA7h	MAIPGH	E87h	—(1)
EE6h	ETXNDL	EC6h	EHT6	EA6h	MAIPGL	E86h	—(1)
EE5h	ETXSTH	EC5h	EHT5	EA5h	—(2)	E85h	MAADR2
EE4h	ETXSTL	EC4h	EHT4	EA4h	MABBIPG	E84h	MAADR1
EE3h	EWRPTH	EC3h	EHT3	EA3h	MACON4	E83h	MAADR4
EE2h	EWRPTL	EC2h	EHT2	EA2h	MACON3	E82h	MAADR3
EE1h	—(1)	EC1h	EHT1	EA1h	—(1)	E81h	MAADR6
EE0h	—(1)	EC0h	EHT0	EA0h	MACON1	E80h	MAADR5

Note 1: Reserved register location; do not modify.
2: Unimplemented registers are read as '0'.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
TOSU	—	—	—	Top-of-Stack Register Upper Byte (TOS<20:16>)					---0 0000	69, 81
TOSH	Top-of-Stack Register High Byte (TOS<15:8>)								0000 0000	69, 81
TOSL	Top-of-Stack Register Low Byte (TOS<7:0>)								0000 0000	69, 81
STKPTR	STKFUL(1)	STKUNF(1)	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	69, 82
PCLATU	—	—	bit 21(2)	Holding Register for PC<20:16>					---0 0000	69, 81
PCLATH	Holding Register for PC<15:8>								0000 0000	69, 81
PCL	PC Low Byte (PC<7:0>)								0000 0000	69, 81
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	69, 108
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	69, 108
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	69, 108
TABLAT	Program Memory Table Latch								0000 0000	69, 108
PRODH	Product Register High Byte								xxxx xxxx	69, 127
PRODL	Product Register Low Byte								xxxx xxxx	69, 127
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	69, 131
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	69, 132
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	69, 133
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A	69, 99
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A	69, 100
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A	69, 100
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A	69, 100
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	69, 100
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- xxxx	69, 99
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	69, 100
WREG	Working Register								xxxx xxxx	69
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A	69, 99
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A	69, 100
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A	69, 100
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A	69, 100
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	69, 100
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- xxxx	69, 99
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	69, 99
BSR	—	—	—	—	Bank Select Register				---- 0000	69, 99
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A	69, 99
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A	69, 100
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A	69, 100
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A	69, 100
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	69, 100
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- xxxx	69, 99
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	69, 99

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	70, 97
TMR0H	Timer0 Register High Byte								0000 0000	70, 171
TMR0L	Timer0 Register Low Byte								xxxx xxxx	70, 171
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	70, 171
OSCCON	IDLEN	—	—	—	OSTS(3)	—	SCS1	SCS0	0--- q-00	70, 53
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	0000 00--	70, 227
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	70, 368
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0-q1 1100	70, 64, 143
TMR1H	Timer1 Register High Byte								xxxx xxxx	70, 175
TMR1L	Timer1 Register Low Byte								xxxx xxxx	70, 175
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	70, 175
TMR2	Timer2 Register								0000 0000	70, 180
PR2	Timer2 Period Register								1111 1111	70, 180
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	70, 180
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								xxxx xxxx	70, 279
SSP1ADD	MSSP1 Address Register (I^2C™ Slave mode), MSSP1 Baud Rate Reload Register (I^2C Master mode)								0000 0000	70, 279
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	70, 270, 280
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	70, 271, 281
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	70, 282
	GCEN	ACKSTAT	ADMSK5(4)	ADMSK4(4)	ADMSK3(4)	ADMSK2(4)	ADMSK1(4)	SEN		
ADRESH	A/D Result Register High Byte								xxxx xxxx	70, 347
ADRESL	A/D Result Register Low Byte								xxxx xxxx	70, 347
ADCON0	ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	0-00 0000	70, 339
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	70, 340
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	70, 341
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	70, 193
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	70, 193
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	70, 198
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	70, 193
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	70, 193
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000	70, 198
CCPR3H	Capture/Compare/PWM Register 3 High Byte								xxxx xxxx	70, 193
CCPR3L	Capture/Compare/PWM Register 3 Low Byte								xxxx xxxx	70, 193
CCP3CON	P3M1	P3M0	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000 0000	70, 198
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	0000 0000	70, 212
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	70, 355
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	70, 349
TMR3H	Timer3 Register High Byte								xxxx xxxx	70, 183
TMR3L	Timer3 Register Low Byte								xxxx xxxx	70, 183

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	71, 183
PSPCON(5)	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	71, 169
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte								0000 0000	71, 320
RCREG1	EUSART1 Receive Register								0000 0000	71, 327
TXREG1	EUSART1 Transmit Register								xxxx xxxx	71, 329
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	71, 320
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	71, 320
EECON2	Program Memory Control Register (not a physical register)								---- ----	71, 106
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	---0 x00-	71, 107
IPR3	SSP2IP(5)	BCL2IP(5)	RC2IP(6)	TX2IP(6)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	1111 1111	71, 142
PIR3	SSP2IF(5)	BCL2IF(5)	RC2IF(6)	TX2IF(6)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	0000 0000	71, 136
PIE3	SSP2IE(5)	BCL2IE(5)	RC2IE(6)	TX2IE(6)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	0000 0000	71, 139
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	1111 1-11	71, 141
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	0000 0-00	71, 135
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	0000 0-00	71, 138
IPR1	PSPIP(9)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	1111 1111	71, 140
PIR1	PSPIF(9)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	71, 134
PIE1	PSPIE(9)	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	0000 0000	71, 137
MEMCON(5,7)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	71, 116
OSCTUNE	PPST1	PLLEN(8)	PPST0	PPRE	—	—	—	—	0000 ----	71, 51
TRISJ(6)	TRISJ7(5)	TRISJ6(5)	TRISJ5(6)	TRISJ4(6)	TRISJ3(5)	TRISJ2(5)	TRISJ1(5)	TRISJ0(5)	1111 1111	71, 167
TRISH(6)	TRISH7(6)	TRISH6(6)	TRISH5(6)	TRISH4(6)	TRISH3(6)	TRISH2(6)	TRISH1(6)	TRISH0(6)	1111 1111	71, 165
TRISG	TRISG7(5)	TRISG6(5)	TRISG5(5)	TRISG4	TRISG3(6)	TRISG2(6)	TRISG1(6)	TRISG0(6)	1111 1111	71, 163
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0(5)	1111 1111	71, 161
TRISE	TRISE7(6)	TRISE6(6)	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	1111 1111	71, 159
TRISD	TRISD7(5)	TRISD6(5)	TRISD5(5)	TRISD4(5)	TRISD3(5)	TRISD2	TRISD1	TRISD0	1111 1111	71, 156
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	71, 153
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	71, 150
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	71, 147
LATJ(6)	LATJ7(5)	LATJ6(5)	LATJ5(6)	LATJ4(6)	LATJ3(5)	LATJ2(5)	LATJ1(5)	LATJ0(5)	xxxx xxxx	71, 167
LATH(6)	LATH7(6)	LATH6(6)	LATH5(6)	LATH4(6)	LATH3(6)	LATH2(6)	LATH1(6)	LATH0(6)	xxxx xxxx	71, 165
LATG	LATG7(5)	LATG6(5)	LATG5(5)	LATG4	LATG3(6)	LATG2(6)	LATG1(6)	LATG0(6)	xxxx xxxx	72, 163
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0(5)	xxxx xxxx	72, 161
LATE	LATE7(6)	LATE6(6)	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx xxxx	72, 159
LATD	LATD7(5)	LATD6(5)	LATD5(5)	LATD4(5)	LATD3(5)	LATD2	LATD1	LATD0	xxxx xxxx	72, 156
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	72, 153
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	72, 150
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	00xx xxxx	72, 147
PORTJ(6)	RJ7(5)	RJ6(5)	RJ5(6)	RJ4(6)	RJ3(5)	RJ2(5)	RJ1(5)	RJ0(5)	xxxx xxxx	72, 167
PORTH(6)	RH7(6)	RH6(6)	RH5(6)	RH4(6)	RH3(6)	RH2(6)	RH1(6)	RH0(6)	0000 xxxx	72, 165
PORTG	RG7(5)	RG6(5)	RG5(5)	RG4	RG3(6)	RG2(6)	RG1(6)	RG0(6)	111x xxxx	72, 163

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0(5)	0000 0000	72, 161
PORTE	RE7(6)	RE6(6)	RE5	RE4	RE3	RE2	RE1	RE0	xxxx xxxx	72, 159
PORTD	RD7(5)	RD6(5)	RD5(5)	RD4(5)	RD3(5)	RD2	RD1	RD0	xxxx xxxx	72, 156
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	72, 153
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	72, 150
PORTA	RJPU(6)	—	RA5	RA4	RA3	RA2	RA1	RA0	0-0x 0000	72, 147
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte								0000 0000	72, 320
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	72, 318
SPBRGH2	EUSART2 Baud Rate Generator Register High Byte								0000 0000	72, 320
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00	72, 318
ERDPTH	—	—	—	Buffer Read Pointer High Byte					---0 0101	72, 223
ERDPTL	Buffer Read Pointer Low Byte								1111 1010	72, 223
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000	72, 211
TMR4	Timer4 Register								0000 0000	72, 187
PR4	Timer4 Period Register								1111 1111	72, 187
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	72, 187
CCPR4H	Capture/Compare/PWM Register 4 High Byte								xxxx xxxx	72, 193
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								xxxx xxxx	72, 193
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	--00 0000	73, 189
CCPR5H	Capture/Compare/PWM Register 5 High Byte								xxxx xxxx	73, 193
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								xxxx xxxx	73, 193
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	--00 0000	73, 189
SPBRG2	EUSART2 Baud Rate Generator Register Low Byte								0000 0000	73, 320
RCREG2	EUSART2 Receive Register								0000 0000	73, 327
TXREG2	EUSART2 Transmit Register								0000 0000	73, 329
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	73, 316
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	73, 317
ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0	0000 0000	73, 212
ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0	0000 0000	73, 211
ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000	73, 212
ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000	73, 211
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								xxxx xxxx	73, 279
SSP2ADD	MSSP2 Address Register (I2C™ Slave mode), MSSP2 Baud Rate Reload Register (I2C Master mode)								0000 0000	73, 279
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	73, 270
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	73, 271, 281
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	73, 282
	GCEN	ACKSTAT	ADMSK5(4)	ADMSK4(4)	ADMSK3(4)	ADMSK2(4)	ADMSK1(4)	SEN		
EDATA	Ethernet Transmit/Receive Buffer Register (EDATA<7:0>)								xxxx xxxx	73, 223
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	-000 0-00	73, 241
ECON2	AUTOINC	PKTDEC	ETHEN	—	—	—	—	—	100- ----	73, 228

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
ESTAT	—	BUFER	—	r	—	RXBUSY	TXABRT	PHYRDY	-0-0 -000	73, 228
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	-000 0-00	73, 240
EDMACSH	DMA Checksum Register High Byte								0000 0000	73, 265
EDMACSL	DMA Checksum Register Low Byte								0000 0000	73, 265
EDMADSTH	—	—	—	DMA Destination Register High Byte					---0 0000	73, 265
EDMADSTL	DMA Destination Register Low Byte								0000 0000	73, 265
EDMANDH	—	—	—	DMA End Register High Byte					---0 0000	73, 265
EDMANDL	DMA End Register Low Byte								0000 0000	73, 265
EDMASTH	—	—	—	DMA Start Register High Byte					---0 0000	73, 265
EDMASTL	DMA Start Register Low Byte								0000 0000	73, 265
ERXWRPTH	—	—	—	Receive Buffer Write Pointer High Byte					---0 0000	73, 225
ERXWRPTL	Receive Buffer Write Pointer Low Byte								0000 0000	73, 225
ERXRDPTH	—	—	—	Receive Buffer Read Pointer High Byte					---0 0101	73, 225
ERXRDPTL	Receive Buffer Read Pointer Low Byte								1111 1010	73, 225
ERXNDH	—	—	—	Receive End Register High Byte					---1 1111	73, 225
ERXNDL	Receive End Register Low Byte								1111 1111	73, 225
ERXSTH	—	—	—	Receive Start Register High Byte					---0 0101	73, 225
ERXSTL	Receive Start Register Low Byte								1111 1010	73, 225
ETXNDH	—	—	—	Transmit End Register High Byte					---0 0000	74, 226
ETXNDL	Transmit End Register Low Byte								0000 0000	74, 226
ETXSTH	—	—	—	Transmit Start Register High Byte					---0 0000	74, 226
ETXSTL	Transmit Start Register Low Byte								0000 0000	74, 226
EWRPTH	—	—	—	Buffer Write Pointer High Byte					---0 0000	74, 223
EWRPTL	Buffer Write Pointer Low Byte								0000 0000	74, 223
EPKTCNT	Ethernet Packet Count Register								0000 0000	74, 252
ERXFCON	UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN	1010 0001	74, 260
EPMOH	—	—	—	Pattern Match Offset Register High Byte					---0 0000	74, 263
EPMOL	Pattern Match Offset Register Low Byte								0000 0000	74, 263
EPMCSH	Pattern Match Checksum Register High Byte								0000 0000	74, 263
EPMCSL	Pattern Match Checksum Register Low Byte								0000 0000	74, 263
EPMM7	Pattern Match Mask Register Byte 7								0000 0000	74, 263
EPMM6	Pattern Match Mask Register Byte 6								0000 0000	74, 263
EPMM5	Pattern Match Mask Register Byte 5								0000 0000	74, 263
EPMM4	Pattern Match Mask Register Byte 4								0000 0000	74, 263
EPMM3	Pattern Match Mask Register Byte 3								0000 0000	74, 263
EPMM2	Pattern Match Mask Register Byte 2								0000 0000	74, 263
EPMM1	Pattern Match Mask Register Byte 1								0000 0000	74, 263
EPMM0	Pattern Match Mask Register Byte 0								0000 0000	74, 263

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

TABLE 6-5: REGISTER FILE SUMMARY (PIC18F97J60 FAMILY) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Values on POR, BOR	Details on Page:
EHT7	Hash Table Register Byte 7								0000 0000	74, 259
EHT6	Hash Table Register Byte 6								0000 0000	74, 259
EHT5	Hash Table Register Byte 5								0000 0000	74, 259
EHT4	Hash Table Register Byte 4								0000 0000	74, 259
EHT3	Hash Table Register Byte 3								0000 0000	74, 259
EHT2	Hash Table Register Byte 2								0000 0000	74, 259
EHT1	Hash Table Register Byte 1								0000 0000	74, 259
EHT0	Hash Table Register Byte 0								0000 0000	74, 259
MIRDH	MII Read Data Register High Byte								0000 0000	74, 232
MIRDL	MII Read Data Register Low Byte								0000 0000	74, 232
MIWRH	MII Write Data Register High Byte								0000 0000	74, 232
MIWRL	MII Write Data Register Low Byte								0000 0000	74, 232
MIREGADR	—	—	—	MII Address Register					---0 0000	74, 232
MICMD	—	—	—	—	—	—	MIISCAN	MIIRD	---- --00	74, 231
MAMXFLH	Maximum Frame Length Register High Byte								0000 0110	74, 245
MAMXFLL	Maximum Frame Length Register Low Byte								0000 0000	74, 245
MAIPGH	—	MAC Non Back-to-Back Inter-Packet Gap Register High Byte							-000 0000	75, 245
MAIPGL	—	MAC Non Back-to-Back Inter-Packet Gap Register Low Byte							-000 0000	75, 245
MABBIPG	—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0	-000 0000	75, 246
MACON4	—	DEFER	r	r	—	—	r	r	-000 --00	75, 231
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	0000 0000	75, 230
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	---0 0000	75, 229
EPAUSH	Pause Timer Value Register High Byte								0001 0000	75, 258
EPAUSL	Pause Timer Value Register Low Byte								0000 0000	75, 258
EFLOCON	—	—	—	—	—	r	FCEN1	FCEN0	---- -000	75, 258
MISTAT	—	—	—	—	r	NVALID	SCAN	BUSY	---- 0000	75, 232
MAADR2	MAC Address Register Byte 2 (MAADR<39:32>), OUI Byte 2								0000 0000	75, 245
MAADR1	MAC Address Register Byte 1 (MAADR<47:40>), OUI Byte 1								0000 0000	75, 245
MAADR4	MAC Address Register Byte 4 (MAADR<23:16>)								0000 0000	75, 245
MAADR3	MAC Address Register Byte 3 (MAADR<31:24>), OUI Byte 3								0000 0000	75, 245
MAADR6	MAC Address Register Byte 6 (MAADR<7:0>)								0000 0000	75, 245
MAADR5	MAC Address Register Byte 5 (MAADR<15:8>)								0000 0000	75, 245

Legend: x = unknown; u = unchanged; - = unimplemented, read as '0'; q = value depends on condition; r = reserved bit, do not modify. Shaded cells are unimplemented, read as '0'.

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

2: Bit 21 of the PC is only available in Serial Programming modes.

3: Reset value is '0' when Two-Speed Start-up is enabled and '1' if disabled.

4: Alternate names and definitions for these bits when the MSSP module is operating in I^2C™ Slave mode.

5: These bits and/or registers are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'. Reset values shown apply only to 100-pin devices.

6: These bits and/or registers are only available in 80-pin and 100-pin devices. In 64-pin devices, they are unimplemented and read as '0'. Reset values are shown for 100-pin devices.

7: In Microcontroller mode, the bits in this register are unwritable and read as '0'.

8: PLLEN is only available when either ECPLL or HSPLL Oscillator mode is selected; otherwise, read as '0'.

9: Implemented in 100-pin devices in Microcontroller mode only.

6.3.6 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 26-2 and Table 26-3.

Note: The C and DC bits operate as a $\overline{\text{Borrow}}$ and Digit $\overline{\text{Borrow}}$ 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 of the result) to change state.
1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is non-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 mode 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 mode 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

```
        LFSRFSR0, 100h;
NEXT    CLRFPOSTINC0; Clear INDF
                    ; register then
                    ; inc pointer
        BTFSSFSR0H, 1; All done with
                    ; Bank1?
        BRA NEXT    ; NO, clear next
CONTINUE            ; YES, continue
```

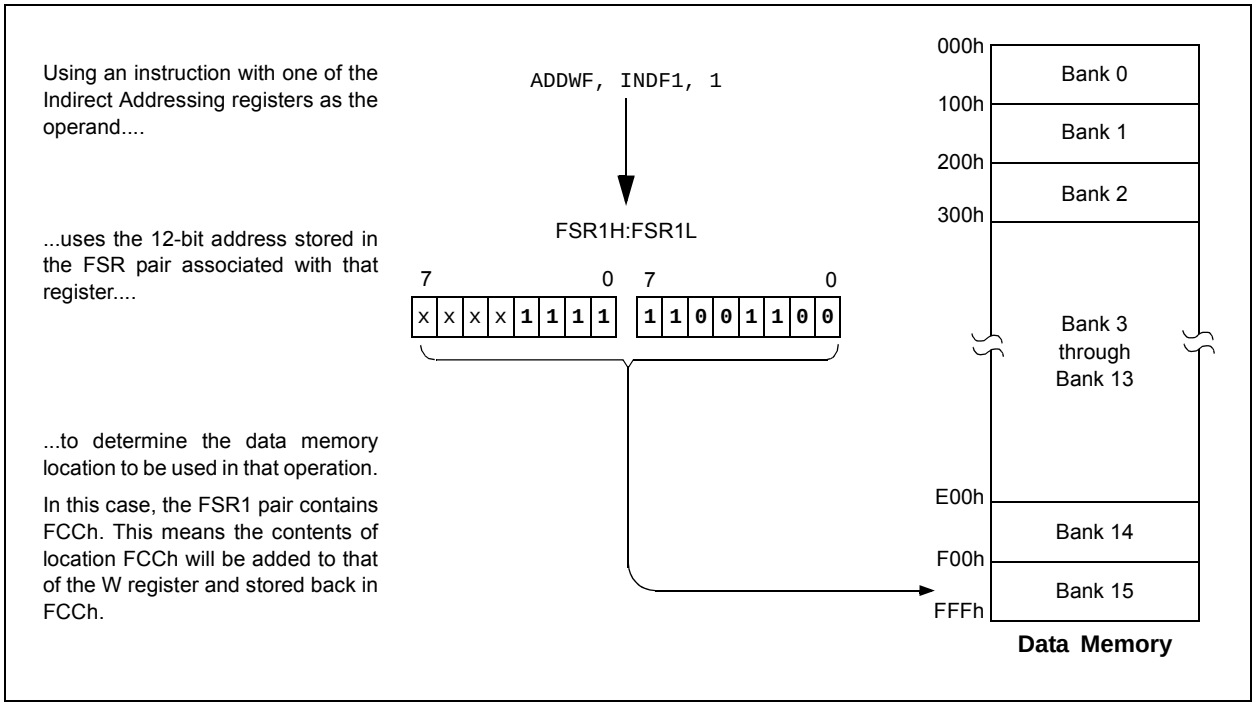
6.4.3.1 FSR Registers and the INDF Operand

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 Access RAM bit have no effect on determining the target address.

FIGURE 6-9: 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 -128 to 127) 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 operation. As a specific case, assume that the FSR0H:FSR0L pair contains FE7h, the address of INDF1. Attempts to read the value of the INDF1, using INDF0 as an operand, will return 00h. Attempts to write to INDF1, using INDF0 as the operand, will result in a NOP.

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

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 half of the standard PIC18 instruction set. Instructions that only use Inherent or Literal Addressing modes are unaffected.

Additionally, byte-oriented and bit-oriented instructions are not affected if they use the Access Bank (Access RAM bit is ‘1’) or include a file address of 60h or above. Instructions meeting these criteria will continue to execute as before. A comparison of the different possible addressing modes when the extended instruction set is enabled is shown in Figure 6-10.

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 26.2.1 “Extended Instruction Syntax”**.

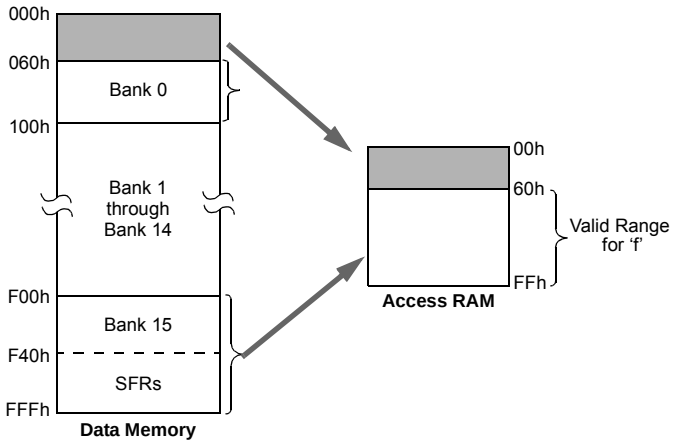
FIGURE 6-10: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)

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

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.

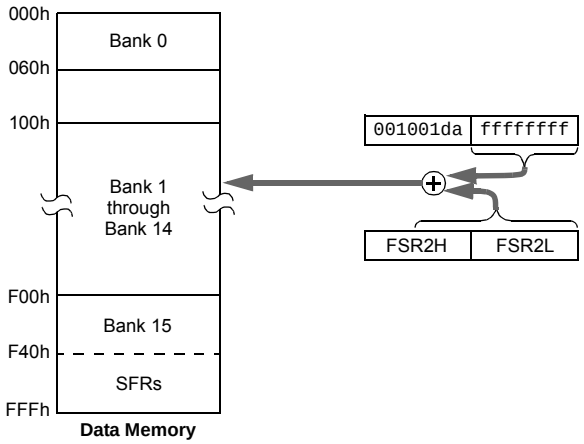
Locations below 060h are not available in this addressing mode.



When a = 0 and f ≤ 5Fh:

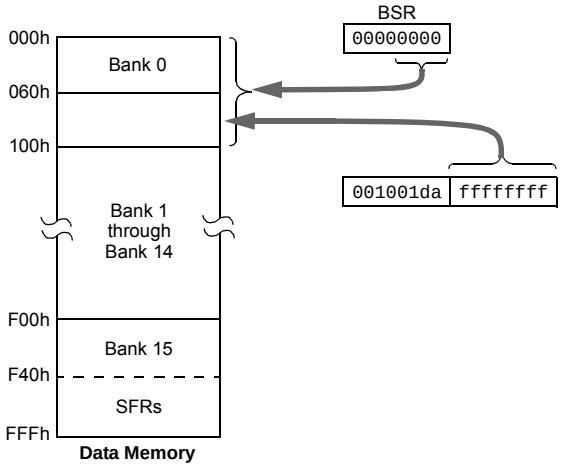
The instruction executes in Indexed Literal Offset mode. 'f' is interpreted as an offset to the address value in FSR2. The two are added together to obtain the address of the target register for the instruction. The address can be anywhere in the data memory space.

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

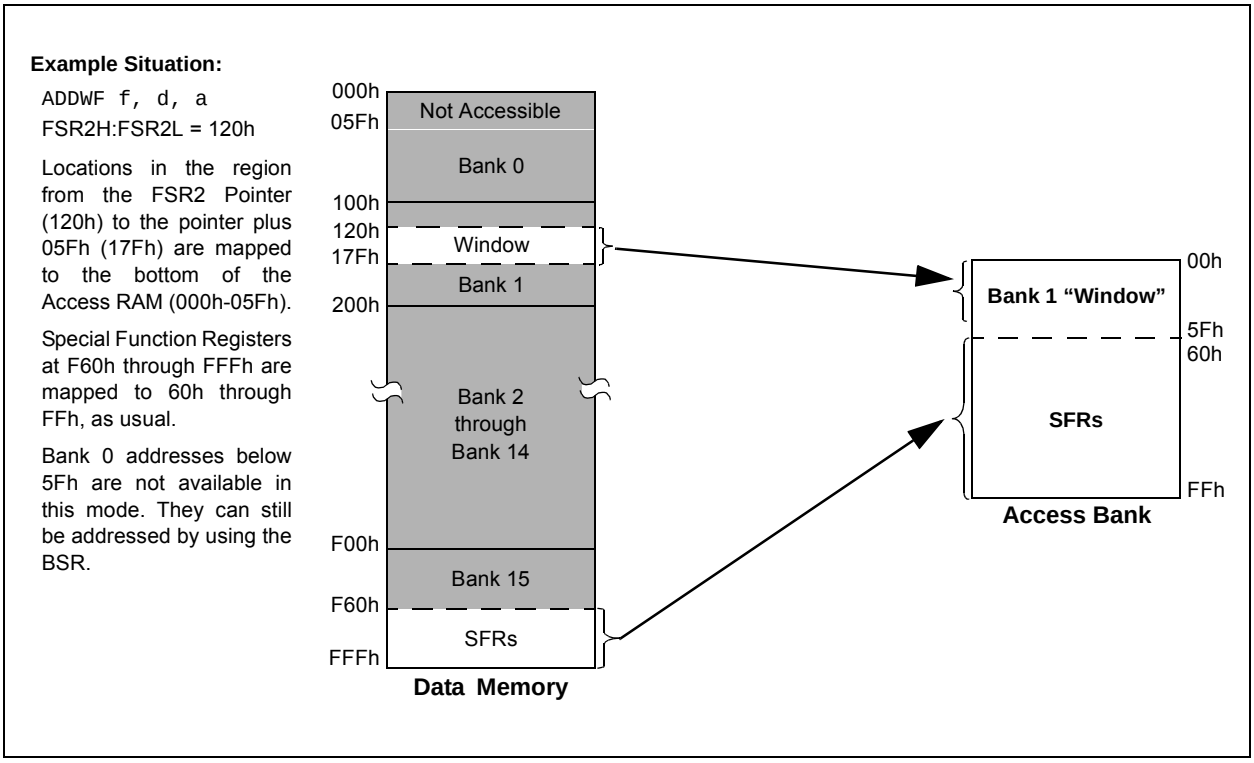
The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined "window" that can be located anywhere in the 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 6-11.

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

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 6-11: 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)

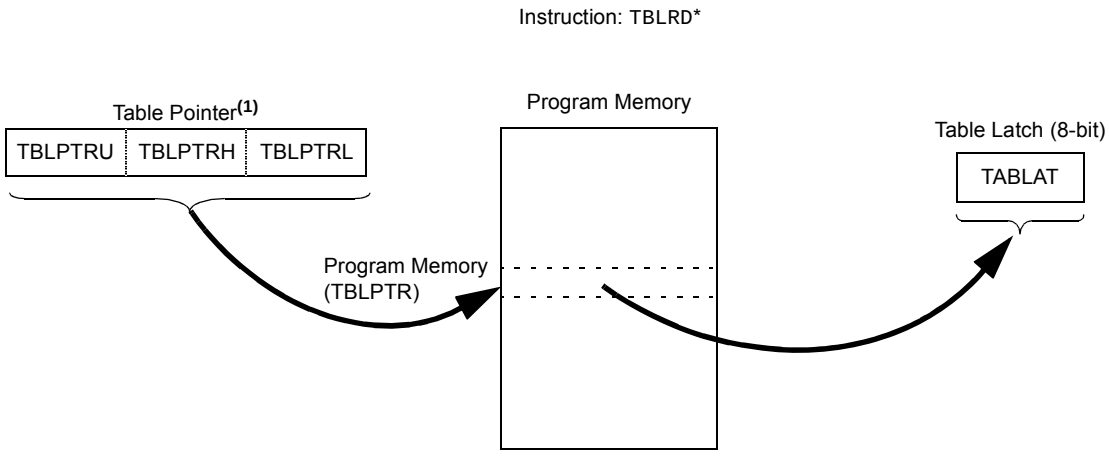
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

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.

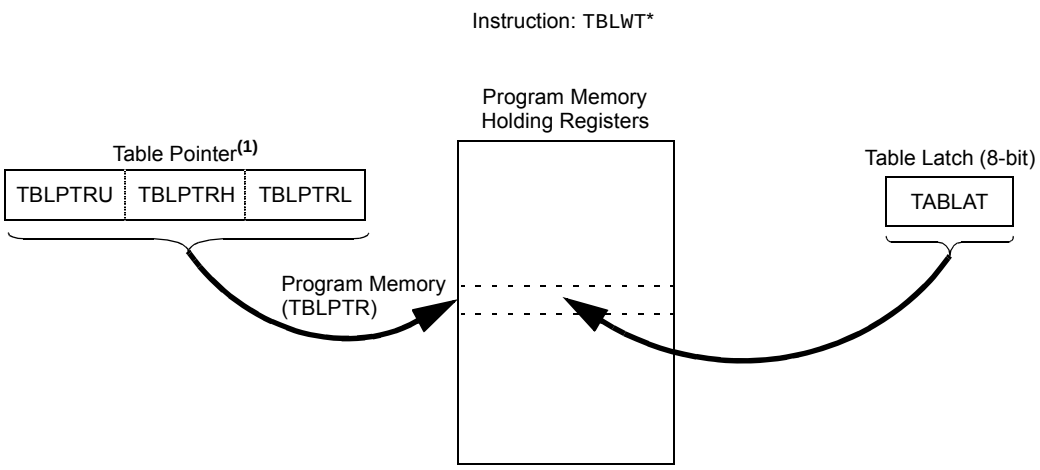
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions 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:	S = Settable 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 7-5 **Unimplemented:** Read as ‘0’

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

bit 3 **WRERR:** Flash Program 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 the write is complete. The WR bit can only be set (not cleared) in software.)
- 0 = Write cycle 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 and 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. The table 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 six LSbs of the Table Pointer register (TBLPTR<5:0>) determine which of the 64 program memory holding registers is written to. When the timed write to program memory begins (via the WR bit), the 15 MSbs of the TBLPTR (TBLPTR<20:6>) determine which program memory block of 64 bytes is written to. For more detail, see **Section 7.5 "Writing to Flash Program Memory"**.

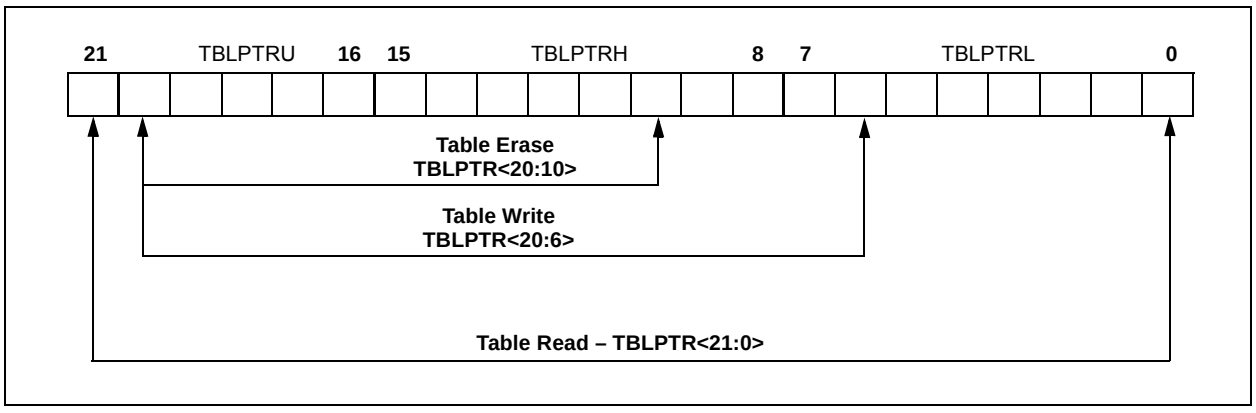
When an erase of program memory is executed, the 11 MSbs of the Table Pointer register (TBLPTR<20:10>) point to the 1024-byte block that will be erased. The Least Significant bits (TBLPTR<9:0>) are ignored.

Figure 7-3 describes the relevant boundaries of 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



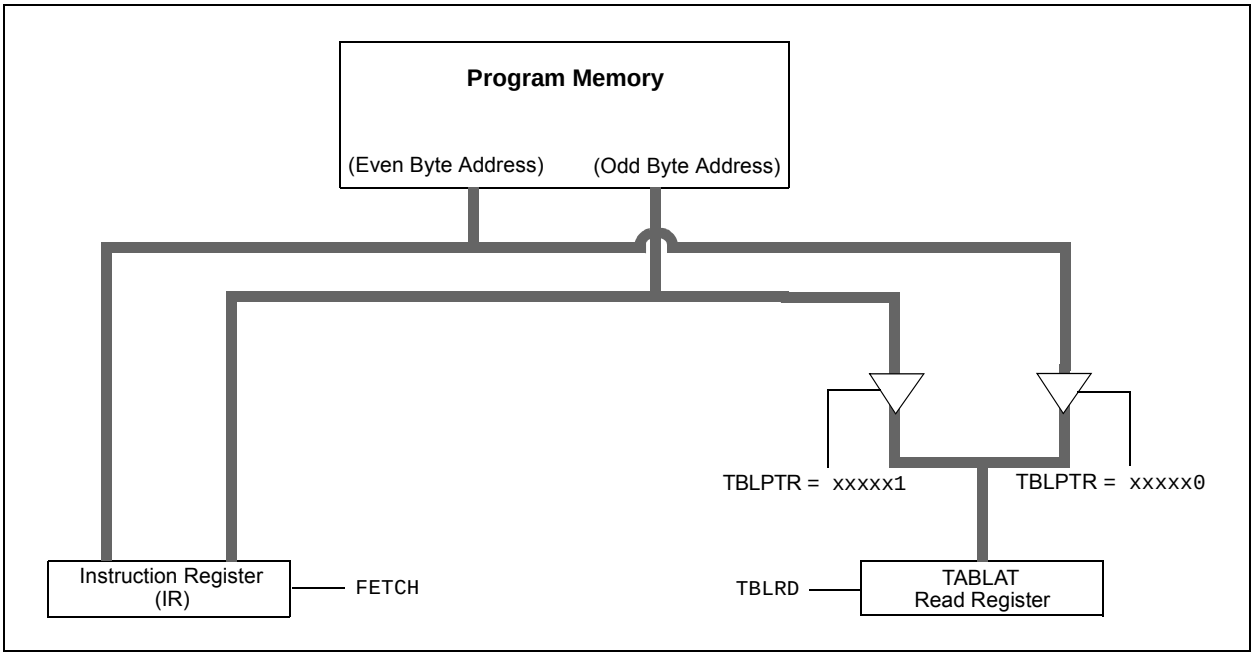
7.3 Reading the Flash Program Memory

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
            MOVFW   TABLAT, W             ; get data
            MOVF    WORD_ODD
```

7.4 Erasing Flash Program Memory

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

When initiating an erase sequence from the microcontroller itself, a block of 1024 bytes of program memory is erased. The Most Significant 11 bits of the TBLPTR<20:10> 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. An on-chip timer controls the erase time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over most of the voltage range of the device. See Parameter D132B (V_{PEW}) for specific limits.

7.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

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

1. Load Table Pointer register with the address of row being erased.
2. Set the EECON1 register for the erase operation:
   - set WREN bit to enable writes;
   - set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit. This will begin the Row Erase cycle.
7. The CPU will stall for the duration of the erase.
8. Re-enable interrupts.

EXAMPLE 7-2: ERASING A FLASH PROGRAM MEMORY ROW

```
                MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
    ERASE_ROW
                BSF     EECON1, WREN            ; enable write to memory
                BSF     EECON1, FREE            ; enable Row Erase operation
                BCF     INTCON, GIE             ; disable interrupts
Required        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.

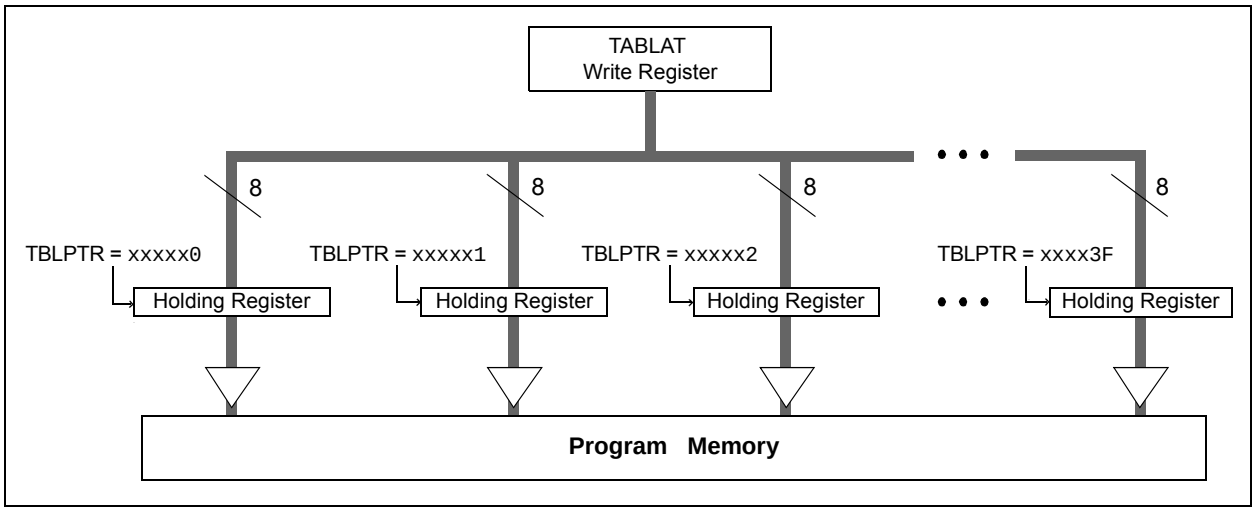
The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

An on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over most of the voltage range of the device. See Parameter D132B (VPEW) for specific limits.

Note 1: Unlike previous PIC MCU devices, members of the PIC18F97J60 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 Row Erase of the target row, 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. If the section of program memory to be written to has been programmed previously, then the memory will need to be erased before the write occurs (see **Section 7.4.1 "Flash Program Memory Erase Sequence"**).
2. Write the 64 bytes into the holding registers with auto-increment.
3. Set the WREN bit to enable byte writes.
4. Disable interrupts.
5. Write 55h to EECON2.
6. Write AAh to EECON2.
7. Set the WR bit. This will begin the write cycle.
8. The CPU will stall for the duration of the write.
9. Re-enable interrupts.
10. 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
                    MOVWF   TBLPTRU            ; address of the memory block
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
ERASE_BLOCK
                    BSF     EECON1, WREN       ; enable write to memory
                    BSF     EECON1, FREE       ; enable Row Erase operation
                    BCF     INTCON, GIE        ; disable interrupts
                    MOVLW   55h
                    MOVWF   EECON2             ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2             ; write 0AAh
                    BSF     EECON1, WR         ; start erase (CPU stall)
                    BSF     INTCON, GIE        ; re-enable interrupts
                    MOVLW   D'16'
                    MOVWF   WRITE_COUNTER      ; Need to write 16 blocks of 64 to write
                                               ; one erase block of 1024
RESTART_BUFFER
                    MOVLW   D'64'
                    MOVWF   COUNTER
                    MOVLW   BUFFER_ADDR_HIGH   ; point to buffer
                    MOVWF   FSR0H
                    MOVLW   BUFFER_ADDR_LOW
                    MOVWF   FSR0L
FILL_BUFFER
                    ...                        ; read the new data from I2C, SPI,
                                               ; PSP, USART, etc.
WRITE_BUFFER
                    MOVLW   D’64               ; number of bytes in holding register
                    MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                    MOVFF   POSTINC0, WREG     ; get low byte of buffer data
                    MOVWF   TABLAT             ; present data to table latch
                    TBLWT+*                    ; write data, perform a short write
                                               ; to internal TBLWT holding register.
                    DECFSZ  COUNTER            ; loop until buffers are full
                    BRA     WRITE_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 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.

7.5.4 PROTECTION AGAINST SPURIOUS WRITES

To protect against spurious writes to Flash program memory, the write initiate sequence must also be followed. See **Section 25.0 "Special Features of the CPU"** for more details.

7.6 Flash Program Operation During Code Protection

See **Section 25.6 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

TABLE 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>)					69
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								69
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								69
TABLAT	Program Memory Table Latch								69
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
EECON2	EEPROM Control Register 2 (not a physical register)								71
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	71

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

NOTES:

8.0 EXTERNAL MEMORY BUS

Note: The external memory bus is not implemented on 64-pin and 80-pin devices.

The External Memory Bus (EMB) 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: PIC18F96J60/96J65/97J60 EXTERNAL MEMORY 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 bit
RJ1/$\overline{OE}$	PORTJ	1	Output Enable ($\overline{OE}$) Control bit
RJ2/$\overline{WRL}$	PORTJ	2	Write Low ($\overline{WRL}$) Control bit
RJ3/$\overline{WRH}$	PORTJ	3	Write High ($\overline{WRH}$) Control bit
RJ4/BA0	PORTJ	4	Byte Address (BA0) Bit 0
RJ5/$\overline{CE}$	PORTJ	5	Chip Enable ($\overline{CE}$) Control bit
RJ6/$\overline{LB}$	PORTJ	6	Lower Byte Enable ($\overline{LB}$) Control bit
RJ7/$\overline{UB}$	PORTJ	7	Upper Byte Enable ($\overline{UB}$) Control bit

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 in 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 operating 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/Os.

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 operating modes are discussed in more detail in **Section 8.6 "16-Bit Data Width Modes"**. The WM 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 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 **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: $\overline{WRH}$ is active when TABLAT is written to and TBLPTR contains an odd address. When TBLPTR contains an even address, writing to TABLAT loads a holding latch with the value written.
- 01 = Byte Select mode: TABLAT data is copied on both MSB and LSB; $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate
- 00 = Byte Write mode: TABLAT data is copied on both MSB and LSB; $\overline{WRH}$ or $\overline{WRL}$ will activate

8.2 Address and Data Width

The PIC18F97J60 family of devices can be independently configured for different address and data widths on the same memory bus. Both address and data widths 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 operating 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 Addressing mode (EMB<1:0> = 01) disables A<19:16> and allows the PORTH<3:0> bits 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 in 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 Data Width 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 is 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 bits located at LATA<7:6> and PORTA<7>. 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.

In Extended Microcontroller mode, the port pull-ups can be useful in preserving the memory state on the external bus while the bus is temporarily disabled (EBDIS = 1).

8.5 Program Memory Modes and the External Memory Bus

The PIC18F97J60 family of devices is 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 the 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 in 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.

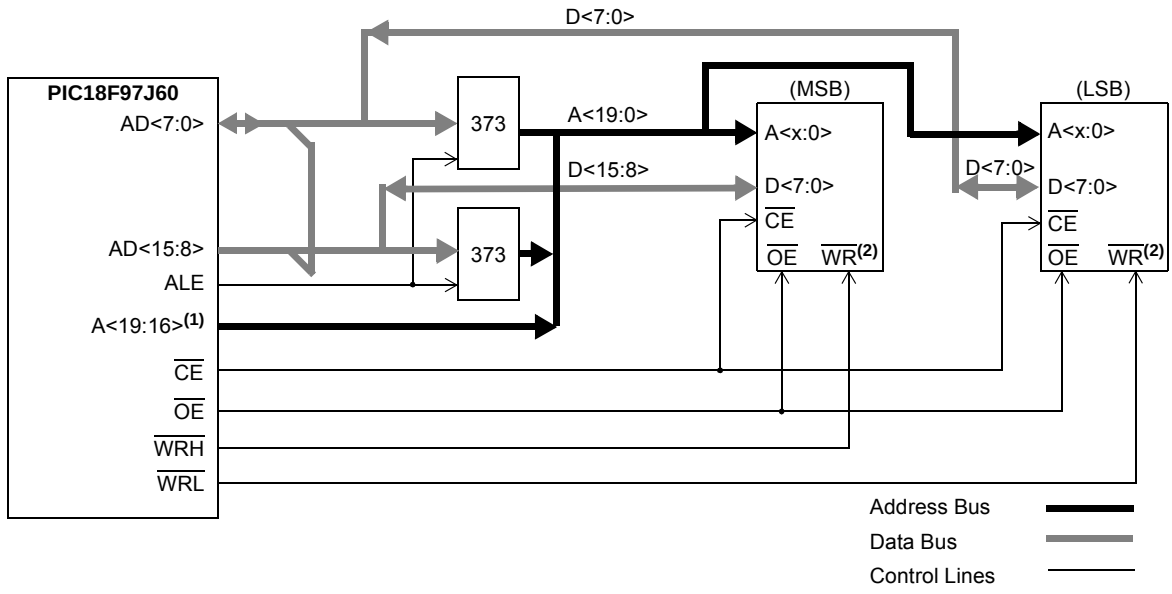
In Byte Select mode, JEDEC standard Flash memories will require BA0 for the byte address line and one I/O line to select between Byte and Word mode. The other 16-Bit Data Width modes do not need BA0. JEDEC standard, static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte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

8.6.1 16-BIT BYTE WRITE MODE

Figure 8-1 shows an example of 16-Bit Byte Write mode for PIC18F97J60 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

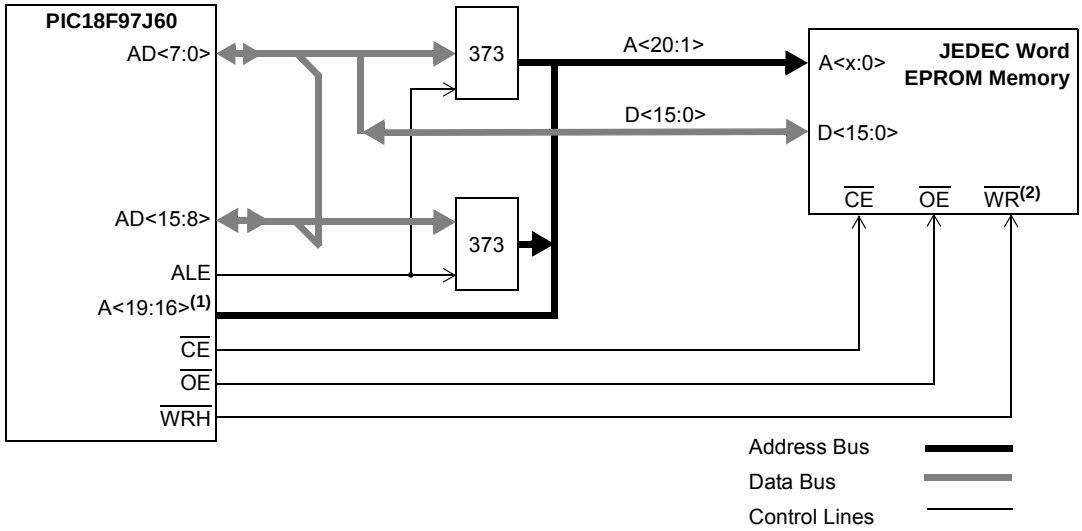
Figure 8-2 shows an example of 16-Bit Word Write mode for PIC18F97J60 family devices. This mode is used for word-wide memories, which include some of the EPROM and Flash type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory, and table writes to any type of word-wide 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"**.

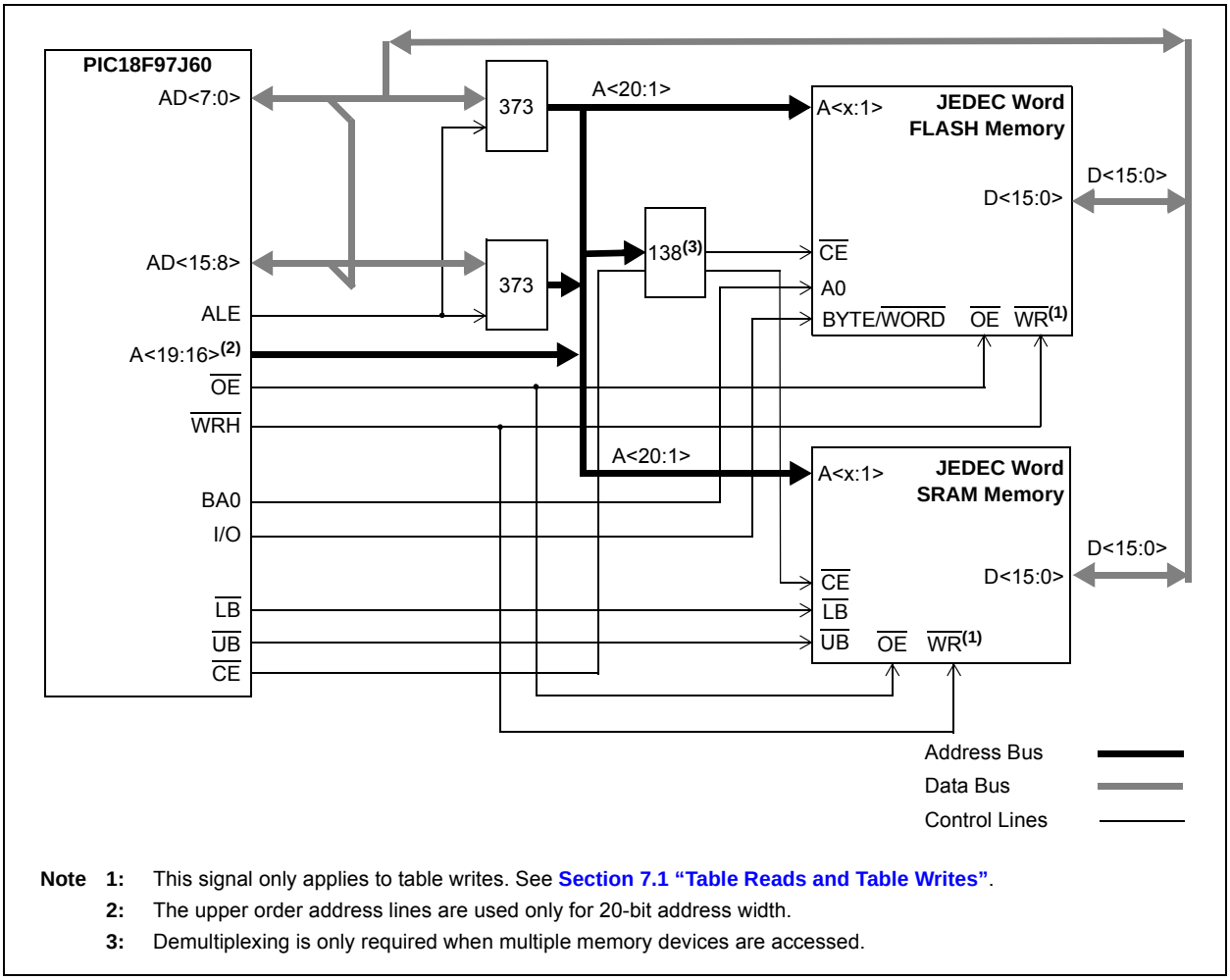
8.6.3 16-BIT BYTE SELECT MODE

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)

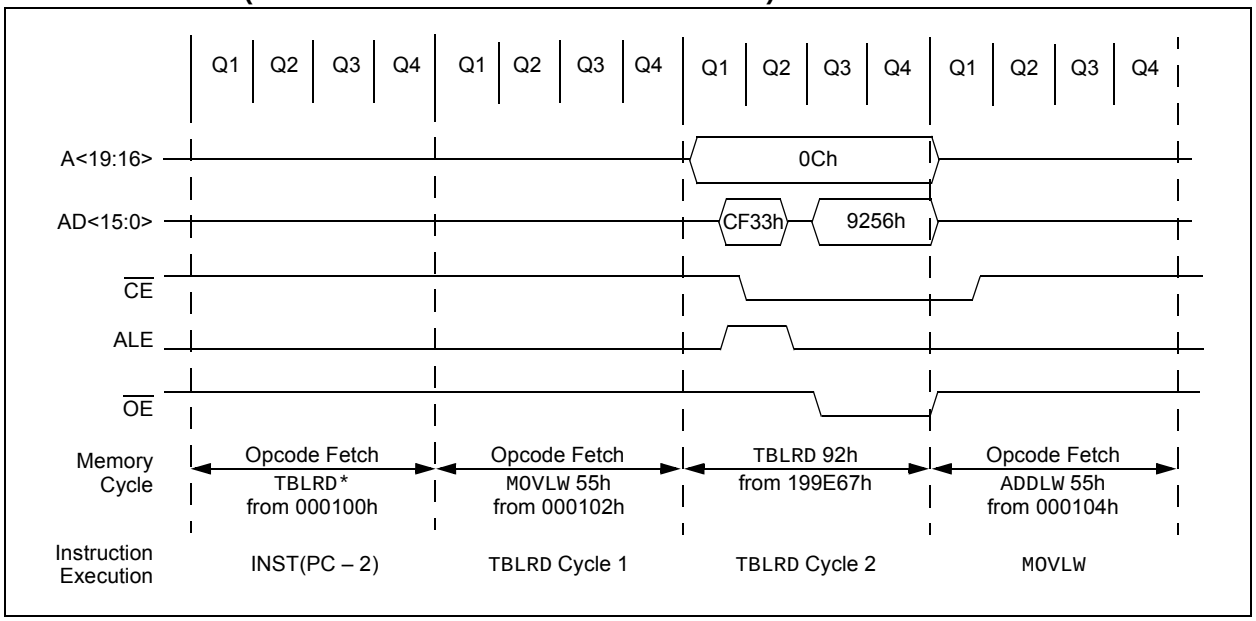
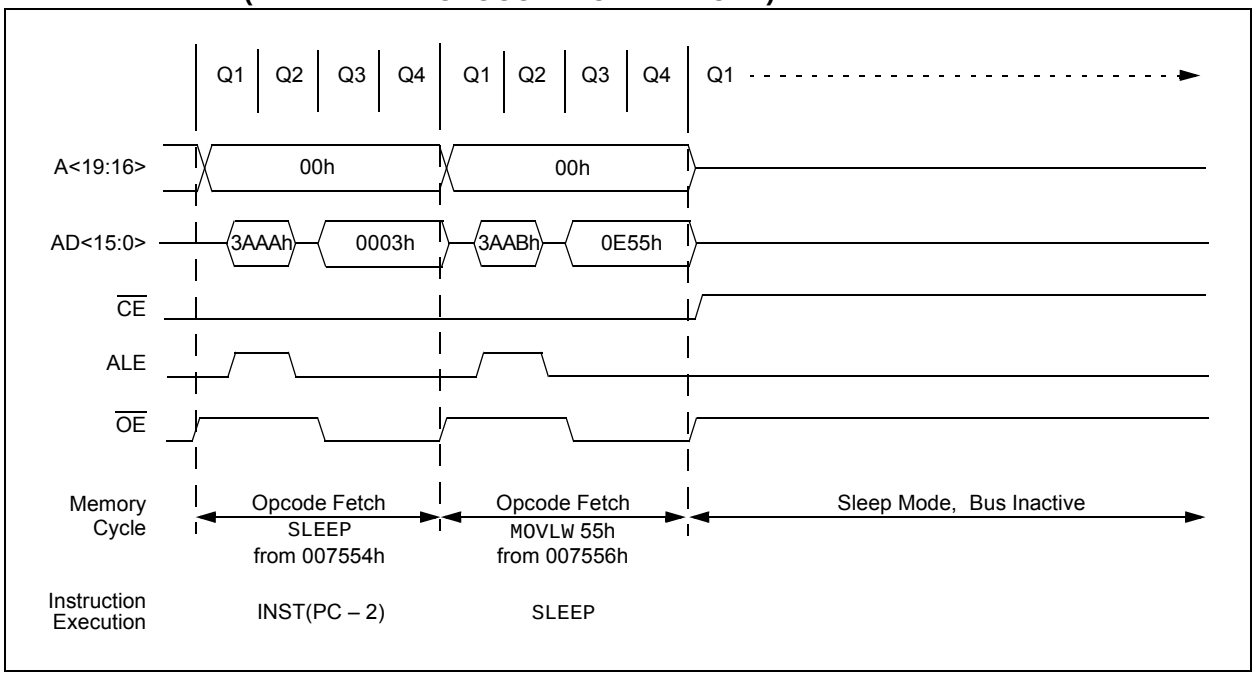


FIGURE 8-5: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



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 eight Least Significant bits of the address bus.

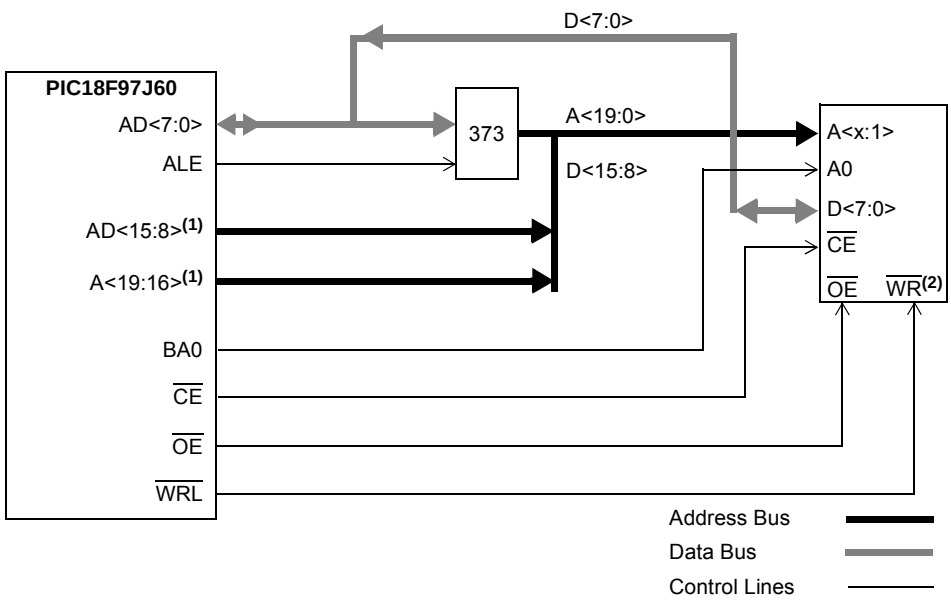
Figure 8-6 shows an example of 8-Bit Multiplexed mode for 100-pin devices. This mode is used for a single 8-bit memory connected for 16-bit operation. The instructions will be fetched as two 8-bit bytes on a shared data/address bus. The two bytes are sequentially fetched within one instruction cycle (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 in 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 process 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 used only for 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)

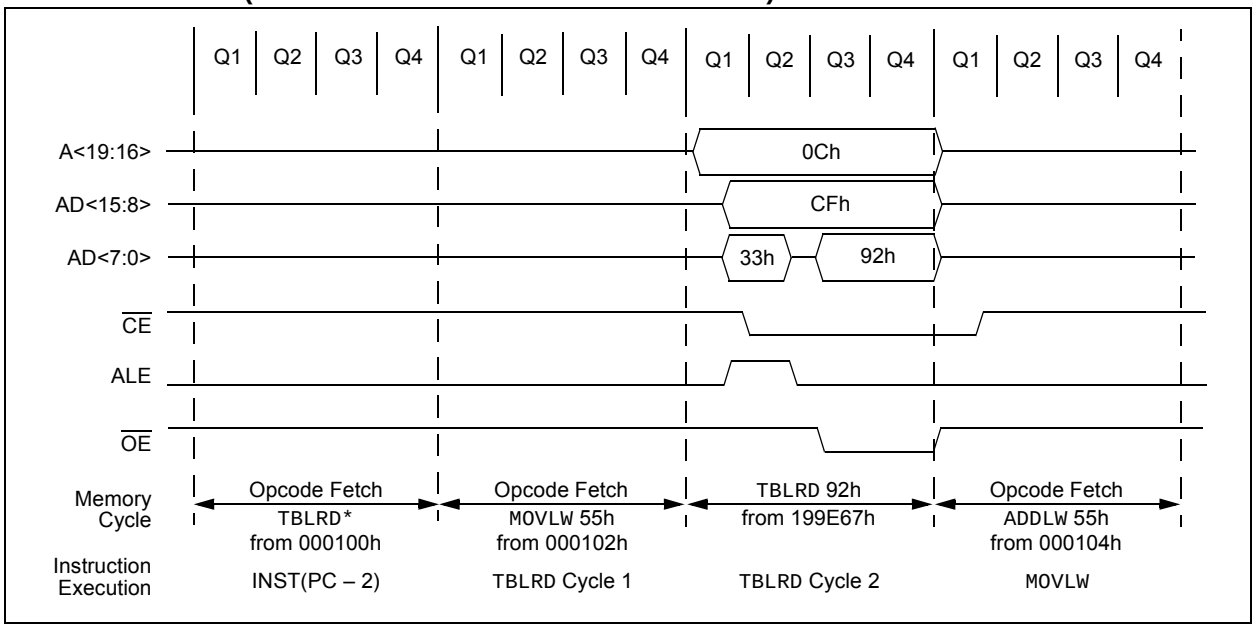
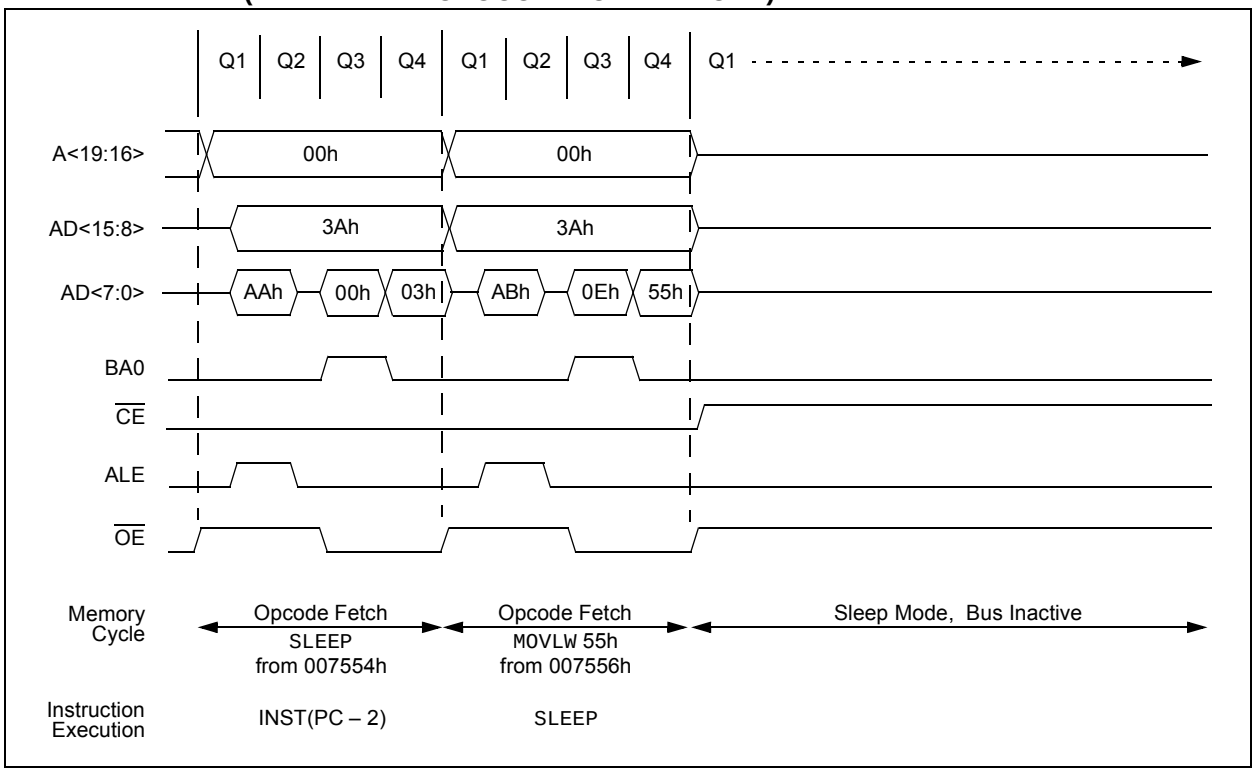


FIGURE 8-8: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



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, user applications should provide memory access time adjustments 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 PIC18F97J60 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® 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.

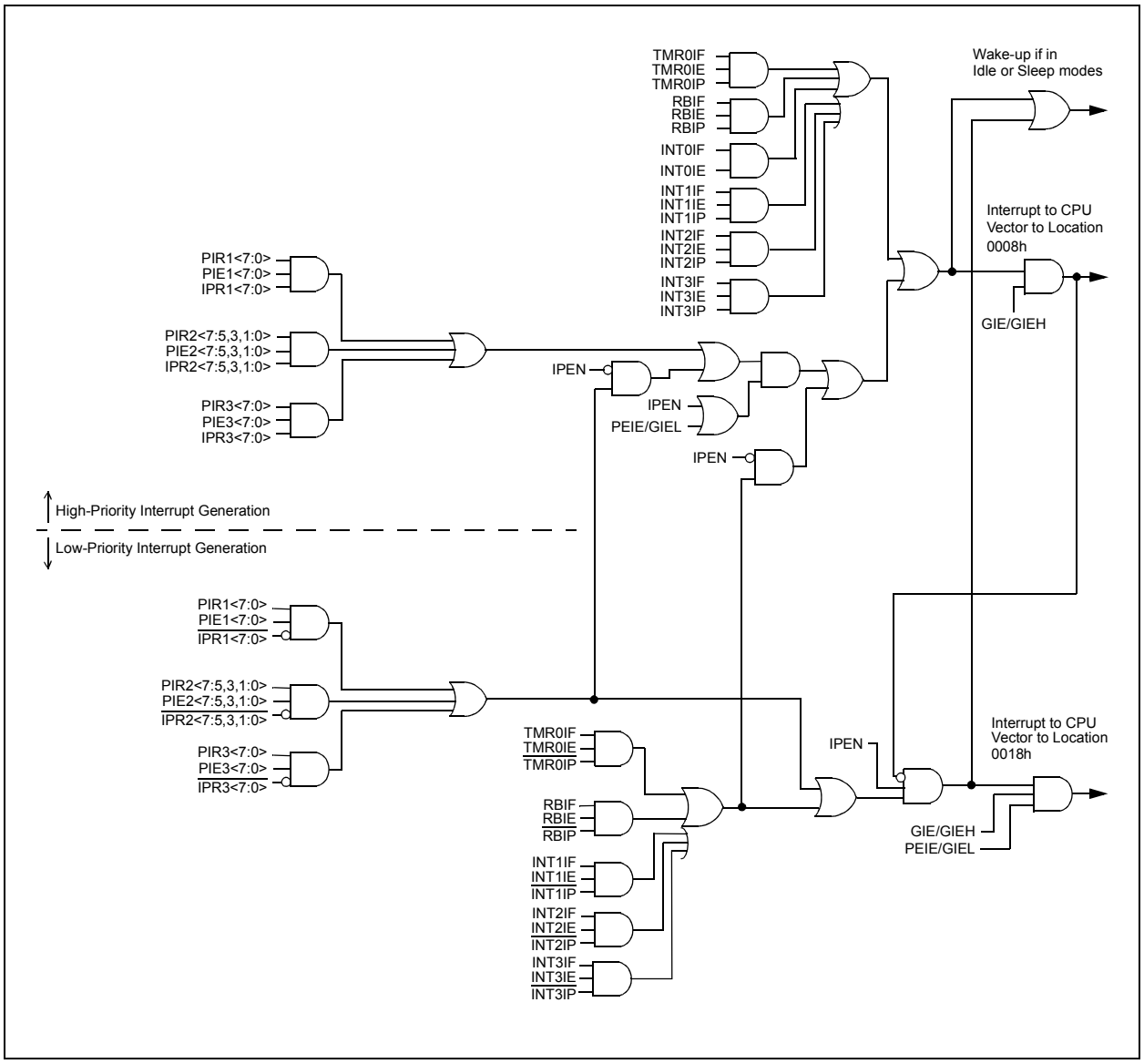
The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine (ISR), the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before 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: PIC18F97J60 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 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	R/W-0	R/W-0	R/W-0
PSPIF(1)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	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)
1 = A read or a write operation has taken place (must be cleared in software)
0 = No read or write has occurred

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

bit 5 **RC1IF:** EUSART1 Receive Interrupt Flag bit
1 = The EUSART1 receive buffer, RCREG1, is full (cleared when RCREG1 is read)
0 = The EUSART1 receive buffer is empty

bit 4 **TX1IF:** EUSART1 Transmit Interrupt Flag bit
1 = The EUSART1 transmit buffer, TXREG1, is empty (cleared when TXREG1 is written)
0 = The EUSART1 transmit buffer is full

bit 3 **SSP1IF:** MSSP1 Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** ECCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

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

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

Note 1: Implemented in 100-pin devices in Microcontroller mode only.

REGISTER 10-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF
bit 7							bit 0

Legend:	r = Reserved 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 **OSCFIF:** Oscillator Fail Interrupt Flag bit
1 = System oscillator failed, clock input has changed to INTRC (must be cleared in software)
0 = System 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 **ETHIF:** Ethernet Module Interrupt Flag bit
1 = An Ethernet module interrupt event has occurred; query EIR register to resolve source
0 = No Ethernet interrupt event has occurred

bit 4 **Reserved:** Maintain as '0'

bit 3 **BCL1IF:** Bus Collision Interrupt Flag bit (MSSP1 module)
1 = A bus collision occurred (must be cleared in software)
0 = No bus collision occurred

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

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

bit 0 **CCP2IF:** ECCP2 Interrupt Flag bit
Capture mode:
1 = A TMR1/TMR3 register capture occurred (must be cleared in software)
0 = No TMR1/TMR3 register capture occurred
Compare mode:
1 = A TMR1/TMR3 register compare match occurred (must be cleared in software)
0 = No TMR1/TMR3 register compare match occurred
PWM mode:
Unused in this mode.

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

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IF(1)	BCL2IF(1)	RC2IF(2)	TX2IF(2)	TMR4IF	CCP5IF	CCP4IF	CCP3IF
bit 7							bit 0

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

bit 7 **SSP2IF:** MSSP2 Interrupt Flag bit(1)
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 6 **BCL2IF:** Bus Collision Interrupt Flag bit (MSSP2 module)(1)
1 = A bus collision occurred (must be cleared in software)
0 = No bus collision occurred

bit 5 **RC2IF:** EUSART2 Receive Interrupt Flag bit(2)
1 = The EUSART2 receive buffer, RCREG2, is full (cleared when RCREG2 is read)
0 = The EUSART2 receive buffer is empty

bit 4 **TX2IF:** EUSART2 Transmit Interrupt Flag bit(2)
1 = The EUSART2 transmit buffer, TXREG2, is empty (cleared when TXREG2 is written)
0 = The EUSART2 transmit buffer is full

bit 3 **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = TMR4 to PR4 match occurred (must be cleared in software)
0 = No TMR4 to PR4 match occurred

bit 2 **CCP5IF:** CCP5 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 **CCP4IF:** CCP4 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 **CCP3IF:** ECCP3 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.

Note 1: Implemented in 100-pin devices only.
2: Implemented in 80-pin and 100-pin devices only.

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	R/W-0	R/W-0	R/W-0
PSPIE(1)	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

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

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

bit 6 **ADIE:** A/D Converter Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **RC1IE:** EUSART1 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TX1IE:** EUSART1 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **SSP1IE:** MSSP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP1IE:** ECCP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

Note 1: Implemented in 100-pin devices in Microcontroller mode only.

REGISTER 10-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE
bit 7							bit 0

Legend:	r = Reserved 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 **OSCFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 5 **ETHIE:** Ethernet Module Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **Reserved:** Maintain as '0'

bit 3 **BCL1IE:** Bus Collision Interrupt Enable bit (MSSP1 module)
1 = Enabled
0 = Disabled

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

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

bit 0 **CCP2IE:** ECCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

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

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IE(1)	BCL2IE(1)	RC2IE(2)	TX2IE(2)	TMR4IE	CCP5IE	CCP4IE	CCP3IE
bit 7							bit 0

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

bit 7 **SSP2IE:** MSSP2 Interrupt Enable bit(1)
1 = Enabled
0 = Disabled

bit 6 **BCL2IE:** Bus Collision Interrupt Enable bit (MSSP2 module)(1)
1 = Enabled
0 = Disabled

bit 5 **RC2IE:** EUSART2 Receive Interrupt Enable bit(2)
1 = Enabled
0 = Disabled

bit 4 **TX2IE:** EUSART2 Transmit Interrupt Enable bit(2)
1 = Enabled
0 = Disabled

bit 3 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP5IE:** CCP5 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **CCP4IE:** CCP4 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **CCP3IE:** ECCP3 Interrupt Enable bit
1 = Enabled
0 = Disabled

Note 1: Implemented in 100-pin devices only.
2: Implemented in 80-pin and 100-pin devices only.

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	R/W-1	R/W-1	R/W-1
PSPIP(1)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

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

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

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

bit 5 **RC1IP:** EUSART1 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX1IP:** EUSART1 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSP1IP:** MSSP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP1IP:** ECCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

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

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

Note 1: Implemented in 100-pin devices in Microcontroller mode only.

REGISTER 10-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1
OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP
bit 7							bit 0

Legend:	r = Reserved 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 **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 **ETHIP:** Ethernet Module Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **Reserved:** Maintain as '1'

bit 3 **BCL1IP:** Bus Collision Interrupt Priority bit (MSSP1 module)
1 = High priority
0 = Low priority

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

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

bit 0 **CCP2IP:** ECCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

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

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SSP2IP(1)	BCL2IP(1)	RC2IP(2)	TX2IP(2)	TMR4IP	CCP5IP	CCP4IP	CCP3IP
bit 7							bit 0

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

bit 7 **SSP2IP:** MSSP2 Interrupt Priority bit(1)
1 = High priority
0 = Low priority

bit 6 **BCL2IP:** Bus Collision Interrupt Priority bit (MSSP2 module)(1)
1 = High priority
0 = Low priority

bit 5 **RC2IP:** EUSART2 Receive Interrupt Priority bit(2)
1 = High priority
0 = Low priority

bit 4 **TX2IP:** EUSART2 Transmit Interrupt Priority bit(2)
1 = High priority
0 = Low priority

bit 3 **TMR4IE:** TMR4 to PR4 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP5IP:** CCP5 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CCP4IP:** CCP4 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CCP3IP:** ECCP3 Interrupt Priority bit
1 = High priority
0 = Low priority

Note 1: Implemented in 100-pin devices only.
2: Implemented in 80-pin and 100-pin devices only.

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 = Enable priority levels on interrupts
0 = Disable 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(2)
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/FLT0, 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 (ISR) 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 (FSR). 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.

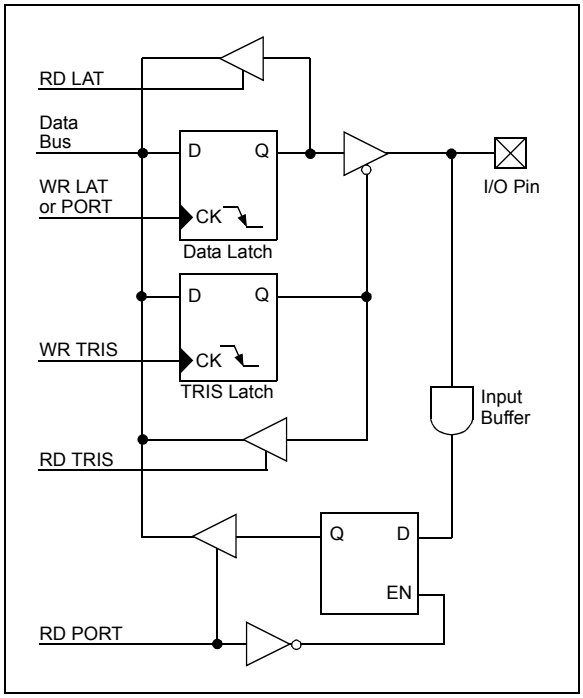
Each port has three registers for its operation. These registers are:

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

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

A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 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 PIN OUTPUT DRIVE

The output pin drive strengths vary for groups of pins intended to meet the needs for a variety of applications. PORTB and PORTC are designed to drive higher loads, such as LEDs. The external memory interface ports (PORTD, PORTE and PORTJ) are designed to drive medium loads. All other ports are designed for small loads, typically indication only. Table 11-1 summarizes the output capabilities. Refer to **Section 28.0 "Electrical Characteristics"** for more details.

TABLE 11-1: OUTPUT DRIVE LEVELS

Port	Drive	Description
PORTA(1)	Minimum	Intended for indication.
PORTF(2)		
PORTG(2)		
PORTH(3)		
PORTD(2)	Medium	Sufficient drive levels for external memory interfacing, as well as indication.
PORTE		
PORTJ(3)		
PORTB	High	Suitable for direct LED drive levels.
PORTC		

Note 1: The exceptions are RA<1:0>, which are capable of directly driving LEDs.

2: Partially implemented on 64-pin and 80-pin devices; fully implemented on 100-pin devices.

3: Unimplemented on 64-pin devices.

11.1.2 INPUT PINS AND VOLTAGE CONSIDERATIONS

The voltage tolerance of pins used as device inputs is dependent on the pin's input function. Pins that are used as digital only inputs are able to handle DC voltages up to 5.5V, a level typical for digital logic circuits. In contrast, pins that also have analog input functions of any kind can only tolerate voltages up to VDD. Voltage excursions beyond VDD on these pins are always to be avoided. Table 11-2 summarizes the input capabilities. Refer to **Section 28.0 "Electrical Characteristics"** for more details.

TABLE 11-2: INPUT VOLTAGE LEVELS

Port or Pin	Tolerated Input	Description
PORTA<5,3:0>	VDD	Only VDD input levels tolerated.
PORTF<6:1>(1)		
PORTH<7:4>(2)		
PORTA<4>	5.5V	Tolerates input levels above VDD, useful for most standard logic.
PORTB<7:0>		
PORTC<7:0>		
PORTD<7:0>(1)		
PORTE<7:0>		
PORTF<7>		
PORTG<7:0>(1)		
PORTH<3:0>(2)		
PORTJ<7:0>(2)		

Note 1: Partially implemented on 64-pin and 80-pin devices; fully implemented on 100-pin devices.

2: Unavailable in 64-pin devices.

11.2 PORTA, TRISA and LATA Registers

PORTA is a 6-bit wide, bidirectional port; it is fully implemented on all devices. The corresponding Data Direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

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

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

The RA4 pin is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The other PORTA pins are multiplexed with the analog VREF+ and VREF- inputs. The operation of pins, RA<5:0>, as A/D Converter inputs is selected by clearing or setting the PCFG<3:0> control bits in the ADCON1 register.

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.

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

The TRISA register controls the direction of the PORTA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

The RA0 and RA1 pins can also be configured as the outputs for the two Ethernet LED indicators. When configured, these two pins are the only pins on PORTA that are capable of high output drive levels.

Although the port is only six bits wide, PORTA<7> is implemented as RJPU, the weak pull-up control bit for PORTJ. In a similar fashion, the LATA<7:6> bits are implemented, not as latch bits, but the pull-up control bits, RDPU and REPU, for PORTD and PORTE. Setting these bits enables the pull-ups for the corresponding port. Because their port pins are not used, the TRISA<7:6> bits are not implemented.

EXAMPLE 11-1: INITIALIZING PORTA

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

TABLE 11-3: PORTA FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RA0/LEDA/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 is enabled.
	LEDA	0	O	DIG	Ethernet LEDA output; takes priority over digital data.
	AN0	1	I	ANA	A/D Input Channel 0. Default input configuration on POR; does not affect digital output.
RA1/LEDB/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 is enabled.
	LEDB	0	O	DIG	Ethernet LEDB output; takes priority over digital data.
	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. Disabled when CVREF output is enabled.
		1	I	TTL	PORTA<2> data input. Disabled when analog functions are enabled; disabled when CVREF output is enabled.
	AN2	1	I	ANA	A/D Input Channel 2 and Comparator C2+ input. Default input configuration on POR; not affected by analog output.
	VREF-	1	I	ANA	A/D and comparator 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 is enabled.
	AN3	1	I	ANA	A/D Input Channel 3. Default input configuration on POR.
	VREF+	1	I	ANA	A/D 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 is enabled.
	AN4	1	I	ANA	A/D Input Channel 4. Default configuration on POR.

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	RJPU(1)	—	RA5	RA4	RA3	RA2	RA1	RA0	72
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	72
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	71
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	70

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

Note 1: Implemented in 80-pin and 100-pin devices only.

11.3 PORTB, TRISB and LATB Registers

PORTB is an 8-bit wide, bidirectional port; it is fully implemented on all devices. The corresponding Data Direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin). All pins on PORTB are digital only and tolerate voltages up to 5.5V.

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

EXAMPLE 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
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all of 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 all Resets.

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 Flag bit, RBIF (INTCON<0>).

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

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

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

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

For 100-pin devices operating in Extended Microcontroller mode, RB3 can be configured as the alternate peripheral pin for the ECCP2 module and Enhanced PWM Output 2A by clearing the CCP2MX Configuration bit. If the devices are in Microcontroller mode, the alternate assignment for ECCP2 is RE7. As with other ECCP2 configurations, the user must ensure that the TRISB<3> bit is set appropriately for the intended operation.

TABLE 11-5: PORTB FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RB0/INT0/FLT0	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.
	FLT0	1	I	ST	Enhanced PWM Fault input (ECCP1 module); enabled in software.
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/ ECCP2/P2A	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.
	ECCP2(1)	0	O	DIG	ECCP2 compare output and PWM output; takes priority over port data.
		1	I	ST	ECCP2 capture input.
	P2A(1)	0	O	DIG	ECCP2 Enhanced PWM output, Channel A. May be configured for tri-state during Enhanced PWM shutdown events. Takes priority over port data.
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	1	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	1	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 ECCP2/P2A when the CCP2MX Configuration bit is cleared (100-pin devices in Extended Microcontroller mode). 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	72
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	72
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	71
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	69
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	69

Legend: Shaded cells are not used by PORTB.

11.4 PORTC, TRISC and LATC Registers

PORTC is an 8-bit wide, bidirectional port; it is fully implemented on all devices. The corresponding Data Direction register is TRISC. Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., put the contents of the output latch on the selected pin). Only PORTC pins, RC2 through RC7, are digital only pins and can tolerate input voltages up to 5.5V.

The Output Latch register (LATC) is also memory mapped. Read-modify-write operations on the LATC register read and write the latched output value for PORTC.

PORTC is multiplexed with several peripheral functions (Table 11-7). The pins have Schmitt Trigger input buffers. RC1 is normally configured by Configuration bit, CCP2MX, as the default peripheral pin for the ECCP2 module and Enhanced PWM output, P2A (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 is enabled. Disables digital I/O.
	T13CKI	1	I	ST	Timer1/Timer3 counter input.
RC1/T1OSI/ ECCP2/P2A	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 is enabled. Disables digital I/O.
	ECCP2(1)	0	O	DIG	ECCP2 compare output and PWM output; takes priority over port data.
		1	I	ST	ECCP2 capture input.
	P2A(1)	0	O	DIG	ECCP2 Enhanced PWM output, Channel A. May be configured for tri-state during Enhanced PWM shutdown events. Takes priority over port data.
RC2/ECCP1/ P1A	RC2	0	O	DIG	LATC<2> data output.
		1	I	ST	PORTC<2> data input.
	ECCP1	0	O	DIG	ECCP1 compare output and PWM output; takes priority over port data.
		1	I	ST	ECCP1 capture input.
	P1A	0	O	DIG	ECCP1 Enhanced PWM output, Channel A. May be configured for tri-state during Enhanced PWM shutdown events. Takes priority over port data.
RC3/SCK1/ SCL1	RC3	0	O	DIG	LATC<3> data output.
		1	I	ST	PORTC<3> data input.
	SCK1	0	O	DIG	SPI clock output (MSSP1 module); takes priority over port data.
		1	I	ST	SPI clock input (MSSP1 module).
	SCL1	0	O	DIG	I^2C™ clock output (MSSP1 module); takes priority over port data.
		1	I	ST	I^2C clock input (MSSP1 module); input type depends on module setting.
RC4/SDI1/ SDA1	RC4	0	O	DIG	LATC<4> data output.
		1	I	ST	PORTC<4> data input.
	SDI1	1	I	ST	SPI data input (MSSP1 module).
	SDA1	1	O	DIG	I^2C data output (MSSP1 module); takes priority over port data.
		1	I	ST	I^2C data input (MSSP1 module); input type depends on module setting.
RC5/SDO1	RC5	0	O	DIG	LATC<5> data output.
		1	I	ST	PORTC<5> data input.
	SDO1	0	O	DIG	SPI data output (MSSP1 module); takes priority over port data.
RC6/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 (EUSART1 module); takes priority over port data.
	CK1	1	O	DIG	Synchronous serial data input (EUSART1 module). User must configure as an input.
		1	I	ST	Synchronous serial clock input (EUSART1 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 (EUSART1 module).
	DT1	1	O	DIG	Synchronous serial data output (EUSART1 module); takes priority over port data.
		1	I	ST	Synchronous serial data input (EUSART1 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 ECCP2/P2A 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	72
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	72
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	71

11.5 PORTD, TRISD and LATD Registers

PORTD is implemented as a bidirectional port in two ways:

- 64-pin and 80-pin devices: 3 bits (RD<2:0>)
- 100-pin devices: 8 bits (RD<7:0>)

The corresponding Data Direction register is TRISD. Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., put the contents of the output latch on the selected pin). All pins on PORTD are digital only and tolerate voltages up to 5.5V.

The Output Latch register (LATD) is also memory mapped. Read-modify-write operations on the LATD register read and write the latched output value for PORTD.

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

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

On 100-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.

Each of the PORTD pins has a weak internal pull-up. A single control bit can turn on all of the pull-ups. This is performed by setting the RDPU bit (LATA<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 100-pin devices, 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 (PSP)"**.

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/P1B)	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input; weak pull-up when RDPU bit is set.
	AD0(1)	x	O	DIG	External memory interface, Address/Data Bit 0 output.(2)
		x	I	TTL	External memory interface, Data Bit 0 input.(2)
	PSP0(1)	x	O	DIG	PSP read output data (LATD<0>); takes priority over port data.
		x	I	TTL	PSP write data input.
	P1B(3)	0	O	DIG	ECCP1 Enhanced PWM output, Channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RD1/AD1/PSP1 (RD1/ECCP3/ P3A)	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input; weak pull-up when RDPU bit is set.
	AD1(1)	x	O	DIG	External memory interface, Address/Data Bit 1 output.(2)
		x	I	TTL	External memory interface, Data Bit 1 input.(2)
	PSP1(1)	x	O	DIG	PSP read output data (LATD<1>); takes priority over port data.
		x	I	TTL	PSP write data input.
	ECCP3(3)	0	O	DIG	ECCP3 compare and PWM output; takes priority over port data.
		1	I	ST	ECCP3 capture input.
	P3A(3)	0	O	DIG	ECCP3 Enhanced PWM output, Channel A; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RD2/AD2/PSP2 (RD2/CCP4/ P3D)	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input; weak pull-up when RDPU bit is set.
	AD2(1)	x	O	DIG	External memory interface, Address/Data Bit 2 output.(2)
		x	I	TTL	External memory interface, Data Bit 2 input.(2)
	PSP2(1)	x	O	DIG	PSP read output data (LATD<2>); takes priority over port data.
		x	I	TTL	PSP write data input.
	CCP4(3)	0	O	DIG	CCP4 compare output and PWM output; takes priority over port data.
		1	I	ST	CCP4 capture input.
	P3D(3)	0	O	DIG	ECCP3 Enhanced PWM output, Channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RD3/AD3/ PSP3(1)	RD3(1)	0	O	DIG	LATD<3> data output.
		1	I	ST	PORTD<3> data input; weak pull-up when RDPU bit is set.
	AD3(1)	x	O	DIG	External memory interface, Address/Data Bit 3 output.(2)
		x	I	TTL	External memory interface, Data Bit 3 input.(2)
	PSP3(1)	x	O	DIG	PSP read output data (LATD<3>); takes priority over port data.
		x	I	TTL	PSP write data input.
RD4/AD4/ PSP4/SDO2(1)	RD4(1)	0	O	DIG	LATD<4> data output.
		1	I	ST	PORTD<4> data input; weak pull-up when RDPU bit is set.
	AD4(1)	x	O	DIG	External memory interface, Address/Data Bit 4 output.(2)
		x	I	TTL	External memory interface, Data Bit 4 input.(2)
	PSP4(1)	x	O	DIG	PSP read output data (LATD<4>); takes priority over port data.
		x	I	TTL	PSP write data input.
	SDO2(1)	0	O	DIG	SPI data output (MSSP2 module); takes priority over port data.

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: These features or port pins are implemented only on 100-pin devices.

2: External memory interface I/O takes priority over all other digital and PSP I/O.

3: These features are implemented on this pin only on 64-pin devices; for all other devices, they are multiplexed with RE6/RH7 (P1B), RG0 (ECCP3/P3A) or RG3 (CCP4/P3D).

TABLE 11-9: PORTD FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD5/AD5/ PSP5/SDI2/ SDA2(1)	RD5(1)	0	O	DIG	LATD<5> data output.
		1	I	ST	PORTD<5> data input; weak pull-up when RDPU bit is set.
	AD5(1)	x	O	DIG	External memory interface, Address/Data Bit 5 output.(2)
		x	I	TTL	External memory interface, Data Bit 5 input.(2)
	PSP5(1)	x	O	DIG	PSP read output data (LATD<5>); takes priority over port data.
		x	I	TTL	PSP write data input.
	SDI2(1)	1	I	ST	SPI data input (MSSP2 module).
	SDA2(1)	1	O	DIG	I^2C™ data output (MSSP2 module); takes priority over port data.
		1	I	ST	I^2C data input (MSSP2 module); input type depends on module setting.
RD6/AD6/ PSP6/SCK2/ SCL2(1)	RD6(1)	0	O	DIG	LATD<6> data output.
		1	I	ST	PORTD<6> data input; weak pull-up when RDPU bit is set.
	AD6(1)	x	O	DIG-3	External memory interface, Address/Data Bit 6 output.(2)
		x	I	TTL	External memory interface, Data Bit 6 input.(2)
	PSP6(1)	x	O	DIG	PSP read output data (LATD<6>); takes priority over port data.
		x	I	TTL	PSP write data input.
	SCK2(1)	0	O	DIG	SPI clock output (MSSP2 module); takes priority over port data.
		1	I	ST	SPI clock input (MSSP2 module).
	SCL2(1)	0	O	DIG	I^2C clock output (MSSP2 module); takes priority over port data.
		1	I	ST	I^2C clock input (MSSP2 module); input type depends on module setting.
RD7/AD7/ PSP7/$\overline{SS2}$(1)	RD7(1)	0	O	DIG	LATD<7> data output.
		1	I	ST	PORTD<7> data input; weak pull-up when RDPU bit is set.
	AD7(1)	x	O	DIG	External memory interface, Address/Data Bit 7 output.(2)
		x	I	TTL	External memory interface, Data Bit 7 input.(2)
	PSP7(1)	x	O	DIG	PSP read output data (LATD<7>); takes priority over port data.
		x	I	TTL	PSP write data input.
	$\overline{SS2}$(1)	x	I	TTL	Slave select input for MSSP2 module.

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: These features or port pins are implemented only on 100-pin devices.

2: External memory interface I/O takes priority over all other digital and PSP I/O.

3: These features are implemented on this pin only on 64-pin devices; for all other devices, they are multiplexed with RE6/RH7 (P1B), RG0 (ECCP3/P3A) or RG3 (CCP4/P3D).

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(1)	RD6(1)	RD5(1)	RD4(1)	RD3(1)	RD2	RD1	RD0	72
LATD	LATD7(1)	LATD6(1)	LATD5(1)	LATD4(1)	LATD3(1)	LATD2	LATD1	LATD0	72
TRISD	TRISD7(1)	TRISD6(1)	TRISD5(1)	TRISD4(1)	TRISD3(1)	TRISD2	TRISD1	TRISD0	71
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	72

Legend: Shaded cells are not used by PORTD.

Note 1: Unimplemented on 64-pin and 80-pin devices; read as '0'.

11.6 PORTE, TRISE and LATE Registers

PORTE is implemented as a bidirectional port in two different ways:

- 64-pin devices: 6 bits wide (RE<5:0>)
- 80-pin and 100-pin devices: 8 bits wide (RE<7:0>)

The corresponding Data Direction register is TRISE. Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., put the contents of the output latch on the selected pin). All pins on PORTE are digital only and tolerate voltages up to 5.5V.

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

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

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

On 100-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.

Each of the PORTE pins has a weak internal pull-up. A single control bit can turn on all of the pull-ups. This is performed by setting bit, REPU (LATA<6>). 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.

PORTE is also multiplexed with Enhanced PWM Outputs B and C for ECCP1 and ECCP3 and Outputs B, C and D for ECCP2. For 80-pin and 100-pin devices, their default assignments are on PORTE<6:0>. For 64-pin devices, their default assignments are on PORTE<5:0> and PORTD<0>. On 80-pin and 100-pin devices, the multiplexing for the outputs of ECCP1 and ECCP3 is controlled by the ECCPMX Configuration bit. Clearing this bit reassigns the P1B/P1C and P3B/P3C outputs to PORTH.

For 80-pin and 100-pin devices operating in Micro-controller mode, pin, RE7, can be configured as the alternate peripheral pin for the ECCP2 module and Enhanced PWM Output 2A. This is done by clearing the CCP2MX Configuration bit.

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.

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/AD8/$\overline{RD}$/ P2D	RE0	0	O	DIG	LATE<0> data output.
		1	I	ST	PORTE<0> data input; weak pull-up when REPU bit is set.
	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)
	$\overline{RD}$(6)	1	I	TTL	Parallel Slave Port read enable control input.
	P2D	0	O	DIG	ECCP2 Enhanced PWM output, Channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE1/AD9/$\overline{WR}$/ P2C	RE1	0	O	DIG	LATE<1> data output.
		1	I	ST	PORTE<1> data input; weak pull-up when REPU bit is set.
	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)
	$\overline{WR}$(6)	1	I	TTL	Parallel Slave Port write enable control input.
	P2C	0	O	DIG	ECCP2 Enhanced PWM output, Channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE2/AD10/$\overline{CS}$/ P2B	RE2	0	O	DIG	LATE<2> data output.
		1	I	ST	PORTE<2> data input; weak pull-up when REPU bit is set.
	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)
	$\overline{CS}$(6)	1	I	TTL	Parallel Slave Port chip select control input.
	P2B	0	O	DIG	ECCP2 Enhanced PWM output, Channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE3/AD11/ P3C	RE3	0	O	DIG	LATE<3> data output.
		1	I	ST	PORTE<3> data input; weak pull-up when REPU bit is set.
	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)
	P3C(3)	0	O	DIG	ECCP3 Enhanced PWM output, Channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE4/AD12/ P3B	RE4	0	O	DIG	LATE<4> data output.
		1	I	ST	PORTE<4> data input; weak pull-up when REPU bit is set.
	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)
	P3B(3)	0	O	DIG	ECCP3 Enhanced PWM output, channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.

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: EMB functions are implemented on 100-pin devices only.

2: External memory interface I/O takes priority over all other digital and PSP I/O.

3: Default assignments for P1B/P1C and P3B/P3C when ECCPMX Configuration bit is set (80-pin and 100-pin devices).

4: Unimplemented on 64-pin devices.

5: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (80-pin and 100-pin devices in Microcontroller mode).

6: Unimplemented on 64-pin and 80-pin devices.

TABLE 11-11: PORTE FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RE5/AD13/ P1C	RE5	0	O	DIG	LATE<5> data output.
		1	I	ST	PORTE<5> data input; weak pull-up when REPU bit is set.
	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)
	P1C(3)	0	O	DIG	ECCP1 Enhanced PWM output, Channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE6/AD14/ P1B(4)	RE6	0	O	DIG	LATE<6> data output.
		1	I	ST	PORTE<6> data input; weak pull-up when REPU bit is set.
	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)
	P1B(3)	0	O	DIG	ECCP1 Enhanced PWM output, Channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RE7/AD15/ ECCP2/P2A(4)	RE7	0	O	DIG	LATE<7> data output.
		1	I	ST	PORTE<7> data input; weak pull-up when REPU bit is set.
	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)
	ECCP2(5)	0	O	DIG	ECCP2 compare output and PWM output; takes priority over port data.
		1	I	ST	ECCP2 capture input.
	P2A(5)	0	O	DIG	ECCP2 Enhanced PWM output, Channel A; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.

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: EMB functions are implemented on 100-pin devices only.
2: External memory interface I/O takes priority over all other digital and PSP I/O.
3: Default assignments for P1B/P1C and P3B/P3C when ECCPMX Configuration bit is set (80-pin and 100-pin devices).
4: Unimplemented on 64-pin devices.
5: Alternate assignment for ECCP2/P2A when CCP2MX Configuration bit is cleared (80-pin and 100-pin devices in Microcontroller mode).
6: Unimplemented on 64-pin and 80-pin devices.

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(1)	RE6(1)	RE5	RE4	RE3	RE2	RE1	RE0	72
LATE	LATE7(1)	LATE6(1)	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	72
TRISE	TRISE7(1)	TRISE6(1)	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	71
LATA	RDPU	REPU	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	72

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 implemented as a bidirectional port in two different ways:

- 64-pin and 80-pin devices: 7 bits wide (RF<7:1>)
- 100-pin devices: 8 bits wide (RF<7:0>)

The corresponding Data Direction register is TRISF. Setting a TRISF bit (= 1) will make the corresponding PORTF pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISF bit (= 0) will make the corresponding PORTF pin an output (i.e., put the contents of the output latch on the selected pin). Only Pin 7 of PORTF has no analog input; it is the only pin that can tolerate voltages up to 5.5V.

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

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

PORTF is multiplexed with several analog peripheral functions, including the A/D Converter and comparator inputs, as well as the comparator outputs. Pins, RF1 through RF6, may be used as comparator inputs or outputs by setting the appropriate bits in the CMCON register. To use RF<6:1> 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 the comparators and set the 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
RF0/AN5(1)	RF0(1)	0	O	DIG	LATF<0> data output; not affected by analog input.
		1	I	ST	PORTF<0> data input; disabled when analog input is enabled.
	AN5(1)	1	I	ANA	A/D Input Channel 5. Default configuration on POR.
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 is 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 is 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 is 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 is 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 is enabled.
		1	I	ST	PORTF<5> data input; disabled when analog input is enabled. Disabled when CVREF output is 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 is 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/$\overline{SS1}$	RF7	0	O	DIG	LATF<7> data output.
		1	I	ST	PORTF<7> data input.
	$\overline{SS1}$	1	I	TTL	Slave select input for MSSP1 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).

Note 1: Implemented on 100-pin devices only.

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	RF0(1)	72
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0(1)	72
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0(1)	71
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	70
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	70
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	70

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

Note 1: Implemented on 100-pin devices only.

11.8 PORTG, TRISG and LATG Registers

Depending on the particular device, PORTG is implemented as a bidirectional port in one of three ways:

- 64-pin devices: 1 bit wide (RG<4>)
- 80-pin devices: 5 bits wide (RG<4:0>)
- 100-pin devices: 8 bits wide (RG<7:0>)

The corresponding Data Direction register is TRISG. Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISG bit (= 0) will make the corresponding PORTG pin an output (i.e., put the contents of the output latch on the selected pin). All pins on PORTG are digital only and tolerate voltages up to 5.5V.

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

PORTG is multiplexed with EUSART2 functions on 80-pin and 100-pin devices (Table 11-15). PORTG pins have Schmitt Trigger input buffers.

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

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/ECCP3/ P3A(1)	RG0(1)	0	O	DIG	LATG<0> data output.
		1	I	ST	PORTG<0> data input.
	ECCP3(1)	0	O	DIG	ECCP3 compare and PWM output; takes priority over port data.
		1	I	ST	ECCP3 capture input.
	P3A(1)	0	O	DIG	ECCP3 Enhanced PWM output, Channel A; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RG1/TX2/ CK2(1)	RG1(1)	0	O	DIG	LATG<1> data output.
		1	I	ST	PORTG<1> data input.
	TX2(1)	1	O	DIG	Synchronous serial data output (EUSART2 module); takes priority over port data.
	CK2(1)	1	O	DIG	Synchronous serial data input (EUSART2 module). User must configure as an input.
		1	I	ST	Synchronous serial clock input (EUSART2 module).
RG2/RX2/ DT2(1)	RG2(1)	0	O	DIG	LATG<2> data output.
		1	I	ST	PORTG<2> data input.
	RX2(1)	1	I	ST	Asynchronous serial receive data input (EUSART2 module).
	DT2(1)	1	O	DIG	Synchronous serial data output (EUSART2 module); takes priority over port data.
		1	I	ST	Synchronous serial data input (EUSART2 module). User must configure as an input.
RG3/CCP4/ P3D(1)	RG3(1)	0	O	DIG	LATG<3> data output.
		1	I	ST	PORTG<3> data input.
	CCP4(1)	0	O	DIG	CCP4 compare output and PWM output; takes priority over port data.
		1	I	ST	CCP4 capture input.
	P3D(1)	0	O	DIG	ECCP3 Enhanced PWM output, Channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RG4/CCP5/ P1D	RG4	0	O	DIG	LATG<4> data output.
		1	I	ST	PORTG<4> data input.
	CCP5	0	O	DIG	CCP5 compare output and PWM output; takes priority over port data.
		1	I	ST	CCP5 capture input.
	P1D	0	O	DIG	ECCP1 Enhanced PWM output, Channel D; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RG5(2)	RG5(2)	0	O	DIG	LATG<0> data output.
		1	I	ST	PORTG<0> data input.
RG6(2)	RG6(2)	0	O	DIG	LATG<0> data output.
		1	I	ST	PORTG<0> data input.
RG7(2)	RG7(2)	0	O	DIG	LATG<0> data output.
		1	I	ST	PORTG<0> 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).

Note 1: Implemented on 80-pin and 100-pin devices only.

2: Implemented on 100-pin devices only.

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	RG7(1)	RG6(1)	RG5(1)	RG4	RG3(2)	RG2(2)	RG1(2)	RG0(2)	72
LATG	LATG7(1)	LATG6(1)	LATG5(1)	LATG4	LATG3(2)	LATG2(2)	LATG1(2)	LATG0(2)	72
TRISG	TRISG7(1)	TRISG6(1)	TRISG5(1)	TRISG4	TRISG3(2)	TRISG2(2)	TRISG1(2)	TRISG0(2)	71

Note 1: Implemented on 100-pin devices only.

2: Implemented on 80-pin and 100-pin devices only.

11.9 PORTH, LATH and TRISH Registers

Note: PORTH is available only on 80-pin and 100-pin devices.

PORTH is an 8-bit wide, bidirectional I/O port; it is fully implemented on 80-pin and 100-pin devices. The corresponding Data Direction register is TRISH. Setting a TRISH bit (= 1) will make the corresponding PORTH pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISH bit (= 0) will make the corresponding PORTH pin an output (i.e., put the contents of the output latch on the selected pin). PORTH<3:0> pins are digital only and tolerate voltages up to 5.5V.

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

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

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.

PORTH pins, RH4 through RH7, are multiplexed with analog converter inputs. The operation of these pins as analog inputs is selected by clearing or setting the PCFG<3:0> control bits in the ADCON1 register.

PORTH can also be configured as the alternate Enhanced PWM Output Channels B and C for the ECCP1 and ECCP3 modules. This is done by clearing the ECCPMX Configuration bit.

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(1)	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(1)	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(1)	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(1)	x	O	DIG	External memory interface, Address Line 19. Takes priority over port data.
RH4/AN12/P3C	RH4	0	O	DIG	LATH<4> data output.
		1	I	ST	PORTH<4> data input.
	AN12		I	ANA	A/D Input Channel 12. Default input configuration on POR; does not affect digital output.
	P3C(2)	0	O	DIG	ECCP3 Enhanced PWM output, Channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RH5/AN13/P3B	RH5	0	O	DIG	LATH<5> data output.
		1	I	ST	PORTH<5> data input.
	AN13		I	ANA	A/D Input Channel 13. Default input configuration on POR; does not affect digital output.
	P3B(2)	0	O	DIG	ECCP3 Enhanced PWM output, Channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RH6/AN14/P1C	RH6	0	O	DIG	LATH<6> data output.
		1	I	ST	PORTH<6> data input.
	AN14		I	ANA	A/D Input Channel 14. Default input configuration on POR; does not affect digital output.
	P1C(2)	0	O	DIG	ECCP1 Enhanced PWM output, Channel C; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.
RH7/AN15/P1B	RH7	0	O	DIG	LATH<7> data output.
		1	I	ST	PORTH<7> data input.
	AN15		I	ANA	A/D Input Channel 15. Default input configuration on POR; does not affect digital output.
	P1B(2)	0	O	DIG	ECCP1 Enhanced PWM output, Channel B; takes priority over port and PSP data. May be configured for tri-state during Enhanced PWM shutdown events.

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: Unimplemented on 80-pin devices.

2: Alternate assignments for P1B/P1C and P3B/P3C when ECCPMX Configuration bit is cleared (80-pin and 100-pin devices only). Default assignments are PORTE<6:3>.

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	72
LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	71
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	71

11.10 PORTJ, TRISJ and LATJ Registers

Note: PORTJ is available only on 80-pin and 100-pin devices.

PORTJ is implemented as a bidirectional port in two different ways:

- 80-pin devices: 2 bits wide (RJ<5:4>)
- 100-pin devices: 8 bits wide (RJ<7:0>)

The corresponding Data Direction register is TRISJ. Setting a TRISJ bit (= 1) will make the corresponding PORTJ pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISJ bit (= 0) will make the corresponding PORTJ pin an output (i.e., put the contents of the output latch on the selected pin). All pins on PORTJ are digital only and tolerate voltages up to 5.5V.

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

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

Note: These pins are configured as digital inputs 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.

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

EXAMPLE 11-9: INITIALIZING PORTJ

```
CLRF    PORTJ   ; Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATJ    ; Alternate method
                ; to clear output
                ; data 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(1)	RJ0(1)	0	O	DIG	LATJ<0> data output.
		1	I	ST	PORTJ<0> data input; weak pull-up when RJPU bit is set.
	ALE(1)	x	O	DIG	External memory interface address latch enable control output; takes priority over digital I/O.
RJ1/$\overline{OE}$(1)	RJ1(1)	0	O	DIG	LATJ<1> data output.
		1	I	ST	PORTJ<1> data input; weak pull-up when RJPU bit is set.
	$\overline{OE}$(1)	x	O	DIG	External memory interface output enable control output; takes priority over digital I/O.
RJ2/$\overline{WRL}$(1)	RJ2(1)	0	O	DIG	LATJ<2> data output.
		1	I	ST	PORTJ<2> data input; weak pull-up when RJPU bit is set.
	$\overline{WRL}$(1)	x	O	DIG	External memory bus write low byte control; takes priority over digital I/O.
RJ3/$\overline{WRH}$(1)	RJ3(1)	0	O	DIG	LATJ<3> data output.
		1	I	ST	PORTJ<3> data input; weak pull-up when RJPU bit is set.
	$\overline{WRH}$(1)	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; weak pull-up when RJPU bit is set.
	BA0(2)	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; weak pull-up when RJPU bit is set.
	$\overline{CE}$(2)	x	O	DIG	External memory interface chip enable control output; takes priority over digital I/O.
RJ6/$\overline{LB}$(1)	RJ6(1)	0	O	DIG	LATJ<6> data output.
		1	I	ST	PORTJ<6> data input; weak pull-up when RJPU bit is set.
	$\overline{LB}$(1)	x	O	DIG	External memory interface lower byte enable control output; takes priority over digital I/O.
RJ7/$\overline{UB}$(1)	RJ7(1)	0	O	DIG	LATJ<7> data output.
		1	I	ST	PORTJ<7> data input; weak pull-up when RJPU bit is set.
	$\overline{UB}$(1)	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).

Note 1: Implemented on 100-pin devices only.

2: EMB functions are implemented on 100-pin devices only.

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(1)	RJ6(1)	RJ5	RJ4	RJ3(1)	RJ2(1)	RJ1(1)	RJ0(1)	72
LATJ	LATJ7(1)	LATJ6(1)	LATJ5	LATJ4	LATJ3(1)	LATJ2(1)	LATJ1(1)	LATJ0(1)	71
TRISJ	TRISJ7(1)	TRISJ6(1)	TRISJ5	TRISJ4	TRISJ3(1)	TRISJ2(1)	TRISJ1(1)	TRISJ0(1)	71
PORTA	RJPU	—	RA5	RA4	RA3	RA2	RA1	RA0	72

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

Note 1: Implemented on 100-pin devices only.

11.11 Parallel Slave Port (PSP)

Note: The Parallel Slave Port is only implemented on 100-pin devices.

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 the $\overline{RD}$ control input pin, RE0/AD8/$\overline{RD}$/P2D and $\overline{WR}$ control input pin, RE1/AD9//$\overline{WR}$/P2C.

Note: 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/AD8/$\overline{RD}$/P2D, to be the $\overline{RD}$ input, RE1/AD9//$\overline{WR}$/P2C to be the $\overline{WR}$ input and RE2/AD10//$\overline{CS}$/P2B 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).

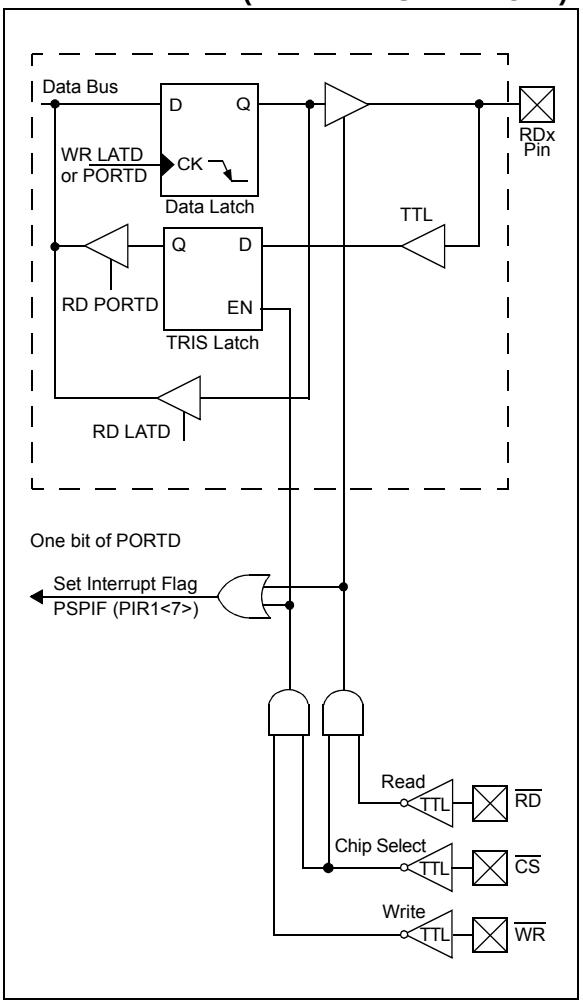
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits are both set when the write ends.

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 is 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 11-3 and Figure 11-4, respectively.

FIGURE 11-2: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



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 11-3: PARALLEL SLAVE PORT WRITE WAVEFORMS

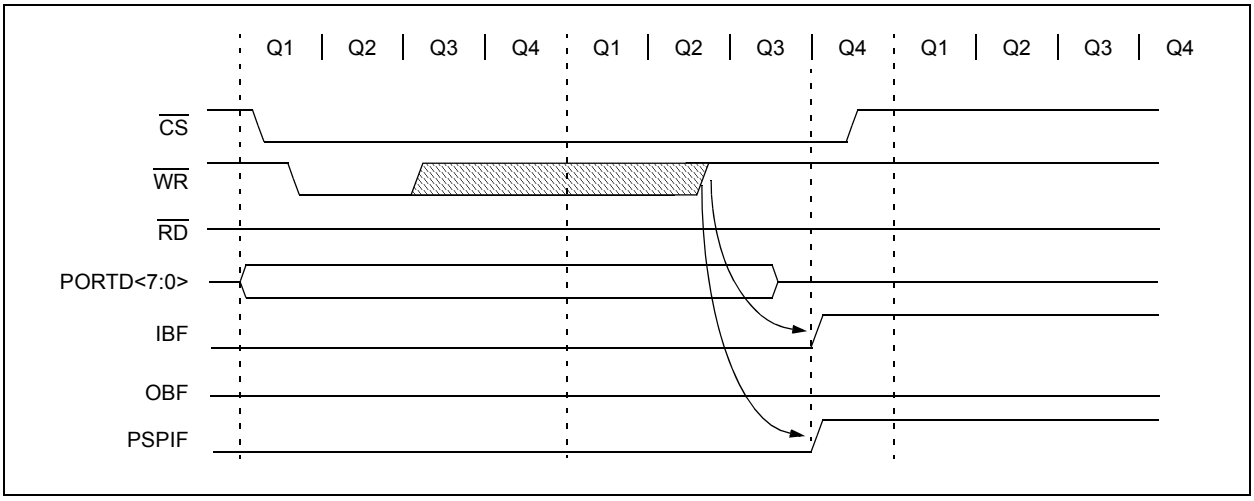


FIGURE 11-4: PARALLEL SLAVE PORT READ WAVEFORMS

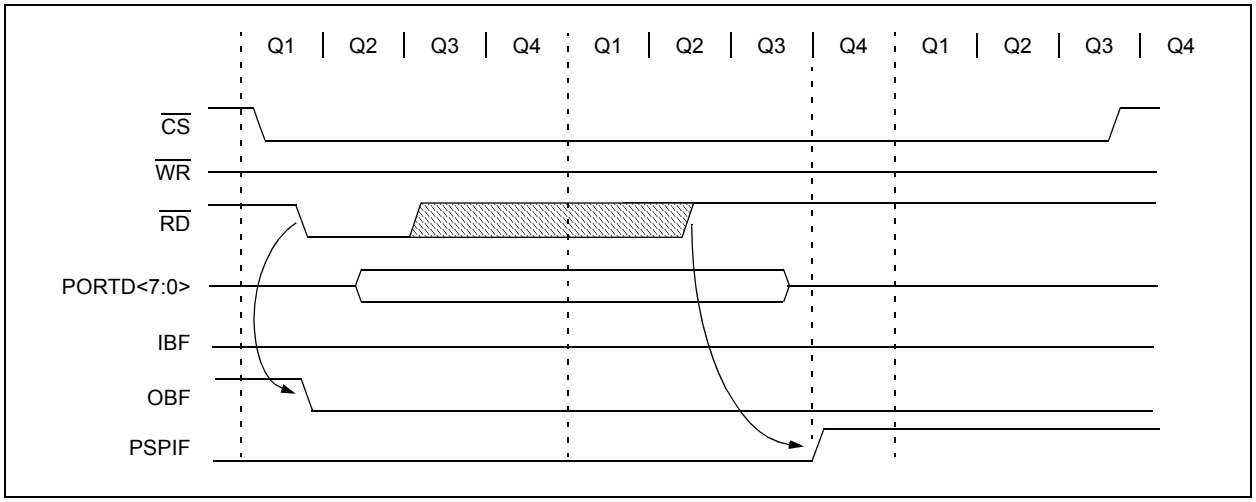


TABLE 11-21: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT

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	72
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	72
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	71
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	72
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	72
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	71
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	71
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71

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
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is NOT assigned; Timer0 clock input bypasses prescaler
0 = Timer0 prescaler is assigned; Timer0 clock input comes from prescaler output

bit 2-0 **T0PS<2:0>:** Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

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.

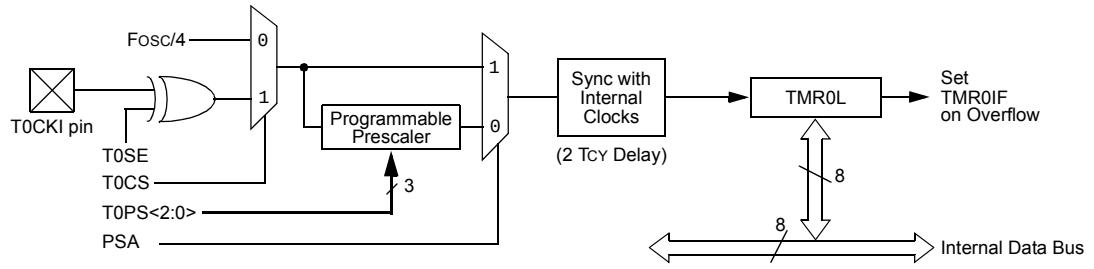
An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the internal phase clock (TOSC). There is a delay between synchronization and the onset of incrementing the timer/counter.

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 was valid, due to a rollover between successive reads of the high and low byte.

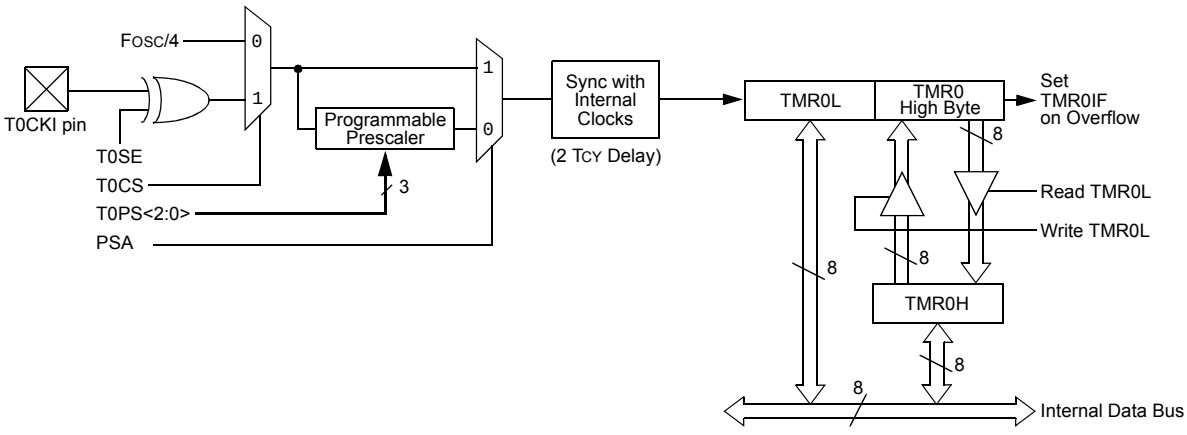
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

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								70
TMR0H	Timer0 Register High Byte								70
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	69
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	70
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	71

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

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 ECCP 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 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

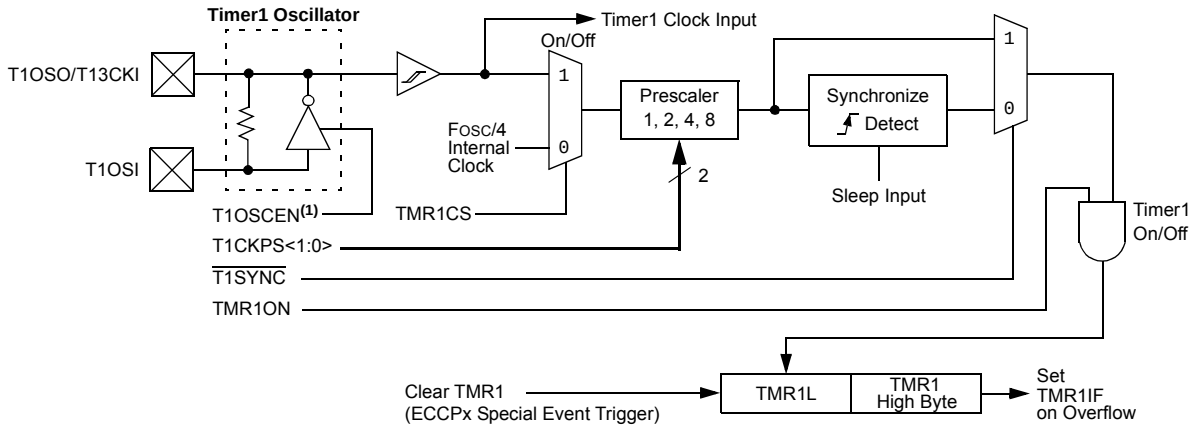
Timer1 can operate in one of these modes:

- 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.

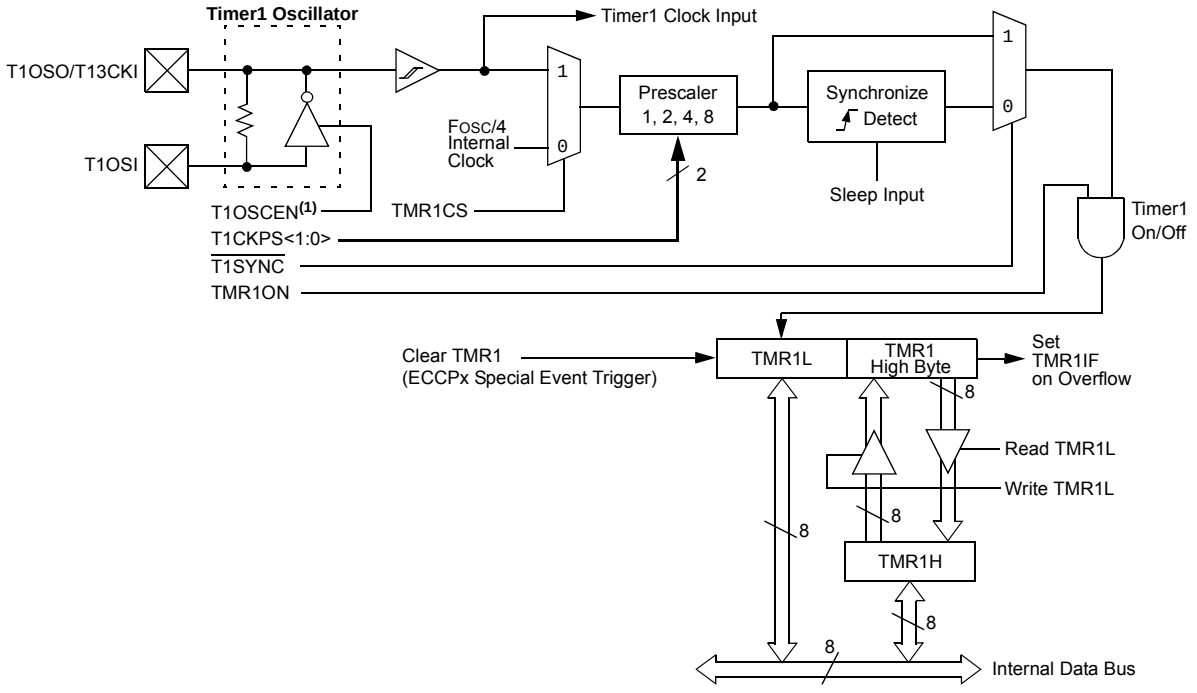
When Timer1 is enabled, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 13-1: TIMER1 BLOCK DIAGRAM



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.

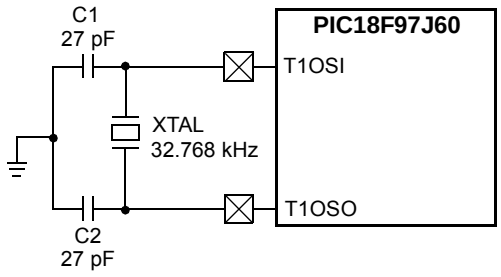
The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

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 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 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 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 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.

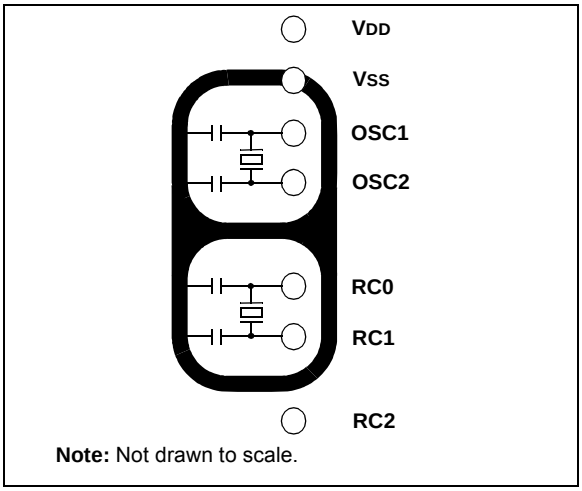
13.3.2 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

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 ECCP1 pin in Output Compare or PWM mode, or the primary oscillator using the OSC2 pin), a grounded guard ring around the oscillator circuit, as 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 ECCPx Special Event Trigger

If ECCP1 or ECCP2 is configured to use Timer1 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3. The trigger from ECCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 18.2.1 "Special Event Trigger"** for more information).

The module must be configured as either a timer or a synchronous counter to take advantage of this feature. When used this way, the 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 ECCPx 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"**) 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.

13.7 Considerations in Asynchronous Counter Mode

Following a Timer1 interrupt and an update to the TMR1 registers, the Timer1 module uses a falling edge on its clock source to trigger the next register update on the rising edge. If the update is completed after the clock input has fallen, the next rising edge will not be counted.

If the application can reliably update TMR1 before the timer input goes low, no additional action is needed. Otherwise, an adjusted update can be performed following a later Timer1 increment. This can be done by monitoring TMR1L within the interrupt routine until it increments, and then updating the TMR1H:TMR1L register pair while the clock is low, or one-half of the period of the clock source. Assuming that Timer1 is being used as a Real-Time Clock, the clock source is a 32.768 kHz crystal oscillator. In this case, one-half period of the clock is 15.25 μs.

The Real-Time Clock application code in Example 13-1 shows a typical ISR for Timer1, as well as the optional code required if the update cannot be done reliably within the required interval.

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
                                ; Insert the next 4 lines of code when TMR1
                                ; can not be reliably updated before clock pulse goes low
            BTFSC   TMR1L,0     ; wait for TMR1L to become clear
            BRA     $-2         ; (may already be clear)
            BTFSS   TMR1L,0     ; wait for TMR1L to become set
            BRA     $-2         ; TMR1 has just incremented
                                ; If TMR1 update can be completed before clock pulse goes low
                                ; Start ISR here
            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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
TMR1L	Timer1 Register Low Byte								70
TMR1H	Timer1 Register High Byte								70
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	70

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

14.0 TIMER2 MODULE

The Timer2 timer module incorporates the following features:

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

This 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 options 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 MSSPx modules operating in SPI mode. Additional information is provided in **Section 20.0 "Master Synchronous Serial Port (MSSP) Module"**.

FIGURE 14-1: TIMER2 BLOCK DIAGRAM

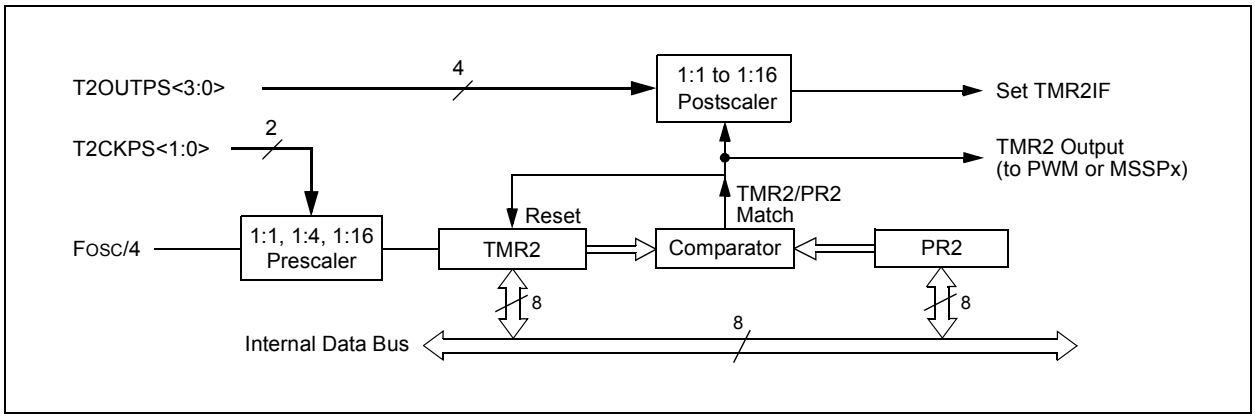


TABLE 14-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
TMR2	Timer2 Register								70
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	70
PR2	Timer2 Period Register								70

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

NOTES:

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/ECCPx 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 CCPx and ECCPx modules; see **Section 17.1.1 "CCPx/ECCPx Modules and Timer Resources"** 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/ECCPx Enable bits
11 = Timer3 and Timer4 are the clock sources for all CCPx/ECCPx modules
10 = Timer3 and Timer4 are the clock sources for ECCP3, CCP4 and CCP5;
Timer1 and Timer2 are the clock sources for ECCP1 and ECCP2
01 = Timer3 and Timer4 are the clock sources for ECCP2, ECCP3, CCP4 and CCP5;
Timer1 and Timer2 are the clock sources for ECCP1
00 = Timer1 and Timer2 are the clock sources for all CCPx/ECCPx 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 Select 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

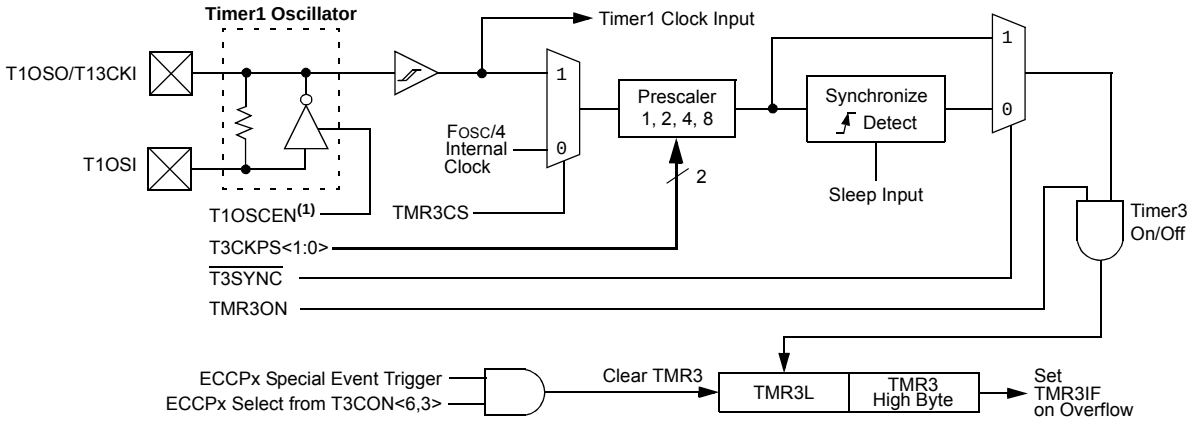
Timer3 can operate in one of three modes:

- 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.

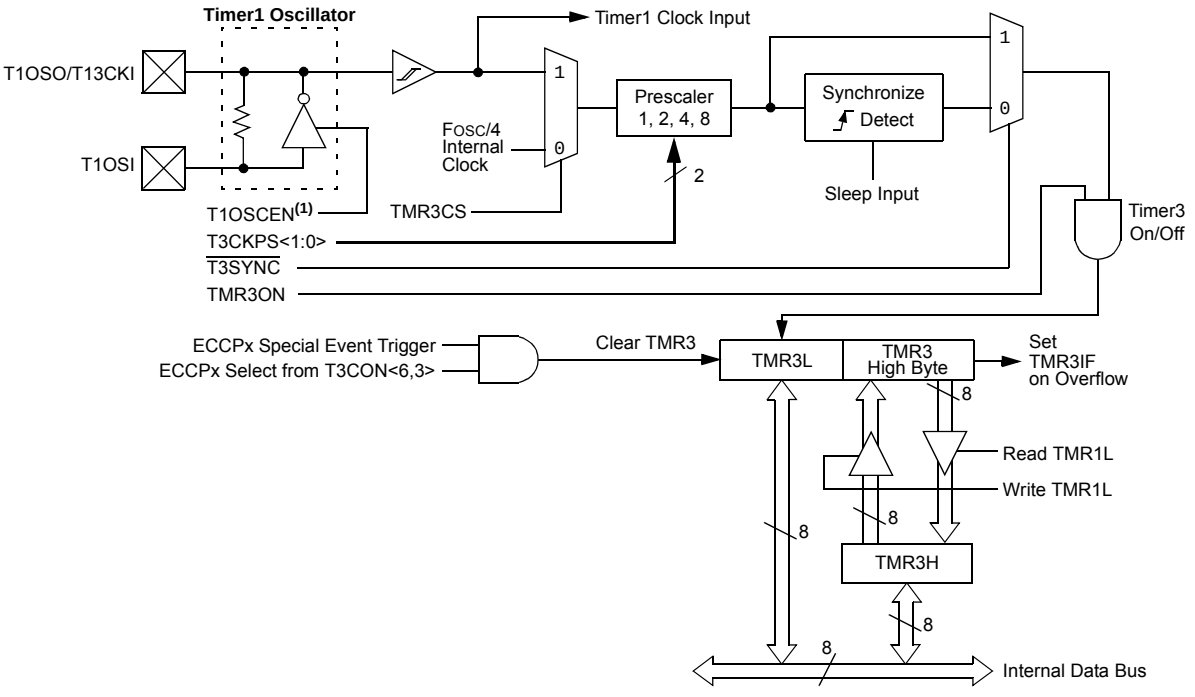
As with Timer1, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs when the Timer1 oscillator is enabled. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 15-1: TIMER3 BLOCK DIAGRAM



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 ECCPx Special Event Trigger

If ECCP1 or ECCP2 is configured to use Timer3 and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timer3. The trigger from ECCP2 will also start an A/D conversion if the A/D module is enabled (see **Section 18.2.1 "Special Event Trigger"** for more information).

The module must be configured as either a timer or synchronous counter to take advantage of this feature. When used this way, the CCPRxH:CCPRxL register pair effectively becomes a Period register for Timer3.

If Timer3 is running in Asynchronous Counter mode, the Reset operation may not work.

In the event that a write to Timer3 coincides with a Special Event Trigger from an ECCPx module, the write will take precedence.

Note:	The Special Event Triggers from the ECCPx 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	69
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
TMR3L	Timer3 Register Low Byte								70
TMR3H	Timer3 Register High Byte								70
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	70
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	71

Legend: — = unimplemented, read as '0', r = reserved. Shaded cells are not used by the Timer3 module.

NOTES:

16.0 TIMER4 MODULE

The Timer4 module has the following features:

- 8-Bit Timer register (TMR4)
- 8-Bit Period register (PR4)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR4 match of PR4

Timer4 has a control register, shown in Register 16-1. Timer4 can be shut off by clearing control bit, TMR4ON (T4CON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer4 is also controlled by this register. Figure 16-1 is a simplified block diagram of the Timer4 module.

16.1 Timer4 Operation

Timer4 can be used as the PWM time base for the PWM mode of the CCP module. The TMR4 register is readable and writable, and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T4CKPS<1:0> (T4CON<1:0>). The match output of TMR4 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR4 interrupt, latched in flag bit, TMR4IF (PIR3<3>).

The prescaler and postscaler counters are cleared when any of the following occurs:

- A write to the TMR4 register
- A write to the T4CON register
- Any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR4 is not cleared when T4CON is written.

REGISTER 16-1: T4CON: TIMER4 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
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **T4OUTPS<3:0>**: Timer4 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR4ON**: Timer4 On bit
1 = Timer4 is on
0 = Timer4 is off

bit 1-0 **T4CKPS<1:0>**: Timer4 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

16.2 Timer4 Interrupt

The Timer4 module has an 8-Bit Period register, PR4, which is both readable and writable. Timer4 increments from 00h until it matches PR4 and then resets to 00h on the next increment cycle. The PR4 register is initialized to FFh upon Reset.

16.3 Output of TMR4

The output of TMR4 (before the postscaler) is used only as a PWM time base for the CCPx/ECCPx modules. It is not used as a baud rate clock for the MSSPx modules as is the Timer2 output.

FIGURE 16-1: TIMER4 BLOCK DIAGRAM

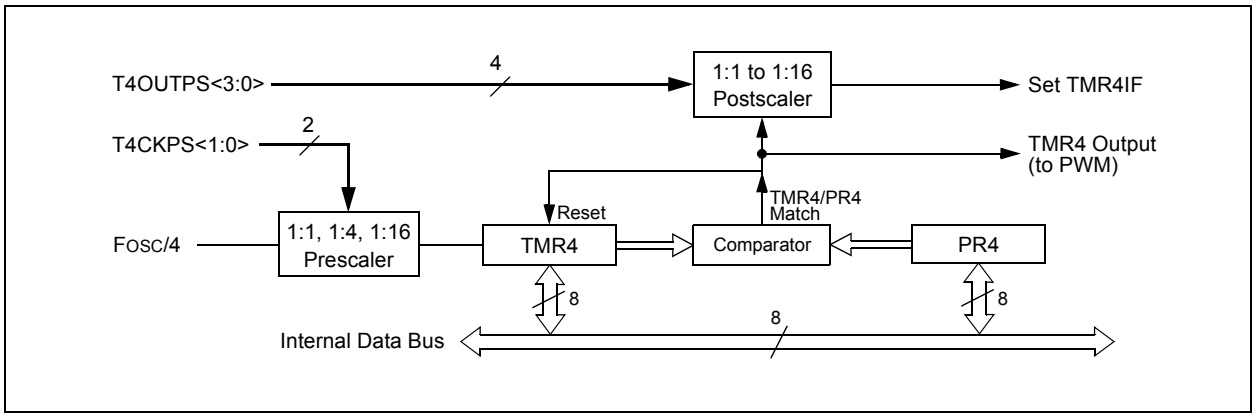


TABLE 16-1: REGISTERS ASSOCIATED WITH TIMER4 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
TMR4	Timer4 Register								72
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	72
PR4	Timer4 Period Register								72

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer4 module.

17.0 CAPTURE/COMPARE/PWM (CCP) MODULES

Members of the PIC18F97J60 family of devices all have a total of five CCP (Capture/Compare/PWM) modules. Two of these (CCP4 and CCP5) implement standard Capture, Compare and Pulse-Width Modulation (PWM) modes and are discussed in this section. The other three modules (ECCP1, ECCP2, ECCP3) implement standard Capture and Compare modes, as well as Enhanced PWM modes. These are discussed in **Section 18.0 "Enhanced Capture/Compare/PWM (ECCP) Modules"**.

Each CCPx/ECCPx 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 CCPx module operation in the following sections is described with respect to CCP4, but is equally applicable to CCP5.

Capture and Compare operations described in this chapter apply to all standard and Enhanced CCPx modules. The operations of PWM mode, described in **Section 17.4 "PWM Mode"**, apply to CCP4 and CCP5 only.

Note: Throughout this section and **Section 18.0 "Enhanced Capture/Compare/PWM (ECCP) Modules"**, references to register and bit names that may be associated with a specific CCP module are referred to generically by the use of 'x' or 'y' in place of the specific module number. Thus, "CCPxCON" might refer to the control register for ECCP1, ECCP2, ECCP3, CCP4 or CCP5.

REGISTER 17-1: CCPxCON: CCPx CONTROL REGISTER (CCP4 AND CCP5)

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>**: CCPx Module PWM Duty Cycle Bit 1 and Bit 0

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 (DCxB<9:2>) 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 = Reserved
11xx = PWM mode

17.1 CCPx 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.

17.1.1 CCPx/ECCPx MODULES AND TIMER RESOURCES

The CCPx/ECCPx modules utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode.

TABLE 17-1: CCPx/ECCPx MODE – TIMER RESOURCE

CCPx/ECCPx Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3
PWM	Timer2 or Timer4

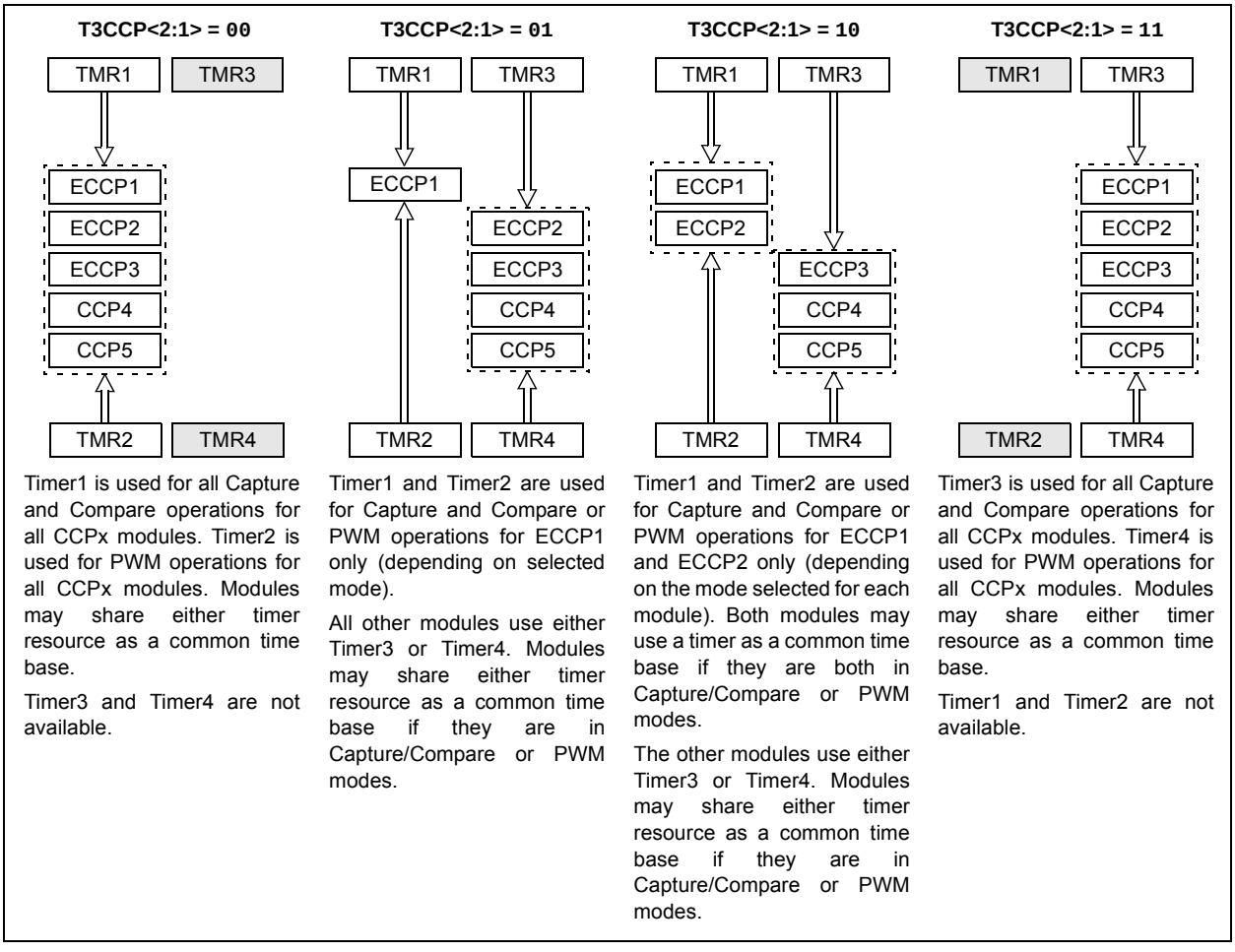
The assignment of a particular timer to a module is determined by the timer to CCPx enable bits in the T3CON register (Register 15-1, page 183). Depending on the configuration selected, up to four timers may be active at once, with modules in the same configuration (Capture/Compare or PWM) sharing timer resources. The possible configurations are shown in Figure 17-1.

17.1.2 ECCP2 PIN ASSIGNMENT

The pin assignment for ECCP2 (Capture input, Compare and PWM output) can change based on device configuration. The CCP2MX Configuration bit determines which pin ECCP2 is multiplexed to. By default, it is assigned to RC1 (CCP2MX = 1). If the Configuration bit is cleared, ECCP2 is multiplexed with RE7 on 80-pin and 100-pin devices in Microcontroller mode and RB3 on 100-pin devices in Extended Microcontroller mode.

Changing the pin assignment of ECCP2 does not automatically change any requirements for configuring the port pin. Users must always verify that the appropriate TRIS register is configured correctly for ECCP2 operation, regardless of where it is located.

FIGURE 17-1: CCPx/ECCPx AND TIMER INTERCONNECT CONFIGURATIONS



Timer1 is used for all Capture and Compare operations for all CCPx modules. Timer2 is used for PWM operations for all CCPx modules. Modules may share either timer resource as a common time base.

Timer3 and Timer4 are not available.

Timer1 and Timer2 are used for Capture and Compare or PWM operations for ECCP1 only (depending on selected mode).

All other modules use either Timer3 or Timer4. Modules may share either timer resource as a common time base if they are in Capture/Compare or PWM modes.

Timer1 and Timer2 are used for Capture and Compare or PWM operations for ECCP1 and ECCP2 only (depending on the mode selected for each module). Both modules may use a timer as a common time base if they are both in Capture/Compare or PWM modes.

The other modules use either Timer3 or Timer4. Modules may share either timer resource as a common time base if they are in Capture/Compare or PWM modes.

Timer3 is used for all Capture and Compare operations for all CCPx modules. Timer4 is used for PWM operations for all CCPx modules. Modules may share either timer resource as a common time base.

Timer1 and Timer2 are not available.

17.2 Capture Mode

In Capture mode, the CCPRxH:CCPRxL register pair captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on the corresponding CCPx pin. An event is defined as one of the following:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

The event is selected by the mode select bits, CCPxM<3:0> (CCPxCON<3:0>). When a capture is made, the interrupt request flag bit, CCPxIF, is set; it must be cleared in software. If another capture occurs before the value in register, CCPRx, is read, the old captured value is overwritten by the new captured value.

17.2.1 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 RG4/CCP5/P1D is configured as an output, a write to the port can cause a capture condition.

17.2.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (Timer1 and/or Timer3) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation will not work. The timer to be used with each CCPx module is selected in the T3CON register (see **Section 17.1.1 "CCPx/ECCPx Modules and Timer Resources"**).

17.2.3 SOFTWARE INTERRUPT

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

17.2.4 CCPx PRESCALER

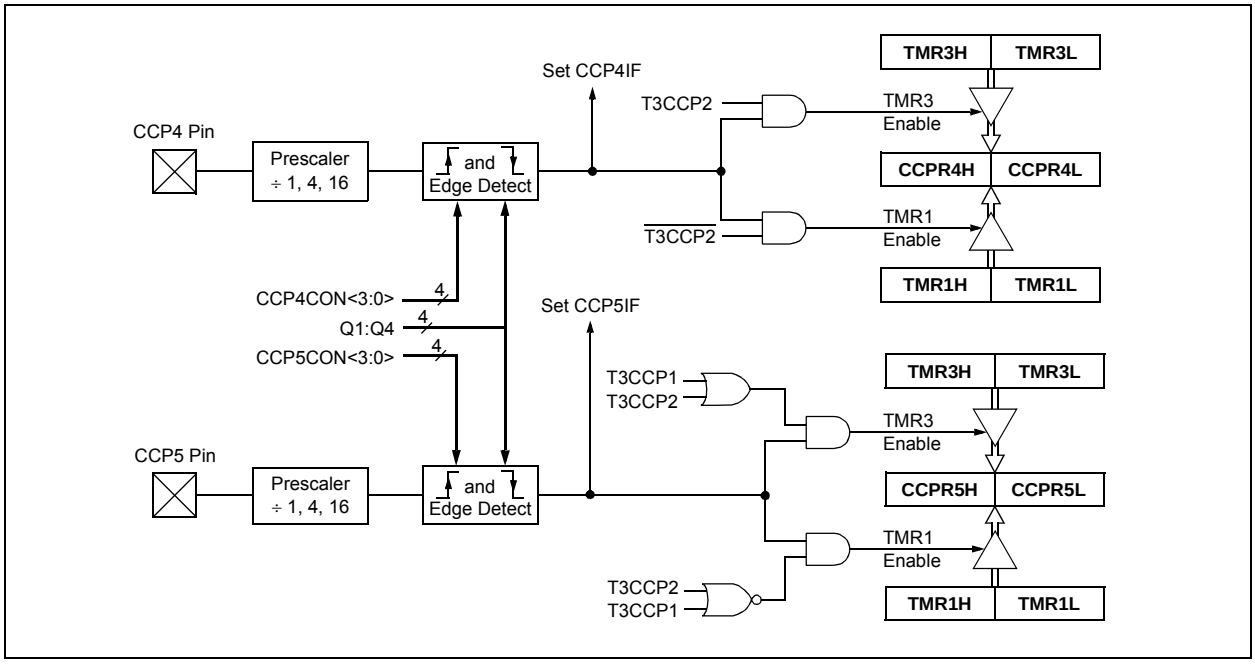
There are four prescaler settings in Capture mode. They are specified as part of the operating mode selected by the mode select bits (CCPxM<3:0>). Whenever the CCPx module is turned off or Capture mode is disabled, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared; therefore, the first capture may be from a non-zero prescaler. Example 17-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

EXAMPLE 17-1: CHANGING BETWEEN CAPTURE PRESCALERS (CCP5 SHOWN)

```
CLRF   CCP5CON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCP5CON      ; Load CCP5CON with
                    ; this value
```

FIGURE 17-2: CAPTURE MODE OPERATION BLOCK DIAGRAM



17.3 Compare Mode

In Compare mode, the 16-bit CCPRx register value is constantly compared against either the TMR1 or TMR3 register pair value. When a match occurs, the CCPx pin:

- Can be driven high
- Can be driven low
- Can be 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 (CCPxM<3:0>). At the same time, the interrupt flag bit, CCPxIF, is set.

17.3.1 CCPx PIN CONFIGURATION

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

Note: Clearing the CCP5CON register will force the RG4 compare output latch (depending on device configuration) to the default low level. This is not the PORTB or PORTC I/O data latch.

17.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode or Synchronized Counter mode if the CCPx module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

17.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCPxM<3:0> = 1010), the corresponding CCPx pin is not affected. Only a CCPx interrupt is generated, if enabled, and the CCPxIE bit is set.

FIGURE 17-3: COMPARE MODE OPERATION BLOCK DIAGRAM

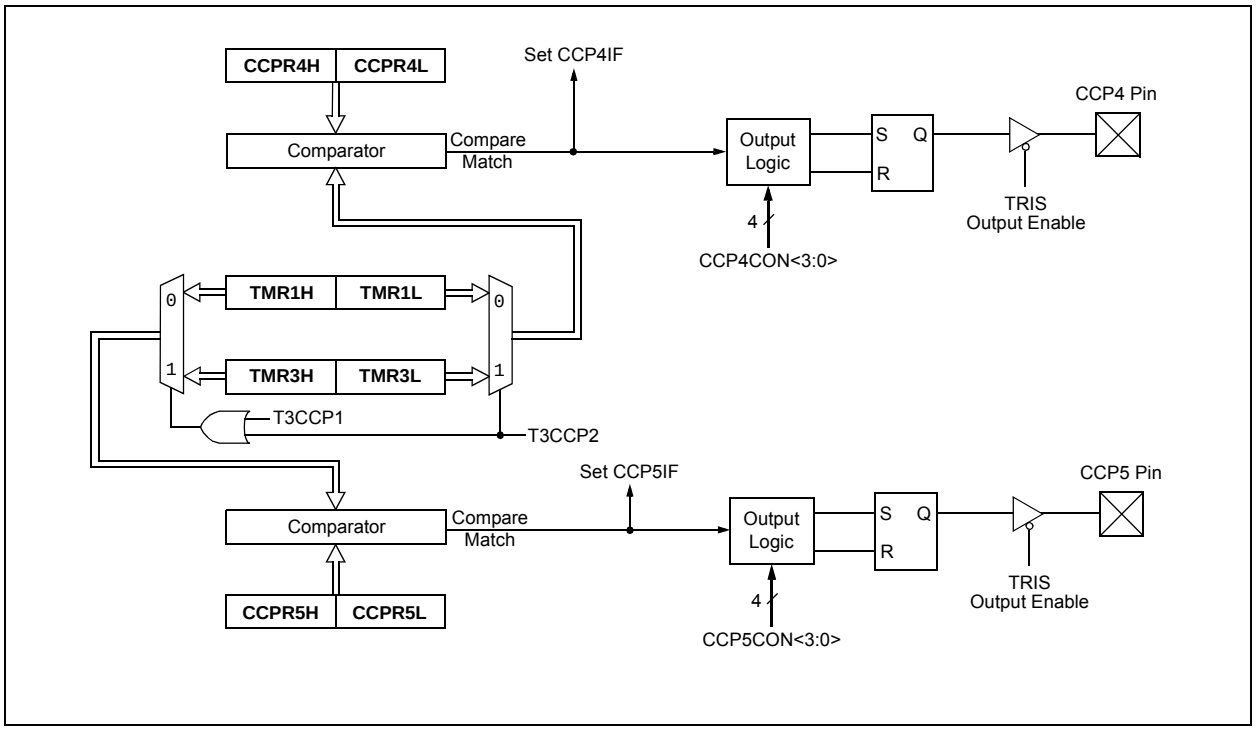


TABLE 17-2: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

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	69
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
TRISG	TRISG7	TRISG6	TRISG5	TRISG4	TRISG3(1)	TRISG2	TRISG1	TRISG0	71
TMR1L	Timer1 Register Low Byte								70
TMR1H	Timer1 Register High Byte								70
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	70
TMR3H	Timer3 Register High Byte								70
TMR3L	Timer3 Register Low Byte								70
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	71
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								72
CCPR4H	Capture/Compare/PWM Register 4 High Byte								72
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								73
CCPR5H	Capture/Compare/PWM Register 5 High Byte								73
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	73
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	73

Legend: — = unimplemented, read as '0', r = reserved. Shaded cells are not used by Capture/Compare, Timer1 or Timer3.

Note 1: This bit is only available in 80-pin and 100-pin devices; otherwise, it is unimplemented and reads as '0'.

17.4 PWM Mode

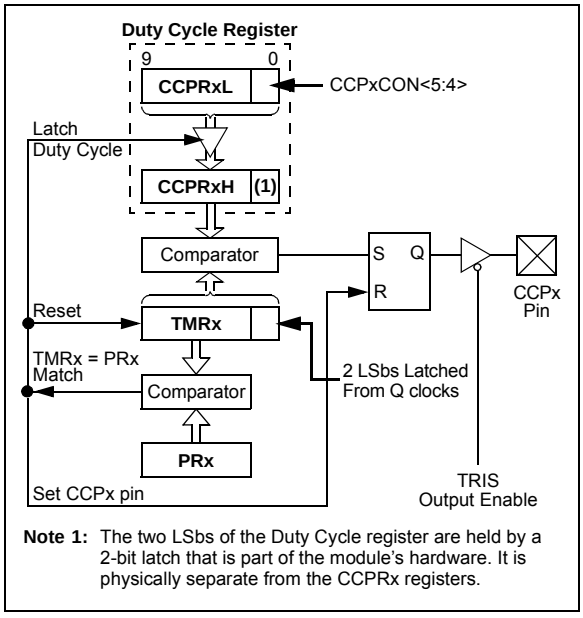
In Pulse-Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP4 and CCP5 pins are multiplexed with a PORTG data latch, the appropriate TRISG bit must be cleared to make the CCP4 or CCP5 pin an output.

Note:	Clearing the CCP4CON or CCP5CON register will force the RG3 or RG4 output latch (depending on device configuration) to the default low level. This is not the PORTG I/O data latch.

Figure 17-4 shows a simplified block diagram of the CCPx module in PWM mode.

For a step-by-step procedure on how to set up a CCPx module for PWM operation, see **Section 17.4.3 "Setup for PWM Operation"**.

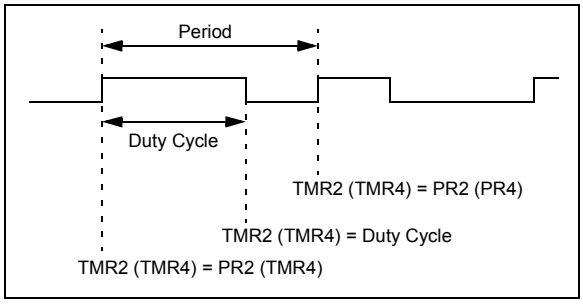
FIGURE 17-4: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: The two LSbs of the Duty Cycle register are held by a 2-bit latch that is part of the module's hardware. It is physically separate from the CCPRx registers.

A PWM output (Figure 17-5) 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 17-5: PWM OUTPUT



17.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 (PR4) register. The PWM period can be calculated using Equation 17-1:

EQUATION 17-1:

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

PWM frequency is defined as 1/[PWM period].

When TMR2 (TMR4) is equal to PR2 (PR4), the following three events occur on the next increment cycle:

- TMR2 (TMR4) is cleared
- The CCPx pin is set (exception: if PWM duty cycle = 0%, the CCPx pin will not be set)
- The PWM duty cycle is latched from CCPRxL into CCPRxH

Note:	The Timer2 and Timer4 postscalers (see **Section 14.0 "Timer2 Module"** and **Section 16.0 "Timer4 Module"**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

17.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPRxL register and to the CCPxCON<5:4> bits. Up to 10-bit resolution is available. The CCPRxL contains the eight MSbs and the CCPxCON<5:4> contains the two LSbs. This 10-bit value is represented by CCPRxL:CCPxCON<5:4>. Equation 17-2 is used to calculate the PWM duty cycle in time.

EQUATION 17-2:

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

CCPRxL and CCPxCON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPRxH until after a match between PR2 (PR4) and TMR2 (TMR4) occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

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

When the CCPRxH and 2-bit latch match TMR2 (TMR4), concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 (TMR4) prescaler, the CCPx pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by Equation 17-3:

EQUATION 17-3:

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

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

17.4.3 SETUP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 (PR4) register.
2. Set the PWM duty cycle by writing to the CCPRxL register and CCPxCON<5:4> bits.
3. Make the CCPx pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 (TMR4) prescale value, then enable Timer2 (Timer4) by writing to T2CON (T4CON).
5. Configure the CCPx module for PWM operation.

TABLE 17-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

TABLE 17-4: REGISTERS ASSOCIATED WITH PWM, TIMER2 AND TIMER4

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
TRISG	TRISG7	TRISG6	TRISG5	TRISG4	TRISG3(1)	TRISG2	TRISG1	TRISG0	71
TMR2	Timer2 Register								70
PR2	Timer2 Period Register								70
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	70
TMR4	Timer4 Register								72
PR4	Timer4 Period Register								72
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	72
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								72
CCPR4H	Capture/Compare/PWM Register 4 High Byte								72
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								73
CCPR5H	Capture/Compare/PWM Register 5 High Byte								73
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	73
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	73

Legend: — = unimplemented, read as '0'. Shaded cells are not used by PWM, Timer2 or Timer4.

Note 1: This bit is only available in 80-pin and 100-pin devices; otherwise, it is unimplemented and reads as '0'.

18.0 ENHANCED CAPTURE/COMPARE/PWM (ECCP) MODULES

In the PIC18F97J60 family of devices, three of the CCP modules are implemented as standard CCP modules with Enhanced PWM capabilities. These include the provision for 2 or 4 output channels, user-selectable polarity, dead-band control and automatic shutdown and restart. The Enhanced features are discussed in detail in **Section 18.4 "Enhanced PWM Mode"**. Capture, Compare and single output PWM functions of the ECCPx modules are the same as described for the standard CCPx modules.

The control register for the Enhanced CCPx module is shown in Register 18-1. It differs from the CCP4CON/CCP5CON registers in that the two Most Significant bits are implemented to control PWM functionality.

In addition to the expanded range of modes available through the Enhanced CCPxCON register, the ECCPx modules each have two additional registers associated with Enhanced PWM operation and auto-shutdown features. They are:

- ECCPxDEL (Dead-Band Delay)
- ECCPxAS (Auto-Shutdown Configuration)

REGISTER 18-1: CCPxCON: ENHANCED CCPx CONTROL REGISTER (ECCP1/ECCP2/ECCP3)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

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

bit 7-6 **PxM<1:0>:** Enhanced PWM Output Configuration bits

If CCPxM<3:2> = 00, 01, 10:

xx = PxA assigned as Capture/Compare input/output; PxB, PxC, PxD assigned as port pins

If CCPxM<3:2> = 11:

00 = Single output: PxA modulated; PxB, PxC, PxD assigned as port pins
01 = Full-bridge output forward: P1D modulated; P1A active; P1B, P1C inactive
10 = Half-bridge output: P1A, P1B modulated with dead-band control; P1C, P1D assigned as port pins
11 = Full-bridge output reverse: P1B modulated; P1C active; P1A, P1D inactive

bit 5-4 **DCxB<1:0>**: ECCPx Module PWM Duty Cycle Bit 1 and Bit 0

Capture mode:

Unused.

Compare mode:

Unused.

PWM mode:

These bits are the 2 LSbs of the 10-bit PWM duty cycle. The 8 MSbs of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>**: ECCPx Module Mode Select bits

0000 = Capture/Compare/PWM disabled (resets ECCPx module)
0001 = Reserved
0010 = Compare mode; toggle output on match
0011 = Capture mode
0100 = Capture mode; every falling edge
0101 = Capture mode; every rising edge
0110 = Capture mode; every 4th rising edge
0111 = Capture mode; every 16th rising edge
1000 = Compare mode; initialize ECCPx pin low; set output on compare match (set CCPxIF)
1001 = Compare mode; initialize ECCPx pin high; clear output on compare match (set CCPxIF)
1010 = Compare mode; generate software interrupt only, ECCPx pin reverts to I/O state)
1011 = Compare mode; trigger special event (ECCPx resets TMR1 or TMR3, sets CCPxIF bit, ECCPx trigger also starts A/D conversion if A/D module is enabled)**(1)**
1100 = PWM mode; PxA, PxC active-high; PxB, PxD active-high
1101 = PWM mode; PxA, PxC active-high; PxB, PxD active-low
1110 = PWM mode; PxA, PxC active-low; PxB, PxD active-high
1111 = PWM mode; PxA, PxC active-low; PxB, PxD active-low

Note 1: Implemented only for ECCP1 and ECCP2; same as '1010' for ECCP3.

18.1 ECCPx Outputs and Configuration

Each of the Enhanced CCPx modules may have up to four PWM outputs, depending on the selected operating mode. These outputs, designated PxA through PxD, are multiplexed with various I/O pins. Some ECCPx pin assignments are constant, while others change based on device configuration. For those pins that do change, the controlling bits are:

- CCP2MX Configuration bit (80-pin and 100-pin devices only)
- ECCPMX Configuration bit (80-pin and 100-pin devices only)
- Program memory operating mode set by the EMB Configuration bits (100-pin devices only)

The pin assignments for the Enhanced CCPx modules are summarized in Table 18-1, Table 18-2 and Table 18-3. To configure the I/O pins as PWM outputs, the proper PWM mode must be selected by setting the PxMx and CCPxMx bits (CCPxCON<7:6> and <3:0>, respectively). The appropriate TRIS direction bits for the corresponding port pins must also be set as outputs.

18.1.1 ECCP1/ECCP3 OUTPUTS AND PROGRAM MEMORY MODE

In 100-pin devices, the use of Extended Microcontroller mode has an indirect effect on the ECCP1 and ECCP3 pins in Enhanced PWM modes. By default, PWM outputs, P1B/P1C and P3B/P3C, are multiplexed to PORTE pins, along with the high-order byte of the external memory bus. When the bus is active in Extended Microcontroller mode, it overrides the Enhanced CCPx outputs and makes them unavailable. Because of this, ECCP1 and ECCP3 can only be used in compatible (single output) PWM modes when the device is in Extended Microcontroller mode with default pin configuration.

An exception to this configuration is when a 12-bit address width is selected for the external bus (EMB<1:0> Configuration bits = 10). In this case, the upper pins of PORTE continue to operate as digital I/O, even when the external bus is active. P1B/P1C and P3B/P3C remain available for use as Enhanced PWM outputs.

If an application requires the use of additional PWM outputs during Extended Microcontroller mode, the P1B/P1C and P3B/P3C outputs can be reassigned to the upper bits of PORTH. This is done by clearing the ECCPMX Configuration bit.

18.1.2 ECCP2 OUTPUTS AND PROGRAM MEMORY MODES

For 100-pin devices, the Program Memory mode of the device (**Section 6.1.3 "PIC18F9XJ60/9XJ65 Program Memory Modes"**) also impacts pin multiplexing for the module.

The ECCP2 input/output (ECCP2/P2A) can be multiplexed to one of three pins. The default assignment (CCP2MX Configuration bit is set) for all devices is RC1. Clearing CCP2MX reassigns ECCP2/P2A to RE7 in 80-pin and 100-pin devices.

An additional option exists for 100-pin devices. When these devices are operating in Microcontroller mode, the multiplexing options described above still apply. In Extended Microcontroller mode, clearing CCP2MX reassigns ECCP2/P2A to RB3.

18.1.3 USE OF CCP4 AND CCP5 WITH ECCP1 AND ECCP3

Only the ECCP2 module has four dedicated, output pins that are available for use. Assuming that the I/O ports or other multiplexed functions on those pins are not needed, they may be used without interfering with any other CCPx module.

ECCP1 and ECCP3, on the other hand, only have three dedicated output pins: ECCPx/PxA, PxB and PxC. Whenever these modules are configured for Quad PWM mode, the pin normally used for CCP4 or CCP5 becomes the PxD output pin for ECCP3 and ECCP1, respectively. The CCP4 and CCP5 modules remain functional but their outputs are overridden.

18.1.4 ECCPx MODULES AND TIMER RESOURCES

Like the standard CCPx modules, the ECCPx modules can utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available for modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode. Additional details on timer resources are provided in **Section 17.1.1 "CCPx/ECCPx Modules and Timer Resources"**.

TABLE 18-1: PIN CONFIGURATIONS FOR ECCP1

ECCP Mode	CCP1CON Configuration	RC2	RD0 or RE6(1)	RE5	RG4	RH7(2)	RH6(2)
64-Pin Devices; 80-Pin Devices, ECCPMX = 1; 100-Pin Devices, ECCPMX = 1, Microcontroller mode or Extended Microcontroller mode with 12-Bit Address Width:							
Compatible CCP	00xx 11xx	ECCP1	RD0/RE6	RE5	RG4/CCP5	RH7/AN15	RH6/AN14
Dual PWM	10xx 11xx	P1A	P1B	RE5	RG4/CCP5	RH7/AN15	RH6/AN14
Quad PWM	x1xx 11xx	P1A	P1B	P1C	P1D	RH7/AN15	RH6/AN14
80-Pin Devices, ECCPMX = 0; 100-Pin Devices, ECCPMX = 0, All Program Memory modes:							
Compatible CCP	00xx 11xx	ECCP1	RD0/RE6	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
Dual PWM	10xx 11xx	P1A	RD0/RE6	RE5/AD13	RG4/CCP5	P1B	RH6/AN14
Quad PWM(3)	x1xx 11xx	P1A	RD0/RE6	RE5/AD13	P1D	P1B	P1C
100-Pin Devices, ECCPMX = 1, Extended Microcontroller mode with 16-Bit or 20-Bit Address Width:							
Compatible CCP	00xx 11xx	ECCP1	RD0/RE6	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14

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

Note 1: P1B is multiplexed with RD0 in 64-pin devices, and RE6 on 80-pin and 100-pin devices.

2: These pin options are not available in 64-pin devices.

3: With ECCP1 in Quad PWM mode, the CCP5 pin's output is overridden by P1D; otherwise, CCP5 is fully operational.

TABLE 18-2: PIN CONFIGURATIONS FOR ECCP2

ECCP Mode	CCP2CON Configuration	RB3	RC1	RE7	RE2	RE1	RE0
All Devices, CCP2MX = 1, All Program Memory modes:							
Compatible CCP	00xx 11xx	RB3/INT3	ECCP2	RE7	RE2	RE1	RE0
Dual PWM	10xx 11xx	RB3/INT3	P2A	RE7	P2B	RE1	RE0
Quad PWM	x1xx 11xx	RB3/INT3	P2A	RE7	P2B	P2C	P2D
80-Pin and 100-Pin Devices, CCP2MX = 0, Microcontroller mode:							
Compatible CCP	00xx 11xx	RB3/INT3	RC1/T1OS1	ECCP2	RE2	RE1	RE0
Dual PWM	10xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	RE1	RE0
Quad PWM	x1xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	P2C	P2D
100-Pin Devices, CCP2MX = 0, Extended Microcontroller mode:							
Compatible CCP	00xx 11xx	ECCP2	RC1/T1OS1	RE7/AD15	RE2/$\overline{CS}$	RE1/$\overline{WR}$	RE0/$\overline{RD}$
Dual PWM	10xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	RE1/$\overline{WR}$	RE0/$\overline{RD}$
Quad PWM	x1xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	P2C	P2D

Legend: x = Don't care. Shaded cells indicate pin assignments not used by ECCP2 in a given mode.

TABLE 18-3: PIN CONFIGURATIONS FOR ECCP3

ECCP Mode	CCP3CON Configuration	RD1 or RG0(1)	RE4	RE3	RD2 or RG3(1)	RH5(2)	RH4(2)
64-Pin Devices; 80-Pin Devices, ECCPMX = 1; 100-Pin Devices, ECCPMX = 1, Microcontroller mode:							
Compatible CCP	00xx 11xx	ECCP3	RE4	RE3	RD2/RG3	RH5/AN13	RH4/AN12
Dual PWM	10xx 11xx	P3A	P3B	RE3	RD2/RG3	RH5/AN13	RH4/AN12
Quad PWM	x1xx 11xx	P3A	P3B	P3C	P3D	RH5/AN13	RH4/AN12
80-Pin Devices, ECCPMX = 0; 100-Pin Devices, ECCPMX = 0, All Program Memory modes:							
Compatible CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RD2/RG3	RH5/AN13	RH4/AN12
Dual PWM	10xx 11xx	P3A	RE6/AD14	RE5/AD13	RD2/RG3	P3B	RH4/AN12
Quad PWM(3)	x1xx 11xx	P3A	RE6/AD14	RE5/AD13	P3D	P3B	P3C
100-Pin Devices, ECCPMX = 1, Extended Microcontroller with 12-Bit Address Width:							
Compatible CCP	00xx 11xx	ECCP3	RE4/AD12	RE3/AD11	RD2/RG3	RH5/AN13	RH4/AN12
Dual PWM	10xx 11xx	P3A	P3B	RE3/AD11	RD2/RG3	RH5/AN13	RH4/AN12
100-Pin Devices, ECCPMX = 1, Extended Microcontroller mode with 16-Bit or 20-Bit Address Width:							
Compatible CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RD2/RG3	RH5/AN13	RH4/AN12

Legend: x = Don't care. Shaded cells indicate pin assignments not used by ECCP3 in a given mode.

Note 1: ECCP3/P3A and CCP4/P3D are multiplexed with RD1 and RD2 in 64-pin devices, and RG0 and RG3 in 80-pin and 100-pin devices.

2: These pin options are not available in 64-pin devices.

3: With ECCP3 in Quad PWM mode, the CCP4 pin's output is overridden by P3D; otherwise, CCP4 is fully operational.

18.2 Capture and Compare Modes

Except for the operation of the Special Event Trigger discussed below, the Capture and Compare modes of the ECCPx modules are identical in operation to that of CCP4. These are discussed in detail in **Section 17.2 "Capture Mode"** and **Section 17.3 "Compare Mode"**.

18.2.1 SPECIAL EVENT TRIGGER

ECCP1 and ECCP2 incorporate an internal hardware trigger that is generated in Compare mode on a match between the CCPRx register pair and the selected timer. This can be used, in turn, to initiate an action. This mode is selected by setting CCPxCON<3:0> to '1011'.

The Special Event Trigger output of either ECCP1 or ECCP2 resets the TMR1 or TMR3 register pair, depending on which timer resource is currently selected. This allows the CCPRx register to effectively be a 16-Bit Programmable Period register for Timer1 or Timer3. In addition, the ECCP2 Special Event Trigger will also start an A/D conversion if the A/D module is enabled.

Special Event Triggers are not implemented for ECCP3, CCP4 or CCP5. Selecting the Special Event Trigger mode for these modules has the same effect as selecting the Compare with Software Interrupt mode (CCPxM<3:0> = 1010).

Note:	The Special Event Trigger from ECCP2 will not set the Timer1 or Timer3 interrupt flag bits.

18.3 Standard PWM Mode

When configured in Single Output mode, the ECCPx modules function identically to the standard CCPx modules in PWM mode, as described in **Section 17.4 "PWM Mode"**. Sometimes this is also referred to as "Compatible CCP" mode, as in Tables 18-1 through 18-3.

Note:	When setting up single output PWM operations, users are free to use either of the processes described in **Section 17.4.3 "Setup for PWM Operation"** or **Section 18.4.9 "Setup for PWM Operation"**. The latter is more generic but will work for either single or multi-output PWM.

18.4 Enhanced PWM Mode

The Enhanced PWM mode provides additional PWM output options for a broader range of control applications. The module is a backward compatible version of the standard CCPx modules and offers up to four outputs, designated PxA through PxD. Users are also able to select the polarity of the signal (either active-high or active-low). The module's output mode and polarity are configured by setting the PxM<1:0> and CCPxM<3:0> bits of the CCPxCON register (CCPxCON<7:6> and <3:0>, respectively).

For the sake of clarity, Enhanced PWM mode operation is described generically throughout this section with respect to ECCP1 and TMR2 modules. Control register names are presented in terms of ECCP1. All three Enhanced modules, as well as the two timer resources, can be used interchangeably and function identically. TMR2 or TMR4 can be selected for PWM operation by selecting the proper bits in T3CON.

Figure 18-1 shows a simplified block diagram of PWM operation. All control registers are double-buffered and are loaded at the beginning of a new PWM cycle (the period boundary when Timer2 resets) in order to prevent glitches on any of the outputs. The exception is the ECCP1 Dead-Band Delay register, ECCP1DEL, which is loaded at either the duty cycle boundary or the boundary period (whichever comes first). Because of the buffering, the module waits until the assigned timer resets instead of starting immediately. This means that Enhanced PWM waveforms do not exactly match the standard PWM waveforms, but are instead, offset by one full instruction cycle (4 TOSC).

As before, the user must manually configure the appropriate TRIS bits for output.

18.4.1 PWM PERIOD

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

EQUATION 18-1:

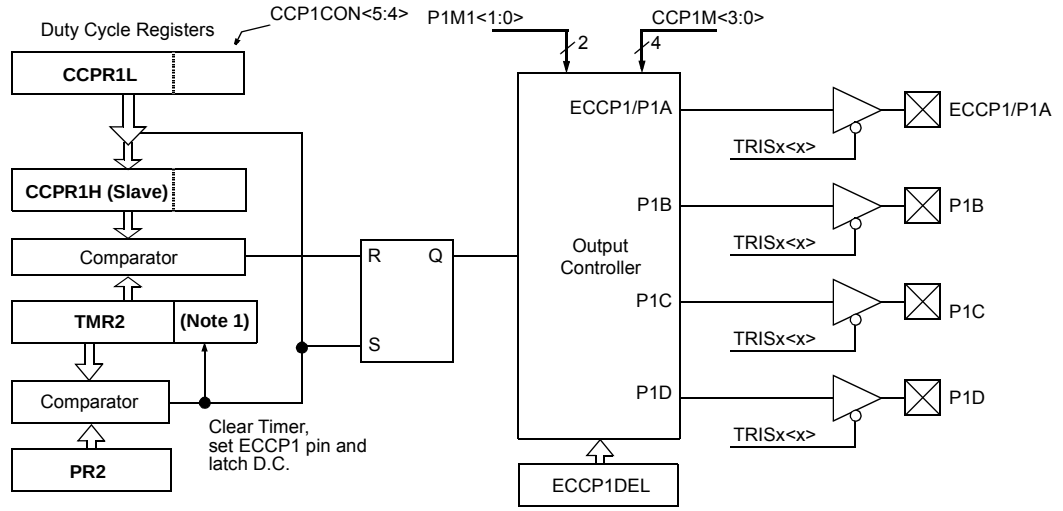
$$\text{PWM Period} = [(\text{PR2}) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

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 ECCP1 pin is set (if PWM duty cycle = 0%, the ECCP1 pin will not be set)
- The PWM duty cycle is copied from CCPR1L into CCPR1H

Note: The Timer2 postscaler (see **Section 14.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

FIGURE 18-1: SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODULE



Note: The 8-bit timer TMR2 register is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

18.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> bits contain the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The PWM duty cycle is calculated by the equation:

EQUATION 18-2:

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

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not copied into CCPR1H until a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation. When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or two bits of the TMR2 prescaler, the ECCP1 pin is cleared. The maximum PWM resolution (bits) for a given PWM frequency is given by the following equation:

EQUATION 18-3:

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

Note: If the PWM duty cycle value is longer than the PWM period, the ECCP1 pin will not be cleared.

18.4.3 PWM OUTPUT CONFIGURATIONS

The P1M<1:0> bits in the CCP1CON register allow one of four configurations:

- Single Output
- Half-Bridge Output
- Full-Bridge Output, Forward mode
- Full-Bridge Output, Reverse mode

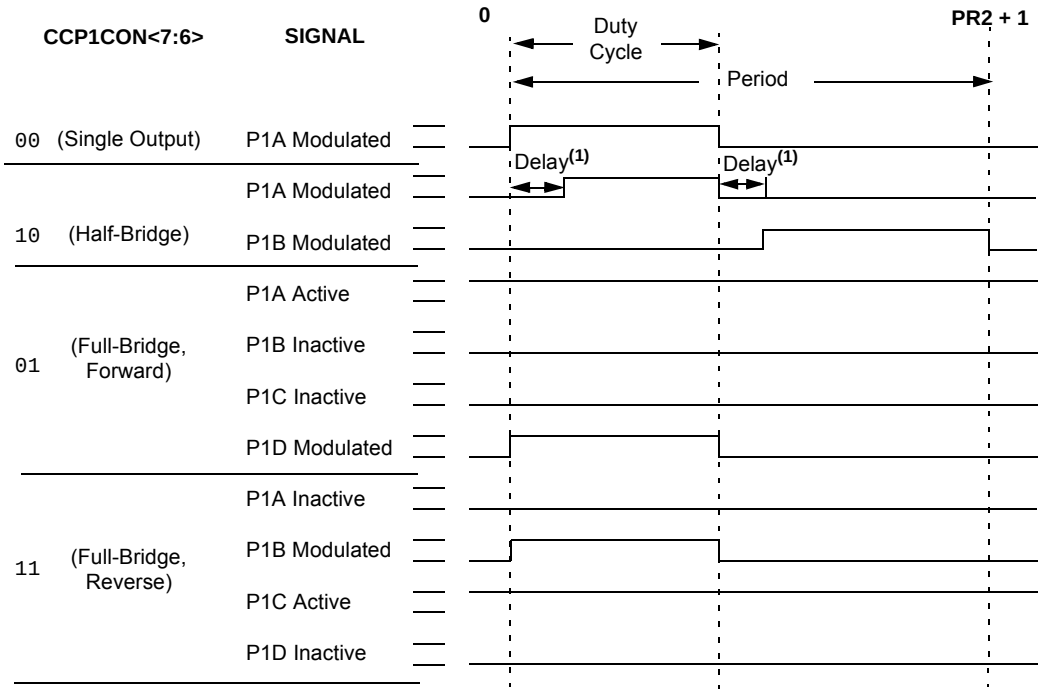
The Single Output mode is the standard PWM mode discussed in **Section 18.4 "Enhanced PWM Mode"**. The Half-Bridge and Full-Bridge Output modes are covered in detail in the sections that follow.

The general relationship of the outputs in all configurations is summarized in Figure 18-2.

TABLE 18-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

FIGURE 18-2: PWM OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)

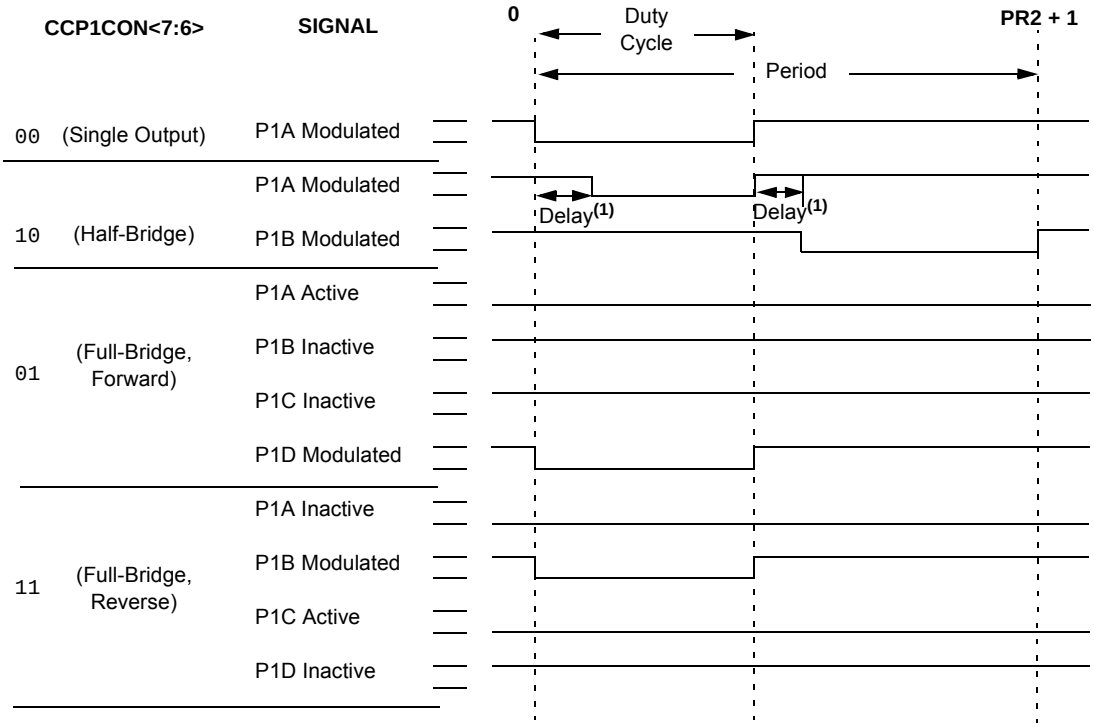


Relationships:

- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Duty Cycle = TOSC * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (ECCP1DEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCP1DEL register (**Section 18.4.6 "Programmable Dead-Band Delay"**).

FIGURE 18-3: PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)



Relationships:

- Period = 4 * Tosc * (PR2 + 1) * (TMR2 Prescale Value)
- Duty Cycle = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * Tosc * (ECCP1DEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCP1DEL register (**Section 18.4.6 "Programmable Dead-Band Delay"**).

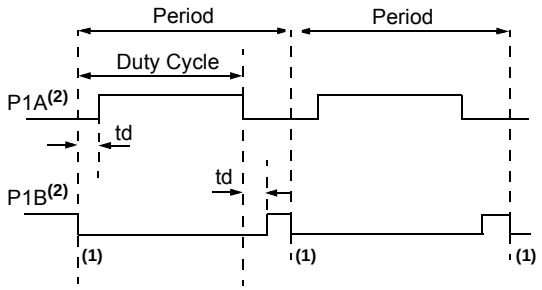
18.4.4 HALF-BRIDGE MODE

In the Half-Bridge Output mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the P1A pin, while the complementary PWM output signal is output on the P1B pin (Figure 18-4). This mode can be used for half-bridge applications, as shown in Figure 18-5, or for full-bridge applications, where four power switches are being modulated with two PWM signals.

In Half-Bridge Output mode, the programmable dead-band delay can be used to prevent shoot-through current in half-bridge power devices. The value of bits, P1DC<6:0>, sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. See **Section 18.4.6 "Programmable Dead-Band Delay"** for more details on dead-band delay operations.

Since the P1A and P1B outputs are multiplexed with the PORTC<2> and PORTE<6> data latches, the TRISC<2> and TRISE<6> bits must be cleared to configure P1A and P1B as outputs.

FIGURE 18-4: HALF-BRIDGE PWM OUTPUT



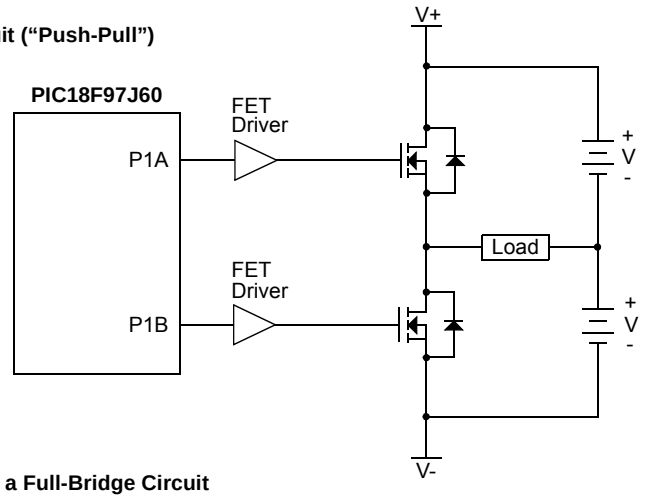
td = Dead Band Delay

Note 1: At this time, the TMR2 register is equal to the PR2 register.

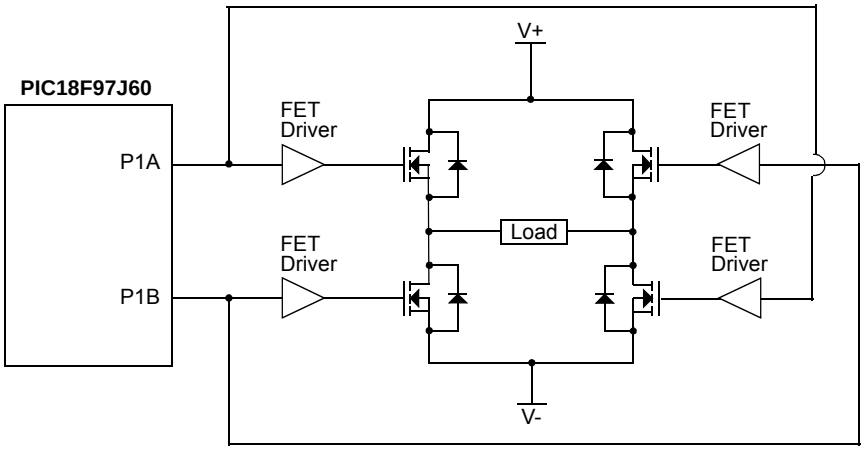
2: Output signals are shown as active-high.

FIGURE 18-5: EXAMPLES OF HALF-BRIDGE OUTPUT MODE APPLICATIONS

Standard Half-Bridge Circuit ("Push-Pull")



Half-Bridge Output Driving a Full-Bridge Circuit

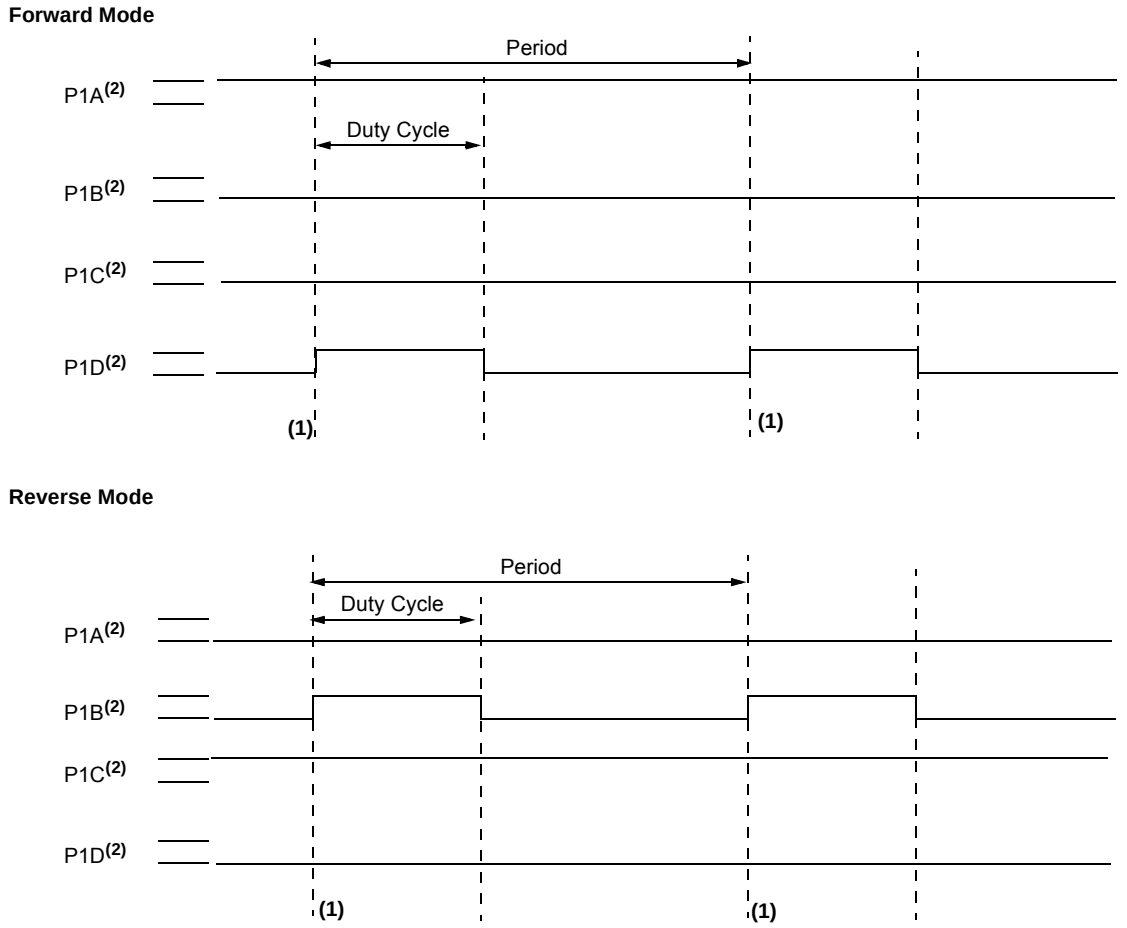


18.4.5 FULL-BRIDGE MODE

In Full-Bridge Output mode, four pins are used as outputs; however, only two outputs are active at a time. In the Forward mode, pin, P1A, is continuously active and pin, P1D, is modulated. In the Reverse mode, pin, P1C, is continuously active and pin, P1B, is modulated. These are illustrated in Figure 18-6.

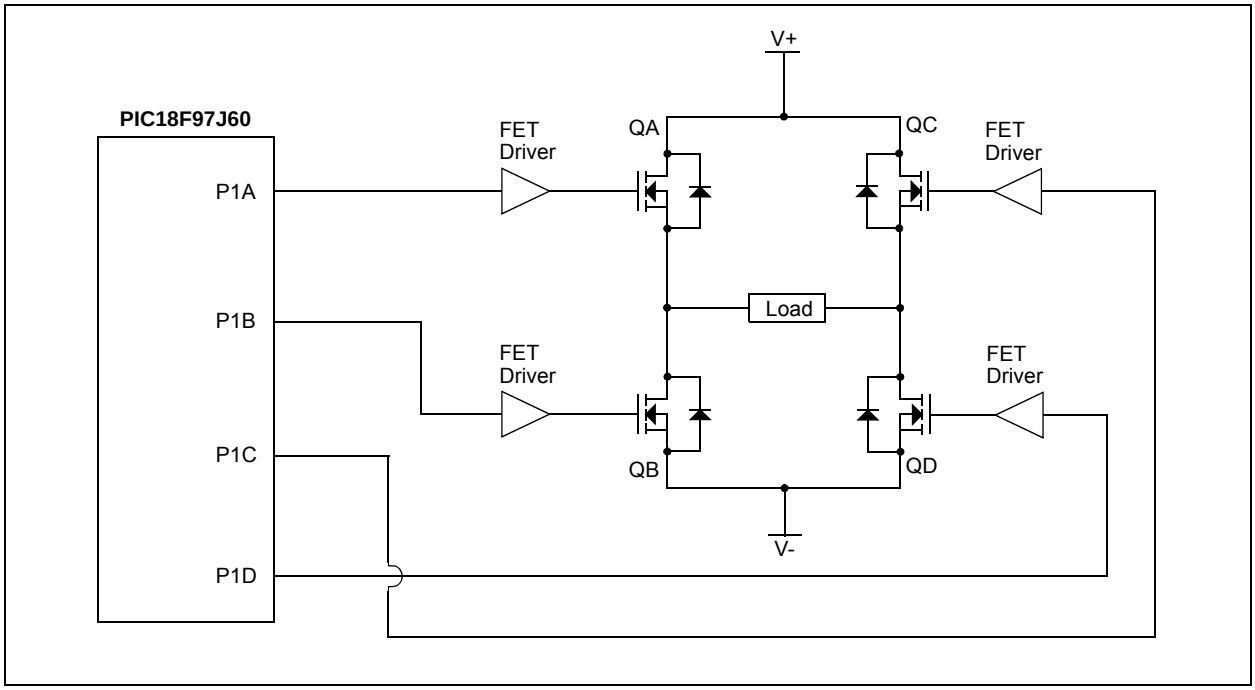
P1A, P1B, P1C and P1D outputs are multiplexed with the data latches of the port pins listed in Table 18-1 and Table 18-3. The corresponding TRIS bits must be cleared to make the P1A, P1B, P1C and P1D pins outputs.

FIGURE 18-6: FULL-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.
2: Output signal is shown as active-high.

FIGURE 18-7: EXAMPLE OF FULL-BRIDGE APPLICATION



18.4.5.1 Direction Change in Full-Bridge Mode

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

Just before the end of the current PWM period, the modulated outputs (P1B and P1D) are placed in their inactive state, while the unmodulated outputs (P1A and P1C) are switched to drive in the opposite direction. This occurs in a time interval of (4 TOSC * (Timer2 Prescale Value) before the next PWM period begins. The Timer2 prescaler will be either 1, 4 or 16, depending on the value of the T2CKPS bits (T2CON<1:0>). During the interval from the switch of the unmodulated outputs to the beginning of the next period, the modulated outputs (P1B and P1D) remain inactive. This relationship is shown in Figure 18-8.

Note that in Full-Bridge Output mode, the ECCP1 module does not provide any dead-band delay. In general, since only one output is modulated at all times, dead-band delay is not required. However, there is a situation where a dead-band delay might be required. This situation occurs when both of the following conditions are true:

1. The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
2. The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn-on time.

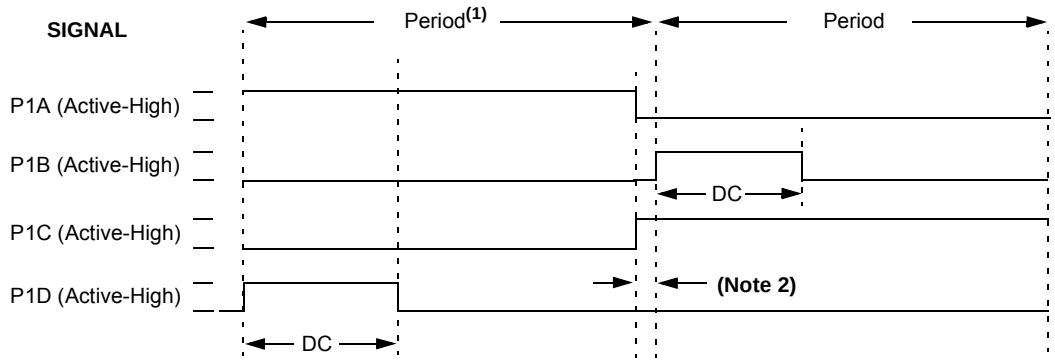
Figure 18-9 shows an example where the PWM direction changes from forward to reverse at a near 100% duty cycle. At time, t1, the outputs, P1A and P1D, become inactive, while output, P1C, becomes active. In this example, since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current may flow through power devices, QC and QD (see Figure 18-7), for the duration of ‘t’. The same phenomenon will occur to power devices, QA and QB, for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, one of the following requirements must be met:

1. Reduce PWM for a PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

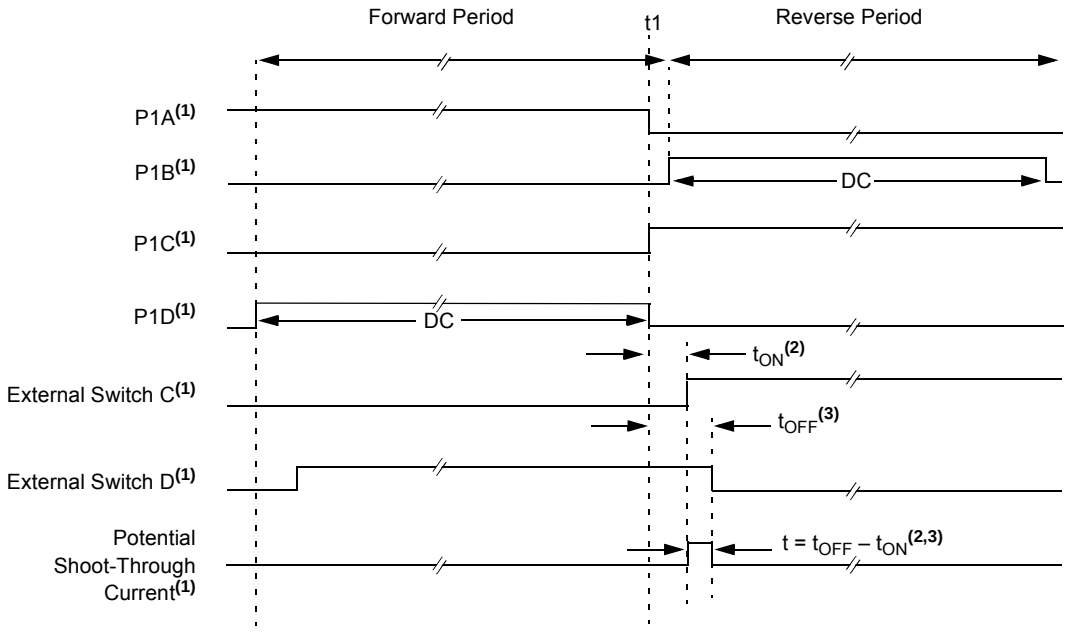
Other options to prevent shoot-through current may exist.

FIGURE 18-8: PWM DIRECTION CHANGE



Note 1: The direction bit in the ECCP1 Control register (CCP1CON<7>) is written at any time during the PWM cycle.
2: When changing directions, the P1A and P1C signals switch before the end of the current PWM cycle at intervals of 4 TOSC, 16 TOSC or 64 TOSC, depending on the Timer2 prescaler value. The modulated P1B and P1D signals are inactive at this time.

FIGURE 18-9: PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



Note 1: All signals are shown as active-high.
2: t_{ON} is the turn-on delay of power switch, QC, and its driver.
3: t_{OFF} is the turn-off delay of power switch, QD, and its driver.

18.4.6 PROGRAMMABLE DEAD-BAND DELAY

In half-bridge applications, where all power switches are modulated at the PWM frequency at all times, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (*shoot-through current*) may flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In the Half-Bridge Output mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 18-4 for the illustration. The lower seven bits of the ECCP1DEL register (Register 18-2) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

18.4.7 ENHANCED PWM AUTO-SHUTDOWN

When the ECCP1 is programmed for any of the Enhanced PWM modes, the active output pins may be configured for auto-shutdown. Auto-shutdown immediately places the Enhanced PWM output pins into a defined shutdown state when a shutdown event occurs.

A shutdown event can be caused by either of the two comparator modules or the FLT0 pin (or any combination of these three sources). The comparators may be used to monitor a voltage input proportional to a current being monitored in the bridge circuit. If the voltage exceeds a threshold, the comparator switches state and triggers a shutdown. Alternatively, a low-level digital signal on the FLT0 pin can also trigger a shutdown. The auto-shutdown feature can be disabled by not selecting any auto-shutdown sources. The auto-shutdown sources to be used are selected using the ECCP1AS<2:0> bits (ECCP1AS<6:4>).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSS1AC<1:0> and PSS1BD<1:0> bits (ECCP1AS<3:0>). Each pin pair (P1A/P1C and P1B/P1D) may be set to drive high, drive low or be tri-stated (not driving). The ECCP1ASE bit (ECCP1AS<7>) is also set to hold the Enhanced PWM outputs in their shutdown states.

The ECCP1ASE bit is set by hardware when a shutdown event occurs. If automatic restarts are not enabled, the ECCP1ASE bit is cleared by firmware when the cause of the shutdown clears. If automatic restarts are enabled, the ECC1PASE bit is automatically cleared when the cause of the auto-shutdown has cleared.

If the ECCP1ASE bit is set when a PWM period begins, the PWM outputs remain in their shutdown state for that entire PWM period. When the ECCP1ASE bit is cleared, the PWM outputs will return to normal operation at the beginning of the next PWM period.

Note:	Writing to the ECCP1ASE bit is disabled while a shutdown condition is active.

REGISTER 18-2: ECCP1DEL: ECCP1 DEAD-BAND DELAY REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0
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 **P1RSEN:** PWM Restart Enable bit

1 = Upon auto-shutdown, the ECCP1ASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically

0 = Upon auto-shutdown, ECCP1ASE must be cleared in software to restart the PWM

bit 6-0 **P1DC<6:0>:** PWM Delay Count bits

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

REGISTER 18-3: ECCP1AS: ECCP1 AUTO-SHUTDOWN CONFIGURATION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0
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 **ECCP1ASE:** ECCP1 Auto-Shutdown Event Status bit
0 = ECCP1 outputs are operating
1 = A shutdown event has occurred; ECCP1 outputs are in shutdown state

bit 6-4 **ECCP1AS<2:0>:** ECCP1 Auto-Shutdown Source Select bits
000 = Auto-shutdown is disabled
001 = Comparator 1 output
010 = Comparator 2 output
011 = Either Comparator 1 or 2
100 = FLT0
101 = FLT0 or Comparator 1
110 = FLT0 or Comparator 2
111 = FLT0 or Comparator 1 or Comparator 2

bit 3-2 **PSS1AC<1:0>:** A and C Pins Shutdown State Control bits
00 = Drive A and C pins to '0'
01 = Drive A and C pins to '1'
1x = A and C pins tri-state

bit 1-0 **PSS1BD<1:0>:** B and D Pins Shutdown State Control bits
00 = Drive B and D pins to '0'
01 = Drive B and D pins to '1'
1x = B and D pins tri-state

18.4.7.1 Auto-Shutdown and Automatic Restart

The auto-shutdown feature can be configured to allow automatic restarts of the module following a shutdown event. This is enabled by setting the P1RSEN bit of the ECCP1DEL register (ECCP1DEL<7>).

In Shutdown mode with P1RSEN = 1 (Figure 18-10), the ECCP1ASE bit will remain set for as long as the cause of the shutdown continues. When the shutdown condition clears, the ECCP1ASE bit is cleared. If P1RSEN = 0 (Figure 18-11), once a shutdown condition occurs, the ECCP1ASE bit will remain set until it is cleared by firmware. Once ECCP1ASE is cleared, the Enhanced PWM will resume at the beginning of the next PWM period.

Note:	Writing to the ECCP1ASE bit is disabled while a shutdown condition is active.

Independent of the P1RSEN bit setting, if the auto-shutdown source is one of the comparators, the shutdown condition is a level. The ECCP1ASE bit cannot be cleared as long as the cause of the shutdown persists.

The Auto-Shutdown mode can be forced by writing a '1' to the ECCP1ASE bit.

18.4.8 START-UP CONSIDERATIONS

When the ECCP1 module is used in the PWM mode, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins. When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the OFF state until the microcontroller drives the I/O pins with the proper signal levels, or activates the PWM output(s).

The CCP1M<1:0> bits (CCP1CON<1:0>) allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (P1A/P1C and P1B/P1D). The PWM output polarities must be selected before the PWM pins are configured as outputs. Changing the polarity configuration while the PWM pins are configured as outputs is not recommended since it may result in damage to the application circuits.

The P1A, P1B, P1C and P1D output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pins for output at the same time as the ECCP1 module may cause damage to the application circuit. The ECCP1 module must be enabled in the proper Output mode and complete a full PWM cycle before configuring the PWM pins as outputs. The completion of a full PWM cycle is indicated by the TMR2IF bit being set as the second PWM period begins.

FIGURE 18-10: PWM AUTO-SHUTDOWN (P1RSEN = 1, AUTO-RESTART ENABLED)

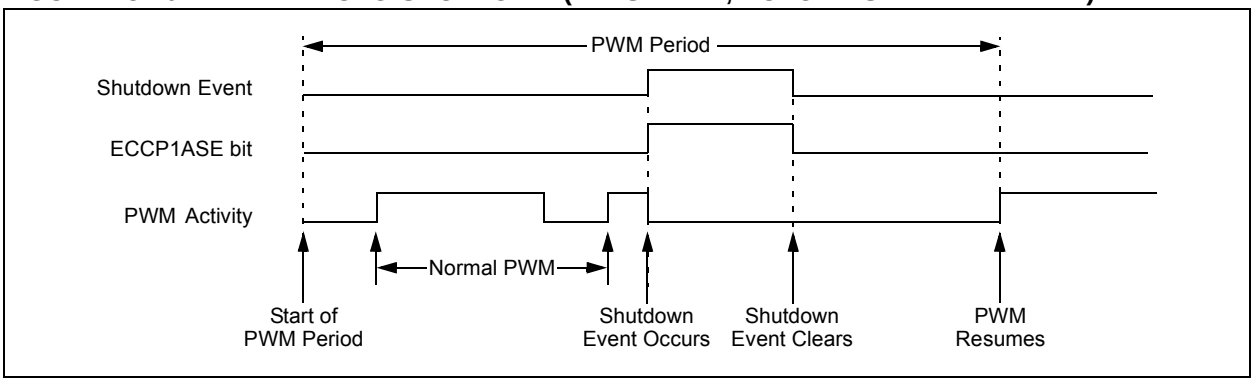
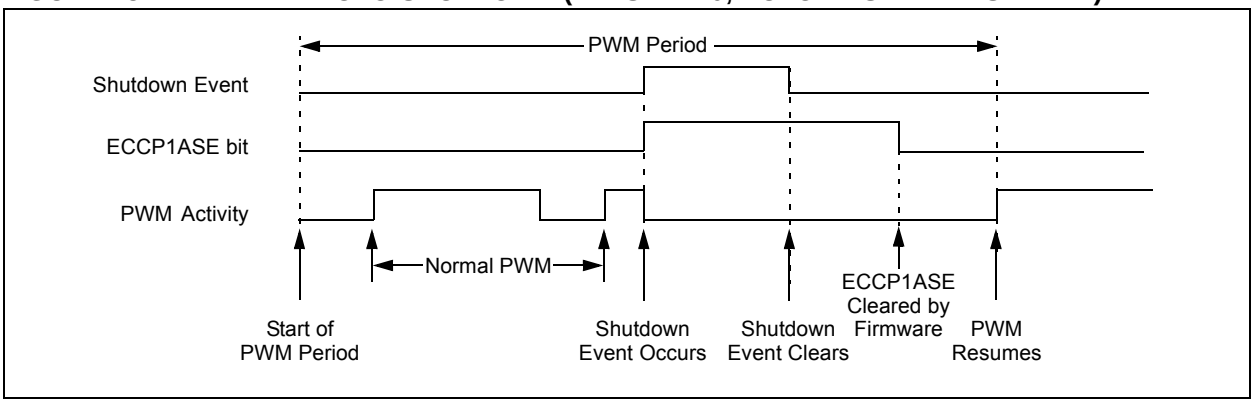


FIGURE 18-11: PWM AUTO-SHUTDOWN (P1RSEN = 0, AUTO-RESTART DISABLED)



18.4.9 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the ECCP1 module for PWM operation:

1. Configure the PWM pins, P1A and P1B (and P1C and P1D, if used), as inputs by setting the corresponding TRIS bits.
2. Set the PWM period by loading the PR2 (PR4) register.
3. Configure the ECCP1 module for the desired PWM mode and configuration by loading the CCP1CON register with the appropriate values:
   - Select one of the available output configurations and direction with the P1M<1:0> bits.
   - Select the polarities of the PWM output signals with the CCP1M<3:0> bits.
4. Set the PWM duty cycle by loading the CCPR1L register and the CCP1CON<5:4> bits.
5. For auto-shutdown:
   - Disable auto-shutdown; ECCP1ASE = 0
   - Configure auto-shutdown source
   - Wait for Run condition
6. For Half-Bridge Output mode, set the dead-band delay by loading ECCP1DEL<6:0> with the appropriate value.
7. If auto-shutdown operation is required, load the ECCP1AS register:
   - Select the auto-shutdown sources using the ECCP1AS<2:0> bits.
   - Select the shutdown states of the PWM output pins using PSS1AC<1:0> and PSS1BD<1:0> bits.
   - Set the ECCP1ASE bit (ECCP1AS<7>).
8. If auto-restart operation is required, set the P1RSEN bit (ECCP1DEL<7>).
9. Configure and start TMR2 (TMR4):
   - Clear the TMRx interrupt flag bit by clearing the TMRxIF bit (PIR1<1> for Timer2 or PIR3<3> for Timer4).
   - Set the TMRx prescale value by loading the TxCKPS bits (T2CON<1:0> for Timer2 or T4CON<1:0> for Timer4).
   - Enable Timer2 (or Timer4) by setting the TMRxON bit (T2CON<2> for Timer2 or T4CON<2> for Timer4).
10. Enable PWM outputs after a new PWM cycle has started:
    - Wait until TMR2 (TMR4) overflows (TMRxIF bit is set).
    - Enable the ECCP1/P1A, P1B, P1C and/or P1D pin outputs by clearing the respective TRIS bits.
    - Clear the ECCP1ASE bit (ECCP1AS<7>).

18.4.10 EFFECTS OF A RESET

Both Power-on Reset and subsequent Resets will force all ports to Input mode and the CCPx/ECCPx registers to their Reset states.

This forces the Enhanced CCPx modules to reset to a state compatible with the standard CCPx modules.

TABLE 18-5: REGISTERS ASSOCIATED WITH ECCPx MODULES AND TIMER1 TO TIMER4

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	71
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	71
TRISD(1)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	71
TRISE	TRISE7(2)	TRISE6(2)	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	71
TRISG	TRISG7	TRISG6	TRISG5	TRISG4	TRISG3(2)	TRISG2	TRISG1	TRISG0(2)	71
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	71
TMR1L	Timer1 Register Low Byte								70
TMR1H	Timer1 Register High Byte								70
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	70
TMR2	Timer2 Register								70
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	70
PR2	Timer2 Period Register								70
TMR3L	Timer3 Register Low Byte								70
TMR3H	Timer3 Register High Byte								70
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	71
TMR4	Timer4 Register								72
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	72
PR4	Timer4 Period Register								72
CCPRxL(3)	Capture/Compare/PWM Register x Low Byte								70
CCPRxH(3)	Capture/Compare/PWM Register x High Byte								70
CCPxCON(3)	PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	70
ECCPxAS(3)	ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0	70, 73
ECCPxDEL(3)	PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0	73

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

Note 1: Applicable in 64-pin devices only.

2: Registers and/or specific bits are unimplemented in 64-pin devices.

3: Generic term for all of the identical registers of this name for all Enhanced CCPx modules, where 'x' identifies the individual module (ECCP1, ECCP2 or ECCP3). Bit assignments and Reset values for all registers of the same generic name are identical.

NOTES:

19.0 ETHERNET MODULE

All members of the PIC18F97J60 family of devices feature an embedded Ethernet controller module. This is a complete connectivity solution, including full implementations of both Media Access Control (MAC) and Physical Layer (PHY) transceiver modules. Two pulse transformers and a few passive components are all that are required to connect the microcontroller directly to an Ethernet network.

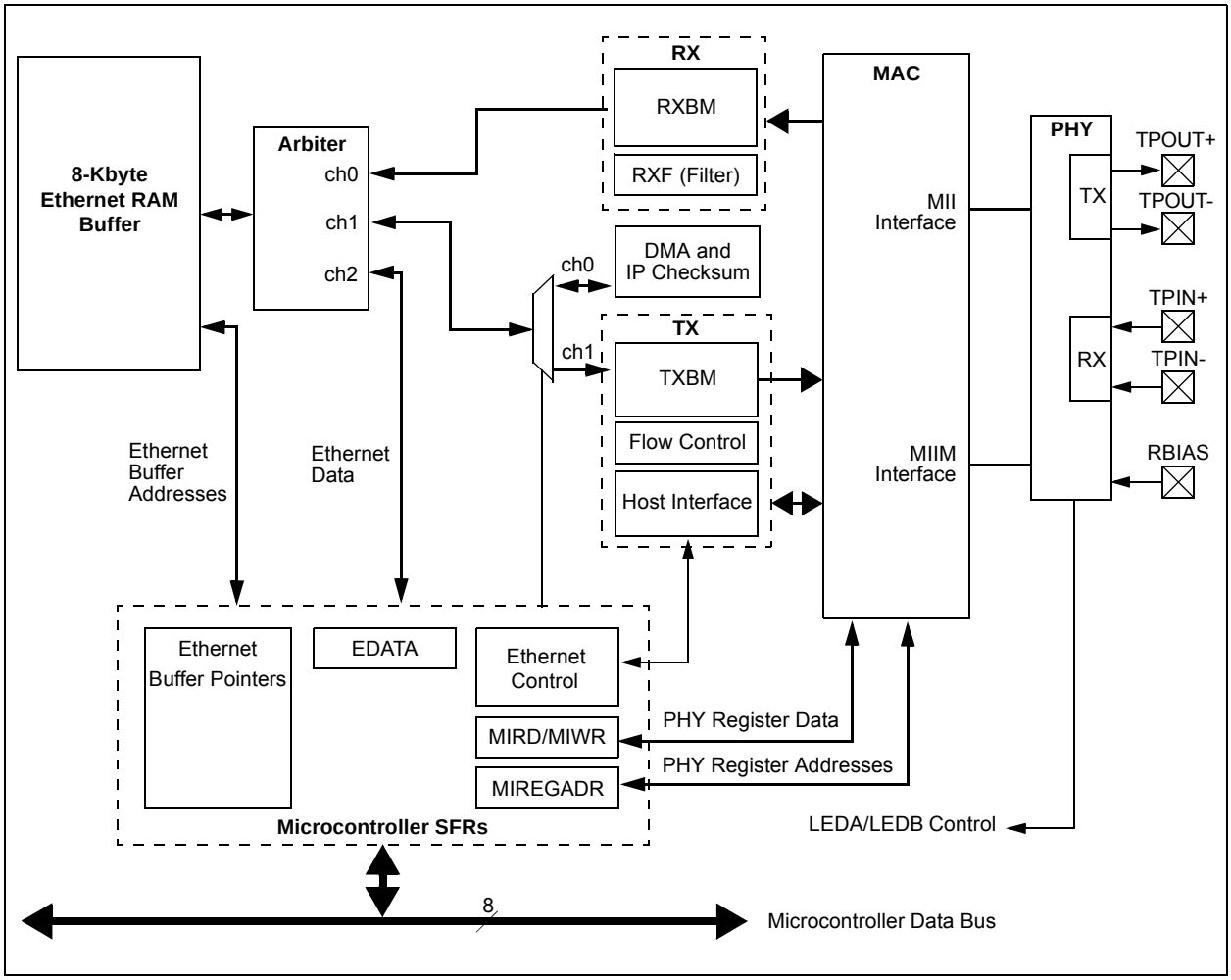
The Ethernet module meets all of the IEEE 802.3™ specifications for 10-BaseT connectivity to a twisted-pair network. It incorporates a number of packet filtering schemes to limit incoming packets. It also provides an internal DMA module for fast data throughput and hardware assisted IP checksum calculations. Provisions are also made for two LED outputs to indicate link and network activity.

A simple block diagram of the module is shown in Figure 19-1.

The Ethernet module consists of five major functional blocks:

1. The PHY transceiver module that encodes and decodes the analog data that is present on the twisted-pair interface and sends or receives it over the network.
2. The MAC module that implements IEEE 802.3 compliant MAC logic and provides Media Independent Interface Management (MIIM) to control the PHY.
3. An independent, 8-Kbyte RAM buffer for storing packets that have been received and packets that are to be transmitted.
4. An arbiter to control access to the RAM buffer when requests are made from the microcontroller core, DMA, transmit and receive blocks.
5. The register interface that functions as an interpreter of commands and internal status signals between the module and the microcontroller's SFRs.

FIGURE 19-1: ETHERNET MODULE BLOCK DIAGRAM



19.1 Physical Interfaces and External Connections

19.1.1 SIGNAL AND POWER INTERFACES

PIC18F97J60 family devices all provide a dedicated 4-pin signal interface for the Ethernet module. No other microcontroller or peripheral functions are multiplexed with these pins, so potential device configuration conflicts do not need to be considered. The pins are:

- TPIN+: Differential plus twisted-pair input
- TPIN-: Differential minus twisted-pair input
- TPOUT+: Differential plus twisted-pair output
- TPOUT-: Differential minus twisted-pair output

No provisions are made for providing or receiving digital Ethernet data from an external Ethernet PHY.

In addition to the signal connections, the Ethernet module has its own independent voltage source and ground connections for the PHY module. Separate connections are provided for the receiver (V_{DDRX} and V_{SSRX}), the transmitter (V_{DDTX} and V_{SSTX}) and the transmitter's internal PLL (V_{DDPLL} and V_{SSPLL}). Although the voltage requirements are the same as V_{DD} and V_{SS} for the microcontroller, the pins are not internally connected. For the Ethernet module to operate properly, supply voltage and ground must be connected to these pins. All of the microcontroller's power and ground supply pins should be externally connected to the same power source or ground node, with no inductors or other filter components between the microcontroller and Ethernet module's V_{DD} pins.

Besides the independent voltage connections, the PHY module has a separate bias current input pin, RBIAS. A bias current, derived from an external resistor, must be applied to RBIAS for proper transceiver operation.

19.1.2 LED CONFIGURATION

The PHY module provides separate outputs to drive the standard Ethernet indicators, LEDA and LEDB. The LED outputs are multiplexed with PORTA pins, RA0 and RA1. Their use as LED outputs is enabled by setting the Configuration bit, ETHLED (Register 25-6, CONFIG3H<2>). When configured as LED outputs, RA0/LEDA and RA1/LEDB have sufficient drive capacity (up to 25 mA) to directly power the LEDs. The pins must always be configured to supply (source) current to the LEDs. Users must also configure the pins as outputs by clearing TRISA<1:0>.

The LEDs can be individually configured to automatically display link status, RX/TX activity, etc. A configurable stretch capability prolongs the LED blink duration for short events, such as a single packet transmit, allowing human perception. The options are controlled by the PHLCON register (Register 19-13). Typical values for blink stretch are listed in Table 19-1.

TABLE 19-1: LED BLINK STRETCH LENGTH

Stretch Length	Typical Stretch (ms)
T_{NSTRCH} (normal)	40
T_{MSTRCH} (medium)	70
T_{LSTRCH} (long)	140

19.1.3 OSCILLATOR REQUIREMENTS

The Ethernet module is designed to operate at 25 MHz. This is provided by the primary microcontroller clock, either with a 25 MHz crystal connected to the OSC1 and OSC2 pins or an external clock source connected to the OSC1 pin. No provision is made to clock the module from a different source.

To maintain the required clock frequency, the microcontroller can operate only from the primary oscillator source (PRI_RUN or PRI_IDLE modes) while the Ethernet module is enabled. Using any other power-managed mode will require that the Ethernet module be disabled.

19.1.3.1 Start-up Timer

The Ethernet module contains a start-up timer, independent of the microcontroller's OST, to ensure that the PHY module's PLL has stabilized before operation. Clearing the module enable bit, ETHEN (ECON2<5>), clears the PHYRDY status bit (ESTAT<0>). Setting the ETHEN bit causes this start-up timer to start counting. When the timer expires, after 1 ms, the PHYRDY bit will be automatically set.

After enabling the module by setting the ETHEN bit, the application software should always poll PHYRDY to determine when normal Ethernet operation can begin.

19.1.4 MAGNETICS, TERMINATION AND OTHER EXTERNAL COMPONENTS

To complete the Ethernet interface, the Ethernet module requires several standard components to be installed externally. These components should be connected, as shown in Figure 19-2.

The internal analog circuitry in the PHY module requires that an external resistor (2.26 kΩ) be attached from RBIAS to ground. The resistor influences the TPOUT+/- signal amplitude. It should be placed as close as possible to the chip with no immediately adjacent signal traces to prevent noise capacitively coupling into the pin and affecting the transmit behavior. It is recommended that the resistor be a surface mount type.

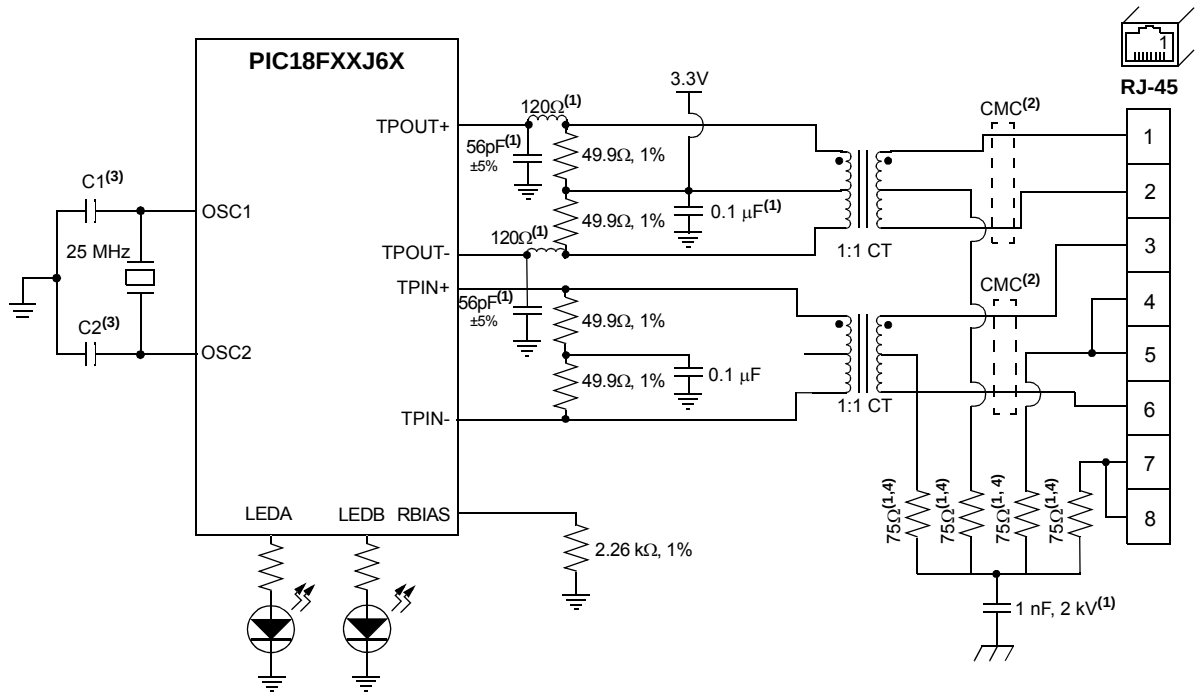
On the TPIN+/TPIN- and TPOUT+/TPOUT- pins, 1:1 center-tapped pulse transformers, rated for Ethernet operations (10/100 or 10/100/1000), are required. When the Ethernet module is enabled, current is continually sunk through both TPOUT pins. When the PHY is actively transmitting, a differential voltage is created on the Ethernet cable by varying the relative current sunk by TPOUT+ compared to TPOUT-.

A common-mode choke on the PHY side of the interface (i.e., between the microcontrollers's TPOUT pins and the Ethernet transformer) is not recommend. If a common-mode choke is used to reduce EMI emissions, it should be placed between the Ethernet transformer and Pins 1 and 2, of the RJ-45 connector. Many Ethernet transformer modules include common-mode chokes inside the same device package. The transformers should have at least the isolation rating specified in Table 28-28 to protect against static voltages and meet IEEE 802.3 isolation requirements (see **Section 28.5 "Ethernet Specifications and Requirements"** for specific transformer requirements). Both transmit and receive interfaces additionally require two resistors and a capacitor to properly terminate the transmission line, minimizing signal reflections.

All power supply pins must be externally connected to the same power source. Similarly, all ground references must be externally connected to the same ground node. Each VDD and VSS pin pair should have a 0.1 μF ceramic bypass capacitor placed as close to the pins as possible.

Since relatively high currents are necessary to operate the twisted-pair interface, all wires should be kept as short as possible. Reasonable wire widths should be used on power wires to reduce resistive loss. If the differential data lines cannot be kept short, they should be routed in such a way as to have a 100Ω characteristic impedance.

FIGURE 19-2: EXTERNAL COMPONENTS REQUIRED FOR ETHERNET OPERATION



Note 1: These components are installed for EMI reduction purposes. See **Section 19.1.5** for more information.

2: Recommended insertion point for Common-Mode Chokes (CMCs) if required for EMI reduction.

3: See **Section 3.3 "Crystal Oscillator/Ceramic Resonators (HS Modes)"** for recommended values.

4: Power over Ethernet applications require capacitors in series with these resistors.

19.1.5 EMI EMISSIONS CONSIDERATIONS

Most locales have limits on unintentional EMI or EMC emissions that govern the amount of electromagnetic energy that may be radiated into the environment across a range of test frequencies. Ethernet applications normally do not include intentional radio frequency emissions sources. They may experience occasional regulatory failures though, due to the relative ease at which high-frequency noise may radiate out of a long attached Ethernet cable. Long cables can act as unintentional antennas.

The PIC18F97J60 family Ethernet module transmit engine internally operates by stepping the 25 MHz base Ethernet clock up to a high frequency via a PLL embedded in the PHY module. Then, the high frequency is used to turn on/turn off small current sinks on the TPOUT+ and TPOUT- pins. This current-mode drive technique allows the PHY to generate an Ethernet TX waveform that resembles an analog signal, with most spectral energy at or below 20 MHz.

However, while low in amplitude, the high frequency used to synthesize the waveform can, in some application circuits, radiate out of the circuit and result in regulatory emissions compliance failures. Such failures caused by the Ethernet module will normally be exhibited as excess emissions at 200 MHz and occasionally 400 MHz.

To minimize the chance of failure, the use of the LC low-pass filter is recommended on the TPOUT+ and TPOUT- pins, as shown Figure 19-2.

In this circuit, 120 ohm ferrite beads are used along with 56 pF±5% capacitors to form a low-pass filter with a -3dB breakpoint that is above 20 MHz, but below 200 MHz. 10Base-T Ethernet signaling requires only about 20 MHz of spectral bandwidth, so minimal distortion is done to the Ethernet signal by using these filters. However, noise at 200 MHz or 400 MHz, generated by the PHY, is reduced by several decibels before having a chance to radiate out of the application and cause a regulatory failure. In this circuit, the ferrite beads must have a saturation current rating of at least 100 mA.

If EMI emissions regulations are stringent in your locale, additional care should be taken when selecting the Ethernet magnetics to further minimize unintentionally radiating common-mode signals out of the Ethernet cable. Ethernet magnetics with a high differential to common-mode rejection ratio should be used.

The differential to common-mode rejection parameter is normally expressed in magnetics manufacturers' data sheets, in units of negative decibels across a test frequency range. In the absence of test data indicating otherwise, a more negative specification at higher frequencies is recommended for the PIC18F97J60 family Ethernet module. For example, a device rated for -40 dB @ 100 MHz is likely preferable to -33 dB @ 100 MHz, even if the performance at 30 MHz is similar or better on the -33 dB @ 100MHz magnetics.

Often, the use of "5-core" magnetics, or magnetics involving a center tapped inductor or auto-transformer on the TX path, is also desirable for EMI emissions reasons.

19.1.6 AUTOMATIC RX POLARITY DETECTION AND CORRECTION

10Base-T Ethernet signaling is performed on the Ethernet cable as a differentially encoded Manchester data stream. This signaling is polarized; therefore, it is required that the RX+ Ethernet signal on the Ethernet cable reach the TPIN+ pin, and the RX- Ethernet signal reach the TPIN- pin. Connecting RX+ to TPIN- and RX- to TPIN+ (by way of Ethernet isolation transformers) will cause the PIC18F97J60 family Ethernet module to successfully link with the remote partner. However, all receive data will be corrupted by the polarity mismatch and will be internally discarded by the PHY as if it were noise on the wire.

Higher speed 100Base-TX and 1000Base-T Ethernet technologies uses different signaling schemes. They use Multi-Level Transition 3 (MLT3) and Five-Level Pulse Amplitude Modulation (PAM5) encoding on the wire, respectively. These encodings are non-polarized. Therefore, swapping the differential wires will have no impact on the Ethernet controller's ability to communicate with the remote node.

A limited number of modern 3rd party 10/100 and 10/100/1000 rated Ethernet devices (switches, routers and end devices) connect their TX+ and TX- signals to the incorrect pins on their RJ-45 Ethernet jack. These devices are not IEEE Standard 802.3 compliant. However, because 100Base-TX and 1000Base-T communications continue to work without correct polarization, some 3rd party vendors mistakenly release their products to production without catching these polarization errors.

Due to these circumstances, current revisions of the Ethernet controller in the PIC18F97J60 family of devices are not compatible with a limited number of 3rd party Ethernet devices. The PIC18FXXJXX devices will link up with the partner and the PHY RX activity LED (if enabled) will blink whenever a packet is transmitted to the PIC18FXXJXX device. However, no packets will be successfully received and written in the Ethernet SRAM buffer when the polarity is incorrect. To eliminate this problem, and obtain maximum interoperability with 3rd party devices, it is possible to externally add an RX polarity swapping circuit to PIC18F97J60 family applications. Figure 19-3 demonstrates the use of bus switches to facilitate the swapping of the RX signals.

In Figure 19-3, a general purpose output pin is used to select the polarity of the RX signals. When the select line is held low, the A ports of the switches will connect with the B0 ports, leaving the B1 ports disconnected. This will allow the TPIN+ pin to be connected to Pin 3 of the RJ-45 jack while TPIN- is connected to Pin 6. These connections accommodate the IEEE Standard 802.3 specified polarity.

When the select line is raised high, the A ports of the switches will connect with the B1 ports, leaving the B0 ports disconnected. This will swap the RX polarity and route the TPIN+ pin to the signal on RJ-45 Jack Pin 6. TPIN- will connect to RJ-45 Pin 3. This swapped polarity can correct an incorrectly wired signal generated at the remote link partner or in the intermediate cabling.

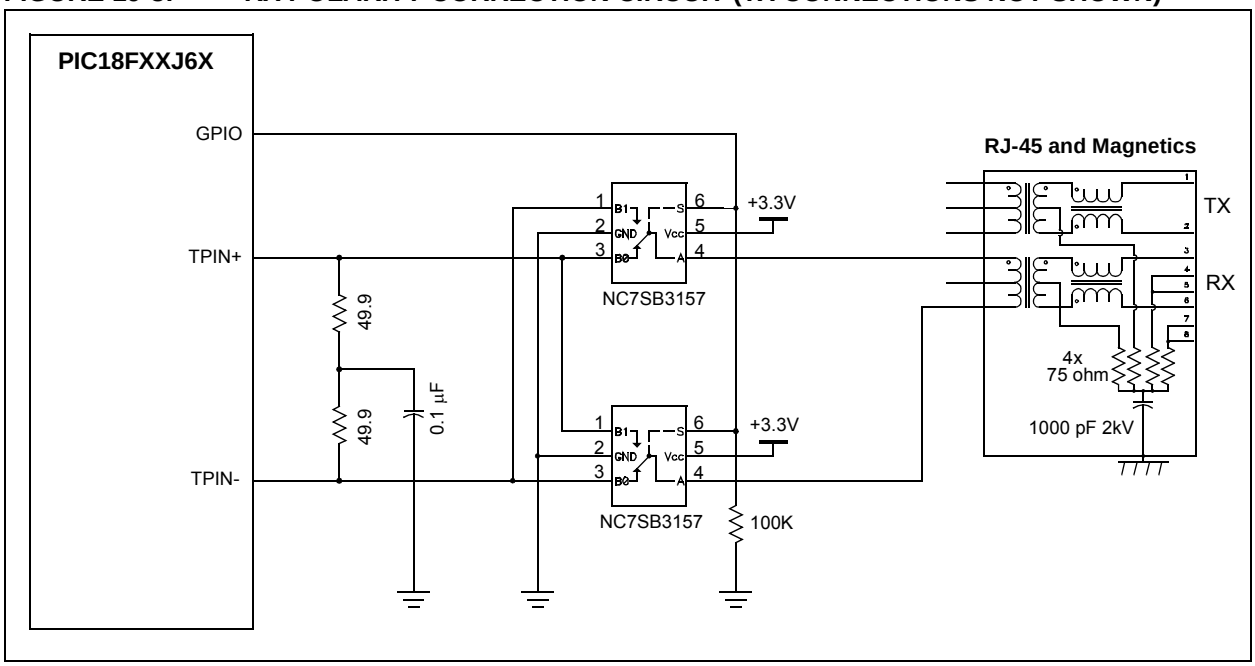
In the MCU, software must be written to toggle the select line state to the correct level based on the connected link partner. This is best achieved by periodically toggling the select line at low frequency (<5 Hz), while watching for a successful packet reception event. When the correct polarity is found, the select line should stop toggling and remain static until the Ethernet link is removed.

In some cases, rather than periodically toggling the polarity select at a low frequency to receive a packet, faster determination of the correct polarity may be determined by transmitting a packet and looking for an immediate response. This method requires the use of a protocol that results in a response packet from the network. In many networks, Dynamic Host Configuration Protocol (DHCP) discovery packets may be used to resolve the correct TPIN polarity quickly.

Care should be taken when selecting the bus switches to ensure that they are capable of passing the Ethernet signals without distortion. The TPIN± pins will weakly bias the RX common-mode voltage to approximately $V_{DD}/2$, and the Ethernet RX waveform will add up to ±1.4V onto this common-mode voltage. Therefore, the switches must be capable of passing signals of at least 3.05V.

Additionally, the bus switches should have low capacitance to minimize the signal loss and impedance discontinuity that may affect the RX signal. The switches, rated -3dB bandwidth, must be well above 20 MHz.

FIGURE 19-3: RX POLARITY CORRECTION CIRCUIT (TX CONNECTIONS NOT SHOWN)



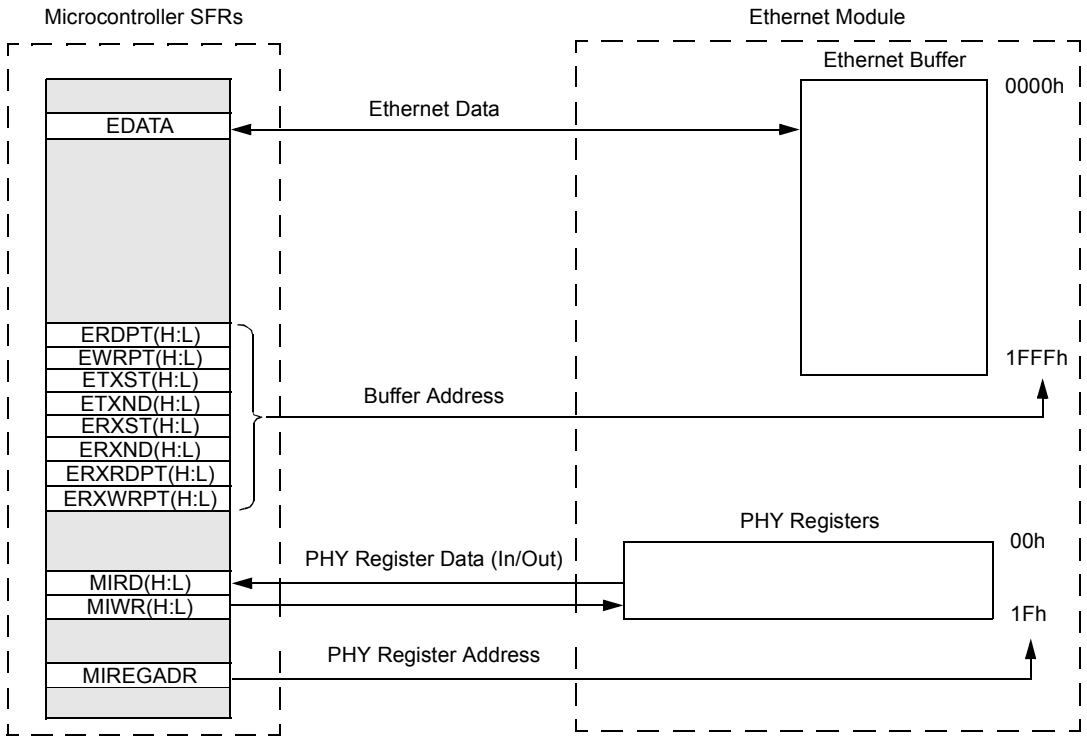
19.2 Ethernet Buffer and Register Spaces

The Ethernet module uses three independent memory spaces for its operations:

- An Ethernet RAM buffer which stores packet data as it is received and being prepared for transmission.
- A set of 8-bit Special Function Registers (SFRs), used to control the module, and pass data back and forth between the module and microcontroller core.
- A separate set of 16-bit PHY registers used specifically for PHY control and status reporting.

The Ethernet buffer and PHY Control registers are contained entirely within the Ethernet module and cannot be accessed directly by the microcontroller. Data is transferred between the Ethernet and microcontroller by using buffer and pointer registers mapped in the microcontroller's SFR space. The relationships between the SFRs and the Ethernet module's memory spaces are shown in Figure 19-4.

FIGURE 19-4: RELATIONSHIP BETWEEN MICROCONTROLLER AND ETHERNET MEMORY SPACES



Note: Microcontroller SFRs are not shown in the order of their placement in the data memory space. Memory areas are not shown to scale.

19.2.1 ETHERNET BUFFER AND BUFFER POINTER REGISTERS

The Ethernet buffer contains the transmit and receive memory used by the Ethernet controller. The entire buffer is 8 Kbytes, divided into separate receive and transmit buffer spaces. The sizes and locations of transmit and receive memory are fully definable using the pointers in the Ethernet SFR space. The organization of the memory space and the relationships of the pointers are shown in Figure 19-5.

The buffer is always accessible through the EDATA and Ethernet Pointer SFRs, regardless of whether or not the Ethernet module is enabled. This makes the buffer potentially useful for applications requiring large amounts of RAM and that do not require Ethernet communication. In these instances, disabling the Ethernet module reduces overall power usage but does not prevent buffer access.

19.2.1.1 Reading and Writing to the Buffer

The Ethernet buffer contents are accessed through the EDATA register, which acts as a window from the microcontroller data bus into the buffer. The location of that window is determined by either the ERDPT or EWRPT Pointers, depending on the operation being performed. For example, writing to EDATA causes a write to the Ethernet buffer at the address currently indicated by the EWRPT register pair. Similarly, moving the contents of EDATA to another register actually moves the buffer contents at the address indicated by the ERDPT Pointer.

When the AUTOINC bit (ECON2<7>) is set, the associated Read or Write Pointer increments by one address following each read or write operation. This eliminates the need to constantly update a pointer after each read or write, simplifying multiple sequential operations. By default, the AUTOINC bit is set.

While sequentially reading from the receive buffer, a wrapping condition will occur at the end of the receive buffer. A read of EDATA, from the address programmed into the ERXND Pointers, will cause the ERDPT registers to be incremented to the value contained in the ERXST Pointers. Writing to the buffer, on the other hand, does not result in automatic wrapping.

By design, the Ethernet memory buffer is unable to support a set of operations where EDATA is used as both an operand and a data destination. Failure to observe these restrictions will result in a corrupted read or write. Also, due to the read-modify-write architecture of the processor core, single-cycle instructions, which write to the EDATA register, will have a side effect of automatically incrementing the ERDPT registers when AUTOINC is set. Using double-cycle `MOVFF`, `MOVSF` and `MOVSS` instructions to write to EDATA will not affect the Read Pointer. See the following note for examples.

Note: Any single instruction that performs both a read and write to the EDATA SFR register will result in a corrupted operation.
Unsupported examples:

```
INCF   EDATA, F
XORWF  EDATA, F
MOVFF  EDATA, EDATA
MOVFF  INDF0, EDATA; (FSR0 = F61h)
```

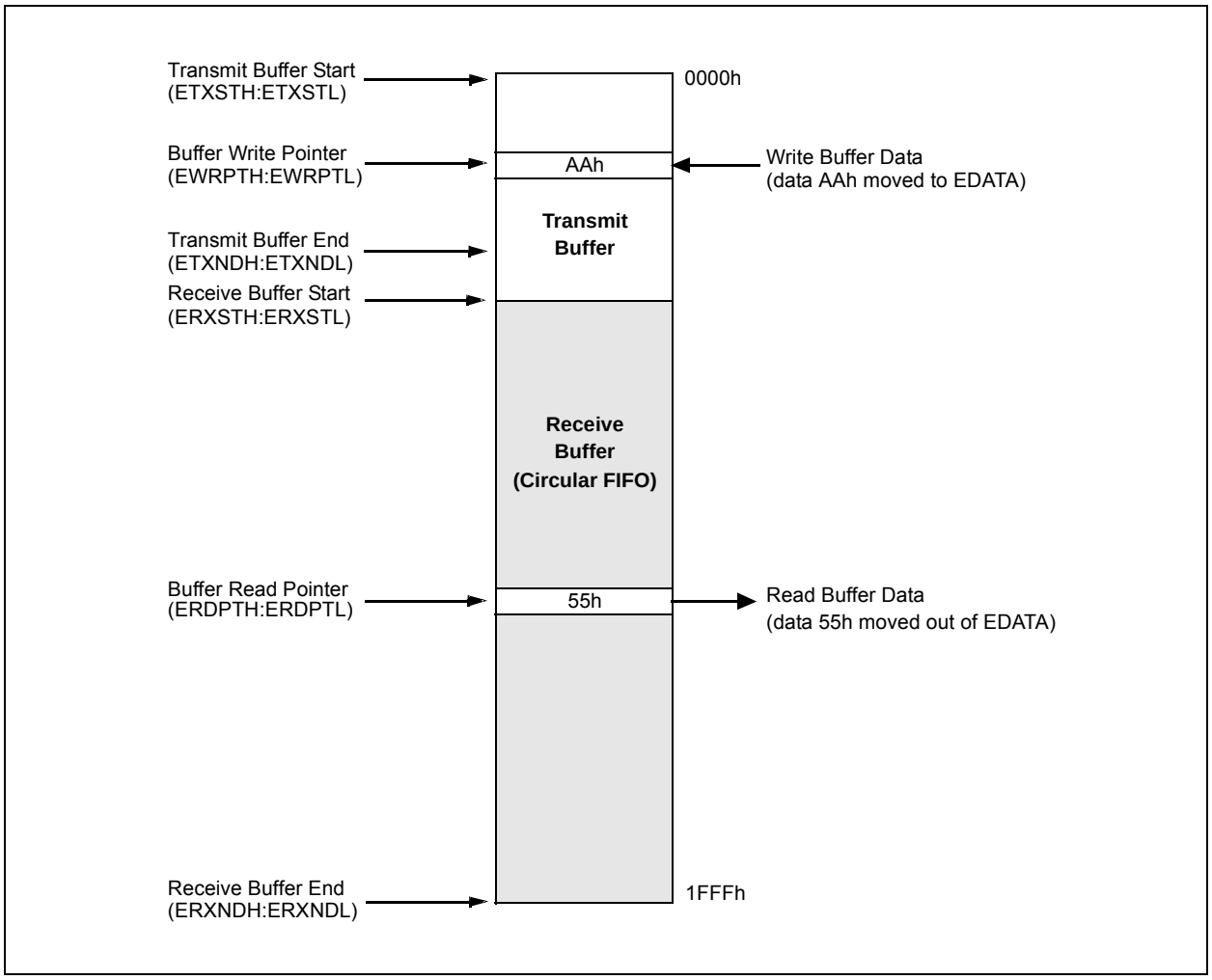
Instructions that only perform one read or one write are permitted.
Supported examples:

```
INCF   EDATA, W
MOVF   EDATA, W
MOVFF  INDF0, EDATA; (FSR0 != F61h)
```

Single-cycle, write-only instructions, while valid, will have a side effect of also incrementing the ERDPT registers when AUTOINC is enabled.
Examples incrementing both ERDPT and EWRPT:

```
CLRF   EDATA
SETF   EDATA
MOVWF  EDATA
```

FIGURE 19-5: ETHERNET BUFFER ORGANIZATION



19.2.1.2 Receive Buffer

The receive buffer constitutes a circular FIFO buffer managed by hardware. The register pairs, ERXSTH:ERXSTL and ERXNDH:ERXNDL, serve as pointers to define the buffer's size and location within the memory. The byte pointed to by the ERXST pair and the byte pointed to by the ERXND pair are both included in the FIFO buffer.

As bytes of data are received from the Ethernet interface, they are written into the receive buffer sequentially. However, after the memory pointed to by the ERXND Pointers is written to, the hardware will automatically write the next byte of received data to the memory pointed to by the ERXST pair. As a result, the receive hardware will never write outside the boundaries of the FIFO.

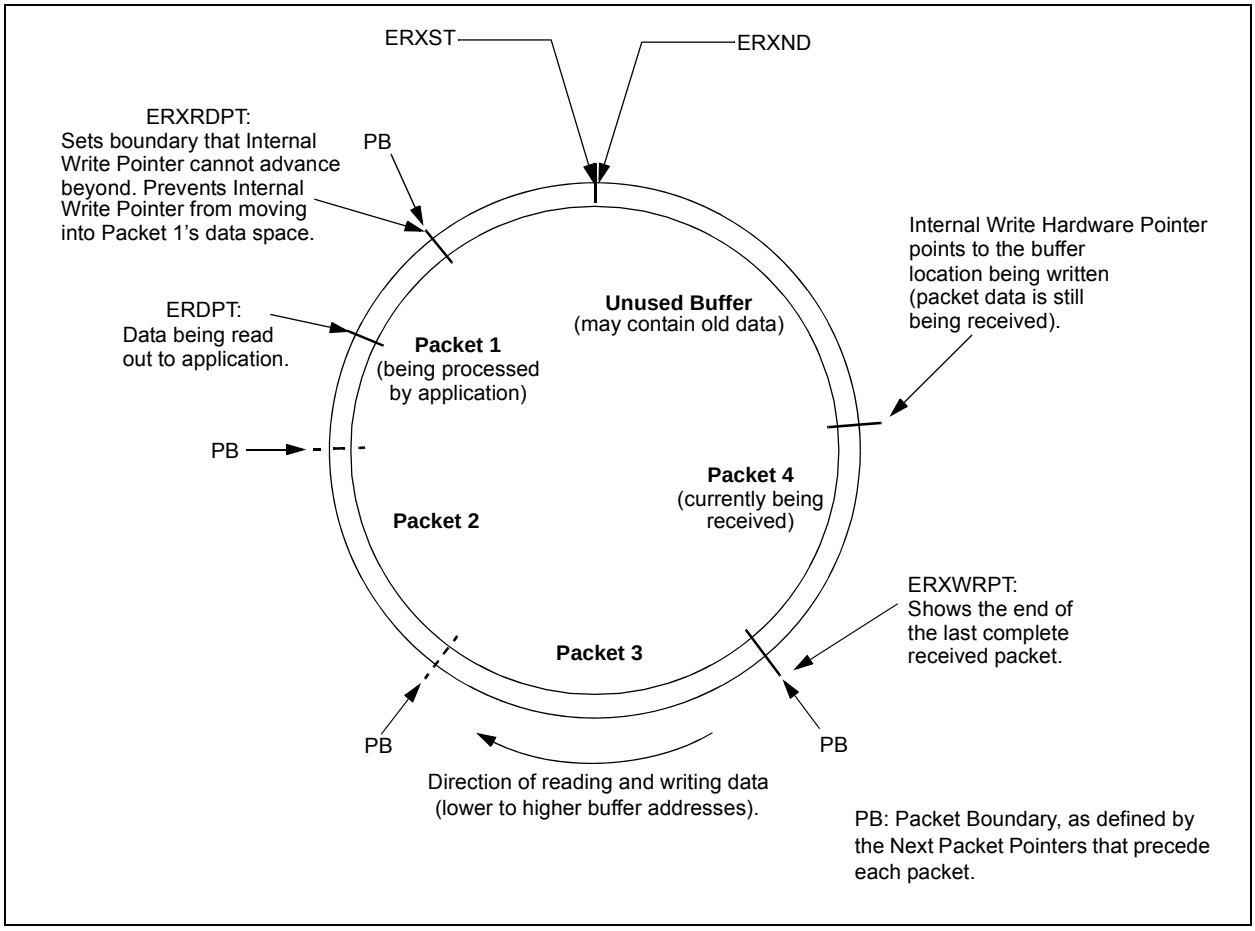
The user may program the ERXST and ERXND Pointers while the receive logic is disabled. The pointers must not be modified while the receive logic is enabled (ERXEN (ECON1<2>) is set).

The buffer hardware uses an Internal Pointer (not mapped to any user-accessible registers) to determine where unvalidated incoming data is to be written. When a packet has been completely received and validated, the read-only ERXWRPTH:ERXWRPTL registers are updated with the Internal Pointer's value. Thus, the ERXWRPT registers define the general area in the receive buffer where data is currently being written. This makes it useful for determining how much free space is available within the FIFO.

The ERXRDPT registers define a location within the FIFO where the receive hardware is forbidden to write to. In normal operation, the receive hardware will write data up to, but not including, the memory pointed to by the ERXRDPT registers. If the FIFO fills up with data and new data continues to arrive, the hardware will not overwrite the previously received data. Instead, the incoming data will be thrown away and the old data will be preserved. In order to continuously receive new data, the application must periodically advance this pointer whenever it finishes processing some, or all, of the old received data.

An example of how the Receive Buffer Pointers and packet data are related in the circular buffer scheme is shown in Figure 19-6. Note that while four packets are shown in this example, the actual number of packets may be greater or lesser.

FIGURE 19-6: CIRCULAR FIFO BUFFER AND THE RELATIONSHIPS OF THE POINTERS



19.2.1.3 Transmit Buffer

Any space within the 8-Kbyte memory which is not programmed as part of the receive FIFO buffer is considered to be the transmit buffer. The responsibility of managing where packets are located in the transmit buffer belongs to the application. Whenever the application decides to transmit a packet, the ETXST and ETXND Pointers are programmed with addresses specifying where, within the transmit buffer, the particular packet to transmit is located. The hardware does not check that the start and end addresses do not overlap with the receive buffer. To prevent buffer corruption, the firmware must not transmit a packet while the ETXST and ETXND Pointers are overlapping the receive buffer, or while the ETXND Pointers are too close to the receive buffer. See **Section 19.5.2 "Transmitting Packets"** for more information.

19.2.1.4 Buffer Arbiter and Access Arbitration

The Ethernet buffer is clocked at one-half of the microcontroller clock rate. Varying amounts of memory access bandwidth are available depending on the clock speed. The total bandwidth available, in bytes per second, is equal to twice the instruction rate (2 * FCY or FOSC/2). For example, at a system clock speed of 41.667 MHz, the total available memory bandwidth that is available is 20.834 Mbyte/s. At an Ethernet signaling rate of 10 Mbit/s, the Ethernet RX engine requires 1.25 Mbyte/s of buffer memory bandwidth to operate without causing an overrun. If Full-Duplex mode is used, an additional 1.25 Mbyte/s is required to allow for simultaneous RX and TX activity.

Because of the finite available memory bandwidth, a three-channel arbiter is used to allocate bandwidth between the RX engine, the TX and DMA engines, and the microcontroller's CPU (i.e., the application accessing EDATA). The arbiter gives the EDATA register accesses first priority, while all remaining bandwidth is shared between the RX and TX/DMA blocks.

With arbitration, bandwidth limitations require that some care be taken in balancing the needs of the module's hardware with that of the application. Accessing the EDATA register too often may result in the RX or TX blocks causing a buffer overrun or underrun, respectively. If such a memory access failure occurs, the BUFER bit (ESTAT<6>), and either the TXERIF or RXERIF interrupt flag, becomes set, and a TX or RX interrupt occurs (if enabled). In either case, the current packet will be lost or aborted.

To eliminate the risk of lost packets, run the microcontroller core at higher speeds. Following the arbitration restrictions, shown in Table 19-2, will prevent memory access failures from occurring. Also, avoid using segments of application code which perform back-to-back accesses of the EDATA register. Instead, insert one or more instructions (including NOP instructions) between each read or write to EDATA.

19.2.1.5 DMA Access to the Buffer

The integrated DMA controller must read from the buffer when calculating a checksum, and it must read and write to the buffer when copying memory. The DMA follows the same wrapping rules as previously described for the receive buffer. While it sequentially reads, it will be subject to a wrapping condition at the end of the receive buffer. All writes it does will not be subject to any wrapping conditions. See **Section 19.9 "Direct Memory Access Controller"** for more information.

TABLE 19-2: BUFFER ARBITRATION RESTRICTIONS VS. CLOCK SPEED

FOSC (MHz)	FCY (MHz)	Available Bandwidth (Mbyte/s)			Application Restrictions to Prevent Underrun/Overrun
		Total	After RX	After TX	
41.667	10.42	20.83	19.58	18.33	Access EDATA no more than once every 2 TCY
31.250	7.81	15.63	14.38	13.13	Access EDATA no more than once every 2 TCY
25.000	6.25	12.50	11.25	10.00	Access EDATA no more than once every 2 TCY
20.833	5.21	10.42	9.17	7.92	Access EDATA no more than once every 2 TCY
13.889	3.47	6.94	5.69	4.44	Access EDATA no more than once every 2 TCY
12.500	3.13	6.25	5.00	3.75	Access EDATA no more than once every 2 TCY
8.333	2.08	4.17	2.92	1.67	Access EDATA no more than once every 3 TCY
6.250	1.56	3.13	1.88	0.63	Access EDATA no more than once every 5 TCY
4.167	1.04	2.08	0.83	< 0	Do not use DMA, do not use full duplex, access EDATA no more than once every 3 TCY
2.778	0.69	1.39	0.14	< 0	Do not use DMA, do not use full duplex, access EDATA no more than once every 10 TCY

19.2.2 SFRs AND THE ETHERNET MODULE

Like other peripherals, direct control of the Ethernet module is accomplished through a set of SFRs. Because of their large number, the majority of these registers is located in the bottom half of Bank 14 of the microcontroller's data memory space.

Five key SFRs for the Ethernet module are located in the microcontroller's regular SFR area in Bank 15, where fast access is possible. They are:

- ECON1
- EDATA
- EIR
- The Ethernet Buffer Read Pointer Pair (ERDPTH and ERDPTL)

ECON1 is described along with other Ethernet control registers in the following section. EDATA and ERDPTH:ERDPTL are the Ethernet Data Buffer registers and its pointers during read operations (see **Section 19.2.1 "Ethernet Buffer and Buffer Pointer Registers"**). EIR is part of the Ethernet interrupt structure and is described in **Section 19.3 "Ethernet Interrupts"**.

Many of the Ethernet SFRs in Bank 14 serve as pointer registers to indicate addresses within the dedicated Ethernet buffer for storage and retrieval of packet data. Others store information for packet pattern masks or checksum operations. Several are used for controlling overall module operations, as well as specific MAC and PHY functions.

19.2.3 ETHERNET CONTROL REGISTERS

The ECON1 register (Register 19-1) is used to control the main functions of the module. Receive enable, transmit request and DMA control bits are all located here. The ECON2 register (Register 19-2) is used to control other top level functions of the module. The ESTAT register (Register 19-3) is used to report the high-level status of the module and Ethernet communications.

The Ethernet SFRs with the 'E' prefix are always accessible, regardless of whether or not the module is enabled.

REGISTER 19-1: ECON1: ETHERNET CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—
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 **TXRST:** Transmit Logic Reset bit
1 = Transmit logic is held in Reset
0 = Normal operation

bit 6 **RXRST:** Receive Logic Reset bit
1 = Receive logic is held in Reset
0 = Normal operation

bit 5 **DMAST:** DMA Start and Busy Status bit
1 = DMA copy or checksum operation is in progress (set by software, cleared by hardware or software)
0 = DMA hardware is Idle

bit 4 **CSUMEN:** DMA Checksum Enable bit
1 = DMA hardware calculates checksums
0 = DMA hardware copies buffer memory

bit 3 **TXRTS:** Transmit Request to Send bit
1 = The transmit logic is attempting to transmit a packet (set by software, cleared by hardware or software)
0 = The transmit logic is Idle

bit 2 **RXEN:** Receive Enable bit
1 = Packets which pass the current filter configuration will be written into the receive buffer
0 = All packets received will be discarded by hardware

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

REGISTER 19-2: ECON2: ETHERNET CONTROL REGISTER 2

R/W-1	R/W-0(1)	R/W-0	U-0	U-0	U-0	U-0	U-0
AUTOINC	PKTDEC	ETHEN	—	—	—	—	—
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 **AUTOINC:** Automatic Buffer Pointer Increment Enable bit
1 = Automatically increment ERDPT or EWRPT registers on reading from, or writing to, EDATA
0 = Do not automatically change ERDPT and EWRPT registers after EDATA is accessed

bit 6 **PKTDEC:** Packet Decrement bit(1)
1 = Decrement the EPKTCNT register by one
0 = Leave EPKTCNT unchanged

bit 5 **ETHEN:** Ethernet Module Enable bit
1 = Ethernet module is enabled
0 = Ethernet module is disabled

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

Note 1: This bit is automatically cleared once it is set.

REGISTER 19-3: ESTAT: ETHERNET STATUS REGISTER

U-0	R/C-0	U-0	R/C-0	U-0	R-0	R/C-0	R-0
—	BUFER	—	r	—	RXBUSY	TXABRT	PHYRDY
bit 7							bit 0

Legend:	r = Reserved bit		
R = Readable bit	C = Clearable 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 **BUFER:** Ethernet Buffer Error Status bit
1 = An Ethernet read or write has generated a buffer error (overrun or underrun)
0 = No buffer error has occurred

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

bit 4 **Reserved:** Write as '0'

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

bit 2 **RXBUSY:** Receive Busy bit
1 = Receive logic is receiving a data packet
0 = Receive logic is Idle

bit 1 **TXABRT:** Transmit Abort Error bit
1 = The transmit request was aborted
0 = No transmit abort error

bit 0 **PHYRDY:** Ethernet PHY Clock Ready bit
1 = Ethernet PHY start-up timer has expired; PHY is ready
0 = Ethernet PHY start-up timer is still counting; PHY is not ready

19.2.4 MAC AND MII REGISTERS

These SFRs are used to control the operations of the MAC, and through the MIIM, the PHY. The MAC and MII registers occupy data addresses, E80h-E85h, E8Ah and EA0h through EB9h.

Although MAC and MII registers appear in the general memory map of the microcontroller, these registers are embedded inside the MAC module. Host interface logic translates the microcontroller data/address bus data to be able to access these registers. The host interface logic imposes restrictions on how firmware is able to access the MAC and MII SFRs. See the following notes.

Note 1: Do not access the MAC and MII SFRs unless the Ethernet module is enabled (ETHEN = 1).

2: Back-to-back accesses of MAC or MII registers are not supported. Between any instruction which addresses a MAC or MII register, at least one NOP or other instruction must be executed.

The three MACON registers control specific MAC operations and packet configuration operations. They are shown in Register 19-4 through Register 19-6.

The MII registers are used to control the MIIM interface and serve as the communication channel with the PHY registers. They are shown in Register 19-7 and Register 19-8.

REGISTER 19-4: MACON1: MAC CONTROL REGISTER 1

U-0	U-0	U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN
bit 7							bit 0

Legend:	r = Reserved 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 7-5 **Unimplemented:** Read as ‘0’

bit 4 **Reserved:** Do not use

bit 3 **TXPAUS:** Pause Control Frame Transmission Enable bit
1 = Allow the MAC to transmit pause control frames (needed for flow control in full duplex)
0 = Disallow pause frame transmissions

bit 2 **RXPAUS:** Pause Control Frame Reception Enable bit
1 = Inhibit transmissions when pause control frames are received (normal operation)
0 = Ignore pause control frames which are received

bit 1 **PASSALL:** Pass All Received Frames Enable bit
1 = Control frames received by the MAC will be written into the receive buffer if not filtered out
0 = Control frames will be discarded after being processed by the MAC (normal operation)

bit 0 **MARXEN:** MAC Receive Enable bit
1 = Enable packets to be received by the MAC
0 = Disable packet reception

REGISTER 19-5: MACON3: MAC CONTROL REGISTER 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX
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 **PADCFG<2:0>:** Automatic Pad and CRC Configuration bits

111 = All short frames are zero-padded to 64 bytes and a valid CRC will then be appended
110 = No automatic padding of short frames
101 = MAC automatically detects VLAN protocol frames which have a 8100h type field and automatically pad to 64 bytes. If the frame is not a VLAN frame, it is padded to 60 bytes. After padding, a valid CRC is appended.
100 = No automatic padding of short frames
011 = All short frames are zero-padded to 64 bytes and a valid CRC is appended
010 = No automatic padding of short frames
001 = All short frames are zero-padded to 60 bytes and a valid CRC is appended
000 = No automatic padding of short frames

bit 4 **TXCRCEN:** Transmit CRC Enable bit

1 = MAC appends a valid CRC to all frames transmitted, regardless of the PADCFG<2:0> bits. TXCRCEN must be set if the PADCFG bits specify that a valid CRC is appended.
0 = MAC does not append a CRC. The last 4 bytes are checked and if it is an invalid CRC, it is reported in the transmit status vector.

bit 3 **PHDREN:** Proprietary Header Enable bit

1 = Frames presented to the MAC contain a 4-byte proprietary header which is not used when calculating the CRC
0 = No proprietary header is present; the CRC covers all data (normal operation)

bit 2 **HFRMEN:** Huge Frame Enable bit

1 = Jumbo frames and frames of any illegal size are allowed to be transmitted and received
0 = Frames bigger than MAMXFL are truncated when transmitted or received

bit 1 **FRMLNEN:** Frame Length Checking Enable bit

1 = The type/length field of transmitted and received frames is checked. If it represents a length, the frame size is compared and mismatches are reported in the transmit/receive status vector.
0 = Frame lengths are not compared with the type/length field

bit 0 **FULDPX:** MAC Full-Duplex Enable bit

1 = MAC operates in Full-Duplex mode; application must also set PDPXMD (PHCON1<8>)
0 = MAC operates in Half-Duplex mode; application must also clear PDPXMD

REGISTER 19-6: MACON4: MAC CONTROL REGISTER 4

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R-0	R-0
—	DEFER	r	r	—	—	r	r
bit 7							bit 0

Legend:	r = Reserved 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 **Unimplemented:** Read as ‘0’

bit 6 **DEFER:** Defer Transmission Enable bit (applies to half duplex only)
1 = When the medium is occupied, the MAC waits indefinitely for it to become free when attempting to transmit (use this setting for IEE 802.3 compliance)
0 = When the medium is occupied, the MAC aborts the transmission after the excessive deferral limit is reached

bit 5-4 **Reserved:** Maintain as ‘0’

bit 3-2 **Unimplemented:** Read as ‘0’

bit 1-0 **Reserved:** Maintain as ‘0’

REGISTER 19-7: MICMD: MII COMMAND REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	MIISCAN	MIIRD
bit 7							bit 0

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

bit 7-2 **Unimplemented:** Read as ‘0’

bit 1 **MIISCAN:** MII Scan Enable bit
1 = PHY register at MIREGADR is continuously read and the data is placed in the MIRD registers
0 = No MII Management scan operation is in progress

bit 0 **MIIRD:** MII Read Enable bit
1 = PHY register at MIREGADR is read once and the data is placed in the MIRD registers
0 = No MII Management read operation is in progress

REGISTER 19-8: MISTAT: MII STATUS REGISTER

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	r	NVALID	SCAN	BUSY
bit 7							bit 0

Legend:	r = Reserved 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 7-4 **Unimplemented:** Read as '0'

bit 3 **Reserved:** Do not use

bit 2 **NVALID:** MII Management Read Data Not Valid bit
- 1 = The contents of the MIRD registers are not valid yet
- 0 = The MII Management read cycle has completed and the MIRD registers have been updated

bit 1 **SCAN:** MII Management Scan Operation bit
- 1 = MII Management scan operation is in progress
- 0 = No MII Management scan operation is in progress

bit 0 **BUSY:** MII Management Busy bit
- 1 = A PHY register is currently being read, or written to. For internal synchronization, the hardware will delay setting this bit for two TCY following a firmware command, which sets the MIISCAN or MIIRD bits, or writes to the MIWRH register.
- 0 = The MII Management interface is Idle

19.2.5 PHY REGISTERS

The PHY registers provide configuration and control of the PHY module, as well as status information about its operation. All PHY registers are 16 bits in width.

PHY registers are accessed with a 5-bit address, for a total of 32 possible registers; of these, only 7 addresses are implemented. The implemented registers are listed in Table 19-3. The main PHY Control registers are described in Register 19-9 through Register 19-13. The other PHY Control and Status registers are described later in this chapter.

Unimplemented registers must never be written to. Reading these locations will return indeterminate data. Within implemented registers, all reserved bit locations that are listed as writable must always be written with the value provided in the register description. When read, these reserved bits can be ignored.

Thy PHY registers are only accessible through the MII Management interface. They must not be read or written to until the PHY start-up timer has expired and the PHYRDY bit (ESTAT<0>) is set.

19.2.5.1 PHSTAT Registers

The PHSTAT1 and PHSTAT2 registers contain read-only bits that show the current status of the PHY module's operations, particularly the conditions of the communications link to the rest of the network.

The PHSTAT1 register (Register 19-10) contains the LLSTAT bit. The bit clears and latches low if the physical layer link has gone down since the last read of the register. The application can periodically poll LLSTAT to determine exactly when the link fails. It may be particularly useful if the link change interrupt is not used.

The PHSTAT2 register (Register 19-12) contains status bits which report if the PHY module is linked to the network and whether or not it is transmitting or receiving.

19.2.5.2 Accessing PHY Registers

As already mentioned, the PHY registers exist in a different memory space and are not directly accessible by the microcontroller. Instead, they are addressed through a special set of MII registers in the Ethernet SFR bank that implement a Media Independent Interface Management (MIIM).

Access is similar to that of the Ethernet buffer, but uses separate read and write buffers (MIRDH:MIRDL and MIWRH:MIWRL) and a 5-bit address register (MIREGADR). In addition, the MICMD and MISTAT registers are used to control read and write operations.

To read from a PHY register:

1. Write the address of the PHY register to be read into the MIREGADR register.
2. Set the MIIRD bit (MICMD<0>). The read operation begins and the BUSY bit (MISTAT<0>) is set after two TCY.
3. Wait 10.24 μs, then poll the BUSY bit to be certain that the operation is complete. When the MAC has obtained the register contents, the BUSY bit will clear itself. While BUSY is set, the user application should not start any MIISCAN operations or write to the MIWRH register.
4. Clear the MIIRD bit.
5. Read the entire 16 bits of the PHY register from the MIRDL and MIRDH registers.

To write to a PHY register:

1. Write the address of the PHY register to be written into the MIREGADR register.
2. Write the lower 8 bits of data to write into the MIWRL register.
3. Write the upper 8 bits of data to write into the MIWRH register. Writing to this register automatically begins the MII transaction, so it must be written to after MIWRL. The BUSY bit is set automatically after two TCY.

The PHY register is written after the MII operation completes, which takes 10.24 μs. When the write operation has completed, the BUSY bit will clear itself. The application should not start any MII scan or read operations while busy.

When a PHY register is written to, the entire 16 bits are written at once; selective bit and/or byte writes are not implemented. If it is necessary to reprogram only select bits in the register, the controller must first read the PHY register, modify the resulting data and then write the data back to the PHY register.

The MAC can also be configured to perform automatic back-to-back read operations on a PHY register. To perform this scan operation:

1. Write the address of the PHY register to be scanned into the MIREGADR register.
2. Set the MIISCAN bit (MICMD<1>). The scan operation begins and the BUSY bit is set after two TCY.

After MIISCAN is set, the NVALID (MISTAT<2>), SCAN and BUSY bits are also set. The first read operation will complete after 10.24 μs. Subsequent reads will be done and the MIRDL and MIRDH registers will be continuously updated automatically at the same interval until the operation is cancelled. The NVALID bit may be polled to determine when the first read operation is complete.

There is no status information which can be used to determine when the MIRD registers are updated. Since only one MII register can be read at a time, it must not be assumed that the values of MIRDL and MIRDH were read from the PHY at exactly the same time during a scan operation.

MIISCAN should remain set as long as the scan operation is desired. The BUSY and SCAN bits are automatically cleared after MIISCAN is set to '0' and the last read sequence is completed. MIREGADR should not be updated while MIISCAN is set.

Starting new PHY operations, such as a read operation or writing to the MIWRH register, must not be done while a scan is underway. The operation can be cancelled by clearing the MIISCAN bit and then polling the BUSY bit. New operations may be started after the BUSY bit is cleared.

TABLE 19-3: PIC18F97J60 FAMILY PHY REGISTER SUMMARY

Addr	Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values
00h	PHCON1	r	r	—	—	r	r	—	PDPXMD	r	—	—	—	—	—	—	—	00-- 00-0 0--- ----
01h	PHSTAT1	—	—	—	r	r	—	—	—	—	—	—	—	—	LLSTAT	r	—	---1 1--- ---- -00-
10h	PHCON2	—	FRCLNK	r	r	r	r	r	HDLDIS	r	r	r	RXAPDIS	r	r	r	r	-000 0000 0000 0000
11h	PHSTAT2	—	—	TXSTAT	RXSTAT	COLSTAT	LSTAT	r	—	—	—	r	—	—	—	—	—	--00 00x- --0- ----
12h	PHIE	r	r	r	r	r	r	r	r	r	r	r	PLNKIE	r	r	PGEIE	r	xxxx xxxx xx00 xx00
13h	PHIR	r	r	r	r	r	r	r	r	r	r	r	PLNKIF	r	PGIF	r	r	xxxx xxxx xx00 00x0
14h	PHLCON	r	r	r	r	LACFG3	LACFG2	LACFG1	LACFG0	LBCFG3	LBCFG2	LBCFG1	LBCFG0	LFRQ1	LFRQ0	STRCH	r	0011 0100 0010 001x

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', r = reserved, do not modify. Shaded cells are unimplemented, read as '0'.

REGISTER 19-9: PHCON1: PHY CONTROL REGISTER 1

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0	R/W-0
r	r	—	—	r	r	—	PDPXMD
bit 15							bit 8

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

Legend:	r = Reserved 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 15-14 **Reserved:** Write as ‘0’

bit 13-12 **Unimplemented:** Read as ‘0’

bit 11-10 **Reserved:** Write as ‘0’

bit 9 **Unimplemented:** Read as ‘0’

bit 8 **PDPXMD:** PHY Duplex Mode bit
1 = PHY operates in Full-Duplex mode; application must also set FULDPX (MACON3<0>)
0 = PHY operates in Half-Duplex mode, application must also clear FULDP

bit 7 **Reserved:** Maintain as ‘0’

bit 6-0 **Unimplemented:** Read as ‘0’

REGISTER 19-10: PHSTAT1: PHYSICAL LAYER STATUS REGISTER 1

U-0	U-0	U-0	R-1	R-1	U-0	U-0	U-0
—	—	—	r	r	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/LL-0	R/LH-0	U-0
—	—	—	—	—	LLSTAT	r	—
bit 7							bit 0

Legend:	‘1’ = Bit is set	r = Reserved bit	
R = Read-only bit	‘0’ = Bit is cleared	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	R/L = Read-Only Latch bit	LL = Latches Low bit	LH = Latches High bit

bit 15-13 **Unimplemented:** Read as ‘0’

bit 12-11 **Reserved:** Read as ‘1’

bit 10-3 **Unimplemented:** Read as ‘0’

bit 2 **LLSTAT:** PHY Latching Link Status bit
1 = Link is up and has been up continously since PHSTAT1 was last read
0 = Link is down or was down for a period since PHSTAT1 was last read

bit 1 **Reserved:** Ignore on read

bit 0 **Unimplemented:** Read as ‘0’

REGISTER 19-11: PHCON2: PHY CONTROL REGISTER 2

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	FRCLNK	r	r	r	r	r	HDLDIS
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
r	r	r	RXAPDIS	r	r	r	r
bit 7							bit 0

Legend:	r = Reserved 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 15 **Unimplemented:** Read as ‘0’

bit 14 **FRCLNK:** PHY Force Linkup bit
1 = Force linkup even when no link partner is detected (transmission is always allowed)
0 = Normal operation (PHY blocks transmission attempts unless a link partner is attached)

bit 13-9 **Reserved:** Write as ‘0’

bit 8 **HDLDIS:** PHY Half-Duplex Loopback Disable bit
1 = Normal PHY operation
0 = Reserved

bit 7-5 **Reserved:** Write as ‘0’

bit 4 **RXAPDIS:** RX+/RX- Operating mode bit
1 = Normal operation
0 = Reserved

bit 3-0 **Reserved:** Write as ‘0’

Note: Improper Ethernet operation may result if HDLDIS or RXAPDIS is cleared, which is the Reset default. Always initialize these bits set before using the Ethernet module.

REGISTER 19-12: PHSTAT2: PHYSICAL LAYER STATUS REGISTER 2

U-0	U-0	R-0	R-0	R-0	R-0	R-x	U-0
—	—	TXSTAT	RXSTAT	COLSTAT	LSTAT	r	—
bit 15							bit 8

U-0	U-0	R-0	U-0	U-0	U-0	U-0	U-0
—	—	r	—	—	—	—	—
bit 7							bit 0

Legend:	r = Reserved 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 15-14 **Unimplemented:** Read as ‘0’

bit 13 **TXSTAT:** PHY Transmit Status bit
1 = PHY is transmitting data
0 = PHY is not transmitting data

bit 12 **RXSTAT:** PHY Receive Status bit
1 = PHY is receiving data
0 = PHY is not receiving data

bit 11 **COLSTAT:** PHY Collision Status bit
1 = A collision is occuring (PHY is both transmitting and receiving while in Half-Duplex mode)
0 = A collision is not occuring

bit 10 **LSTAT:** PHY Collision Status bit
1 = Link is up
0 = Link is down

bit 9 **Reserved:** Ignore on read

bit 8-6 **Unimplemented:** Read as ‘0’

bit 5 **Reserved:** Ignore on read

bit 4-0 **Unimplemented:** Read as ‘0’

REGISTER 19-13: PHLCON: PHY MODULE LED CONTROL REGISTER

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0
r	r	r	r	LACFG3	LACFG2	LACFG1	LACFG0
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-x
LBCFG3	LBCFG2	LBCFG1	LBCFG0	LFRQ1	LFRQ0	STRCH	r
bit 7							bit 0

Legend:	r = Reserved 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 15-14 **Reserved:** Write as '0'

bit 13-12 **Reserved:** Write as '1'

bit 11-8 **LACFG<3:0>:** LEDA Configuration bits
0000 = Reserved
0001 = Display transmit activity (stretchable)
0010 = Display receive activity (stretchable)
0011 = Display collision activity (stretchable)
0100 = Display link status
0101 = Display duplex status
0110 = Reserved
0111 = Display transmit and receive activity (stretchable)
1000 = On
1001 = Off
1010 = Blink fast
1011 = Blink slow
1100 = Display link status and receive activity (always stretched)
1101 = Display link status and transmit/receive activity (always stretched)
111x = Reserved

bit 7-4 **LBCFG<3:0>:** LEDB Configuration bits
0000 = Reserved
0001 = Display transmit activity (stretchable)
0010 = Display receive activity (stretchable)
0011 = Display collision activity (stretchable)
0100 = Display link status
0101 = Display duplex status
0110 = Reserved
0111 = Display transmit and receive activity (stretchable)
1000 = On
1001 = Off
1010 = Blink fast
1011 = Blink slow
1100 = Display link status and receive activity (always stretched)
1101 = Display link status and transmit/receive activity (always stretched)
111x = Reserved

bit 3-2 **LFRQ<1:0>:** LED Pulse Stretch Time Configuration bits (see Table 19-1)
11 = Reserved
10 = Stretch LED events by TLSTRCH
01 = Stretch LED events by TMSTRCH
00 = Stretch LED events by TNSTRCH

bit 1 **STRCH:** LED Pulse Stretching Enable bit
1 = Stretchable LED events will cause lengthened LED pulses based on LFRQ<1:0> configuration
0 = Stretchable LED events will only be displayed while they are occurring

bit 0 **Reserved:** Write as '0'

19.3 Ethernet Interrupts

The Ethernet module can generate multiple interrupt conditions. To accommodate all of these sources, the module has its own interrupt logic structure, similar to that of the microcontroller. Separate sets of registers are used to enable and flag different interrupt conditions.

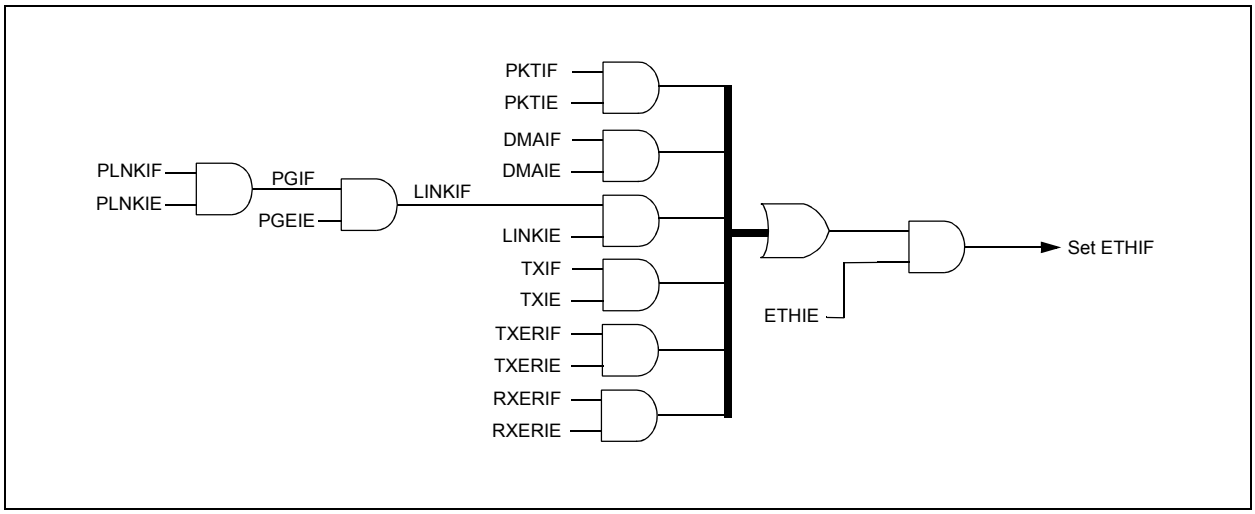
The EIE register contains the individual interrupt enable bits for each source, while the EIR register contains the corresponding interrupt flag bits. When an interrupt occurs, the interrupt flag is set. If the interrupt is enabled in the EIE register, and the corresponding ETHIE Global Interrupt Enable bit is set, the microcontroller's master Ethernet Interrupt Flag (ETHIF) is set, as appropriate (see Figure 19-7).

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

19.3.1 CONTROL INTERRUPT (ETHIE)

The four registers associated with the control interrupts are shown in Register 19-14 through Register 19-17.

FIGURE 19-7: ETHERNET MODULE INTERRUPT LOGIC



REGISTER 19-14: EIE: ETHERNET INTERRUPT ENABLE REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE
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 **PKTIE:** Receive Packet Pending Interrupt Enable bit
1 = Enable receive packet pending interrupt
0 = Disable receive packet pending interrupt

bit 5 **DMAIE:** DMA Interrupt Enable bit
1 = Enable DMA interrupt
0 = Disable DMA interrupt

bit 4 **LINKIE:** Link Status Change Interrupt Enable bit
1 = Enable link change interrupt from the PHY
0 = Disable link change interrupt

bit 3 **TXIE:** Transmit Enable bit
1 = Enable transmit interrupt
0 = Disable transmit interrupt

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

bit 1 **TXERIE:** Transmit Error Interrupt Enable bit
1 = Enable transmit error interrupt
0 = Disable transmit error interrupt

bit 0 **RXERIE:** Receive Error Interrupt Enable bit
1 = Enable receive error interrupt
0 = Disable receive error interrupt

REGISTER 19-15: EIR: ETHERNET INTERRUPT REQUEST (FLAG) REGISTER

U-0	R-0	R/C-0	R-0	R/C-0	U-0	R/C-0	R/C-0
—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF
bit 7							bit 0

Legend:			
R = Readable bit	C = Clearable 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 **PKTIF:** Receive Packet Pending Interrupt Flag bit
- 1 = Receive buffer contains one or more unprocessed packets; cleared only when EPKTCNT is decremented to 0 by setting PKTDEC (ECON2<6>)
- 0 = Receive buffer is empty

bit 5 **DMAIF:** DMA Interrupt Flag bit
- 1 = DMA copy or checksum calculation has completed
- 0 = No DMA interrupt is pending

bit 4 **LINKIF:** Link Change Interrupt Flag bit
- 1 = PHY reports that the link status has changed; read PHIR register to clear
- 0 = Link status has not changed

bit 3 **TXIF:** Transmit Interrupt Flag bit
- 1 = Transmit request has ended
- 0 = No transmit interrupt is pending

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

bit 1 **TXERIF:** Transmit Error Interrupt Flag bit
- 1 = A transmit error has occurred
- 0 = No transmit error has occurred

bit 0 **RXERIF:** Receive Error Interrupt Flag bit
- 1 = A packet was aborted because there is insufficient buffer space, or a buffer overrun has occurred
- 0 = No receive error interrupt is pending

REGISTER 19-16: PHIE: PHY INTERRUPT ENABLE REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
r	r	r	r	r	r	r	r
bit 15							bit 8

R-0	R-0	R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0
r	r	r	PLNKIE	r	r	PGEIE	r
bit 7							bit 0

Legend:	r = Reserved 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 15-6 **Reserved:** Write as '0', ignore on read

bit 5 **Reserved:** Maintain as '0'

bit 4 **PLNKIE:** PHY Link Change Interrupt Enable bit
1 = PHY link change interrupt is enabled
0 = PHY link change interrupt is disabled

bit 3-2 **Reserved:** Write as '0', ignore on read

bit 1 **PGEIE:** PHY Global Interrupt Enable bit
1 = PHY interrupts are enabled
0 = PHY interrupts are disabled

bit 0 **Reserved:** Maintain as '0'

REGISTER 19-17: PHIR: PHY INTERRUPT REQUEST (FLAG) REGISTER

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
r	r	r	r	r	r	r	r
bit 15							bit 8

R-x	R-x	R-0	R/SC-0	R-0	R/SC-0	R-x	R-0
r	r	r	PLNKIF	r	PGIF	r	r
bit 7							bit 0

Legend:	r = Reserved bit		
R = Readable bit	SC = Self-Clearable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-6 **Reserved:** Ignore on read

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

bit 4 **PLNKIF:** PHY Link Change Interrupt Flag bit
1 = PHY link status has changed since PHIR was last read; resets to '0' when read
0 = PHY link status has not changed since PHIR was last read

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

bit 2 **PGIF:** PHY Global Interrupt Flag bit
1 = One or more enabled PHY interrupts have occurred since PHIR was last read; resets to '0' when read
0 = No PHY interrupts have occurred

bit 1 **Reserved:** Ignore on read

bit 0 **Reserved:** Read as '0'

19.3.1.1 Receive Error Interrupt (RXERIF)

The receive error interrupt is used to indicate that a packet being received was aborted due to an error condition. Three errors are possible:

1. No buffer space is available to store the incoming packet (buffer overflow);
2. Receiving another packet would cause the EPKTCNT counter to overflow, because it already contains the value, 255; or
3. The Ethernet RX hardware was not allocated enough memory bandwidth to write the incoming data to the buffer.

When a packet is being received and the receive error occurs, the packet being received will be aborted (permanently lost) and the RXERIF bit will be set to '1'. Once set, RXERIF can only be cleared by firmware or by a Reset condition. If the receive error interrupt and Ethernet interrupt are enabled (both RXERIE and ETHIE are set), an Ethernet interrupt is generated. If the receive error interrupt is not enabled (either RXERIE or ETHIE is cleared), the application may poll RXERIF and take appropriate action.

Normally, upon the first two receive error conditions (buffer overflow or potential EPKTCNT overflow), the application would process any packets pending from the receive buffer, and then make additional room for future packets by advancing the ERXRDPT registers (low byte first) and decrementing the EPKTCNT register. See **Section 19.5.3.3 "Freeing Receive Buffer Space"** for more information on processing packets. Once processed, the application should clear the RXERIF bit.

The third condition (insufficient RX memory bandwidth) can be identified by checking if the BUFER bit (ESTAT<6>) has been set. Memory access errors that set BUFER are generally transient in nature, and do not require run-time resolution. Adjustments to the application and its allocation of buffer memory bandwidth may be necessary if BUFER errors are frequent or persistent.

19.3.1.2 Transmit Error Interrupt (TXERIF)

The transmit error interrupt is used to indicate that a transmit abort has occurred. An abort can occur because of any of the following conditions:

1. More than 15 collisions occurred while attempting to transmit a given packet.
2. A late collision (collision after 64 bytes of a packet had been transmitted) has occurred.
3. The transmission was unable to gain an opportunity to transmit the packet because the medium was constantly occupied for too long. The deferral limit was reached and the DEFER bit (MACON4<6>) was clear.
4. An attempt to transmit a packet larger than the maximum frame length, defined by the MAMXFL registers, was made without setting the HFRMEN bit (MACON3<2>) or per-packet POVERRIDE and PHUGEEN bits.
5. The Ethernet buffer did not have enough memory bandwidth to maintain the required 10 Mbit/s transfer rate (buffer underrun).

Upon any of these conditions, the TXERIF flag is set to '1'. Once set, it can only be cleared by firmware or by a Reset condition. If the transmit error interrupt is enabled (TXERIE and ETHIE are both set), an Ethernet interrupt is generated. If the transmit error interrupt is not enabled (either TXERIE or ETHIE is cleared), the application may poll TXERIF and take appropriate action. Once the interrupt is processed, the flag bit should be cleared.

After a transmit abort, the TXRTS bit (ECON1<3>) will be cleared, the TXABRT bit (ESTAT<1>) becomes set and the transmit status vector will be written at the ETXND registers + 1. The MAC will not automatically attempt to retransmit the packet. The application may wish to read the transmit status vector and BUFER bit to determine the cause of the abort. After determining the problem and solution, the application should clear the BUFER (if set) and TXABRT bits so that future aborts can be detected accurately.

In Full-Duplex mode, Conditions 4 and 5 are the only ones that should cause this interrupt. Condition 5 can be further distinguished as it also sets the BUFER bit. Collisions and other problems related to sharing the network are not possible on full-duplex networks. The conditions, which cause the transmit error interrupt, meet the requirements of the transmit interrupt. As a result, when this interrupt occurs, TXIF will also be simultaneously set.

19.3.1.3 Transmit Interrupt (TXIF)

The transmit interrupt is used to indicate that the requested packet transmission has ended (the TXRTS bit has transitioned from '1' to '0'). Upon transmission completion, abort, or transmission cancellation by the application, the TXIF flag will be set to '1'. If the application did not clear the TXRTS bit, and the TXABRT bit is not set, the packet was successfully transmitted. Once TXIF is set, it can only be cleared in software or by a Reset condition. If the transmit interrupt is enabled (TXIE and ETHIE are both set), an interrupt is generated. If the transmit interrupt is not enabled (either TXIE or ETHIE is cleared), the application may poll the TXIF bit and take appropriate action.

19.3.1.4 Link Change Interrupt (LINKIF)

The LINKIF indicates that the link status has changed. The actual current link status can be obtained from the LLSTAT (PHSTAT1<2>) or LSTAT (PHSTAT2<10>) bits (see Register 19-10 and Register 19-12). Unlike other interrupt sources, the link status change interrupt is created in the integrated PHY module; additional steps must be taken to enable it.

By Reset default, LINKIF is never set for any reason. To receive it, both the PLNKIE and PGEIE bits must be set. When the interrupt is enabled, the LINKIF bit will shadow the contents of the PGIF bit. The PHY only supports one interrupt, so the PGIF bit will always be the same as the PLNKIF bit (when both PHY enable bits are set).

Once LINKIF is set, it can only be cleared in software or by a Reset. If the link change interrupt is enabled (LINKIE, PLNKIE, PGEIE and ETHIE are all set), an interrupt is generated. If the link change interrupt is not enabled (LINKIE, PLNKIE, PGEIE or ETHIE are cleared), the user application may poll the PLNKIF flag and take appropriate action.

The LINKIF bit is read-only. Because reading PHY registers requires a non-negligible period of time, the application may instead set PLNKIE and PGEIE, then poll the LINKIF flag bit. Performing an MII read on the PHIR register will clear the LINKIF, PGIF and PLNKIF bits automatically, and allow for future link status change interrupts. See **Section 19.2.5 "PHY Registers"** for information on accessing the PHY registers.

19.3.1.5 DMA Interrupt (DMAIF)

The DMA interrupt indicates that the DMA module has completed its memory copy or checksum calculation (the DMAST bit has transitioned from '1' to '0'). Additionally, this interrupt will be caused if the application cancels a DMA operation by manually clearing the DMAST bit. Once set, DMAIF can only be cleared by the firmware or by a Reset condition. If the DMA interrupt is enabled, an Ethernet interrupt is generated. If the DMA interrupt is not enabled, the user application may poll the DMAIF flag status and take appropriate action. Once processed, the flag bit should be cleared.

19.3.1.6 Receive Packet Pending Interrupt (PKTIF)

The receive packet pending interrupt is used to indicate the presence of one or more data packets in the receive buffer and to provide a notification means for the arrival of new packets. When the receive buffer has at least one packet in it, the PKTIF flag bit is set. In other words, this interrupt flag will be set any time the Ethernet Packet Count register (EPKTCNT) is non-zero.

When the receive packet pending interrupt is enabled (both PKTIE and ETHIE are set), an Ethernet interrupt is generated whenever a new packet is successfully received and written into the receive buffer. If the receive packet pending interrupt is not enabled (either PKTIE or ETHIE is cleared), the user application may poll the PKTIF bit and take appropriate action.

The PKTIF bit can only be cleared indirectly in software, by decrementing the EPKTCNT register to '0', or by a Reset condition. See **Section 19.5.3 "Receiving Packets"** for more information about clearing the EPKTCNT register. When the last data packet in the receive buffer is processed, EPKTCNT becomes zero and the PKTIF bit is automatically cleared.

19.3.2 ETHERNET INTERRUPTS AND WAKE-ON-LAN

The Ethernet interrupt structure implements a version of Wake-on-LAN, also called Remote Wake-up, using a Magic Packet data packet. This allows the application to conserve power in Idle mode, and then return to full-power operation only when a specific wake-up packet is received.

For Remote Wake-up to work, the Ethernet module must remain enabled at all times. It is also necessary to configure the receive filters to select for Magic Packets. For more information on filter configuration, see **Section 19.8 "Receive Filters"**.

To configure the microcontroller for Remote Wake-up:

1. With the Ethernet module enabled and in normal operating configuration, enable the CRC post-filter and Magic Packets filter (ERXFCON<5,3> = 1).
2. Finish processing any pending packets in the Ethernet buffer.
3. Enable Ethernet interrupts at the microcontroller level (PIE2<5> = 1) and the receive packet pending interrupt at the module level (EIE<6> = 1).
4. Place the microcontroller in PRI_IDLE mode (with the primary clock source selected and OSCCON<7> = 1, execute the SLEEP instruction).

In this configuration, the receipt of a Magic Packet data packet will cause a receive packet pending interrupt. This, in turn, will cause the microcontroller to wake-up from the interrupt.

19.4 Module Initialization

Before the Ethernet module can be used to transmit and receive packets, certain device settings must be initialized. Depending on the application, some configuration options may need to be changed. Normally, these tasks may be accomplished once after Reset and do not need to be changed thereafter.

Before any other configuration actions are taken, it is recommended that the module be enabled by setting the ETHEN bit (ECON2<5>). This reduces the Idle time that might otherwise result while waiting for the PHYRDY flag to become set.

19.4.1 RECEIVE BUFFER

Before receiving any packets, the receive buffer must be initialized by setting the ERXST and ERXND Pointers. All memory between and including the ERXST and ERXND addresses will be dedicated to the receive hardware. The ERXST Pointers must be programmed with an even address while the ERXND Pointers must be programmed with an odd address.

Applications expecting large amounts of data and frequent packet delivery may wish to allocate most of the memory as the receive buffer. Applications that may need to save older packets, or have several packets ready for transmission, should allocate less memory.

When programming the ERXST or ERXND Pointers, the ERXWRPT Pointer registers will automatically be updated with the value in the ERXST registers. The address in the ERXWRPT registers will be used as the starting location when the receive hardware begins writing received data. When the ERXST and ERXND Pointers are initialized, the ERXRDPT registers should additionally be programmed with the value of the ERXND registers. To program the ERXRDPT registers, write to ERXRDPTL first, followed by ERXRDPTH. See **Section 19.5.3.3 "Freeing Receive Buffer Space"** for more information.

19.4.2 TRANSMISSION BUFFER

All memory which is not used by the receive buffer is considered to be the transmission buffer. Data which is to be transmitted should be written into any unused space. After a packet is transmitted, however, the hardware will write a 7-byte status vector into memory after the last byte in the packet. Therefore, the application should leave at least 7 bytes between each packet and the beginning of the receive buffer.

19.4.3 RECEIVE FILTERS

The appropriate receive filters should be enabled or disabled by writing to the ERXFCON register. See **Section 19.8 "Receive Filters"** for information on how to configure it.

19.4.4 WAITING FOR THE PHY START-UP TIMER

If the initialization procedure is being executed immediately after enabling the module (setting the ETHEN bit to '1'), the PHYRDY bit should be polled to make certain that enough time (1 ms) has elapsed before proceeding to modify the PHY registers. For more information on the PHY start-up timer, see **Section 19.1.3.1 "Start-up Timer"**.

19.4.5 MAC INITIALIZATION SETTINGS

Several of the MAC registers require configuration during initialization. This only needs to be done once during initialization; the order of programming is unimportant.

1. Set the MARXEN bit (MACON1<0>) to enable the MAC to receive frames. If using full duplex, most applications should also set TXPAUS and RXPAUS to allow IEEE defined flow control to function.
2. Configure the PADCFG<2:0>, TXCRCEN and FULDPX bits in the MACON3 register. Most applications should enable automatic padding to at least 60 bytes and always append a valid CRC. For convenience, many applications may wish to set the FRMLNEN bit as well to enable frame length status reporting. The FULDPX bit should be set if the application will be connected to a full-duplex configured remote node; otherwise leave it clear.
3. Configure the bits in MACON4. For maintaining compliance with IEEE 802.3, be certain to set the DEFER bit (MACON4<6>).
4. Program the MAMXFL registers with the maximum frame length to be permitted to be received or transmitted. Normal network nodes are designed to handle packets that are 1518 bytes or less; larger packets are not supported by IEEE 802.3.
5. Configure the MAC Back-to-Back Inter-Packet Gap register, MABBIPG, with 15h (when Full-Duplex mode is used) or 12h (when Half-Duplex mode is used). Refer to Register 19-18 for a more detailed description of configuring the inter-packet gap.
6. Configure the MAC Non Back-to-Back Inter-Packet Gap Low Byte register, MAIPGL, with 12h.
7. If half duplex is used, configure the MAC Non Back-to-Back Inter-Packet Gap High Byte register, MAIPGH, with 0Ch.
8. Program the local MAC address into the MAADR1:MAADR6 registers.

REGISTER 19-18: MABBIPG: MAC BACK-TO-BACK INTER-PACKET GAP REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0
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 6-0 **BBIPG<6:0>:** Back-to-Back Inter-Packet Gap Delay Time bits

When FULDPX (MACON3<0>) = 1:
Nibble time offset delay between the end of one transmission and the beginning of the next in a back-to-back sequence. The register value should be programmed to the desired period in nibble times minus 3. The recommended setting is 15h which represents the minimum IEEE specified Inter-Packet Gap (IPG) of 9.6 μs.

When FULDPX (MACON3<0>) = 0:
Nibble time offset delay between the end of one transmission and the beginning of the next in a back-to-back sequence. The register value should be programmed to the desired period in nibble times minus 6. The recommended setting is 12h which represents the minimum IEEE specified Inter-Packet Gap (IPG) of 9.6 μs.

19.4.6 PHY INITIALIZATION SETTINGS

Depending on the application, bits in three of the PHY module's registers may also require configuration.

The PDPXMD bit (PHCON1<8>) controls the PHY half/full-duplex configuration. The application must program the bit properly, along with the FULDPX bit (MACON3<0>).

The HDLDIS bit (PHCON2<8>) disables automatic loopback of data. For proper operation, always set both HDLDIS and RXAPDIS (PHCON2<4>).

The PHY register, PHLCON (Register 19-13), controls the outputs of LEDA and LEDB. If an application requires a LED configuration other than the default, alter this register to match the new requirements. The settings for LED operation are discussed in **Section 19.1.2 "LED Configuration"**.

19.4.7 DISABLING THE ETHERNET MODULE

There may be circumstances during which the Ethernet module is not needed for prolonged periods. For example, in situations where the application only needs to transmit or receive Ethernet packets on the occurrence of a particular event. In these cases, the module can be selectively powered down.

To selectively disable the module:

1. Turn off packet reception by clearing the RXEN bit.
2. Wait for any in-progress packets to finish being received by polling the RXBUSY bit (ESTAT<2>). This bit should be clear before proceeding.
3. Wait for any current transmissions to end by confirming that the TXRTS bit (ECON1<3>) is clear.
4. Clear the ETHEN bit. This removes power and clock sources from the module, and makes the PHY registers inaccessible. The PHYRDY bit is also cleared automatically.

19.5 Transmitting and Receiving Data

The Ethernet protocol (IEEE Standard 802.3) provides an extremely detailed description of the 10 Mbps, frame-based serial communications system. Before discussing the actual use of the Ethernet module, a brief review of the structure of a typical Ethernet data frame may be appropriate. It is assumed that users already have some familiarity with IEEE 802.3. Those requiring more information should refer to the official standard, or other Ethernet reference texts, for a more comprehensive explanation.

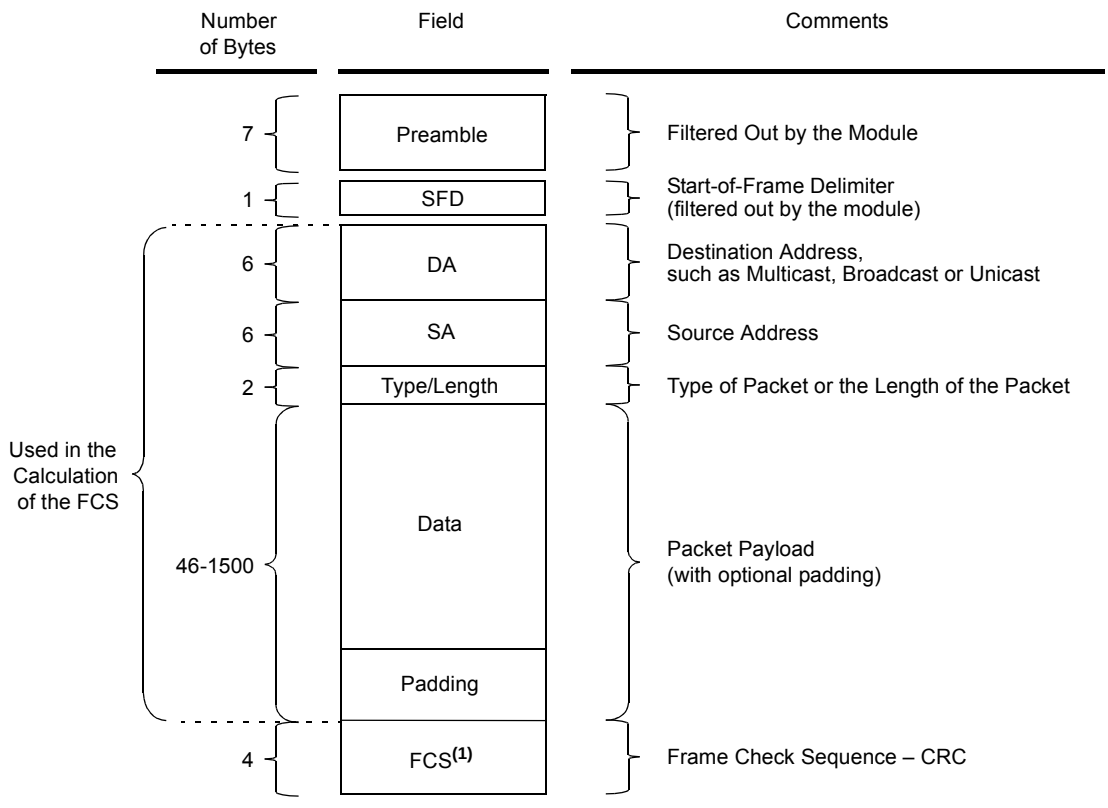
19.5.1 PACKET FORMAT

Normal IEEE 802.3 compliant Ethernet frames are between 64 and 1518 bytes long. They are made up of five or six different fields: a destination MAC address, a source MAC address, a type/length field, data payload, an optional padding field and a Cyclic Redundancy Check (CRC). Additionally, when transmitted on the Ethernet medium, a 7-byte preamble field and Start-of-Frame (SOF) delimiter byte are appended to the beginning of the Ethernet packet. Thus, traffic seen on the twisted-pair cabling will appear as shown in Figure 19-8.

19.5.1.1 Preamble/Start-of-Frame Delimiter

When transmitting and receiving data with the Ethernet module, the preamble and Start-of-Frame delimiter bytes are automatically generated, or stripped from the packets, when they are transmitted or received. It can also automatically generate CRC fields and padding as needed on transmission, and verify CRC data on reception. The user application does not need to create or process these fields, or manually verify CRC data. However, the padding and CRC fields are written into the receive buffer when packets arrive, so they may be evaluated by the user application as needed.

FIGURE 19-8: ETHERNET PACKET FORMAT



Note 1: The FCS is transmitted starting with bit 31 and ending with bit 0.

19.5.1.2 Destination Address

The destination address field is a 6-byte field filled with the MAC address of the device that the packet is directed to. If the Least Significant bit in the first byte of the MAC address is set, the address is a Multicast destination. For example, 01-00-00-00-F0-00 and 33-45-67-89-AB-CD are Multicast addresses, while 00-00-00-00-F0-00 and 32-45-67-89-AB-CD are not.

Packets with Multicast destination addresses are designed to arrive and be important to a selected group of Ethernet nodes. If the destination address field is the reserved Multicast address, FF-FF-FF-FF-FF-FF, the packet is a Broadcast packet and it will be directed to everyone sharing the network. If the Least Significant bit in the first byte of the MAC address is clear, the address is a Unicast address and will be designed for usage by only the addressed node.

The Ethernet module incorporates receive filters which can be used to discard or accept packets with Multicast, Broadcast and/or Unicast destination addresses. When transmitting packets, the application is responsible for writing the desired destination address into the transmit buffer.

19.5.1.3 Source Address

The source address field is a 6-byte field filled with the MAC address of the node which created the Ethernet packet. Users of the Ethernet module must generate a unique MAC address for each and every microcontroller used.

MAC addresses consist of two portions. The first three bytes are known as the Organizationally Unique Identifier (OUI). OUIs are distributed by the IEEE. The last three bytes are address bytes at the discretion of the company that purchased the OUI.

When transmitting packets, the assigned source MAC address must be written into the transmit buffer by the application. The module will not automatically transmit the contents of the MAADR registers which are used for the Unicast receive filter.

19.5.1.4 Type/Length

The type/length field is a 2-byte field which defines which protocol the following packet data belongs to. Alternately, if the field is filled with the contents of 05DCh (1500) or any smaller number, the field is considered a length field, and it specifies the amount of non-padding data which follows in the data field. Users implementing proprietary networks may choose to treat this field as a length field, while applications implementing protocols, such as the Internet Protocol (IP) or Address Resolution Protocol (ARP), should program this field with the appropriate type defined by the protocol's specification when transmitting packets.

19.5.1.5 Data

The data field is a variable length field anywhere from 0 to 1500 bytes. Larger data packets will violate Ethernet standards and will be dropped by most Ethernet nodes. The Ethernet module, however, is capable of transmitting and receiving larger packets when the Huge Frame Enable bit, HFRMEN, is set (MACON3<2> = 1).

19.5.1.6 Padding

The padding field is a variable length field added to meet IEEE 802.3 specification requirements when small data payloads are used. The destination, source, type, data and padding of an Ethernet packet must be no smaller than 60 bytes. Adding the required 4-byte CRC field, packets must be no smaller than 64 bytes. If the data field is less than 46 bytes long, a padding field is required.

When transmitting packets, the Ethernet module automatically generates zero-padding if the PADCFG<2:0> bits (MACON3<7:5>) are configured for this. Otherwise, the user application will need to add any padding to the packet before transmitting it. The module will not prevent the transmission of undersized packets should the application command such an action.

When receiving packets, the module automatically rejects packets which are less than 18 bytes. All packets, 18 bytes and larger, will be subject to the standard receive filtering criteria and may be accepted as normal traffic. Since the module only rejects packets smaller than 18 bytes, it is important that the firmware check the length of every received packet and reject packets which are smaller than 64 bytes to meet IEEE 802.3 specification requirements.

19.5.1.7 CRC

The CRC field is a 4-byte field which contains an industry standard, 32-bit CRC, calculated with the data from the destination, source, type, data and padding fields. It provides a way of detecting corrupted Ethernet frames, as well as junk data fragments resulting from packet collisions or another host's aborted transmissions.

When receiving packets, the Ethernet module will check the CRC of each incoming packet. If the CRCEN bit is set, packets with invalid CRCs will automatically be discarded. If CRCEN is clear and the packet meets all other receive filtering criteria, the packet will be written into the receive buffer and the application will be able to determine if the CRC was valid by reading the receive status vector (see **Section 19.5.3 "Receiving Packets"**).

When transmitting packets, the module automatically generates a valid CRC and transmits it if the PADCFG<2:0> bits are configured for this. Otherwise, the user application must generate the CRC and place it in the transmit buffer. Given the complexity of calculating a CRC, it is highly recommended to allow the module to automatically calculate and include the CRC.

19.5.2 TRANSMITTING PACKETS

The Ethernet module's MAC will automatically generate the preamble and Start-of-Frame (SOF) delimiter fields when transmitting. Additionally, the MAC can generate any padding (if needed) and the CRC if configured to do so. The application must generate and write all other frame fields into the buffer memory for transmission.

In addition, the Ethernet module requires a single per-packet control byte to precede the packet for transmission. The control byte is organized as shown in Figure 19-9. Before transmitting packets, the MAC registers, which alter the transmission characteristics, should be initialized as documented in **Section 19.4 "Module Initialization"**.

FIGURE 19-9: FORMAT FOR PER-PACKET CONTROL BYTES

—	—	—	—	PHUGEEN	PPADN	PCRCEN	POVERRIDE
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **PHUGEEN:** Per-Packet Huge Frame Enable bit

When POVERRIDE = 1:

1 = The packet will be transmitted in whole

0 = The MAC will transmit up to the number of bytes specified by the MAMXFL registers. If the packet is larger than the bytes specified, it will be aborted after the MAMXFL registers specification is reached.

When POVERRIDE = 0:

This bit is ignored.

bit 2 **PPADN:** Per-Packet Padding Enable bit

When POVERRIDE = 1:

1 = The packet will be zero-padded to 60 bytes if it is less than 60 bytes

0 = The packet will be transmitted without adding any padding bytes

When POVERRIDE = 0:

This bit is ignored.

bit 1 **PCRCEN:** Per-Packet CRC Enable bit

When POVERRIDE = 1:

1 = A valid CRC will be calculated and attached to the frame

0 = No CRC will be appended. The last 4 bytes of the frame will be checked for validity as a CRC.

When POVERRIDE = 0:

This bit is ignored.

bit 0 **POVERRIDE:** Per-Packet Override bit

1 = The values of PCRCEN, PPADN and PHUGEEN will override the configuration defined by MACON3

0 = The values in MACON3 will be used to determine how the packet will be transmitted

An example of how the entire assembled transmit packet looks in memory is shown in Figure 19-10. To construct and transmit a packet in this fashion:

1. Set the ETXST Pointers to an appropriate unused location in the buffer. This will be the location of the per-packet control byte. In the example, it would be 0120h. It is recommended that an even address be used for the ETXST Pointers.
2. Using EDATA and the EWRPT registers, sequentially write the packet data to the Ethernet buffer. In order, write the data for the per-packet control byte, the destination address, the source MAC address, the type/length and the data payload.
3. Set the ETXND Pointers to point to the last byte in the data payload. In the example, it would be programmed to 0156h.
4. Clear the TXIF flag bit (EIR<3>), and set the TXIE (EIE<3>) and ETHIE bits to enable an interrupt when done (if desired).
5. Start the transmission process by setting the TXRTS bit (ECON1<3>).

If a DMA operation was in progress while the TXRTS bit was set, the module will wait until the DMA operation is complete before attempting to transmit the packet. This possible delay is required because the DMA and transmission engine share the same memory arbiter channel. Similarly, if the DMAST bit is set after TXRTS is already set, the DMA will wait until the TXRTS bit becomes clear before doing anything.

While the transmission is in progress, the ETXST and ETXND Pointers should not be modified. If it is necessary to cancel the transmission, clear the TXRTS bit.

When the packet is finished transmitting, or was aborted due to an error/cancellation, several things occur:

- The TXRTS bit is cleared.
- A 7-byte transmit status vector is written to the buffer at the location pointed to by the ETXND Pointers + 1.
- The TXIF flag is set.
- An interrupt will be generated (if enabled).
- The ETXST and ETXND Pointers will not be modified.

To check if the packet was successfully transmitted, read the TXABRT bit. If it has been set, poll the BUFER bit in addition to the various fields in the transmit status vector to determine the cause. The transmit status vector is organized as shown in Table 19-4. Multi-byte fields are written in little-endian format.

FIGURE 19-10: SAMPLE TRANSMIT PACKET LAYOUT

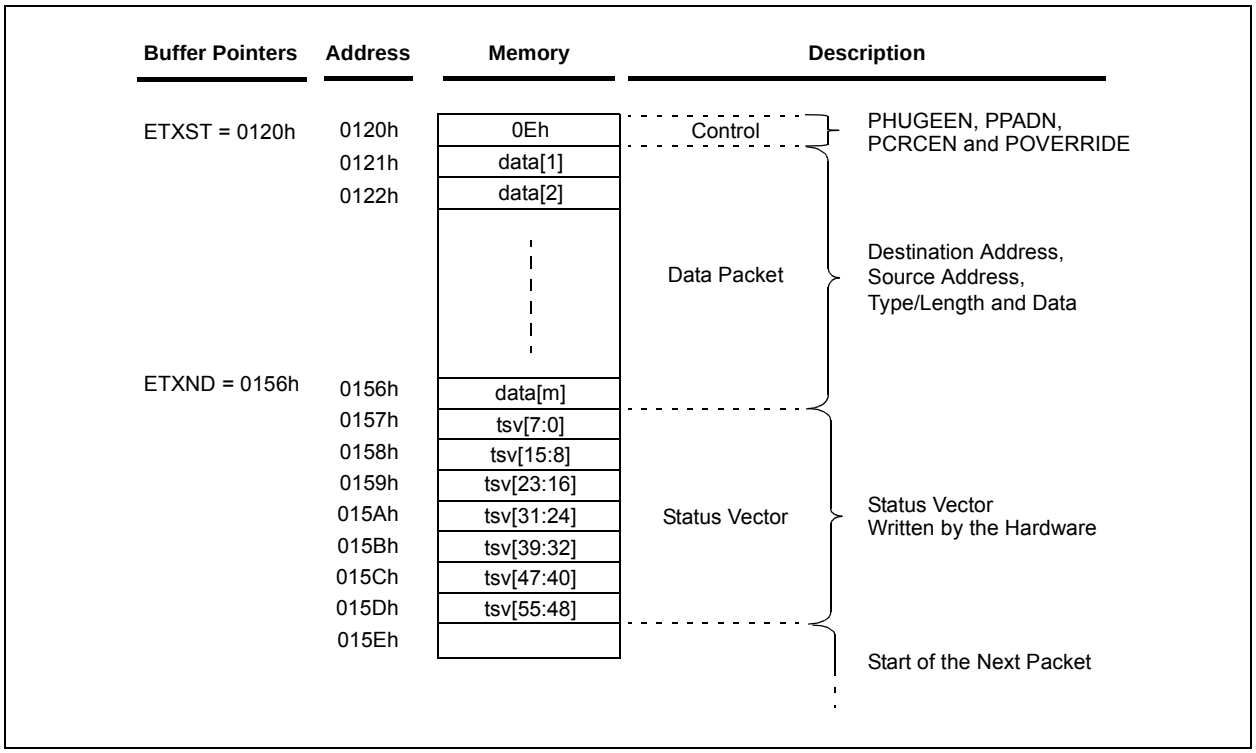


TABLE 19-4: TRANSMIT STATUS VECTORS

Bit	Field	Description
55-52	Zero	0
51	Transmit VLAN Tagged Frame	Frame's length/type field contained 8100h which is the VLAN protocol identifier.
50	Backpressure Applied	Reserved, do not use.
49	Transmit Pause Control Frame	The frame transmitted was a control frame with a valid pause opcode.
48	Transmit Control Frame	The frame transmitted was a control frame.
47-32	Total Bytes Transmitted on Wire	Total bytes transmitted on the wire for the current packet, including all bytes from collided attempts.
31	Transmit Underrun	The transmission was aborted due to insufficient buffer memory bandwidth to sustain the 10 Mbit/s transmit rate.
30	Transmit Giant	Byte count for frame was greater than the MAMXFL registers.
29	Transmit Late Collision	Collision occurred after 64 bytes had already been transmitted.
28	Transmit Excessive Collision	Packet was aborted after the number of collisions exceeded 15, the retransmission maximum.
27	Transmit Excessive Defer	Packet was deferred in excess of 24,287 bit times (2.4287 ms), due to a continuously occupied medium.
26	Transmit Packet Defer	Packet was deferred for at least one attempt, but less than an excessive defer.
25	Transmit Broadcast	Packet's destination address was a Broadcast address.
24	Transmit Multicast	Packet's destination address was a Multicast address.
23	Transmit Done	Transmission of the packet was completed successfully.
22	Transmit Length Out of Range	Indicates that frame type/length field was larger than 1500 bytes (type field).
21	Transmit Length Check Error	Indicates that the frame length field value in the packet does not match the actual data byte length and is not a type field. The FRMLNEN bit (MACON3<1>) must be set to get this error.
20	Transmit CRC Error	The attached CRC in the packet did not match the internally generated CRC.
19-16	Transmit Collision Count	Number of collisions the current packet incurred during transmission attempts. It applies to successfully transmitted packets, and as such, will not show the possible maximum count of 16 collisions.
15-0	Transmit Byte Count	Total bytes in frame, not counting collided bytes.

19.5.3 RECEIVING PACKETS

Assuming that the receive buffer has been initialized, the MAC has been properly configured and the receive filters have been configured, the application should perform these steps to receive Ethernet packets:

1. Set the PKTIE and ETHIE bits to generate an Ethernet interrupt whenever a packet is received (if desired).
2. Clear the RXERIF flag and set both RXERIE and ETHIE to generate an interrupt whenever a packet is dropped, due to insufficient buffer space or memory access bandwidth (if desired).
3. Enable reception by setting the RXEN bit (ECON1<2>).

After setting RXEN, the Duplex mode and the Receive Buffer Start and End Pointers should not be modified. Additionally, to prevent unexpected packets from arriving, it is recommended that RXEN be cleared before altering the receive filter configuration (ERXFCON) and MAC address.

After reception is enabled, packets which are not filtered out will be written into the circular receive buffer. Any packet which does not meet the necessary filter criteria will be discarded and the application will not have any means of identifying that a packet was thrown away. When a packet is accepted and completely written into the buffer:

- The EPKTCNT register is incremented
- The PKTIF bit is set
- An interrupt is generated (if enabled)
- The Hardware Write Pointers, ERXWRPT, are automatically advanced

19.5.3.1 Receive Packet Layout

Figure 19-11 shows the layout of a received packet. The packets are preceded by a 6-byte header which contains a Next Packet Pointer in addition to a receive status vector which contains receive statistics, including the packet's size. The receive status vectors are shown in Table 19-5.

If the last byte in the packet ends on an odd value address, the hardware will automatically add a padding byte when advancing the Hardware Write Pointer. As such, all packets will start on an even boundary.

FIGURE 19-11: SAMPLE RECEIVE PACKET LAYOUT

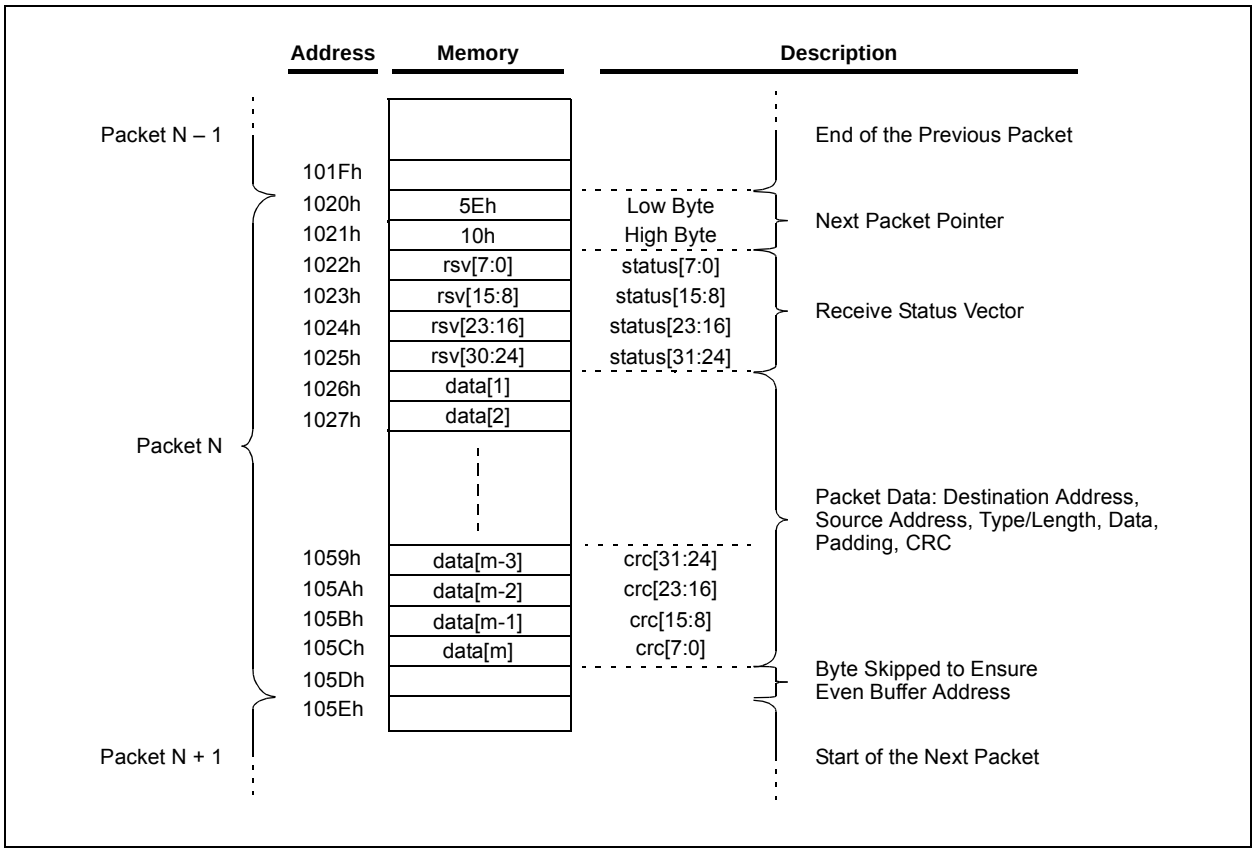


TABLE 19-5: RECEIVE STATUS VECTORS

Bit	Field	Description
31	Zero	'0'
30	Receive VLAN Type Detected	Current frame was recognized as a VLAN tagged frame.
29	Receive Unknown Opcode	Current frame was recognized as a control frame, but it contained an unknown opcode.
28	Receive Pause Control Frame	Current frame was recognized as a control frame containing a valid pause frame opcode and a valid destination address.
27	Receive Control Frame	Current frame was recognized as a control frame for having a valid type/length designating it as a control frame.
26	Dribble Nibble	Indicates that after the end of this packet, an additional 1 to 7 bits were received. The extra bits were thrown away.
25	Receive Broadcast Packet	Indicates packet received had a valid Broadcast address.
24	Receive Multicast Packet	Indicates packet received had a valid Multicast address.
23	Received OK	Indicates that the packet had a valid CRC and no symbol errors.
22	Length Out of Range	Indicates that frame type/length field was larger than 1500 bytes (type field).
21	Length Check Error	Indicates that frame length field value in the packet does not match the actual data byte length.
20	CRC Error	Indicates that the frame CRC field value does not match the CRC calculated by the MAC.
19	Reserved	
18	Carrier Event Previously Seen	Indicates that at some time since the last receive, a carrier event was detected. The carrier event is not associated with this packet. A carrier event is activity on the receive channel that does not result in a packet receive attempt being made.
17	Reserved	
16	Long Event/Drop Event	Indicates a packet over 50,000 bit times occurred or that a packet was dropped since the last receive.
15-0	Received Byte Count	Indicates length of the received frame. This includes the destination address, source address, type/length, data, padding and CRC fields. This field is stored in little-endian format.

19.5.3.2 Reading Received Packets

To process the packet, an application will normally start reading from the beginning of the Next Packet Pointer. The application will save the Next Packet Pointer, any necessary bytes from the receive status vector, and then proceed to read the actual packet contents. If the AUTOINC bit is set, it will be able to sequentially read the entire packet without ever modifying the ERDPT registers. The Read Pointer would automatically wrap at the end of the circular receive buffer to the beginning.

In the event that the application needed to randomly access the packet, it would be necessary to manually calculate the proper ERDPT registers, taking care to not exceed the end of the receive buffer if the packet spans the ERXND to ERXST buffer boundary. In other words, given the packet start address and a desired offset, the application should follow the logic shown in Equation 19-1.

EQUATION 19-1: RANDOM ACCESS ADDRESS CALCULATION

```
If Packet Start Address + Offset > ERXND, then
        ERDPT = Packet Start Address + Offset – (ERXND – ERXST + 1)
else:
        ERDPT = Packet Start Address + Offset
```

19.5.3.3 Freeing Receive Buffer Space

After the user application has processed a packet (or part of the packet) and needs to free the occupied buffer space used by the processed data, it must advance the Receive Buffer Read Pointer pair, ERXRDPT. The module always writes up to, but not over, the memory pointed to by the ERXRDPT registers. If an attempt to overwrite the Receive Buffer Read Pointer location occurs, the packet in progress is aborted, the RXERIF flag is set and an interrupt is generated (if enabled). In this manner, the hardware will never overwrite unprocessed packets. Normally, the ERXRDPT pair is advanced close to a value pointed to by the Next Packet Pointer, which precedes the receive status vector for the current packet.

The Receive Buffer Read Pointer Low Byte (ERXRDPTL register) is internally buffered to prevent the pointer from moving when only one byte is updated. To move the ERXRDPT pair, the application must write to ERXRDPTL first. The write will update the internal buffer but will not affect the register. When the application writes to ERXRDPTH, the internally buffered low byte will be loaded into the ERXRDPTL register at the same time. The ERXRDPT bytes can be read in any order. When they are read, the actual value of the registers will be returned. As a result, the buffered low byte is not readable.

In addition to advancing the Receive Buffer Read Pointer, after each packet is fully processed, the application must set the PKTDEC bit (ECON2<6>). This causes the EPKTCNT register to decrement by 1. After decrementing, if EPKTCNT is '0', the PKTIF flag bit is automatically cleared. Otherwise, it remains set, indicating that additional packets are in the receive buffer and are waiting to be processed. Attempting to decrement EPKTCNT below 0 does not cause an underflow to 255, but may cause an unintentional interrupt. The application should avoid decrementing EPKTCNT in this situation.

Additionally, if the EPKTCNT register ever maximizes at 255, all new packets which are received will be aborted, even if buffer space is available. To indicate the error, the RXERIF is set and an interrupt is generated (if enabled). To prevent this condition, the user application must properly decrement the counter whenever a packet is processed.

Because only one pointer is available to control buffer area ownership, the application must process packets in the order they are received. If a packet is to be saved and processed later, the application should copy the packet to an unused location in memory. This can be done efficiently using the integrated DMA controller (see **Section 19.9 "Direct Memory Access Controller"**).

19.5.3.4 Receive Buffer Free Space

At any time the application needs to know how much receive buffer space is remaining, it should read the Hardware Write Pointers (ERXWRPT registers) and compare it with the ERXRDPT registers. Combined with the known size of the receive buffer, the free space can be derived.

Note:	The ERXWRPT registers only update when a packet has been successfully received. If the application reads it just before another packet is to be successfully completed, the value returned could be stale and off by the maximum frame length permitted (MAMXFLH:MAMXFLL) plus 8. Furthermore, as the application reads one byte of the ERXWRPT registers, a new packet may arrive and update the 13-bit pointer before the application has an opportunity to read the other byte of the ERXWRPT registers.

When reading the ERXWRPT registers with the receive hardware enabled, special care must be taken to ensure the low and high bytes are read as a matching set.

To be assured that a matching set is obtained:

1. Read the EPKTCNT register and save its contents.
2. Read ERXWRPTL and ERXWRPTH.
3. Read the EPKTCNT register again.
4. Compare the two packet counts. If they are not the same, go back to Step 2.

With the Hardware Write Pointers obtained, the free space can be calculated as shown in Equation 19-2. The hardware prohibits moving the Write Pointer to the same value occupied by the ERXRDPT registers, so at least one byte will always go unused in the buffer. The Equation 19-2 calculation reflects the lost byte.

EQUATION 19-2: RECEIVE BUFFER FREE SPACE CALCULATION

```
If ERXWRPT > ERXRDPT, then
        Free Space = (ERXND – ERXST) – (ERXWRPT – ERXRDPT)
else:
if ERXWRPT = ERXRDPT, then
        Free Space = (ERXND – ERXST)
else:
        Free Space = ERXRDPT – ERXWRPT – 1
```

TABLE 19-6: SUMMARY OF REGISTERS ASSOCIATED WITH PACKET TRANSMISSION

Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	73
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	73
ESTAT	—	BUFER	—	r	—	RXBUSY	TXABRT	PHYRDY	73
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	70
ETXSTL	Transmit Start Register Low Byte (ETXST<7:0>)								74
ETXSTH	—	—	—	Transmit Start Register High Byte (ETXST<12:8>)					74
ETXNDL	Transmit End Register Low Byte (ETXND<7:0>)								74
ETXNDH	—	—	—	Transmit End Register High Byte (ETXND<12:8>)					74
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	75
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	75
MACON4	—	DEFER	r	r	—	—	r	r	75
MABBIPG	—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0	75
MAIPGL	—	MAC Non Back-to-Back Inter-Packet Gap Register Low Byte (MAIPGL<6:0>)							75
MAIPGH	—	MAC Non Back-to-Back Inter-Packet Gap Register High Byte (MAIPGH<6:0>)							75
MAMXFLL	Maximum Frame Length Register Low Byte (MAMXFL<7:0>)								74
MAMXFLH	Maximum Frame Length Register High Byte (MAMXFL<15:8>)								74

Legend: — = unimplemented, r = reserved bit. Shaded cells are not used.

TABLE 19-7: SUMMARY OF REGISTERS ASSOCIATED WITH PACKET RECEPTION

Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	73
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	73
ESTAT	—	BUFER	—	r	—	RXBUSY	TXABRT	PHYRDY	73
ECON2	AUTOINC	PKTDEC	ETHEN	—	—	—	—	—	73
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	70
ERXSTL	Receive Start Register Low Byte (ERXST<7:0>)								74
ERXSTH	—	—	—	Receive Start Register High Byte (ERXST<12:8>)					74
ERXNDL	Receive End Register Low Byte (ERXND<7:0>)								74
ERXNDH	—	—	—	Receive End Register High Byte (ERXND<12:8>)					74
ERXRDPTL	Receive Buffer Read Pointer Low Byte (ERXRDPT<7:0>)								73
ERXRDPTH	—	—	—	Receive Buffer Read Pointer High Byte (ERXRDPT<12:8>)					73
ERXFCON	UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN	74
EPKTCNT	Ethernet Packet Count Register								74
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	75
MACON3	PADCFG2	PADCFG1	PADCFG0	TXCRCEN	PHDREN	HFRMEN	FRMLNEN	FULDPX	75
MAMXFLL	Maximum Frame Length Register Low Byte (MAMXFL<7:0>)								74
MAMXFLH	Maximum Frame Length Register High Byte (MAMXFL<15:8>)								74

Legend: — = unimplemented, r = reserved bit. Shaded cells are not used.

19.6 Duplex Mode Configuration and Negotiation

The Ethernet module does not support Automatic Duplex mode negotiation. If it is connected to an automatic duplex negotiation-enabled network switch or Ethernet controller, the module will be detected as a half-duplex device. To communicate in full duplex, the module and the remote node (switch, router or Ethernet controller) must be manually configured for full-duplex operation.

19.6.1 HALF-DUPLEX OPERATION

The Ethernet module operates in Half-Duplex mode when the FULDPX (MACON3<0>) and PDPXMD (PHCON1<8>) bits are cleared (= 0). If only one of these two bits is set, the module will be in an indeterminate state and not function correctly. Since switching between Full and Half-Duplex modes may result in this indeterminate state, it is recommended that the application not transmit any packets (maintain the TXRTS bit clear), and disable packet reception (maintain the RXEN bit clear) during this period.

In Half-Duplex mode, only one Ethernet controller may be transmitting on the physical medium at any time. If the application requests a packet to be transmitted by setting the TXRTS bit while another Ethernet controller is already transmitting, the Ethernet module will delay, waiting for the remote transmitter to stop. When it stops, the module will attempt to transmit its packet. Should another Ethernet controller start transmitting at approximately the same time, the data on the wire will become corrupt and a collision will occur.

The hardware will handle this condition in one of two ways. If the collision occurs before 64 bytes have been transmitted, the following events occur:

1. The TXRTS bit remains set
2. The transmit error interrupt does not occur
3. A random exponential backoff delay elapses, as defined by the IEEE 802.3 specification
4. A new attempt to transmit the packet from the beginning occurs. The application does not need to intervene.

If the number of retransmission attempts reaches 15 and another collision occurs, the packet is aborted and the TXRTS bit is cleared. The application will then be responsible for taking appropriate action. The application will be able to determine that the packet was aborted instead of being successfully transmitted by reading the TXABRT flag. For more information, see **Section 19.5.2 "Transmitting Packets"**.

If the collision occurs after 64 bytes have already been transmitted, the packet is immediately aborted without any retransmission attempts. Ordinarily, in IEEE 802.3 compliant networks which are properly configured, this late collision will not occur. User intervention may be required to correct the issue. This problem may occur as a result of a full-duplex node attempting to transmit on the half-duplex medium. Alternately, the module may be attempting to operate in Half-Duplex mode while it may be connected to a full-duplex network. Excessively long cabling and network size may also be a possible cause of late collisions.

19.6.2 FULL-DUPLEX OPERATION

The Ethernet module operates in Full-Duplex mode when the FULDPX (MACON3<0>) and PDPXMD (PHCON1<8>) bits are both set (= 1). If only one of these two bits is clear, the module will be in an indeterminate state and not function correctly. Again, since switching between Full and Half-Duplex modes may result in this indeterminate state, it is recommended that the application not transmit any packets and should disable packet reception during this period.

In Full-Duplex mode, packets will be transmitted while simultaneously packets may be received. Given this, it is impossible to cause any collisions when transmitting packets.

19.7 Flow Control

The Ethernet module implements hardware flow control for both Full and Half-Duplex modes. The operation of this feature differs depending on which mode is being used.

19.7.1 HALF-DUPLEX MODE

In Half-Duplex mode, setting the FCEN0 bit (EFLOCON<0>) causes flow control to be enabled. When FCEN0 is set, a continuous preamble pattern of alternating '1's and '0's (55h) will automatically be transmitted on the Ethernet medium. Any connected nodes will see the transmission and either not transmit anything, waiting for the transmission to end, or will attempt to transmit and immediately cause a collision. Because a collision will always occur, no nodes on the network will be able to communicate with each other and no new packets will arrive.

When the application causes the module to transmit a packet by setting the TXRTS bit, the preamble pattern will stop being transmitted. An inter-packet delay will pass as configured by register, MABBIPG, and then the module will attempt to transmit its packet. After the inter-packet delay, other nodes may begin to transmit. Because all traffic was jammed previously, several nodes may begin transmitting and a series of collisions may occur. When the module successfully finishes transmitting its packet or aborts it, the transmission of the preamble pattern will automatically restart. When the application wishes to no longer jam the network, it should clear the FCEN0 bit. The preamble transmission will cease and normal network operation will resume.

Given the detrimental network effects that are possible and lack of effectiveness, it is not recommended that half-duplex flow control be used unless the application will be in a closed network environment with proper testing.

19.7.2 FULL-DUPLEX MODE

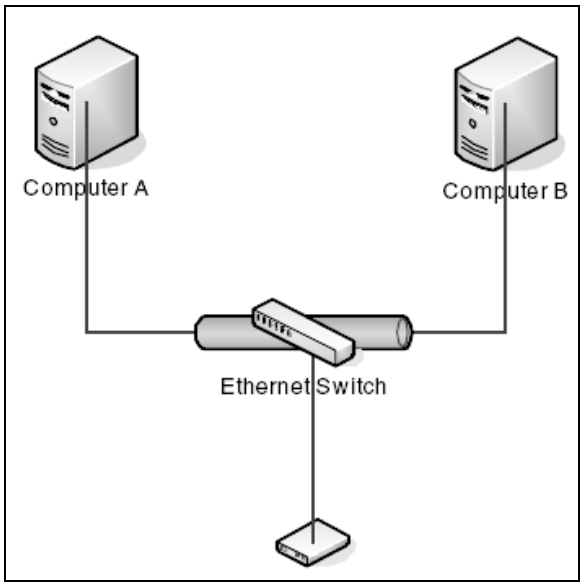
In Full-Duplex mode (MACON3<0> = 1), hardware flow control is implemented by means of transmitting pause control frames, as defined by the IEEE 802.3 specification. Pause control frames are 64-byte frames consisting of the reserved Multicast destination address of 01-80-C2-00-00-01, the source address of the sender, a special pause opcode, a 2-byte pause timer value and padding/CRC.

Normally, when a pause control frame is received by a MAC, the MAC will finish the packet it is transmitting and then stop transmitting any new frames. The pause timer value will be extracted from the control frame and used to initialize an internal timer. The timer will automatically decrement every 512 bit times, or 51.2 μs. While the timer is counting down, reception of packets is still enabled. If new pause frames arrive, the timer will be re-initialized with the new pause timer value. When the timer reaches zero, or was sent a frame with a zero pause timer value, the MAC that received the pause frame will resume transmitting any pending packets. To prevent a pause frame from stopping all traffic on the entire network, Ethernet switches and routers do not propagate pause control frames in Full-Duplex mode. The pause operation only applies to the direct recipient.

A sample network is shown in Figure 19-12. If Computer A were to be transmitting too much data to the microcontroller-based application in Full-Duplex mode, the Ethernet module could transmit a pause control frame to stop the data which is being sent to it. The Ethernet switch would take the pause frame and stop sending data to the application. If Computer A continues to send data, the Ethernet switch will buffer the data so it can be transmitted later when its pause timer expires. If the Ethernet switch begins to run out of buffer space, it will likely transmit a pause control frame of its own to Computer A.

If, for some reason the Ethernet switch does not generate a pause control frame of its own, or one of the nodes does not properly handle the pause frame it receives, then packets will inevitably be dropped. In any event, any communication between Computer A and Computer B will always be completely unaffected.

FIGURE 19-12: SAMPLE FULL-DUPLEX NETWORK



To enable flow control in Full-Duplex mode, set the TXPAUS and RXPAUS bits in the MACON1 register. Then, at any time that the receiver buffer is running out of space, set the Flow Control Enable bits, FCEN<1:0> (EFLOCON<1:0>). The module will automatically finish transmitting anything that was in progress and then send a valid pause frame, loaded with the selected pause timer value. Depending on the mode selected, the application may need to eventually clear Flow Control mode by again writing to the FCEN bits.

When the RXPAUS bit is set and a valid pause frame arrives with a non-zero pause timer value, the module will automatically inhibit transmissions. If the TXRTS bit becomes set to send a packet, the hardware will simply wait until the pause timer expires before attempting to send the packet and subsequently, clearing the TXRTS bit. Normally, this is transparent to the microcontroller, and it will never know that a pause frame had been received. Should it be desirable to know when the MAC is paused or not, the user should set the PASSALL bit (MACON1<1>), then manually interpret the pause control frames which may arrive.

REGISTER 19-19: EFLOCON: ETHERNET FLOW CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	R-0	R/W-0	R/W-0
—	—	—	—	—	r	FCEN1	FCEN0
bit 7							bit 0

Legend:	r = Reserved 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 7-3 **Unimplemented:** Read as ‘0’

bit 2 **Reserved:** Do not use

bit 1-0 **FCEN<1:0>:** Flow Control Enable bits

When FULDPX (MACON3<0>) = 1:

11 = Send one pause frame with a ‘0’ timer value and then turn flow control off

10 = Send pause frames periodically

01 = Send one pause frame then turn flow control off

00 = Flow control off

When FULDPX (MACON3<0>) = 0:

x1 = Flow control on

x0 = Flow control off

TABLE 19-8: SUMMARY OF REGISTERS USED WITH FLOW CONTROL

Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	70
MACON1	—	—	—	r	TXPAUS	RXPAUS	PASSALL	MARXEN	75
MABBIPG	—	BBIPG6	BBIPG5	BBIPG4	BBIPG3	BBIPG2	BBIPG1	BBIPG0	75
EFLOCON	—	—	—	—	—	r	FCEN1	FCEN0	75
EPAUSL	Pause Timer Value Register Low Byte (EPAUS<7:0>)								75
EPAUSH	Pause Timer Value Register High Byte (EPAUS<15:8>)								75

Legend: — = unimplemented, r = reserved bit. Shaded cells are not used.

19.8 Receive Filters

To minimize microcontroller processing overhead, the Ethernet module incorporates a range of different receive filters which can automatically reject packets which are not needed. Six different types of packet filters are implemented:

- Unicast
- Multicast
- Broadcast
- Pattern Match
- Magic Packet™
- Hash Table

The individual filters are all configured by the ERXFCON register (Register 19-20). More than one filter can be active at any given time. Additionally, the filters can be configured by the ANDOR bit to either logically AND or logically OR the tests of several filters. In other words, the filters may be set so that only packets accepted by all active filters are accepted, or a packet accepted by any one filter is accepted. The flowcharts in Figure 19-13 and Figure 19-14 show the effect that each of the filters will have, depending on the setting of ANDOR.

The device can enter Promiscuous mode and receive all legal packets by setting the ERXFCON register to 20h (enabling only the CRC filter for valid packets). The proper setting of the register will depend on the application requirements.

19.8.1 UNICAST FILTER

The Unicast receive filter checks the destination address of all incoming packets. If the destination address exactly matches the contents of the MAADR registers, the packet meets the Unicast filter criteria.

19.8.2 MULTICAST FILTER

The Multicast receive filter checks the destination address of all incoming packets. If the Least Significant bit of the first byte of the destination address is set, the packet meets the Multicast filter criteria.

19.8.3 BROADCAST FILTER

The Broadcast receive filter checks the destination address of all incoming packets. If the destination address is FF-FF-FF-FF-FF-FF, the packet meets the Broadcast filter criteria.

19.8.4 HASH TABLE FILTER

The Hash Table receive filter is typically used to receive traffic sent to a specific Multicast group address. Because it checks the specific destination address of packets, it is capable of filtering out more unwanted packets than the Multicast filter.

The filter performs a 32-bit CRC over the six destination address bytes in the packet, using the polynomial, 4C11DB7h. From the resulting 32-bit binary number, a 6-bit value is derived from bits<28:23>. This value, in turn, points to a location in a table formed by the Ethernet Hash Table registers, ETH0 through ETH7. If the bit in that location is set, the packet meets the Hash Table filter criteria and is accepted. The specific pointer values for each bit location in the table are shown in Table 19-9.

An example of the Hash Table operation is shown in Example 19-1. In this case, the destination address, 01-00-00-00-01-2C, produces a Table Pointer value of 34h, which points to bit 4 of ETH6. If this bit is '1', the packet will be accepted.

By extension, clearing every bit in the Hash Table registers means that the filter criteria will never be met. Similarly, if every bit in the Hash Table is set, the filter criteria will always be met.

TABLE 19-9: BIT ASSIGNMENTS IN HASH TABLE REGISTERS

Register	Bit Number in Hash Table							
	7	6	5	4	3	2	1	0
EHT0	07	06	05	04	03	02	01	00
EHT1	0F	0E	0D	0C	0B	0A	09	08
EHT2	17	16	15	14	13	12	11	10
EHT3	1F	1E	1D	1C	1B	1A	19	18
EHT4	27	26	25	24	23	22	21	20
EHT5	2F	2E	2D	2C	2B	2A	29	28
EHT6	37	36	35	34	33	32	31	30
EHT7	3F	3E	3D	3C	3B	3A	39	38

EXAMPLE 19-1: DERIVING A HASH TABLE LOCATION

Packet Destination Address:

01-00-00-00-01-2C (hex)

Result of CRC-32 with 4C11DB7h:

1101 1010 0000 1011 0100 0101 0111 0101 (binary)

Pointer Derived from bits<28:23> of CRC Result:

110100 (binary) or 34 (hex)

Corresponding Hash Table Location:

ETH6<4>

REGISTER 19-20: ERXFCON: ETHERNET RECEIVE FILTER CONTROL REGISTER

R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
UCEN	ANDOR	CRCEN	PMEN	MPEN	HTEN	MCEN	BCEN
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 **UCEN:** Unicast Filter Enable bit
When ANDOR = 1:
1 = Packets not having a destination address matching the local MAC address will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Packets with a destination address matching the local MAC address will be accepted
0 = Filter is disabled

bit 6 **ANDOR:** AND/OR Filter Select bit
1 = AND: Packets will be rejected unless all enabled filters accept the packet
0 = OR: Packets will be accepted unless all enabled filters reject the packet

bit 5 **CRCEN:** Post-Filter CRC Check Enable bit
1 = All packets with an invalid CRC will be discarded
0 = The CRC validity will be ignored

bit 4 **PMEN:** Pattern Match Filter Enable bit
When ANDOR = 1:
1 = Packets must meet the Pattern Match criteria or they will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Packets which meet the Pattern Match criteria will be accepted
0 = Filter is disabled

bit 3 **MPEN:** Magic Packet Filter Enable bit
When ANDOR = 1:
1 = Packets must be Magic Packets for the local MAC address or they will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Magic Packets for the local MAC address will be accepted
0 = Filter is disabled

bit 2 **HTEN:** Hash Table Filter Enable bit
When ANDOR = 1:
1 = Packets must meet the Hash Table criteria or they will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Packets which meet the Hash Table criteria will be accepted
0 = Filter is disabled

bit 1 **MCEN:** Multicast Filter Enable bit
When ANDOR = 1:
1 = The LSb of the first byte of the packet's destination address must be set or it will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Packets which have the LSb of the first byte in the destination address set will be accepted
0 = Filter is disabled

bit 0 **BCEN:** Broadcast Filter Enable bit
When ANDOR = 1:
1 = Packets must have a destination address of FF-FF-FF-FF-FF-FF or they will be discarded
0 = Filter is disabled
When ANDOR = 0:
1 = Packets which have a destination address of FF-FF-FF-FF-FF-FF will be accepted
0 = Filter is disabled

FIGURE 19-13: RECEIVE FILTERING USING OR LOGIC

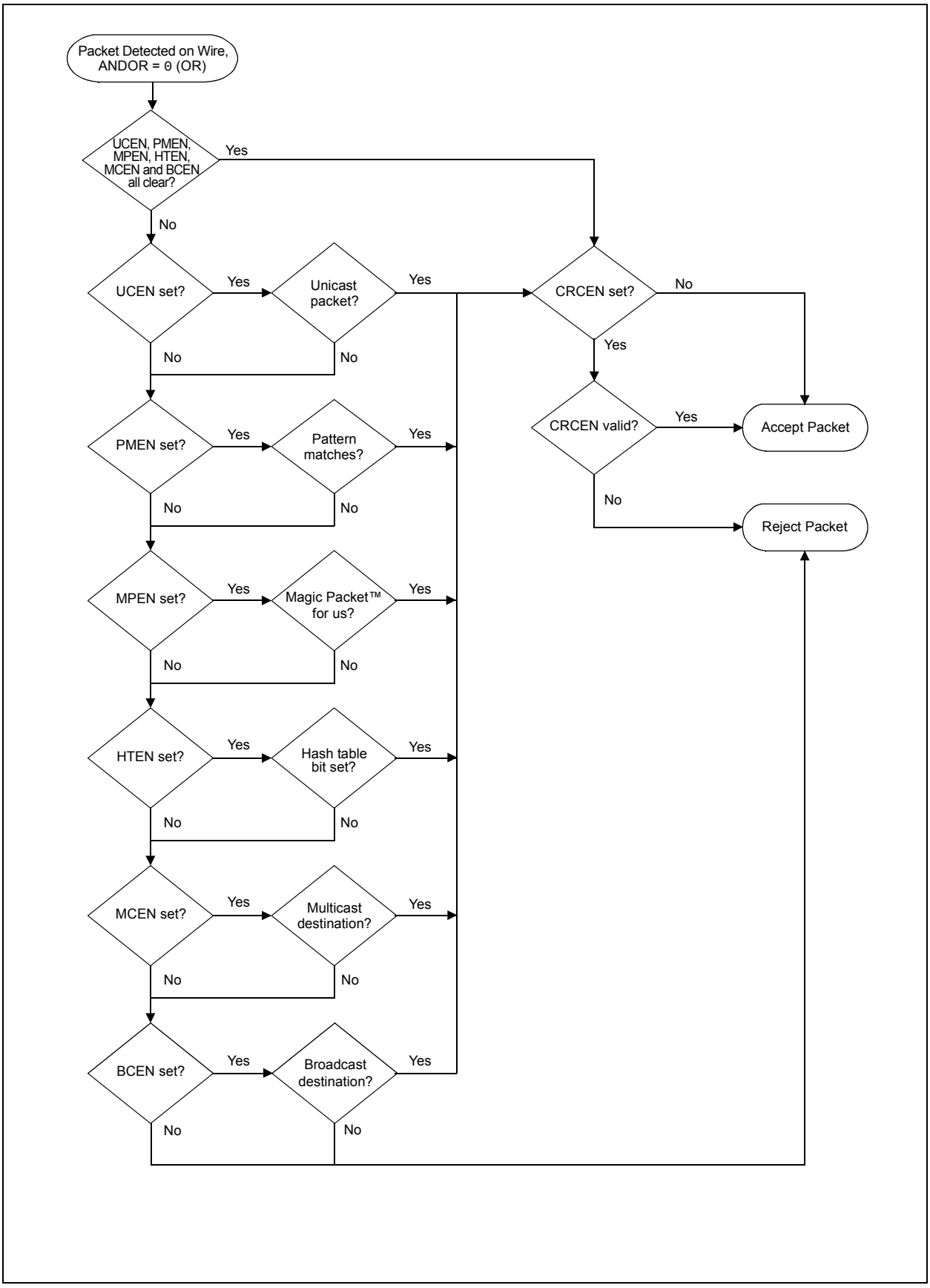
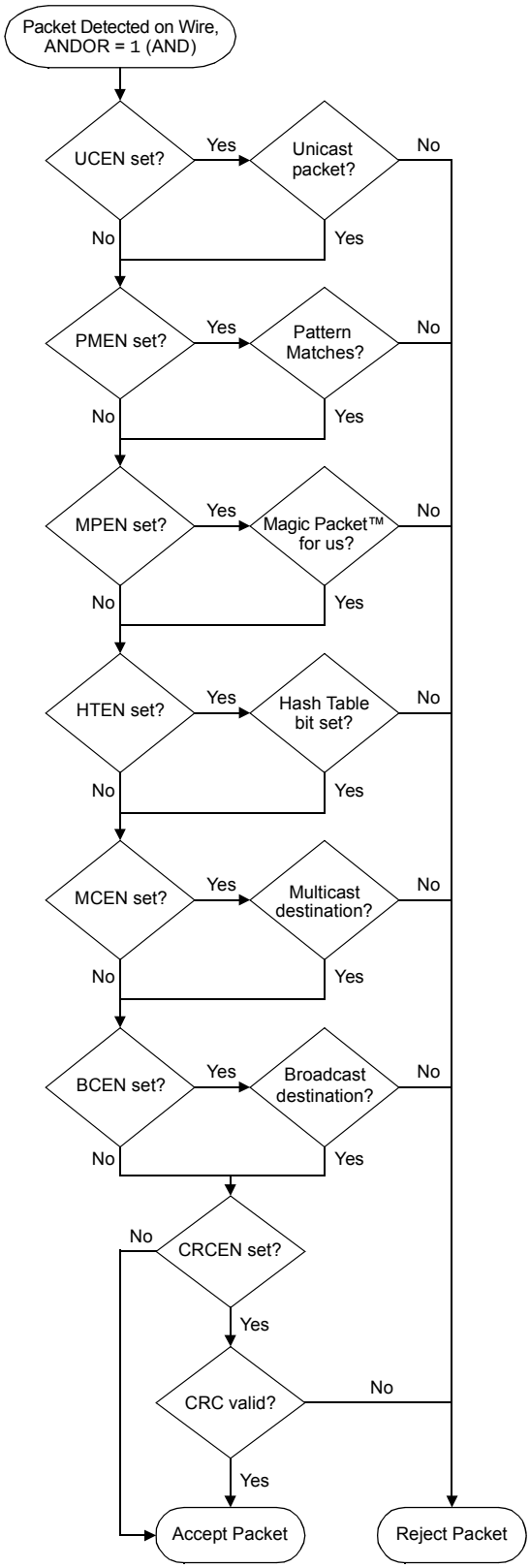


FIGURE 19-14: RECEIVE FILTERING USING AND LOGIC



19.8.5 PATTERN MATCH FILTER

The Pattern Match filter selects up to 64 bytes from the incoming packet and calculates an IP checksum of the bytes. The checksum is then compared to the EPMCS registers. The packet meets the Pattern Match filter criteria if the calculated checksum matches the EPMCS registers. The Pattern Match filter may be useful for filtering packets which have expected data inside them.

To use the Pattern Match filter, the application must program the Pattern Match offset (EPMOH:EPMOL), all of the Pattern Match mask bytes (EPMM0:EPMM7) and the Pattern Match Checksum register pair (EPMCSH:EPMCSL). The Pattern Match offset should be loaded with the offset from the beginning of the destination address field to the 64-byte window which will be used for the checksum computation. Within the 64-byte window, each individual byte can be selectively included or excluded from the checksum computation by setting or clearing the respective bit in the Pattern Match mask. If a packet is received which would cause the 64-byte window to extend past the end of the CRC, the filter criteria will immediately not be met, even if the corresponding mask bits are all '0'.

Note:	In all cases, the value of the Pattern Match offset must be even for proper operation. Programming the EMPO register pair with an odd value will cause unpredictable results.

The Pattern Match Checksum registers should be programmed to the checksum which is expected for the selected bytes. The checksum is calculated in the same manner that the DMA module calculates checksums (see **Section 19.9.2 "Checksum Calculations"**). Data bytes which have corresponding mask bits programmed to '0' are completely removed for purposes of calculating the checksum, as opposed to treating the data bytes as zero.

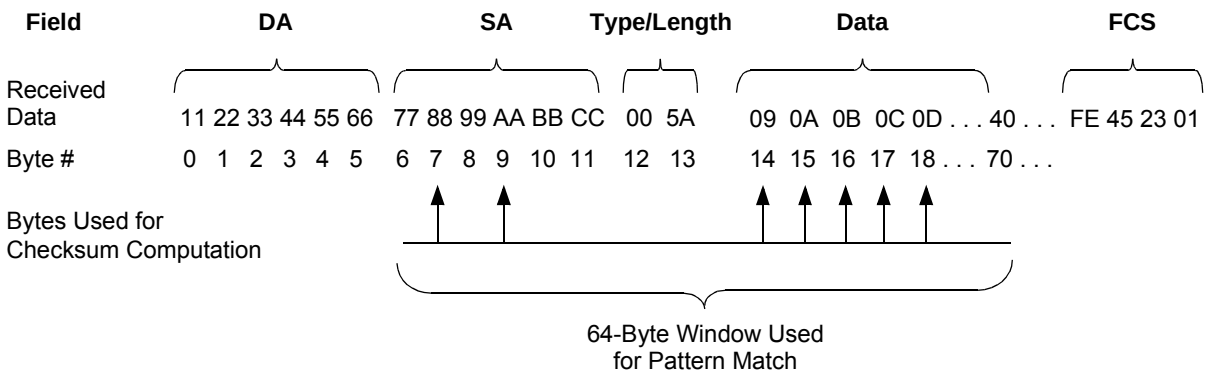
As an example, if the application wished to filter all packets having a particular source MAC address of 00-04-A3-FF-FF-FF, it could program the Pattern Match offset to 0000h and then set bits 6 and 7 of EPMM0 and bits 0, 1, 2 and 3 of EPMM1 (assuming all other mask bits are '0'). The proper checksum to program into the EPMCS registers would be 5BFCh. As an alternative configuration, it could program the offset to 0006h and set bits 0, 1, 2, 3, 4 and 5 of EPMM0. The checksum would still be 5BFCh. However, the second case would be less desirable as packets less than 70 bytes long could never meet the Pattern Match criteria, even if they would generate the proper checksum given the mask configuration.

Another example of a Pattern Match filter is illustrated in Figure 19-15.

FIGURE 19-15: SAMPLE PATTERN MATCH FORMAT

Input Configuration:

EMPOH:EPMOL = 0006h
EPMM7:EPMM0 = 0000000000001F0Ah
EPMCSH:EPMCSL = 563Fh



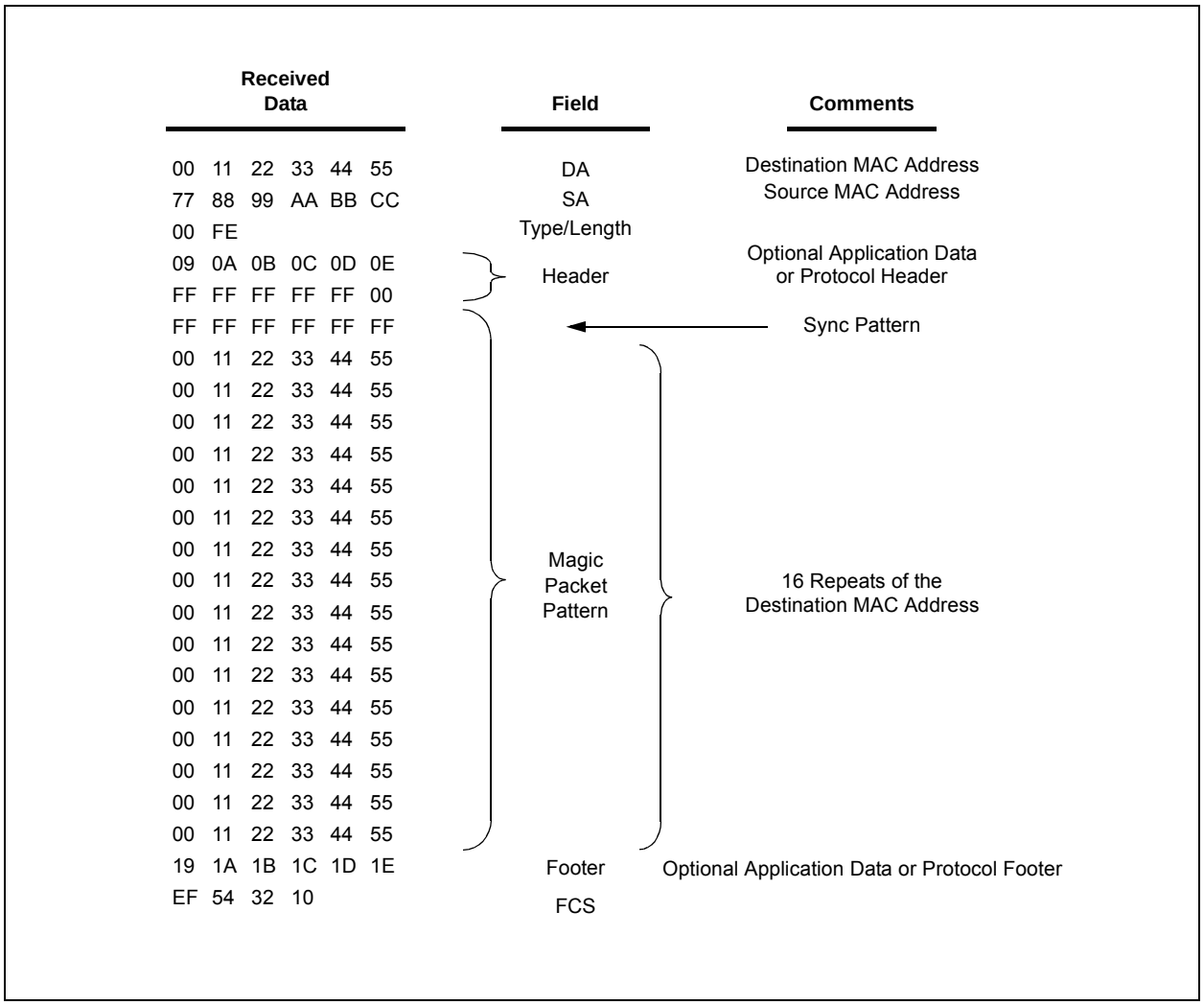
Values Used for Checksum Computation = {88h, AAh, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 00h}
(00h padding byte added by hardware)

Note: Received data is shown in hexadecimal. Byte numbers are shown in decimal format.

19.8.6 MAGIC PACKET FILTER

The Magic Packet pattern consists of a sync pattern of 6 FFh bytes, followed by 16 repeats of the destination address (Figure 19-16). The Magic Packet filter checks the destination address and data fields of all incoming packets. If the destination address matches the MAADR registers and the data field holds a valid Magic Packet pattern someplace within it, then the packet will meet the Magic Packet filter criteria.

FIGURE 19-16: SAMPLE MAGIC PACKET™ FORMAT



19.9 Direct Memory Access Controller

The Ethernet module incorporates a dual purpose DMA controller, which can be used to copy data between locations within the 8-Kbyte memory buffer. It can also be used to calculate a 16-bit checksum which is compatible with various industry standard communication protocols, including TCP, UDP, IP, ICMP, etc.

The DMA is controlled using three pointers and several status/control bits:

- EDMASTH:EDMASTL: Source Start Address
- EDMANDH:EDMANDL: Source End Address
- EDMADSTH:EDMADSTL: Destination Start Address
- DMAST and CSUMEN (ECON1<5,4>): DMA Start/Busy and Checksum Enable bits
- DMAIE and DMAIF (EIE<5> and EIR<5>): DMA Interrupt Enable and Flag bits

The Source and End Pointers define what data will be copied or checksumed. The Destination Pointer, used only when copying data, defines where copied data will be placed. All three pointers are with respect to the 8-Kbyte Ethernet memory and cannot be used to access memory in the PIC® microcontroller data memory space.

When a DMA operation begins, the EDMAST register pair is copied into an Internal Source Pointer. The DMA will execute on one byte at a time and then increment the Internal Source Pointer. However, if a byte is processed and the Internal Source Pointer is equal to the Receive Buffer End Pointer pair, ERXND, the Source Pointer will not be incremented. Instead, the Internal Source Pointer will be loaded with the Receive Buffer Start Pointer pair, ERXST. In this way, the DMA will follow the circular FIFO structure of the receive buffer and received packets can be processed using one operation. The DMA operation will end when the Internal Source Pointer matches the EDMAND Pointers.

While any DMA operation is in progress, the DMA Pointers and the CSUMEN bit (ECON1<4>) should not be modified. The DMA operation can be canceled at any time by clearing the DMAST bit (ECON1<5>). No registers will change; however, some memory bytes may already have been copied if a DMA copy was in progress.

Some operational requirements must always be kept in mind when using the DMA. Failure to observe these requirements may result in a loss of Ethernet buffer data, or even complete failure of Ethernet operation:

- If the EDMAND Pointers cannot be reached because of the receive buffer wrapping behavior, the DMA operation will never end.
- By design, the DMA module cannot be used to copy or calculate a checksum over only one byte (EDMAST = EDMAND). An attempt to do so may overwrite all memory in the buffer and never end.
- After termination of a DMA operation (DMAST is cleared by hardware or firmware), the application must not set DMAST again within 4 instruction cycles.
- To ensure reliable operation, avoid having the application access EDATA during a DMA copy operation. EDATA may be safely accessed during DMA checksum operations.

19.9.1 COPYING MEMORY

To copy memory within the buffer:

1. Program the EDMAST, EDMAND and EDMADST register pairs with the appropriate start, end and destination addresses. The EDMAST registers should point to the first byte to copy from, the EDMAND registers should point to the last byte to copy and the EDMADST registers should point to the first byte in the destination range. The destination range will always be linear, never wrapping at any values except from 8191 to 0 (the 8-Kbyte memory boundary). Extreme care should be taken when calculating the End Pointer to prevent a never ending DMA operation which would overwrite the entire 8-Kbyte buffer.
2. If desired, set the DMAIE (EIE<5>) and ETHIE (PIE2<5>) bits, and clear the DMAIF (EIR<5>) flag bit to enable an interrupt at the end of the copy process.
3. Clear the CSUMEN (ECON1<4>) bit.
4. Start the DMA copy by setting the DMAST (ECON1<5>) bit.

If a transmit operation is in progress (TXRTS bit is set) while the DMAST bit is set, the module will wait until the transmit operation is complete before attempting to do the DMA copy. This possible delay is required because the DMA and transmission engine are unable to access the buffer at the same time.

When the copy is complete, the DMA hardware will clear the DMAST bit, set the DMAIF bit and generate an interrupt (if enabled). The pointers and the EDMACS registers will not be modified.

After the DMA module has been initialized and has begun its copy, one instruction cycle (T_{CY}) will be required for each byte copied. However, if the Ethernet receive hardware accumulates one byte of data, the DMA will stall that cycle, yielding to the higher priority operation. If a maximum size, 1518-byte packet was copied while no other memory bandwidth was being used, the DMA module would require slightly more than 145.7 μs to complete at a core frequency of 41.667 MHz. The time required to copy a minimum size packet of 64 bytes would be approximately 6.2 μs (at 41.667 MHz), plus register configuration time.

19.9.2 CHECKSUM CALCULATIONS

The checksum calculation logic treats the source data as a series of 16-bit big-endian integers. If the source range contains an odd number of bytes, a padding byte of 00h is effectively added to the end of the series for purposes of calculating the checksum.

The calculated checksum is the 16-bit, one's complement of the one's complement sum of all 16-bit integers. For example, if the bytes included in the checksum were {89h, ABh, CDh}, the checksum would begin by computing: 89ABh + CD00h. A carry out of the 16th bit would occur in the example, so in 16-bit one's complement arithmetic, it would be added back to the first bit. The resulting value of 56ACh would finally be complemented to achieve a checksum of A953h.

To calculate a checksum:

1. Set the EDMAST and EDMAND register pairs to point to the first and last bytes of buffer data to be included in the checksum. Care should be taken when programming these pointers to prevent a never-ending checksum calculation due to receive buffer wrapping.
2. To generate an optional interrupt when the checksum calculation is done, set the DMAIE (EIE<5>) and ETHIE (PIE2<5>) bits and clear the DMAIF (EIR<5>) bit.
3. Start the calculation by setting the CSUMEN (ECON1<4>) and DMAST (ECON1<5>) bits.

When the checksum is finished being calculated, the hardware will clear the DMAST bit, set the DMAIF bit and an interrupt will be generated, if enabled. The DMA Pointers will not be modified and no memory will be written to. The EDMACSH and EDMACSL registers will contain the calculated checksum. The application may write this value into a packet, compare this value with zero (to validate a received block of data containing a checksum field in it), or compare it with some other checksum, such as a pseudo header checksum used in various protocols (TCP, UDP, etc.).

When operating the DMA in Checksum mode, it takes one instruction cycle (T_{CY}) for every byte included in the checksum. As a result, if a checksum over 1446 bytes was performed, the DMA module would require slightly more than 138.8 μs to complete the operation at 41.667 MHz.

At the same frequency, a small 20-byte header field would take approximately 1.9 μs plus DMA setup time to calculate a sum. These estimated times assume that the Ethernet receive hardware does not need memory access bandwidth and the CPU does not issue any reads or writes to the EDATA register while the DMA is computing.

Like the DMA Copy mode, the checksum operation will not start until the TXRTS bit (ECON1<3>) is clear. This may considerably increase the checksum calculation time if the application transmits a large packet and immediately attempts to validate a checksum on a received packet.

TABLE 19-10: SUMMARY OF REGISTERS ASSOCIATED WITH THE DMA CONTROLLER

Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
EIE	—	PKTIE	DMAIE	LINKIE	TXIE	—	TXERIE	RXERIE	73
EIR	—	PKTIF	DMAIF	LINKIF	TXIF	—	TXERIF	RXERIF	73
ECON1	TXRST	RXRST	DMAST	CSUMEN	TXRTS	RXEN	—	—	70
ERXNDL	Receive End Register Low Byte (ERXND<7:0>)								73
ERXNDH	—	—	—	Receive End Register High Byte (ERXND<12:8>)					73
EDMASTL	DMA Start Register Low Byte (EDMAST<7:0>)								73
EDMASTH	—	—	—	DMA Start Register High Byte (EDMAST<12:8>)					73
EDMANDL	DMA End Register Low Byte (EDMAND<7:0>)								73
EDMANDH	—	—	—	DMA End Register High Byte (EDMAND<12:8>)					73
EDMADSTL	DMA Destination Register Low Byte (EDMADST<7:0>)								73
EDMADSTH	—	—	—	DMA Destination Register High Byte (EDMADST<12:8>)					73
EDMACSL	DMA Checksum Register Low Byte (EDMACS<7:0>)								73
EDMACSH	DMA Checksum Register High Byte (EDMACS<15:8>)								73

Legend: — = unimplemented. Shaded cells are not used.

19.10 Module Resets

The Ethernet module provides selective module Resets:

- Transmit Only Reset
- Receive Only Reset

19.10.1 MICROCONTROLLER RESETS

Following any standard Reset event, the Ethernet module returns to a known state. The contents of the Ethernet buffer memory are unknown. All SFR and PHY registers are loaded with their specified Reset values, depending on the type of Reset event. However, the PHY registers must not be accessed until the PHY start-up timer has expired and the PHYRDY bit (ESTAT<0>) becomes set, or at least 1 ms has passed since the ETHEN bit was set. For more details, see **Section 19.1.3.1 "Start-up Timer"**.

19.10.2 TRANSMIT ONLY RESET

The Transmit Only Reset is performed by writing a '1' to the TXRST bit (ECON1<7>). This resets the transmit logic only. Other register and control blocks, such as buffer management and host interface, are not affected by a Transmit Only Reset event. To return to normal operation, the TXRST bit must be cleared in software. After clearing TXRST, firmware must not write to any Ethernet module SFRs for at least 1.6 μs. After the delay, normal operation can resume.

19.10.3 RECEIVE ONLY RESET

The Receive Only Reset is performed by writing a '1' to the RXRST bit (ECON1<6>). This action resets receive logic only. Other register and control blocks, such as the buffer management and host interface blocks, are not affected by a Receive Only Reset event. To return to normal operation, the RXRST bit is cleared in software. After clearing RXRST, firmware must not write to any Ethernet module SFRs for at least 1.6 μs. After the delay, normal operation can resume.

NOTES:

20.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

20.1 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

The 64-pin and 80-pin devices of the PIC18F97J60 family have one MSSP module, designated as MSSP1. The 100-pin devices have two MSSP modules, designated as MSSP1 and MSSP2. Each module operates independently of the other.

Note:	Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names and module I/O signals use the generic designator, 'x', to indicate the use of a numeral to distinguish a particular module when required. Control bit names are not individuated.

20.2 Control Registers

Each MSSP module has three associated control registers. These include a status register (SSPxSTAT) and two control registers (SSPxCON1 and SSPxCON2). The use of these registers and their individual configuration bits differ significantly depending on whether the MSSP module is operating in SPI or I^2C mode.

Additional details are provided under the individual sections.

Note:	In devices with more than one MSSP module, it is very important to pay close attention to the SSPxCON register names. SSP1CON1 and SSP1CON2 control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

20.3 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:

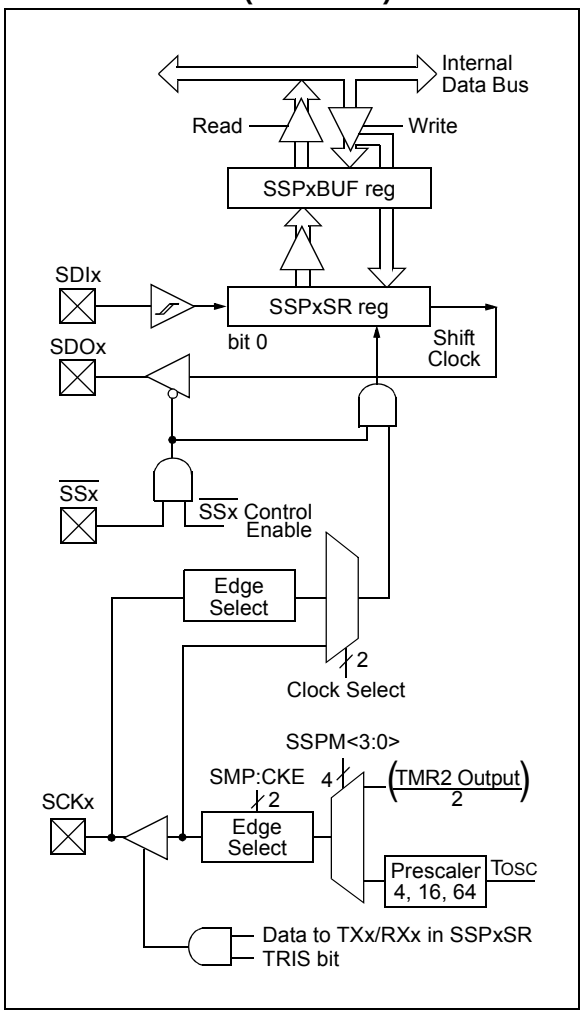
- Serial Data Out (SDOx) – RC5/SDO1 (or RD4/SDO2 for 100-pin devices)
- Serial Data In (SDIx) – RC4/SDI1/SDA1 (or RD5/SDI2/SDA2 for 100-pin devices)
- Serial Clock (SCKx) – RC3/SCK1/SCL1 (or RD6/SCK2/SCL2 for 100-pin devices)

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SSx}$) – RF7/$\overline{SS1}$ (or RD7/$\overline{SS2}$ for 100-pin devices)

Figure 20-1 shows the block diagram of the MSSP module when operating in SPI mode.

FIGURE 20-1: MSSP BLOCK DIAGRAM (SPI MODE)



20.3.1 REGISTERS

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

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

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

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

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

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

REGISTER 20-1: SSPxSTAT: MSSPx STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE(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 sampled at end of data output time
0 = Input data sampled at middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Select bit(1)
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/$\overline{Address}$ bit
Used in I^2C mode only.

bit 4 **P:** Stop bit
Used in I^2C mode only. This bit is cleared when the MSSP module is disabled, SSPEN is cleared.

bit 3 **S:** Start bit
Used in I^2C mode only.

bit 2 **R/$\overline{W}$:** Read/$\overline{Write}$ Information bit
Used in I^2C mode only.

bit 1 **UA:** Update Address bit
Used in I^2C mode only

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
1 = Receive complete, SSPxBUF is full
0 = Receive not complete, SSPxBUF is empty

Note 1: Polarity of clock state is set by the CKP bit (SSPxCON1<4>).

REGISTER 20-2: SSPxCON1: MSSPx CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(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 SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit(1)
SPI Slave mode:
1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. The user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(2)
1 = Enables serial port and configures SCKx, SDOx, SDIx and $\overline{SSx}$ as serial port pins
0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
1 = Idle state for clock is a high level
0 = Idle state for clock is a low level

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(3)
0101 = SPI Slave mode, Clock = SCKx pin, $\overline{SSx}$ pin control disabled, $\overline{SSx}$ can be used as I/O pin
0100 = SPI Slave mode, Clock = SCKx pin, $\overline{SSx}$ pin control enabled
0011 = SPI Master mode, Clock = TMR2 output/2
0010 = SPI Master mode, Clock = FOSC/64
0001 = SPI Master mode, Clock = FOSC/16
0000 = SPI Master mode, Clock = FOSC/4

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
2: When this bit is enabled, these pins must be properly configured as input or output.
3: Bit combinations not specifically listed here are either reserved or implemented in I^2C™ mode only.

20.3.2 OPERATION

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

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

Each MSSP module consists of a transmit/receive shift register (SSPxSR) and a buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full detect bit, BF (SSPxSTAT<0>), and the interrupt flag bit, SSPxIF, are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPxBUF register during transmission/reception of data will be ignored and the Write Collision detect bit, WCOL (SSPxCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPxBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF (SSPxSTAT<0>), indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPxBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 20-1 shows the loading of the SSP1BUF (SSP1SR) for data transmission.

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

EXAMPLE 20-1: LOADING THE SSP1BUF (SSP1SR) REGISTER

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

20.3.3 ENABLING SPI I/O

To enable the serial port, MSSP Enable bit, SSPEN (SSPxCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPxCON registers and then set the SSPEN bit. This configures the SDIx, SDOx, SCKx and $\overline{SS}$x 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:

- SDIx is automatically controlled by the SPI module
- SDOx must have TRISC<5> (or TRISD<4>) bit cleared
- SCKx (Master mode) must have TRISC<3> (or TRISD<6>) bit cleared
- SCKx (Slave mode) must have TRISC<3> (or TRISD<6>) bit set
- $\overline{SS}$x must have TRISF<7> (or TRISD<7>) bit set

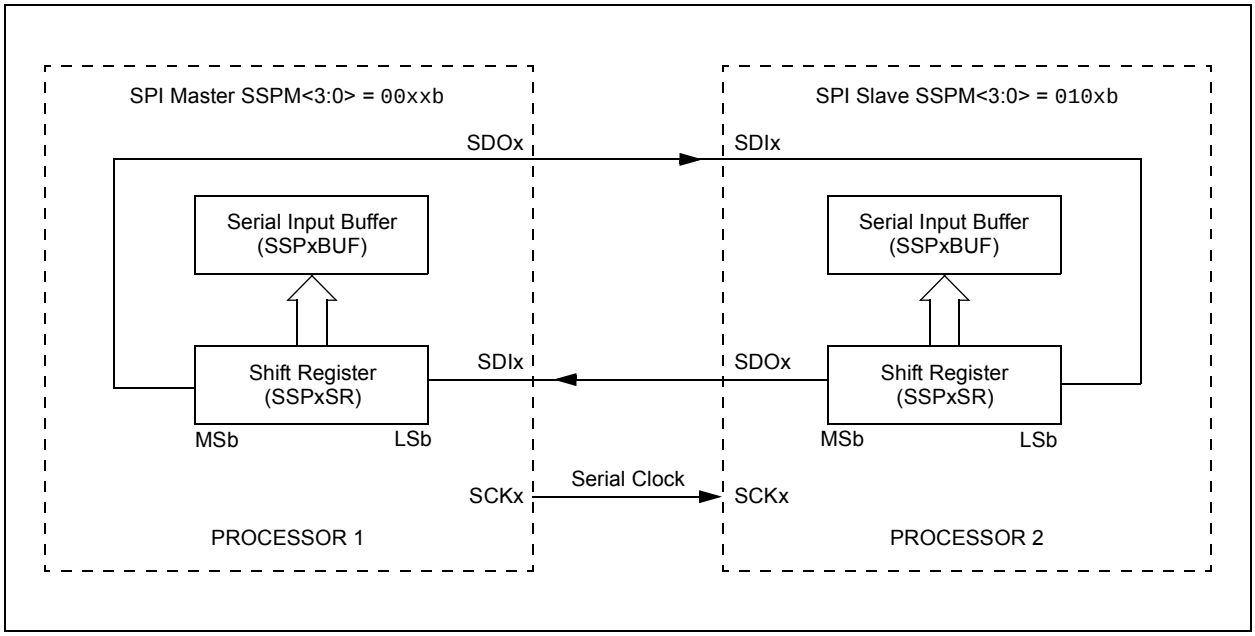
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

20.3.4 TYPICAL CONNECTION

Figure 20-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCKx signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

FIGURE 20-2: SPI MASTER/SLAVE CONNECTION



20.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx. The master determines when the slave (Processor 2, Figure 20-2) will broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDOx output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDIx pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

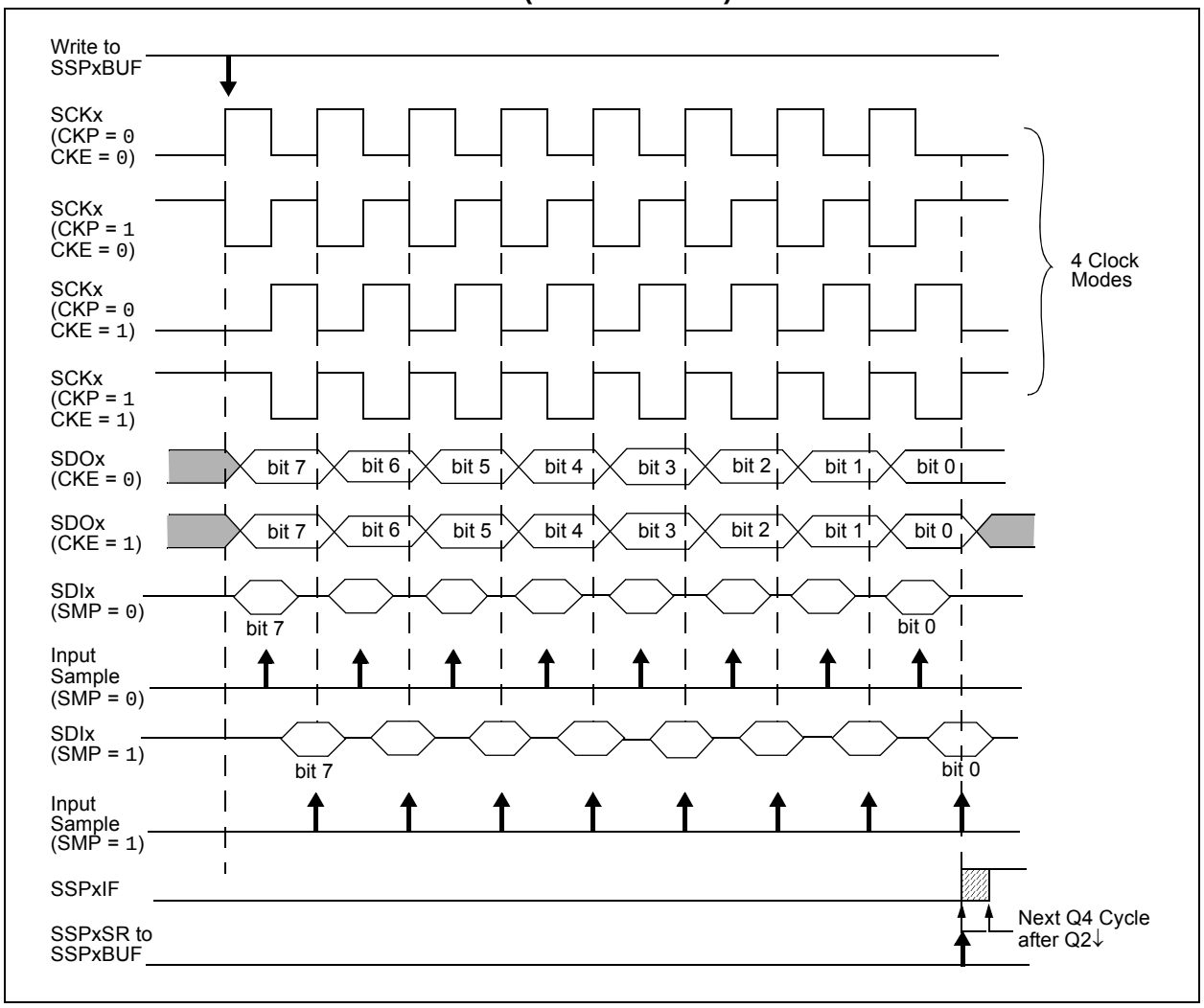
The clock polarity is selected by appropriately programming the CKP bit (SSPxCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 20-3, Figure 20-5 and Figure 20-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to 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 20-3 shows the waveforms for Master mode. When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 20-3: SPI MODE WAVEFORM (MASTER MODE)



20.3.6 SLAVE MODE

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

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

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

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

20.3.7 SLAVE SELECT SYNCHRONIZATION

The $\overline{SSx}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 04h). When the $\overline{SSx}$ pin is low, transmission and reception are enabled and the SDOx pin is driven. When the $\overline{SSx}$ pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte, and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note 1: When the SPI is in Slave mode with $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SSx}$ pin is set to VDD.

2: If the SPI is used in Slave mode with CKE set, then the $\overline{SSx}$ pin control must be enabled.

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

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

FIGURE 20-4: SLAVE SYNCHRONIZATION WAVEFORM

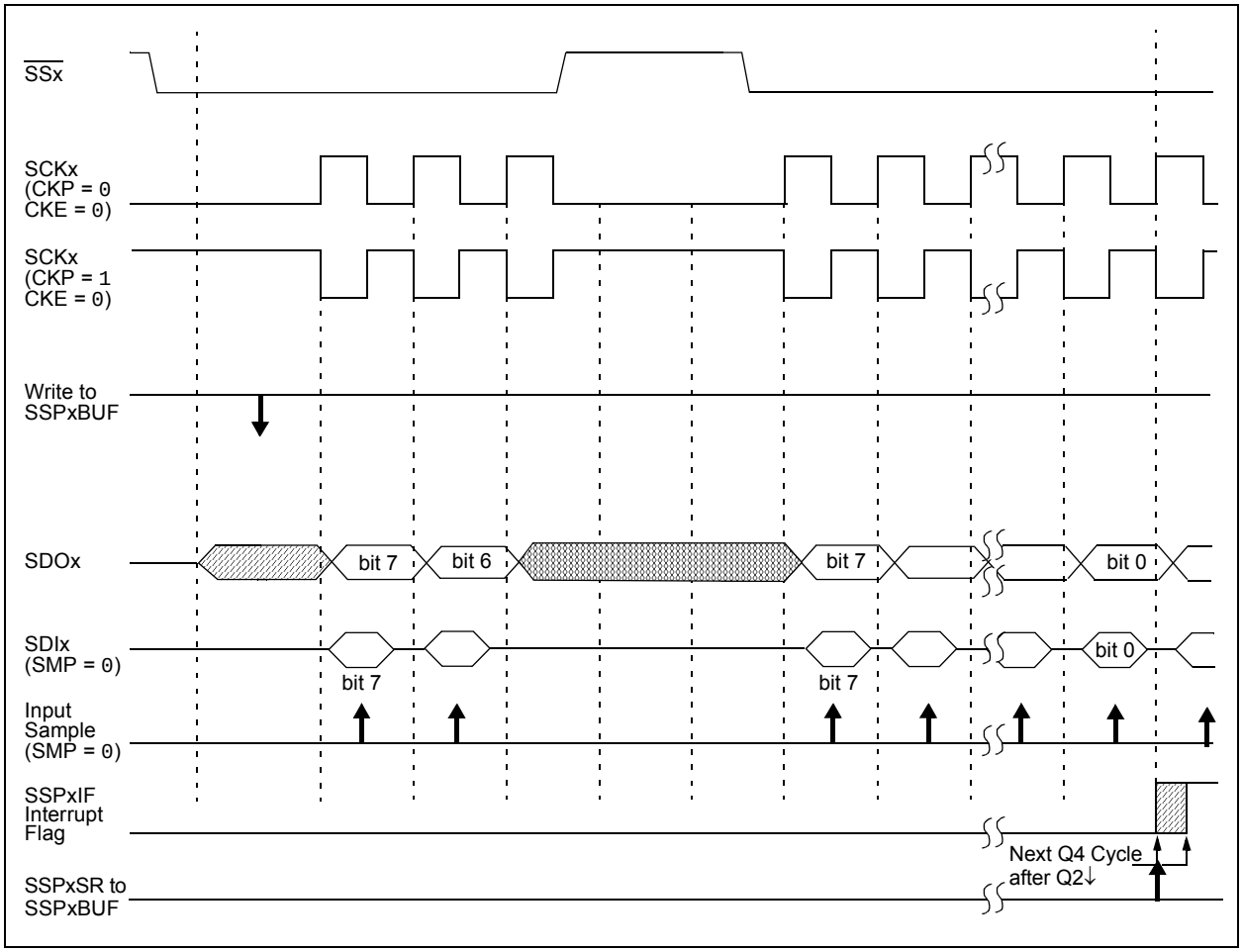


FIGURE 20-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

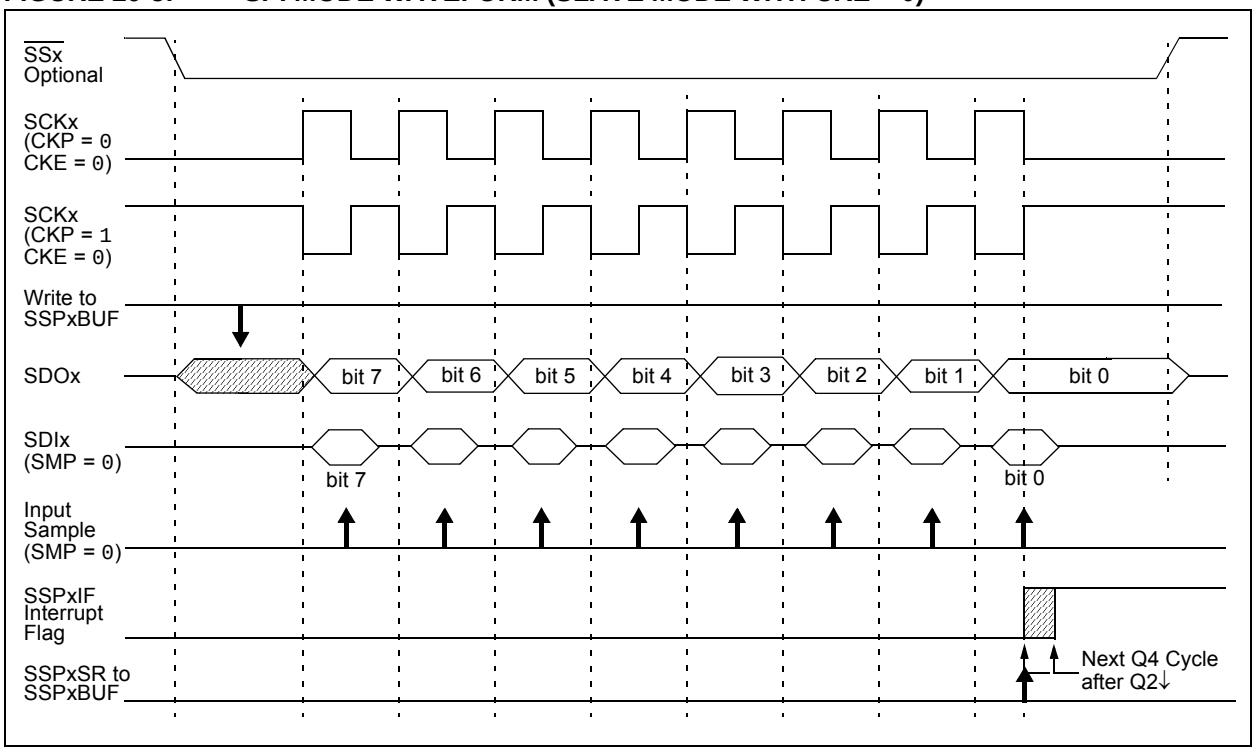
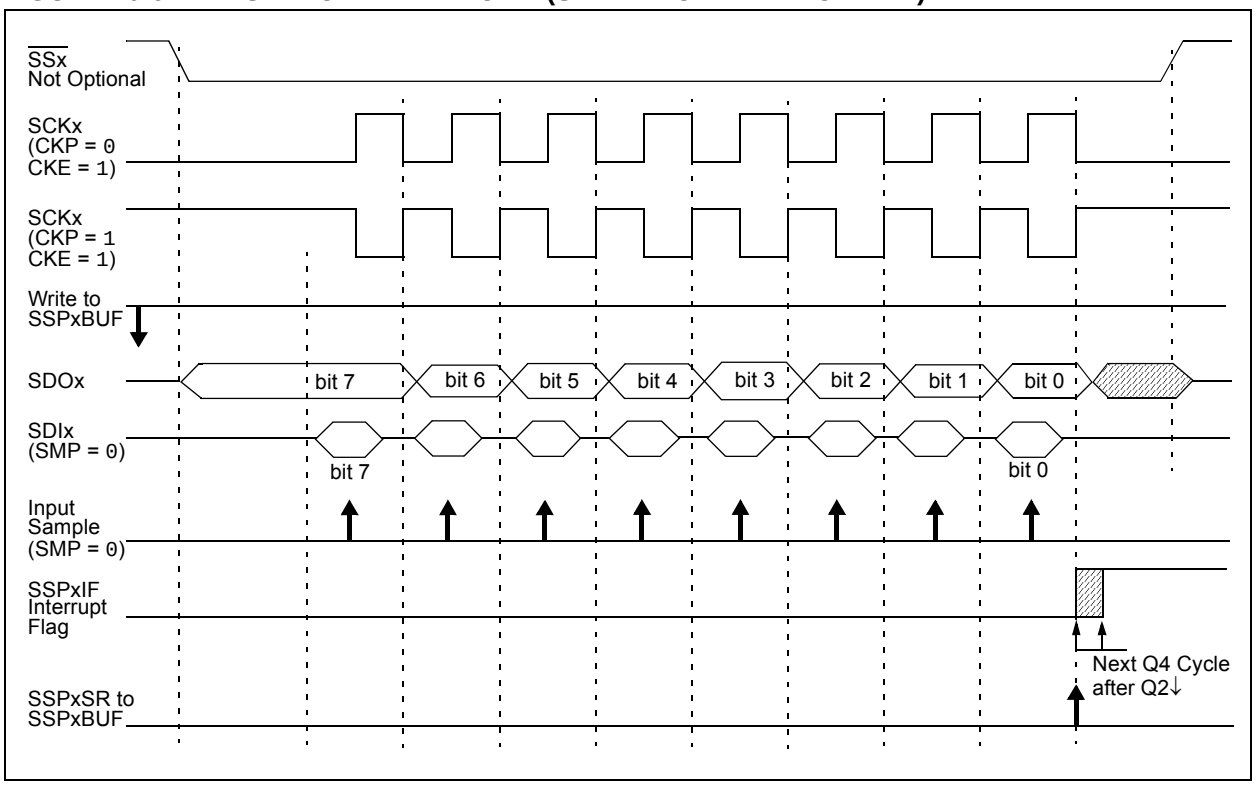


FIGURE 20-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



20.3.8 OPERATION IN POWER-MANAGED MODES

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

In Idle modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTRC source. See **Section 3.7 "Clock Sources and Oscillator Switching"** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the 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.

20.3.9 EFFECTS OF A RESET

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

20.3.10 BUS MODE COMPATIBILITY

Table 20-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 20-1: 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.

20.3.11 SPI CLOCK SPEED AND MODULE INTERACTIONS

Because MSSP1 and MSSP2 are independent modules, they can operate simultaneously at different data rates. Setting the SSPM<3:0> bits of the SSPxCON1 register determines the rate for the corresponding module.

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

TABLE 20-2: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF(1)	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE(1)	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP(1)	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	71
TRISD	TRISD7(1)	TRISD6(1)	TRISD5(1)	TRISD4(1)	TRISD3	TRISD2	TRISD1	TRISD0	71
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	71
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								70
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	70
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	70
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								73
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	73
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	73

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

Note 1: These bits are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'.

20.4 I²C Mode

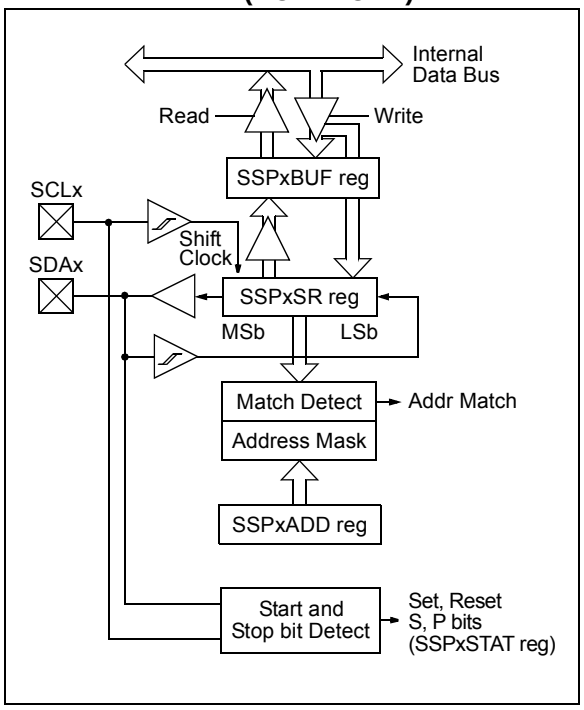
The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications, as well as 7-bit and 10-bit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

Two pins are used for data transfer:

- Serial clock (SCLx) – RC3/SCK1/SCL1 (or RD6/SCK2/SCL2 for 100-pin devices)
- Serial data (SDAx) – RC4/SDI1/SDA1 (or RD5/SDI2/SDA2 for 100-pin devices)

The user must configure these pins as inputs by setting the TRISC<4:3> or TRISD<5:4> bits.

FIGURE 20-7: MSSP BLOCK DIAGRAM (I²C™ MODE)



20.4.1 REGISTERS

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

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

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

Many of the bits in SSPxCON2 assume different functions, depending on whether the module is operating in Master or Slave mode. SSPxCON2<5:1> also assume different names in Slave mode. The different aspects of SSPxCON2 are shown in Register 20-5 (for Master mode) and Register 20-6 (Slave mode).

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

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

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

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

REGISTER 20-3: SSPxSTAT: MSSPx STATUS REGISTER (I²C™ MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$(2,3)	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/$\overline{Address}$ 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/$\overline{Write}$ Information bit (I²C mode only)(2,3)

In Slave mode:

1 = Read

0 = Write

In Master mode:

1 = Transmit is in progress

0 = Transmit is not in progress

bit 1 **UA:** Update Address bit (10-Bit Slave mode only)

1 = Indicates that the user needs to update the address in the SSPxADD register

0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit

In Transmit mode:

1 = SSPxBUF is full

0 = SSPxBUF is empty

In Receive mode:

1 = SSPxBUF is full (does not include the $\overline{ACK}$ and Stop bits)

0 = SSPxBUF is empty (does not include the $\overline{ACK}$ 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 $\overline{ACK}$ bit.

3: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Active mode.

REGISTER 20-4: SSPxCON1: MSSPx CONTROL REGISTER 1 (I²C™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

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

bit 7 **WCOL:** Write Collision Detect bit

In Master Transmit mode:
1 = A write to the SSPxBUF register was attempted while the I²C conditions were not valid for a transmission to be started (must be cleared in software)
0 = No collision

In Slave Transmit mode:
1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

In Receive mode (Master or Slave modes):
This is a "don't care" bit.

bit 6 **SSPOV:** Receive Overflow Indicator bit

In Receive mode:
1 = A byte is received while the SSPxBUF register is still holding the previous byte (must be cleared in software)
0 = No overflow

In Transmit mode:
This is a "don't care" bit in Transmit mode.

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

1 = Enables the serial port and configures the SDAx and SCLx pins as the serial port pins(1)
0 = Disables serial port and configures these pins as I/O port pins(1)

bit 4 **CKP:** SCKx Release Control bit

In Slave mode:
1 = Releases clock
0 = Holds clock low (clock stretch); used to ensure data setup time

In Master mode:
Unused in this mode.

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits

1111 = I²C Slave mode, 10-bit addressing with Start and Stop bit interrupts enabled(2)
1110 = I²C Slave mode, 7-bit addressing with Start and Stop bit interrupts enabled(2)
1011 = I²C Firmware Controlled Master mode (slave Idle)(2)
1000 = I²C Master mode, Clock = FOSC/(4 * (SSPADD + 1))(2)
0111 = I²C Slave mode, 10-bit addressing(2)
0110 = I²C Slave mode, 7-bit addressing(2)

Note 1: When enabled, the SDAx and SCLx pins must be configured as inputs.
2: Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

REGISTER 20-5: SSPxCON2: MSSPx 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 (Slave mode only)
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 Acknowledged
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit(2)
1 = Initiate Acknowledge sequence on SDAx and SCLx pins and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence Idle

bit 3 **RCEN:** Receive Enable bit (Master Receive mode only)(2)
1 = Enables Receive mode for I^2C
0 = Receive Idle

bit 2 **PEN:** Stop Condition Enable bit(2)
1 = Initiate Stop condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit(2)
1 = Initiate Repeated Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enable/Stretch Enable bit(2)
1 = Initiate Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Start condition Idle

Note 1: Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.
2: If the I^2C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

REGISTER 20-6: SSPxCON2: MSSPx 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 (Slave mode only)
1 = Enable interrupt when a general call address (0000h) is received in the SSPxSR
0 = General call address disabled

bit 6 **ACKSTAT:** Acknowledge Status bit
Unused in Slave mode.

bit 5-2 **ADMSK5:ADMSK2:** Slave Address Mask Select bits
1 = Masking of corresponding bits of SSPxADD is enabled
0 = Masking of corresponding bits of SSPxADD is disabled

bit 1 **ADMSK1:** Slave Address Least Significant Mask Select bit
In 7-Bit Addressing mode:
1 = Masking of SSPxADD<1> is only enabled
0 = Masking of SSPxADD<1> is only disabled
In 10-Bit Addressing mode:
1 = Masking of SSPxADD<1:0> is enabled
0 = Masking of SSPxADD<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 SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

20.4.2 OPERATION

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

The SSPxCON1 register allows control of the I^2C operation. Four mode selection bits (SSPxCON1<3:0>) allow one of the following I^2C modes to be selected:

- I^2C Master mode, clock = (FOSC/4) x (SSPxADD + 1)
- I^2C Slave mode (7-bit addressing)
- I^2C Slave mode (10-bit addressing)
- I^2C Slave mode (7-bit addressing) with Start and Stop bit interrupts enabled
- I^2C Slave mode (10-bit addressing) 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 SCLx and SDAx pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC or TRISD bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCLx and SDAx pins.

20.4.3 SLAVE MODE

In Slave mode, the SCLx and SDAx pins must be configured as inputs (TRISC<4:3> or TRISD<5:4> are 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 SSPxBUF register with the received value currently in the SSPxSR register.

Any combination of the following conditions will cause the MSSP module not to give this $\overline{ACK}$ pulse:

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

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

The SCLx clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification, as well as the requirement of the MSSP module, are shown in timing Parameter 100 and Parameter 101.

20.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPxSR register. All incoming bits are sampled with the rising edge of the clock (SCLx) line. The value of register SSPxSR<7:1> is compared to the value of the SSPxADD register. The address is compared on the falling edge of the eighth clock (SCLx) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPxSR register value is loaded into the SSPxBUF register.
2. The Buffer Full bit, BF, is set.
3. An $\overline{ACK}$ pulse is generated.
4. The MSSP Interrupt Flag bit, SSPxIF, is set (and the interrupt is generated, if enabled) on the falling edge of the ninth SCLx pulse.

In 10-Bit Addressing mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit, R/$\overline{W}$ (SSPxSTAT<2>), must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '11110 A9 A8 0', where 'A9' and 'A8' are the two MSbs of the address. The sequence of events for 10-bit addressing is as follows, with Steps 7 through 9 for the slave-transmitter:

1. Receive first (high) byte of address (bits, SSPxIF, BF and UA, are set).
2. Update the SSPxADD register with second (low) byte of address (clears bit, UA, and releases the SCLx line).
3. Read the SSPxBUF register (clears bit, BF) and clear flag bit, SSPxIF.
4. Receive second (low) byte of address (bits, SSPxIF, BF and UA, are set).
5. Update the SSPxADD register with the first (high) byte of address. If match releases SCLx line, this will clear bit, UA.
6. Read the SSPxBUF register (clears bit, BF) and clear flag bit, SSPxIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits, SSPxIF and BF, are set).
9. Read the SSPxBUF register (clears bit, BF) and clear flag bit, SSPxIF.

20.4.3.2 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 mode, and up to 63 addresses in 10-bit mode (see Example 20-2).

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 SSPxBUF.

In 7-Bit Addressing mode, address mask bits, ADMSK<5:1> (SSPxCON2<5:1>), mask the corresponding address bits in the SSPxADD register. For any ADMSK bits that are set (ADMSK<n> = 1), the corresponding address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

In 10-Bit Addressing mode, bits, ADMSK<5:2>, mask the corresponding address bits in the SSPxADD register. In addition, ADMSK1 simultaneously masks the two LSbs of the address (SSPxADD<1:0>). For any ADMSK bits that are active (ADMSK<n> = 1), the corresponding address bit is ignored (SSPxADD<n> = x). Also note, that although in 10-Bit Addressing mode, the upper address bits re-use part of the SSPxADD register bits. The address mask bits do not interact with those bits; they only affect the lower address bits.

Note 1: ADMSK1 masks the two Least Significant bits of the address.

2: The two Most Significant bits of the address are not affected by address masking.

EXAMPLE 20-2: ADDRESS MASKING EXAMPLES

7-Bit Addressing:

SSPxADD<7:1> = A0h (1010000) (SSPxADD<0> is assumed to be '0')

ADMSK<5:1> = 00111

Addresses Acknowledged: A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

10-Bit Addressing:

SSPxADD<7:0> = A0h (10100000) (the two MSb 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

20.4.3.3 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 SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and the SDAx line is held low ($\overline{ACK}$).

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit, BF (SSPxSTAT<0>), is set, or bit, SSPOV (SSPxCON1<6>), is set.

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

If SEN is enabled (SSPxCON2<0> = 1), SCKx/SCLx (RC3 or RD6) will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPxCON1<4>). See **Section 20.4.4 "Clock Stretching"** for more details.

20.4.3.4 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 SSPxSTAT register is set. The received address is loaded into the SSPxBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RC3 or RD6 is held low, regardless of SEN (see **Section 20.4.4 "Clock Stretching"** for more details). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPxBUF register which also loads the SSPxSR register. Then, pin, RC3 or RD6, should be enabled by setting bit, CKP (SSPxCON1<4>). The eight data bits are shifted out on the falling edge of the SCLx input. This ensures that the SDAx signal is valid during the SCLx high time (Figure 20-10).

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCLx input pulse. If the SDAx line is high (not $\overline{ACK}$), then the data transfer is complete. In this case, when the $\overline{ACK}$ is latched by the slave, the slave logic is reset (resets SSPxSTAT register) and the slave monitors for another occurrence of the Start bit. If the SDAx line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPxBUF register. Again, pin, RC3 or RD6, must be enabled by setting bit, CKP.

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

FIGURE 20-8: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)

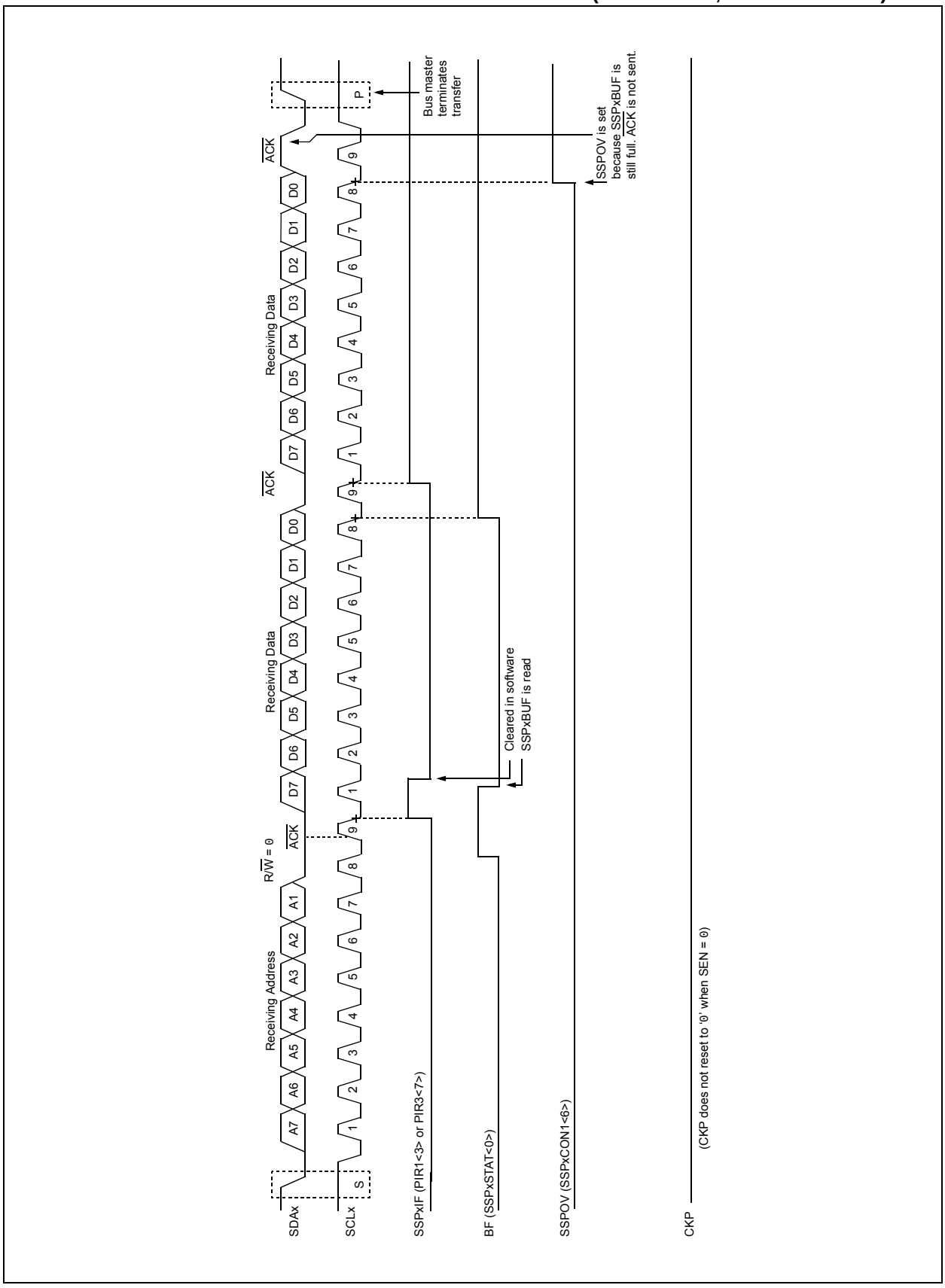
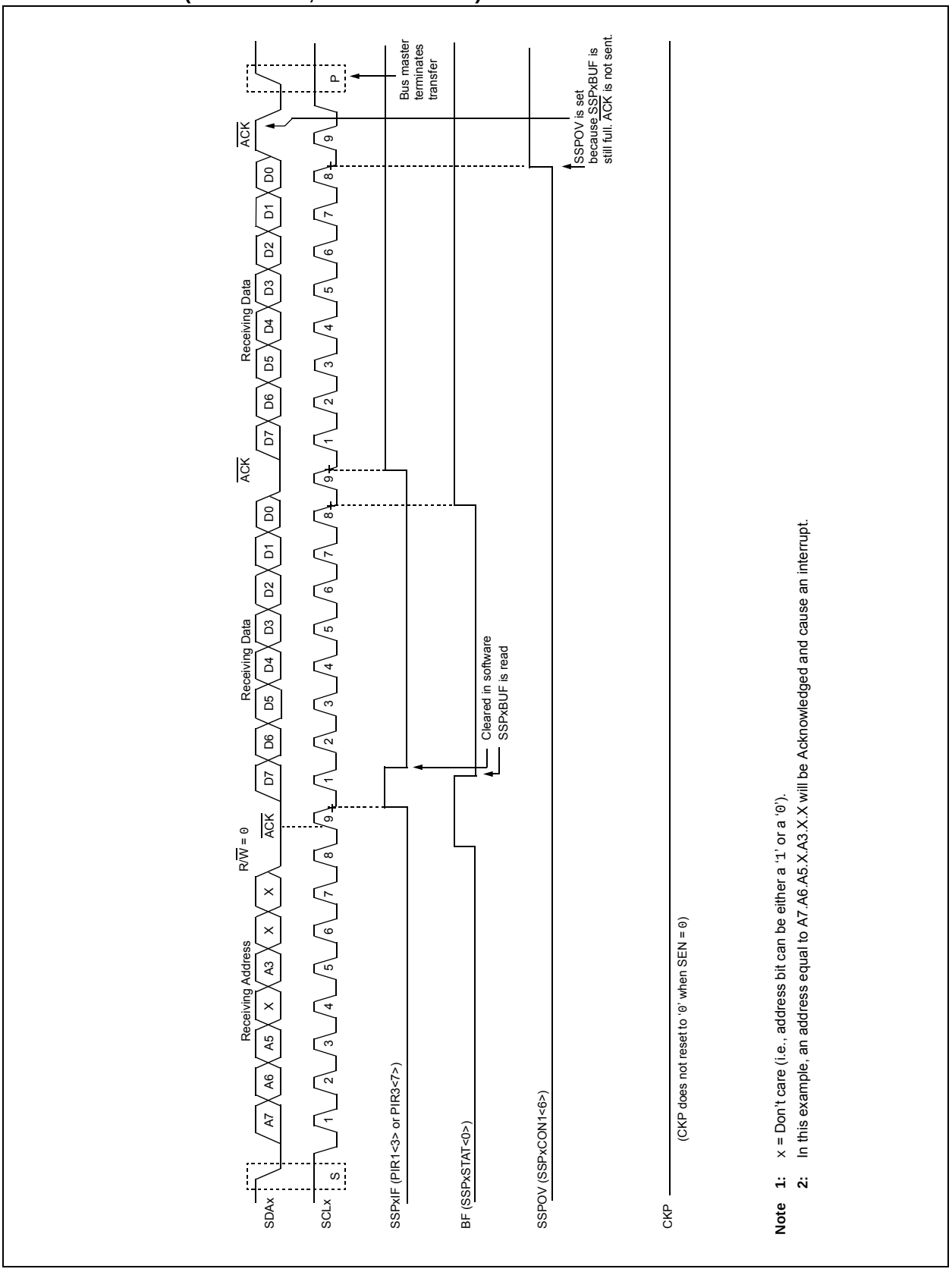


FIGURE 20-9: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01011 (RECEPTION, 7-BIT ADDRESS)



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 20-10: I²C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

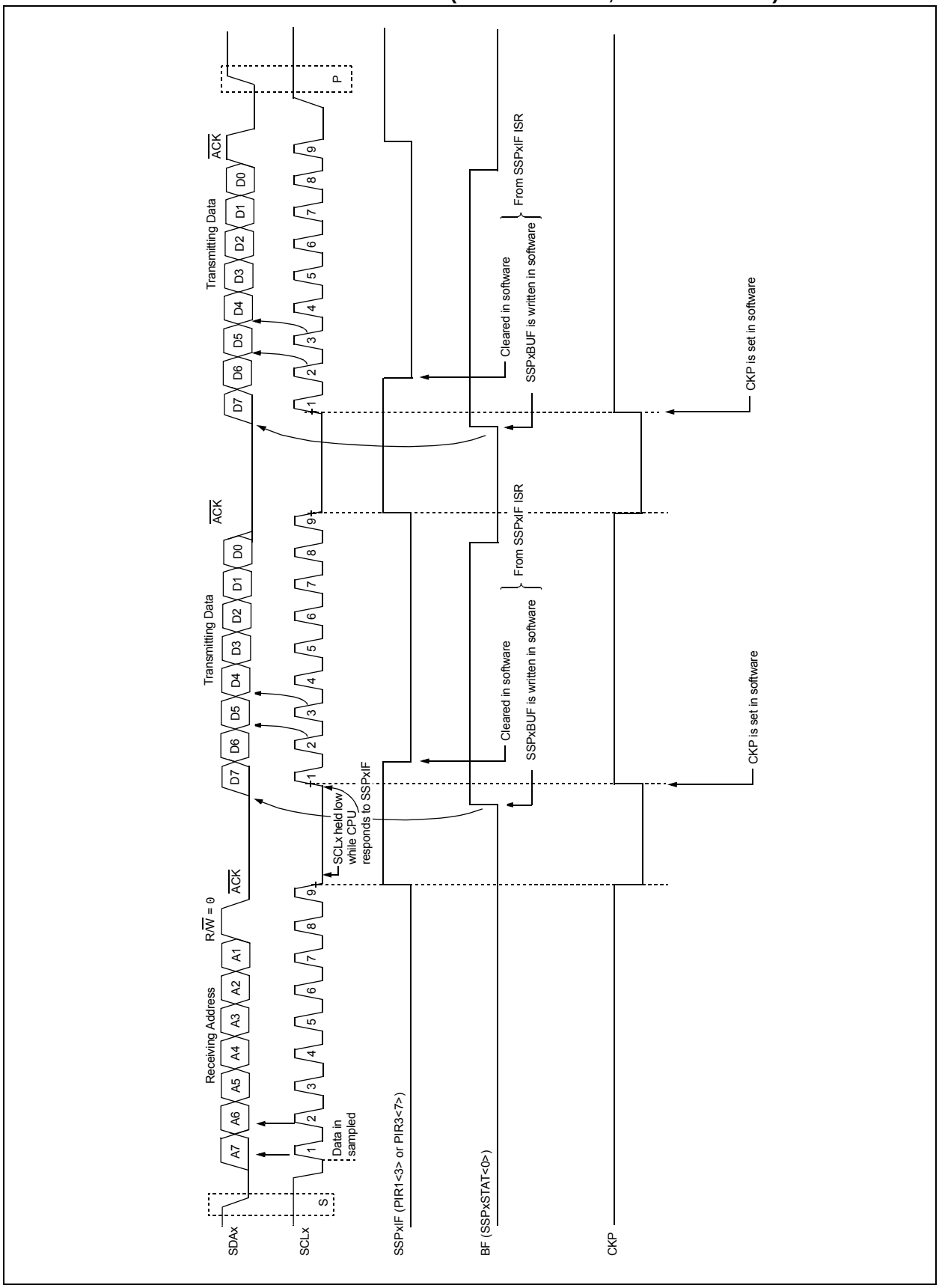


FIGURE 20-11: I²C™ SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)

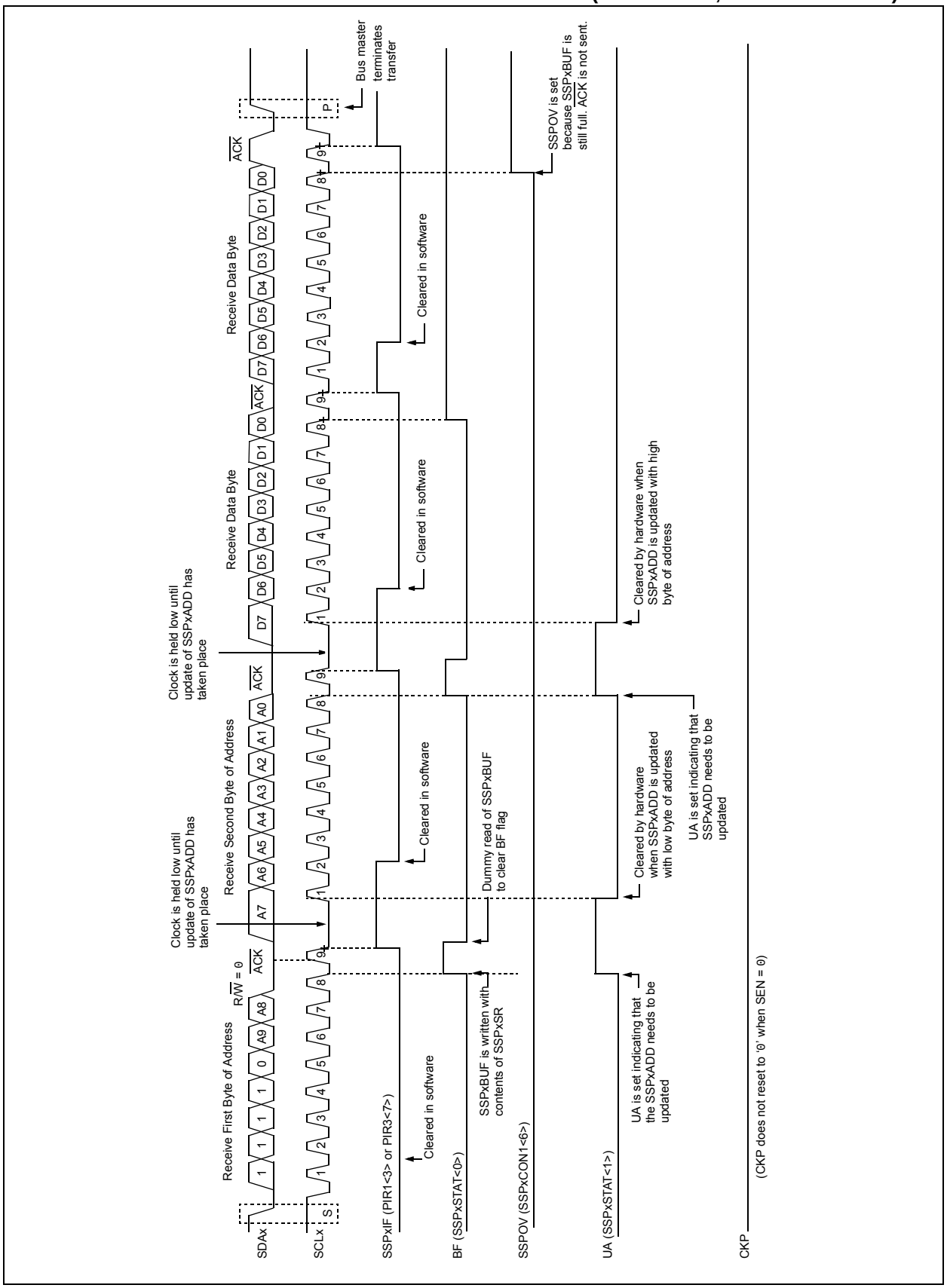
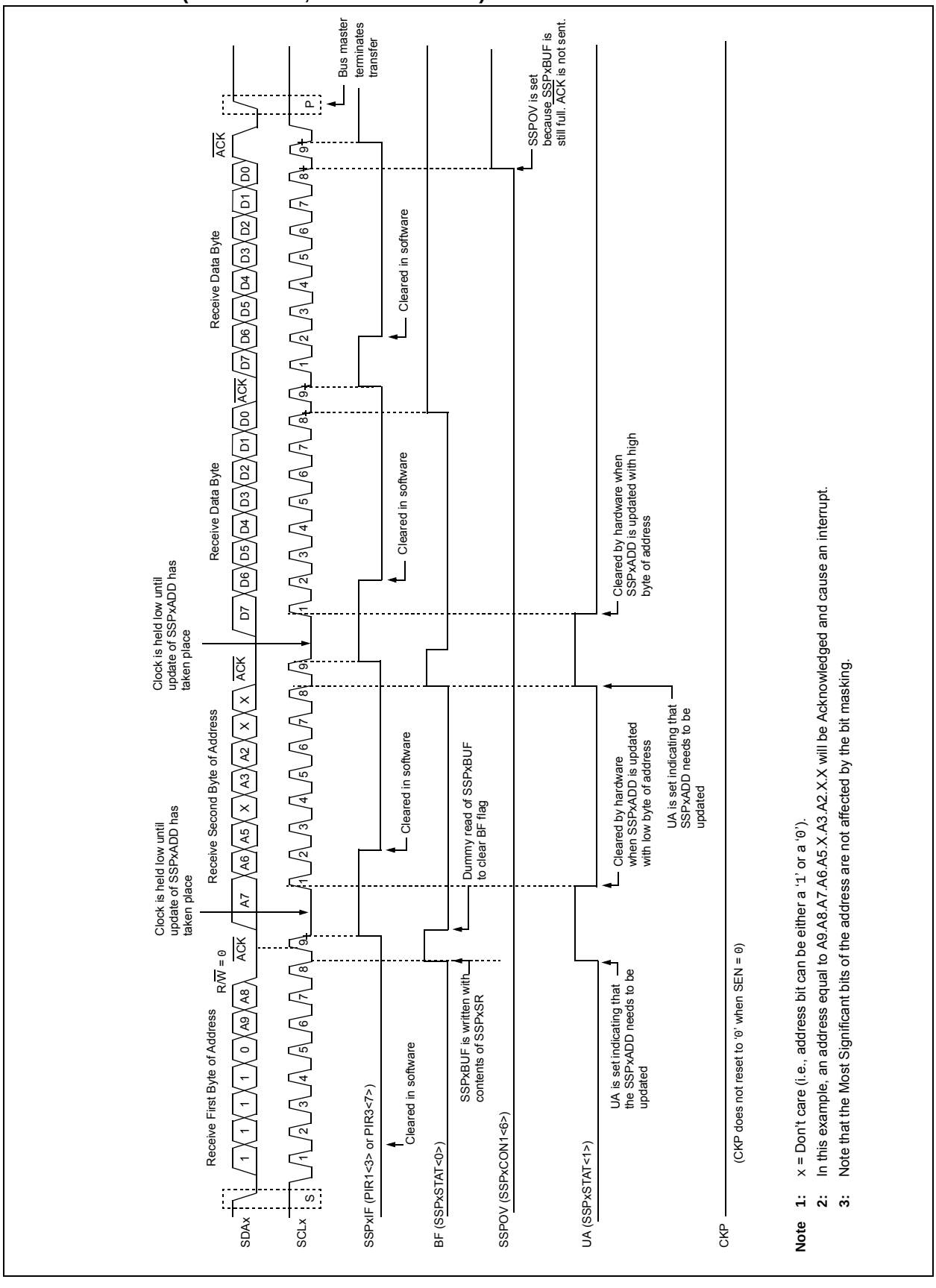


FIGURE 20-12: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01001 (RECEPTION, 10-BIT ADDRESS)

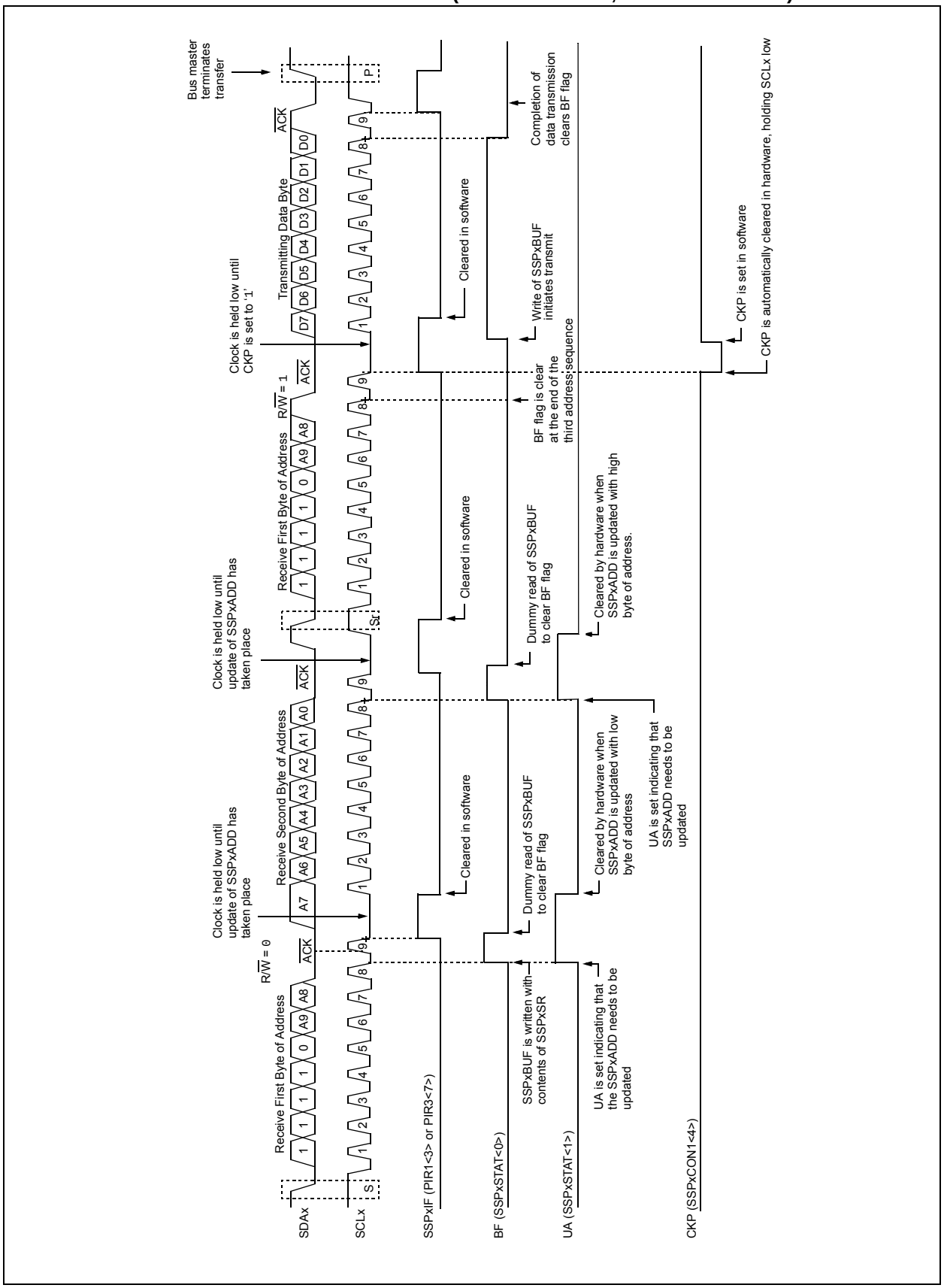


Note 1: x = Don't care (i.e., address bit can be either a '1' or a '0').

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 20-13: I²C™ SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



20.4.4 CLOCK STRETCHING

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

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

20.4.4.1 Clock Stretching for 7-Bit Slave Receive Mode (SEN = 1)

In 7-Bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence, if the BF bit is set, the CKP bit in the SSPxCON1 register is automatically cleared, forcing the SCLx output to be held low. The CKP being cleared to '0' will assert the SCLx line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and read the contents of the SSPxBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 20-15).

Note 1: If the user reads the contents of the SSPxBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

20.4.4.2 Clock Stretching for 10-Bit Slave Receive Mode (SEN = 1)

In 10-Bit Slave Receive mode during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPxADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

Note: If the user polls the UA bit and clears it by updating the SSPxADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPxBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

20.4.4.3 Clock Stretching for 7-Bit Slave Transmit Mode

The 7-Bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock, if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and load the contents of the SSPxBUF before the master device can initiate another transmit sequence (see Figure 20-10).

Note 1: If the user loads the contents of SSPxBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software regardless of the state of the BF bit.

20.4.4.4 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 20-13).

20.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCLx output is forced to '0'. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external I^2C master device has already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I^2C bus have deasserted SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 20-14).

FIGURE 20-14: CLOCK SYNCHRONIZATION TIMING

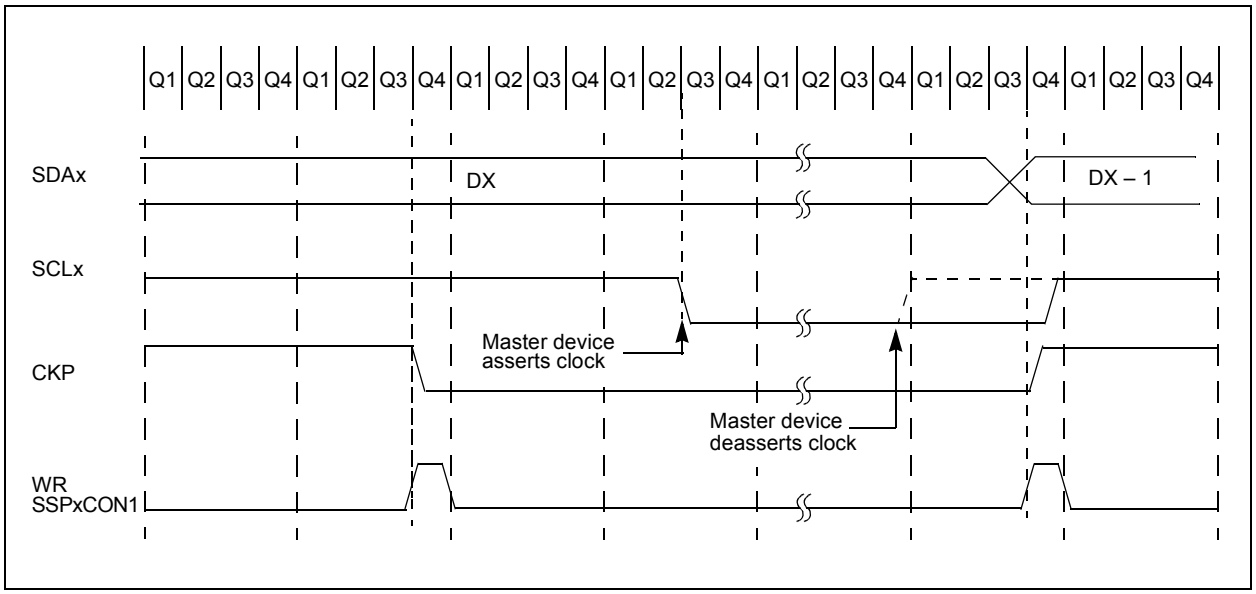


FIGURE 20-15: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)

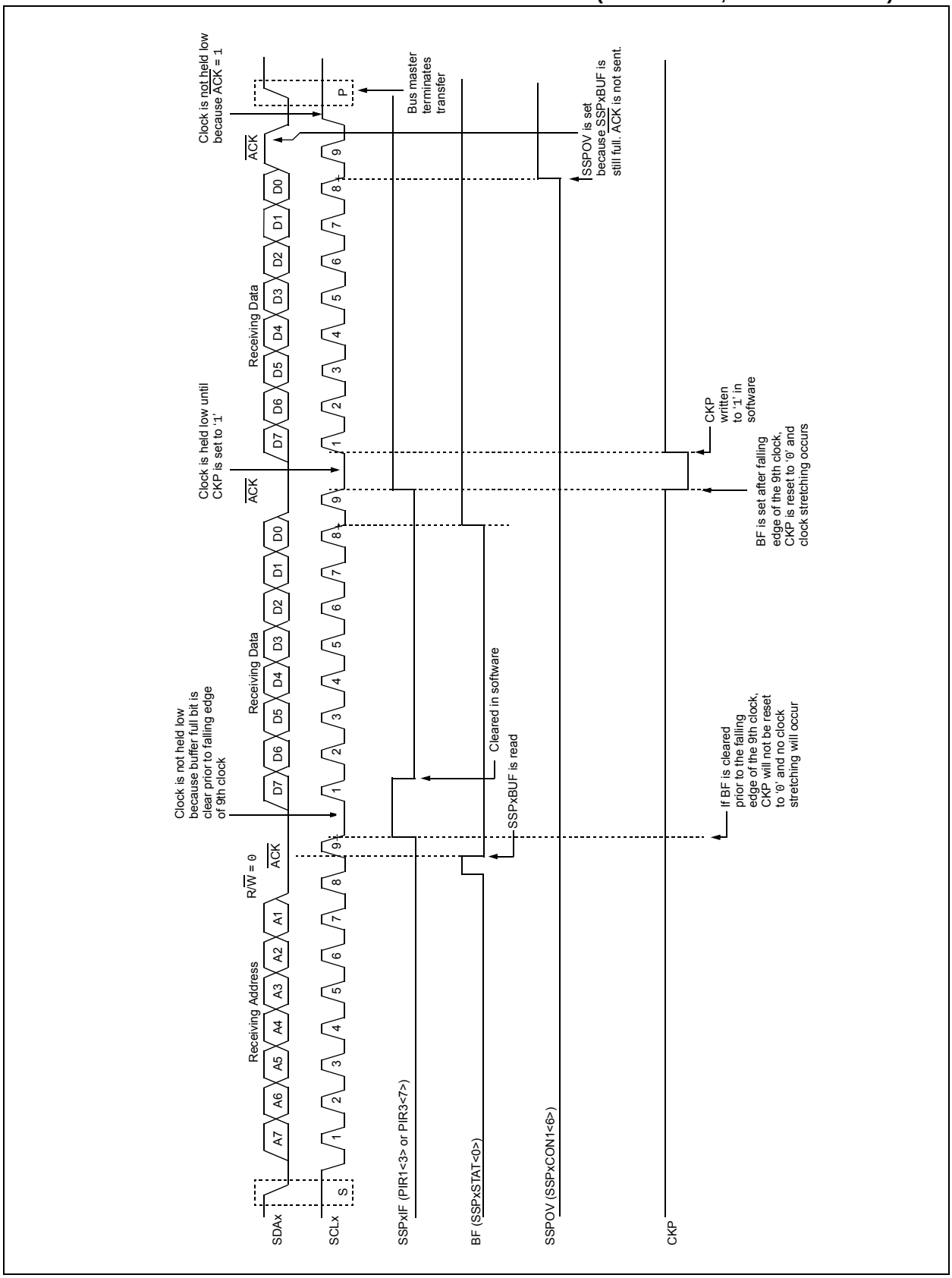
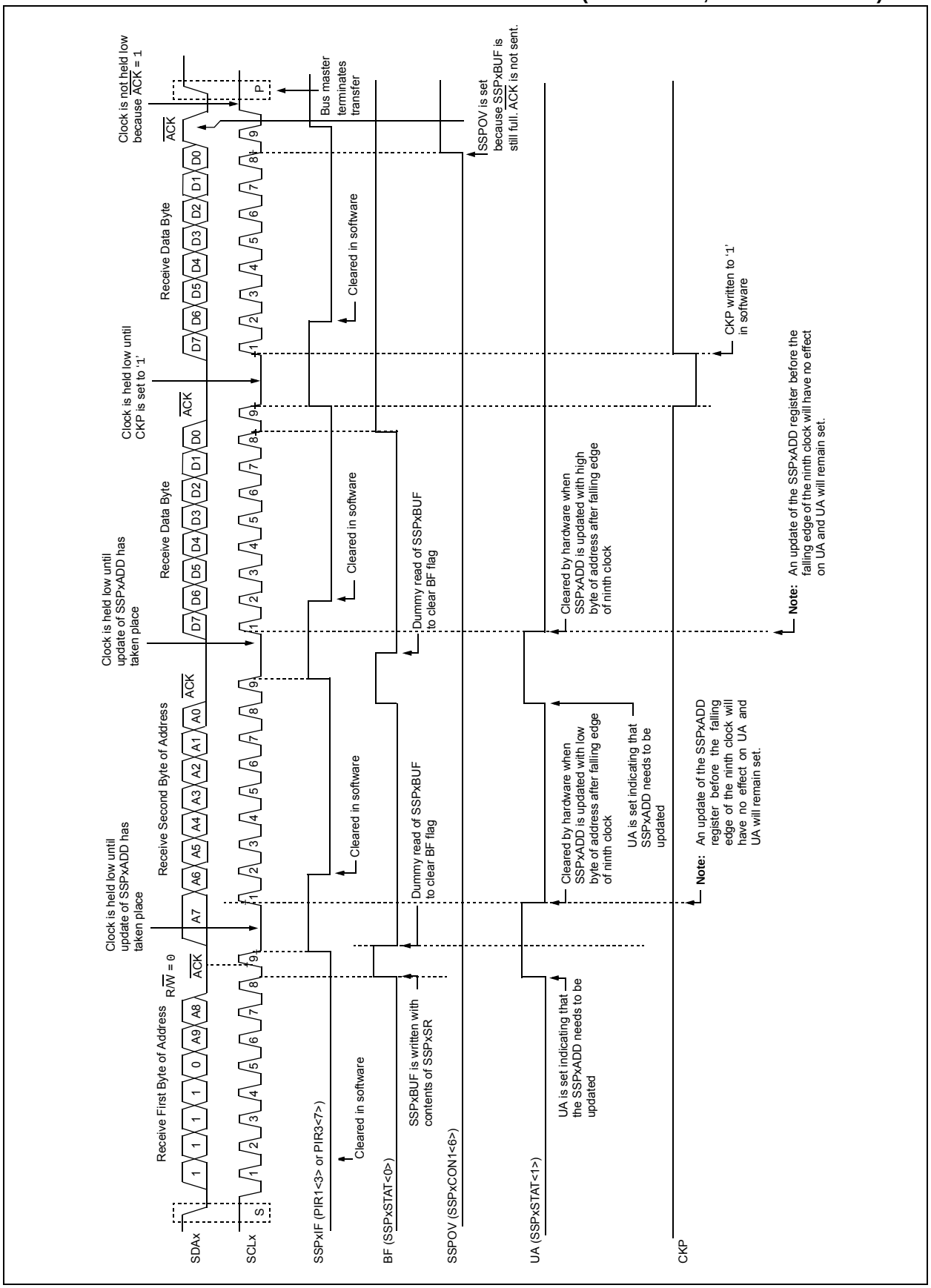


FIGURE 20-16: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)



20.4.5 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.

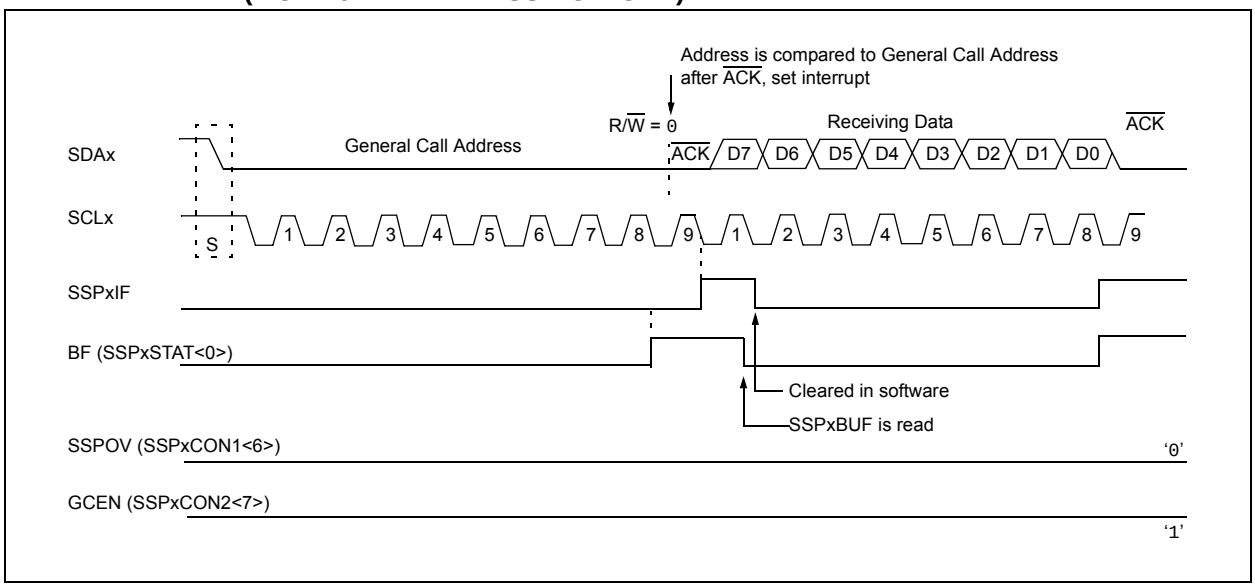
The general call address is recognized when the General Call Enable bit, GCEN, is enabled (SSPxCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPxSR and the address is compared against the SSPxADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPxSR is transferred to the SSPxBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPxIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPxBUF. The value can be used to determine if the address was device-specific or a general call address.

In 10-Bit Addressing mode, the SSPxADD is required to be updated for the second half of the address to match, and the UA bit is set (SSPxSTAT<1>). If the general call address is sampled when the GCEN bit is set, while the slave is configured in 10-Bit Addressing mode, then the second half of the address is not necessary, the UA bit will not be set and the slave will begin receiving data after the Acknowledge (Figure 20-17).

FIGURE 20-17: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESSING MODE)



20.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPxCON1 and by setting the SSPEN bit. In Master mode, the SCLx and SDAx lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the 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.

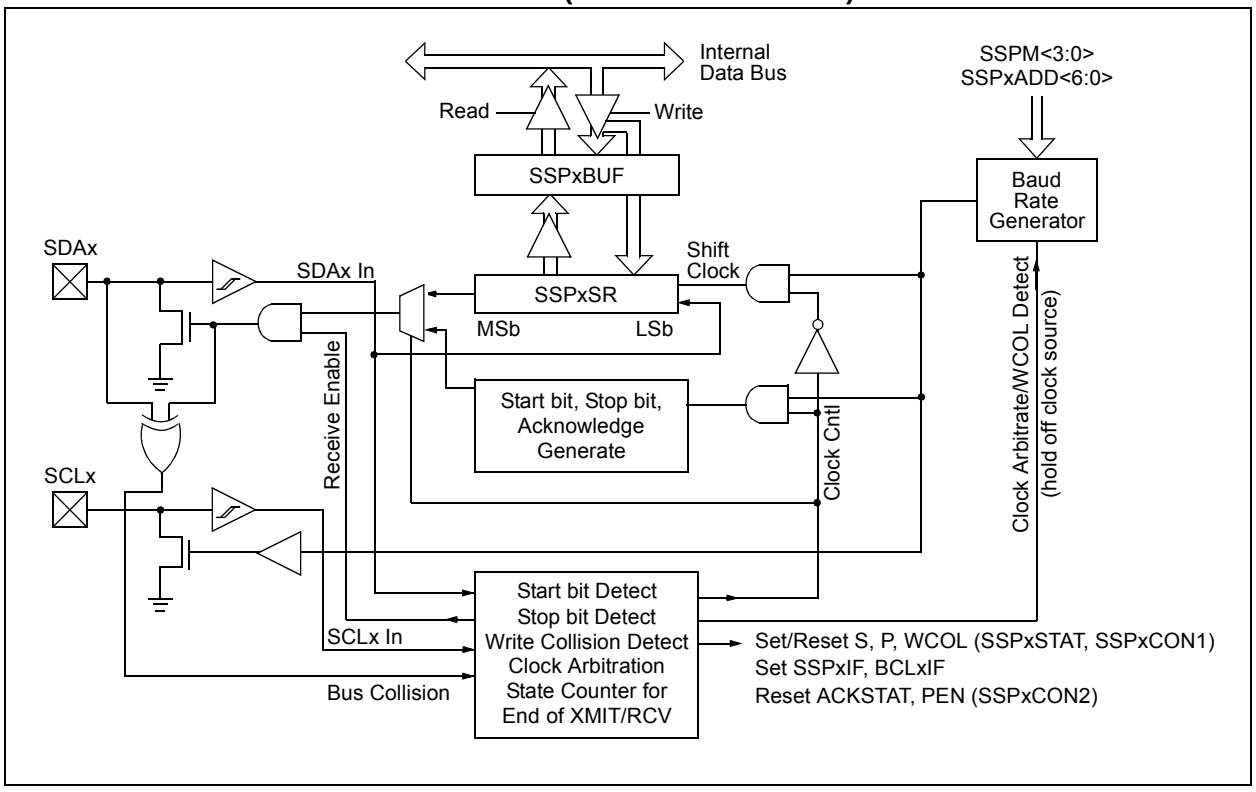
1. Assert a Start condition on SDAx and SCLx.
2. Assert a Repeated Start condition on SDAx and SCLx.
3. Write to the SSPxBUF register, initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDAx and SCLx.

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 SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur.

The following events will cause the MSSP Interrupt Flag bit, SSPxIF, to be set (and MSSP interrupt, if enabled):

- Start Condition
- Stop Condition
- Data Transfer Byte Transmitted/Received
- Acknowledge Transmit
- Repeated Start

FIGURE 20-18: MSSP BLOCK DIAGRAM (I^2C™ MASTER MODE)



20.4.6.1 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 SDAx, while SCLx outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) 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 SDAx, while SCLx outputs the serial clock. Serial data is received, 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. 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 SCLx clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 20.4.7 "Baud Rate"** for more detail.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start Enable bit, SEN (SSPxCON2<0>).
2. SSPxIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPxBUF with the slave address to transmit.
4. Address is shifted out on the SDAx pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPxCON2 register (SSPxCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
7. The user loads the SSPxBUF with eight bits of data.
8. Data is shifted out on the SDAx pin until all 8 bits are transmitted.
9. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPxCON2 register (SSPxCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
11. The user generates a Stop condition by setting the Stop Enable bit, PEN (SSPxCON2<2>).
12. Interrupt is generated once the Stop condition is complete.

20.4.7 BAUD RATE

In I^2C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPxADD register (Figure 20-19). When a write occurs to SSPxBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (T_{CY}) 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 SCLx pin will remain in its last state.

Table 20-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

20.4.7.1 Baud Rate and Module Interdependence

Because MSSP1 and MSSP2 are independent, they can operate simultaneously in I^2C Master mode at different baud rates. This is done by using different BRG reload values for each module.

Because this mode derives its basic clock source from the system clock, any changes to the clock will affect both modules in the same proportion. It may be possible to change one or both baud rates back to a previous value by changing the BRG reload value.

FIGURE 20-19: BAUD RATE GENERATOR BLOCK DIAGRAM

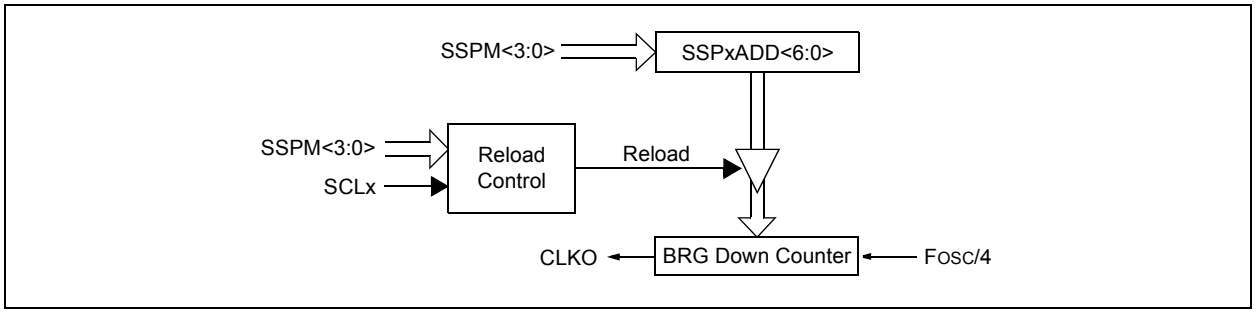


TABLE 20-3: I^2C™ CLOCK RATE w/BRG

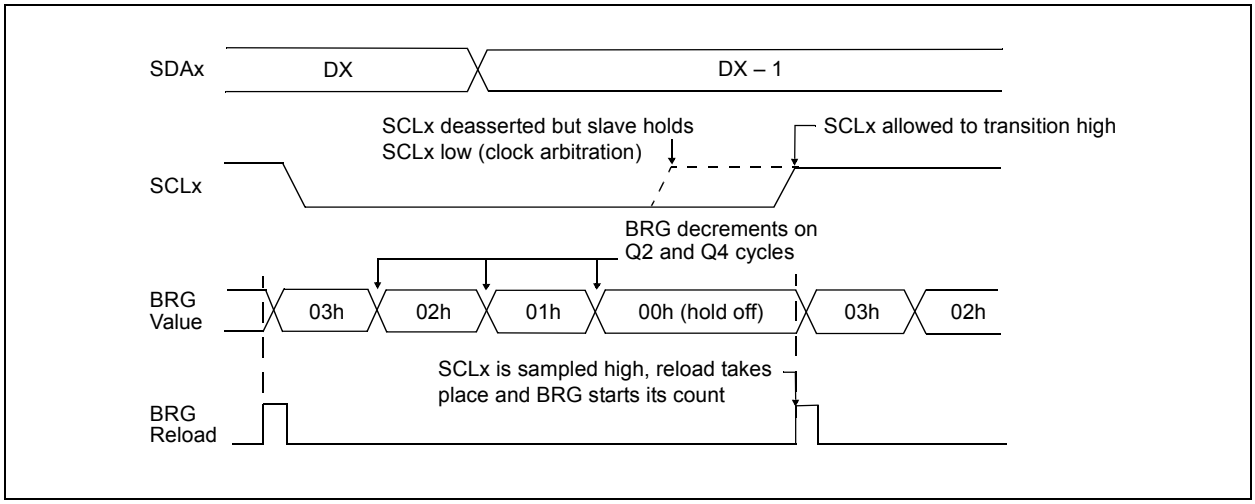
FOSC	BRG Value	FSCL (2 Rollovers of BRG)
41.667 MHz	19h	400 kHz(1)
41.667 MHz	67h	100 kHz
31.25 MHz	13h	400 kHz(1)
31.25 MHz	4Dh	100 kHz
20.833 MHz	09h	400 kHz(1)
20.833 MHz	33h	100 kHz

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.

20.4.7.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 20-20).

FIGURE 20-20: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



20.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPxCON2<0>). If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit (SSPxSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPxCON2<0>) will be automatically cleared by hardware. The Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

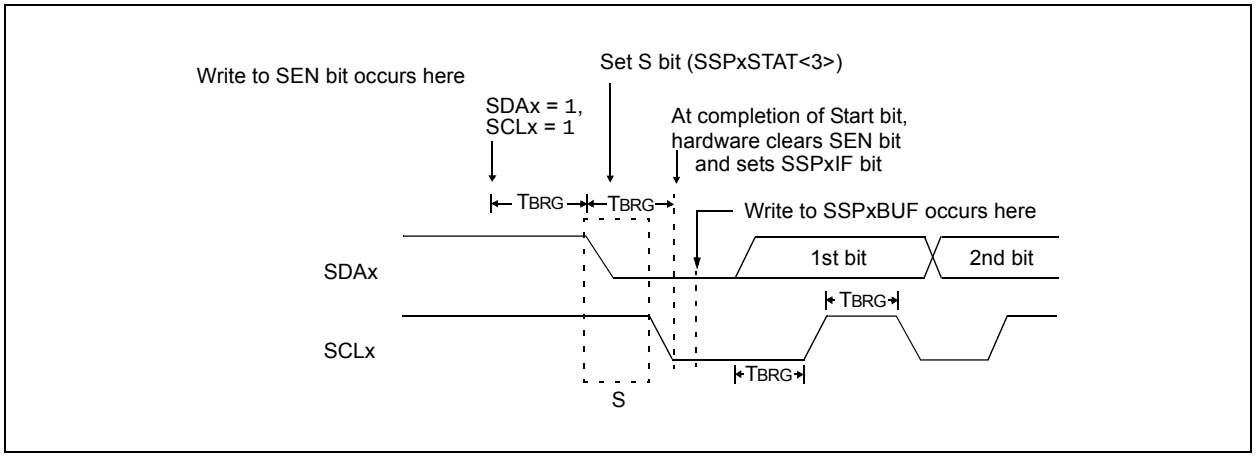
Note:	If, at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs. The Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

20.4.8.1 WCOL Status Flag

If the user writes the SSPxBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPxCON2 is disabled until the Start condition is complete.

FIGURE 20-21: FIRST START BIT TIMING



20.4.9 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPxCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPxADD<6:0> and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. Following this, the RSEN bit (SSPxCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit (SSPxSTAT<3>) will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

Note 1: If RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

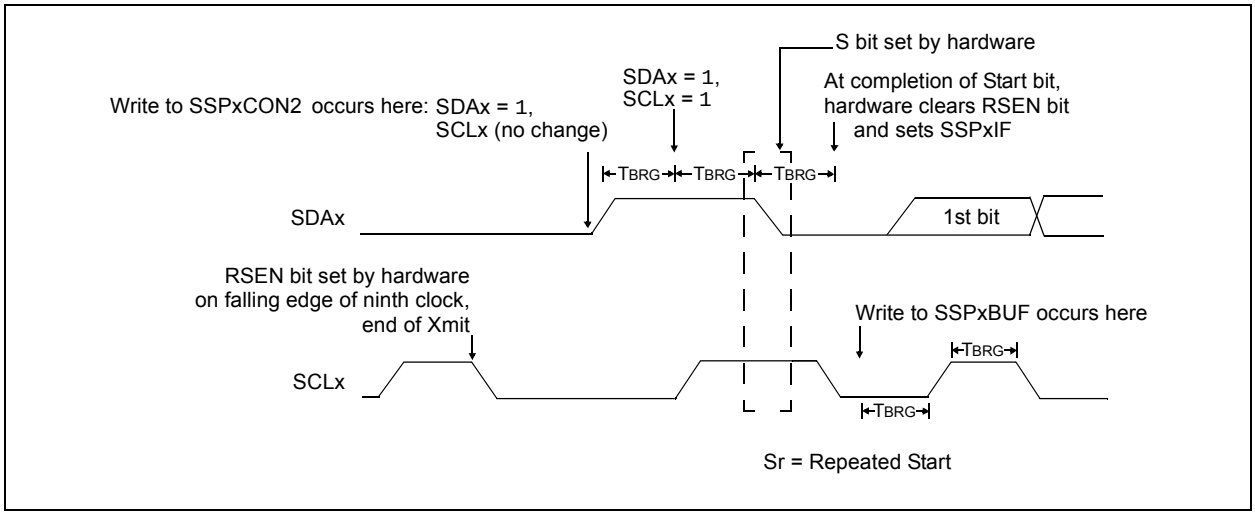
Immediately following the SSPxIF bit getting set, the user may write the SSPxBUF with the 7-bit address in 7-bit mode or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

20.4.9.1 WCOL Status Flag

If the user writes the SSPxBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPxCON2 is disabled until the Repeated Start condition is complete.

FIGURE 20-22: REPEATED START CONDITION WAVEFORM



20.4.10 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPxBUF register. This action will set the Buffer Full flag bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDAx pin after the falling edge of SCLx is asserted (see data hold time specification Parameter 106). SCLx is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCLx is released high (see data setup time specification Parameter 107). When the SCLx pin is released high, it is held that way for TBRG. The data on the SDAx pin must remain stable for that duration and some hold time after the next falling edge of SCLx. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDAx. This allows the slave device being addressed to respond with an $\overline{ACK}$ 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 SSPxIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 20-23).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCLx until all seven address bits and the R/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the master will deassert the SDAx pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDAx pin to see if the address was recognized by a slave. The status of the $\overline{ACK}$ bit is loaded into the ACKSTAT status bit (SSPxCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPxIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

20.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPxSTAT<0>) is set when the CPU writes to SSPxBUF, and is cleared when all 8 bits are shifted out.

20.4.10.2 WCOL Status Flag

If the user writes to the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur) after 2 TCY after the SSPxBUF write. If SSPxBUF is rewritten within 2 TCY, the WCOL bit is set and SSPxBUF is updated. This may result in a corrupted transfer.

The user should verify that the WCOL is clear after each write to SSPxBUF to ensure the transfer is correct. In all cases, WCOL must be cleared in software.

20.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPxCON2<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.

20.4.11 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPxCON2<3>).

Note:	The MSSP module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting and on each rollover. The state of the SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx low. The MSSP is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPxCON2<4>).

20.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

20.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

20.4.11.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 20-23: I²C™ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)

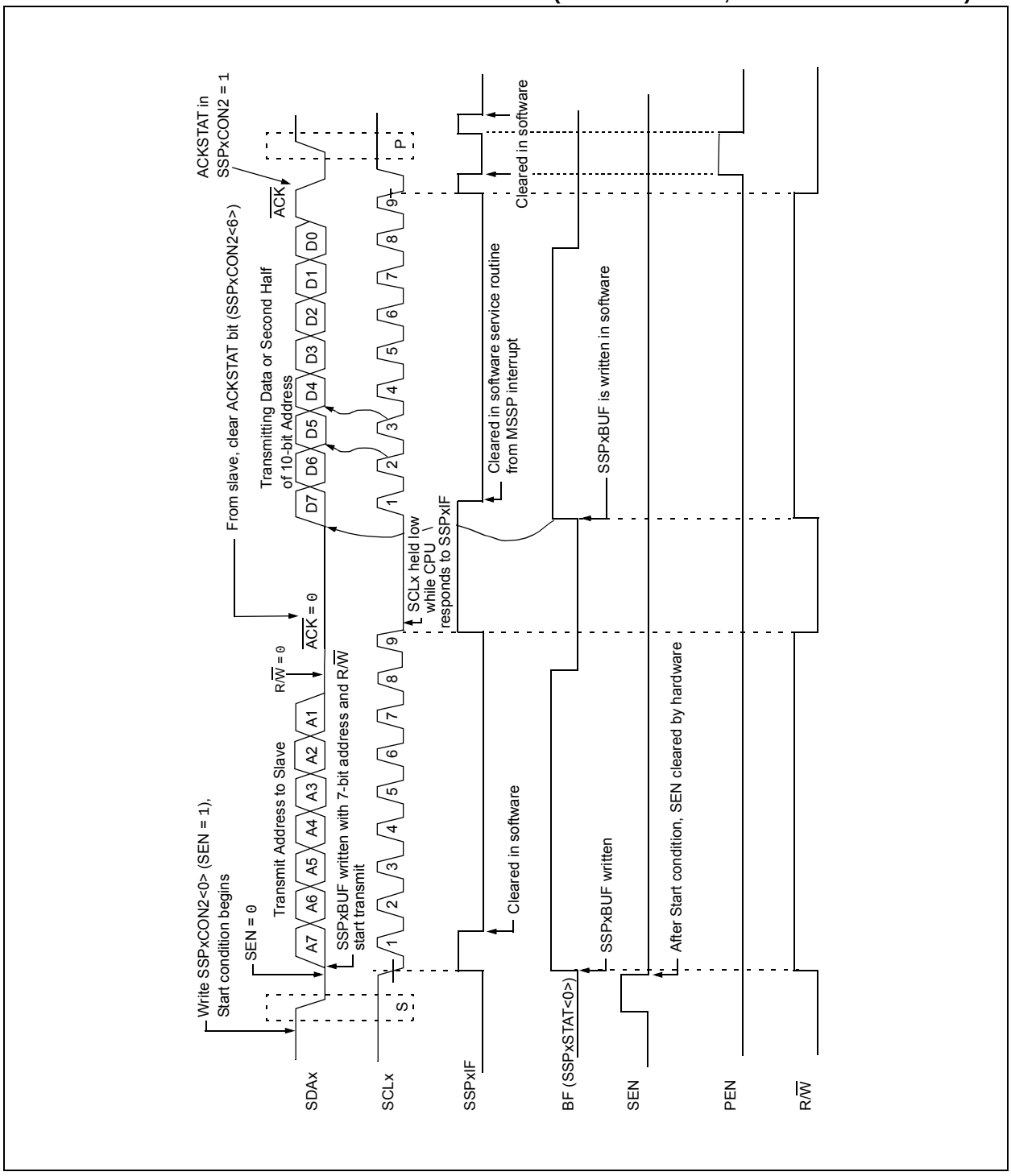
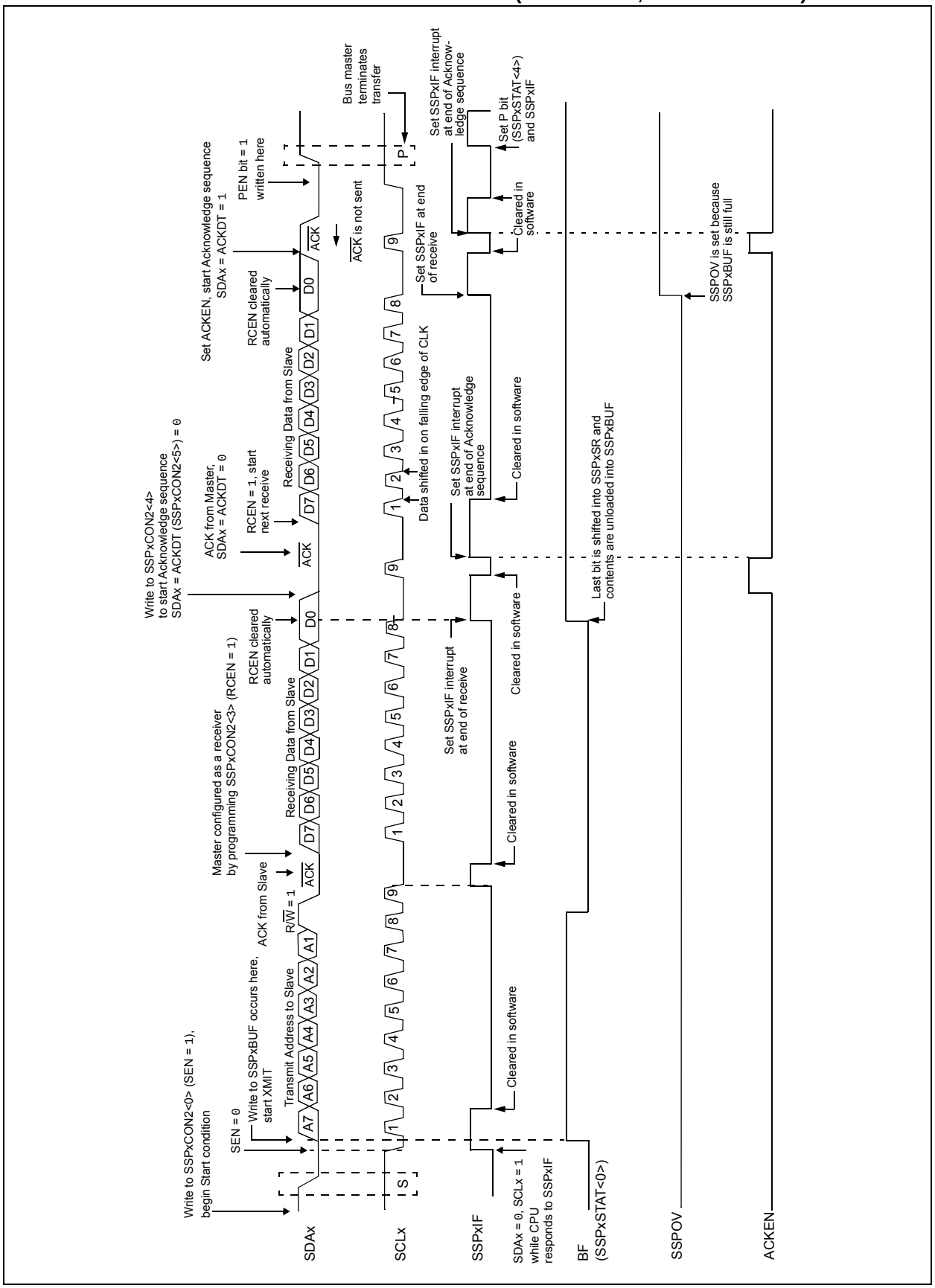


FIGURE 20-24: I²C™ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



20.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPxCON2<4>). When this bit is set, the SCLx pin is pulled low and the contents of the Acknowledge data bit are presented on the SDAx pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCLx pin is deasserted (pulled high). When the SCLx pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCLx pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 20-25).

20.4.12.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

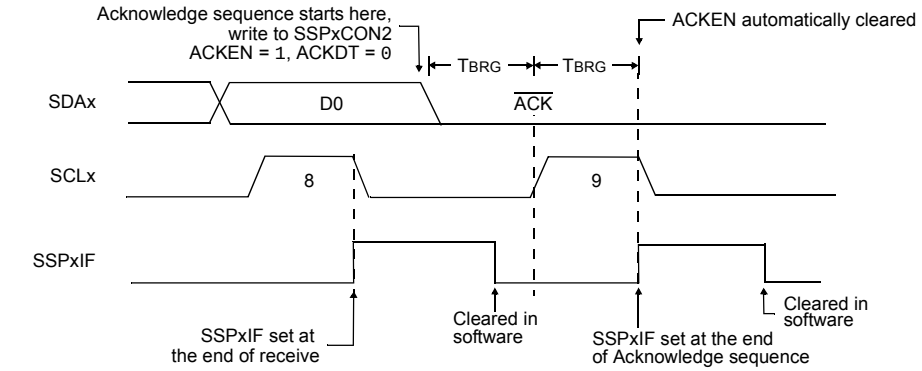
20.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDAx pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPxCON2<2>). At the end of a receive/transmit, the SCLx line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDAx line low. When the SDAx line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCLx pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the P bit (SSPxSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 20-26).

20.4.13.1 WCOL Status Flag

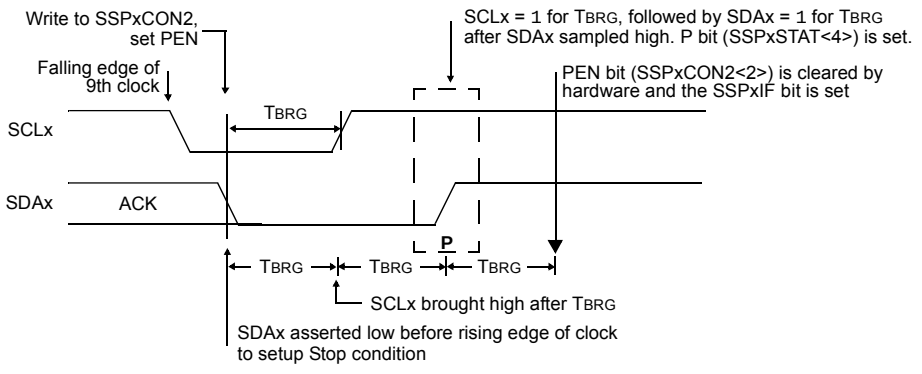
If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 20-25: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 20-26: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one Baud Rate Generator period.

20.4.14 SLEEP OPERATION

While in Sleep mode, the I²C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

20.4.15 EFFECTS OF A RESET

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

20.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I²C bus may be taken when the P bit (SSPxSTAT<4>) is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the MSSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDAx line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed in hardware with the result placed in the BCLxIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

20.4.17 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF and reset the I²C port to its Idle state (Figure 20-27).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I²C bus is free, the user can resume communication by asserting a Start condition.

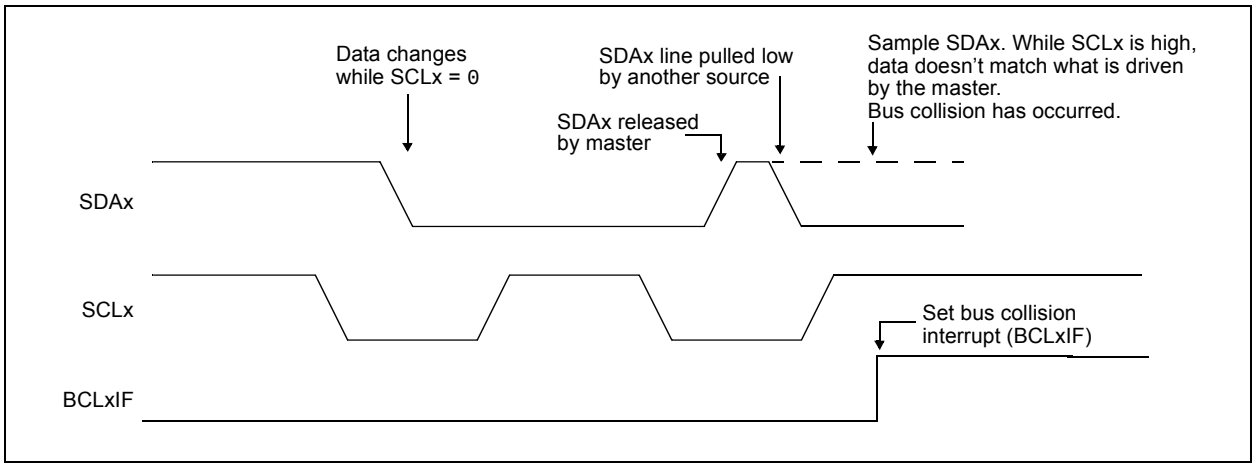
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I²C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

A write to the SSPxBUF will start the transmission of data at the first data bit regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I²C bus can be taken when the P bit is set in the SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 20-27: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



20.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDAx or SCLx are sampled low at the beginning of the Start condition (Figure 20-28).

b) SCLx is sampled low before SDAx is asserted low (Figure 20-29).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

- the Start condition is aborted;
- the BCLxIF flag is set; and
- the MSSP module is reset to its Idle state (Figure 20-28).

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded from SSPxADD<6:0> and counts down to 0. If the SCLx pin is sampled low while SDAx is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 20-30). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to 0. If the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx pin is asserted low.

Note: The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 20-28: BUS COLLISION DURING START CONDITION (SDAx ONLY)

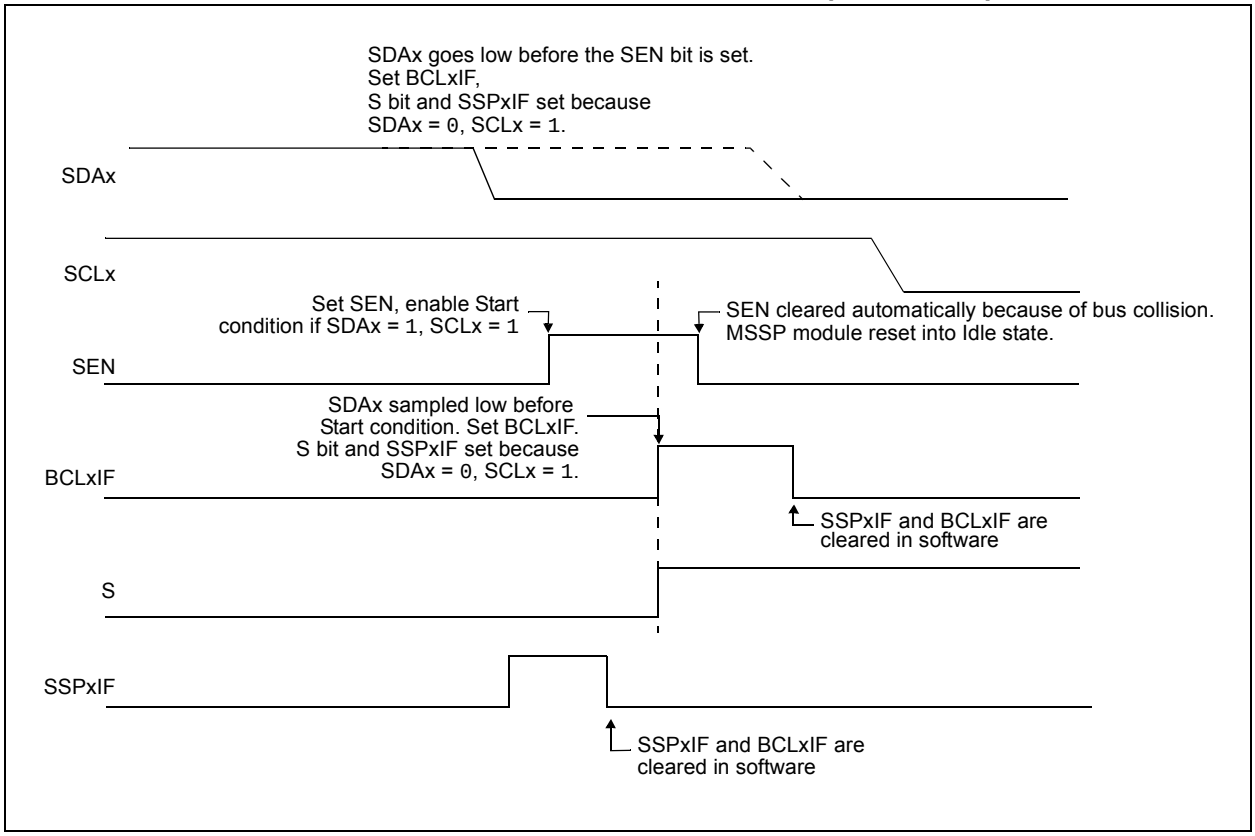


FIGURE 20-29: BUS COLLISION DURING START CONDITION (SCLx = 0)

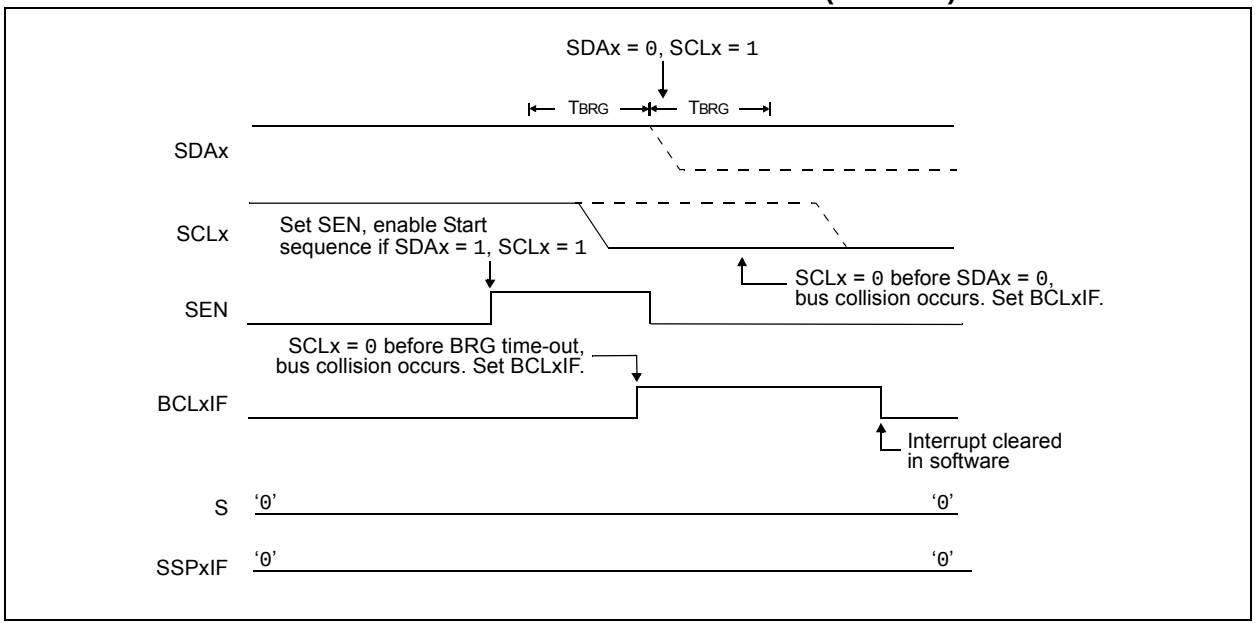
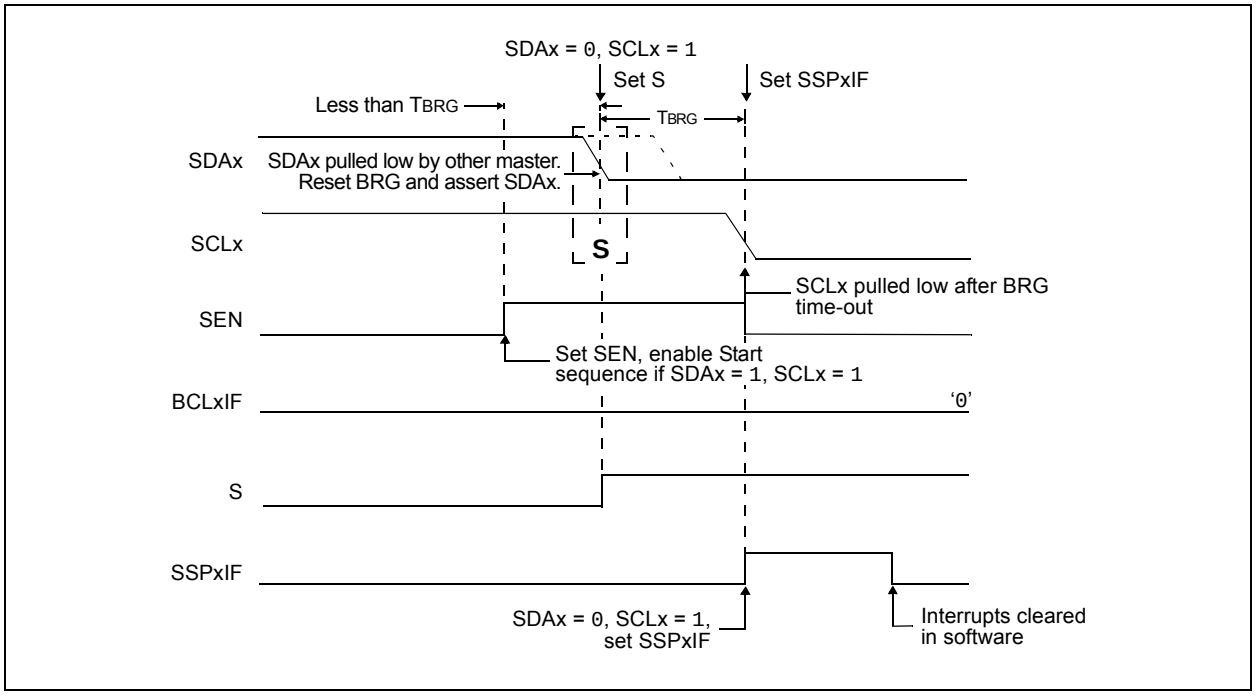


FIGURE 20-30: BRG RESET DUE TO SDAx ARBITRATION DURING START CONDITION



20.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDAx when SCLx goes from low level to high level.

b) SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD<6:0> and counts down to 0. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', see Figure 20-31). If SDAx is sampled high, the BRG is reloaded and begins counting. If SDAx goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDAx at exactly the same time.

If SCLx goes from high-to-low before the BRG times out and SDAx has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 20-32).

If, at the end of the BRG time-out, both SCLx and SDAx are still high, the SDAx pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCLx pin, the SCLx pin is driven low and the Repeated Start condition is complete.

FIGURE 20-31: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

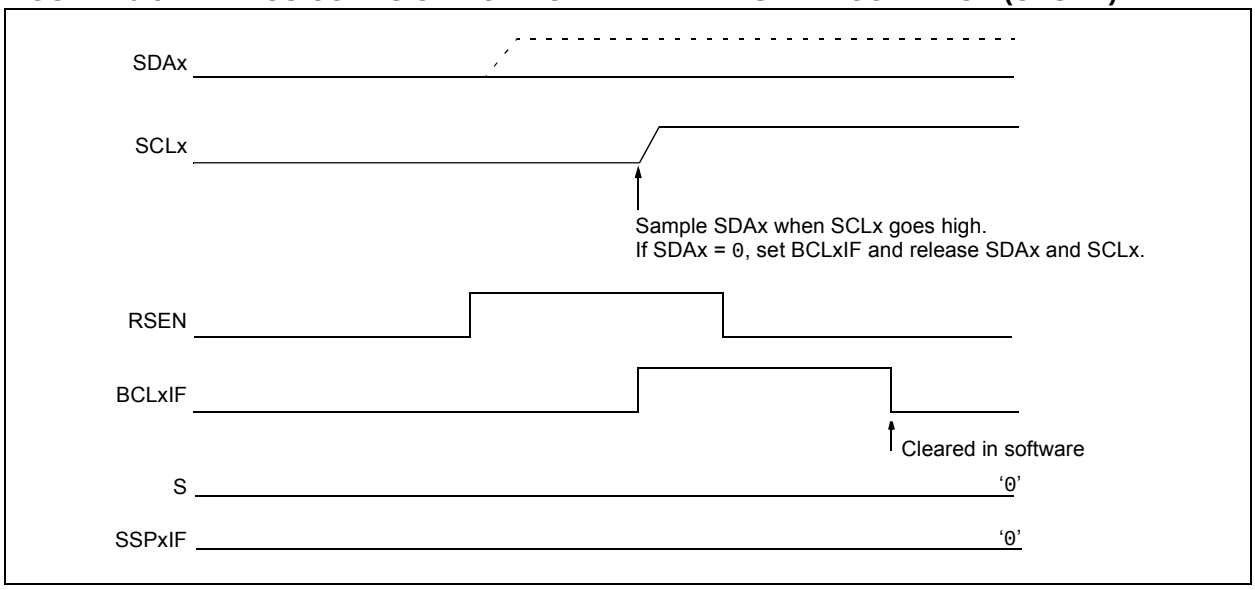
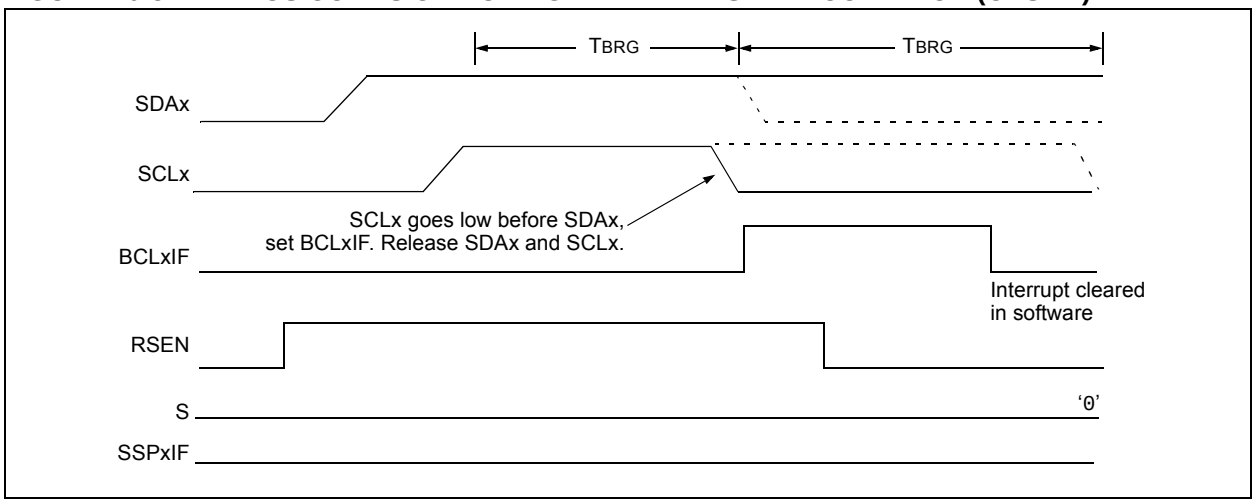


FIGURE 20-32: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



20.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out.

b) After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high.

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD<6:0> and counts down to 0. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 20-33). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 20-34).

FIGURE 20-33: BUS COLLISION DURING A STOP CONDITION (CASE 1)

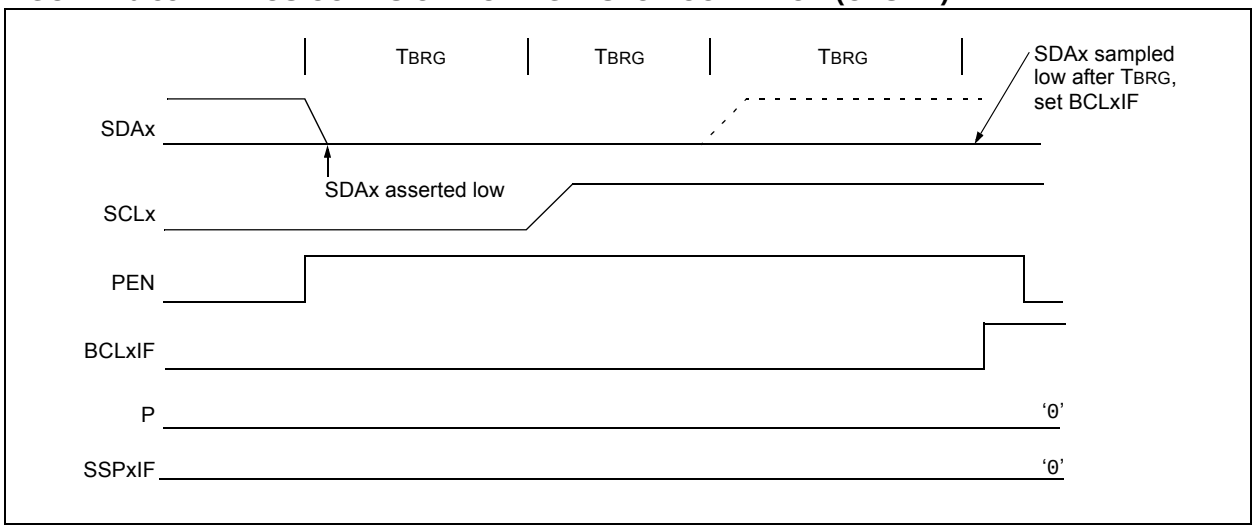


FIGURE 20-34: BUS COLLISION DURING A STOP CONDITION (CASE 2)

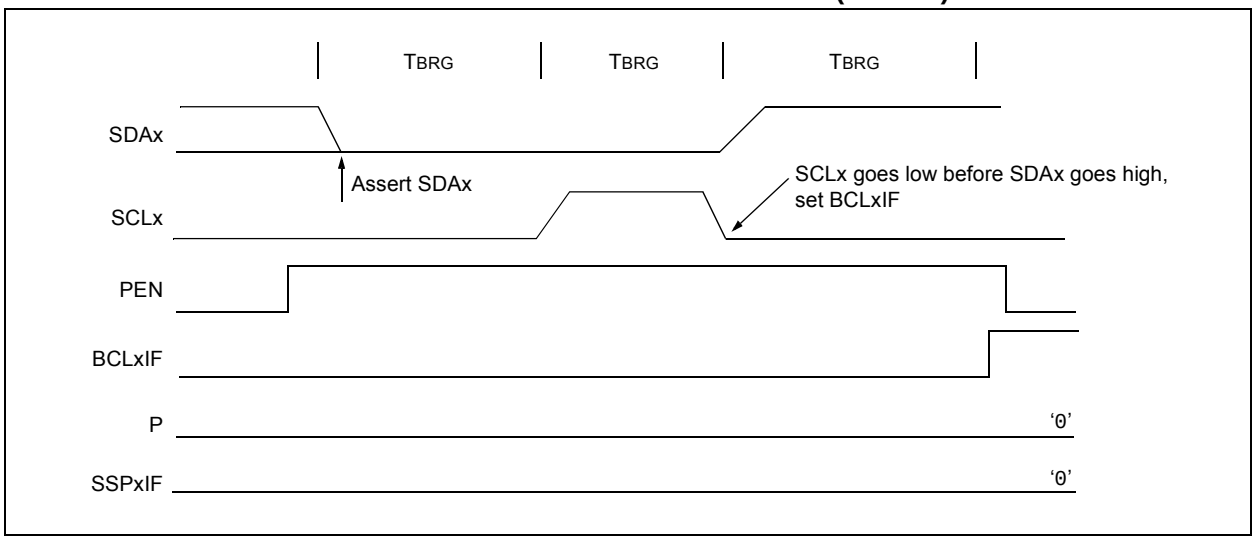


TABLE 20-4: REGISTERS ASSOCIATED WITH I²C™ OPERATION

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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
PIR3	SSP2IF(1)	BCL2IF(1)	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE(1)	BCL2IE(1)	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP(1)	BCL2IP(1)	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	71
TRISD	TRISD7	TRISD6(1)	TRISD5(1)	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	71
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								70
SSP1ADD	MSSP1 Address Register (I²C™ Slave mode), MSSP1 Baud Rate Reload Register (I²C Master mode)								73
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	70
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	70
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN	70
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	70
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								70
SSP2ADD	MSSP2 Address Register (I²C Slave mode), MSSP2 Baud Rate Reload Register (I²C Master mode)								73
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	73
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	73
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN	73
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	73

Legend: — = unimplemented, read as '0', r = reserved. Shaded cells are not used by the MSSP module in I²C™ mode.

Note 1: These bits are only available in 100-pin devices; otherwise, they are unimplemented and read as '0'.

2: Alternate bit definitions in I²C™ Slave mode.

NOTES:

21.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of two serial I/O modules. (Generically, the EUSART is also known as a Serial Communications Interface or SCI.) The EUSART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These features make it ideally suited for use in Local Interconnect Network bus (LIN/J2602 bus) systems.

The 64-pin devices of the PIC18F97J60 family are equipped with one EUSART module, referred to as EUSART1. The 80-pin and 100-pin devices each have two independent EUSART modules, referred to as EUSART1 and EUSART2. They can be configured in the following modes:

- Asynchronous (full-duplex) with:
  - Auto-Wake-up on Character Reception
  - Auto-Baud Calibration
  - 12-Bit Break Character Transmission
- Synchronous – Master (half-duplex) with Selectable Clock Polarity
- Synchronous – Slave (half-duplex) with Selectable Clock Polarity

The pins of EUSART1 and EUSART2 are multiplexed with the functions of PORTC (RC6/TX1/CK1 and RC7/RX1/DT1) and PORTG (RG1/TX2/CK2 and RG2/RX2/DT2), respectively. In order to configure these pins as an EUSART:

- For EUSART1:
  - SPEN bit (RCSTA1<7>) must be set (= 1)
  - TRISC<7> bit must be set (= 1)
  - TRISC<6> bit must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - TRISC<6> bit must be set (= 1) for Synchronous Slave mode
- For EUSART2:
  - SPEN bit (RCSTA2<7>) must be set (= 1)
  - 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 EUSARTx control will automatically reconfigure the pin from input to output as needed.

The operation of each Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTAx)
- Receive Status and Control (RCSTAx)
- Baud Rate Control (BAUDCONx)

These are detailed on the following pages in Register 21-1, Register 21-2 and Register 21-3, respectively.

Note:	Throughout this section, references to register and bit names that may be associated with a specific EUSART module are referred to generically by the use of ‘x’ in place of the specific module number. Thus, “RCSTAx” might refer to the Receive Status register for either EUSART1 or EUSART2.

REGISTER 21-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER x

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:** EUSARTx 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 is 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 21-2: RCSTAx: RECEIVE STATUS AND CONTROL REGISTER x

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 RXx/DTx and TXx/CKx 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
9-Bit Asynchronous mode (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
9-Bit Asynchronous mode (RX9 = 0):
Don't care.

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be updated by reading RCREGx register and receiving next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit
1 = Overrun error (can be cleared by clearing bit, CREN)
0 = No overrun error

bit 0 **RX9D:** 9th bit of Received Data
This can be an address/data bit or a parity bit and must be calculated by user firmware.

REGISTER 21-3: BAUDCONx: BAUD RATE CONTROL REGISTER x

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:
1 = Receive data (RXx) is inverted. Idle state is a low level.
0 = No inversion of receive data (RXx). Idle state is a high level.
Synchronous modes:
1 = Data (DTx) is inverted; Idle state is a low level
0 = No inversion of data (DTx); Idle state is a high level

bit 4 **TXCKP**: Clock and Data Polarity Select bit
Asynchronous mode:
1 = Transmit data (TXx) is inverted; Idle state is a low level
0 = No inversion of transmit data (TXx); Idle state is a high level
Synchronous modes:
1 = Idle state for clock (CKx) is a high level
0 = Idle state for clock (CKx) is a low level

bit 3 **BRG16:** 16-Bit Baud Rate Register Enable bit
1 = 16-bit Baud Rate Generator – SPBRGHx and SPBRGx
0 = 8-bit Baud Rate Generator – SPBRGx only, SPBRGHx value ignored (Compatible mode)

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

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSARTx will continue to sample the RXx pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
0 = RXx pin not monitored or rising edge detected
Synchronous mode:
Unused in this mode.

bit 0 **ABDEN**: Auto-Baud Detect Enable bit
Asynchronous mode:
1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in hardware upon completion.
0 = Baud rate measurement disabled or completed
Synchronous mode:
Unused in this mode.

21.1 Baud Rate Generator (BRG)

The BRG is a dedicated, 8-bit or 16-bit generator that supports both the Asynchronous and Synchronous modes of the EUSARTx. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCONx<3>) selects 16-bit mode.

The SPBRGHx:SPBRGx register pair controls the period of a free-running timer. In Asynchronous mode, bits BRGH (TXSTAx<2>) and BRG16 (BAUDCONx<3>) also control the baud rate. In Synchronous mode, BRGH is ignored. Table 21-1 shows the formula for computation of the baud rate for different EUSARTx modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGHx:SPBRGx registers can be calculated using the formulas in Table 21-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 21-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 21-2. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGHx:SPBRGx registers causes the BRG timer to be reset (or cleared). This ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

21.1.1 OPERATION IN POWER-MANAGED MODES

The device clock is used to generate the desired baud rate. When one of the power-managed modes is entered, the new clock source may be operating at a different frequency. This may require an adjustment to the value in the SPBRGx register pair.

21.1.2 SAMPLING

The data on the RXx pin (either RC7/RX1/DT1 or RG2/RX2/DT2) is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RXx pin.

TABLE 21-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSARTx 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 SPBRGHx:SPBRGx register pair

EQUATION 21-1: 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 ([SPBRGHx:SPBRGx] + 1))
Solving for SPBRGHx:SPBRGx:

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 21-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

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

TABLE 21-3: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRG16 = 0, BRGH = 0											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 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	—	—	—	—	—	—	—	—	—	1.271	5.96	255
2.4	2.543	5.96	255	2.405	0.22	202	2.396	-0.15	162	2.393	-0.27	135
9.6	9.574	-0.27	67	9.574	-0.27	50	9.527	-0.76	40	9.574	-0.27	33
19.2	19.148	-0.27	33	19.531	1.73	24	19.531	1.73	19	19.147	-0.27	16
57.6	59.186	2.75	10	61.035	5.96	7	55.804	-3.12	6	54.253	-5.81	5
115.2	108.508	-5.81	5	122.070	5.96	3	130.208	13.03	2	108.505	-5.81	2

BAUD RATE (K)	SYNC = 0, BRG16 = 0, BRGH = 0								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 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	216
1.2	1.198	-0.08	180	1.206	0.47	80	1.206	0.48	53
2.4	2.411	0.47	89	2.382	-0.76	40	2.411	0.48	26
9.6	9.435	-1.71	22	9.766	1.73	9	9.301	-3.11	6
19.2	19.279	2.75	10	19.531	1.73	4	21.703	13.04	2
57.6	54.254	-5.81	3	48.828	-15.23	1	65.109	13.04	0
115.2	108.508	-5.81	1	97.656	-15.23	0	65.109	-43.48	0

BAUD RATE (K)	SYNC = 0, BRG16 = 0, BRGH = 1											
	Fosc = 41.667 MHz			Fosc = 31.25 MHz			Fosc = 25.000 MHz			Fosc = 20.833 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	—	—	—	—	—	—	—	—	—	—	—	—
9.6	10.172	5.96	255	9.621	0.22	202	9.586	-0.15	162	9.573	-0.27	135
19.2	19.148	-0.27	135	19.148	-0.27	101	19.290	0.47	80	19.147	-0.27	67
57.6	57.871	0.47	44	57.445	-0.27	33	57.870	0.47	26	56.611	-1.72	22
115.2	113.226	-1.71	22	114.890	-0.27	16	111.607	-3.12	13	118.369	2.75	10

BAUD RATE (K)	SYNC = 0, BRG16 = 0, BRGH = 1								
	Fosc = 13.889 MHz			Fosc = 6.250 MHz			Fosc = 4.167 MHz		
	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	—	—	—	—	—	—	1.200	0.01	216
2.4	—	—	—	2.396	-0.15	162	2.389	-0.44	108
9.6	9.645	0.47	89	9.527	-0.76	40	9.645	0.48	26
19.2	19.290	0.47	44	19.531	1.73	19	18.603	-3.11	13
57.6	57.871	0.47	14	55.804	-3.12	6	52.088	-9.57	4
115.2	108.508	-5.81	7	130.208.	13.03	2	130.219	13.04	1

TABLE 21-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRG16 = 1, BRGH = 0											
	FOSC = 41.667 MHz			FOSC = 31.25 MHz			FOSC = 25.000 MHz			FOSC = 20.833 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	8680	0.300	0.00	6509	0.300	0.01	5207	0.300	0.00	4339
1.2	1.200	0.01	2169	1.200	-0.02	1627	1.200	0.01	1301	1.200	0.00	1084
2.4	2.400	0.01	1084	2.399	-0.02	813	2.400	0.01	650	2.398	-0.09	542
9.6	9.609	0.10	270	9.621	0.22	202	9.586	-0.15	162	9.574	-0.27	135
19.2	19.148	-0.27	135	19.148	-0.27	101	19.290	0.47	80	19.148	-0.27	67
57.6	57.871	0.47	44	57.444	-0.27	33	57.870	0.47	26	56.611	-1.72	22
115.2	113.226	-1.71	22	114.890	-0.27	16	111.607	-3.12	13	118.369	2.75	10

BAUD RATE (K)	SYNC = 0, BRG16 = 1, BRGH = 0								
	FOSC = 13.889 MHz			FOSC = 6.250 MHz			FOSC = 4.167 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	-0.02	2893	0.300	0.01	1301	0.300	0.01	867
1.2	1.201	0.05	722	1.198	-0.15	325	1.200	0.01	216
2.4	2.398	-0.08	361	2.396	-0.15	162	2.389	-0.44	108
9.6	9.645	0.47	89	9.527	-0.76	40	9.646	0.48	26
19.2	19.290	0.47	44	19.531	1.73	19	18.603	-3.11	13
57.6	57.871	0.47	14	55.804	-3.12	6	52.088	-9.57	4
115.2	108.508	-5.81	7	130.208	13.03	2	130.218	13.04	1

BAUD RATE (K)	SYNC = 0, BRG16 = 1, BRGH = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 41.667 MHz			FOSC = 31.25 MHz			FOSC = 25.000 MHz			FOSC = 20.833 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	34722	0.300	0.00	26041	0.300	0.00	20832	0.300	0.00	17360
1.2	1.200	0.00	8680	1.200	0.01	6509	1.200	0.01	5207	1.200	0.00	4339
2.4	2.400	0.01	4339	2.400	0.01	3254	2.400	0.01	2603	2.400	0.00	2169
9.6	9.601	0.01	1084	9.598	-0.02	813	9.601	0.01	650	9.592	-0.09	542
19.2	19.184	-0.08	542	19.195	-0.02	406	19.172	-0.15	325	19.219	0.10	270
57.6	57.551	-0.08	180	57.445	-0.27	135	57.339	-0.45	108	57.869	0.47	89
115.2	115.742	0.47	89	114.890	-0.27	67	115.741	0.47	53	115.739	0.47	44

BAUD RATE (K)	SYNC = 0, BRG16 = 1, BRGH = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 13.889 MHz			FOSC = 6.250 MHz			FOSC = 4.167 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	0.00	11573	0.300	0.01	5207	0.300	-0.01	3472
1.2	1.200	-0.02	2893	1.200	0.01	1301	1.200	0.01	867
2.4	2.400	-0.02	1446	2.400	0.01	650	2.400	0.01	433
9.6	9.592	-0.08	361	9.586	-0.15	162	9.557	-0.44	108
19.2	19.184	-0.08	180	19.290	0.47	80	19.292	0.48	53
57.6	57.870	0.47	59	57.870	0.47	26	57.875	0.48	17
115.2	115.742	0.47	29	111.607	-3.12	13	115.750	0.48	8

21.1.3 AUTO-BAUD RATE DETECT

The Enhanced USARTx 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 21-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RXx signal, the RXx signal is timing the BRG. In ABD mode, the internal 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 SPBRGx begins counting up, using the preselected clock source on the first rising edge of RXx. After eight bits on the RXx pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGHx:SPBRGx register pair. Once the 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 (BAUDCONx<7>). It is set in hardware by BRG rollovers and can be set or cleared by the user in software. ABD mode remains active after rollover events and the ABDEN bit remains set (Figure 21-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. Note that the BRG clock will be configured by the BRG16 and BRGH bits. Independent of the BRG16 bit setting, both the SPBRGx and SPBRGHx will be used as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGHx register. Refer to Table 21-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSARTx state machine is held in Idle. The RCxIF interrupt is set once the fifth rising edge on RXx is detected. The value in the RCREGx needs to be read to clear the RCxIF interrupt. The contents of RCREGx should be discarded.

Note 1:	If the WUE bit is set with the ABDEN bit, Auto-Baud Rate Detection will occur on the byte *following* the Break character.
2:	It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSARTx 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 21-4: BRG COUNTER CLOCK RATES

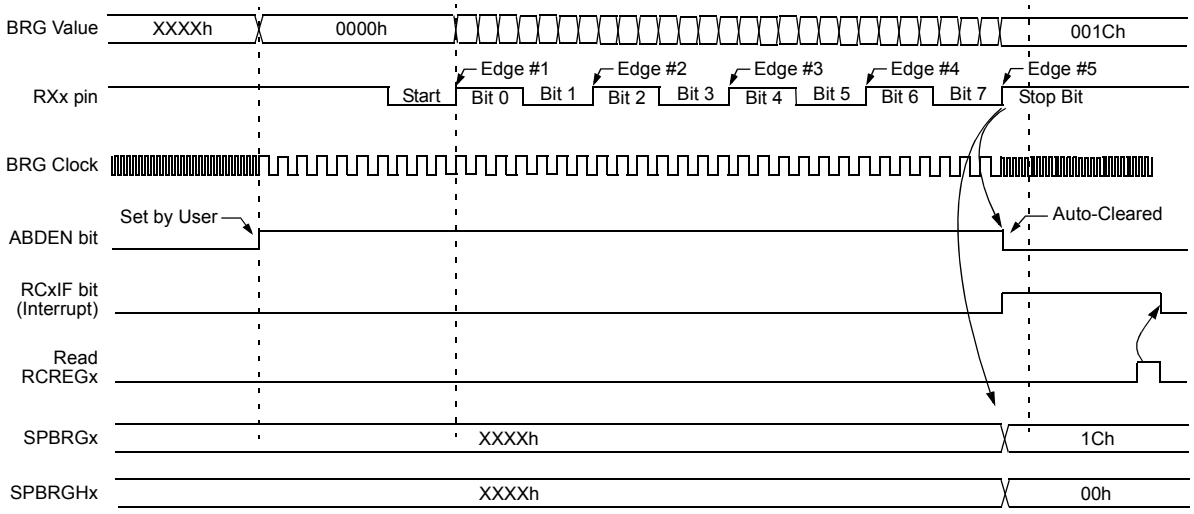
BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

Note: During the ABD sequence, SPBRGx and SPBRGHx are both used as a 16-bit counter, independent of the BRG16 setting.

21.1.3.1 ABD and EUSARTx Transmission

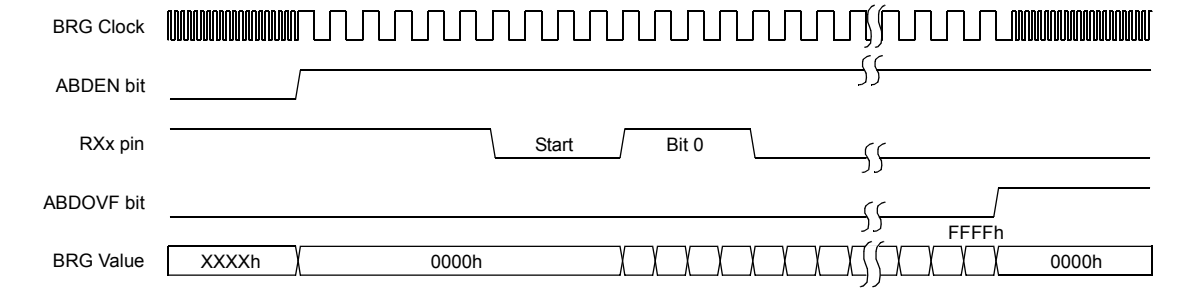
Since the BRG clock is reversed during ABD acquisition, the EUSARTx transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREGx cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSARTx operation.

FIGURE 21-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the EUSARTx module to be configured in Asynchronous mode and WUE = 0.

FIGURE 21-2: BRG OVERFLOW SEQUENCE



21.2 EUSARTx Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTAx<4>). In this mode, the EUSARTx 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 EUSARTx transmits and receives the LSb first. The EUSARTx module'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 (TXSTAx<2> and BAUDCONx<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

The TXCKP (BAUDCONx<4>) and RXDTP (BAUDCONx<5>) bits allow the TXx and RXx signals to be inverted (polarity reversed). Devices that buffer signals between TTL and RS-232 levels also invert the signal. Setting the TXCKP and RXDTP bits allows for the use of circuits that provide buffering without inverting the signal.

When operating in Asynchronous mode, the EUSARTx 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

21.2.1 EUSARTx ASYNCHRONOUS TRANSMITTER

The EUSARTx transmitter block diagram is shown in Figure 21-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREGx. The TXREGx register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREGx register (if available).

Once the TXREGx register transfers the data to the TSR register (occurs in one TCY), the TXREGx register is empty and the TXxIF flag bit is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF will be set regardless of the state of TXxIE; it cannot be cleared in software. TXxIF is also not cleared immediately upon loading TXREGx, but becomes valid in the second instruction cycle following the load instruction. Polling TXxIF, immediately following a load of TXREGx, will return invalid results.

While TXxIF indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<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, TXxIF, is set when enable bit TXEN is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting bit, SPEN.
3. If the signal from the TXx pin is to be inverted, set the TXCKP bit.
4. If interrupts are desired, set enable bit, TXxIE.
5. If 9-bit transmission is desired, set transmit bit, TX9. Can be used as address/data bit.
6. Enable the transmission by setting the TXEN bit which will also set bit, TXxIF.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
8. Load data to the TXREGx register (starts transmission).
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-3: EUSARTx TRANSMIT BLOCK DIAGRAM

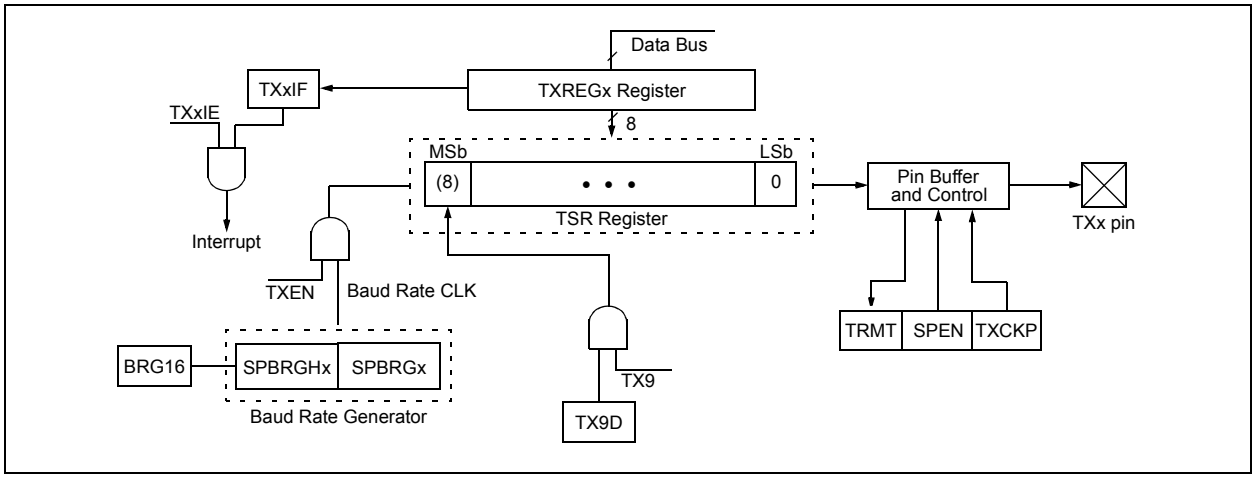


FIGURE 21-4: ASYNCHRONOUS TRANSMISSION, TXCKP = 0 (TXx NOT INVERTED)

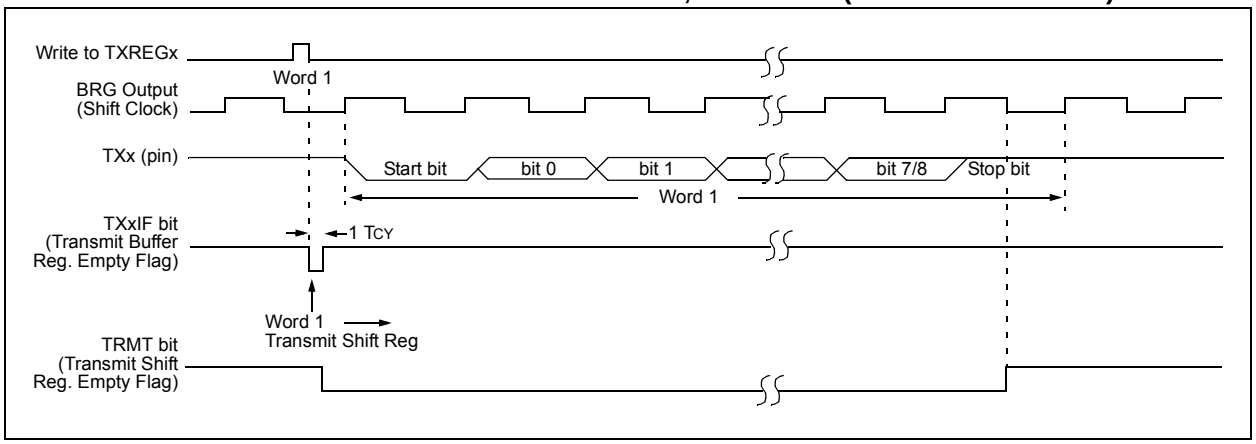
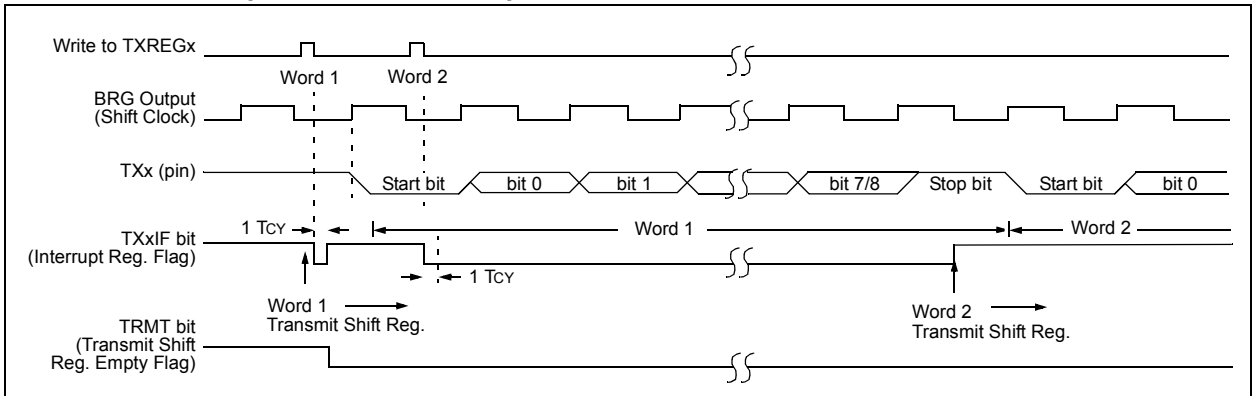


FIGURE 21-5: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK), TXCKP = 0 (TXx NOT INVERTED)



Note: This timing diagram shows two consecutive transmissions.

TABLE 21-5: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
TXREGx	EUSARTx Transmit Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as '0'.

21.2.2 EUSARTx ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 21-6. The data is received on the RXx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

The RXDTP bit (BAUDCON<5>) allows the RXx signal to be inverted (polarity reversed). Devices that buffer signals from RS-232 to TTL levels also perform an inversion of the signal (when RS-232 = positive, TTL = 0). Inverting the polarity of the RXx pin data by setting the RXDTP bit allows for the use of circuits that provide buffering without inverting the signal.

To set up an Asynchronous Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting bit, SPEN.
3. If the signal at the RXx pin is to be inverted, set the RXDTP bit.
4. If interrupts are desired, set enable bit, RCxIE.
5. If 9-bit reception is desired, set bit, RX9.
6. Enable the reception by setting bit, CREN.
7. Flag bit, RCxIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCxIE, was set.
8. Read the RCSTAx 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 RCREGx register.
10. If any error occurred, clear the error by clearing enable bit, CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

21.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If the signal at the RXx pin is to be inverted, set the RXDTP bit. If the signal from the TXx pin is to be inverted, set the TXCKP bit.
4. If interrupts are required, set the RCEN bit and select the desired priority level with the RCxIP bit.
5. Set the RX9 bit to enable 9-bit reception.
6. Set the ADDEN bit to enable address detect.
7. Enable reception by setting the CREN bit.
8. The RCxIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCxIE and GIE bits are set.
9. Read the RCSTAx register to determine if any error occurred during reception, as well as read Bit 9 of data (if applicable).
10. Read RCREGx to determine if the device is being addressed.
11. If any error occurred, clear the CREN bit.
12. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 21-6: EUSARTx RECEIVE BLOCK DIAGRAM

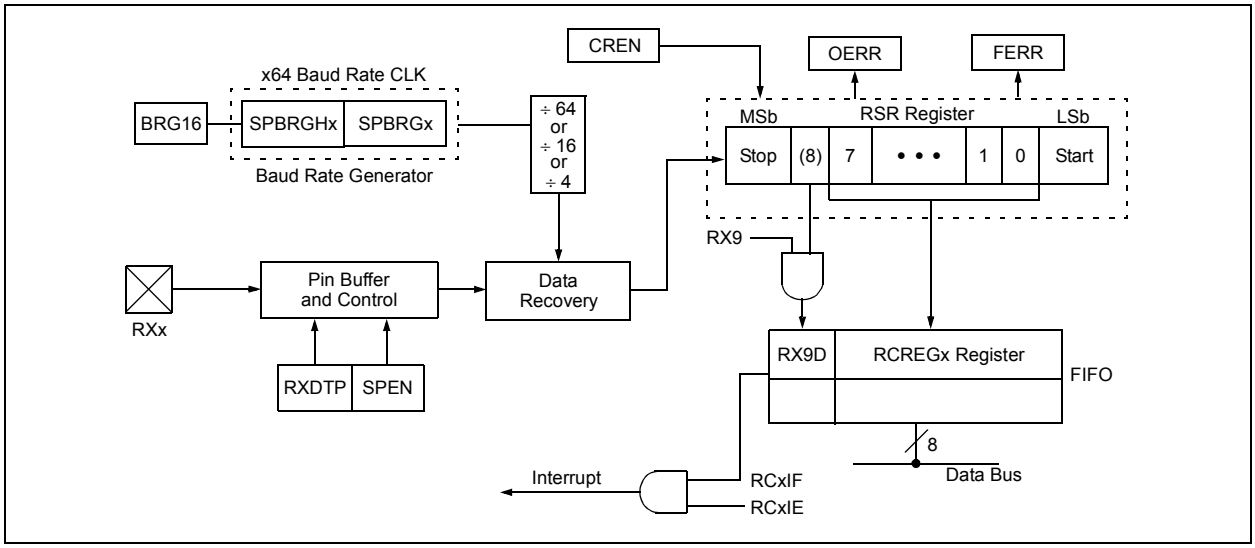
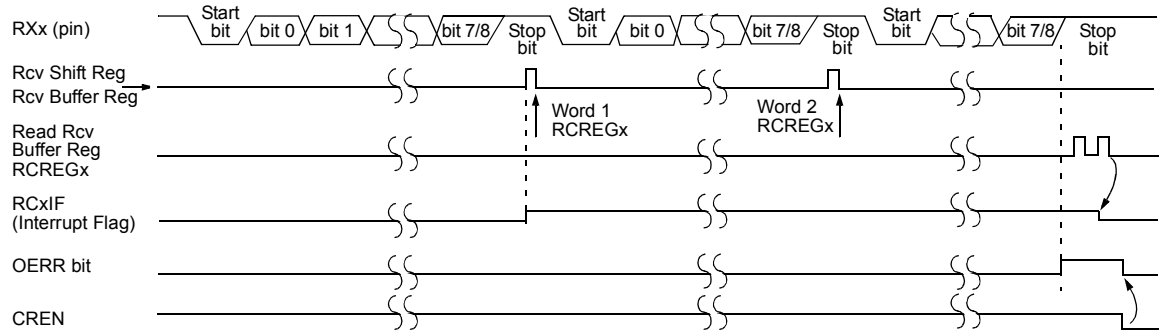


FIGURE 21-7: ASYNCHRONOUS RECEPTION, RXDTP = 0 (RXx NOT INVERTED)



Note: This timing diagram shows three words appearing on the RXx input. The RCREGx (Receive Buffer) is read after the third word, causing the OERR (Overrun Error) bit to be set.

TABLE 21-6: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
RCREGx	EUSARTx Receive Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as '0'.

21.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSARTx 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 RXx/DTx line while the EUSARTx is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCONx<1>). Once set, the typical receive sequence on RXx/DTx is disabled and the EUSARTx remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RXx/DTx line. (This coincides with the start of a Sync Break or a Wake-up Signal character for the LIN/J2602 protocol.)

Following a wake-up event, the module generates an RCxIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 21-8) and asynchronously if the device is in Sleep mode (Figure 21-9). The interrupt condition is cleared by reading the RCREGx register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RXx line following the wake-up event. At this point, the EUSARTx module is in Idle mode and returns to normal operation. This signals to the user that the Sync Break event is over.

21.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RXx/DTx, information with any state changes before the Stop bit may signal a false End-of-Character (EOC) and cause data or framing errors. To work properly, therefore, the initial character in the transmission must be all '0's. This can be 00h (8 bytes) for standard RS-232 devices or 000h (12 bits) for LIN/J2602 bus.

Oscillator start-up time must also be considered, especially in applications using oscillators with longer start-up intervals (i.e., HS or HSPLL mode). The Sync Break (or Wake-up Signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the EUSARTx.

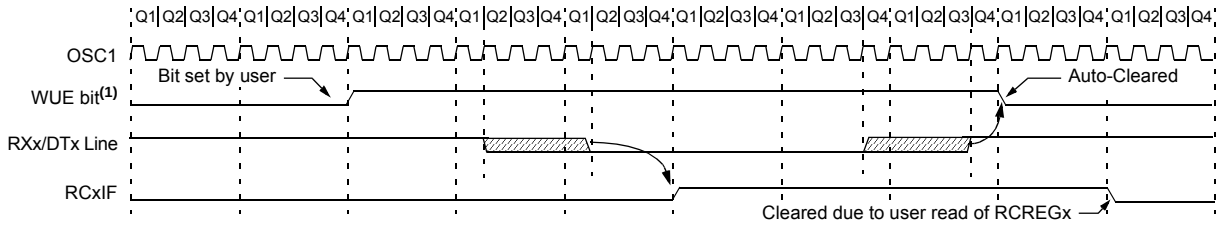
21.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCxIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the EUSARTx in an Idle mode. The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared after this when a rising edge is seen on RXx/DTx. The interrupt condition is then cleared by reading the RCREGx register. Ordinarily, the data in RCREGx will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set), and the RCxIF flag is set, should not be used as an indicator of the integrity of the data in RCREGx. Users should consider implementing a parallel method in firmware to verify received data integrity.

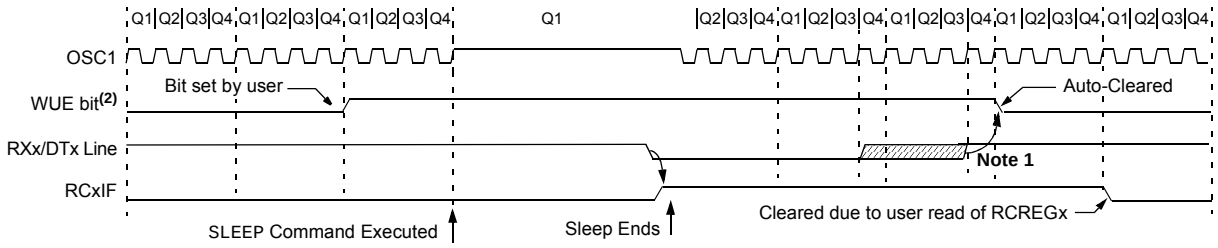
To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 21-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION



Note 1: The EUSARTx remains in Idle while the WUE bit is set.

FIGURE 21-9: 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 before the oscillator is ready. This sequence should not depend on the presence of Q clocks.

2: The EUSARTx remains in Idle while the WUE bit is set.

21.2.5 BREAK CHARACTER SEQUENCE

The EUSARTx 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 (TXSTAx<3> and TXSTAx<5>) are set while the Transmit Shift Register (TSR) is loaded with data. Note that the value of data written to TXREGx will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN/J2602 support specification).

Note that the data value written to the TXREGx for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 21-10 for the timing of the Break character sequence.

21.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN/J2602 bus master.

1. Configure the EUSARTx for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.
3. Load the TXREGx with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREGx to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREGx becomes empty, as indicated by the TXxIF, the next data byte can be written to TXREGx.

21.2.6 RECEIVING A BREAK CHARACTER

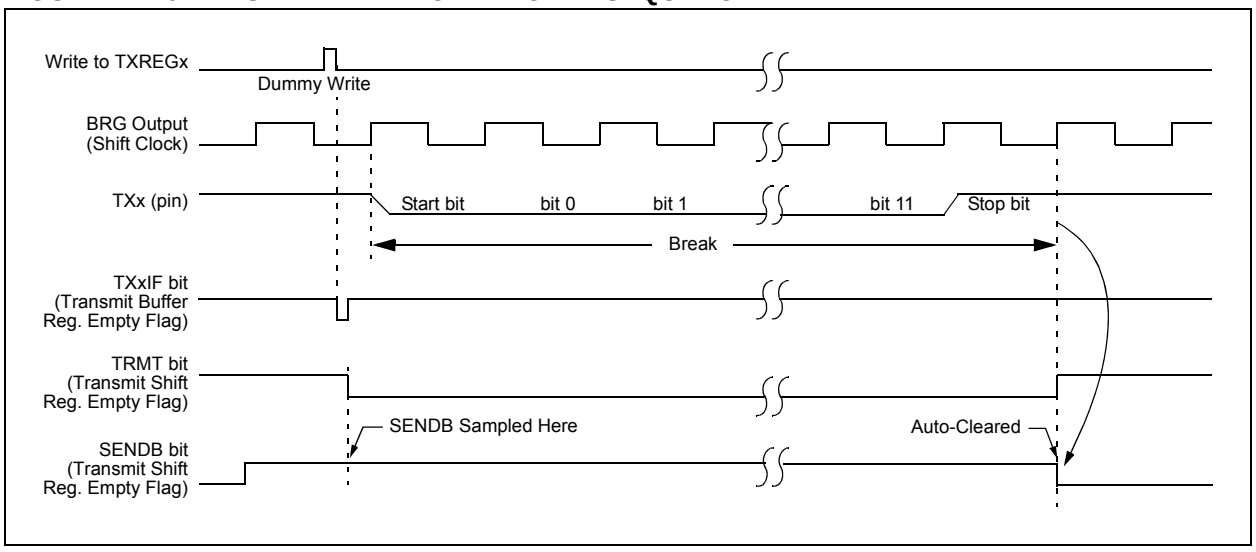
The Enhanced USARTx module can receive a Break character in two ways.

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 21.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSARTx will sample the next two transitions on RXx/DTx, cause an RCxIF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABDEN bit once the TXxIF interrupt is observed.

FIGURE 21-10: SEND BREAK CHARACTER SEQUENCE



21.3 EUSARTx Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTAx<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit, SYNC (TXSTAx<4>). In addition, enable bit, SPEN (RCSTAx<7>), is set in order to configure the TXx and RXx pins to CKx (clock) and DTx (data) lines, respectively.

Clock polarity (CKx) is selected with the TXCKP bit (BAUDCON<4>). Setting TXCKP sets the Idle state on CKx as high, while clearing the bit sets the Idle state as low. Data polarity (DTx) is selected with the RXDTP bit (BAUDCONx<5>). Setting RXDTP sets the Idle state on DTx as high, while clearing the bit sets the Idle state as low. DTx is sampled when CKx returns to its Idle state. This option is provided to support Microwire devices with this module.

21.3.1 EUSARTx SYNCHRONOUS MASTER TRANSMISSION

The EUSARTx transmitter block diagram is shown in Figure 21-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Transmit Shift register obtains its data from the Read/Write Transmit Buffer register, TXREGx. The TXREGx register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREGx (if available).

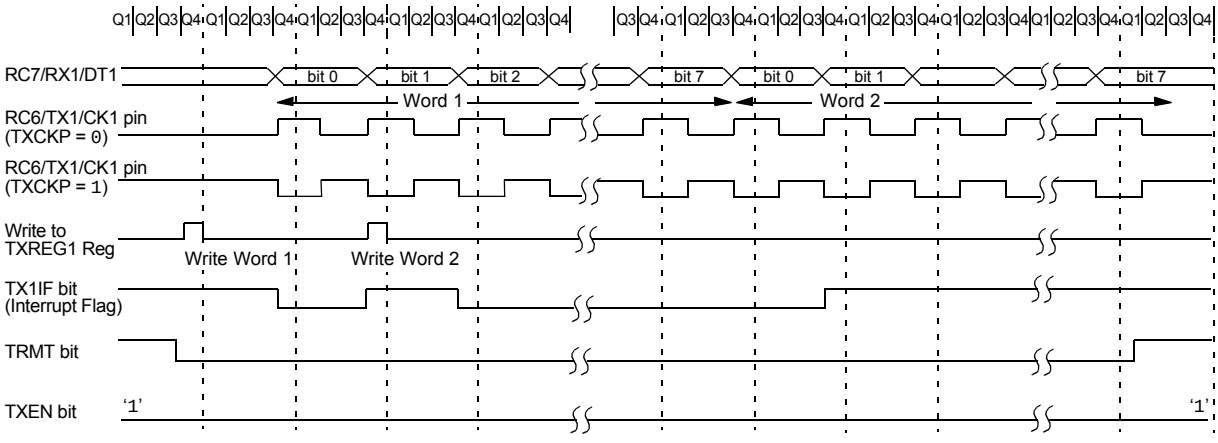
Once the TXREGx register transfers the data to the TSR register (occurs in one TCY), the TXREGx is empty and the TXxIF flag bit is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF is set regardless of the state of enable bit, TXxIE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREGx register.

While flag bit, TXxIF, indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user must poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

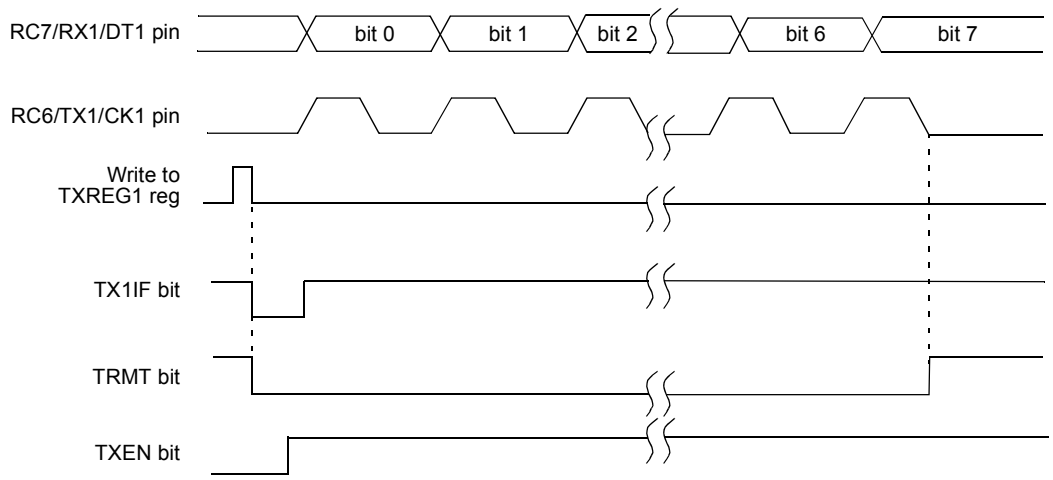
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission 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 TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-11: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG1 = 0; continuous transmission of two 8-bit words. This example is equally applicable to EUSART2 (RG1/TX2/CK2 and RG2/RX2/DT2).

FIGURE 21-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



Note: This example is equally applicable to EUSART2 (RG1/TX2/CK2 and RG2/RX2/DT2).

TABLE 21-7: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
TXREGx	EUSARTx Transmit Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as '0'.

21.3.2 EUSARTx SYNCHRONOUS MASTER RECEPTION

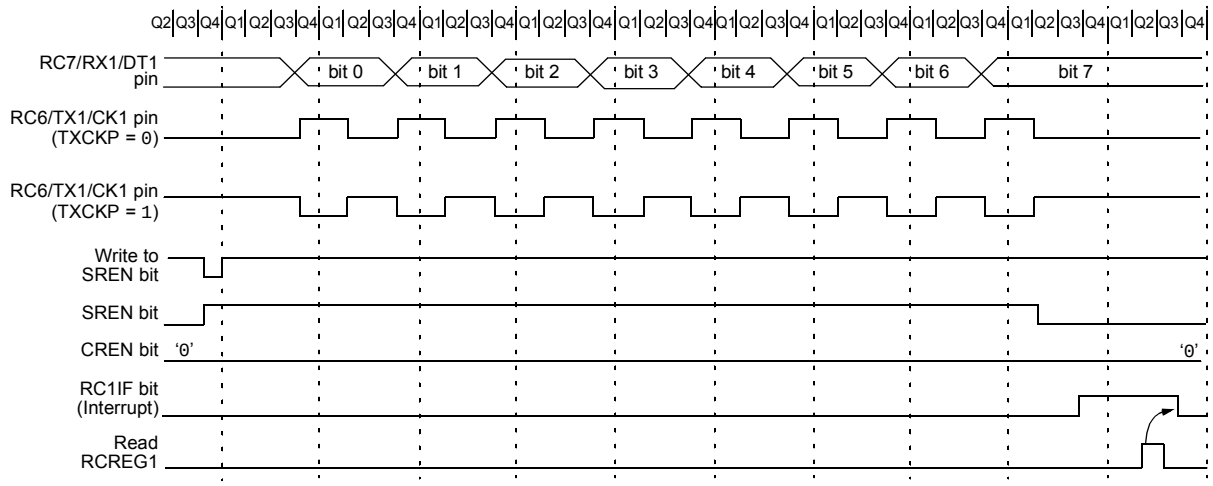
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTAx<5>), or the Continuous Receive Enable bit, CREN (RCSTAx<4>). Data is sampled on the RXx pin on the falling edge of the clock.

If enable bit, SREN, is set, only a single word is received. If enable bit, CREN, is set, the reception is continuous until CREN is cleared. If both bits are set, then CREN takes precedence.

To set up a Synchronous Master Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. Ensure bits, CREN and SREN, are clear.
4. If the signal from the CKx pin is to be inverted, set the TXCKP bit. If the signal from the DTx pin is to be inverted, set the RXDTP bit.
5. If interrupts are desired, set enable bit, RCxIE.
6. If 9-bit reception is desired, set bit, RX9.
7. If a single reception is required, set bit, SREN. For continuous reception, set bit, CREN.
8. Interrupt flag bit, RCxIF, will be set when reception is complete and an interrupt will be generated if the enable bit, RCxIE, was set.
9. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
10. Read the 8-bit received data by reading the RCREGx register.
11. If any error occurred, clear the error by clearing bit, CREN.
12. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0. This example is equally applicable to EUSART2 (RG1/TX2/CK2 and RG2/RX2/DT2).

TABLE 21-8: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
RCREGx	EUSARTx Receive Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as '0'.

21.4 EUSARTx Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

21.4.1 EUSARTx SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep mode.

If two words are written to the TXREGx and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREGx register.
c) Flag bit, TXxIF, will not be set.
d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit, TXxIF, will now be set.
e) If enable bit, TXxIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. Clear bits, CREN and SREN.
3. If the signal from the CKx pin is to be inverted, set the TXCKP bit. If the signal from the DTx pin is to be inverted, set the RXDTP bit.
4. If interrupts are desired, set enable bit, TXxIE.
5. If 9-bit transmission is desired, set bit, TX9.
6. Enable the transmission by setting enable bit, TXEN.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
8. Start transmission by loading data to the TXREGx register.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 21-9: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF(1)	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE(1)	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP(1)	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
TXREGx	EUSARTx Transmit Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as '0'.

21.4.2 EUSARTx SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep or any Idle mode, and bit, SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this low-power mode. Once the word is received, the RSR register will transfer the data to the RCREGx register. If the RCxIE enable bit is set, the interrupt generated will wake the chip from the low-power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. If interrupts are desired, set enable bit, RCxIE.
3. If the signal from the CKx pin is to be inverted, set the TXCKP bit. If the signal from the DTx pin is to be inverted, set the RXDTP bit.
4. If 9-bit reception is desired, set bit, RX9.
5. To enable reception, set enable bit, CREN.
6. Flag bit, RCxIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCxIE, was set.
7. Read the RCSTAx 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 RCREGx register.
9. If any error occurred, clear the error by clearing bit, CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 21-10: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR3	SSP2IF	BCL2IF	RC2IF(1)	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	71
PIE3	SSP2IE	BCL2IE	RC2IE(1)	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	71
IPR3	SSP2IP	BCL2IP	RC2IP(1)	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	71
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	71
RCREGx	EUSARTx Receive Register								71
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	71
BAUDCONx	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	72
SPBRGHx	EUSARTx Baud Rate Generator Register High Byte								72
SPBRGx	EUSARTx Baud Rate Generator Register Low Byte								72

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

Note 1: These bits are only available in 80-pin and 100-pin devices; otherwise, they are unimplemented and read as ‘0’.

22.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has 11 inputs for the 64-pin devices, 15 inputs for the 80-pin devices and 16 inputs for the 100-pin devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has five registers:

- A/D Result 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 22-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 22-2, configures the functions of the port pins. The ADCON2 register, shown in Register 22-3, configures the A/D clock source, programmed acquisition time and justification.

REGISTER 22-1: 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 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 0 (AN0)
0001 = Channel 1 (AN1)
0010 = Channel 2 (AN2)
0011 = Channel 3 (AN3)
0100 = Channel 4 (AN4)
0101 = Channel 5 (AN5)(1,3)
0110 = Channel 6 (AN6)
0111 = Channel 7 (AN7)
1000 = Channel 8 (AN8)
1001 = Channel 9 (AN9)
1010 = Channel 10 (AN10)
1011 = Channel 11 (AN11)
1100 = Channel 12 (AN12)(2,3)
1101 = Channel 13 (AN13)(2,3)
1110 = Channel 14 (AN14)(2,3)
1111 = Channel 15 (AN15)(2,3)

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

Note 1: This channel is implemented on 100-pin devices only.
2: These channels are implemented on 80-pin and 100-pin devices only.
3: Performing a conversion on unimplemented channels will return random values.

REGISTER 22-2: 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>	AN15(1)	AN14(1)	AN13(1)	AN12(1)	AN11	AN10	AN9	AN8	AN7	AN6	AN5(2)	AN4	AN3	AN2	AN1(3)	AN0(3)
0000	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input D = Digital I/O

Note 1: AN12 through AN15 are available in 80-pin and 100-pin devices only.
2: AN5 is available in 100-pin devices only.
3: AN0 and AN1 can also operate as Ethernet LED outputs in either Analog or Digital I/O modes.

REGISTER 22-3: 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 Converter's internal RC oscillator.

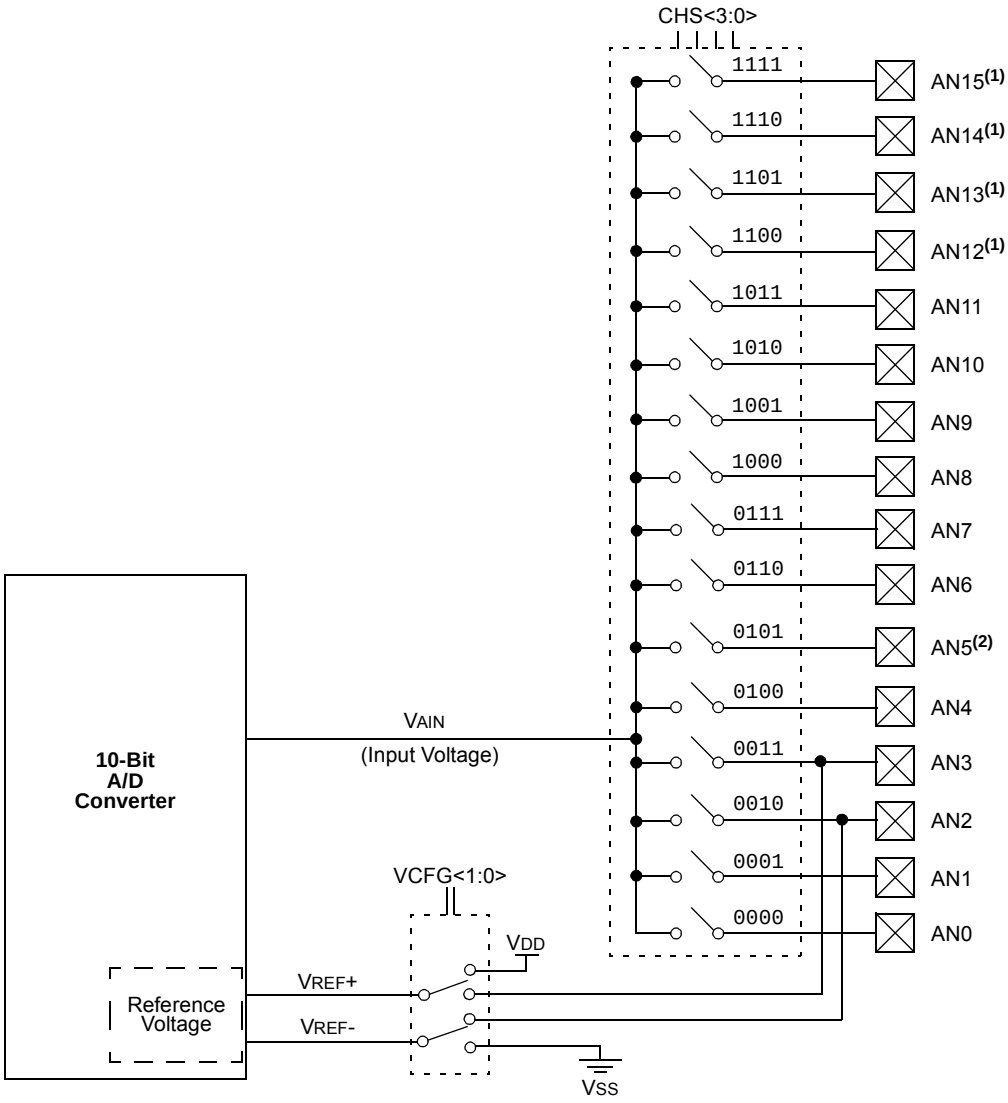
The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

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/$\overline{DONE}$ 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 22-1.

FIGURE 22-1: A/D BLOCK DIAGRAM



Note 1: Channels AN15 through AN12 are not available in 64-pin devices.
2: Channel AN5 is implemented in 100-pin devices only.

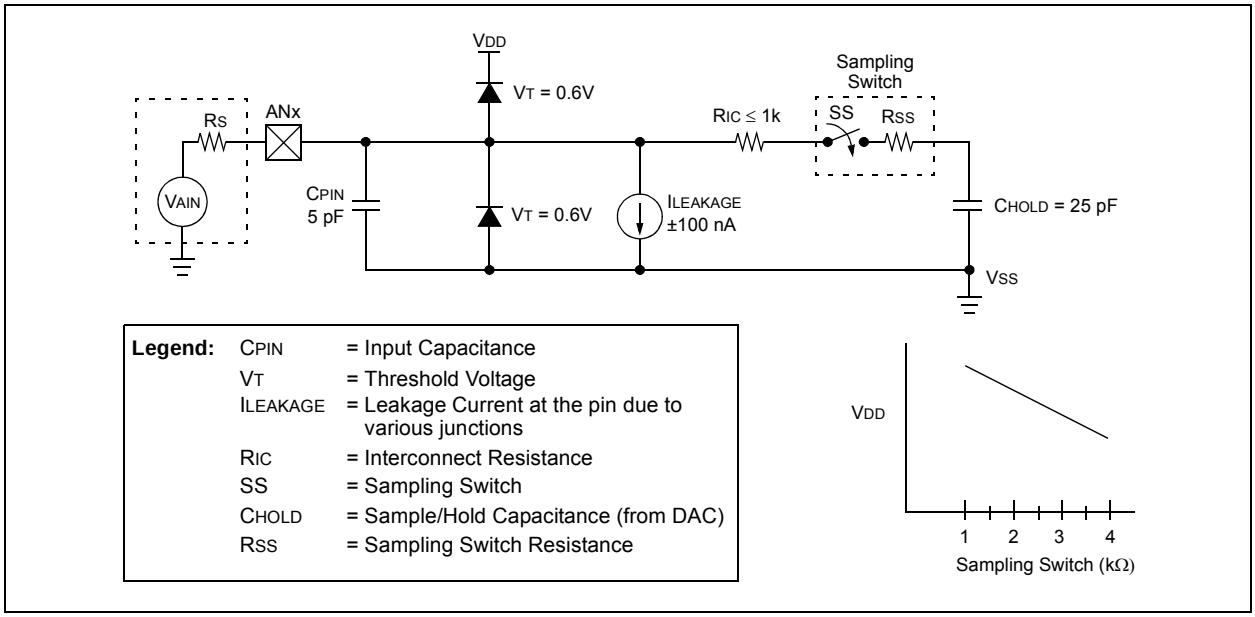
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs. To determine acquisition time, see **Section 22.1 "A/D Acquisition Requirements"**. After this acquisition time has elapsed, the A/D conversion can be started. An acquisition time can be programmed to occur between setting the GO/$\overline{\text{DONE}}$ bit and the actual start of the conversion.

The following steps should be followed to do an A/D conversion:

1. Configure the A/D module:
   - Configure analog pins, voltage reference and digital I/O (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D acquisition time (ADCON2)
   - Select A/D conversion clock (ADCON2)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set GIE bit
3. Wait the required acquisition time (if required).
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0<1>)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR
   - Waiting for the A/D interrupt
6. Read A/D Result registers (ADRESH:ADRESL); clear bit, ADIF, if required.
7. For 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 next acquisition starts.

FIGURE 22-2: ANALOG INPUT MODEL



22.1 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 22-2. 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 22-1 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 22-3 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 22-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 22-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

or

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)$$

EQUATION 22-3: 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.

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)\ \mu s$$
$$-(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1\ \mu s + 1.2\ \mu s$$
$$2.4\ \mu s$$

22.2 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.

22.3 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 **Section 28.0 "Electrical Characteristics"**, A/D Parameter 130 in Table 28-27 for more information.

Table 22-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 22-1: 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	41.67 MHz
64 TOSC	110	41.67 MHz
RC(2)	x11	1.00 MHz(1)

Note 1: The RC source has a typical TAD time of 4 ms.

2: See Parameter 130 in Table 28-27 for A/D RC clock specifications.

22.4 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.

22.5 A/D Conversions

Figure 22-3 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 22-4 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set, the ACQT<2:0> bits are 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 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.

22.6 Use of the ECCP2 Trigger

An A/D conversion can be started by the "Special Event Trigger" of the ECCP2 module. This requires that the 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 22-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)

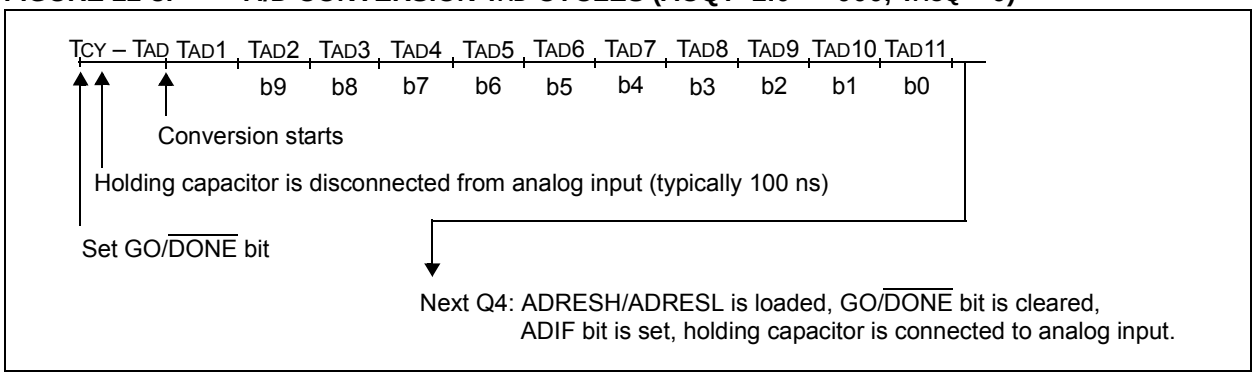
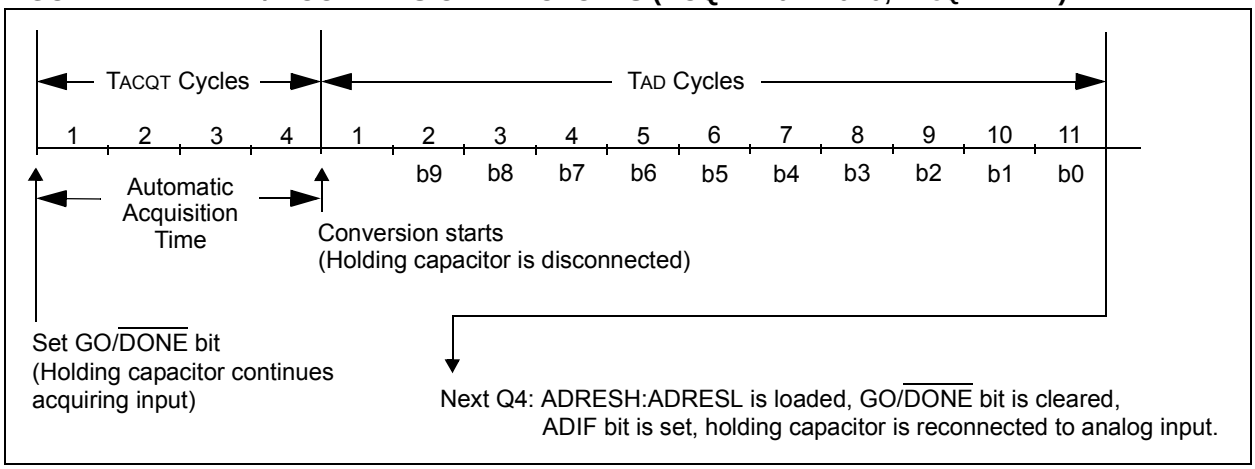


FIGURE 22-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)



22.7 A/D Converter Calibration

The A/D Converter in the PIC18F97J60 family of devices includes a self-calibration feature which compensates for any offset generated within the module. The calibration process is automated and is initiated by setting the ADCAL bit (ADCON0<7>). The next time the GO/DONE bit is set, the module will perform a "dummy" conversion (that is, with reading none of the input channels) and store the resulting value internally to compensate for offset. Thus, subsequent offsets will be compensated.

The calibration process assumes that the device is in a relatively steady-state operating condition. If A/D calibration is used, it should be performed after each device Reset, or if there are other major changes in operating conditions.

22.8 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 Sleep mode requires the A/D RC clock to be selected. If bits, ACQT<2:0>, are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS bits in the OSCCON register must have already been cleared prior to starting the conversion.

TABLE 22-2: 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	69
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	71
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	71
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	71
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
ADRESH	A/D Result Register High Byte								70
ADRESL	A/D Result Register Low Byte								70
ADCON0	ADCAL	—	CHS3	CHS3	CHS1	CHS0	GO/DONE	ADON	70
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	70
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	70
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	70
PORTA	RJPU	—	RA5	RA4	RA3	RA2	RA1	RA0	72
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	71
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0(1)	72
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0(1)	71
PORTH(2)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	72
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	71

Legend: — = unimplemented, read as '0', r = reserved. Shaded cells are not used for A/D conversion.

Note 1: Implemented in 100-pin devices only.

2: This register is not implemented in 64-pin devices.

NOTES:

23.0 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 24.0 "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 23-1) selects the comparator input and output configuration. Block diagrams of the various comparator configurations are shown in Figure 23-1.

REGISTER 23-1: CMCON: COMPARATOR CONTROL REGISTER

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

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

bit 7 **C2OUT**: Comparator 2 Output bit
When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
When C2INV = 1:
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT**: Comparator 1 Output bit
When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
When C1INV = 1:
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5 **C2INV**: Comparator 2 Output Inversion bit
1 = C2 output 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 RA4/AN9

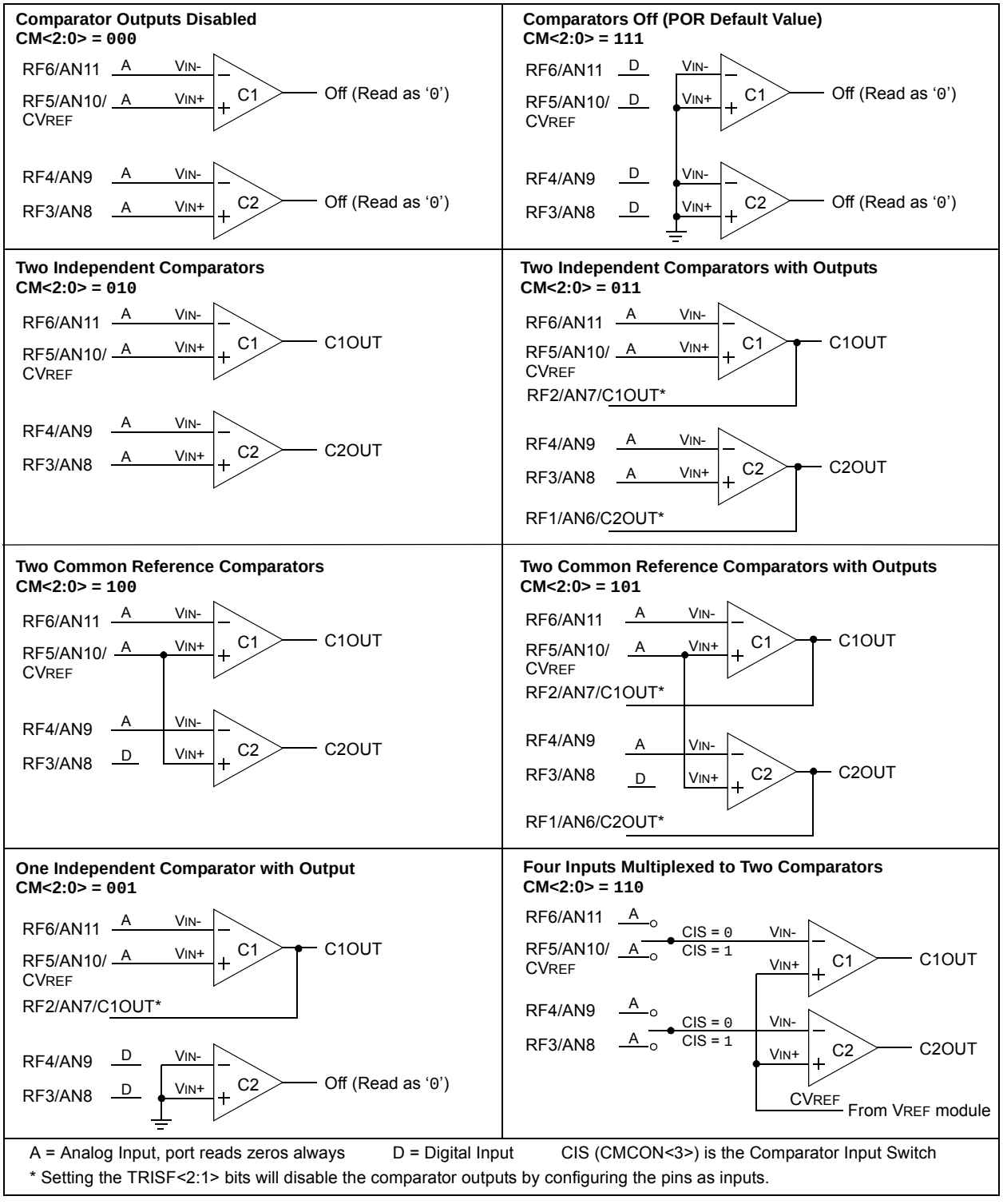
bit 2-0 **CM<2:0>**: Comparator Mode bits
Figure 23-1 shows the Comparator modes and the CM<2:0> bit settings.

23.1 Comparator Configuration

There are eight modes of operation for the comparators, shown in Figure 23-1. 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 28.0 "Electrical Characteristics"**.

Note: Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur.

FIGURE 23-1: 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.

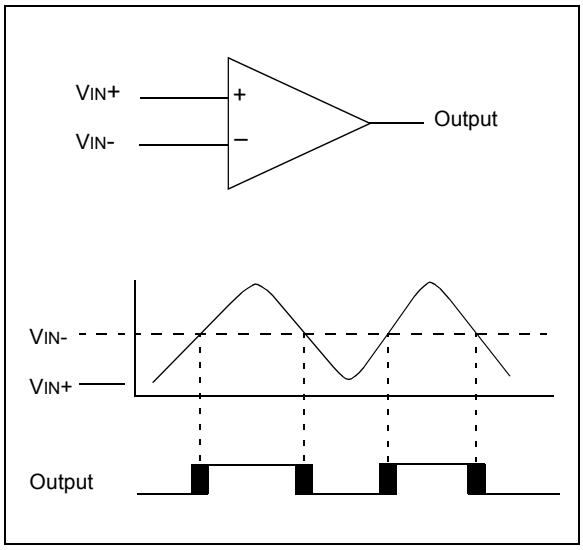
23.2 Comparator Operation

A single comparator is shown in Figure 23-2, 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 23-2 represent the uncertainty due to input offsets and response time.

23.3 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 23-2).

FIGURE 23-2: SINGLE COMPARATOR



23.3.1 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).

23.3.2 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 24.0 "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.

23.4 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 28.0 "Electrical Characteristics"**).

23.5 Comparator Outputs

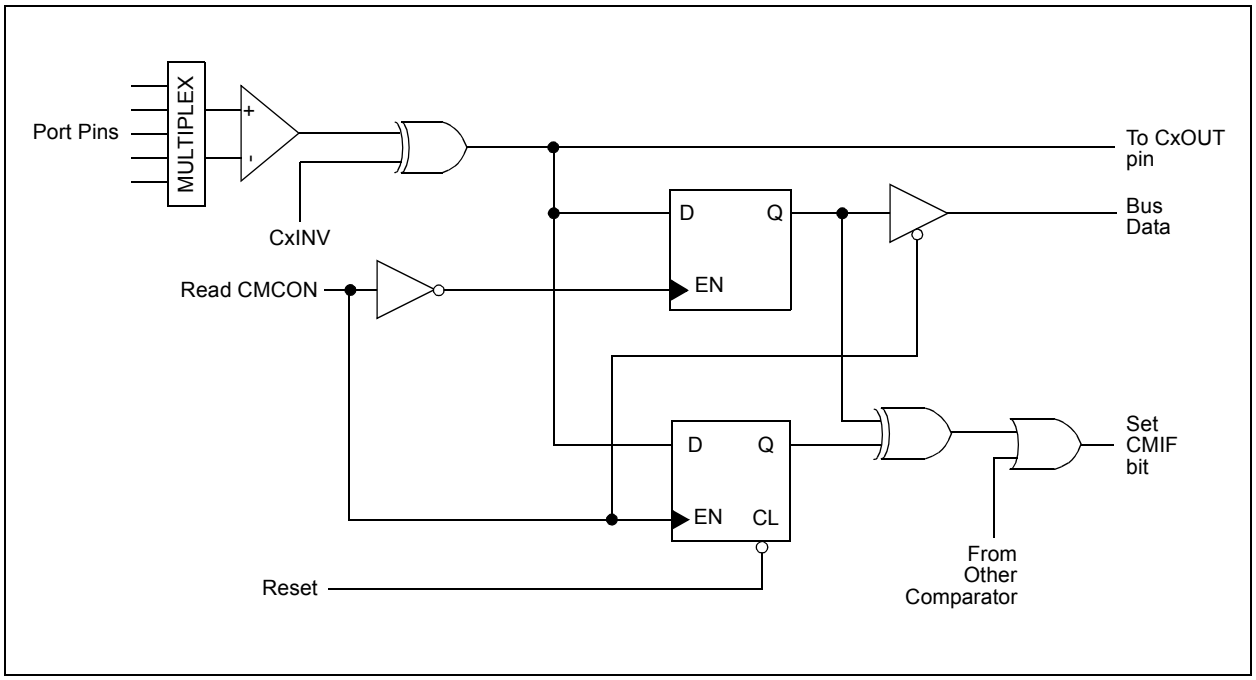
The comparator outputs are read through the CMCON register. These bits are read-only. The comparator outputs may also be directly output to the RF1 and RF2 I/O pins. When enabled, multiplexors in the output path of the RF1 and RF2 pins will switch and the output of each pin will be the unsynchronized 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 23-3 shows the comparator output block diagram.

The TRISF bits will still function as an output enable/disable 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 23-3: COMPARATOR OUTPUT BLOCK DIAGRAM



23.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR2<6>) is the Comparator Interrupt Flag. The CMIF bit must be reset by clearing it. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

Both the CMIE bit (PIE2<6>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

Note:	If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR2 register) interrupt flag may not get set.

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit, CMIF.

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

23.7 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.

23.8 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.

23.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 23-4. 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 23-4: COMPARATOR ANALOG INPUT MODEL

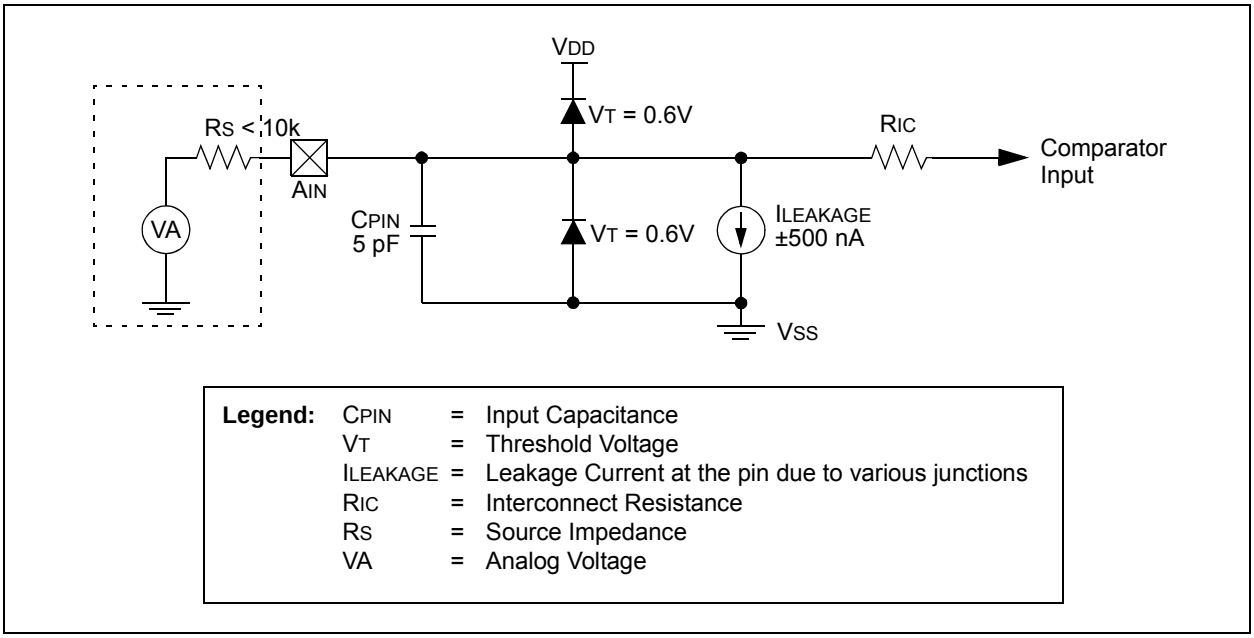


TABLE 23-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE

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	69
PIR2	OSCFIF	CMIF	ETHIF	r	BCL1IF	—	TMR3IF	CCP2IF	71
PIE2	OSCFIE	CMIE	ETHIE	r	BCL1IE	—	TMR3IE	CCP2IE	71
IPR2	OSCFIP	CMIP	ETHIP	r	BCL1IP	—	TMR3IP	CCP2IP	71
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	70
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	70
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	72
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	71

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

NOTES:

24.0 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 24-1. 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.

24.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 24-1). 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 28-3 in **Section 28.0 "Electrical Characteristics"**).

REGISTER 24-1: 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 powered on
0 = CVREF circuit powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RF5/AN10/CVREF pin
0 = CVREF voltage is disconnected from the RF5/AN10/CVREF pin

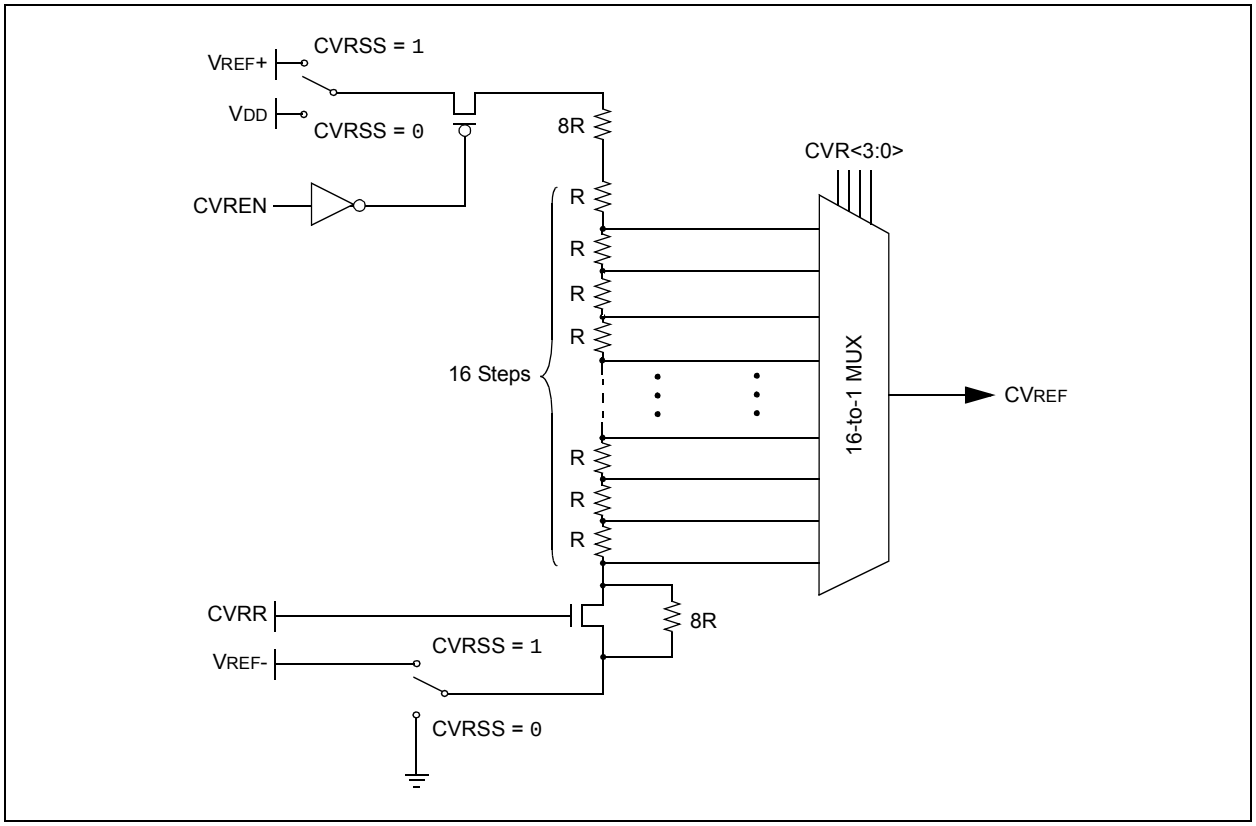
bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)

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 24-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



24.2 Comparator Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 24-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in **Section 28.0 "Electrical Characteristics"**.

24.3 Operation During Sleep

When the device wakes up from Sleep through an interrupt, or a Watchdog Timer time-out, the contents of the CVRCON register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

24.4 Effects of a Reset

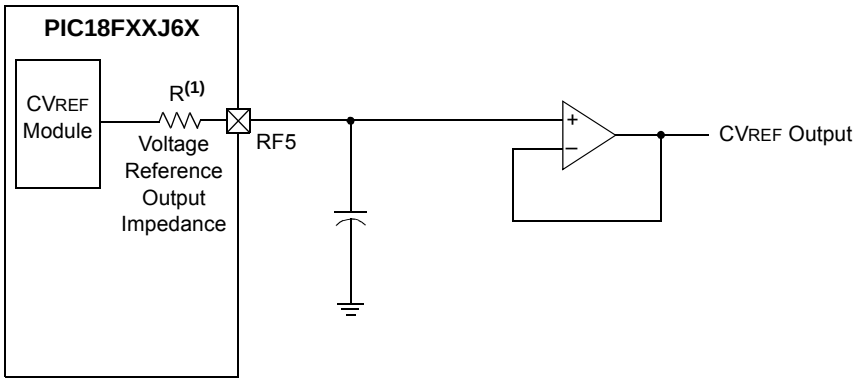
A device Reset disables the voltage reference by clearing bit, CVREN (CVRCON<7>). This Reset also disconnects the reference from the RA2 pin by clearing bit, CVROE (CVRCON<6>), and selects the high-voltage range by clearing bit, CVRR (CVRCON<5>). The CVR value select bits are also cleared.

24.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RF5 pin if the 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 voltage reference output for external connections to VREF. Figure 24-2 shows an example buffering technique.

FIGURE 24-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the comparator voltage reference configuration bits, CVRCON<5> and CVRCON<3:0>.

TABLE 24-1: 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	70
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	70
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	71

Legend: — = unimplemented, read as '0'. Shaded cells are not used with the comparator voltage reference module.

NOTES:

25.0 SPECIAL FEATURES OF THE CPU

PIC18F97J60 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 PIC18F97J60 family of devices has 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.

25.1 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 25-1. A detailed explanation of the various bit functions is provided in Register 25-1 through Register 25-8.

25.1.1 CONSIDERATIONS FOR CONFIGURING THE PIC18F97J60 FAMILY DEVICES

Devices of the PIC18F97J60 family do not use persistent memory registers to store configuration information. The configuration bytes are implemented as volatile memory which means that configuration data must be programmed each time the device is powered up.

Configuration data is stored in the four words at the top of the on-chip program memory space, known as the Flash Configuration Words, which are located in the program memory space, as shown in Table 6-1. The Configuration Words are stored in the same order shown in Table 25-1, with CONFIG1L at the lowest address and CONFIG3H at the highest. The data is automatically loaded in the proper Configuration registers during device power-up.

When creating applications for these devices, users should always specifically allocate the location of the Flash Configuration Word for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The volatile memory cells used for the Configuration bits always reset to '1' on Power-on Resets. For all other 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 25-1: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value(1)
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN	110- ---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	WAIT(4)	BW(4)	EMB1(4)	EMB0(4)	EASHFT(4)	—	—	—	1111 1---
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	—	ETHLED	ECCPMX(5)	CCP2MX(5)	---- -111
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(6)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	xxxx xxxx(6)

Legend: x = unknown, u = unchanged, - = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: Values reflect the unprogrammed state as received from the factory and following Power-on Resets. In all other Reset states, the configuration bytes maintain their previously programmed states.

2: The value of these bits in program memory should always be '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

3: This bit should always be maintained as '0'.

4: Implemented in 100-pin devices only.

5: Implemented in 80-pin and 100-pin devices only.

6: See Register 25-7 and Register 25-8 for DEVID values. These registers are read-only and cannot be programmed by the user.

REGISTER 25-1: CONFIG1L: CONFIGURATION REGISTER 1 LOW (BYTE ADDRESS 300000h)

R/WO-1	R/WO-1	R/WO-0	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 is enabled
0 = WDT is disabled (control is placed on SWDTEN bit)

REGISTER 25-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-0	U-0	U-0	U-0	U-0(1)	R/WO-1	U-0	U-0
—(2)	—(2)	—(2)	—(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 7-3 **Unimplemented:** Read as '0'

bit 2 **CP0:** Code Protection bit
1 = Program memory is not code-protected
0 = Program memory is code-protected

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

Note 1: This bit should always be maintained as '0'.
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.

REGISTER 25-3: 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 **FOSC2:** Default/Reset System Clock Select bit
1 = Clock selected by FOSC<1:0> as system clock is enabled when OSCCON<1:0> = 00
0 = INTRC enabled as system clock when OSCCON<1:0> = 00

bit 1-0 **FOSC<1:0>:** Oscillator Selection bits
11 = EC oscillator, PLL is enabled and under software control, CLKO function on OSC2
10 = EC oscillator, CLKO function on OSC2
01 = HS oscillator, PLL is enabled and under software control
00 = HS oscillator

REGISTER 25-4: 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 '0'

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 25-5: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT(1)	BW(1)	EMB1(1)	EMB0(1)	EASHFT(1)	—	—	—
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)
- 1 = Wait states for operations on external memory bus is disabled
- 0 = Wait states for operations on external memory bus is enabled and selected by MEMCON<5:4>

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

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

bit 3 **EASHFT:** External Address Bus Shift Enable bit(1)
- 1 = Address shifting is enabled; address on external bus is offset to start at 000000h
- 0 = Address shifting is disabled; address on external bus reflects the PC value

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

Note 1: Implemented on 100-pin devices only.

REGISTER 25-6: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

U-0	U-0	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1
—(1)	—(1)	—(1)	—(1)	—	ETHLED	ECCPMX(2)	CCP2MX(2)
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 7-3 **Unimplemented:** Read as '0'

bit 2 **ETHLED:** Ethernet LED Enable bit
- 1 = RA0/RA1 are multiplexed with LEDA/LEDB when the Ethernet module is enabled and function as I/O when the Ethernet is disabled
- 0 = RA0/RA1 function as I/O regardless of Ethernet module status

bit 1 **ECCPMX:** ECCP MUX bit(2)
- 1 = ECCP1 outputs (P1B/P1C) are multiplexed with RE6 and RE5; ECCP3 outputs (P3B/P3C) are multiplexed with RE4 and RE3
- 0 = ECCP1 outputs (P1B/P1C) are multiplexed with RH7 and RH6; ECCP3 outputs (P3B/P3C) are multiplexed with RH5 and RH4

bit 0 **CCP2MX:** ECCP2 MUX bit(2)
- 1 = ECCP2/P2A is multiplexed with RC1
- 0 = ECCP2/P2A is multiplexed with RE7 in Microcontroller mode (80-pin and 100-pin devices) or with RB3 in Extended Microcontroller mode (100-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.

2: Implemented in 80-pin and 100-pin devices only.

REGISTER 25-7: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F97J60 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	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		u = Unchanged from programmed state

bit 7-5 **DEV<2:0>:** Device ID bits
See Register 25-8 for a complete listing.

bit 4-0 **REV<4:0>:** Revision ID bits
These bits are used to indicate the device revision.

REGISTER 25-8: DEVID2: DEVICE ID REGISTER 2 FOR PIC18F97J60 FAMILY DEVICES

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

Legend:		
R = Read-only bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		u = Unchanged from programmed state

bit 7-0 **DEV<10:3>:** Device ID bits:

DEV<10:3> (DEVID2<7:0>)	DEV<2:0> (DEVID1<7:5>)	Device
0001 1000	000	PIC18F66J60
0001 1111	000	PIC18F66J65
0001 1111	001	PIC18F67J60
0001 1000	001	PIC18F86J60
0001 1111	010	PIC18F86J65
0001 1111	011	PIC18F87J60
0001 1000	010	PIC18F96J60
0001 1111	100	PIC18F96J65
0001 1111	101	PIC18F97J60

25.2 Watchdog Timer (WDT)

For PIC18F97J60 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.

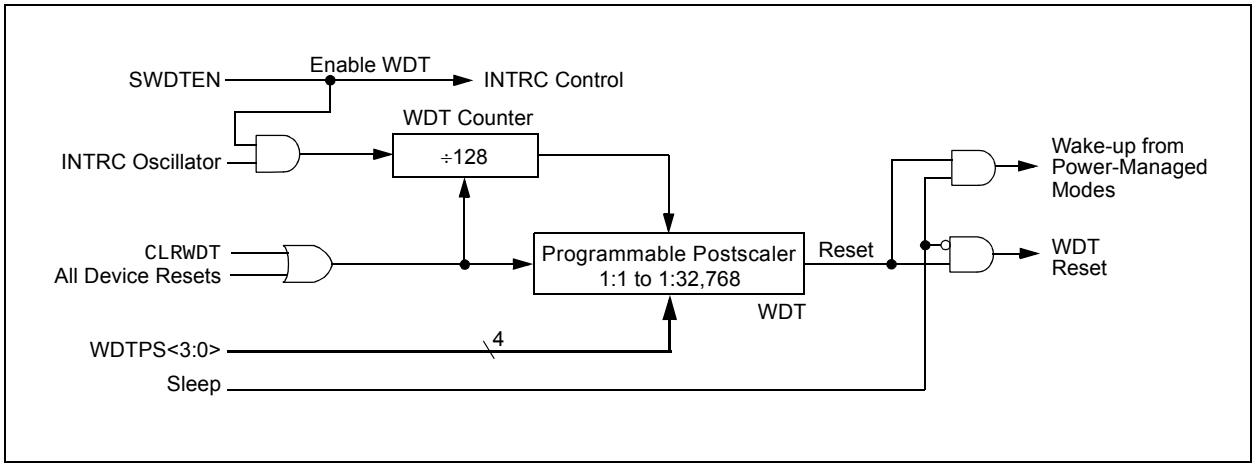
Note 1: The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.

2: When a CLRWDT instruction is executed, the postscaler count will be cleared.

25.2.1 CONTROL REGISTER

The WDTCON register (Register 25-9) 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 25-1: WDT BLOCK DIAGRAM



REGISTER 25-9: WDTCON: WATCHDOG TIMER CONTROL REGISTER

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

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

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

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit(1)
1 = Watchdog Timer is on
0 = Watchdog Timer is off

Note 1: This bit has no effect if the Configuration bit, WDTEN, is enabled.

TABLE 25-2: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
WDTCON	—	—	—	—	—	—	—	SWDTEN	70

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

25.3 On-Chip Voltage Regulator

All of the PIC18F97J60 family devices power their core digital logic at a nominal 2.5V. This may create an issue for designs that are required to operate at a higher typical voltage, such as 3.3V. To simplify system design, all devices in the PIC18F97J60 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 25-2). This helps to maintain the stability of the regulator. The recommended value for the filter capacitor is provided in **Section 28.3 "DC Characteristics: PIC18F97J60 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 25-2 for possible configurations.

25.3.1 ON-CHIP REGULATOR AND BOR

When the on-chip regulator is enabled, PIC18F97J60 family devices also have a simple brown-out capability. If the voltage supplied to the regulator is inadequate to maintain a regulated level, the regulator Reset circuitry will generate a Brown-out Reset. This event is captured by the $\overline{\text{BOR}}$ flag bit (RCON<0>).

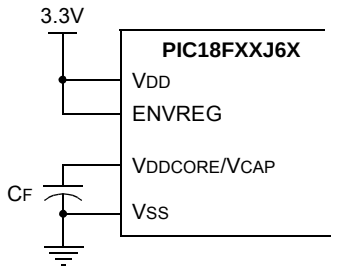
The operation of the BOR is described in more detail in **Section 5.4 "Brown-out Reset (BOR)"** and **Section 5.4.1 "Detecting BOR"**. The Brown-out Reset voltage levels are specific in **Section 28.1 "DC Characteristics: Supply Voltage, PIC18F97J60 Family (Industrial)"**

25.3.2 POWER-UP REQUIREMENTS

The on-chip regulator is designed to meet the power-up requirements for the device. If the application does not use the regulator, then strict power-up conditions must be adhered to. While powering up, VDDCORE must never exceed VDD by 0.3 volts.

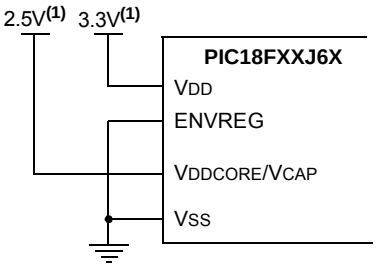
FIGURE 25-2: CONNECTIONS FOR THE ON-CHIP REGULATOR

Regulator Enabled (ENVREG tied to VDD):

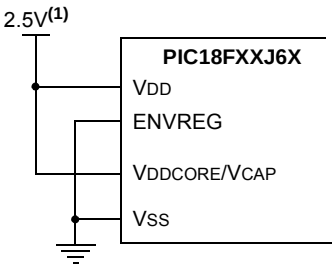


Regulator Disabled (ENVREG tied to ground):

(VDD > VDDCORE)



(VDD = VDDCORE)



Note 1: These are typical operating voltages. Refer to **Section 28.1 "DC Characteristics: Supply Voltage"** for the full operating ranges of VDD and VDDCORE.

25.4 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period, from oscillator start-up to code execution, by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is HS or HSPLL (Crystal-Based) modes. Since the EC and ECPLL modes do not require an Oscillator Start-up Timer 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.

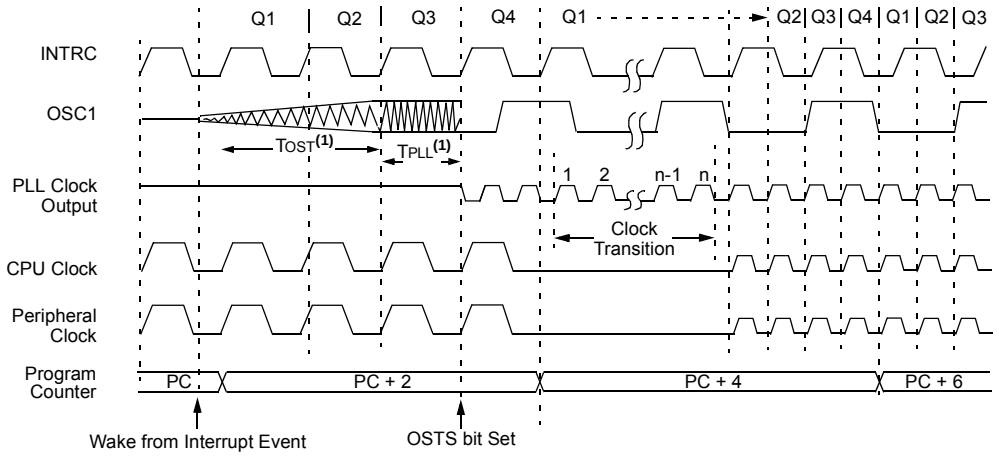
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

25.4.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial SLEEP instructions (refer to **Section 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 25-3: TIMING TRANSITION FOR TWO-SPEED START-UP (INTRC TO HSPLL)



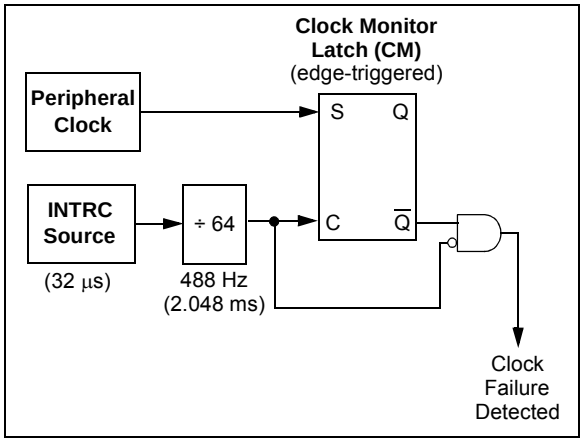
Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

25.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 25-4) is accomplished by creating a sample clock signal which is the INTRC output, divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor (CM) latch. The CM is set on the falling edge of the device clock source but cleared on the rising edge of the sample clock.

FIGURE 25-4: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 25-5). 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)
- 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 25.4.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

25.5.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

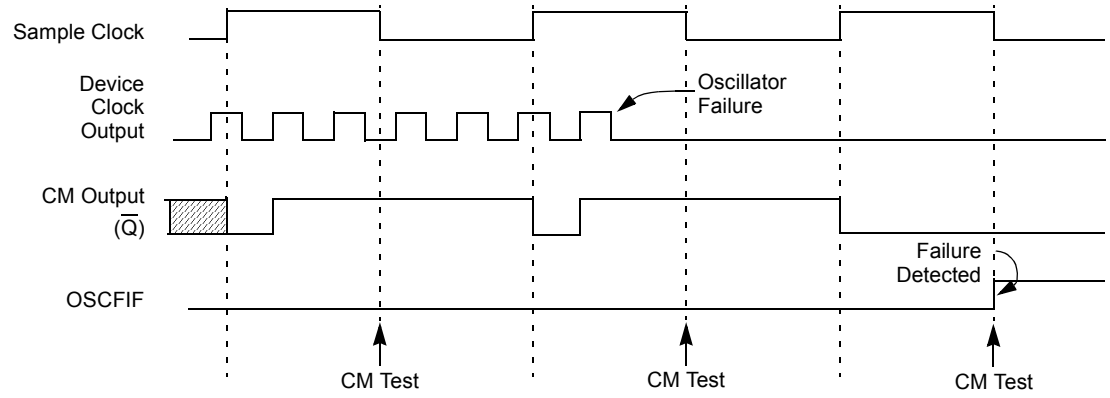
As already noted, the clock source is switched to the INTRC clock when a clock failure is detected. This may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock 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.

25.5.2 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 INTRC oscillator. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

FIGURE 25-5: 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.

25.5.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexor selects the clock source selected by the OSCCON register. Fail-Safe Monitoring of the power-managed clock source resumes in the power-managed mode.

If an oscillator failure occurs during power-managed operation, the subsequent events depend on whether or not the oscillator failure interrupt is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTRC multiplexor. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, subsequent interrupts while in Idle mode will cause the CPU to begin executing instructions while being clocked by the INTRC source.

25.5.4 POR OR WAKE-UP FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary device clock is either EC or INTRC, 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 25.4.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration and enter an alternate power-managed mode while waiting for the primary clock to become stable. When the new power-managed mode is selected, the primary clock is disabled.

25.6 Program Verification and Code Protection

For all devices in the PIC18F97J60 family, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by one Configuration bit, CP0. This bit inhibits external reads and writes to the program memory space. It has no direct effect in normal execution mode.

25.6.1 CONFIGURATION REGISTER PROTECTION

The Configuration registers are protected against untoward changes or reads in two ways. The primary protection is the write-once feature of the Configuration bits which prevents reconfiguration once the bit has been programmed during a power cycle. To safeguard against unpredictable events, Configuration bit changes resulting from individual cell level disruptions (such as ESD events) will cause a parity error and trigger a device Reset.

The data for the Configuration registers is derived from the Flash Configuration Words in program memory. When the CP0 bit is programmed (cleared), the source data for device configuration is also protected as a consequence.

25.7 In-Circuit Serial Programming

PIC18F97J60 family microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

25.8 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ Configuration bit is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 25-3 shows which resources are required by the background debugger.

TABLE 25-3: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	512 bytes
Data Memory:	10 bytes

NOTES:

26.0 INSTRUCTION SET SUMMARY

The PIC18F97J60 family of devices incorporates 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.

26.1 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 26-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 26-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator ‘f’ specifies which file register is to be used by the instruction. The destination designator ‘d’ specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator ‘b’ selects the number of the bit affected by the operation, while the file register designator ‘f’ represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 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 26-1 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 26-2, lists the standard instructions recognized by the Microchip MPASM™ Assembler.

Section 26.1.1 “Standard Instruction Set” provides a description of each instruction.

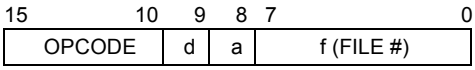
TABLE 26-1: OPCODE FIELD DESCRIPTIONS

Field	Description
a	RAM access bit: a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
C, DC, Z, OV, N	ALU Status bits: **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 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 26-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations

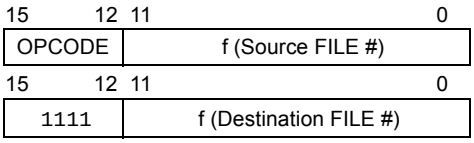
Example Instruction




```
ADDWF MYREG, W, B
```

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

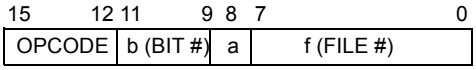
Byte to Byte move operations (2-word)




```
MOVFF MYREG1, MYREG2
```

f = 12-bit file register address

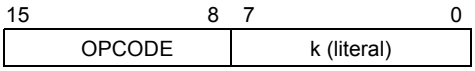
Bit-oriented file register operations




```
BSF MYREG, bit, B
```

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Literal operations

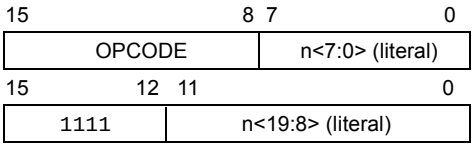



```
MOVLW 7Fh
```

k = 8-bit immediate value

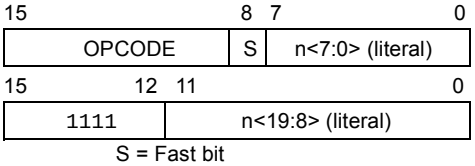
Control operations

CALL, GOTO and Branch operations



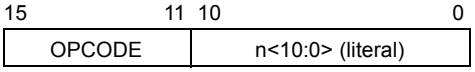

```
GOTO Label
```

n = 20-bit immediate value

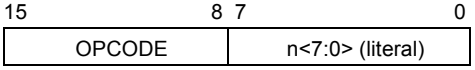



```
CALL MYFUNC
```

S = Fast bit




```
BRA MYFUNC
```




```
BC MYFUNC
```

TABLE 26-2: PIC18F97J60 FAMILY INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
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 an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 26-2: PIC18F97J60 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 an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 26-2: PIC18F97J60 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 an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

26.1.1 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’ (default).

If ‘a’ is ‘0’, the Access Bank is selected. If ‘a’ is ‘1’, the BSR is used to select the GPR bank (default).

If ‘a’ is ‘0’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: 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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: ADDWFC REG, 0, 1

Before Instruction
Carry bit = 1
REG = 02h
W = 4Dh
After Instruction
Carry bit = 0
REG = 02h
W = 50h

ANDLW **AND Literal with W**

Syntax: ANDLW k

Operands: $0 \le k \le 255$

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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `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 ≤ n ≤ 1023

Operation: (PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	0nnn	nnnn	nnnn

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: HERE BRA Jump

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)

BSF **Bit Set f**

Syntax: BSF f, b {,a}

Operands: 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

Operation: 1 → f<b>

Status Affected: None

Encoding:

1000	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is set.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write 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 (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(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 (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(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 (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write 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 (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `HERE CALL THERE,1`

Before Instruction
PC = address (HERE)
After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS = STATUS

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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: 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 post-scaler 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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `COMF REG, 0, 0`

Before Instruction
REG = 13h
After Instruction
REG = 13h
W = ECh

CPFSEQ **Compare f with W, Skip if f = W**

Syntax: CPFSEQ f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ 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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```
Before Instruction
```
PC  =  Address (HERE)
W   =  ?
```
After Instruction
```
If REG  >  W;
    PC  =  Address (GREATER)
If REG  ≤  W;
    PC  =  Address (NGREATER)
```

CPFSLT **Compare f with W, Skip if f < W**

Syntax: CPFSLT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

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 (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```
Before Instruction
```
PC  =  Address (HERE)
W   =  ?
```
After Instruction
```
If REG  <  W;
    PC  =  Address (LESS)
If REG  ≥  W;
    PC  =  Address (NLESS)
```

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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: 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' (default).

If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    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' (default).

If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: INCF CNT, 1, 0

Before Instruction
CNT = FFh
Z = 0
C = ?
DC = ?
After Instruction
CNT = 00h
Z = 1
C = 1
DC = 1

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'. (default)

If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INCFSZ   CNT, 1, 0
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' (default).

If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INFSNZ  REG, 1, 0
ZERO
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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: 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}kk 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' (default). Location 'f' can be anywhere in the 256-byte bank.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 22h
W = FFh
After Instruction
REG = 22h
W = 22h

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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: MOVWF REG, 0

Before Instruction
W = 4Fh
REG = FFh
After Instruction
W = 4Fh
REG = 4Fh

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 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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: MULWF REG, 1

Before Instruction
W = C4h
REG = B5h
PRODH = ?
PRODL = ?
After Instruction
W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

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 (default).

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 26.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register ‘f’

Example:

```
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 ≤ n ≤ 1023

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	1nnn	nnnn	nnnn

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: HERE RCALL Jump

Before Instruction
PC = Address (HERE)
After Instruction
PC = Address (Jump)
TOS = Address (HERE + 2)

RESET **Reset**

Syntax: RESET

Operands: None

Operation: Reset all registers and flags that are affected by a MCLR Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a MCLR Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: RESET

After Instruction
Registers = Reset Value
Flags* = Reset Value

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 (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	POP PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: RETFIE 1

After Interrupt
PC = TOS
W = WS
BSR = BSRS
STATUS = 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 (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: RETURN

After Instruction:
PC = TOS

RLCF **Rotate Left f through Carry**

Syntax: RLCF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f<n>) → dest<n + 1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

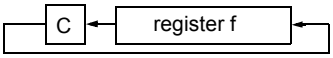
Encoding:

0011	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RLCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

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

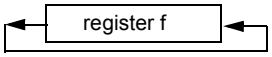
Encoding:

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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: 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

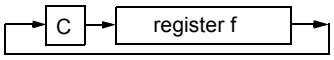
Encoding:

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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: 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

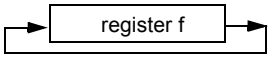
Encoding:

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' (default).

If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1:
```
RRNCF   REG, 1, 0
```
Before Instruction
REG = 1101 0111
After Instruction
REG = 1110 1011

Example 2:
```
RRNCF   REG, 0, 0
```
Before Instruction
W = ?
REG = 1101 0111
After Instruction
W = 1110 1011
REG = 1101 0111

SETF **Set f**

Syntax: SETF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example:
```
SETF    REG,1
```
Before Instruction
REG = 5Ah
After Instruction
REG = FFh

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’ (default).

If ‘a’ is ‘0’, the Access Bank is selected. If ‘a’ is ‘1’, the BSR is used to select the GPR bank (default).

If ‘a’ is ‘0’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example 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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   3
    W     =   2
    C     =   ?
After Instruction
    REG   =   1
    W     =   2
    C     =   1     ; result is positive
    Z     =   0
    N     =   0
```

Example 2:
```
SUBWF   REG, 0, 0
Before Instruction
    REG   =   2
    W     =   2
    C     =   ?
After Instruction
    REG   =   2
    W     =   0
    C     =   1     ; result is zero
    Z     =   1
    N     =   0
```

Example 3:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   1
    W     =   2
    C     =   ?
After Instruction
    REG   =   FFh   ;(2's complement)
    W     =   2
    C     =   0     ; result is negative
    Z     =   0
    N     =   1
```

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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 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' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `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 (default).

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 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 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 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(W) .XOR. (f) → dest
Status Affected:	N, Z

Encoding:

0001	10da	ffff	ffff

Description: Exclusive OR the contents of W with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register 'f' (default).

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `XORWF REG, 1, 0`

Before Instruction
REG = AFh
W = B5h
After Instruction
REG = 1Ah
W = B5h

26.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, the PIC18F97J60 family of devices also provide an optional extension to the core CPU functionality. The added features include eight additional instructions that augment Indirect and Indexed Addressing operations and the implementation of Indexed Literal Offset Addressing 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 reentrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- Dynamic allocation and deallocation of software stack space when entering and leaving subroutines
- Function Pointer invocation
- Software Stack Pointer manipulation
- Manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 26-3. Detailed descriptions are provided in **Section 26.2.2 "Extended Instruction Set"**. The opcode field descriptions in Table 26-1 (page 376) apply to both the standard and extended PIC18 instruction sets.

Note: The instruction set extension and the Indexed Literal Offset Addressing mode were designed for optimizing applications written in C; the user may likely never use these instructions directly in assembler. The syntax for these commands is provided as a reference for users who may be reviewing code that has been generated by a compiler.

26.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the 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 26.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**.

Note: In the past, square brackets have been used to denote optional arguments in the PIC18 and earlier instruction sets. In this text and going forward, optional arguments are denoted by braces ("{ }").

TABLE 26-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
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

26.2.2 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:

1110	1010	kkkk	kkkk

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation.

This instruction allows users to push values onto a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process data	Write to destination

Example: PUSHL 08h

Before Instruction
FSR2H:FSR2L = 01ECh
Memory (01ECh) = 00h

After Instruction
FSR2H:FSR2L = 01EBh
Memory (01ECh) = 08h

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)

26.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note: Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing (**Section 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 register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward-compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 26.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

26.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument 'f' in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within the brackets, will generate an error in the MPASM Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled), when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument 'd' functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

26.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18F97J60 family, it is very important to consider the type of code. A large, reentrant 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' (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process Data	Write to destination

Example: ADDWF [OFST],0

Before Instruction
W = 17h
OFST = 2Ch
FSR2 = 0A00h
Contents of 0A2Ch = 20h
After Instruction
W = 37h
Contents of 0A2Ch = 20h

BSF **Bit Set Indexed (Indexed Literal Offset mode)**

Syntax: BSF [k], b

Operands: $0 \le f \le 95$
$0 \le b \le 7$

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

26.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set for the PIC18F97J60 family. This includes the MPLAB C18 C Compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default Configuration bits for that device. The default setting for the XINST Configuration bit is '0', disabling the extended instruction set and Indexed Literal Offset Addressing. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option or dialog box within the environment that allows the user to configure the language tool and its settings for the project
- A command line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

27.0 DEVELOPMENT SUPPORT

The PIC® 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

27.1 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.

27.2 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.

27.3 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.

27.4 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

27.5 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

27.6 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

27.7 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.

27.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The 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.

27.9 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.

27.10 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.

27.11 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.

27.12 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.

27.13 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.

28.0 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 input pin or $\overline{MCLR}$ with respect to V_{SS} (except V_{DD}) ..... -0.3V to 6.0V
Voltage on any combined digital and analog pin with respect to V_{SS} ..... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{DDCORE} with respect to V_{SS} ..... -0.3V to 2.75V
Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to 4.0V
Total power dissipation **(Note 1)** 1.0W
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin ..... 250 mA
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) **(Note 2)** ..... ±0 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) **(Note 2)** ..... ±0 mA
Maximum output current sunk by 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 any PORTA, PORTF, PORTG and PORTH I/O pins **(Note 3)** ..... 2 mA
Maximum output current sourced by 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 any PORTA, PORTF, PORTG and PORTH I/O pins **(Note 3)** ..... 2 mA
Maximum current sunk by all ports combined ..... 200 mA
Maximum current sourced by all ports combined ..... 200 mA

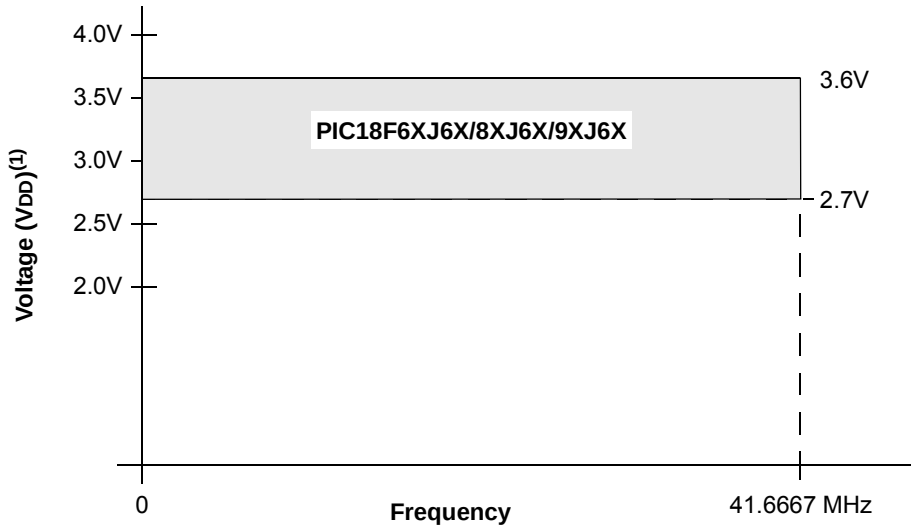
Note 1: Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL}) + \sum (V_{TPOUT} \times I_{TPOUT})$

2: No clamping diodes are present.

3: Exceptions are RA<1> and RA<0>, which are capable of directly driving LEDs up to 25 mA.

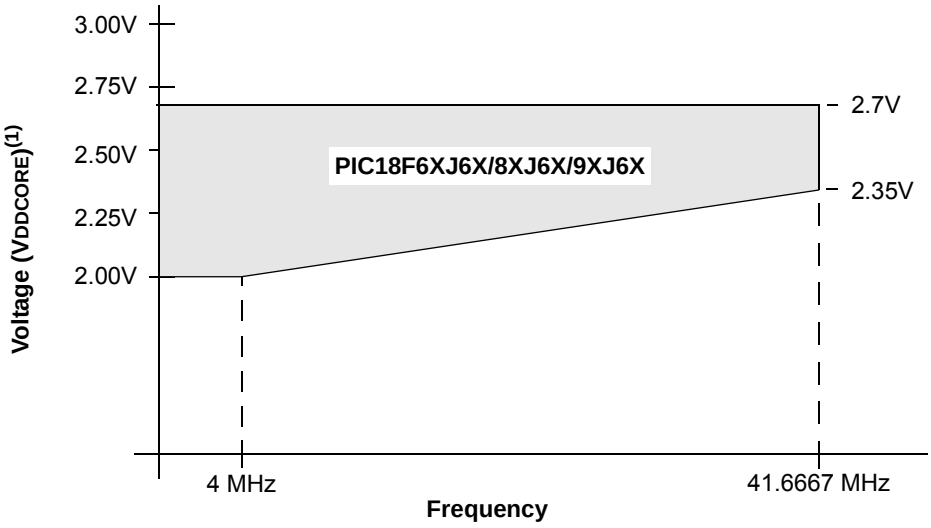
† **NOTICE:** Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 28-1: PIC18F97J60 FAMILY VOLTAGE-FREQUENCY GRAPH, REGULATOR ENABLED (ENVREG TIED TO VDD)



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.

FIGURE 28-2: PIC18F97J60 FAMILY VOLTAGE-FREQUENCY GRAPH, REGULATOR DISABLED (ENVREG TIED TO VSS)



For frequencies between 4 MHz and 41.6667 MHz, FMAX = (107.619 MHz/V) * (VDDCORE – 2V) + 4 MHz.

Note 1: When the on-chip voltage regulator is disabled, VDD and VDDCORE must be maintained so that VDDCORE ≤ VDD ≤ 3.6V.

28.1 DC Characteristics: Supply Voltage, PIC18F97J60 Family (Industrial)

PIC18F97J60 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 2.7 3.1	— — —	3.6 3.6 3.6	V V V	ENVREG tied to VSS ENVREG tied to VDD Ethernet module enabled (ECON2<5> = 1)
D001B	VDDCORE	**External Supply for Microcontroller Core**	2.0	—	2.7	V	
D001C	AVDD	**Analog Supply Voltage**	VDD – 0.3	—	VDD + 0.3	V	
D002	VDR	**RAM Data Retention Voltage(1)**	1.5	—	—	V	
D003	VPOR	**VDD Power-on Reset Voltage**	—	—	0.7	V	See **Section 5.3 "Power-on Reset (POR)"** for details
D004	SVDD	**VDD Rise Rate** to Ensure Internal Power-on Reset	0.05	—	—	V/ms	See **Section 5.3 "Power-on Reset (POR)"** for details
D005	BOR	Brown-out Reset	2.35	2.4	2.7	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.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial)

PIC18F97J60 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	19.0	69.0	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4) (**Sleep** mode)
		21.0	69.0	µA	+25°C	
		45.0	149.0	µA	+85°C	
	All devices	26.0	104.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4) (**Sleep** mode)
		29.0	104.0	µA	+25°C	
		60.0	184.0	µA	+85°C	
	All devices	40.0	203.0	µA	-40°C	VDD = 3.3V(5) (**Sleep** mode)
		44.0	203.0	µA	+25°C	
		105.0	209.0	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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 (I_{DD})(2,3)						
	All devices	12.0	34.0	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 31 kHz (**RC_RUN** mode, Internal Oscillator Source)
		12.0	34.0	µA	+25°C		
		74.0	108.0	µA	+85°C		
	All devices	20.0	45.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		20.0	45.0	µA	+25°C		
		82.0	126.0	µA	+85°C		
	All devices	105.0	168.0	µA	-40°C	VDD = 3.3V(5)	
		105.0	168.0	µA	+25°C		
		182.0	246.0	µA	+85°C		
	All devices	8.0	32.0	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 31 kHz (**RC_IDLE** mode, Internal Oscillator Source)
		8.0	32.0	µA	+25°C		
		62.0	98.0	µA	+85°C		
	All devices	12.0	35.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		12.0	35.0	µA	+25°C		
		70.0	95.0	µA	+85°C		
	All devices	90.0	152.0	µA	-40°C	VDD = 3.3V(5)	
		90.0	152.0	µA	+25°C		
		170.0	225.0	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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	0.8	1.5	mA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**PRI_RUN** mode, EC oscillator)
		0.8	1.5	mA	+25°C		
		0.9	1.7	mA	+85°C		
	All devices	1.1	1.8	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		1.1	1.8	mA	+25°C		
		1.2	2.0	mA	+85°C		
	All devices	2.1	3.4	mA	-40°C	VDD = 3.3V(5)	
		2.0	3.4	mA	+25°C		
		2.1	3.4	mA	+85°C		
	All devices	9.2	14.5	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz (**PRI_RUN** mode, EC oscillator)
		9.0	14.5	mA	+25°C		
		9.2	14.5	mA	+85°C		
	All devices	13.0	18.4	mA	-40°C	VDD = 3.3V(5)	
		12.4	18.4	mA	+25°C		
		13.0	18.4	mA	+85°C		
	All devices	13.4	19.8	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 41.6667 MHz (**PRI_RUN** mode, EC oscillator)
		13.0	19.8	mA	+25°C		
		13.4	19.8	mA	+85°C		
	All devices	14.5	21.6	mA	-40°C	VDD = 3.3V(5)	
		14.4	21.6	mA	+25°C		
		14.5	21.6	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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 (I_{DD})(2)						
	All devices	2.8	5.2	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz, 2.7778 MHz internal (**PRI_RUN HS** mode)
		2.5	5.2	mA	+25°C		
		2.8	5.2	mA	+85°C		
	All devices	3.6	6.4	mA	-40°C	VDD = 3.3V(5)	
		3.3	6.4	mA	+25°C		
		3.6	6.4	mA	+85°C		
	All devices	6.4	11.0	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz, 13.8889 MHz internal (**PRI_RUN HSPLL** mode)
		6.0	11.0	mA	+25°C		
		6.4	11.0	mA	+85°C		
	All devices	7.8	12.5	mA	-40°C	VDD = 3.3V(5)	
		7.4	12.5	mA	+25°C		
		7.8	12.5	mA	+85°C		
	All devices	9.2	14.5	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz, 25 MHz internal (**PRI_RUN HS** mode)
		9.0	14.5	mA	+25°C		
		9.2	14.5	mA	+85°C		
	All devices	13.0	18.4	mA	-40°C	VDD = 3.3V(5)	
		12.4	18.4	mA	+25°C		
		13.0	18.4	mA	+85°C		
	All devices	13.4	19.8	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz, 41.6667 MHz internal (**PRI_RUN HSPLL** mode)
		13.0	19.8	mA	+25°C		
		13.4	19.8	mA	+85°C		
	All devices	14.5	21.6	mA	-40°C	VDD = 3.3V(5)	
		14.4	21.6	mA	+25°C		
		14.5	21.6	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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	0.5	1.1	mA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator)
		0.5	1.1	mA	+25°C		
		0.6	1.2	mA	+85°C		
	All devices	0.9	1.4	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		0.9	1.4	mA	+25°C		
		1.0	1.5	mA	+85°C		
	All devices	1.9	2.6	mA	-40°C	VDD = 3.3V(5)	
		1.8	2.6	mA	+25°C		
		1.9	2.6	mA	+85°C		
	All devices	5.9	9.5	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 25 MHz (**PRI_IDLE** mode, EC oscillator)
		5.6	9.5	mA	+25°C		
		5.9	9.5	mA	+85°C		
	All devices	7.5	13.2	mA	-40°C	VDD = 3.3V(5)	
		7.2	13.2	mA	+25°C		
		7.5	13.2	mA	+85°C		
	All devices	8.6	14.0	mA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	FOSC = 41.6667 MHz (**PRI_IDLE** mode, EC oscillator)
		8.0	14.0	mA	+25°C		
		8.6	14.0	mA	+85°C		
	All devices	9.8	16.0	mA	-40°C	VDD = 3.3V(5)	
		9.4	16.0	mA	+25°C		
		9.8	16.0	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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 (I_{DD})(2)						
	All devices	22.0	45.0	µA	-10°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 32 kHz(3) (**SEC_RUN** mode, Timer1 as clock)
		22.0	45.0	µA	+25°C		
		78.0	114.0	µA	+70°C		
	All devices	27.0	52.0	µA	-10°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		27.0	52.0	µA	+25°C		
		92.0	135.0	µA	+70°C		
	All devices	106.0	168.0	µA	-10°C	VDD = 3.3V(5)	
		106.0	168.0	µA	+25°C		
		188.0	246.0	µA	+70°C		
	All devices	18.0	37.0	µA	-10°C	VDD = 2.0V, VDDCORE = 2.0V(4)	FOSC = 32 kHz(3) (**SEC_IDLE** mode, Timer1 as clock)
		18.0	37.0	µA	+25°C		
		75.0	105.0	µA	+70°C		
	All devices	21.0	40.0	µA	-10°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		21.0	40.0	µA	+25°C		
		84.0	98.0	µA	+70°C		
	All devices	94.0	152.0	µA	-10°C	VDD = 3.3V(5)	
		94.0	152.0	µA	+25°C		
		182.0	225.0	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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.2 DC Characteristics: Power-Down and Supply Current PIC18F97J60 Family (Industrial) (Continued)

PIC18F97J60 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, ΔIETH)						
D022 (ΔIWDT)	**Watchdog Timer**	2.4	7.0	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	
		2.4	7.0	µA	+25°C		
		12.0	19.0	µA	+85°C		
		3.0	8.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		3.0	8.0	µA	+25°C		
		14.0	22.0	µA	+85°C		
		5.0	12.0	µA	-40°C	VDD = 3.3V(5)	
		5.0	12.0	µA	+25°C		
		19.0	30.0	µA	+85°C		
D025 (ΔIOSCB)	**Timer1 Oscillator**	12.0	20.0	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V(4)	32 kHz on Timer1(3)
		12.0	20.0	µA	+25°C		
		24.0	36.0	µA	+85°C		
		13.0	21.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V(4)	32 kHz on Timer1(3)
		13.0	21.0	µA	+25°C		
		26.0	38.0	µA	+85°C		
		14.0	25.0	µA	-40°C	VDD = 3.3V(5)	32 kHz on Timer1(3)
		14.0	25.0	µA	+25°C		
		29.0	40.0	µA	+85°C		
D026 (ΔIAD)	**A/D Converter**	1.2	10.0	µA	-40°C to +85°C	VDD = 2.0V, VDDCORE = 2.0V(4)	A/D on, not converting
		1.2	10.0	µA	-40°C to +85°C	VDD = 2.5V, VDDCORE = 2.5V(4)	
		1.2	11.0	µA	-40°C to +85°C	VDD = 3.3V(5)	
D027 ΔIETH(6)	**Ethernet Module**	130.0	156.0	mA	-40°C to +85°C	VDD = 3.3V(5)	No transmit activity
		180.0	214.0	mA	-40°C to +85°C		Transmission in progress

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, 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 = 0, tied to VSS).

5: Voltage regulator enabled (ENVREG = 1, tied to VDD).

6: For ΔIETH, the specified current includes current sunk through TPOUT+ and TPOUT-. LEDA and LEDB are disabled for all testing.

28.3 DC Characteristics: PIC18F97J60 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	0.15VDD	V	VDD<2.7V
			VSS	0.8	V	2.7V ≤ VDD ≤ 3.6V
D031		with Schmitt Trigger Buffer	VSS	0.2 VDD	V	
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 mode
D034		T13CKI	VSS	0.3	V	
	VIH	**Input High Voltage**				
		I/O Ports, with Analog Functions:				
D040		with TTL Buffer	0.25 VDD + 0.8V	VDD	V	
D041		with Schmitt Trigger Buffer	0.8 VDD	VDD	V	
		I/O Ports, Digital Only:				
		with TTL Buffer	0.25 VDD + 0.8V	5.5	V	
		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 mode
D044		T13CKI	1.6	VDD	V	
	IIL	**Input Leakage Current(1)**				
D060		I/O Ports	—	±1	μA	VSS ≤ VPIN ≤ VDD, 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, PORTD, PORTE, PORTJ	80	400	μA	VDD = 3.3V, VPIN = VSS

Note 1: Negative current is defined as current sourced by the pin.

28.3 DC Characteristics: PIC18F97J60 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:				
		PORTD, PORTE, PORTJ	—	0.4	V	IOL = 4 mA, VDD = 3.3V, -40°C to +85°C
		PORTA<5:2>, PORTF, PORTG, PORTH	—	0.4	V	IOL = 2 mA, VDD = 3.3V, -40°C to +85°C
		PORTA<1:0>, PORTB, PORTC	—	0.4	V	IOL = 8 mA, VDD = 3.3V, -40°C to +85°C
D083		OSC2/CLKO (EC, ECPLL modes)	—	0.4	V	IOL = 2 mA, VDD = 3.3V, -40°C to +85°C
	VOH	**Output High Voltage(1)**				
D090		I/O Ports:			V	
		PORTD, PORTE, PORTJ	2.4	—	V	IOH = -4 mA, VDD = 3.3V, -40°C to +85°C
		PORTA<5:2>, PORTF, PORTG, PORTH	2.4	—	V	IOH = -2 mA, VDD = 3.3V, -40°C to +85°C
		PORTA<1:0>, PORTB, PORTC	2.4	—	V	IOH = -8 mA, VDD = 3.3V, -40°C to +85°C
D092		OSC2/CLKO (EC, ECPLL modes)	2.4	—	V	IOH = -1.0 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 (in Internal RC mode, EC, ECPLL)	—	50	pF	To meet the AC timing specifications
D102	CB	SCLx, SDAx	—	400	pF	I^2C™ specification

Note 1: Negative current is defined as current sourced by the pin.

TABLE 28-1: 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
D132B	VPEW	Voltage for Self-Timed Erase or Write					
		VDD	2.70	—	3.6	V	ENVREG tied to VDD
		VDDCORE	2.35	—	2.7	V	ENVREG tied to VSS
D133A	TIW	Self-Timed Write Cycle Time	—	2.8	—	ms	
D134	TRETD	Characteristic Retention	20	—	—	Year	Provided no other specifications are violated
D135	IDDP	Supply Current during Programming	—	10	—	mA	Ethernet module disabled

† Data in “Typ” column is at 3.3V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 28-2: COMPARATOR SPECIFICATIONS

Operating Conditions: 3.0V ≤ 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	
300	TRESP	Response Time(1)*	—	150	400	ns	
301	TMC2OV	Comparator Mode Change to Output Valid*	—	—	10	μs	

* These parameters are characterized but not tested.

Note 1: Response time measured with one comparator input at (AVDD – 1.5)/2, while the other input transitions from VSS to AVDD.

TABLE 28-3: 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	
310	TSET	Settling Time(1)	—	—	10	μs	

Note 1: Settling time measured while CVRR = 1 and CVR<3:0> transitions from '0000' to '1111'.

TABLE 28-4: 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	
	CF	External Filter Capacitor Value	1	10	—	μF	Capacitor must be low series resistance

28.4 AC (Timing) Characteristics

28.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
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	ECCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDIx	sc	SCKx
do	SDOx	ss	$\overline{SS}$x
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		

28.4.2 TIMING CONDITIONS

The temperature and voltages specified in Table 28-5 apply to all timing specifications unless otherwise noted. Figure 28-3 specifies the load conditions for the timing specifications.

TABLE 28-5: 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 DC spec **Section 28.1 "DC Characteristics: Supply Voltage, PIC18F97J60 Family (Industrial)"** and **Section 28.3 "DC Characteristics: PIC18F97J60 Family (Industrial)"**.

FIGURE 28-3: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS

RL = 464Ω

CL = 50 pF for all pins except OSC2/CLKO and including D and E outputs as ports

CL = 15 pF for OSC2/CLKO

28.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 28-4: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)

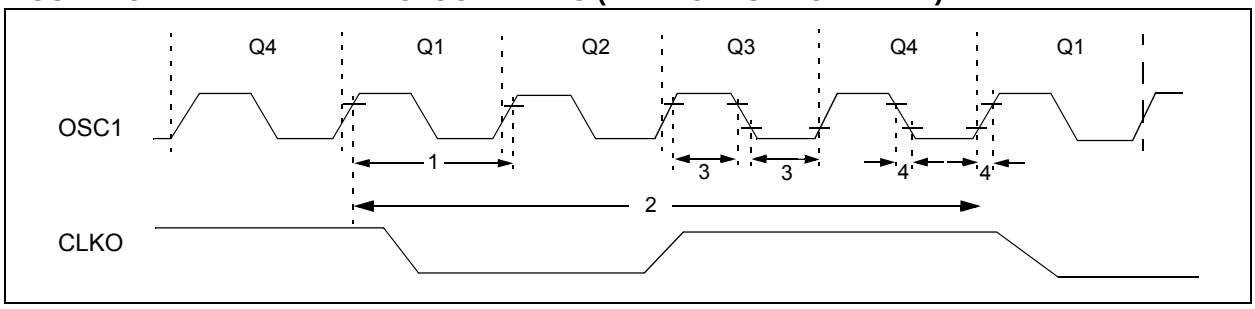


TABLE 28-6: EXTERNAL CLOCK TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	FOSC	External CLKI Frequency(1)	DC	41.6667	MHz	EC Oscillator mode
		Oscillator Frequency(1)	6	25	MHz	HS Oscillator mode
1	TOSC	External CLKI Period(1)	24	—	ns	EC Oscillator mode
		Oscillator Period(1)	40	167	ns	HS Oscillator mode
2	TCY	Instruction Cycle Time(1)	96	—	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
5		Clock Frequency Tolerance	—	±50	ppm	Ethernet module enabled

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

TABLE 28-7: PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 2.6V TO 3.6V)

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
F10	FOSC	Oscillator Frequency Range	8	—	25	MHz	HSPLL mode
			8	—	37.5	MHz	ECPLL mode
F11	FSYS	On-Chip VCO System Frequency	20	—	62.5	MHz	
F12	t_{rc}	PLL Start-up Time (Lock Time)	—	—	2	ms	
F13	ΔCLK	CLKO Stability (Jitter)	-2	—	+2	%	

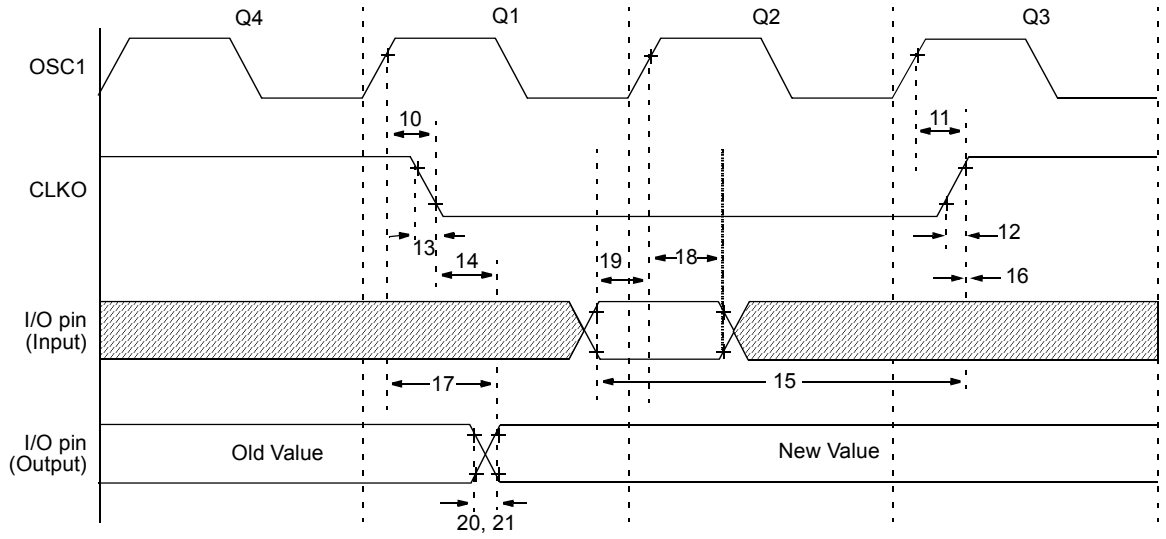
† Data in "Typ" column is at 3.3V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 28-8: AC CHARACTERISTICS: INTERNAL RC ACCURACY PIC18F97J60 FAMILY (INDUSTRIAL)

Param No.	Characteristic	Min	Typ	Max	Units	Conditions
	INTRC Accuracy @ Freq = 31 kHz(1)	21.7	—	40.3	kHz	

Note 1: INTRC frequency changes as V_{DDCORE} changes.

FIGURE 28-5: CLKO AND I/O TIMING



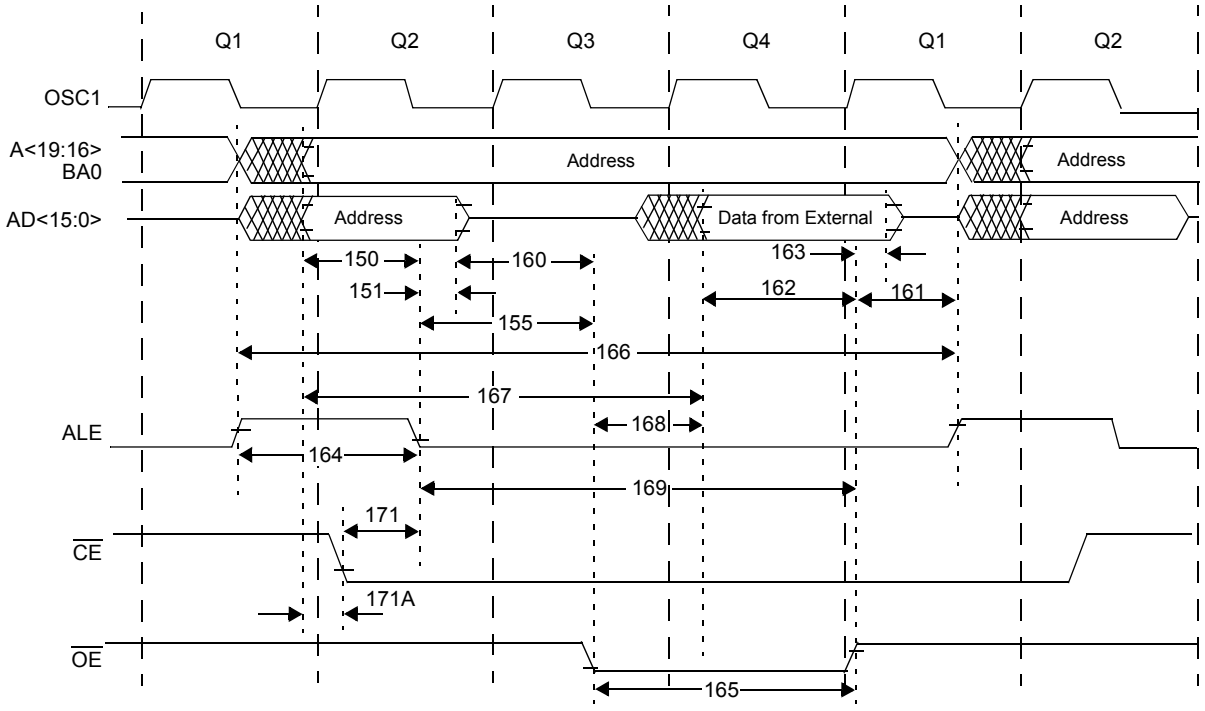
Note: Refer to Figure 28-3 for load conditions.

TABLE 28-9: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
10	TOSH2CKL	OSC1 ↑ to CLKO ↓	—	75	200	ns	
11	TOSH2CKH	OSC1 ↑ to CLKO ↑	—	75	200	ns	
12	TCKR	CLKO Rise Time	—	15	30	ns	
13	TCKF	CLKO Fall Time	—	15	30	ns	
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.

FIGURE 28-6: PROGRAM MEMORY READ TIMING DIAGRAM

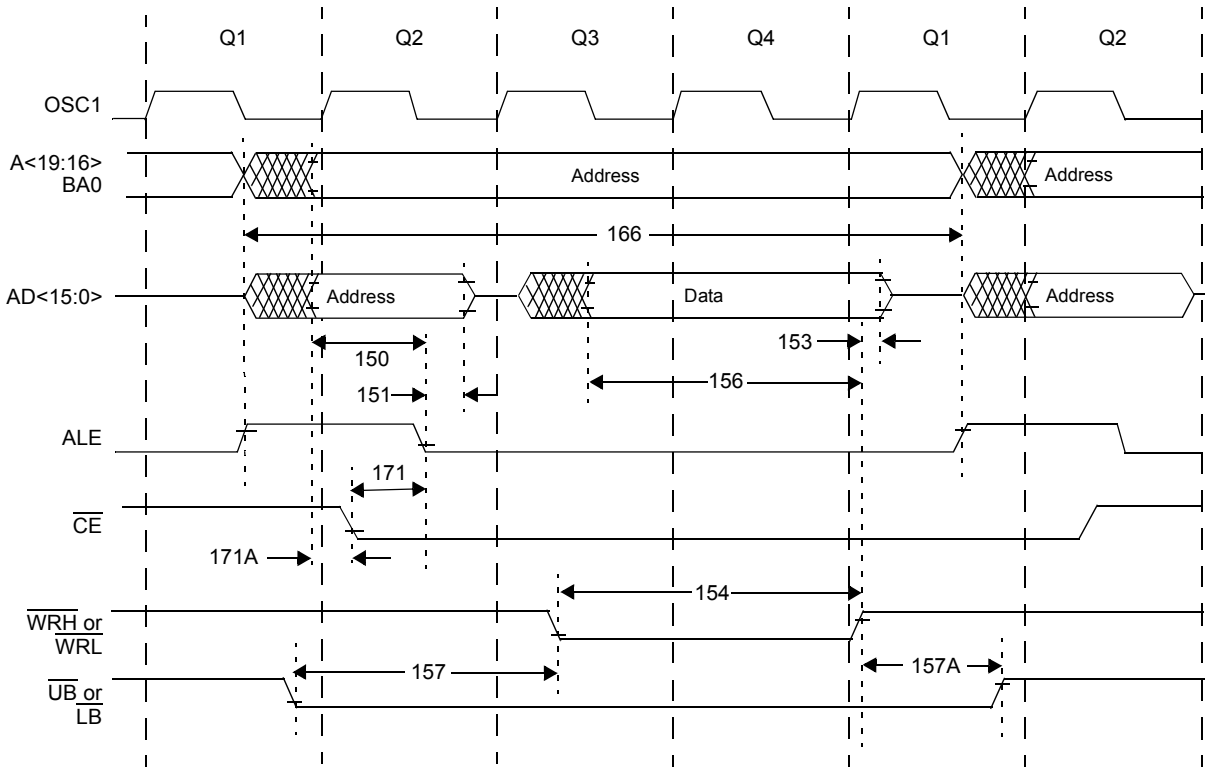


Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C, unless otherwise stated.

TABLE 28-10: 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	Least Significant 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	—	TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ Pulse Width	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	0.25 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 28-7: PROGRAM MEMORY WRITE TIMING DIAGRAM

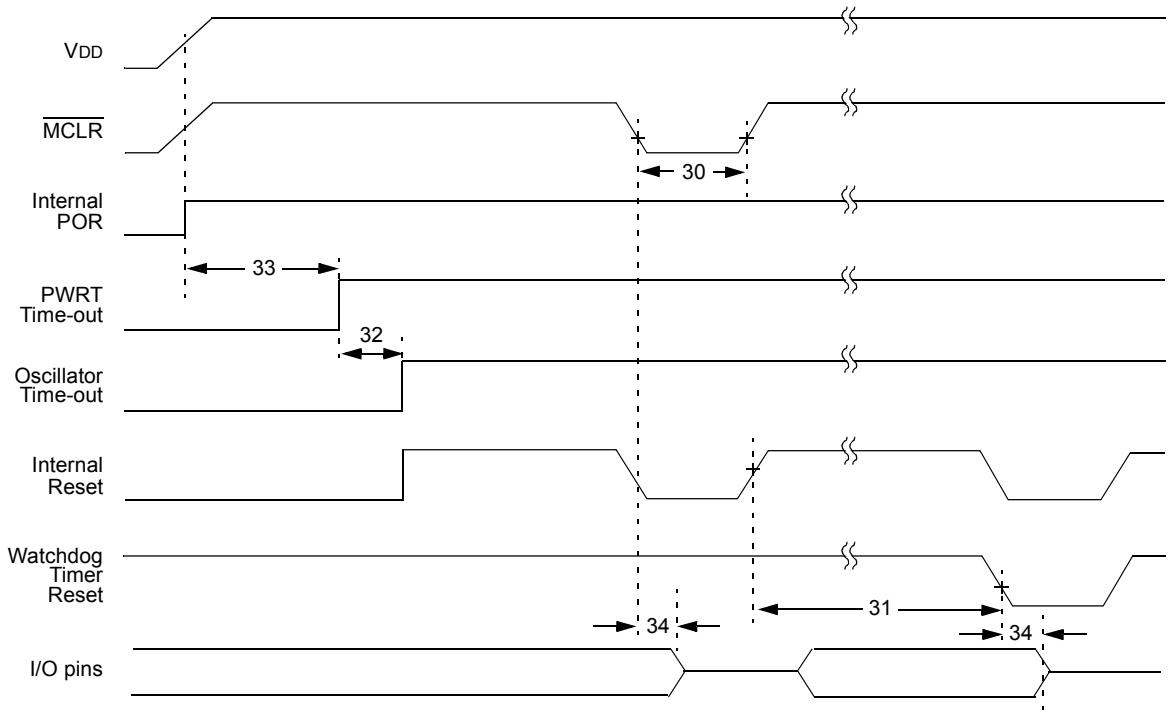


Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C, unless otherwise stated.

TABLE 28-11: 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	TwrH2bsI	$\overline{WRn}$ ↑ to Byte Select Invalid (byte select hold time)	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	0.25 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 28-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

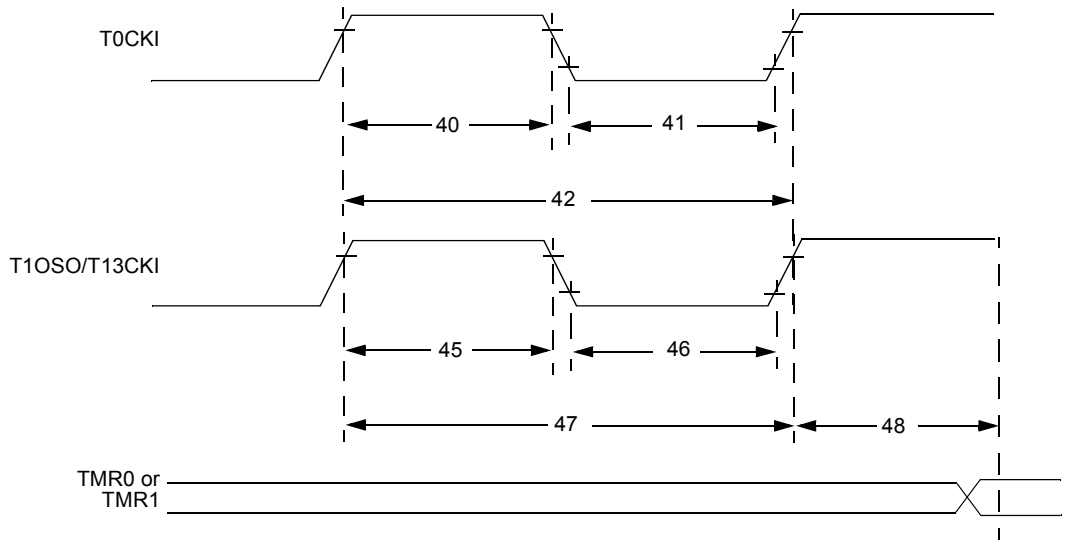


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-12: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2	—	—	µs	
31	TWDT	Watchdog Timer Time-out Period (no postscaler)	2.8	4.1	5.4	ms	
32	TOST	Oscillation Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	46.2	66	85.8	ms	
34	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	—	3 TCY + 2	µs	System clock available
			—	—	415	µs	System clock unavailable (Sleep mode or primary oscillator off)
38	TCSD	CPU Start-up Time	—	200	—	µs	

FIGURE 28-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

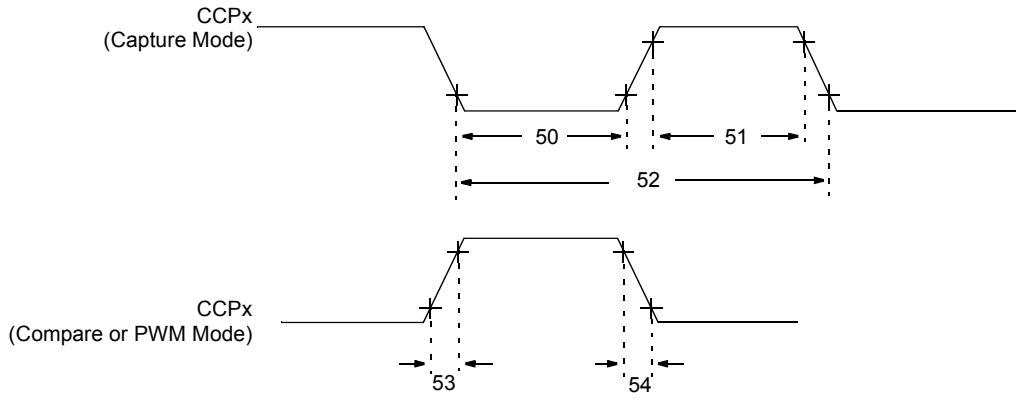


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-13: 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 28-10: CAPTURE/COMPARE/PWM TIMINGS (INCLUDING ECCPx MODULES)



Note: Refer to Figure 28-3 for load conditions.

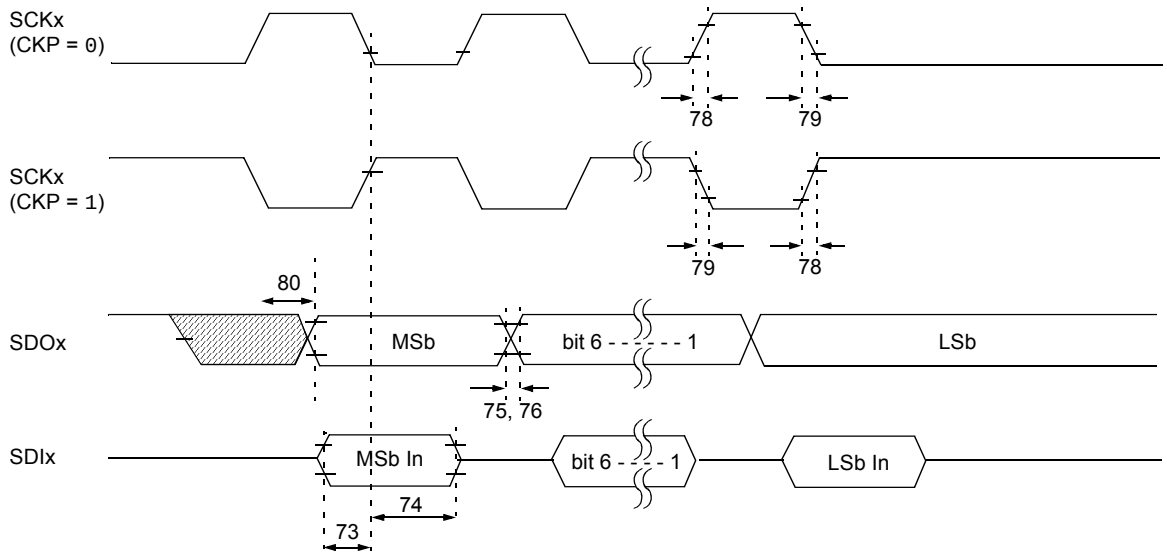
TABLE 28-14: CAPTURE/COMPARE/PWM REQUIREMENTS (INCLUDING ECCPx MODULES)

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	

TABLE 28-15: PARALLEL SLAVE PORT REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
62	TdtV2wrH	Data In Valid before $\overline{WR}$ ↑ or $\overline{CS}$ ↑ (setup time)	20	—	ns	
63	TwrH2dtI	$\overline{WR}$ ↑ or $\overline{CS}$ ↑ to Data–In Invalid (hold time)	20	—	ns	
64	TrdL2dtV	$\overline{RD}$ ↓ and $\overline{CS}$ ↓ to Data–Out Valid	—	80	ns	
65	TrdH2dtI	$\overline{RD}$ ↑ or $\overline{CS}$ ↓ to Data–Out Invalid	10	30	ns	
66	TibfINH	Inhibit of the IBF Flag bit being Cleared from $\overline{WR}$ ↑ or $\overline{CS}$ ↑	—	3 TCY		

FIGURE 28-11: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)

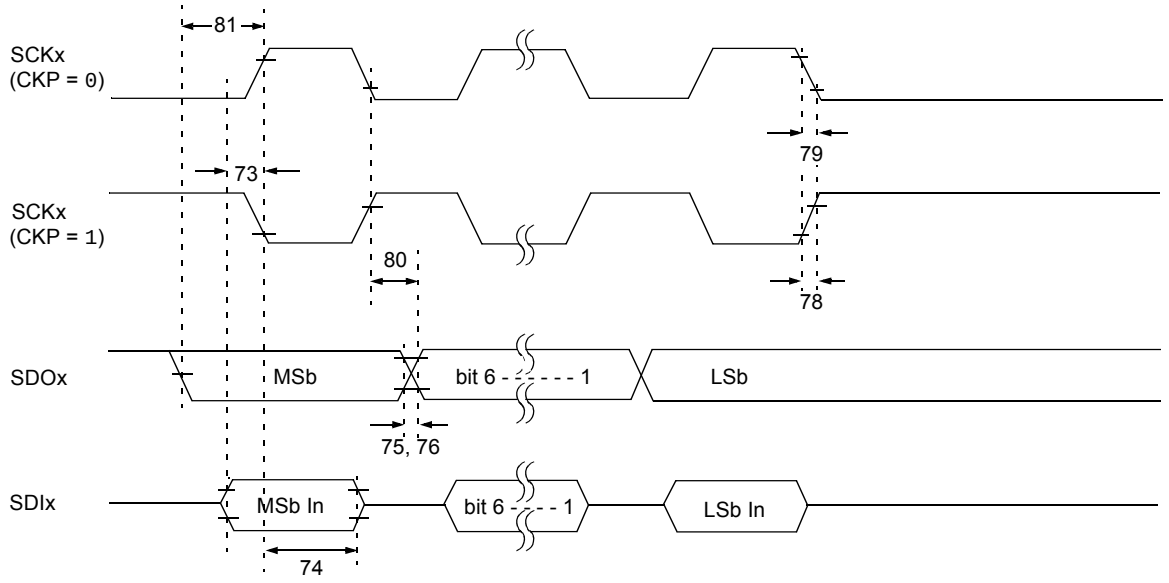


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-16: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDIx Data Input to SCKx Edge	100	—	ns	
74	TSCH2DIL, TSCL2DIL	Hold Time of SDIx Data Input to SCKx Edge	100	—	ns	
75	TDOR	SDOx Data Output Rise Time	—	25	ns	
76	TDOF	SDOx Data Output Fall Time	—	25	ns	
78	TSCR	SCKx Output Rise Time	—	25	ns	
79	TSCF	SCKx Output Fall Time	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDOx Data Output Valid after SCKx Edge	—	50	ns	

FIGURE 28-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)

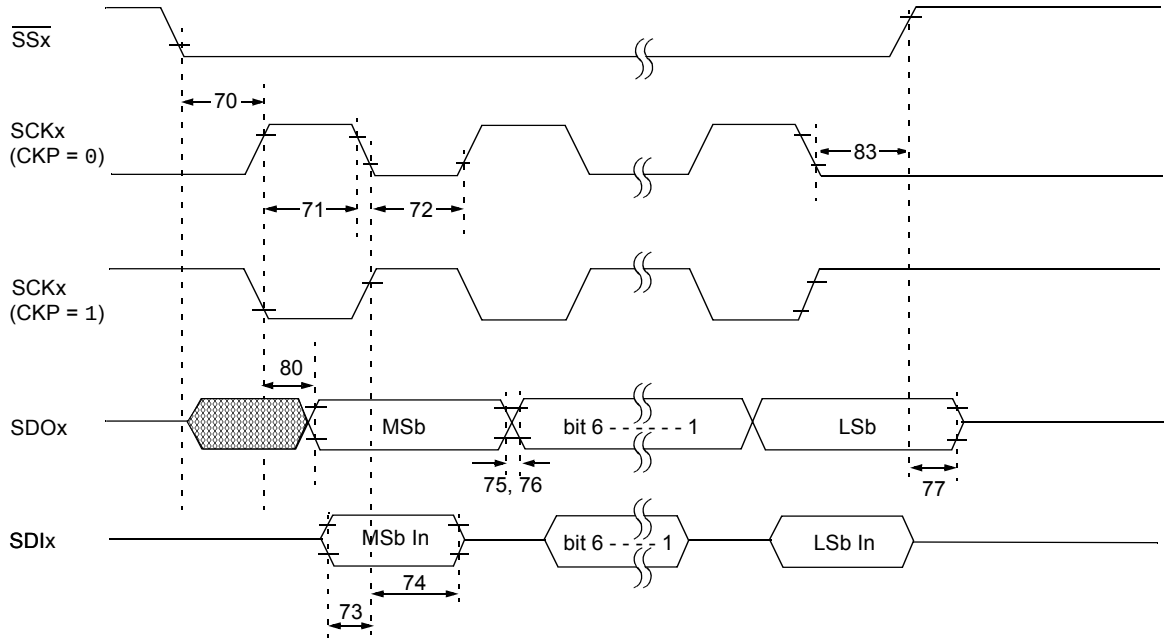


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-17: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDIx Data Input to SCKx Edge	100	—	ns	
74	TSCH2DIL, TSCL2DIL	Hold Time of SDIx Data Input to SCKx Edge	100	—	ns	
75	TDOR	SDOx Data Output Rise Time	—	25	ns	
76	TDOF	SDOx Data Output Fall Time	—	25	ns	
78	TSCR	SCKx Output Rise Time	—	25	ns	
79	TSCF	SCKx Output Fall Time	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDOx Data Output Valid after SCKx Edge	—	50	ns	
81	TDOV2SCH, TDOV2SCL	SDOx Data Output Setup to SCKx Edge	TCY	—	ns	

FIGURE 28-13: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 28-3 for load conditions.

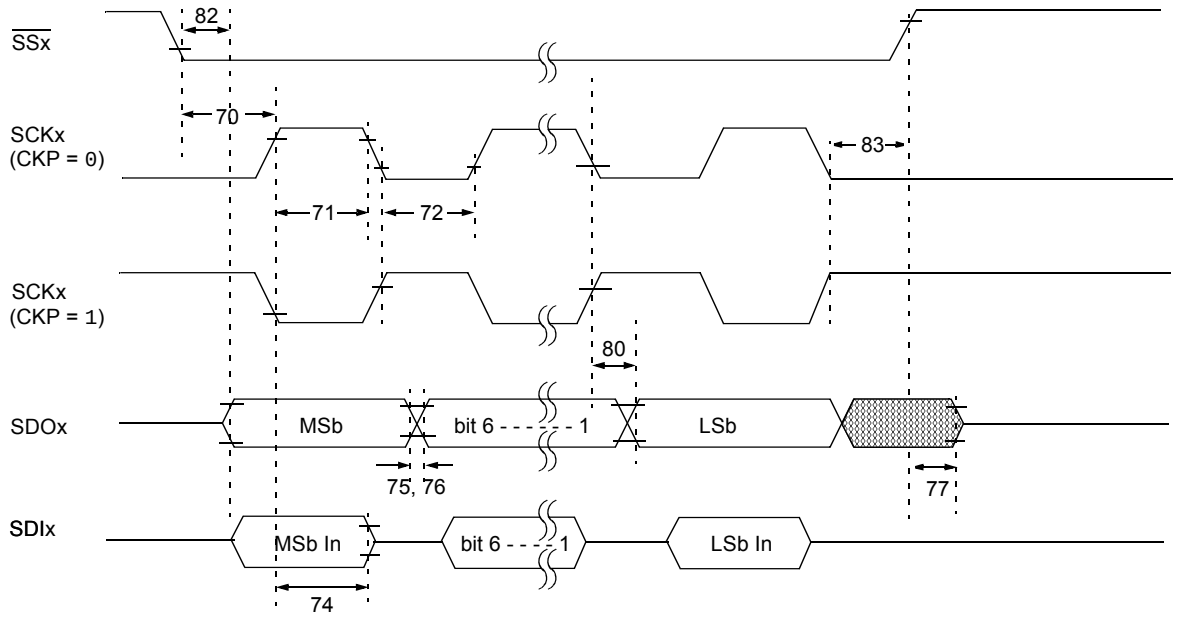
TABLE 28-18: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓ to SCKx ↓ or SCKx ↑ Input		TCY	—	ns	
71	TscH	SCKx Input High Time	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCKx Input Low Time	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge		100	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge		100	—	ns	
75	TdoR	SDOx Data Output Rise Time		—	25	ns	
76	TdoF	SDOx Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑ to SDOx Output High-impedance		10	50	ns	
80	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SSx}$ ↑ after SCKx Edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 28-14: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)



Note: Refer to Figure 28-3 for load conditions.

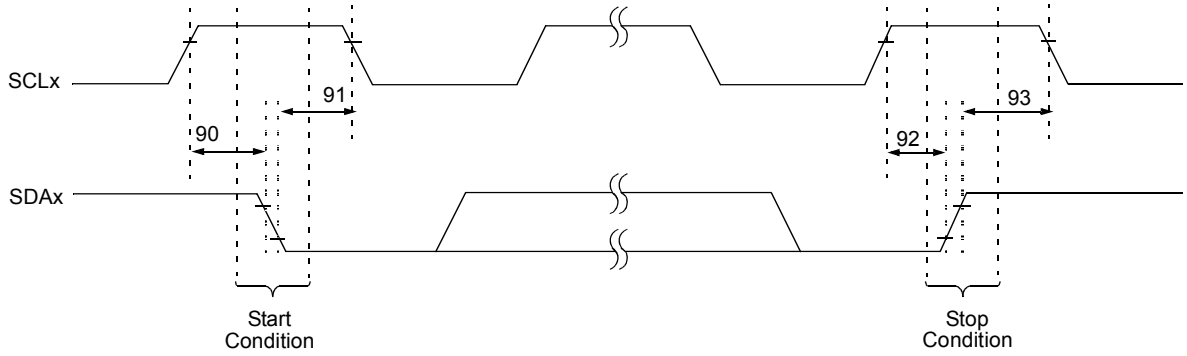
TABLE 28-19: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓ to SCKx ↓ or SCKx ↑ Input		Tcy	—	ns	
71	TscH	SCKx Input High Time	Continuous	1.25 Tcy + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCKx Input Low Time	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 SDIx Data Input to SCKx Edge		100	—	ns	
75	TdoR	SDOx Data Output Rise Time		—	25	ns	
76	TdoF	SDOx Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑ to SDOx Output High-Impedance		10	50	ns	
80	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge		—	50	ns	
82	TssL2doV	SDOx Data Output Valid after $\overline{SSx}$ ↓ Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SSx}$ ↑ after SCKx Edge		1.5 Tcy + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 28-15: I²C™ BUS START/STOP BITS TIMING

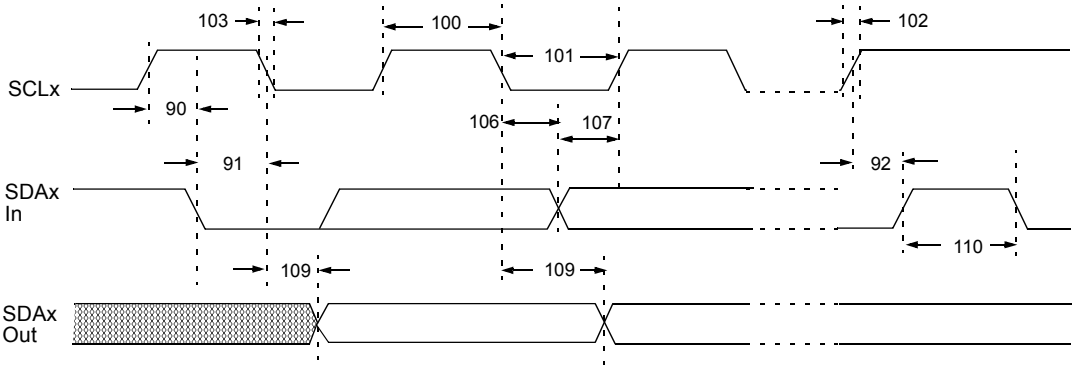


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-20: I²C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition	100 kHz mode	4700	—	ns	Only relevant for Repeated Start condition
		Setup Time	400 kHz mode	600	—		
91	THD:STA	Start Condition	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
		Hold Time	400 kHz mode	600	—		
92	TSU:STO	Stop Condition	100 kHz mode	4700	—	ns	
		Setup Time	400 kHz mode	600	—		
93	THD:STO	Stop Condition	100 kHz mode	4000	—	ns	
		Hold Time	400 kHz mode	600	—		

FIGURE 28-16: I²C™ BUS DATA TIMING



Note: Refer to Figure 28-3 for load conditions.

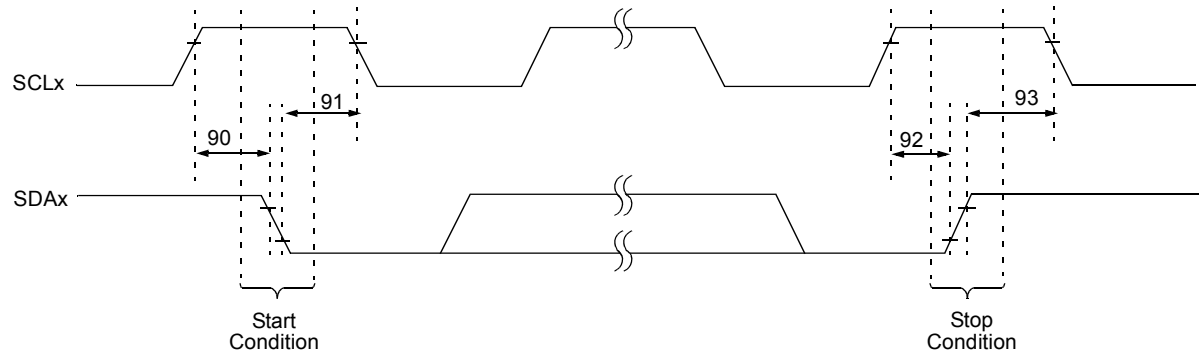
TABLE 28-21: I²C™ BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	μs	PIC18F97J60 family must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	μs	PIC18F97J60 family must operate at a minimum of 10 MHz
			MSSP module	1.5 TCY	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	μs	PIC18F97J60 family must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	μs	PIC18F97J60 family must operate at a minimum of 10 MHz
			MSSP module	1.5 TCY	—		
102	TR	SDAx and SCLx 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	SDAx and SCLx 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 SCLx to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I²C™ bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line,
TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCLx line is released.

FIGURE 28-17: MASTER SSP I²C™ BUS START/STOP BITS TIMING WAVEFORMS



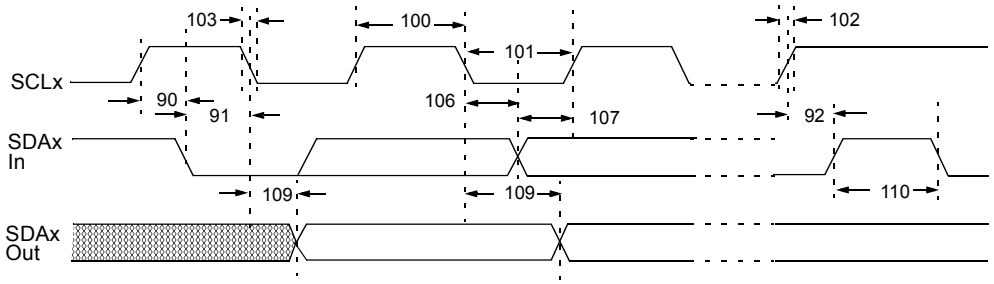
Note: Refer to Figure 28-3 for load conditions.

TABLE 28-22: MASTER SSP I²C™ BUS START/STOP BITS REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

FIGURE 28-18: MASTER SSP I²C™ BUS DATA TIMING



Note: Refer to Figure 28-3 for load conditions.

TABLE 28-23: MASTER SSP I²C™ BUS DATA REQUIREMENTS

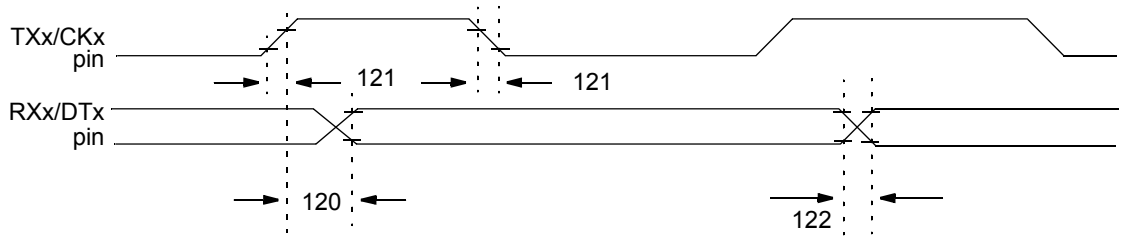
Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	Clock Low Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	300	ns	
103	TF	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	100	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	ms	
			1 MHz mode(1)	TBD	—	ns	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	TBD	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	ms	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	ms	
			1 MHz mode(1)	TBD	—	ms	
D102	CB	Bus Capacitive Loading		—	400	pF	

Legend: TBD = To Be Determined

Note 1: Maximum pin capacitance = 10 pF for all I²C™ pins.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but Parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, Parameter #102 + Parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCLx line is released.

FIGURE 28-19: EUSARTx SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

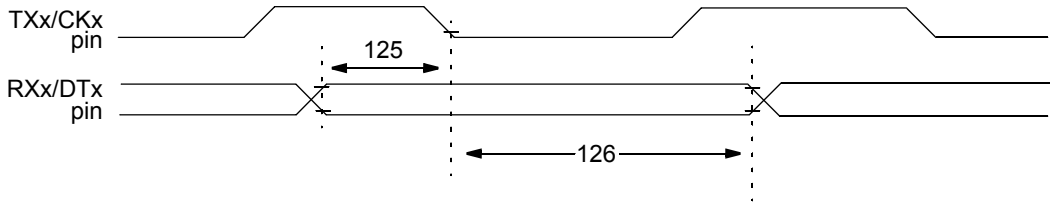


Note: Refer to Figure 28-3 for load conditions.

TABLE 28-24: EUSARTx 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 28-20: EUSARTx SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 28-3 for load conditions.

TABLE 28-25: EUSARTx 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	

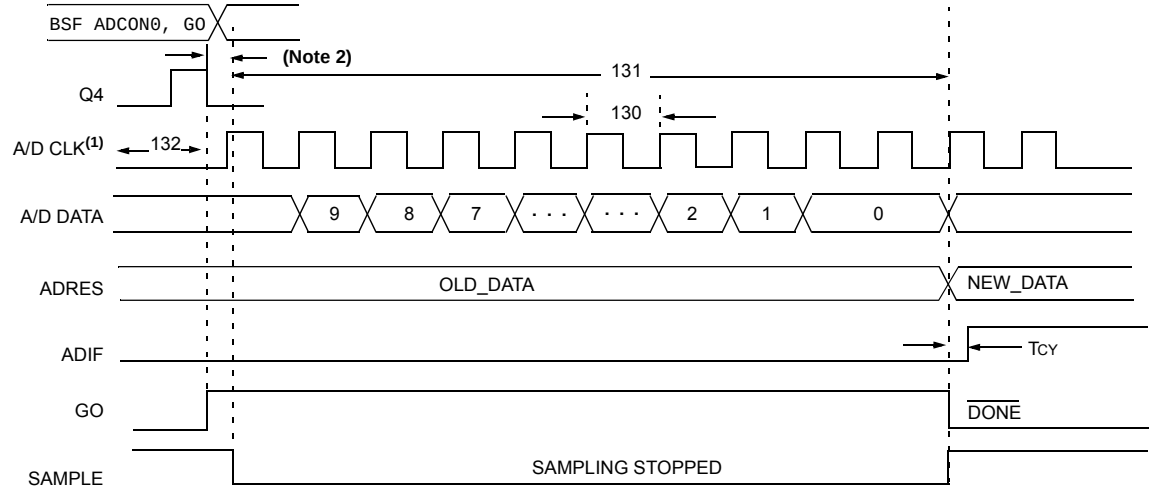
TABLE 28-26: A/D CONVERTER CHARACTERISTICS: PIC18F97J60 FAMILY (INDUSTRIAL)

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
A01	NR	Resolution	—	—	10	bit	ΔVREF ≥ 2.0V
A03	EIL	Integral Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 2.0V
A04	EDL	Differential Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 2.0V
A06	EOFF	Offset Error	—	—	<±3	LSb	ΔVREF ≥ 2.0V
A07	EGN	Gain Error	—	—	<±3	LSb	ΔVREF ≥ 2.0V
A10	—	Monotonicity	Guaranteed(1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	Reference Voltage Range (VREFH – VREFL) Reference Voltage Range (VREFH – VREFL)	1.8 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
	VREFSUM	Reference Voltage Sum (VREFH + VREFL)	—	—	AVDD + 0.5	V	
A21	VREFH	Reference Voltage High	VREFL	—	AVDD	V	
A22	VREFL	Reference Voltage Low	AVSS	—	VREFH	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) VREF Input Current(2)	— —	— —	5 1000	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 RA3/AN3/VREF+ pin or AVDD, whichever is selected as the VREFH source. VREFL current is from RA2/AN2/VREF- pin or AVSS, whichever is selected as the VREFL source.

FIGURE 28-21: 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 28-27: 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 ≥ 2.0V
			TBD	1	μs	A/D RC mode
131	TCNV	Conversion Time (not including acquisition time) **(Note 2)**	11	12	TAD	
132	TACQ	Acquisition Time **(Note 3)**	1.4	—	μs	-40°C to +85°C
135	TSWC	Switching Time from Convert → Sample	—	**(Note 4)**		
TBD	TDIS	Discharge Time	0.2	—	μs	

Legend: TBD = To Be Determined

Note 1: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

2: ADRES registers may be read on the following TCY cycle.

3: The time for the holding capacitor to acquire the "New" input voltage when the voltage changes full scale after the conversion (VDD to VSS or VSS to VDD). The source impedance (*RS*) on the input channels is 50Ω.

4: On the following cycle of the device clock.

28.5 Ethernet Specifications and Requirements

TABLE 28-28: REQUIREMENTS FOR ETHERNET TRANSCEIVER EXTERNAL MAGNETICS

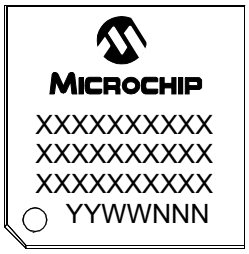
Parameter	Min	Norm	Max	Units	Conditions
RX Turns Ratio	—	1:1	—	—	
TX Turns Ratio	—	1:1	—	—	Transformer Center Tap = 3.3V
Insertion Loss	—	—	-1.1	dB	
Primary Inductance	350	—	—	μH	8 mA bias
Transformer Isolation	1.5	—	—	kVrms	Required to meet IEEE 802.3™ requirements
Differential to Common-Mode Rejection	40	—	—	dB	0.1 to 10 MHz
Return Loss	-16	—	—	dB	

NOTES:

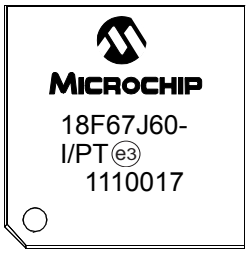
29.0 PACKAGING INFORMATION

29.1 Package Marking Information

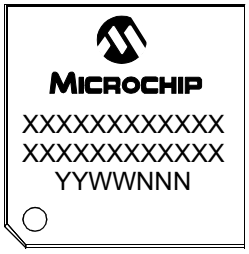
64-Lead TQFP



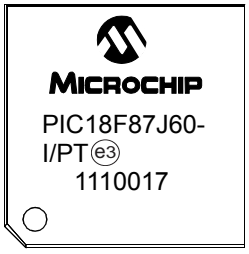
Example



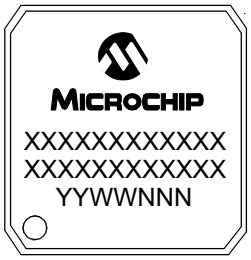
80-Lead TQFP



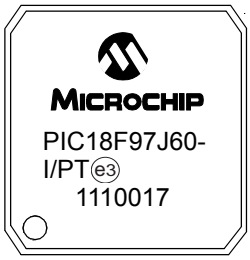
Example



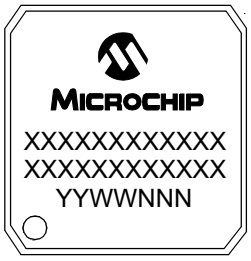
100-Lead TQFP (12x12x1 mm)



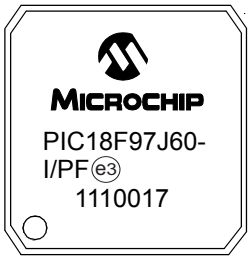
Example



100-Lead TQFP (14x14x1 mm)



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.

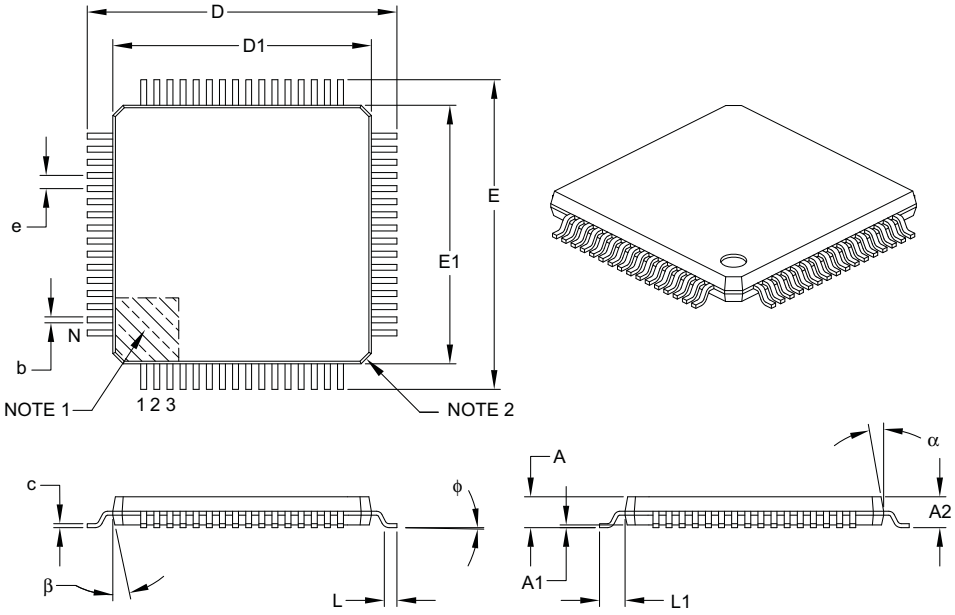
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

29.2 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.

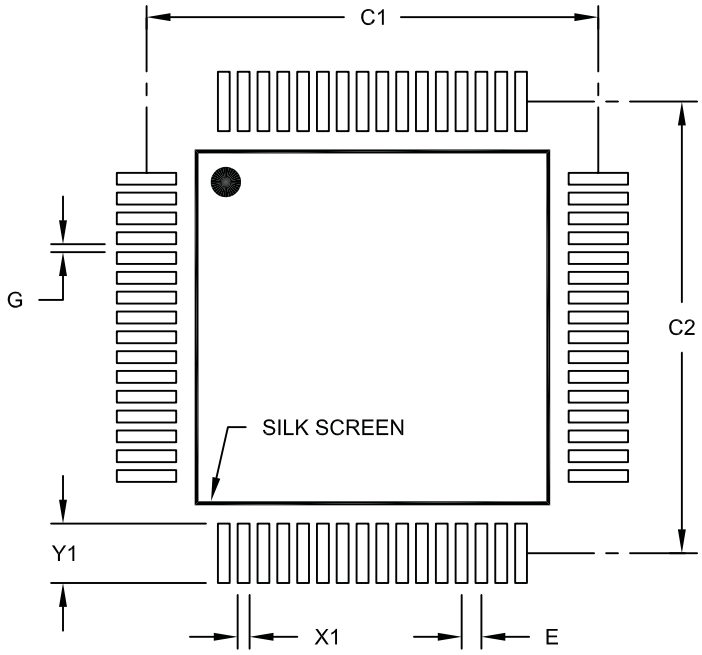
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

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:

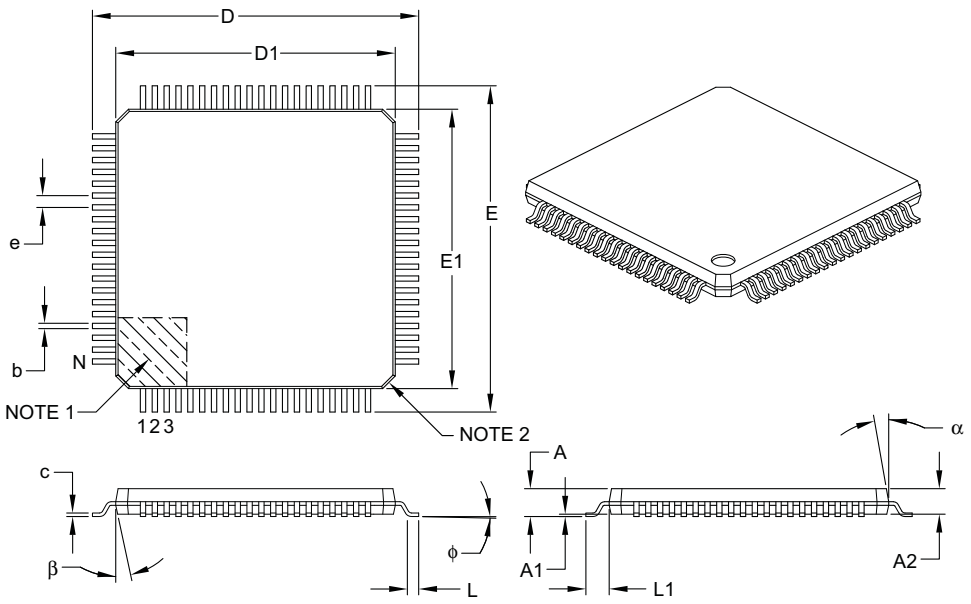
1. Dimensioning and tolerancing per ASME Y14.5M

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.

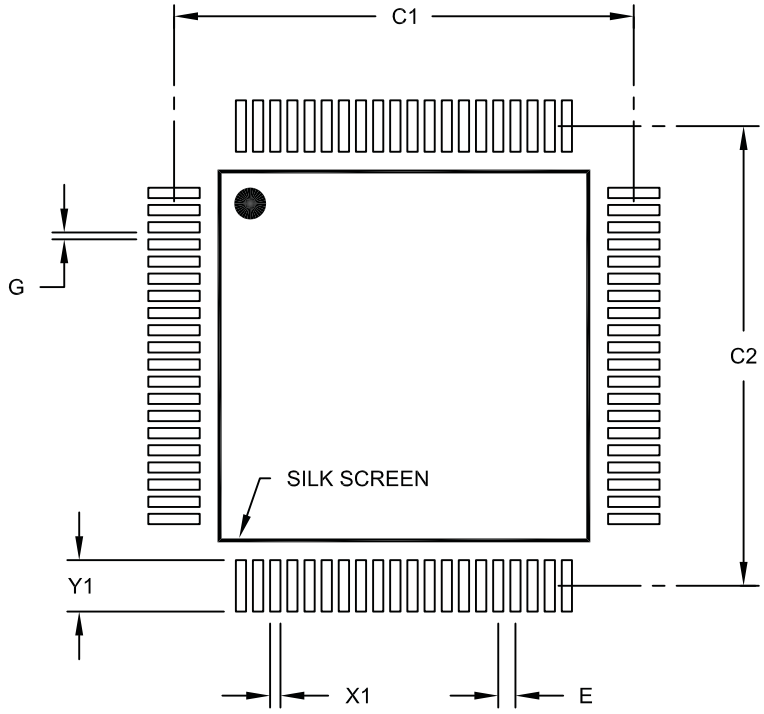
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

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:

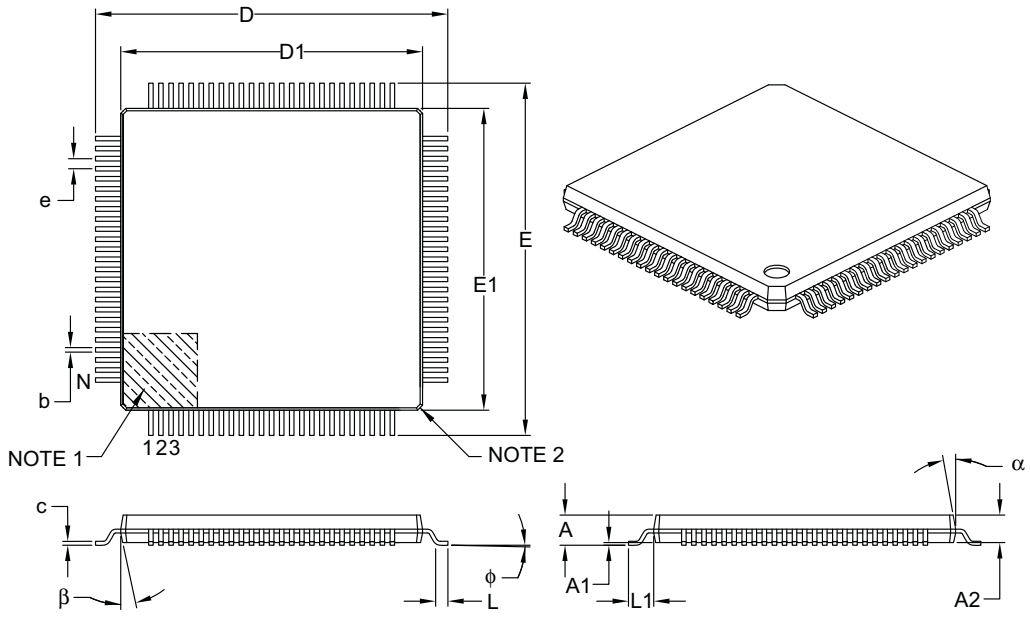
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2092A

100-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		100	
Lead Pitch	e		0.40 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	0.13	0.18	0.23
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

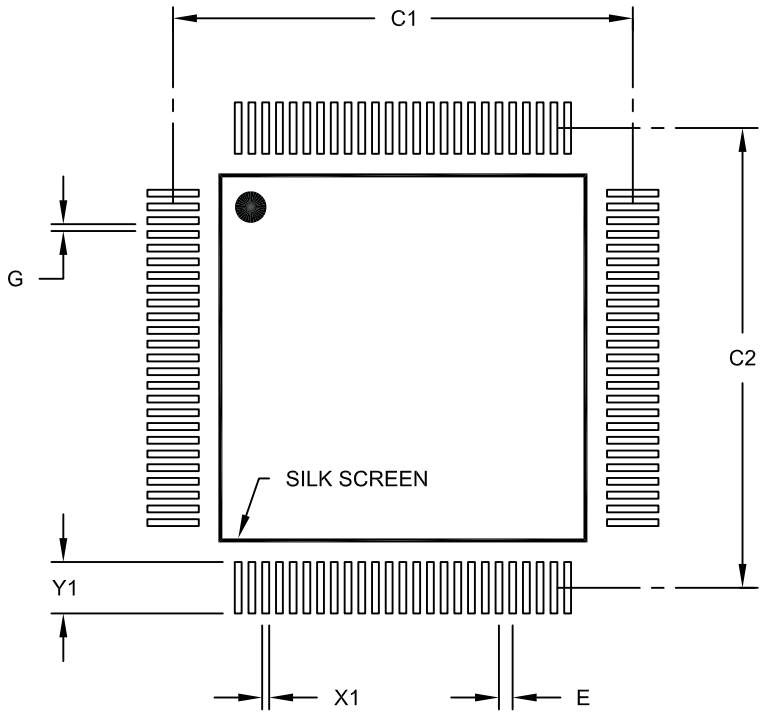
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-100B

100-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		0.40 BSC	
Contact Pad Spacing	C1		13.40	
Contact Pad Spacing	C2		13.40	
Contact Pad Width (X100)	X1			0.20
Contact Pad Length (X100)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

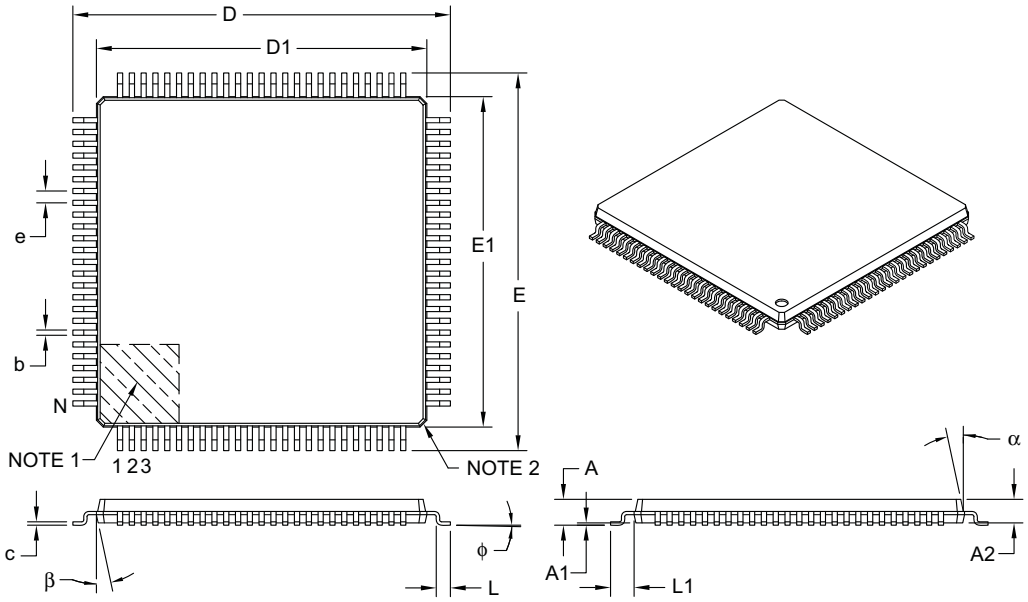
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2100A

100-Lead Plastic Thin Quad Flatpack (PF) – 14x14x1 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		100	
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		16.00 BSC	
Overall Length	D		16.00 BSC	
Molded Package Width	E1		14.00 BSC	
Molded Package Length	D1		14.00 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:

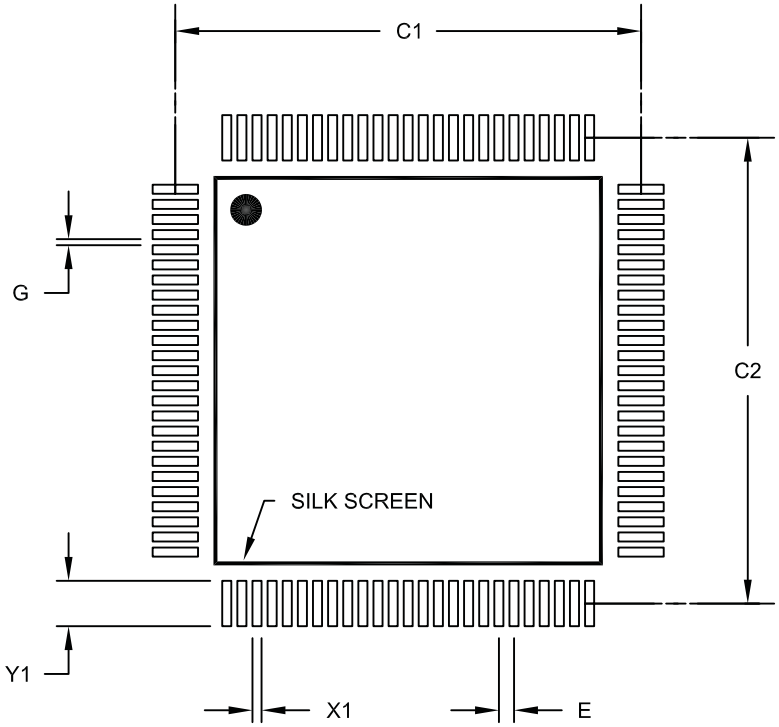
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-110B

100-Lead Plastic Thin Quad Flatpack (PF) – 14x14x1 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		15.40	
Contact Pad Spacing	C2		15.40	
Contact Pad Width (X100)	X1			0.30
Contact Pad Length (X100)	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. C04-2110A

NOTES:

APPENDIX A: REVISION HISTORY

Revision A (March 2006)

Original data sheet for the PIC18F97J60 family of devices.

Revision B (October 2006)

First revision. Includes preliminary electrical specifications; revised and updated material on the Ethernet module; updated material on Reset integration; and updates to the device memory map.

Revision C (June 2007)

Corrected Table 10.2: Input Voltage Levels; added content on Ethernet module's reading and writing to the buffer; added new, 100-lead PT 12x12x1 mm TQFP package to "Package Marking Information" and "Package Details" sections; updated other package details drawings; changed Product Identification System examples.

Revision D (January 2008)

Added one line to "Ethernet Features" description. Added land pattern schematics for each package.

Revision E (October 2009)

Updated to remove Preliminary status.

Revision F (April 2011)

Added Brown-out Reset (BOR) specs, added Ethernet RX Auto-Polarity circuit section, added EMI filter section, added **Section 2.0 "Guidelines for Getting Started with PIC18FJ Microcontrollers"**, changed the opcode encoding of the PUSHL instruction to 1110 1010 kkk kkkk and changed the 2 TOSC Maximum Device Frequency in Table 22-1 from 2.68 MHz to the correct value of 2.86 MHz. Updated comparator input offset voltage maximum to the correct value of 25 mV.

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

TABLE B-1: DEVICE DIFFERENCES BETWEEN PIC18F97J60 FAMILY MEMBERS

Features	PIC18F66J60	PIC18F66J65	PIC18F67J60	PIC18F86J60	PIC18F86J65	PIC18F87J60	PIC18F96J60	PIC18F96J65	PIC18F97J60
Program Memory (Bytes)	64K	96K	128K	64K	96K	128K	64K	96K	128K
Program Memory (Instructions)	32764	49148	65532	32764	49148	65532	32764	49148	65532
Interrupt Sources	26			27			29		
I/O Ports (Pins)	Ports A, B, C, D, E, F, G (39)			Ports A, B, C, D, E, F, G, H, J (55)			Ports A, B, C, D, E, F, G, H, J (70)		
Enhanced USART Modules	1			2					
MSSP Modules	1						2		
Parallel Slave Port Communications (PSP)	No						Yes		
External Memory Bus	No						Yes		
10-Bit Analog-to-Digital Module	11 input channels			15 input channels			16 input channels		
Packages	64-pin TQFP			80-pin TQFP			100-pin TQFP		

INDEX

A

B

C

D

E

Q

R

S

V

W

X

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", 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://microchip.com/support

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: PIC18F97J60 Family Literature Number: DS39762F

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	PIC18F66J60/66J65/67J60, PIC18F86J60/86J65/87J60, PIC18F96J60/96J65/97J60, PIC18F66J60/66J65/67J60T(1), PIC18F86J60/86J65/87J60T(1), PIC18F96J60/96J65/97J60T(1)
Temperature Range	I = -40°C to +85°C (Industrial)
Package	PT = 64, 80 and 100-Lead, 12x12x1 mm TQFP (Thin Quad Flatpack) PF = 100-Lead, 14x14x1 mm TQFP
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)

Examples:

a) PIC18F67J60-I/PT 301 = Industrial temp., TQFP package, QTP pattern #301.

b) PIC18F67J60T-I/PT = Tape and reel, Industrial temp., TQFP package.

Note 1: 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://www.microchip.com/support
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

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

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: 886-7-213-7830
Fax: 886-7-330-9305

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

02/18/11

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC18F66J60-I/PT PIC18F66J60T-I/PT PIC18F66J65-I/PT PIC18F66J65T-I/PT PIC18F67J60-I/PT PIC18F67J60T-I/PT PIC18F86J60-I/PT PIC18F86J60T-I/PT PIC18F86J65-I/PT PIC18F86J65T-I/PT PIC18F87J60-I/PT PIC18F87J60T-I/PT PIC18F96J60-I/PT PIC18F96J65-I/PT PIC18F96J65T-I/PT PIC18F97J60-I/PF PIC18F97J60T-I/PT PIC18F96J60-I/PF PIC18F96J60T-I/PF PIC18F96J60T-I/PT PIC18F96J65-I/PF PIC18F96J65T-I/PF PIC18F97J60-I/PT PIC18F97J60T-I/PF

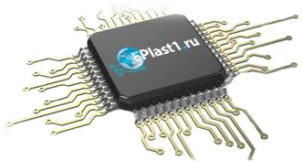
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

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.