PIC18F6520/8520/6620/8620/6720/8720
Data Sheet

64/80-Pin High-Performance, 256 Kbit to 1 Mbit Enhanced Flash Microcontrollers with A/D

© 2004 Microchip Technology Inc.

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

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

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

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

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

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

Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, rfLAB, rfPIC, Select Mode, SmartSensor, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2004, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

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

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

PIC18F6520/8520/6620/8620/6720/8720

64/80-Pin High-Performance, 256 Kbit to 1 Mbit Enhanced Flash Microcontrollers with A/D

High-Performance RISC CPU:

- C compiler optimized architecture/instruction set:
  - Source code compatible with the PIC16 and PIC17 instruction sets
- Linear program memory addressing to 128 Kbytes
- Linear data memory addressing to 3840 bytes
- 1 Kbyte of data EEPROM
- Up to 10 MIPs operation:
  - DC – 40 MHz osc./clock input
  - 4 MHz – 10 MHz osc./clock input with PLL active
- 16-bit wide instructions, 8-bit wide data path
- Priority levels for interrupts
- 31-level, software accessible hardware stack
- 8 x 8 Single Cycle Hardware Multiplier

External Memory Interface (PIC18F8X20 Devices Only):

- Address capability of up to 2 Mbytes
- 16-bit interface

Peripheral Features:

- High current sink/source 25 mA/25 mA
- Four external interrupt pins
- Timer0 module: 8-bit/16-bit timer/counter
- Timer1 module: 16-bit timer/counter
- Timer2 module: 8-bit timer/counter
- Timer3 module: 16-bit timer/counter
- Timer4 module: 8-bit timer/counter
- Secondary oscillator clock option – Timer1/Timer3
- Five Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 6.25 ns (TCY/16)
  - Compare is 16-bit, max. resolution 100 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Master Synchronous Serial Port (MSSP) module with two modes of operation:
  - 3-wire SPI™ (supports all 4 SPI modes)
  - I^2C™ Master and Slave mode
- Two Addressable USART modules:
  - Supports RS-485 and RS-232
- Parallel Slave Port (PSP) module

Analog Features:

- 10-bit, up to 16-channel Analog-to-Digital Converter (A/D):
  - Conversion available during Sleep
- Programmable 16-level Low-Voltage Detection (LVD) module:
  - Supports interrupt on Low-Voltage Detection
- Programmable Brown-out Reset (PBOR)
- Dual analog comparators:
  - Programmable input/output configuration

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- 1 second programming time
- Flash/Data EEPROM Retention: > 40 years
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options including:
  - 4X Phase Lock Loop (of primary oscillator)
  - Secondary Oscillator (32 kHz) clock input
- In-Circuit Serial Programming™ (ICSP™) via two pins
- MPLAB® In-Circuit Debug (ICD) via two pins

CMOS Technology:

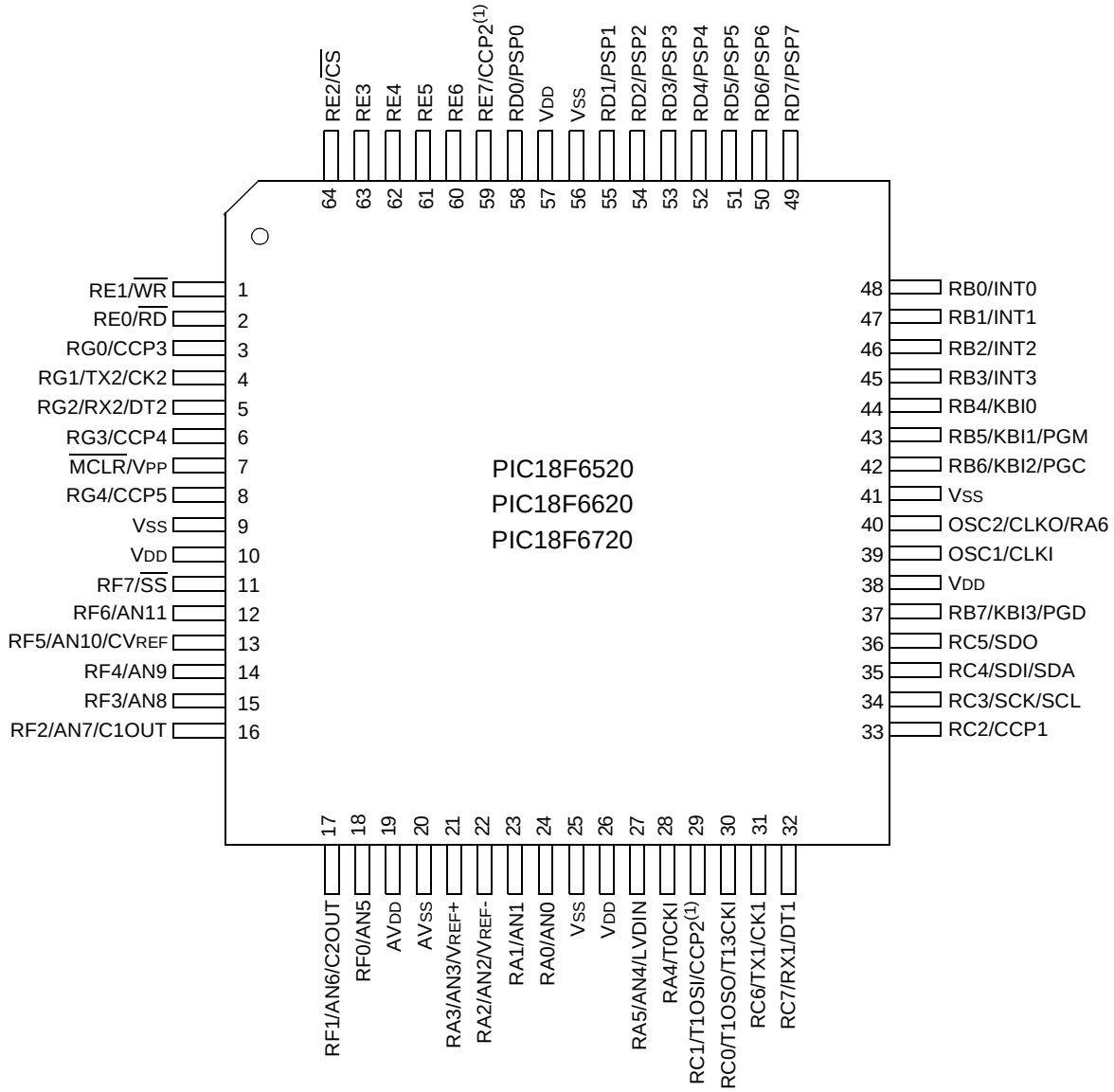
- Low-power, high-speed Flash technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Industrial and Extended temperature ranges

Device	Program Memory		Data Memory		I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8-bit/16-bit	Ext Bus	Max FOSC (MHz)
	Bytes	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI	Master I^2C				
PIC18F6520	32K	16384	2048	1024	52	12	5	Y	Y	2	2/3	N	40
PIC18F6620	64K	32768	3840	1024	52	12	5	Y	Y	2	2/3	N	25
PIC18F6720	128K	65536	3840	1024	52	12	5	Y	Y	2	2/3	N	25
PIC18F8520	32K	16384	2048	1024	68	16	5	Y	Y	2	2/3	Y	40
PIC18F8620	64K	32768	3840	1024	68	16	5	Y	Y	2	2/3	Y	25
PIC18F8720	128K	65536	3840	1024	68	16	5	Y	Y	2	2/3	Y	25

© 2004 Microchip Technology Inc.

Pin Diagrams

64-Pin TQFP

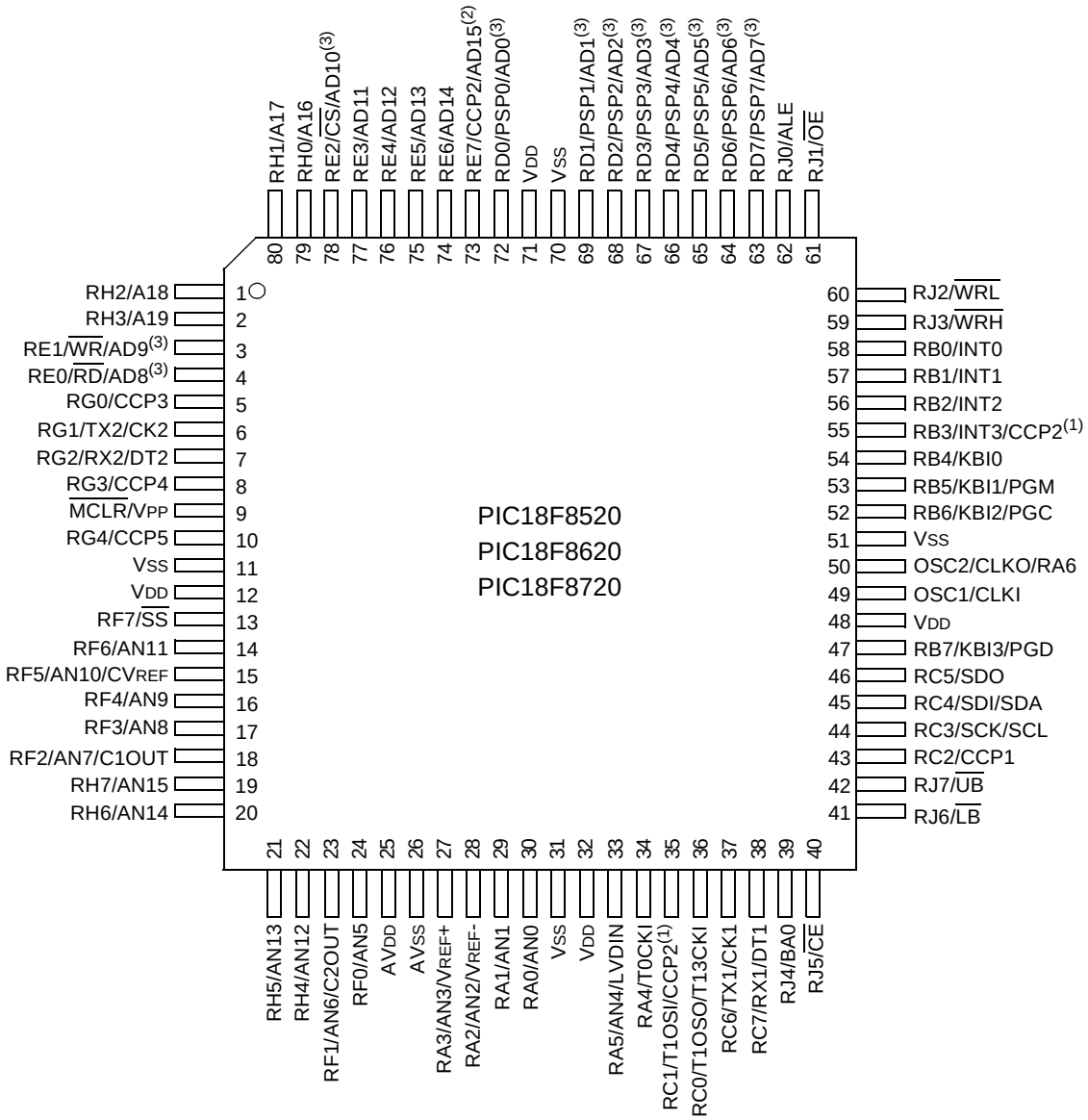


Note 1: CCP2 is multiplexed with RC1 when CCP2MX is set.

© 2004 Microchip Technology Inc.

Pin Diagrams (Continued)

80-Pin TQFP



Note 1: CCP2 is multiplexed with RC1 when CCP2MX is set.
2: CCP2 is multiplexed by default with RE7 when the device is configured in Microcontroller mode.
3: PSP is available only in Microcontroller mode.

Table of Contents

1.0 Device Overview 7
2.0 Oscillator Configurations 21
3.0 Reset 29
4.0 Memory Organization 39
5.0 Flash Program Memory 61
6.0 External Memory Interface 71
7.0 Data EEPROM Memory 79
8.0 8 X 8 Hardware Multiplier 85
9.0 Interrupts 87
10.0 I/O Ports 103
11.0 Timer0 Module 131
12.0 Timer1 Module 135
13.0 Timer2 Module 141
14.0 Timer3 Module 143
15.0 Timer4 Module 147
16.0 Capture/Compare/PWM (CCP) Modules 149
17.0 Master Synchronous Serial Port (MSSP) Module 157
18.0 Addressable Universal Synchronous Asynchronous Receiver Transmitter (USART) 197
19.0 10-Bit Analog-to-Digital Converter (A/D) Module 213
20.0 Comparator Module 223
21.0 Comparator Voltage Reference Module 229
22.0 Low-Voltage Detect 233
23.0 Special Features of the CPU 239
24.0 Instruction Set Summary 259
25.0 Development Support 301
26.0 Electrical Characteristics 307
27.0 DC and AC Characteristics Graphs and Tables 343
28.0 Packaging Information 357
Appendix A: Revision History 361
Appendix B: Device Differences 361
Appendix C: Conversion Considerations 362
Appendix D: Migration from Mid-Range to Enhanced Devices 362
Appendix E: Migration from High-End to Enhanced Devices 363
Index 365
On-Line Support 375
Systems Information and Upgrade Hot Line 375
Reader Response 376
PIC18F6520/8520/6620/8620/6720/8720 Product Identification System 377

© 2004 Microchip Technology Inc.

TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at docerrors@mail.microchip.com 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)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

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

Customer Notification System

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

NOTES:

© 2004 Microchip Technology Inc.

1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC18F6520
- PIC18F8520
- PIC18F6620
- PIC18F8620
- PIC18F6720
- PIC18F8720

This family offers the same advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price – with the addition of high endurance Enhanced Flash program memory. The PIC18FXX20 family also provides an enhanced range of program memory options and versatile analog features that make it ideal for complex, high-performance applications.

1.1 Key Features

1.1.1 EXPANDED MEMORY

The PIC18FXX20 family introduces the widest range of on-chip, Enhanced Flash program memory available on PICmicro® microcontrollers – up to 128 Kbyte (or 65,536 words), the largest ever offered by Microchip. For users with more modest code requirements, the family also includes members with 32 Kbyte or 64 Kbyte.

Other memory features are:

- Data RAM and Data EEPROM: The PIC18FXX20 family also provides plenty of room for application data. Depending on the device, either 2048 or 3840 bytes of data RAM are available. All devices have 1024 bytes of data EEPROM for long-term retention of nonvolatile data.
- Memory Endurance: The Enhanced Flash cells for both program memory and data EEPROM are rated to last for many thousands of erase/write cycles – up to 100,000 for program memory and 1,000,000 for EEPROM. Data retention without refresh is conservatively estimated to be greater than 40 years.

1.1.2 EXTERNAL MEMORY INTERFACE

In the event that 128 Kbytes of program memory is inadequate for an application, the PIC18F8X20 members of the family also implement an External Memory Interface. 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.

With the addition of new operating modes, the External Memory Interface offers many new options, including:

- Operating the microcontroller entirely from external memory
- 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

1.1.3 EASY MIGRATION

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

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

1.1.4 OTHER SPECIAL FEATURES

- Communications: The PIC18FXX20 family incorporates a range of serial communications peripherals, including 2 independent USARTs and a Master SSP module, capable of both SPI and I^2C (Master and Slave) modes of operation. For PIC18F8X20 devices, 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 five Capture/Compare/PWM modules to maximize flexibility in control applications. Up to four different time bases may be used to perform several different operations at once.
- Analog Features: All devices in the family feature 10-bit A/D converters, with up to 16 input channels, as well as the ability to perform conversions during Sleep mode. Also included are dual analog comparators with programmable input and output configuration, a programmable Low-Voltage Detect module and a programmable Brown-out Reset module.
- Self-programmability: These devices can write to their own program memory spaces under internal software control. By using a bootloader routine located in the protected Boot Block at the top of program memory, it becomes possible to create an application that can update itself in the field.

1.2 Details on Individual Family Members

The PIC18FXX20 devices are available in 64-pin and 80-pin packages. They are differentiated from each other in five ways:

1. Flash program memory (32 Kbytes for PIC18FX520 devices, 64 Kbytes for PIC18FX620 devices and 128 Kbytes for PIC18FX720 devices)
2. Data RAM (2048 bytes for PIC18FX520 devices, 3840 bytes for PIC18FX620 and PIC18FX720 devices)
3. A/D channels (12 for PIC18F6X20 devices, 16 for PIC18F8X20)
4. I/O pins (52 on PIC18F6X20 devices, 68 on PIC18F8X20)
5. External program memory interface (present only on PIC18F8X20 devices)

All other features for devices in the PIC18FXX20 family are identical. These are summarized in Table 1-1.

Block diagrams of the PIC18F6X20 and PIC18F8X20 devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2.

TABLE 1-1: PIC18FXX20 DEVICE FEATURES

Features	PIC18F6520	PIC18F6620	PIC18F6720	PIC18F8520	PIC18F8620	PIC18F8720
Operating Frequency	DC – 40 MHz	DC – 25 MHz	DC – 25 MHz	DC – 40 MHz	DC – 25 MHz	DC – 25 MHz
Program Memory (Bytes)	32K	64K	128K	32K	64K	128K
Program Memory (Instructions)	16384	32768	65536	16384	32768	65536
Data Memory (Bytes)	2048	3840	3840	2048	3840	3840
Data EEPROM Memory (Bytes)	1024	1024	1024	1024	1024	1024
External Memory Interface	No	No	No	Yes	Yes	Yes
Interrupt Sources	17	17	17	18	18	18
I/O Ports	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
Timers	5	5	5	5	5	5
Capture/Compare/ PWM Modules	5	5	5	5	5	5
Serial Communications	MSSP, Addressable USART (2)	MSSP, Addressable USART (2)	MSSP, Addressable USART (2)	MSSP, Addressable USART (2)	MSSP, Addressable USART (2)	MSSP, Addressable USART (2)
Parallel Communications	PSP	PSP	PSP	PSP	PSP	PSP
10-bit Analog-to-Digital Module	12 input channels	12 input channels	12 input channels	16 input channels	16 input channels	16 input channels
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes	Yes	Yes
Instruction Set	77 Instructions	77 Instructions	77 Instructions	77 Instructions	77 Instructions	77 Instructions
Package	64-pin TQFP	64-pin TQFP	64-pin TQFP	80-pin TQFP	80-pin TQFP	80-pin TQFP

© 2004 Microchip Technology Inc.

FIGURE 1-1: PIC18F6X20 BLOCK DIAGRAM

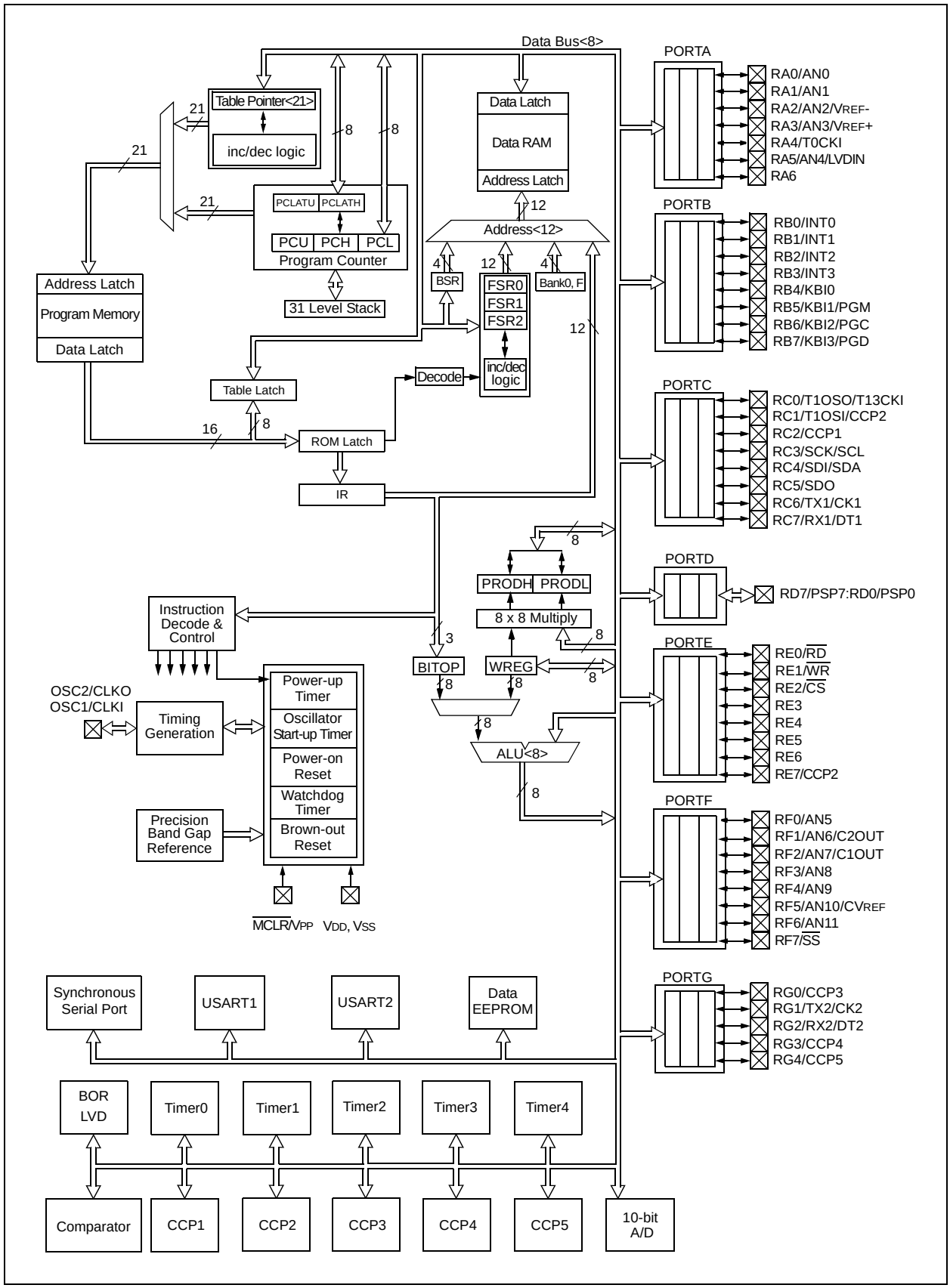
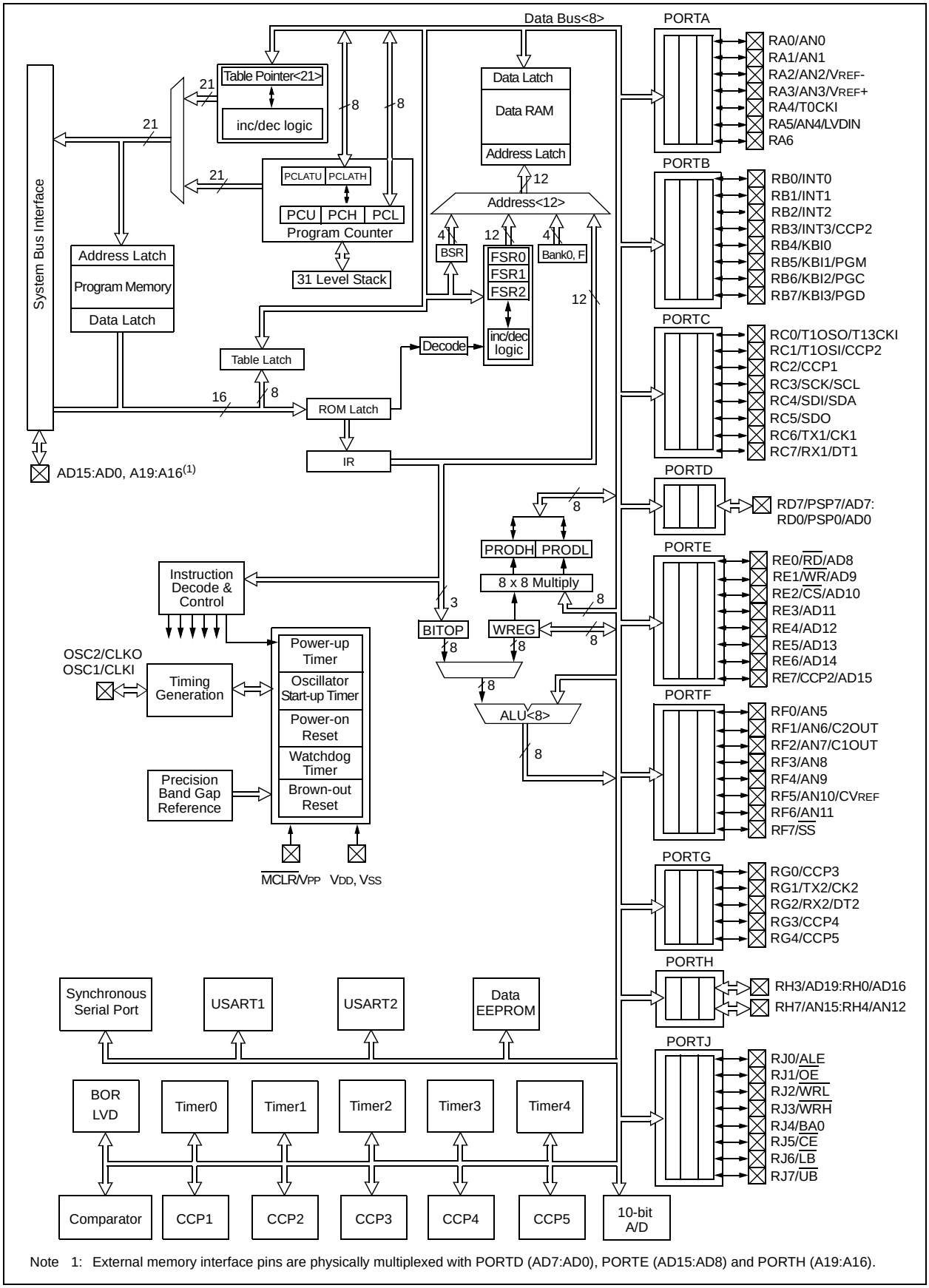


FIGURE 1-2: PIC18F8X20 BLOCK DIAGRAM



Note 1: External memory interface pins are physically multiplexed with PORTD (AD7:AD0), PORTE (AD15:AD8) and PORTH (A19:A16).

© 2004 Microchip Technology Inc.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
$\overline{MCLR}$/VPP	7	9			Master Clear (input) or programming voltage (output).
$\overline{MCLR}$			I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
VPP			P		Programming voltage input.
OSC1/CLKI	39	49			Oscillator crystal or external clock input.
OSC1			I	CMOS/ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS.
CLKI			I	CMOS	External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO/RA6	40	50			Oscillator crystal or clock output.
OSC2			O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	—	In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6			I/O	TTL	General purpose I/O pin.

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTA is a bidirectional I/O port.
RA0/AN0	24	30			
RA0			I/O	TTL	Digital I/O.
AN0			I	Analog	Analog input 0.
RA1/AN1	23	29			
RA1			I/O	TTL	Digital I/O.
AN1			I	Analog	Analog input 1.
RA2/AN2/VREF-	22	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+	21	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	28	34			
RA4			I/O	ST/OD	Digital I/O – Open-drain when configured as output.
T0CKI			I	ST	Timer0 external clock input.
RA5/AN4/LVDIN	27	33			
RA5			I/O	TTL	Digital I/O.
AN4			I	Analog	Analog input 4.
LVDIN			I	Analog	Low-Voltage Detect input.
RA6					See the OSC2/CLKO/RA6 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 CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

© 2004 Microchip Technology Inc.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0	48	58			
RB0			I/O	TTL	Digital I/O.
INT0			I	ST	External interrupt 0.
RB1/INT1	47	57			
RB1			I/O	TTL	Digital I/O.
INT1			I	ST	External interrupt 1.
RB2/INT2	46	56			
RB2			I/O	TTL	Digital I/O.
INT2			I	ST	External interrupt 2.
RB3/INT3/CCP2	45	55			
RB3			I/O	TTL	Digital I/O.
INT3			I/O	ST	External interrupt 3.
CCP2(1)			I/O	ST	Capture2 input, Compare2 output, PWM2 output.
RB4/KBI0	44	54			
RB4			I/O	TTL	Digital I/O.
KBI0			I	ST	Interrupt-on-change pin.
RB5/KBI1/PGM	43	53			
RB5			I/O	TTL	Digital I/O.
KBI1			I	ST	Interrupt-on-change pin.
PGM			I/O	ST	Low-Voltage ICSP Programming enable pin.
RB6/KBI2/PGC	42	52			
RB6			I/O	TTL	Digital I/O.
KBI2			I	ST	Interrupt-on-change pin.
PGC			I/O	ST	In-Circuit Debugger and ICSP programming clock.
RB7/KBI3/PGD	37	47			
RB7			I/O	TTL	Digital I/O.
KBI3			I/O	ST	Interrupt-on-change pin.
PGD					In-Circuit Debugger and ICSP programming data.

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AV_{DD} must be connected to a positive supply and AV_{SS} must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTC is a bidirectional I/O port.
RC0/T1OSO/T13CKI	30	36			
RC0			I/O	ST	Digital I/O.
T1OSO			O	—	Timer1 oscillator output.
T13CKI			I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI/CCP2	29	35			
RC1			I/O	ST	Digital I/O.
T1OSI			I	CMOS	Timer1 oscillator input.
CCP2(2)			I/O	ST	Capture2 input/Compare2 output/ PWM2 output.
RC2/CCP1	33	43			
RC2			I/O	ST	Digital I/O.
CCP1			I/O	ST	Capture1 input/Compare1 output/ PWM1 output.
RC3/SCK/SCL	34	44			
RC3			I/O	ST	Digital I/O.
SCK			I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL			I/O	ST	Synchronous serial clock input/output for I^2C mode.
RC4/SDI/SDA	35	45			
RC4			I/O	ST	Digital I/O.
SDI			I	ST	SPI data in.
SDA			I/O	ST	I^2C data I/O.
RC5/SDO	36	46			
RC5			I/O	ST	Digital I/O.
SDO			O	—	SPI data out.
RC6/TX1/CK1	31	37			
RC6			I/O	ST	Digital I/O.
TX1			O	—	USART 1 asynchronous transmit.
CK1			I/O	ST	USART 1 synchronous clock (see RX1/DT1).
RC7/RX1/DT1	32	38			
RC7			I/O	ST	Digital I/O.
RX1			I	ST	USART 1 asynchronous receive.
DT1			I/O	ST	USART 1 synchronous data (see TX1/CK1).

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

© 2004 Microchip Technology Inc.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTD is a bidirectional I/O port. These pins have TTL input buffers when external memory is enabled.
RD0/PSP0/AD0	58	72			
RD0			I/O	ST	Digital I/O.
PSP0			I/O	TTL	Parallel Slave Port data.
AD0(3)			I/O	TTL	External memory address/data 0.
RD1/PSP1/AD1	55	69			
RD1			I/O	ST	Digital I/O.
PSP1			I/O	TTL	Parallel Slave Port data.
AD1(3)			I/O	TTL	External memory address/data 1.
RD2/PSP2/AD2	54	68			
RD2			I/O	ST	Digital I/O.
PSP2			I/O	TTL	Parallel Slave Port data.
AD2(3)			I/O	TTL	External memory address/data 2.
RD3/PSP3/AD3	53	67			
RD3			I/O	ST	Digital I/O.
PSP3			I/O	TTL	Parallel Slave Port data.
AD3(3)			I/O	TTL	External memory address/data 3.
RD4/PSP4/AD4	52	66			
RD4			I/O	ST	Digital I/O.
PSP4			I/O	TTL	Parallel Slave Port data.
AD4(3)			I/O	TTL	External memory address/data 4.
RD5/PSP5/AD5	51	65			
RD5			I/O	ST	Digital I/O.
PSP5			I/O	TTL	Parallel Slave Port data.
AD5(3)			I/O	TTL	External memory address/data 5.
RD6/PSP6/AD6	50	64			
RD6			I/O	ST	Digital I/O.
PSP6			I/O	TTL	Parallel Slave Port data.
AD6(3)			I/O	TTL	External memory address/data 6.
RD7/PSP7/AD7	49	63			
RD7			I/O	ST	Digital I/O.
PSP7			I/O	TTL	Parallel Slave Port data.
AD7(3)			I/O	TTL	External memory address/data 7.

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 CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AV_{DD} must be connected to a positive supply and AV_{SS} must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTE is a bidirectional I/O port.
RE0/$\overline{RD}$/AD8	2	4			
RE0			I/O	ST	Digital I/O.
$\overline{RD}$			I	TTL	Read control for Parallel Slave Port (see $\overline{WR}$ and $\overline{CS}$ pins).
AD8(3)			I/O	TTL	External memory address/data 8.
RE1/$\overline{WR}$/AD9	1	3			
RE1			I/O	ST	Digital I/O.
$\overline{WR}$			I	TTL	Write control for Parallel Slave Port (see $\overline{CS}$ and $\overline{RD}$ pins).
AD9(3)			I/O	TTL	External memory address/data 9.
RE2/$\overline{CS}$/AD10	64	78			
RE2			I/O	ST	Digital I/O.
$\overline{CS}$			I	TTL	Chip select control for Parallel Slave Port (see $\overline{RD}$ and $\overline{WR}$).
AD10(3)			I/O	TTL	External memory address/data 10.
RE3/AD11	63	77			
RE3			I/O	ST	Digital I/O.
AD11(3)			I/O	TTL	External memory address/data 11.
RE4/AD12	62	76			
RE4			I/O	ST	Digital I/O.
AD12			I/O	TTL	External memory address/data 12.
RE5/AD13	61	75			
RE5			I/O	ST	Digital I/O.
AD13(3)			I/O	TTL	External memory address/data 13.
RE6/AD14	60	74			
RE6			I/O	ST	Digital I/O.
AD14(3)			I/O	TTL	External memory address/data 14.
RE7/CCP2/AD15	59	73			
RE7			I/O	ST	Digital I/O.
CCP2(1,4)			I/O	ST	Capture2 input/Compare2 output/ PWM2 output.
AD15(3)			I/O	TTL	External memory address/data 15.

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 CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

© 2004 Microchip Technology Inc.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTF is a bidirectional I/O port.
RF0/AN5	18	24			
RF0			I/O	ST	Digital I/O.
AN5			I	Analog	Analog input 5.
RF1/AN6/C2OUT	17	23			
RF1			I/O	ST	Digital I/O.
AN6			I	Analog	Analog input 6.
C2OUT			O	ST	Comparator 2 output.
RF2/AN7/C1OUT	16	18			
RF2			I/O	ST	Digital I/O.
AN7			I	Analog	Analog input 7.
C1OUT			O	ST	Comparator 1 output.
RF3/AN8	15	17			
RF1			I/O	ST	Digital I/O.
AN8			I	Analog	Analog input 8.
RF4/AN9	14	16			
RF1			I/O	ST	Digital I/O.
AN9			I	Analog	Analog input 9.
RF5/AN10/CVREF	13	15			
RF1			I/O	ST	Digital I/O.
AN10			I	Analog	Analog input 10.
CVREF			O	Analog	Comparator VREF output.
RF6/AN11	12	14			
RF6			I/O	ST	Digital I/O.
AN11			I	Analog	Analog input 11.
RF7/$\overline{SS}$	11	13			
RF7			I/O	ST	Digital I/O.
$\overline{SS}$			I	TTL	SPI slave select input.

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTG is a bidirectional I/O port.
RG0/CCP3	3	5			
RG0			I/O	ST	Digital I/O.
CCP3			I/O	ST	Capture3 input/Compare3 output/ PWM3 output.
RG1/TX2/CK2	4	6			
RG1			I/O	ST	Digital I/O.
TX2			O	—	USART 2 asynchronous transmit.
CK2			I/O	ST	USART 2 synchronous clock (see RX2/DT2).
RG2/RX2/DT2	5	7			
RG2			I/O	ST	Digital I/O.
RX2			I	ST	USART 2 asynchronous receive.
DT2			I/O	ST	USART 2 synchronous data (see TX2/CK2).
RG3/CCP4	6	8			
RG3			I/O	ST	Digital I/O.
CCP4			I/O	ST	Capture4 input/Compare4 output/ PWM4 output.
RG4/CCP5	8	10			
RG4			I/O	ST	Digital I/O.
CCP5			I/O	ST	Capture5 input/Compare5 output/ PWM5 output.

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AV_{DD} must be connected to a positive supply and AV_{SS} must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

© 2004 Microchip Technology Inc.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTH is a bidirectional I/O port(5).
RH0/A16	—	79			
RH0			I/O	ST	Digital I/O.
A16			O	TTL	External memory address 16.
RH1/A17	—	80			
RH1			I/O	ST	Digital I/O.
A17			O	TTL	External memory address 17.
RH2/A18	—	1			
RH2			I/O	ST	Digital I/O.
A18			O	TTL	External memory address 18.
RH3/A19	—	2			
RH3			I/O	ST	Digital I/O.
A19			O	TTL	External memory address 19.
RH4/AN12	—	22			
RH4			I/O	ST	Digital I/O.
AN12			I	Analog	Analog input 12.
RH5/AN13	—	21			
RH5			I/O	ST	Digital I/O.
AN13			I	Analog	Analog input 13.
RH6/AN14	—	20			
RH6			I/O	ST	Digital I/O.
AN14			I	Analog	Analog input 14.
RH7/AN15	—	19			
RH7			I/O	ST	Digital I/O.
AN15			I	Analog	Analog input 15.

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 CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTJ is a bidirectional I/O port(5).
RJ0/ALE	—	62			
RJ0			I/O	ST	Digital I/O.
ALE			O	TTL	External memory address latch enable.
RJ1/$\overline{OE}$	—	61			
RJ1			I/O	ST	Digital I/O.
$\overline{OE}$			O	TTL	External memory output enable.
RJ2/$\overline{WRL}$	—	60			
RJ2			I/O	ST	Digital I/O.
$\overline{WRL}$			O	TTL	External memory write low control.
RJ3/$\overline{WRH}$	—	59			
RJ3			I/O	ST	Digital I/O.
$\overline{WRH}$			O	TTL	External memory write high control.
RJ4/BA0	—	39			
RJ4			I/O	ST	Digital I/O.
BA0			O	TTL	External memory Byte Address 0 control.
RJ5/$\overline{CE}$	—	40			
RJ5			I/O	ST	Digital I/O.
$\overline{CE}$			O	TTL	External memory chip enable control.
RJ6/$\overline{LB}$	—	41			
RJ6			I/O	ST	Digital I/O.
$\overline{LB}$			O	TTL	External memory low byte select.
RJ7/$\overline{UB}$	—	42			
RJ7			I/O	ST	Digital I/O.
$\overline{UB}$			O	TTL	External memory high byte select.
VSS	9, 25, 41, 56	11, 31, 51, 70	P	—	Ground reference for logic and I/O pins.
VDD	10, 26, 38, 57	12, 32, 48, 71	P	—	Positive supply for logic and I/O pins.
AVSS(6)	20	26	P	—	Ground reference for analog modules.
AVDD(6)	19	25	P	—	Positive supply for analog modules.

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

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

© 2004 Microchip Technology Inc.

2.0 OSCILLATOR CONFIGURATIONS

2.1 Oscillator Types

The PIC18FXX20 devices can be operated in eight different oscillator modes. The user can program three configuration bits (FOSC2, FOSC1 and FOSC0) to select one of these eight modes:

1. LP Low-Power Crystal
2. XT Crystal/Resonator
3. HS High-Speed Crystal/Resonator
4. HS+PLL High-Speed Crystal/Resonator with PLL enabled
5. RC External Resistor/Capacitor
6. RCIO External Resistor/Capacitor with I/O pin enabled
7. EC External Clock
8. ECIO External Clock with I/O pin enabled

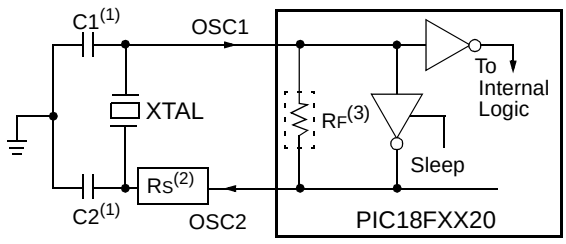
2.2 Crystal Oscillator/Ceramic Resonators

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

The PIC18FXX20 oscillator design requires the use of a parallel cut crystal.

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

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



Note 1: See Table 2-1 and Table 2-2 for recommended values of C1 and C2.

2: A series resistor (RS) may be required for AT strip cut crystals.

3: RF varies with the oscillator mode chosen.

TABLE 2-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Ranges Tested:			
Mode	Freq	C1	C2
XT	455 kHz 2.0 MHz 4.0 MHz	68-100 pF 15-68 pF 15-68 pF	68-100 pF 15-68 pF 15-68 pF
HS	8.0 MHz 16.0 MHz	10-68 pF 10-22 pF	10-68 pF 10-22 pF
These values are for design guidance only. See notes following this table.			
Resonators Used:			
2.0 MHz	Murata Erie CSA2.00MG	± 0.5%	
4.0 MHz	Murata Erie CSA4.00MG	± 0.5%	
8.0 MHz	Murata Erie CSA8.00MT	± 0.5%	
16.0 MHz	Murata Erie CSA16.00MX	± 0.5%	
All resonators used did not have built-in capacitors.			

Note 1: Higher capacitance increases the stability of the oscillator, but also increases the start-up time.

2: When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use high gain HS mode, try a lower frequency resonator, or switch to a crystal oscillator.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components, or verify oscillator performance.

TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Ranges Tested:			
Mode	Freq	C1	C2
LP	32 kHz	15-22 pF	15-22 pF
	200 kHz		
XT	1 MHz	15-22 pF	15-22 pF
	4 MHz		
HS	4 MHz	15-22 pF	15-22 pF
	8 MHz		
	20 MHz		

Capacitor values are for design guidance only.

These capacitors were tested with the above crystal frequencies for basic start-up and operation. These values are not optimized .

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

Note 1: Higher capacitance increases the stability of the oscillator, but also increases the start-up time.

2: When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.

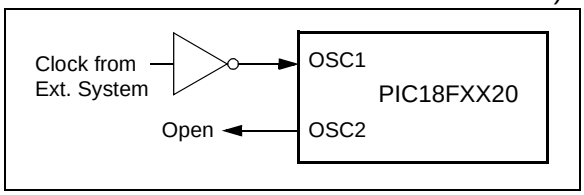
3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components, or verify oscillator performance.

4: RS may be required to avoid overdriving crystals with low drive level specification.

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

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

FIGURE 2-2: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

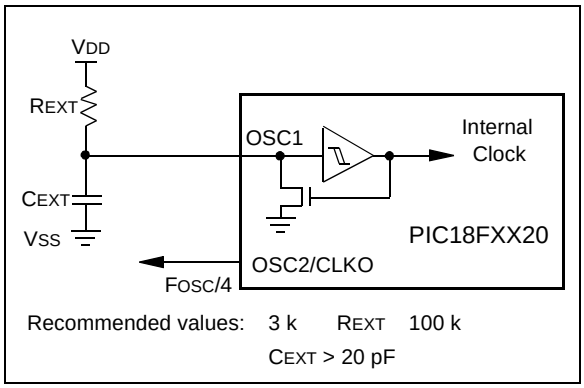


2.3 RC Oscillator

For timing insensitive applications, the “RC” and “RCIO” device options offer additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit, due to normal process parameter variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 2-3 shows how the R/C combination is connected.

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

FIGURE 2-3: RC OSCILLATOR MODE



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

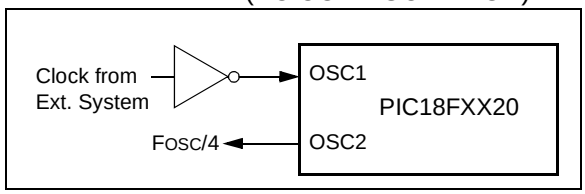
© 2004 Microchip Technology Inc.

2.4 External Clock Input

The EC and ECIO Oscillator modes require an external clock source to be connected to the OSC1 pin. The feedback device between OSC1 and OSC2 is turned off in these modes to save current. There is a maximum 1.5 µs start-up required after a Power-on Reset, or wake-up from Sleep mode.

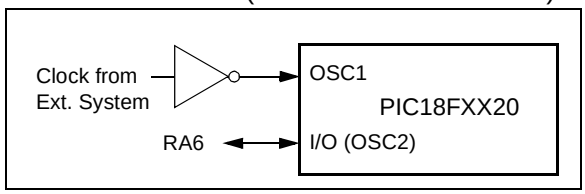
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-4 shows the pin connections for the EC Oscillator mode.

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



The ECIO Oscillator mode functions like the EC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 2-5 shows the pin connections for the ECIO Oscillator mode.

FIGURE 2-5: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)



2.5 HS/PLL

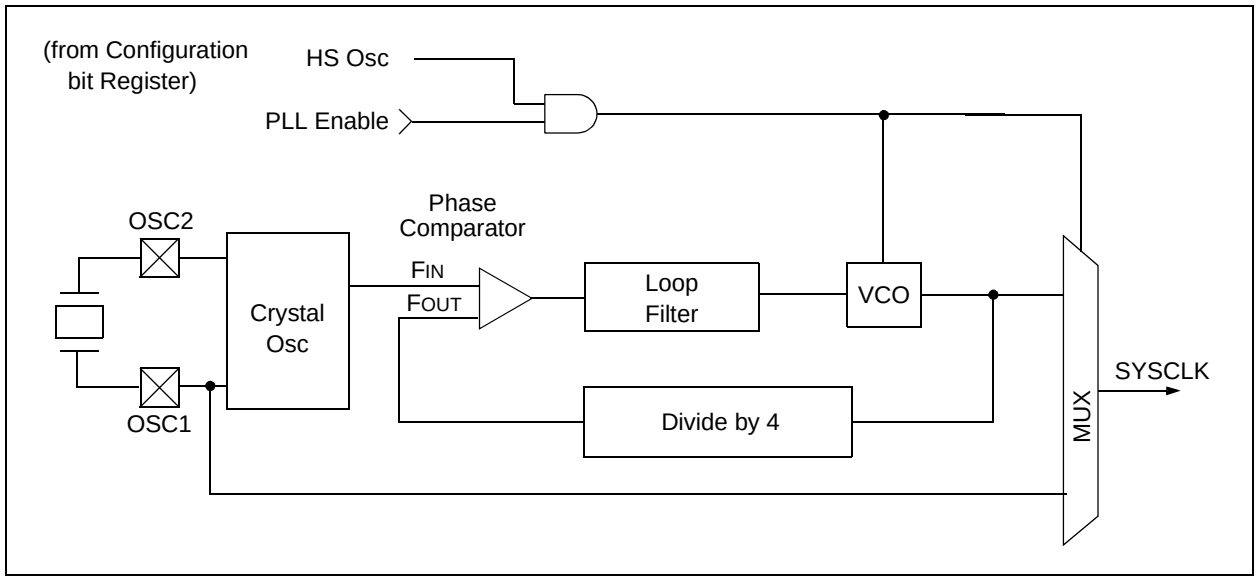
A Phase Locked Loop circuit (PLL) is provided as a programmable option for users that want to multiply the frequency of the incoming crystal oscillator signal by 4. For an input clock frequency of 10 MHz, the internal clock frequency will be multiplied to 40 MHz. This is useful for customers who are concerned with EMI due to high-frequency crystals.

The PLL is one of the modes of the FOSC<2:0> configuration bits. The oscillator mode is specified during device programming.

The PLL can only be enabled when the oscillator configuration bits are programmed for HS mode. If they are programmed for any other mode, the PLL is not enabled and the system clock will come directly from OSC1. Also, PLL operation cannot be changed "on-the-fly". To enable or disable it, the controller must either cycle through a Power-on Reset, or switch the clock source from the main oscillator to the Timer1 oscillator and back again. See Section 2.6 "Oscillator Switching Feature" for details on oscillator switching.

A PLL lock timer is used to ensure that the PLL has locked before device execution starts. The PLL lock timer has a time-out that is called TPLL.

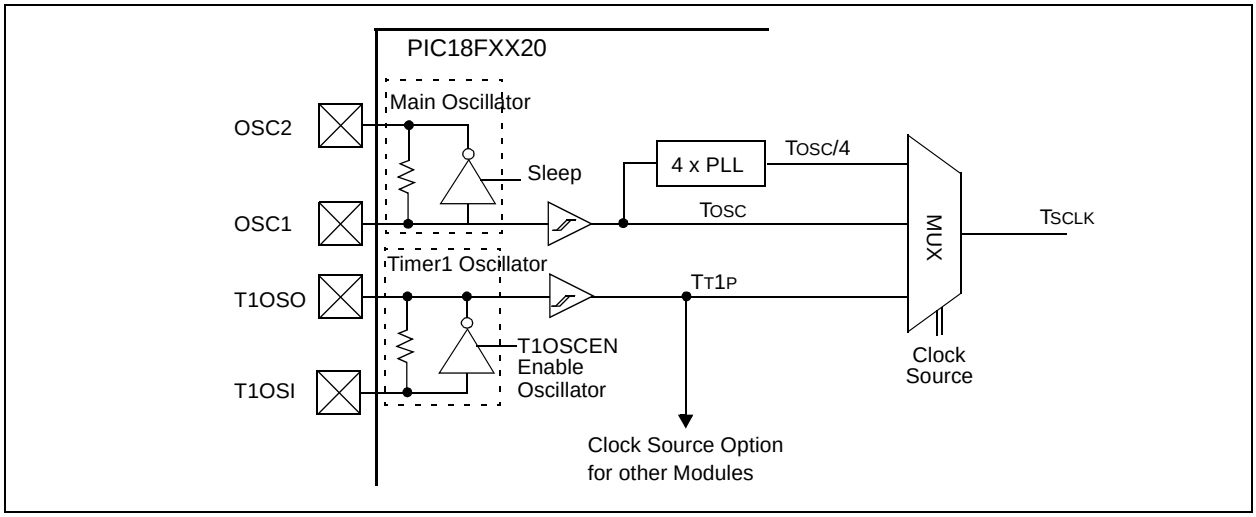
FIGURE 2-6: PLL BLOCK DIAGRAM



2.6 Oscillator Switching Feature

The PIC18FXX20 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low-frequency clock source. For the PIC18FXX20 devices, this alternate clock source is the Timer1 oscillator. If a low-frequency crystal (32 kHz, for example) has been attached to the Timer1 oscillator pins and the Timer1 oscillator has been enabled, the device can switch to a low-power execution mode. Figure 2-7 shows a block diagram of the system clock sources. The clock switching feature is enabled by programming the Oscillator Switching Enable (OSCSEN) bit in Configuration Register 1H to a '0'. Clock switching is disabled in an erased device. See Section 12.0 "Timer1 Module" for further details of the Timer1 oscillator. See Section 23.0 "Special Features of the CPU" for Configuration register details.

FIGURE 2-7: DEVICE CLOCK SOURCES



© 2004 Microchip Technology Inc.

2.6.1 SYSTEM CLOCK SWITCH BIT

The system clock source switching is performed under software control. The system clock switch bit, SCS (OSCCON<0>), controls the clock switching. When the SCS bit is '0', the system clock source comes from the main oscillator that is selected by the FOSC configuration bits in Configuration Register 1H. When the SCS bit is set, the system clock source will come from the Timer1 oscillator. The SCS bit is cleared on all forms of Reset.

Note:	The Timer1 oscillator must be enabled and operating to switch the system clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON). If the Timer1 oscillator is not enabled, then any write to the SCS bit will be ignored (SCS bit forced cleared) and the main oscillator will continue to be the system clock source.

REGISTER 2-1: OSCCON REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-1
—	—	—	—	—	—	—	SCS
bit 7							bit 0

bit 7-1 Unimplemented: Read as '0'

bit 0 SCS: System Clock Switch bit

When $\overline{\text{OSCSEN}}$ Configuration bit = 0 and T1OSCEN bit is set:
1 = Switch to Timer1 oscillator/clock pin
0 = Use primary oscillator/clock input pin

When $\overline{\text{OSCSEN}}$ and T1OSCEN are in other states:
Bit is forced clear.

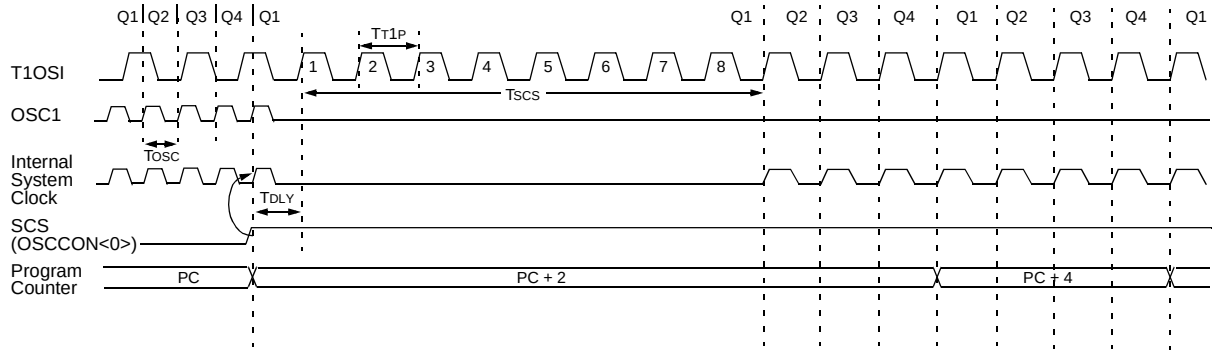
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.6.2 OSCILLATOR TRANSITIONS

PIC18FXX20 devices contain circuitry to prevent “glitches” when switching between oscillator sources. Essentially, the circuitry waits for eight rising edges of the clock source that the processor is switching to. This ensures that the new clock source is stable and that its pulse width will not be less than the shortest pulse width of the two clock sources.

A timing diagram indicating the transition from the main oscillator to the Timer1 oscillator is shown in Figure 2-8. The Timer1 oscillator is assumed to be running all the time. After the SCS bit is set, the processor is frozen at the next occurring Q1 cycle. After eight synchronization cycles are counted from the Timer1 oscillator, operation resumes. No additional delays are required after the synchronization cycles.

FIGURE 2-8: TIMING DIAGRAM FOR TRANSITI ON FROM OSC1 TO TIMER1 OSCILLATOR

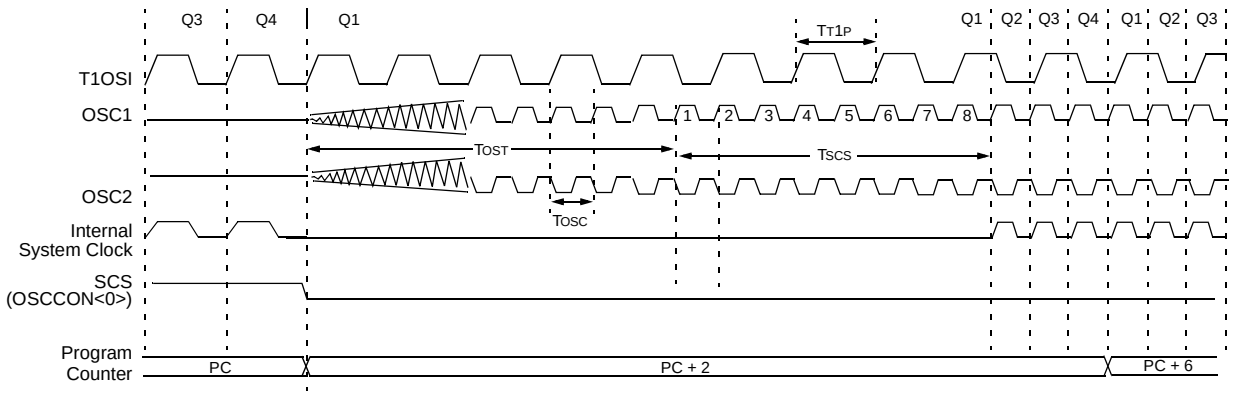


Note 1: Delay on internal system clock is eight oscillator cycles for synchronization.

The sequence of events that takes place when switching from the Timer1 oscillator to the main oscillator will depend on the mode of the main oscillator. In addition to eight clock cycles of the main oscillator, additional delays may take place.

If the main oscillator is configured for an external crystal (HS, XT, LP), then the transition will take place after an oscillator start-up time (TOST) has occurred. A timing diagram, indicating the transition from the Timer1 oscillator to the main oscillator for HS, XT and LP modes, is shown in Figure 2-9.

FIGURE 2-9: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (HS, XT, LP)

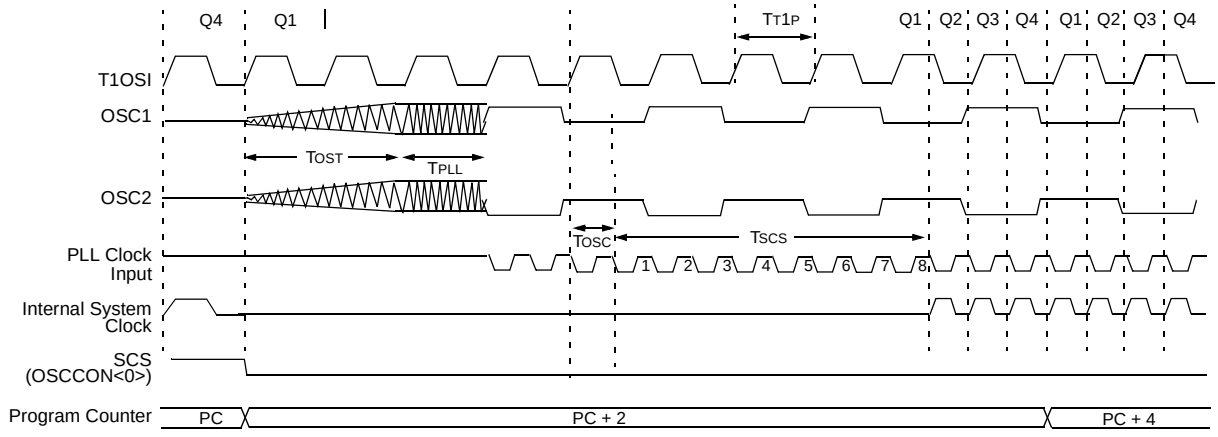


Note 1: TOST = 1024 TOSC (drawing not to scale).

© 2004 Microchip Technology Inc.

If the main oscillator is configured for HS-PLL mode, an oscillator start-up time (TOST), plus an additional PLL time-out (TPLL), will occur. The PLL time-out is typically 2 ms and allows the PLL to lock to the main oscillator frequency. A timing diagram, indicating the transition from the Timer1 oscillator to the main oscillator for HS-PLL mode, is shown in Figure 2-10.

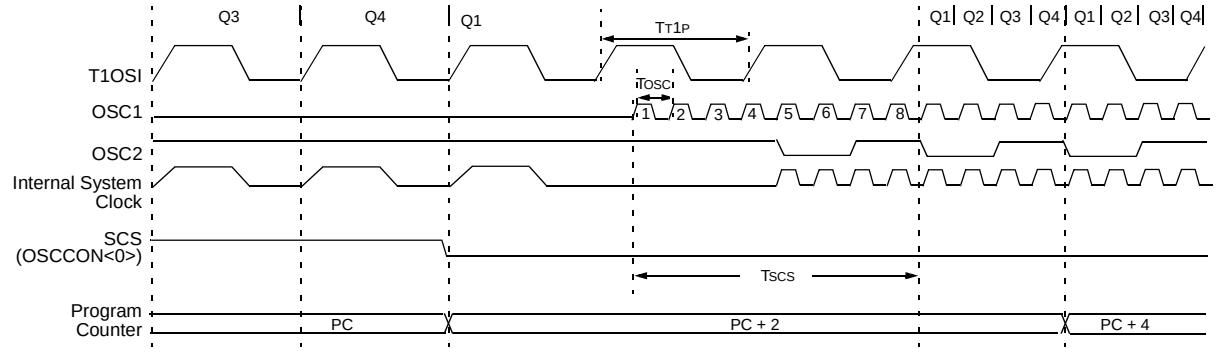
FIGURE 2-10: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (HS WITH PLL)



Note 1: TOST = 1024 TOSC (drawing not to scale).

If the main oscillator is configured in the RC, RCIO, EC or ECIO modes, there is no oscillator start-up time-out. Operation will resume after eight cycles of the main oscillator have been counted. A timing diagram, indicating the transition from the Timer1 oscillator to the main oscillator for RC, RCIO, EC and ECIO modes, is shown in Figure 2-11.

FIGURE 2-11: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (RC, EC)



Note 1: RC Oscillator mode assumed.

2.7 Effects of Sleep Mode on the On-Chip Oscillator

When the device executes a SLEEPinstruction, the on-chip clocks and oscillator are turned off and the device is held at the beginning of an instruction cycle (Q1 state). With the oscillator off, the OSC1 and OSC2 signals will stop oscillating. Since all the transistor switching currents have been removed, 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 user can wake from Sleep through external Reset, Watchdog Timer Reset or through an interrupt.

2.8 Power-up Delays

Power up delays are controlled by two timers so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply and clock are stable. For additional information on Reset operation, see Section 3.0 “Reset” .

The first timer is the Power-up Timer (PWRT), which optionally provides a fixed delay of 72 ms (nominal) on power-up only (POR and BOR). The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable.

With the PLL enabled (HS/PLL Oscillator mode), the time-out sequence following a Power-on Reset is different from other oscillator modes. The time-out sequence is as follows: First, the PWRT time-out is invoked after a POR time delay has expired. Then, the Oscillator Start-up Timer (OST) is invoked. However, this is still not a sufficient amount of time to allow the PLL to lock at high frequencies. The PWRT timer is used to provide an additional fixed 2 ms (nominal) time-out to allow the PLL ample time to lock to the incoming clock frequency.

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

OSC Mode	OSC1 Pin	OSC2 Pin
RC	Floating, external resistor should pull high	At logic low
RCIO	Floating, external resistor should pull high	Configured as PORTA, bit 6
ECIO	Floating	Configured as PORTA, bit 6
EC	Floating	At logic low
LP, XT and HS	Feedback inverter disabled at quiescent voltage level	Feedback inverter disabled at quiescent voltage level

Note: See Table 3-1 in Section 3.0 “Reset” for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

© 2004 Microchip Technology Inc.

3.0 RESET

The PIC18FXX20 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) MCLR Reset during normal operation
c) MCLR Reset during Sleep
d) Watchdog Timer (WDT) Reset (during normal operation)
e) Programmable Brown-out Reset (PBOR)
f) RESETInstruction
g) Stack Full Reset
h) Stack Underflow Reset

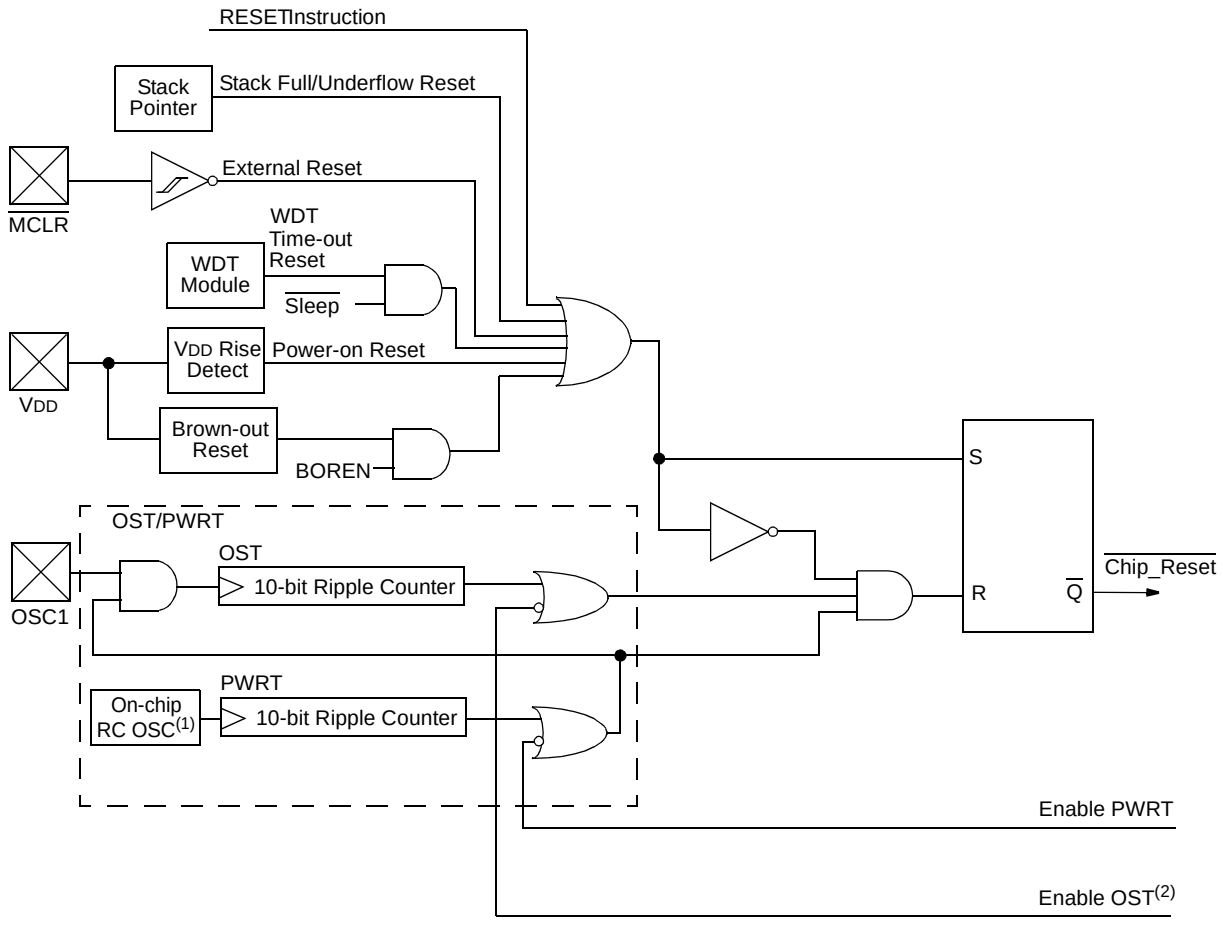
Most registers are unaffected by a Reset. Their status is unknown on POR and unchanged by all other Resets. The other registers are forced to a "Reset state" on Power-on Reset, MCLR, WDT Reset, Brown-out Reset, MCLR Reset during Sleep and by the RESETinstruction.

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, RI, TO, PD, POR and BOR, are set or cleared differently in different Reset situations, as indicated in Table 3-2. These bits are used in software to determine the nature of the Reset. See Table 3-3 for a full description of the Reset states of all registers.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 3-1.

The Enhanced MCU devices have a MCLR noise filter in the MCLR Reset path. The filter will detect and ignore small pulses. The MCLR pin is not driven low by any internal Resets, including the WDT.

FIGURE 3-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



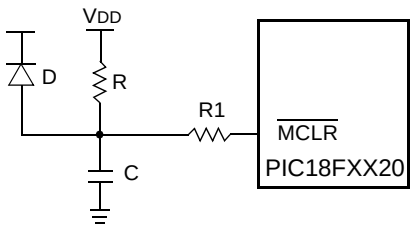
Note 1: This is a separate oscillator from the RC oscillator of the CLKI pin.
2: See Table 3-1 for time-out situations.

3.1 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected. To take advantage of the POR circuitry, tie the MCLR pin through a 1 k to 10 k resistor 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 3-2.

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.

FIGURE 3-2: 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 k is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 = 1 k to 10 k will limit any current flowing into MCLR from external capacitor C, in the event of MCLR/VPP pin breakdown due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

3.2 Power-up Timer (PWRT)

The Power-up Timer provides a fixed nominal time-out (parameter #33) only on power-up from the POR. The Power-up Timer operates on an internal RC oscillator. The chip is kept in Reset as long as the PWRT is active. The PWRT's time delay allows VDD to rise to an acceptable level. A configuration bit is provided to enable/disable the PWRT.

The power-up time delay will vary from chip-to-chip due to VDD, temperature and process variation. See DC parameter #33 for details.

3.3 Oscillator Start-up Timer (OST)

The Oscillator Start-up Timer (OST) provides 1024 oscillator cycles (from OSC1 input) delay after the PWRT delay is over (parameter #32). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP and HS modes and only on Power-on Reset, or wake-up from Sleep.

3.4 PLL Lock Time-out

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

3.5 Brown-out Reset (BOR)

A configuration bit, BOREN, can disable (if clear/programmed), or enable (if set) the Brown-out Reset circuitry. If VDD falls below parameter D005 for greater than parameter #35, the brown-out situation will reset the chip. A Reset may not occur if VDD falls below parameter D005 for less than parameter #35. The chip will remain in Brown-out Reset until VDD rises above BVDD. If the Power-up Timer is enabled, it will be invoked after VDD rises above BVDD; it then will keep the chip in Reset for an additional time delay (parameter #33). If VDD drops below BVDD 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 BVDD, the Power-up Timer will execute the additional time delay.

3.6 Time-out Sequence

On power-up, the time-out sequence is as follows: First, PWRT time-out is invoked after the POR time delay has expired. Then, OST is activated. The total time-out will vary based on oscillator configuration and the status of the PWRT. For example, in RC mode with the PWRT disabled, there will be no time-out at all. Figures 3-3 through 3-7 depict time-out sequences on power-up.

Since the time-outs occur from the POR pulse, the time-outs will expire if MCLR is kept low long enough. Bringing MCLR high will begin execution immediately (Figure 3-5). This is useful for testing purposes, or to synchronize more than one PIC18FXX20 device operating in parallel.

Table 3-2 shows the Reset conditions for some Special Function Registers, while Table 3-3 shows the Reset conditions for all of the registers.

© 2004 Microchip Technology Inc.

TABLE 3-1: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up (2)		Brown-out	Wake-up from Sleep or Oscillator Switch
	$\overline{PWRTE}$ = 0	$\overline{PWRTE}$ = 1		
HS with PLL enabled(1)	72 ms + 1024 TOSC + 2ms	1024 TOSC + 2 ms	72 ms(2) + 1024 TOSC + 2 ms	1024 TOSC + 2 ms
HS, XT, LP	72 ms + 1024 TOSC	1024 TOSC	72 ms(2) + 1024 TOSC	1024 TOSC
EC	72 ms	1.5 µs	72 ms(2)	1.5 µs(3)
External RC	72 ms	—	72 ms(2)	—

Note 1: 2 ms is the nominal time required for the 4xPLL to lock.

2: 72 ms is the nominal power-up timer delay, if implemented.

3: 1.5 µs is the recovery time from Sleep. There is no recovery time from oscillator switch.

REGISTER 3-1: RCON REGISTER BITS AND POSITIONS

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

Note 1: Refer to Section 4.14 “RCON Register” for bit definitions.

TABLE 3-2: STATUS BITS, THEIR SIGNIFICANC E AND THE INITIALIZATION CONDITION FOR RCON REGISTER

Condition	Program Counter	RCON Register	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
Power-on Reset	0000h	0--1 1100	1	1	1	0	0	u	u
$\overline{MCLR}$ Reset during normal operation	0000h	0--u uuuu	u	u	u	u	u	u	u
Software Reset during normal operation	0000h	0--0 uuuu	0	u	u	u	u	u	u
Stack Full Reset during normal operation	0000h	0--u uu11	u	u	u	u	u	u	1
Stack Underflow Reset during normal operation	0000h	0--u uu11	u	u	u	u	u	1	u
$\overline{MCLR}$ Reset during Sleep	0000h	0--u 10uu	u	1	0	u	u	u	u
WDT Reset	0000h	0--u 01uu	1	0	1	u	u	u	u
WDT Wake-up	PC + 2	u--u 00uu	u	0	0	u	u	u	u
Brown-out Reset	0000h	0--1 11u0	1	1	1	1	0	u	u
Interrupt wake-up from Sleep	PC + 2(1)	u--u 00uu	u	1	0	u	u	u	u

Legend: u = unchanged, x = unknown, – = unimplemented bit, read as ‘0’

Note 1: When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (0x000008h or 0x000018h).

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESETInstruction Stack Resets	Wake-up via WDT or Interrupt
TOSU	PIC18F6X20	PIC18F8X20	---0 0000	---0 0000	---0 uuuu (3)
TOSH	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu (3)
TOSL	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu (3)
STKPTR	PIC18F6X20	PIC18F8X20	00-0 0000	uu-0 0000	uu-u uuuu (3)
PCLATU	PIC18F6X20	PIC18F8X20	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6X20	PIC18F8X20	0000 000x	0000 000u	uuuu uuuu (1)
INTCON2	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu (1)
INTCON3	PIC18F6X20	PIC18F8X20	1100 0000	1100 0000	uuuu uuuu (1)
INDF0	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTINC0	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTDEC0	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PREINC0	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PLUSW0	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
FSR0H	PIC18F6X20	PIC18F8X20	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTINC1	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTDEC1	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PREINC1	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PLUSW1	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read '0'.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read '0'.

© 2004 Microchip Technology Inc.

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESETInstruction Stack Resets	Wake-up via WDT or Interrupt
FSR1H	PIC18F6X20	PIC18F8X20	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6X20	PIC18F8X20	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTINC2	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
POSTDEC2	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PREINC2	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
PLUSW2	PIC18F6X20	PIC18F8X20	N/A	N/A	N/A
FSR2H	PIC18F6X20	PIC18F8X20	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F6X20	PIC18F8X20	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6X20	PIC18F8X20	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6X20	PIC18F8X20	---- ---0	---- ---0	---- ---u
LVDCON	PIC18F6X20	PIC18F8X20	--00 0101	--00 0101	--uu uuuu
WDTCON	PIC18F6X20	PIC18F8X20	---- ---0	---- ---0	---- ---u
RCON(4)	PIC18F6X20	PIC18F8X20	0--q 11qq	0--q qquu	u--u qquu
TMR1H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6X20	PIC18F8X20	0-00 0000	u-uu uuuu	u-uu uuuu
TMR2	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6X20	PIC18F8X20	-000 0000	-000 0000	-uuu uuuu
SSPBUF	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
SSPCON1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
SSPCON2	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read ‘0’.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read ‘0’.

TABLE 3-3: INITIALIZATION CONDITI ONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESETInstruction Stack Resets	Wake-up via WDT or Interrupt
ADRESH	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
ADCON1	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6X20	PIC18F8X20	0--- -000	0--- -000	u--- -uuu
CCPR1H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
CCPR2H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
CCPR3H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TMR3H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6X20	PIC18F8X20	0000 0000	uuuu uuuu	uuuu uuuu
PSPCON	PIC18F6X20	PIC18F8X20	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TXSTA1	PIC18F6X20	PIC18F8X20	0000 -010	0000 -010	uuuu -uuu
RCSTA1	PIC18F6X20	PIC18F8X20	0000 000x	0000 000x	uuuu uuuu
EEADRH	PIC18F6X20	PIC18F8X20	---- --00	---- --00	---- --uu
EEADR	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
EEDATA	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
EECON2	PIC18F6X20	PIC18F8X20	---- ----	---- ----	---- ----
EECON1	PIC18F6X20	PIC18F8X20	xx-0 x000	uu-0 u000	uu-0 u000

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read '0'.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read '0'.

© 2004 Microchip Technology Inc.

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESETInstruction Stack Resets	Wake-up via WDT or Interrupt
IPR3	PIC18F6X20	PIC18F8X20	--11 1111	--11 1111	--uu uuuu
PIR3	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
PIE3	PIC18F6X20	PIC18F8X20	--00 0000	--00 0000	--uu uuuu
IPR2	PIC18F6X20	PIC18F8X20	-1-1 1111	-1-1 1111	-u-u uuuu
PIR2	PIC18F6X20	PIC18F8X20	-0-0 0000	-0-0 0000	-u-u uuuu (1)
PIE2	PIC18F6X20	PIC18F8X20	-0-0 0000	-0-0 0000	-u-u uuuu
IPR1	PIC18F6X20	PIC18F8X20	0111 1111	0111 1111	uuuu uuuu
PIR1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu (1)
PIE1	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6X20	PIC18F8X20	0-00 --00	0-00 --00	u-uu --uu
TRISJ	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6X20	PIC18F8X20	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
TRISA(5,6)	PIC18F6X20	PIC18F8X20	-111 1111 (5)	-111 1111 (5)	-uuu uuuu (5)
LATJ	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6X20	PIC18F8X20	---x xxxx	---u uuuu	---u uuuu
LATF	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5,6)	PIC18F6X20	PIC18F8X20	-xxx xxxx (5)	-uuu uuuu (5)	-uuu uuuu (5)

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read '0'.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read '0'.

TABLE 3-3: INITIALIZATION CONDITI ONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESETInstruction Stack Resets	Wake-up via WDT or Interrupt
PORTJ	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6X20	PIC18F8X20	0000 xxxx	0000 uuuu	uuuu uuuu
PORTG	PIC18F6X20	PIC18F8X20	---x xxxx	uuuu uuuu	---u uuuu
PORTF	PIC18F6X20	PIC18F8X20	x000 0000	u000 0000	u000 0000
PORTE	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5,6)	PIC18F6X20	PIC18F8X20	-x0x 0000 (5)	-u0u 0000 (5)	-uuu uuuu (5)
TMR4	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
PR4	PIC18F6X20	PIC18F8X20	1111 1111	1111 1111	uuuu uuuu
T4CON	PIC18F6X20	PIC18F8X20	-000 0000	-000 0000	-uuu uuuu
CCPR4H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR4L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP4CON	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
CCPR5H	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR5L	PIC18F6X20	PIC18F8X20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP5CON	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
SPBRG2	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6X20	PIC18F8X20	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6X20	PIC18F8X20	0000 -010	0000 -010	uuuu -uuu
RCSTA2	PIC18F6X20	PIC18F8X20	0000 000x	0000 000x	uuuu uuuu

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

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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

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

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read ‘0’.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read ‘0’.

© 2004 Microchip Technology Inc.

FIGURE 3-3: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD VIA 1 k RESISTOR)

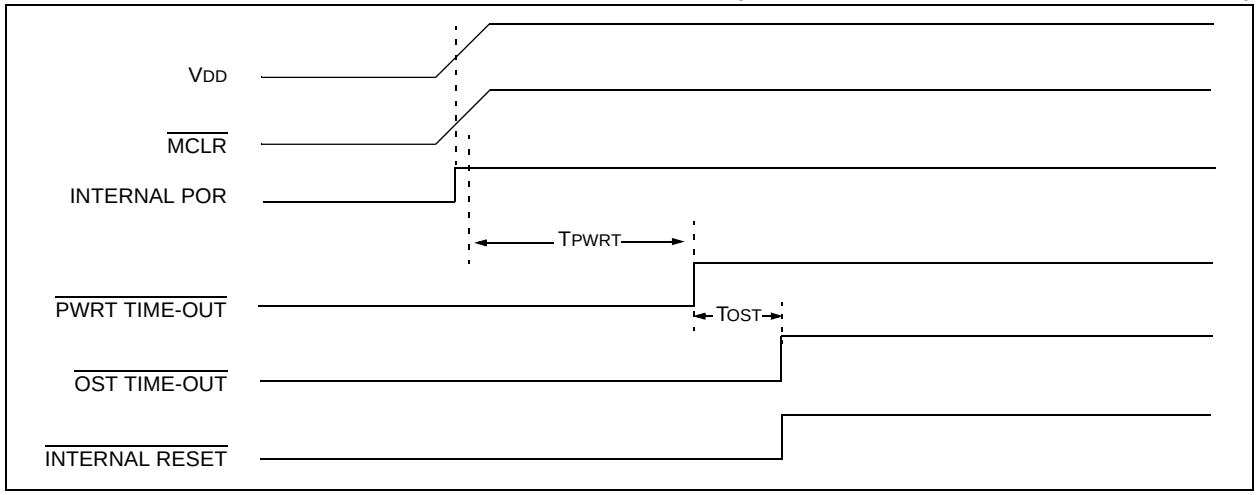


FIGURE 3-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1

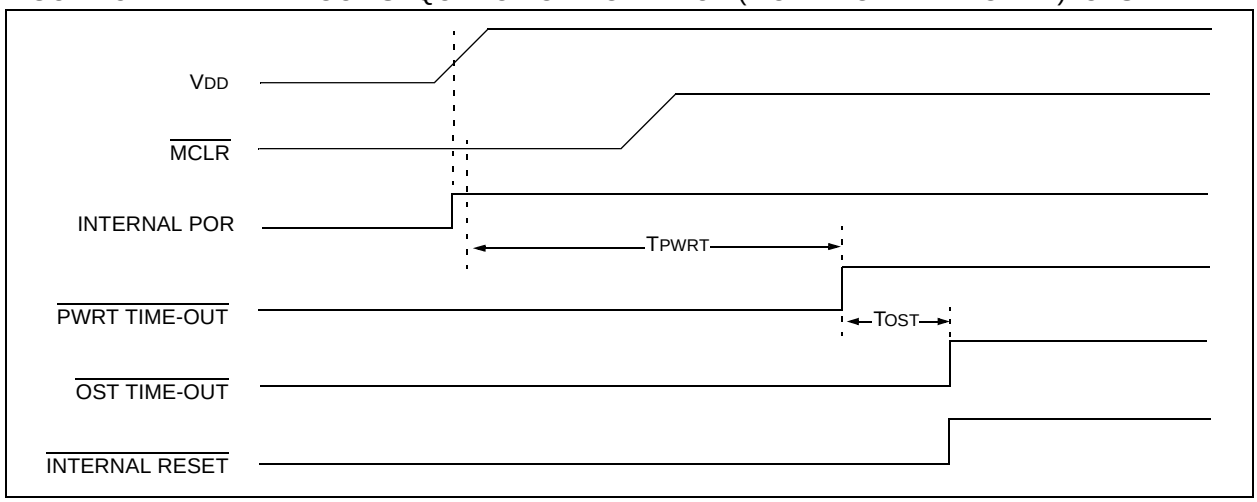


FIGURE 3-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

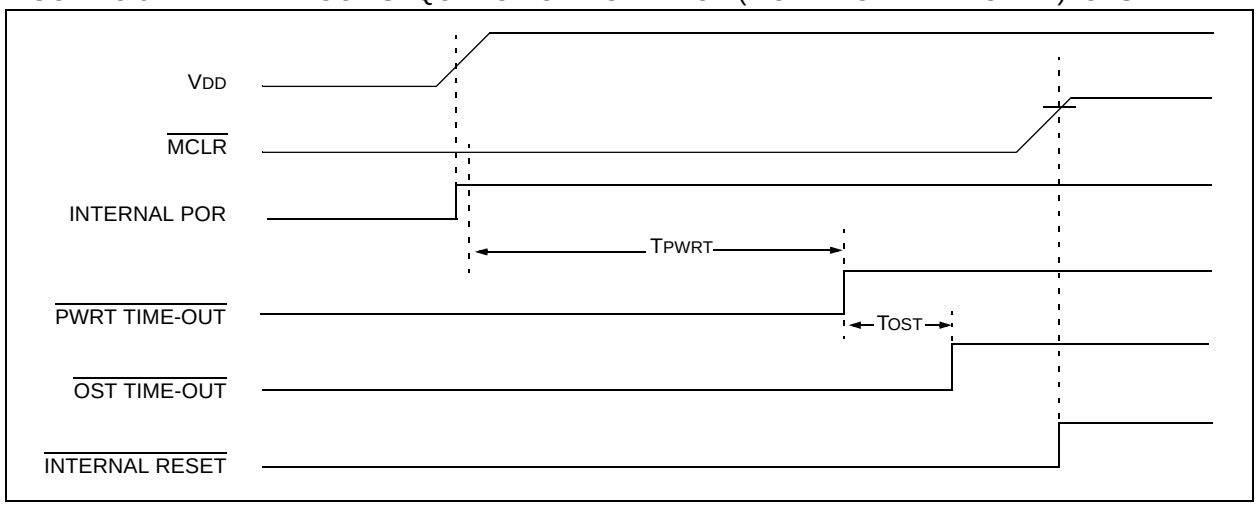


FIGURE 3-6: SLOW RISE TIME ($\overline{MCLR}$ TIED TO V_{DD} VIA 1 k RESISTOR)

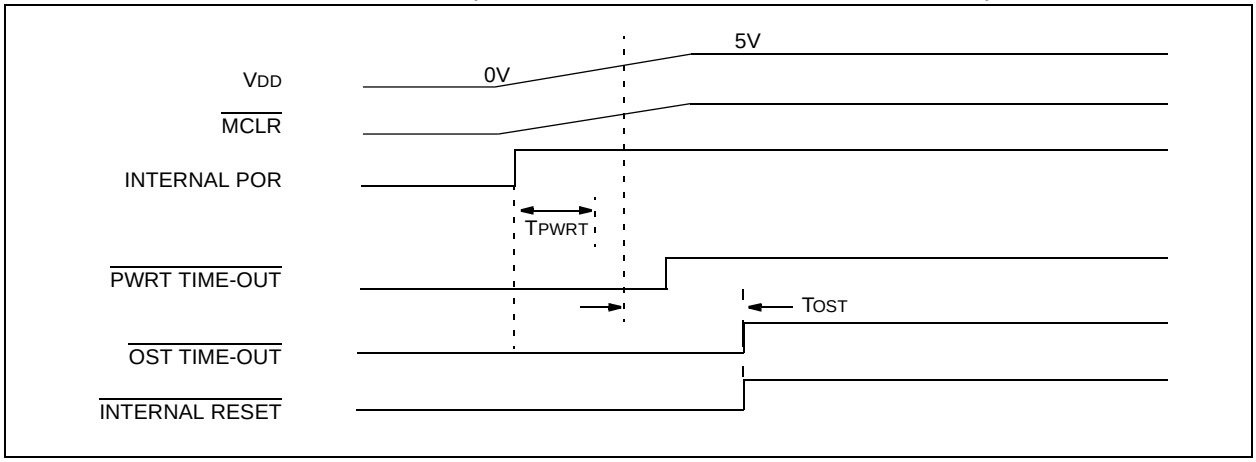
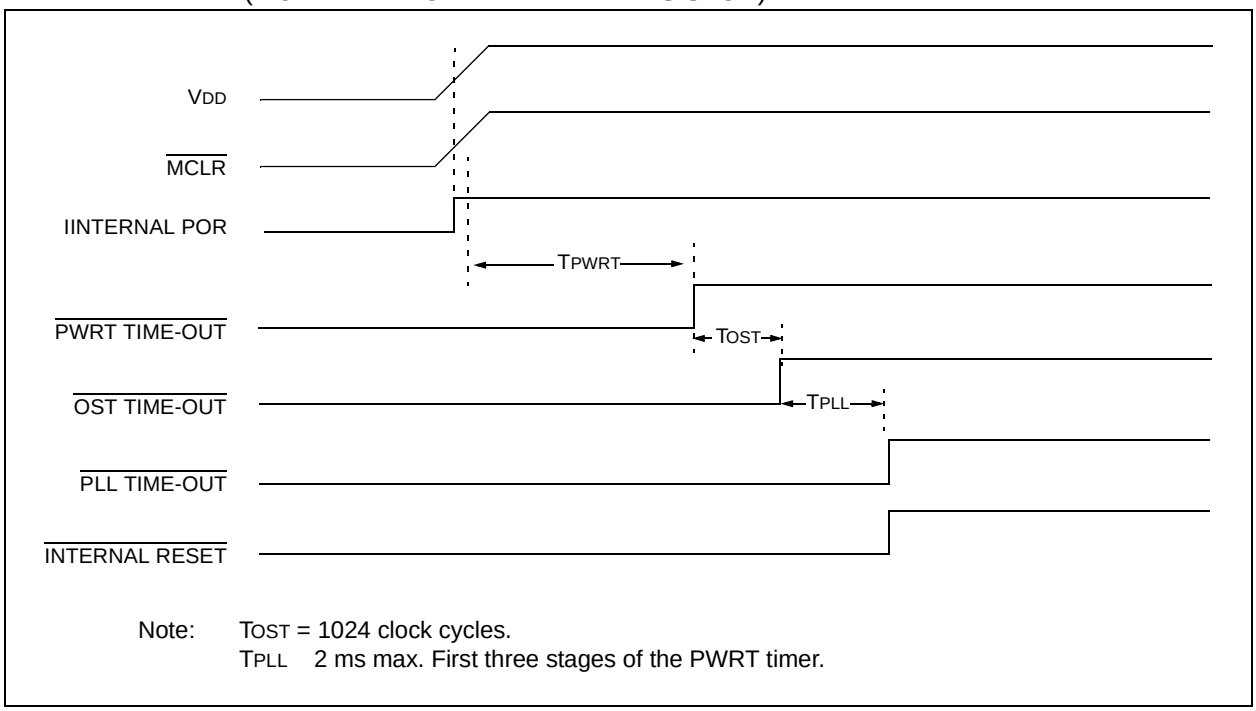


FIGURE 3-7: TIME-OUT SEQUENCE ON POR W/PLL ENABLED (MCLR TIED TO V_{DD} VIA 1 k RESISTOR)



Note: T_{OST} = 1024 clock cycles.
T_{PLL} 2 ms max. First three stages of the PWRT timer.

© 2004 Microchip Technology Inc.

4.0 MEMORY ORGANIZATION

There are three memory blocks in PIC18FXX20 devices. They are:

- Program Memory
- Data RAM
- Data EEPROM

Data and program memory use separate busses, which allows for concurrent access of these blocks. Additional detailed information for Flash program memory and data EEPROM is provided in Section 5.0 “Flash Program Memory” and Section 7.0 “Data EEPROM Memory” , respectively.

In addition to on-chip Flash, the PIC18F8X20 devices are also capable of accessing external program memory through an external memory bus. Depending on the selected operating mode (discussed in Section 4.1.1 “PIC18F8X20 Program Memory Modes”), the controllers may access either internal or external program memory exclusively, or both internal and external memory in selected blocks. Additional information on the External Memory Interface is provided in Section 6.0 “External Memory Interface” .

4.1 Program Memory Organization

A 21-bit program counter is capable of addressing the 2-Mbyte program memory space. Accessing a location between the physically implemented memory and the 2-Mbyte address will cause a read of all ‘0’s (a NOP instruction).

Devices in the PIC18FXX20 family can be divided into three groups, based on program memory size. The PIC18FX520 devices (PIC18F6520 and PIC18F8520) have 32 Kbytes of on-chip Flash memory, equivalent to 16,384 single-word instructions. The PIC18FX620 devices (PIC18F6620 and PIC18F8620) have 64 Kbytes of on-chip Flash memory, equivalent to 32,768 single-word instructions. Finally, the PIC18FX720 devices (PIC18F6720 and PIC18F8720) have 128 Kbytes of on-chip Flash memory, equivalent to 65,536 single-word instructions.

For all devices, the Reset vector address is at 0000h and the interrupt vector addresses are at 0008h and 0018h.

The program memory maps for all of the PIC18FXX20 devices are compared in Figure 4-1.

4.1.1 PIC18F8X20 PROGRAM MEMORY MODES

PIC18F8X20 devices differ significantly from their PIC18 predecessors in their utilization of program memory. In addition to available on-chip Flash program memory, these controllers can also address up to 2 Mbytes of external program memory through the External Memory Interface. There are four distinct operating modes available to the controllers:

- Microprocessor (MP)
- Microprocessor with Boot Block (MPBB)
- Extended Microcontroller (EMC)
- Microcontroller (MC)

The Program Memory mode is determined by setting the two Least Significant bits of the CONFIG3L configuration byte, as shown in Register 4-1. (See also Section 23.1 “Configuration Bits” for additional details on the device configuration bits.)

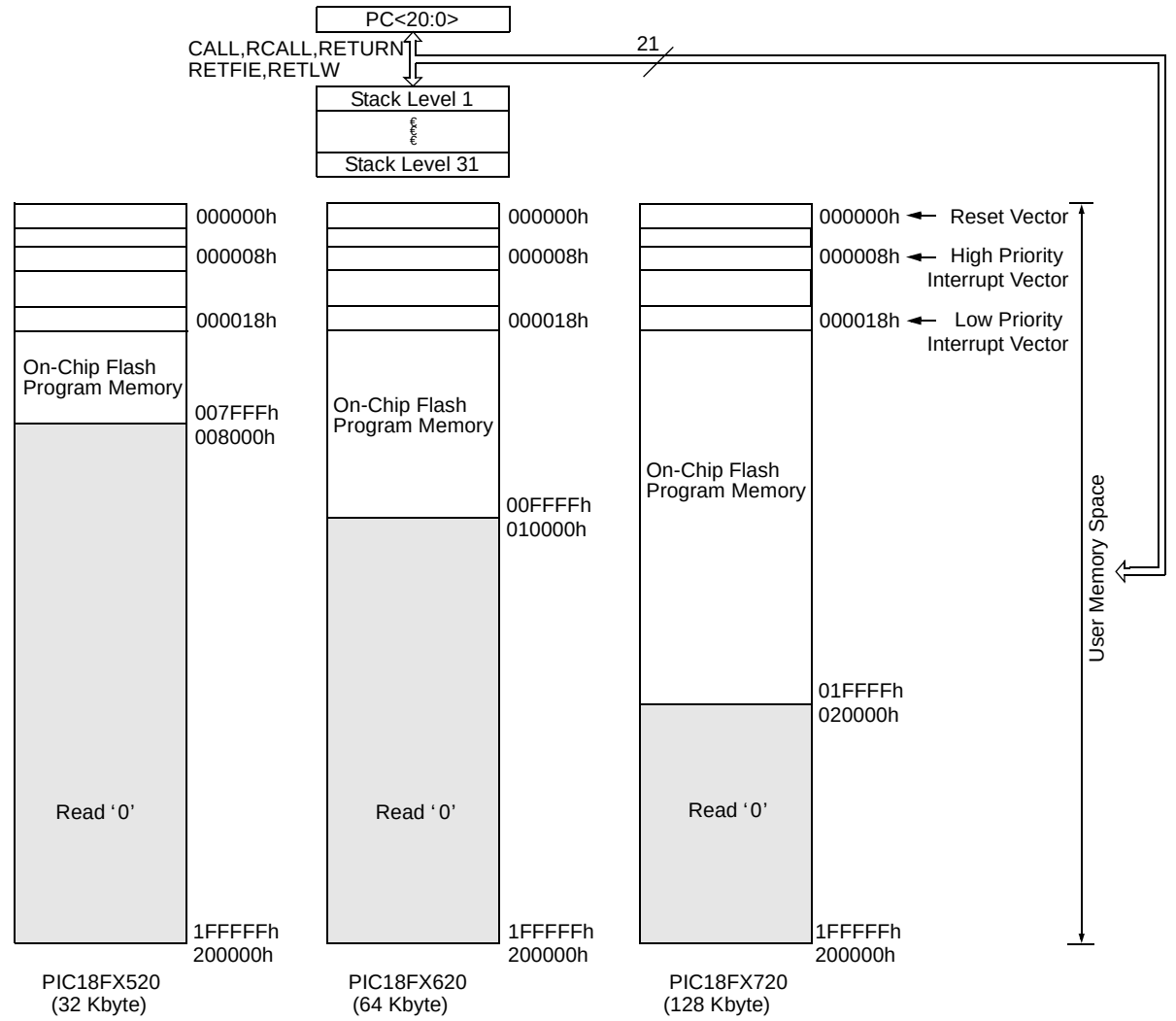
The Program Memory modes operate as follows:

- The Microprocessor Mode permits access only to external program memory; the contents of the on-chip Flash memory are ignored. The 21-bit program counter permits access to a 2-Mbyte linear program memory space.
- The Microprocessor with Boot Block Mode accesses on-chip Flash memory from addresses 000000h to 0007FFh for PIC18F8520 devices and from 000000h to 0001FFh for PIC18F8620 and PIC18F8720 devices. Above this, external program memory is accessed all the way up to the 2-Mbyte limit. Program execution automatically switches between the two memories, as required.
- The Microcontroller Mode accesses only on-chip Flash memory. Attempts to read above the physical limit of the on-chip Flash (7FFFh for the PIC18F8520, 0FFFFh for the PIC18F8620, 1FFFFh for the PIC18F8720) causes a read of all ‘0’s (a NOPinstruction). The Microcontroller mode is also the only operating mode available to PIC18F6X20 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 Flash memory; above this, the device accesses external program memory up to the 2-Mbyte program space limit. As with Boot Block mode, execution automatically switches between the two memories, as required.

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

Figure 4-2 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 4-1.

FIGURE 4-1: INTERNAL PROGRAM MEMORY MAP AND STACK FOR PIC18FXX20 DEVICES



Note: Size of memory regions not to scale.

TABLE 4-1: MEMORY ACCESS FOR PIC18F 8X20 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
Microprocessor	No Access	No Access	No Access	Yes	Yes	Yes
Microprocessor with Boot Block	Yes	Yes	Yes	Yes	Yes	Yes
Microcontroller	Yes	Yes	Yes	No Access	No Access	No Access
Extended Microcontroller	Yes	Yes	Yes	Yes	Yes	Yes

© 2004 Microchip Technology Inc.

REGISTER 4-1: CONFIG3L CONFIGURATION BYTE

R/P-1	U-0	U-0	U-0	U-0	U-0	R/P-1	R/P-1
WAIT	—	—	—	—	—	PM1	PM0
bit 7							bit 0

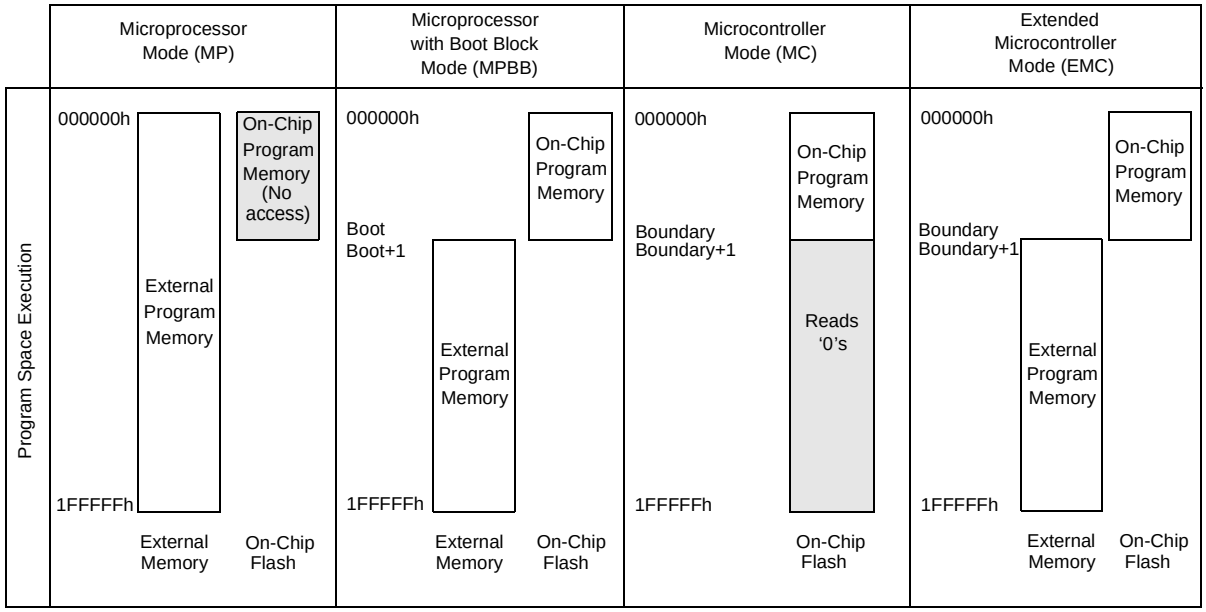
bit 7 WAIT: External Bus Data Wait Enable bit
1 = Wait selections unavailable, device will not wait
0 = Wait programmed by WAIT1 and WAIT0 bits of MEMCOM register (MEMCOM<5:4>)

bit 6-2 Unimplemented: Read as ‘0’

bit 1-0 PM1:PM0: Processor Data Memory Mode Select bits
11 = Microcontroller mode
10 = Microprocessor mode
01 = Microcontroller with Boot Block mode
00 = Extended Microcontroller mode

Legend:			
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’	
- n = Value after erase	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

FIGURE 4-2: MEMORY MAPS FOR PIC18F8X20 PROGRAM MEMORY MODES



Boundary Values for Microprocessor with Boot Block, Microcontroller and Extended Microcontroller modes (1)

Device	Boot	Boot+1	Boundary	Boundary+1	Available Memory Mode(s)
PIC18F6520	0007FFh	000800h	007FFFh	008000h	MC
PIC18F6620	0001FFh	000200h	00FFFFh	010000h	MC
PIC18F6720	0001FFh	000200h	01FFFFh	020000h	MC
PIC18F8520	0007FFh	000800h	007FFFh	008000h	MP, MPBB, MC, EMC
PIC18F8620	0001FFh	000200h	00FFFFh	010000h	MP, MPBB, MC, EMC
PIC18F8720	0001FFh	000200h	01FFFFh	020000h	MP, MPBB, MC, EMC

Note 1: PIC18F6X20 devices are included here for completeness, to show the boundaries of their Boot Blocks and program memory spaces.

4.2 Return Address Stack

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

The stack operates as a 31-word by 21-bit RAM and a 5-bit stack pointer, with the stack pointer initialized to 00000b after all Resets. There is no RAM associated with stack pointer 00000b. This is only a Reset value. During a CALL type instruction, causing a push onto the stack, the stack pointer is first incremented and the RAM location pointed to by the stack pointer is written with the contents of the PC. During a RETURN type instruction, causing a pop from the stack, the contents of the RAM location pointed to by the STKPTR are transferred to the PC and then the stack pointer is decremented.

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 SFR registers. Data can also be pushed to, or popped from the stack using the top-of-stack SFRs. Status bits indicate if the stack pointer is at, or beyond the 31 levels provided.

4.2.1 TOP-OF-STACK ACCESS

The top of the stack is readable and writable. Three register locations, TOSU, TOSH and TOSL, hold the contents of the stack location pointed to by the STKPTR register. This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt, the software can read the pushed value by reading the TOSU, TOSH and TOSL registers. These values can be placed on a user defined software stack. At return time, the software can replace the TOSU, TOSH and TOSL and do a return.

The user must disable the global interrupt enable bits during this time to prevent inadvertent stack operations.

4.2.2 RETURN STACK POINTER (STKPTR)

The STKPTR register contains the stack pointer value, the STKFUL (Stack Full) status bit and the STKUNF (Stack Underflow) status bits. Register 4-2 shows the STKPTR register. The value of the stack pointer can be 0 through 31. The stack pointer increments when values are pushed onto the stack and decrements when values are popped off the stack. At Reset, the stack pointer value will be ‘0’. The user may read and write the stack pointer value. This feature can be used by a Real-Time Operating System 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 can only be cleared in 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 24.0 “Instruction Set Summary” 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 ‘0’.

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

When the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC and sets the STKUNF bit, while the stack pointer remains at ‘0’. The STKUNF bit will remain set until cleared in software or 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.

© 2004 Microchip Technology Inc.

REGISTER 4-2: STKPTR REGISTER

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

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

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

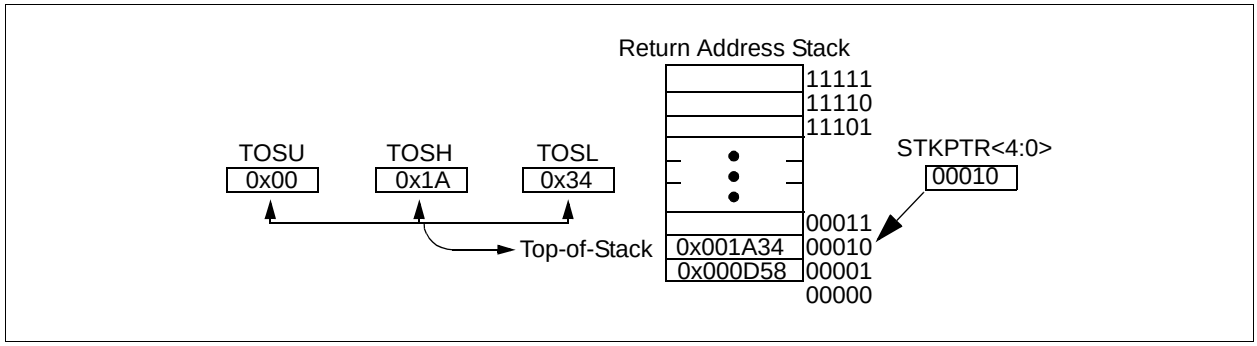
bit 5 Unimplemented: Read as '0'

bit 4-0 SP4:SP0: Stack Pointer Location bits

Note 1: Bit 7 and bit 6 can only be cleared in user software or by a POR.

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

FIGURE 4-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



4.2.3 PUSH AND POP INSTRUCTIONS

Since the Top-of-Stack (TOS) 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 option. To push the current PC value onto the stack, a PUSH instruction can be executed. This will increment the stack pointer and load the current PC value onto the stack. TOSU, TOSH and TOSL can then be modified to place a return address on the stack.

The ability to pull the TOS value off of the stack and replace it with the value that was previously pushed onto the stack, without disturbing normal execution, is achieved by using the POP instruction. The POP instruction discards the current TOS by decrementing the stack pointer. The previous value pushed onto the stack then becomes the TOS value.

4.2.4 STACK FULL/UNDERFLOW RESETS

These Resets are enabled by programming the STVREN configuration bit. When the STVREN bit is disabled, a full or underflow condition will set the appropriate STKFUL or STKUNF bit, but not cause a device Reset. When the STVREN bit is enabled, a full or underflow condition will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. The STKFUL or STKUNF bits are only cleared by the user software or a POR Reset.

4.3 Fast Register Stack

A “fast interrupt return” option is available for interrupts. A Fast Register Stack is provided for the Status, WREG and BSR registers and is only one in depth. The stack is not readable or writable and is loaded with the current value of the corresponding register when the processor vectors for an interrupt. The values in the registers are then loaded back into the working registers, if the FAST RETURNinstruction is used to return from the interrupt.

A low or high priority interrupt source will push values into the stack registers. If both low and high priority interrupts are enabled, the stack registers cannot be used reliably for 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.

If high priority interrupts are not disabled during low priority interrupts, users must save the key registers in software during a low priority 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 FAST CALL instruction must be executed.

Example 4-1 shows a source code example that uses the fast register stack.

EXAMPLE 4-1: 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
```

4.4 PCL, PCLATH and PCLATU

The program counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide. The low byte is called the PCL register; this register is readable and writable. The high byte is called the PCH register. This register contains the PC<15:8> bits and is not directly readable or writable; updates to the PCH register may be performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits and is not directly readable or writable; updates to the PCU register may be performed through the PCLATU register.

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

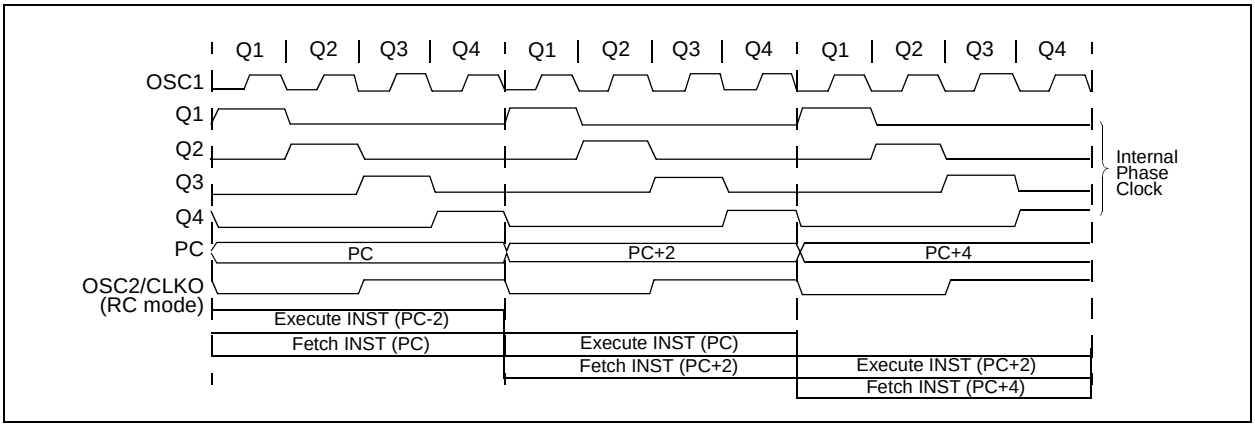
The CALL, RCALL, GOTO and program branch instructions write to the program counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the program counter.

The contents of PCLATH and PCLATU will be transferred to the program counter by an operation that writes PCL. Similarly, the upper two bytes of the program counter will be transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see Section 4.8.1 “Computed GOTO”).

4.5 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the program counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 4-4.

FIGURE 4-4: CLOCK/ INSTRUCTION CYCLE



© 2004 Microchip Technology Inc.

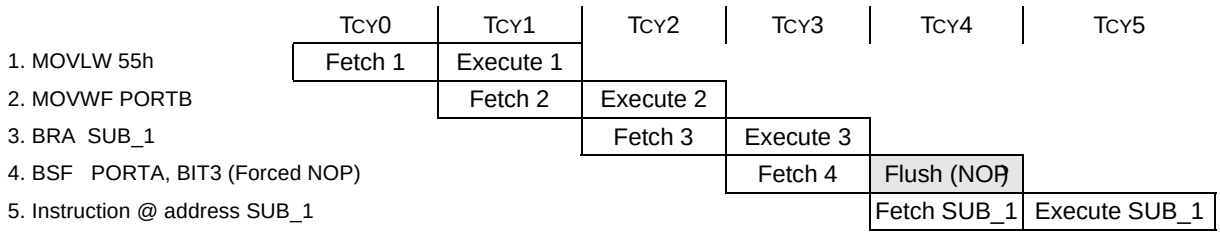
4.6 Instruction Flow/Pipelining

An “Instruction Cycle” consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined, such that fetch takes one instruction cycle, while decode and execute takes 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 4-2).

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).

EXAMPLE 4-2: INSTRUCTION PIPELINE FLOW



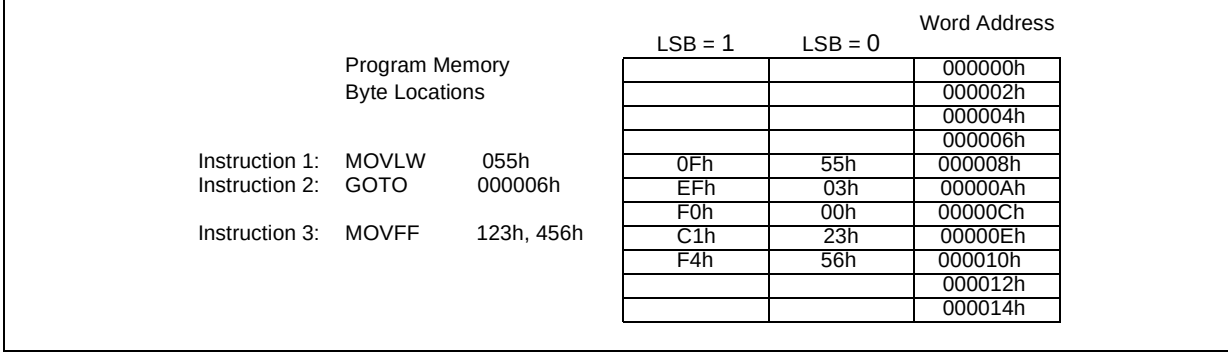
All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is “flushed” from the pipeline, while the new instruction is being fetched and then executed.

4.7 Instructions in Program Memory

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). Figure 4-5 shows an example of how instruction words are stored in the program memory. To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read ‘0’ (see Section 4.4 “PCL, PCLATH and PCLATU”).

The CALL and GOTOinstructions have an 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 4-5 shows how the instruction “GOTO 000006h” is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. Section 24.0 “Instruction Set Summary” provides further details of the instruction set.

FIGURE 4-5: INSTRUCTIONS IN PROGRAM MEMORY



4.7.1 TWO-WORD INSTRUCTIONS

The PIC18FXX20 devices have four two-word instructions: MOVFF, CALL, GOTO and LFSR. The second word of these instructions has the 4 MSBs set to '1's and is a special kind of NOP instruction. The lower 12 bits of the second word contain data to be used by the instruction. If the first word of the instruction is executed, the data in the second word is accessed. If the second word of the instruction is executed by itself (first word was skipped), it will execute as a NOP. This action is necessary when the two-word instruction is preceded by a conditional instruction that changes the PC. A program example that demonstrates this concept is shown in Example 4-3. Refer to Section 24.0 "Instruction Set Summary" for further details of the instruction set.

EXAMPLE 4-3: 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, execute 2-word instruction
1111 0100 0101 0110			; 2nd operand holds address of REG2
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
1111 0100 0101 0110			; 2nd operand becomes NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code

4.8 Look-up Tables

Look-up tables are implemented two ways. These are:

- Computed GOTO
- Table Reads

4.8.1 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL).

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW 0xnn instructions. WREG 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 0xnn instructions, that returns the value 0xnn to the calling function.

The offset value (value in WREG) specifies the number of bytes that the program counter should advance.

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

4.8.2 TABLE READS/TABLE WRITES

A better method of storing data in program memory allows 2 bytes of data to be stored in each instruction location.

Look-up table data may be stored 2 bytes per program word by using table reads and writes. The Table Pointer (TBLPTR) specifies the byte address and the Table Latch (TABLAT) contains the data that is read from, or written to program memory. Data is transferred to/from program memory, one byte at a time.

A description of the table read/table write operation is shown in Section 5.0 "Flash Program Memory" .

© 2004 Microchip Technology Inc.

4.9 Data Memory Organization

The data memory is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4096 bytes of data memory. The data memory map is in turn divided into 16 banks of 256 bytes each. The lower 4 bits of the Bank Select Register (BSR<3:0>) select which bank will be accessed. The upper 4 bits of the BSR are not implemented.

The data memory space contains both Special Function Registers (SFR) and General Purpose Registers (GPR). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratch pad operations in the user's application. The SFRs start at the last location of Bank 15 (0FFFh) and extend downwards. Any remaining space beyond the SFRs in the Bank may be implemented as GPRs. GPRs start at the first location of Bank 0 and grow upwards. Any read of an unimplemented location will read as '0's.

PIC18FX520 devices have 2048 bytes of data RAM, extending from Bank 0 to Bank 7 (000h through 7FFh). PIC18FX620 and PIC18FX720 devices have 3840 bytes of data RAM, extending from Bank 0 to Bank 14 (000h through EFFh). The organization of the data memory space for these devices is shown in Figure 4-6 and Figure 4-7.

The entire data memory may be accessed directly or indirectly. Direct addressing may require the use of the BSR register. Indirect addressing requires the use of a File Select Register (FSRn) and a corresponding Indirect File Operand (INDFn). Each FSR holds a 12-bit address value that can be used to access any location in the data memory map without banking.

The instruction set and architecture allow operations across all banks. This may be accomplished by indirect addressing, or by the use of the MOVFFinstruction. The MOVFFinstruction is a two-word/two-cycle instruction that moves a value from one register to another.

To ensure that commonly used registers (SFRs and select GPRs) can be accessed in a single cycle, regardless of the current BSR values, an Access Bank is implemented. A segment of Bank 0 and a segment of Bank 15 comprise the Access RAM. Section 4.10 "Access Bank" provides a detailed description of the Access RAM.

4.9.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly or indirectly. Indirect addressing operates using a File Select Register and corresponding Indirect File Operand. The operation of indirect addressing is shown in Section 4.12 "Indirect Addressing, INDF and FSR Registers" .

Enhanced MCU devices may have banked memory in the GPR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

Data RAM is available for use as General Purpose Registers by all instructions. The top section of Bank 15 (F60h to FFFh) contains SFRs. All other banks of data memory contain GPR registers, starting with Bank 0.

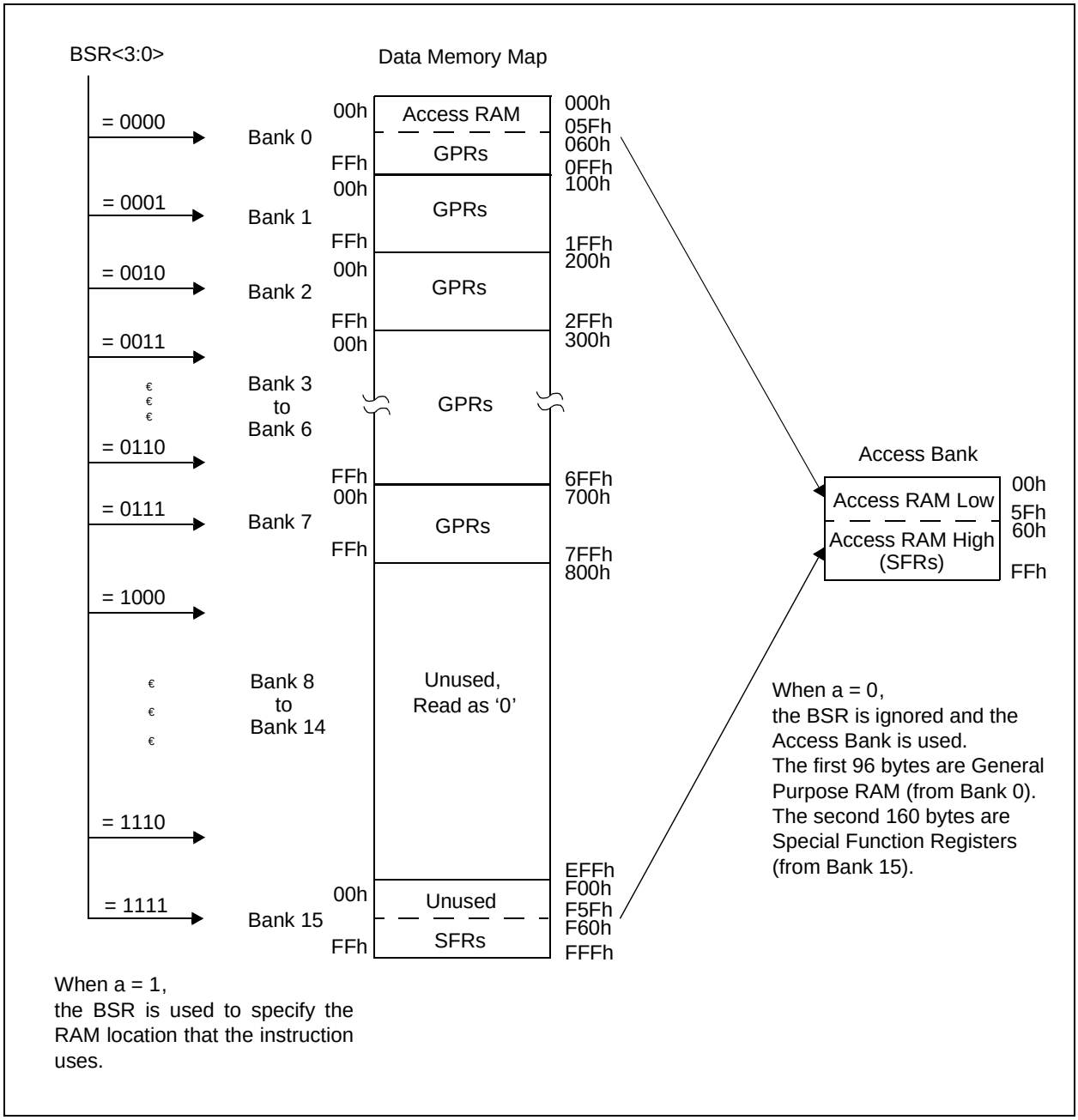
4.9.2 SPECIAL FUNCTION REGISTERS

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. A list of these registers is given in Table 4-2 and Table 4-3.

The SFRs can be classified into two sets: those associated with the "core" function and those related to the peripheral functions. Those registers related to the "core" are described in this section, while those related to the operation of the peripheral features are described in the section of that peripheral feature. The SFRs are typically distributed among the peripherals whose functions they control.

The unused SFR locations are unimplemented and read as '0's. The addresses for the SFRs are listed in Table 4-2.

FIGURE 4-6: DATA MEMORY MAP FOR PIC18FX520 DEVICES



© 2004 Microchip Technology Inc.

FIGURE 4-7: DATA MEMORY MAP FOR PIC18F X620 AND PIC18FX720 DEVICES

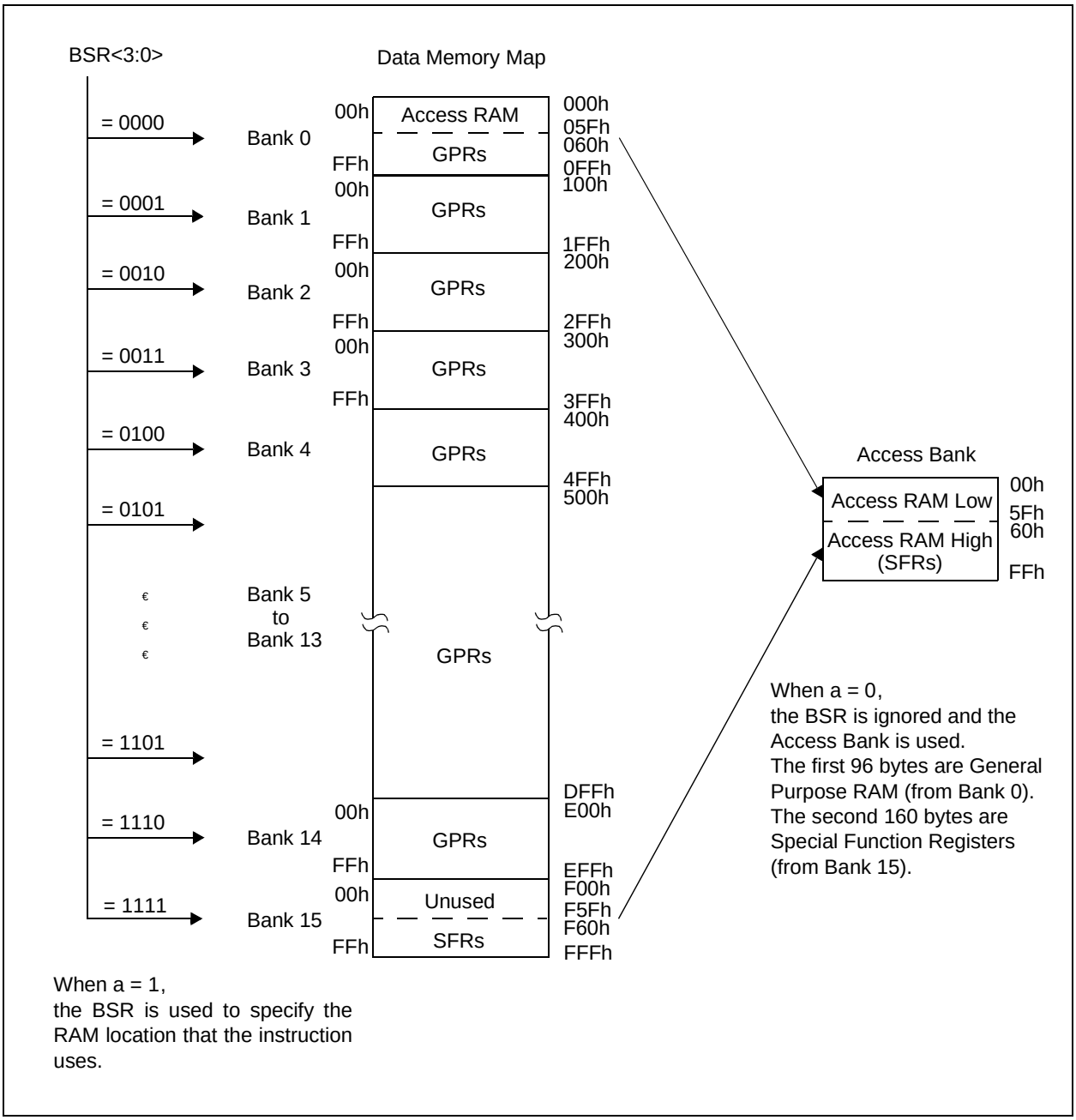


TABLE 4-2: SPECIAL FUNCTION REGISTER MAP

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(3)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(3)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(3)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(3)	FBCh	CCPR2H	F9Ch	MEMCON(2)
FFBh	PCLATU	FDBh	PLUSW2(3)	FBBh	CCPR2L	F9Bh	—(1)
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—(1)	F96h	TRISE
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD
FF4h	PRODH	FD4h	—(1)	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	LVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH
FEFh	INDF0(3)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG
FEEh	POSTINC0(3)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF
FEDh	POSTDEC0(3)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE
FECh	PREINC0(3)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD
FEBh	PLUSW0(3)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	EEADRH	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	PORTJ
FE7h	INDF1(3)	FC7h	SSPSTAT	FA7h	EECON2	F87h	PORTH
FE6h	POSTINC1(3)	FC6h	SSPCON1	FA6h	EECON1	F86h	PORTG
FE5h	POSTDEC1(3)	FC5h	SSPCON2	FA5h	IPR3	F85h	PORTF
FE4h	PREINC1(3)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE
FE3h	PLUSW1(3)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

Note 1: Unimplemented registers are read as ‘0’.

2: This register is unused on PIC18F6X20 devices. Always maintain this register clear.

3: This is not a physical register.

© 2004 Microchip Technology Inc.

TABLE 4-2: SPECIAL FUNCTION REGISTER MAP (CONTINUED)

Address	Name	Address	Name	Address	Name	Address	Name
F7Fh	—(1)	F5Fh	—(1)	F3Fh	—(1)	F1Fh	—(1)
F7Eh	—(1)	F5Eh	—(1)	F3Eh	—(1)	F1Eh	—(1)
F7Dh	—(1)	F5Dh	—(1)	F3Dh	—(1)	F1Dh	—(1)
F7Ch	—(1)	F5Ch	—(1)	F3Ch	—(1)	F1Ch	—(1)
F7Bh	—(1)	F5Bh	—(1)	F3Bh	—(1)	F1Bh	—(1)
F7Ah	—(1)	F5Ah	—(1)	F3Ah	—(1)	F1Ah	—(1)
F79h	—(1)	F59h	—(1)	F39h	—(1)	F19h	—(1)
F78h	TMR4	F58h	—(1)	F38h	—(1)	F18h	—(1)
F77h	PR4	F57h	—(1)	F37h	—(1)	F17h	—(1)
F76h	T4CON	F56h	—(1)	F36h	—(1)	F16h	—(1)
F75h	CCPR4H	F55h	—(1)	F35h	—(1)	F15h	—(1)
F74h	CCPR4L	F54h	—(1)	F34h	—(1)	F14h	—(1)
F73h	CCP4CON	F53h	—(1)	F33h	—(1)	F13h	—(1)
F72h	CCPR5H	F52h	—(1)	F32h	—(1)	F12h	—(1)
F71h	CCPR5L	F51h	—(1)	F31h	—(1)	F11h	—(1)
F70h	CCP5CON	F50h	—(1)	F30h	—(1)	F10h	—(1)
F6Fh	SPBRG2	F4Fh	—(1)	F2Fh	—(1)	F0Fh	—(1)
F6Eh	RCREG2	F4Eh	—(1)	F2Eh	—(1)	F0Eh	—(1)
F6Dh	TXREG2	F4Dh	—(1)	F2Dh	—(1)	F0Dh	—(1)
F6Ch	TXSTA2	F4Ch	—(1)	F2Ch	—(1)	F0Ch	—(1)
F6Bh	RCSTA2	F4Bh	—(1)	F2Bh	—(1)	F0Bh	—(1)
F6Ah	—(1)	F4Ah	—(1)	F2Ah	—(1)	F0Ah	—(1)
F69h	—(1)	F49h	—(1)	F29h	—(1)	F09h	—(1)
F68h	—(1)	F48h	—(1)	F28h	—(1)	F08h	—(1)
F67h	—(1)	F47h	—(1)	F27h	—(1)	F07h	—(1)
F66h	—(1)	F46h	—(1)	F26h	—(1)	F06h	—(1)
F65h	—(1)	F45h	—(1)	F25h	—(1)	F05h	—(1)
F64h	—(1)	F44h	—(1)	F24h	—(1)	F04h	—(1)
F63h	—(1)	F43h	—(1)	F23h	—(1)	F03h	—(1)
F62h	—(1)	F42h	—(1)	F22h	—(1)	F02h	—(1)
F61h	—(1)	F41h	—(1)	F21h	—(1)	F01h	—(1)
F60h	—(1)	F40h	—(1)	F20h	—(1)	F00h	—(1)

Note 1: Unimplemented registers are read as ‘0’.

2: This register is not available on PIC18F6X20 devices.

3: This is not a physical register.

TABLE 4-3: REGISTER FILE SUMMARY

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000	32, 42
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	32, 42
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	32, 42
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					00-0 0000	32, 43
PCLATU	—	—	bit 21	Holding Register for PC<20:16>					--10 0000	32, 44
PCLATH	Holding Register for PC<15:8>								0000 0000	32, 44
PCL	PC Low Byte (PC<7:0>)								0000 0000	32, 44
TBLPTRU	—	—	bit 21(2)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	32, 64
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	32, 64
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	32, 64
TABLAT	Program Memory Table Latch								0000 0000	32, 64
PRODH	Product Register High Byte								xxxx xxxx	32, 85
PRODL	Product Register Low Byte								xxxx xxxx	32, 85
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	32, 89
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	32, 90
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	32, 91
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								n/a	57
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								n/a	57
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								n/a	57
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								n/a	57
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by value in WREG								n/a	57
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- 0000	32, 57
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	32, 57
WREG	Working Register								xxxx xxxx	32
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								n/a	57
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								n/a	57
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								n/a	57
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								n/a	57
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by value in WREG								n/a	57
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000	33, 57
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	33, 57
BSR	—	—	—	—	Bank Select Register				---- 0000	33, 56
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								n/a	57
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								n/a	57
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								n/a	57

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

Note 1: RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator modes only and read '0' in all other oscillator modes.

2: Bit 21 of the TBLPTRU allows access to the device configuration bits.

3: These registers are unused on PIC18F6X20 devices; always maintain these clear.

© 2004 Microchip Technology Inc.

TABLE 4-3: REGISTER FILE SUMMARY (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								n/a	57
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by value in WREG								n/a	57
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000	33, 57
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	33, 57
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	33, 59
TMR0H	Timer0 Register High Byte								0000 0000	33, 133
TMR0L	Timer0 Register Low Byte								xxxx xxxx	33, 133
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	33, 131
OSCCON	—	—	—	—	—	—	—	SCS	---- ---0	25, 33
LVDCON	—	—	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0	--00 0101	33, 235
WDTCON	—	—	—	—	—	—	—	SWDTE	---- ---0	33, 250
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 11qq	33, 60, 101
TMR1H	Timer1 Register High Byte								xxxx xxxx	33, 135
TMR1L	Timer1 Register Low Byte								xxxx xxxx	33, 135
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0-00 0000	33, 135
TMR2	Timer2 Register								0000 0000	33, 141
PR2	Timer2 Period Register								1111 1111	33, 142
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	33, 141
SSPBUF	SSP Receive Buffer/Transmit Register								xxxx xxxx	33, 157
SSPADD	SSP Address Register in I^2C Slave mode. SSP Baud Rate Reload Register in I^2C Master mode.								0000 0000	33, 166
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	33, 158
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	33, 168
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	33, 169
ADRESH	A/D Result Register High Byte								xxxx xxxx	34, 215
ADRESL	A/D Result Register Low Byte								xxxx xxxx	34, 215
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	--00 0000	34, 213
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	34, 214
ADCON2	ADFM	—	—	—	—	ADCS2	ADCS1	ADCS0	0--- -000	34, 215
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	34, 151, 152
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	34, 151, 152
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	34, 149
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	34, 151, 152
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	34, 151, 152
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	34, 149

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

Note 1: RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator modes only and read '0' in all other oscillator modes.

2: Bit 21 of the TBLPTRU allows access to the device configuration bits.

3: These registers are unused on PIC18F6X20 devices; always maintain these clear.

TABLE 4-3: REGISTER FILE SUMMARY (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
CCPR3H	Capture/Compare/PWM Register 3 High Byte								xxxx xxxx	34, 151, 152
CCPR3L	Capture/Compare/PWM Register 3 Low Byte								xxxx xxxx	34, 151, 152
CCP3CON	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	--00 0000	34, 149
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	34, 229
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	34, 223
TMR3H	Timer3 Register High Byte								xxxx xxxx	34, 143
TMR3L	Timer3 Register Low Byte								xxxx xxxx	34, 143
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	34, 143
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	34, 129
SPBRG1	USART1 Baud Rate Generator								0000 0000	34, 205
RCREG1	USART1 Receive Register								0000 0000	34, 206
TXREG1	USART1 Transmit Register								0000 0000	34, 204
TXSTA1	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	34, 198
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	34, 199
EEADRH	—	—	—	—	—	—	EE Adr Register High		---- --00	34, 79
EEADR	Data EEPROM Address Register								0000 0000	34, 79
EEDATA	Data EEPROM Data Register								0000 0000	34, 79
EECON2	Data EEPROM Control Register 2 (not a physical register)								---- ----	34, 79
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	34, 80
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	35, 100
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	35, 94
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	35, 97
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	-1-1 1111	35, 99
PIR2	—	CMIF	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	-0-0 0000	35, 93
PIE2	—	CMIE	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	-0-0 0000	35, 96
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	35, 98
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	35, 92
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	35, 95
MEMCON(3)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	35, 71
TRISJ(3)	Data Direction Control Register for PORTJ								1111 1111	35, 125
TRISH(3)	Data Direction Control Register for PORTH								1111 1111	35, 122
TRISG	—	—	—	Data Direction Control Register for PORTG					---1 1111	35, 120
TRISF	Data Direction Control Register for PORTF								1111 1111	35, 117
TRISE	Data Direction Control Register for PORTE								1111 1111	35, 114
TRISD	Data Direction Control Register for PORTD								1111 1111	35, 111
TRISC	Data Direction Control Register for PORTC								1111 1111	35, 109
TRISB	Data Direction Control Register for PORTB								1111 1111	35, 106
TRISA	—	TRISA6(1)	Data Direction Control Register for PORTA						-111 1111	35, 103

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

Note 1: RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator modes only and read '0' in all other oscillator modes.

2: Bit 21 of the TBLPTRU allows access to the device configuration bits.

3: These registers are unused on PIC18F6X20 devices; always maintain these clear.

© 2004 Microchip Technology Inc.

TABLE 4-3: REGISTER FILE SUMMARY (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
LATJ(3)	Read PORTJ Data Latch, Write PORTJ Data Latch								xxxx xxxx	35, 125
LATH(3)	Read PORTH Data Latch, Write PORTH Data Latch								xxxx xxxx	35, 122
LATG	—	—	—	Read PORTG Data Latch, Write PORTG Data Latch					---x xxxx	35, 120
LATF	Read PORTF Data Latch, Write PORTF Data Latch								xxxx xxxx	35, 117
LATE	Read PORTE Data Latch, Write PORTE Data Latch								xxxx xxxx	35, 114
LATD	Read PORTD Data Latch, Write PORTD Data Latch								xxxx xxxx	35, 111
LATC	Read PORTC Data Latch, Write PORTC Data Latch								xxxx xxxx	35, 109
LATB	Read PORTB Data Latch, Write PORTB Data Latch								xxxx xxxx	35, 106
LATA	—	LATA6(1)	Read PORTA Data Latch, Write PORTA Data Latch(1)						-xxx xxxx	35, 103
PORTJ(3)	Read PORTJ pins, Write PORTJ Data Latch								xxxx xxxx	36, 125
PORTH(3)	Read PORTH pins, Write PORTH Data Latch								xxxx xxxx	36, 122
PORTG	—	—	—	Read PORTG pins, Write PORTG Data Latch					---x xxxx	36, 120
PORTF	Read PORTF pins, Write PORTF Data Latch								xxxx xxxx	36, 117
PORTE	Read PORTE pins, Write PORTE Data Latch								xxxx xxxx	36, 114
PORTD	Read PORTD pins, Write PORTD Data Latch								xxxx xxxx	36, 111
PORTC	Read PORTC pins, Write PORTC Data Latch								xxxx xxxx	36, 109
PORTB	Read PORTB pins, Write PORTB Data Latch								xxxx xxxx	36, 106
PORTA	—	RA6(1)	Read PORTA pins, Write PORTA Data Latch(1)						-x0x 0000	36, 103
TMR4	Timer4 Register								0000 0000	36, 148
PR4	Timer4 Period Register								1111 1111	36, 148
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	36, 147
CCPR4H	Capture/Compare/PWM Register 4 High Byte								xxxx xxxx	36, 151, 152
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								xxxx xxxx	36, 151, 152
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	0000 0000	36, 149
CCPR5H	Capture/Compare/PWM Register 5 High Byte								xxxx xxxx	36, 151, 152
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								xxxx xxxx	36, 151, 152
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	0000 0000	36, 149
SPBRG2	USART2 Baud Rate Generator								0000 0000	36, 205
RCREG2	USART2 Receive Register								0000 0000	36, 206
TXREG2	USART2 Transmit Register								0000 0000	36, 204
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	36, 198
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	36, 199

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

Note 1: RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator modes only and read '0' in all other oscillator modes.

2: Bit 21 of the TBLPTRU allows access to the device configuration bits.

3: These registers are unused on PIC18F6X20 devices; always maintain these clear.

4.10 Access Bank

The Access Bank is an architectural enhancement, which is very useful for C compiler code optimization. The techniques used by the C compiler may also be useful for programs written in assembly.

This data memory region can be used for:

- Intermediate computational values
- Local variables of subroutines
- Faster context saving/switching of variables
- Common variables
- Faster evaluation/control of SFRs (no banking)

The Access Bank is comprised of the upper 160 bytes in Bank 15 (SFRs) and the lower 96 bytes in Bank 0. These two sections will be referred to as Access RAM High and Access RAM Low, respectively. Figure 4-7 indicates the Access RAM areas.

A bit in the instruction word specifies if the operation is to occur in the bank specified by the BSR register or in the Access Bank. This bit is denoted by the ‘a’ bit (for access bit).

When forced in the Access Bank (a = 0), the last address in Access RAM Low is followed by the first address in Access RAM High. Access RAM High maps the Special Function Registers, so that these registers can be accessed without any software overhead. This is useful for testing status flags and modifying control bits.

4.11 Bank Select Register (BSR)

The need for a large general purpose memory space dictates a RAM banking scheme. The data memory is partitioned into sixteen banks. When using direct addressing, the BSR should be configured for the desired bank.

BSR<3:0> holds the upper 4 bits of the 12-bit RAM address. The BSR<7:4> bits will always read ‘0’s and writes will have no effect.

A MOVLB instruction has been provided in the instruction set to assist in selecting banks.

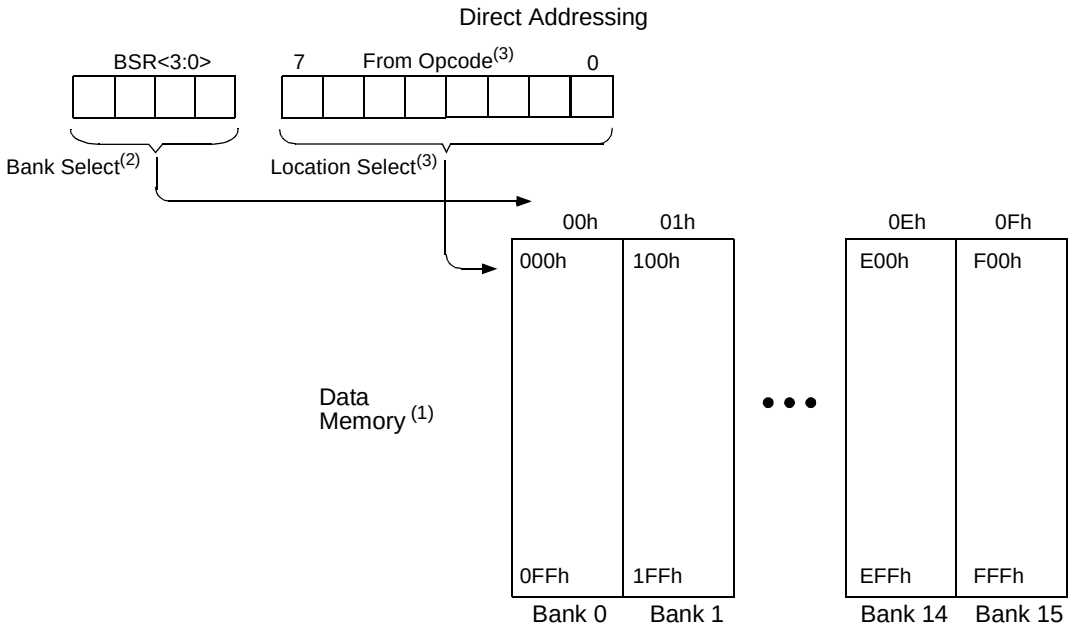
If the currently selected bank is not implemented, any read will return all ‘0’s and all writes are ignored. The Status register bits will be set/cleared as appropriate for the instruction performed.

Each Bank extends up to FFh (256 bytes). All data memory is implemented as static RAM.

A MOVFFinstruction ignores the BSR, since the 12-bit addresses are embedded into the instruction word.

Section 4.12 “Indirect Addressing, INDF and FSR Registers” provides a description of indirect addressing, which allows linear addressing of the entire RAM space.

FIGURE 4-8: DIRECT ADDRESSING



Note 1: For register file map detail, see Table 4-2.

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

3: The MOVFFinstruction embeds the entire 12-bit address in the instruction.

© 2004 Microchip Technology Inc.

4.12 Indirect Addressing, INDF and FSR Registers

Indirect addressing is a mode of addressing data memory, where the data memory address in the instruction is not fixed. An FSR register is used as a pointer to the data memory location that is to be read or written. Since this pointer is in RAM, the contents can be modified by the program. This can be useful for data tables in the data memory and for software stacks. Figure 4-9 shows the operation of indirect addressing. This shows the moving of the value to the data memory address, specified by the value of the FSR register.

Indirect addressing is possible by using one of the INDF registers. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = 0), will read 00h. Writing to the INDF register indirectly, results in a no operation. The FSR register contains a 12-bit address, which is shown in Figure 4-10.

The INDFn register is not a physical register. Addressing INDFn actually addresses the register whose address is contained in the FSRn register (FSRn is a pointer). This is indirect addressing.

Example 4-4 shows a simple use of indirect addressing to clear the RAM in Bank 1 (locations 100h-1FFh) in a minimum number of instructions.

EXAMPLE 4-4: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR  FSR0  ,0x100  ;
NEXT    CLRF  POSTINC0      ; Clear INDF
                            ; register and
                            ; inc pointer
        BTFSS FSR0H, 1      ; All done with
                            ; Bank 1?
        GOTO  NEXT          ; NO, clear next
CONTINUE                    ; YES, continue
```

There are three indirect addressing registers. To address the entire data memory space (4096 bytes), these registers are 12 bits wide. To store the 12 bits of addressing information, two 8-bit registers are required. These indirect addressing registers are:

1. FSR0: composed of FSR0H:FSR0L
2. FSR1: composed of FSR1H:FSR1L
3. FSR2: composed of FSR2H:FSR2L

In addition, there are registers INDF0, INDF1 and INDF2, which are not physically implemented. Reading or writing to these registers activates indirect addressing, with the value in the corresponding FSR register being the address of the data. If an instruction writes a value to INDF0, the value will be written to the address pointed to by FSR0H:FSR0L. A read from INDF1 reads the data from the address pointed to by FSR1H:FSR1L. INDFn can be used in code anywhere an operand can be used.

If INDF0, INDF1 or INDF2 are read indirectly via an FSR, all '0's are read (zero bit is set). Similarly, if INDF0, INDF1 or INDF2 are written to indirectly, the operation will be equivalent to a NOPinstruction and the Status bits are not affected.

4.12.1 INDIRECT ADDRESSING OPERATION

Each FSR register has an INDF register associated with it, plus four additional register addresses. Performing an operation on one of these five registers determines how the FSR will be modified during indirect addressing.

When data access is done to one of the five INDFn locations, the address selected will configure the FSRn register to:

- Do nothing to FSRn after an indirect access (no change) – INDFn.
- Auto-decrement FSRn after an indirect access (post-decrement) – POSTDECn.
- Auto-increment FSRn after an indirect access (post-increment) – POSTINCn.
- Auto-increment FSRn before an indirect access (pre-increment) – PREINCn.
- Use the value in the WREG register as an offset to FSRn. Do not modify the value of the WREG or the FSRn register after an indirect access (no change) – PLUSWn.

When using the auto-increment or auto-decrement features, the effect on the FSR is not reflected in the Status register. For example, if the indirect address causes the FSR to equal '0', the Z bit will not be set.

Incrementing or decrementing an FSR affects all 12 bits. That is, when FSRnL overflows from an increment, FSRnH will be incremented automatically.

Adding these features allows the FSRn to be used as a stack pointer, in addition to its uses for table operations in data memory.

Each FSR has an address associated with it that performs an indexed indirect access. When a data access to this INDFn location (PLUSWn) occurs, the FSRn is configured to add the signed value in the WREG register and the value in FSR to form the address before an indirect access. The FSR value is not changed.

If an FSR register contains a value that points to one of the INDFn, an indirect read will read 00h (zero bit is set), while an indirect write will be equivalent to a NOP (Status bits are not affected).

If an indirect addressing operation is done where the target address is an FSRnH or FSRnL register, the write operation will dominate over the pre- or post-increment/decrement functions.

FIGURE 4-9: INDIRECT ADDRESSING OPERATION

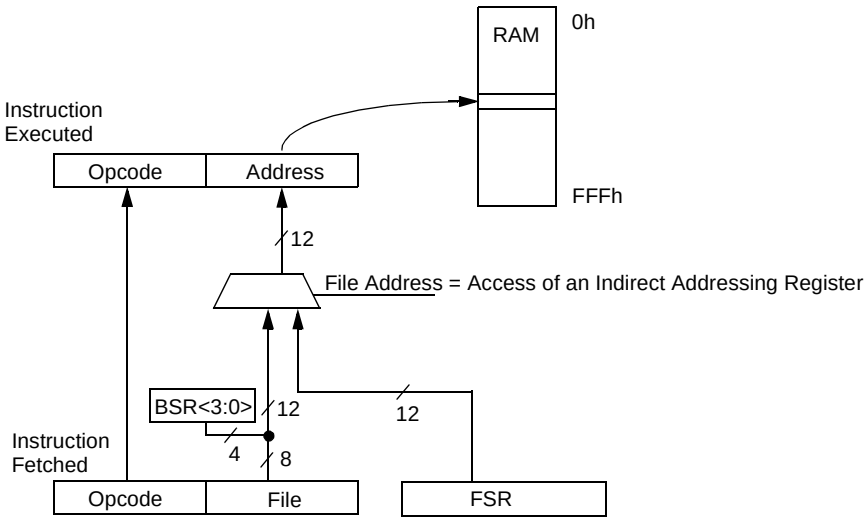
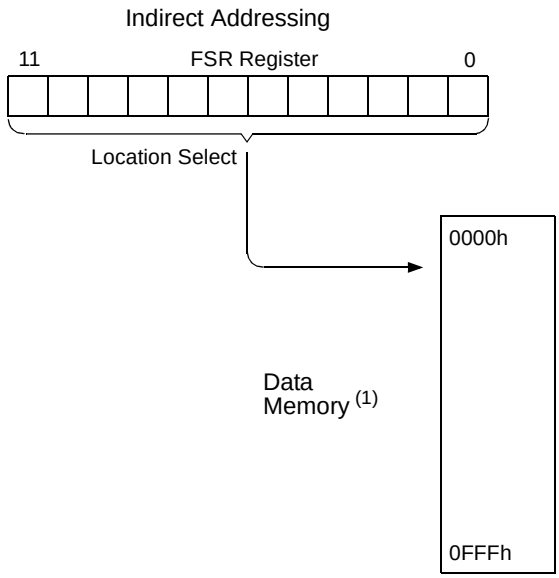


FIGURE 4-10: INDIRECT ADDRESSING



Note 1: For register file map detail, see Table 4-2.

© 2004 Microchip Technology Inc.

4.13 Status Register

The Status register, shown in Register 4-3, contains the arithmetic status of the ALU. The Status register can be the destination 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 clear the upper three bits and set the Z bit. This leaves the Status register as 000u u1uu (where u = unchanged).

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 from the Status register. For other instructions not affecting any status bits, see Table 24-1.

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

REGISTER 4-3: 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	C
bit 7							bit 0

bit 7-5 Unimplemented: Read as '0'

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

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

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

bit 1 DC: Digit carry/$\overline{\text{borrow}}$ bit
For ADDWF, ADDLW, SUBLW and SUBWF instructions:
1 = A carry-out from the 4th low-order bit of the result occurred
0 = No carry-out from the 4th low-order bit of the result

Note: For $\overline{\text{borrow}}$, 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.

bit 0 C: Carry/$\overline{\text{borrow}}$ bit
For ADDWF, ADDLW, SUBLW and SUBWF instructions:
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Note: For $\overline{\text{borrow}}$, 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.

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

4.14 RCON Register

The Reset Control (RCON) register contains flag bits that allow differentiation between the sources of a device Reset. These flags include the $\overline{TO}$, $\overline{PD}$, $\overline{POR}$, $\overline{BOR}$ and $\overline{RI}$ bits. This register is readable and writable.

Note 1:	If the BOREN configuration bit is set (Brown-out Reset enabled), the $\overline{BOR}$ bit is '1' on a Power-on Reset. After a Brown-out Reset has occurred, the BOR bit will be cleared and must be set by firmware to indicate the occurrence of the next Brown-out Reset.
2:	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.

REGISTER 4-4: RCON REGISTER

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

bit 7 IPEN: Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6-5 Unimplemented: Read as '0'

bit 4 $\overline{RI}$: RESETInstruction Flag bit
1 = The RESETinstruction was not executed
0 = The RESETinstruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 $\overline{TO}$: Watchdog Time-out Flag bit
1 = After power-up, CLRWDTinstruction, or SLEEPinstruction
0 = A WDT time-out occurred

bit 2 $\overline{PD}$: Power-down Detection Flag bit
1 = After power-up or by the CLRWDTinstruction
0 = By execution of the SLEEPinstruction

bit 1 $\overline{POR}$: Power-on Reset Status bit
1 = A Power-on Reset has not occurred
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
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

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

© 2004 Microchip Technology Inc.

5.0 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 8 bytes at a time. Program memory is erased in blocks of 64 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.

5.1 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)

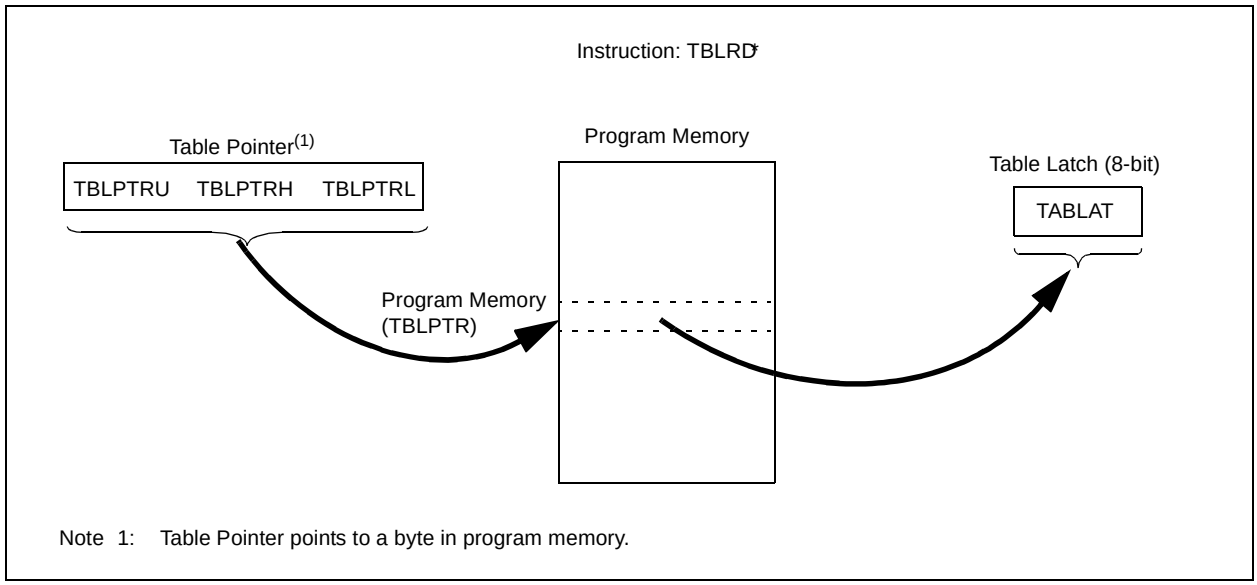
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into the data RAM space. Figure 5-1 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 5.5 "Writing to Flash Program Memory" . Figure 5-2 shows the operation of a table write with program memory and data RAM.

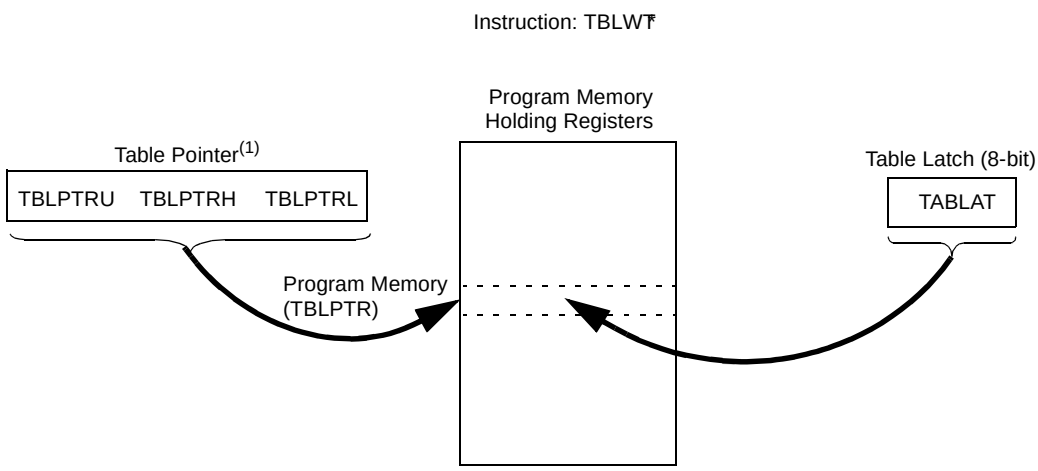
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions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

FIGURE 5-1: TABLE READ OPERATION



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

FIGURE 5-2: TABLE WRITE OPERATION



Note 1: Table Pointer actually points to one of eight holding registers, the address of which is determined by TBLPTRL<2:0>. The process for physically writing data to the Program Memory Array is discussed in Section 5.5 “Writing to Flash Program Memory” .

5.2 Control Registers

Several control registers are used in conjunction with the TBLRDand TBLWTinstructions. These include the:

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

5.2.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all ‘0’s. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit EEPGD determines if the access will be a program or data EEPROM memory access. When clear, any subsequent operations will operate on the data EEPROM memory. When set, any subsequent operations will operate on the program memory.

Control bit CFGS determines if the access will be to the configuration/calibration registers, or to program memory/data EEPROM memory. When set, subsequent operations will operate on configuration registers, regardless of EEPGD (see Section 23.0 “Special Features of the CPU”). When clear, memory selection access is determined by EEPGD.

The FREE bit, when set, will allow a program memory erase operation. When the FREE bit 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 when a write operation is interrupted by a MCLR Reset, or a WDT Time-out Reset during normal operation. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR), due to Reset values of zero.

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. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

Note:	Interrupt flag bit, EEIF in the PIR2 register, is set when the write is complete. It must be cleared in software.

© 2004 Microchip Technology Inc.

REGISTER 5-1: EECON1 REGISTER (ADDRESS FA6h)

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

bit 7 EEPGD: Flash Program or Data EEPROM Memory Select bit
1 = Access Flash program memory
0 = Access data EEPROM memory

bit 6 CFGS: Flash Program/Data EEPROM or Configuration Select bit
1 = Access configuration registers
0 = Access Flash program or data EEPROM memory

bit 5 Unimplemented: Read as ‘0’

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

bit 3 WRERR: Flash Program/Data EEPROM Error Flag bit
1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation)
0 = The write operation completed

Note: When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

bit 2 WREN: Flash Program/Data EEPROM Write Enable bit
1 = Allows write cycles to Flash program/data EEPROM
0 = Inhibits write cycles to Flash program/data EEPROM

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

bit 0 RD: Read Control bit
1 = Initiates an EEPROM read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1.)
0 = Does not initiate an EEPROM read

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

5.2.2 TABLAT – TABLE LATCH REGISTER

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

5.2.3 TBLPTR – TABLE POINTER REGISTER

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

The Table Pointer, TBLPTR, is used by the TBLRDand TBLWTinstructions. These instructions can update the TBLPTR in one of four ways, based on the table operation. These operations are shown in Table 5-1. These operations on the TBLPTR only affect the low-order 21 bits.

5.2.4 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 Table Pointer determine which byte is read from program memory into TABLAT.

When a TBLWTis executed, the three LSbs of the Table Pointer (TBLPTR<2:0>) determine which of the eight program memory holding registers is written to. When the timed write to program memory (long write) begins, the 19 MSbs of the Table Pointer, TBLPTR (TBLPTR<21:3>), will determine which program memory block of 8 bytes is written to. For more detail, see Section 5.5 “Writing to Flash Program Memory” .

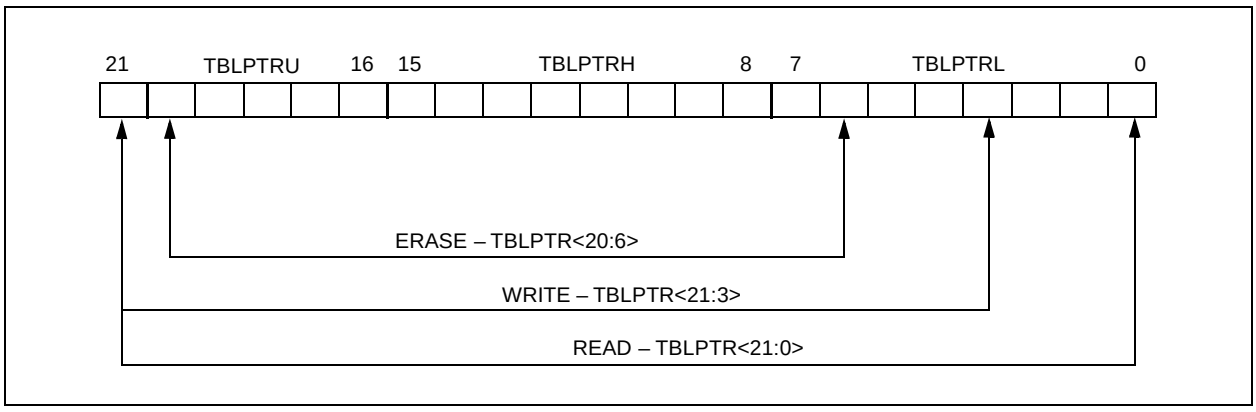
When an erase of program memory is executed, the 16 MSbs of the Table Pointer (TBLPTR<21:6>) point to the 64-byte block that will be erased. The Least Significant bits (TBLPTR<5:0>) are ignored.

Figure 5-3 describes the relevant boundaries of TBLPTR based on Flash program memory operations.

TABLE 5-1: TABLE POINTER OPERATIONS WITH TBLRDAND TBLWTINSTRUCTIONS

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 5-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



© 2004 Microchip Technology Inc.

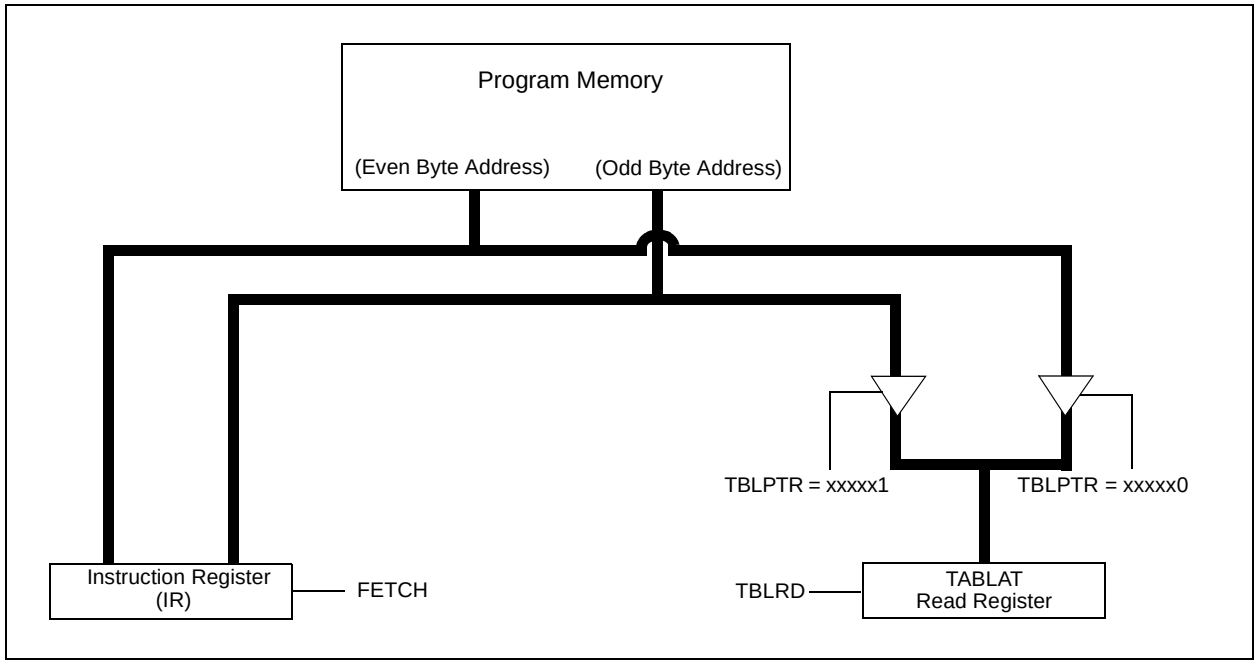
5.3 Reading the Flash Program Memory

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 5-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 5-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 5-1: 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
        MOVWF   WORD_ODD
```

5.4 Erasing Flash Program Memory

The minimum erase block is 32 words or 64 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 64 bytes of program memory is erased. The Most Significant 16 bits of the TBLPTR<21:6> point to the block being erased. TBLPTR<5:0> are ignored.

The EECON1 register commands the erase operation. The EEPGD bit must be set to point to the Flash program memory. 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.

5.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

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

1. Load Table Pointer with address of row being erased.
2. Set the EECON1 register for the erase operation:
   - set EEPGD bit to point to program memory;
   - clear the CFGS bit to access program memory;
   - set WREN bit to enable writes;
   - set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write AAh to EECON2.
6. Set the WR bit. This will begin the row erase cycle.
7. The CPU will stall for duration of the erase (about 2 ms using internal timer).
8. Execute a NOP.
9. Re-enable interrupts.

EXAMPLE 5-2: 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, EEPGD       ; point to Flash program memory
                    BCF     EECON1, CFGS        ; access Flash program memory
                    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
Required            MOVLW   AAh
Sequence            MOVWF   EECON2              ; write AAH
                    BSF     EECON1, WR          ; start erase (CPU stall)
                    NOP
                    BSF     INTCON, GIE         ; re-enable interrupts
```


© 2004 Microchip Technology Inc.

5.5 Writing to Flash Program Memory

The minimum programming block is 4 words or 8 bytes. Word or byte programming is not supported.

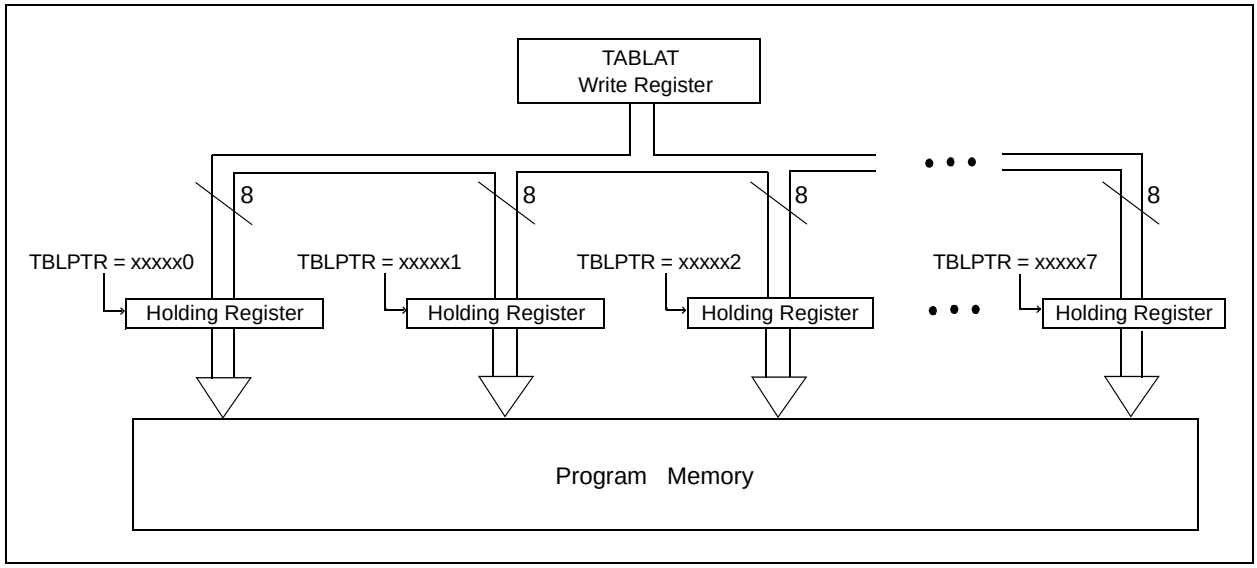
Table writes are used internally to load the holding registers needed to program the Flash memory. There are 8 holding registers used by the table writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWTinstruction has to be executed 8 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 8 registers, the EECON1 register must be written to, to start the programming operation with a long write.

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

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

FIGURE 5-5: TABLE WRITES TO FLASH PROGRAM MEMORY



5.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

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

1. Read 64 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer with address being erased.
4. Do the row erase procedure.
5. Load Table Pointer with address of first byte being written.
6. Write the first 8 bytes into the holding registers with auto-increment.
7. Set the EECON1 register for the write operation:
   - set EEPGD bit to point to program memory
   - clear the CFGS bit to access program memory
   - set WREN to enable byte writes
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write AAh to EECON2.
11. Set the WR bit. This will begin the write cycle.
12. The CPU will stall for duration of the write (about 2 ms using internal timer).
13. Execute a NOP
14. Re-enable interrupts.
15. Repeat steps 6-14 seven times, to write 64 bytes.
16. Verify the memory (table read).

This procedure will require about 18 ms to update one row of 64 bytes of memory. An example of the required code is given in Example 5-3.

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

EXAMPLE 5-3: WRITING TO FLASH PROGRAM MEMORY

```
                MOVLW   D'64'                   ; number of bytes in erase block
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
                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
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVF    TABLAT, W               ; get data
                MOVWF   POSTINC0                ; store data
                DECFSZ  COUNTER                 ; done?
                BRA     READ_BLOCK              ; repeat
MODIFY_WORD
                MOVLW   DATA_ADDR_HIGH          ; point to buffer
                MOVWF   FSR0H
                MOVLW   DATA_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   NEW_DATA_LOW            ; update buffer word
                MOVWF   POSTINC0
                MOVLW   NEW_DATA_HIGH
                MOVWF   INDF0
ERASE_BLOCK
                MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
                BSF     EECON1, EEPGD           ; point to Flash program memory
                BCF     EECON1, CFGS            ; access Flash program memory
                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
    Required    MOVLW   AAh
    Sequence    MOVWF   EECON2                  ; write AAH
                BSF     EECON1, WR              ; start erase (CPU stall)
                NOP
                BSF     INTCON, GIE             ; re-enable interrupts
                TBLRD*-                         ; dummy read decrement
WRITE_BUFFER_BACK
                MOVLW   8                       ; number of write buffer groups of 8 bytes
                MOVWF   COUNTER_HI
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
PROGRAM_LOOP
                MOVLW   8                       ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_WORD_TO_HREGS
                MOVFF   POSTINC0, WREG          ; get low byte of buffer data
                                                ; present data to table latch
                TBLWT+*                         ; write data, perform a short write
                                                ; to internal TBLWT holding register.
                DECFSZ  COUNTER                 ; loop until buffers are full
                BRA     WRITE_WORD_TO_HREGS
```


© 2004 Microchip Technology Inc.

EXAMPLE 5-3: WRITING TO FLASH PROGRAM MEMORY (CONTINUED)

```
PROGRAM_MEMORY
                BSF     EECON1, EEPGD       ; point to Flash program memory
                BCF     EECON1, CFGS        ; access Flash program memory
                BSF     EECON1, WREN        ; enable write to memory
                BCF     INTCON, GIE         ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2              ; write 55H
    Required    MOVLW   AAh
    Sequence    MOVWF   EECON2              ; write AAH
                BSF     EECON1, WR          ; start program (CPU stall)
                NOP
                BSF     INTCON, GIE         ; re-enable interrupts
                DECFSZ  COUNTER_HI          ; loop until done
                BRA     PROGRAM_LOOP
                BCF     EECON1, WREN        ; disable write to memory
```

5.5.2 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.

5.5.3 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. The WRERR bit is set when a write operation is interrupted by a MCLR Reset, or a WDT Time-out Reset during normal operation. In these situations, users can check the WRERR bit and rewrite the location.

5.5.4 PROTECTION AGAINST SPURIOUS WRITES

To protect against spurious writes to Flash program memory, the write initiate sequence must also be followed. See Section 23.0 “Special Features of the CPU” for more detail.

5.6 Flash Program Operation During Code Protection

See Section 23.0 “Special Features of the CPU” for details on code protection of Flash program memory.

TABLE 5-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	--00 0000
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	0000 0000
TBLPTRL	Program Memory Table Pointer High Byte (TBLPTR<7:0>)								0000 0000	0000 0000
TABLAT	Program Memory Table Latch								0000 0000	0000 0000
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 0000	0000 0000
EECON2	EEPROM Control Register 2 (not a physical register)								—	—
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	uu-0 u000
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111	---1 1111
PIR2	—	CMIF	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000	---0 0000
PIE2	—	CMIE	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000	---0 0000

Legend: x = unknown, u = unchanged, r = reserved, - = unimplemented, read as ‘0’.
Shaded cells are not used during Flash/EEPROM access.

NOTES:

© 2004 Microchip Technology Inc.

6.0 EXTERNAL MEMORY INTERFACE

Note:	The External Memory Interface is not implemented on PIC18F6X20 (64-pin) devices.

The External Memory Interface is a feature of the PIC18F8X20 devices that allows the controller to access external memory devices (such as Flash, EPROM, SRAM, etc.) as program or data memory.

The physical implementation of the interface uses 27 pins. These pins are reserved for external address/data bus functions; they are multiplexed with I/O port pins on four ports. Three I/O ports are multiplexed with the address/data bus, while the fourth port is multiplexed with the bus control signals. The I/O port functions are enabled when the EBDIS bit in the MEMCON register is set (see Register 6-1). A list of the multiplexed pins and their functions is provided in Table 6-1.

As implemented in the PIC18F8X20 devices, the interface operates in a similar manner to the external memory interface introduced on PIC18C601/801 microcontrollers. The most notable difference is that the interface on PIC18F8X20 devices only operates in 16-bit modes. The 8-bit mode is not supported.

For a more complete discussion of the operating modes that use the external memory interface, refer to Section 4.1.1 “PIC18F8X20 Program Memory Modes” .

6.1 Program Memory Modes and the External Memory Interface

As previously noted, PIC18F8X20 controllers are capable of operating in any one of four 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 Microprocessor Mode , the external bus is always active and the port pins have only the external bus function.

In Microcontroller Mode, the bus is not active and the pins have their port functions only. Writes to the MEMCOM register are not permitted.

In Microprocessor with Boot Block or 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.

Note:	Maximum FOSC for the PIC18FX520 is limited to 25 MHz when using the external memory interface.

REGISTER 6-1: MEMCON 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
bit7							bit0

bit 7 EBDIS: External Bus Disable bit
1 = External system bus disabled, all external bus drivers are mapped as I/O ports
0 = External system bus enabled and 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>: TBLWRTOperation with 16-bit Bus bits
1x = Word Write mode: TABLAT<0> and TABLAT<1> word output, $\overline{WRH}$ active when TABLAT<1> written
01 = Byte Select mode: TABLAT data copied on both MSB and LSB, $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate
00 = Byte Write mode: TABLAT data copied on both MSB and LSB, $\overline{WRH}$ or $\overline{WRL}$ will activate

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

If the device fetches or accesses external memory while EBDIS = 1, the pins will switch to 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.

When the device is executing out of internal memory (EBDIS = 0) in Microprocessor with Boot Block mode, or Extended Microcontroller mode, the control signals will NOT be active. They will go to a state where the AD<15:0> and A<19:16> 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'.

TABLE 6-1: PIC18F8X20 EXTERNAL BUS – I/O PORT FUNCTIONS

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

© 2004 Microchip Technology Inc.

6.2 16-bit Mode

The External Memory Interface implemented in PIC18F8X20 devices operates only in 16-bit mode. The mode selection is not software configurable, but is programmed via the configuration bits.

The WM<1:0> bits in the MEMCON register determine three types of connections in 16-bit mode. They are referred to as:

- 16-bit Byte Write
- 16-bit Word Write
- 16-bit Byte Select

These three different configurations allow the designer maximum flexibility in using 8-bit and 16-bit memory devices.

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

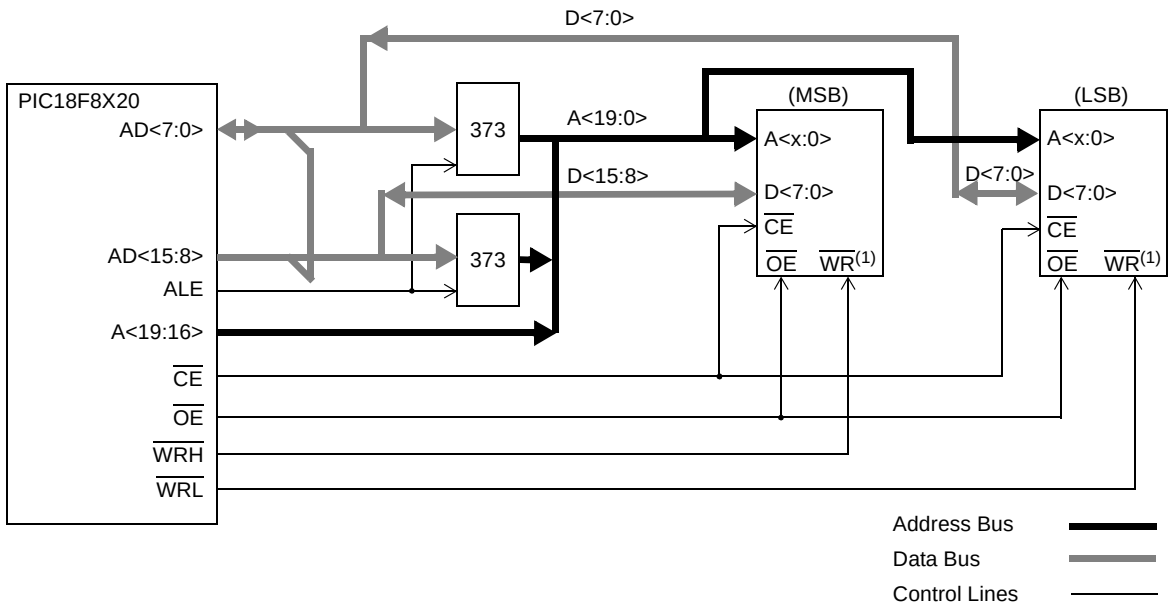
In Byte Select mode, JEDEC standard Flash memories will require BA0 for the byte address line and one I/O line to select between Byte and Word mode. The other 16-bit modes do not need BA0. JEDEC standard static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte selection.

6.2.1 16-BIT BYTE WRITE MODE

Figure 6-1 shows an example of 16-bit Byte Write mode for PIC18F8X20 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 TBLWTinstruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD15:AD0 bus. The appropriate $\overline{WRH}$ or $\overline{WRL}$ control line is strobed on the LSb of the TBLPTR.

FIGURE 6-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: This signal only applies to table writes. See Section 5.1 “Table Reads and Table Writes” .

6.2.2 16-BIT WORD WRITE MODE

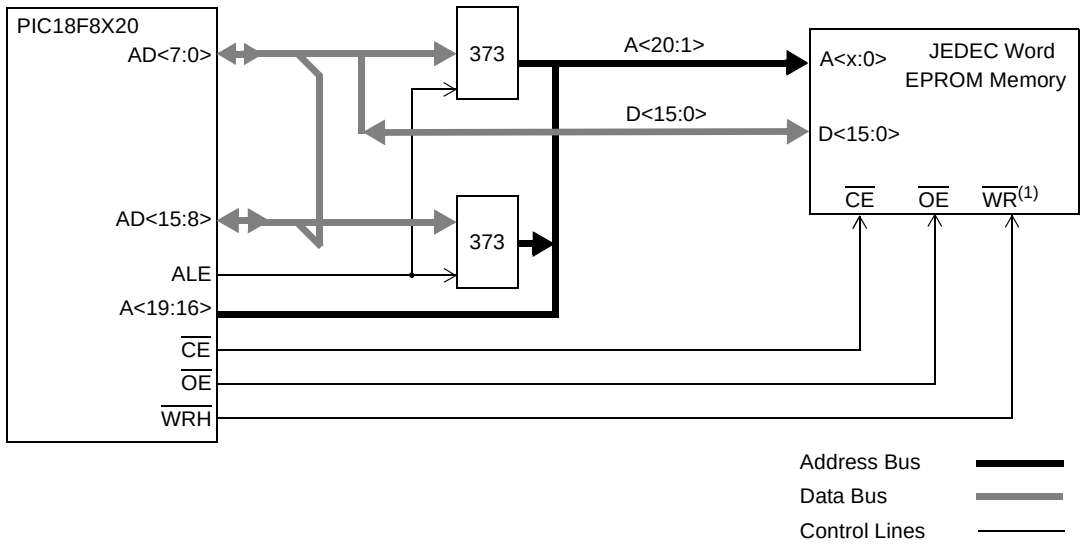
Figure 6-2 shows an example of 16-bit Word Write mode for PIC18F8X20 devices. This mode is used for word-wide memories, which includes some of the EPROM and Flash type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory and table writes to any type of word-wide external memories. This method makes a distinction between TBLWTcycles 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 AD15:AD0 bus. The contents of the holding latch are presented on the lower byte of the AD15:AD0 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 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 6-2: 16-BIT WORD WRITE MODE EXAMPLE



Note 1: This signal only applies to table writes. See Section 5.1 “Table Reads and Table Writes” .

© 2004 Microchip Technology Inc.

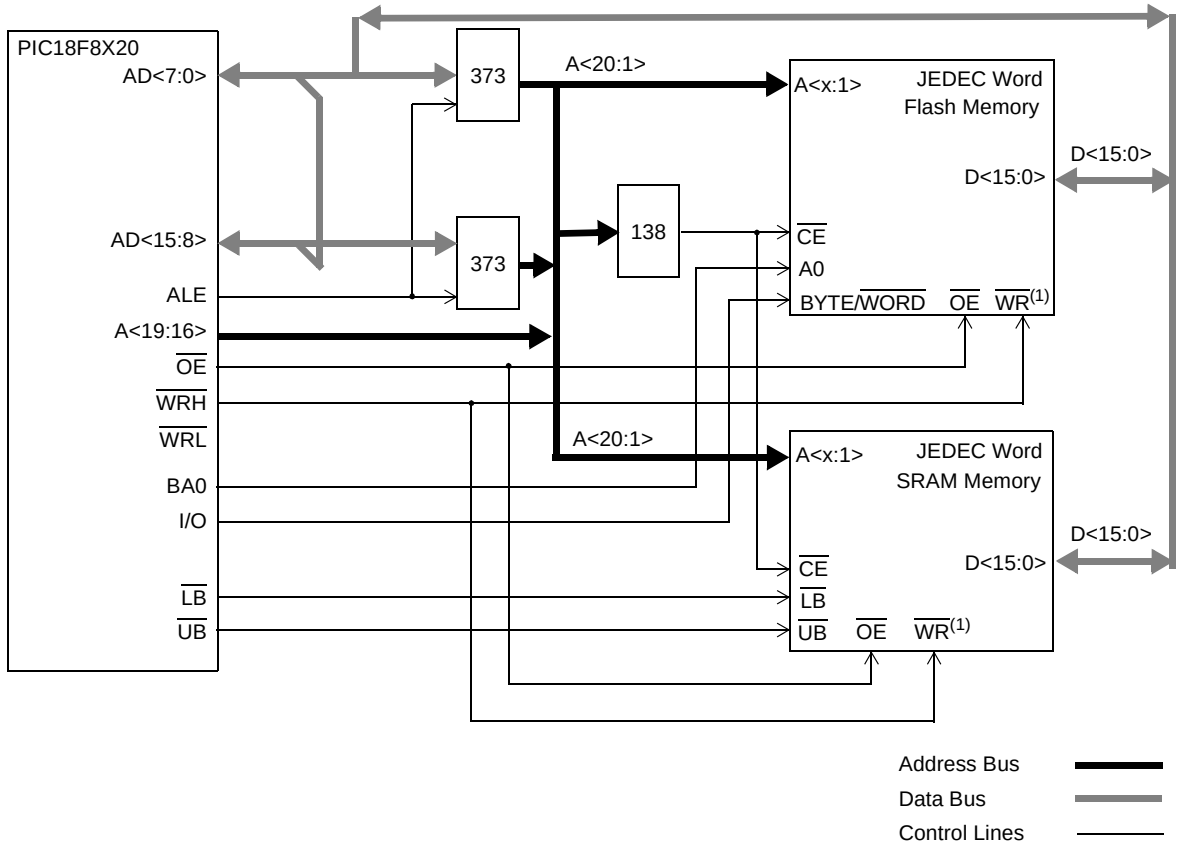
6.2.3 16-BIT BYTE SELECT MODE

Figure 6-3 shows an example of 16-bit Byte Select mode for PIC18F8X20 devices. 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 TBLWTcycle, the TABLAT data is presented on the upper and lower byte of the AD15:AD0 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 6-3: 16-BIT BYTE SELECT MODE EXAMPLE



Note 1: This signal only applies to table writes. See Section 5.1 "Table Reads and Table Writes" .

6.2.4 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 6-4 through Figure 6-6.

FIGURE 6-4: EXTERNAL MEMORY BUS TIMING FOR TBLRD (MICROPROCESSOR MODE)

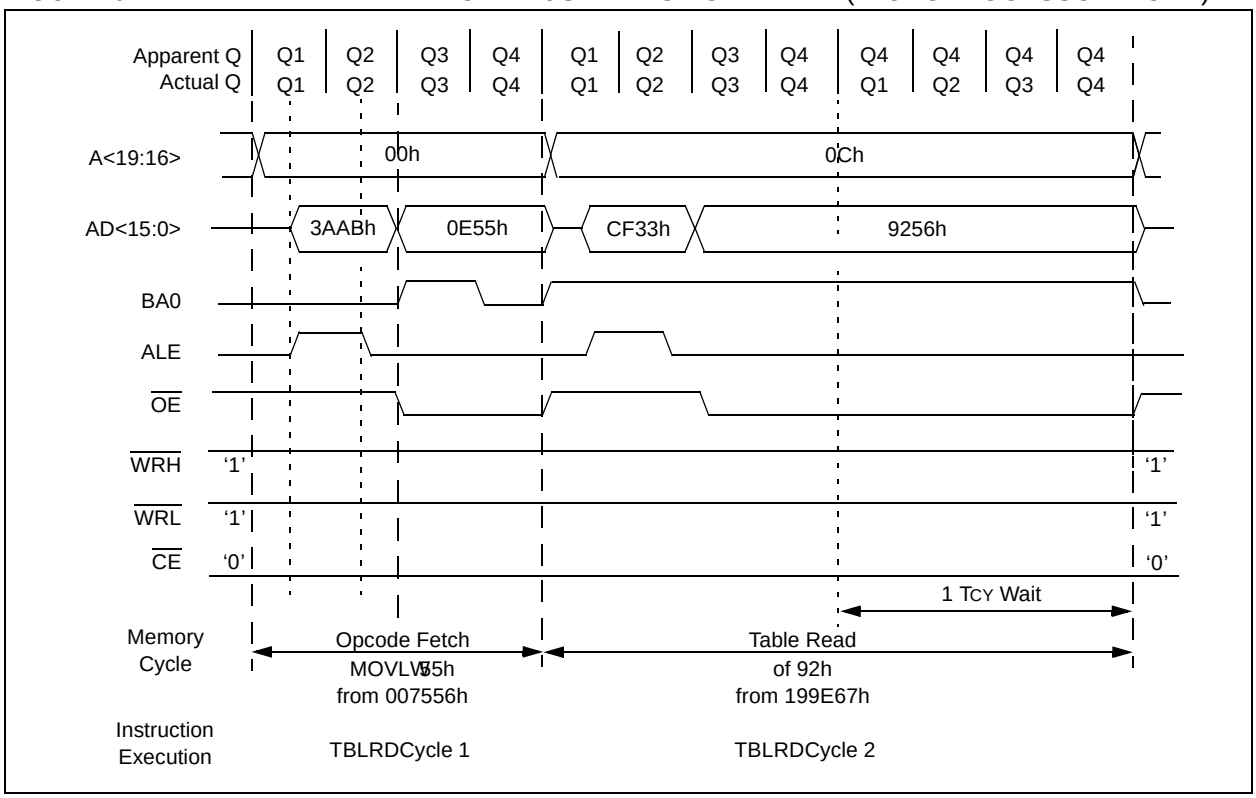
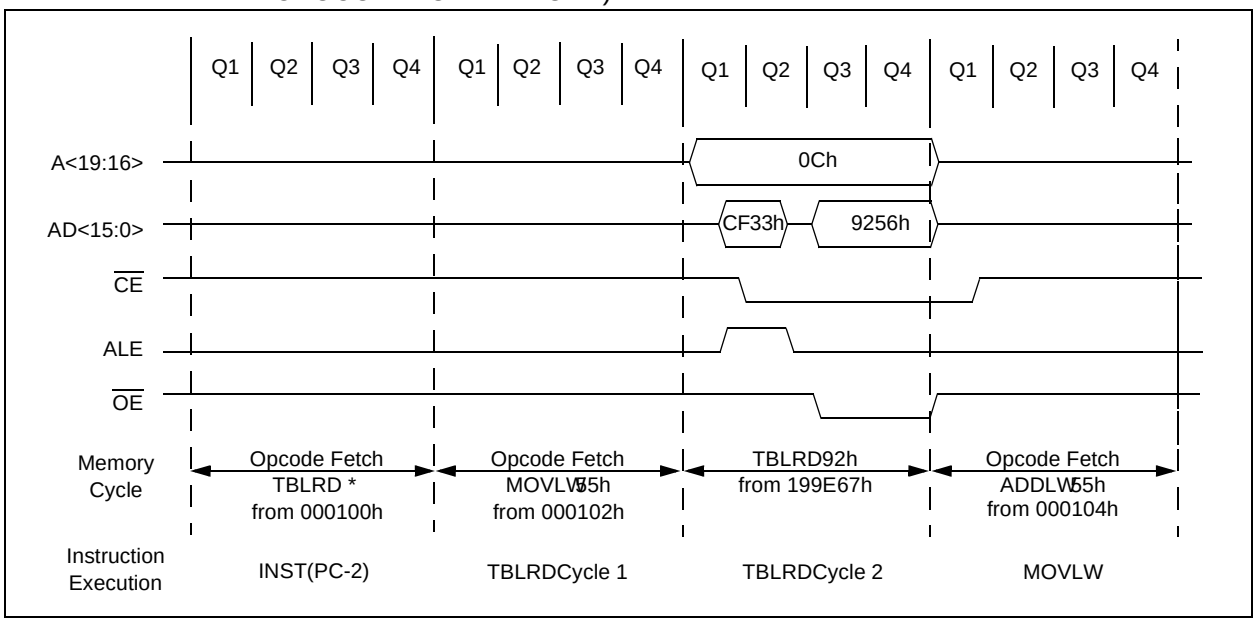
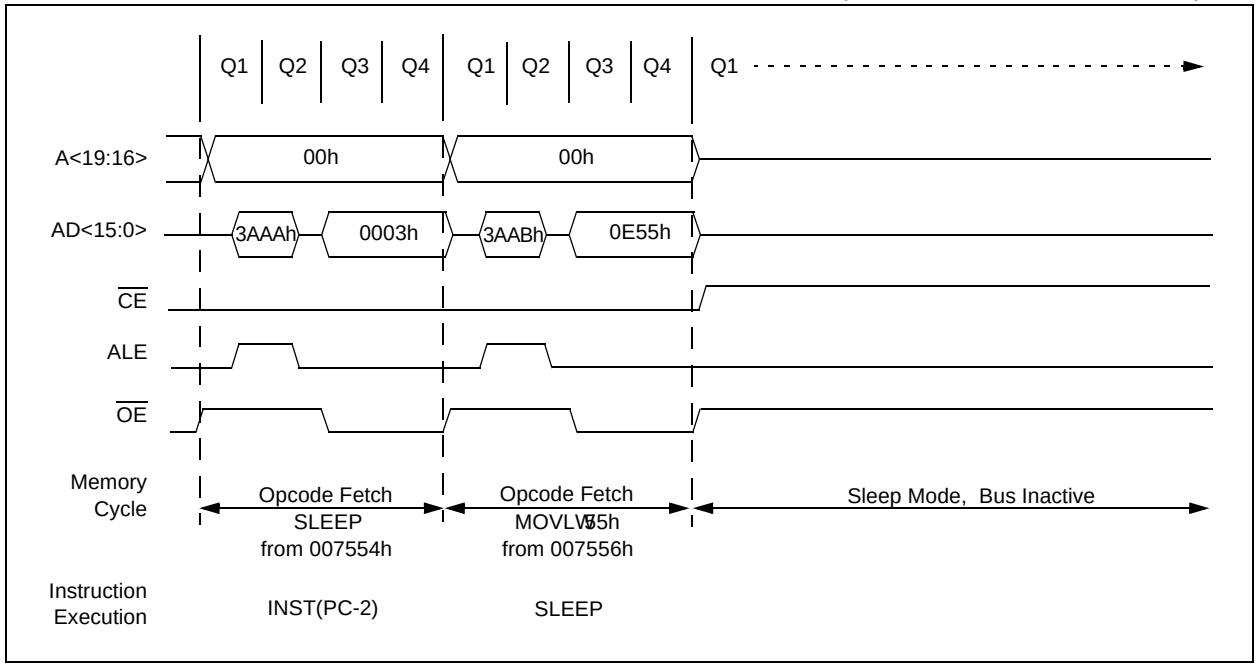


FIGURE 6-5: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)



© 2004 Microchip Technology Inc.

FIGURE 6-6: EXTERNAL MEMORY BUS TIMING FOR SLEEP (MICROPROCESSOR MODE)



NOTES:

© 2004 Microchip Technology Inc.

7.0 DATA EEPROM MEMORY

The data EEPROM is readable and writable during normal operation over the entire VDD range. The data memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers (SFR).

There are five SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADRH
- EEADR

The EEPROM data memory allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write. EEADR and EEADRH hold the address of the EEPROM location being accessed. These devices have 1024 bytes of data EEPROM with an address range from 00h to 3FFh.

The EEPROM data memory is rated for high erase/write cycles. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature, as well as from chip to chip. Please refer to parameter D122 (see Section 26.0 “Electrical Characteristics”) for exact limits.

7.1 EEADR and EEADRH

The address register pair can address up to a maximum of 1024 bytes of data EEPROM. The two Most Significant bits of the address are stored in EEADRH, while the remaining eight Least Significant bits are stored in EEADR. The six Most Significant bits of EEADRH are unused and are read as ‘0’.

7.2 EECON1 and EECON2 Registers

EECON1 is the control register for EEPROM memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all ‘0’s. The EECON2 register is used exclusively in the EEPROM write sequence.

Control bits, RD and WR, initiate read and write operations, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at the completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a $\overline{\text{MCLR}}$ Reset or a WDT Time-out Reset during normal operation. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR) due to the Reset condition forcing the contents of the registers to zero.

Note:	Interrupt flag bit, EEIF in the PIR2 register, is set when write is complete. It must be cleared in software.

REGISTER 7-1: EECON1 REGISTER (ADDRESS FA6h)

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

bit 7 EEPGD: Flash Program/Data EEPROM Memory Select bit

1 = Access Flash program memory
0 = Access data EEPROM memory

bit 6 CFGS: Flash Program/Data EEPROM or Configuration Select bit

1 = Access configuration or calibration registers
0 = Access Flash program or data EEPROM memory

bit 5 Unimplemented: Read as '0'

bit 4 FREE: Flash Row Erase Enable bit

1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation)
0 = Perform write only

bit 3 WRERR: Flash Program/Data EEPROM Error Flag bit

1 = A write operation is prematurely terminated (any MCLR or any WDT Reset during self-timed programming in normal operation)
0 = The write operation completed

Note: When a WRERR occurs, the EEPGD or FREE bits are not cleared. This allows tracing of the error condition.

bit 2 WREN: Flash Program/Data EEPROM Write Enable bit

1 = Allows write cycles to Flash program/data EEPROM
0 = Inhibits write cycles to Flash program/data EEPROM

bit 1 WR: Write Control bit

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

bit 0 RD: Read Control bit

1 = Initiates an EEPROM read. (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1.)
0 = Does not initiate an EEPROM read

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

© 2004 Microchip Technology Inc.

7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADRH:EEADR register pair, clear the EEPGD control bit (EECON1<7>), clear the CFGS control bit (EECON1<6>) and then set the RD control bit (EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation, or until it is written to by the user (during a write operation).

EXAMPLE 7-1: DATA EEPROM READ

```
MOVLW   DATA_EE_ADDRH   ;
MOVWF   EEADRH          ; Upper bits of Data Memory Address to read
MOVLW   DATA_EE_ADDR    ;
MOVWF   EEADR           ; Lower bits of Data Memory Address to read
BCF     EECON1, EEPGD   ; Point to DATA memory
BCF     EECON1, CFGS    ; Access EEPROM
BSF     EECON1, RD      ; EEPROM Read
MOVF    EEDATA, W       ; W = EEDATA
```

7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADRH:EEADR register pair and the data written to the EEDATA register. Then the sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not initiate if the above sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution (i.e., runaway programs). The WREN bit should be kept clear at all times, except when updating the EEPROM. The WREN bit is not cleared by hardware

After a write sequence has been initiated, EECON1, EEADRH, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Write Complete Interrupt Flag bit (EEIF) is set. The user may either enable this interrupt, or poll this bit. EEIF must be cleared by software.

EXAMPLE 7-2: DATA EEPROM WRITE

```
                MOVLW   DATA_EE_ADDRH   ;
                MOVWF   EEADRH          ; Upper bits of Data Memory Address to write
                MOVLW   DATA_EE_ADDR    ;
                MOVWF   EEADR           ; Lower bits of Data Memory Address to write
                MOVLW   DATA_EE_DATA    ;
                MOVWF   EEDATA          ; Data Memory Value to write
                BCF     EECON1, EEPGD   ; Point to DATA memory
                BCF     EECON1, CFGS    ; Access EEPROM
                BSF     EECON1, WREN    ; Enable writes

                BCF     INTCON, GIE     ; Disable Interrupts
                MOVLW   55h             ;
Required        MOVWF   EECON2          ; Write 55h
Sequence        MOVLW   AAh             ;
                MOVWF   EECON2          ; Write AAh
                BSF     EECON1, WR      ; Set WR bit to begin write
                BSF     INTCON, GIE     ; Enable Interrupts

                                        ; User code execution
                BCF     EECON1, WREN    ; Disable writes on write complete (EEIF set)
```

7.5 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.

7.6 Protection Against Spurious Write

There are conditions when the user may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

7.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect mechanism. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal data EEPROM, regardless of the state of the code-protect configuration bit. Refer to Section 23.0 “Special Features of the CPU” for additional information.

7.8 Using the Data EEPROM

The data EEPROM is a high endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 7-3.

Note:	If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124.

EXAMPLE 7-3: DATA EEPROM REFRESH ROUTINE

```
        CLRF    EEADR               ; Start at address 0
        CLRF    EEADRH              ;
        BCF     EECON1, CFGS        ; Set for memory
        BCF     EECON1, EEPGD       ; Set for Data EEPROM
        BCF     INTCON, GIE         ; Disable interrupts
        BSF     EECON1, WREN        ; Enable writes
Loop                                ; Loop to refresh array
        BSF     EECON1, RD          ; Read current address
        MOVLW   55h                 ;
        MOVWF   EECON2              ; Write 55h
        MOVLW   AAh                 ;
        MOVWF   EECON2              ; Write AAh
        BSF     EECON1, WR          ; Set WR bit to begin write
        BTFSC   EECON1, WR          ; Wait for write to complete
        BRA     $-2
        INCFSZ  EEADR, F            ; Increment address
        BRA     Loop                ; Not zero, do it again
        INCFSZ  EEADRH, F           ; Increment the high address
        BRA     Loop                ; Not zero, do it again

        BCF     EECON1, WREN        ; Disable writes
        BSF     INTCON, GIE         ; Enable interrupts
```


© 2004 Microchip Technology Inc.

TABLE 7-1: REGISTERS ASSOCIATED WITH DATA EEPROM MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
EEADRH	—	—	—	—	—	—	EE Addr Register High		---- --00	---- --00
EEADR	EEPROM Address Register								0000 0000	0000 0000
EEDATA	EEPROM Data Register								0000 0000	0000 0000
EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	---- ----
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	uu-0 u000
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111	---1 1111
PIR2	—	CMIF	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000	---0 0000
PIE2	—	CMIE	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000	---0 0000

Legend: x = unknown, u = unchanged, r = reserved, – = unimplemented, read as ‘0’.
Shaded cells are not used during Flash/EEPROM access.

NOTES:

© 2004 Microchip Technology Inc.

8.0 8 X 8 HARDWARE MULTIPLIER

8.1 Introduction

An 8 x 8 hardware multiplier is included in the ALU of the PIC18FXX20 devices. By making the multiply a hardware operation, it completes in a single instruction cycle. This is an unsigned multiply that gives a 16-bit result. The result is stored in the 16-bit product register pair (PRODH:PRODL). The multiplier does not affect any flags in the ALUSTA register.

Making the 8 x 8 multiplier execute in a single cycle gives the following advantages:

- Higher computational throughput
- Reduces code size requirements for multiply algorithms

The performance increase allows the device to be used in applications previously reserved for Digital Signal Processors.

Table 8-1 shows a performance comparison between enhanced devices using the single-cycle hardware multiply and performing the same function without the hardware multiply.

8.2 Operation

Example 8-1 shows the sequence to do an 8 x 8 unsigned multiply. Only one instruction is required when one argument of the multiply is already loaded in the WREG register.

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

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

EXAMPLE 8-2: 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 8-1: PERFORMANCE COMPARISON

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 8-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 8-1 shows the algorithm that is used. The 32-bit result is stored in four registers, RES3:RES0.

EQUATION 8-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

RES3:RES0 = ARG1H:ARG1L € ARG2H:ARG2L
= (ARG1H € ARG2H € 2^{16}) +
(ARG1H € ARG2L € 2^{8}) +
(ARG1L € ARG2H € 2^{8}) +
(ARG1L € ARG2L)

EXAMPLE 8-3: 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 8-4 shows the sequence to do a 16 x 16 signed multiply. Equation 8-2 shows the algorithm used. The 32-bit result is stored in four registers, RES3:RES0. To account for the sign bits of the arguments, each argument pairs' Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EQUATION 8-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

RES3:RES0
= ARG1H:ARG1L € ARG2H:ARG2L
= (ARG1H € ARG2H € 2^{16}) +
(ARG1H € ARG2L € 2^{8}) +
(ARG1L € ARG2H € 2^{8}) +
(ARG1L € ARG2L) +
(-1 € ARG2H<7> € ARG1H:ARG1L € 2^{16}) +
(-1 € ARG1H<7> € ARG2H:ARG2L € 2^{16})

EXAMPLE 8-4: 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
   :
```


© 2004 Microchip Technology Inc.

9.0 INTERRUPTS

The PIC18FXX20 devices have multiple interrupt sources and an interrupt priority feature that allows each interrupt source to be assigned a high or a low priority level. The high priority interrupt vector is at 000008h, while the low priority interrupt vector is at 000018h. High priority interrupt events will override any low priority interrupts that may be in progress.

There are thirteen registers which are used to control interrupt operation. They 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.

Each interrupt source has three bits to control its operation. The functions of these bits 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. Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared. When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h, depending on the priority level. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PICmicro® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 000008h in Compatibility mode.

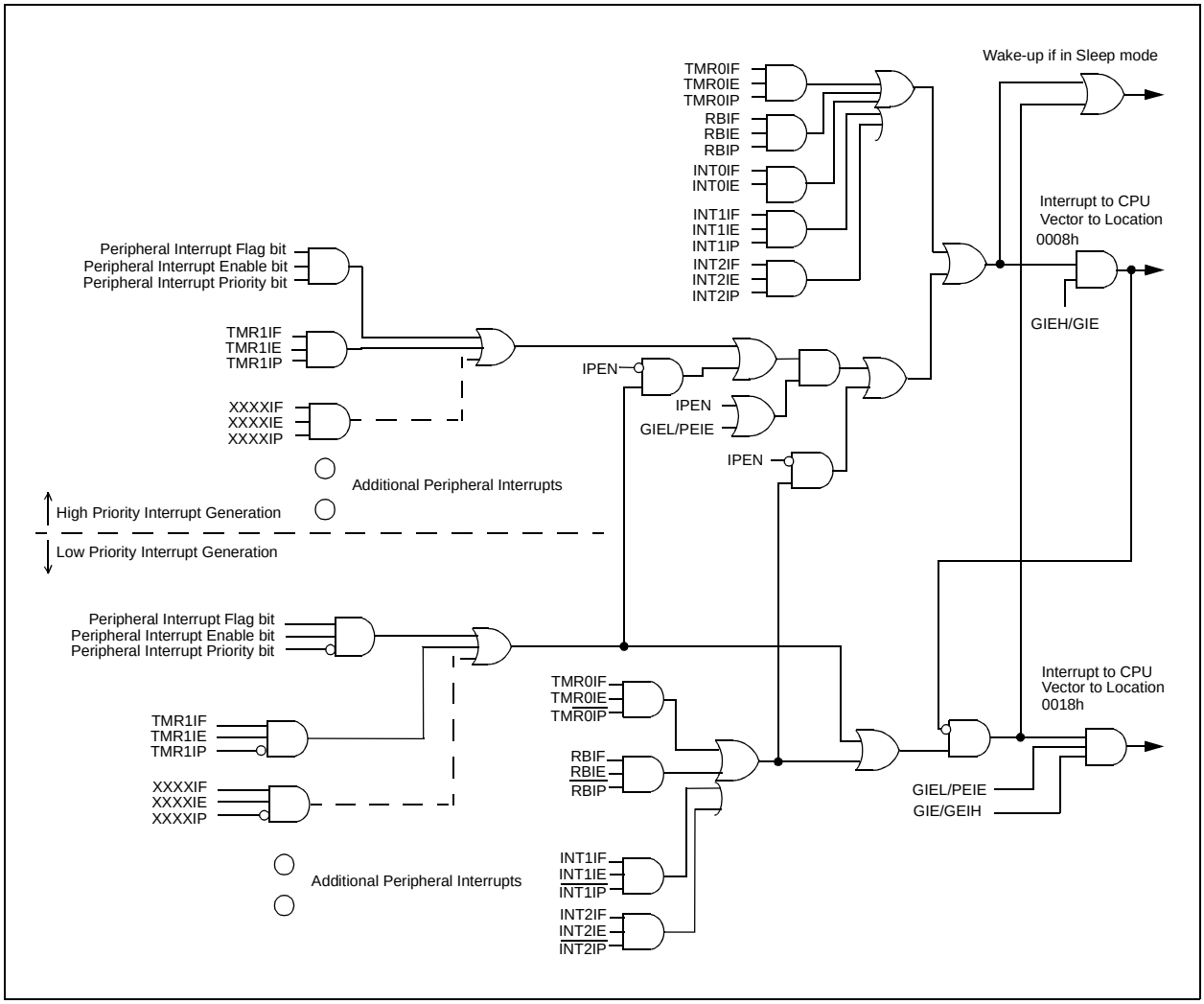
When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High priority interrupt sources can interrupt a low priority interrupt.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

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

For external interrupt events, such as the INT pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

FIGURE 9-1: INTERRUPT LOGIC



© 2004 Microchip Technology Inc.

9.1 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 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 9-1: INTCON REGISTER

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

bit 7 GIE/GIEH: Global Interrupt Enable bit
When IPEN (RCON<7>) = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN (RCON<7>) = 1:
1 = Enables all high priority interrupts
0 = Disables all interrupts

bit 6 PEIE/GIEL: Peripheral Interrupt Enable bit
When IPEN (RCON<7>) = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN (RCON<7>) = 1:
1 = Enables all low priority peripheral interrupts
0 = Disables all low priority peripheral interrupts

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

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

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

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

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

bit 0 RBIF: RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

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

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

REGISTER 9-2: INTCON2 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
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
bit 7							bit 0

bit 7 $\overline{\text{RBPU}}$: PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 INTEDG0: External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 INTEDG1: External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 INTEDG2: External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 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

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

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

© 2004 Microchip Technology Inc.

REGISTER 9-3: INTCON3 REGISTER

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

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

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

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

9.2 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 Flag Registers (PIR1, PIR2 and 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 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 9-4: 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	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit 7 PSPIF: Parallel Slave Port Read/Write Interrupt Flag bit(1)
1 = A read or a write operation has taken place (must be cleared in software)
0 = No read or write has occurred

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: USART1 Receive Interrupt Flag bit
1 = The USART1 receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The USART1 receive buffer is empty

bit 4 TX1IF: USART Transmit Interrupt Flag bit
1 = The USART1 transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The USART1 transmit buffer is full

bit 3 SSPIF: Master Synchronous Serial Port Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 CCP1IF: CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 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: Enabled only in Microcontroller mode for PIC18F8X20 devices.

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

© 2004 Microchip Technology Inc.

REGISTER 9-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CMIF	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF
bit 7							bit 0

bit 7 Unimplemented: Read as '0'

bit 6 CMIF: Comparator Interrupt Flag bit
1 = The comparator input has changed (must be cleared in software)
0 = The comparator input has not changed

bit 5 Unimplemented: Read as '0'

bit 4 EEIF: Data EEPROM/Flash Write Operation Interrupt Flag bit
1 = The write operation is complete (must be cleared in software)
0 = The write operation is not complete, or has not been started

bit 3 BCLIF: Bus Collision Interrupt Flag bit
1 = A bus collision occurred while the SSP module (configured in I^2C Master mode) was transmitting (must be cleared in software)
0 = No bus collision occurred

bit 2 LVDIF: Low-Voltage Detect Interrupt Flag bit
1 = A low-voltage condition occurred (must be cleared in software)
0 = The device voltage is above the Low-Voltage Detect trip point

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

bit 0 CCP2IF: CCP2 Interrupt Flag bit
Capture mode:
1 = A TMR1 or TMR3 register capture occurred (must be cleared in software)
0 = No TMR1 or TMR3 register capture occurred
Compare mode:
1 = A TMR1 or TMR3 register compare match occurred (must be cleared in software)
0 = No TMR1 or TMR3 register compare match occurred
PWM mode:
Unused in this mode.

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

REGISTER 9-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

U-0	U-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF
bit 7							bit 0

bit 7- 6 Unimplemented: Read as ‘0’

bit 5 RC2IF: USART2 Receive Interrupt Flag bit

1 = The USART2 receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The USART2 receive buffer is empty

bit 4 TX2IF: USART2 Transmit Interrupt Flag bit

1 = The USART2 transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The USART2 transmit buffer is full

bit 3 TMR4IF: TMR3 Overflow Interrupt Flag bit

1 = TMR4 register overflowed (must be cleared in software)
0 = TMR4 register did not overflow

bit 2-0 CCPxIF: CCPx Interrupt Flag bit (CCP Modules 3, 4 and 5)

Capture mode:
1 = A TMR1 or TMR3 register capture occurred (must be cleared in software)
0 = No TMR1 or TMR3 register capture occurred

Compare mode:
1 = A TMR1 or TMR3 register compare match occurred (must be cleared in software)
0 = No TMR1 or TMR3 register compare match occurred

PWM mode:
Unused in this mode.

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

© 2004 Microchip Technology Inc.

9.3 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 and PIE3). When the IPEN bit (RCON<7>) is '0', the PEIE bit must be set to enable any of these peripheral interrupts.

REGISTER 9-7: 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	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7 PSPIE: Parallel Slave Port Read/Write Interrupt Enable bit(1)
1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

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

bit 5 RC1IE: USART1 Receive Interrupt Enable bit
1 = Enables the USART1 receive interrupt
0 = Disables the USART1 receive interrupt

bit 4 TX1IE: USART1 Transmit Interrupt Enable bit
1 = Enables the USART1 transmit interrupt
0 = Disables the USART1 transmit interrupt

bit 3 SSPIE: Master Synchronous Serial Port Interrupt Enable bit
1 = Enables the MSSP interrupt
0 = Disables the MSSP interrupt

bit 2 CCP1IE: CCP1 Interrupt Enable bit
1 = Enables the CCP1 interrupt
0 = Disables the CCP1 interrupt

bit 1 TMR2IE: TMR2 to PR2 Match Interrupt Enable bit
1 = Enables the TMR2 to PR2 match interrupt
0 = Disables the TMR2 to PR2 match interrupt

bit 0 TMR1IE: TMR1 Overflow Interrupt Enable bit
1 = Enables the TMR1 overflow interrupt
0 = Disables the TMR1 overflow interrupt

Note 1: Enabled only in Microcontroller mode for PIC18F8X20 devices.

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

REGISTER 9-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CMIE	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE
bit 7							bit 0

bit 7 Unimplemented: Read as '0'

bit 6 CMIE: Comparator Interrupt Enable bit
1 = Enables the comparator interrupt
0 = Disables the comparator interrupt

bit 5 Unimplemented: Read as '0'

bit 4 EEIE: Data EEPROM/Flash Write Operation Interrupt Enable bit
1 = Enables the write operation interrupt
0 = Disables the write operation interrupt

bit 3 BCLIE: Bus Collision Interrupt Enable bit
1 = Enables the bus collision interrupt
0 = Disables the bus collision interrupt

bit 2 LVDIE: Low-Voltage Detect Interrupt Enable bit
1 = Enables the Low-Voltage Detect interrupt
0 = Disables the Low-Voltage Detect interrupt

bit 1 TMR3IE: TMR3 Overflow Interrupt Enable bit
1 = Enables the TMR3 overflow interrupt
0 = Disables the TMR3 overflow interrupt

bit 0 CCP2IE: CCP2 Interrupt Enable bit
1 = Enables the CCP2 interrupt
0 = Disables the CCP2 interrupt

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

© 2004 Microchip Technology Inc.

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

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE
bit 7							bit 0

bit 7-6 Unimplemented: Read as ‘0’

bit 5 RC2IE: USART2 Receive Interrupt Enable bit
1 = Enables the USART2 receive interrupt
0 = Disables the USART2 receive interrupt

bit 4 TX2IE: USART2 Transmit Interrupt Enable bit
1 = Enables the USART2 transmit interrupt
0 = Disables the USART2 transmit interrupt

bit 3 TMR4IE: TMR4 to PR4 Match Interrupt Enable bit
1 = Enables the TMR4 to PR4 match interrupt
0 = Disables the TMR4 to PR4 match interrupt

bit 2-0 CCPxIE: CCPx Interrupt Enable bit (CCP Modules 3, 4 and 5)
1 = Enables the CCPx interrupt
0 = Disables the CCPx interrupt

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

9.4 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 and IPR3). The operation of the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 9-10: 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	SSPIP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

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

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

bit 5 RC1IP: USART1 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 TX1IP: USART1 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 2 CCP1IP: CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

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

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

Note 1: Enabled only in Microcontroller mode for PIC18F8X20 devices.

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

© 2004 Microchip Technology Inc.

REGISTER 9-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

U-0	R/W-1	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP
bit 7							bit 0

bit 7 Unimplemented: Read as ‘0’

bit 6 CMIP: Comparator Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 Unimplemented: Read as ‘0’

bit 4 EEIP: Data EEPROM/Flash Write Operation Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 BCLIP: Bus Collision Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 LVDIP: Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

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

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

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

REGISTER 9-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP
bit 7							bit 0

bit 7-6 Unimplemented: Read as '0'

bit 5 RC2IP: USART2 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 TX2IP: USART2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 TMR4IP: TMR4 to PR4 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2-0 CCPxIP: CCPx Interrupt Priority bit (CCP Modules 3, 4 and 5)
1 = High priority
0 = Low priority

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

© 2004 Microchip Technology Inc.

9.5 RCON Register

The RCON register contains the IPEN bit, which is used to enable prioritized interrupts. The functions of the other bits in this register are discussed in more detail in Section 4.14 “RCON Register” .

REGISTER 9-13: RCON REGISTER

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

bit 7 IPEN: Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16 Compatibility mode)

bit 6-5 Unimplemented: Read as ‘0’

bit 4 $\overline{RI}$: RESETInstruction Flag bit
For details of bit operation, see Register 4-4.

bit 3 $\overline{TO}$: Watchdog Time-out Flag bit
For details of bit operation, see Register 4-4.

bit 2 $\overline{PD}$: Power-Down Detection Flag bit
For details of bit operation, see Register 4-4.

bit 1 $\overline{POR}$: Power-on Reset Status bit
For details of bit operation, see Register 4-4.

bit 0 $\overline{BOR}$: Brown-out Reset Status bit
For details of bit operation, see Register 4-4.

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

9.6 INT0 Interrupt

External interrupts on the RB0/INT0, RB1/INT1, RB2/INT2 and RB3/INT3 pins are edge-triggered: either rising, if the corresponding INTEDGx bit is set in the INTCON2 register, or falling, if the INTEDGx bit is clear. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit, INTxF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxE. Flag bit, INTxF, must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt. All external interrupts (INT0, INT1, INT2 and INT3) can wake-up the processor from Sleep if bit INTxIE was set prior to going into Sleep. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

The interrupt priority for INT, 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.

9.7 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 registers (FFFFh 0000h) will set flag bit TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See Section 11.0 “Timer0 Module” for further details on the Timer0 module.

9.8 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>).

9.9 Context Saving During Interrupts

During an interrupt, the return PC value is saved on the stack. Additionally, the WREG, Status and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (see Section 4.3 “Fast Register Stack”), the user may need to save the WREG, Status and BSR registers in software. Depending on the user’s application, other registers may also need to be saved. Example 9-1 saves and restores the WREG, Status and BSR registers during an Interrupt Service Routine.

EXAMPLE 9-1: 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 located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR           ; Restore BSR
MOVF    W_TEMP, W               ; Restore WREG
MOVFF   STATUS_TEMP, STATUS     ; Restore STATUS
```


© 2004 Microchip Technology Inc.

10.0 I/O PORTS

Depending on the device selected, there are either seven or nine I/O ports available on PIC18FXX20 devices. Some of their pins are multiplexed with one or more alternate functions from the other peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

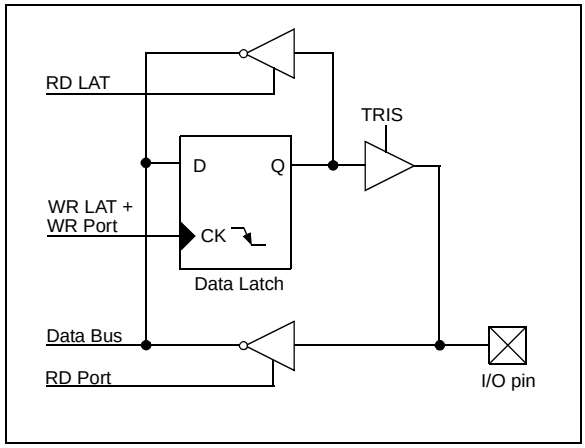
Each port has three registers for its operation. These registers are:

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

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

A simplified version of a generic I/O port and its operation is shown in Figure 10-1.

FIGURE 10-1: SIMPLIFIED BLOCK DIAGRAM OF PORT/LAT/TRIS OPERATION



10.1 PORTA, TRISA and LATA Registers

PORTA is a 7-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

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

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

The RA4 pin is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The RA4/T0CKI pin is a Schmitt Trigger input and an open-drain output. All other RA port pins have TTL input levels and full CMOS output drivers.

The RA6 pin is only enabled as a general I/O pin in ECIO and RCIO Oscillator modes.

The other PORTA pins are multiplexed with analog inputs and the analog $V_{REF}+$ and $V_{REF}-$ inputs. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register 1).

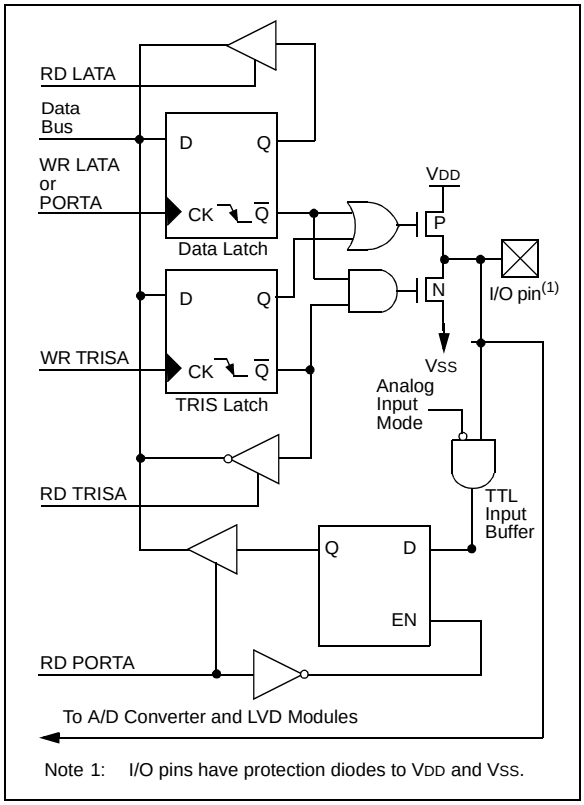
Note:	On a Power-on Reset, RA5 and RA3:RA0 are configured as analog inputs and read as '0'. RA6 and RA4 are configured as digital inputs.

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

EXAMPLE 10-1: INITIALIZING PORTA

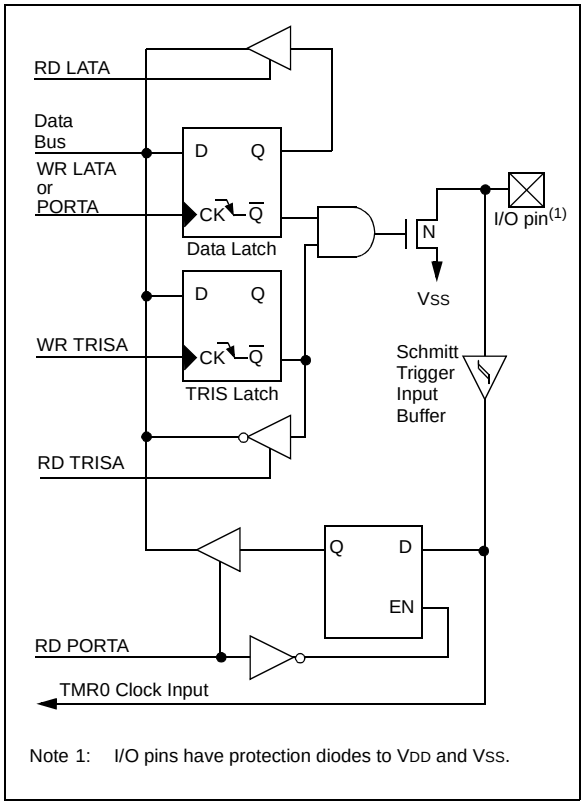
```
CLRF    PORTA    ; Initialize PORTA by
                 ; clearing output
                 ; data latches
CLRF    LATA     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0x0F     ; Configure A/D
MOVWF   ADCON1   ; for digital inputs
MOVLW   0xCF     ;  Value used to
                 ; initialize data
                 ; direction
MOVWF   TRISA    ;  Set RA<3:0> as inputs
                 ;  RA<5:4> as outputs
```

FIGURE 10-2: BLOCK DIAGRAM OF RA3:RA0 AND RA5 PINS



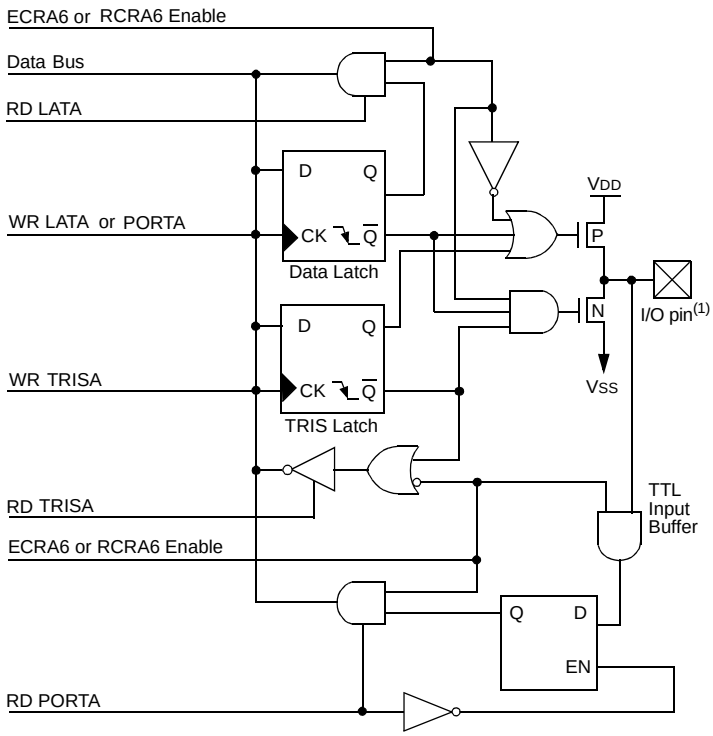
Note 1: I/O pins have protection diodes to VDD and VSS.

FIGURE 10-3: BLOCK DIAGRAM OF RA4/T0CKI PIN



Note 1: I/O pins have protection diodes to VDD and VSS.

FIGURE 10-4: BLOCK DIAGRAM OF RA6 PIN (WHEN ENABLED AS I/O)



Note 1: I/O pins have protection diodes to VDD and VSS.

© 2004 Microchip Technology Inc.

TABLE 10-1: PORTA FUNCTIONS

Name	Bit#	Buffer	Function
RA0/AN0	bit 0	TTL	Input/output or analog input.
RA1/AN1	bit 1	TTL	Input/output or analog input.
RA2/AN2/VREF-	bit 2	TTL	Input/output or analog input or VREF-.
RA3/AN3/VREF+	bit 3	TTL	Input/output or analog input or VREF+.
RA4/T0CKI	bit 4	ST	Input/output or external clock input for Timer0. Output is open-drain type.
RA5/AN4/LVDIN	bit 5	TTL	Input/output or slave select input for synchronous serial port or analog input, or Low-Voltage Detect input.
OSC2/CLKO/RA6	bit 6	TTL	OSC2 or clock output, or I/O pin.

Legend: TTL = TTL input, ST = Schmitt Trigger input

TABLE 10-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTA	—	RA6	RA5	RA4	RA3	RA2	RA1	RA0	-x0x 0000	-u0u 0000
LATA	—	LATA Data Output Register							-xxx xxxx	-uuu uuuu
TRISA	—	PORTA Data Direction Register							-111 1111	-111 1111
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, – = unimplemented locations read as ‘0’.
Shaded cells are not used by PORTA.

10.2 PORTB, TRISB and LATB Registers

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

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

EXAMPLE 10-2: INITIALIZING PORTB

```
CLRF    PORTB    ; Initialize PORTB by
                 ; clearing output
                 ; data latches
CLRF    LATB     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0xCF     ;Value used to
                 ; initialize data
                 ; direction
MOVWF   TRISB    ;Set RB<3:0> as inputs
                 ;  RB<5:4> as outputs
                 ;  RB<7:6> as inputs
```

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

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

Four of the PORTB pins (RB3:RB0) are the external interrupt pins, INT3 through INT0. In order to use these pins as external interrupts, the corresponding TRISB bit must be set to '1'.

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

This interrupt can wake the device from Sleep. 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 MOVFFinstruction). 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.

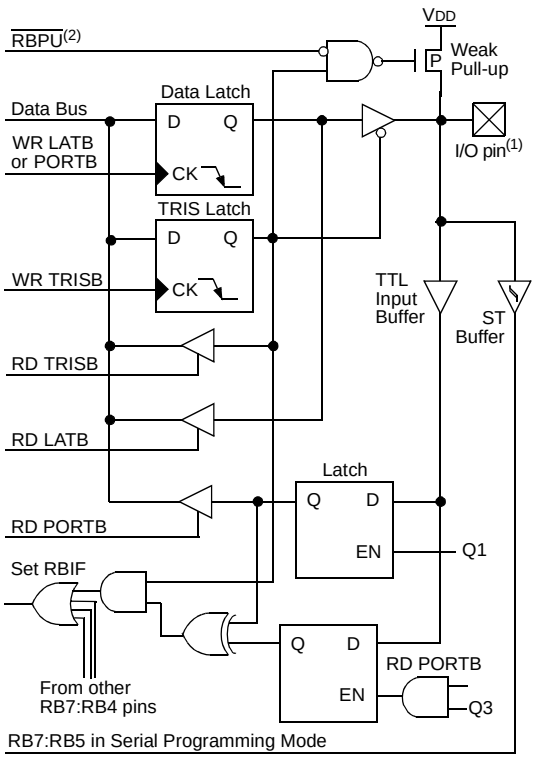
The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

RB3 can be configured by the configuration bit CCP2MX, as the alternate peripheral pin for the CCP2 module. This is only available when the device is configured in Microprocessor, Microprocessor with Boot Block, or Extended Microcontroller operating modes.

The RB5 pin is used as the LVP programming pin. When the LVP configuration bit is programmed, this pin loses the I/O function and become a programming test function.

Note:	When LVP is enabled, the weak pull-up on RB5 is disabled.

FIGURE 10-5: BLOCK DIAGRAM OF RB7:RB4 PINS

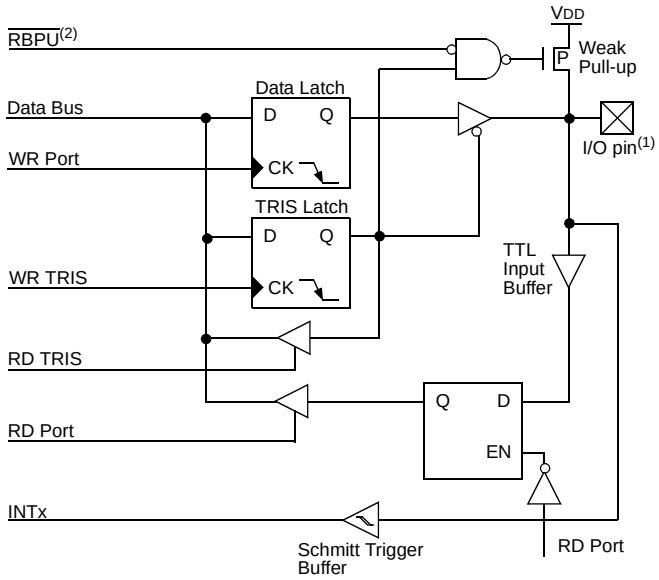


Note 1: I/O pins have diode protection to VDD and VSS.

2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit (INTCON2<7>).

© 2004 Microchip Technology Inc.

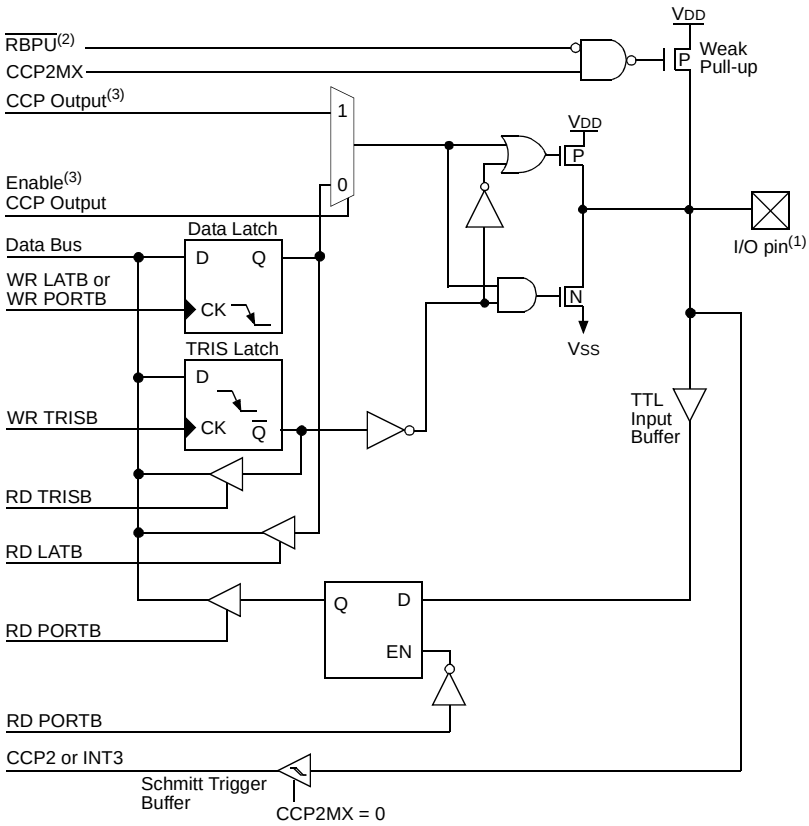
FIGURE 10-6: BLOCK DIAGRAM OF RB2:RB0 PINS



Note 1: I/O pins have diode protection to VDD and VSS.

2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit (OPTION_REG<7>).

FIGURE 10-7: BLOCK DIAGRAM OF RB3 PIN



Note 1: I/O pin has diode protection to VDD and VSS.

2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit (INTCON2<7>).

3: The CCP2 input/output is multiplexed with RB3 if the CCP2MX bit is enabled (= 0) in the Configuration register and the device is operating in Microprocessor, Microprocessor with Boot Block or Extended Microcontroller mode.

TABLE 10-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/INT0	bit 0	TTL/ST(1)	Input/output pin or external interrupt input 0. Internal software programmable weak pull-up.
RB1/INT1	bit 1	TTL/ST(1)	Input/output pin or external interrupt input 1. Internal software programmable weak pull-up.
RB2/INT2	bit 2	TTL/ST(1)	Input/output pin or external interrupt input 2. Internal software programmable weak pull-up.
RB3/INT3/CCP2(3)	bit 3	TTL/ST(4)	Input/output pin or external interrupt input 3. Capture2 input/Compare2 output/PWM output (when CCP2MX configuration bit is enabled, all PIC18F8X20 operating modes except Microcontroller mode). Internal software programmable weak pull-up.
RB4/KBI0	bit 4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5/KBI1/PGM	bit 5	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Low-voltage ICSP enable pin.
RB6/KBI2/PGC	bit 6	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock.
RB7/KBI3/PGD	bit 7	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.

2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

3: RC1 is the alternate assignment for CCP2 when CCP2MX is not set (all operating modes except Microcontroller mode).

4: This buffer is a Schmitt Trigger input when configured as the CCP2 input.

TABLE 10-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
LATB	LATB Data Output Register								xxxx xxxx	uuuu uuuu
TRISB	PORTB Data Direction Register								1111 1111	1111 1111
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	1111 1111
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	1100 0000

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

© 2004 Microchip Technology Inc.

10.3 PORTC, TRISC and LATC Registers

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

The Data Latch register (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 10-5). PORTC pins have Schmitt Trigger input buffers.

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

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

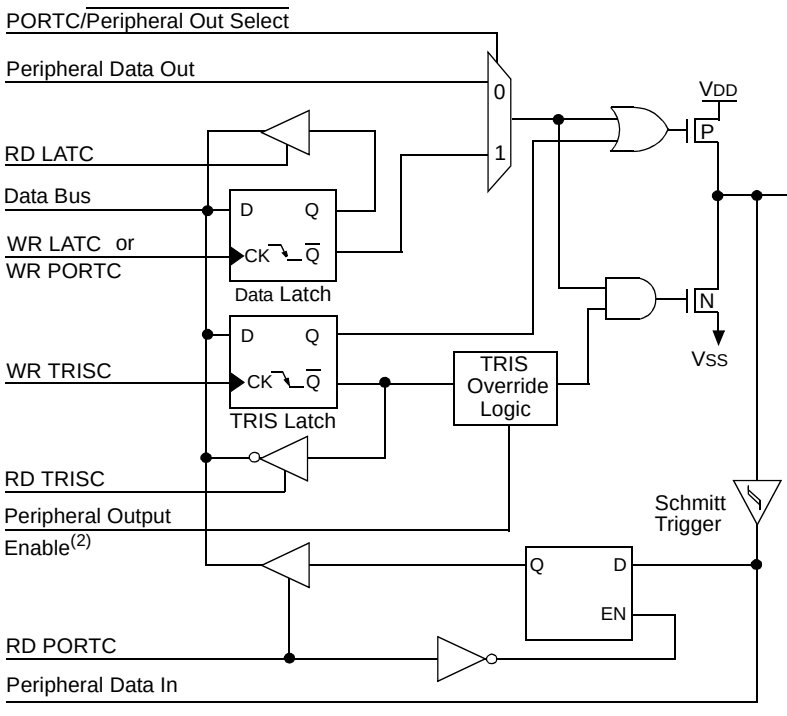
The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register, without concern due to peripheral overrides.

RC1 is normally configured by configuration bit, CCP2MX, as the default peripheral pin of the CCP2 module (default/erased state, CCP2MX = 1).

EXAMPLE 10-3: INITIALIZING PORTC

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

FIGURE 10-8: PORTC BLOCK DIAGRAM (PERIPHERAL OUTPUT OVERRIDE)



TRIS OVERRIDE

Pin	Override	Peripheral
RC0	Yes	Timer1 Osc for Timer1/Timer3
RC1	Yes	Timer1 Osc for Timer1/Timer3, CCP2 I/O
RC2	Yes	CCP1 I/O
RC3	Yes	SPI/I^2C Master Clock
RC4	Yes	I^2C Data Out
RC5	Yes	SPI Data Out
RC6	Yes	USART1 Async Xmit, Sync Clock
RC7	Yes	USART1 Sync Data Out

Note 1: I/O pins have diode protection to VDD and VSS.

2: Peripheral Output Enable is only active if Peripheral Select is active.

TABLE 10-5: PORTC FUNCTIONS

Name	Bit#	Buffer Type	Function
RC0/T1OSO/T13CKI	bit 0	ST	Input/output port pin, Timer1 oscillator output or Timer1/Timer3 clock input.
RC1/T1OSI/CCP2(1)	bit 1	ST	Input/output port pin, Timer1 oscillator input or Capture2 input/ Compare2 output/PWM output (when CCP2MX configuration bit is disabled).
RC2/CCP1	bit 2	ST	Input/output port pin or Capture1 input/Compare1 output/ PWM1 output.
RC3/SCK/SCL	bit 3	ST	RC3 can also be the synchronous serial clock for both SPI and I^2C modes.
RC4/SDI/SDA	bit 4	ST	RC4 can also be the SPI data in (SPI mode) or data I/O (I^2C mode).
RC5/SDO	bit 5	ST	Input/output port pin or synchronous serial port data output.
RC6/TX1/CK1	bit 6	ST	Input/output port pin, addressable USART1 asynchronous transmit or addressable USART1 synchronous clock.
RC7/RX1/DT1	bit 7	ST	Input/output port pin, addressable USART1 asynchronous receive or addressable USART1 synchronous data.

Legend: ST = Schmitt Trigger input

Note 1: RB3 is the alternate assignment for CCP2 when CCP2MX is set.

TABLE 10-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	uuuu uuuu
LATC	LATC Data Output Register								xxxx xxxx	uuuu uuuu
TRISC	PORTC Data Direction Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged

© 2004 Microchip Technology Inc.

10.4 PORTD, TRISD and LATD Registers

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

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

PORTD is an 8-bit port with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

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

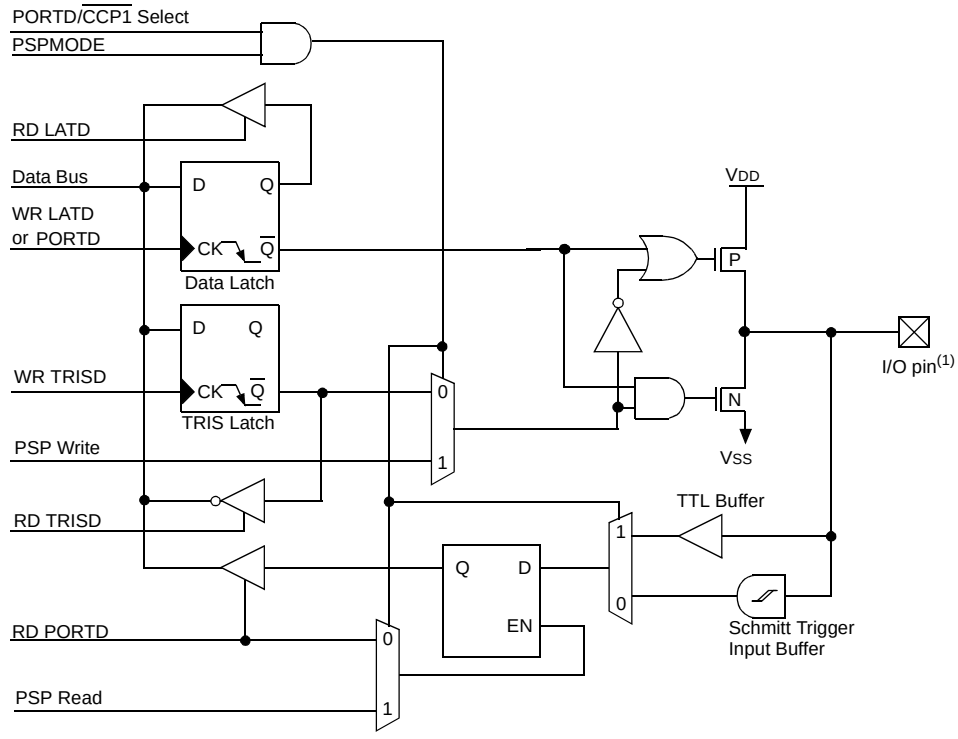
PORTD is multiplexed with the system bus as the external memory interface; I/O port functions are only available when the system bus is disabled, by setting the EBDIS bit in the MEMCOM register (MEMCON<7>). When operating as the external memory interface, PORTD is the low-order byte of the multiplexed address/data bus (AD7:AD0).

PORTD can also be configured as an 8-bit wide microprocessor port (Parallel Slave Port) by setting control bit PSPMODE (TRISE<4>). In this mode, the input buffers are TTL. See Section 10.10 “Parallel Slave Port” for additional information on the Parallel Slave Port (PSP).

EXAMPLE 10-4: INITIALIZING PORTD

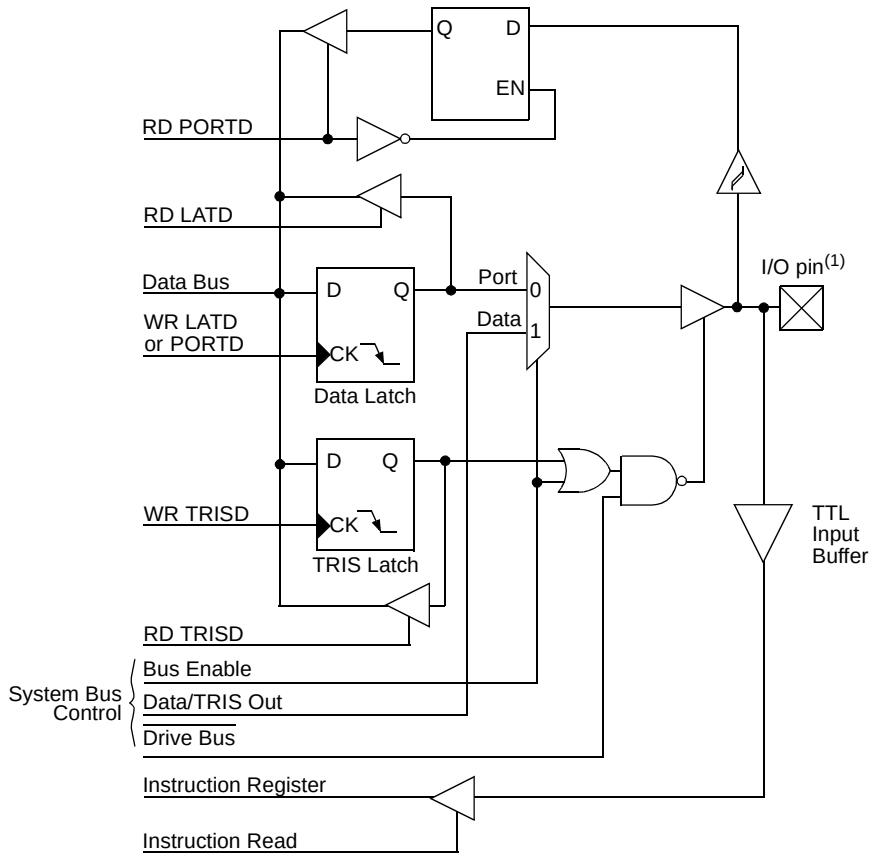
```
CLRF    PORTD    ; Initialize PORTD by
                 ; clearing output
                 ; data latches
CLRF    LATD     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0xCF     ;Value used to
                 ; initialize data
                 ; direction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

FIGURE 10-9: PORTD BLOCK DI AGRAM IN I/O PORT MODE



Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-10: PORTD BLOCK DIAGRAM IN SYSTEM BUS MODE



Note 1: I/O pins have protection diodes to VDD and VSS.

© 2004 Microchip Technology Inc.

TABLE 10-7: PORTD FUNCTIONS

Name	Bit#	Buffer Type	Function
RD0/PSP0/AD0	bit 0	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 0 or address/data bus bit 0.
RD1/PSP1/AD1	bit 1	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 1 or address/data bus bit 1.
RD2/PSP2/AD2	bit 2	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 2 or address/data bus bit 2.
RD3/PSP3/AD3	bit 3	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 3 or address/data bus bit 3.
RD4/PSP4/AD4	bit 4	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 4 or address/data bus bit 4.
RD5/PSP5/AD5	bit 5	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 5 or address/data bus bit 5.
RD6/PSP6/AD6	bit 6	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 6 or address/data bus bit 6.
RD7/PSP7/AD7	bit 7	ST/TTL(1)	Input/output port pin, Parallel Slave Port bit 7 or address/data bus bit 7.

Legend: ST = Schmitt Trigger input, TTL = TTL input

Note 1: Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in System Bus or Parallel Slave Port mode.

TABLE 10-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	uuuu uuuu
LATD	LATD Data Output Register								xxxx xxxx	uuuu uuuu
TRISD	PORTD Data Direction Register								1111 1111	1111 1111
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	0000 ----
MEMCON	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	0-00 --00

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by PORTD.

10.5 PORTE, TRISE and LATE Registers

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

Read-modify-write operations on the LATE register, read and write the latched output value for PORTE.

PORTE is an 8-bit port with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output. PORTE is multiplexed with the CCP module (Table 10-9).

On PIC18F8X20 devices, PORTE is also multiplexed with the system bus as the external memory interface; the I/O bus is available only when the system bus is disabled, by setting the EBDIS bit in the MEMCON register (MEMCON<7>). If the device is configured in Microprocessor or Extended Microcontroller mode, then the PORTE<7:0> becomes the high byte of the address/data bus for the external program memory interface. In Microcontroller mode, the PORTE<2:0> pins become the control inputs for the Parallel Slave Port when bit PSPMODE (PSPCON<4>) is set. (Refer to Section 4.1.1 “PIC18F8X20 Program Memory Modes” for more information on program memory modes.)

When the Parallel Slave Port is active, three PORTE pins (RE0/$\overline{RD}$/AD8, RE1/$\overline{WR}$/AD9 and RE2/$\overline{CS}$/AD10) function as its control inputs. This automatically occurs when the PSPMODE bit (PSPCON<4>) is set. Users must also make certain that bits TRISE<2:0> are set to configure the pins as digital inputs and the ADCON1 register is configured for digital I/O. The PORTE PSP control functions are summarized in Table 10-9.

Pin RE7 can be configured as the alternate peripheral pin for CCP module 2 when the device is operating in Microcontroller mode. This is done by clearing the configuration bit, CCP2MX, in configuration register, CONFIG3H (CONFIG3H<0>).

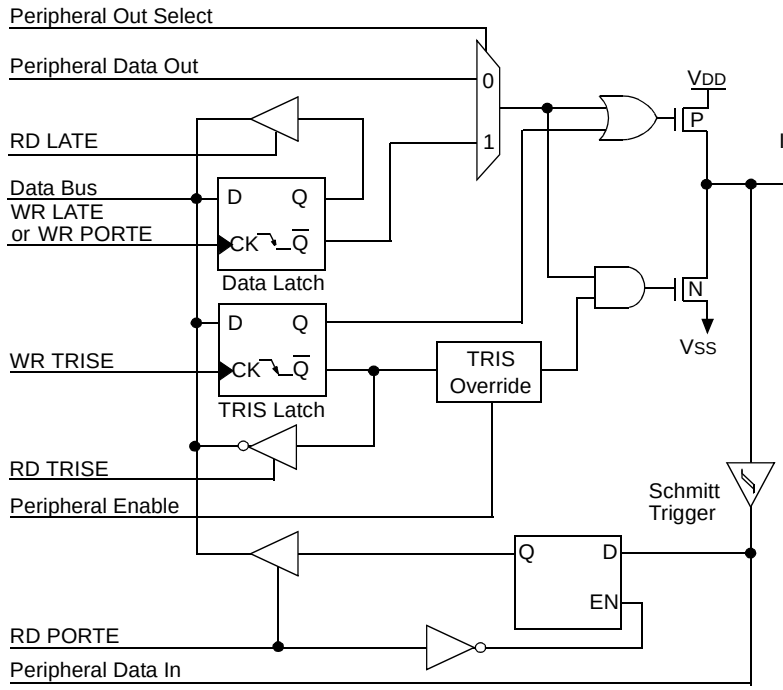
Note:	For PIC18F8X20 (80-pin) devices operating in Extended Microcontroller mode, PORTE defaults to the system bus on Power-on Reset.

EXAMPLE 10-5: INITIALIZING PORTE

```
CLRF    PORTE    ; Initialize PORTE by
                 ; clearing output
                 ; data latches
CLRF    LATE     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0x03     ;   Value used to
                 ;initializedata
                 ; direction
MOVWF   TRISE    ;   Set RE1:RE0 as inputs
                 ; RE7:RE2 as outputs
```


© 2004 Microchip Technology Inc.

FIGURE 10-11: PORTE BLOCK DIAGRAM IN I/O MODE

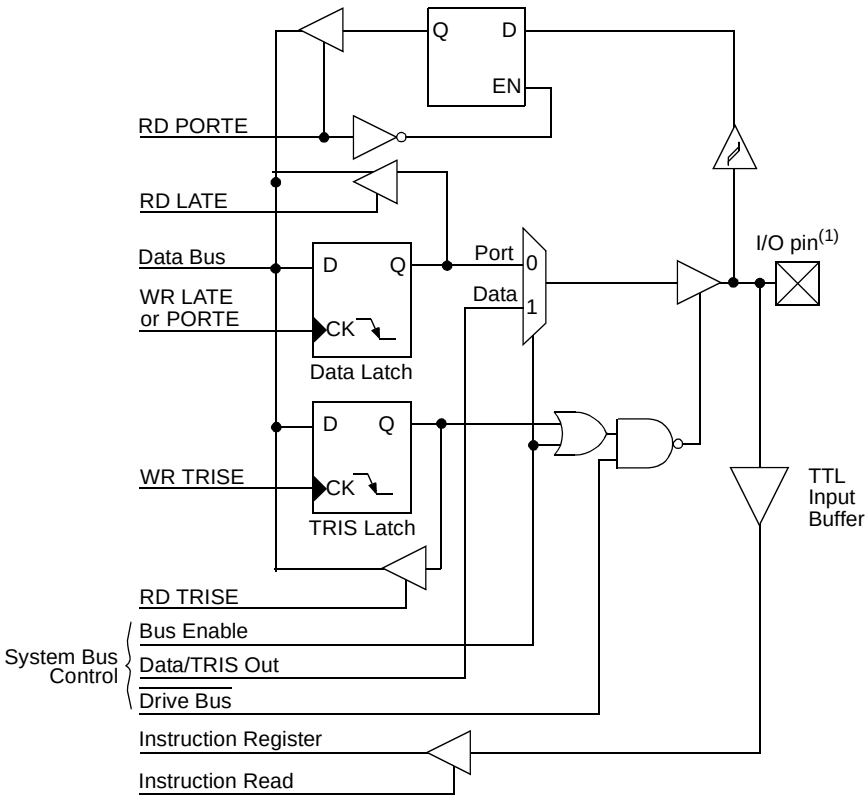


TRIS OVERRIDE

Pin	Override	Peripheral
RE0	Yes	External Bus
RE1	Yes	External Bus
RE2	Yes	External Bus
RE3	Yes	External Bus
RE4	Yes	External Bus
RE5	Yes	External Bus
RE6	Yes	External Bus
RE7	Yes	External Bus

Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-12: PORTE BLOCK DIAGRAM IN SYSTEM BUS MODE



Note 1: I/O pins have protection diodes to VDD and VSS.

TABLE 10-9: PORTE FUNCTIONS

Name	Bit#	Buffer Type	Function
RE0/$\overline{RD}$/AD8	bit 0	ST/TTL(1)	Input/output port pin, read control for Parallel Slave Port or address/data bit 8 For $\overline{RD}$ (PSP Control mode): 1 = Not a read operation 0 = Read operation, reads PORTD register (if chip selected)
RE1/$\overline{WR}$/AD9	bit 1	ST/TTL(1)	Input/output port pin, write control for Parallel Slave Port or address/data bit 9 For $\overline{WR}$ (PSP Control mode): 1 = Not a write operation 0 = Write operation, writes PORTD register (if chip selected)
RE2/$\overline{CS}$/AD10	bit 2	ST/TTL(1)	Input/output port pin, chip select control for Parallel Slave Port or address/data bit 10 For $\overline{CS}$ (PSP Control mode): 1 = Device is not selected 0 = Device is selected
RE3/AD11	bit 3	ST/TTL(1)	Input/output port pin or address/data bit 11.
RE4/AD12	bit 4	ST/TTL(1)	Input/output port pin or address/data bit 12.
RE5/AD13	bit 5	ST/TTL(1)	Input/output port pin or address/data bit 13.
RE6/AD14	bit 6	ST/TTL(1)	Input/output port pin or address/data bit 14.
RE7/CCP2/AD15	bit 7	ST/TTL(1)	Input/output port pin, Capture2 input/Compare2 output/PWM output (PIC18F8X20 devices in Microcontroller mode only) or address/data bit 15.

Legend: ST = Schmitt Trigger input, TTL = TTL input

Note 1: Input buffers are Schmitt Triggers when in I/O or CCP mode and TTL buffers when in System Bus or PSP Control mode.

TABLE 10-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TRISE	PORTE Data Direction Control Register								1111 1111	1111 1111
PORTE	Read PORTE pin/Write PORTE Data Latch								xxxx xxxx	uuuu uuuu
LATE	Read PORTE Data Latch/Write PORTE Data Latch								xxxx xxxx	uuuu uuuu
MEMCON	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	0000 --00
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	0000 ----

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTE.

© 2004 Microchip Technology Inc.

10.6 PORTF, LATF and TRISF Registers

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

Read-modify-write operations on the LATF register, read and write the latched output value for PORTF.

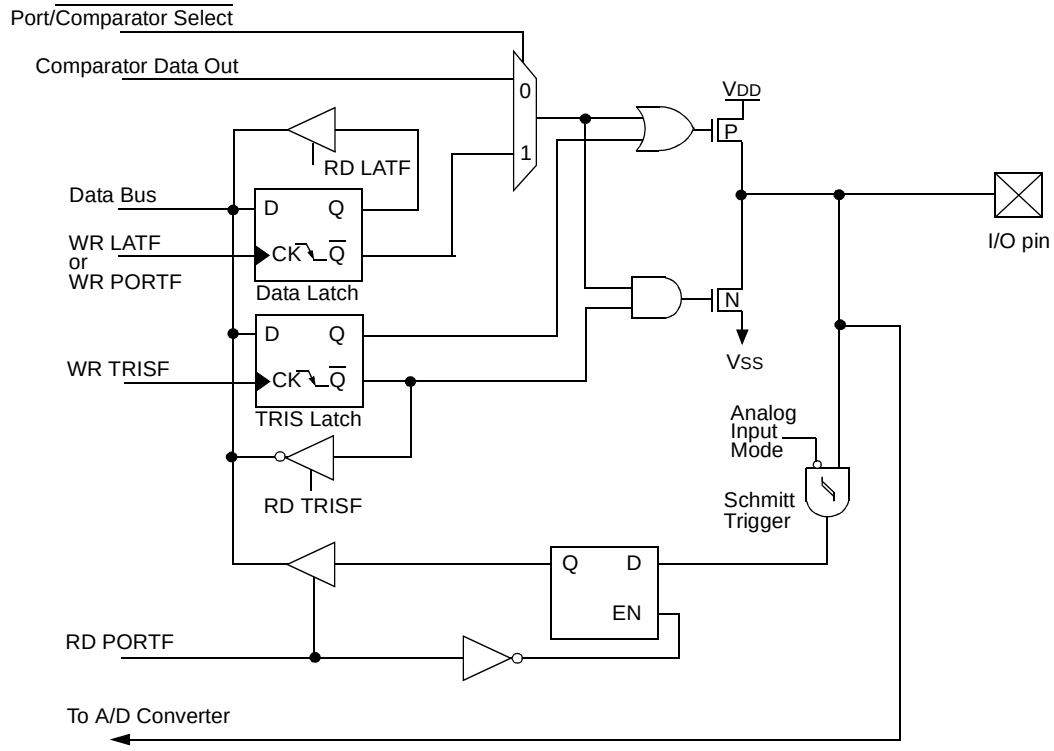
PORTF is multiplexed with several analog peripheral functions, including the A/D converter inputs and comparator inputs, outputs and voltage reference.

Note 1:	On a Power-on Reset, the RF6:RF0 pins are configured as inputs and read as '0'.
2:	To configure PORTF as digital I/O, turn off comparators and set ADCON1 value.

EXAMPLE 10-6: INITIALIZING PORTF

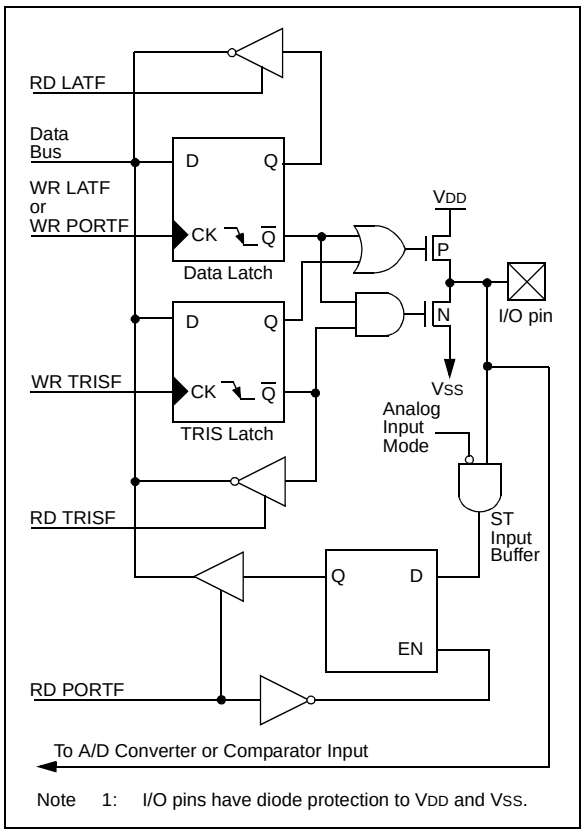
```
CLRF    PORTF    ; Initialize PORTF by
                 ; clearing output
                 ; data latches
CLRF    LATF     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0x07     ;
MOVWF   CMCON    ; Turn off comparators
MOVLW   0x0F     ;
MOVWF   ADCON1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

FIGURE 10-13: PORTF RF1/AN6/C2OUT, RF2/AN 7/C1OUT PINS BLOCK DIAGRAM



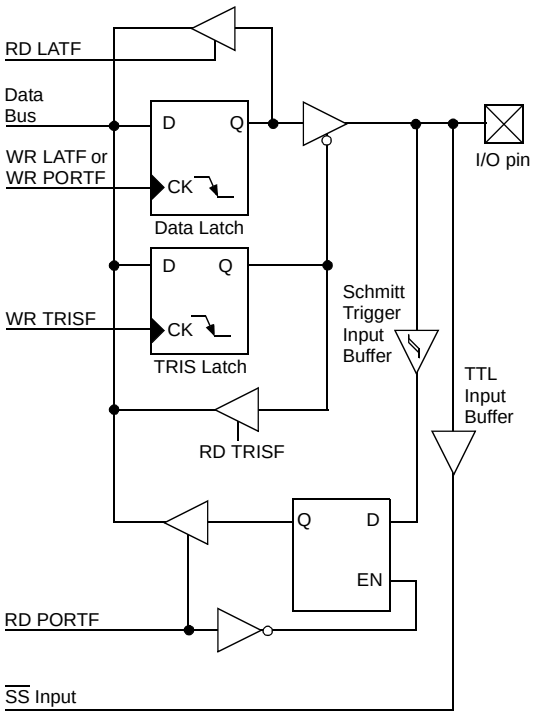
Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-14: RF6:RF3 AND RF0 PINS BLOCK DIAGRAM



Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-15: RF7 PIN BLOCK DIAGRAM



Note: I/O pins have diode protection to VDD and VSS.

© 2004 Microchip Technology Inc.

TABLE 10-11: PORTF FUNCTIONS

Name	Bit#	Buffer Type	Function
RF0/AN5	bit 0	ST	Input/output port pin or analog input.
RF1/AN6/C2OUT	bit 1	ST	Input/output port pin, analog input or comparator 2 output.
RF2/AN7/C1OUT	bit 2	ST	Input/output port pin, analog input or comparator 1 output.
RF3/AN8	bit 3	ST	Input/output port pin or analog input/comparator input.
RF4/AN9	bit 4	ST	Input/output port pin or analog input/comparator input.
RF5/AN10/CVREF	bit 5	ST	Input/output port pin, analog input/comparator input or comparator reference output.
RF6/AN11	bit 6	ST	Input/output port pin or analog input/comparator input.
RF7/$\overline{SS}$	bit 7	ST/TTL	Input/output port pin or slave select pin for synchronous serial port.

Legend: ST = Schmitt Trigger input, TTL = TTL input

TABLE 10-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TRISF	PORTF Data Direction Control Register								1111 1111	1111 1111
PORTF	Read PORTF pin/Write PORTF Data Latch								xxxx xxxx	uuuu uuuu
LATF	Read PORTF Data Latch/Write PORTF Data Latch								0000 0000	uuuu uuuu
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	--00 0000
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	0000 0000

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTF.

10.7 PORTG, TRISG and LATG Registers

PORTG is a 5-bit wide, bidirectional port. The corresponding data direction register is TRISG. Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISG bit (= 0) will make the corresponding PORTC pin an output (i.e., put the contents of the output latch on the selected pin).

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

PORTG is multiplexed with both CCP and USART functions (Table 10-13). 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.

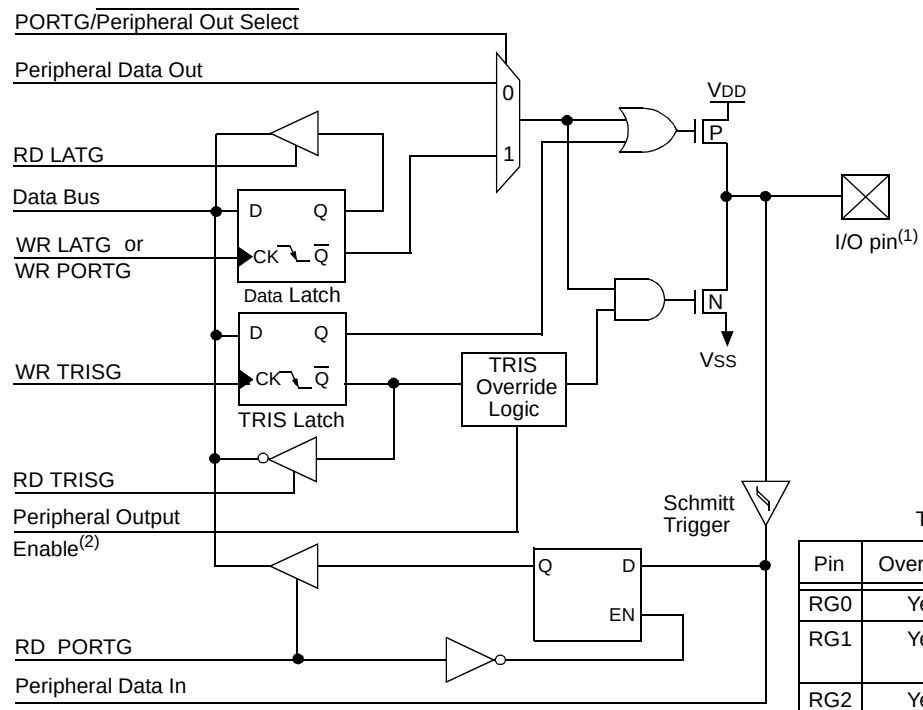
Note:	On a Power-on Reset, these pins are configured as digital inputs.

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 10-7: INITIALIZING PORTG

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

FIGURE 10-16: PORTG BLOCK DIAGRAM (PERIPHERAL OUTPUT OVERRIDE)



TRIS OVERRIDE

Pin	Override	Peripheral
RG0	Yes	CCP3 I/O
RG1	Yes	USART1 Async Xmit, Sync Clock
RG2	Yes	USART1 Async Rcv, Sync Data Out
RG3	Yes	CCP4 I/O
RG4	Yes	CCP5 I/O

Note 1: I/O pins have diode protection to VDD and VSS.

2: Peripheral Output Enable is only active if Peripheral Select is active.

© 2004 Microchip Technology Inc.

TABLE 10-13: PORTG FUNCTIONS

Name	Bit#	Buffer Type	Function
RG0/CCP3	bit 0	ST	Input/output port pin or Capture3 input/Compare3 output/PWM3 output.
RG1/TX2/CK2	bit 1	ST	Input/output port pin, addressable USART2 asynchronous transmit or addressable USART2 synchronous clock.
RG2/RX2/DT2	bit 2	ST	Input/output port pin, addressable USART2 asynchronous receive or addressable USART2 synchronous data.
RG3/CCP4	bit 3	ST	Input/output port pin or Capture4 input/Compare4 output/PWM4 output.
RG4/CCP5	bit 4	ST	Input/output port pin or Capture5 input/Compare5 output/PWM5 output.

Legend: ST = Schmitt Trigger input

TABLE 10-14: SUMMARY OF REGISTERS ASSOCIATED WITH PORTG

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTG	—	—	—	Read PORTF pin/Write PORTF Data Latch					---x xxxx	---u uuuu
LATG	—	—	—	LATG Data Output Register					---x xxxx	---u uuuu
TRISG	—	—	—	Data Direction Control Register for PORTG					---1 1111	---1 1111

Legend: x = unknown, u = unchanged

10.8 PORTH, LATH and TRISH Registers

Note:	PORTH is available only on PIC18F8X20 devices.

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

Read-modify-write operations on the LATH register, read and write the latched output value for PORTH.

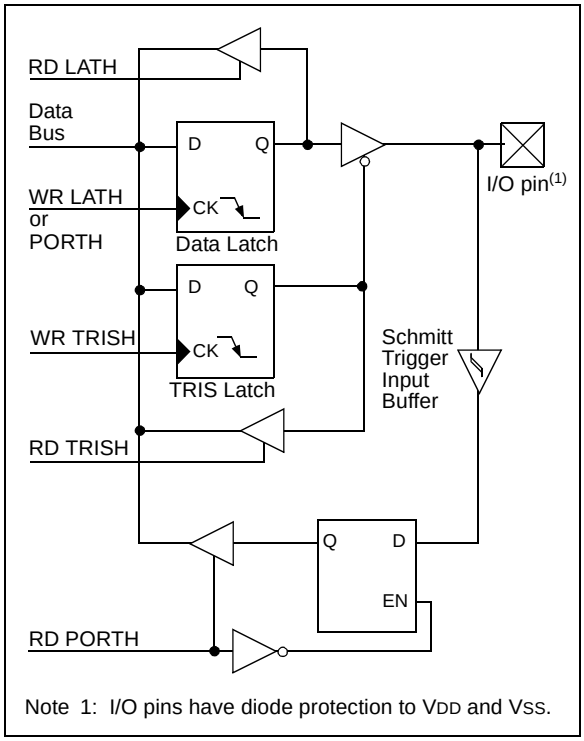
Pins RH7:RH4 are multiplexed with analog inputs AN15:AN12. Pins RH3:RH0 are multiplexed with the system bus as the external memory interface; they are the high-order address bits, A19:A16. By default, pins RH7:RH4 are enabled as A/D inputs and pins RH3:RH0 are enabled as the system address bus. Register ADCON1 configures RH7:RH4 as I/O or A/D inputs. Register MEMCON configures RH3:RH0 as I/O or system bus pins.

Note 1:	On Power-on Reset, PORTH pins RH7:RH4 default to A/D inputs and read as '0'.
2:	On Power-on Reset, PORTH pins RH3:RH0 default to system bus signals.

EXAMPLE 10-8: INITIALIZING PORTH

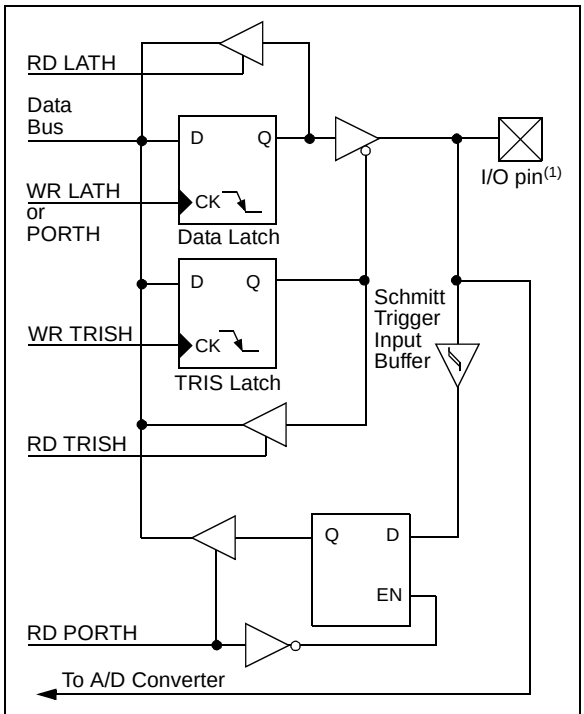
```
CLRF    PORTH    ; Initialize PORTH by
                 ; clearing output
                 ; data latches
CLRF    LATH     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0Fh      ;
MOVWF   ADCON1   ;
MOVLW   0CFh     ; Value used to
                 ; initialize data
                 ; direction
MOVWF   TRISH    ; Set RH3:RH0 as inputs
                 ; RH5:RH4 as outputs
                 ; RH7:RH6 as inputs
```

FIGURE 10-17: RH3:RH0 PINS BLOCK DIAGRAM IN I/O MODE



Note 1: I/O pins have diode protection to VDD and VSS.

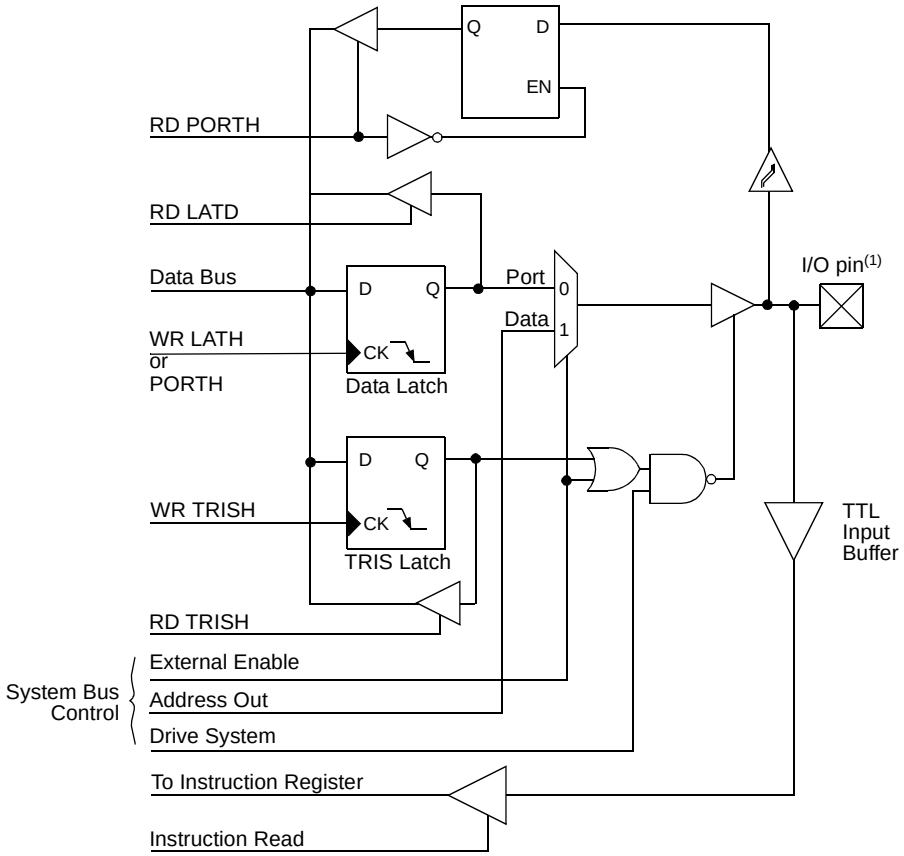
FIGURE 10-18: RH7:RH4 PINS BLOCK DIAGRAM IN I/O MODE



Note 1: I/O pins have diode protection to VDD and VSS.

© 2004 Microchip Technology Inc.

FIGURE 10-19: RH3:RH0 PINS BLOC K DIAGRAM IN SYSTEM BUS MODE



Note 1: I/O pins have diode protection to VDD and VSS.

TABLE 10-15: PORTH FUNCTIONS

Name	Bit#	Buffer Type	Function
RH0/A16	bit 0	ST/TTL(1)	Input/output port pin or address bit 16 for external memory interface.
RH1/A17	bit 1	ST/TTL(1)	Input/output port pin or address bit 17 for external memory interface.
RH2/A18	bit 2	ST/TTL(1)	Input/output port pin or address bit 18 for external memory interface.
RH3/A19	bit 3	ST/TTL(1)	Input/output port pin or address bit 19 for external memory interface.
RH4/AN12	bit 4	ST	Input/output port pin or analog input channel 12.
RH5/AN13	bit 5	ST	Input/output port pin or analog input channel 13.
RH6/AN14	bit 6	ST	Input/output port pin or analog input channel 14.
RH7/AN15	bit 7	ST	Input/output port pin or analog input channel 15.

Legend: ST = Schmitt Trigger input, TTL = TTL input

Note 1: Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in System Bus or Parallel Slave Port mode.

TABLE 10-16: SUMMARY OF REGISTERS ASSOCIATED WITH PORTH

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TRISH	PORTH Data Direction Control Register								1111 1111	1111 1111
PORTH	Read PORTH pin/Write PORTH Data Latch								xxxx xxxx	uuuu uuuu
LATH	Read PORTH Data Latch/Write PORTH Data Latch								xxxx xxxx	uuuu uuuu
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	--00 0000
MEMCON	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	0-00 --00

Legend: x = unknown, u = unchanged, – = unimplemented. Shaded cells are not used by PORTH.

© 2004 Microchip Technology Inc.

10.9 PORTJ, TRISJ and LATJ Registers

Note:	PORTJ is available only on PIC18F8X20 devices.

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

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

PORTJ is multiplexed with the system bus as the external memory interface; I/O port functions are only available when the system bus is disabled. When operating as the external memory interface, PORTJ provides the control signal to external memory devices. The RJ5 pin is not multiplexed with any system bus functions.

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

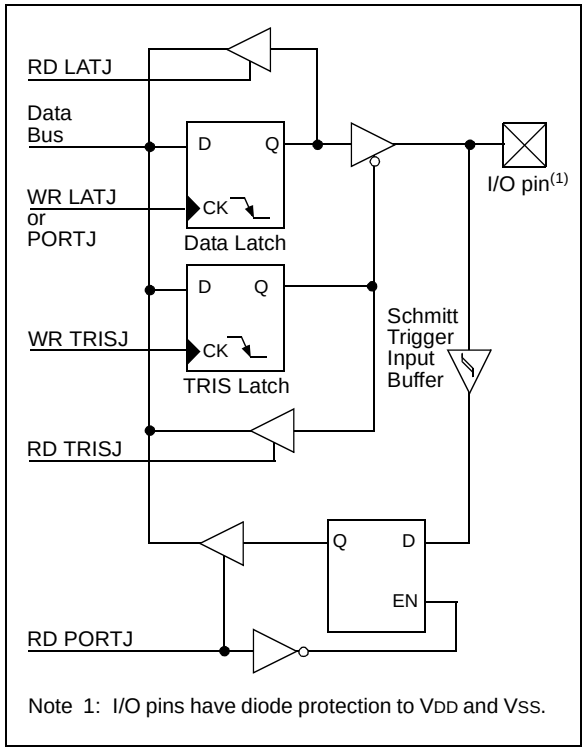
Note:	On a Power-on Reset, these pins are configured as digital inputs.

The 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 10-9: INITIALIZING PORTJ

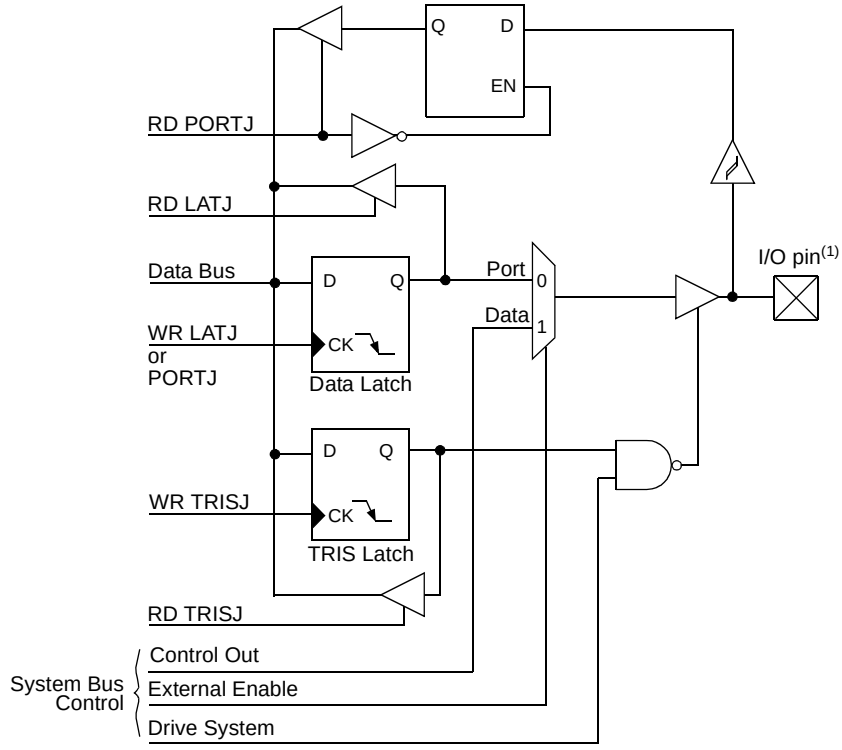
```
CLRF    PORTJ    ; Initialize PORTG by
                 ; clearing output
                 ; data latches
CLRF    LATJ     ; Alternate method
                 ; to clear output
                 ; data latches
MOVLW   0xCF     ;Value used to
                 ;initialize data
                 ; direction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

FIGURE 10-20: PORTJ BLOCK DIAGRAM IN I/O MODE



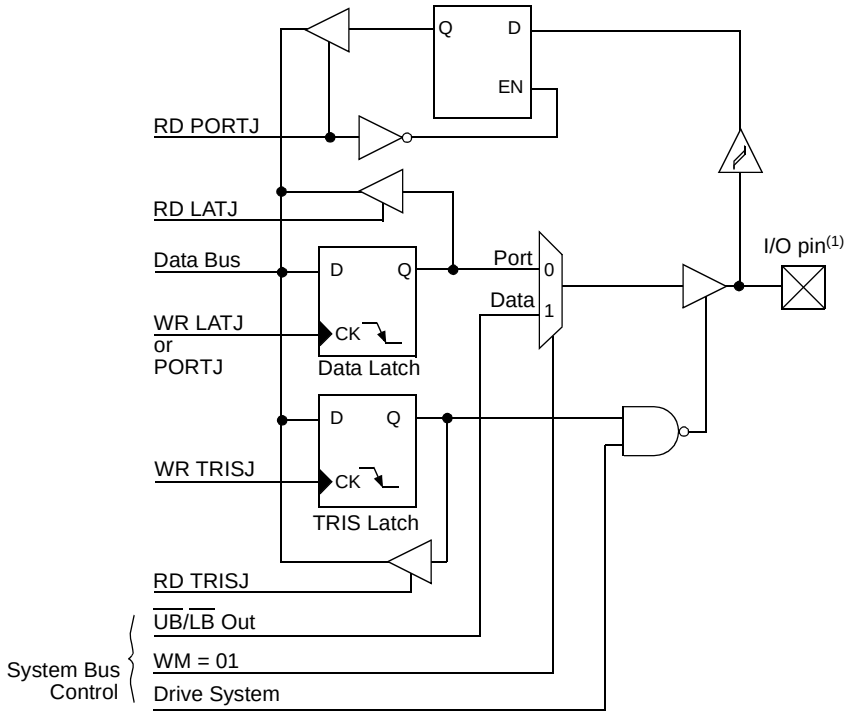
Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-21: RJ4:RJ0 PINS BL OCK DIAGRAM IN SYSTEM BUS MODE



Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-22: RJ7:RJ6 PINS BLO CK DIAGRAM IN SYSTEM BUS MODE



Note 1: I/O pins have diode protection to VDD and VSS.

© 2004 Microchip Technology Inc.

TABLE 10-17: PORTJ FUNCTIONS

Name	Bit#	Buffer Type	Function
RJ0/ALE	bit 0	ST	Input/output port pin or address latch enable control for external memory interface.
RJ1/$\overline{OE}$	bit 1	ST	Input/output port pin or output enable control for external memory interface.
RJ2/$\overline{WRL}$	bit 2	ST	Input/output port pin or write low byte control for external memory interface.
RJ3/$\overline{WRH}$	bit 3	ST	Input/output port pin or write high byte control for external memory interface.
RJ4/BA0	bit 4	ST	Input/output port pin or byte address 0 control for external memory interface.
RJ5/$\overline{CE}$	bit 5	ST	Input/output port pin or chip enable control for external memory interface.
RJ6/$\overline{LB}$	bit 6	ST	Input/output port pin or lower byte select control for external memory interface.
RJ7/$\overline{UB}$	bit 7	ST	Input/output port pin or upper byte select control for external memory interface.

Legend: ST = Schmitt Trigger input

TABLE 10-18: SUMMARY OF REGISTERS ASSOCIATED WITH PORTJ

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTJ	Read PORTJ pin/Write PORTJ Data Latch								xxxx xxxx	uuuu uuuu
LATJ	LATJ Data Output Register								xxxx xxxx	uuuu uuuu
TRISJ	Data Direction Control Register for PORTJ								1111 1111	1111 1111

Legend: x = unknown, u = unchanged

10.10 Parallel Slave Port

PORTD also operates 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/$\overline{RD}$/AD8 and the $\overline{WR}$ control input pin, RE1/$\overline{WR}$/AD9.

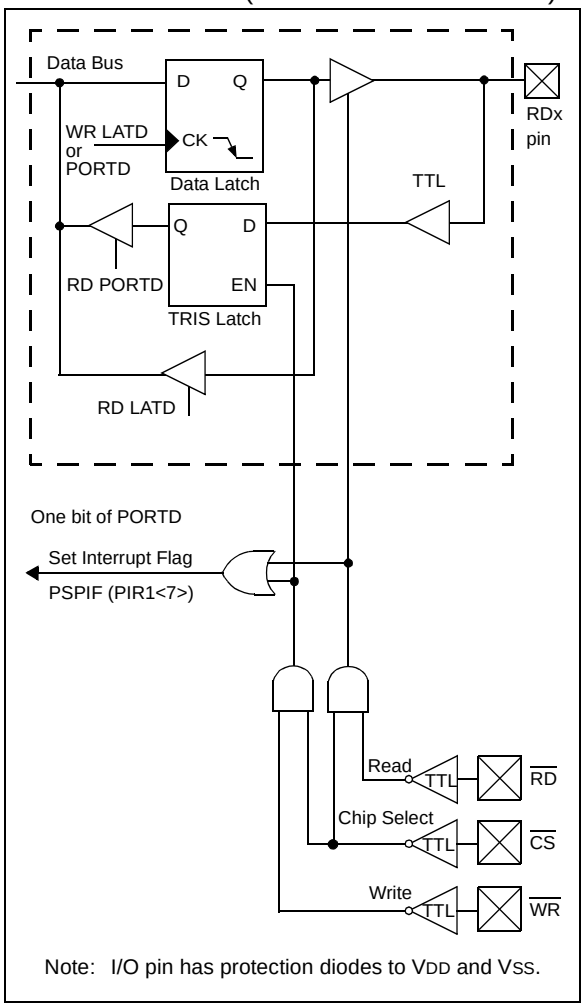
Note:	For PIC18F8X20 devices, the Parallel Slave Port is available only in Microcontroller mode.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an 8-bit latch. Setting bit PSPMODE enables port pin RE0/$\overline{RD}$/AD8 to be the $\overline{RD}$ input, RE1/$\overline{WR}$/AD9 to be the $\overline{WR}$ input and RE2/$\overline{CS}$/AD10 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). The A/D port configuration bits, PCFG2:PCFG0 (ADCON1<2:0>), must be set which will configure pins RE2:RE0 as digital I/O.

A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low. A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low.

The PORTE I/O pins become control inputs for the microprocessor port when bit PSPMODE (PSPCON<4>) is set. In this mode, the user must make sure that the TRISE<2:0> bits are set (pins are configured as digital inputs) and the ADCON1 is configured for digital I/O. In this mode, the input buffers are TTL.

FIGURE 10-23: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: I/O pin has protection diodes to V_{DD} and V_{SS}.

© 2004 Microchip Technology Inc.

REGISTER 10-1: PSPCON 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

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’

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

FIGURE 10-24: PARALLEL SLAVE PORT WRITE WAVEFORMS

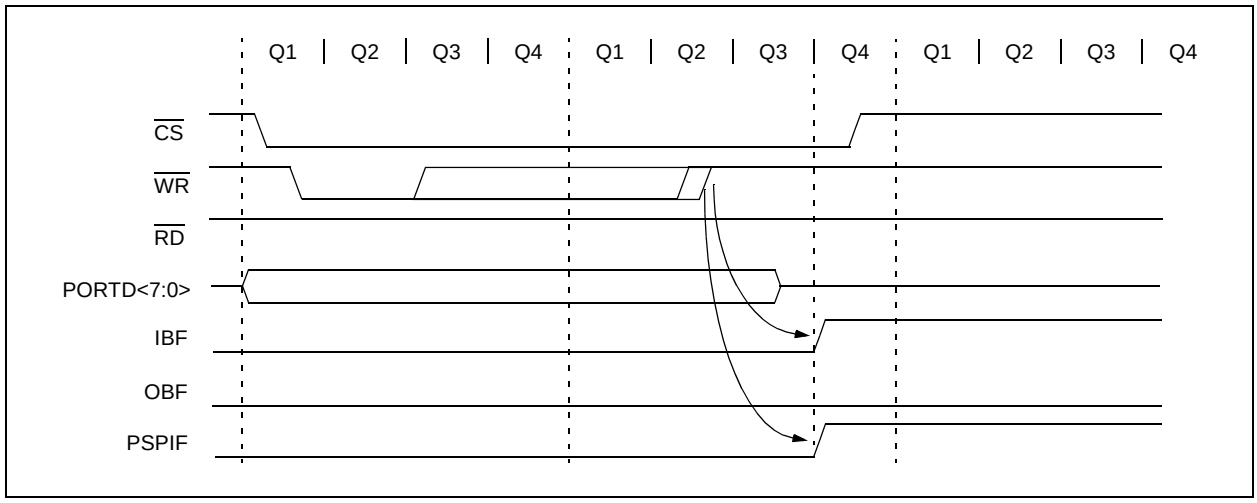


FIGURE 10-25: PARALLEL SLAVE PORT READ WAVEFORMS

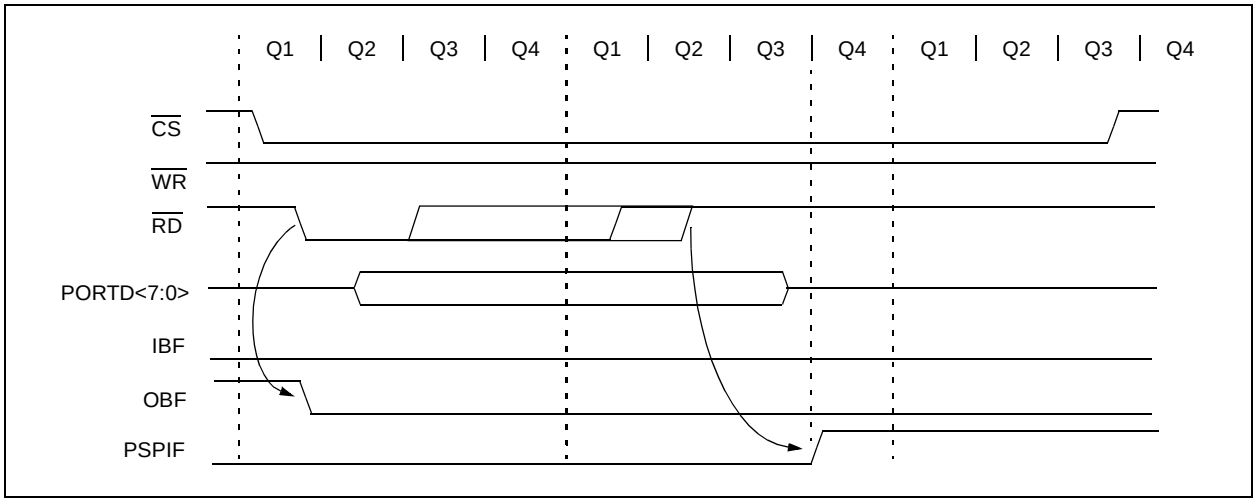


TABLE 10-19: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
PORTD	Port Data Latch when written; Port pins when read								xxxx xxxx	uuuu uuuu
LATD	LATD Data Output bits								xxxx xxxx	uuuu uuuu
TRISD	PORTD Data Direction bits								1111 1111	1111 1111
PORTE	—	—	—	—	—	Read PORTE pin/ Write PORTE Data Latch			0000 0000	0000 0000
LATE	—	—	—	—	—	LATE Data Output bits			xxxx xxxx	uuuu uuuu
TRISE	—	—	—	—	—	PORTE Data Direction bits			1111 1111	1111 1111
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	0000 ----
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IF	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the Parallel Slave Port.

Note 1: Enabled only in Microcontroller mode for PIC18F8X20 devices.

© 2004 Microchip Technology Inc.

11.0 TIMER0 MODULE

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler
- Clock source selectable to be external or internal
- Interrupt-on-overflow from FFh to 00h in 8-bit mode and FFFFh to 0000h in 16-bit mode
- Edge select for external clock

Figure 11-1 shows a simplified block diagram of the Timer0 module in 8-bit mode and Figure 11-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

The T0CON register (Register 11-1) is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

REGISTER 11-1: 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

bit 7 TMR0ON: Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 T08BIT: Timer0 8-bit/16-bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 T0CS: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

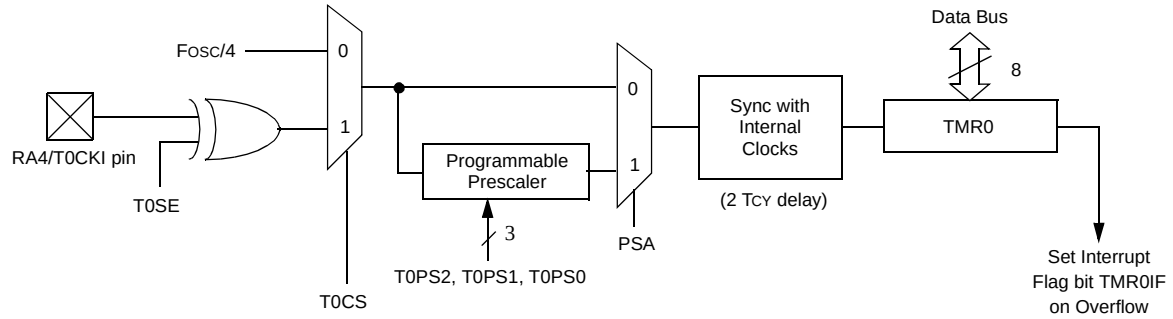
bit 4 T0SE: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

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

bit 2-0 T0PS2:T0PS0: Timer0 Prescaler Select bits
111 = 1:256 prescale value
110 = 1:128 prescale value
101 = 1:64 prescale value
100 = 1:32 prescale value
011 = 1:16 prescale value
010 = 1:8 prescale value
001 = 1:4 prescale value
000 = 1:2 prescale value

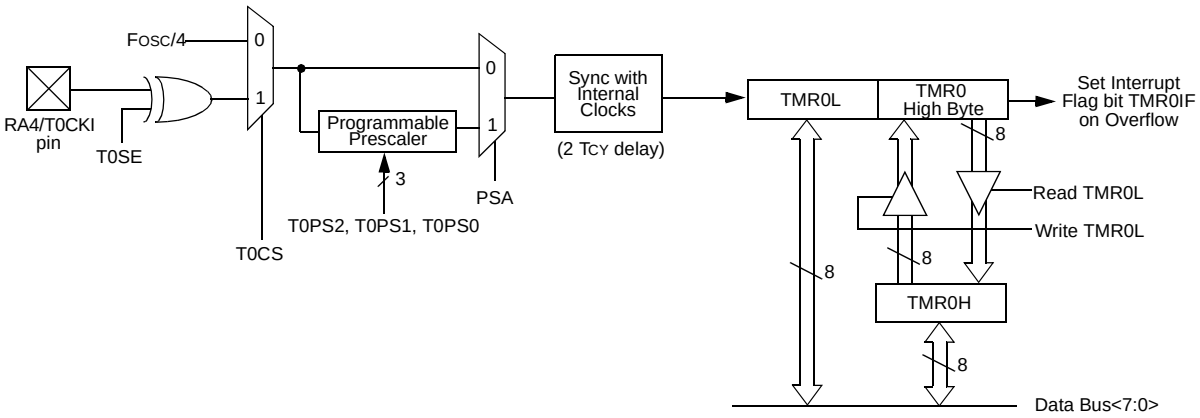
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

FIGURE 11-1: TIMER0 BLOCK DIAGRAM IN 8-BIT MODE



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 11-2: TIMER0 BLOCK DIAGRAM IN 16-BIT MODE



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

© 2004 Microchip Technology Inc.

11.1 Timer0 Operation

Timer0 can operate as a timer or as a counter.

Timer mode is selected by clearing the T0CS bit. In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, 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.

Counter mode is selected by setting the T0CS bit. In Counter mode, Timer0 will increment, either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit (T0SE). Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed below.

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

11.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not readable or writable.

The PSA and T0PS2:T0PS0 bits determine the prescaler assignment and prescale ratio.

Clearing bit PSA will assign the prescaler to the Timer0 module. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4, ..., 1:256 are selectable.

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

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

11.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control, (i.e., it can be changed “on-the-fly” during program execution).

11.3 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF bit. The interrupt can be masked by clearing the TMR0IE bit. The TMR0IF bit must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Sleep, since the timer is shut-off during Sleep.

11.4 16-Bit Mode Timer Reads and Writes

TMR0H is not the high byte of the timer/counter in 16-bit mode, but is actually a buffered version of the high byte of Timer0 (refer to Figure 11-2). The high byte of the Timer0 counter/timer is not directly readable nor writable. TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

A write to the high byte of Timer0 must also take place through the TMR0H Buffer register. Timer0 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.

TABLE 11-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TMR0L	Timer0 Module Low Byte Register								xxxx xxxx	uuuu uuuu
TMR0H	Timer0 Module High Byte Register								0000 0000	0000 0000
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	1111 1111
TRISA	—	PORTA Data Direction Register							-111 1111	-111 1111

Legend: x = unknown, u = unchanged, – = unimplemented locations, read as ‘0’.
Shaded cells are not used by Timer0.

NOTES:

© 2004 Microchip Technology Inc.

12.0 TIMER1 MODULE

The Timer1 module timer/counter has the following features:

- 16-bit timer/counter
  (two 8-bit registers: TMR1H and TMR1L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module special event trigger

Figure 12-1 is a simplified block diagram of the Timer1 module.

Register 12-1 details the Timer1 Control register. This register controls the operating mode of the Timer1 module and contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

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.

REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 RD16: 16-bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 Unimplemented: Read as ‘0’

bit 5-4 T1CKPS1:T1CKPS0: Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 T1OSCEN: Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 $\overline{\text{T1SYNC}}$: 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 pin RC0/T1OSO/T13CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 TMR1ON: Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

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

12.1 Timer1 Operation

Timer1 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

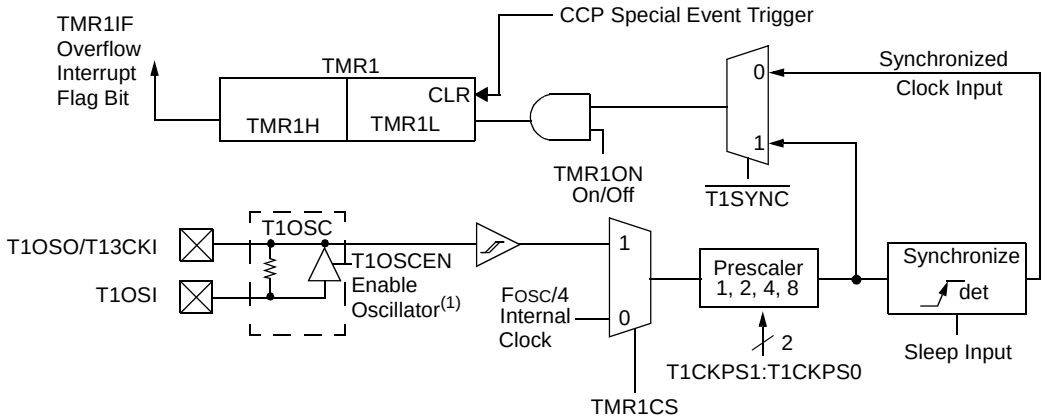
The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

When TMR1CS = 0, Timer1 increments every instruction cycle. When TMR1CS = 1, Timer1 increments on every rising edge of the external clock input or the Timer1 oscillator, if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. That is, the TRISC<1:0> value is ignored and the pins are read as '0'.

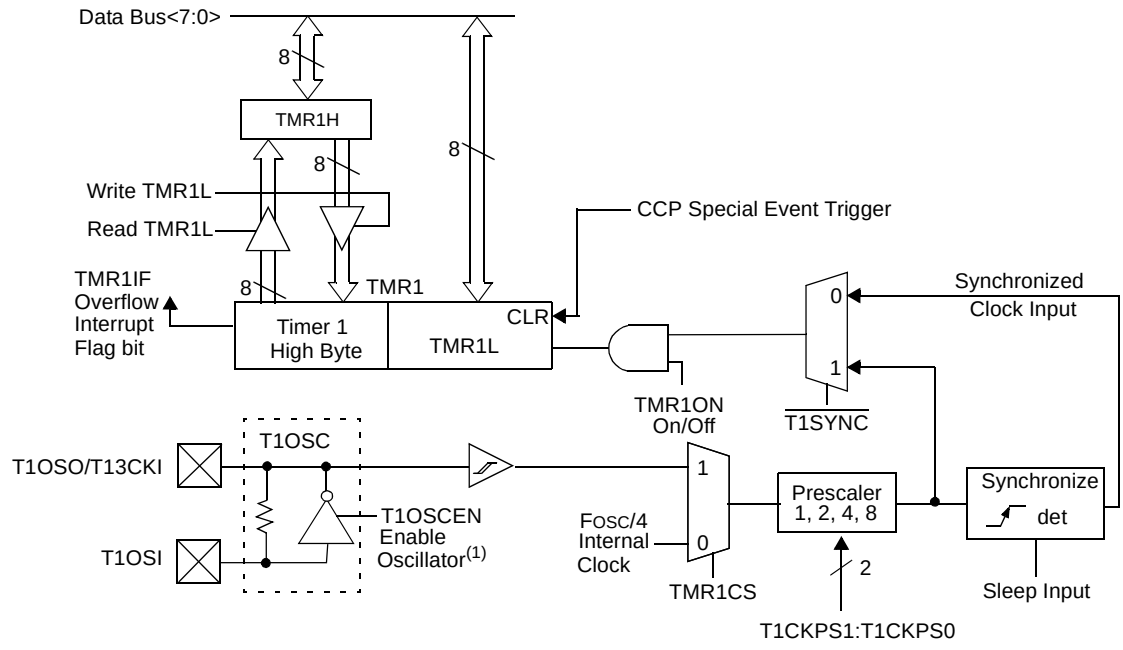
Timer1 also has an internal "Reset input". This Reset can be generated by the CCP module (see Section 16.0 "Capture/Compare/PWM (CCP) Modules").

FIGURE 12-1: TIMER1 BLOCK DIAGRAM



Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

FIGURE 12-2: TIMER1 BLOCK DIAG RAM: 16-BIT READ/WRITE MODE



Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

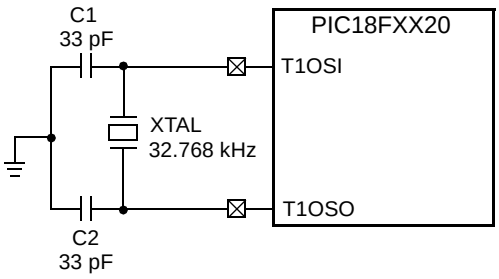
© 2004 Microchip Technology Inc.

12.2 Timer1 Oscillator

A crystal oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator, rated up to 200 kHz. It will continue to run during Sleep. It is primarily intended for a 32 kHz crystal. The circuit for a typical LP oscillator is shown in Figure 12-3. Table 12-1 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 12-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the Notes with Table 12-1 for additional information about capacitor selection.

TABLE 12-1: CAPACITOR SELECTION FOR THE ALTERNATE OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	TBD(1)	TBD(1)
Crystal to be Tested:			
32.768 kHz	Epson C-001R32.768K-A		± 20 PPM

Note 1: Microchip suggests 33 pF 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.

12.2.1 LOW-POWER TIMER1 OPTION (PIC18FX520 DEVICES ONLY)

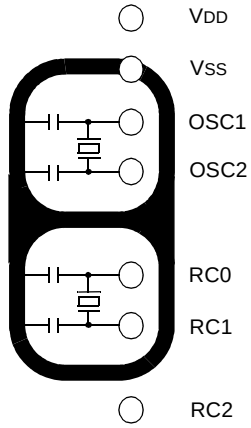
The Timer1 oscillator for PIC18LFX520 devices incorporates a low-power feature, which allows the oscillator to automatically reduce its power consumption when the microcontroller is in Sleep mode.

As high noise environments may cause excessive oscillator instability in Sleep mode, this option is best suited for low noise applications where power conservation is an important design consideration. 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 12-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high-speed circuit must be located near the oscillator (such as the CCP1 pin in output compare or PWM mode, or the primary oscillator using the OSC2 pin), a grounded guard ring around the oscillator circuit, as shown in Figure 12-4, may be helpful when used on a single-sided PCB or in addition to a ground plane.

FIGURE 12-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING



Note: Not drawn to scale.

Note: PIC18FX620/X720 devices have the standard Timer1 oscillator permanently selected. PIC18LFX620/X720 devices have the low-power Timer1 oscillator permanently selected.

12.3 Timer1 Interrupt

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

12.4 Resetting Timer1 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a "special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Note:	The special event triggers from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. 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 from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer1.

12.5 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 12-2). 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. 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, is valid, due to a rollover between reads.

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

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

12.6 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in Section 12.2 "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 12-1, 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 overflow.

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.

© 2004 Microchip Technology Inc.

EXAMPLE 12-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```
RTCinit
        MOVLW   0x80            ; Preload TMR1 register pair
        MOVWF   TMR1H           ; for 1 second overflow
        CLRF    TMR1L
        MOVLW   b'00001111'     ; Configure for external clock,
        MOVWF   T1OSC           ; Asynchronous operation, external oscillator
        CLRF    secs            ; Initialize timekeeping registers
        CLRF    mins            ;
        MOVLW   .12
        MOVWF   hours
        BSF     PIE1, TMR1IE    ; Enable Timer1 interrupt
        RETURN
RTCisr
        BSF     TMR1H, 7        ; Preload for 1 sec overflow
        BCF     PIR1, TMR1IF    ; Clear interrupt flag
        INCF    secs, F         ; Increment seconds
        MOVLW   .59             ; 60 seconds elapsed?
        CPFSGT  secs
        RETURN                  ; No, done
        CLRF    secs            ; Clear seconds
        INCF    mins, F         ; Increment minutes
        MOVLW   .59             ; 60 minutes elapsed?
        CPFSGT  mins
        RETURN                  ; No, done
        CLRF    mins            ; clear minutes
        INCF    hours, F        ; Increment hours
        MOVLW   .23             ; 24 hours elapsed?
        CPFSGT  hours
        RETURN                  ; No, done
        MOVLW   .01             ; Reset hours to 1
        MOVWF   hours
        RETURN                  ; Done
```

TABLE 12-2: 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	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

NOTES:

© 2004 Microchip Technology Inc.

13.0 TIMER2 MODULE

The Timer2 module timer has the following features:

- 8-bit timer (TMR2 register)
- 8-bit period register (PR2)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match of PR2
- SSP module optional use of TMR2 output to generate clock shift

Timer2 has a control register shown in Register 13-1. Timer2 can be shut-off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption. Figure 13-1 is a simplified block diagram of the Timer2 module. Register 13-1 shows the Timer2 Control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

13.1 Timer2 Operation

Timer2 can be used as the PWM time base for the PWM mode of the CCP module. The TMR2 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, T2CKPS1:T2CKPS0 (T2CON<1:0>). The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit, TMR2IF (PIR1<1>)).

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

- 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 13-1: T2CON: TIMER2 CONTROL REGISTER

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

bit 7 Unimplemented: Read as ‘0’

bit 6-3 T2OUTPS3:T2OUTPS0: Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

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

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

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

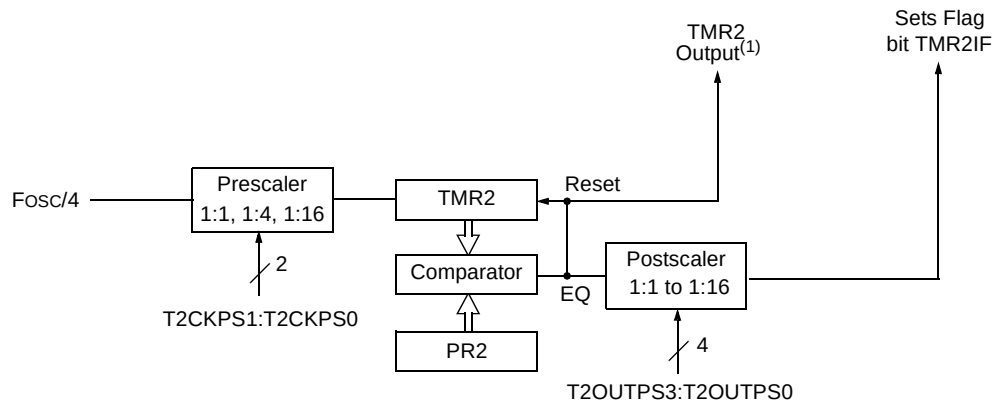
13.2 Timer2 Interrupt

The Timer2 module has an 8-bit period register, PR2. Timer2 increments from 00h until it matches PR2 and then resets to 00h on the next increment cycle. PR2 is a readable and writable register. The PR2 register is initialized to FFh upon Reset.

13.3 Output of TMR2

The output of TMR2 (before the postscaler) is fed to the synchronous serial port module, which optionally uses it to generate the shift clock.

FIGURE 13-1: TIMER2 BLOCK DIAGRAM



Note 1: TMR2 register output can be software selected by the SSP module as a baud clock.

TABLE 13-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
TMR2	Timer2 Module Register								0000 0000	0000 0000
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
PR2	Timer2 Period Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the Timer2 module.

© 2004 Microchip Technology Inc.

14.0 TIMER3 MODULE

The Timer3 module timer/counter has the following features:

- 16-bit timer/counter
  (two 8-bit registers; TMR3H and TMR3L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module trigger

Figure 14-1 is a simplified block diagram of the Timer3 module.

Register 14-1 shows the Timer3 Control register. This register controls the operating mode of the Timer3 module and sets the CCP clock source.

Register 12-1 shows the Timer1 Control register. This register controls the operating mode of the Timer1 module, as well as contains the Timer1 Oscillator Enable bit (T1OSCEN), which can be a clock source for Timer3.

REGISTER 14-1: 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

bit 7 RD16: 16-bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer3 in one 16-bit operation
0 = Enables register read/write of Timer3 in two 8-bit operations

bit 6, 3 T3CCP2:T3CCP1: Timer3 and Timer1 to CCPx Enable bits
11 = Timer3 and Timer4 are the clock sources for CCP1 through CCP5
10 = Timer3 and Timer4 are the clock sources for CCP3 through CCP5;
Timer1 and Timer2 are the clock sources for CCP1 and CCP2
01 = Timer3 and Timer4 are the clock sources for CCP2 through CCP5;
Timer1 and Timer2 are the clock sources for CCP1
00 = Timer1 and Timer2 are the clock sources for CCP1 through CCP5

bit 5-4 T3CKPS1:T3CKPS0: Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 2 $\overline{\text{T3SYNC}}$: Timer3 External Clock Input Synchronization Control bit
(Not usable if the system 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

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

14.1 Timer3 Operation

Timer3 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

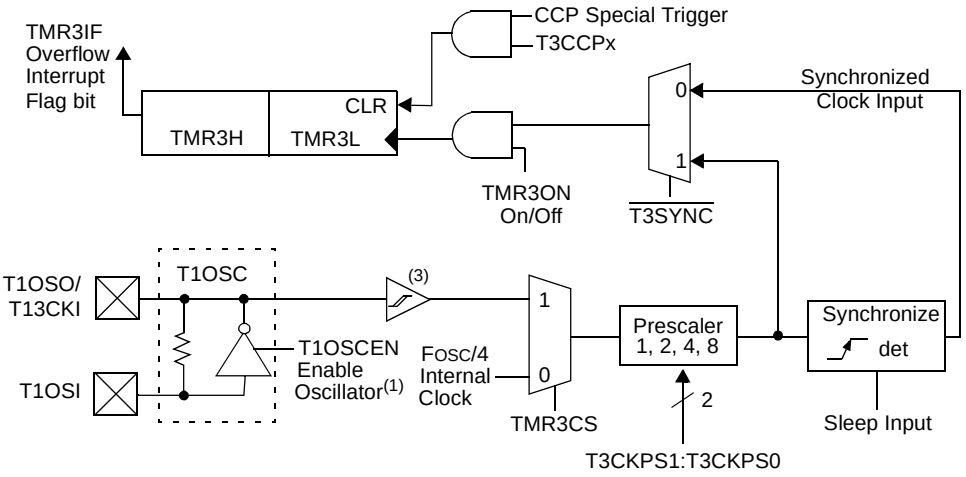
The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>).

When TMR3CS = 0, Timer3 increments every instruction cycle. When TMR3CS = 1, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. That is, the TRISC<1:0> value is ignored and the pins are read as '0'.

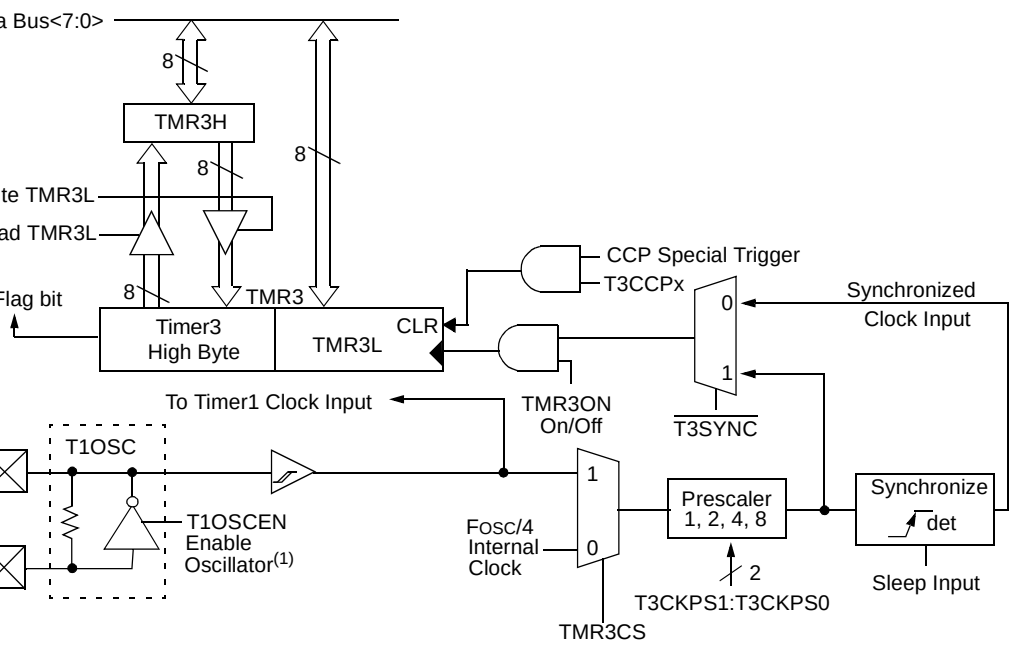
Timer3 also has an internal "Reset input". This Reset can be generated by the CCP module (see Section 14.0 "Timer3 Module").

FIGURE 14-1: TIMER3 BLOCK DIAGRAM



Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

FIGURE 14-2: TIMER3 BLOCK DIAGRAM CO NFIGURED IN 16-BIT READ/WRITE MODE



Note 1: When the T1OSCEN bit is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

© 2004 Microchip Technology Inc.

14.2 Timer1 Oscillator

The Timer1 oscillator may be used as the clock source for Timer3. The Timer1 oscillator is enabled by setting the T1OSCEN (T1CON<3>) bit. The oscillator is a low-power oscillator rated up to 200 kHz. See Section 12.0 “Timer1 Module” for further details.

14.3 Timer3 Interrupt

The TMR3 register pair (TMR3H:TMR3L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR3 interrupt, if enabled, is generated on overflow, which is latched in interrupt flag bit, TMR3IF (PIR2<1>). This interrupt can be enabled/disabled by setting/clearing TMR3 Interrupt Enable bit, TMR3IE (PIE2<1>).

14.4 Resetting Timer3 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a “special event trigger” (CCP1M3:CCP1M0 = 1011), this signal will reset Timer3.

Note:	The special event triggers from the CCP module will not set interrupt flag bit, TMR3IF (PIR1<0>).

Timer3 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer3 is running in Asynchronous Counter mode, this Reset operation may not work. In the event that a write to Timer3 coincides with a special event trigger from CCP1, the write will take precedence. In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer3.

TABLE 14-1: 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	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR2	—	—	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	---0 0000	---0 0000
PIE2	—	—	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	---0 0000	---0 0000
IPR2	—	—	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	---1 1111	---1 1111
TMR3L	Holding Register for the Least Significant Byte of the 16-bit TMR3 Register								xxxx xxxx	uuuu uuuu
TMR3H	Holding Register for the Most Significant Byte of the 16-bit TMR3 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	uuuu uuuu

Legend: x = unknown, u = unchanged, – = unimplemented, read as ‘0’. Shaded cells are not used by the Timer3 module.

NOTES:

© 2004 Microchip Technology Inc.

15.0 TIMER4 MODULE

The Timer4 module timer has the following features:

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

Timer4 has a control register shown in Register 15-1. Timer4 can be shut-off by clearing control bit, TMR4ON (T4CON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer4 are also controlled by this register. Figure 15-1 is a simplified block diagram of the Timer4 module.

15.1 Timer4 Operation

Timer4 can be used as the PWM time base for the PWM mode of the 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 T4CKPS1:T4CKPS0 (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 15-1: 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

bit 7 Unimplemented: Read as ‘0’

bit 6-3 T4OUTPS3:T4OUTPS0: 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 T4CKPS1:T4CKPS0: Timer4 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

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

15.2 Timer4 Interrupt

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

15.3 Output of TMR4

The output of TMR4 (before the postscaler) is used only as a PWM time base for the CCP modules. It is not used as a baud rate clock for the MSSP, as is the Timer2 output.

FIGURE 15-1: TIMER4 BLOCK DIAGRAM

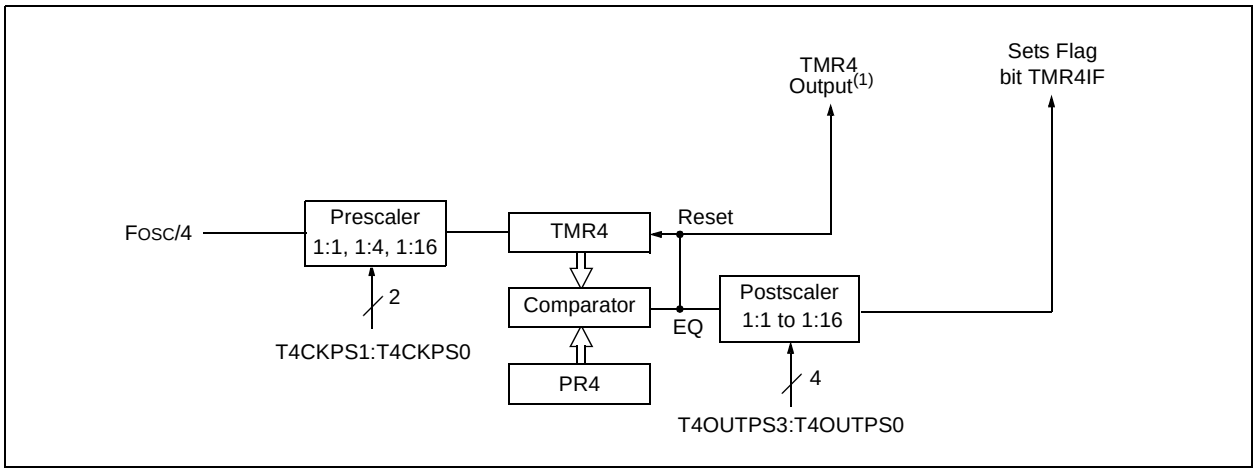


TABLE 15-1: REGISTERS ASSOCIATED WITH TIMER4 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--00 0000
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
TMR4	Timer4 Module Register								0000 0000	0000 0000
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	-000 0000
PR4	Timer4 Period Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the Timer4 module.

© 2004 Microchip Technology Inc.

16.0 CAPTURE/COMPARE/PWM (CCP) MODULES

The PIC18FXX20 devices all have five CCP (Capture/Compare/PWM) modules. Each module contains a 16-bit register, which can operate as a 16-bit Capture register, a 16-bit Compare register or a Pulse Width Modulation (PWM) Master/Slave Duty Cycle register. Table 16-1 shows the timer resources of the CCP module modes.

The operation of all CCP modules are identical, with the exception of the special event trigger present on CCP1 and CCP2.

For the sake of clarity, CCP module operation in the following sections is described with respect to CCP1. The descriptions can be applied (with the exception of the special event triggers) to any of the modules.

Note:	Throughout this section, 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 CCP1, CCP2, CCP3, CCP4 or CCP5.

REGISTER 16-1: CCPxCON REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 Unimplemented: Read as '0'

bit 5-4 DCxB1:DCxB0: PWM Duty Cycle bit 1 and bit 0 for CCP Module x

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two Least Significant bits (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight Most Significant bits (DCx9:DCx2) of the duty cycle are found in CCPRxL.

bit 3-0 CCPxM3:CCPxM0: CCP Module x 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 CCP pin Low; on compare match, force CCP pin High (CCPIF bit is set)
1001 = Compare mode, Initialize CCP pin High; on compare match, force CCP pin Low (CCPIF bit is set)
1010 = Compare mode, Generate software interrupt on compare match (CCPIF bit is set, (CCP pin is unaffected)
1011 = Compare mode, trigger special event (CCPIF bit is set):

For CCP1 and CCP2:
Timer1 or Timer3 is reset on event.

For all other modules:
CCPx pin is unaffected and is configured as an I/O port (same as CCPxM<3:0> = 1010 , above).

11xx = PWM mode

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

16.1 CCP Module Configuration

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

16.1.1 CCP MODULES AND TIMER RESOURCES

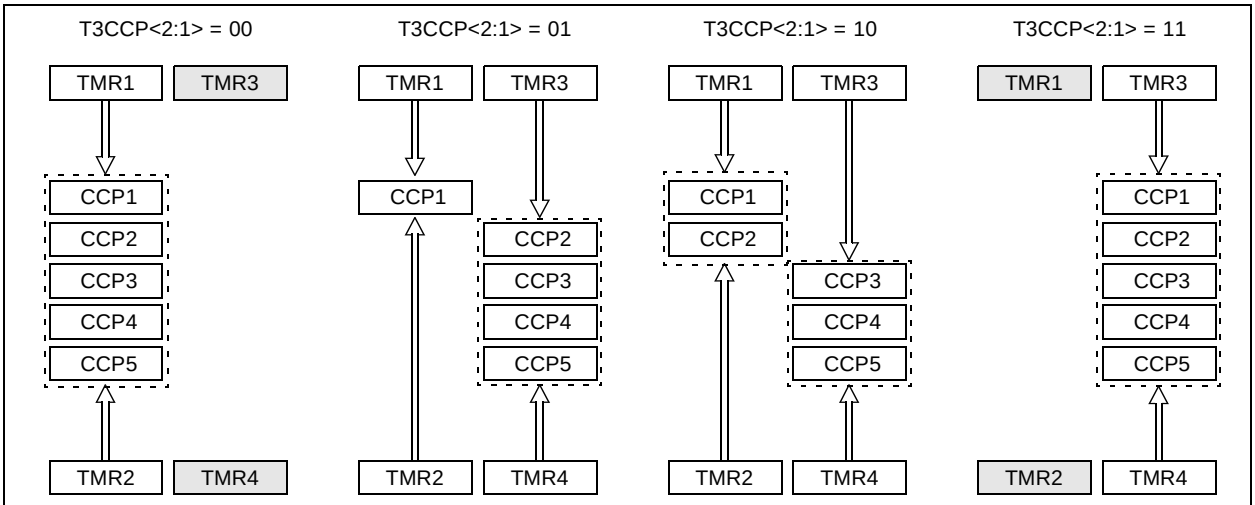
The CCP modules utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode.

TABLE 16-1: CCP MODE – TIMER RESOURCE

CCP 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-CCP Enable bits in the T3CON register (Register 14-1). Depending on the configuration selected, up to four timers may be active at once, with modules in the same configuration (Capture/Compare or PWM) sharing timer resources. The possible configurations are shown in Figure 16-1.

FIGURE 16-1: CCP AND TIMER INTERCONNECT CONFIGURATIONS



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

Timer3 and Timer4 are not available.

Timer1 and Timer2 are used for Capture and Compare or PWM operations for CCP1 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 CCP1 and CCP2 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 CCP modules. Timer4 is used for PWM operations for all CCP modules. Modules may share either timer resource as a common time base.

Timer1 and Timer2 are not available.

© 2004 Microchip Technology Inc.

16.2 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on pin RC2/CCP1. 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 control bits, CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit, CCP1IF (PIR1<2>), is set; it must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten by the new captured value.

16.2.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1 pin should be configured as an input by setting the TRISC<2> bit.

Note:	If the RC2/CCP1 is configured as an output, a write to the port can cause a capture condition.

16.2.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (Timer1 and/or Timer3) must be running in Timer mode, or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation may not work. The timer to be used with each CCP module is selected in the T3CON register (see Section 16.1.1 “CCP Modules and Timer Resources”).

16.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP1IE (PIE1<2>) clear to avoid false interrupts and should clear the flag bit, CCP1IF, following any such change in operating mode.

16.2.4 CCP PRESCALER

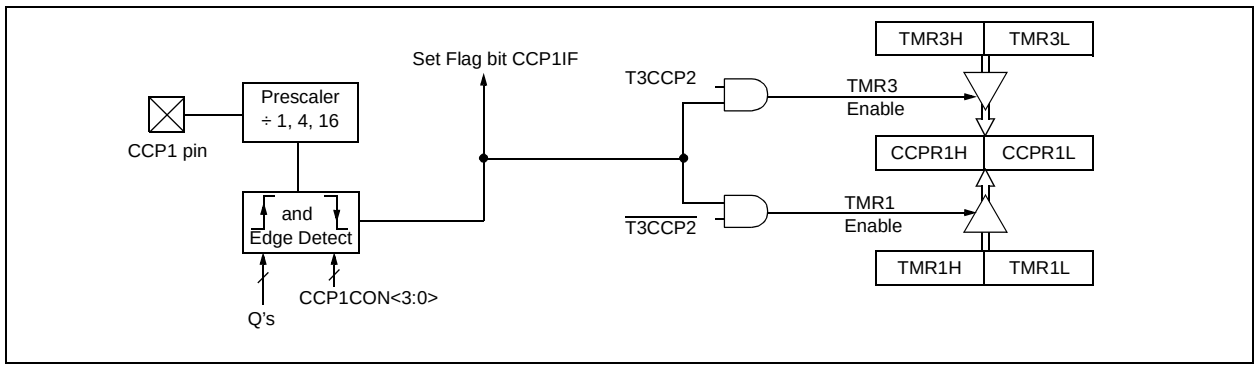
There are four prescaler settings, specified by bits CCP1M3:CCP1M0. Whenever the CCP module is turned off or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 16-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 16-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF    CCP1CON, F    ; Turn CCP module off
MOVLW   NEW_CAPT_PS   ; Load WREG with the
                      ; new prescaler mode
                      ; value and CCP ON
MOVWF   CCP1CON       ; Load CCP1CON with
                      ; this value
```

FIGURE 16-2: CAPTURE MODE OPERATION BLOCK DIAGRAM



16.3 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against either the TMR1 register pair value or the TMR3 register pair value. When a match occurs, the CCP1 pin:

- is driven High
- is driven Low
- toggles output (high-to-low or low-to-high)
- remains unchanged

The action on the pin is based on the value of control bits, CCP1M3:CCP1M0. At the same time, interrupt flag bit CCP1IF (CCP2IF) is set.

16.3.1 CCP PIN CONFIGURATION

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

Note:	Clearing the CCP1CON register will force the RC2/CCP1 compare output latch to the default low level. This is not the PORTC I/O data latch.

16.3.2 TIMER1/TIMER3 MODE SELECTION

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

16.3.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

16.3.4 SPECIAL EVENT TRIGGER

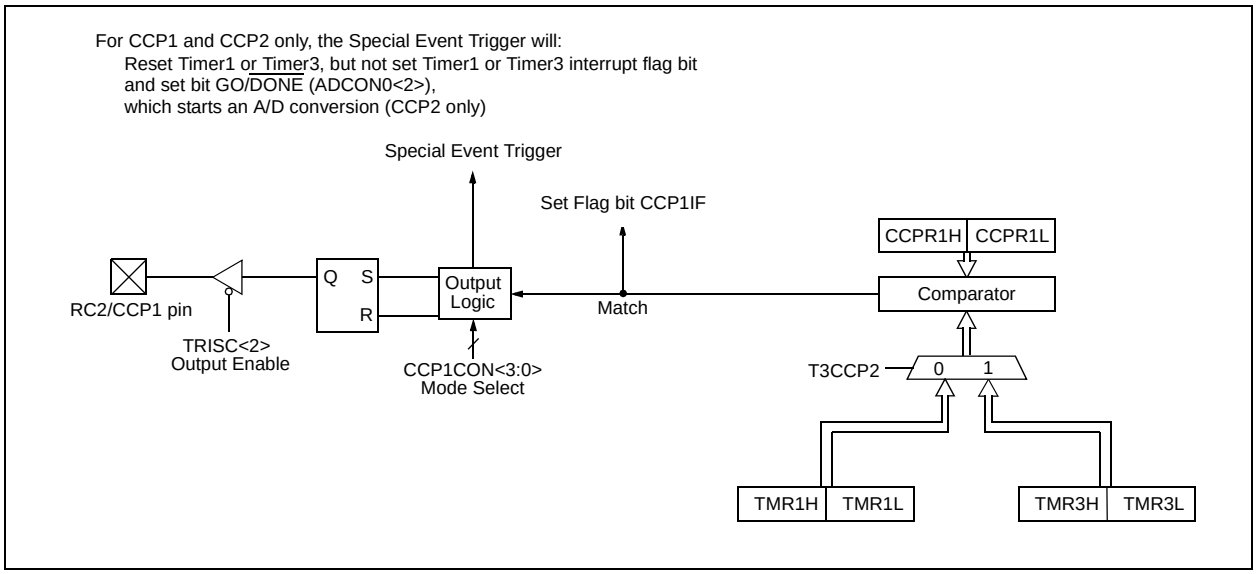
In this mode, an internal hardware trigger is generated, which may be used to initiate an action.

The special event trigger output of either CCP1 or CCP2, resets the TMR1 or TMR3 register pair, depending on which timer resource is currently selected. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1 or Timer3.

The CCP2 Special Event Trigger will also start an A/D conversion if the A/D module is enabled.

Note:	The special event trigger from the CCP2 module will not set the Timer1 or Timer3 interrupt flag bits.

FIGURE 16-3: COMPARE MODE OPERATION BLOCK DIAGRAM



© 2004 Microchip Technology Inc.

TABLE 16-2: 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	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 11qq	0--q qquu
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR2	—	CMIE	—	EEIE	BCLIF	LVDIF	TMR3IF	CCP2IF	-0-0 0000	---0 0000
PIE2	—	CMIF	—	EEIF	BCLIE	LVDIE	TMR3IE	CCP2IE	-0-0 0000	---0 0000
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	-1-1 1111	---1 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu
TMR3H	Timer3 Register High Byte								xxxx xxxx	uuuu uuuu
TMR3L	Timer3 Register Low Byte								xxxx xxxx	uuuu uuuu
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	uuuu uuuu
CCPRxL(1)	Capture/Compare/PWM Register x (LSB)								xxxx xxxx	uuuu uuuu
CCPRxH(1)	Capture/Compare/PWM Register x (MSB)								xxxx xxxx	uuuu uuuu
CCPxCON(1)	—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'.
Shaded cells are not used by Capture and Compare, Timer1 or Timer3.

Note 1: Generic term for all of the identical registers of this name for all CCP modules, where 'x' identifies the individual module (CCP1 through CCP5). Bit assignments and Reset values for all registers of the same generic name are identical.

16.4 PWM Mode

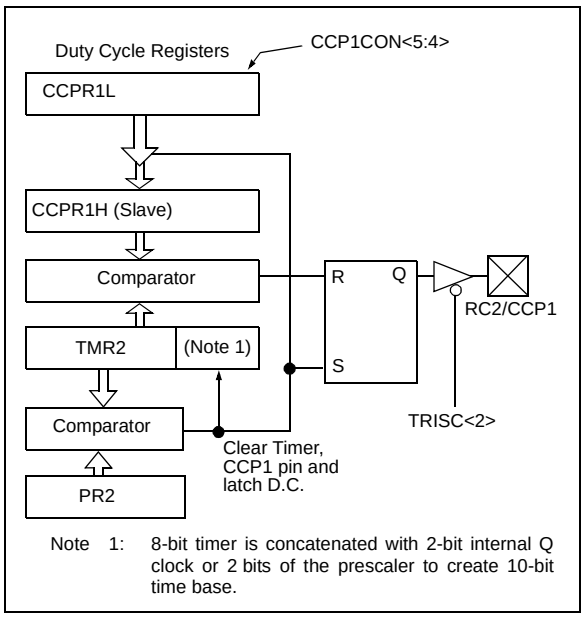
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 16-4 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see Section 16.4.3 “Setup for PWM Operation” .

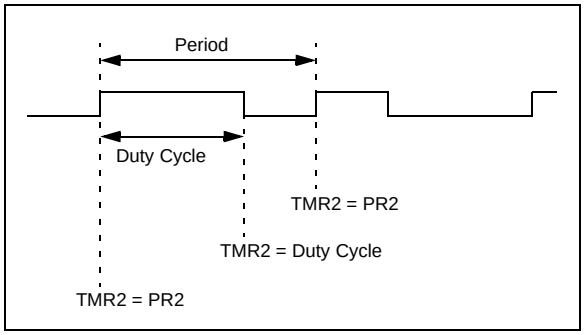
FIGURE 16-4: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time base.

A PWM output (Figure 16-5) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 16-5: PWM OUTPUT



16.4.1 PWM PERIOD

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

EQUATION 16-1:

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

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

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

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 and Timer4 postscalers (see Section 13.0 “Timer2 Module”) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

16.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the 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> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

EQUATION 16-2:

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

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after 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 doublebuffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

© 2004 Microchip Technology Inc.

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

EQUATION 16-3:

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

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

16.4.3 SETUP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

TABLE 16-3: 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)	14 10	12 10	10	8	7	6.58

TABLE 16-4: REGISTERS ASSOCIATED WITH PWM, TIMER2 AND TIMER4

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 11qq	0--q qquu
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR2	—	CMIE	—	EEIE	BCLIF	LVDIF	TMR3IF	CCP2IF	-0-0 0000	---0 0000
PIE2	—	CMIF	—	EEIF	BCLIE	LVDIE	TMR3IE	CCP2IE	-0-0 0000	---0 0000
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	-1-1 1111	---1 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
TMR2	Timer2 Module Register								0000 0000	0000 0000
PR2	Timer2 Module Period Register								1111 1111	1111 1111
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	uuuu uuuu
TMR4	Timer4 Register								0000 0000	uuuu uuuu
PR4	Timer4 Period Register								1111 1111	uuuu uuuu
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	uuuu uuuu
CCPRxL(1)	Capture/Compare/PWM Register x (LSB)								xxxx xxxx	uuuu uuuu
CCPRxH(1)	Capture/Compare/PWM Register x (MSB)								xxxx xxxx	uuuu uuuu
CCPxCON(1)	—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by PWM, Timer2, or Timer4.

Note 1: Generic term for all of the identical registers of this name for all CCP modules, where 'x' identifies the individual module (CCP1 through CCP5). Bit assignments and Reset values for all registers of the same generic name are identical.

NOTES:

© 2004 Microchip Technology Inc.

17.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

17.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I^2C)
  - Full Master mode
  - Slave mode (with general address call)

The I^2C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

17.2 Control Registers

The MSSP module has three associated registers. These include a status register (SSPSTAT) and two control registers (SSPCON1 and SSPCON2). The use of these registers and their individual configuration bits differ significantly, depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections.

17.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

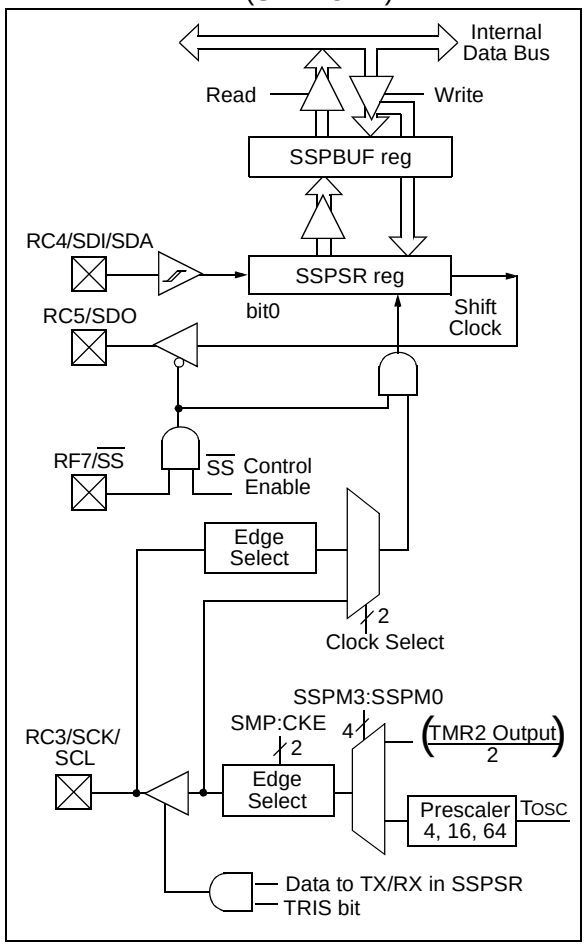
- Serial Data Out (SDO) – RC5/SDO
- Serial Data In (SDI) – RC4/SDI/SDA
- Serial Clock (SCK) – RC3/SCK/SCL

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) – RF7/$\overline{SS}$

Figure 17-1 shows the block diagram of the MSSP module when operating in SPI mode.

FIGURE 17-1: MSSP BLOCK DIAGRAM (SPI MODE)



17.3.1 REGISTERS

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

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

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

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

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

REGISTER 17-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 SMP: Sample bit
SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 CKE: SPI Clock Select bit
1 = Transmit occurs on transition from active to Idle clock state
0 = Transmit occurs on transition from Idle to active clock state
Note: Polarity of clock state is set by the CKP bit (SSPCON1<4>).

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/$\overline{Write}$ bit information
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, SSPBUF is full
0 = Receive not complete, SSPBUF is empty

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

© 2004 Microchip Technology Inc.

REGISTER 17-2: SSPCON1: MSSP CONTROL REGISTER1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 WCOL: Write Collision Detect bit (Transmit mode only)

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

bit 6 SSPOV: Receive Overflow Indicator bit

SPI Slave mode:

1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).

0 = No overflow

Note: In Master mode, the overflow bit is not set, since each new reception (and transmission) is initiated by writing to the SSPBUF register.

bit 5 SSPEN: Synchronous Serial Port Enable bit

1 = Enables serial port and configures SCK, SDO, SDI and $\overline{SS}$ as serial port pins

0 = Disables serial port and configures these pins as I/O port pins

Note: When enabled, these pins must be properly configured as input or output.

bit 4 CKP: Clock Polarity Select bit

1 = Idle state for clock is a high level

0 = Idle state for clock is a low level

bit 3-0 SSPM3:SSPM0: Synchronous Serial Port Mode Select bits

0101 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control disabled, $\overline{SS}$ can be used as I/O pin

0100 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control enabled

0011 = SPI Master mode, clock = TMR2 output/2

0010 = SPI Master mode, clock = FOSC/64

0001 = SPI Master mode, clock = FOSC/16

0000 = SPI Master mode, clock = FOSC/4

Note: Bit combinations not specifically listed here are either reserved, or implemented in I^2C mode only.

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

17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data input sample phase (middle or end of data output time)
- Clock edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

The MSSP consists of a Transmit/Receive Shift Register (SSPSR) and a Buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then, the Buffer Full detect bit, BF (SSPSTAT<0>) and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored and the Write Collision detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

EQUATION 17-1: LOADING THE SSPBUF (SSPSR) REGISTER

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


© 2004 Microchip Technology Inc.

17.3.3 ENABLING SPI I/O

To enable the serial port, SSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPCON registers and then set the SSPEN bit. This configures the SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- SDI is automatically controlled by the SPI module
- SDO must have TRISC<5> bit cleared
- SCK (Master mode) must have TRISC<3> bit cleared
- SCK (Slave mode) must have TRISC<3> bit set
- $\overline{SS}$ must have TRISF<7> bit set

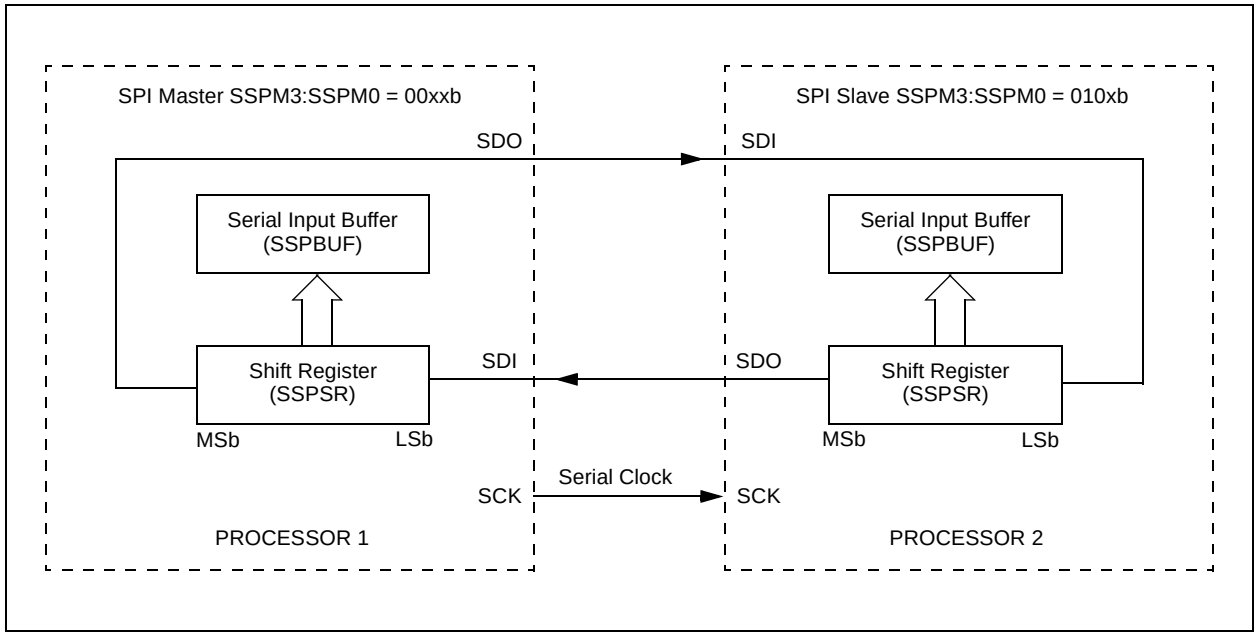
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

17.3.4 TYPICAL CONNECTION

Figure 17-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

FIGURE 17-2: SPI MASTER/SLAVE CONNECTION



17.3.5 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 17-2) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

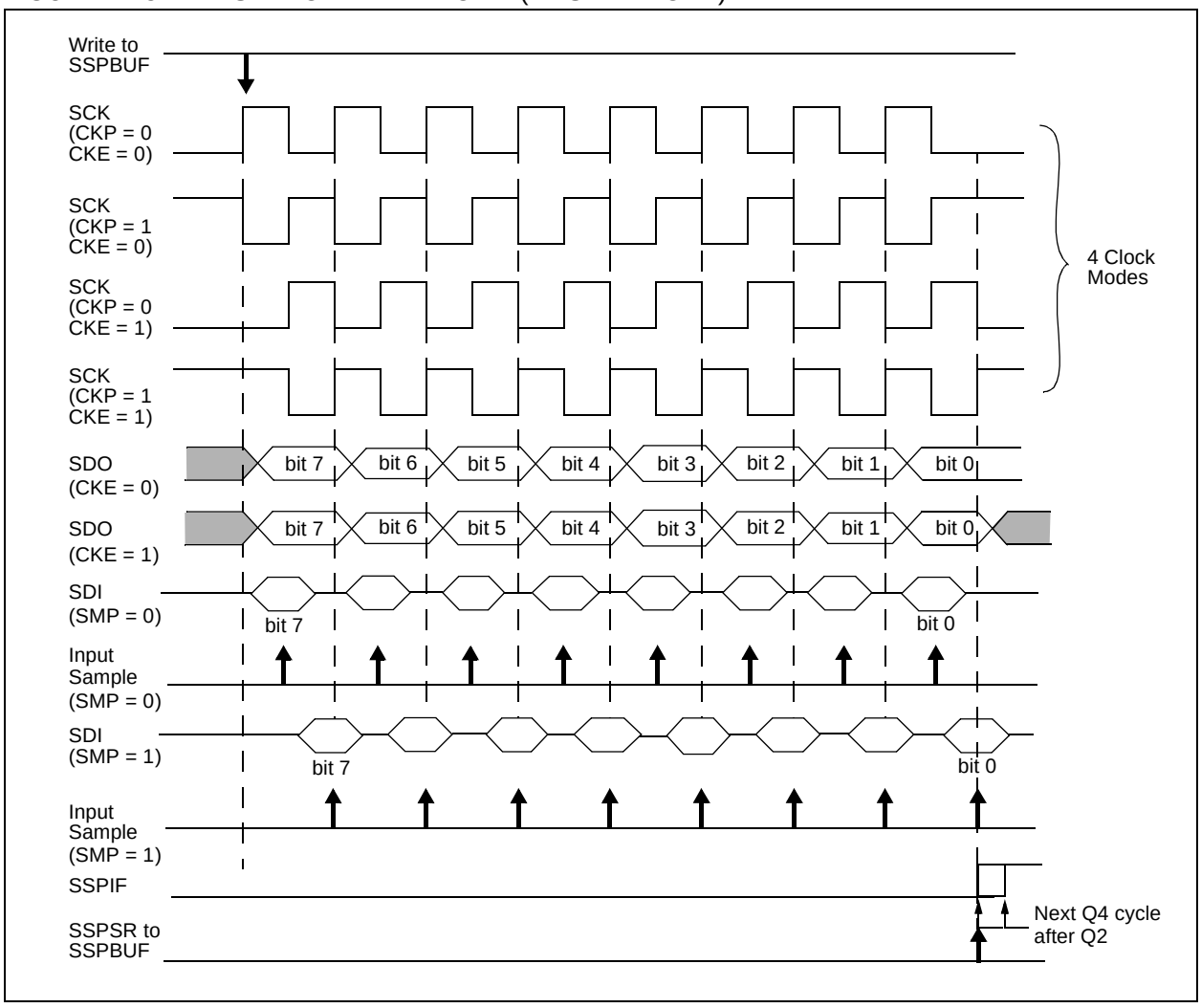
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication, as shown in Figure 17-3, Figure 17-5 and Figure 17-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

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

Figure 17-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

FIGURE 17-3: SPI MODE WAVEFORM (MASTER MODE)



© 2004 Microchip Technology Inc.

17.3.6 SLAVE MODE

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

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

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

17.3.7 SLAVE SELECT SYNCHRONIZATION

The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 04h). The pin must not be driven low for the $\overline{SS}$ pin to function as an input. The Data Latch must be high. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable, depending on the application.

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

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

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

FIGURE 17-4: SLAVE SYNCHRONIZATION WAVEFORM

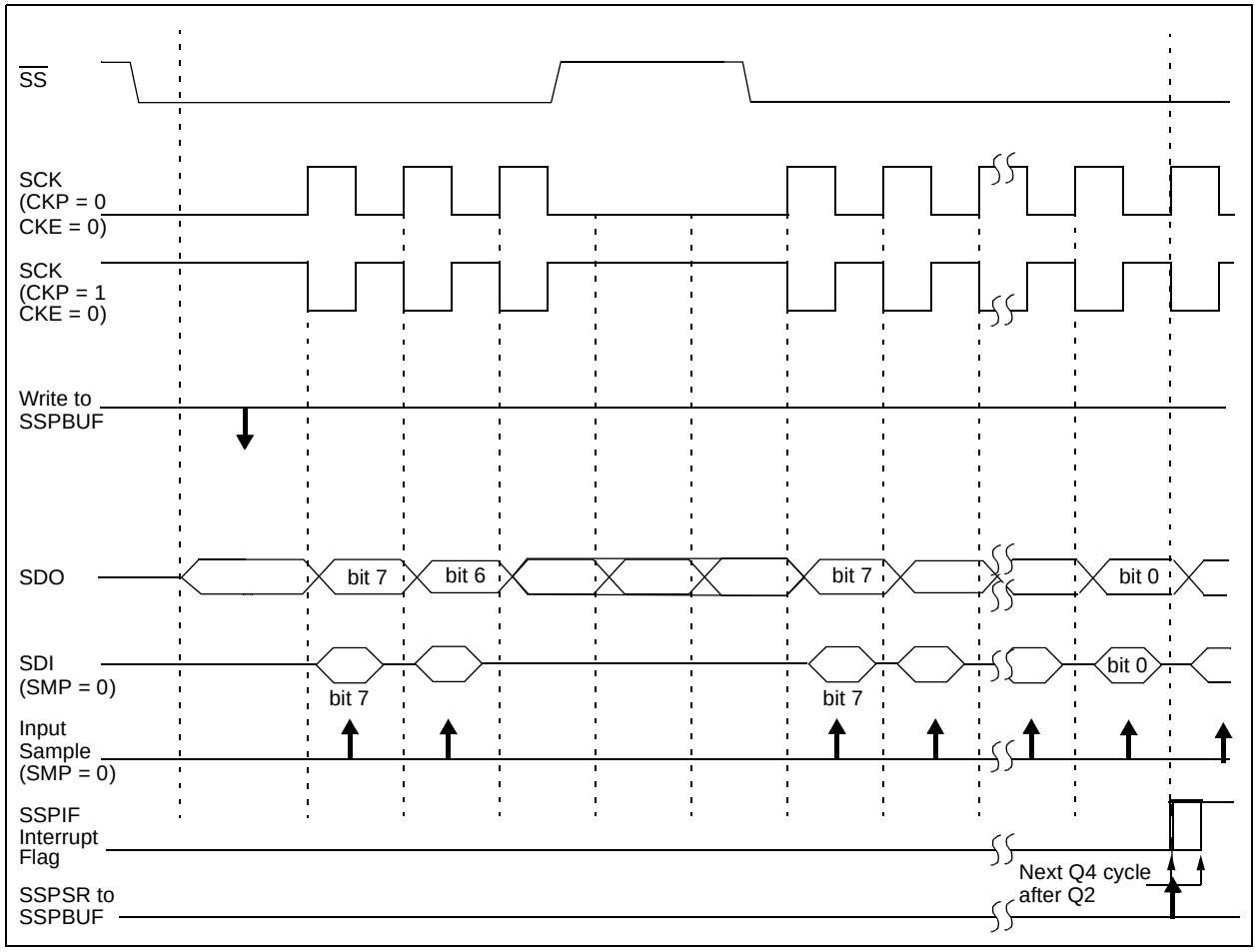


FIGURE 17-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

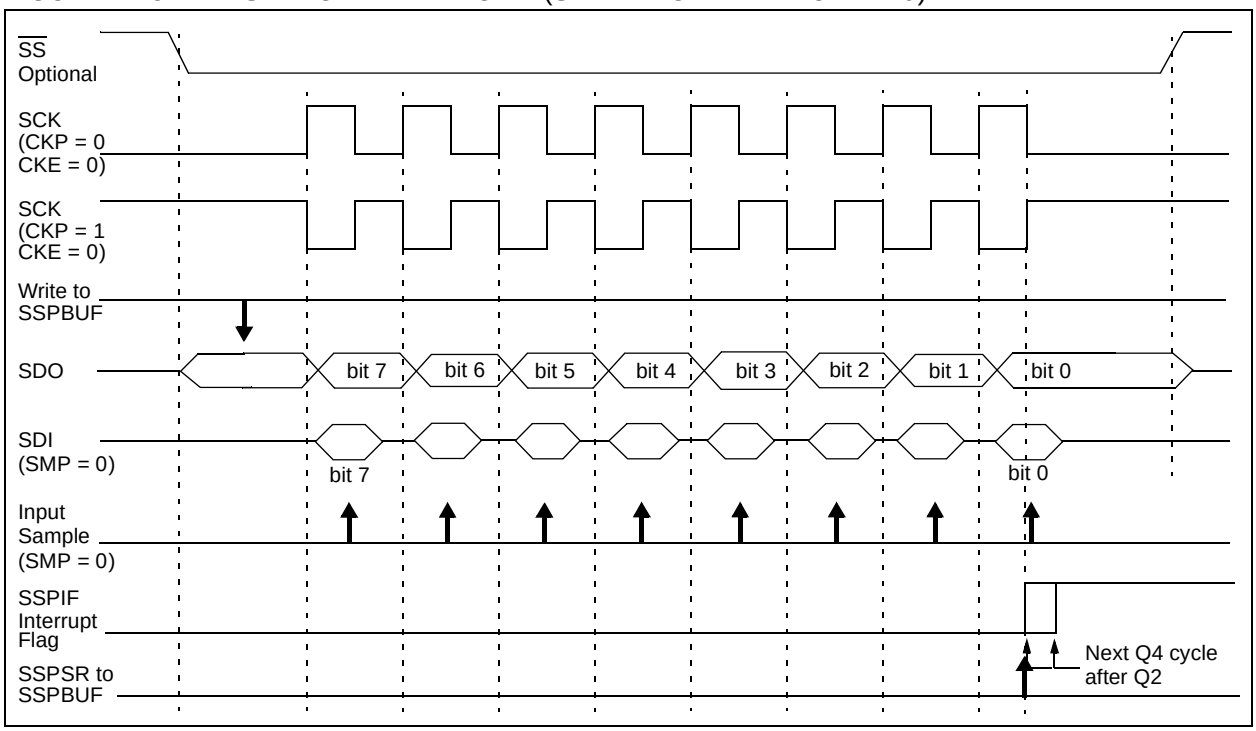
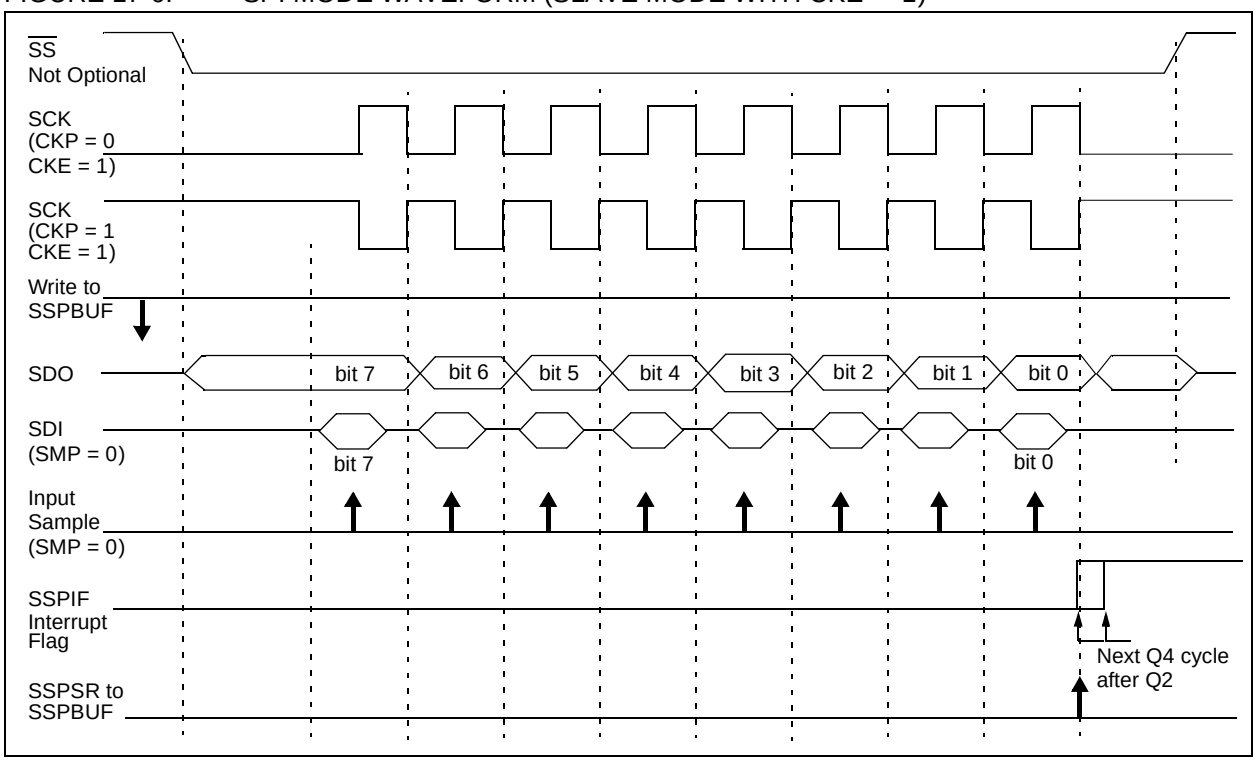


FIGURE 17-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



© 2004 Microchip Technology Inc.

17.3.8 SLEEP OPERATION

In Master mode, all module clocks are halted and the transmission/reception will remain in that state until the device wakes from Sleep. After the device returns to normal mode, the module will continue to transmit/ receive data.

In Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in Sleep 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 from Sleep.

17.3.9 EFFECTS OF A RESET

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

17.3.10 BUS MODE COMPATIBILITY

Table 17-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 17-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also an SMP bit, which controls when the data is sampled.

TABLE 17-2: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	1111 1111	uuuu uuuu
SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

17.4 I^2C Mode

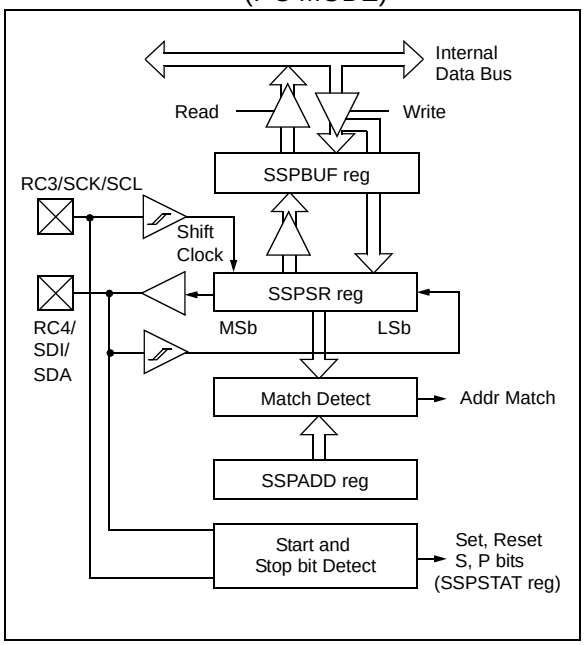
The MSSP module in I^2C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial clock (SCL) – RC3/SCK/SCL
- Serial data (SDA) – RC4/SDI/SDA

The user must configure these pins as inputs or outputs through the TRISC<4:3> bits.

FIGURE 17-7: MSSP BLOCK DIAGRAM (I^2C MODE)



17.4.1 REGISTERS

The MSSP module has six registers for I^2C operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON, SSPCON2 and SSPSTAT are the control and status registers in I^2C mode operation. The SSPCON and SSPCON2 registers are readable and writable. The lower six bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

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

SSPADD register holds the slave device address when the SSP is configured in I^2C Slave mode. When the SSP is configured in Master mode, the lower seven bits of SSPADD act as the Baud Rate Generator reload value.

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

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

© 2004 Microchip Technology Inc.

REGISTER 17-3: SSPSTAT: MSSP STATUS REGISTER (I ^{2}C 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	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 SMP: Slew Rate Control bit

In Master or Slave mode:

1 = Slew rate control disabled for standard speed mode (100 kHz and 1 MHz)

0 = Slew rate control enabled for high-speed mode (400 kHz)

bit 6 CKE: SMBus Select bit

In Master or Slave mode:

1 = Enable SMBus specific inputs

0 = Disable SMBus specific inputs

bit 5 D/$\overline{A}$: 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 = Indicates that a Stop bit has been detected last

0 = Stop bit was not detected last

Note: This bit is cleared on Reset and when SSPEN is cleared.

bit 3 S: Start bit

1 = Indicates that a Start bit has been detected last

0 = Start bit was not detected last

Note: This bit is cleared on Reset and when SSPEN is cleared.

bit 2 R/$\overline{W}$: Read/$\overline{Write}$ bit Information (I^{2}C mode only)

In Slave mode:

1 = Read

0 = Write

Note: 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.

In Master mode:

1 = Transmit is in progress

0 = Transmit is not in progress

Note: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in active mode.

bit 1 UA: Update Address bit (10-bit Slave mode only)

1 = Indicates that the user needs to update the address in the SSPADD register

0 = Address does not need to be updated

bit 0 BF: Buffer Full Status bit

In Transmit mode:

1 = SSPBUF is full

0 = SSPBUF is empty

In Receive mode:

1 = SSPBUF is full (does not include the $\overline{ACK}$ and Stop bits)

0 = SSPBUF is empty (does not include the $\overline{ACK}$ and Stop bits)

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

REGISTER 17-4: SSPCON1: MSSP CONTROL REGISTER 1 (I^2C MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 WCOL: Write Collision Detect bit

In Master Transmit mode:

1 = A write to the SSPBUF register was attempted while the I^2C conditions were not valid for a transmission to be started (must be cleared in software)

0 = No collision

In Slave Transmit mode:

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

In Receive mode (Master or Slave modes):

This is a “don’t care” bit.

bit 6 SSPOV: Receive Overflow Indicator bit

In Receive mode:

1 = A byte is received while the SSPBUF register is still holding the previous byte (must be cleared in software)

0 = No overflow

In Transmit mode:

This is a “don’t care” bit in Transmit mode.

bit 5 SSPEN: Synchronous Serial Port Enable bit

1 = Enables the serial port and configures the SDA and SCL pins as the serial port pins

0 = Disables serial port and configures these pins as I/O port pins

Note: When enabled, the SDA and SCL pins must be properly configured as input or output.

bit 4 CKP: SCK Release Control bit

In Slave mode:

1 = Release clock

0 = Holds clock low (clock stretch), used to ensure data setup time

In Master mode:

Unused in this mode.

bit 3-0 SSPM3:SSPM0: Synchronous Serial Port Mode Select bits

1111 = I^2C Slave mode, 10-bit address with Start and Stop bit interrupts enabled

1110 = I^2C Slave mode, 7-bit address with Start and Stop bit interrupts enabled

1011 = I^2C Firmware Controlled Master mode (Slave Idle)

1000 = I^2C Master mode, clock = FOSC/(4 * (SSPADD + 1))

0111 = I^2C Slave mode, 10-bit address

0110 = I^2C Slave mode, 7-bit address

Note: Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

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

© 2004 Microchip Technology Inc.

REGISTER 17-5: SSPCON2: MSSP CONTROL REGISTER 2 (I^2C MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 7 GCEN: General Call Enable bit (Slave mode only)
1 = Enable interrupt when a general call address (0000h) is received in the SSPSR
0 = General call address disabled

bit 6 ACKSTAT: Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave

bit 5 ACKDT: Acknowledge Data bit (Master Receive mode only)
1 = Not Acknowledge
0 = Acknowledge

Note: Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.

bit 4 ACKEN: Acknowledge Sequence Enable bit (Master Receive mode only)
1 = Initiate Acknowledge sequence on SDA and SCL pins and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence Idle

bit 3 RCEN: Receive Enable bit (Master mode only)
1 = Enables Receive mode for I^2C
0 = Receive Idle

bit 2 PEN: Stop Condition Enable bit (Master mode only)
1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 RSEN: Repeated Start Condition Enabled bit (Master mode only)
1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 SEN: Start Condition Enabled/Stretch Enabled bit
In Master mode:
1 = Initiate Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Start condition Idle
In Slave mode:
1 = Clock stretching is enabled for both Slave Transmit and Slave Receive (stretch enabled)
0 = Clock stretching is disabled

Note: For bits ACKEN, RCEN, PEN, RSEN, SEN: If the I^2C module is not in the Idle mode, this bit may not be set (no spooling) and the SSPBUF may not be written (or writes to the SSPBUF are disabled).

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

17.4.2 OPERATION

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

The SSPCON1 register allows control of the I^2C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I^2C modes to be selected:

- I^2C Master mode, clock = (FOSC/4) x (SSPADD + 1)
- I^2C Slave mode (7-bit address)
- I^2C Slave mode (10-bit address)
- I^2C Slave mode (7-bit address), with Start and Stop bit interrupts enabled
- I^2C Slave mode (10-bit address), with Start and Stop bit interrupts enabled
- I^2C Firmware Controlled Master mode, slave is Idle

Selection of any I^2C mode, with the SSPEN bit set, forces the SCL and SDA pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISC bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCL and SDA pins.

17.4.3 SLAVE MODE

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

The I^2C Slave mode hardware will always generate an interrupt on an address match. Through the mode select bits, the user can also choose to interrupt on Start and Stop bits

When an address is matched or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) pulse and load the SSPBUF register with the received value currently in the SSPSR register.

Any combination of the following conditions will cause the MSSP module not to give this $\overline{ACK}$ pulse:

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

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit SSPIF (PIR1<3>) is set. The BF bit is cleared by reading the SSPBUF register, while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification, as well as the requirement of the MSSP module, are shown in timing parameter #100 and parameter #101.

17.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPSR register value is loaded into the SSPBUF register.
2. The Buffer Full bit BF is set.
3. An $\overline{ACK}$ pulse is generated.
4. MSSP Interrupt Flag bit, SSPIF (PIR1<3>), is set (interrupt is generated, if enabled) on the falling edge of the ninth SCL pulse.

In 10-bit Address mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{W}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '11110 A9 A8 0 ', where 'A9' and 'A8' are the two MSbs of the address. The sequence of events for 10-bit address is as follows, with steps 7 through 9 for the slave-transmitter:

1. Receive first (high) byte of address (bits SSPIF, BF and bit UA (SSPSTAT<1>) are set).
2. Update the SSPADD register with second (low) byte of address (clears bit UA and releases the SCL line).
3. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
4. Receive second (low) byte of address (bits SSPIF, BF and UA are set).
5. Update the SSPADD register with the first (high) byte of address. If match releases SCL line, this will clear bit UA.
6. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits SSPIF and BF are set).
9. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.

© 2004 Microchip Technology Inc.

17.4.3.2 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register and the SDA line is held low ($\overline{ACK}$).

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON1<6>) is set.

An MSSP interrupt is generated for each data transfer byte. Flag bit SSPIF (PIR1<3>) must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

If SEN is enabled (SSPCON1<0> = 1), RC3/SCK/SCL will be held low (clock stretch) following each data transfer. The clock must be released by setting bit CKP (SSPCON<4>). See Section 17.4.4 “Clock Stretching” for more detail.

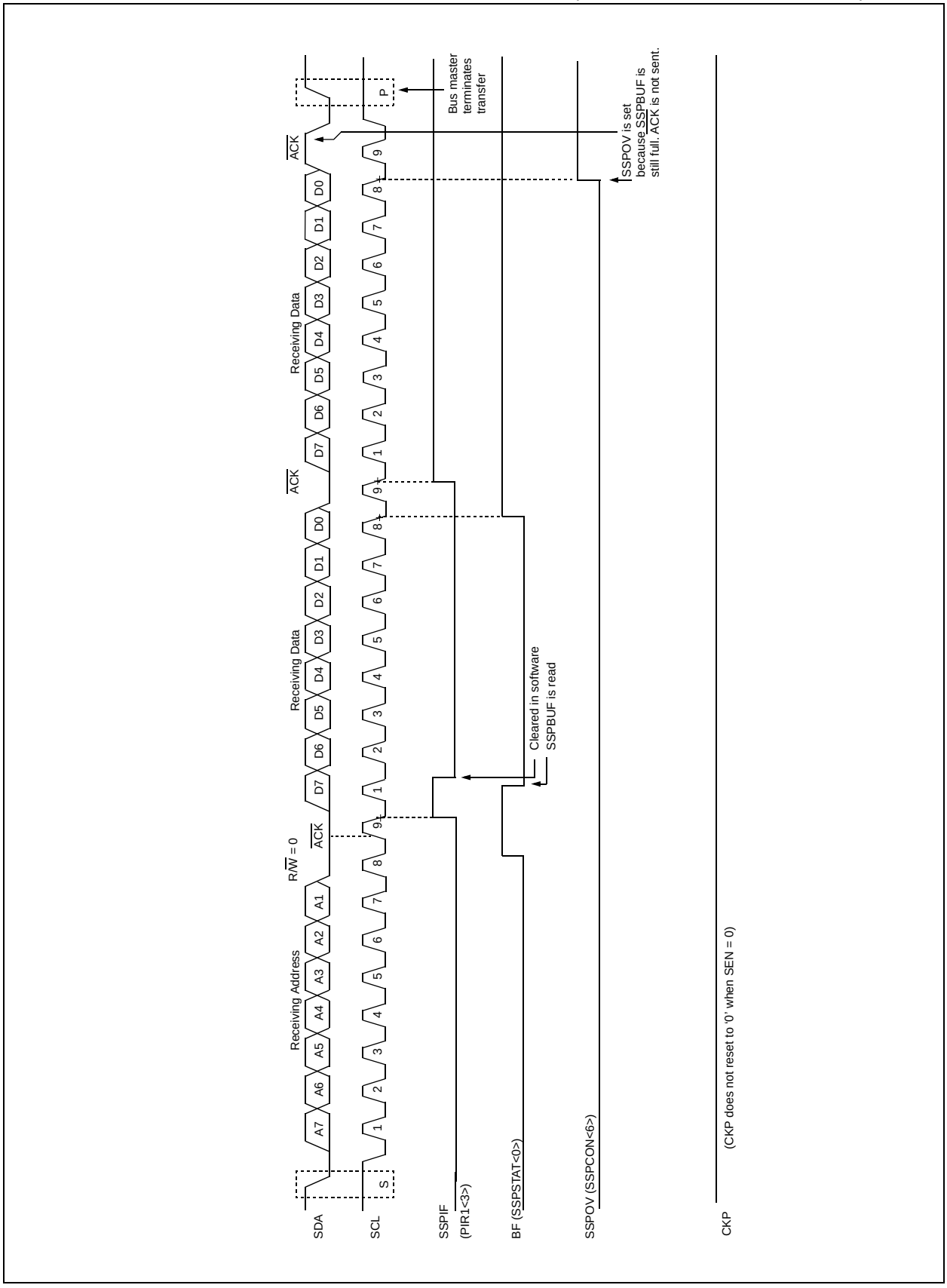
17.4.3.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RC3/SCK/SCL is held low, regardless of SEN (see Section 17.4.4 “Clock Stretching” , for more detail). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then pin RC3/SCK/SCL should be enabled by setting bit CKP (SSPCON1<4>). The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 17-9).

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not $\overline{ACK}$), then the data transfer is complete. In this case, when the $\overline{ACK}$ is latched by the slave, the slave logic is reset (resets SSPSTAT register) and the slave monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPBUF register. Again, pin RC3/SCK/SCL must be enabled by setting bit CKP.

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

FIGURE 17-8: I^2C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)



© 2004 Microchip Technology Inc.

FIGURE 17-9: I^2C SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

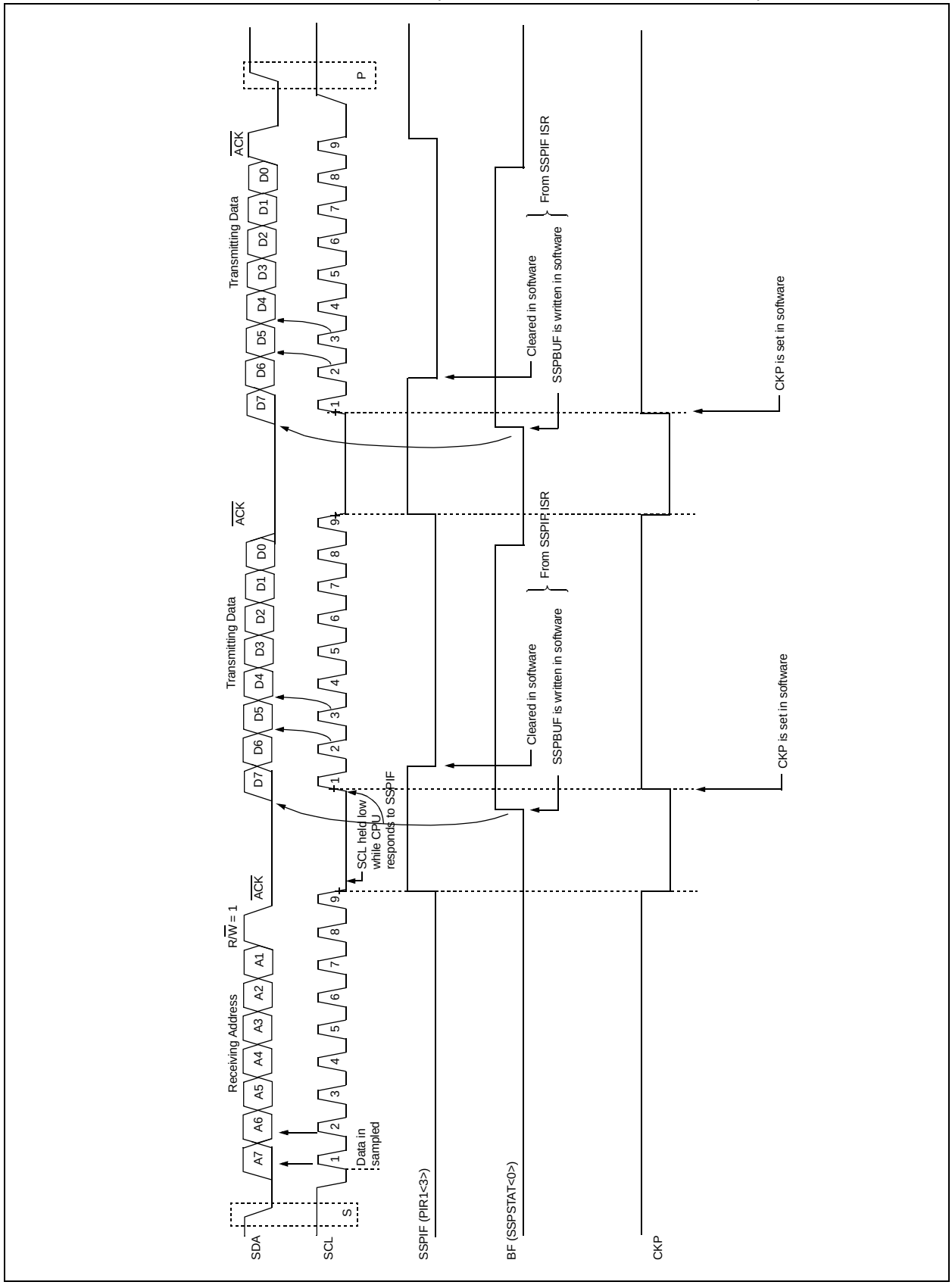
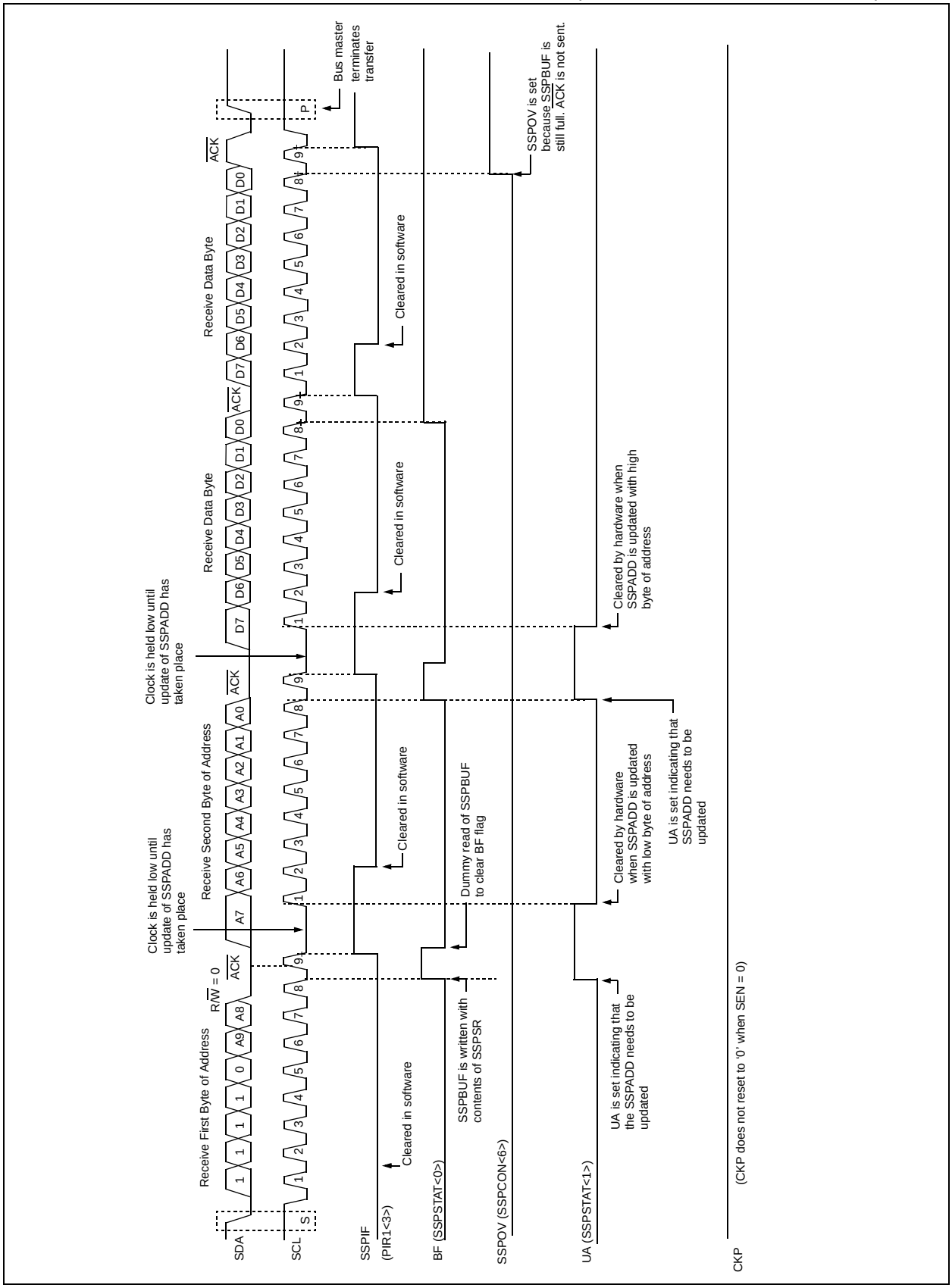
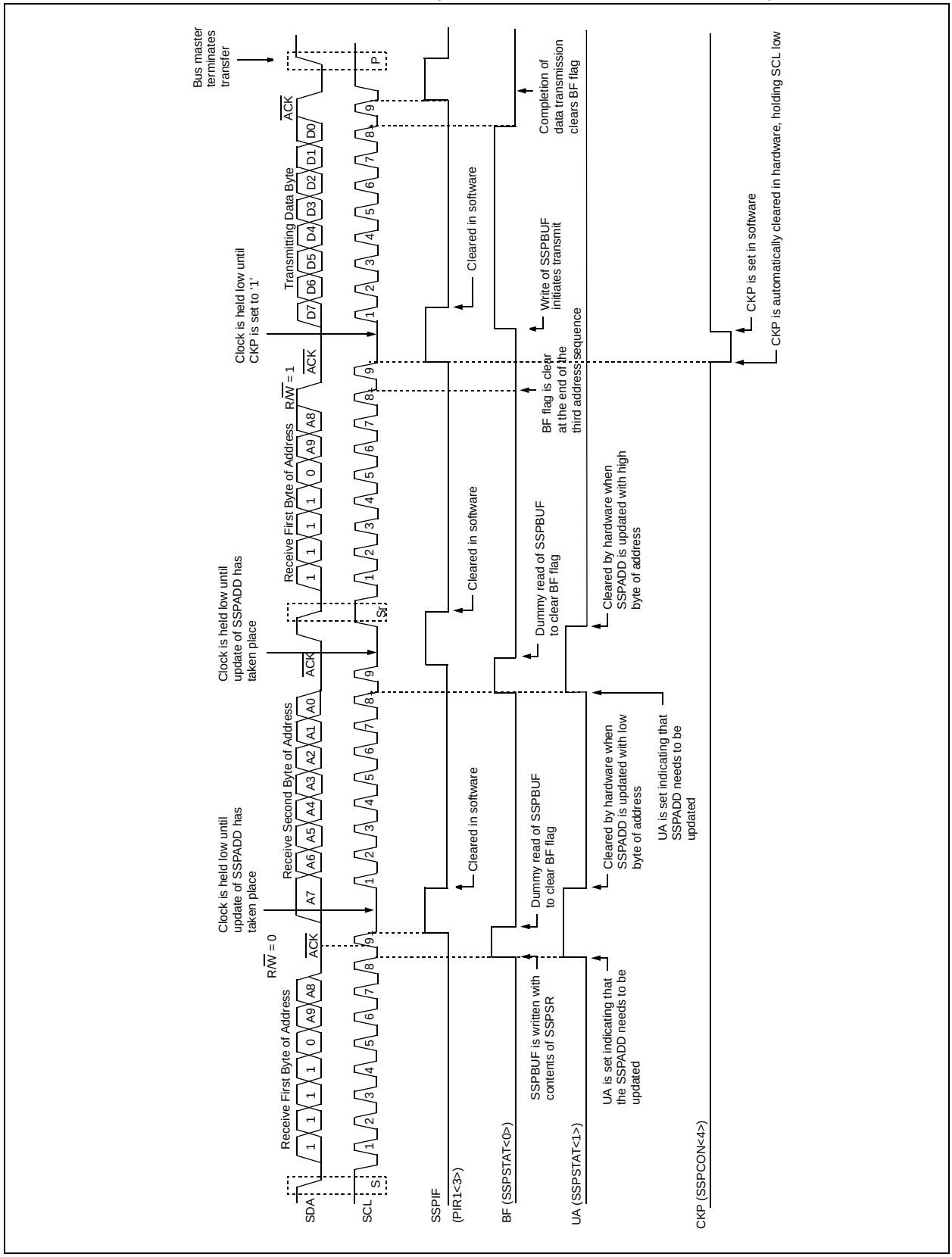


FIGURE 17-10: I^2C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)



© 2004 Microchip Technology Inc.

FIGURE 17-11: I^2C SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



17.4.4 CLOCK STRETCHING

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

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

17.4.4.1 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 $\overline{ACK}$ sequence, if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-13).

Note 1:	If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.
2:	The CKP bit can be set in software, regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence, in order to prevent an overflow condition.

17.4.4.2 Clock Stretching for 10-bit Slave Receive Mode (SEN = 1)

In 10-bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but 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 SSPADD. Clock stretching will occur on each data receive sequence, as described in 7-bit mode.

Note:	If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

17.4.4.3 Clock Stretching for 7-bit Slave Transmit Mode

7-bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock, if the BF bit is clear. This occurs, regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-9).

Note 1:	If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.
2:	The CKP bit can be set in software, regardless of the state of the BF bit.

17.4.4.4 Clock Stretching for 10-bit Slave Transmit Mode

In 10-bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-bit Slave Receive mode. The first two addresses are followed by a third address sequence, which contains the high-order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled as in 7-bit Slave Transmit mode (see Figure 17-11).

© 2004 Microchip Technology Inc.

17.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCL output is forced to '0'. However, clearing the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external I^2C master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the I^2C bus have deasserted SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 17-12).

FIGURE 17-12: CLOCK SYNCHRONIZATION TIMING

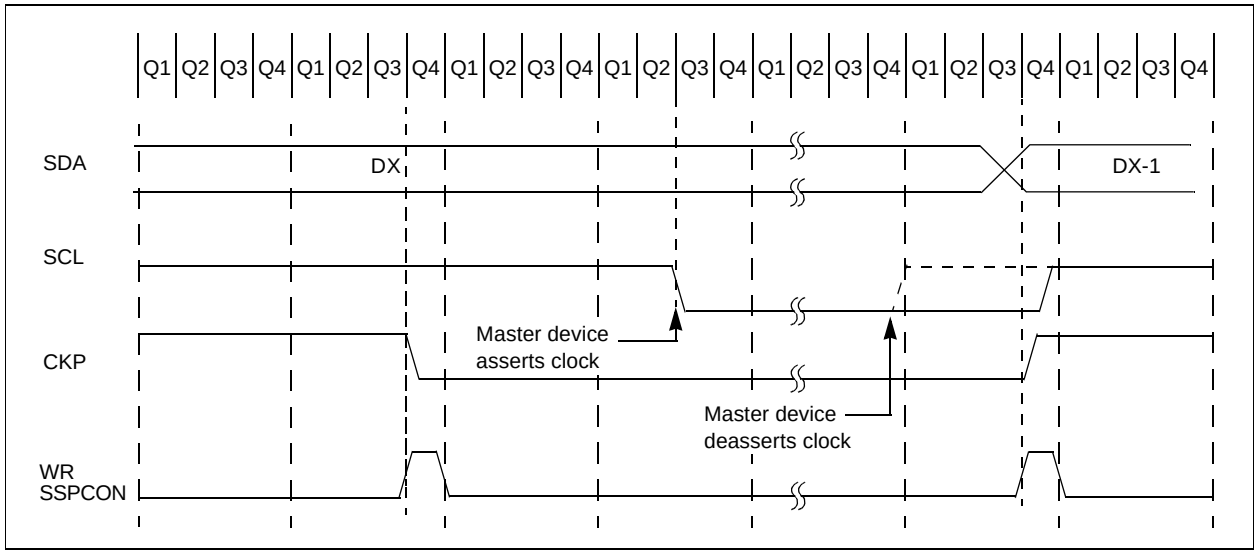
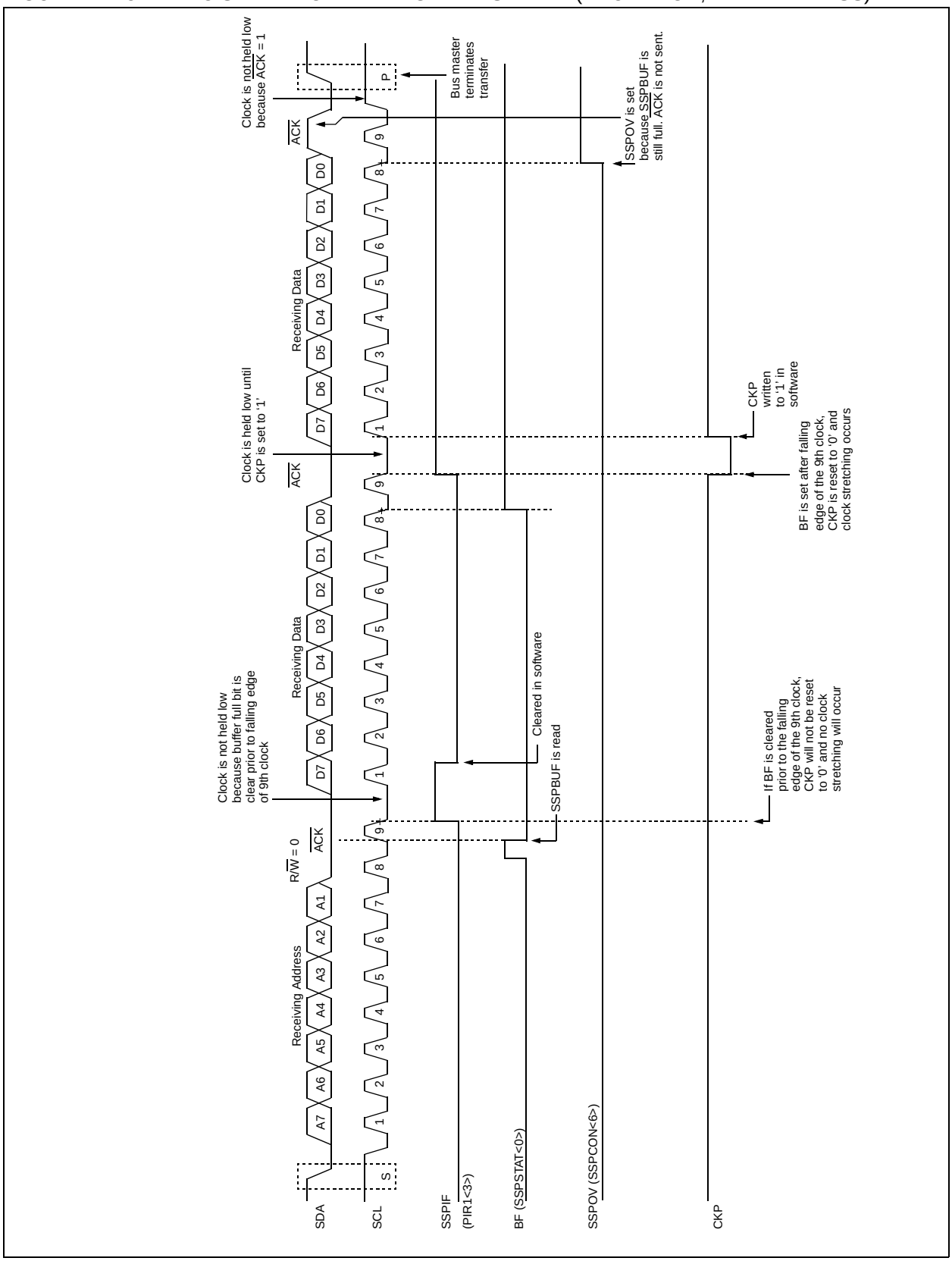
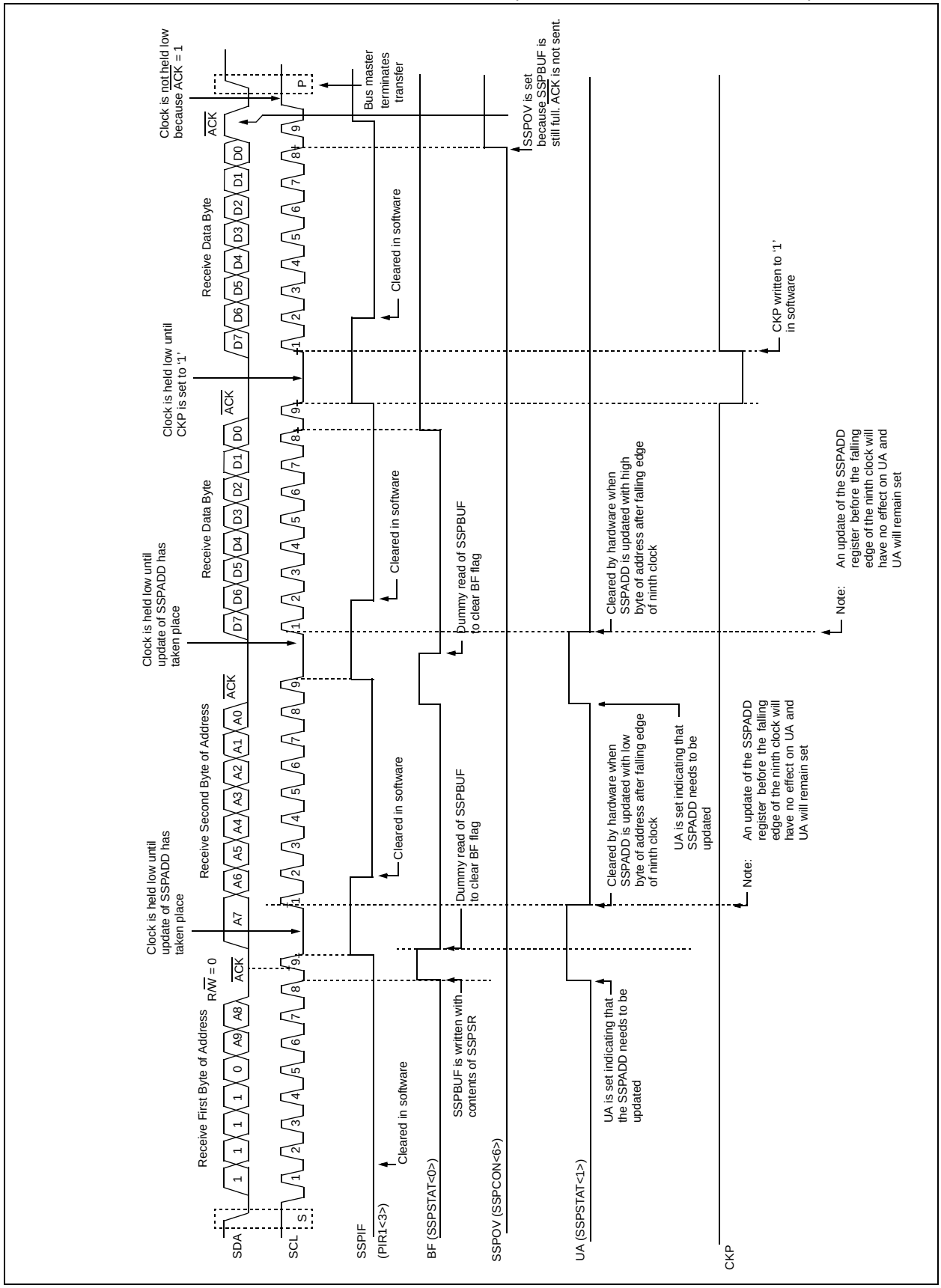


FIGURE 17-13: I^2C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)



© 2004 Microchip Technology Inc.

FIGURE 17-14: I^2C SLAVE MODE TIMING SEN = 1 (RECEPTION, 10-BIT ADDRESS)



17.4.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I^2C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address, which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I^2C protocol. It consists of all '0's with R/$\overline{W}$ = 0.

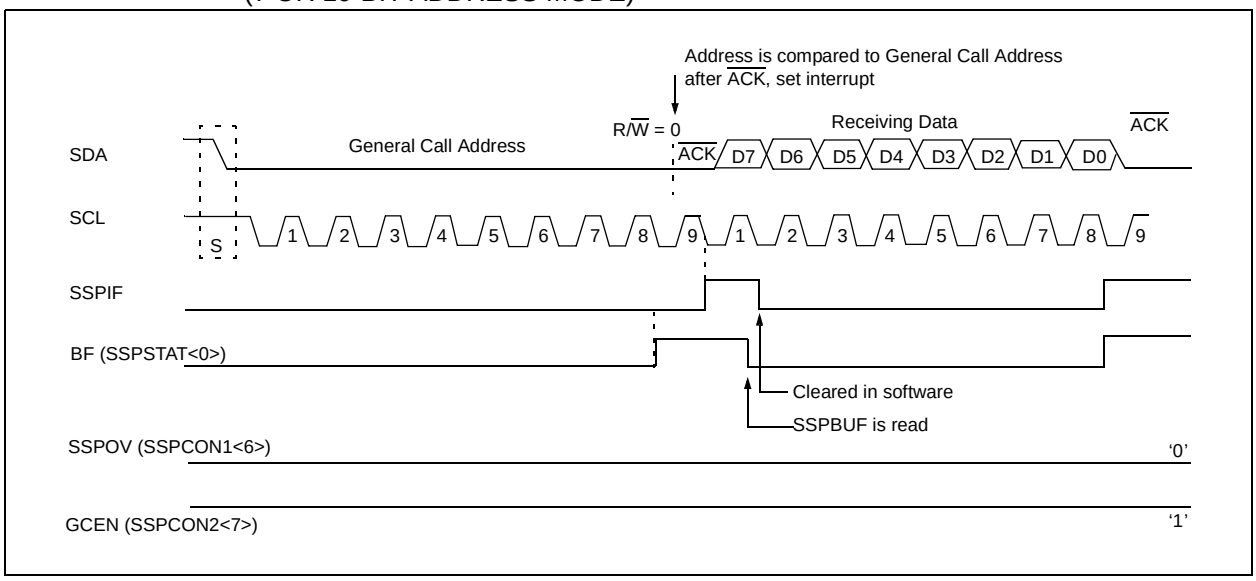
The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPBUF. The value can be used to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPADD is required to be updated for the second half of the address to match and the UA bit is set (SSPSTAT<1>). If the general call address is sampled when the GCEN bit is set, while the slave is configured in 10-bit Address mode, then the second half of the address is not necessary, the UA bit will not be set and the slave will begin receiving data after the Acknowledge (Figure 17-15).

FIGURE 17-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESS MODE)



© 2004 Microchip Technology Inc.

17.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset, or when the MSSP module is disabled. Control of the I^2C bus may be taken when the P bit is set or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I^2C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

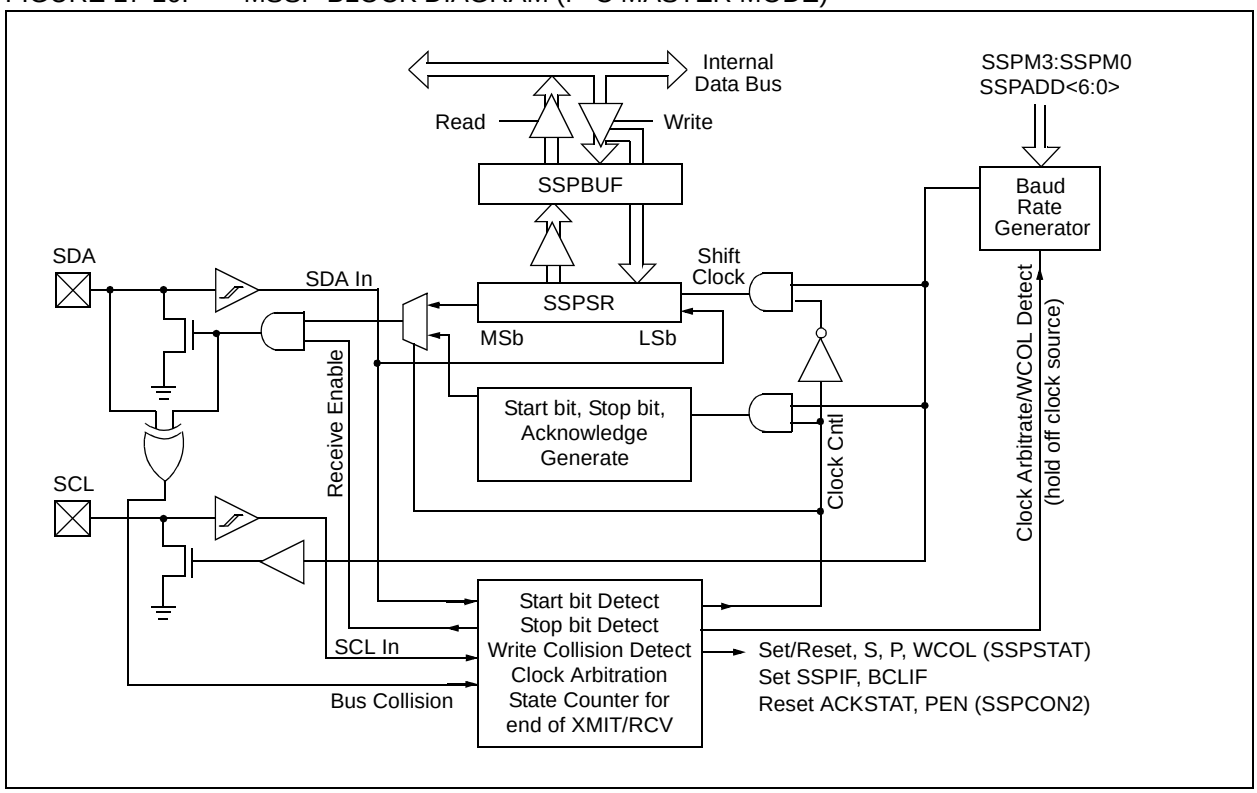
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

Note:	The MSSP module, when configured in I^2C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

The following events will cause SSP Interrupt Flag bit, SSPIF, to be set (SSP interrupt if enabled):

- Start Condition
- Stop Condition
- Data Transfer Byte Transmitted/received
- Acknowledge Transmit
- Repeated Start

FIGURE 17-16: MSSP BLOCK DIAGRAM (I^2C MASTER MODE)



17.4.6.1 I^2C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I^2C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator used for the SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz or 1 MHz I^2C operation. See Section 17.4.7 "Baud Rate Generator" , for more information.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. Address is shifted out the SDA pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with eight bits of data.
8. Data is shifted out the SDA pin until all 8 bits are transmitted.
9. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
11. The user generates a Stop condition by setting the Stop enable bit PEN (SSPCON2<2>).
12. Interrupt is generated once the Stop condition is complete.

© 2004 Microchip Technology Inc.

17.4.7 BAUD RATE GENERATOR

In I^2C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPADD register (Figure 17-17). When a write occurs to SSPBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to '0' and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (TCY) on the Q2 and Q4 clocks. In I^2C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by $\overline{ACK}$), the internal clock will automatically stop counting and the SCL pin will remain in its last state.

Table 15-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD.

FIGURE 17-17: BAUD RATE GENERATOR BLOCK DIAGRAM

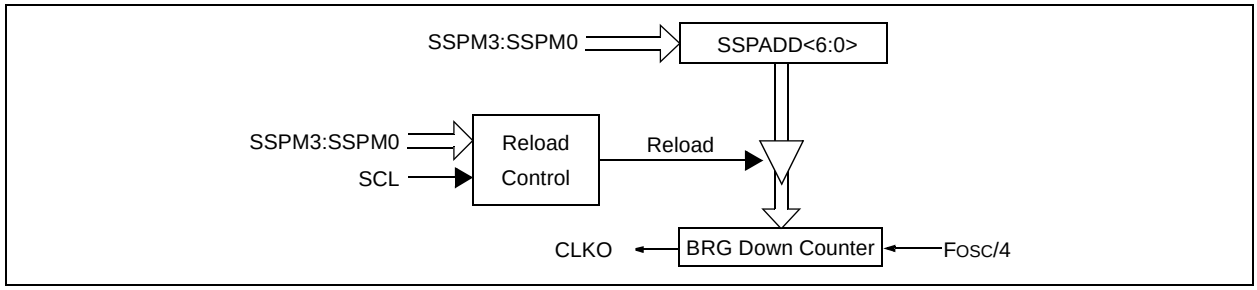


TABLE 17-3: I^2C CLOCK RATE W/BRG

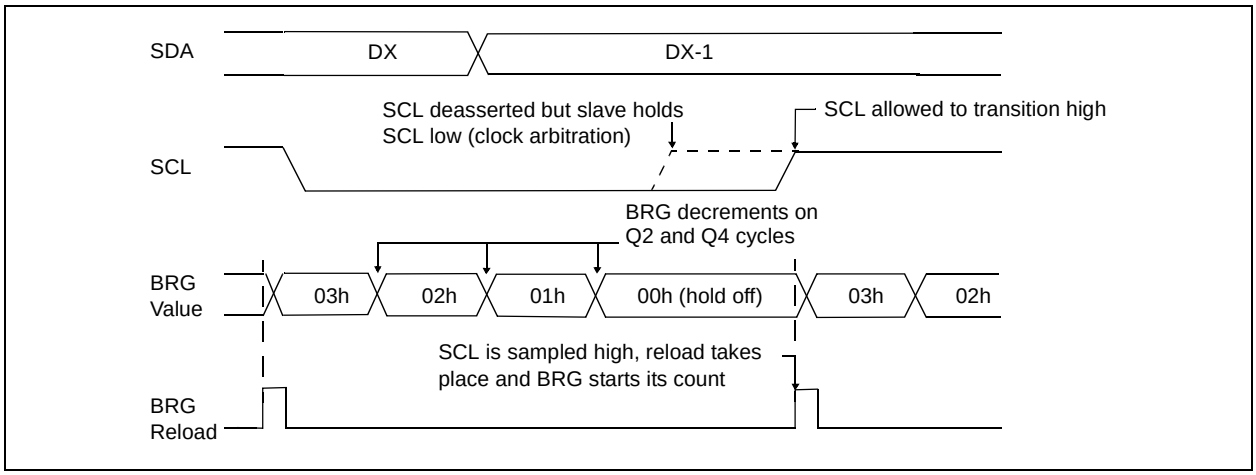
FCY	FCY*2	BRG VALUE	FSCL (2 rollovers of BRG)
10 MHz	20 MHz	19h	400 kHz(1)
10 MHz	20 MHz	20h	312.5 kHz
10 MHz	20 MHz	3Fh	100 kHz
4 MHz	8 MHz	0Ah	400 kHz(1)
4 MHz	8 MHz	0Dh	308 kHz
4 MHz	8 MHz	28h	100 kHz
1 MHz	2 MHz	03h	333 kHz(1)
1 MHz	2 MHz	0Ah	100 kHz
1 MHz	2 MHz	00h	1 MHz(1)

Note 1: The I^2C interface does not conform to the 400 kHz I^2C specification (which applies to rates greater than 100 kHz) in all details, but may be used with care where higher rates are required by the application.

17.4.7.1 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count, in the event that the clock is held low by an external device (Figure 15-18).

FIGURE 17-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



© 2004 Microchip Technology Inc.

17.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Condition Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low, while SCL is high, is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware, the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

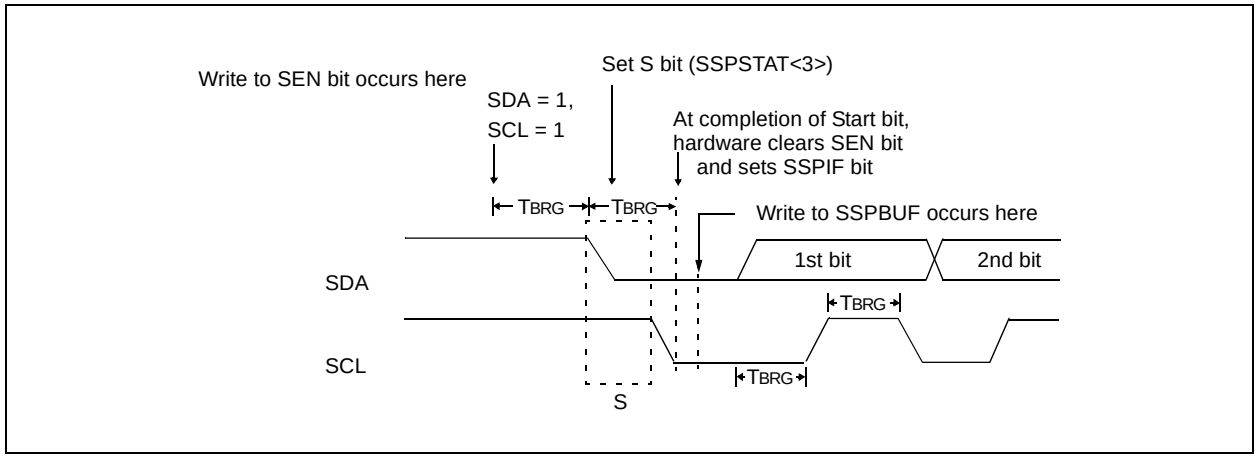
Note:	If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

17.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.

FIGURE 17-19: FIRST START BIT TIMING



17.4.9 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPADD<5:0> and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = 0) for one TBRG, while SCL is high. Following this, the RSEN bit (SSPCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the Baud Rate Generator has timed out.

Note 1: If RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDA is sampled low when SCL goes from low-to-high.
- SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data '1'.

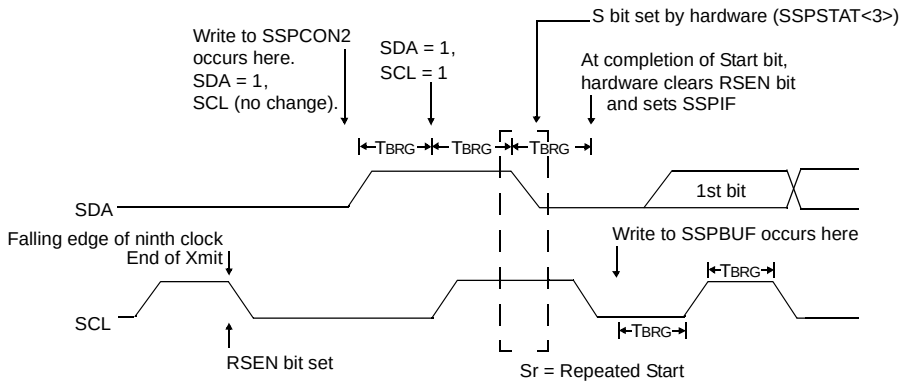
Immediately following the setting of the SSPIF bit, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

17.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write does not occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

FIGURE 17-20: REPEAT START CONDITION WAVEFORM



© 2004 Microchip Technology Inc.

17.4.10 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address, or the other half of a 10-bit address is accomplished by simply writing a value to the SSPBUF register. This action will set the Buffer Full flag bit, BF and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted (see data hold time specification parameter #106). SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high (see data setup time specification parameter #107). When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time, after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. This allows the slave device being addressed to respond with an $\overline{ACK}$ bit during the ninth bit time if an address match occurred, or if data was received properly. The status of $\overline{ACK}$ is written into the ACKDT bit on the falling edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared. If not, the bit is set. After the ninth clock, the SSPIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPBUF, leaving SCL low and SDA unchanged (Figure 17-21).

After the write to the SSPBUF, each address bit will be shifted out on the falling edge of SCL, until all seven address bits and the R/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the master will deassert the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the $\overline{ACK}$ bit is loaded into the ACKSTAT status bit (SSPCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

17.4.10.1 BF Status Flag

In Transmit mode, the BF bit (SSPSTAT<0>) is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

17.4.10.2 WCOL Status Flag

If the user writes the SSPBUF when a transmit is already in progress (i.e., SSPSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

WCOL must be cleared in software.

17.4.10.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPCON2<6>) is cleared when the slave has sent an Acknowledge ($\overline{ACK}$ = 0) and is set when the slave does not Acknowledge ($\overline{ACK}$ = 1). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

17.4.11 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPCON2<3>).

Note:	The MSSP module must be in an Idle state before the RCEN bit is set, or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting and on each rollover, the state of the SCL pin changes (high-to-low/low-to-high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR are loaded into the SSPBUF, the BF flag bit is set, the SSPIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCL low. The MSSP is now in Idle state, awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>).

17.4.11.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPBUF from SSPSR. It is cleared when the SSPBUF register is read.

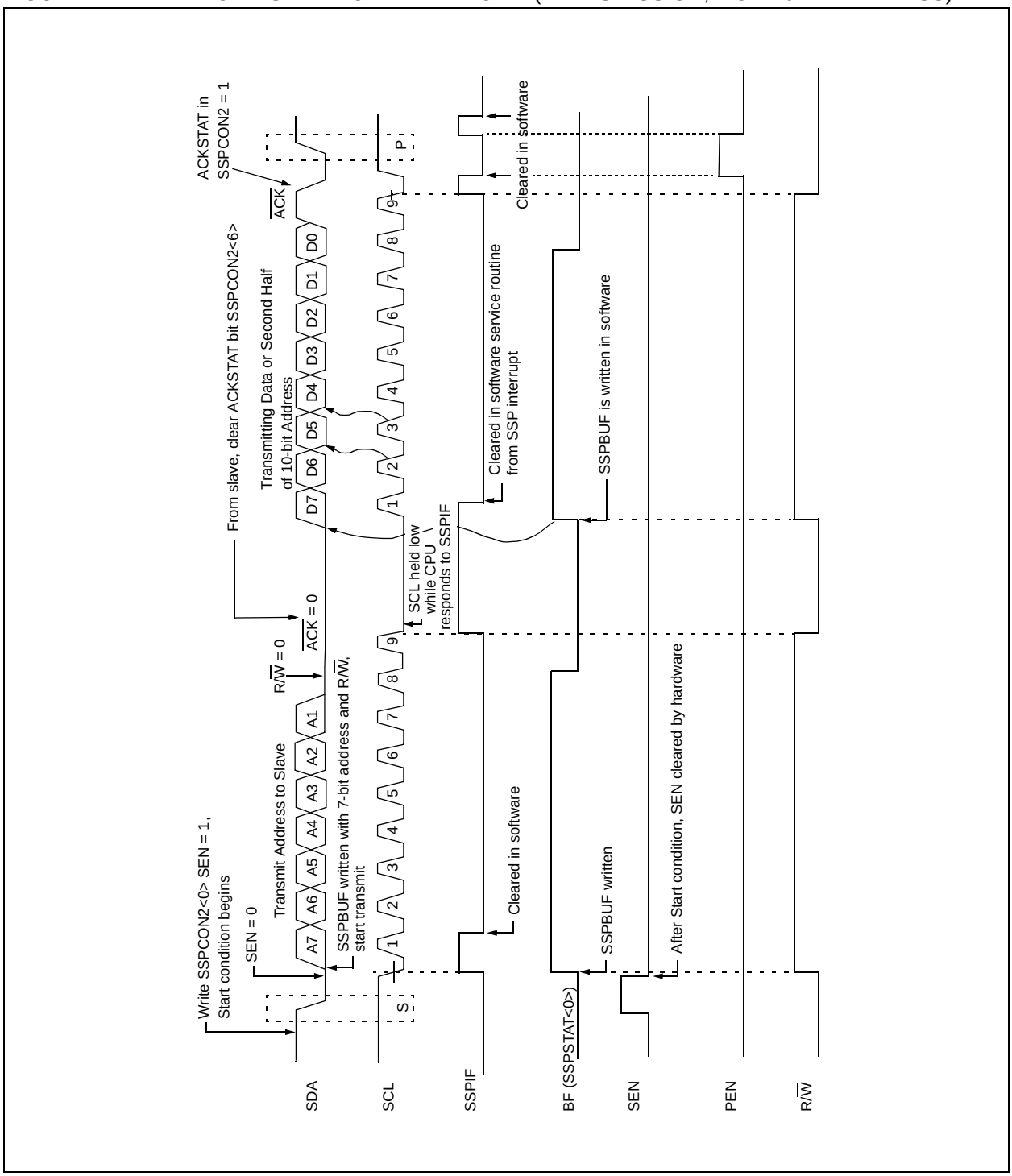
17.4.11.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

17.4.11.3 WCOL Status Flag

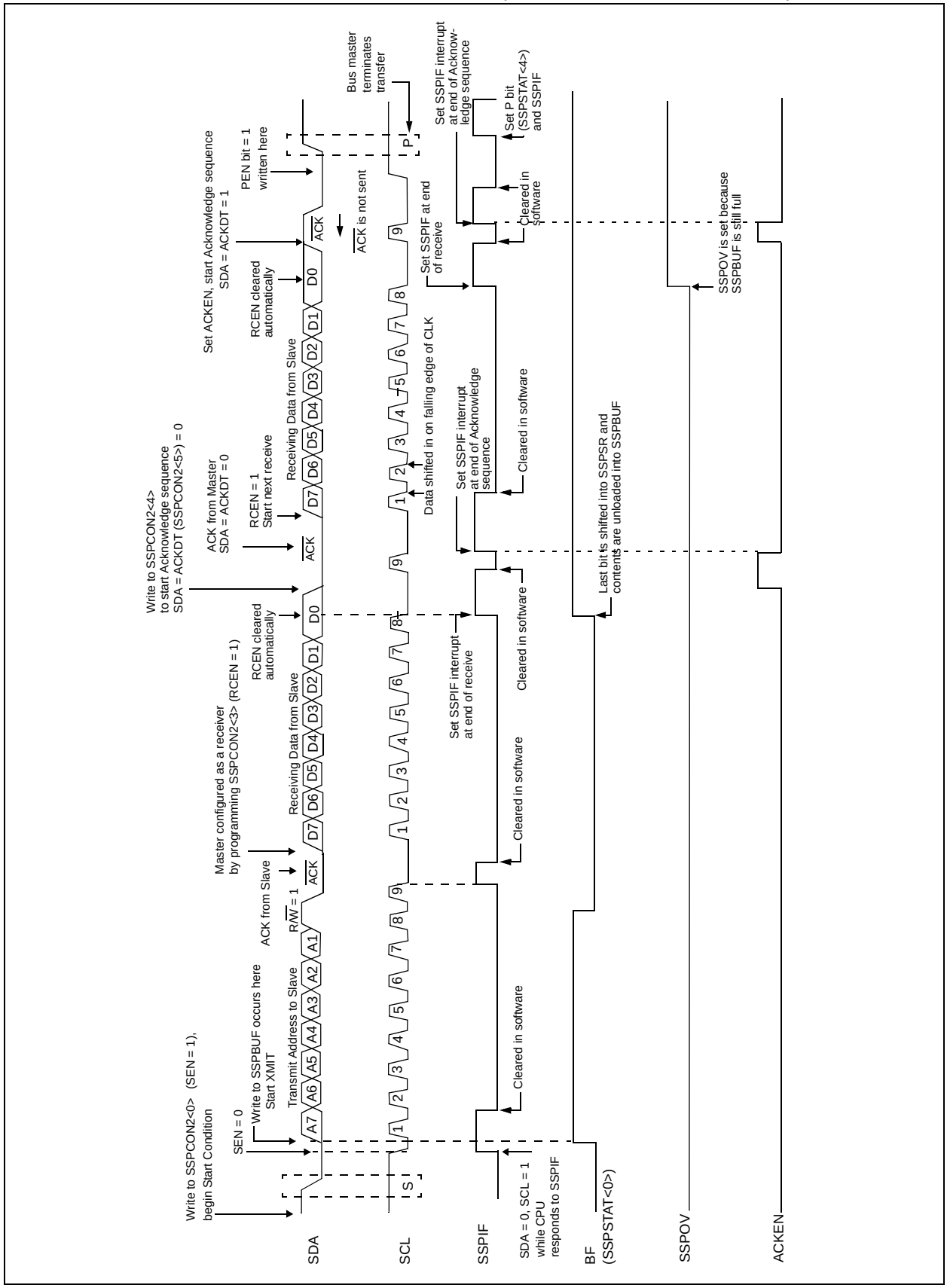
If the user writes the SSPBUF when a receive is already in progress (i.e., SSPSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 17-21: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)



© 2004 Microchip Technology Inc.

FIGURE 17-22: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



17.4.12 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 17-23).

17.4.12.1 WCOL Status Flag

If the user writes the SSPBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

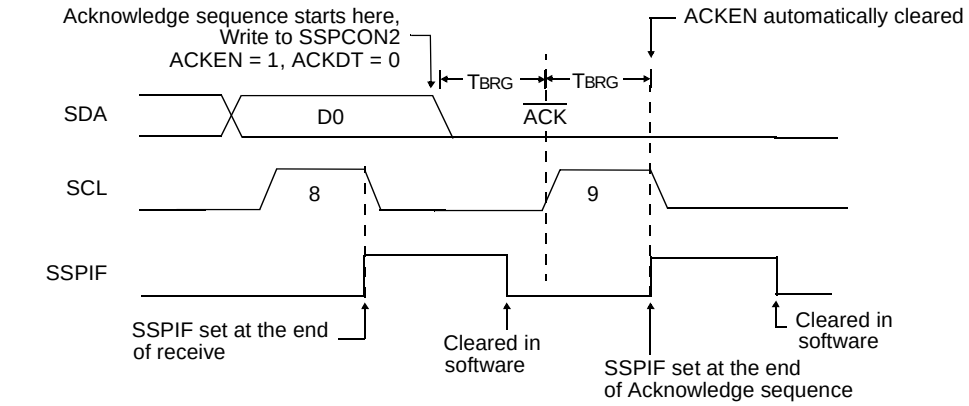
17.4.13 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 17-24).

17.4.13.1 WCOL Status Flag

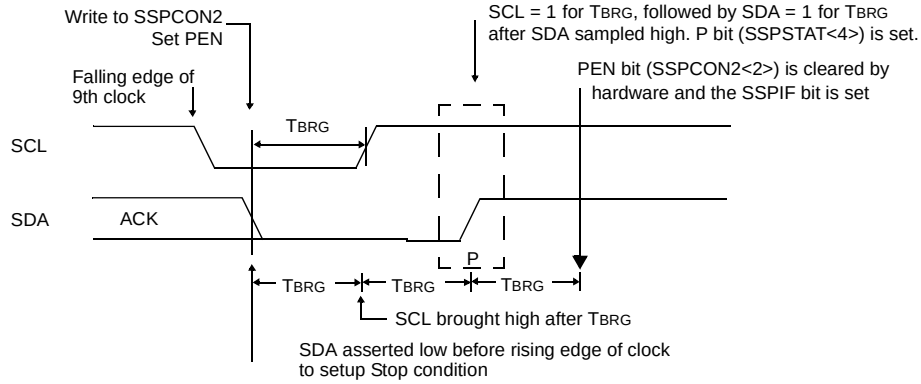
If the user writes the SSPBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 17-23: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 17-24: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one Baud Rate Generator period.

© 2004 Microchip Technology Inc.

17.4.14 SLEEP OPERATION

While in Sleep mode, the I^2C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

17.4.15 EFFECT OF A RESET

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

17.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I^2C bus may be taken when the P bit (SSPSTAT<4>) is set, or the bus is idle with both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA line must be monitored for arbitration, to see if the signal level is the expected output level. This check is performed in hardware, with the result placed in the BCLIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

17.4.17 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA, by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the I^2C port to its Idle state (Figure 17-25).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

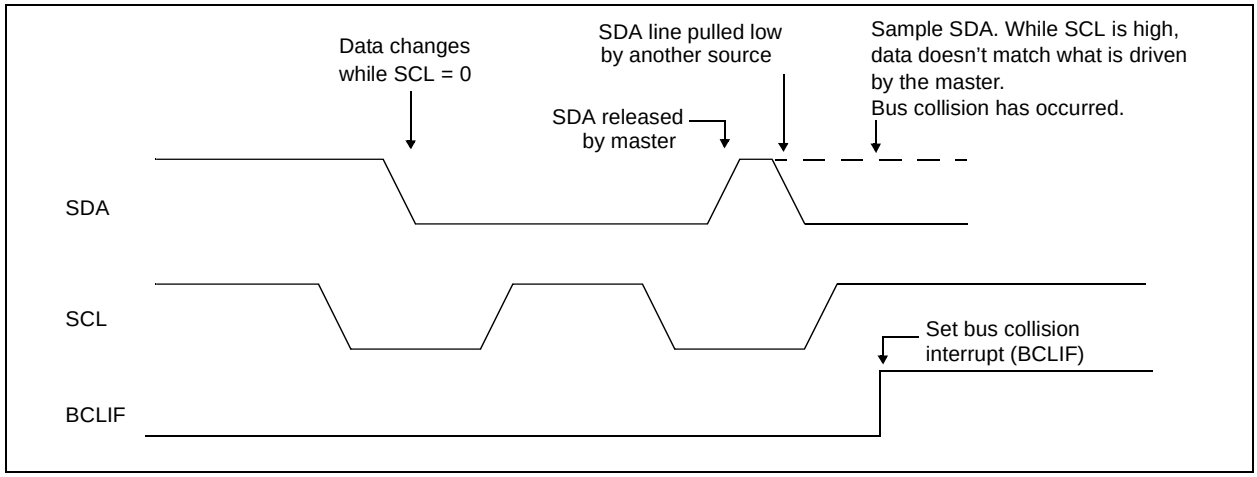
If a Start, Repeated Start, Stop, or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL pins. If a Stop condition occurs, the SSPIF bit will be set.

A write to the SSPBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I^2C bus can be taken when the P bit is set in the SSPSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 17-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



17.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDA or SCL are sampled low at the beginning of the Start condition (Figure 17-26).

b) SCL is sampled low before SDA is asserted low (Figure 17-27).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low, or the SCL pin is already low, then all of the following occur:

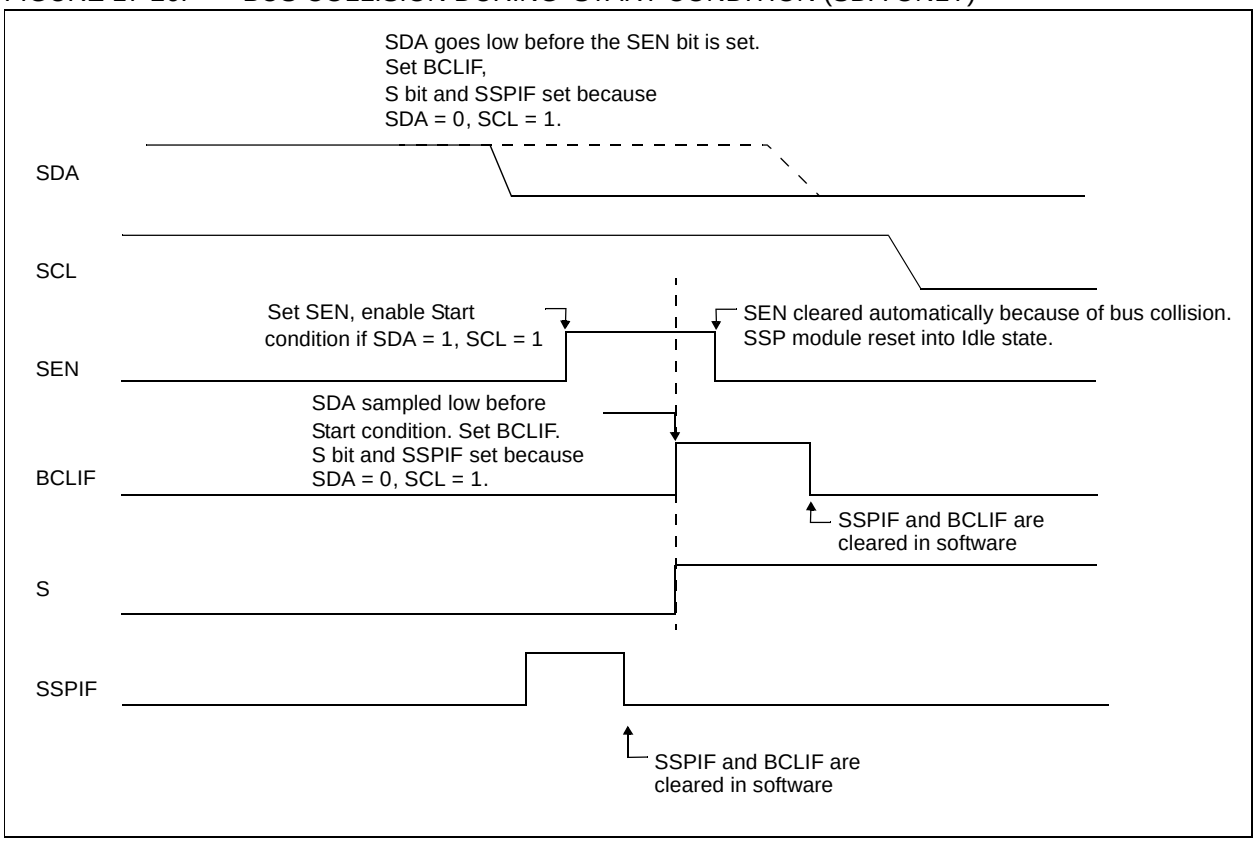
- the Start condition is aborted,
- the BCLIF flag is set and
- the MSSP module is reset to its Idle state (Figure 17-26).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded from SSPADD<6:0> and counts down to '0'. If the SCL pin is sampled low while SDA is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 17-28). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to '0' and during this time, if the SCL pin is sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

Note:	The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 17-26: BUS COLLISION DURING START CONDITION (SDA ONLY)



© 2004 Microchip Technology Inc.

FIGURE 17-27: BUS COLLISION D URING START CONDITION (SCL = 0)

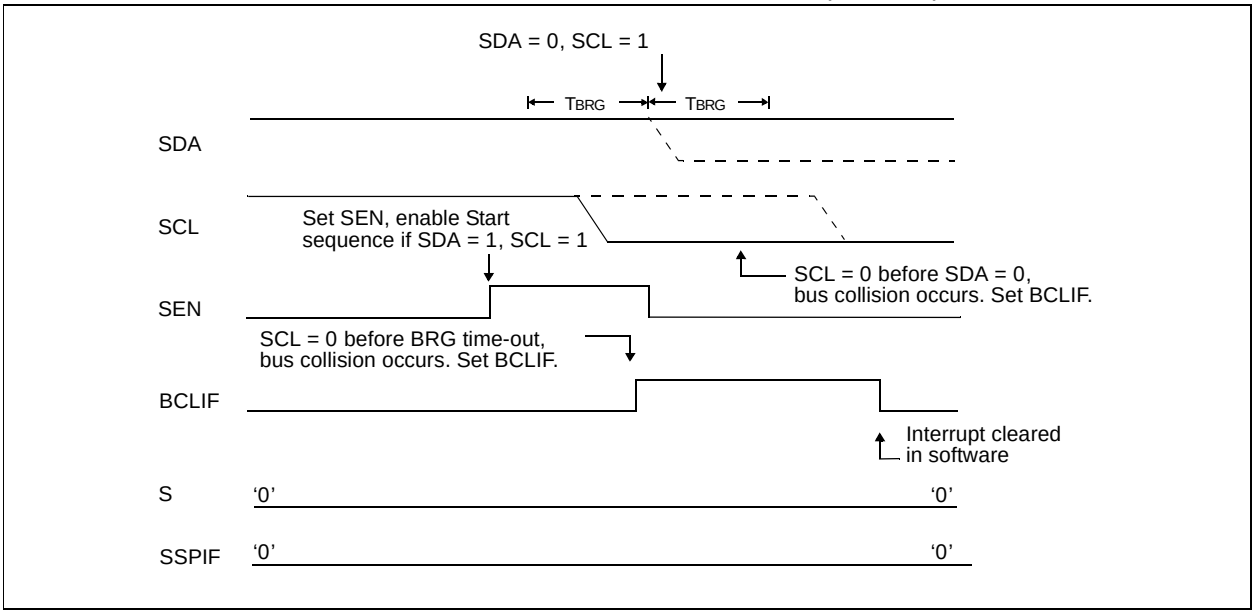
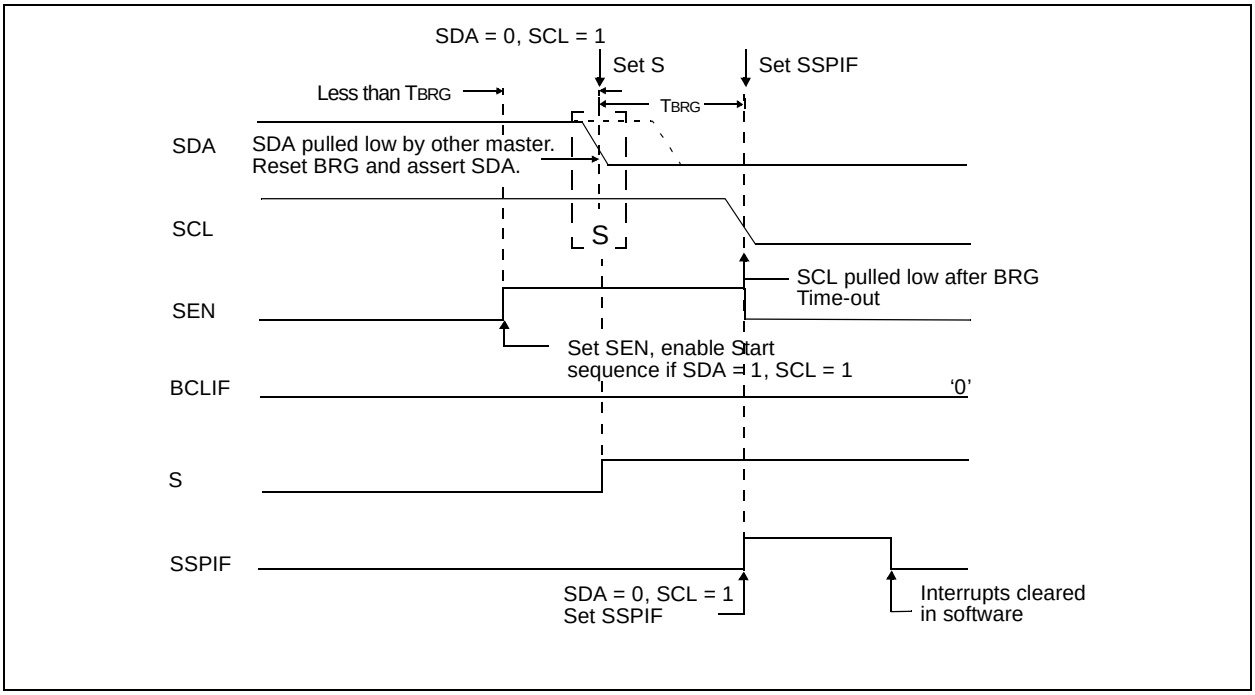


FIGURE 17-28: BRG RESET DUE TO SDA ARBITRAT ION DURING START CONDITION



17.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level.

b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<6:0> and counts down to '0'. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 17-29). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high-to-low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition, Figure 17-30.

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

FIGURE 17-29: BUS COLLISION DURING A REPEAT ED START CONDITION (CASE 1)

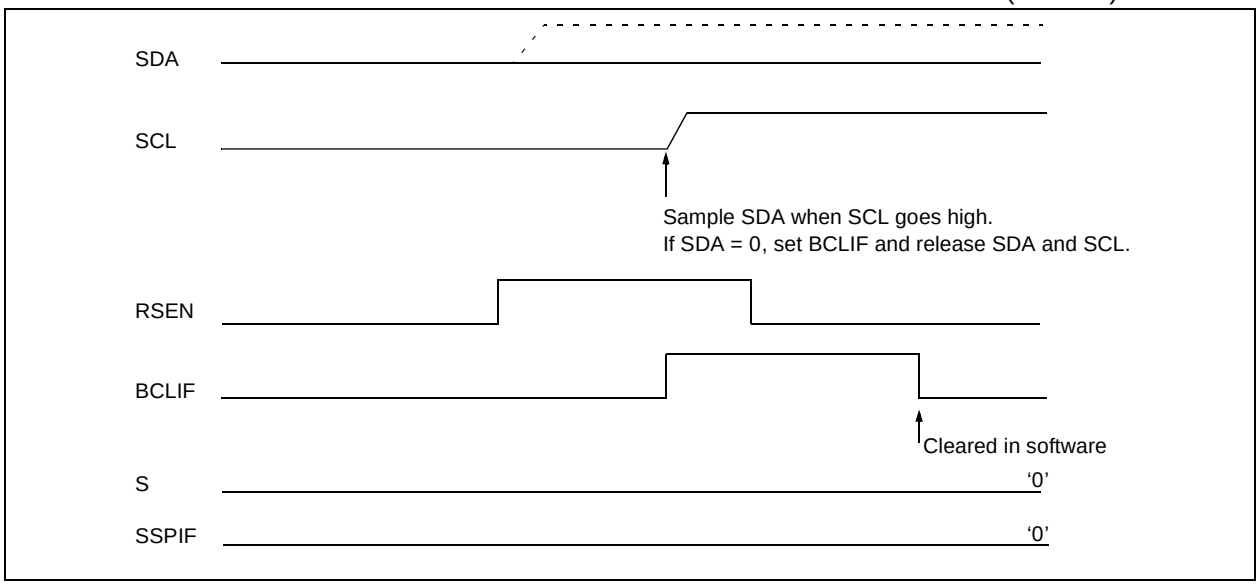
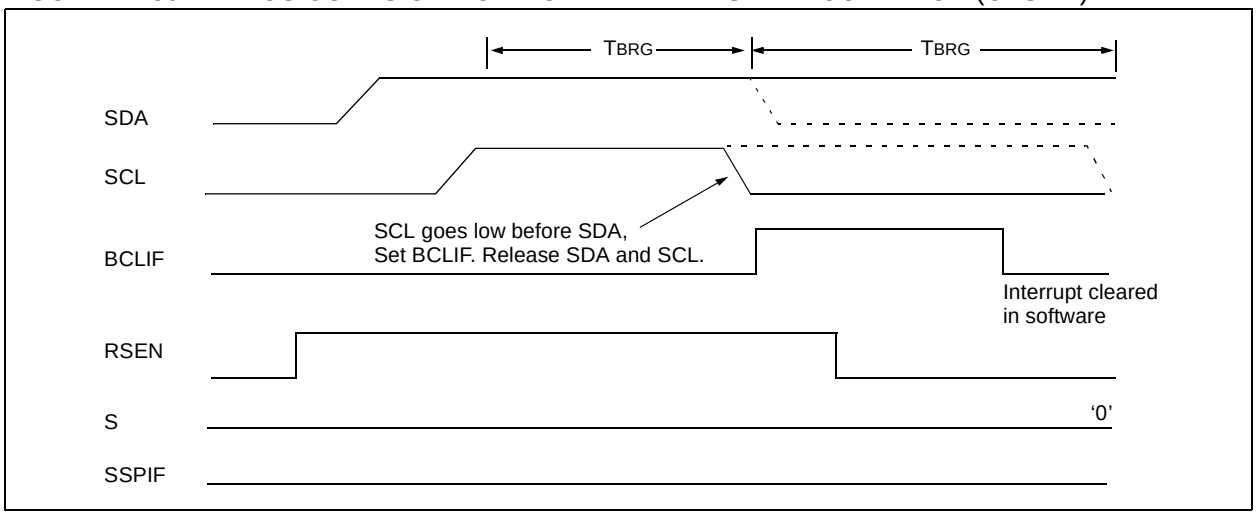


FIGURE 17-30: BUS COLLISION DURING REPEAT ED START CONDITION (CASE 2)



© 2004 Microchip Technology Inc.

17.4.17.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out.

b) After the SCL pin is deasserted, SCL is sampled low before SDA goes high.

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPADD<6:0> and counts down to '0'. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 17-31). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 17-32).

FIGURE 17-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)

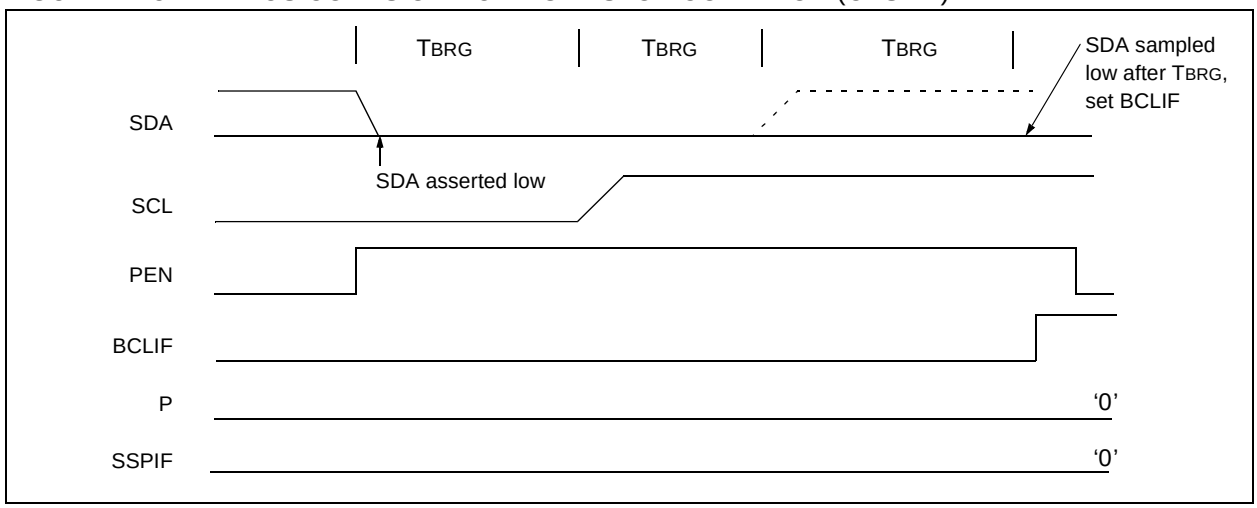
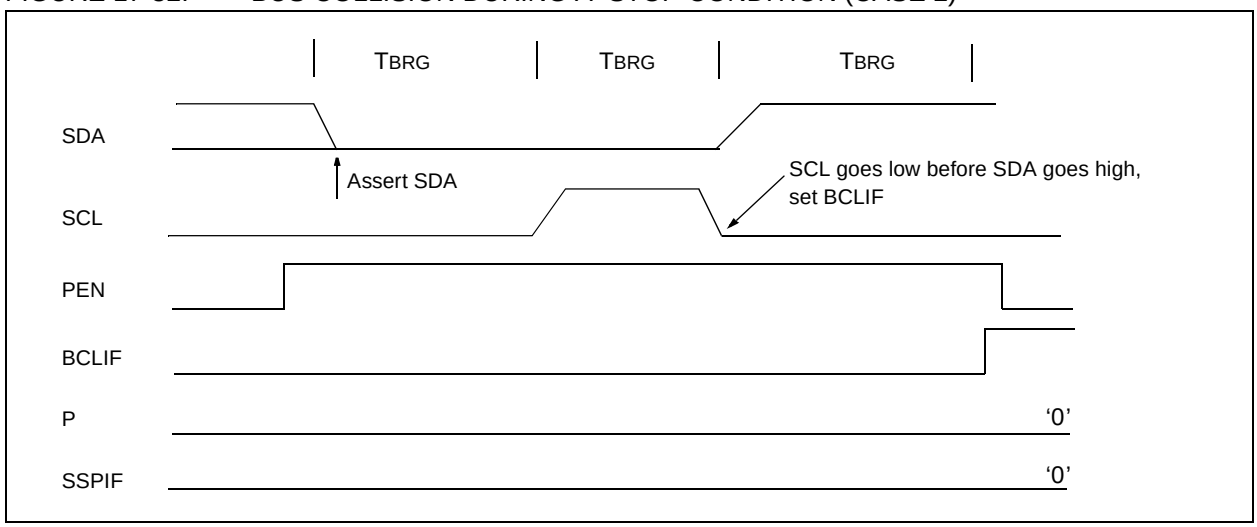


FIGURE 17-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)



NOTES:

© 2004 Microchip Technology Inc.

18.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module (also known as a Serial Communications Interface or SCI) is one of the two types of serial I/O modules available on PIC18FXX20 devices. Each device has two USARTs, which can be configured independently of each other. Each can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or 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 USART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous – Master (half-duplex)
- Synchronous – Slave (half-duplex)

The pins of USART1 and USART2 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 a USART:

- For USART1:
  - bit SPEN (RCSTA1<7>) must be set (= 1)
  - bit TRISC<7> must be set (= 1)
  - bit TRISC<6> must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - bit TRISC<6> must be set (= 1) for Synchronous Slave mode
- For USART2:
  - bit SPEN (RCSTA2<7>) must be set (= 1)
  - bit TRISG<2> must be set (= 1)
  - bit TRISG<1> must be cleared (= 0) for Asynchronous and Synchronous Master modes
  - bit TRISC<6> must be set (= 1) for Synchronous Slave mode

Register 18-1 shows the layout of the Transmit Status and Control registers (TXSTAx) and Register 18-2 shows the layout of the Receive Status and Control registers (RCSTAx). USART1 and USART2 each have their own independent and distinct pairs of transmit and receive control registers, which are identical to each other apart from their names. Similarly, each USART has its own distinct set of transmit, receive and baud rate registers.

Note:	Throughout this section, references to register and bit names that may be associated with a specific USART 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 USART1 or USART2.

REGISTER 18-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 CSRC: Clock Source Select bit

Asynchronous mode:
Don't care.

Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 TX9: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 TXEN: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

Note: SREN/CREN overrides TXEN in Sync mode.

bit 4 SYNC: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 Unimplemented: Read as '0'

bit 2 BRGH: High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode.

bit 1 TRMT: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 TX9D: 9th bit of Transmit Data
Can be address/data bit or a parity bit.

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

© 2004 Microchip Technology Inc.

REGISTER 18-2: RCSTAx: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 SPEN: Serial Port Enable bit
1 = Serial port enabled (configures RX/DT and TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 RX9: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 SREN: Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don't care.

bit 4 CREN: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables receiver
0 = Disables receiver
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 ADDEN: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of 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

bit 2 FERR: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 OERR: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 RX9D: 9th bit of Received Data
This can be address/data bit or a parity bit and must be calculated by user firmware.

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

18.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USARTs. It is a dedicated 8-bit Baud Rate Generator. The SPBRG register controls the period of a free running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTAx<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 18-1 shows the formula for computation of the baud rate for different USART modes, which only apply in Master mode (internal clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGx register can be calculated using the formula in Table 18-1. From this, the error in baud rate can be determined.

Example 18-1 shows the calculation of the baud rate error for the following conditions:

- FOSC = 16 MHz
- Desired Baud Rate = 9600
- BRGH = 0
- SYNC = 0

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the equation in Example 18-1 can reduce the baud rate error in some cases.

Writing a new value to the SPBRGx register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

18.1.1 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 pin.

EXAMPLE 18-1: CALCULATING BAUD RATE ERROR

Desired Baud Rate = FOSC/(64 (X + 1))

Solving for X:

X = ((FOSC/Desired Baud Rate)/64) – 1
X = ((16000000/9600)/64) – 1
X = [25.042] = 25

Calculated Baud Rate = 16000000/(64 (25 + 1))
= 9615

$$\text{Error} = \frac{\text{(Calculated Baud Rate – Desired Baud Rate)}}{\text{Desired Baud Rate}}$$
= (9615 – 9600)/9600
= 0.16%

TABLE 18-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = FOSC/(64(X + 1))	Baud Rate = FOSC/(16(X + 1))
1	(Synchronous) Baud Rate = FOSC/(4(X + 1))	N/A

Legend: X = value in SPBRGx (0 to 255)

TABLE 18-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TXSTAx	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
SPBRGx	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as '0'. Shaded cells are not used by the BRG.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where 'x' indicates the particular module. Bit names and Reset values are identical between modules.

© 2004 Microchip Technology Inc.

TABLE 18-3: BAUD RATES FOR SYNCHRONOUS MODE

BAUD RATE (Kbps)	FOSC = 40 MHz			33 MHz			25 MHz			20 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	NA	-	-	NA	-	-
9.6	NA	-	-	NA	-	-	NA	-	-	NA	-	-
19.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
76.8	76.92	+0.16	129	77.10	+0.39	106	77.16	+0.47	80	76.92	+0.16	64
96	96.15	+0.16	103	95.93	-0.07	85	96.15	+0.16	64	96.15	+0.16	51
300	303.03	+1.01	32	294.64	-1.79	27	297.62	-0.79	20	294.12	-1.96	16
500	500	0	19	485.30	-2.94	16	480.77	-3.85	12	500	0	9
HIGH	10000	-	0	8250	-	0	6250	-	0	5000	-	0
LOW	39.06	-	255	32.23	-	255	24.41	-	255	19.53	-	255

BAUD RATE (Kbps)	FOSC = 16 MHz			10 MHz			7.15909 MHz			5.0688 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	NA	-	-	NA	-	-
9.6	NA	-	-	NA	-	-	9.62	+0.23	185	9.60	0	131
19.2	19.23	+0.16	207	19.23	+0.16	129	19.24	+0.23	92	19.20	0	65
76.8	76.92	+0.16	51	75.76	-1.36	32	77.82	+1.32	22	74.54	-2.94	16
96	95.24	-0.79	41	96.15	+0.16	25	94.20	-1.88	18	97.48	+1.54	12
300	307.70	+2.56	12	312.50	+4.17	7	298.35	-0.57	5	316.80	+5.60	3
500	500	0	7	500	0	4	447.44	-10.51	3	422.40	-15.52	2
HIGH	4000	-	0	2500	-	0	1789.80	-	0	1267.20	-	0
LOW	15.63	-	255	9.77	-	255	6.99	-	255	4.95	-	255

BAUD RATE (Kbps)	FOSC = 4 MHz			3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	0.30	+1.14	26
1.2	NA	-	-	NA	-	-	1.20	+0.16	207	1.17	-2.48	6
2.4	NA	-	-	NA	-	-	2.40	+0.16	103	2.73	+13.78	2
9.6	9.62	+0.16	103	9.62	+0.23	92	9.62	+0.16	25	8.20	-14.67	0
19.2	19.23	+0.16	51	19.04	-0.83	46	19.23	+0.16	12	NA	-	-
76.8	76.92	+0.16	12	74.57	-2.90	11	83.33	+8.51	2	NA	-	-
96	1000	+4.17	9	99.43	+3.57	8	83.33	-13.19	2	NA	-	-
300	333.33	+11.11	2	298.30	-0.57	2	250	-16.67	0	NA	-	-
500	500	0	1	447.44	-10.51	1	NA	-	-	NA	-	-
HIGH	1000	-	0	894.89	-	0	250	-	0	8.20	-	0
LOW	3.91	-	255	3.50	-	255	0.98	-	255	0.03	-	255

TABLE 18-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)

BAUD RATE (Kbps)	FOSC = 40 MHz			33 MHz			25 MHz			20 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	2.40	-0.07	214	2.40	-0.15	162	2.40	+0.16	129
9.6	9.62	+0.16	64	9.55	-0.54	53	9.53	-0.76	40	9.47	-1.36	32
19.2	18.94	-1.36	32	19.10	-0.54	26	19.53	+1.73	19	19.53	+1.73	15
76.8	78.13	+1.73	7	73.66	-4.09	6	78.13	+1.73	4	78.13	+1.73	3
96	89.29	-6.99	6	103.13	+7.42	4	97.66	+1.73	3	104.17	+8.51	2
300	312.50	+4.17	1	257.81	-14.06	1	NA	-	-	312.50	+4.17	0
500	625	+25.00	0	NA	-	-	NA	-	-	NA	-	-
HIGH	625	-	0	515.63	-	0	390.63	-	0	312.50	-	0
LOW	2.44	-	255	2.01	-	255	1.53	-	255	1.22	-	255

BAUD RATE (Kbps)	FOSC = 16 MHz			10 MHz			7.15909 MHz			5.0688 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	1.20	+0.16	207	1.20	+0.16	129	1.20	+0.23	92	1.20	0	65
2.4	2.40	+0.16	103	2.40	+0.16	64	2.38	-0.83	46	2.40	0	32
9.6	9.62	+0.16	25	9.77	+1.73	15	9.32	-2.90	11	9.90	+3.13	7
19.2	19.23	+0.16	12	19.53	+1.73	7	18.64	-2.90	5	19.80	+3.13	3
76.8	83.33	+8.51	2	78.13	+1.73	1	111.86	+45.65	0	79.20	+3.13	0
96	83.33	-13.19	2	78.13	-18.62	1	NA	-	-	NA	-	-
300	250	-16.67	0	156.25	-47.92	0	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	250	-	0	156.25	-	0	111.86	-	0	79.20	-	0
LOW	0.98	-	255	0.61	-	255	0.44	-	255	0.31	-	255

BAUD RATE (Kbps)	FOSC = 4 MHz			3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	0.30	-0.16	207	0.30	+0.23	185	0.30	+0.16	51	0.26	-14.67	1
1.2	1.20	+1.67	51	1.19	-0.83	46	1.20	+0.16	12	NA	-	-
2.4	2.40	+1.67	25	2.43	+1.32	22	2.23	-6.99	6	NA	-	-
9.6	8.93	-6.99	6	9.32	-2.90	5	7.81	-18.62	1	NA	-	-
19.2	20.83	+8.51	2	18.64	-2.90	2	15.63	-18.62	0	NA	-	-
76.8	62.50	-18.62	0	55.93	-27.17	0	NA	-	-	NA	-	-
96	NA	-	-	NA	-	-	NA	-	-	NA	-	-
300	NA	-	-	NA	-	-	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	62.50	-	0	55.93	-	0	15.63	-	0	0.51	-	0
LOW	0.24	-	255	0.22	-	255	0.06	-	255	0.002	-	255

© 2004 Microchip Technology Inc.

TABLE 18-5: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)

BAUD RATE (Kbps)	FOSC = 40 MHz			33 MHz			25 MHz			20 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	NA	-	-	NA	-	-
9.6	NA	-	-	9.60	-0.07	214	9.59	-0.15	162	9.62	+0.16	129
19.2	19.23	+0.16	129	19.28	+0.39	106	19.30	+0.47	80	19.23	+0.16	64
76.8	75.76	-1.36	32	76.39	-0.54	26	78.13	+1.73	19	78.13	+1.73	15
96	96.15	+0.16	25	98.21	+2.31	20	97.66	+1.73	15	96.15	+0.16	12
300	312.50	+4.17	7	294.64	-1.79	6	312.50	+4.17	4	312.50	+4.17	3
500	500	0	4	515.63	+3.13	3	520.83	+4.17	2	416.67	-16.67	2
HIGH	2500	-	0	2062.50	-	0	1562.50	-	0	1250	-	0
LOW	9.77	-	255	8,06	-	255	6.10	-	255	4.88	-	255

BAUD RATE (Kbps)	FOSC = 16 MHz			10 MHz			7.15909 MHz			5.0688 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	2.41	+0.23	185	2.40	0	131
9.6	9.62	+0.16	103	9.62	+0.16	64	9.52	-0.83	46	9.60	0	32
19.2	19.23	+0.16	51	18.94	-1.36	32	19.45	+1.32	22	18.64	-2.94	16
76.8	76.92	+0.16	12	78.13	+1.73	7	74.57	-2.90	5	79.20	+3.13	3
96	100	+4.17	9	89.29	-6.99	6	89.49	-6.78	4	105.60	+10.00	2
300	333.33	+11.11	2	312.50	+4.17	1	447.44	+49.15	0	316.80	+5.60	0
500	500	0	1	625	+25.00	0	447.44	-10.51	0	NA	-	-
HIGH	1000	-	0	625	-	0	447.44	-	0	316.80	-	0
LOW	3.91	-	255	2.44	-	255	1.75	-	255	1.24	-	255

BAUD RATE (Kbps)	FOSC = 4 MHz			3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	0.30	+0.16	207	0.29	-2.48	6
1.2	1.20	+0.16	207	1.20	+0.23	185	1.20	+0.16	51	1.02	-14.67	1
2.4	2.40	+0.16	103	2.41	+0.23	92	2.40	+0.16	25	2.05	-14.67	0
9.6	9.62	+0.16	25	9.73	+1.32	22	8.93	-6.99	6	NA	-	-
19.2	19.23	+0.16	12	18.64	-2.90	11	20.83	+8.51	2	NA	-	-
76.8	NA	-	-	74.57	-2.90	2	62.50	-18.62	0	NA	-	-
96	NA	-	-	111.86	+16.52	1	NA	-	-	NA	-	-
300	NA	-	-	223.72	-25.43	0	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	250	-	0	55.93	-	0	62.50	-	0	2.05	-	0
LOW	0.98	-	255	0.22	-	255	0.24	-	255	0.008	-	255

18.2 USART Asynchronous Mode

In this mode, the USARTs use standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip dedicated 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The USART's transmitter and receiver are functionally independent, but use the same data format and baud rate. The Baud Rate Generator produces a clock, either 16 or 64 times the bit shift rate, depending on bit BRGH (TXSTAx<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during Sleep.

Asynchronous mode is selected by clearing bit SYNC (TXSTAx<4>).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

18.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 18-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer register, 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 flag bit, TXx1IF (PIR1<4> for USART1, PIR3<4> for USART2), is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXxIE (PIE1<4> for USART1, PIE<4> for USART2). Flag bit TXxIF will be set, regardless of the state of enable bit TXxIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREGx register. While flag bit TXIF indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<1>), shows the status of the TSR register. Status bit 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.

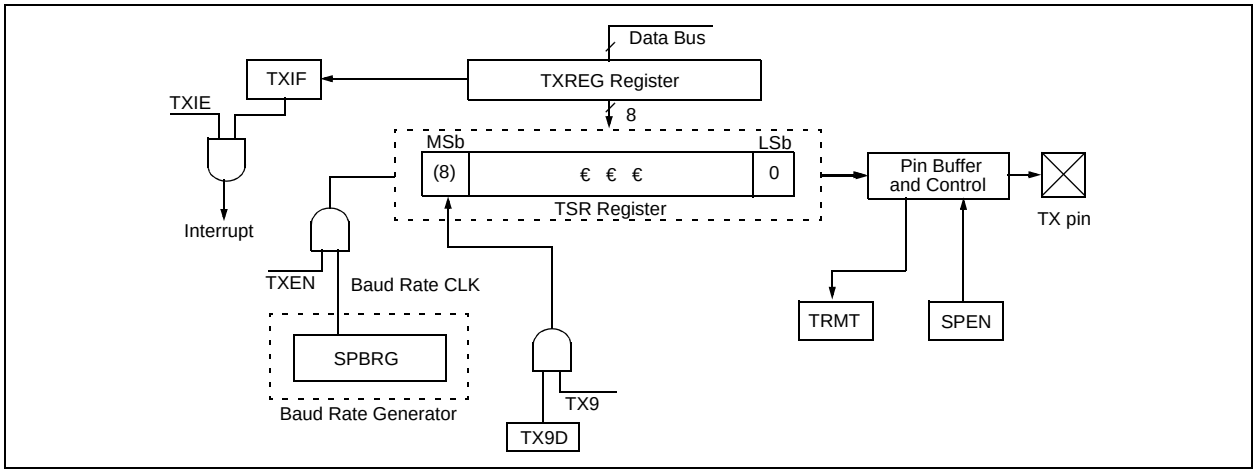
Note 1:	The TSR register is not mapped in data memory, so it is not available to the user.
2:	Flag bit TXIF is set when enable bit TXEN is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGx register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (Section 18.1 "USART Baud Rate Generator (BRG)").
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXxIE in the appropriate PIE register.
4. If 9-bit transmission is desired, set transmit bit TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit TXEN, which will also set bit TXxIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREGx register (starts transmission).

Note:	TXIF is not cleared immediately upon loading data into the transmit buffer TXREG. The flag bit becomes valid in the second instruction cycle following the load instruction.

FIGURE 18-1: USART TRANSMIT BLOCK DIAGRAM



© 2004 Microchip Technology Inc.

FIGURE 18-2: ASYNCHRONOUS TRANSMISSION

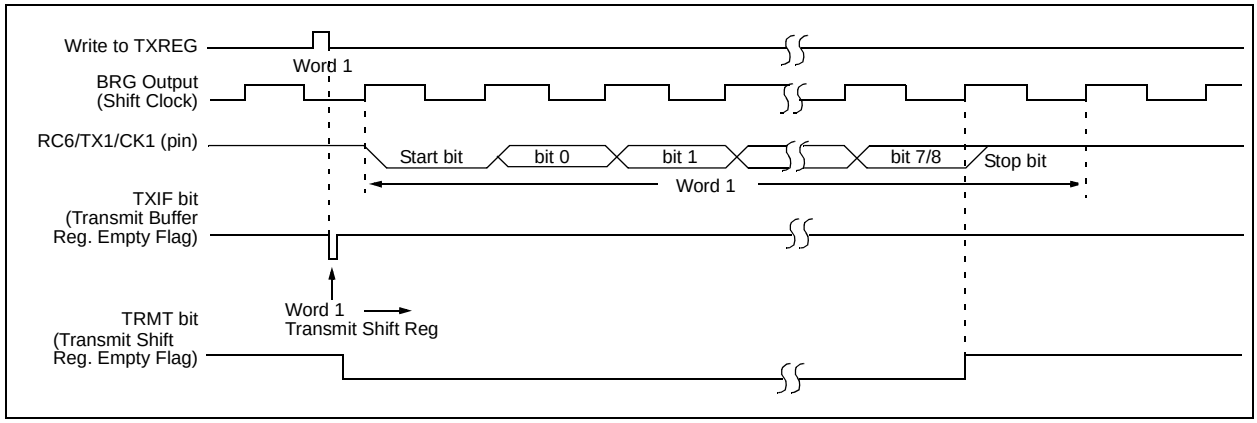
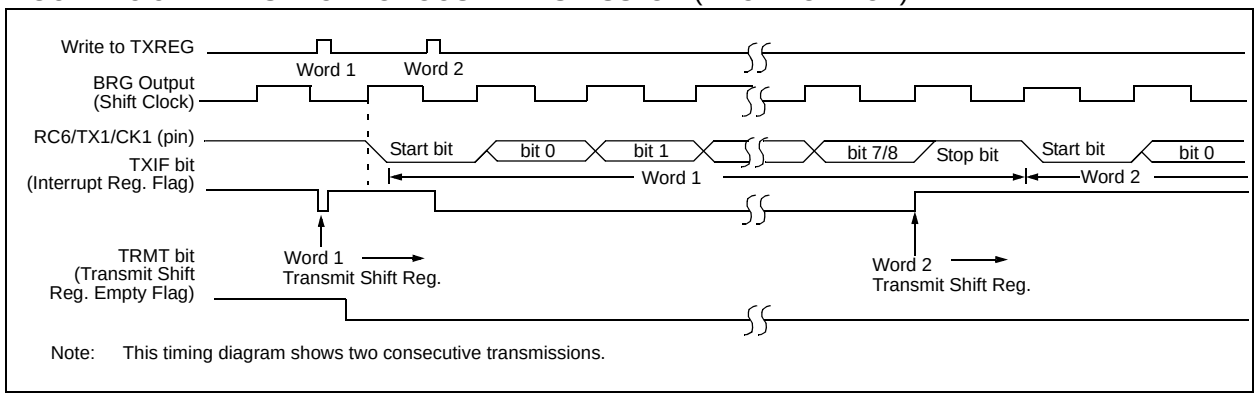


FIGURE 18-3: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 18-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREGx(1)	USART Transmit Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where 'x' indicates the particular module. Bit names and Reset values are identical between modules.

18.2.2 USART ASYNCHRONOUS RECEIVER

The USART receiver block diagram is shown in Figure 18-4. The data is received on the pin (RC7/RX1/DT1 or RG2/RX2/DT2) and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

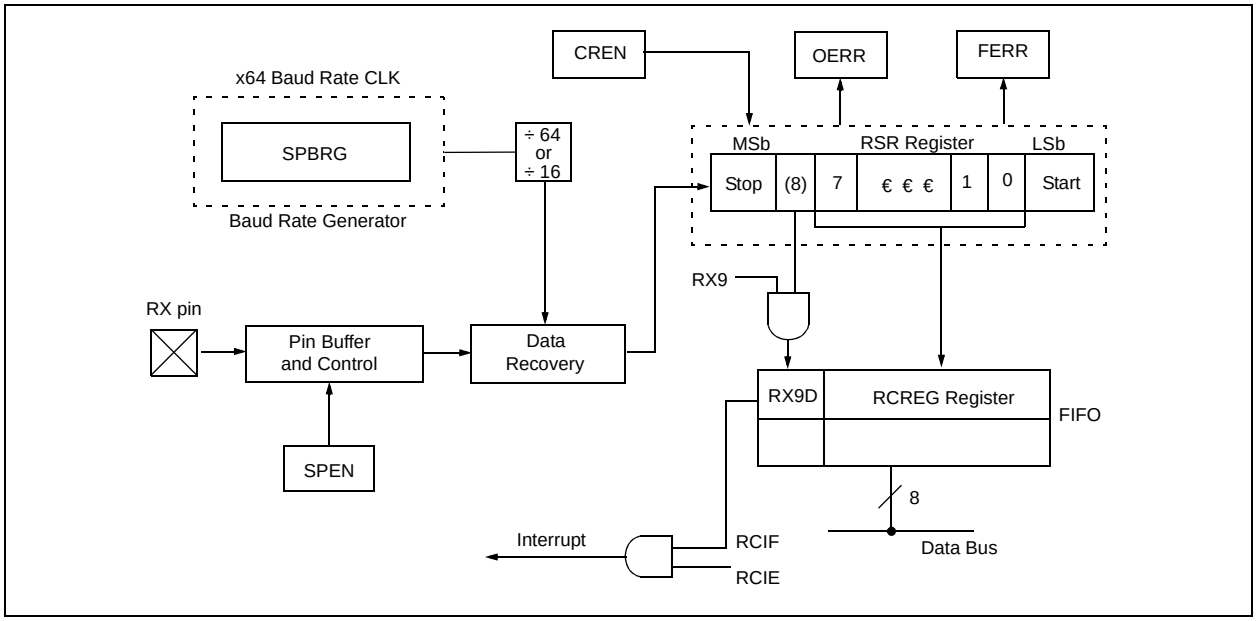
1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (Section 18.1 “USART Baud Rate Generator (BRG)”).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit RCxIE.
4. If 9-bit reception is desired, set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCxIF will be set when reception is complete and an interrupt will be generated if enable bit RCxIE was set.
7. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

18.2.3 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:

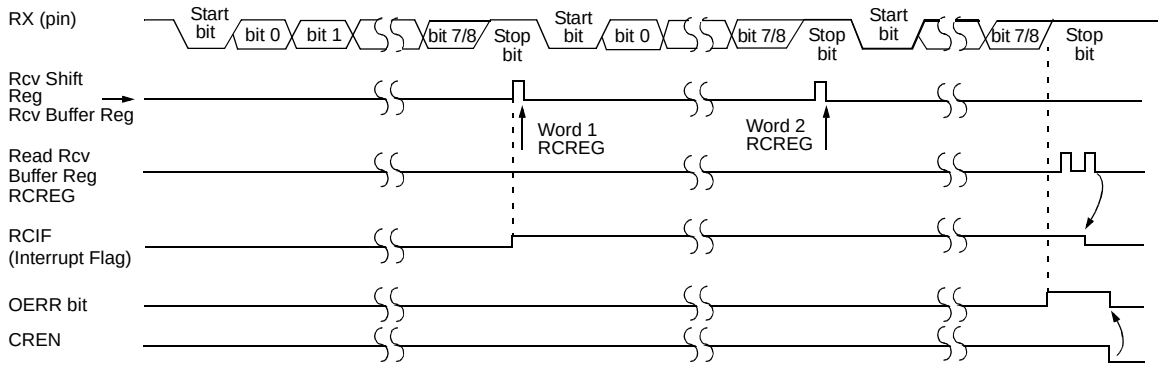
1. Initialize the SPBRGx register for the appropriate baud rate. If a high-speed baud rate is required, set the BRGH bit.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCIP bit.
4. Set the RX9 bit to enable 9-bit reception.
5. Set the ADDEN bit to enable address detect.
6. Enable reception by setting the CREN bit.
7. The RCxIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCxIE and GIE bits are set.
8. Read the RCSTAx register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREGx to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 18-4: USART RECEIVE BLOCK DIAGRAM



© 2004 Microchip Technology Inc.

FIGURE 18-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The RCREG (receive buffer) is read after the third word, causing the OERR (overrun) bit to be set.

TABLE 18-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RC1IF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RC1IE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RC1IP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx(1)	USART Receive Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where 'x' indicates the particular module. Bit names and Reset values are identical between modules.

18.3 USART Synchronous Master Mode

In Synchronous Master 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 appropriate I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTAx<7>).

18.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 18-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. 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 TCYCLE), the TXREGx is empty and interrupt bit TXxIF (PIR1<4> for USART1, PIR3<4> for USART2) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXxIE (PIE1<4> for USART1, PIE3<4> for USART2). Flag bit TXxIF will be set, regardless of the state of enable bit TXxIE and 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 has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (Section 18.1 “USART Baud Rate Generator (BRG)”).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXxIE in the appropriate PIE register.
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.

Note:	TXIF is not cleared immediately upon loading data into the transmit buffer TXREG. The flag bit becomes valid in the second instruction cycle following the load instruction.

TABLE 18-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

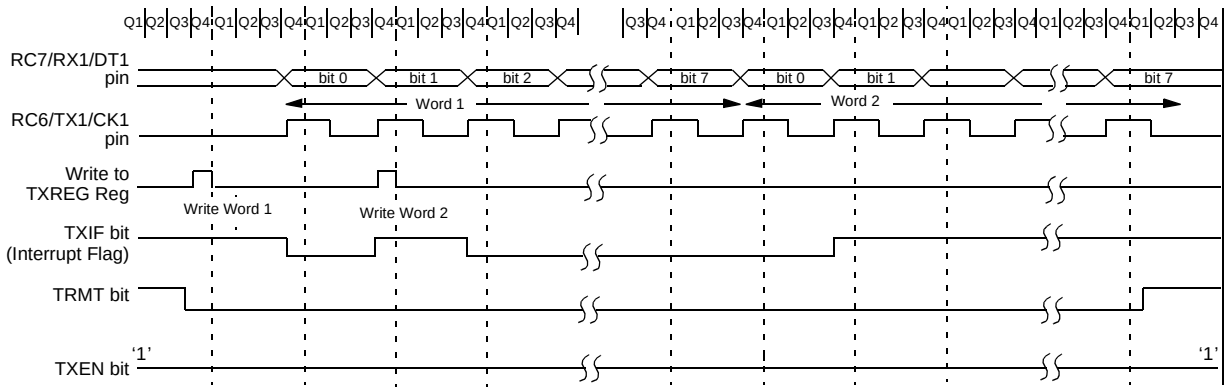
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREGx(1)	USART Transmit Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as ‘0’. Shaded cells are not used for synchronous master transmission.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where ‘x’ indicates the particular module. Bit names and Reset values are identical between modules.

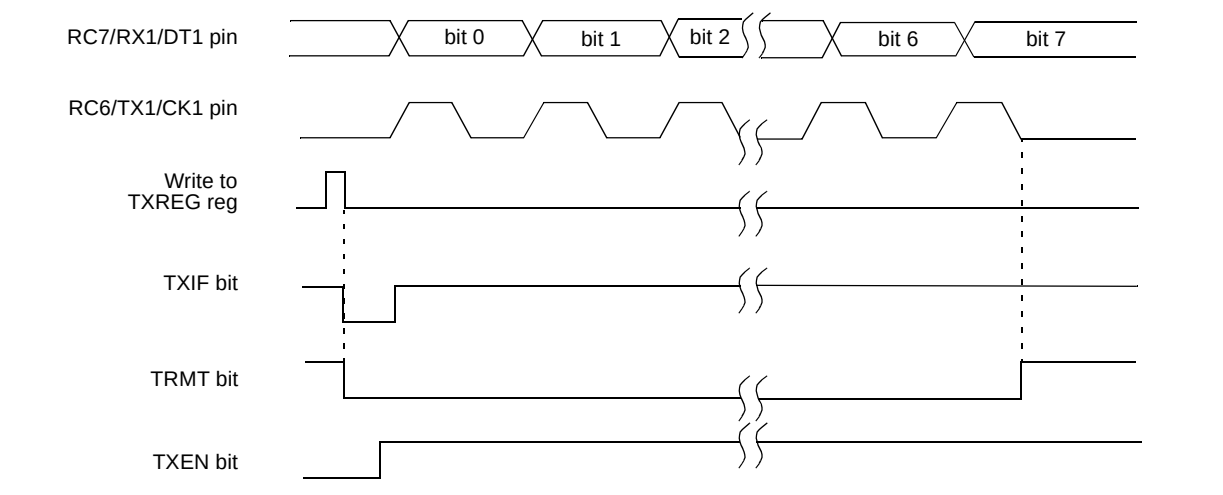
© 2004 Microchip Technology Inc.

FIGURE 18-6: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode; SPBRG = 0. Continuous transmission of two 8-bit words.

FIGURE 18-7: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



18.3.2 USART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either enable bit SREN (RCSTAx<5>) or enable bit CREN (RCSTAx<4>). Data is sampled on the RXx pin (RC7/RX1/DT1 or RG2/RX2/DT2) 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 SPBRGx register for the appropriate baud rate (Section 18.1 “USART Baud Rate Generator (BRG)”).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, set enable bit RCxIE in the appropriate PIE register.
5. If 9-bit reception is desired, set bit RX9.
6. If a single reception is required, set bit SREN. For continuous reception, set bit CREN.
7. Interrupt flag bit RCxIF will be set when reception is complete and an interrupt will be generated if the enable bit RCxIE was set.
8. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREGx register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

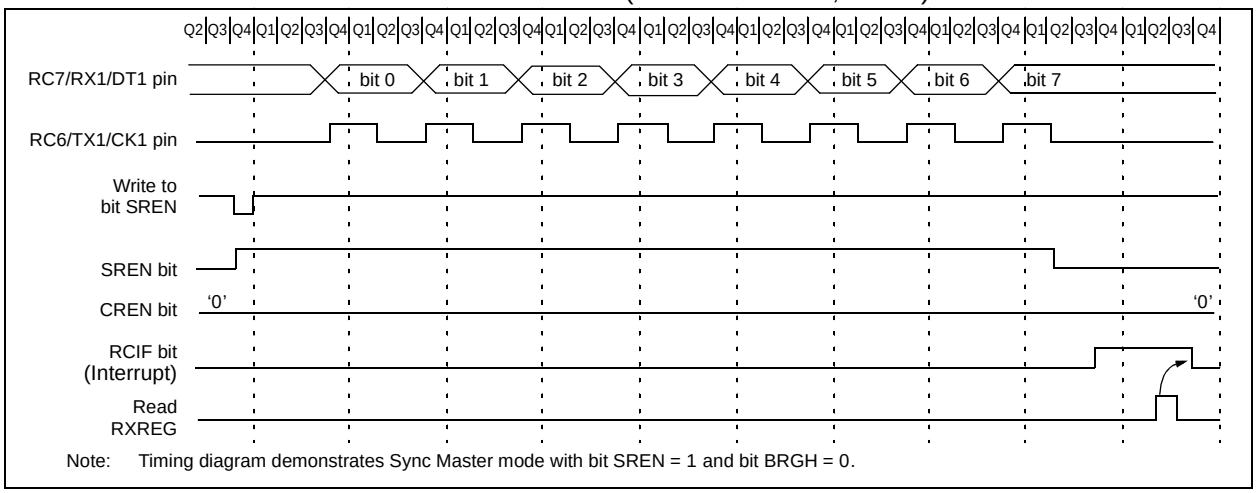
TABLE 18-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx(1)	USART Receive Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as ‘0’. Shaded cells are not used for synchronous master reception.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where ‘x’ indicates the particular module. Bit names and Reset values are identical between modules.

FIGURE 18-8: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

© 2004 Microchip Technology Inc.

18.4 USART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the TXx pin (RC6/TX1/CK1 or RG1/TX2/CK2), instead of being supplied internally in Master mode. TRISC<6> must be set for this mode. This allows the device to transfer or receive data while in Sleep mode. Slave mode is entered by clearing bit CSRC (TXSTAx<7>).

18.4.1 USART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREG register.
c) Flag bit TXxIF will not be set.
d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit TXxIF will now be set.
e) If enable bit TXxIE is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, set enable bit TXxIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 18-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREGx(1)	USART Transmit Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where 'x' indicates the particular module. Bit names and Reset values are identical between modules.

18.4.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the Sleep mode and bit SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting bit CREN prior to the SLEEPinstruction, then a word may be received during Sleep. On completely receiving the word, the RSR register will transfer the data to the RCREG register and if enable bit RCxIE bit is set, the interrupt generated 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 Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCxIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCxIF will be set when reception is complete. An interrupt will be generated if enable bit RCxIE was set.
6. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREGx register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 18-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	--00 0000
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	--00 0000
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	--11 1111
RCSTAx(1)	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx(1)	USART Receive Register								0000 0000	0000 0000
TXSTAx(1)	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRGx(1)	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

Note 1: Register names generically refer to both of the identically named registers for the two USART modules, where ‘x’ indicates the particular module. Bit names and Reset values are identical between modules.

© 2004 Microchip Technology Inc.

19.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The analog-to-digital (A/D) converter module has 12 inputs for the PIC18F6X20 devices and 16 for the PIC18F8X20 devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 19-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 19-2, configures the functions of the port pins. The ADCON2 register, shown in Register 19-3, configures the A/D clock source and justification.

REGISTER 19-1: ADCON0 REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

bit 7-6 Unimplemented: Read as '0'

bit 5-2 CHS3:CHS0: Analog Channel Select bits

0000 = Channel 0 (AN0)
0001 = Channel 1 (AN1)
0010 = Channel 2 (AN2)
0011 = Channel 3 (AN3)
0100 = Channel 4 (AN4)
0101 = Channel 5 (AN5)
0110 = Channel 6 (AN6)
0111 = Channel 7 (AN7)
1000 = Channel 8 (AN8)
1001 = Channel 9 (AN9)
1010 = Channel 10 (AN10)
1011 = Channel 11 (AN11)
1100 = Channel 12 (AN12)(1)
1101 = Channel 13 (AN13)(1)
1110 = Channel 14 (AN14)(1)
1111 = Channel 15 (AN15)(1)

Note 1: These channels are not available on the PIC18F6X20 (64-pin) devices.

bit 1 GO/$\overline{\text{DONE}}$: A/D Conversion Status bit

When ADON = 1:

1 = A/D conversion in progress (setting this bit starts the A/D conversion, which is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

bit 0 ADON: A/D On bit

1 = A/D converter module is enabled
0 = A/D converter module is disabled

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

REGISTER 19-2: ADCON1 REGISTER

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

bit 7-6 Unimplemented: Read as '0'

bit 5-4 VCFG1:VCFG0: Voltage Reference Configuration bits:

VCFG1 VCFG0	A/D V_{REF+}	A/D V_{REF-}
00	AV_{DD}	AV_{SS}
01	External V_{REF+}	AV_{SS}
10	AV_{DD}	External V_{REF-}
11	External V_{REF+}	External V_{REF-}

bit 3-0 PCFG3:PCFG0: A/D Port Configuration Control bits:

PCFG3 PCFG0	AN15	AN14	AN13	AN12	AN11	AN10	AN9	AN8	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A	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: Shaded cells indicate A/D channels available only on PIC18F8X20 devices.

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

© 2004 Microchip Technology Inc.

REGISTER 19-3: ADCON2 REGISTER

R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	—	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 7 ADFM: A/D Result Format Select bit
1 = Right justified
0 = Left justified

bit 6-3 Unimplemented: Read as ‘0’

bit 2-0 ADCS1:ADCS0: A/D Conversion Clock Select bits
000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC (clock derived from an RC oscillator = 1 MHz max)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC (clock derived from an RC oscillator = 1 MHz max)

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

The analog reference voltage is software selectable to either the device’s positive and negative supply voltage (V_{DD} and V_{SS}), or the voltage level on the RA3/AN3/V_{REF}+ pin and RA2/AN2/V_{REF}- pin.

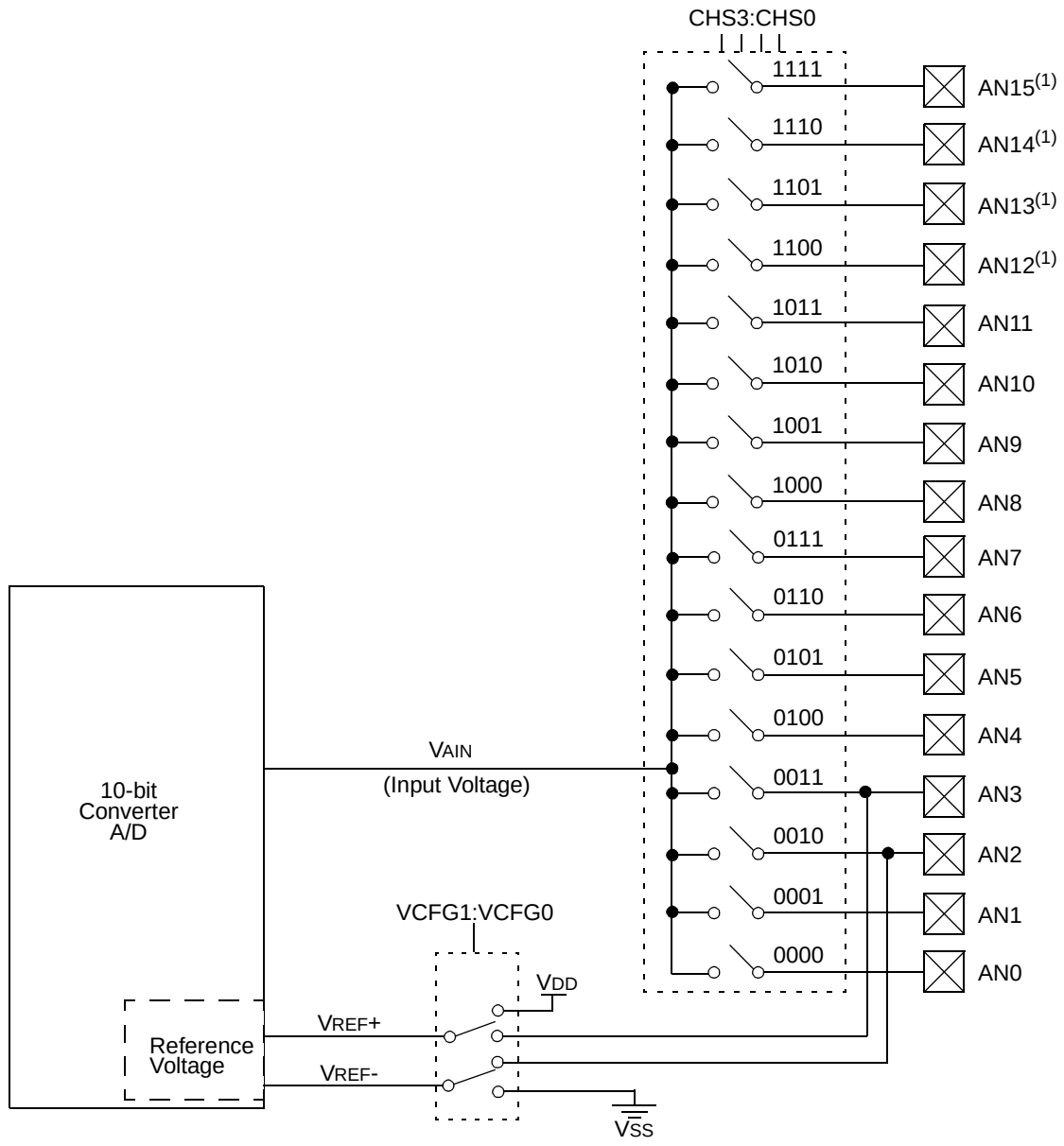
The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D’s internal RC oscillator.

The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

A device Reset forces all registers to their Reset state. This forces the A/D module to be turned off and any conversion is aborted.

Each port pin associated with the A/D converter can be configured as an analog input (RA3 can also be a voltage reference), or as a digital I/O. The ADRESH and ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRESH/ADRESL registers, the GO/$\overline{\text{DONE}}$ bit (ADCON0 register) is cleared and A/D interrupt flag bit, ADIF, is set. The block diagram of the A/D module is shown in Figure 19-1.

FIGURE 19-1: A/D BLOCK DIAGRAM



Note 1: Channels AN15 through AN12 are not available on PIC18F6X20 devices.

2: I/O pins have diode protection to VDD and VSS.

© 2004 Microchip Technology Inc.

The value in the ADRESH/ADRESL registers is not modified for a Power-on Reset. The ADRESH/ ADRESL registers will contain unknown data after a Power-on Reset.

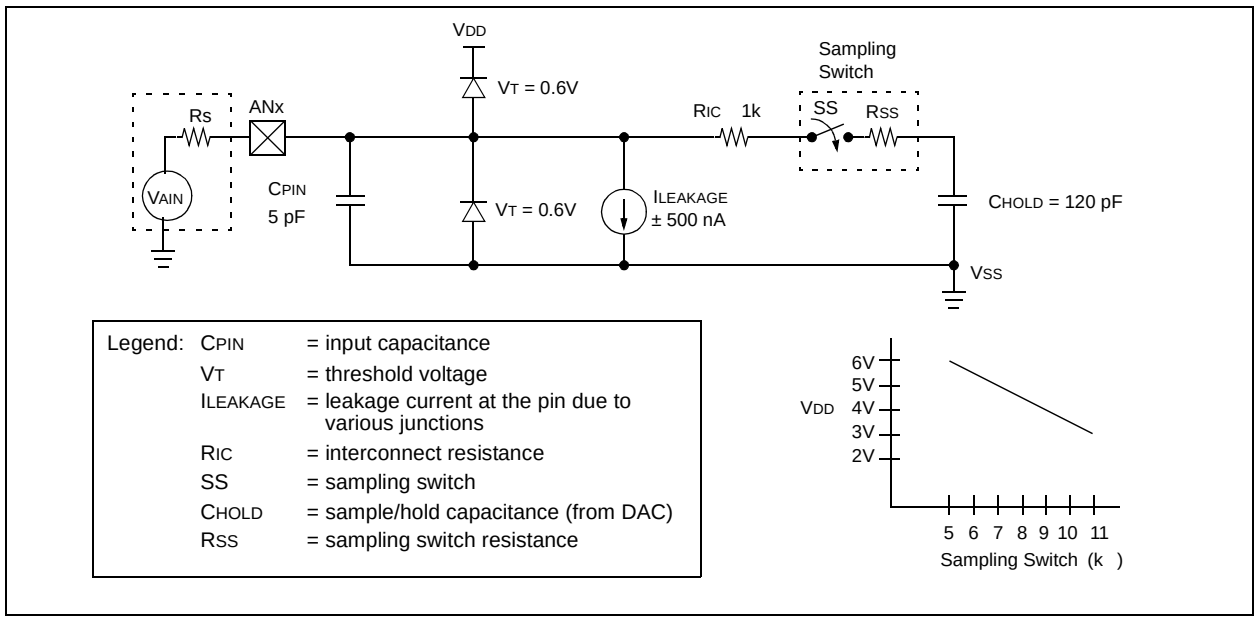
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as an input. To determine acquisition time, see Section 19.1 “A/D Acquisition Requirements” . After this acquisition time has elapsed, the A/D conversion can be started.

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 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.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0 register)
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 the next acquisition starts.

FIGURE 19-2: ANALOG INPUT MODEL



19.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 19-2. The source impedance (R_S) and the internal sampling switch (R_{SS}) impedance directly affect the time required to charge the capacitor C_{HOLD}. The sampling switch (R_{SS}) impedance varies over the device voltage (V_{DD}). 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 k . After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

Note:	When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 19-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 19-1 shows the calculation of the minimum required acquisition time, T_{ACQ}. This calculation is based on the following application system assumptions:

C_{HOLD}	=	120 pF
Rs	=	2.5 k
Conversion Error		1/2 LSb
V_{DD}	=	5V Rss = 7 k
Temperature	=	50°C (system max.)
V_{HOLD}	=	0V @ time = 0

Note:	When using external voltage references with the A/D converter, the source impedance of the external voltage references must be less than 20 to obtain the A/D performance specified in parameters A01-A06. Higher reference source impedances will increase both offset and gain errors. Resistive voltage dividers will not provide a sufficiently low source impedance. To maintain the best possible performance in A/D conversions, external V_{REF} inputs should be buffered with an operational amplifier or other low output impedance circuit. If deviating from the operating conditions specified for parameters A03-A06, the effect of parameter A50 (V_{REF} input current) must be considered.

EQUATION 19-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 19-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-Tc/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(120\ pF)(1\ k + R_{SS} + R_S)\ \ln(1/2047)$$

EXAMPLE 19-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

Temperature coefficient is only required for temperatures > 25°C.

$T_{ACQ} = 2\ \mu s + T_C + [(Temp - 25°C)(0.05\ \mu s/°C)]$

$T_C = -C_{HOLD}\ (R_{IC} + R_{SS} + R_S)\ \ln(1/2047)$
$-120\ pF\ (1\ k + 7\ k + 2.5\ k)\ \ln(0.0004885)$
$-120\ pF\ (10.5\ k)\ \ln(0.0004885)$
$-1.26\ \mu s\ (-7.6241)$
$9.61\ \mu s$

$T_{ACQ} = 2\ \mu s + 9.61\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C)]$
$11.61\ \mu s + 1.25\ \mu s$
$12.86\ \mu s$

© 2004 Microchip Technology Inc.

19.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 12 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 selected to ensure a minimum TAD time of 1.6 µs.

Table 19-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

19.3 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 CHS3:CHS0 bits and the TRIS bits.

Note 1:	When reading the port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert as an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.
2:	Analog levels on any pin defined as a digital input may cause the input buffer to consume current out of the device's specification limits.

TABLE 19-1: T AD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (T AD)		Maximum Device Frequency	
Operation	ADCS2:ADCS0	PIC18FXX20	PIC18LFXX20
2 TOSC	000	1.25 MHz	666 kHz
4 TOSC	100	2.50 MHz	1.33 MHz
8 TOSC	001	5.00 MHz	2.67 MHz
16 TOSC	101	10.0 MHz	5.33 MHz
32 TOSC	010	20.0 MHz	10.67 MHz
64 TOSC	110	40.0 MHz	21.33 MHz
RC	x11	—	—

19.4 A/D Conversions

Figure 19-3 shows the operation of the A/D converter after the GO bit has been set. Clearing the GO/DONE bit during a conversion will abort the current conversion. The A/D Result register pair will NOT be updated with the partially completed A/D conversion sample. That is, 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 aborted, a 2 TAD wait is required before the next acquisition is started. After this 2 TAD wait, acquisition on the selected channel is automatically started.

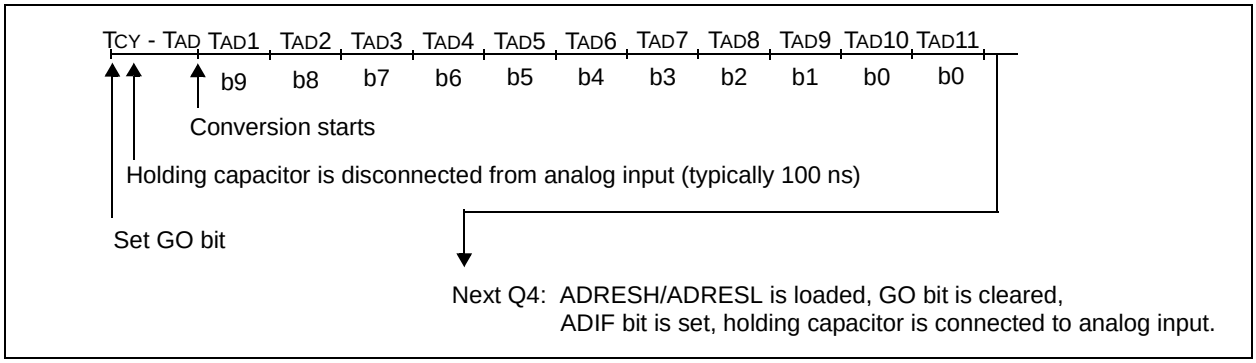
Note:	The GO/DONE bit should NOT be set in the same instruction that turns on the A/D.

19.5 Use of the CCP2 Trigger

An A/D conversion can be started by the "special event trigger" of the CCP2 module. This requires that the CCP2M3:CCP2M0 bits (CCP2CON<3:0>) be programmed as '1011' and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/DONE bit will be set, starting the A/D 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 done before the "special event trigger" sets the GO/DONE bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), the "special event trigger" will be ignored by the A/D module, but will still reset the Timer1 (or Timer3) counter.

FIGURE 19-3: A/D CONVERSION TAD CYCLES



© 2004 Microchip Technology Inc.

TABLE 19-2: SUMMARY OF A/D REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0111 1111	0111 1111
PIR2	—	CMIF	—	—	BCLIF	LVDIF	TMR3IF	CCP2IF	-0-- 0000	-0-- 0000
PIE2	—	CMIE	—	—	BCLIE	LVDIE	TMR3IE	CCP2IE	-0-- 0000	-0-- 0000
IPR2	—	CMIP	—	—	BCLIP	LVDIP	TMR3IP	CCP2IP	-0-- 0000	-0-- 0000
ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
ADRESL	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
ADCON0	—	—	CHS3	CHS3	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	--00 0000	--00 0000
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	--00 0000
ADCON2	ADFM	—	—	—	—	ADCS2	ADCS1	ADCS0	0--- -000	0--- -000
PORTA	—	RA6	RA5	RA4	RA3	RA2	RA1	RA0	--0x 0000	--0u 0000
TRISA	—	PORTA Data Direction Register							--11 1111	--11 1111
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	x000 0000	u000 0000
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	xxxx xxxx	uuuu uuuu
TRISF	PORTF Data Direction Control Register								1111 1111	1111 1111
PORTH(1)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	0000 xxxx	0000 xxxx
LATH(1)	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	xxxx xxxx	uuuu uuuu
TRISH(1)	PORTH Data Direction Control Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: Only available on PIC18F8X20 devices.

NOTES:

© 2004 Microchip Technology Inc.

20.0 COMPARATOR MODULE

The comparator module contains two analog comparators. The inputs to the comparators are multiplexed with the RF1 through RF6 pins. The on-chip voltage reference (Section 21.0 “Comparator Voltage Reference Module”) can also be an input to the comparators.

The CMCON register, shown as Register 20-1, controls the comparator input and output multiplexers. A block diagram of the various comparator configurations is shown in Figure 20-1.

REGISTER 20-1: CMCON REGISTER

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 C2OUT: Comparator 2 Output bit
When C2INV = 0:
1 = C2 V_{IN+} > C2 V_{IN-}
0 = C2 V_{IN+} < C2 V_{IN-}
When C2INV = 1:
1 = C2 V_{IN+} < C2 V_{IN-}
0 = C2 V_{IN+} > C2 V_{IN-}

bit 6 C1OUT: Comparator 1 Output bit
When C1INV = 0:
1 = C1 V_{IN+} > C1 V_{IN-}
0 = C1 V_{IN+} < C1 V_{IN-}
When C1INV = 1:
1 = C1 V_{IN+} < C1 V_{IN-}
0 = C1 V_{IN+} > C1 V_{IN-}

bit 5 C2INV: Comparator 2 Output Inversion bit
1 = C2 output inverted
0 = C2 output not inverted

bit 4 C1INV: Comparator 1 Output Inversion bit
1 = C1 output inverted
0 = C1 output not inverted

bit 3 CIS: Comparator Input Switch bit
When CM2:CM0 = 110:
1 = C1 V_{IN-} connects to RF5/AN10
C2 V_{IN-} connects to RF3/AN8
0 = C1 V_{IN-} connects to RF6/AN11
C2 V_{IN-} connects to RF4/AN9

bit 2-0 CM2:CM0: Comparator Mode bits
Figure 20-1 shows the Comparator modes and the CM2:CM0 bit settings.

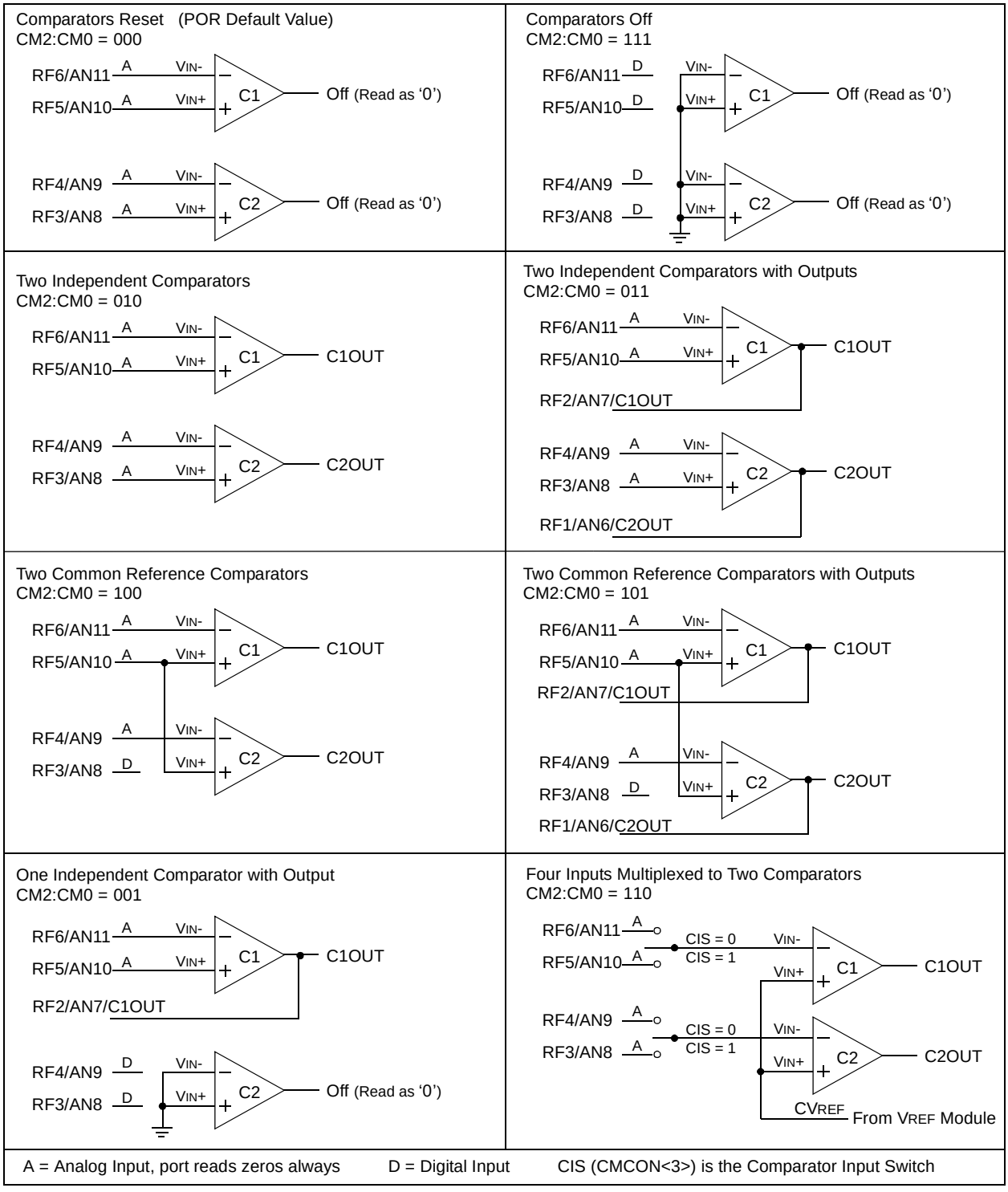
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

20.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 20-1 shows the eight possible 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 the Electrical Specifications (Section 26.0 “Electrical Characteristics”).

Note:	Comparator interrupts should be disabled during a Comparator mode change. Otherwise, a false interrupt may occur.

FIGURE 20-1: COMPARATOR I/O OPERATING MODES



© 2004 Microchip Technology Inc.