PIC18F6X2X/8X2X
Data Sheet

64/80-Pin High Performance, 64-Kbyte Enhanced FLASH Microcontrollers with A/D

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

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

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

Trademarks

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

PIC18F6X2X/8X2X

64/80-Pin High Performance, 64-Kbyte Enhanced FLASH Microcontrollers with A/D

High Performance RISC CPU:

- Linear program memory addressing to 64 Kbytes
- Linear data memory addressing to 4 Kbytes
- 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

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
- Two Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 6.25 ns (TCY/16)
  - Compare is 16-bit, max. resolution 100 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Three Enhanced Capture/Compare/PWM (ECCP) modules:
  - Same Capture/Compare features as CCP
  - One, two, or four PWM outputs
  - Selectable polarity
  - Programmable dead-time
  - Auto shutdown on external event
  - Auto Restart
- 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 Enhanced USART modules:
  - Supports RS-485, RS-232, and LIN 1.2
  - Auto wake-up on START bit
  - Auto baud detect
- Parallel Slave Port (PSP) module

External Memory Interface (PIC18F8X2X Devices Only):

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

Analog Features:

- 10-bit, up to 16-channel Analog-to-Digital Converter (A/D):
  - Auto acquisition
  - Conversion available during SLEEP
- Programmable 16-level Low Voltage Detection (LVD) module:
  - Supports interrupt on Low Voltage Detection
- Programmable Brown-out Reset (BOR)
- 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: > 100 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 2) via two pins

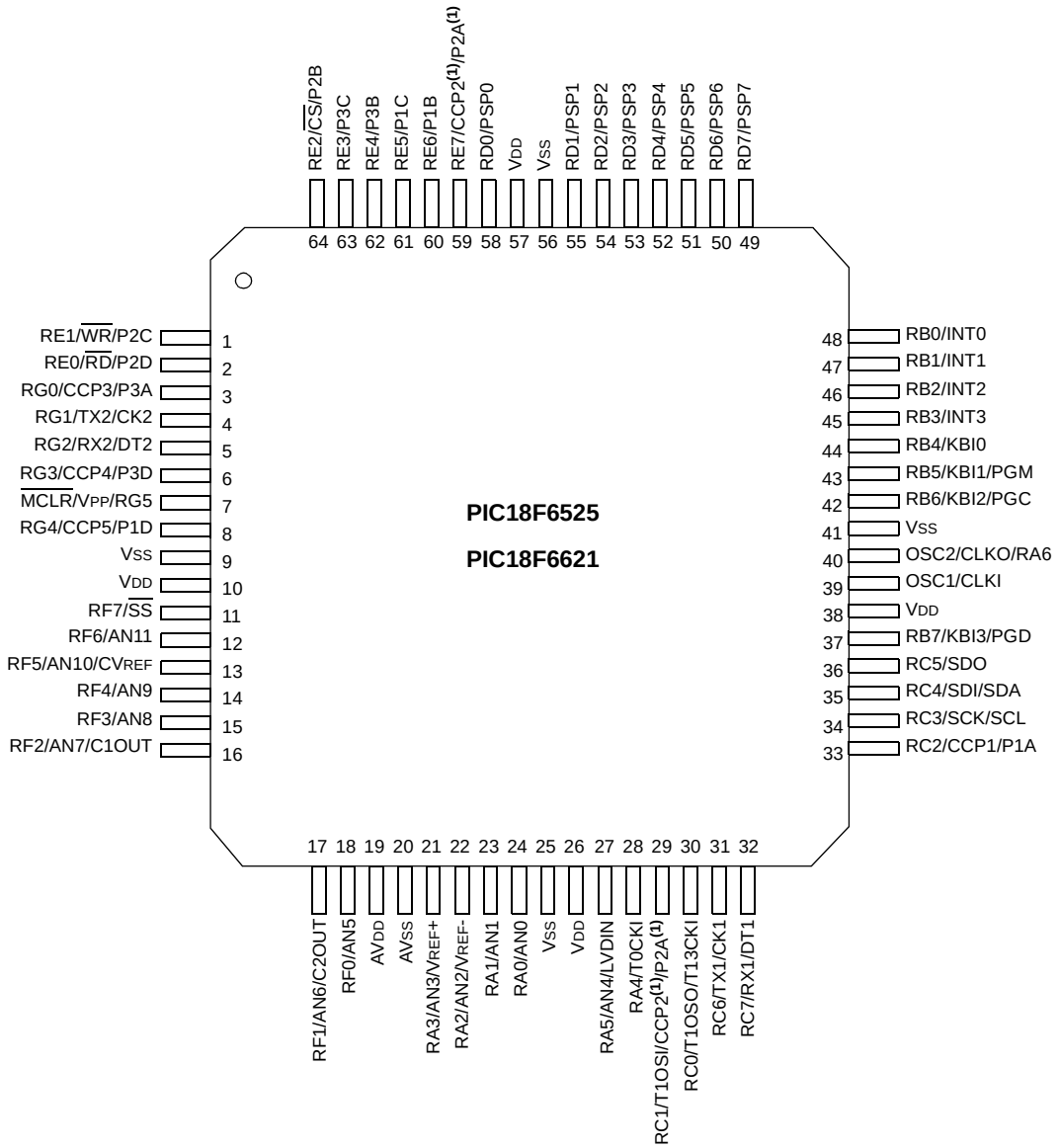
CMOS Technology:

- 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/ ECCP	PWM	MSSP/ SPI/ Master I^2C	EUSART	Timers 8-bit/16-bit	EMI
	Bytes	# Single Word Instructions	SRAM (bytes)	EEPROM (bytes)								
PIC18F6525	48K	24576	3840	1024	53	12	2/3	14	Y	2	2/3	N
PIC18F6621	64K	32768	3840	1024	53	12	2/3	14	Y	2	2/3	N
PIC18F8525	48K	24576	3840	1024	69	16	2/3	14	Y	2	2/3	Y
PIC18F8621	64K	32768	3840	1024	69	16	2/3	14	Y	2	2/3	Y

Pin Diagrams

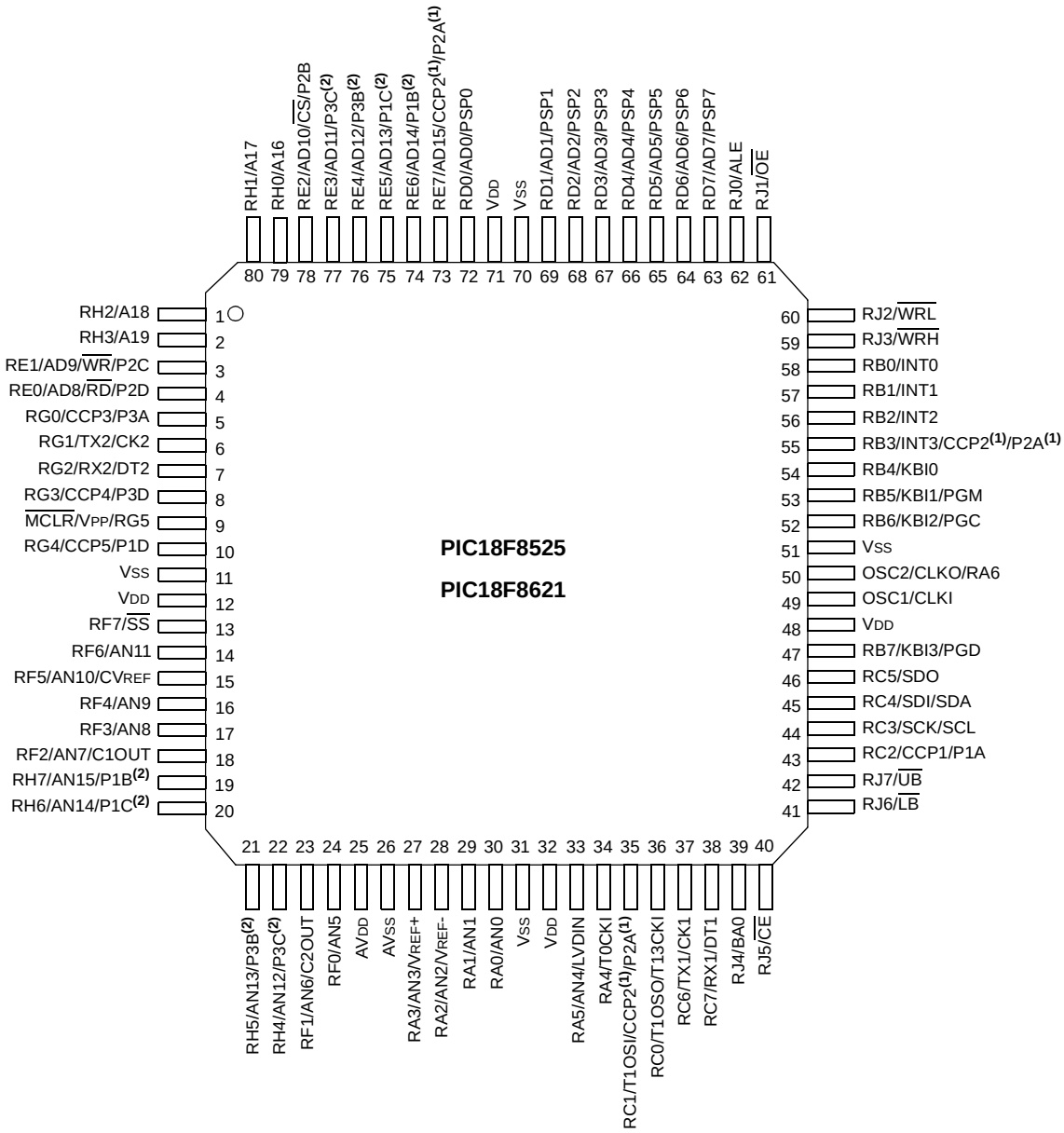
64-Pin TQFP



Note 1: CCP2/P2A are multiplexed with RC1 when CCP2MX is set, or RE7 when CCP2MX is not set.

Pin Diagrams (Cont.'d)

80-Pin TQFP



Note 1: CCP2/P2A are multiplexed with RC1 when CCP2MX is set; with RE7 when CCP2MX is cleared and the device is configured in Microcontroller mode; or with RB3 when CCP2MX is cleared in all other Program Memory modes.

2: P1B/P1C/P3B/P3C are multiplexed with RE6:RE3 when ECCPMX is set, and with RH7:RH4 when ECCPMX is not set.

Table of Contents

TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@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:

1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC18F6525
- PIC18F6621
- PIC18F8525
- PIC18F8621

This family offers the advantages of all PIC18 microcontrollers - namely, high computational performance at an economical price - with the addition of high endurance enhanced FLASH program memory. The PIC18F6X2X/8X2X 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 PIC18F6X2X/8X2X family provides ample room for application code, and includes members with 48 Kbytes or 64 Kbytes of code space.

Other memory features are:

- **Data RAM and Data EEPROM:** The PIC18F6X2X/8X2X family also provides plenty of room for application data. The devices have 3840 bytes of data RAM as well as 1024 bytes of data EEPROM for long term retention of non-volatile 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 unlikely event that 64 Kbytes of program memory is inadequate for an application, the PIC18F8X2X 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 PIC18F6X2X/8X2X family incorporates a range of serial communication peripherals, including 2 independent enhanced USARTs and a Master SSP module capable of both SPI and I^2C (Master and Slave) modes of operation. Also, for PIC18F6X2X/8X2X 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 two Capture/Compare/PWM (CCP) modules and three enhanced CCP modules to maximize flexibility in control applications. Up to four different time bases may be used to perform several different operations at once. Each of the three ECCPs offer up to four PWM outputs, allowing for a total of 12 PWMs. The ECCPs also offer many beneficial features, including polarity selection, programmable dead-time, auto shutdown and restart, and Half-Bridge and Full-Bridge Output modes.
- **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 and auto-acquisition conversions. 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 PIC18F6X2X/8X2X devices are available in 64-pin and 80-pin packages. They are differentiated from each other in four ways:

1. FLASH program memory (48 Kbytes for PIC18FX525 devices and 64 Kbytes for PIC18FX621 devices)
2. A/D channels (12 for PIC18F6X2X devices, 16 for PIC18F8X2X)
3. I/O ports (7 on PIC18F6X2X devices, 9 on PIC18F8X2X)
4. External program memory interface (present only on PIC18F8X2X devices)

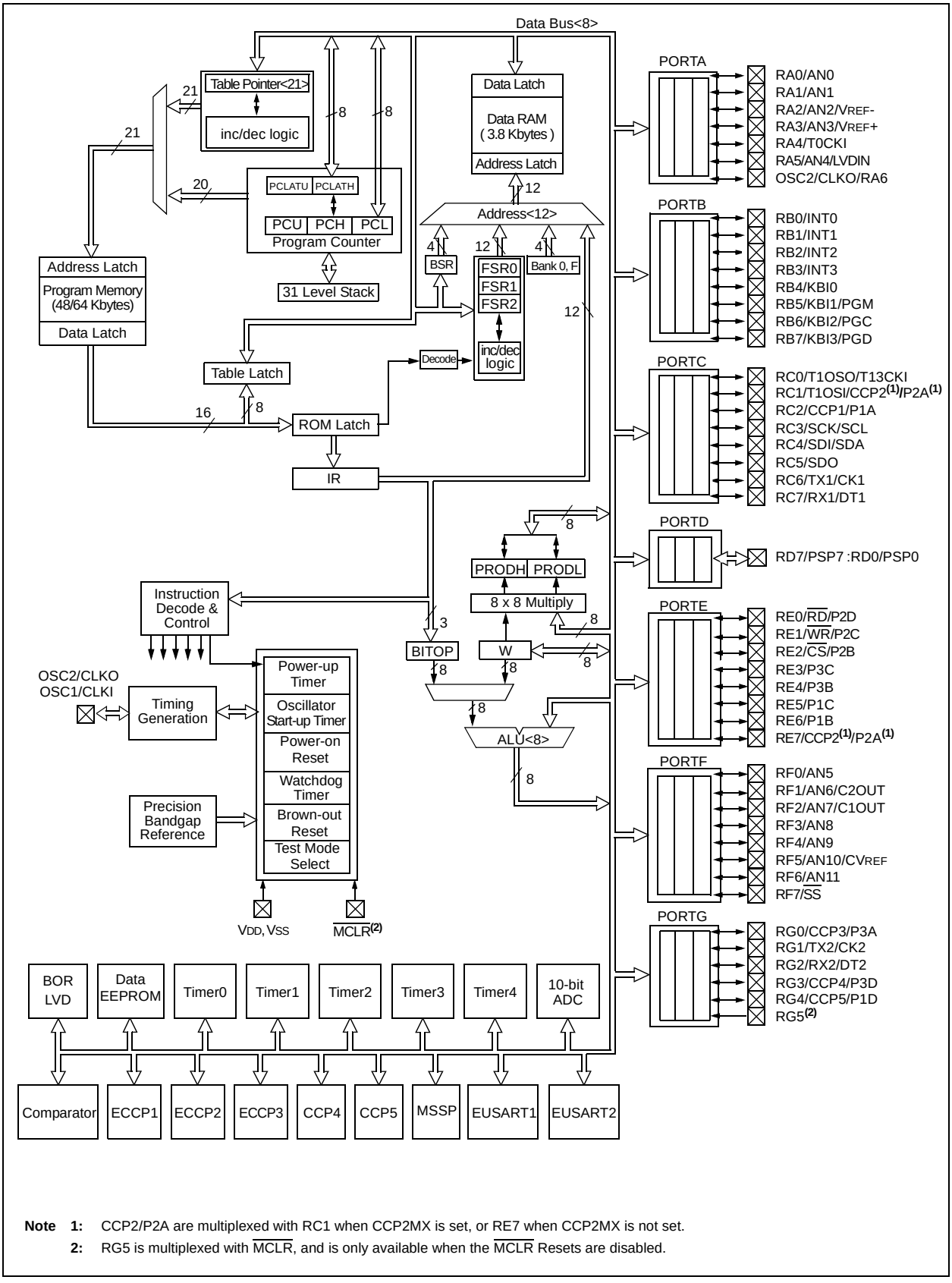
All other features for devices in the PIC18F6X2X/8X2X family are identical. These are summarized in Table 1-1.

Block diagrams of the PIC18F6X2X and PIC18F8X2X 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: PIC18F6X2X/8X2X DEVICE FEATURES

Features	PIC18F6525	PIC18F6621	PIC18F8525	PIC18F8621
Operating Frequency	DC - 40 MHz	DC - 40 MHz	DC - 40 MHz	DC - 40 MHz
Program Memory (Bytes)	48K	64K	48K	64K
Program Memory (Instructions)	24576	32768	24576	32768
Data Memory (Bytes)	3840	3840	3840	3840
Data EEPROM Memory (Bytes)	1024	1024	1024	1024
External Memory Interface	No	No	Yes	Yes
Interrupt Sources	17	17	17	17
I/O Ports	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
Timers	5	5	5	5
Capture/Compare/PWM Modules	2	2	2	2
Enhanced Capture/Compare/ PWM Module	3	3	3	3
Serial Communications	MSSP, Addressable EUSART (2)	MSSP, Addressable EUSART (2)	MSSP, Addressable EUSART (2)	MSSP, Addressable EUSART (2)
Parallel Communications	PSP	PSP	PSP	PSP
10-bit Analog-to-Digital Module	12 input channels	12 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)
Programmable Low Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	77 Instructions	77 Instructions	77 Instructions	77 Instructions
Package	64-pin TQFP	64-pin TQFP	80-pin TQFP	80-pin TQFP

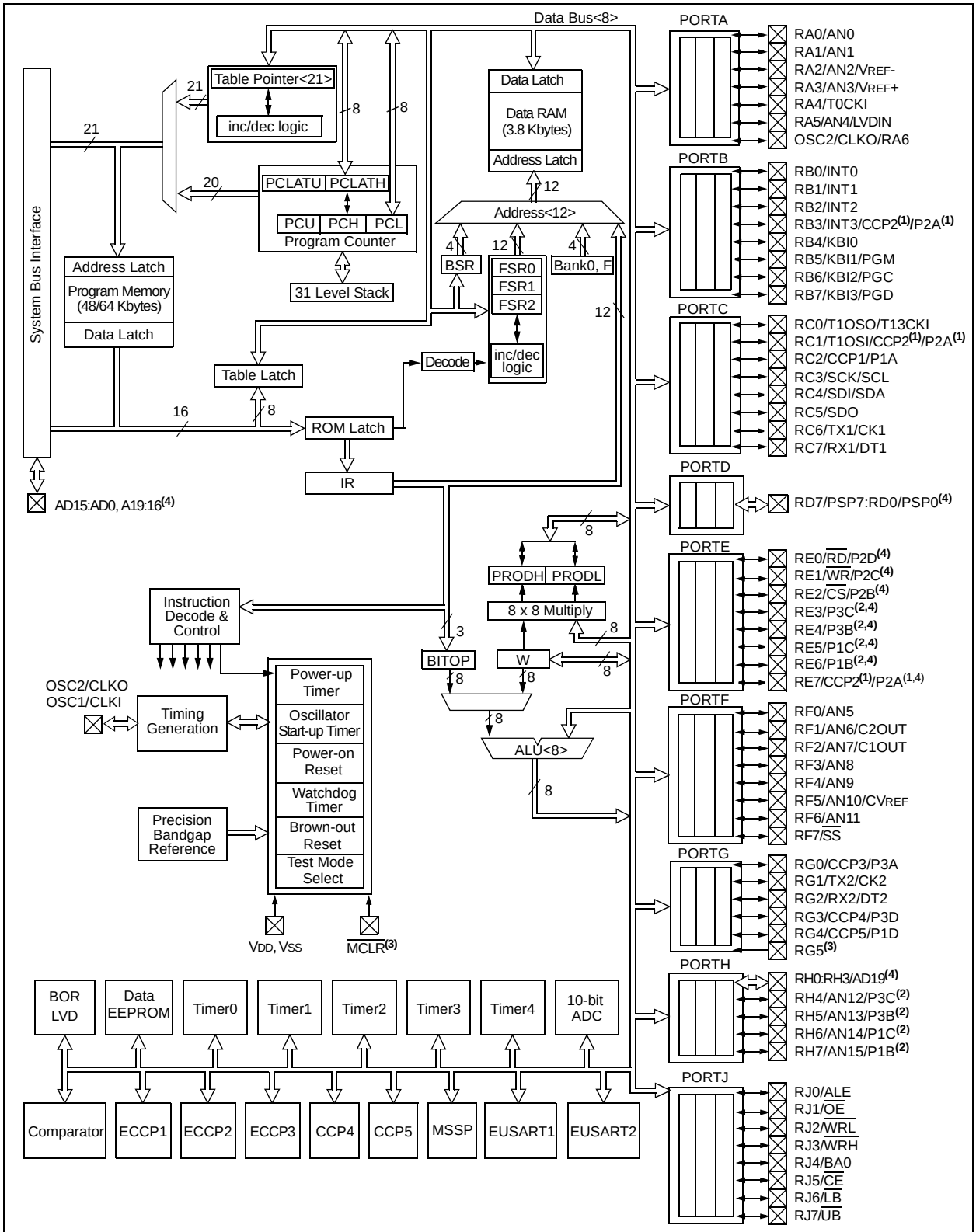
FIGURE 1-1: PIC18F6525/6621 BLOCK DIAGRAM



Note 1: CCP2/P2A are multiplexed with RC1 when CCP2MX is set, or RE7 when CCP2MX is not set.

2: RG5 is multiplexed with MCLR, and is only available when the MCLR Resets are disabled.

FIGURE 1-2: PIC18F8525/8621 BLOCK DIAGRAM



Note 1: CCP2/P2A are multiplexed with RC1 when CCP2MX is set; with RE7 when CCP2MX is cleared and the device is configured in Microcontroller mode; or with RB3 when CCP2MX is cleared in all other Program Memory modes.

2: P1B/P1C/P3B/P3C are multiplexed with RE6:RE3 when ECCPMX is set, and with RH7:RH4 when ECCPMX is not set.

3: RG5 is multiplexed with MCLR, and is only available when the MCLR Resets are disabled.

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

TABLE 1-2: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
$\overline{MCLR}$/VPP/RG5	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.
RG5			I	ST	Digital 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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTA is a bi-directional 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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTB is a bi-directional 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/P2A	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.
P2A(1)			O	—	Enhanced CCP2 output P2A.
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	ST	Interrupt-on-change pin.
PGD			I/O	ST	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 VDD)

Note 1: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTC is a bi-directional 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/P2A	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.
P2A(2)			O	—	Enhanced CCP2 output P2A.
RC2/CCP1/P1A	33	43			
RC2			I/O	ST	Digital I/O.
CCP1			I/O	ST	Capture1 input, Compare1 output, PWM1 output.
P1A			O	—	Enhanced CCP1 output P1A.
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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTD is a bi-directional I/O port. These pins have TTL input buffers when external memory is enabled.
RD0/AD0/PSP0	58	72			
RD0			I/O	ST	Digital I/O.
AD0(3)			I/O	TTL	External memory address/data 0.
PSP0			I/O	TTL	Parallel slave port data.
RD1/AD1/PSP1	55	69			
RD1			I/O	ST	Digital I/O.
AD1(3)			I/O	TTL	External memory address/data 1.
PSP1			I/O	TTL	Parallel slave port data.
RD2/AD2/PSP2	54	68			
RD2			I/O	ST	Digital I/O.
AD2(3)			I/O	TTL	External memory address/data 2.
PSP2			I/O	TTL	Parallel slave port data.
RD3/AD3/PSP3	53	67			
RD3			I/O	ST	Digital I/O.
AD3(3)			I/O	TTL	External memory address/data 3.
PSP3			I/O	TTL	Parallel slave port data.
RD4/AD4/PSP4	52	66			
RD4			I/O	ST	Digital I/O.
AD4(3)			I/O	TTL	External memory address/data 4.
PSP4			I/O	TTL	Parallel slave port data.
RD5/AD5/PSP5	51	65			
RD5			I/O	ST	Digital I/O.
AD5(3)			I/O	TTL	External memory address/data 5.
PSP5			I/O	TTL	Parallel Slave Port data.
RD6/AD6/PSP6	50	64			
RD6			I/O	ST	Digital I/O.
AD6(3)			I/O	TTL	External memory address/data 6.
PSP6			I/O	TTL	Parallel slave port data.
RD7/AD7/PSP7	49	63			
RD7			I/O	ST	Digital I/O.
AD7(3)			I/O	TTL	External memory address/data 7.
PSP7			I/O	TTL	Parallel slave port data.

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

Note 1: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTE is a bi-directional I/O port.
RE0/AD8/$\overline{RD}$/P2D	2	4			
RE0			I/O	ST	Digital I/O.
AD8(3)			I/O	TTL	External memory address/data 8.
$\overline{RD}$			I	TTL	Read control for parallel slave port.
P2D			O	—	Enhanced CCP2 output P2D.
RE1/AD9/$\overline{WR}$/P2C	1	3			
RE1			I/O	ST	Digital I/O.
AD9(3)			I/O	TTL	External memory address/data 9.
$\overline{WR}$			I	TTL	Write control for parallel slave port.
P2C			O	ST	Enhanced CCP2 output P2C.
RE2/AD10/$\overline{CS}$/P2B	64	78			
RE2			I/O	ST	Digital I/O.
AD10(3)			I/O	TTL	External memory address/data 10.
$\overline{CS}$			I	TTL	Chip select control for parallel slave port.
P2B			O	—	Enhanced CCP2 output P2B.
RE3/AD11/P3C	63	77			
RE3			I/O	ST	Digital I/O.
AD11(3)			I/O	TTL	External memory address/data 11.
P3C(4)			O	—	Enhanced CCP3 output P3C.
RE4/AD12/P3B	62	76			
RE4			I/O	ST	Digital I/O.
AD12(3)			I/O	TTL	External memory address/data 12.
P3B(4)			O	—	Enhanced CCP3 output P3B.
RE5/AD13/P1C	61	75			
RE5			I/O	ST	Digital I/O.
AD13(3)			I/O	TTL	External memory address/data 13.
P1C(4)			O	—	Enhanced CCP1 output P1C.
RE6/AD14/P1B	60	74			
RE6			I/O	ST	Digital I/O.
AD14(3)			I/O	TTL	External memory address/data 14.
P1B(4)			O	—	Enhanced CCP1 output P1B.
RE7/AD15/CCP2/P2A	59	73			
RE7			I/O	ST	Digital I/O.
AD15(3)			I/O	TTL	External memory address/data 15.
CCP2(5)			I/O	ST	Capture2 input, Compare2 output, PWM2 output.
P2A(5)			O	—	Enhanced CCP2 output P2A.

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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTF is a bi-directional 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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTG is a bi-directional I/O port.
RG0/CCP3/P3A	3	5			
RG0			I/O	ST	Digital I/O.
CCP3			I/O	ST	Capture3 input, Compare3 output, PWM3 output.
P3A			O	—	Enhanced CCP3 output P3A.
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/P3D	6	8			
RG3			I/O	ST	Digital I/O.
CCP4			I/O	ST	Capture4 input, Compare4 output, PWM4 output.
P3D			O	—	Enhanced CCP3 output P3D.
RG4/CCP5/P1D	8	10			
RG4			I/O	ST	Digital I/O.
CCP5			I/O	ST	Capture5 input, Compare5 output, PWM5 output.
P1D			O	—	Enhanced CCP1 output P1D.
RG5	7	9	—	—	See $\overline{MCLR}$/VPP/RG5 pin.

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

Note 1: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTH is a bi-directional I/O port[6].
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/P3C	—	22			
RH4			I/O	ST	Digital I/O.
AN12			I	Analog	Analog input 12.
P3C[7]			O	—	Enhanced CCP3 output P3C.
RH5/AN13/P3B	—	21			
RH5			I/O	ST	Digital I/O.
AN13			I	Analog	Analog input 13.
P3B[7]			O	—	Enhanced CCP3 output P3B.
RH6/AN14/P1C	—	20			
RH6			I/O	ST	Digital I/O.
AN14			I	Analog	Analog input 14.
P1C[7]			O	—	Enhanced CCP1 output P1C.
RH7/AN15/P1B	—	19			
RH7			I/O	ST	Digital I/O.
AN15			I	Analog	Analog input 15.
P1B[7]			O	—	Enhanced CCP1 output P1B.

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/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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: PIC18F6X2X/8X2X PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X2X	PIC18F8X2X			
					PORTJ is a bi-directional I/O port[6].
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	System bus byte address 0 control.
RJ5/$\overline{CE}$	—	40			
RJ5			I/O	ST	Digital I/O
$\overline{CE}$			O	TTL	External memory access indicator.
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.
V_{SS}	9, 25, 41, 56	11, 31, 51, 70	P	—	Ground reference for logic and I/O pins.
V_{DD}	10, 26, 38, 57	12, 32, 48, 71	P	—	Positive supply for logic and I/O pins.
AV_{SS}[8]	20	26	P	—	Ground reference for analog modules.
AV_{DD}[8]	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 V_{DD})

Note 1: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX (CONFIG3H<0>) is not set (all Program Memory modes except Microcontroller).

2: Default assignment for CCP2/P2A when CCP2MX is set (all devices).

3: External memory interface functions are only available on PIC18F8X2X devices.

4: Default assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is set, and for all PIC18F6X2X devices.

5: Alternate assignment for CCP2/P2A in PIC18F8X2X devices when CCP2MX is not set (Microcontroller mode).

6: PORTH and PORTJ (and their multiplexed functions) are only available on PIC18F8X2X devices.

7: Alternate assignment for P1B/P1C/P3B/P3C for PIC18F8X2X devices when ECCPMX (CONFIG3H<1>) is not set.

8: 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.

2.0 OSCILLATOR CONFIGURATIONS

2.1 Oscillator Types

The PIC18F6X2X/8X2X devices can be operated in twelve different Oscillator modes. The user can program four configuration bits (FOSC3, 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. RC External Resistor/Capacitor
5. EC External Clock
6. ECIO External Clock with I/O pin enabled
7. HS+PLL High Speed Crystal/Resonator with PLL enabled
8. RCIO External Resistor/Capacitor with I/O pin enabled
9. ECIO+SPLL External Clock with software controlled PLL
10. ECIO+PLL External Clock with PLL and I/O pin enabled
11. HS+SPLL High Speed Crystal/Resonator with software control
12. RCIO External Resistor/Capacitor with I/O pin enabled

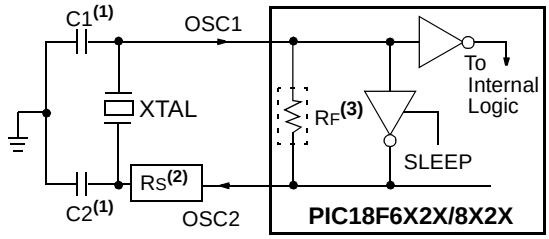
2.2 Crystal Oscillator/Ceramic Resonators

In XT, LP, HS, HS+PLL or HS+SPLL 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 PIC18F6X2X/8X2X 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 manufacturers 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	68 - 100 pF	68 - 100 pF
	2.0 MHz	15 - 68 pF	15 - 68 pF
	4.0 MHz	15 - 68 pF	15 - 68 pF
HS	8.0 MHz	10 - 68 pF	10 - 68 pF
	16.0 MHz	10 - 22 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	32.0 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1.0 MHz	15 pF	15 pF
	4.0 MHz	15 pF	15 pF
HS	4.0 MHz	15 pF	15 pF
	8.0 MHz	15-33 pF	15-33 pF
	20.0 MHz	15-33 pF	15-33 pF
	25.0 MHz	TBD	TBD
These values are for design guidance only. See notes following this table.			
Crystals Used			
32.0 kHz	Epson C-001R32.768K-A		± 20 PPM
200 kHz	STD XTL 200.000KHz		± 20 PPM
1.0 MHz	ECS ECS-10-13-1		± 50 PPM
4.0 MHz	ECS ECS-40-20-1		± 50 PPM
8.0 MHz	Epson CA-301 8.000M-C		± 30 PPM
20.0 MHz	Epson CA-301 20.000M-C		± 30 PPM

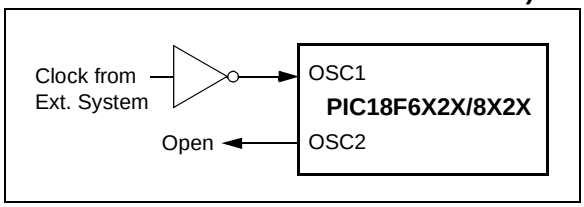
Note 1: Higher capacitance increases the stability of the oscillator but also increases the start-up time.

2: Rs (see Figure 2-1) may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification.

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.

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)

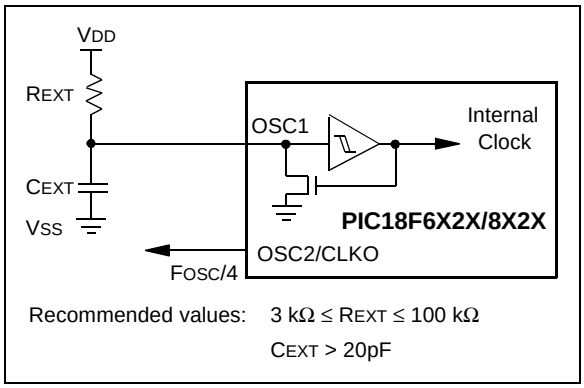


2.3 RC Oscillator

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



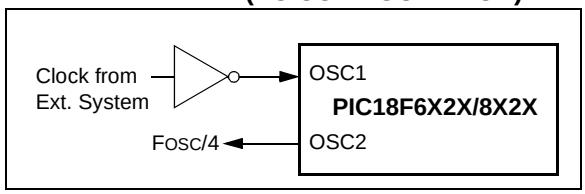
The RCIO Oscillator mode functions like the RC mode except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6).

2.4 External Clock Input

The EC, ECIO, EC+PLL and EC+SPLL 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.

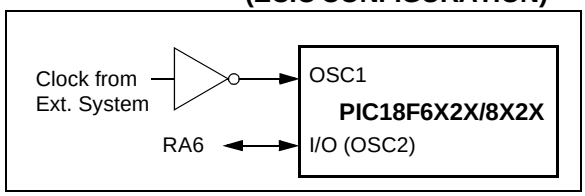
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-4 shows the pin connections for the EC Oscillator mode.

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 Phase Locked Loop (PLL)

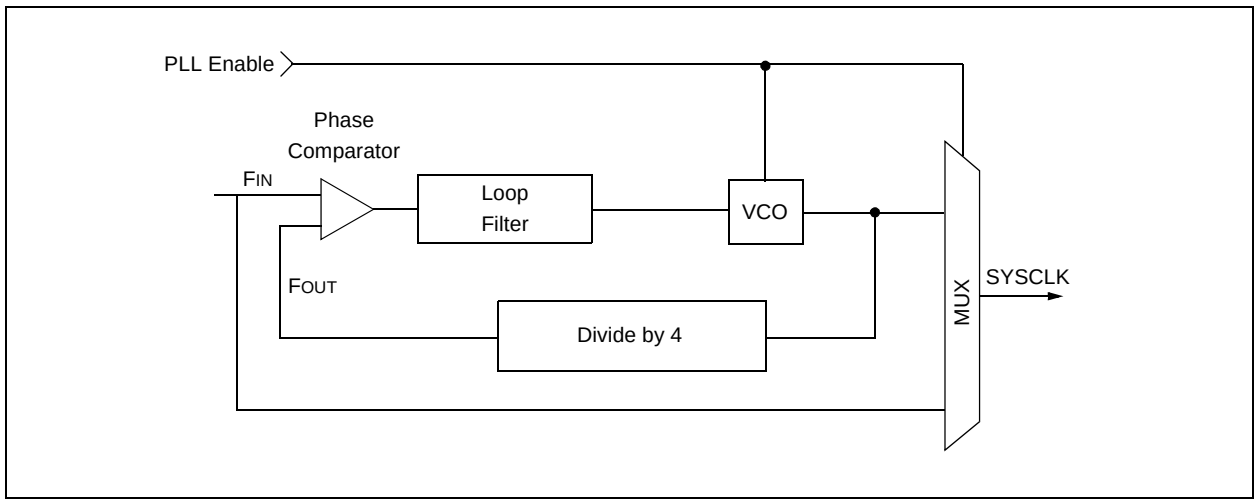
A phase locked loop circuit is provided as a programmable option for users that want to multiply the frequency of the incoming oscillator signal by 4. For an input clock frequency of 10 MHz, the internal clock frequency will be multiplied to 40 MHz. This is useful for customers who are concerned with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

The PLL can only be enabled when the oscillator configuration bits are programmed for High Speed Oscillator or External Clock mode. If they are programmed for any other mode, the PLL is not enabled and the system clock will come directly from OSC1. There are two types of PLL modes - Software Controlled PLL and Configuration bits Controlled PLL. In Software Controlled PLL mode, PIC18F6X2X/8X2X executes at regular clock frequency after all RESET conditions. During execution, the application can enable PLL and switch to 4x clock frequency operation by setting the PLLEN bit in the OSCCON register. In Configuration bits Controlled PLL, the 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 for details on oscillator switching).

The type of PLL is selected by programming FOSC<3:0> configuration bits in CONFIG1H Configuration register. The Oscillator mode is specified during device programming.

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.

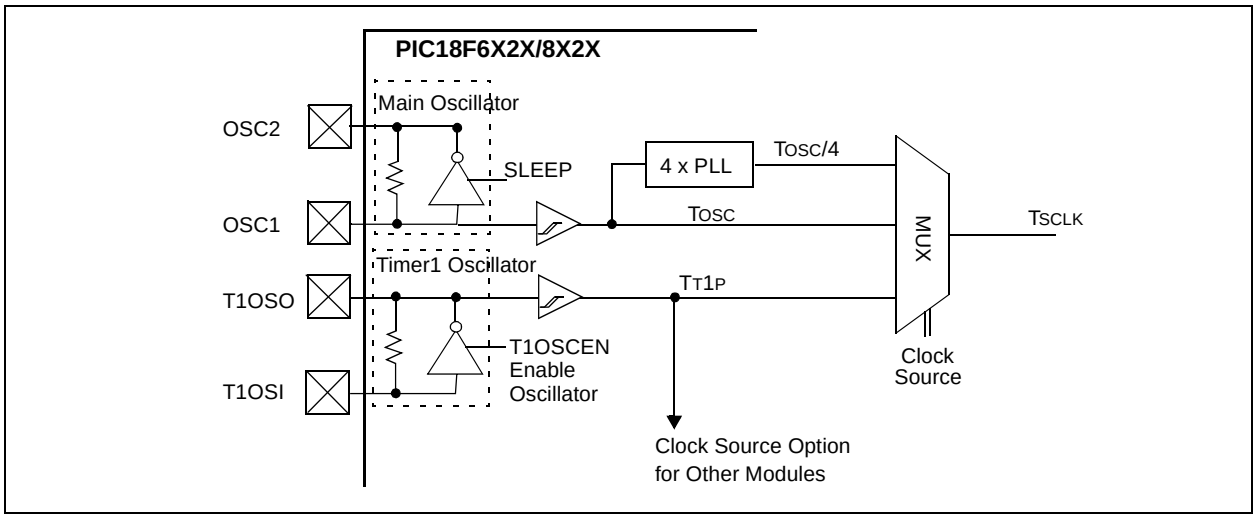
FIGURE 2-6: PLL BLOCK DIAGRAM



2.6 Oscillator Switching Feature

The PIC18F6X2X/8X2X 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 PIC18F6X2X/8X2X 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 the CONFIG1H Configuration register to a '0'. Clock switching is disabled in an erased device. See Section 12.0 for further details of the Timer1 oscillator. See Section 24.0 for Configuration register details.

FIGURE 2-7: DEVICE CLOCK SOURCES



2.6.1 SYSTEM CLOCK SWITCH BIT

The system clock source switching is performed under software control. The system clock switch bits, SCS1:SCS0 (OSCCON<1:0>), control the clock switching. When the SCS0 bit is '0', the system clock source comes from the main oscillator that is selected by the FOSC configuration bits in CONFIG1H Configuration register. When the SCS0 bit is set, the system clock source will come from the Timer1 oscillator. The SCS0 bit is cleared on all forms of RESET.

When FOSC bits are programmed for Software PLL mode, SCS1 bit can be used to select between primary oscillator/clock and PLL output. SCS1 bit will only have an effect on the system clock if the PLL is enabled (PLLEN = 1) and locked (LOCK = 1), else it will be forced cleared. When programmed with Configuration Controlled PLL, SCS1 bit will be forced clear.

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 SCS0 bit will be ignored (SCS0 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	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	LOCK	PLLEN	SCS1	SCS0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **LOCK:** Phase Lock Loop Lock Status bit
1 = Phase lock loop output is stable as system clock
0 = Phase lock loop output is not stable and output cannot be used as system clock

bit 2 **PLLEN(1):** Phase Lock Loop Enable bit
1 = Enable phase lock loop output as system clock
0 = Disable phase lock loop

bit 1 **SCS1:** System Clock Switch bit 1
When PLLEN and LOCK bits are set:
1 = Use PLL output
0 = Use primary oscillator/clock input pin
When PLLEN or LOCK bit is cleared:
Bit is forced clear.

bit 0 **SCS0(2):** System Clock Switch bit 0
When $\overline{\text{OSCSEN}}$ configuration bit = 0 and T1OSCEN bit = 1:
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.

Note 1: PLLEN bit is forced set when configured for ECIO+PLL and HS+PLL. This bit is writable for ECIO+SPLL and HS+SPLL modes only; forced cleared for all other Oscillator modes.

2: The setting of SCS0 = 1 supersedes SCS1 = 1.

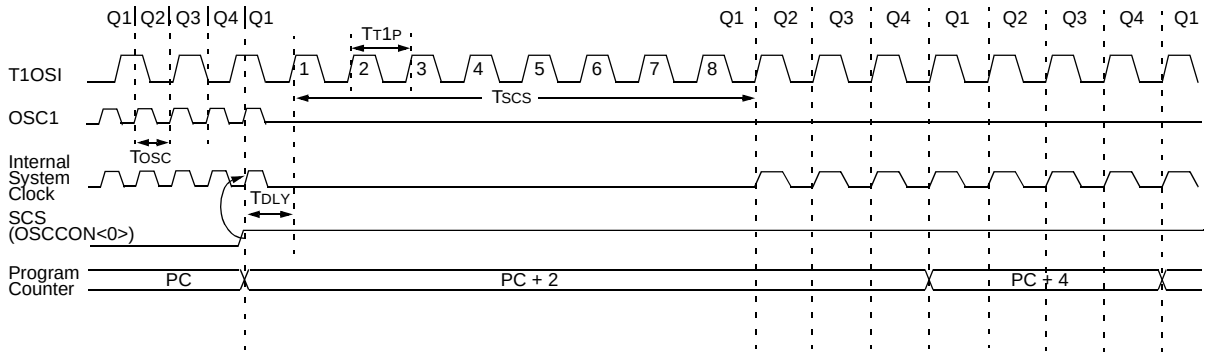
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.6.2 OSCILLATOR TRANSITIONS

PIC18F6X2X/8X2X 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 SCS0 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 TRANSITION FROM OSC1 TO TIMER1 OSCILLATOR

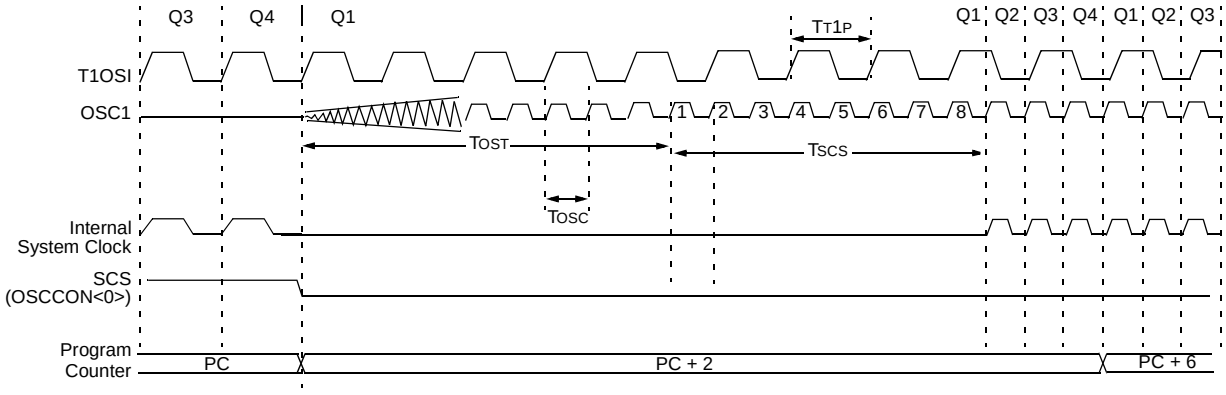


Note: TDLY is delay from SCS high to first count of transition circuit.

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.

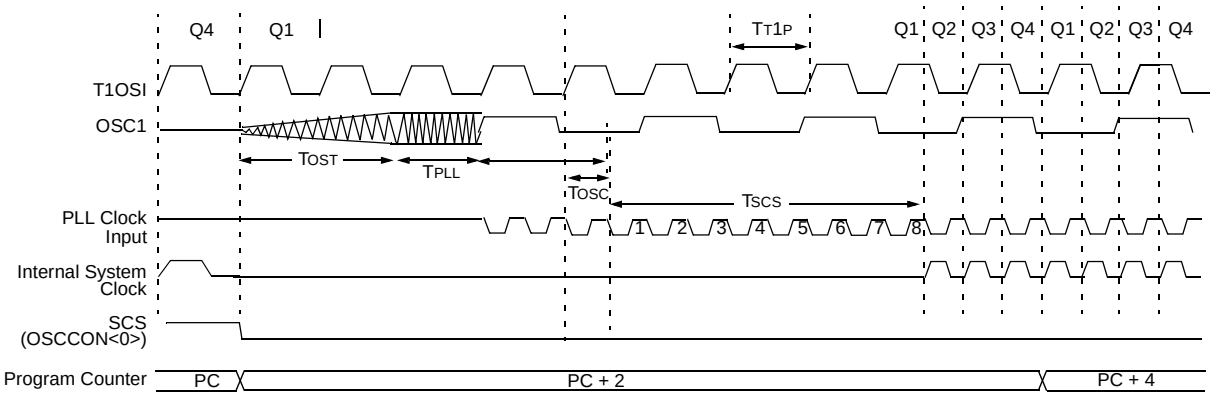
FIGURE 2-9: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (HS, XT, LP)



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

If the main oscillator is configured for HS mode with PLL active, 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.

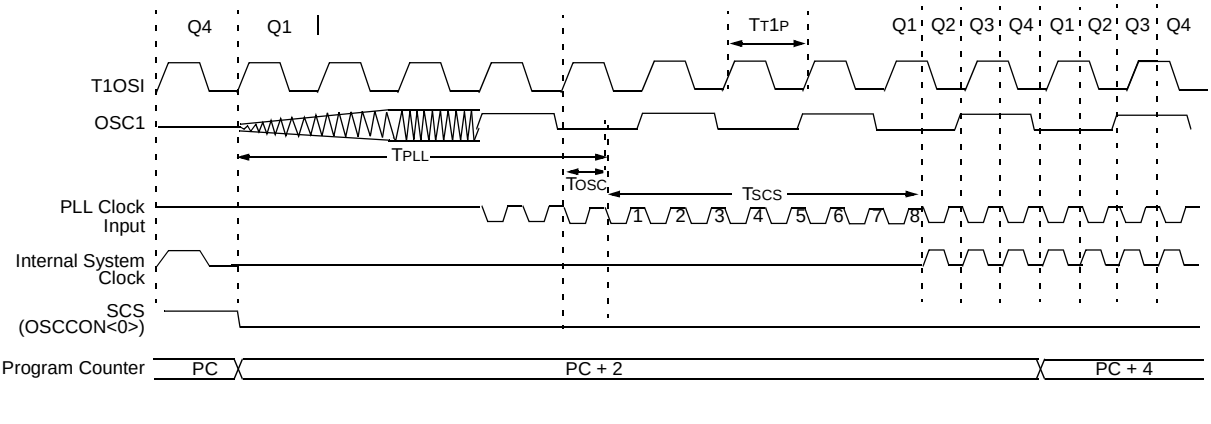
FIGURE 2-10: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (HS WITH PLL ACTIVE, SCS1 = 1)



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

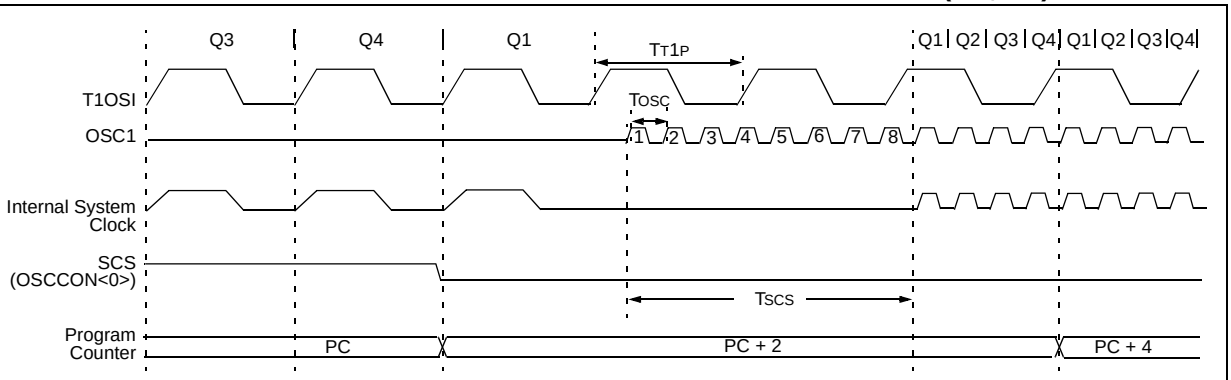
If the main oscillator is configured for EC mode with PLL active, only PLL time-out (TPLL) will occur. The PLL time-out is typically 2 ms and allows the PLL to lock to the main oscillator frequency. A timing diagram, indicating the transition from the Timer1 oscillator to the main oscillator for EC with PLL active, is shown in Figure 2-11.

FIGURE 2-11: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (EC WITH PLL ACTIVE, SCS1 = 1)



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 2-12.

FIGURE 2-12: TIMING FOR TRANSITION BETWEEN TIMER1 AND OSC1 (RC, EC)



Note: RC Oscillator mode assumed.

2.7 Effects of SLEEP Mode on the On-Chip Oscillator

When the device executes a `SLEEP` instruction, 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.

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.

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 and EC+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.

NOTES:

3.0 RESET

The PIC18F6X2X/8X2X 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 (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

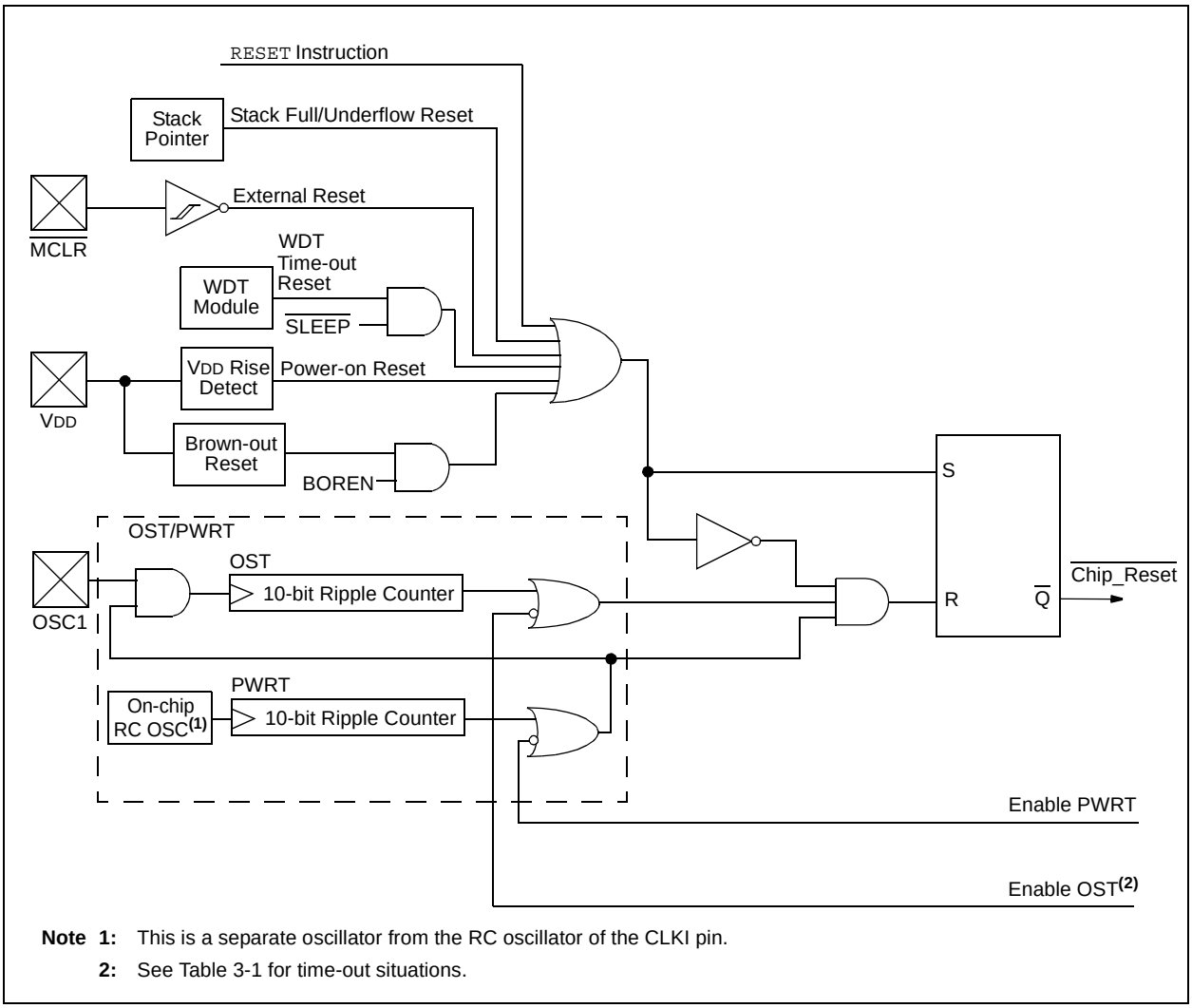
Most registers are unaffected by a RESET. Their status is unknown on POR and unchanged by all other RESETS. The other registers are forced to a "RESET state" on Power-on Reset, MCLR, WDT Reset, Brown-out Reset, MCLR Reset during SLEEP and by the RESET instruction.

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



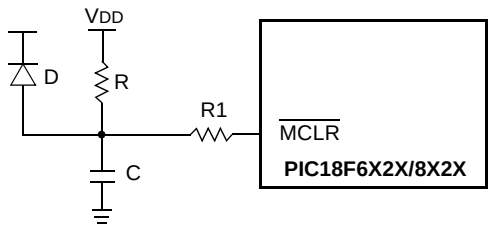
Note 1: This is a separate oscillator from the RC oscillator of the CLKI pin.
2: See Table 3-1 for time-out situations.

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 kΩ to 10 kΩ 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 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 = 1 kΩ to 10 kΩ 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 cycle (from OSC1 input) delays 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. Figure 3-3, Figure 3-4, Figure 3-5, Figure 3-6 and Figure 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 PIC18F6X2X/8X2X 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.

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 4x PLL 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[1]

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

Note 1: Refer to Section 4.14 (page 61) for bit definitions.

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

Condition	Program Counter	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
Power-on Reset	0000h	1	1	1	0	0	u	u
$\overline{MCLR}$ Reset during normal operation	0000h	u	u	u	u	u	u	u
Software Reset during normal operation	0000h	0	u	u	u	u	u	u
Stack Full Reset during normal operation	0000h	u	u	u	u	u	u	1
Stack Underflow Reset during normal operation	0000h	u	u	u	u	u	1	u
$\overline{MCLR}$ Reset during SLEEP	0000h	u	1	0	u	u	u	u
WDT Reset	0000h	1	0	1	u	u	u	u
WDT Wake-up	PC + 2	u	0	0	u	u	u	u
Brown-out Reset	0000h	1	1	1	1	0	u	u
Interrupt wake-up from SLEEP	PC + 2[1]	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 (0008h or 0018h).

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TOSU	PIC18F6X2X	PIC18F8X2X	---0 0000	---0 0000	---0 uuuu(3)
TOSH	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	PIC18F6X2X	PIC18F8X2X	00-0 0000	uu-0 0000	uu-u uuuu(3)
PCLATU	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
PCLATH	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6X2X	PIC18F8X2X	0000 000x	0000 000x	uuuu uuuu(1)
INTCON2	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu(1)
INTCON3	PIC18F6X2X	PIC18F8X2X	1100 0000	1100 0000	uuuu uuuu(1)
INDF0	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTINC0	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTDEC0	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PREINC0	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PLUSW0	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
FSR0H	PIC18F6X2X	PIC18F8X2X	---- 0000	---- 0000	---- uuuu
FSR0L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTINC1	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTDEC1	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PREINC1	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PLUSW1	PIC18F6X2X	PIC18F8X2X	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'.

7: If $\overline{\text{MCLR}}$ function is disabled, PORTG<5> is a read only bit.

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
FSR1H	PIC18F6X2X	PIC18F8X2X	---- 0000	---- 0000	---- uuuu
FSR1L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6X2X	PIC18F8X2X	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTINC2	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
POSTDEC2	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PREINC2	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
PLUSW2	PIC18F6X2X	PIC18F8X2X	N/A	N/A	N/A
FSR2H	PIC18F6X2X	PIC18F8X2X	---- 0000	---- 0000	---- uuuu
FSR2L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F6X2X	PIC18F8X2X	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6X2X	PIC18F8X2X	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6X2X	PIC18F8X2X	---- 0000	---- 0000	---- uuuu
LVDCON	PIC18F6X2X	PIC18F8X2X	--00 0101	--00 0101	--uu uuuu
WDTCON	PIC18F6X2X	PIC18F8X2X	---- ---0	---- ---0	---- ---u
RCON(4)	PIC18F6X2X	PIC18F8X2X	0--1 11qq	0--1 qquu	u--1 qquu
TMR1H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6X2X	PIC18F8X2X	0-00 0000	u-uu uuuu	u-uu uuuu
TMR2	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6X2X	PIC18F8X2X	-000 0000	-000 0000	-uuu uuuu
SSPBUF	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
SSPCON1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
SSPCON2	PIC18F6X2X	PIC18F8X2X	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'.

7: If $\overline{\text{MCLR}}$ function is disabled, PORTG<5> is a read only bit.

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
ADRESH	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
ADCON1	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6X2X	PIC18F8X2X	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
CCPR3H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6X2X	PIC18F8X2X	0000 0000	uuuu uuuu	uuuu uuuu
ECCP1AS	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TMR3H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6X2X	PIC18F8X2X	0000 0000	uuuu uuuu	uuuu uuuu
PSPCON	PIC18F6X2X	PIC18F8X2X	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TXSTA1	PIC18F6X2X	PIC18F8X2X	0000 -010	0000 -010	uuuu -uuu
RCSTA1	PIC18F6X2X	PIC18F8X2X	0000 000x	0000 000x	uuuu uuuu
EEADRH	PIC18F6X2X	PIC18F8X2X	---- --00	---- --00	---- --uu
EEADR	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
EEDATA	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
EECON2	PIC18F6X2X	PIC18F8X2X	xx-0 x000	uu-0 u000	uu-0 u000
EECON1	PIC18F6X2X	PIC18F8X2X	xx-x x000	uu-u u000	uu-u 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'.

7: If $\overline{\text{MCLR}}$ function is disabled, PORTG<5> is a read only bit.

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
IPR3	PIC18F6X2X	PIC18F8X2X	--11 1111	--11 1111	--uu uuuu
PIR3	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
PIE3	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
IPR2	PIC18F6X2X	PIC18F8X2X	-1-1 1111	-1-1 1111	-u-u uuuu
PIR2	PIC18F6X2X	PIC18F8X2X	-0-0 0000	-0-0 0000	-u-u uuuu(1)
PIE2	PIC18F6X2X	PIC18F8X2X	-0-0 0000	-0-0 0000	-u-u uuuu
IPR1	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu(1)
PIE1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6X2X	PIC18F8X2X	0-00 --00	0-00 --00	u-uu --uu
TRISJ	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6X2X	PIC18F8X2X	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
TRISA(5,6)	PIC18F6X2X	PIC18F8X2X	-111 1111(5)	-111 1111(5)	-uuu uuuu(5)
LATJ	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6X2X	PIC18F8X2X	---x xxxx	---u uuuu	---u uuuu
LATF	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5,6)	PIC18F6X2X	PIC18F8X2X	-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'.

7: If $\overline{\text{MCLR}}$ function is disabled, PORTG<5> is a read only bit.

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
PORTJ	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	xxxx uuuu	uuuu uuuu
PORTG(7)	PIC18F6X2X	PIC18F8X2X	---x xxxx	---u uuuu	---u uuuu
PORTF	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
PORTE	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	0000 0000	uuuu uuuu
PORTD	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5,6)	PIC18F6X2X	PIC18F8X2X	-xxx 0000(5)	-uuu 0000(5)	-uuu uuuu(5)
SPBRGH1	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6X2X	PIC18F8X2X	-1-0 0-00	-1-0 0-00	-1-u u-uu
SPBRGH2	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
BAUDCON2	PIC18F6X2X	PIC18F8X2X	-1-0 0-00	-1-0 0-00	-1-u u-uu
ECCP1DEL	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TMR4	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
PR4	PIC18F6X2X	PIC18F8X2X	1111 1111	1111 1111	uuuu uuuu
T4CON	PIC18F6X2X	PIC18F8X2X	-000 0000	-000 0000	-uuu uuuu
CCPR4H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR4L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP4CON	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR5L	PIC18F6X2X	PIC18F8X2X	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP5CON	PIC18F6X2X	PIC18F8X2X	--00 0000	--00 0000	--uu uuuu
SPBRG2	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6X2X	PIC18F8X2X	0000 -010	0000 -010	uuuu -u1u
RCSTA2	PIC18F6X2X	PIC18F8X2X	0000 000x	0000 000x	uuuu uuu-
ECCP3AS	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
ECCP2AS	PIC18F6X2X	PIC18F8X2X	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F6X2X	PIC18F8X2X	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'.

7: If $\overline{\text{MCLR}}$ function is disabled, PORTG<5> is a read only bit.

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

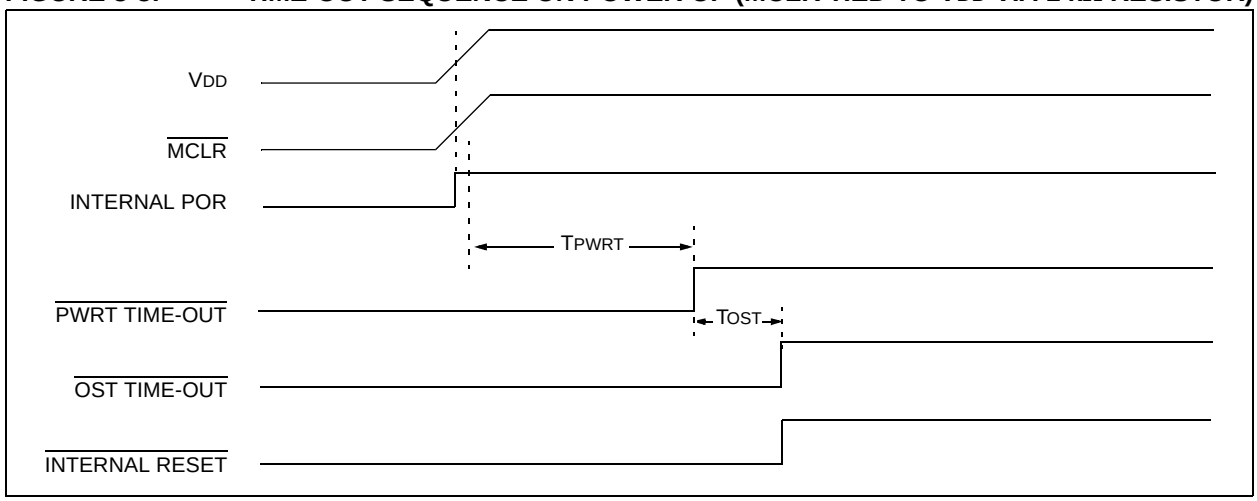


FIGURE 3-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1

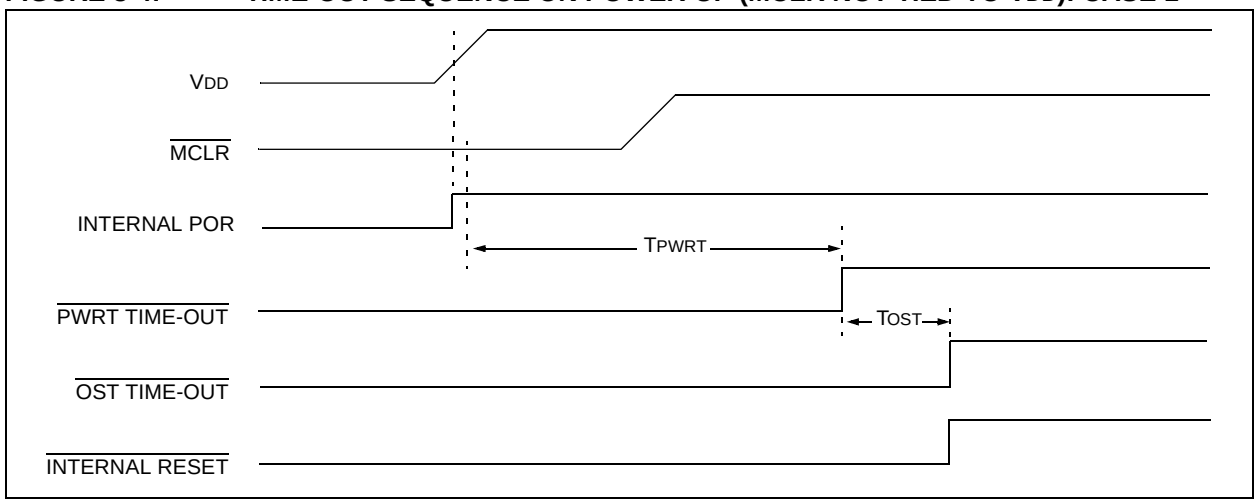


FIGURE 3-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

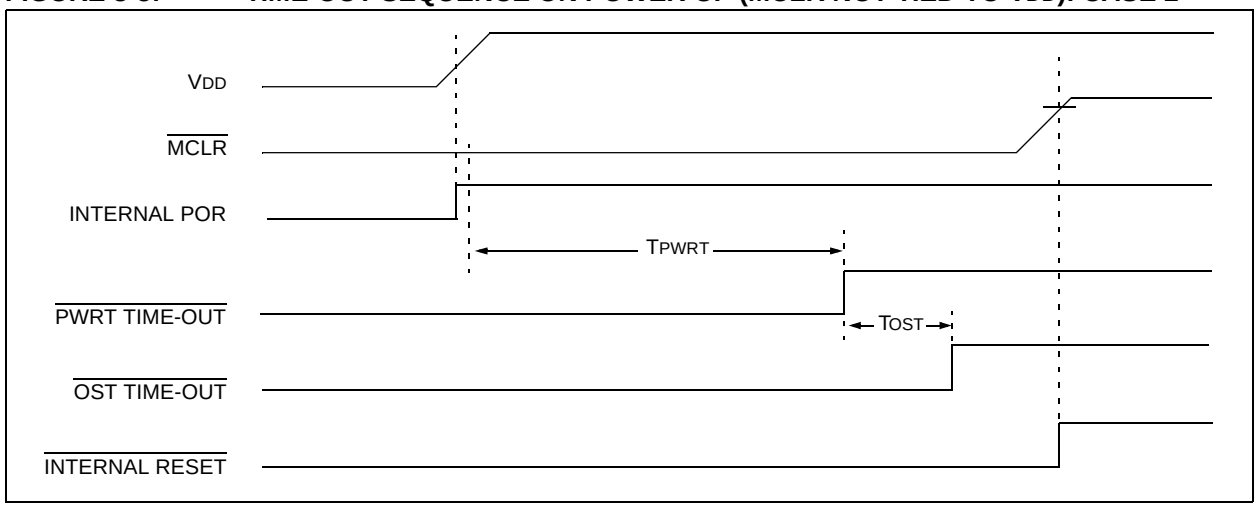


FIGURE 3-6: SLOW RISE TIME ($\overline{\text{MCLR}}$ TIED TO V_{DD} VIA 1 kΩ RESISTOR)

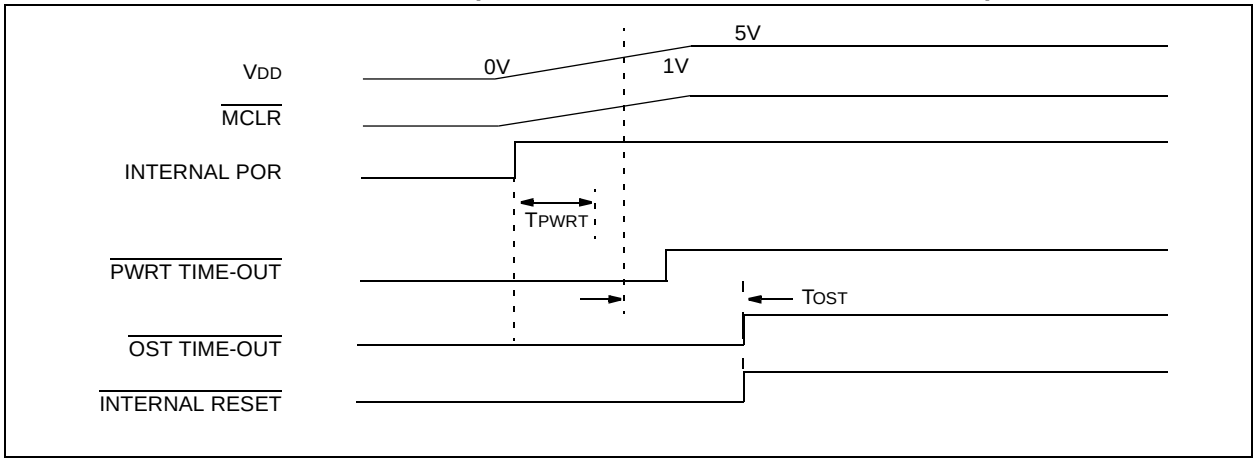
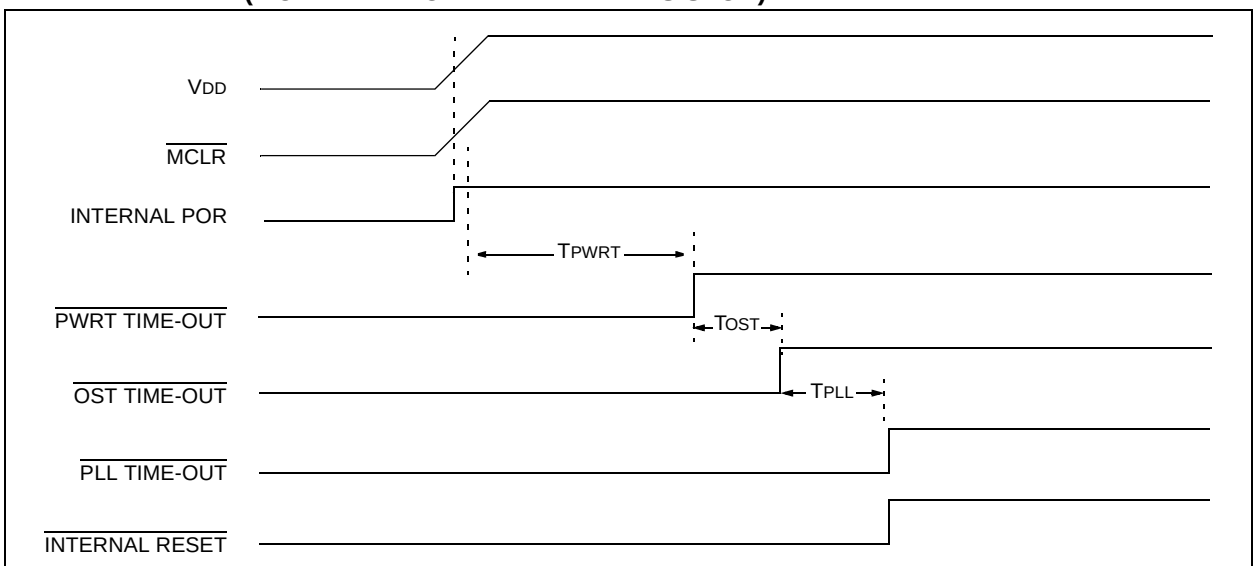


FIGURE 3-7: TIME-OUT SEQUENCE ON POR W/ PLL ENABLED (MCLR TIED TO V_{DD} VIA 1 kΩ RESISTOR)



Note: TOST = 1024 clock cycles.
TPLL ≈ 2 ms max. First three stages of the PWRT timer.

4.0 MEMORY ORGANIZATION

There are three memory blocks in PIC18F6X2X/8X2X 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 and Section 7.0, respectively.

In addition to on-chip FLASH, the PIC18F8X2X 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), 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.

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).

The PIC18F6525 and PIC18F8525 each have 48 Kbytes of on-chip FLASH memory, while the PIC18F6621 and PIC18F8621 have 64 Kbytes of FLASH. This means that PIC18FX525 devices can store internally up to 24,576 single word instructions, and PIC18FX621 devices can store up to 32,768 single word instructions.

The RESET vector address is at 0000h, and the interrupt vector addresses are at 0008h and 0018h.

Figure 4-1 shows the program memory map for PIC18FX525 devices, while Figure 4-2 shows the program memory map for PIC18FX621 devices.

4.1.1 PIC18F6X2X/8X2X PROGRAM MEMORY MODES

PIC18F8X2X 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 register as shown in Register 4-1. (See also Section 24.1 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. 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 (BFFFh for the PIC18FX525, FFFFh for the PIC18FX621) causes a read of all ‘0’s (a `NOP` instruction). The Microcontroller mode is also the only operating mode available to PIC18F6X2X 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 4-3 compares the memory maps of the different Program Memory modes. The differences between on-chip and external memory access limitations are more fully explained in Table 4-1.

FIGURE 4-1: INTERNAL PROGRAM MEMORY MAP AND STACK FOR PIC18FX525

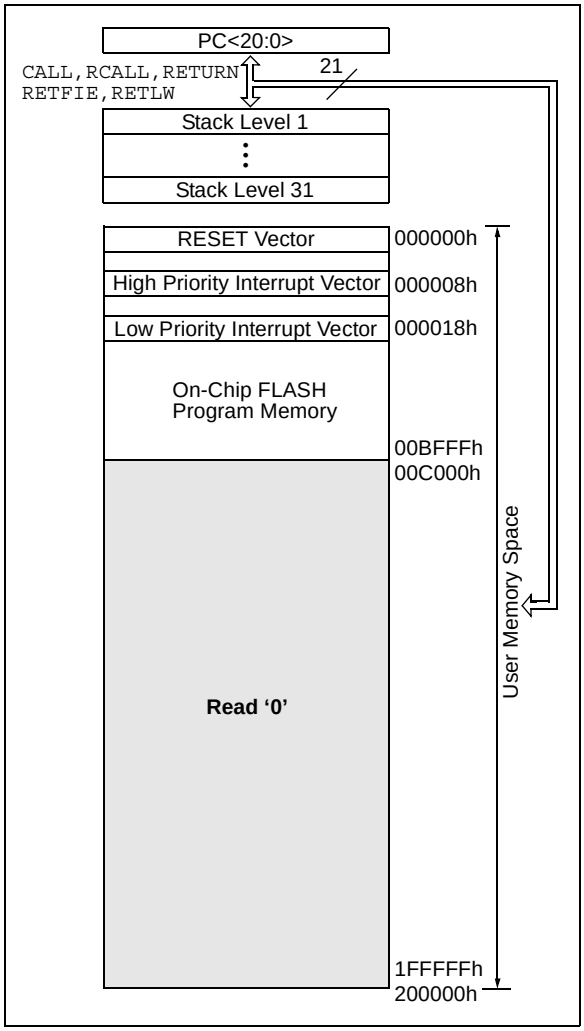


FIGURE 4-2: INTERNAL PROGRAM MEMORY MAP AND STACK FOR PIC18FX621

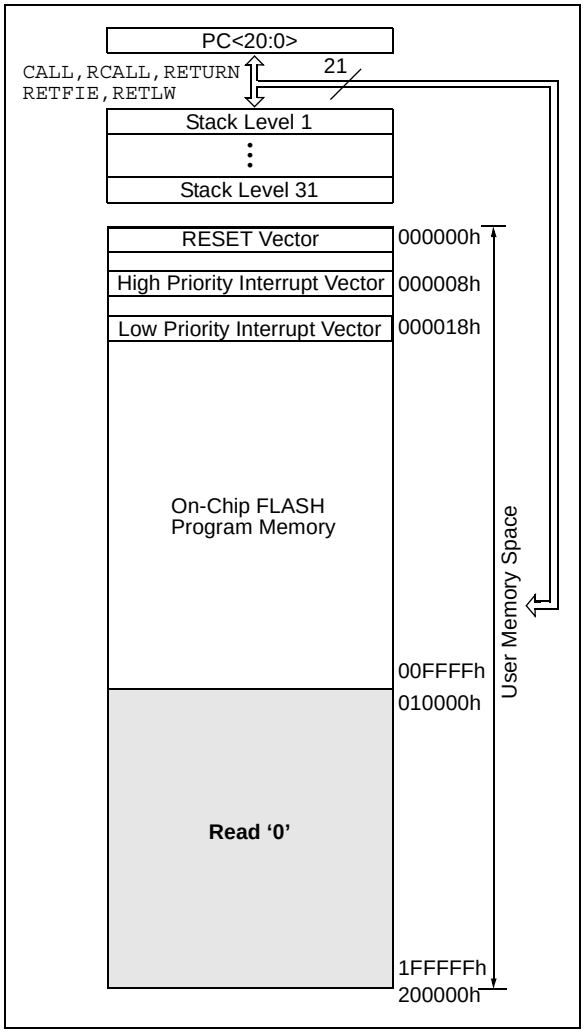


TABLE 4-1: MEMORY ACCESS FOR PIC18F8X2X PROGRAM MEMORY MODES

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 w/ 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

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>)

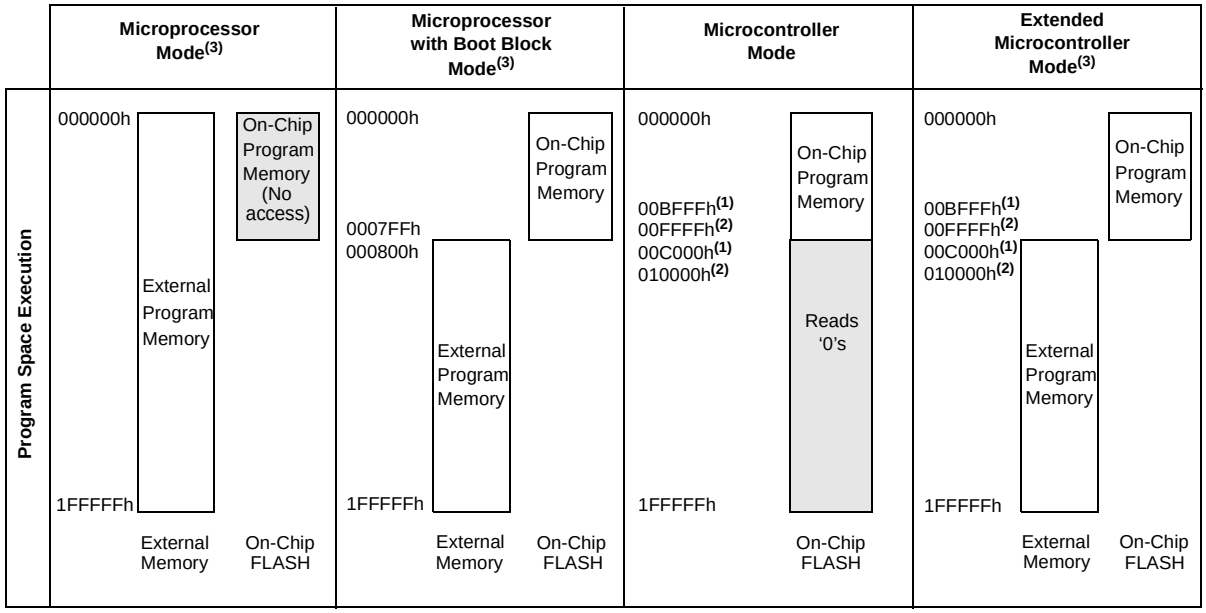
bit 6-2 **Unimplemented:** Read as '0'

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

Note 1: This mode is available only on PIC18F8X2X devices.

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 4-3: MEMORY MAPS FOR PIC18F6X2X/8X2X PROGRAM MEMORY MODES



Note 1: PIC18F8525 and PIC18F6525.
2: PIC18F8621 and PIC18F6621.
3: This mode is available only on PIC18F8X2X devices.

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 25.0 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.

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

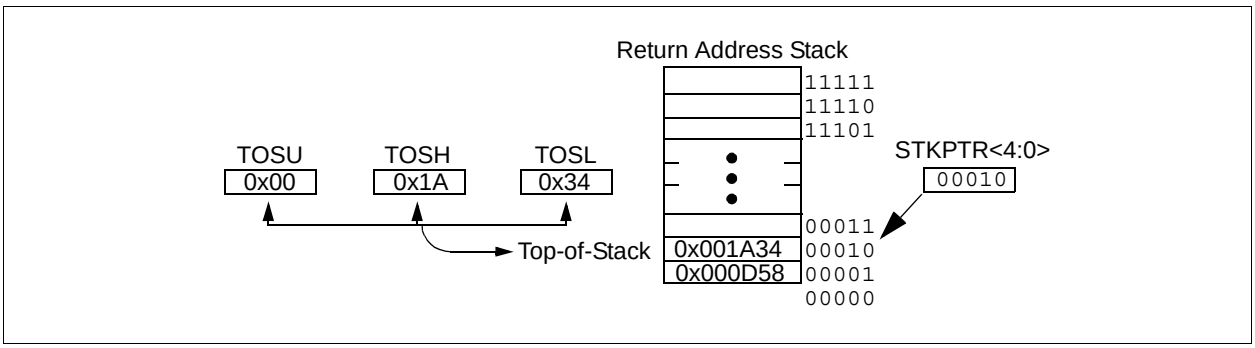
bit 5 **Unimplemented:** Read as '0'

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 4-4: 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 RETURN` instruction 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.

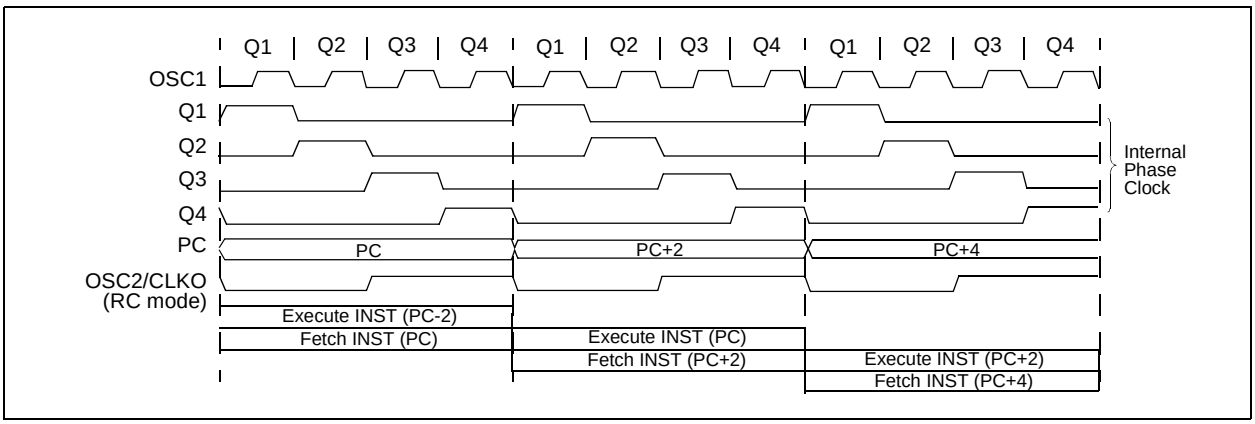
The `CALL, RCALL, GOTO` and program branch instructions write to the program counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the program counter.

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).

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 4-5.

FIGURE 4-5: CLOCK/INSTRUCTION CYCLE



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 take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO), then two cycles are required to complete the instruction (Example 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

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	Fetch 1	Execute 1				
2. MOVWF PORTB		Fetch 2	Execute 2			
3. BRA SUB_1			Fetch 3	Execute 3		
4. BSF PORTA, BIT3 (Forced NOP)				Fetch 4	Flush (NOP)	
5. Instruction @ address SUB_1					Fetch SUB_1	Execute SUB_1

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 4-6 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).

The CALL and GOTO instructions 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 4-6 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 25.0 provides further details of the instruction set.

FIGURE 4-6: INSTRUCTIONS IN PROGRAM MEMORY

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

4.7.1 TWO-WORD INSTRUCTIONS

The PIC18F6X2X/8X2X 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 25.0 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 Lookup Tables

Lookup 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 lookup 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.

Lookup 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.

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. Figure 4-7 shows the data memory organization for the PIC18F6X2X/8X2X devices.

The data memory map is divided into 16 banks that contain 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 for the BSR are not implemented.

The data memory contains 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.

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 `MOVFF` instruction. The `MOVFF` instruction 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 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.

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 GPRs, 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 4-7: DATA MEMORY MAP FOR PIC18F6X2X/8X2X DEVICES

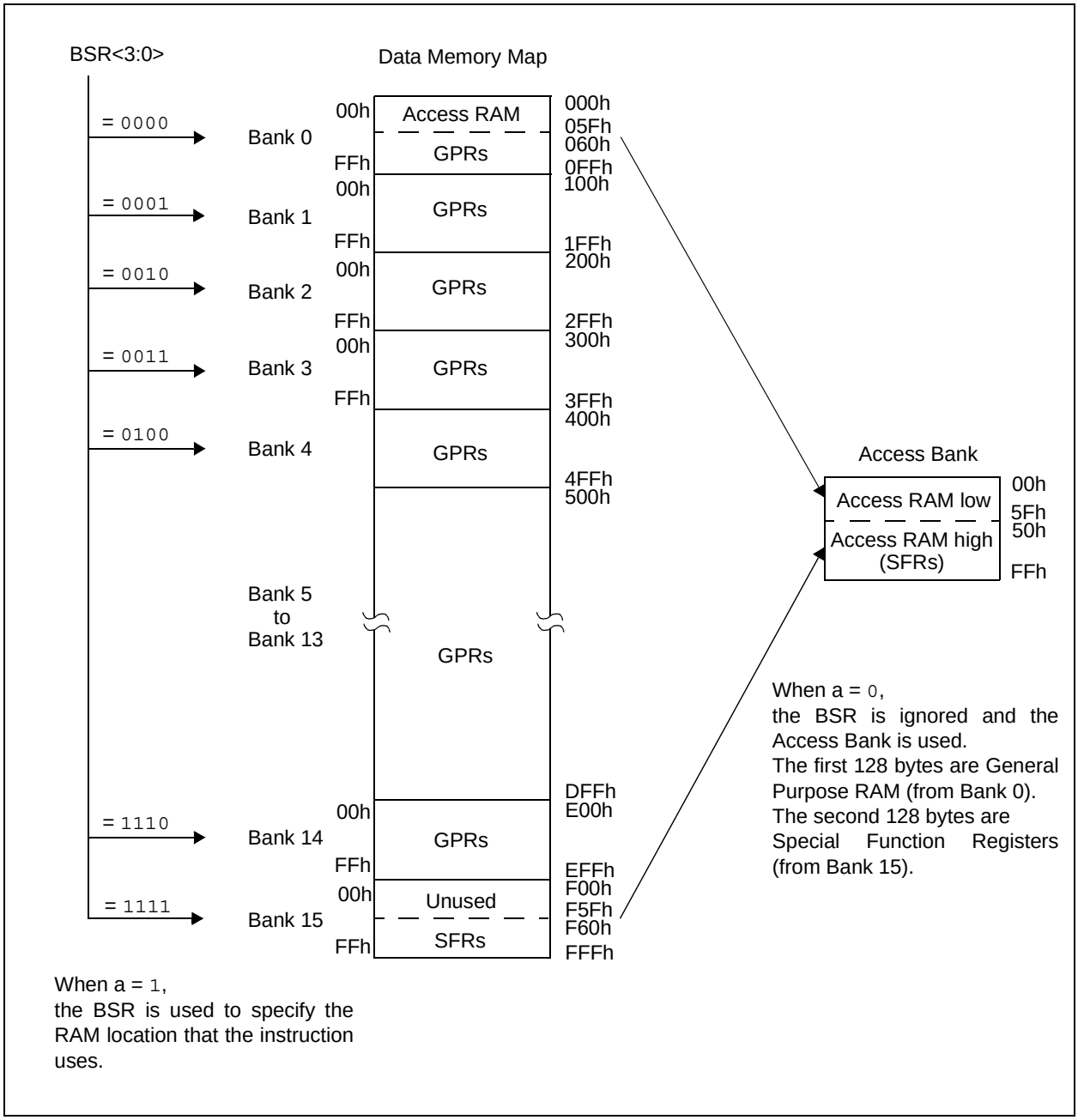


TABLE 4-2: SPECIAL FUNCTION REGISTER MAP

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(2)
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH(2)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	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(2)
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(2)
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(2)
FE7h	INDF1(3)	FC7h	SSPSTAT	FA7h	EECON2	F87h	PORTH(2)
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 not available on PIC18F6X2X devices.

3: This is not a physical register.

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

Address	Name	Address	Name	Address	Name	Address	Name
F7Fh	SPBRGH1	F5Fh	—(1)	F3Fh	—(1)	F1Fh	—(1)
F7Eh	BAUDCON1	F5Eh	—(1)	F3Eh	—(1)	F1Eh	—(1)
F7Dh	SPBRGH2	F5Dh	—(1)	F3Dh	—(1)	F1Dh	—(1)
F7Ch	BAUDCON2	F5Ch	—(1)	F3Ch	—(1)	F1Ch	—(1)
F7Bh	—(1)	F5Bh	—(1)	F3Bh	—(1)	F1Bh	—(1)
F7Ah	—(1)	F5Ah	—(1)	F3Ah	—(1)	F1Ah	—(1)
F79h	ECCP1DEL	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	ECCP3AS	F4Ah	—(1)	F2Ah	—(1)	F0Ah	—(1)
F69h	ECCP3DEL	F49h	—(1)	F29h	—(1)	F09h	—(1)
F68h	ECCP2AS	F48h	—(1)	F28h	—(1)	F08h	—(1)
F67h	ECCP2DEL	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 PIC18F6X2X 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	34, 44
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	34, 44
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	34, 44
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					00-0 0000	34, 45
PCLATU	—	—	bit 21	Holding Register for PC<20:16>					--10 0000	34, 46
PCLATH	Holding Register for PC<15:8>								0000 0000	34, 46
PCL	PC Low Byte (PC<7:0>)								0000 0000	34, 46
TBLPTRU	—	—	bit 21(2)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	34, 71
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	34, 71
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	34, 71
TABLAT	Program Memory Table Latch								0000 0000	34, 71
PRODH	Product Register High Byte								xxxx xxxx	34, 87
PRODL	Product Register Low Byte								xxxx xxxx	34, 87
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	34, 91
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	34, 92
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	34, 93
INDF0	Uses contents of FSR0 to address data memory - value of FSR0 not changed (not a physical register)								n/a	58
POSTINC0	Uses contents of FSR0 to address data memory - value of FSR0 post-incremented (not a physical register)								n/a	58
POSTDEC0	Uses contents of FSR0 to address data memory - value of FSR0 post-decremented (not a physical register)								n/a	58
PREINC0	Uses contents of FSR0 to address data memory - value of FSR0 pre-incremented (not a physical register)								n/a	58
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	58
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- 0000	34, 58
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	34, 58
WREG	Working Register								xxxx xxxx	34
INDF1	Uses contents of FSR1 to address data memory - value of FSR1 not changed (not a physical register)								n/a	58
POSTINC1	Uses contents of FSR1 to address data memory - value of FSR1 post-incremented (not a physical register)								n/a	58
POSTDEC1	Uses contents of FSR1 to address data memory - value of FSR1 post-decremented (not a physical register)								n/a	58
PREINC1	Uses contents of FSR1 to address data memory - value of FSR1 pre-incremented (not a physical register)								n/a	58
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	58
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000	35, 58
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	35, 58
BSR	—	—	—	—	Bank Select Register				---- 0000	35, 57
INDF2	Uses contents of FSR2 to address data memory - value of FSR2 not changed (not a physical register)								n/a	58
POSTINC2	Uses contents of FSR2 to address data memory - value of FSR2 post-incremented (not a physical register)								n/a	58
POSTDEC2	Uses contents of FSR2 to address data memory - value of FSR2 post-decremented (not a physical register)								n/a	58

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 mode 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 PIC18F6X2X devices; always maintain these clear.

4: RG5 is available only if $\overline{\text{MCLR}}$ function is disabled in configuration.

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	58
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	58
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000	35, 58
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	35, 58
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	35, 60
TMR0H	Timer0 Register High Byte								0000 0000	35, 135
TMR0L	Timer0 Register Low Byte								xxxx xxxx	35, 135
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	35, 133
OSCCON	—	—	—	—	—	—	—	SCS	---- ---0	25, 35
LVDCON	—	—	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0	--00 0101	35, 255
WDTCON	—	—	—	—	—	—	—	SWDTE	---- ---0	35, 269
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 11qq	35, 61, 103
TMR1H	Timer1 Register High Byte								xxxx xxxx	35, 141
TMR1L	Timer1 Register Low Byte								xxxx xxxx	35, 141
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0-00 0000	35, 141
TMR2	Timer2 Register								0000 0000	35, 144
PR2	Timer2 Period Register								1111 1111	35, 144
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	35, 144
SSPBUF	SSP Receive Buffer/Transmit Register								xxxx xxxx	35, 183
SSPADD	SSP Address Register in I^2C Slave mode. SSP Baud Rate Reload Register in I^2C Master mode.								0000 0000	35, 183
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	35, 176
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	35, 177
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	35, 187
ADRESH	A/D Result Register High Byte								xxxx xxxx	36, 242
ADRESL	A/D Result Register Low Byte								xxxx xxxx	36, 242
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	--00 0000	36, 235
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	36, 236
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	36, 237
CCPR1H	Enhanced Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	36, 174
CCPR1L	Enhanced Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	36, 174
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	36, 159
CCPR2H	Enhanced Capture/Compare/PWM Register 2 High Byte								xxxx xxxx	36, 174
CCPR2L	Enhanced Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx	36, 174
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000	36, 159
CCPR3H	Enhanced Capture/Compare/PWM Register 3 High Byte								xxxx xxxx	36, 174
CCPR3L	Enhanced Capture/Compare/PWM Register 3 Low Byte								xxxx xxxx	36, 174
CCP3CON	P3M1	P3M0	CCP3X	CCP3Y	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000 0000	36, 159
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	0000 0000	36, 171
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	36, 249

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 mode 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 PIC18F6X2X devices; always maintain these clear.

4: RG5 is available only if $\overline{MCLR}$ function is disabled in configuration.

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:
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	36, 243
TMR3H	Timer3 Register High Byte								xxxx xxxx	36, 147
TMR3L	Timer3 Register Low Byte								xxxx xxxx	36, 147
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	0000 0000	36, 147
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	36, 131
SPBRG1	USART1 Baud Rate Generator								0000 0000	36, 219
RCREG1	USART1 Receive Register								0000 0000	36, 226
TXREG1	USART1 Transmit Register								0000 0000	36, 224
TXSTA1	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	36, 216
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	36, 217
EEADRH	—	—	—	—	—	—	EE Adr Register High		---- --00	36, 85
EEADR	Data EEPROM Address Register								0000 0000	36, 85
EEDATA	Data EEPROM Data Register								0000 0000	36, 85
EECON2	Data EEPROM Control Register 2 (not a physical register)								---- ----	36, 85
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	36, 82
IPR3	—	—	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	--11 1111	37, 102
PIR3	—	—	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	--00 0000	37, 96
PIE3	—	—	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	--00 0000	37, 99
IPR2	—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP	-1-1 1111	37, 101
PIR2	—	CMIF	—	EEIF	BCLIF	LVDIF	TMR3IF	CCP2IF	-0-0 0000	37, 95
PIE2	—	CMIE	—	EEIE	BCLIE	LVDIE	TMR3IE	CCP2IE	-0-0 0000	37, 98
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111	37, 100
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	37, 94
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	37, 97
MEMCON(3)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	37, 73
TRISJ(3)	Data Direction Control Register for PORTJ								1111 1111	37, 129
TRISH(3)	Data Direction Control Register for PORTH								1111 1111	37, 126
TRISG	—	—	—	Data Direction Control Register for PORTG					---1 1111	37, 121
TRISF	Data Direction Control Register for PORTF								1111 1111	37, 118
TRISE	Data Direction Control Register for PORTE								1111 1111	37, 115
TRISD	Data Direction Control Register for PORTD								1111 1111	37, 112
TRISC	Data Direction Control Register for PORTC								1111 1111	37, 110
TRISB	Data Direction Control Register for PORTB								1111 1111	37, 107
TRISA	—	TRISA6(1)	Data Direction Control Register for PORTA						-111 1111	37, 123
LATJ(3)	Read PORTJ Data Latch, Write PORTJ Data Latch								xxxx xxxx	37, 129
LATH(3)	Read PORTH Data Latch, Write PORTH Data Latch								xxxx xxxx	37, 126
LATG	—	—	—	Read PORTG Data Latch, Write PORTG Data Latch					---x xxxx	37, 123
LATF	Read PORTF Data Latch, Write PORTF Data Latch								xxxx xxxx	37, 121
LATE	Read PORTE Data Latch, Write PORTE Data Latch								xxxx xxxx	37, 118
LATD	Read PORTD Data Latch, Write PORTD Data Latch								xxxx xxxx	37, 115
LATC	Read PORTC Data Latch, Write PORTC Data Latch								xxxx xxxx	37, 112
LATB	Read PORTB Data Latch, Write PORTB Data Latch								xxxx xxxx	37, 110
LATA	—	LATA6(1)	Read PORTA Data Latch, Write PORTA Data Latch(1)						-xxx xxxx	37, 107

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 mode 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 PIC18F6X2X devices; always maintain these clear.

4: RG5 is available only if $\overline{\text{MCLR}}$ function is disabled in configuration.

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:
PORTJ(3)	Read PORTJ pins, Write PORTJ Data Latch								xxxx xxxx	38, 129
PORTH(3)	Read PORTH pins, Write PORTH Data Latch								xxxx xxxx	38, 126
PORTG	—	—	RG5(4)	Read PORTG pins, Write PORTG Data Latch					--0x xxxx	38, 123
PORTF	Read PORTF pins, Write PORTF Data Latch								xxxx xxxx	38, 121
PORTE	Read PORTE pins, Write PORTE Data Latch								xxxx xxxx	38, 118
PORTD	Read PORTD pins, Write PORTD Data Latch								xxxx xxxx	38, 115
PORTC	Read PORTC pins, Write PORTC Data Latch								xxxx xxxx	38, 112
PORTB	Read PORTB pins, Write PORTB Data Latch								xxxx xxxx	38, 110
PORTA	—	RA6(1)	Read PORTA pins, Write PORTA Data Latch(1)						-x0x 0000	38, 107
SPBRGH1	Enhanced USART1 Baud Rate Generator High Byte								0000 0000	38, 219
BAUDCON1	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	38, 218
SPBRGH2	Enhanced USART2 Baud Rate Generator High Byte								0000 0000	38, 219
BAUDCON2	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-0 0-00	38, 218
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000	38, 170
TMR4	Timer4 Register								0000 0000	38, 150
PR4	Timer4 Period Register								1111 1111	38, 150
T4CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	38, 149
CCPR4H	Capture/Compare/PWM Register 4 High Byte								xxxx xxxx	38, 155
CCPR4L	Capture/Compare/PWM Register 4 Low Byte								xxxx xxxx	38, 155
CCP4CON	—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0	--00 0000	38, 151
CCPR5H	Capture/Compare/PWM Register 5 High Byte								xxxx xxxx	38, 155
CCPR5L	Capture/Compare/PWM Register 5 Low Byte								xxxx xxxx	38, 155
CCP5CON	—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0	--00 0000	38, 151
SPBRG2	USART2 Baud Rate Generator								0000 0000	38, 219
RCREG2	USART2 Receive Register								0000 0000	38, 226
TXREG2	USART2 Transmit Register								0000 0000	38, 224
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 -010	38, 224
RCSTA2	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 000x	38, 224
ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0	0000 0000	38, 171
ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0	0000 0000	38, 170
ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000	38, 171
ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000	38, 170

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 mode 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 PIC18F6X2X devices; always maintain these clear.

4: RG5 is available only if $\overline{\text{MCLR}}$ function is disabled in configuration.

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.

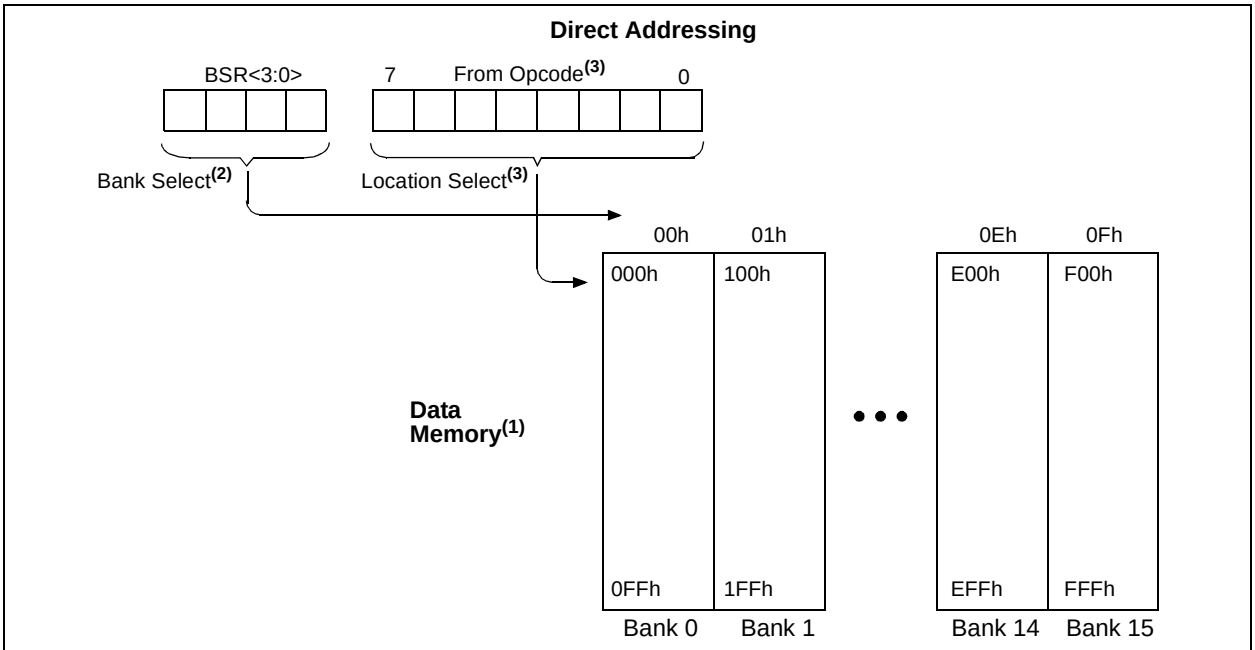
If the currently selected bank is not implemented, any read will return all ‘0’s and all writes are ignored. The STATUS register bits will be set/cleared as appropriate for the instruction performed.

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

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

Section 4.12 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 MOVFF instruction embeds the entire 12-bit address in the instruction.

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
                           ; Bank1?
        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 NOP instruction 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

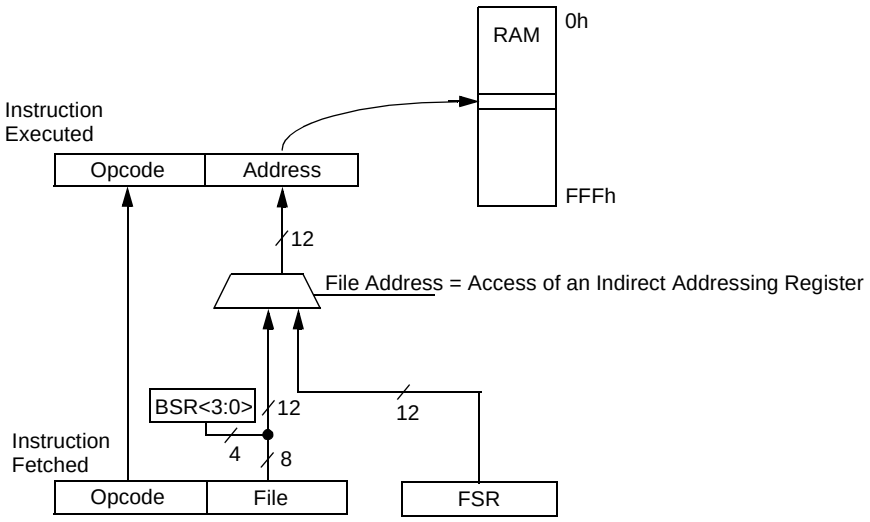
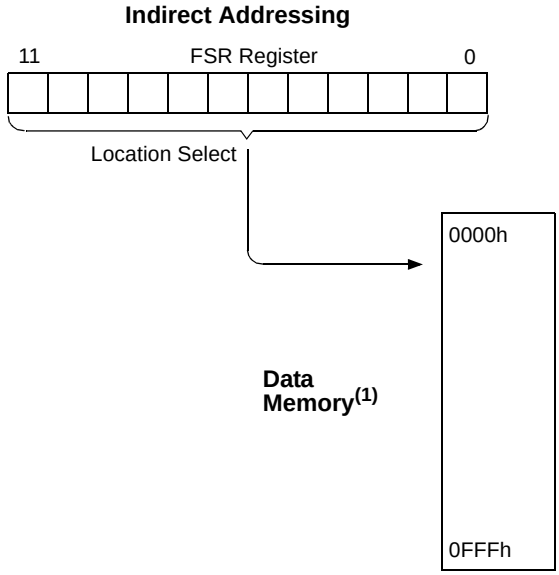


FIGURE 4-10: INDIRECT ADDRESSING



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

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 25-2.

Note:	The C and DC bits operate as a $\overline{\text{borrow}}$ and $\overline{\text{digit}}$ 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 two’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 two’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: 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 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed
0 = The RESET instruction was executed causing a device RESET
(must be set in software after a Brown-out Reset occurs)

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

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

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

NOTES:

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)

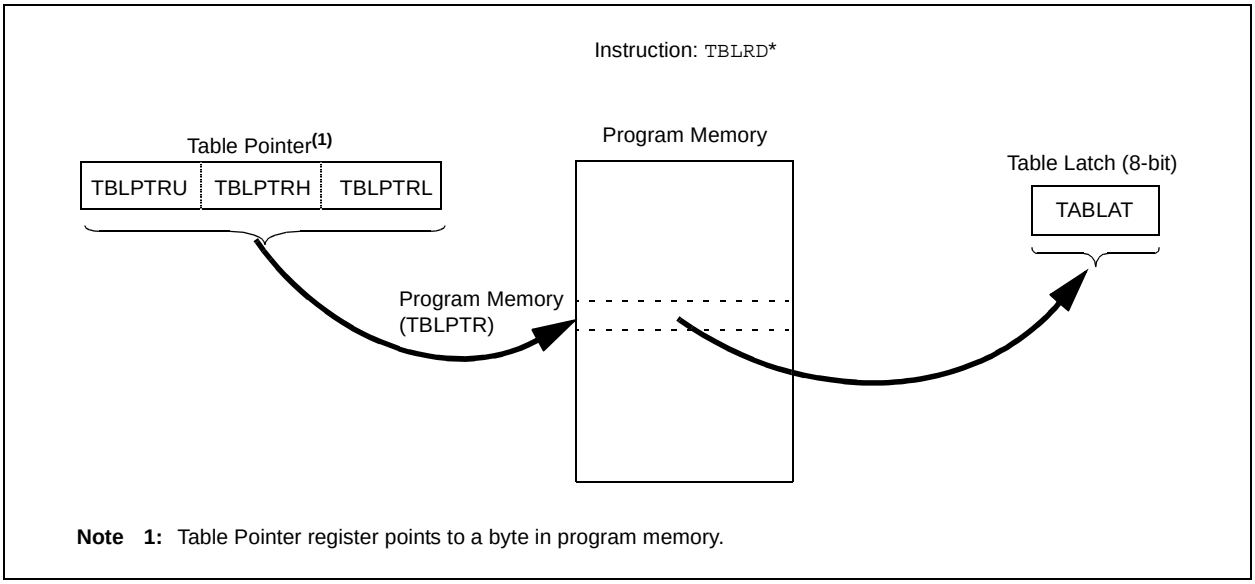
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and places 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.

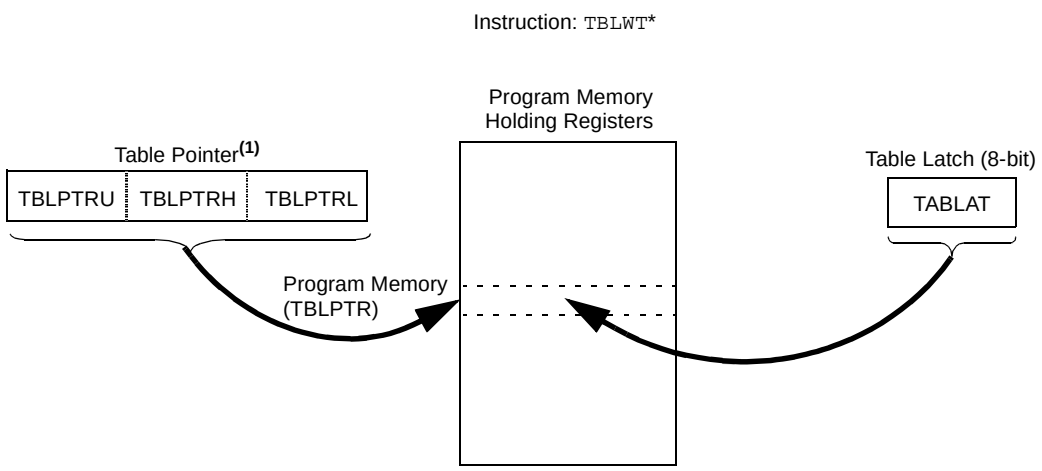
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions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

FIGURE 5-1: TABLE READ OPERATION



Note 1: Table Pointer register 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.

5.2 Control Registers

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

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

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 24.0, “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.

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 register 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 register (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 `TBLRD` and `TBLWT` instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations are shown in Table 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 TBLPTR determine which byte is read from program memory into TABLAT.

When a `TBLWT` is executed, the three LSbs of the Table Pointer register (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 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”.

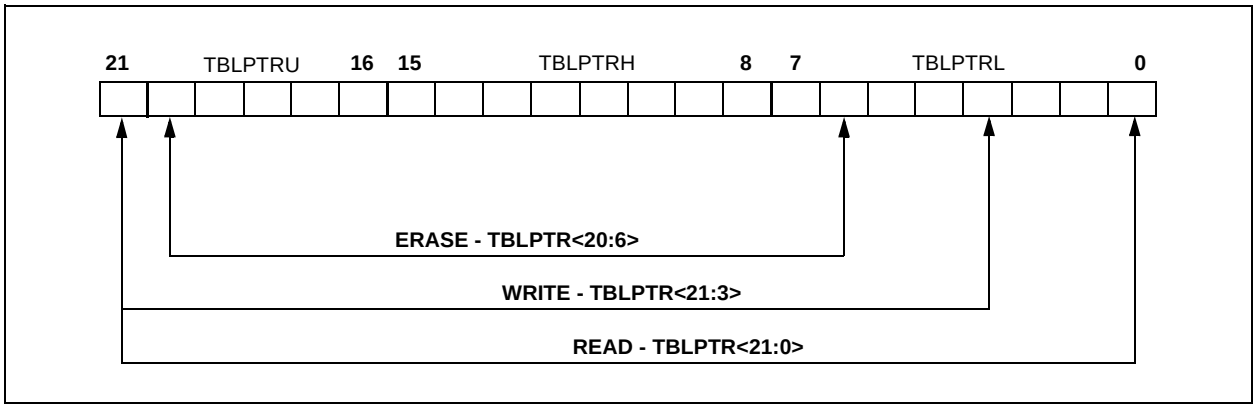
When an erase of program memory is executed, the 16 MSbs of the Table Pointer register (TBLPTR<21:6>) point to the 64-byte block that will be erased. The Least Significant bits (TBLPTR<5:0>) are ignored.

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

TABLE 5-1: TABLE POINTER OPERATIONS WITH `TBLRD` AND `TBLWT` INSTRUCTIONS

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

FIGURE 5-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



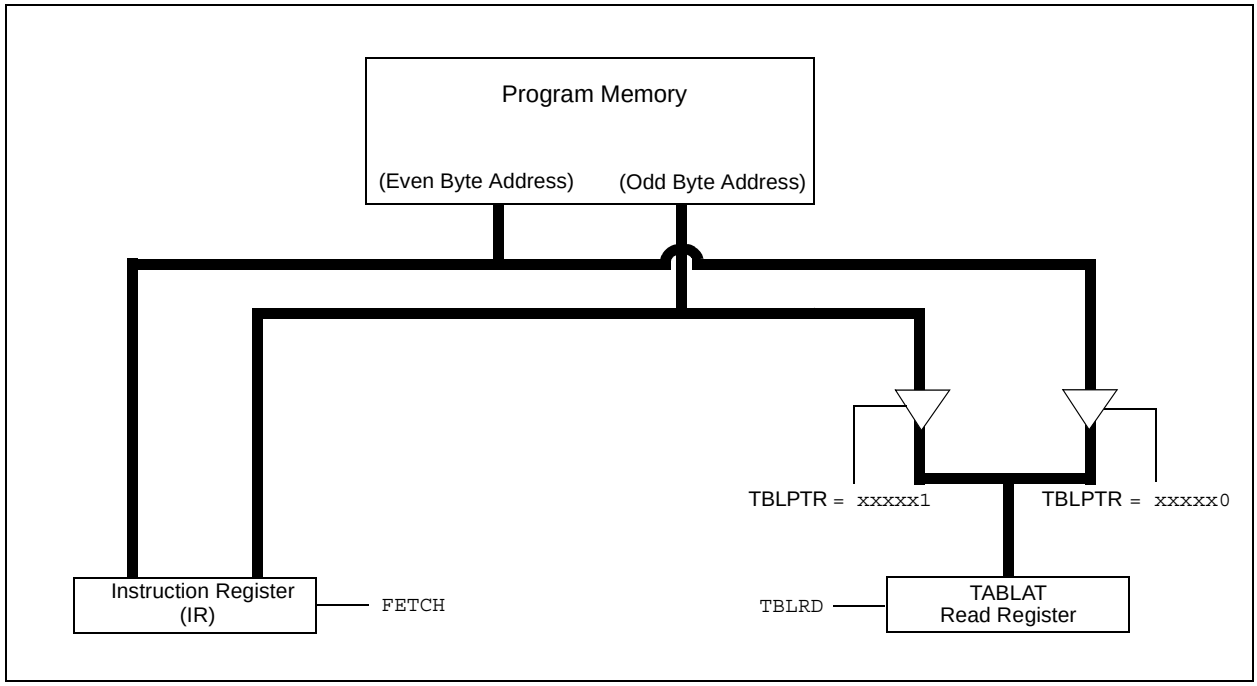
5.3 Reading the FLASH Program Memory

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 register 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. 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
    Required        MOVLW   55h
    Sequence        MOVWF   EECON2                ; write 55H
                    MOVLW   AAh
                    MOVWF   EECON2                ; write AAH
                    BSF     EECON1,WR             ; start erase (CPU stall)
                    BSF     INTCON,GIE            ; re-enable interrupts
```


5.5 Writing to FLASH Program Memory

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

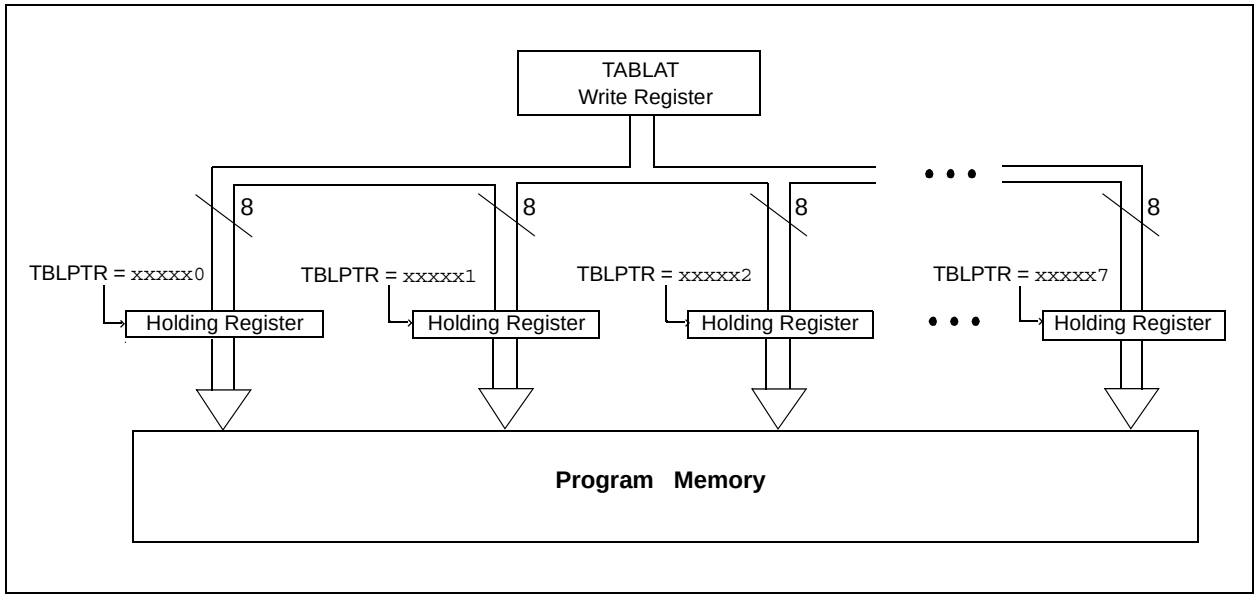
Table writes are used internally to load the holding registers needed to program the FLASH memory. There are 8 holding registers used by the Table Writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction 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 register with address being erased.
4. Do the row erase procedure.
5. Load Table Pointer register 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. Re-enable interrupts.
14. Repeat steps 6-14 seven times to write 64 bytes.
15. 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
Required        MOVWF   EECON2                  ; write 55H
Sequence        MOVLW   AAh
                MOVWF   EECON2                  ; write AAH
                BSF     EECON1,WR               ; start erase (CPU stall)
                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
                MOVFW   POSTINC0, W             ; get low byte of buffer data
                MOVWF   TABLAT                  ; present data to table latch
                TBLWT+*                         ; write data, perform a short write
                                                ; to internal TBLWT holding register.
                DECFSZ  COUNTER                 ; loop until buffers are full
                BRA     WRITE_WORD_TO_HREGS
```


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
    Required    MOVWF   EECON2                ; write 55H
    Sequence    MOVLW   AAh
                MOVWF   EECON2                ; write AAH
                BSF     EECON1,WR             ; start program (CPU stall)
                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 24.0, “Special Features of the CPU” for more detail.

5.6 FLASH Program Operation During Code Protection

See Section 24.0, “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	—	—	bit21	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:

6.0 EXTERNAL MEMORY INTERFACE

Note: The external memory interface is not implemented on PIC18F6X2X (64-pin) devices.

The external memory interface is a feature of the PIC18F8X2X 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 PIC18F8X2X 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 PIC18F8X2X 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, “PIC18F6X2X/8X2X Program Memory Modes”.

6.1 Program Memory Modes and the External Memory Interface

As previously noted, PIC18F8X2X 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 depends 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.

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 **WAIT1:WAIT0**: 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 **WM1:WM0**: TBLWRT Operation 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 MS and LS Byte, $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate
00 = Byte Write mode: TABLAT data copied on both MS and LS Byte, $\overline{WRH}$ or $\overline{WRL}$ will activate

Note: The MEMCON register is unimplemented and reads all ‘0’s when the device is in Microcontroller 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

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 $\overline{CE}$, $\overline{OE}$, WRH, $\overline{WRL}$, $\overline{UB}$ and $\overline{LB}$ signals are '1', and ALE and BA0 are '0'.

TABLE 6-1: PIC18F8X2X 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

6.2 16-bit Mode

The external memory interface implemented in PIC18F8X2X devices operates only in 16-bit mode. The mode selection is not software configurable but is programmed via the configuration bits.

The WM1:WM0 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 A15:A0 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.

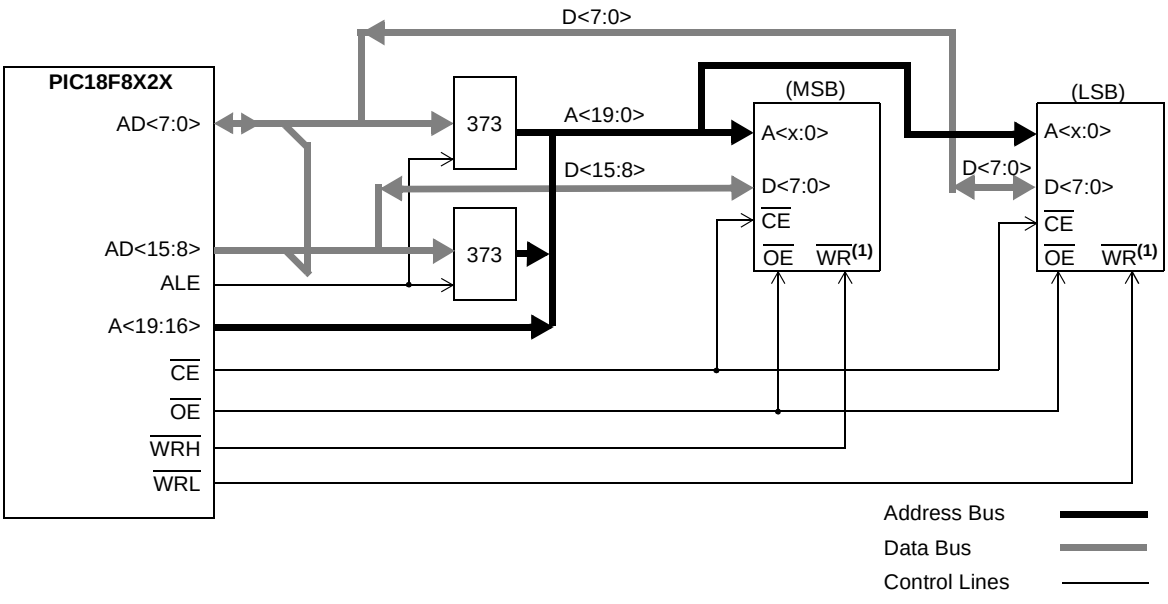
In Byte Select mode, JEDEC standard FLASH memories will require BA0 for the byte address line and one I/O line, to select between Byte and Word mode. The other 16-bit modes do not need BA0. JEDEC standard static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte selection.

6.2.1 16-BIT BYTE WRITE MODE

Figure 6-1 shows an example of 16-bit Byte Write mode for PIC18F8X2X devices. This mode is used for two separate 8-bit memories connected for 16-bit operation. This generally includes basic EPROM and FLASH devices. It allows table writes to byte-wide external memories.

During a `TBLWT` instruction cycle, the TABLAT data is presented on the upper and lower bytes of the 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 Writes”.

6.2.2 16-BIT WORD WRITE MODE

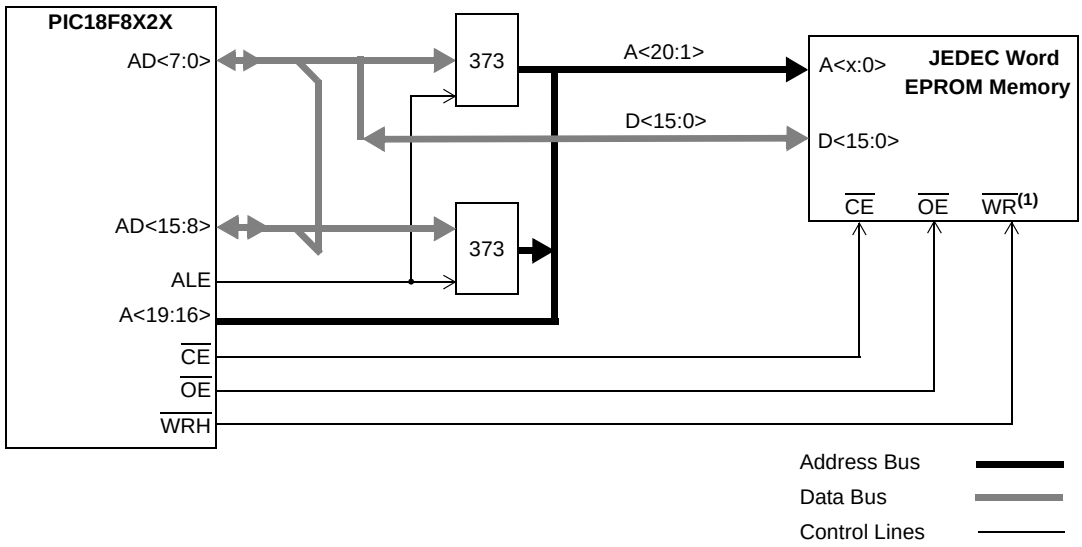
Figure 6-2 shows an example of 16-bit Word Write mode for PIC18F8X2X devices. This mode is used for word-wide memories which includes some of the EPROM and FLASH type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory, and table writes to any type of word-wide external memories. This method makes a distinction between TBLWT cycles to even or odd addresses.

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

During a TBLWT cycle to an odd address (TBLPTR<0> = 1), the TABLAT data is presented on the upper byte of the 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 LSbit of the TBLPTR but it is left unconnected. Instead, the $\overline{UB}$ and $\overline{LB}$ signals are active to select both bytes. The obvious limitation to this method is that the table write must be done in pairs on a specific word boundary to correctly write a word location.

FIGURE 6-2: 16-BIT WORD WRITE MODE EXAMPLE



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

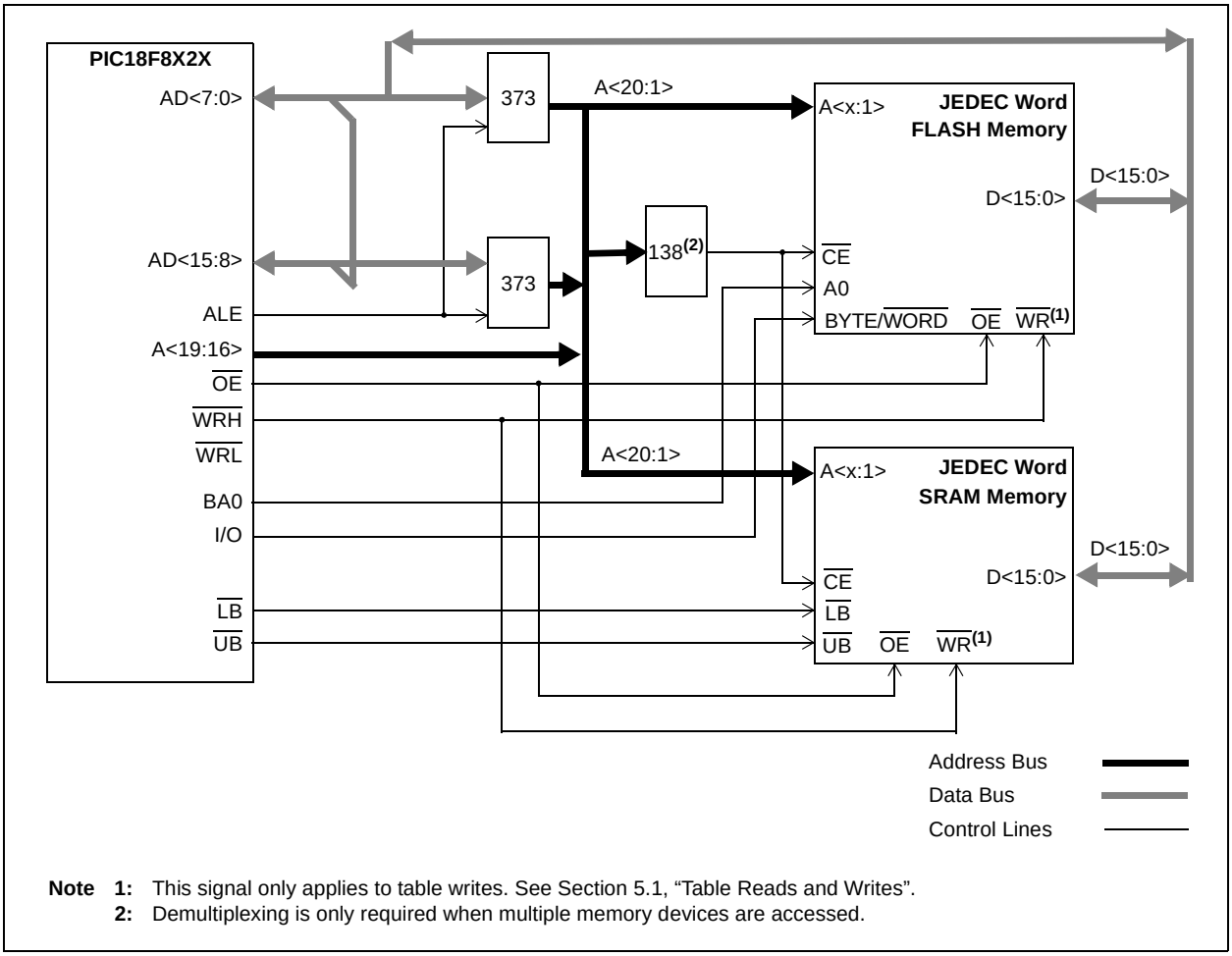
6.2.3 16-BIT BYTE SELECT MODE

Figure 6-3 shows an example of 16-bit Byte Select mode for PIC18F8X2X 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 TBLWT cycle, 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 Writes".
2: Demultiplexing is only required when multiple memory devices are accessed.

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)

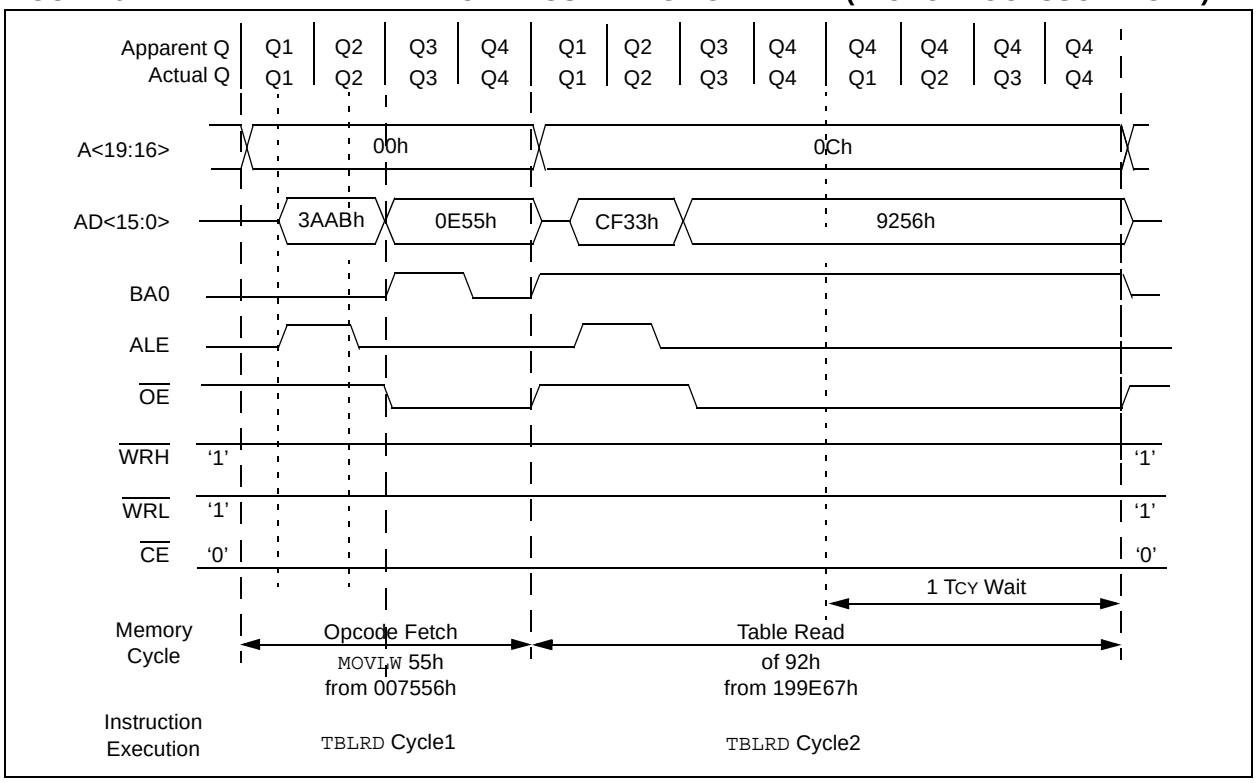


FIGURE 6-5: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)

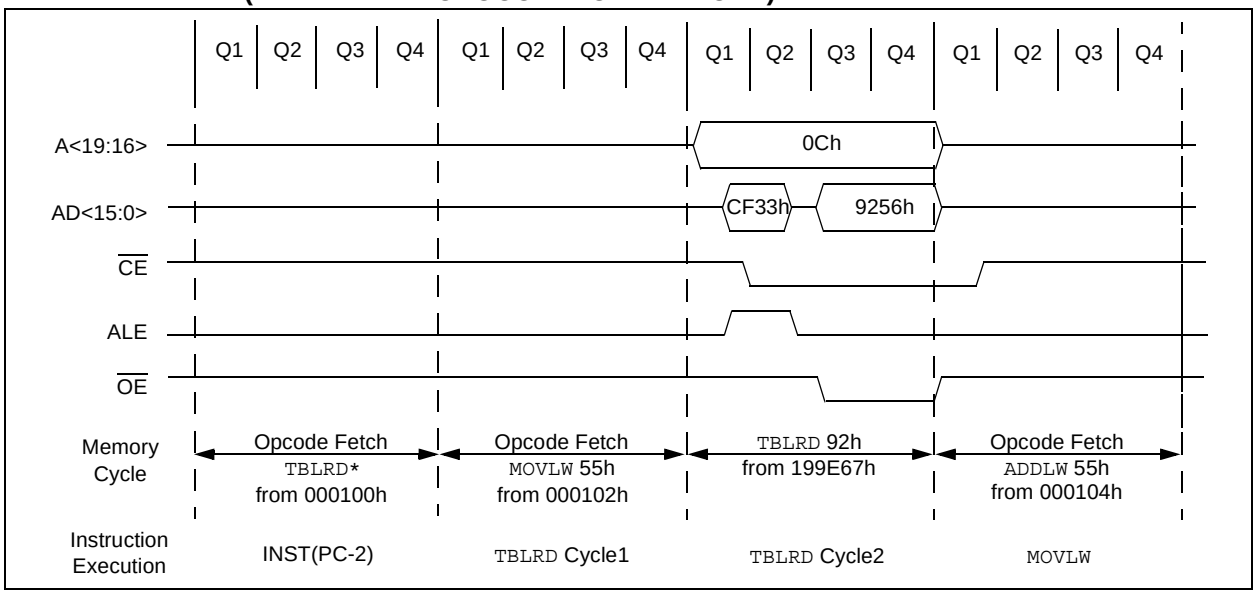
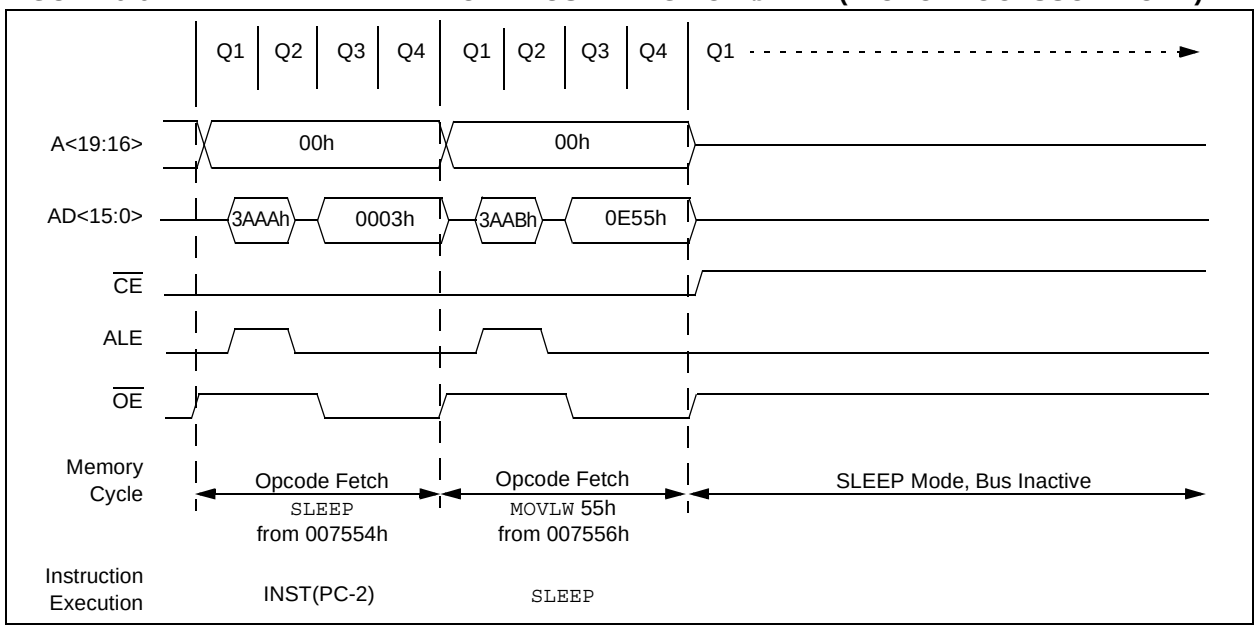


FIGURE 6-6: EXTERNAL MEMORY BUS TIMING FOR SLEEP (MICROPROCESSOR MODE)



NOTES:

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 (Section 27.0, “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 MSbits of the address are stored in EEADRH, while the remaining eight LSbits 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

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. The WREN bit must be set on a previous instruction. 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 device 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 24.0, “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
```


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:

8.0 8 X 8 HARDWARE MULTIPLIER

8.1 Introduction

An 8 x 8 hardware multiplier is included in the ALU of the PIC18F6X2X/8X2X 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	24	24	2.4 µs	9.6 µs	24 µs
16 x 16 signed	Without hardware multiply	52	254	25.4 µs	102.6 µs	254 µs
	Hardware multiply	36	36	3.6 µs	14.4 µs	36 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

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

EXAMPLE 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

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

EXAMPLE 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
      :
```

9.0 INTERRUPTS

The PIC18F6X2X/8X2X 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.

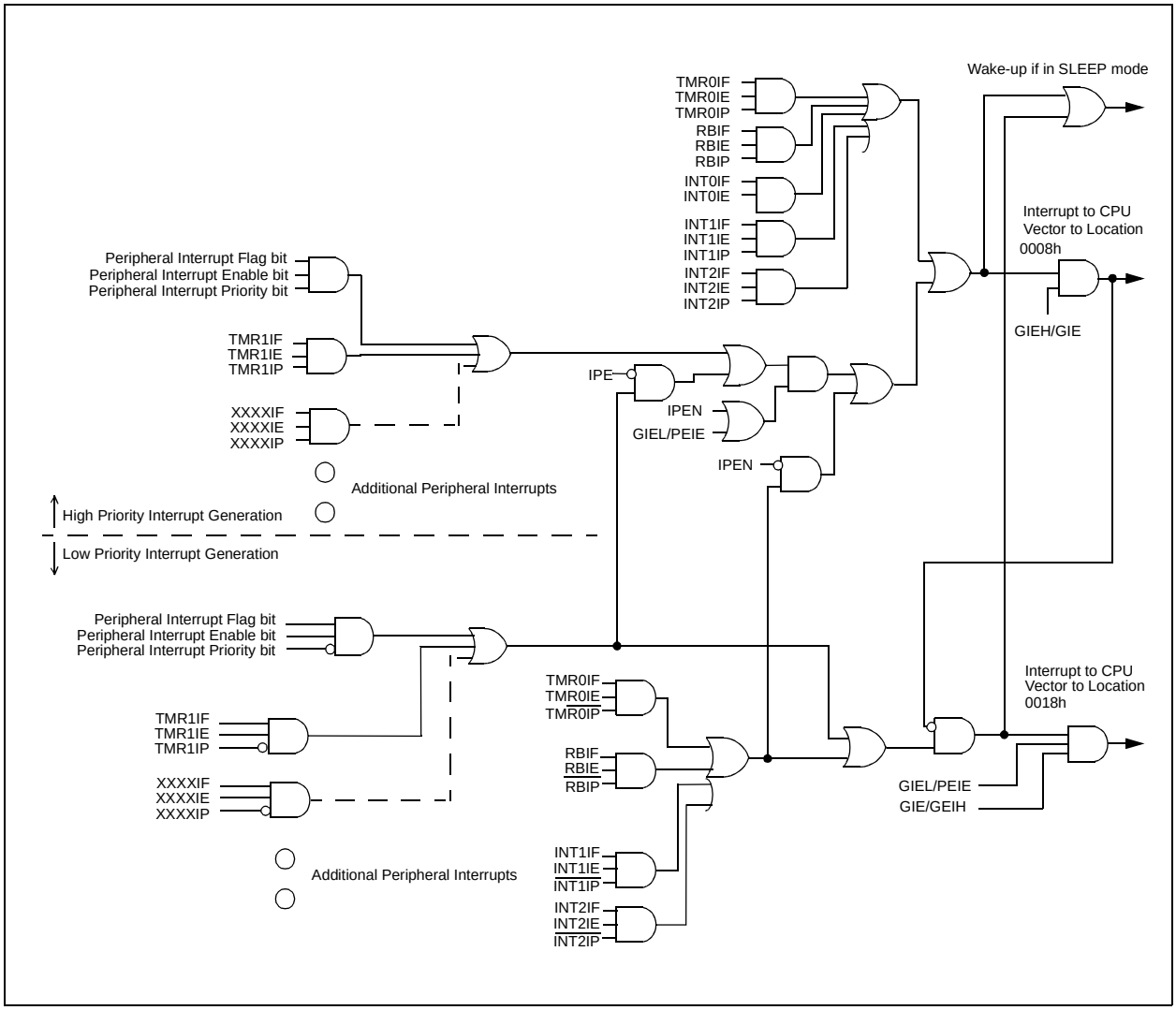
When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High priority interrupt sources can interrupt a low priority interrupt.

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



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.

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: 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

Note 1: Enabled only in Microcontroller mode for PIC18F8X2X devices.

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, RCREGx, is full (cleared when RCREGx is read)
0 = The USART1 receive buffer is empty

bit 4 **TX1IF**: USART Transmit Interrupt Flag bit
1 = The USART1 transmit buffer, TXREGx, is empty (cleared when TXREGx 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

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 9-5: 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, RCREGx, is full (cleared when RCREGx is read)
0 = The USART2 receive buffer is empty

bit 4 **TX2IF**: USART2 Transmit Interrupt Flag bit
1 = The USART2 transmit buffer, TXREGx, is empty (cleared when TXREGx 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 (ECCP3, CCP4 and CCP5)
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

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

Note 1: Enabled only in Microcontroller mode for PIC18F8X2X devices.

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

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

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 (ECCP3, CCP4 and CCP5)
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

Note 1: Enabled only in Microcontroller mode for PIC18F8X2X devices.

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

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 9-11: 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 (ECCP3, CCP4 and CCP5)
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

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.

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 **$\overline{RI}$:** RESET Instruction 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 INT1, INT2 and INT3 is determined by the value contained in the interrupt priority bits: INT1IP (INTCON3<6>), INT2IP (INTCON3<7>) and INT3IP (INTCON2<1>). There is no priority bit associated with INT0; it is always a high priority interrupt source.

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 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), 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
```


10.0 I/O PORTS

Depending on the device selected, there are either seven or nine I/O ports available on PIC18F6X2X/8X2X 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.

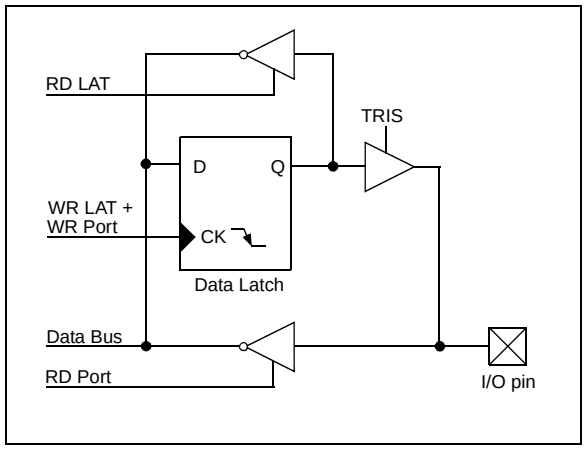
Each port has three registers for its operation. These registers are:

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

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, bi-directional 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 VREF+ and VREF- inputs. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register 1).

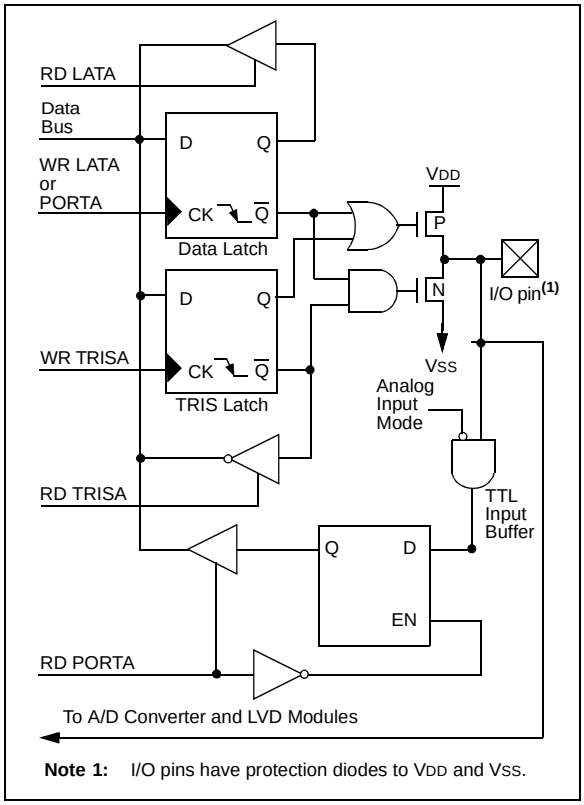
Note:	On a Power-on Reset, RA5 and RA3:RA0 are configured as analog inputs and read as '0'. RA6 and RA4 are configured as digital inputs.

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

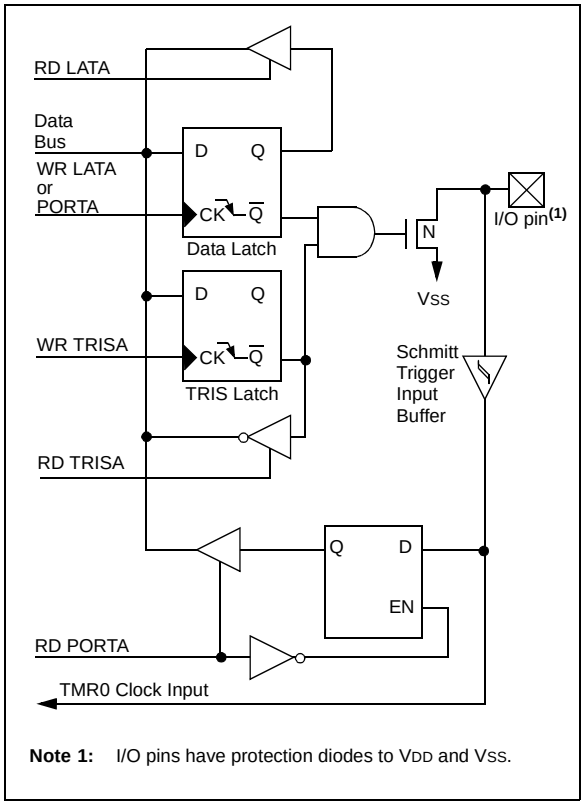
```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x0F    ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   0xCF    ; Value used 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



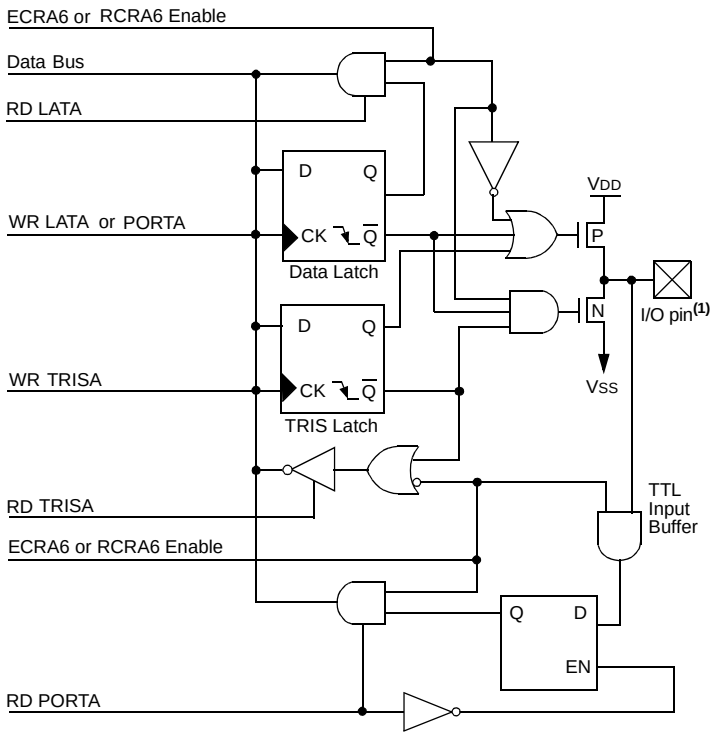
Note 1: I/O pins have protection diodes to VDD and VSS.

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.

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, bi-directional 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 ORed 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 MOVFF instruction). This will end the mismatch condition.

b) Clear flag bit RBIF.

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

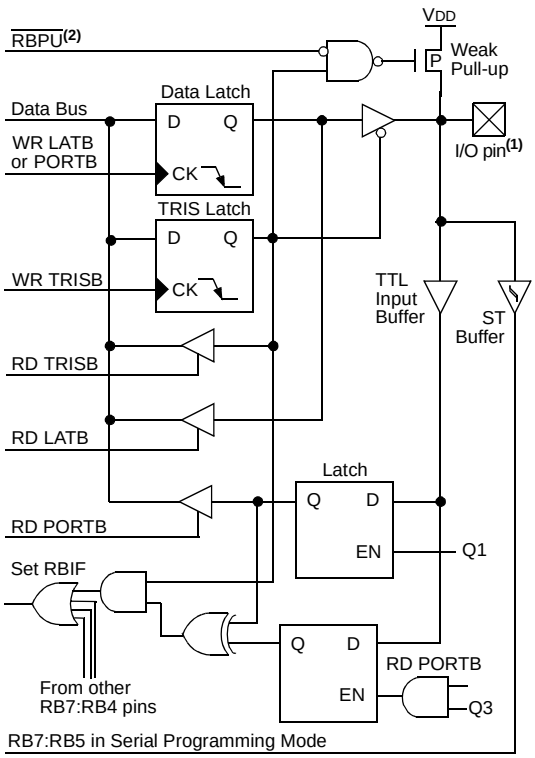
The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

For PIC18F8X2X devices, 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.

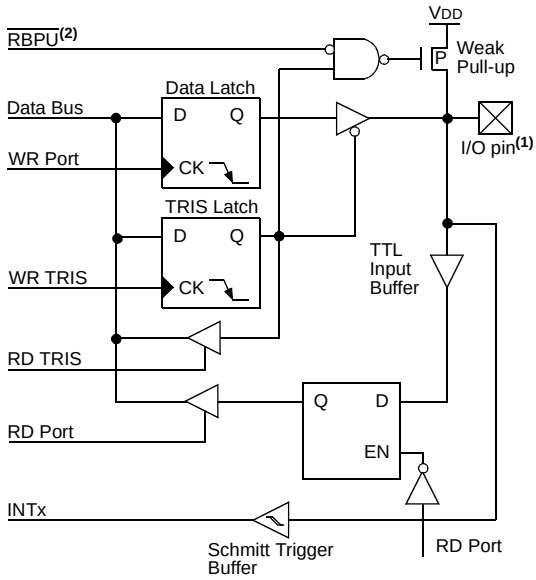
FIGURE 10-5: BLOCK DIAGRAM OF RB7:RB4 PINS



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

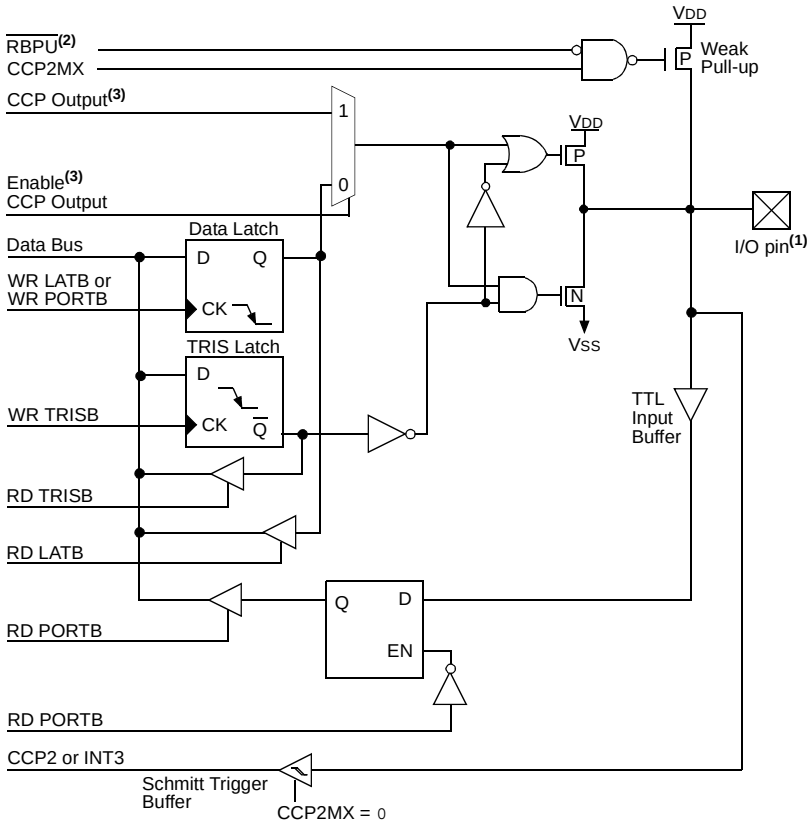
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit (INTCON2<7>).

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: For PIC18F8X2X parts, 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)/P2A(3)	bit 3	TTL/ST(4)	Input/output pin, external interrupt input 3, Capture2 input/Compare2 output/PWM2 output, or Enhanced PWM output P2A. 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: Valid for PIC18F8X2X devices in all operating modes except Microcontroller mode when CCP2MX is not set. RC1 is the default assignment for CCP2/PA2 when CCP2MX is set in all devices; RE7 is the alternate assignment for PIC18F8X2X devices in Microcontroller mode when CCP2MX is clear.

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	$\overline{\text{RBPU}}$	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.

10.3 PORTC, TRISC and LATC Registers

PORTC is an 8-bit wide, bi-directional 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.

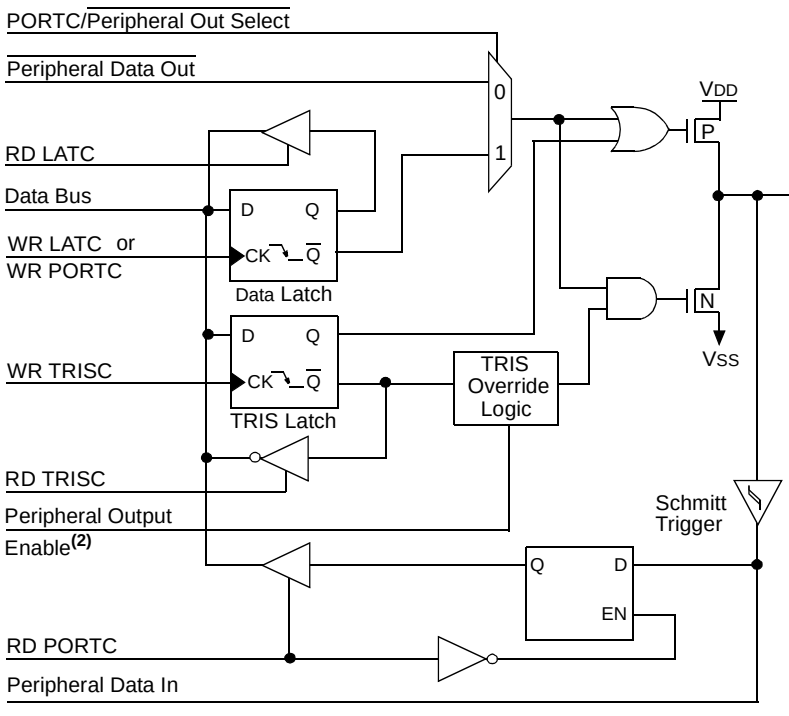
The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register without concern due to peripheral overrides.

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)/ P2A(1)	bit 1	ST	Input/output port pin, Timer1 oscillator input, standard Capture2 input/Compare2 output/PWM output, or Enhanced PWM output P2A.
RC2/CCP1/P1A	bit 2	ST	Input/output port pin, Capture1 input/Compare1 output/PWM1 output, or Enhanced PWM output P1A.
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: Valid when CCP2MX is set in all devices and in all operating modes (default). RE7 is the alternate assignment for CCP2/P2A for all PIC18F6X2X devices and PIC18F8X2X devices in Microcontroller modes when CCP2MX is not set; RB3 is the alternate assignment for PIC18F8X2X devices in all other operating modes.

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

10.4 PORTD, TRISD and LATD Registers

PORTD is an 8-bit wide, bi-directional 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.

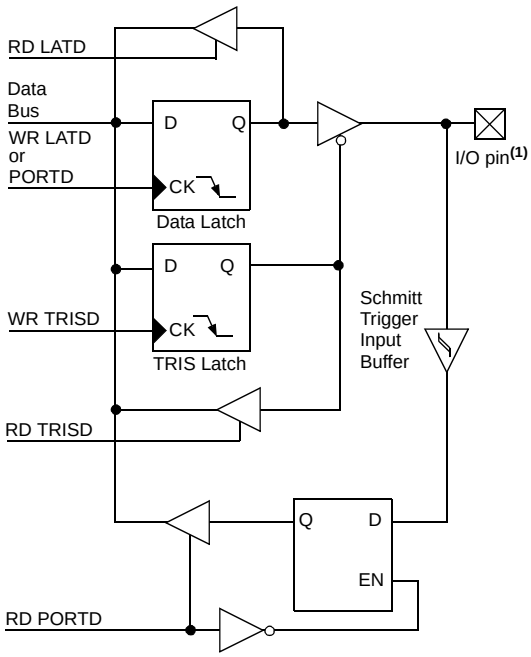
PORTD is multiplexed with the system bus as the external memory interface; I/O port functions are only available when the system bus is disabled by setting the EBDIS bit in the MEMCOM register (MEMCON<7>). When operating as the external memory interface, PORTD is the low order byte of the multiplexed 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 for additional information on the Parallel Slave Port (PSP).

EXAMPLE 10-4: INITIALIZING PORTD

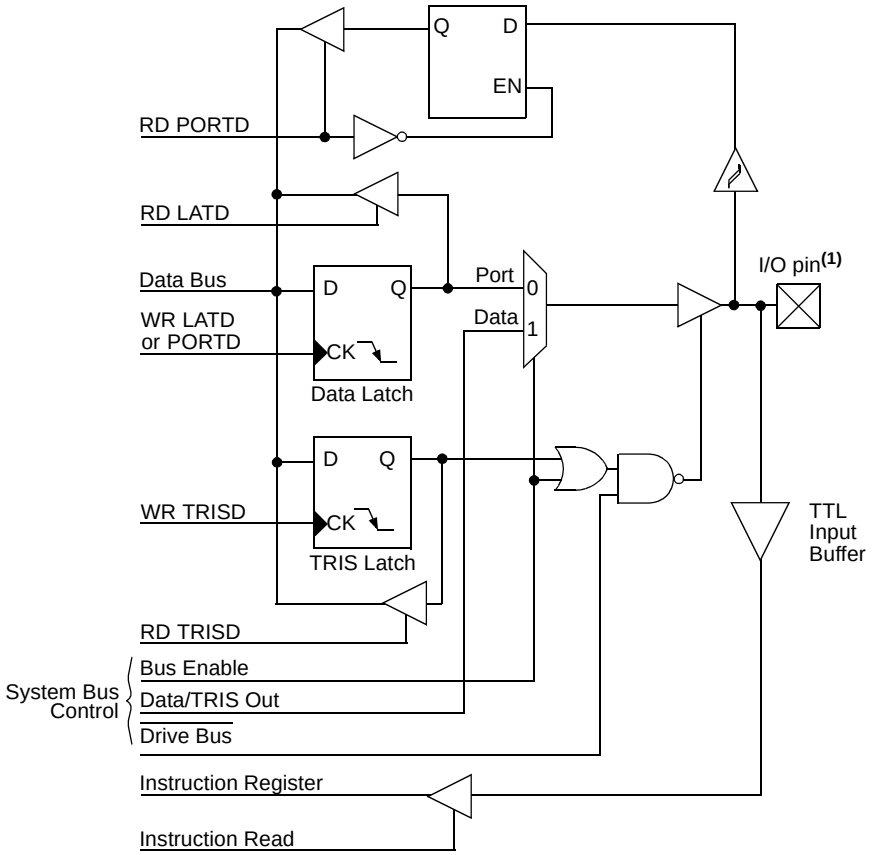
```
CLRF   PORTD   ; Initialize PORTD by
               ; clearing output
               ; data latches
CLRF   LATD    ; Alternate method
               ; to clear output
               ; data latches
MOVLW  0xCF    ; Value used to
               ; initialize data
               ; direction
MOVWF  TRISD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

FIGURE 10-9: PORTD BLOCK DIAGRAM 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.

TABLE 10-7: PORTD FUNCTIONS

Name	Bit#	Buffer Type	Function
RD0/AD0/PSP0	bit 0	ST/TTL(1)	Input/output port pin, address/data bus bit 0 or parallel slave port bit 0.
RD1/AD1/PSP1	bit 1	ST/TTL(1)	Input/output port pin, address/data bus bit 1 or parallel slave port bit 1.
RD2/AD2/PSP2	bit 2	ST/TTL(1)	Input/output port pin, address/data bus bit 2 or parallel slave port bit 2.
RD3/AD3/PSP3	bit 3	ST/TTL(1)	Input/output port pin, address/data bus bit 3 or parallel slave port bit 3.
RD4/AD4/PSP4	bit 4	ST/TTL(1)	Input/output port pin, address/data bus bit 4 or parallel slave port bit 4.
RD5/AD5/PSP5	bit 5	ST/TTL(1)	Input/output port pin, address/data bus bit 5 or parallel slave port bit 5.
RD6/AD6/PSP6	bit 6	ST/TTL(1)	Input/output port pin, address/data bus bit 6 or parallel slave port bit 6.
RD7/AD7/PSP7	bit 7	ST/TTL(1)	Input/output port pin, address/data bus bit 7 or parallel slave port 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, bi-directional 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 PIC18F8X2X 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 for more information on Program Memory modes.)

When the parallel slave port is active, three PORTE pins (RE0/AD8/$\overline{RD}$, RE1/AD9/$\overline{WR}$, and RE2/AD10/$\overline{CS}$) 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.

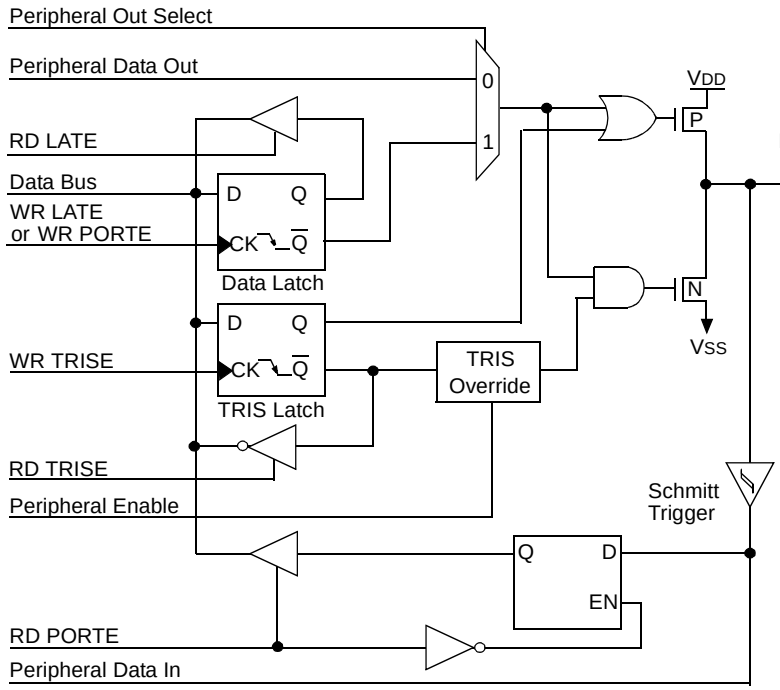
Pin RE7 can be configured as the alternate peripheral pin for the CCP2 module when the device is operating in Microcontroller mode. This is done by clearing the configuration bit CCP2MX in the CONFIG3H Configuration register (CONFIG3H<0>).

Note:	For PIC18F8X2X (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
                ; initialize data
                ; direction
MOVWF   TRISE   ; Set RE1:RE0 as inputs
                ; RE7:RE2 as outputs
```


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

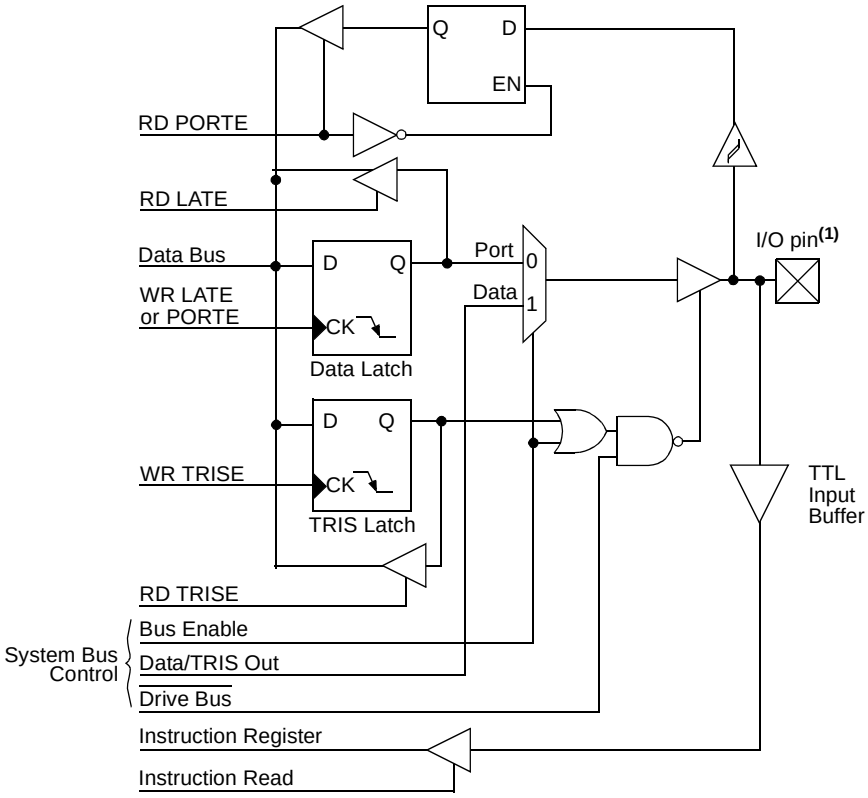


TRIS OVERRIDE

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/AD8/$\overline{RD}$/P2D	bit 0	ST/TTL(1)	Input/output port pin, address/data bit 8, Read control for parallel slave port, or Enhanced PWM2 output P2D For $\overline{RD}$ (PSP Control mode): 1 = Not a read operation 0 = Read operation, reads PORTD register (if chip selected)
RE1/AD9/$\overline{WR}$/P2C	bit 1	ST/TTL(1)	Input/output port pin, address/data bit 9, Write control for parallel slave port, or Enhanced PWM2 output P2C For $\overline{WR}$ (PSP Control mode): 1 = Not a write operation 0 = Write operation, writes PORTD register (if chip selected)
RE2/AD10/$\overline{CS}$/P2B	bit 2	ST/TTL(1)	Input/output port pin, address/data bit 10, Chip Select control for parallel slave port, or Enhanced PWM2 output P2B For $\overline{CS}$ (PSP Control mode): 1 = Device is not selected 0 = Device is selected
RE3/AD11/P3C(2)	bit 3	ST/TTL(1)	Input/output port pin, address/data bit 11 or Enhanced PWM3 output P3C.
RE4/AD12/P3B(2)	bit 4	ST/TTL(1)	Input/output port pin, address/data bit 12 or Enhanced PWM3 output P3B.
RE5/AD13/P1C(2)	bit 5	ST/TTL(1)	Input/output port pin, address/data bit 13 or Enhanced PWM1 output P1C.
RE6/AD14/P1B(2)	bit 6	ST/TTL(1)	Input/output port pin, address/data bit 14 or Enhanced PWM1 output P1B.
RE7/AD15/CCP2(3)/ P2A(3)	bit 7	ST/TTL(1)	Input/output port pin, address/data bit 15, Capture2 input/Compare2 output/PWM output or Enhanced PWM output P2A.

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

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

2: Valid for all PIC18F6X2X devices and PIC18F8X2X devices when ECCPMX is set. Alternate assignments for P1B/P1C/P3B/P3C are RH7, RH6, RH5 and RH4, respectively.

3: Valid for all PIC18F6X2X devices and PIC18F8X2X devices in Microcontroller mode when CCP2MX is not set. RC1 is the default assignment for CCP2/P2A for all devices in Microcontroller mode when CCP2MX is set; RB3 is the alternate assignment for PIC18F8X2X devices in operating modes except Microcontroller mode when CCP2MX is not set.

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.

10.6 PORTF, LATF, and TRISF Registers

PORTF is an 8-bit wide, bi-directional 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.

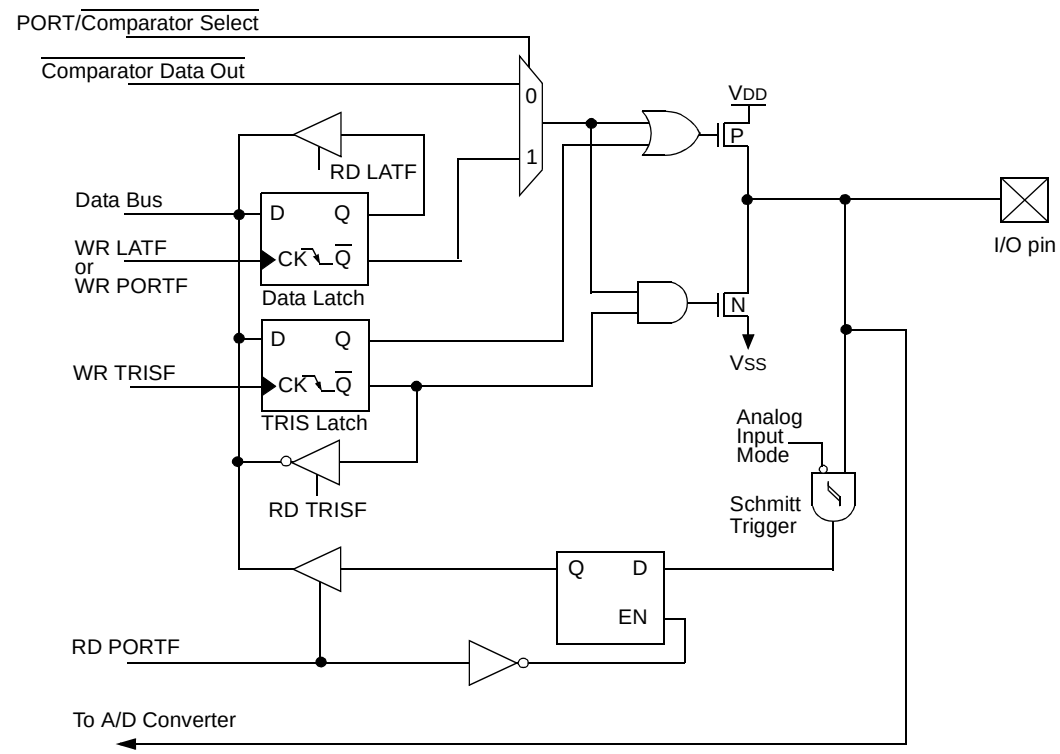
PORTF is multiplexed with several analog peripheral functions, including the A/D converter inputs and comparator inputs, outputs, and voltage reference.

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

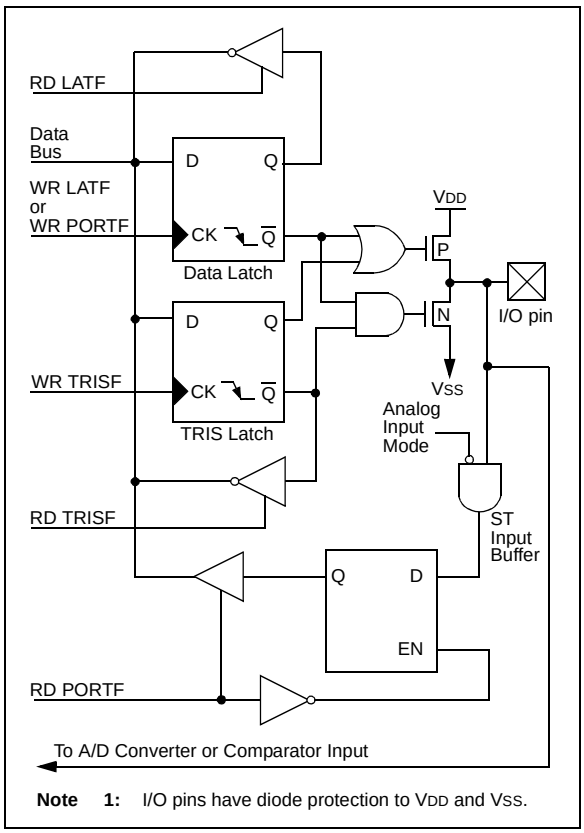
```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x07    ;
MOVWF   CMCON   ; Turn off comparators
MOVLW   0x0F    ;
MOVWF   ADCON1  ; Set 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/AN7/C1OUT PINS BLOCK DIAGRAM



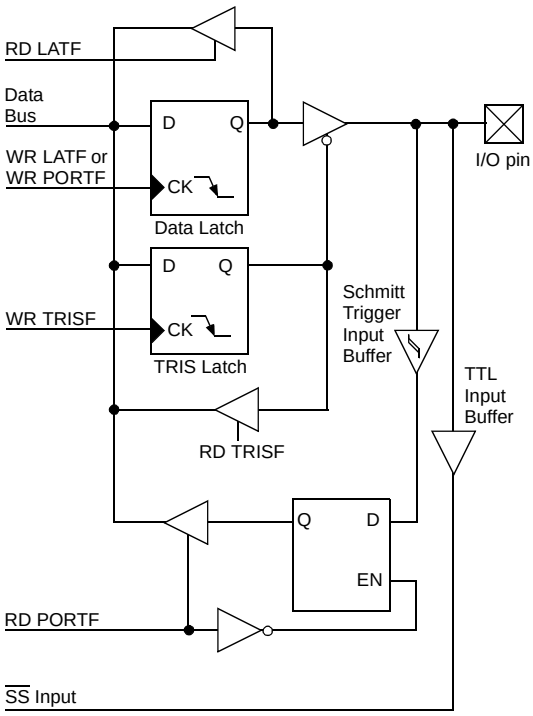
Note 1: I/O pins have diode protection to VDD and VSS.

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.

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/ CV_{REF}	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 6-bit wide port with 5 bi-directional pins (RG0:RG4) and one optional input only pin (RG5). 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 operations of the TRIS register without concern due to peripheral overrides.

The sixth pin of PORTG ($\overline{MCLR}$/VPP/RG5) is a digital input pin. Its operation is controlled by the MCLRE configuration bit in Configuration Register 3H (CONFIG3H<7>). In its default configuration (MCLRE = 1), the pin functions as the device Master Clear input. When selected as a port pin (MCLRE = 0), it functions as an input only pin; as such, it does not have TRISG or LATG bits associated with it.

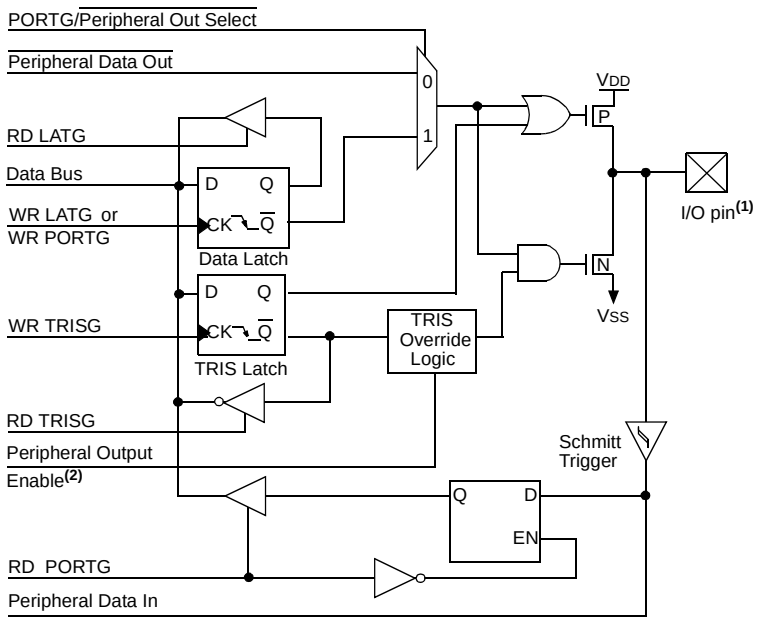
In either configuration, RG5 also functions as the programming voltage input during device programming.

Note 1:	On a Power-on Reset, RG5 is enabled as a digital input only if Master Clear functionality is disabled (MCLRE = 0).
2:	If the device Master Clear is disabled, verify that either of the following is done to ensure proper entry into ICSP mode: a.) disable low voltage programming (CONFIG4L<2> = 0); or b.) make certain that RB5/KBI1/PGM is held low during entry into ICSP.

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.

FIGURE 10-17: $\overline{MCLR}$/VPP/RG5 PIN BLOCK DIAGRAM

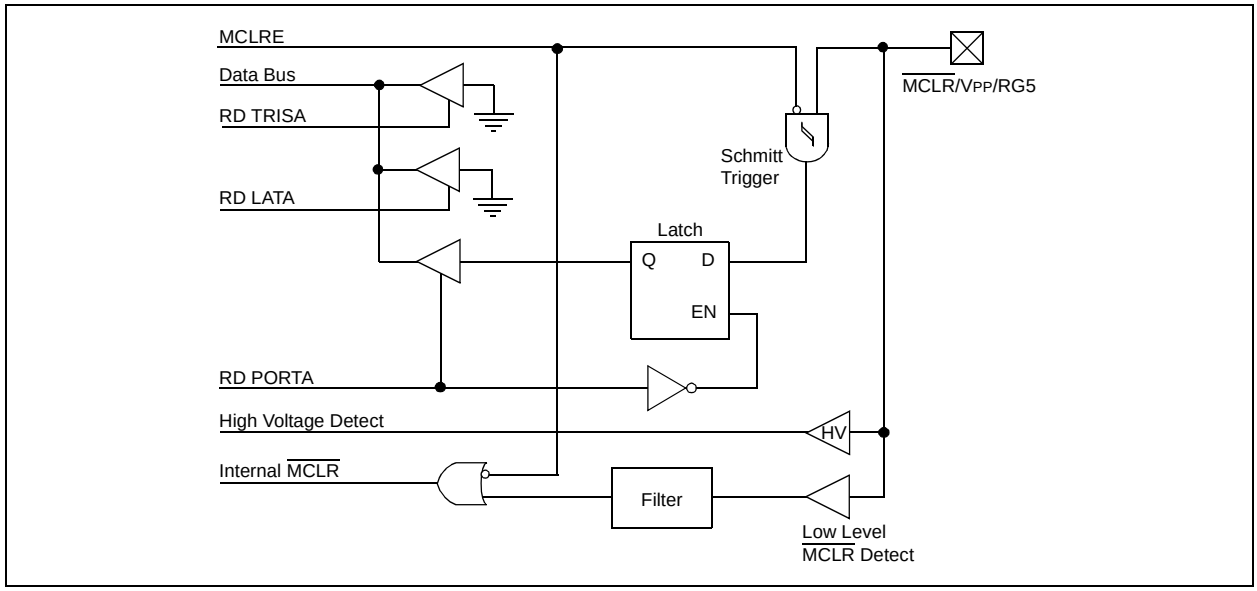


TABLE 10-13: PORTG FUNCTIONS

Name	Bit#	Buffer Type	Function
RG0/CCP3/P3A	bit 0	ST	Input/output port pin, Capture3 input/Compare3 output/PWM3 output, or Enhanced PWM3 output P3A.
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/P3D	bit 3	ST	Input/output port pin, Capture4 input/Compare4 output/PWM4 output, or Enhanced PWM3 output P3D.
RG4/CCP5/P1D	bit 4	ST	Input/output port pin, Capture5 input/Compare5 output/PWM5 output, or Enhanced PWM1 output P1D.
$\overline{MCLR}$/VPP/RG5	bit 5	ST	Master Clear input or programming voltage input (if $\overline{MCLR}$ is enabled). Input only port pin or programming voltage input (if $\overline{MCLR}$ is disabled).

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	—	—	RG5(1)	Read PORTG pin/Write PORTG Data Latch					--0x xxxx	--0u 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

Note 1: RG5 is available as an input only when $\overline{MCLR}$ is disabled.

10.8 PORTH, LATH, and TRISH Registers

Note:	PORTH is available only on PIC18F8X2X devices.

PORTH is an 8-bit wide, bi-directional 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.

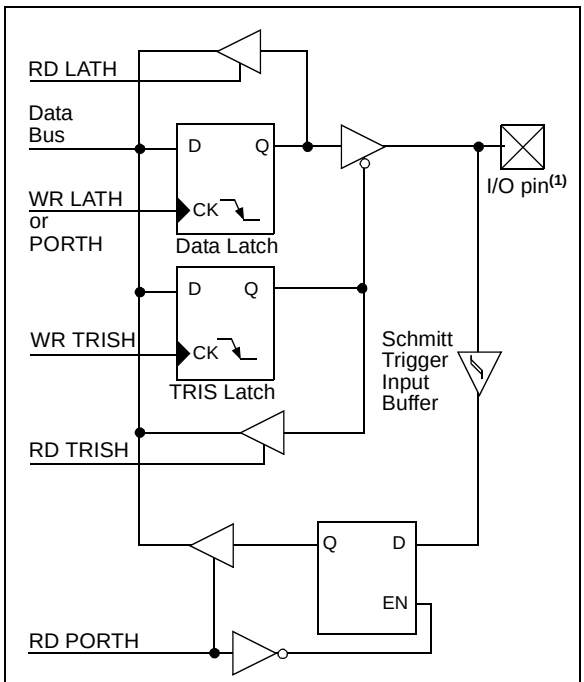
Pins RH7:RH4 are multiplexed with analog inputs AN15:AN12. Pins RH3:RH0 are multiplexed with the system bus as the external memory interface; they are the high order address bits A19:A16. By default, pins RH7:RH4 are enabled as A/D inputs and pins RH3:RH0 are enabled as the system address bus. Register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

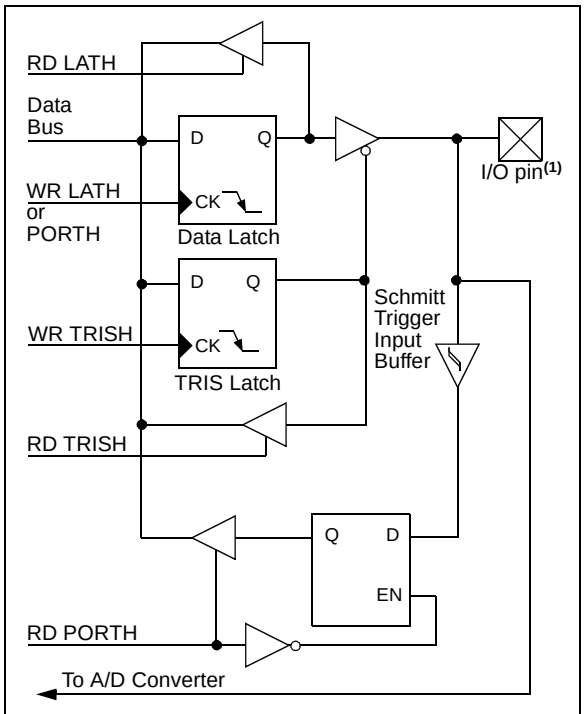
```
CLRF    PORTH   ; Initialize PORTH by
                ; clearing output
                ; data latches
CLRF    LATH    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0Fh     ;
MOVWF   ADCON1  ;
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISH   ; Set RH3:RH0 as inputs
                ; RH5:RH4 as outputs
                ; RH7:RH6 as inputs
```

FIGURE 10-18: RH3:RH0 PINS BLOCK DIAGRAM IN I/O MODE



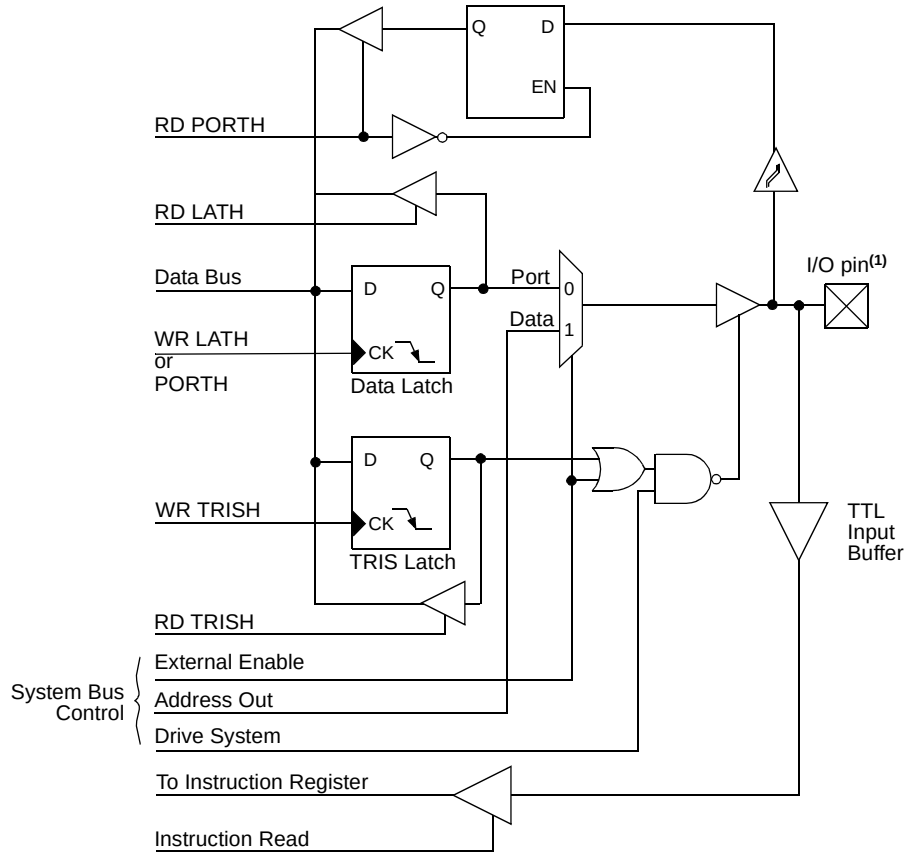
Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-19: RH7:RH4 PINS BLOCK DIAGRAM IN I/O MODE



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

FIGURE 10-20: RH3:RH0 PINS BLOCK 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/P3C(2)	bit 4	ST	Input/output port pin, analog input channel 12 or Enhanced PWM output P3C.
RH5/AN13/P3B(2)	bit 5	ST	Input/output port pin, analog input channel 13 or Enhanced PWM output P3B.
RH6/AN14/P1C(2)	bit 6	ST	Input/output port pin, analog input channel 14 or Enhanced PWM output P1C.
RH7/AN15/P1B(2)	bit 7	ST	Input/output port pin, analog input channel 15 or Enhanced PWM3 output P1B.

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.

2: Valid only for PIC18F8X2X devices when ECCPMX is not set. The alternate assignments for P1B/P1C/P3B/P3C in all PIC18F6X2X devices and in PIC18F8X2X devices when ECCPMX is set are RE6, RE5, RE4 and RE3, respectively.

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.

10.9 PORTJ, TRISJ and LATJ Registers

Note:	PORTJ is available only on PIC18F8X2X devices.

PORTJ is an 8-bit wide, bi-directional 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.

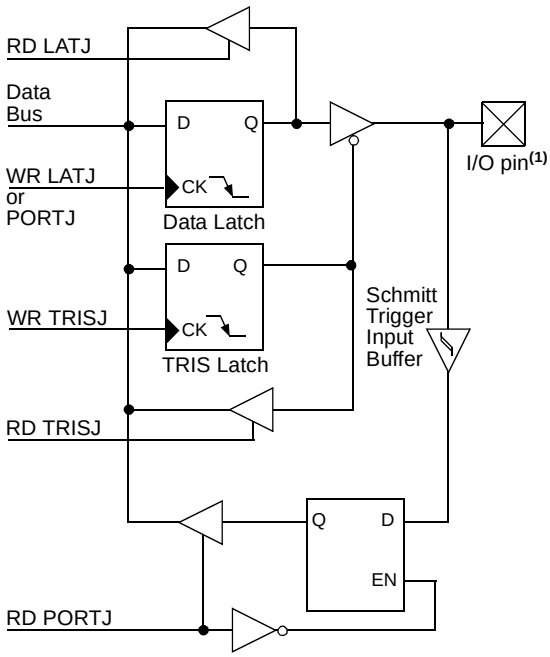
Note:	On a Power-on Reset, these pins are configured as digital inputs.

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

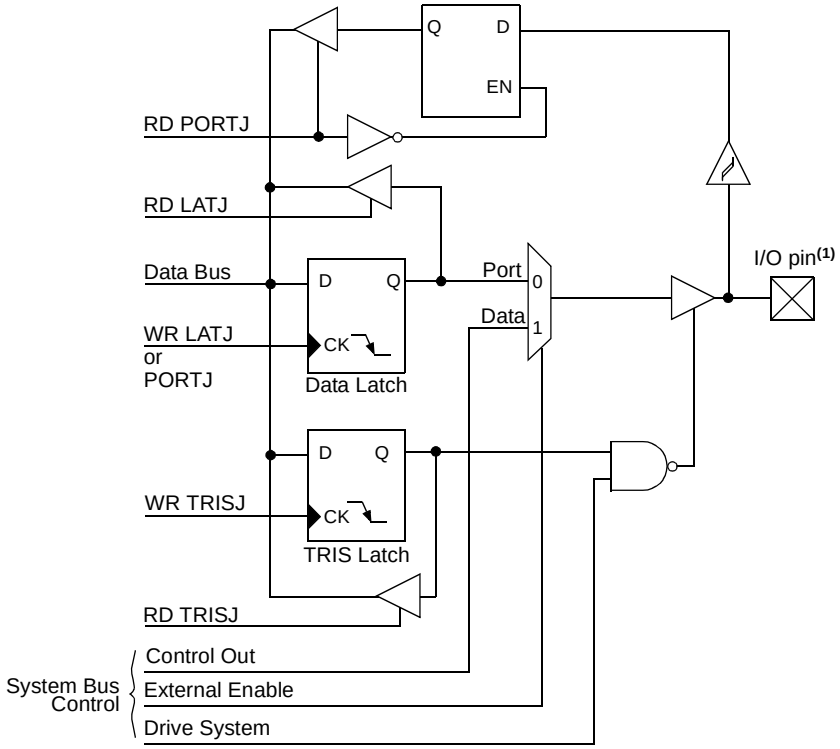
```
CLRF    PORTJ   ; Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATJ    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction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

FIGURE 10-21: PORTJ BLOCK DIAGRAM IN I/O MODE



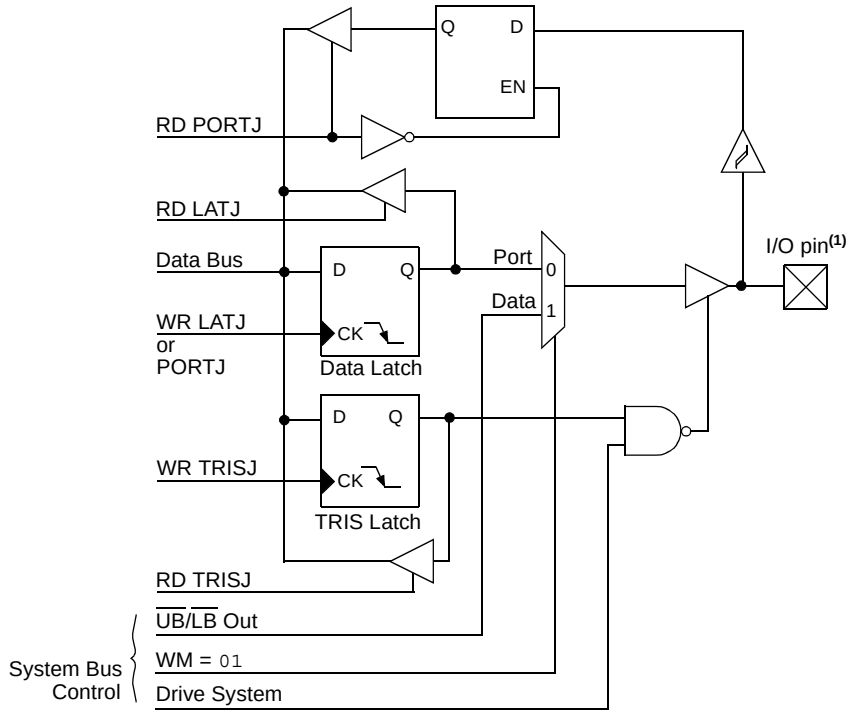
Note 1: I/O pins have diode protection to VDD and VSS.

FIGURE 10-22: RJ4:RJ0 PINS BLOCK DIAGRAM IN SYSTEM BUS MODE



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

FIGURE 10-23: RJ7:RJ6 PINS BLOCK DIAGRAM IN SYSTEM BUS MODE



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

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 $\overline{RD}$ control input pin, RE0/$\overline{RD}$ and $\overline{WR}$ control input pin, RE1/$\overline{WR}$.

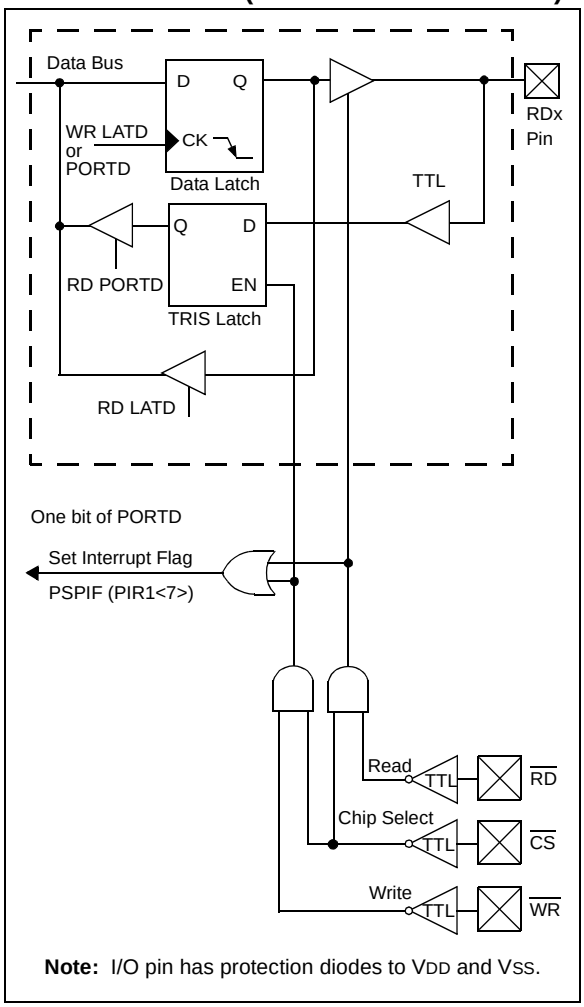
Note:	For PIC18F8X2X devices, the parallel slave port is available only in Microcontroller mode.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an 8-bit latch. Setting bit PSPMODE enables port pin RE0/$\overline{RD}$ to be the $\overline{RD}$ input, RE1/$\overline{WR}$ to be the $\overline{WR}$ input and RE2/$\overline{CS}$ to be the $\overline{CS}$ (chip select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (set). 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 10-24: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: I/O pin has protection diodes to V_{DD} and V_{SS}.

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 10-25: PARALLEL SLAVE PORT WRITE WAVEFORMS

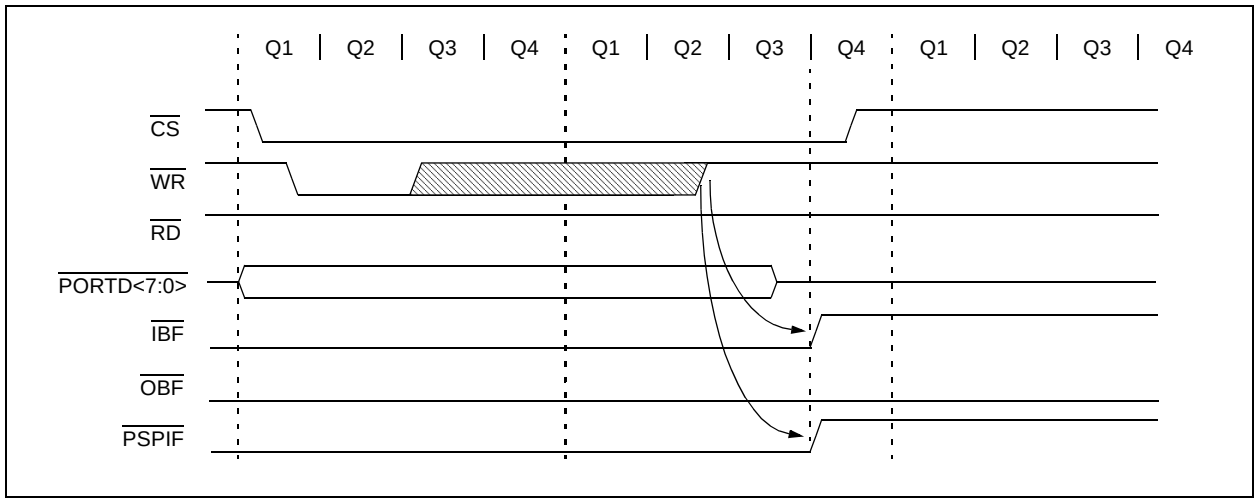


FIGURE 10-26: PARALLEL SLAVE PORT READ WAVEFORMS

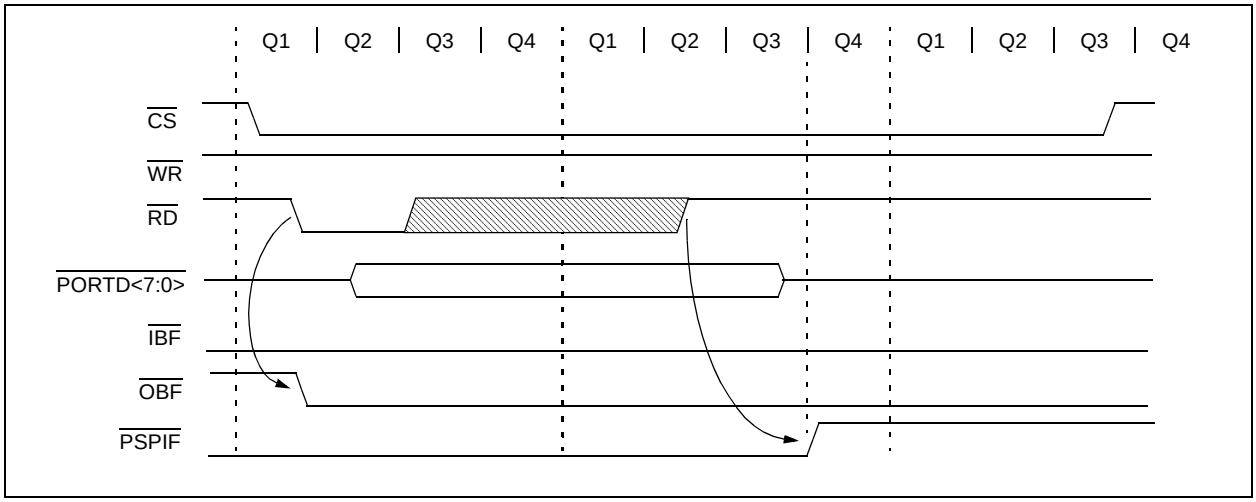


TABLE 10-19: REGISTERS ASSOCIATED WITH PARALLEL SLAVE PORT

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	1111 1111	1111 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 PIC18F8X2X devices.

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)

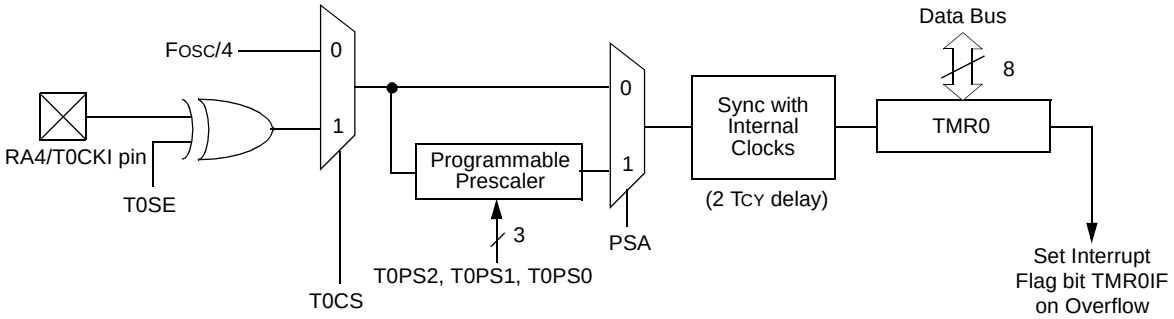
bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

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

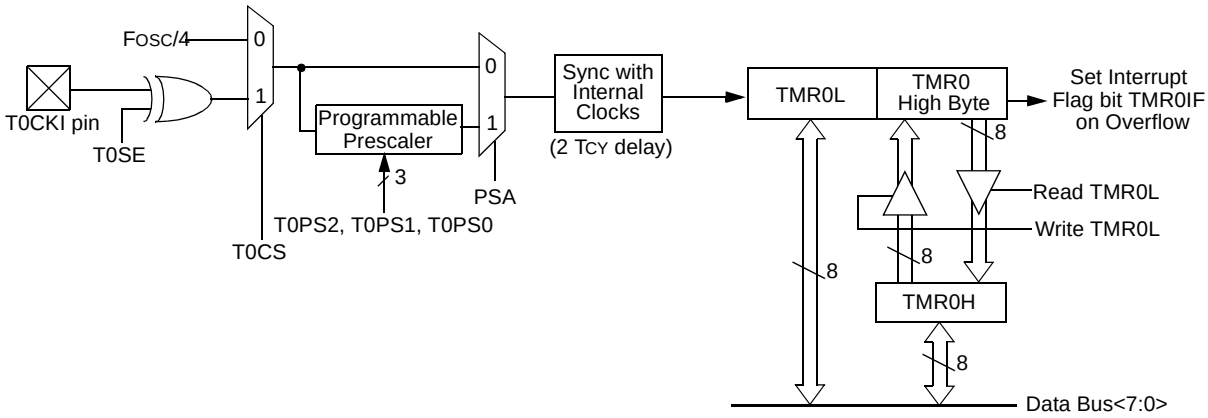
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

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.

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`, and so on) 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 TMR0IE 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:

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	$\overline{\text{T1SYNC}}$	$\overline{\text{TMR1CS}}$	$\overline{\text{TMR1ON}}$
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 **$\overline{\text{TMR1CS}}$:** Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T13CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **$\overline{\text{TMR1ON}}$:** 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

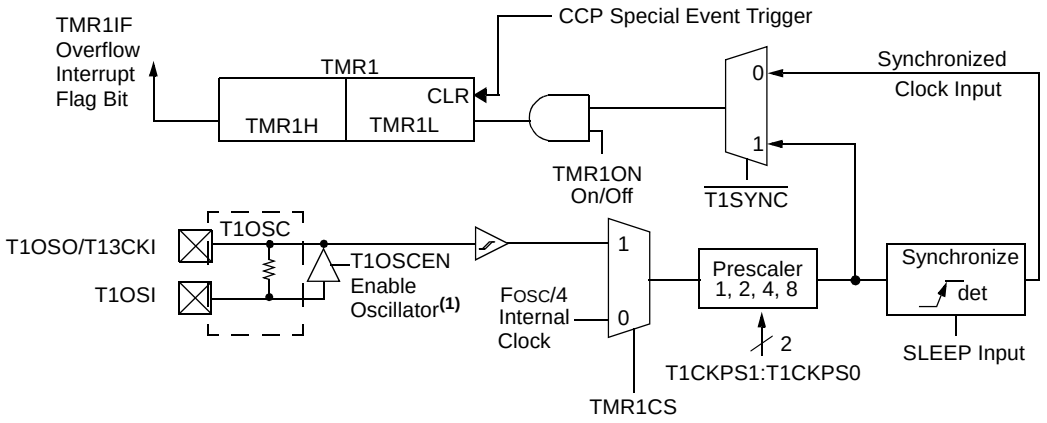
The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

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'.

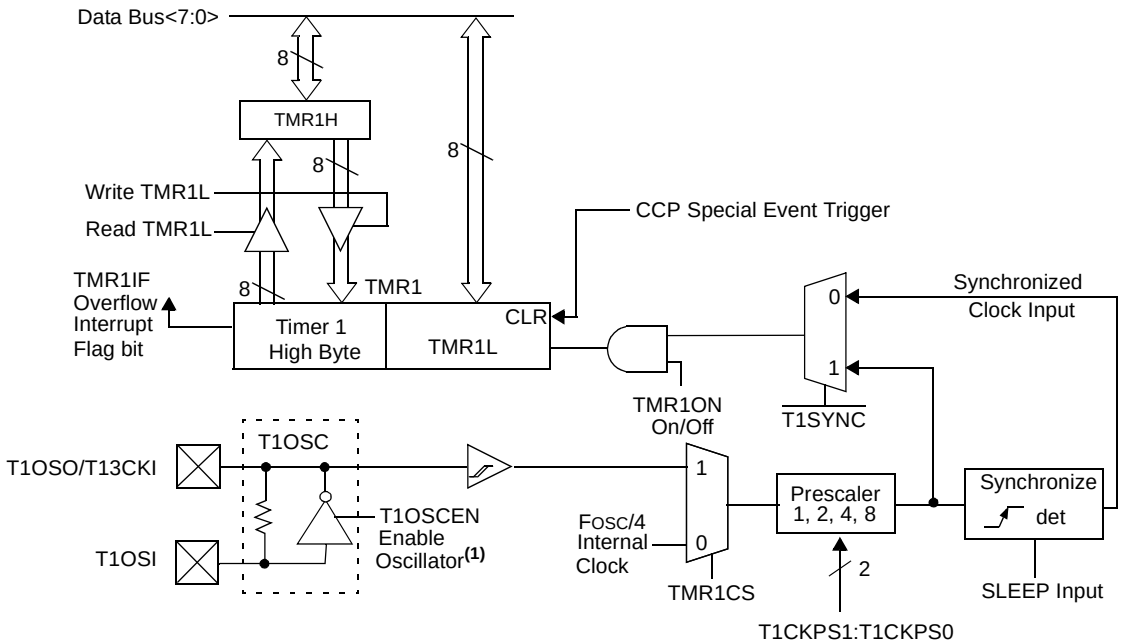
Timer1 also has an internal "RESET input". This RESET can be generated by ECCP1 or ECCP2 special event trigger. This is discussed in detail in Section 12.4.

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 DIAGRAM: 16-BIT READ/WRITE MODE



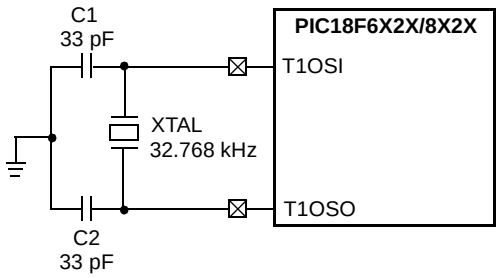
Note 1: When enable bit T1OSCEN is cleared, the inverter and feedback resistor are turned off. This eliminates power drain.

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.

12.3 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 an ECCP Special Trigger Output

If either the ECCP1 or ECCP2 module is configured in Compare mode to generate a “special event trigger” (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1. The trigger for ECCP2 will also 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 ECCP1, 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) 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 MSbit of TMR1H with a `BSF` instruction. Note that the TMR1L register is never preloaded or altered; doing so may introduce cumulative error over many cycles.

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

EXAMPLE 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	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	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:

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

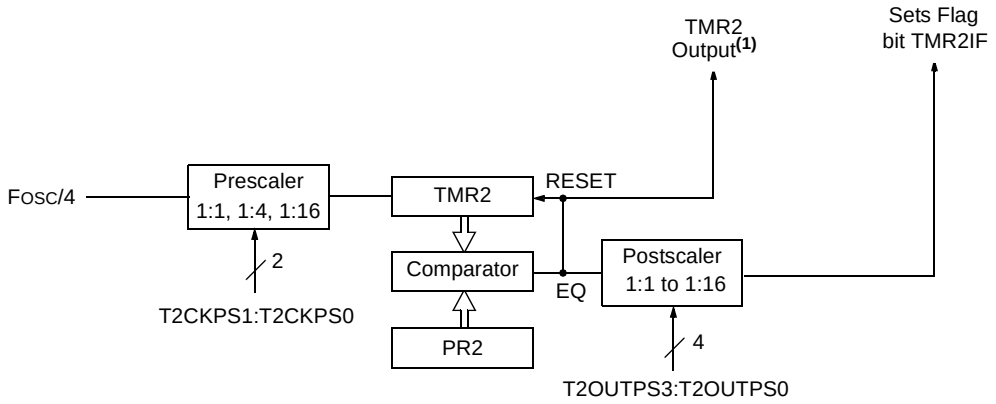
13.2 Timer2 Interrupt

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	1111 1111	1111 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.

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

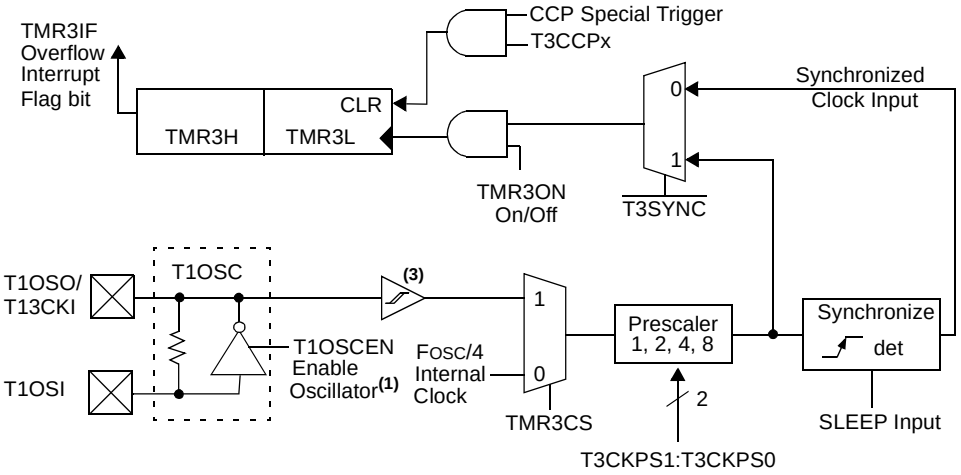
The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>).

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'.

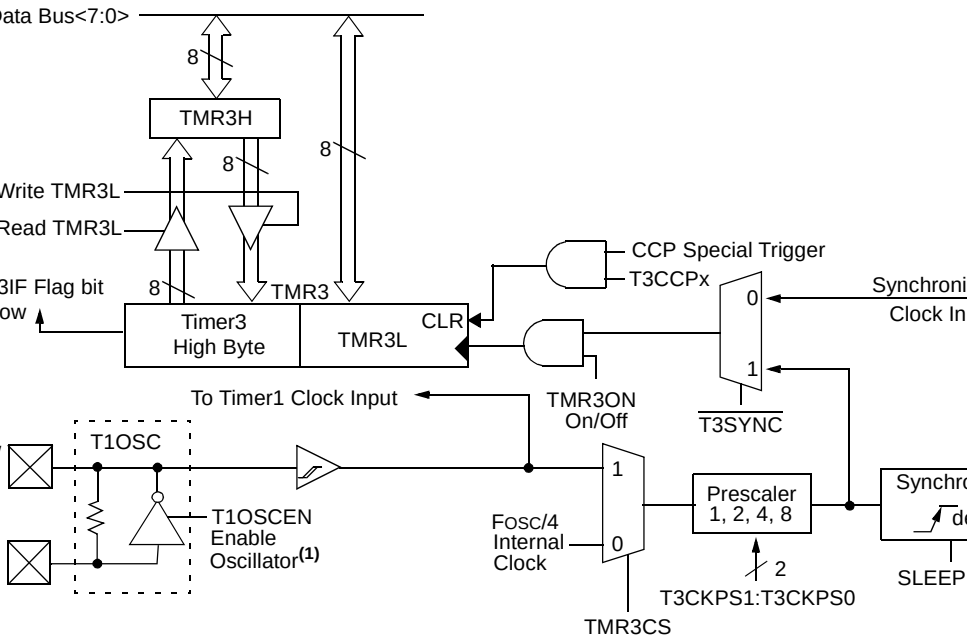
Timer3 also has an internal "RESET input". This RESET can be generated by the CCP module (Section 14.0).

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 CONFIGURED 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.

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 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 an ECCP Special Trigger Output

If either the ECCP1 or ECCP2 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 ECCP1, 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:

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

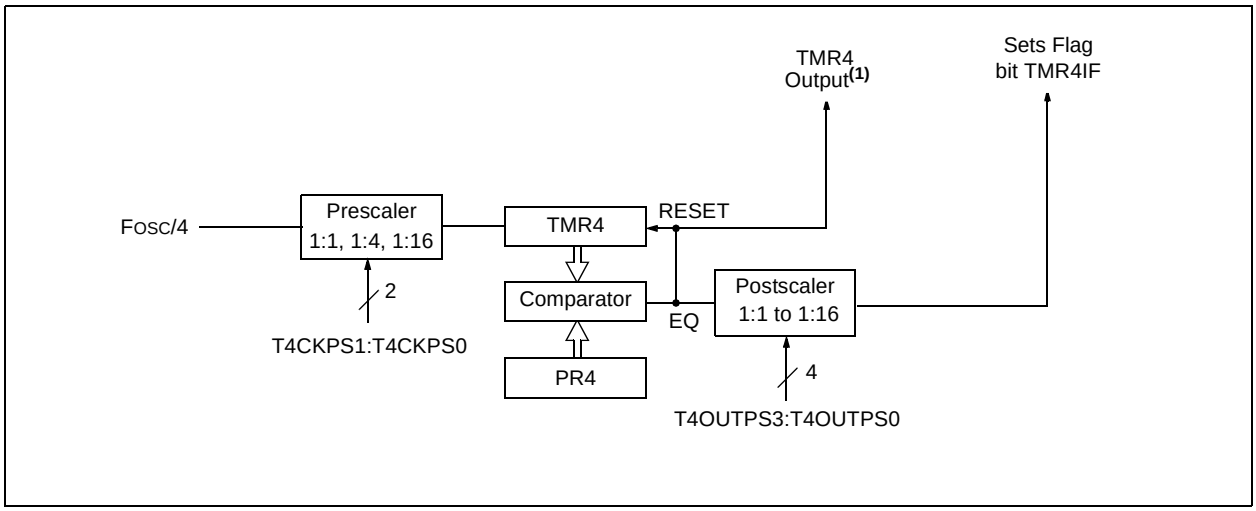


TABLE 15-1: REGISTERS ASSOCIATED WITH TIMER4 AS A TIMER/COUNTER

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.

16.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18F6X2X/8X2X devices all have a total of five CCP (Capture/Compare/PWM) modules. Two of these (CCP4 and CCP5) implement standard Capture, Compare and Pulse Width Modulation (PWM) modes, and are discussed in this section. The other three modules (ECCP1, ECCP2, ECCP3) implement standard Capture and Compare modes, as well as enhanced PWM modes. These are discussed in Section 17.0.

Each CCP/ECCP module contains a 16-bit register which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. For the sake of clarity, all CCP module operation in the following sections is described with respect to CCP4, but is equally applicable to CCP5.

Capture and Compare operations described in this chapter apply to all standard and enhanced CCP modules. The operations of PWM mode described in Section 16.4 applies to CCP4 and CCP5 only.

Note:	Throughout this section and Section 17.0, 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 CCP4 or CCP5, or ECCP1, ECCP2 or ECCP3. "CCPxCON" is used throughout these sections to refer to the module control register, regardless of whether the CCP module is a standard or enhanced implementation.

REGISTER 16-1: CCPxCON REGISTER (CCP4 AND CCP5 MODULES)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

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 LSbits (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight MSbits (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 reflects I/O state)
1011 = Reserved
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

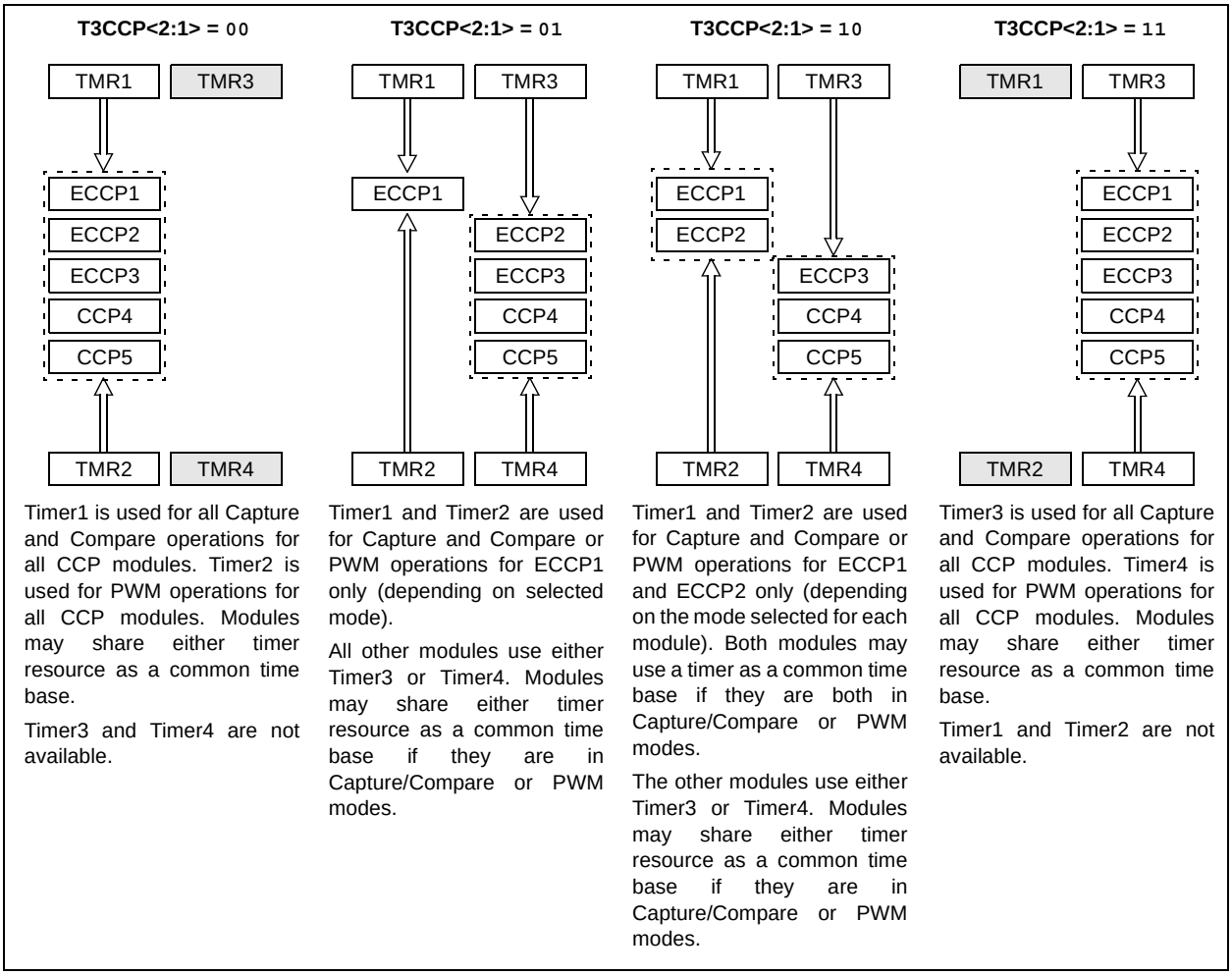
The CCP/ECCP modules utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode.

TABLE 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, page 145). 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 ECCP1 only (depending on selected mode).

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

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

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

Timer3 is used for all Capture and Compare operations for all 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.

16.2 Capture Mode

In Capture mode, the CCPR4H:CCPR4L register pair captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on pin RG3/CCP4. An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

The event is selected by the mode select bits, CCP4M3:CCP4M0 (CCP4CON<3:0>). When a capture is made, the interrupt request flag bit CCP4IF (PIR3<1>) is set; it must be cleared in software. If another capture occurs before the value in register CCPR4 is read, the old captured value is overwritten by the new captured value.

16.2.1 CCP PIN CONFIGURATION

In Capture mode, the RG3/CCP4 pin should be configured as an input by setting the TRISG<3> bit.

Note:	If the RG3/CCP4 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).

16.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP4IE (PIE3<1>) clear to avoid false interrupts and should clear the flag bit, CCP4IF, following any such change in operating mode.

16.2.4 CCP PRESCALER

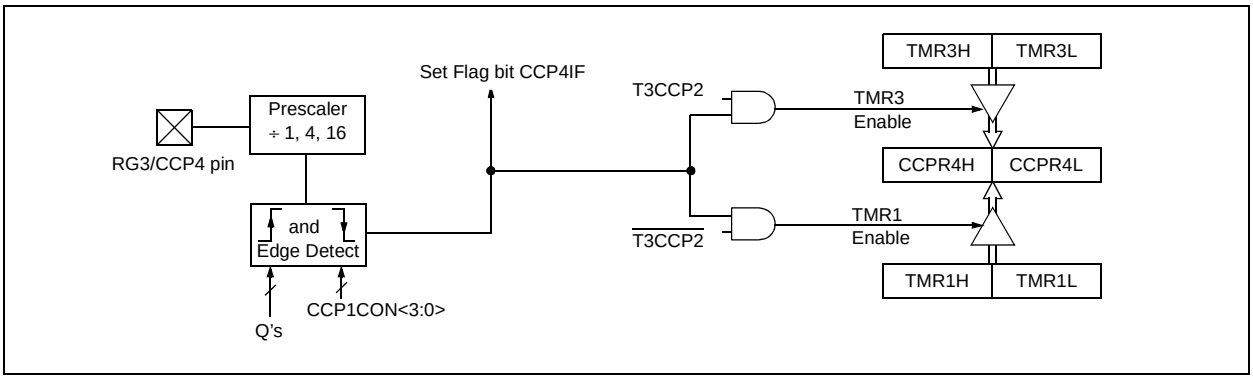
There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCP4M3:CCP4M0). Whenever the CCP module is turned off or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any RESET will clear 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    CCP4CON      ; Turn CCP module off
MOVLW   NEW_CAPT_PS  ; Load WREG with the
                     ; new prescaler mode
                     ; value and CCP ON
MOVWF   CCP4CON      ; 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 or TMR3 register pair value. When a match occurs, the CCP4 pin can be:

- driven high
- driven low
- toggled (high to low or low to high)
- remain unchanged (that is, reflects the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCP4M3:CCP4M0). At the same time, the interrupt flag bit CCP4IF 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 CCP4CON register will force the RG3/CCP4 compare output latch to the default low level. This is not the PORTG 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 the Generate Software Interrupt mode is chosen (CCP4M3:CCP4M0 = `1010`), the CCP4 pin is not affected. Only a CCP interrupt is generated if enabled and the CCP4IE bit is set.

16.3.4 SPECIAL EVENT TRIGGER

Although shown in Figure 16-3, the compare on match special event triggers are not implemented on CCP4 or CCP5; they are only available on ECCP1 and ECCP2. Their operation is discussed in detail in Section 17.2.1.

FIGURE 16-3: COMPARE MODE OPERATION BLOCK DIAGRAM

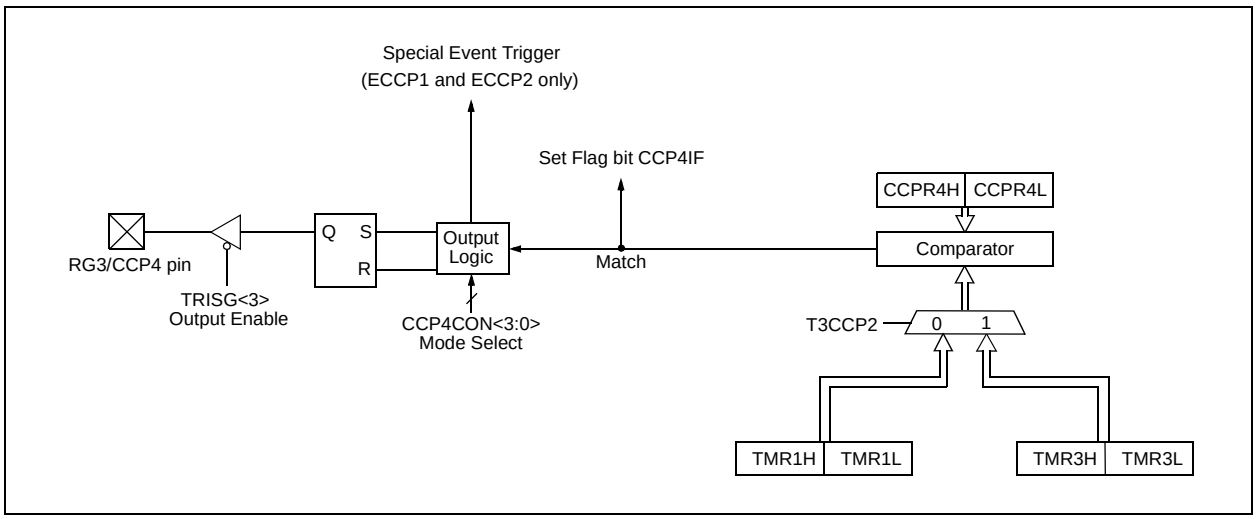


TABLE 16-2: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

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	1111 1111	1111 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 (CCP4 and CCP5). Bit assignments and RESET values for all registers of the same generic name are identical.

16.4 PWM Mode

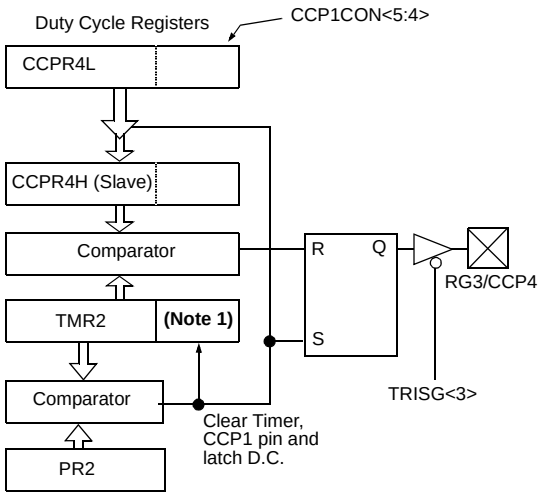
In Pulse Width Modulation (PWM) mode, the CCP4 pin produces up to a 10-bit resolution PWM output. Since the CCP4 pin is multiplexed with the PORTG data latch, the TRISG<3> bit must be cleared to make the CCP4 pin an output.

Note:	Clearing the CCP4CON register will force the CCP4 PWM output latch to the default low level. This is not the PORTG 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.

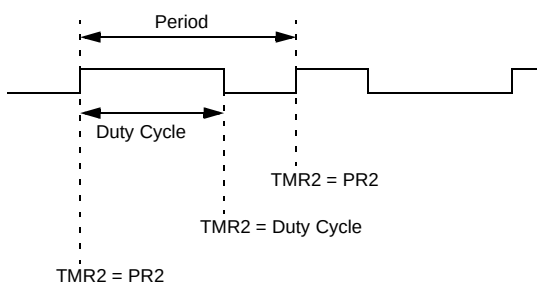
FIGURE 16-4: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer TMR2 or TMR4 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 (PR4) register. The PWM period can be calculated using the following formula:

$$\text{PWM period} = (PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

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

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

- TMR2 (TMR4) is cleared
- The CCP4 pin is set (exception: if PWM duty cycle = 0%, the CCP4 pin will not be set)
- The PWM duty cycle is latched from CCPR4L into CCPR4H

Note:	The Timer2 and Timer4 postscalers (see Section 13.0) 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 CCPR4L register and to the CCP4CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP4CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR4L:CCP4CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCPR4L:CCP4CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

CCPR4L and CCP4CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR4H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR4H is a read only register.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPR4H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP4 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

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

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP4 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. Select TMR2 or TMR4 by setting or clearing the T3CCP2:T3CCP1 bits in T3CON register.
2. Set the PWM period by writing to the PR2 or PR4 register
3. Set the PWM duty cycle by writing to the CCPR4L register and CCP4CON<5:4> bits.
4. Make the CCP4 pin an output by clearing the TRISG<3> bit.
5. Set TMR2 or TMR4 prescale value, enable Timer2 or Timer4 by writing to T2CON or T4CON.
6. Configure the CCP4 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	12	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	1111 1111	1111 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 (CCP4 and CCP5). Bit assignments and RESET values for all registers of the same generic name are identical.

NOTES:

17.0 ENHANCED CAPTURE/COMPARE/PWM (ECCP) MODULE

The enhanced CCP (ECCP) modules differ from the standard CCP modules by the addition of enhanced PWM capabilities. These allow for 2 or 4 output channels, user selectable polarity, dead band control, and automatic shutdown and restart, and are discussed in detail in Section 17.4. Except for the addition of the special event trigger, Capture and Compare functions of the ECCP module are the same as the standard CCP module.

The prototype control register for the enhanced CCP module is shown in Register 17-1. In addition to the expanded range of modes available through the CCPxCON register, the ECCP modules each have two additional registers associated with enhanced PWM operation and auto shutdown features. They are:

- ECCPxDEL (Dead band delay)
- ECCPxAS (Auto shutdown configuration)

REGISTER 17-1: CCPxCON REGISTER (ECCP1, ECCP2 AND ECCP3 MODULES)

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

bit 7-6 **PxM1:PxM0:** Enhanced PWM Output Configuration bits

If CCPxM3:CCPxM2 = 00, 01, 10:

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

If CCPxM3:CCPxM2 = 11:

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

bit 5-4 **DCxB1:DCxB0**: PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM3:CCPxM0**: Enhanced CCP Mode Select bits

0000 = Capture/Compare/PWM off (resets CCPx module)
0001 = Reserved
0010 = Compare mode, toggle output on match
0011 = Capture mode
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, initialize CCP pin low, set output on compare match (set CCPxIF)
1001 = Compare mode, initialize CCP pin high, clear output on compare match (set CCPxIF)
1010 = Compare mode, generate software interrupt only, CCP pin reverts to I/O state
1011 = Compare mode, trigger special event (ECCP resets TMR1 or TMR3, sets CCxIF bit, ECCP2 trigger starts A/D conversion if A/D module is enabled)**(1)**
1100 = PWM mode; PxA, PxC active high; PxB, PxD active high
1101 = PWM mode; PxA, PxC active high; PxB, PxD active low
1110 = PWM mode; PxA, PxC active low; PxB, PxD active high
1111 = PWM mode; PxA, PxC active low; PxB, PxD active low

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

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

17.1 ECCP Outputs and Configuration

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

- CCP2MX configuration bit (CONFIG3H<0>)
- ECCPMX configuration bit (CONFIG3H<1>)
- Program Memory mode (set by configuration bits CONFIG3L<1:0>)

The pin assignments for the enhanced CCP modules are summarized in Table 17-1, Table 17-2 and Table 17-3. To configure the I/O pins as PWM outputs, the proper PWM mode must be selected by setting the PxMx and CCPxMx bits (CCPxCON<7:6> and <3:0>, respectively). The appropriate TRIS direction bits for the corresponding port pins must also be set as outputs.

17.1.1 USE OF CCP4 AND CCP5 WITH ECCP1 AND ECCP3

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

ECCP1 and ECCP3, on the other hand, only have three dedicated output pins: CCPx/P3A, PxB, and PxC. Whenever these modules are configured for Quad PWM mode, the pin normally used for CCP4 or CCP5 becomes the D output pins for ECCP3 and ECCP1, respectively. The CCP4 and CCP5 modules remain functional but their outputs are overridden.

17.1.2 ECCP MODULE OUTPUTS AND PROGRAM MEMORY MODES

For PIC18F8X2X devices, the Program Memory mode of the device (Section 4.1.1) impacts both pin multiplexing and the operation of the module.

The ECCP2 input/output (CCP2/P2A) can be multiplexed to one of three pins. By default, this is RC1 for all devices; in this case, the default is when CCP2MX is set and the device is operating in Microcontroller mode. With PIC18F8X2X devices, three other options exist. When CCP2MX is not set (= 0) and the device is in Microcontroller mode, CCP2/P2A is multiplexed to RE7; in all other Program Memory modes, it is multiplexed to RB3.

The final option is for CCP2MX to be set while the device is operating in one of the three other Program Memory modes. In this case, ECCP1 and ECCP3 operate as compatible (i.e., single output) CCP modules. The pins used by their other outputs (PxB through PxD) are available for other multiplexed functions. ECCP2 continues to operate as an enhanced CCP module regardless of the Program Memory mode.

TABLE 17-1: PIN CONFIGURATIONS FOR ECCP1

ECCP Mode	CCP1CON Configuration	RC2	RE6	RE5	RG4	RH7	RH6
All PIC18F6X2X devices:							
Compatible CCP	00xx11xx	CCP1	RE6	RE5	RG4/CCP5	n/a	n/a
Dual PWM	10xx11xx	P1A	P1B	RE5	RG4/CCP5	n/a	n/a
Quad PWM	x1xx11xx	P1A	P1B	P1C	P1D	n/a	n/a
PIC18F8X2X devices, ECCPMX = 1, Microcontroller mode:							
Compatible CCP	00xx11xx	CCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
Dual PWM	10xx11xx	P1A	P1B	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
Quad PWM	x1xx11xx	P1A	P1B	P1C	P1D	RH7/AN15	RH6/AN14
PIC18F8X2X devices, ECCPMX = 0, Microcontroller mode:							
Compatible CCP	00xx11xx	CCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
Dual PWM	10xx11xx	P1A	RE6/AD14	RE5/AD13	RG4/CCP5	P1B	RH6/AN14
Quad PWM	x1xx11xx	P1A	RE6/AD14	RE5/AD13	P1D	P1B	P1C
PIC18F8X2X devices, ECCPMX = 1, all other Program Memory modes:							
Compatible CCP	00xx11xx	CCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14

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

Note 1: With ECCP1 in Quad PWM mode, CCP5's output is overridden by P1D; otherwise CCP5 is fully operational.

TABLE 17-2: PIN CONFIGURATIONS FOR ECCP2

ECCP Mode	CCP2CON Configuration	RB3	RC1	RE7	RE2	RE1	RE0
All devices, CCP2MX = 1, Microcontroller mode:							
Compatible CCP	00xx11xx	RB3/INT3	CCP2	RE7	RE2	RE1	RE0
Dual PWM	10xx11xx	RB3/INT3	P2A	RE7	P2B	RE1	RE0
Quad PWM	x1xx11xx	RB3/INT3	P2A	RE7	P2B	P2C	P2D
All devices, CCP2MX = 0, Microcontroller mode:							
Compatible CCP	00xx11xx	RB3/INT3	RC1/T1OS1	CCP2	RE2	RE1	RE0
Dual PWM	10xx11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	RE1	RE0
Quad PWM	x1xx11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	P2C	P2D
PIC18F8X2X devices, CCP2MX = 0, all other Program Memory modes:							
Compatible CCP	00xx11xx	CCP2	RC1/T1OS1	RE7/AD15	RE2/$\overline{CS}$	RE1/$\overline{WR}$	RE0/$\overline{RD}$
Dual PWM	10xx11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	RE1/$\overline{WR}$	RE0/$\overline{RD}$
Quad PWM	x1xx11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	P2C	P2D

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

TABLE 17-3: PIN CONFIGURATIONS FOR ECCP3

ECCP Mode	CCP3CON Configuration	RG0	RE4	RE3	RG3	RH5	RH4
All PIC18F6X2X devices:							
Compatible CCP	00xx11xx	CCP3	RE4	RE3	RG3/CCP4	n/a	n/a
Dual PWM	10xx11xx	P3A	P3B	RE3	RG3/CCP4	n/a	n/a
Quad PWM	x1xx11xx	P3A	P3B	P3C	P3D	n/a	n/a
PIC18F8X2X devices, ECCPMX = 1, Microcontroller mode:							
Compatible CCP	00xx11xx	CCP3	RE4/AD12	RE3/AD11	RG3/CCP4	RH5/AN13	RH4/AN12
Dual PWM	10xx11xx	P3A	P3B	RE3/AD11	RG3/CCP4	RH5/AN13	RH4/AN12
Quad PWM	x1xx11xx	P3A	P3B	P3C	P3D	RH5/AN13	RH4/AN12
PIC18F8X2X devices, ECCPMX = 0, Microcontroller mode:							
Compatible CCP	00xx11xx	CCP3	RE6/AD14	RE5/AD13	RG3/CCP4	RH7/AN15	RH6/AN14
Dual PWM	10xx11xx	P3A	RE6/AD14	RE5/AD13	RG3/CCP4	P3B	RH6/AN14
Quad PWM	x1xx11xx	P3A	RE6/AD14	RE5/AD13	P3D	P3B	P3C
PIC18F8X2X devices, ECCPMX = 1, all other Program Memory modes:							
Compatible CCP	00xx11xx	CCP3	RE6/AD14	RE5/AD13	RG3/CCP4	RH7/AN15	RH6/AN14

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

Note 1: With ECCP3 in Quad PWM mode, CCP4's output is overridden by P1D; otherwise CCP4 is fully operational.

17.1.3 ECCP MODULES AND TIMER RESOURCES

Like the standard CCP modules, the ECCP modules can utilize Timers 1, 2, 3 or 4, depending on the mode selected. Timer1 and Timer3 are available for modules in Capture or Compare modes, while Timer2 and Timer4 are available for modules in PWM mode. Additional details on timer resources are provided in Section 16.1.1.

17.2 Capture and Compare Modes

Except for the operation of the special event trigger discussed below, the Capture and Compare modes of the ECCP module are identical in operation to that of CCP4. These are discussed in detail in Section 16.2 and Section 16.3.

17.2.1 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated in Compare mode, on a match between the CCPR register pair and the selected timer. This can be used in turn to initiate an action.

The special event trigger output of either ECCP1 or ECCP2 resets the TMR1 or TMR3 register pair, depending on which timer resource is currently selected. This allows the CCPRx register to effectively be a 16-bit programmable period register for Timer1 or Timer3. In addition, the ECCP2 special event trigger will also start an A/D conversion if the A/D module is enabled.

The triggers are not implemented for ECCP3, CCP4 or CCP5. Selecting the Special Event mode (CCPxM3:CCPxM0 = `1011`) for these modules has the same effect as selecting the Compare with Software Interrupt mode (CCPxM3:CCPxM0 = `1010`).

Note:	The special event trigger from ECCP2 will not set the Timer1 or Timer3 interrupt flag bits.

17.3 Standard PWM Mode

When configured in Single Output mode, the ECCP module functions identically to the standard CCP module in PWM mode as described in Section 16.4. This is also sometimes referred to as “Compatible CCP” mode as in Tables 17-1 through 17-3.

Note:	When setting up single output PWM operations, users are free to use either of the processes described in Section 16.4.3 or Section 17.4.9. The latter is more generic but will work for either single or multi-output PWM.

17.4 Enhanced PWM Mode

The Enhanced PWM mode provides additional PWM output options for a broader range of control applications. The module is a backward compatible version of the standard CCP module and offers up to four outputs, designated PxA through PxD. Users are also able to select the polarity of the signal (either active high or active low). The module’s output mode and polarity are configured by setting the PxM1:PxM0 and CCPxM3CCPxM0 bits of the CCPxCON register (CCPxCON<7:6> and CCPxCON<3:0>, respectively).

For the sake of clarity, Enhanced PWM mode operation is described generically throughout this section with respect to ECCP1 and TMR2 modules. Control register names are presented in terms of ECCP1. All three enhanced modules, as well as the two timer resources, can be used interchangeably and function identically. TMR2 or TMR4 can be selected for PWM operation by selecting the proper bits in T3CON.

Figure 17-1 shows a simplified block diagram of PWM operation. All control registers are double-buffered and are loaded at the beginning of a new PWM cycle (the period boundary when Timer2 resets) in order to prevent glitches on any of the outputs. The exception is the PWM delay register, ECCP1DEL, which is loaded at either the duty cycle boundary or the boundary period (whichever comes first). Because of the buffering, the module waits until the assigned timer resets instead of starting immediately. This means that enhanced PWM waveforms do not exactly match the standard PWM waveforms but are instead offset by one full instruction cycle (4 TOSC).

As before, the user must manually configure the appropriate TRIS bits for output.

17.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the equation:

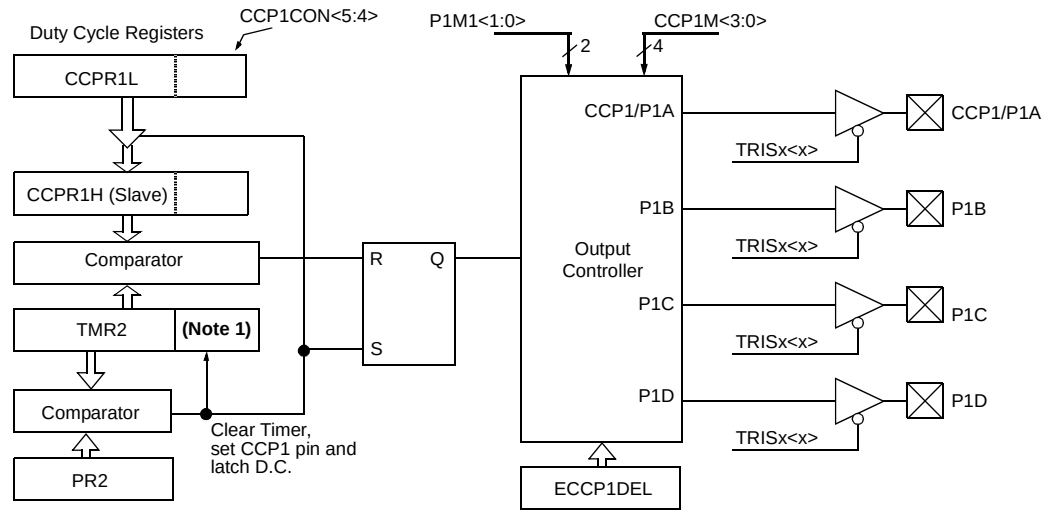
$$\text{PWM Period} = [(\text{PR2}) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1 / [PWM period]. When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is copied from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see Section 13.0) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

FIGURE 17-1: SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODULE



Note: The 8-bit timer TMR2 register is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

17.4.2 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 CCPRxL:CCPxCON<5:4>. The PWM duty cycle is calculated by the equation:

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \bullet T_{OSC} \bullet (\text{TMR2 Prescale Value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time but the duty cycle value is not copied into CCPR1H until a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read only register.

The CCPRxH register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation. When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or two bits of the TMR2 prescaler, the CCP1 pin is cleared. The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

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

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

TABLE 17-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

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

17.4.3 PWM OUTPUT CONFIGURATIONS

The P1M1:P1M0 bits in the CCP1CON register allow one of four configurations:

- Single Output
- Half-Bridge Output
- Full-Bridge Output, Forward mode
- Full-Bridge Output, Reverse mode

The Single Output mode is the standard PWM mode discussed in Section 17.4. The Half-Bridge and Full-Bridge Output modes are covered in detail in the sections that follow.

The general relationship of the outputs in all configurations is summarized in Figure 17-2.

FIGURE 17-2: PWM OUTPUT RELATIONSHIPS (ACTIVE HIGH STATE)

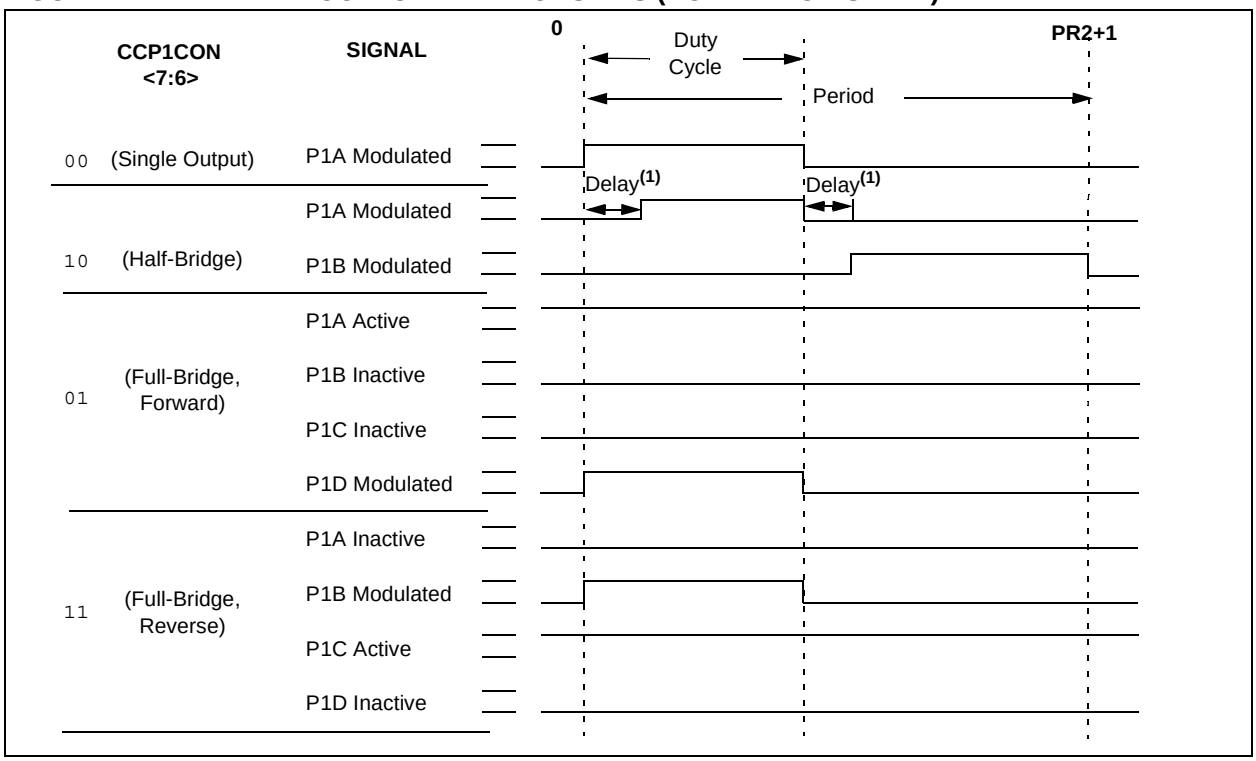
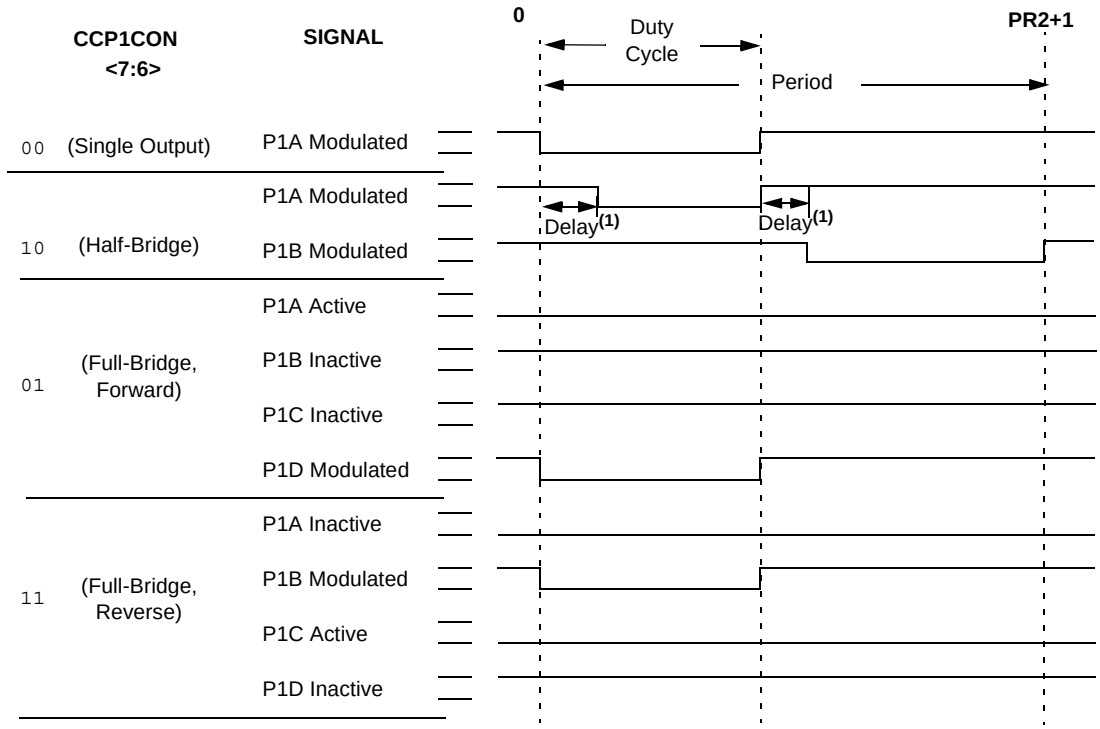


FIGURE 17-3: PWM OUTPUT RELATIONSHIPS (ACTIVE LOW STATE)



Relationships:

- Period = 4 * Tosc * (PR2 + 1) * (TMR2 prescale value)
- Duty Cycle = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 prescale value)
- Delay = 4 * Tosc * (ECCP1DEL<6:0>)

Note 1: Dead band delay is programmed using the ECCP1DEL register (Section 17.4.6).

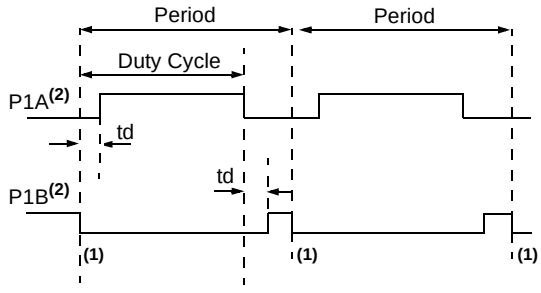
17.4.4 HALF-BRIDGE MODE

In the Half-Bridge Output mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the P1A pin, while the complementary PWM output signal is output on the P1B pin (Figure 17-4). This mode can be used for half-bridge applications, as shown in Figure 17-5, or for full-bridge applications, where four power switches are being modulated with two PWM signals.

In Half-Bridge Output mode, the programmable dead band delay can be used to prevent shoot-through current in half-bridge power devices. The value of bits PDC6:PDC0 sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. See Section 17.4.6 for more details of the dead band delay operations.

Since the P1A and P1B outputs are multiplexed with the PORTC<2> and PORTE<6> data latches, the TRISC<2> and TRISE<6> bits must be cleared to configure P1A and P1B as outputs.

FIGURE 17-4: HALF-BRIDGE PWM OUTPUT

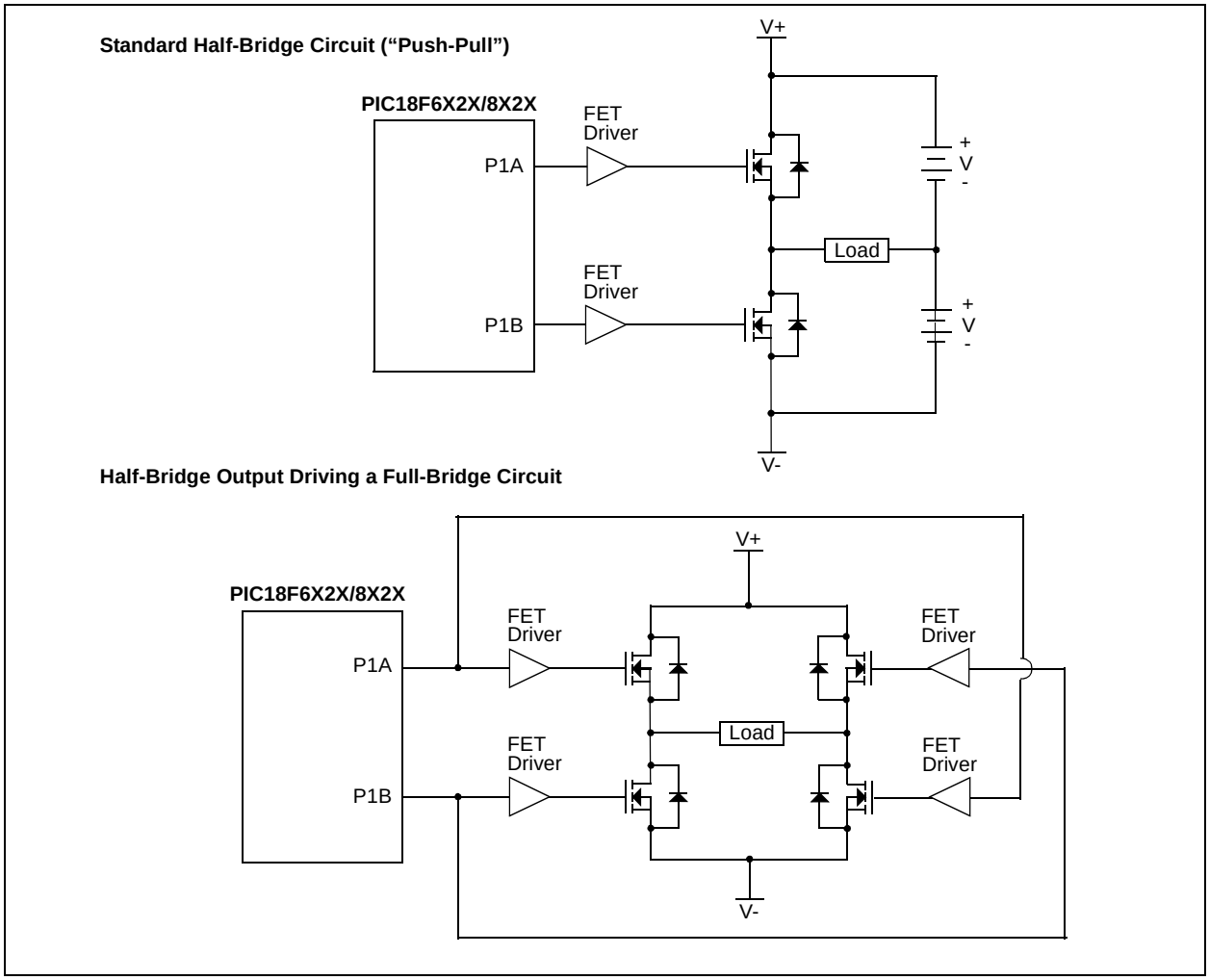


td = Dead Band Delay

Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: Output signals are shown as active high.

FIGURE 17-5: EXAMPLES OF HALF-BRIDGE OUTPUT MODE APPLICATIONS

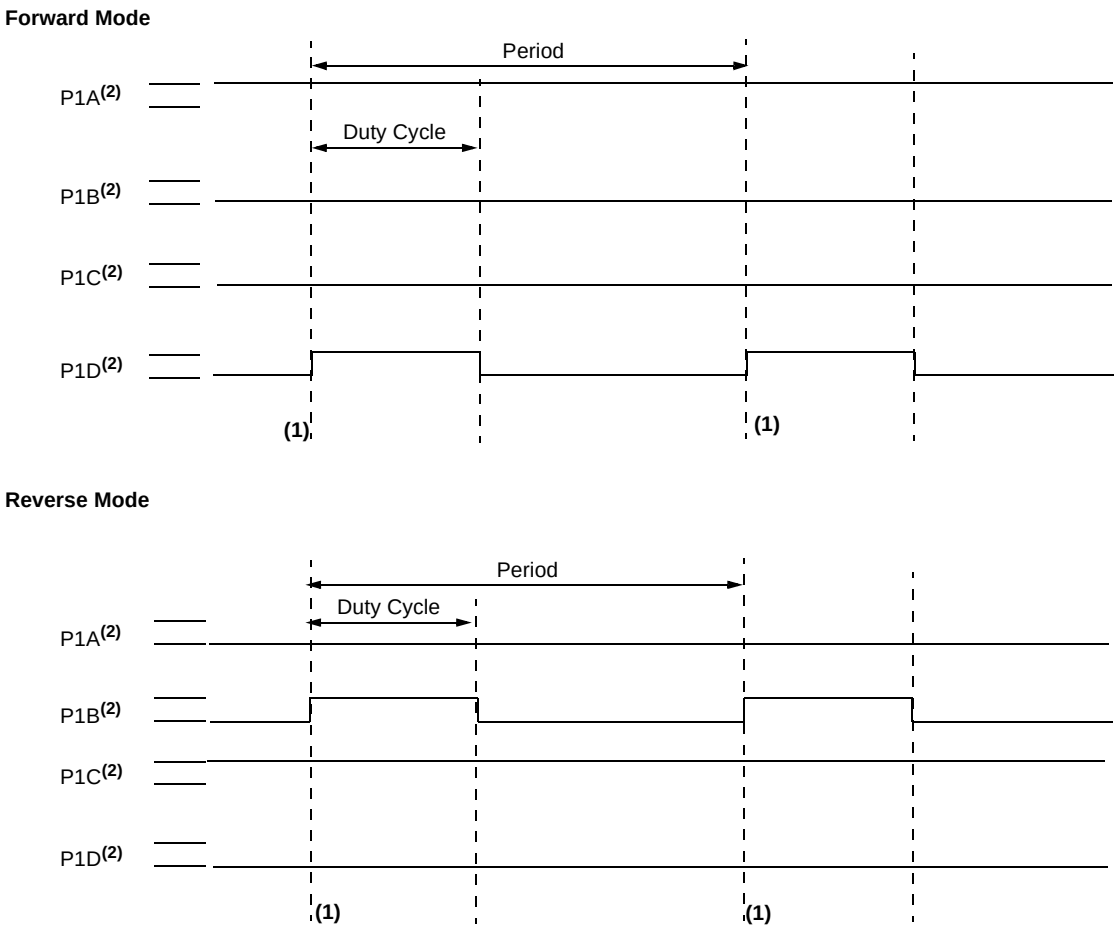


17.4.5 FULL-BRIDGE MODE

In Full-Bridge Output mode, four pins are used as outputs; however, only two outputs are active at a time. In the Forward mode, pin P1A is continuously active and pin P1D is modulated. In the Reverse mode, pin P1C is continuously active and pin P1B is modulated. These are illustrated in Figure 17-6.

P1A, P1B, P1C and P1D outputs are multiplexed with the PORTC<2>, PORTE<6:5>, and PORTG<4> data latches. The TRISC<2>, TRISC<6:5>, and TRISG<4> bits must be cleared to make the P1A, P1B, P1C, and P1D pins outputs.

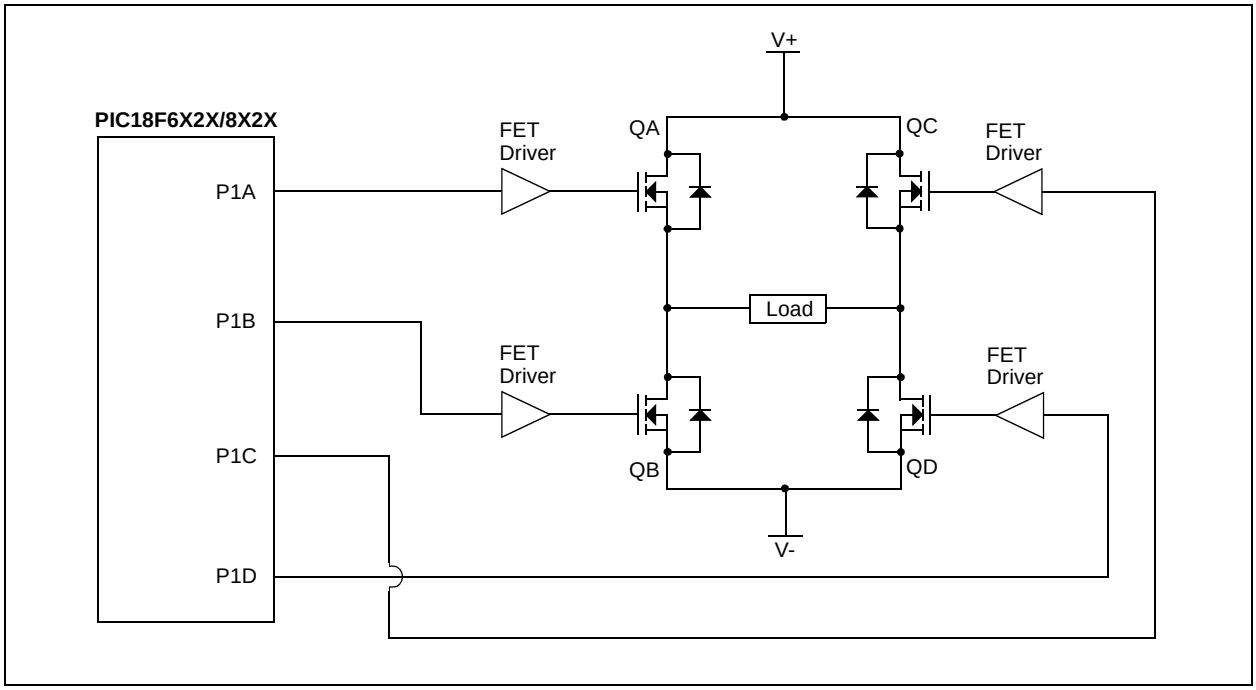
FIGURE 17-6: FULL-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: Output signal is shown as active high.

FIGURE 17-7: EXAMPLE OF FULL-BRIDGE APPLICATION



17.4.5.1 Direction Change in Full-Bridge Mode

In the Full-Bridge Output mode, the P1M1 bit in the CCP1CON register allows user to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will assume the new direction on the next PWM cycle.

Just before the end of the current PWM period, the modulated outputs (P1B and P1D) are placed in their inactive state, while the unmodulated outputs (P1A and P1C) are switched to drive in the opposite direction. This occurs in a time interval of (4 TOSC * (Timer2 Prescale value) before the next PWM period begins. The Timer2 prescaler will be either 1, 4 or 16, depending on the value of the T2CKPS bit (T2CON<1:0>). During the interval from the switch of the unmodulated outputs to the beginning of the next period, the modulated outputs (P1B and P1D) remain inactive. This relationship is shown in Figure 17-8.

Note that in the Full-Bridge Output mode, the CCP1 module does not provide any dead band delay. In general, since only one output is modulated at all times, dead band delay is not required. However, there is a situation where a dead band delay might be required. This situation occurs when both of the following conditions are true:

1. The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
2. The turn off time of the power switch, including the power device and driver circuit, is greater than the turn on time.

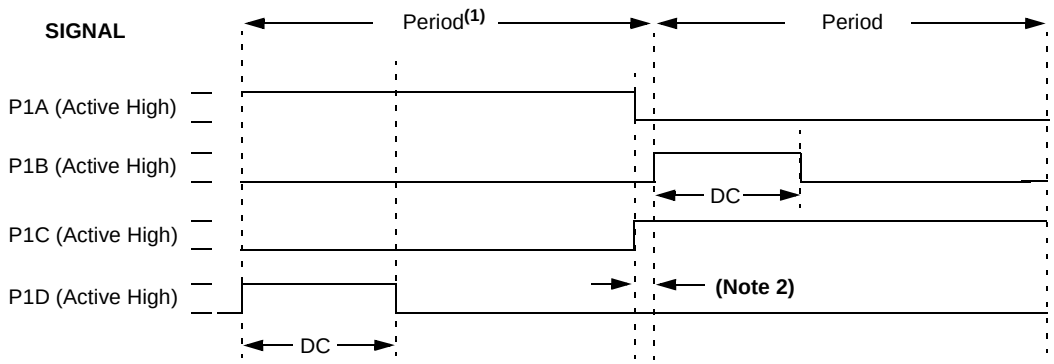
Figure 17-9 shows an example where the PWM direction changes from forward to reverse at a near 100% duty cycle. At time t1, the output P1A and P1D become inactive, while output P1C becomes active. In this example, since the turn off time of the power devices is longer than the turn on time, a shoot-through current may flow through power devices QC and QD (see Figure 17-7) for the duration of ‘t’. The same phenomenon will occur to power devices QA and QB for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, one of the following requirements must be met:

1. Reduce PWM for a PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

Other options to prevent shoot-through current may exist.

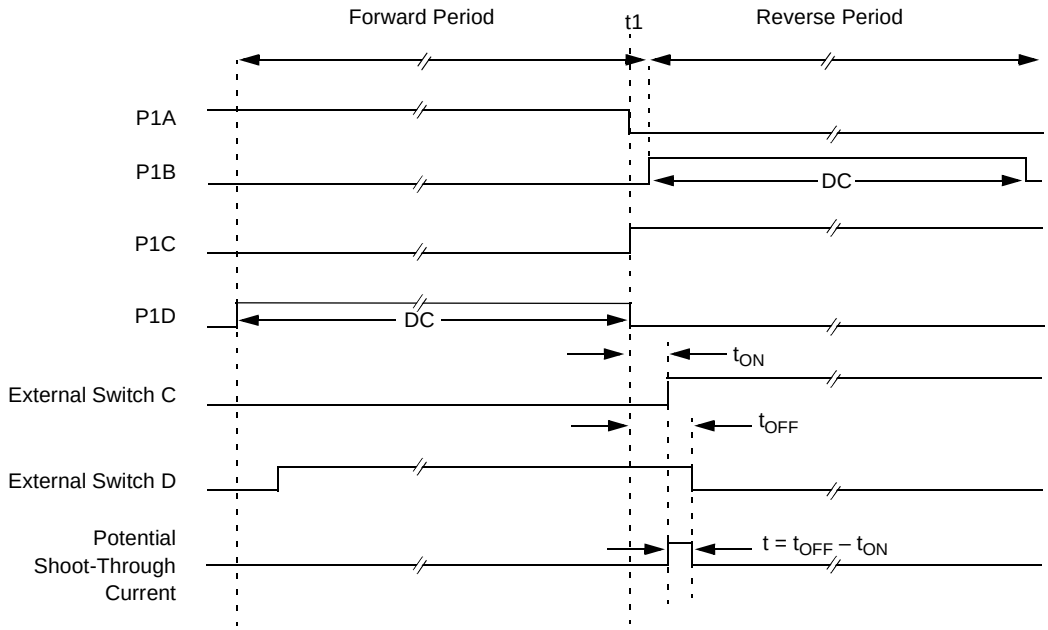
FIGURE 17-8: PWM DIRECTION CHANGE



Note 1: The direction bit in the CCP1 Control register (CCP1CON<7>) is written any time during the PWM cycle.

2: When changing directions, the P1A and P1C signals switch before the end of the current PWM cycle at intervals of 4 TOSC, 16 TOSC or 64 TOSC, depending on the Timer2 prescaler value. The modulated P1B and P1D signals are inactive at this time.

FIGURE 17-9: PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



Note 1: All signals are shown as active high.

2: t_{ON} is the turn on delay of power switch QC and its driver.

3: t_{OFF} is the turn off delay of power switch QD and its driver.

17.4.6 PROGRAMMABLE DEAD BAND DELAY

In half-bridge applications where all power switches are modulated at the PWM frequency at all times, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on, and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (*shoot-through current*) may flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In the Half-Bridge Output mode, a digitally programmable dead band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 17-4 for illustration. The lower seven bits of the ECCPxDEL register (Register 17-2) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

17.4.7 ENHANCED PWM AUTO SHUTDOWN

When the CCP1 is programmed for any of the enhanced PWM modes, the active output pins may be configured for auto shutdown. Auto shutdown immediately places the enhanced PWM output pins into a defined shutdown state when a shutdown event occurs.

A shutdown event can be caused by either of the two comparator modules or the INT0 pin (or any combination of these three sources). The comparators may be used to monitor a voltage input proportional to a current being monitored in the bridge circuit. If the voltage exceeds a threshold, the comparator switches state and triggers a shutdown. Alternatively, a digital signal on the INT0 pin can also trigger a shutdown. The auto shutdown feature can be disabled by not selecting any auto shutdown sources. The auto shutdown sources to be used are selected using the ECCP1AS2:ECCP1AS0 bits (bits<6:4> of the ECCP1AS register).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSS1AC1:PSS1AC0 and PSS1BD1:PSS1BD0 bits (ECCP1AS3:ECCP1AS0). Each pin pair (P1A/P1C and P1B/P1D) may be set to drive high, drive low, or be tri-stated (not driving). The ECCP1ASE bit (ECCP1AS<7>) is also set to hold the enhanced PWM outputs in their shutdown states.

The ECCP1ASE bit is set by hardware when a shutdown event occurs. If automatic restarts are not enabled, the ECCPASE bit is cleared by firmware when the cause of the shutdown clears. If automatic restarts are enabled, the ECCPASE bit is automatically cleared when the cause of the auto shutdown has cleared.

If the ECCPASE bit is set when a PWM period begins, the PWM outputs remain in their shutdown state for that entire PWM period. When the ECCPASE bit is cleared, the PWM outputs will return to normal operation at the beginning of the next PWM period.

Note:	Writing to the ECCPASE bit is disabled while a shutdown condition is active.

REGISTER 17-2: ECCPxDEL: PWM CONFIGURATION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
bit 7							bit 0

bit 7 **PxRSEN:** PWM Restart Enable bit

1 = Upon auto shutdown, the ECCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically

0 = Upon auto shutdown, ECCPxASE must be cleared in software to restart the PWM

bit 6-0 **PxDC6:PxDC0:** PWM Delay Count bits

Delay time, in number of FOSC/4 (4*TOSC) cycles, between the scheduled and actual time for a PWM signal to transition to active.

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 17-3: ECCPxAS: ENHANCED CAPTURE/COMPARE/PWM AUTO SHUTDOWN 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
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
bit 7							bit 0

bit 7 **ECCPxASE:** ECCP Auto Shutdown Event Status bit
0 = ECCP outputs are operating
1 = A shutdown event has occurred; ECCP outputs are in shutdown state

bit 6-4 **ECCPxAS2:ECCPxAS0:** ECCP Auto Shutdown Source Select bits
000 = Auto shutdown is disabled
001 = Comparator 1 output
010 = Comparator 2 output
011 = Either Comparator 1 or 2
100 = INT0
101 = INT0 or Comparator 1
110 = INT0 or Comparator 2
111 = INT0 or Comparator 1 or Comparator 2

bit 3-2 **PSSxAC1:PSSxAC0:** Pin A and C Shutdown State Control bits
00 = Drive Pins A and C to ‘0’
01 = Drive Pins A and C to ‘1’
1x = Pins A and C tri-state

bit 1-0 **PSSxBD1:PSSxBD0:** Pin B and D Shutdown State Control bits
00 = Drive Pins B and D to ‘0’
01 = Drive Pins B and D to ‘1’
1x = Pins B and D tri-state

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

17.4.7.1 Auto Shutdown and Automatic Restart

The auto shutdown feature can be configured to allow automatic restarts of the module following a shutdown event. This is enabled by setting the P1RSEN bit of the ECCP1DEL register (ECCP1DEL<7>).

In Shutdown mode with PRSEN = 1 (Figure 17-10), the ECCPASE bit will remain set for as long as the cause of the shutdown continues. When the shutdown condition clears, the ECCP1ASE bit is cleared. If PRSEN = 0 (Figure 17-11), once a shutdown condition occurs, the ECCP1ASE bit will remain set until it is cleared by firmware. Once ECCP1ASE is cleared, the enhanced PWM will resume at the beginning of the next PWM period.

Note:	Writing to the ECCPASE bit is disabled while a shutdown condition is active.

Independent of the P1RSEN bit setting, if the auto shutdown source is one of the comparators, the shutdown condition is a level. The ECCP1ASE bit cannot be cleared as long as the cause of the shutdown persists.

The Auto Shutdown mode can be forced by writing a '1' to the ECCPASE bit.

17.4.8 START-UP CONSIDERATIONS

When the ECCP module is used in the PWM mode, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins. When the microcontroller is released from RESET, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the off state until the microcontroller drives the I/O pins with the proper signal levels, or activates the PWM output(s).

The CCP1M1:CCP1M0 bits (CCP1CON<1:0>) allow the user to choose whether the PWM output signals are active high or active low for each pair of PWM output pins (P1A/P1C and P1B/P1D). The PWM output polarities must be selected before the PWM pins are configured as outputs. Changing the polarity configuration while the PWM pins are configured as outputs is not recommended since it may result in damage to the application circuits.

The P1A, P1B, P1C and P1D output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pins for output at the same time as the ECCP module may cause damage to the application circuit. The ECCP module must be enabled in the proper output mode and complete a full PWM cycle before configuring the PWM pins as outputs. The completion of a full PWM cycle is indicated by the TMR2IF bit being set as the second PWM period begins.

FIGURE 17-10: PWM AUTO SHUTDOWN (PRSEN = 1, AUTO RESTART ENABLED)

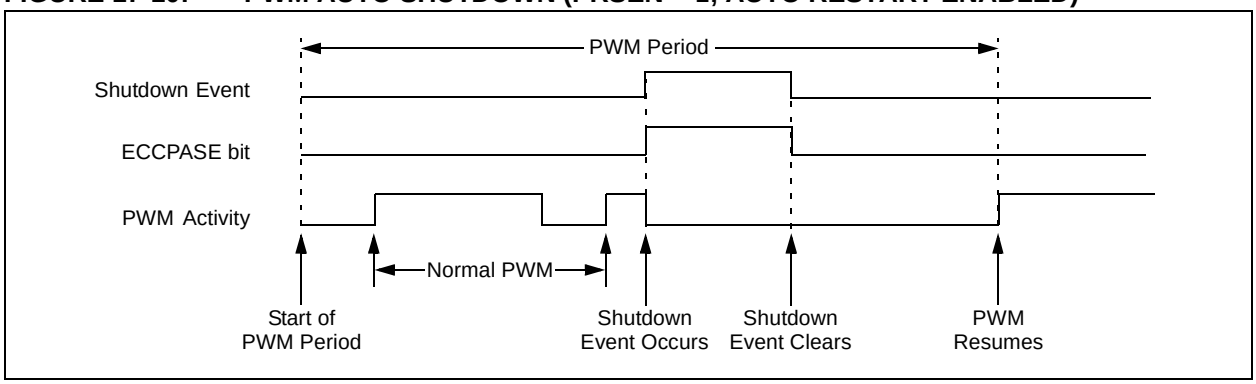
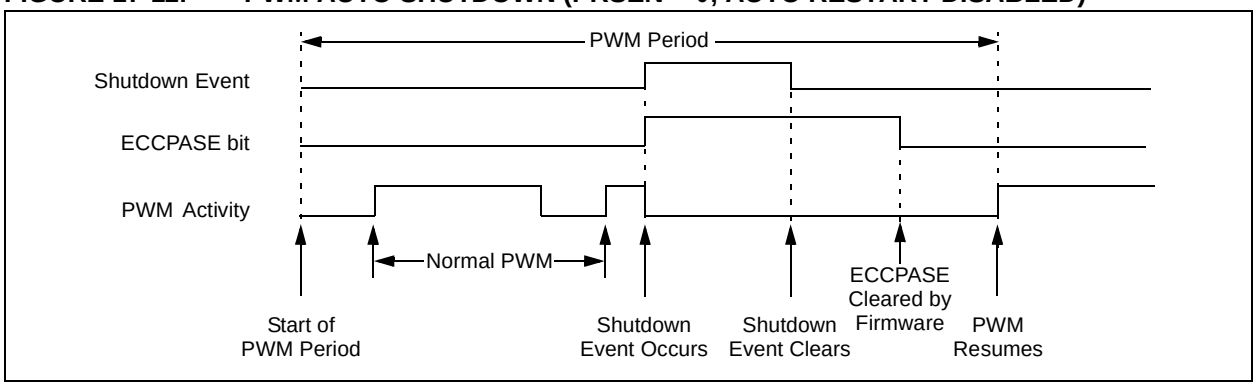


FIGURE 17-11: PWM AUTO SHUTDOWN (PRSEN = 0, AUTO RESTART DISABLED)



17.4.9 SETUP FOR PWM OPERATION

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

1. Configure the PWM pins PxA and PxB (and PxC and PxD, if used) as inputs by setting the corresponding TRIS bits.
2. Set the PWM period by loading the PR2 (PR4) register.
3. Configure the ECCPx module for the desired PWM mode and configuration by loading the CCPxCON register with the appropriate values:
   - Select one of the available output configurations and direction with the PxM1:PxM0 bits.
   - Select the polarities of the PWM output signals with the CCPxM3:CCPxM0 bits.
4. Set the PWM duty cycle by loading the CCPRxL register and CCPxCON<5:4> bits.
5. For Half-Bridge Output mode, set the dead band delay by loading ECCPxDEL<6:0> with the appropriate value.
6. If auto shutdown operation is required, load the ECCPxAS register:
   - Select the auto shutdown sources using the ECCPxAS2:ECCPxAS0 bits.
   - Select the shutdown states of the PWM output pins using PSSxAC1:PSSxAC0 and PSSxBD1:PSSxBD0 bits.
   - Set the ECCPxASE bit (ECCPxAS<7>).
   - Configure the comparators using the CMCON register.
   - Configure the comparator inputs as analog inputs.
7. If auto restart operation is required, set the PxRSEN bit (ECCPxDEL<7>).
8. Configure and start TMRn (TMR2 or TMR4):
   - Clear the TMRn interrupt flag bit by clearing the TMRnIF bit (PIR1<1> for Timer2 or PIR3<3> for Timer4).
   - Set the TMRn prescale value by loading the TnCKPS bits (TnCON<1:0>).
   - Enable Timer2 (or Timer4) by setting the TMRnON bit (TnCON<2>).
9. Enable PWM outputs after a new PWM cycle has started:
   - Wait until TMRn overflows (TMRnIF bit is set).
   - Enable the CCPx/PxA, PxB, PxC and/or PxD pin outputs by clearing the respective TRIS bits.
   - Clear the ECCPASE bit (ECCPxAS<7>).

17.4.10 EFFECTS OF A RESET

Both Power-on Reset and subsequent RESETS will force all ports to Input mode and the CCP registers to their RESET states.

This forces the enhanced CCP module to reset to a state compatible with the standard CCP module.

TABLE 17-5: REGISTERS ASSOCIATED WITH ECCP MODULES AND TIMER1 TO 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 000x	0000 000u
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	TMR1IF	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IF	1111 1111	1111 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
TRISB	PORTB Data Direction Register								1111 1111	1111 1111
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
TRISE	PORTE Data Direction Register								1111 1111	1111 1111
TRISG	—	—	—	PORTG Data Direction Register					---1 1111	---1 1111
TRISH	PORTH 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	T1SYNC	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu
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
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
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	uuuu uuuu
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
CCPRxL(1)	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
CCPRxH(1)	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
CCPxCON(1)	PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	0000 0000	0000 0000
ECCPxAS(1)	ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0	0000 0000	0000 0000
ECCPxDEL(1)	PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0	0000 0000	uuuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used during ECCP operation.

Note 1: Generic term for all of the identical registers of this name for all enhanced CCP modules, where 'x' identifies the individual module (ECCP1, ECCP2 or ECCP3). Bit assignments and RESET values for all registers of the same generic name are identical.

18.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

18.1 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

18.2 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.

18.3 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:

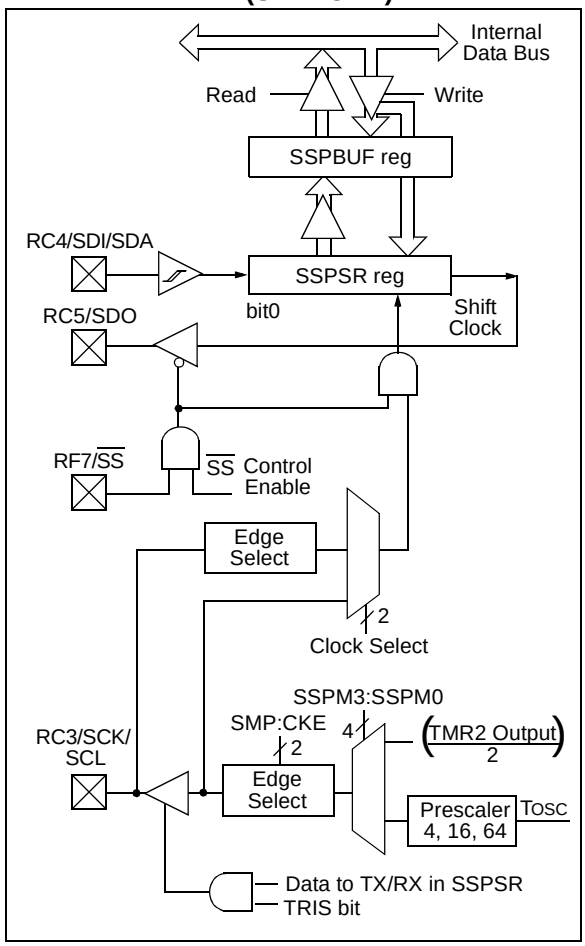
- Serial Data Out (SDO) - RC5/SDO
- Serial Data In (SDI) - RC4/SDI/SDA
- Serial Clock (SCK) - RC3/SCK/SCL

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) - RF7/$\overline{SS}$

Figure 18-1 shows the block diagram of the MSSP module when operating in SPI mode.

FIGURE 18-1: MSSP BLOCK DIAGRAM (SPI MODE)



18.3.1 REGISTERS

The MSSP module has four registers for SPI mode operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) - Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower 6 bits of the SSPSTAT are read only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPSR and SSPBUF together create a double-buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double-buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

REGISTER 18-1: 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 Edge Select bit
When CKP = 0:
1 = Data transmitted on rising edge of SCK
0 = Data transmitted on falling edge of SCK
When CKP = 1:
1 = Data transmitted on falling edge of SCK
0 = Data transmitted on rising edge of SCK

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ 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/$\overline{\text{Write}}$ 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

REGISTER 18-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	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

18.3.2 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 18-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable, and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP status register (SSPSTAT) indicates the various status conditions.

EXAMPLE 18-1: LOADING THE SSPBUF (SSPSR) REGISTER

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```


18.3.3 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, re-initialize 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

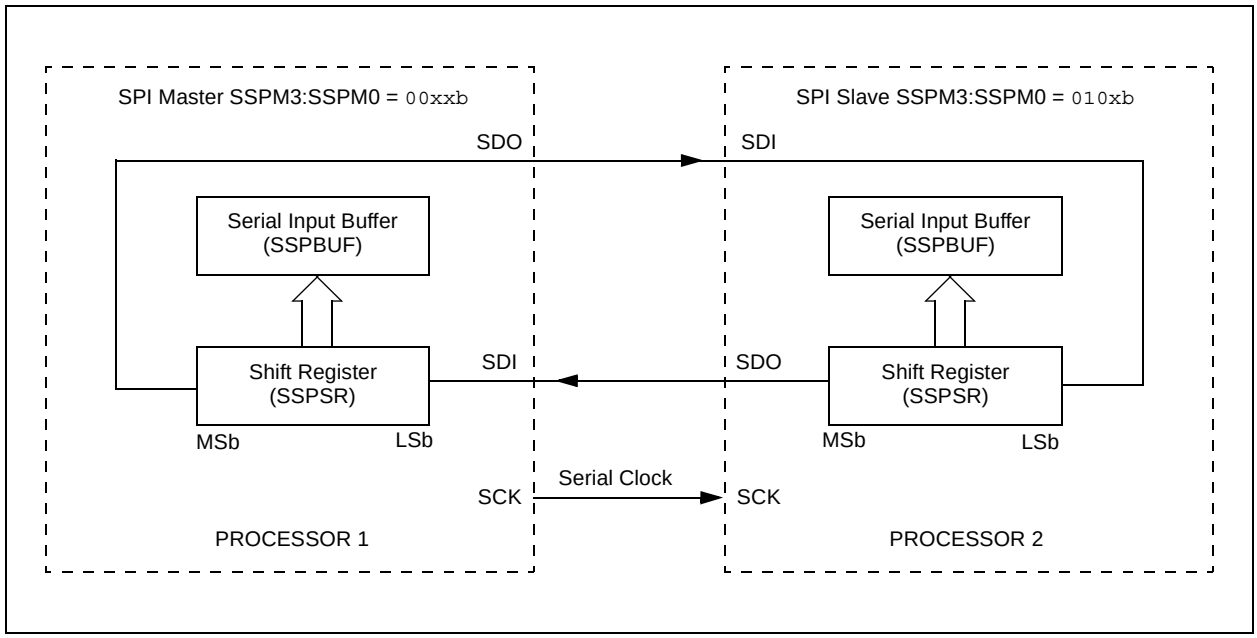
Any serial port function that is not desired may be overridden by programming the corresponding Data Direction (TRIS) register to the opposite value.

18.3.4 TYPICAL CONNECTION

Figure 18-2 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 18-2: SPI MASTER/SLAVE CONNECTION



18.3.5 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 18-2) 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.

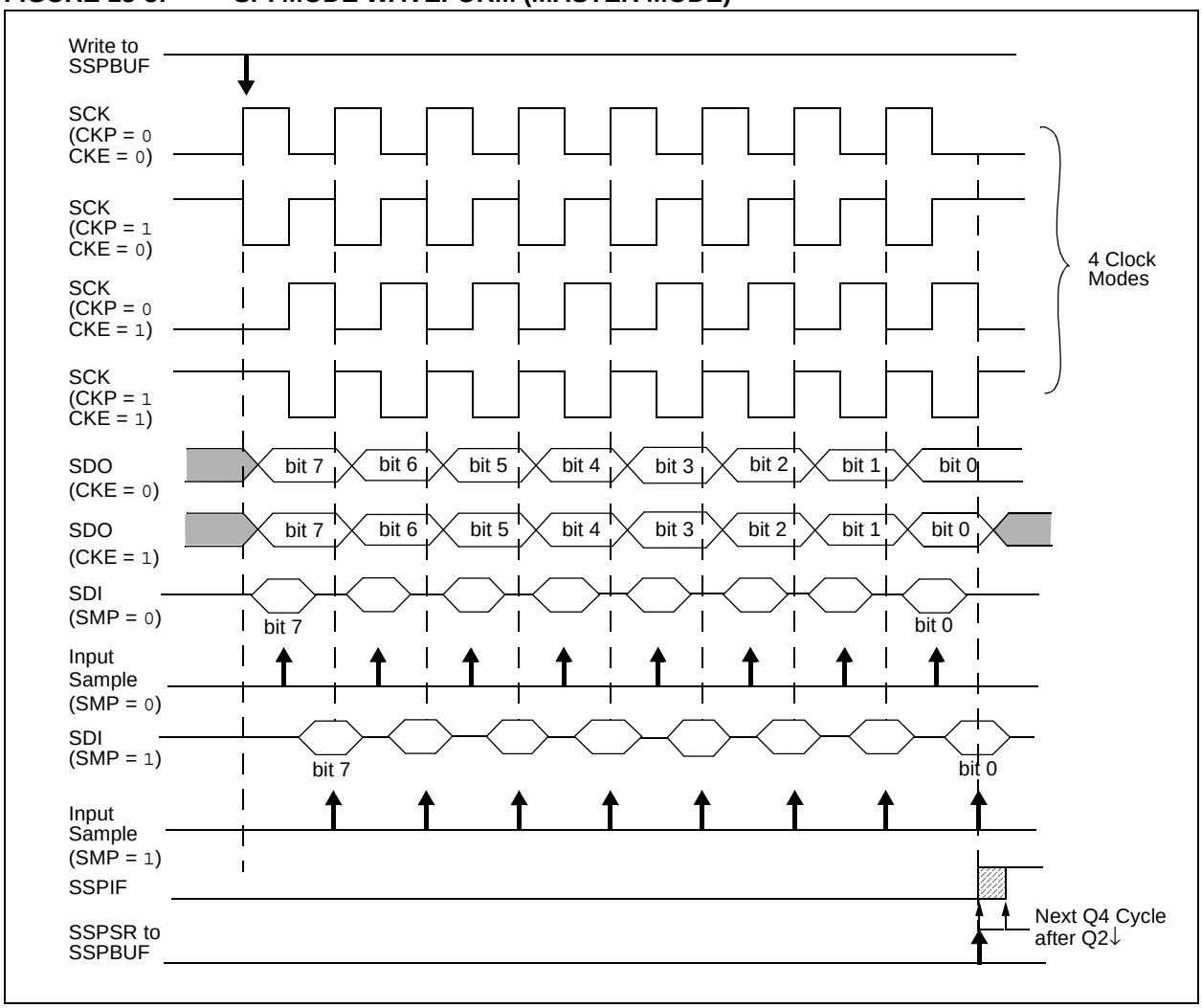
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 18-3, Figure 18-5, and Figure 18-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to 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 18-3 shows the waveforms for Master mode.

FIGURE 18-3: SPI MODE WAVEFORM (MASTER MODE)



18.3.6 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.

18.3.7 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 18-4: SLAVE SYNCHRONIZATION WAVEFORM

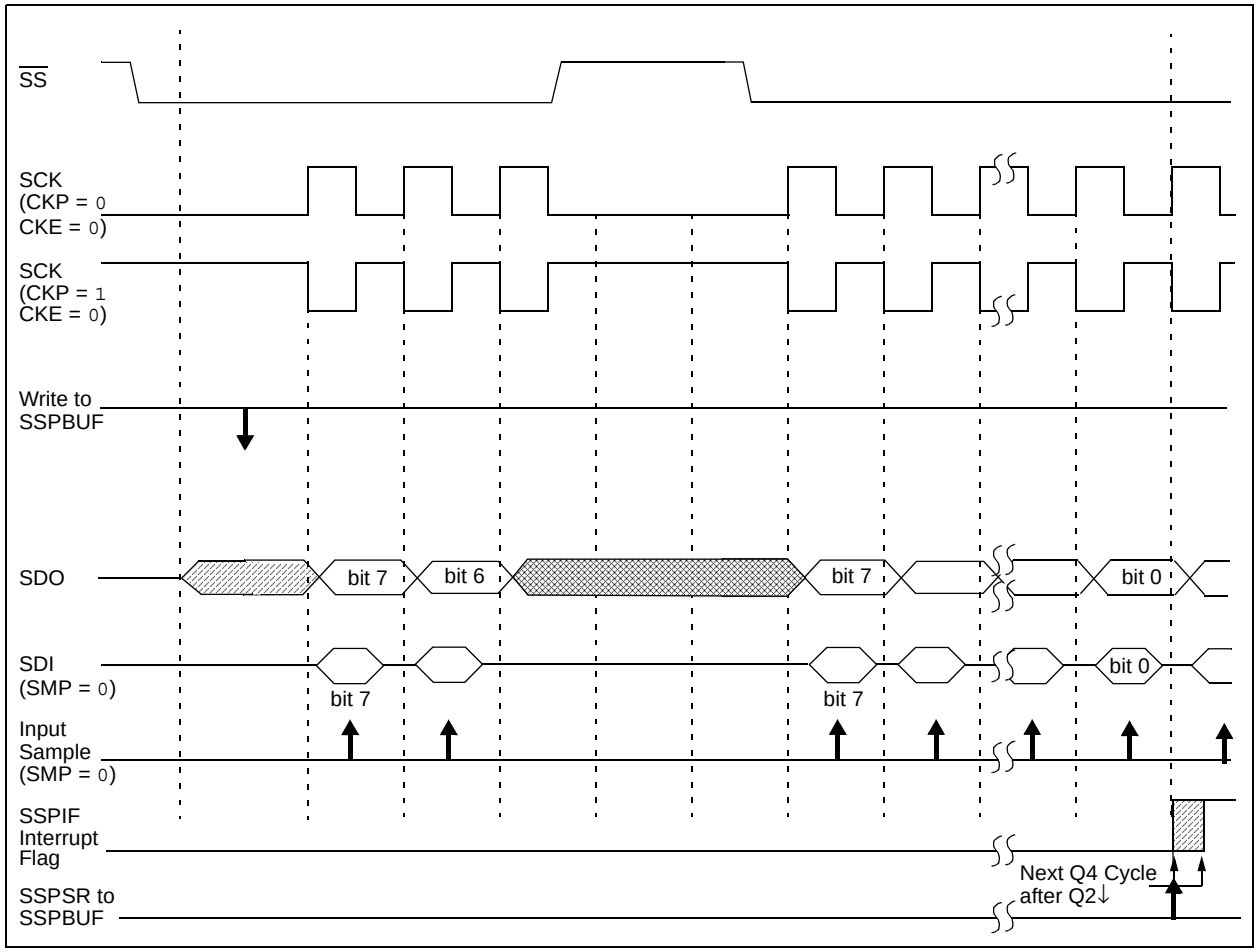


FIGURE 18-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

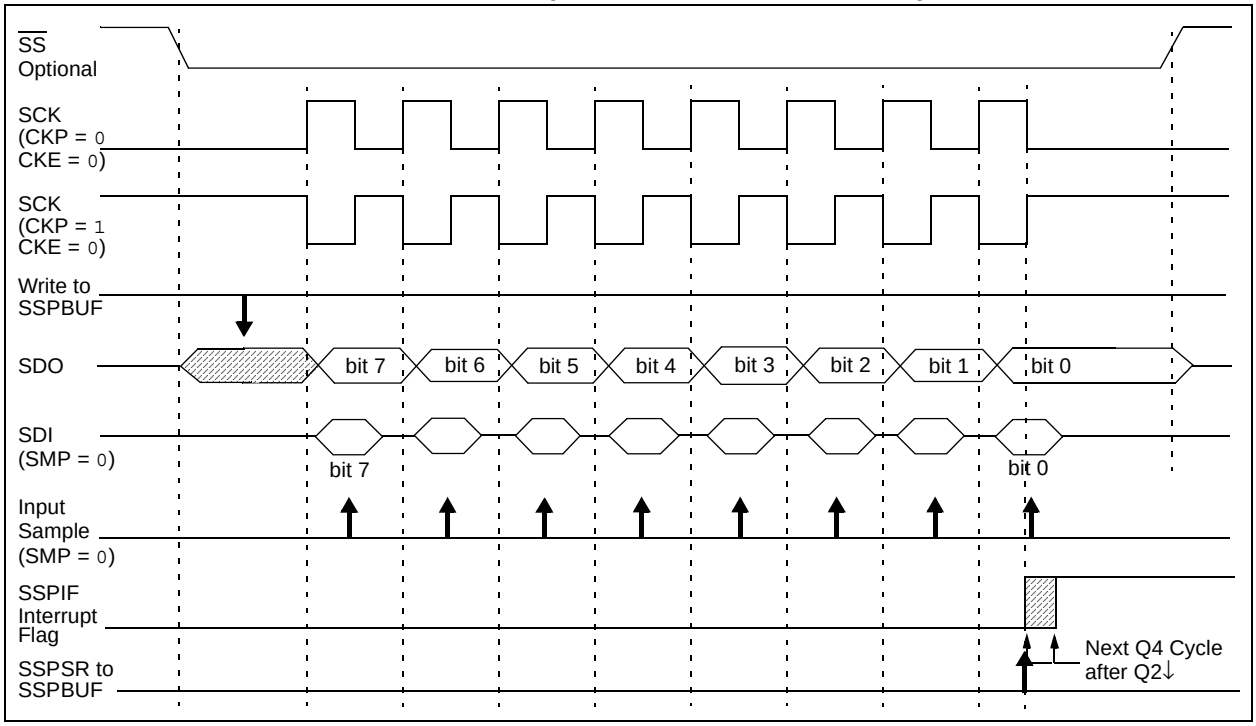
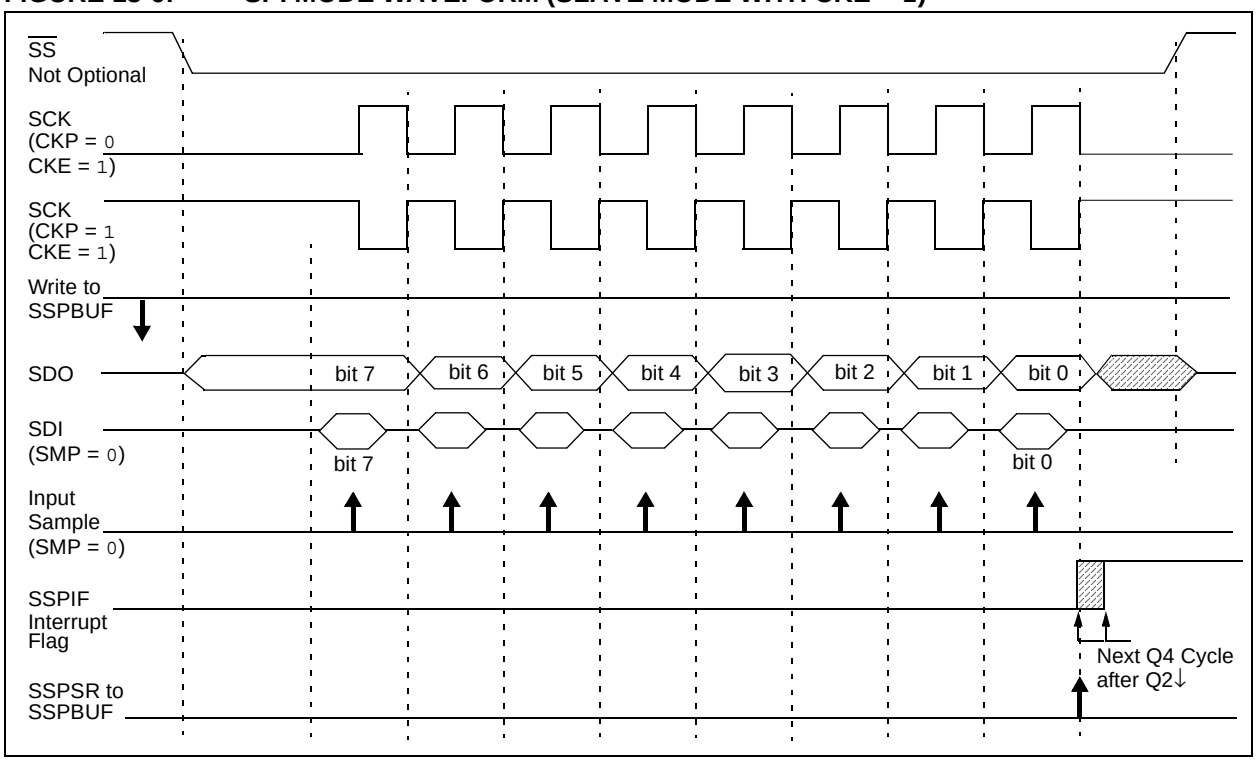


FIGURE 18-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



18.3.8 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.

18.3.9 EFFECTS OF A RESET

A RESET disables the MSSP module and terminates the current transfer.

18.3.10 BUS MODE COMPATIBILITY

Table 18-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 18-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also a SMP bit which controls when the data is sampled.

TABLE 18-2: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	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	1111 1111	1111 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
SSPCON1	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.

18.4 I²C Mode

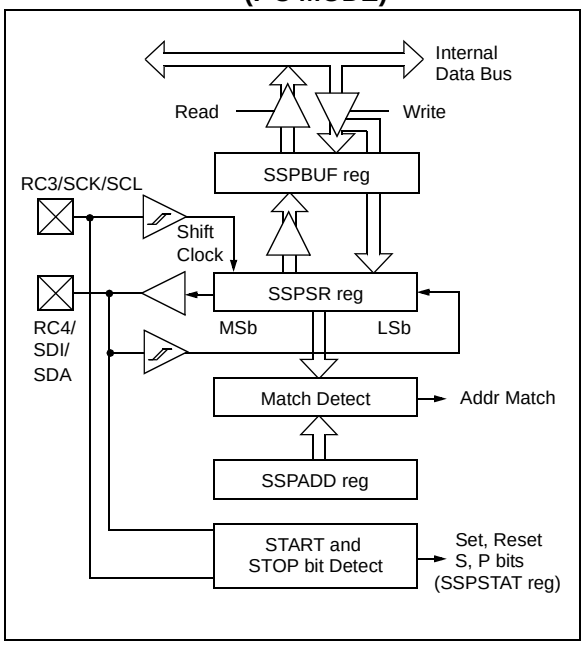
The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on START and STOP bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial clock (SCL) - RC3/SCK/SCL
- Serial data (SDA) - RC4/SDI/SDA

The user must configure these pins as inputs or outputs through the TRISC<4:3> bits.

FIGURE 18-7: MSSP BLOCK DIAGRAM (I²C MODE)



18.4.1 REGISTERS

The MSSP module has six registers for I²C operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) - Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON1, SSPCON2 and SSPSTAT are the control and status registers in I²C mode operation. The SSPCON1 and SSPCON2 registers are readable and writable. The lower 6 bits of the SSPSTAT are read only. The upper two bits of the SSPSTAT are read/write.

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 I²C 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.

REGISTER 18-3: SSPSTAT: MSSP STATUS REGISTER (I^2C 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/$\overline{\text{Address}}$ bit

In Master mode:

Reserved.

In Slave mode:

1 = Indicates that the last byte received or transmitted was data

0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** STOP bit

1 = 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/$\overline{\text{Write}}$ bit Information (I^2C 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 $\overline{\text{ACK}}$ 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 IDLE 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 = Receive complete, SSPBUF is full

0 = Receive not complete, SSPBUF is empty

In Receive mode:

1 = Data transmit in progress (does not include the $\overline{\text{ACK}}$ and STOP bits), SSPBUF is full

0 = Data transmit complete (does not include the $\overline{\text{ACK}}$ and STOP bits), 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

REGISTER 18-4: 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

REGISTER 18-5: 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

18.4.2 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 operation, 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.

18.4.3 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.

18.4.3.1 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.

18.4.3.2 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 18.4.4, “Clock Stretching” for more detail.

18.4.3.3 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 18.4.4, “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 18-9).

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 18-8: I^2C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)

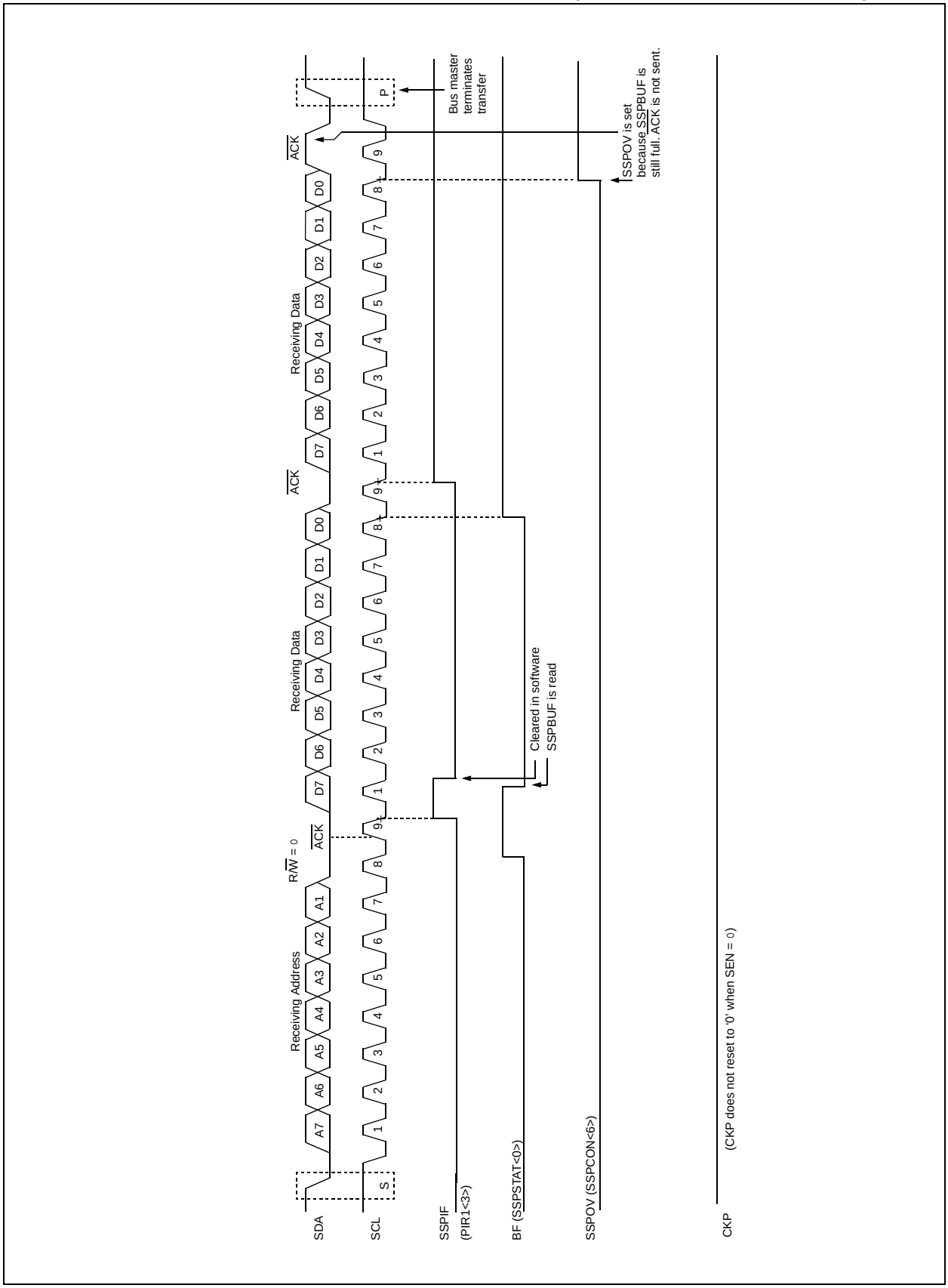


FIGURE 18-9: I²C SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

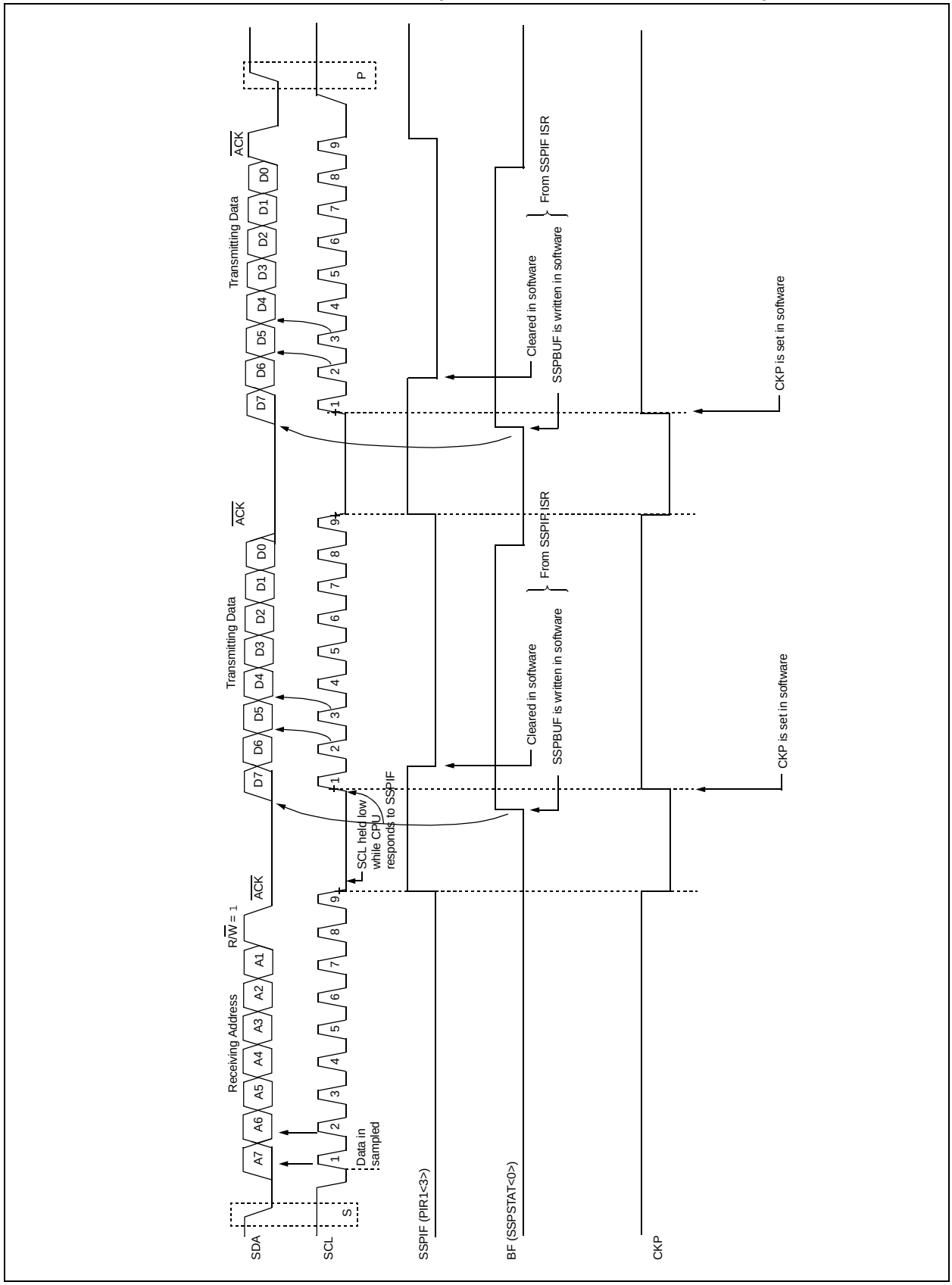


FIGURE 18-10: I²C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)

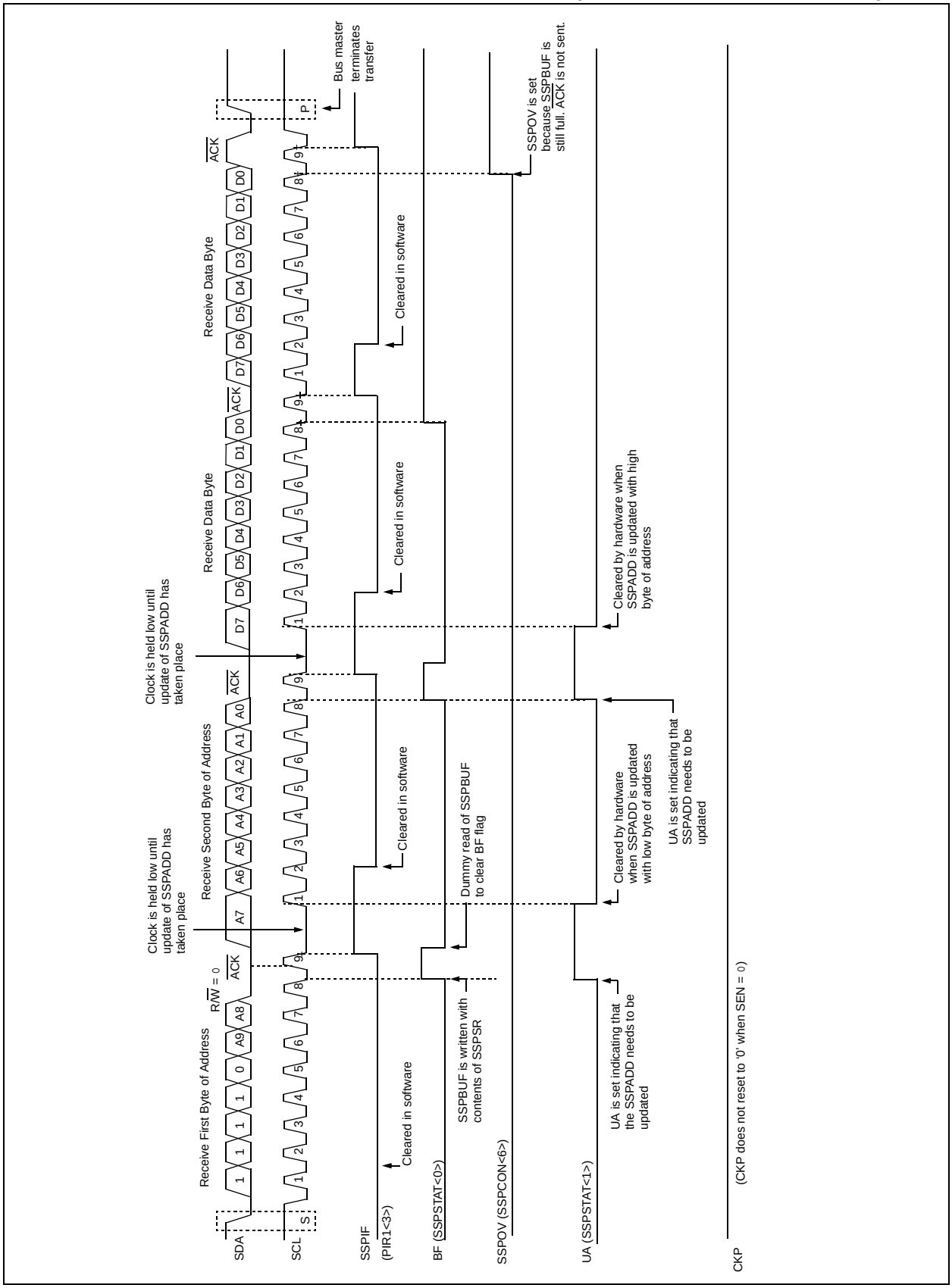
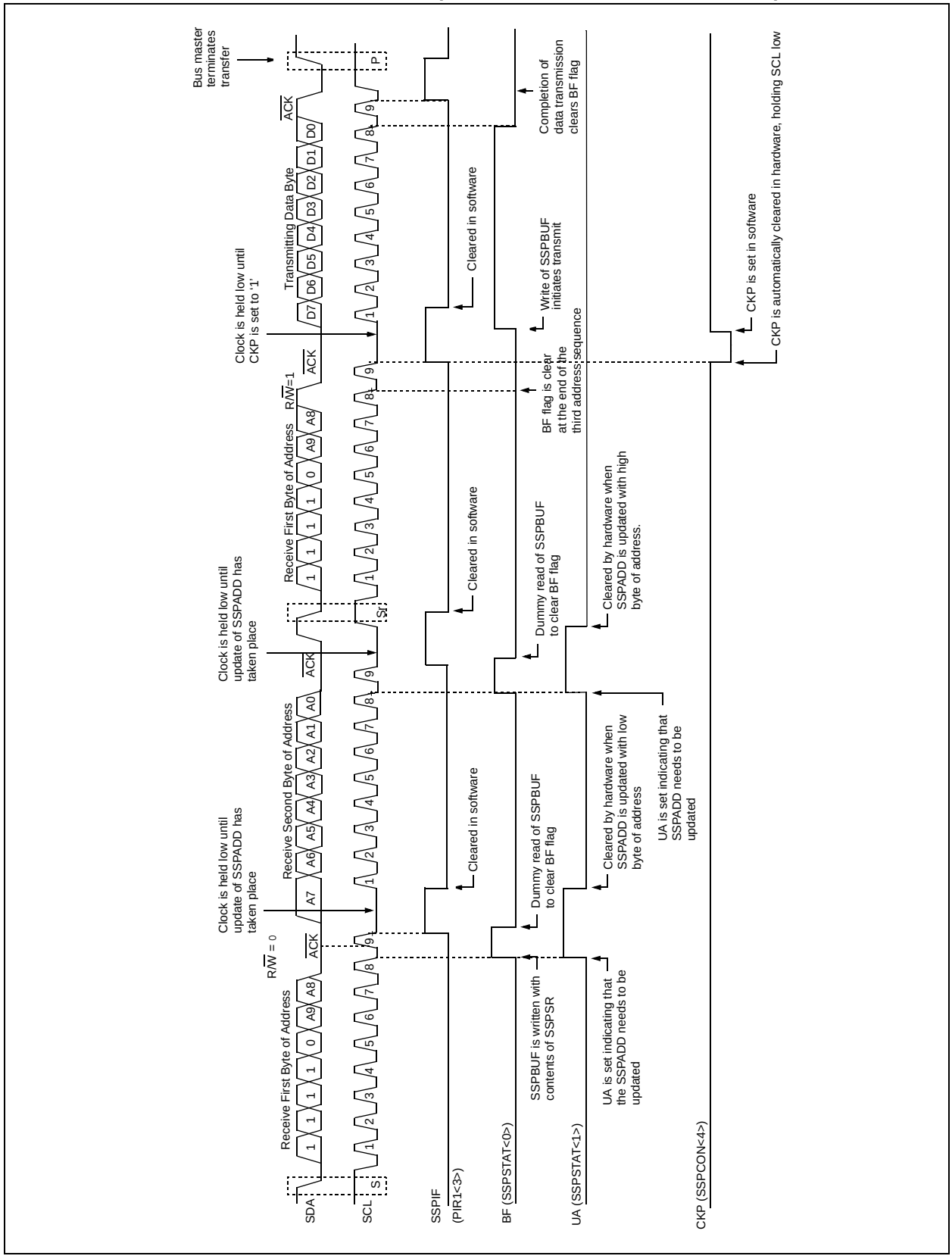


FIGURE 18-11: I²C SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



18.4.4 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.

18.4.4.1 Clock Stretching for 7-bit Slave Receive Mode (SEN = 1)

In 7-bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 18-13).

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.

18.4.4.2 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.

18.4.4.3 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 18-9).

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.

18.4.4.4 Clock Stretching for 10-bit Slave Transmit Mode

In 10-bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode, and clock stretching is controlled by the BF flag as in 7-bit Slave Transmit mode (see Figure 18-11).

18.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCL output is forced to '0'. However, setting the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external I^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 18-12).

FIGURE 18-12: CLOCK SYNCHRONIZATION TIMING

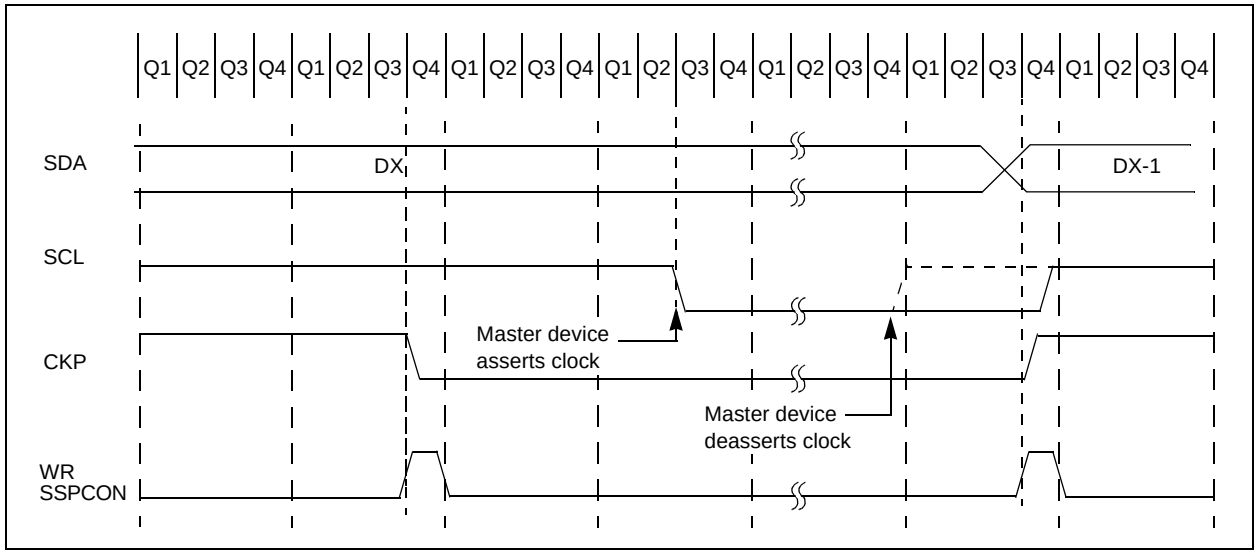


FIGURE 18-13: I^2C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)

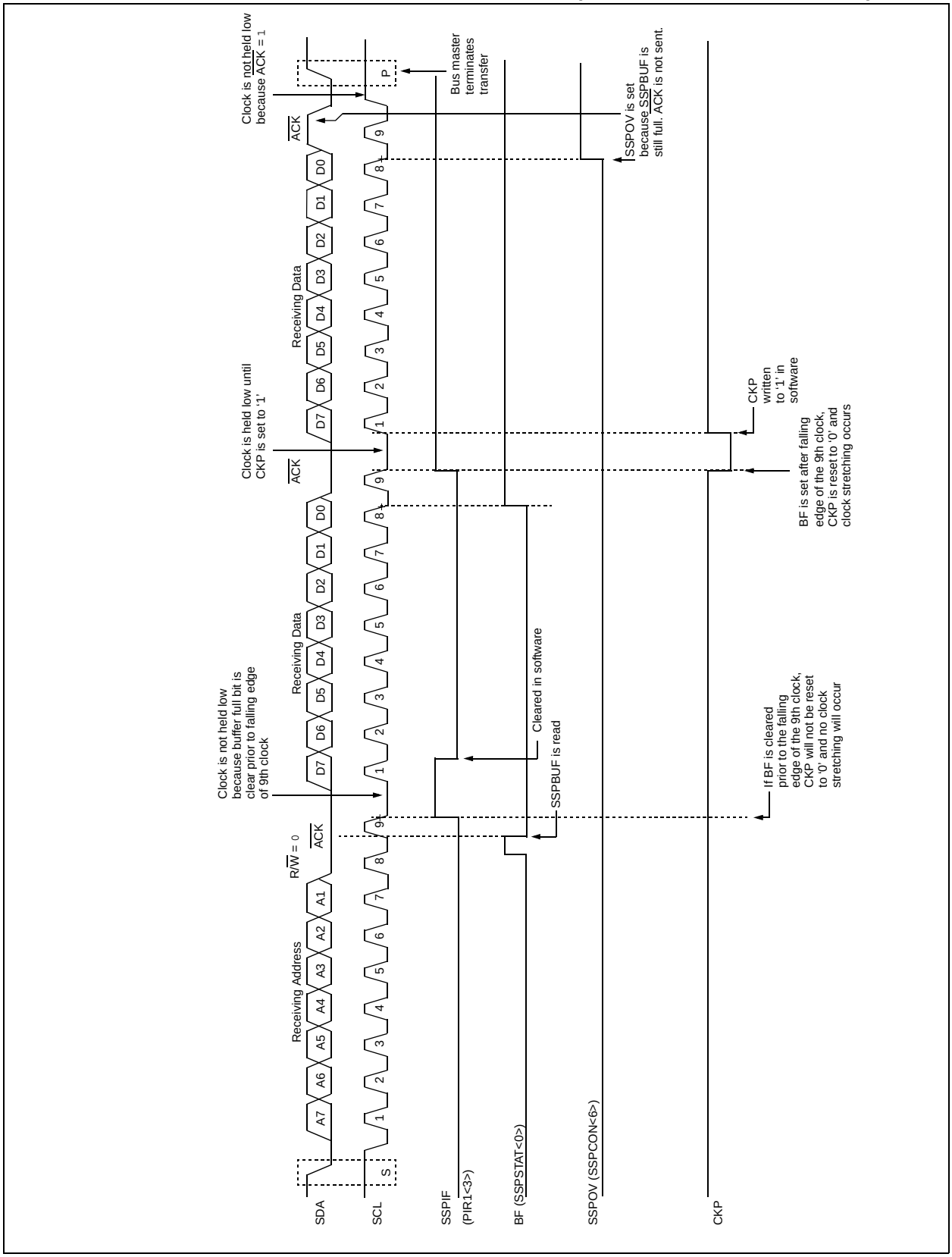
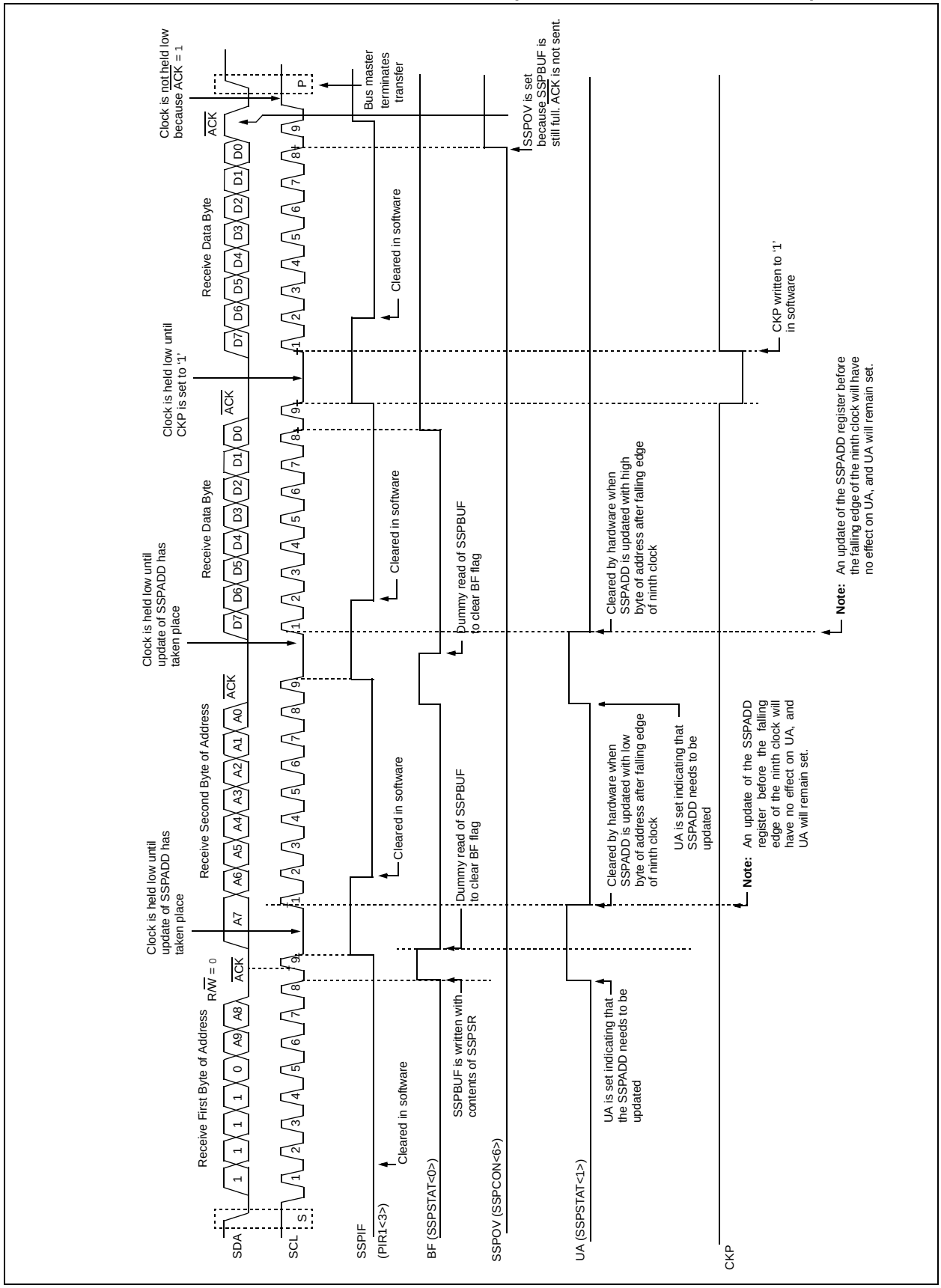


FIGURE 18-14: I²C SLAVE MODE TIMING SEN = 1 (RECEPTION, 10-BIT ADDRESS)



18.4.5 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.

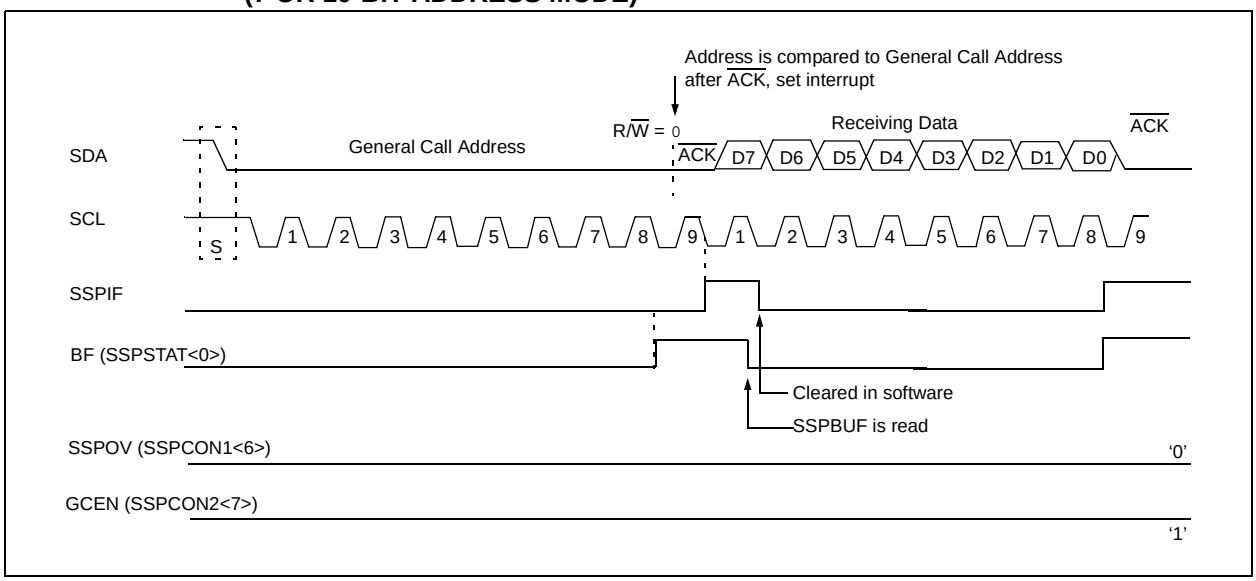
The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a START bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

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 18-15).

FIGURE 18-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESS MODE)



18.4.6 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.

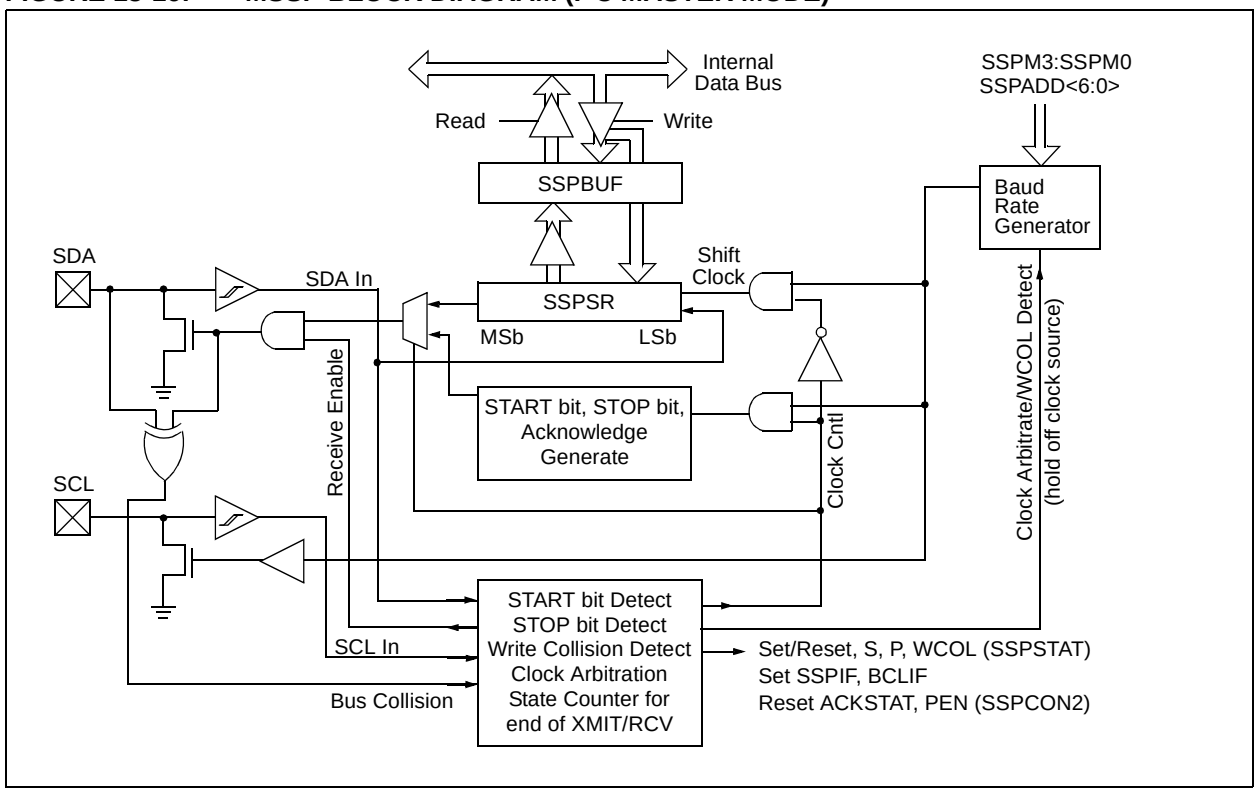
1. Assert a START condition on SDA and SCL.
2. Assert a Repeated START condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate 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 18-16: MSSP BLOCK DIAGRAM (I^2C MASTER MODE)



18.4.6.1 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 18.4.7, "Baud Rate Generator" for more detail.

A typical transmit sequence would go as follows:

1. The user generates a START condition by setting the START enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. Address is shifted out the SDA pin until all 8 bits are transmitted.
5. The MSSP Module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with 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.

18.4.7 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 18-17). 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 18-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPADD.

FIGURE 18-17: BAUD RATE GENERATOR BLOCK DIAGRAM

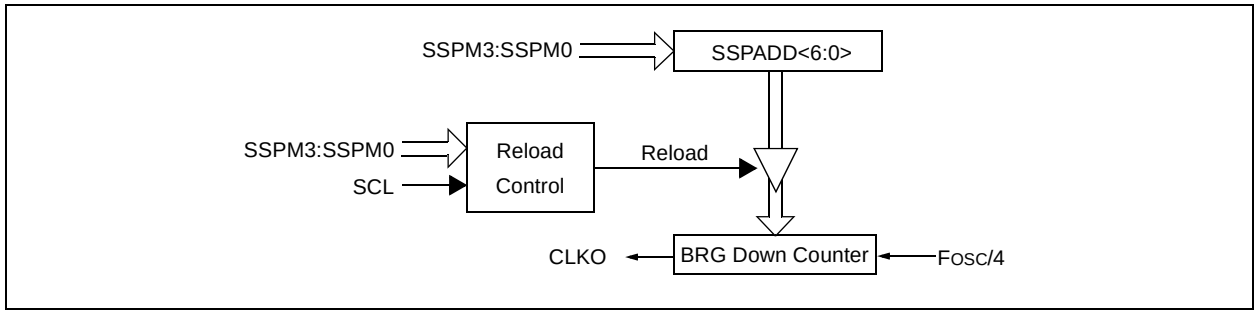


TABLE 18-3: I^2C CLOCK RATE W/BRG

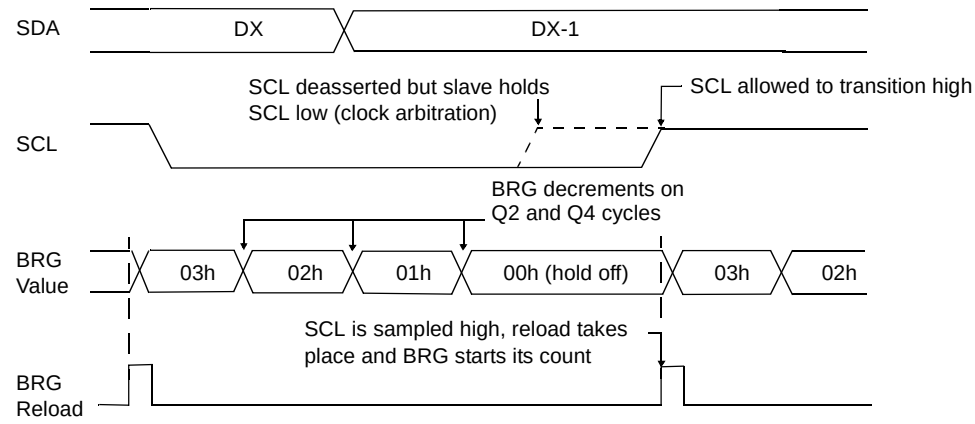
FCY	FCY*2	BRG Value	FSCL (2 Rollovers of BRG)
10 MHz	20 MHz	19h	400 kHz(1)
10 MHz	20 MHz	20h	312.5 kHz
10 MHz	20 MHz	3Fh	100 kHz
4 MHz	8 MHz	0Ah	400 kHz(1)
4 MHz	8 MHz	0Dh	308 kHz
4 MHz	8 MHz	28h	100 kHz
1 MHz	2 MHz	03h	333 kHz(1)
1 MHz	2 MHz	0Ah	100 kHz
1 MHz	2 MHz	00h	1 MHz(1)

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.

18.4.7.1 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 18-18).

FIGURE 18-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



18.4.8 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.

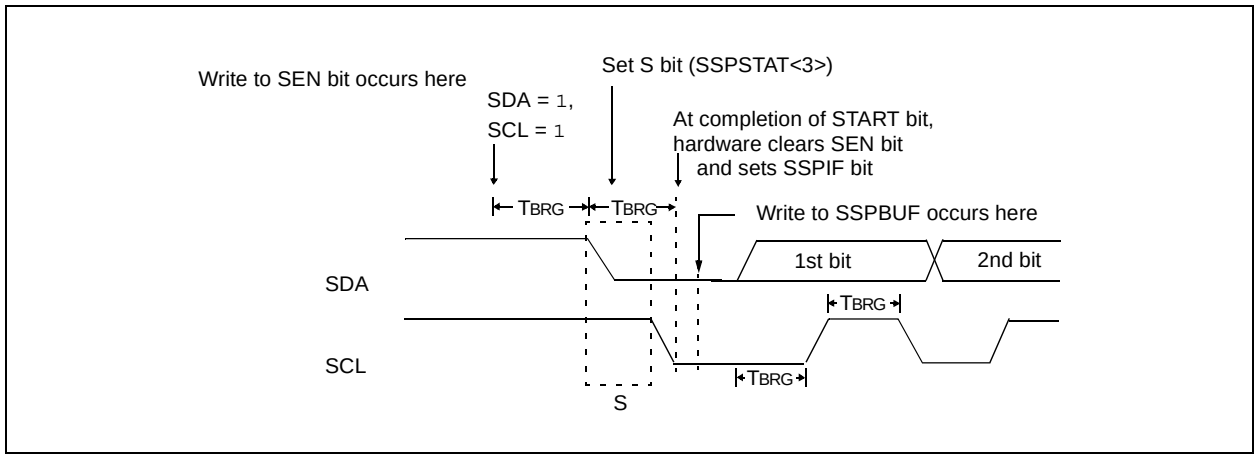
Note:	If at the beginning of the START condition, the SDA and SCL pins are already sampled low, or if during the START condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the START condition is aborted, and the I²C module is reset into its IDLE state.

18.4.8.1 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 18-19: FIRST START BIT TIMING



18.4.9 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'.

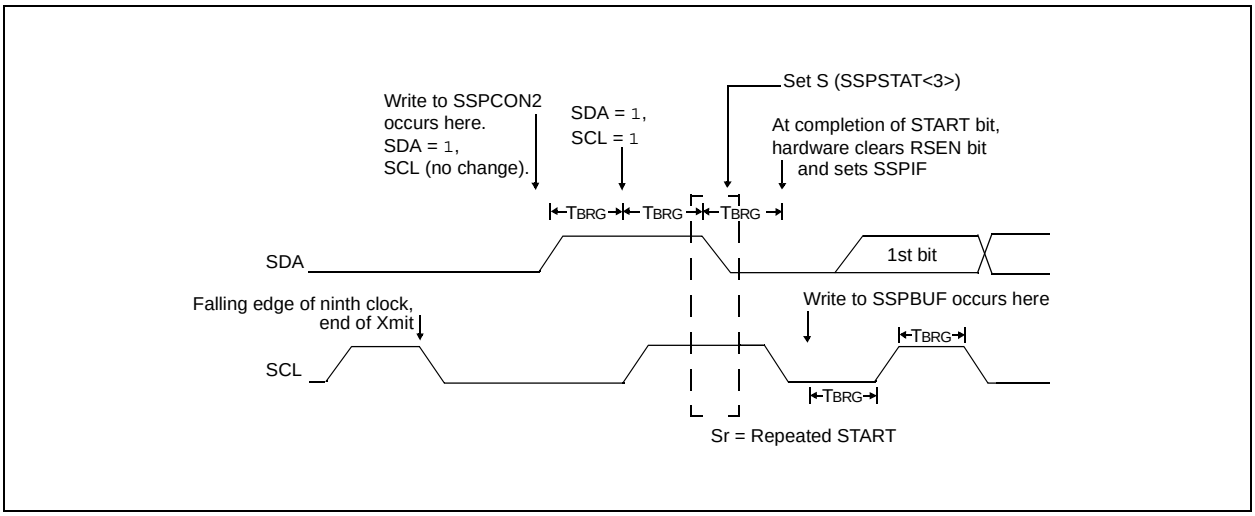
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

18.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated START sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated START condition is complete.

FIGURE 18-20: REPEAT START CONDITION WAVEFORM



18.4.10 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 18-21).

After the write to the SSPBUF, each bit of address will be shifted out on the falling edge of SCL until all seven address bits and the $\overline{R/W}$ bit are completed. On the falling edge of the eighth clock, the master will deassert the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the $\overline{ACK}$ 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.

18.4.10.1 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.

18.4.10.2 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.

18.4.10.3 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.

18.4.11 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>).

18.4.11.1 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.

18.4.11.2 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.

18.4.11.3 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 18-21: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)

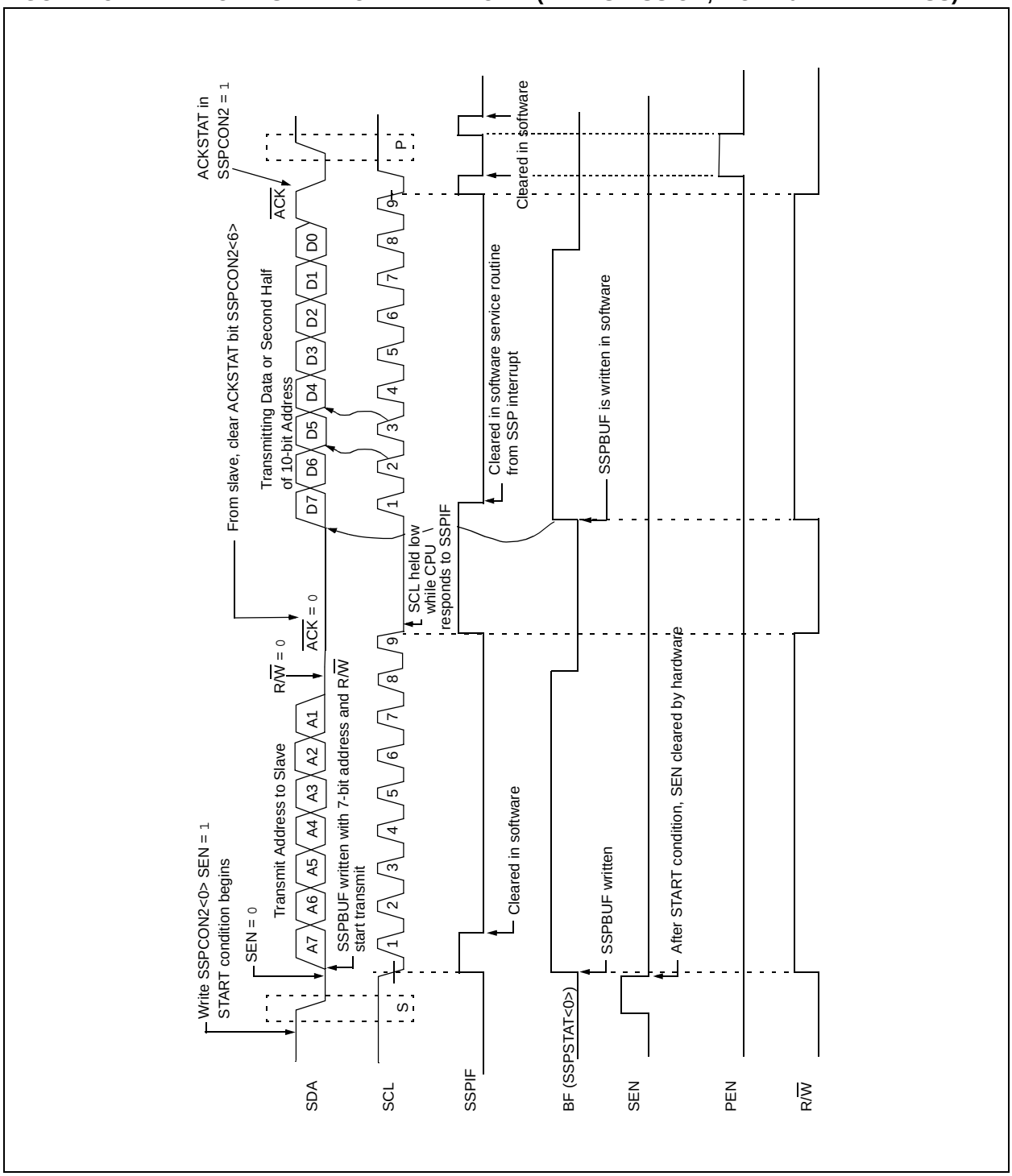
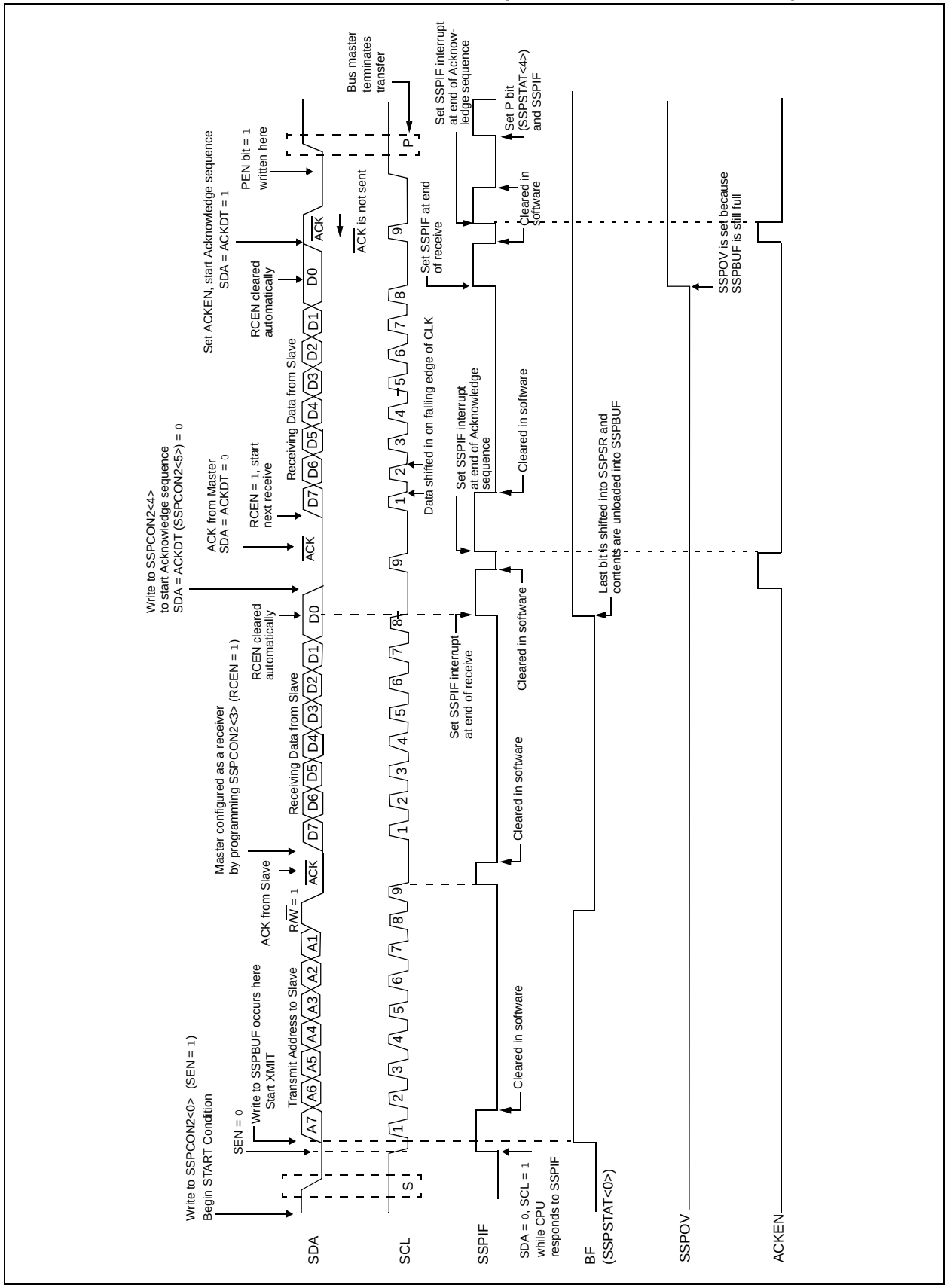


FIGURE 18-22: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



18.4.12 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 18-23).

18.4.12.1 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).

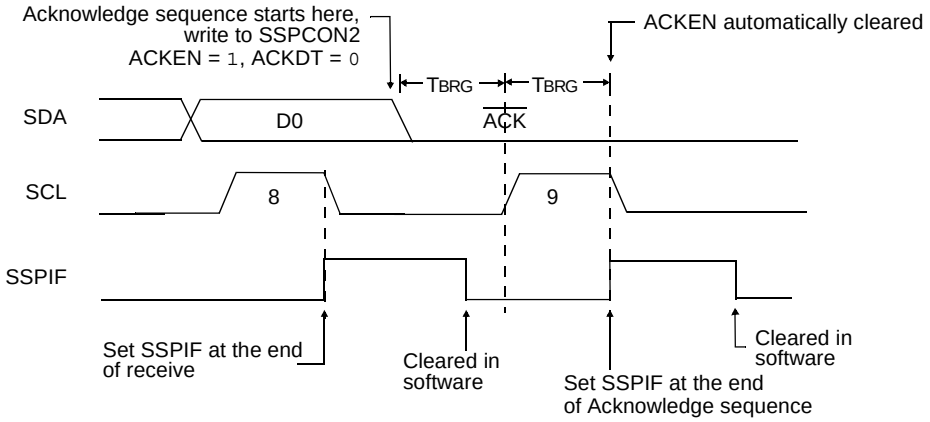
18.4.13 STOP CONDITION TIMING

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 18-24).

18.4.13.1 WCOL Status Flag

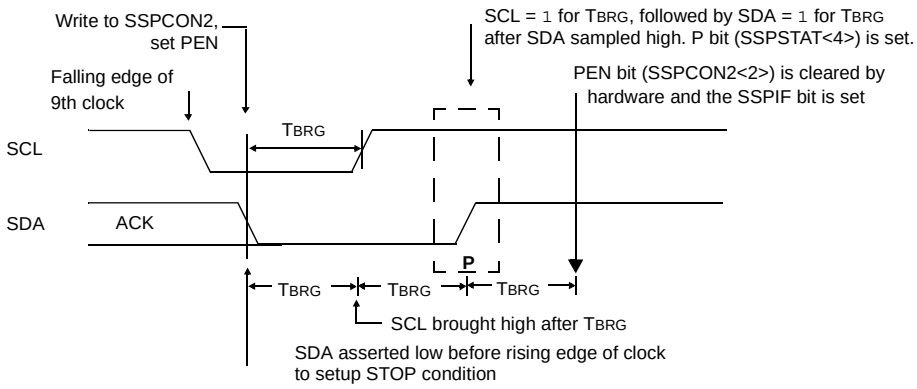
If the user writes the SSPBUF when a STOP sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 18-23: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one baud rate generator period.

FIGURE 18-24: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one baud rate generator period.

18.4.14 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).

18.4.15 EFFECT OF A RESET

A RESET disables the MSSP module and terminates the current transfer.

18.4.16 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

18.4.17 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 18-25).

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.

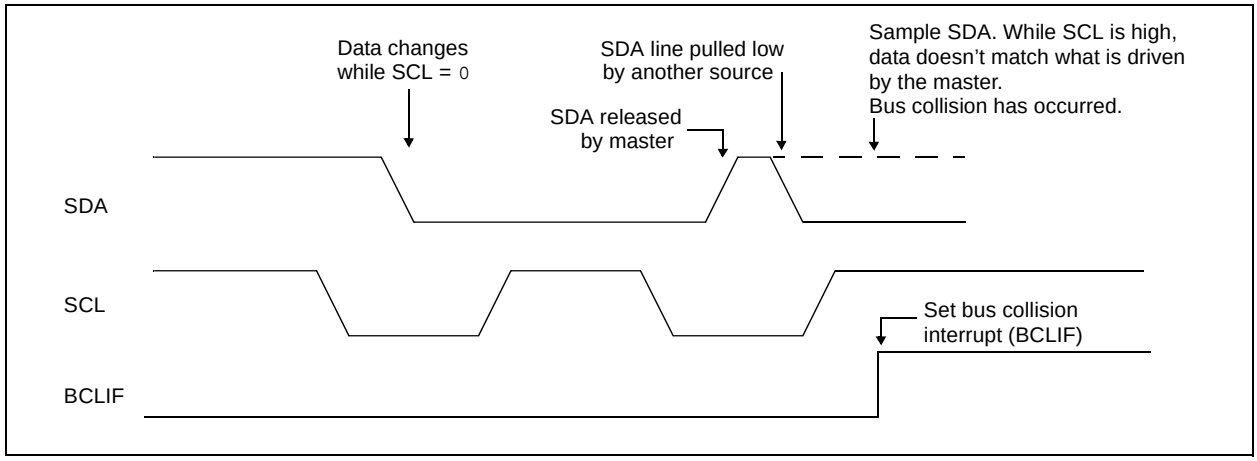
If a START, Repeated START, STOP, or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted, and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine, 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 18-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



18.4.17.1 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 18-26).

b) SCL is sampled low before SDA is asserted low (Figure 18-27).

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 18-26).

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 18-28). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The baud rate generator is then reloaded and counts down to '0', and during this time, if the SCL pins are sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

Note:	The reason that bus collision is not a factor during a START condition is that no two bus masters can assert a START condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the START condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated START or STOP conditions.

FIGURE 18-26: BUS COLLISION DURING START CONDITION (SDA ONLY)

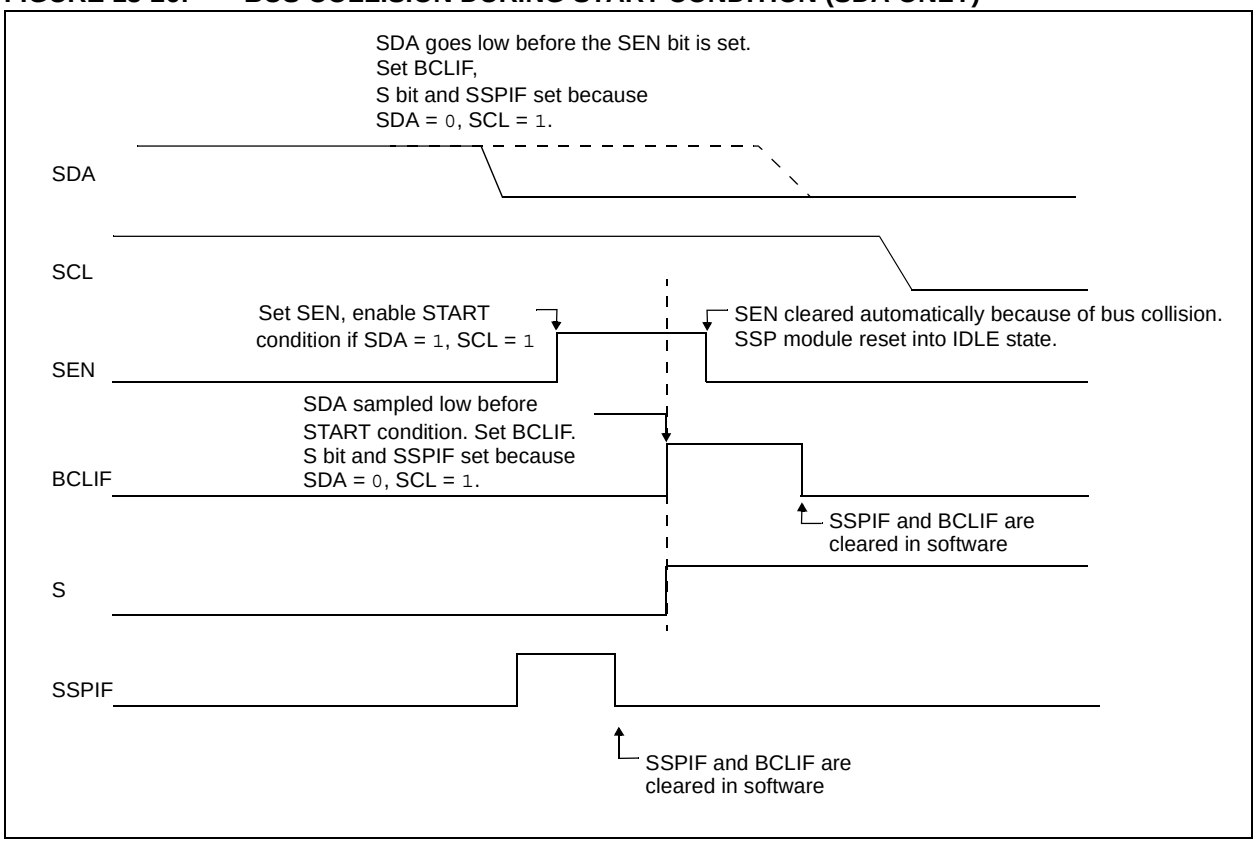


FIGURE 18-27: BUS COLLISION DURING START CONDITION (SCL = 0)

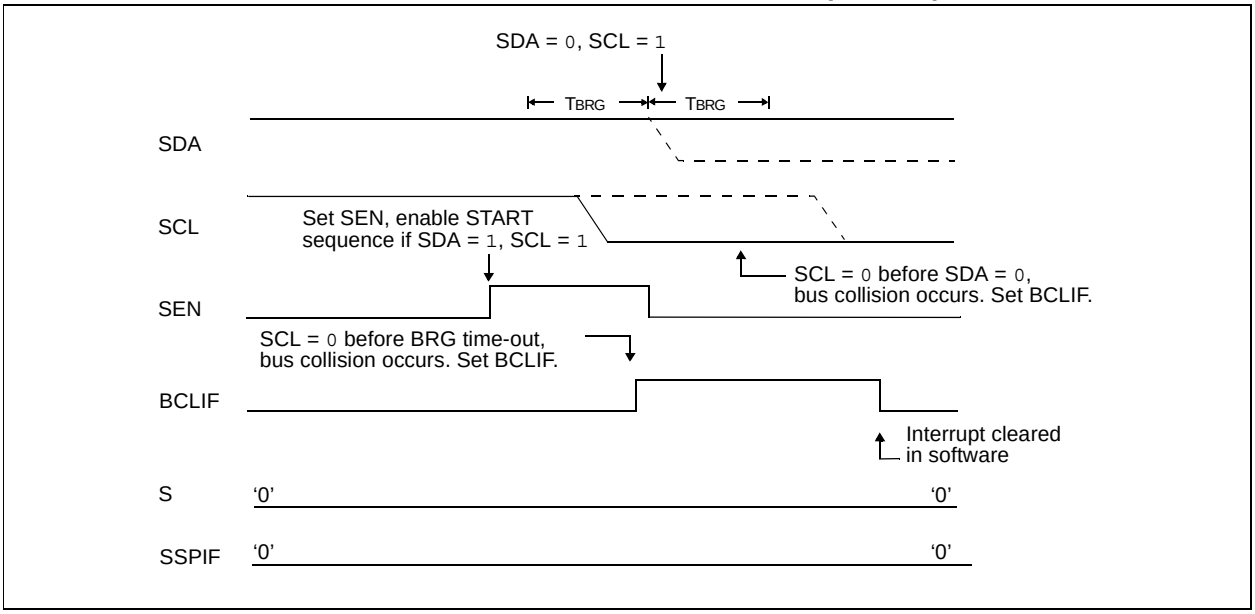
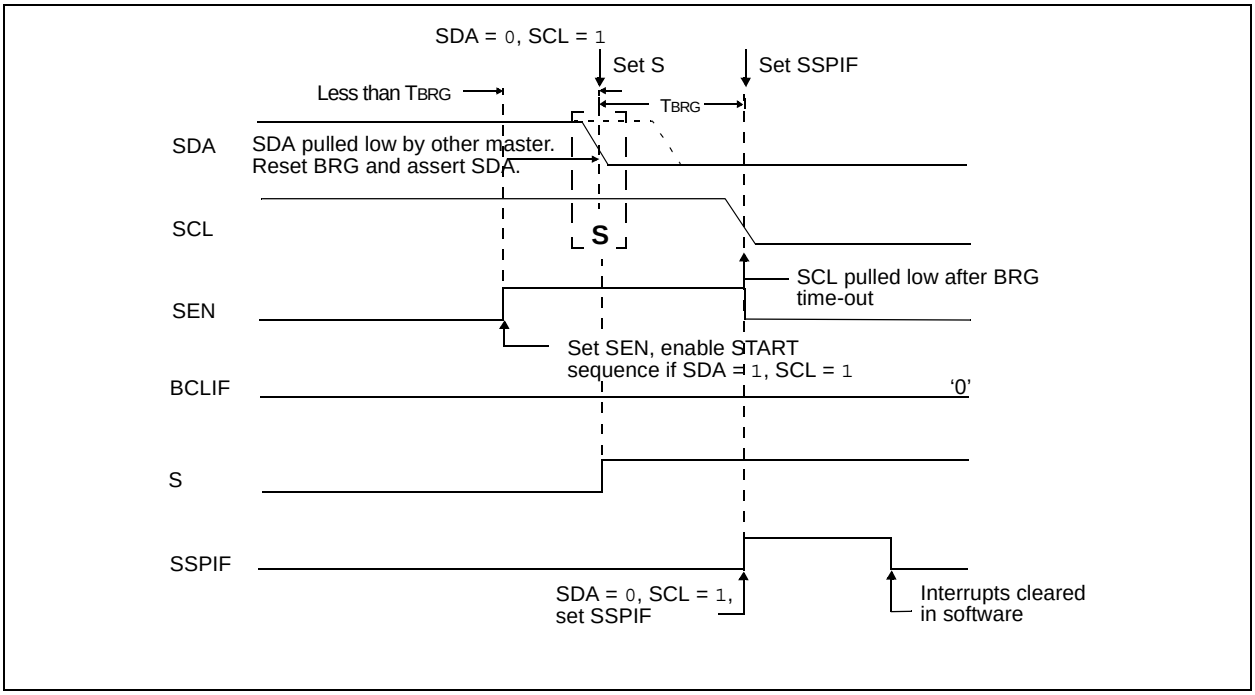


FIGURE 18-28: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION



18.4.17.2 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 18-29). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high to low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high to low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated START condition, see Figure 18-30.

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 18-29: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

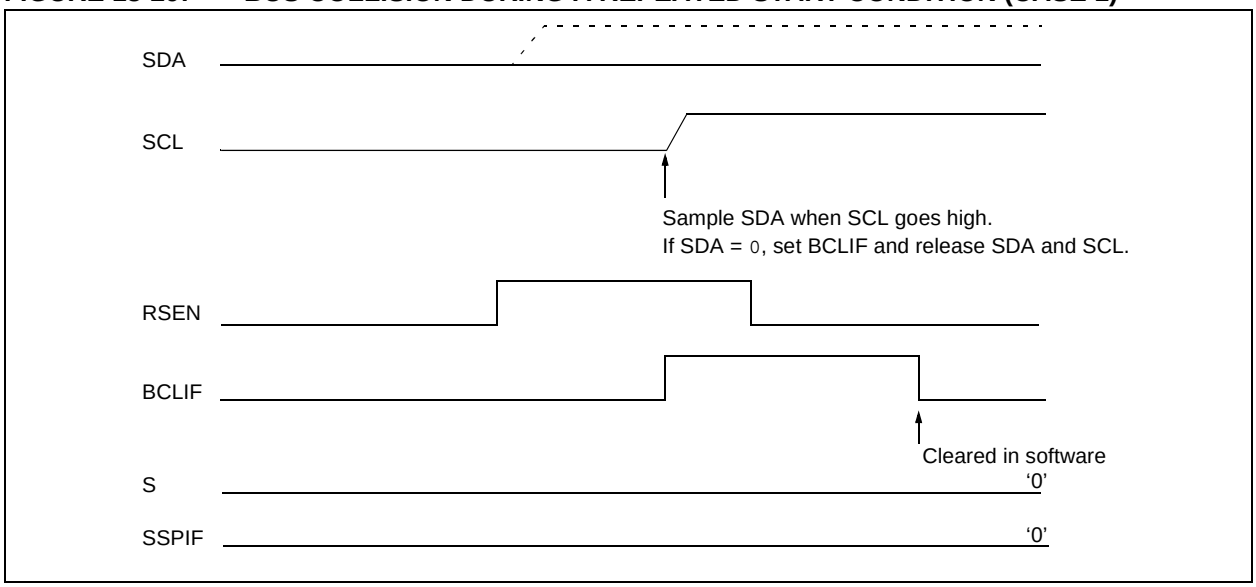
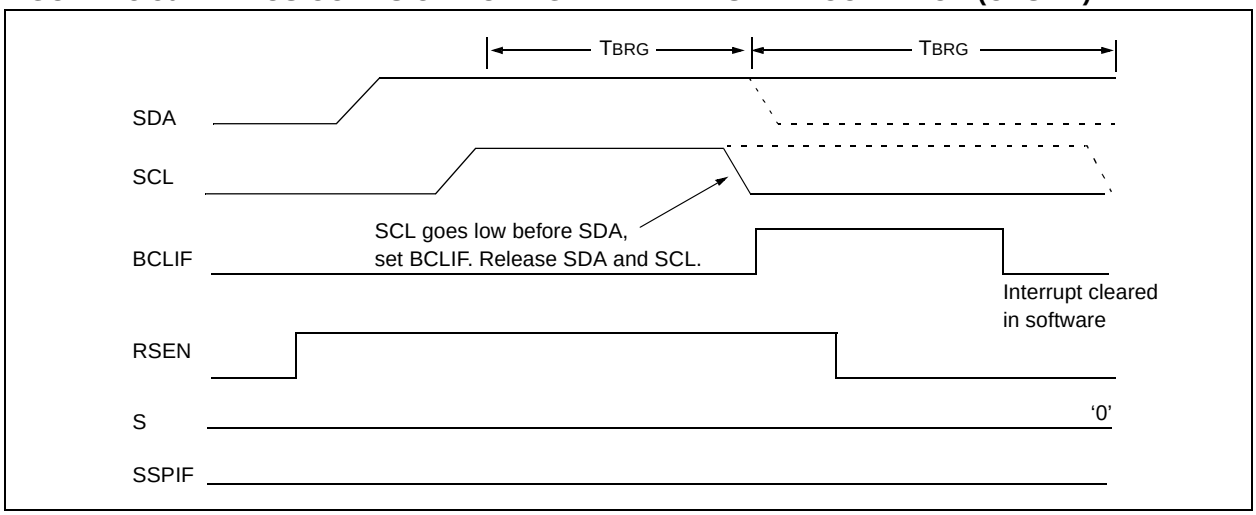


FIGURE 18-30: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



18.4.17.3 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 18-31). 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 18-32).

FIGURE 18-31: BUS COLLISION DURING A STOP CONDITION (CASE 1)

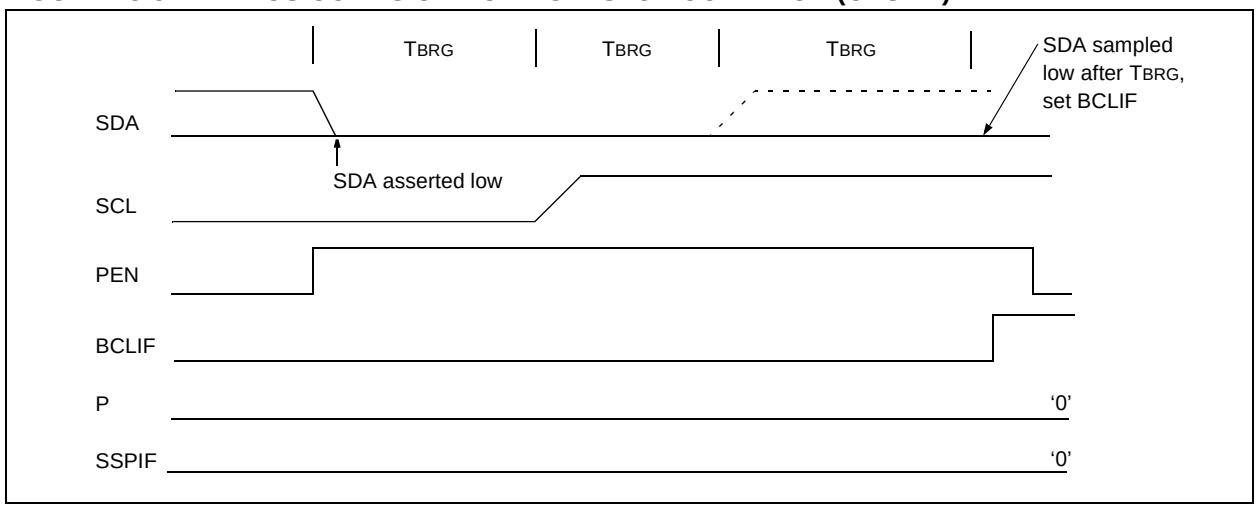
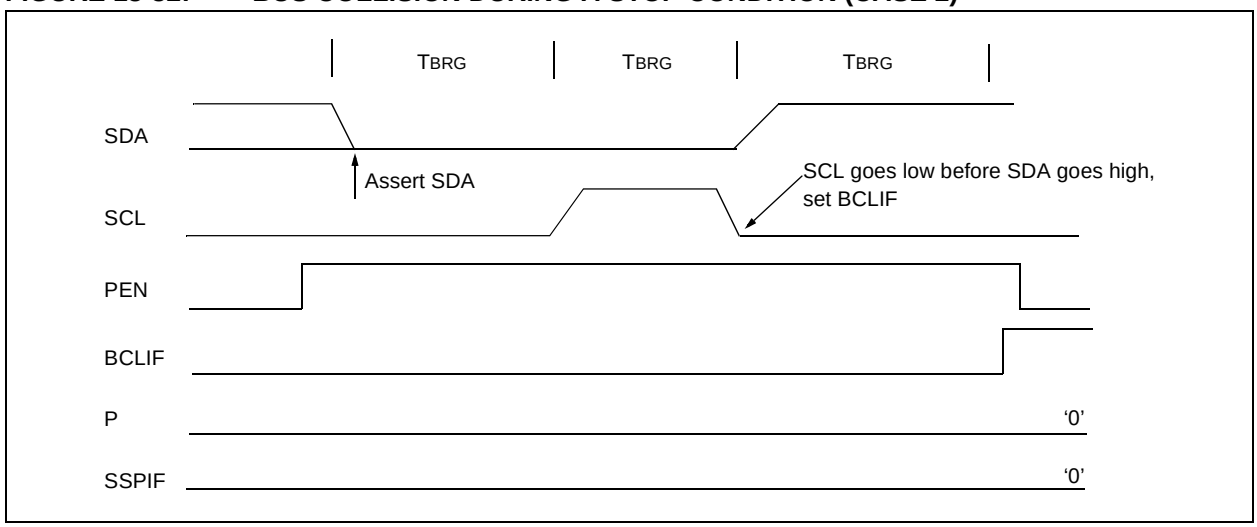


FIGURE 18-32: BUS COLLISION DURING A STOP CONDITION (CASE 2)



NOTES:

19.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN bus) systems.

The USART can be configured in the following modes:

- Asynchronous (full-duplex) with:
  - Auto wake-up on character reception
  - Auto baud calibration
  - 12-bit Break character transmission
- Synchronous - Master (half-duplex) with selectable clock polarity
- Synchronous - Slave (half-duplex) with selectable clock polarity

The pins of USART1 and USART2 are multiplexed with the functions of PORTC (RC6/TX1/CK1 and RC7RX1/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

Note:	The USART control will automatically reconfigure the pin from input to output as needed.

The operation of each enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTAx)
- Receive Status and Control (RCSTAx)
- Baud Rate Control (BAUDCONx)

These are detailed on the following pages in Register 19-1, Register 19-2 and Register 19-3, respectively.

Note:	Throughout this section, references to register and bit names that may be associated with a specific 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 19-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	SENDB	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 **SENDB:** Send Break Character bit

Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission completed

Synchronous mode:
Don't care.

bit 2 **BRGH:** High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode.

bit 1 **TRMT:** Transmit Shift Register Status bit

1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data

Can be address/data bit or a parity bit.

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: 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 (held in RESET)

bit 6 **RX9:** 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode - Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode - Slave:
Don't care.

bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
1 = Enables receiver
0 = Disables receiver
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and loads the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 9-bit (RX9 = 0):
Don't care.

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be updated by reading RCREGx register and 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

REGISTER 19-3: BAUDCONx: BAUD RATE CONTROL REGISTER

U-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **RCIDL**: Receive Operation IDLE Status bit
1 = Receive operation is IDLE
0 = Receive operation is active

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **SCKP**: Synchronous Clock Polarity Select bit
Asynchronous mode:
Unused in this mode.
Synchronous mode:
1 = IDLE state for clock (CK) is a high level
0 = IDLE state for clock (CK) is a low level

bit 3 **BRG16:** 16-bit Baud Rate Register Enable bit
1 = 16-bit baud rate generator - SPBRGHx and SPBRGx
0 = 8-bit baud rate generator - SPBRGx only (Compatible mode), SPBRGHx value ignored

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = USART will continue to sample the RX pin - interrupt generated on falling edge; bit cleared in hardware on following rising edge
0 = RX pin not monitored or rising edge detected
Synchronous mode:
Unused in this mode.

bit 0 **ABDEN**: Auto Baud Detect Enable bit
Asynchronous mode:
1 = Enable baud rate measurement on the next character - requires reception of a Sync field (55h); cleared in hardware upon completion
0 = Baud rate measurement disabled or completed
Synchronous mode:
Unused in this mode.

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

19.1 USART Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit or 16-bit generator that supports both the Asynchronous and Synchronous modes of the USART. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCONx<3>) selects 16-bit mode.

The SPBRGHx:SPBRGx register pair controls the period of a free running timer. In Asynchronous mode, bits BRGH (TXSTAx<2>) and BRG16 also control the baud rate. In Synchronous mode, bit BRGH is ignored. Table 19-1 shows the formula for computation of the baud rate for different USART modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGHx:SPBRGx registers can be calculated using the formulas in Table 19-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 19-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 19-2. It may be advantageous to use the high baud rate (BRGH = 1) or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGHx:SPBRGx registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

19.1.1 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 RX pin.

TABLE 19-1: BAUD RATE FORMULAS

Configuration Bits			BRG/USART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	FOSC / [64 (n+1)]
0	0	1	8-bit/Asynchronous	FOSC / [16 (n+1)]
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	FOSC / [4 (n+1)]
1	0	x	8-bit/Synchronous	
1	1	x	16-bit/Synchronous	

Legend: x = Don't care, n = value of SPBRGHx:SPBRGx register pair

EXAMPLE 19-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

Desired Baud Rate = FOSC / (64 ([SPBRGHx:SPBRGx] + 1))

Solving for SPBRGHx:SPBRGx:

X = ((FOSC / Desired Baud Rate)/64) – 1
= ((16000000 / 9600) / 64) – 1
= [25.042] = 25

Calculated Baud Rate= 16000000 / (64 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate) / Desired Baud Rate
= (9615 – 9600) / 9600 = 0.16%

TABLE 19-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other RESETS
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 -010	0000 -010
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

TABLE 19-3: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.16	207	300	-0.16	103	300	-0.16	51
1.2	1.202	0.16	51	1201	-0.16	25	1201	-0.16	12
2.4	2.404	0.16	25	2403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	300	-0.16	207
1.2	1.202	0.16	207	1201	-0.16	103	1201	-0.16	51
2.4	2.404	0.16	103	2403	-0.16	51	2403	-0.16	25
9.6	9.615	0.16	25	9615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

TABLE 19-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.04	832	300	-0.16	415	300	-0.16	207
1.2	1.202	0.16	207	1201	-0.16	103	1201	-0.16	51
2.4	2.404	0.16	103	2403	-0.16	51	2403	-0.16	25
9.6	9.615	0.16	25	9615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.01	3332	300	-0.04	1665	300	-0.04	832
1.2	1.200	0.04	832	1201	-0.16	415	1201	-0.16	207
2.4	2.404	0.16	415	2403	-0.16	207	2403	-0.16	103
9.6	9.615	0.16	103	9615	-0.16	51	9615	-0.16	25
19.2	19.231	0.16	51	19230	-0.16	25	19230	-0.16	12
57.6	58.824	2.12	16	55555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

19.1.2 AUTO BAUD RATE DETECT

The enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 19-1) begins whenever a START bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RX signal, the RX signal is timing the BRG. In ABD mode, the internal baud rate generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a START bit. The auto baud detect must receive a byte with the value 55h (ASCII "U", which is also the LIN bus Sync character), in order to calculate the proper bit rate. The measurement is taken over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a START bit, the SPBRGx begins counting up using the preselected clock source on the first rising edge of RX. After eight bits on the RX pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGHx:SPBRGx register pair. Once the 5th edge is seen (this should correspond to the STOP bit), the ABDEN bit is automatically cleared.

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. Note that the BRG clock will be configured by the BRG16 and BRGH bits. Independent of the BRG16 bit setting, both the SPBRGx and SPBRGHx will be used as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGHx register. Refer to Table 19-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the USART state machine is held in IDLE. The RCIF interrupt is set once the fifth rising edge on RX is detected. The value in the RCREGx needs to be read to clear the RCIF interrupt. RCREGx content should be discarded.

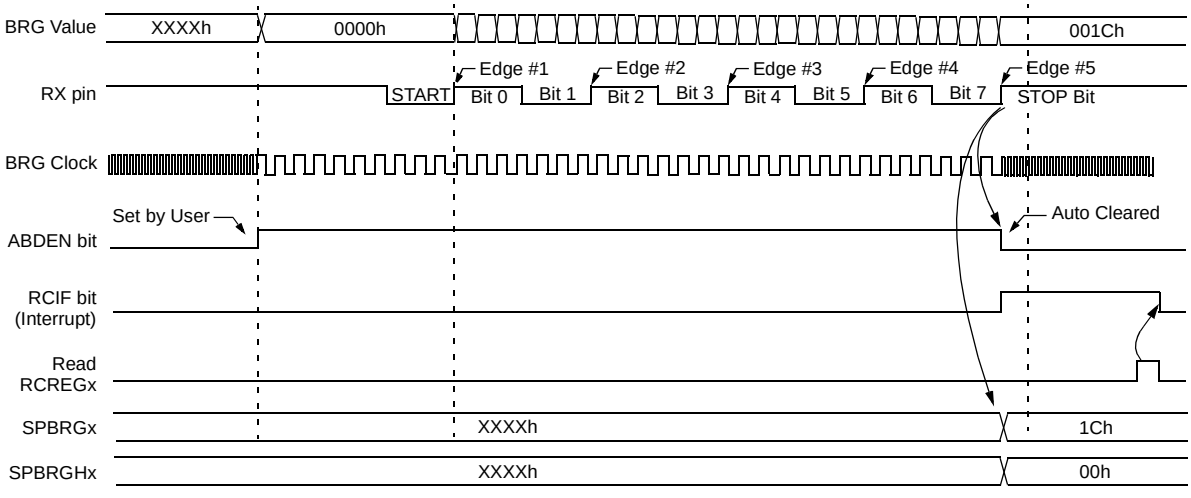
Note 1:	If the WUE bit is set with the ABDEN bit, auto baud rate detection will occur on the byte *following* the Break character.
2:	It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and USART baud rates are not possible due to bit error rates. Overall system timing and communication baud rates must be taken into consideration when using the auto baud rate detection feature.

TABLE 19-4: BRG COUNTER CLOCK RATES

BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

Note: During the ABD sequence, SPBRGx and SPBRGHx are both used as a 16-bit counter, independent of BRG16 setting.

FIGURE 19-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the USART module to be configured in Asynchronous mode and WUE = 0.

19.2 USART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTAx<4>). In this mode, the USART uses standard non-return-to-zero (NRZ) format (one START bit, eight or nine data bits and one STOP bit). The most common data format is 8 bits. An on-chip dedicated 8-bit/16-bit baud rate generator can be used to derive standard baud rate frequencies from the oscillator.

The 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 x16 or x64 of the bit shift rate depending on the BRGH and BRG16 bits (TXSTAx<2> and BAUDCONx<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the USART module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto Wake-up on Sync Break Character
- 12-bit Break Character Transmit
- Auto Baud Rate Detection

19.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 19-2. 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 TXIF (PIR1<4>) is set. This interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. Flag bit TXIF is not cleared immediately upon loading the Transmit Buffer register, TXREGx. TXIF becomes valid in the second instruction cycle following the load instruction. Polling TXIF immediately following a load of TXREGx will return invalid results.

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 SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXIE.
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 TXIF.
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).

If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 19-2: USART TRANSMIT BLOCK DIAGRAM

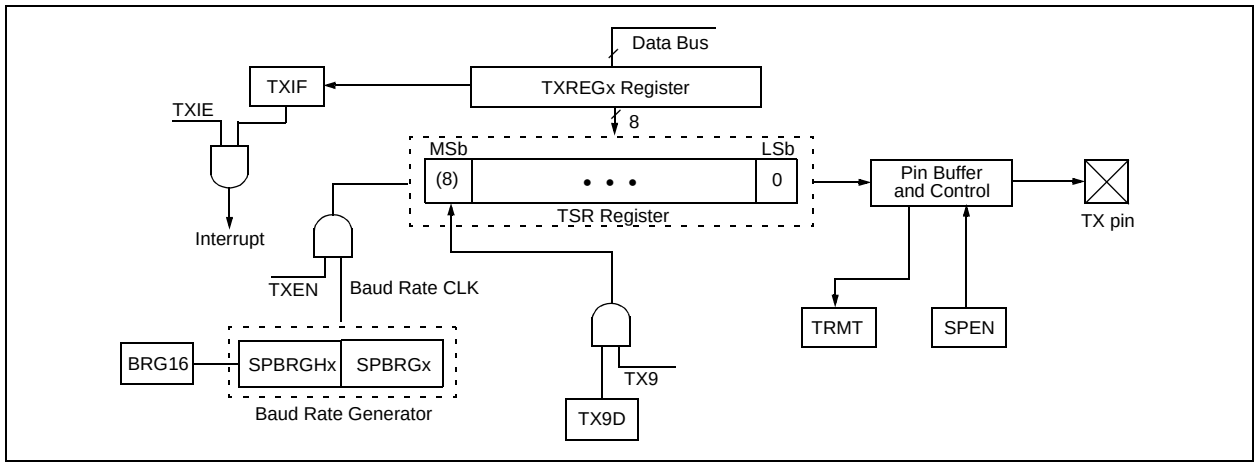


FIGURE 19-3: ASYNCHRONOUS TRANSMISSION

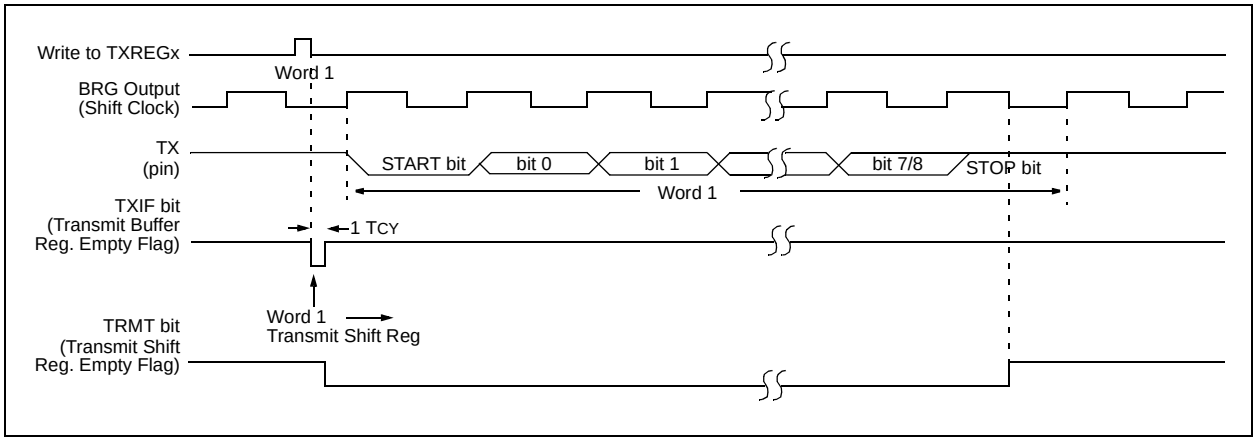
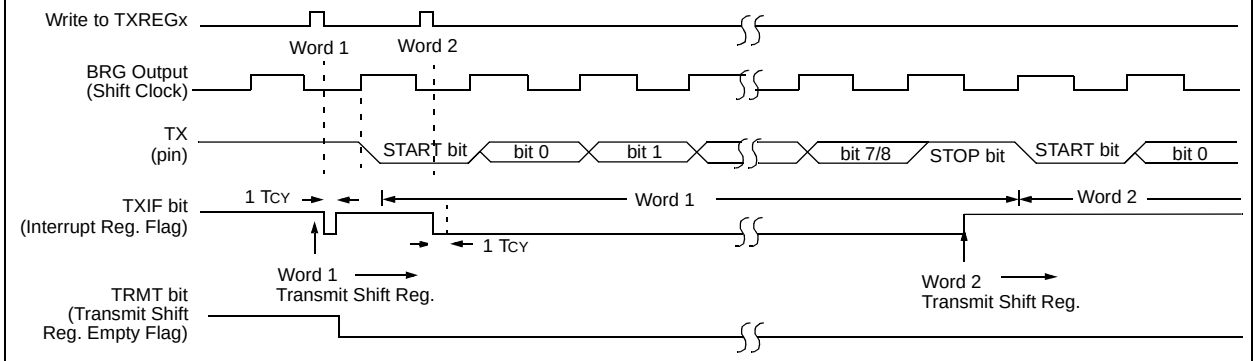


FIGURE 19-4: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 19-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other RESETS
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	-000 -000	-000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 -000	-000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	-111 -111	-111 -111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREGx	USART Transmit Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

19.2.2 USART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 19-5. The data is received on the RXx pin and drives the data recovery block. The data recovery block is actually a high speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit RCIE.
4. If 9-bit reception is desired, set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
7. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREGx register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

19.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the 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 RCIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCIE and GIE bits are set.
8. Read the RCSTAx register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREGx to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 19-5: USART RECEIVE BLOCK DIAGRAM

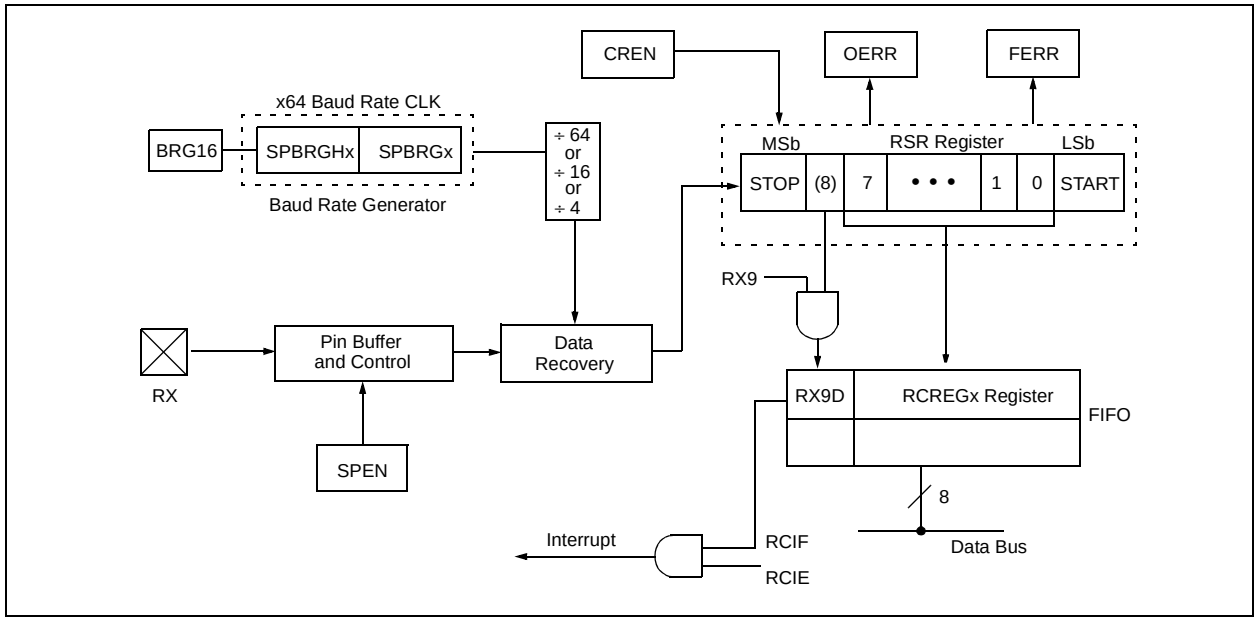
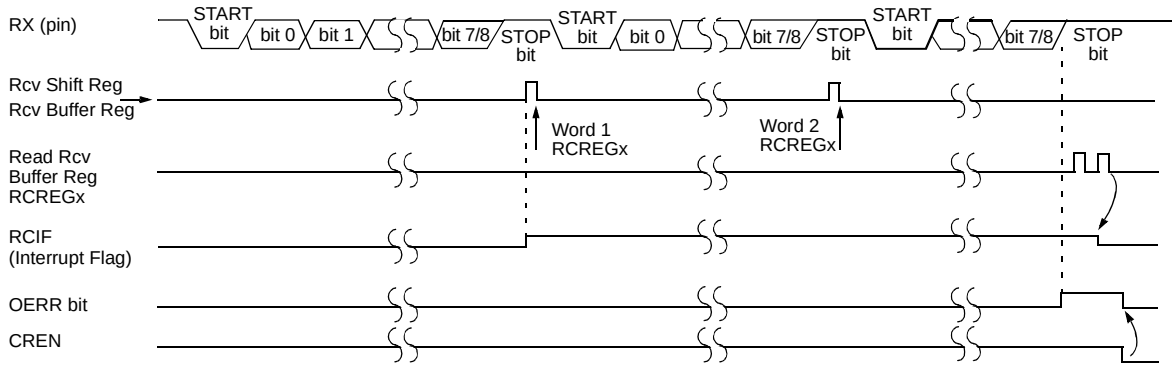


FIGURE 19-6: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The RCREGx (receive buffer) is read after the third word causing the OERR (overrun) bit to be set.

TABLE 19-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other RESETS
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	1111 1111	1111 1111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx	USART Receive Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

19.2.4 AUTO WAKE-UP ON SYNC BREAK CHARACTER

During SLEEP mode, all clocks to the USART are suspended. Because of this, the baud rate generator is inactive and a proper byte reception cannot be performed. The auto wake-up feature allows the controller to wake-up due to activity on the RX/DT line while the USART is operating in Asynchronous mode.

The auto wake-up feature is enabled by setting the WUE bit (BAUDCONx<1>). Once set, the typical receive sequence on RX/DT is disabled, and the USART remains in an IDLE state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RX/DT line. (This coincides with the start of a Sync Break or a Wake-up Signal character for the LIN protocol.)

Following a wake-up event, the module generates an RCIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 19-7) and asynchronously, if the device is in SLEEP mode (Figure 19-8). The interrupt condition is cleared by reading the RCREGx register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RX line following the wake-up event. At this point, the USART module is in IDLE mode and returns to normal operation. This signals to the user that the Sync Break event is over.

19.2.4.1 Special Considerations Using Auto Wake-up

Since auto wake-up functions by sensing rising edge transitions on RX/DT, information with any state changes before the STOP bit may signal a false end-of-character and cause data or framing errors. To work properly, therefore, the initial character in the transmission must be all '0's. This can be 00h (8 bytes) for standard RS-232 devices, or 000h (12 bits) for LIN bus.

Oscillator start-up time must also be considered, especially in applications using oscillators with longer start-up intervals (i.e., XT or HS mode). The Sync Break (or Wake-up Signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the USART.

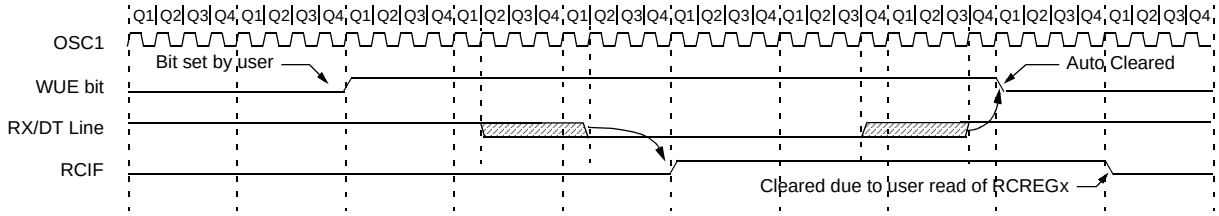
19.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the USART in an IDLE mode. The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared after this when a rising edge is seen on RX/DT. The interrupt condition is then cleared by reading the RCREGx register. Ordinarily, the data in RCREGx will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set) and the RCIF flag is set should not be used as an indicator of the integrity of the data in RCREGx. Users should consider implementing a parallel method in firmware to verify received data integrity.

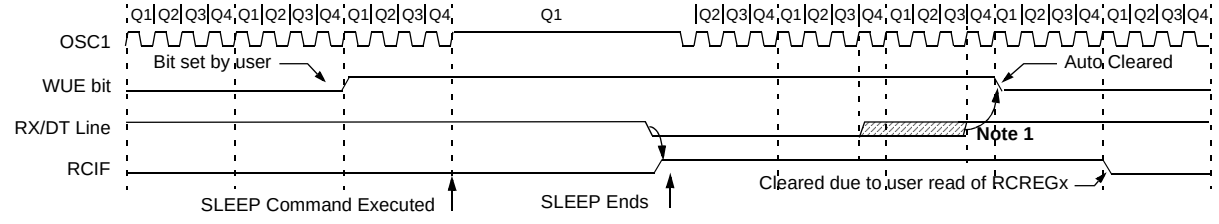
To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the SLEEP mode.

FIGURE 19-7: AUTO WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION



Note: The USART remains in IDLE while the WUE bit is set.

FIGURE 19-8: AUTO WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the auto clear of the WUE bit can occur while the *stposc* signal is still active. This sequence should not depend on the presence of Q clocks.

2: The USART remains in IDLE while the WUE bit is set.

19.2.5 BREAK CHARACTER SEQUENCE

The enhanced USART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. The Break character transmit consists of a START bit, followed by twelve ‘0’ bits and a STOP bit. The frame Break character is sent whenever the SENDB and TXEN bits (TXSTAx<3> and TXSTAx<5>) are set while the Transmit Shift register is loaded with data. Note that the value of data written to TXREGx will be ignored and all ‘0’s will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding STOP bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

Note that the data value written to the TXREGx for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or IDLE, just as it does during normal transmission. See Figure 19-9 for the timing of the Break character sequence.

19.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an auto baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the USART for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.
3. Load the TXREGx with a dummy character to initiate transmission (the value is ignored).
4. Write ‘55h’ to TXREGx to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREGx becomes empty, as indicated by the TXIF, the next data byte can be written to TXREGx.

19.2.6 RECEIVING A BREAK CHARACTER

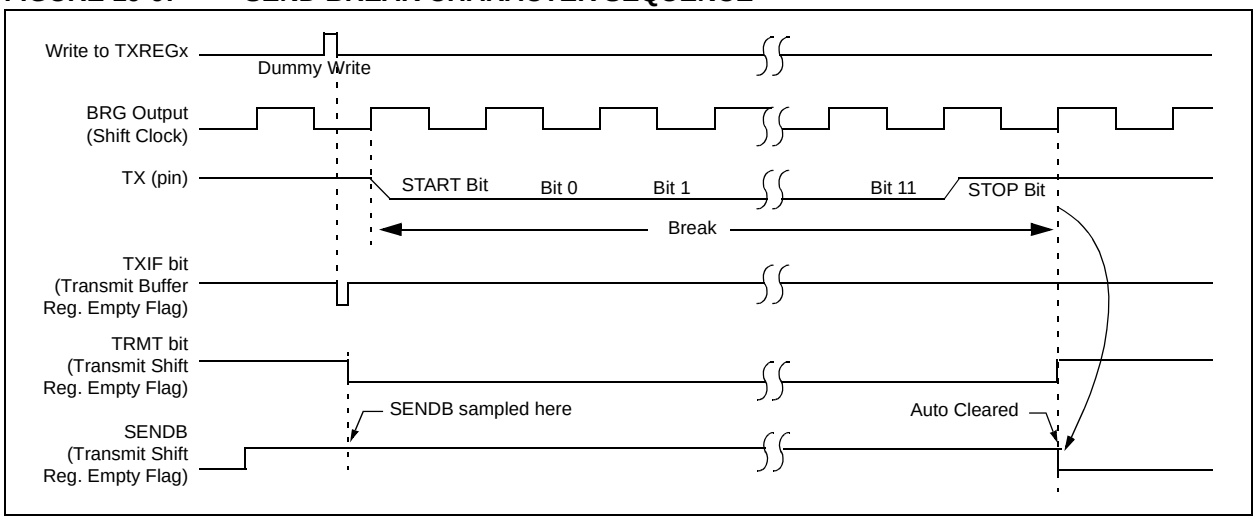
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the STOP bit transition to be at the correct sampling location (13 bits for Break versus START bit and 8 data bits for typical data).

The second method uses the auto wake-up feature described in Section 19.2.4. By enabling this feature, the USART will sample the next two transitions on RX/DT, cause an RCIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the auto baud rate detect feature. For both methods, the user can set the ABD bit once the TXIF interrupt is observed.

FIGURE 19-9: SEND BREAK CHARACTER SEQUENCE



19.3 USART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTAx<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTAx<4>). In addition, enable bit SPEN (RCSTAx<7>) is set in order to configure the TXx and RXx pins to CK (clock) and DT (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK line. Clock polarity is selected with the SCKP bit (BAUDCONx<4>); setting SCKP sets the IDLE state on CK as high, while clearing the bit sets the IDLE state as low. This option is provided to support Microwire® devices with this module.

19.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 19-2. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREGx. The TXREGx register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREGx (if available).

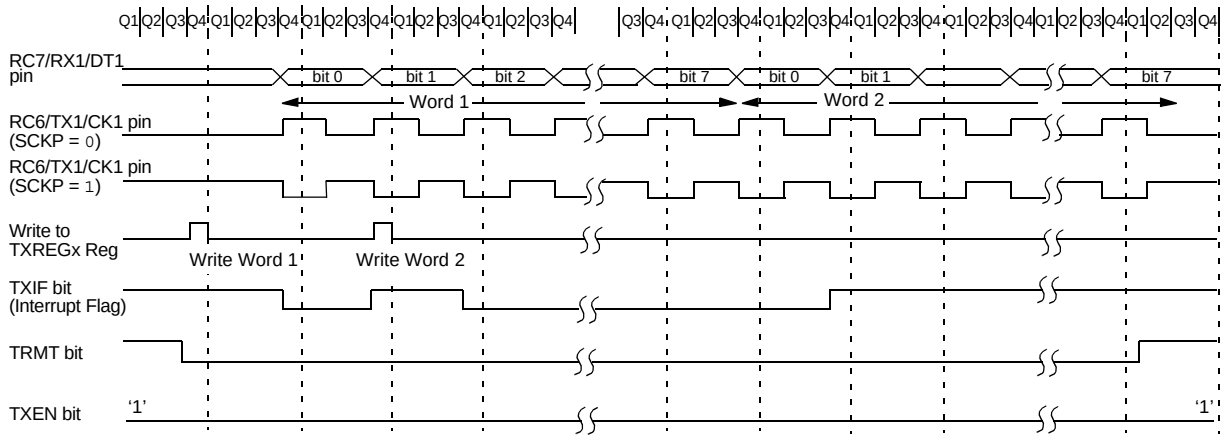
Once the TXREGx register transfers the data to the TSR register (occurs in one TCYCLE), the TXREGx is empty and interrupt bit TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE, 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. 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 SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 19-10: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRGx = 0, continuous transmission of two 8-bit words.

FIGURE 19-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

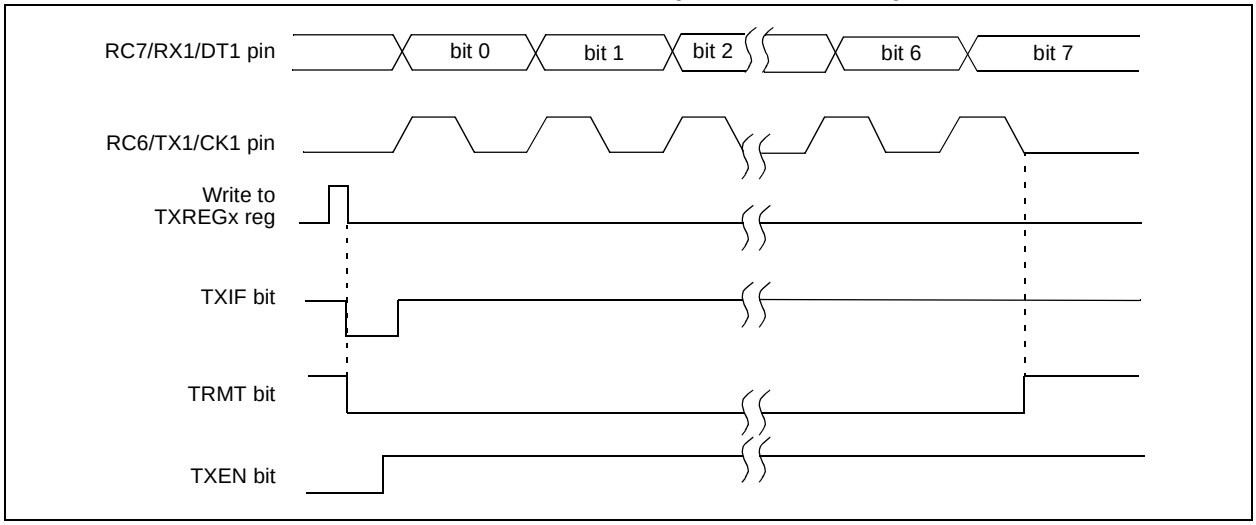


TABLE 19-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	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	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	1111 1111	1111 1111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREGx	USART Transmit Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-0 0-00	-1-0 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

19.3.2 USART SYNCHRONOUS MASTER RECEPTION

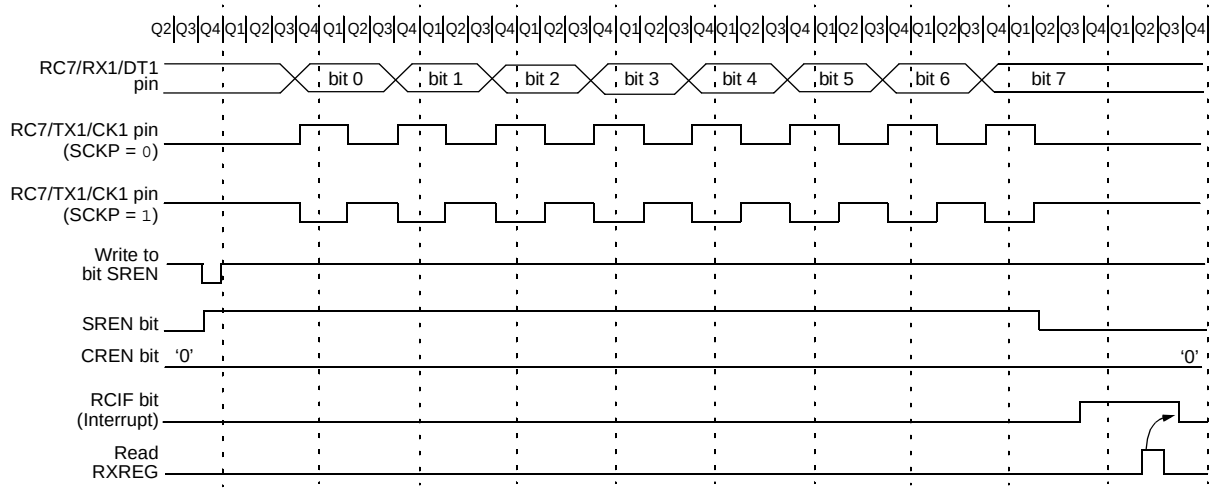
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTAx<5>), or the Continuous Receive Enable bit, CREN (RCSTAx<4>). Data is sampled on the RXx pin on the falling edge of the clock.

If enable bit SREN is set, only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set, then CREN takes precedence.

To set up a Synchronous Master Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, set enable bit RCIE.
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 RCIF will be set when reception is complete and an interrupt will be generated if the enable bit RCIE was set.
8. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREGx register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 19-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

TABLE 19-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	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	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	1111 1111	1111 1111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx	USART Receive Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-0 0-00	-1-0 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

19.4 USART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any Low Power mode.

19.4.1 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 TXREGx and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREGx register.
c) Flag bit TXIF 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 TXIF will now be set.
e) If enable bit TXIE 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 TXIE.
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 19-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other RESETS
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	-000 -000	-000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 -000	-000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	-111 -111	-111 -111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREGx	USART Transmit Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

19.4.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical except in the case of SLEEP or any IDLE mode and bit SREN, which is a “don't care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering SLEEP or any IDLE mode, then a word may be received while in this Low Power mode. Once the word is received, the RSR register will transfer the data to the RCREGx register; if the RCIE enable bit is set, the interrupt generated will wake the chip from Low Power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete. An interrupt will be generated if enable bit RCIE was set.
6. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREGx register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 19-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	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	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	1111 1111	1111 1111
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREGx	USART Receive Register								0000 0000	0000 0000
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCONx	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-0 0-00	-1-0 0-00
SPBRGHx	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRGx	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

NOTES:

20.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The analog-to-digital (A/D) converter module has 12 inputs for the PIC18F6X2X devices and 16 for the PIC18F8X2X devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

A new feature for the A/D converter is the addition of programmable acquisition time. This feature allows the user to select a new channel for conversion and setting the GO/$\overline{\text{DONE}}$ bit immediately. When the GO/$\overline{\text{DONE}}$ bit is set, the selected channel is sampled for the programmed acquisition time before a conversion is actually started. This removes the firmware overhead that may have been required to allow for an acquisition (sampling) period (see Register 20-3 and Section 20.4).

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 20-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 20-2, configures the functions of the port pins. The ADCON2, shown in Register 20-3, configures the A/D clock source, justification and auto-acquisition time.

REGISTER 20-1: 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 PIC18F6X2X (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 20-2: 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 PIC18F8X2X 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 20-3: ADCON2 REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified
0 = Left justified

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

bit 5-3 **ACQT2:ACQT0:** A/D Acquisition Time Select bits
000 = 0 TAD(1)
001 = 2 TAD
010 = 4 TAD
011 = 6 TAD
100 = 8 TAD
101 = 12 TAD
110 = 16 TAD
111 = 20 TAD

bit 2-0 **ADCS2:ADCS0:** A/D Conversion Clock Select bits
000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC (clock derived from A/D RC oscillator)(1)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC (clock derived from A/D RC oscillator)(1)

Note 1: If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

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.

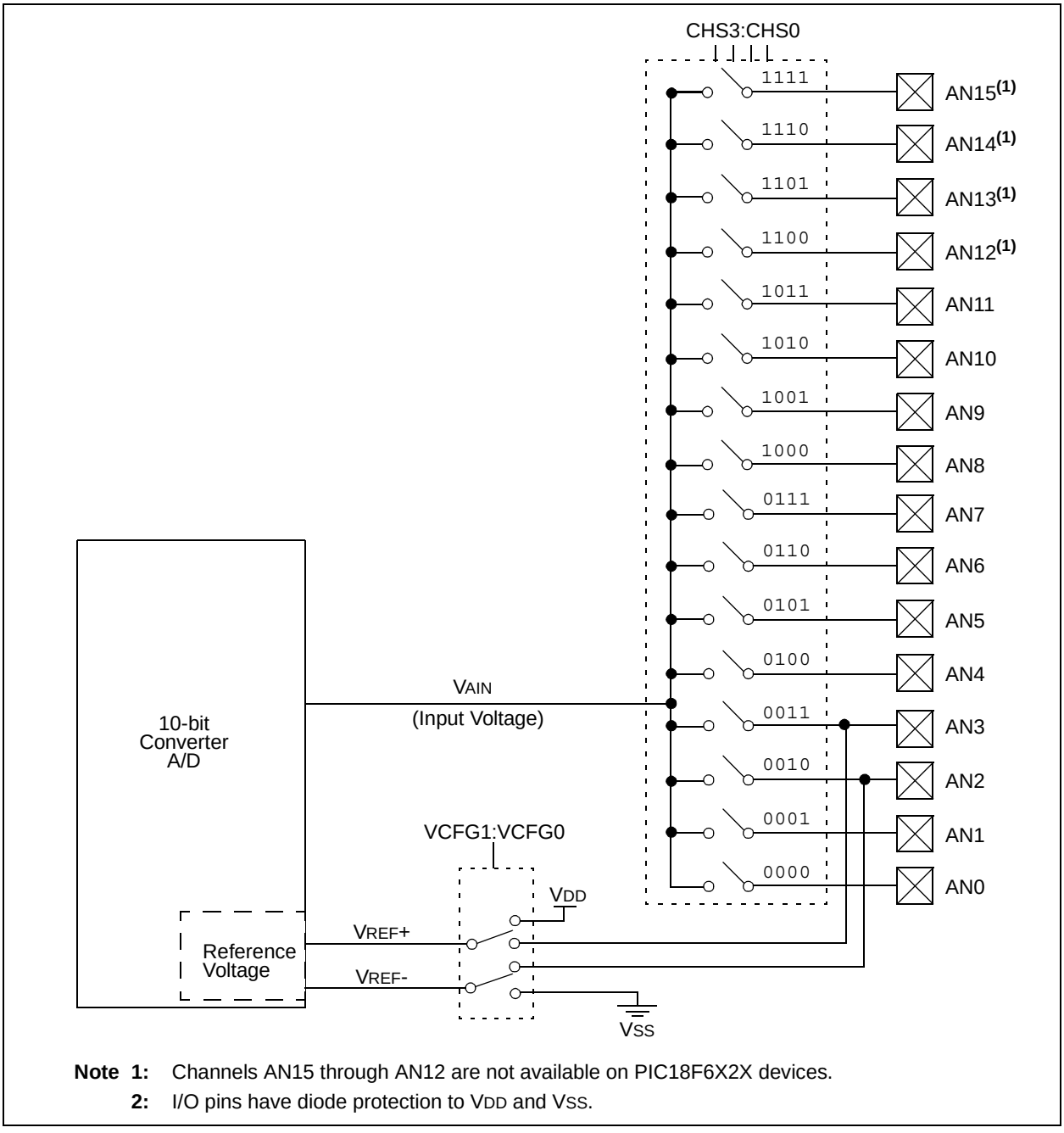
The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

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 20-1.

FIGURE 20-1: A/D BLOCK DIAGRAM



Note 1: Channels AN15 through AN12 are not available on PIC18F6X2X devices.

2: I/O pins have diode protection to V_{DD} and V_{SS}.

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.

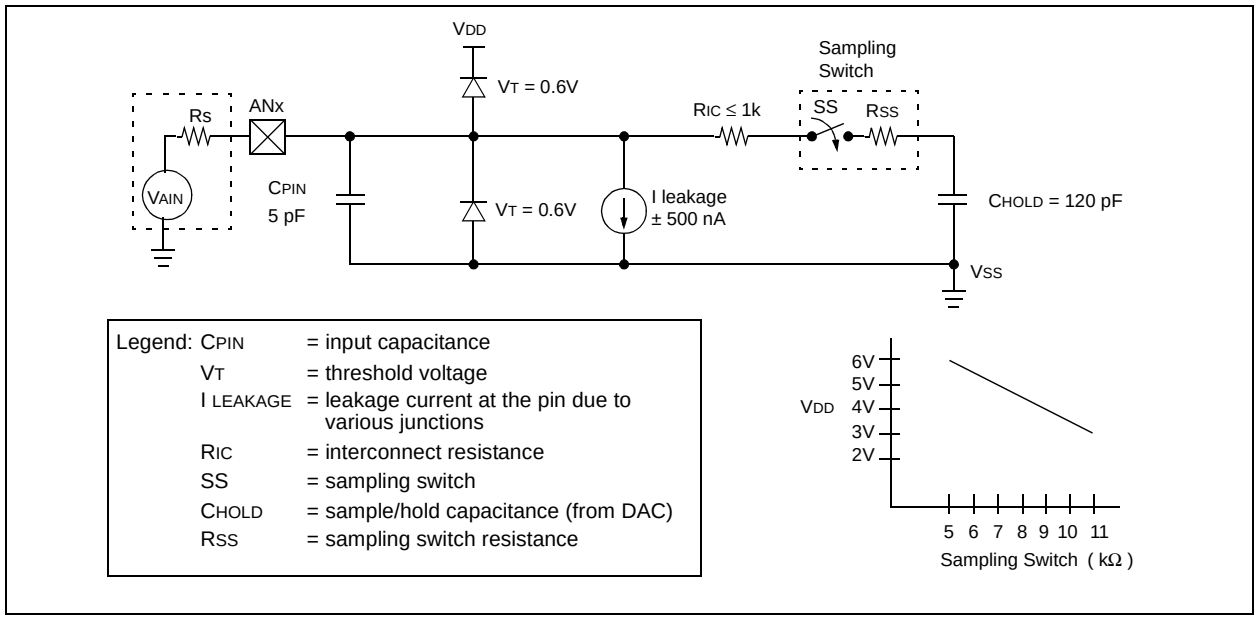
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as an input. To determine acquisition time, see Section 20.1. After this acquisition time has elapsed, 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 (not required in case of auto-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 next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2 TAD is required before next acquisition starts.

FIGURE 20-2: ANALOG INPUT MODEL



20.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 20-2. The source impedance (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 kΩ.** 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 20-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 20-1 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 kΩ
Conversion Error	≤	1/2 LSb
V_{DD}	=	5V → Rss = 7 kΩ
Temperature	=	50°C (system max.)
V_{HOLD}	=	0V @ time = 0

EQUATION 20-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 20-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(120\ pF)(1\ k\Omega + R_{SS} + R_S)\ \ln(1/2047)$$

EXAMPLE 20-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

Temperature coefficient is only required for temperatures > 25°C.

$T_{ACQ} = 2\ \mu s + T_C + [(Temp - 25°C)(0.05\ \mu s/°C)]$

$T_C = -C_{HOLD}(R_{IC} + R_{SS} + R_S)\ \ln(1/2047)$
$-120\ pF\ (1\ k\Omega + 7\ k\Omega + 2.5\ k\Omega)\ \ln(0.0004885)$
$-120\ pF\ (10.5\ k\Omega)\ \ln(0.0004885)$
$-1.26\ \mu s\ (-7.6241)$
$9.61\ \mu s$

$T_{ACQ} = 2\ \mu s + 9.61\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C)]$
$11.61\ \mu s + 1.25\ \mu s$
$12.86\ \mu s$

20.2 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 20-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

20.3 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 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.

20.4 A/D Conversions

Figure 20-3 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.

20.5 Use of the ECCP2 Trigger

An A/D conversion can be started by the special event trigger of the ECCP2 module. This requires that the 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 and starts a conversion.

If the A/D module is not enabled (ADON is cleared), the special event trigger will be ignored by the A/D module but will still reset the Timer1 (or Timer3) counter.

TABLE 20-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency
Operation	**ADCS2:ADCS0**	**PIC18F6X2X/8X2X**
2 TOSC	000	1.25 MHz
4 TOSC	100	2.50 MHz
8 TOSC	001	5.00 MHz
16 TOSC	101	10.0 MHz
32 TOSC	010	20.0 MHz
64 TOSC	110	40.0 MHz
RC	x11	—

FIGURE 20-3: A/D CONVERSION TAD CYCLES

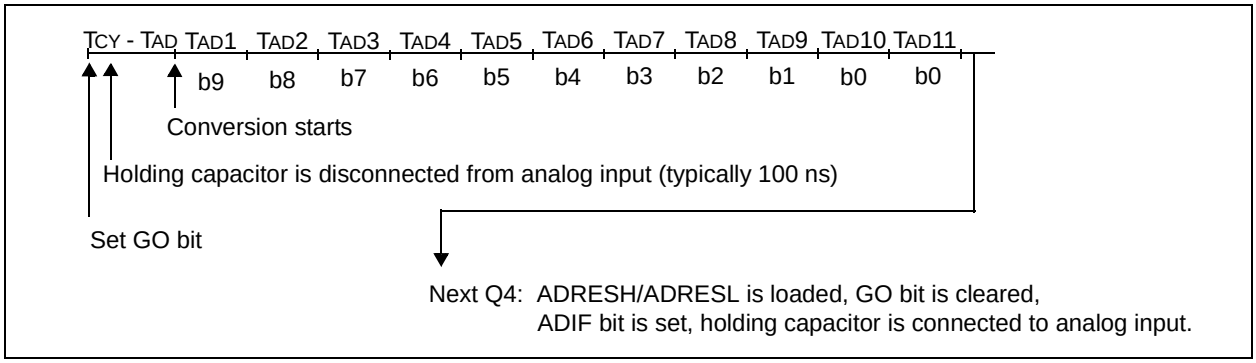


TABLE 20-2: SUMMARY OF A/D REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	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	1111 1111	1111 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	—	ACQT2	ACQT1	ACQT0	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: Implemented on PIC18F8X2X devices only.

21.0 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 22.0) can also be an input to the comparators.

The CMCON register, shown as Register 21-1, controls the comparator input and output multiplexers. A block diagram of the various comparator configurations is shown in Figure 21-1.

REGISTER 21-1: 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 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
When C2INV = 1:
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT**: Comparator 1 Output bit
When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
When C1INV = 1:
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5 **C2INV**: Comparator 2 Output Inversion bit
1 = C2 output inverted
0 = C2 output not inverted

bit 4 **C1INV**: Comparator 1 Output Inversion bit
1 = C1 output inverted
0 = C1 output not inverted

bit 3 **CIS**: Comparator Input Switch bit
When CM2:CM0 = 110:
1 = C1 VIN- connects to RF5/AN10
C2 VIN- connects to RF3/AN8
0 = C1 VIN- connects to RF6/AN11
C2 VIN- connects to RF4/AN9

bit 2-0 **CM2:CM0**: Comparator Mode bits
Figure 21-1 shows the Comparator modes and CM2:CM0 bit settings.

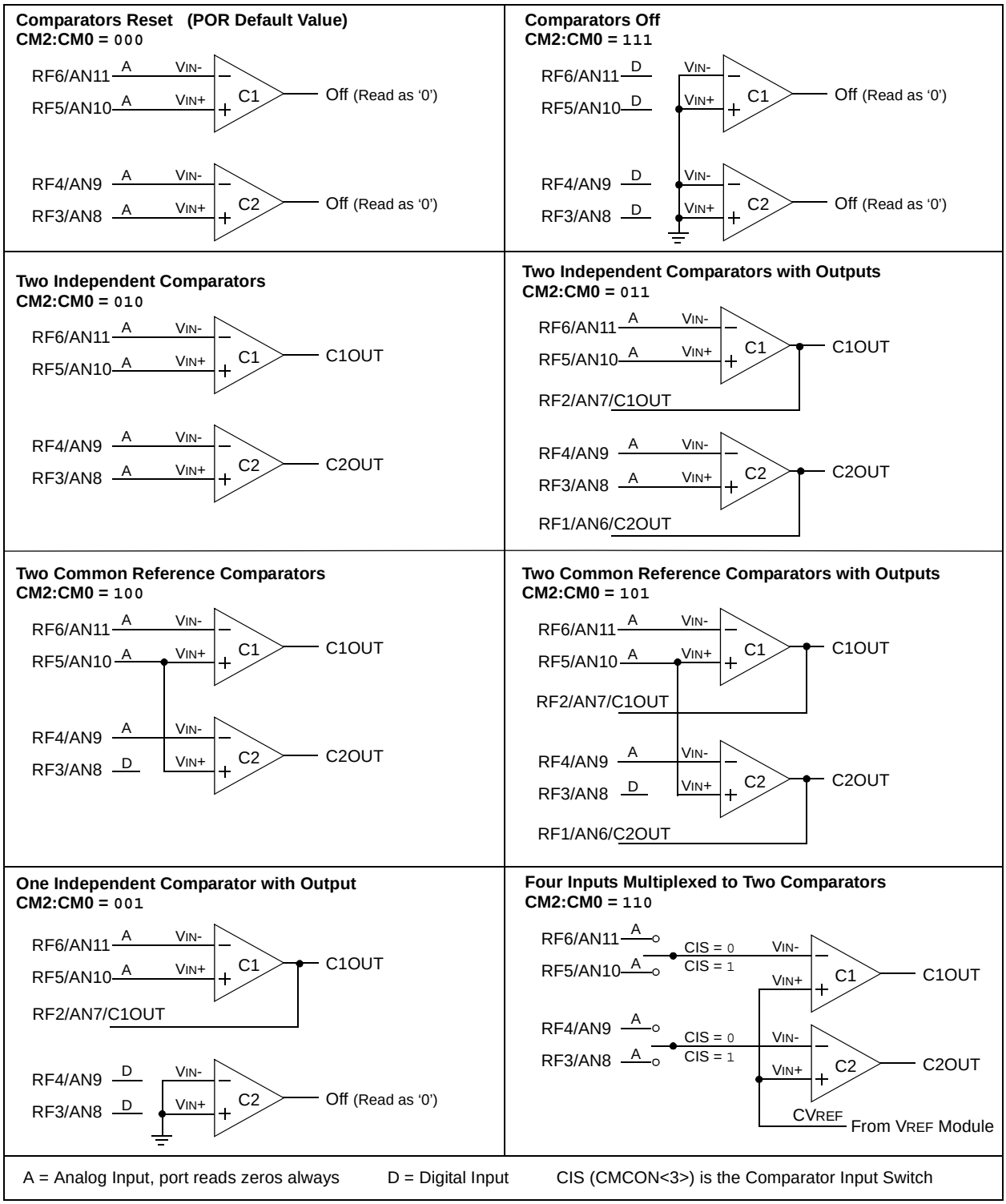
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

21.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 21-1 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 Section 27.0, Electrical Specifications.

Note:	Comparator interrupts should be disabled during a Comparator mode change. Otherwise, a false interrupt may occur.

FIGURE 21-1: COMPARATOR I/O OPERATING MODES



A = Analog Input, port reads zeros always D = Digital Input CIS (CMCON<3>) is the Comparator Input Switch

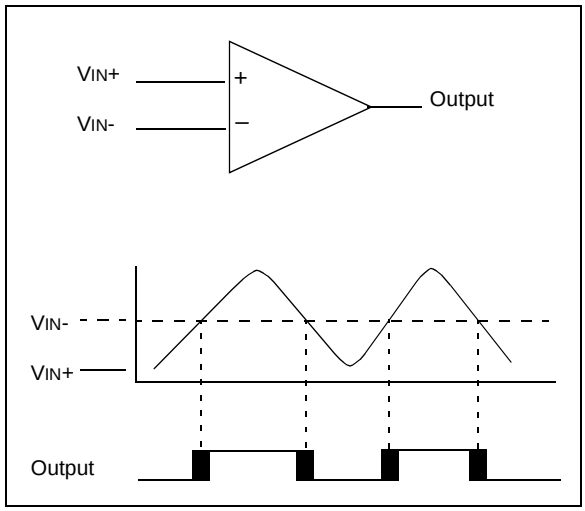
21.2 Comparator Operation

A single comparator is shown in Figure 21-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 21-2 represent the uncertainty due to input offsets and response time.

21.3 Comparator Reference

An external or internal reference signal may be used depending on the comparator operating mode. The analog signal present at VIN- is compared to the signal at VIN+, and the digital output of the comparator is adjusted accordingly (Figure 21-2).

FIGURE 21-2: SINGLE COMPARATOR



21.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same, or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD and can be applied to either pin of the comparator(s).

21.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference for the comparators. Section 22.0 contains a detailed description of the comparator voltage reference module that provides this signal. The internal reference signal is used when comparators are in mode CM<2:0> = 110 (Figure 21-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

21.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise, the maximum delay of the comparators should be used (Section 27.0).

21.5 Comparator Outputs

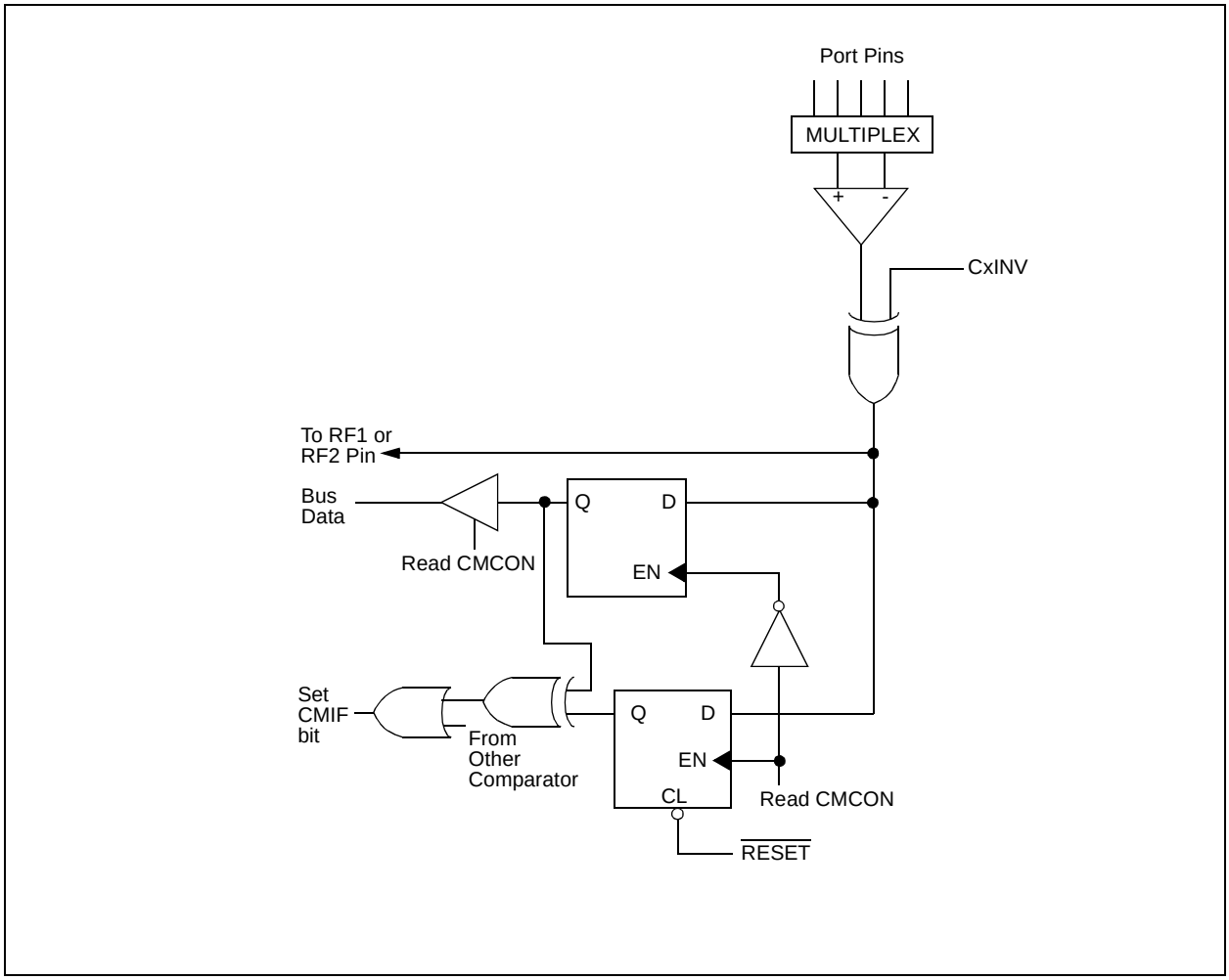
The comparator outputs are read through the CMCON register. These bits are read only. The comparator outputs may also be directly output to the RF1 and RF2 I/O pins. When enabled, multiplexors in the output path of the RF1 and RF2 pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 21-3 shows the comparator output block diagram.

The TRISA bits will still function as an output enable/ disable for the RF1 and RF2 pins while in this mode.

The polarity of the comparator outputs can be changed using the C2INV and C1INV bits (CMCON<4:5>).

Note 1:	When reading the Port register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert an analog input according to the Schmitt Trigger input specification.
2:	Analog levels on any pin defined as a digital input may cause the input buffer to consume more current than is specified.

FIGURE 21-3: COMPARATOR OUTPUT BLOCK DIAGRAM



21.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR registers) is the comparator interrupt flag. The CMIF bit must be reset by clearing '0'. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

The CMIE bit (PIE registers) and the PEIE bit (INTCON register) must be set to enable the interrupt. In addition, the GIE bit must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

Note:	If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR registers) interrupt flag may not get set.

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit CMIF.

A mismatch condition will continue to set flag bit CMIF. Reading CMCON will end the mismatch condition and allow flag bit CMIF to be cleared.

21.7 Comparator Operation During SLEEP

When a comparator is active and the device is placed in SLEEP mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will wake-up the device from SLEEP mode when enabled. While the comparator is powered up, higher SLEEP currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in SLEEP mode, turn off the comparators, CM<2:0> = 111, before entering SLEEP. If the device wakes up from SLEEP, the contents of the CMCON register are not affected.

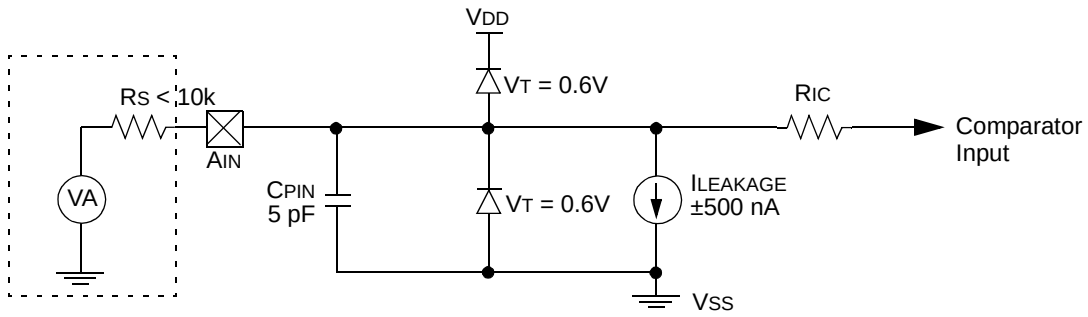
21.8 Effects of a RESET

A device RESET forces the CMCON register to its RESET state, causing the comparator module to be in the comparator RESET mode, CM<2:0> = 000. This ensures that all potential inputs are analog inputs. Device current is minimized when analog inputs are present at RESET time. The comparators will be powered down during the RESET interval.

21.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 21-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latchup condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 21-4: COMPARATOR ANALOG INPUT MODEL



Legend:
CPIN = Input Capacitance
VT = Threshold Voltage
ILEAKAGE = Leakage Current at the pin due to various junctions
RIC = Interconnect Resistance
RS = Source Impedance
VA = Analog Voltage

TABLE 21-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other RESETS
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
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 0000	0000 0000
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	-1-- 1111	-1-- 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	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are unused by the comparator module.

22.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable voltage reference. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The CVRCON register controls the operation of the reference as shown in Register 22-1. The block diagram is given in Figure 22-1.

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF- that are multiplexed with RA3 and RA2. The comparator reference supply voltage is controlled by the CVRSS bit.

22.1 Configuring the Comparator Voltage Reference

The comparator voltage reference can output 16 distinct voltage levels for each range. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = (CVR<3:0>/24) x CVRSRC

If CVRR = 0:
CVREF = (CVDD x 1/4) + (CVR<3:0>/32) x CVRSRC

The settling time of the comparator voltage reference must be considered when changing the CVREF output (Section 27.0).

REGISTER 22-1: CVRCON REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN**: Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RF5/AN10/CVREF pin
0 = CVREF voltage is disconnected from the RF5/AN10/CVREF pin

Note 1: If enabled for output, RF5 must also be configured as an input by setting TRISF<5> to '1'.

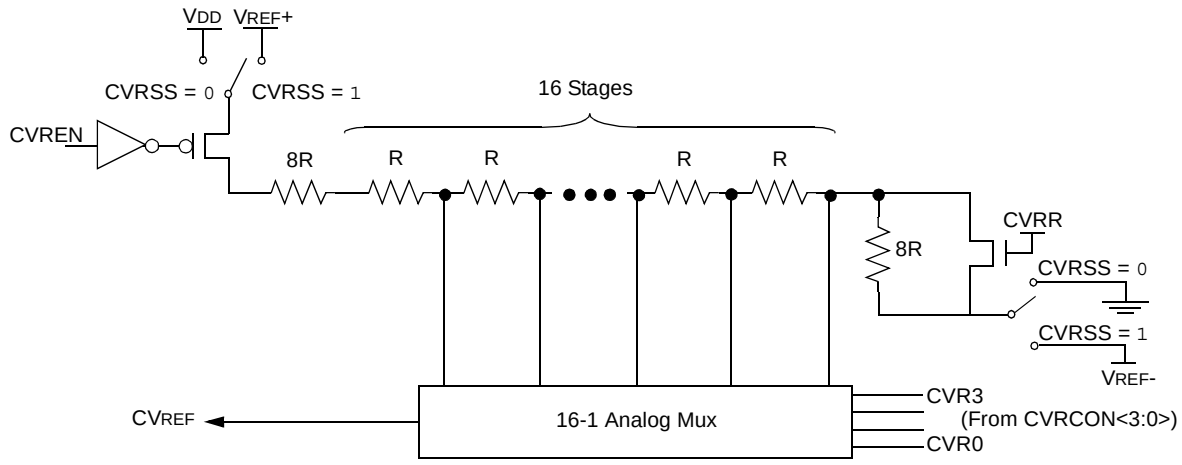
bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0.00 CVRSRC to 0.75 CVRSRC, with CVRSRC/24 step size
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size

bit 4 **CVRSS**: Comparator VREF Source Selection bit
1 = Comparator reference source CVRSRC = VREF+ – VREF-
0 = Comparator reference source CVRSRC = VDD – VSS

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ VR3:VR0 ≤ 15)
When CVRR = 1:
CVREF = (CVR<3:0>/ 24) • (CVRSRC)
When CVRR = 0:
CVREF = 1/4 • (CVRSRC) + (CVR3:CVR0/32) • (CVRSRC)

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 22-1: VOLTAGE REFERENCE BLOCK DIAGRAM



Note: R is defined in Section 27.0.

22.2 Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 22-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in Section 27.0.

22.3 Operation During SLEEP

When the device wakes up from SLEEP through an interrupt or a Watchdog Timer time-out, the contents of the CVRCON register are not affected. To minimize current consumption in SLEEP mode, the voltage reference should be disabled.

22.4 Effects of a RESET

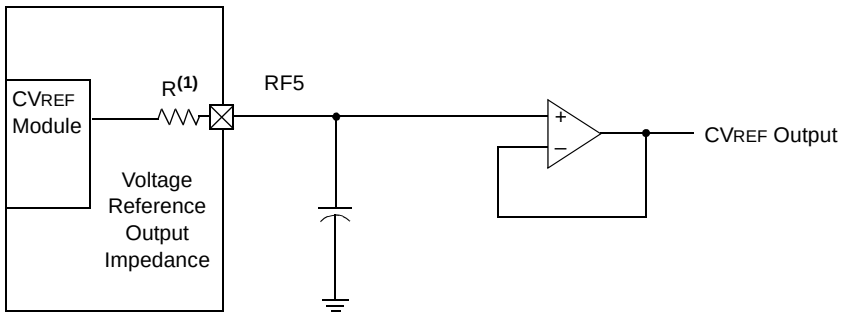
A device RESET disables the voltage reference by clearing bit CVREN (CVRCON<7>). This RESET also disconnects the reference from the RA2 pin by clearing bit CVROE (CVRCON<6>) and selects the high voltage range by clearing bit CVRR (CVRCON<5>). The VRSS value select bits, CVRCON<3:0>, are also cleared.

22.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RF5 pin if the TRISF<5> bit is set and the CVROE bit is set. Enabling the voltage reference output onto the RF5 pin with an input signal present will increase current consumption. Connecting RF5 as a digital output with VRSS enabled will also increase current consumption.

The RF5 pin can be used as a simple D/A output with limited drive capability. Due to the limited current drive capability, a buffer must be used on the voltage reference output for external connections to VREF. Figure 22-2 shows an example buffering technique.

FIGURE 22-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the voltage reference configuration bits CVRCON<3:0> and CVRCON<5>.

TABLE 22-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other RESETS
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	0000 0000
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are not used with the comparator voltage reference.

NOTES:

23.0 LOW VOLTAGE DETECT

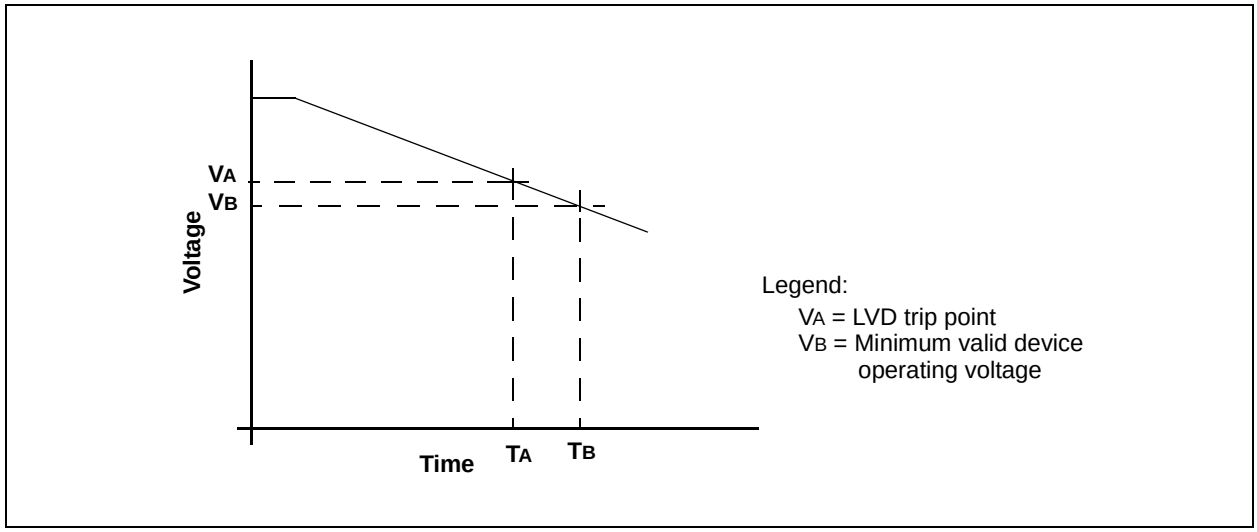
In many applications, the ability to determine if the device voltage (V_{DD}) is below a specified voltage level is a desirable feature. A window of operation for the application can be created, where the application software can do “housekeeping tasks” before the device voltage exits the valid operating range. This can be done using the Low Voltage Detect module.

This module is a software programmable circuitry, where a device voltage trip point can be specified. When the voltage of the device becomes lower then the specified point, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to that interrupt source.

The Low Voltage Detect circuitry is completely under software control. This allows the circuitry to be “turned off” by the software which minimizes the current consumption for the device.

Figure 23-1 shows a possible application voltage curve (typically for batteries). Over time, the device voltage decreases. When the device voltage equals voltage V_A, the LVD logic generates an interrupt. This occurs at time T_A. The application software then has the time, until the device voltage is no longer in valid operating range, to shutdown the system. Voltage point V_B is the minimum valid operating voltage specification. This occurs at time T_B. The difference $T_B - T_A$ is the total time for shut-down.

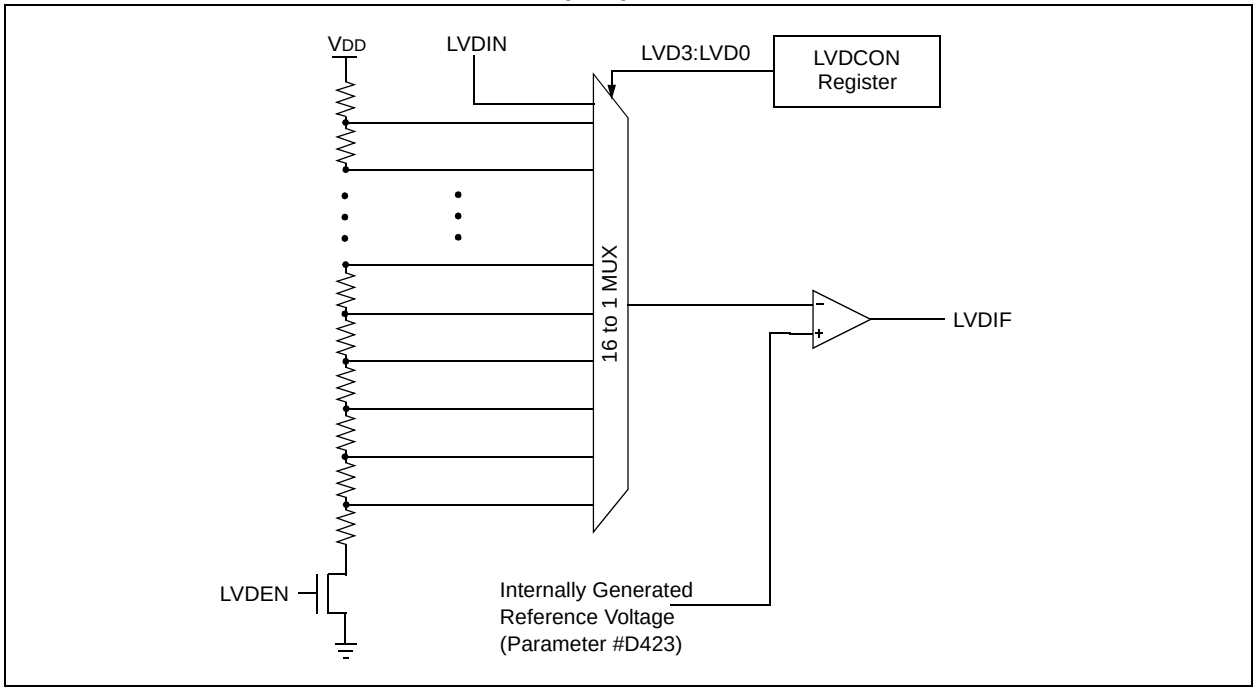
FIGURE 23-1: TYPICAL LOW VOLTAGE DETECT APPLICATION



The block diagram for the LVD module is shown in Figure 23-2. A comparator uses an internally generated reference voltage as the set point. When the selected tap output of the device voltage crosses the set point (is lower than), the LVDIF bit is set.

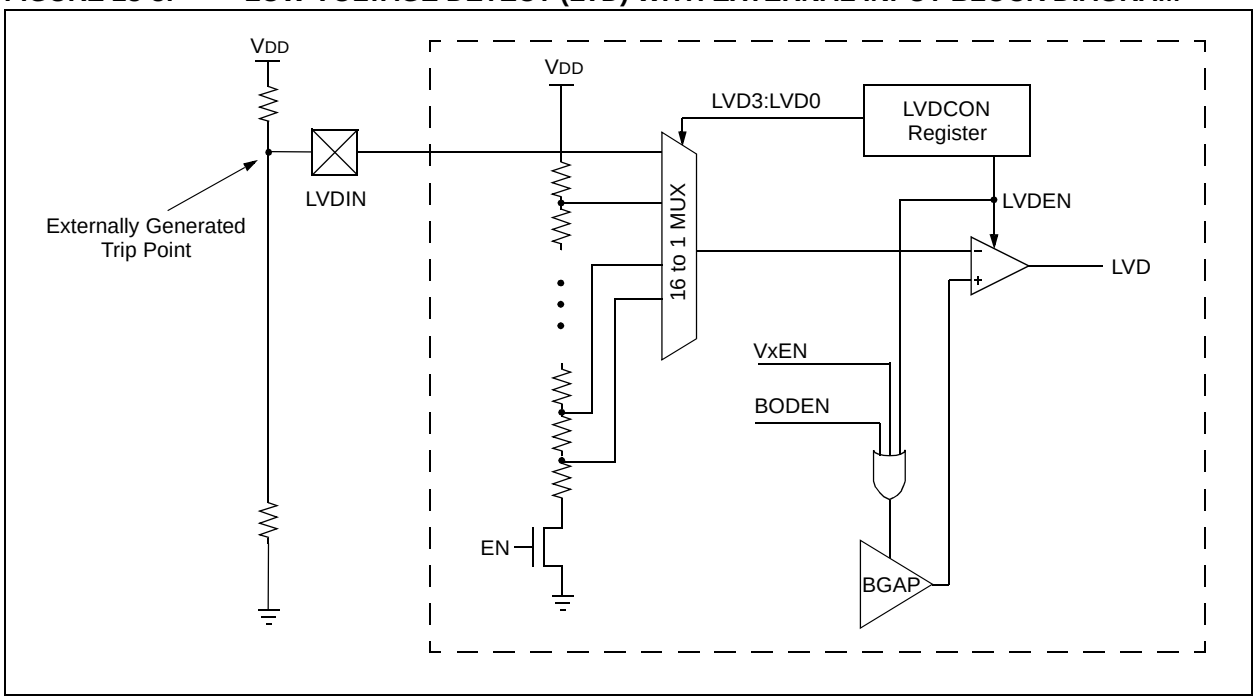
Each node in the resistor divider represents a “trip point” voltage. The “trip point” voltage is the minimum supply voltage level at which the device can operate before the LVD module asserts an interrupt. When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the 1.2V internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal setting the LVDIF bit. This voltage is software programmable to any one of 16 values (see Figure 23-2). The trip point is selected by programming the LVDL3:LVDL0 bits (LVDCON<3:0>).

FIGURE 23-2: LOW VOLTAGE DETECT (LVD) BLOCK DIAGRAM



The LVD module has an additional feature that allows the user to supply the trip voltage to the module from an external source. This mode is enabled when bits LVDL3:LVDL0 are set to '1111'. In this state, the comparator input is multiplexed from the external input pin, LVDIN (Figure 23-3). This gives users flexibility because it allows them to configure the Low Voltage Detect interrupt to occur at any voltage in the valid operating range.

FIGURE 23-3: LOW VOLTAGE DETECT (LVD) WITH EXTERNAL INPUT BLOCK DIAGRAM



23.1 Control Register

The Low Voltage Detect Control register (Register 23-1) controls the operation of the Low Voltage Detect circuitry.

REGISTER 23-1: LVDCON REGISTER

U-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
—	—	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit
1 = Indicates that the Low Voltage Detect logic will generate the interrupt flag at the specified voltage range
0 = Indicates that the Low Voltage Detect logic will not generate the interrupt flag at the specified voltage range and the LVD interrupt should not be enabled

bit 4 **LVDEN:** Low Voltage Detect Power Enable bit
1 = Enables LVD, powers up LVD circuit
0 = Disables LVD, powers down LVD circuit

bit 3-0 **LVDL3:LVDL0:** Low Voltage Detection Limit bits
1111 = External analog input is used (input comes from the LVDIN pin)
1110 = 4.45V - 4.83V
1101 = 4.16V - 4.5V
1100 = 3.96V - 4.3V
1011 = 3.76V - 3.92V
1010 = 3.57V - 3.87V
1001 = 3.47V - 3.75V
1000 = 3.27V - 3.55V
0111 = 2.98V - 3.22V
0110 = 2.77V - 3.01V
0101 = 2.67V - 2.89V
0100 = 2.48V - 2.68V
0011 = 2.37V - 2.57V
0010 = 2.18V - 2.36V
0001 = 1.98V - 2.14V
0000 = Reserved

Note: LVDL3:LVDL0 modes, which result in a trip point below the valid operating voltage of the device, are not tested.

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

23.2 Operation

Depending on the power source for the device voltage, the voltage normally decreases relatively slowly. This means that the LVD module does not need to be constantly operating. To decrease the current requirements, the LVD circuitry only needs to be enabled for short periods where the voltage is checked. After doing the check, the LVD module may be disabled.

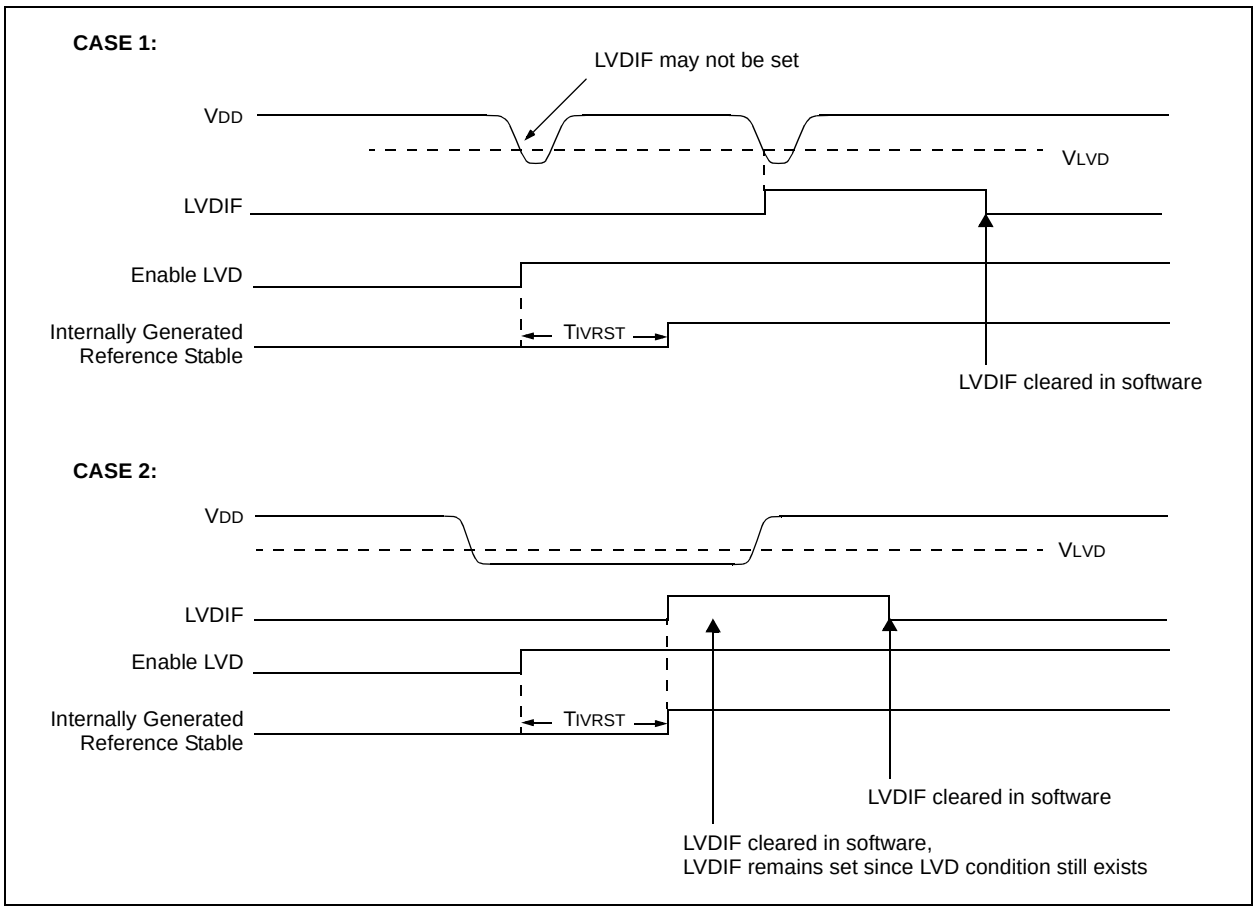
Each time that the LVD module is enabled, the circuitry requires some time to stabilize. After the circuitry has stabilized, all status flags may be cleared. The module will then indicate the proper state of the system.

The following steps are needed to set up the LVD module:

1. Write the value to the LVDL3:LVDL0 bits (LVDCON register) which selects the desired LVD trip point.
2. Ensure that LVD interrupts are disabled (the LVDIE bit is cleared or the GIE bit is cleared).
3. Enable the LVD module (set the LVDEN bit in the LVDCON register).
4. Wait for the LVD module to stabilize (the IRVST bit to become set).
5. Clear the LVD interrupt flag, which may have falsely become set, until the LVD module has stabilized (clear the LVDIF bit).
6. Enable the LVD interrupt (set the LVDIE and the GIE bits).

Figure 23-4 shows typical waveforms that the LVD module may be used to detect.

FIGURE 23-4: LOW VOLTAGE DETECT WAVEFORMS



23.2.1 REFERENCE VOLTAGE SET POINT

The internal reference voltage of the LVD module, specified in electrical specification parameter #D423, may be used by other internal circuitry (the Programmable Brown-out Reset). If these circuits are disabled (lower current consumption), the reference voltage circuit requires a time to become stable before a low voltage condition can be reliably detected. This time is invariant of system clock speed. This start-up time is specified in electrical specification parameter #36. The low voltage interrupt flag will not be enabled until a stable reference voltage is reached. Refer to the waveform in Figure 23-4.

23.2.2 CURRENT CONSUMPTION

When the module is enabled, the LVD comparator and voltage divider are enabled and will consume static current. The voltage divider can be tapped from multiple places in the resistor array. Total current consumption, when enabled, is specified in electrical specification parameter #D022B.

23.3 Operation During SLEEP

When enabled, the LVD circuitry continues to operate during SLEEP. If the device voltage crosses the trip point, the LVDIF bit will be set and the device will wake-up from SLEEP. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

23.4 Effects of a RESET

A device RESET forces all registers to their RESET state. This forces the LVD module to be turned off.

NOTES:

24.0 SPECIAL FEATURES OF THE CPU

There are several features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These are:

- OSC Selection
- RESET
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code Protection
- ID Locations
- In-Circuit Serial Programming

All PIC18F6X2X/8X2X devices have a Watchdog Timer which is permanently enabled via the configuration bits, or software controlled. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in RESET until the crystal oscillator is stable. The other is the Power-up Timer (PWRT) which provides a fixed delay on power-up only, designed to keep the part in RESET while the power supply stabilizes. With these two timers on-chip, most applications need no external RESET circuitry.

SLEEP mode is designed to offer a very low current power-down mode. The user can wake-up from SLEEP through external RESET, Watchdog Timer wake-up, or through an interrupt. Several oscillator options are also made available to allow the part to fit the application. The RC oscillator option saves system cost, while the LP crystal option saves power. A set of configuration bits are used to select various options.

24.1 Configuration Bits

The configuration bits can be programmed (read as ‘0’) or left unprogrammed (read as ‘1’), to select various device configurations. These bits are mapped, starting at program memory location 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h through 3FFFFFh) which can only be accessed using table reads and table writes.

Programming the Configuration registers is done in a manner similar to programming the FLASH memory. The EECON1 register WR bit starts a self-timed write to the Configuration register. In normal operation mode, a `TBLWT` instruction, with the TBLPTR pointed to the Configuration register, sets up the address and the data for the Configuration register write. Setting the WR bit starts a long write to the Configuration register. The Configuration registers are written a byte at a time. To write or erase a configuration cell, a `TBLWT` instruction can write a ‘1’ or a ‘0’ into the cell.

TABLE 24-1: CONFIGURATION BITS AND DEVICE IDS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h	CONFIG1H	—	—	$\overline{\text{OSCSEN}}$	—	FOSC3	FOSC2	FOSC1	FOSC0	--1- 1111
300002h	CONFIG2L	—	—	—	—	BORV1	BORV0	BODEN	$\overline{\text{PWRTEN}}$	---- 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300004h(1)	CONFIG3L	WAIT	—	—	—	—	—	PM1	PM0	1--- --11
300005h	CONFIG3H	MCLRE	—	—	—	—	—	ECCPMX(1)	CCP2MX	1--- --11
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN	1--- -1-1
300008h	CONFIG5L	—	—	—	—	CP3(2)	CP2	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	WRT3(2)	WRT2	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	EBTR3(2)	EBTR2	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	**(Note 3)**
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 1010

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

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: See Register 24-13 for DEVID1 values.

REGISTER 24-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-0	U-0	R/P-1	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	$\overline{\text{OSCSEN}}$	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **$\overline{\text{OSCSEN}}$**: Oscillator System Clock Switch Enable bit

1 = Oscillator system clock switch option is disabled (main oscillator is source)
0 = Timer1 oscillator system clock switch option is enabled (oscillator switching is enabled)

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

bit 3-0 **Fosc3:Fosc0**: Oscillator Selection bits

1111 = RC oscillator with OSC2 configured as RA6
1110 = HS oscillator with SW enabled 4x PLL
1101 = EC oscillator with OSC2 configured as RA6 and SW enabled 4x PLL
1100 = EC oscillator with OSC2 configured as RA6 and HW enabled 4x PLL
1011 = Reserved; do not use
1010 = Reserved; do not use
1001 = Reserved; do not use
1000 = Reserved; do not use
0111 = RC oscillator with OSC2 configured as RA6
0110 = HS oscillator with HW enabled 4x PLL
0101 = EC oscillator with OSC2 configured as RA6
0100 = EC oscillator with OSC2 configured as divide by 4 clock output
0011 = RC oscillator with OSC2 configured as divide by 4 clock output
0010 = HS oscillator
0001 = XT oscillator
0000 = LP oscillator

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-2: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	BORV1	BORV0	BOREN	$\overline{\text{PWRTEN}}$
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3-2 **BORV1:BORV0:** Brown-out Reset Voltage bits

11 = V_{BOR} set to 2.0V
10 = V_{BOR} set to 2.7V
01 = V_{BOR} set to 4.2V
00 = V_{BOR} set to 4.5V

bit 1 **BOREN:** Brown-out Reset Enable bit

1 = Brown-out Reset enabled
0 = Brown-out Reset disabled

bit 0 **$\overline{\text{PWRTEN}}$:** Power-up Timer Enable bit

1 = PWRT disabled
0 = PWRT enabled

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-3: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

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

bit 4-1 **WDTPS2:WDTPS0:** Watchdog Timer Postscaler Select bits

1111 = 1:32768
1110 = 1:16384
1101 = 1:8192
1100 = 1:4096
1011 = 1:2048
1010 = 1:1024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

bit 0 **WDTEN:** Watchdog Timer Enable bit

1 = WDT enabled
0 = WDT disabled (control is placed on the SWDTEN bit)

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-4: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)(1)

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 for table reads and table writes
0 = Wait selections for table reads and table writes are determined by WAIT1:WAIT0 bits (MEMCOM<5:4>)

bit 6-2 **Unimplemented:** Read as '0'

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

Note 1: This register is unimplemented for PIC18F6X2X devices; maintain these bits set.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-5: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

R/P-1	U-0	U-0	U-0	U-0	U-0	R/P-1	R/P-1
MCLRE(1)	—	—	—	—	—	ECCPMX(2)	CCP2MX
bit 7							bit 0

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Enable bit(1)
1 = $\overline{\text{MCLR}}$ pin enabled, RG5 input pin disabled
0 = RG5 input enabled, $\overline{\text{MCLR}}$ disabled

bit 6-2 **Unimplemented:** Read as '0'

bit 1 **ECCPMX:** ECCP Mux bit(2)
1 = Enhanced CCP1 (P1B/P1C) and Enhanced CCP3 (P3B/P3C) PWM outputs are multiplexed with RE6 through RE3
0 = Enhanced CCP1 (P1B/P1C) and Enhanced CCP3 (P3B/P3C) PWM outputs are multiplexed with RH7 through RH4

bit 0 **CCP2MX:** CCP2 Mux bit
In Microcontroller mode:
1 = CCP2 input/output is multiplexed with RC1
0 = CCP2 input/output is multiplexed with RE7
In Microprocessor, Microprocessor with Boot Block and Extended Microcontroller modes (PIC18F8X8X devices only):
1 = CCP2 input/output is multiplexed with RC1
0 = CCP2 input/output is multiplexed with RB3

Note 1: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.
2: This register is unimplemented for PIC18F6X2X devices; maintain these bits set.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-6: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/P-1	U-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN
bit 7							bit 0

bit 7 **$\overline{\text{DEBUG}}$:** Background Debugger Enable bit
1 = Background debugger disabled. RB6 and RB7 configured as general purpose I/O pins.
0 = Background debugger enabled. RB6 and RB7 are dedicated to in-circuit debug.

bit 6-3 **Unimplemented:** Read as '0'

bit 2 **LVP:** Low Voltage ICSP Enable bit
1 = Low voltage ICSP enabled
0 = Low voltage ICSP disabled

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

bit 0 **STVREN:** Stack Full/Underflow Reset Enable bit
1 = Stack full/underflow will cause RESET
0 = Stack full/underflow will not cause RESET

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-7: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3(1)	CP2	CP1	CP0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **CP3:** Code Protection bit(1)

1 = Block 3 (00C000-00FFFFh) not code protected
0 = Block 3 (00C000-00FFFFh) code protected

Note 1: Unimplemented in PIC18FX525 devices; maintain this bit set.

bit 2 **CP2:** Code Protection bit

1 = Block 2 (008000-00BFFFh) not code protected
0 = Block 2 (008000-00BFFFh) code protected

bit 1 **CP1:** Code Protection bit

1 = Block 1 (004000-007FFFh) not code protected
0 = Block 1 (004000-007FFFh) code protected

bit 0 **CP0:** Code Protection bit

1 = Block 0 (000800-003FFFh) not code protected
0 = Block 0 (000800-003FFFh) code protected

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-8: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
bit 7							bit 0

bit 7 **CPD:** Data EEPROM Code Protection bit

1 = Data EEPROM not code protected
0 = Data EEPROM code protected

bit 6 **CPB:** Boot Block Code Protection bit

1 = Boot block (000000-0007FFh) not code protected
0 = Boot block (000000-0007FFh) code protected

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

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-9: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	WRT3(1)	WRT2	WRT1	WRT0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **WRT3:** Write Protection bit(1)
1 = Block 3 (00C000-00FFFFh) not write protected
0 = Block 3 (00C000-00FFFFh) write protected

Note 1: Unimplemented in PIC18FX525 devices; maintain this bit set.

bit 2 **WRT2:** Write Protection bit
1 = Block 2 (008000-00BFFFh) not write protected
0 = Block 2 (008000-00BFFFh) write protected

bit 1 **WRT1:** Write Protection bit
1 = Block 1 (004000-007FFFh) not write protected
0 = Block 1 (004000-007FFFh) write protected

bit 0 **WR0:** Write Protection bit
1 = Block 0 (000800-003FFFh) not write protected
0 = Block 0 (000800-003FFFh) write protected

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-10: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)

R/C-1	R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC	—	—	—	—	—
bit 7							bit 0

bit 7 **WRTD:** Data EEPROM Write Protection bit
1 = Data EEPROM not write protected
0 = Data EEPROM write protected

bit 6 **WRTB:** Boot Block Write Protection bit
1 = Boot block (000000-0007FFh) not write protected
0 = Boot block (000000-0007FFh) write protected

bit 5 **WRTC:** Configuration Register Write Protection bit
1 = Configuration registers (300000-3000FFh) not write protected
0 = Configuration registers (300000-3000FFh) write protected

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

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-11: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3(1)	EBTR2	EBTR1	EBTR0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **EBTR3:** Table Read Protection bit(1)

1 = Block 3 (00C000-00FFFFh) not protected from table reads executed in other blocks
0 = Block 3 (00C000-00FFFFh) protected from table reads executed in other blocks

Note 1: Unimplemented in PIC18FX525 devices; maintain this bit set.

bit 2 **EBTR2:** Table Read Protection bit

1 = Block 2 (008000-00BFFFh) not protected from table reads executed in other blocks
0 = Block 2 (008000-00BFFFh) protected from table reads executed in other blocks

bit 1 **EBTR1:** Table Read Protection bit

1 = Block 1 (004000-007FFFh) not protected from table reads executed in other blocks
0 = Block 1 (004000-007FFFh) protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit

1 = Block 0 (000800-003FFFh) not protected from table reads executed in other blocks
0 = Block 0 (000800-003FFFh) protected from table reads executed in other blocks

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-12: CONFIG7H: CONFIGURATION REGISTER 7 HIGH (BYTE ADDRESS 30000Dh)

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
bit 7							bit 0

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

bit 6 **EBTRB:** Boot Block Table Read Protection bit

1 = Boot block (000000-0007FFh) not protected from table reads executed in other blocks
0 = Boot block (000000-0007FFh) protected from table reads executed in other blocks

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

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-13: DEVICE ID REGISTER 1 FOR PIC18F6X2X/8X2X DEVICES (ADDRESS 3FFFFEh)

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

bit 7-5 **DEV2:DEV0:** Device ID bits

100 = PIC18F8621
101 = PIC18F6621
110 = PIC18F8525
111 = PIC18F6525

bit 4-0 **REV4:REV0:** Revision ID bits

These bits are used to indicate the device revision.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 24-14: DEVICE ID REGISTER 2 FOR PIC18F6X2X/8X2X DEVICES (ADDRESS 3FFFFFh)

R-0	R-0	R-0	R-0	R-1	R-0	R-1	R-0
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3:** Device ID bits

These bits are used with the DEV2:DEV0 bits in the Device ID Register 1 to identify the part number.

0000 1010 = PIC18F6X2X/8X2X

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

24.2 Watchdog Timer (WDT)

The Watchdog Timer is a free running on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKI pin. That means that the WDT will run even if the clock on the OSC1/CLKI and OSC2/CLKO/RA6 pins of the device has been stopped, for example, by execution of a SLEEP instruction.

During normal operation, a WDT time-out generates a device RESET (Watchdog Timer Reset). If the device is in SLEEP mode, a WDT time-out causes the device to wake-up and continue with normal operation (Watchdog Timer wake-up). The $\overline{TO}$ bit in the RCON register will be cleared upon a WDT time-out.

The Watchdog Timer is enabled or disabled by a device configuration bit, WDTEN (CONFIG2H<0>). If WDTEN is set, software execution may not disable this function. When WDTEN is cleared, the SWDTEN bit enables or disables the operation of the WDT.

The WDT time-out period values may be found in the Electrical Specifications section under parameter #31. Values for the WDT postscaler may be assigned using the configuration bits.

Note 1:	The CLRWDT and SLEEP instructions clear the WDT and the postscaler if assigned to the WDT and prevent it from timing out and generating a device RESET condition.
2:	When a CLRWDT instruction is executed and the postscaler is assigned to the WDT, the postscaler count will be cleared but the postscaler assignment is not changed.

24.2.1 CONTROL REGISTER

Register 24-15 shows the WDTCON register. This is a readable and writable register which contains a control bit that allows software to override the WDT enable configuration bit only when the configuration bit has disabled the WDT.

REGISTER 24-15: WDTCON REGISTER

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

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

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit

1 = Watchdog Timer is on

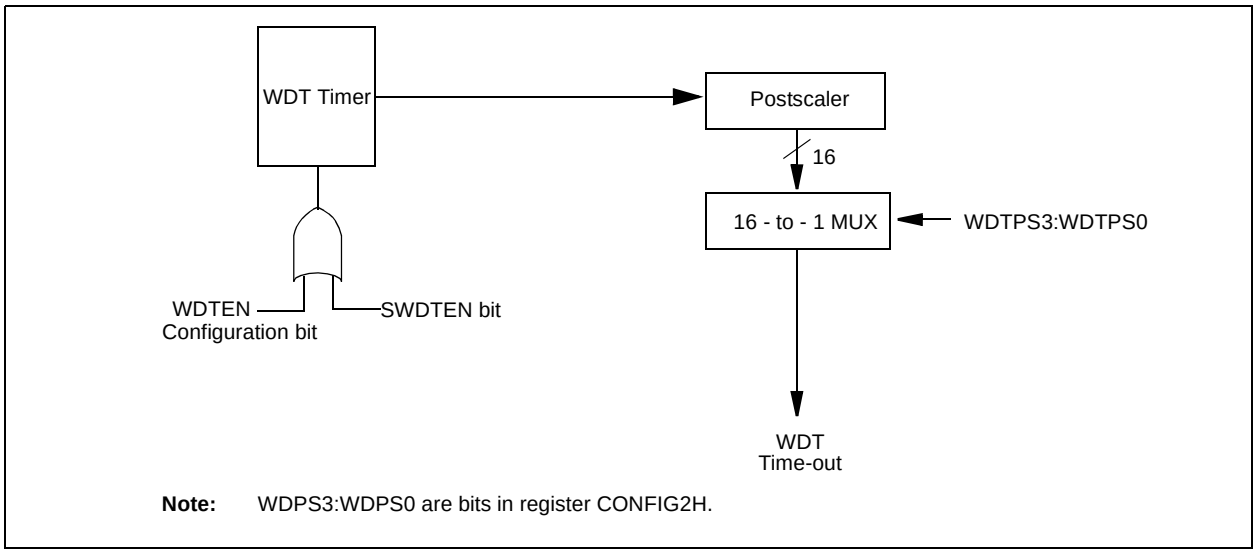
0 = Watchdog Timer is turned off (if CONFIG2H<0> = 0)

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

24.2.2 WDT POSTSCALER

The WDT has a postscaler that can extend the WDT Reset period. The postscaler is selected at the time of the device programming by the value written to the CONFIG2H Configuration register.

FIGURE 24-1: WATCHDOG TIMER BLOCK DIAGRAM



Note: WDPS3:WDPS0 are bits in register CONFIG2H.

TABLE 24-2: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS2	WDTPS0	WDTEN
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
WDTCON	—	—	—	—	—	—	—	SWDTEN

Legend: Shaded cells are not used by the Watchdog Timer.

24.3 Power-down Mode (SLEEP)

Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{PD}$ bit (RCON<3>) is cleared, the $\overline{TO}$ (RCON<4>) bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, place all I/O pins at either V_{DD} or V_{SS}, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D and disable external clocks. Pull all I/O pins that are hi-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on-chip pull-ups on PORTB should be considered.

The $\overline{MCLR}$ pin must be at a logic high level (V_{IHMC}).

24.3.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

1. External RESET input on $\overline{MCLR}$ pin.
2. Watchdog Timer wake-up (if WDT was enabled).
3. Interrupt from INT pin, RB port change or a peripheral interrupt.

The following peripheral interrupts can wake the device from SLEEP:

1. PSP read or write.
2. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
3. TMR3 interrupt. Timer3 must be operating as an asynchronous counter.
4. CCP Capture mode interrupt.
5. Special event trigger (Timer1 in Asynchronous mode using an external clock).
6. MSSP (START/STOP) bit detect interrupt.
7. MSSP transmit or receive in Slave mode (SPI/I^2C).
8. USART RX or TX (Synchronous Slave mode).
9. A/D conversion (when A/D clock source is RC).
10. EEPROM write operation complete.
11. LVD interrupt.

Other peripherals cannot generate interrupts since during SLEEP, no on-chip clocks are present.

External $\overline{MCLR}$ Reset will cause a device RESET. All other events are considered a continuation of program execution and will cause a “wake-up”. The $\overline{TO}$ and $\overline{PD}$ bits in the RCON register can be used to determine the cause of the device RESET. The $\overline{PD}$ bit, which is set on power-up, is cleared when SLEEP is invoked. The $\overline{TO}$ bit is cleared if a WDT time-out occurred (and caused wake-up).

When the SLEEP instruction is being executed, the next instruction (PC + 2) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address. In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

24.3.2 WAKE-UP USING INTERRUPTS

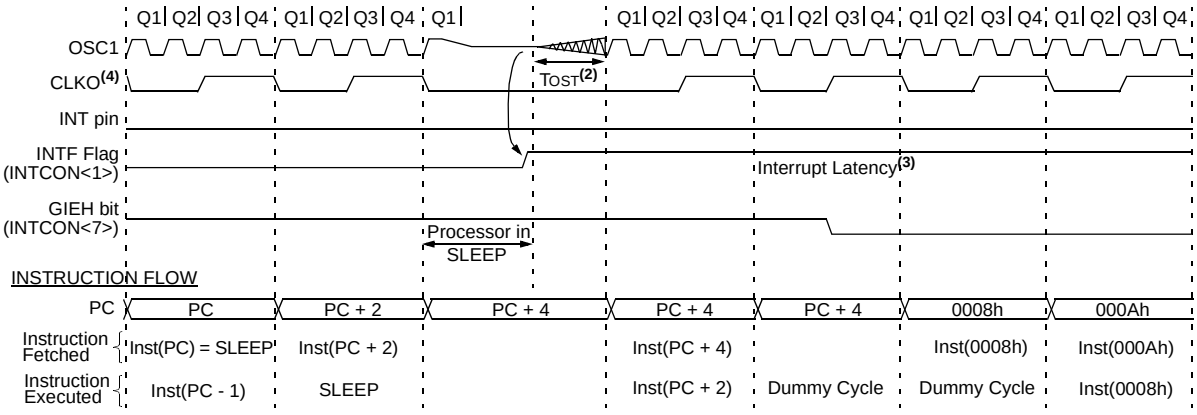
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If an interrupt condition (interrupt flag bit and interrupt enable bits are set) occurs **before** the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT postscaler will not be cleared, the $\overline{TO}$ bit will not be set and $\overline{PD}$ bits will not be cleared.
- If the interrupt condition occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake-up from SLEEP. The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

FIGURE 24-2: WAKE-UP FROM SLEEP THROUGH INTERRUPT(1,2)



Note 1: XT, HS or LP Oscillator mode assumed.
2: GIE = 1 assumed. In this case, after wake-up, the processor jumps to the interrupt routine. If GIE = 0, execution will continue in-line.
3: TOST = 1024 TOSC (drawing not to scale). This delay will not occur for RC and EC Osc modes.
4: CLKO is not available in these Osc modes but shown here for timing reference.

24.4 Program Verification and Code Protection

The overall structure of the code protection on the PIC18 FLASH devices differs significantly from other PICmicro® devices.

The user program memory is divided on binary boundaries into four blocks of 16 Kbytes each. The first block is further divided into a boot block of 2048 bytes and a second block (Block 0) of 14 Kbytes.

Each of the blocks has three code protection bits associated with them. They are:

- Code Protect bit (CPn)
- Write Protect bit (WRTn)
- External Block Table Read bit (EBTRn)

Figure 24-3 shows the program memory organization for 48 and 64-Kbyte devices and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 24-3.

FIGURE 24-3: CODE PROTECTED PROGRAM MEMORY FOR PIC18F6X2X/8X2X DEVICES

MEMORY SIZE/DEVICE			
48 Kbytes (PIC18FX525	**64 Kbytes (PIC18FX621)**	**Address Range**	**Block Code Protection Controlled By:**
Boot Block	Boot Block	000000h 0007FFh	CPB, WRTB, EBTRB
Block 0	Block 0	000800h 003FFFh	CP0, WRT0, EBTR0
Block 1	Block 1	004000h 007FFFh	CP1, WRT1, EBTR1
Block 2	Block 2	008000h 00BFFFh	CP2, WRT2, EBTR2
Unimplemented Read '0'	Block 3	00C000h 00FFFFh	CP3, WRT3, EBTR3

TABLE 24-3: SUMMARY OF CODE PROTECTION REGISTERS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3*	CP2	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3*	WRT2	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3*	EBTR2	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

Legend: Shaded cells are unimplemented.

* Unimplemented in PIC18FX585 devices.

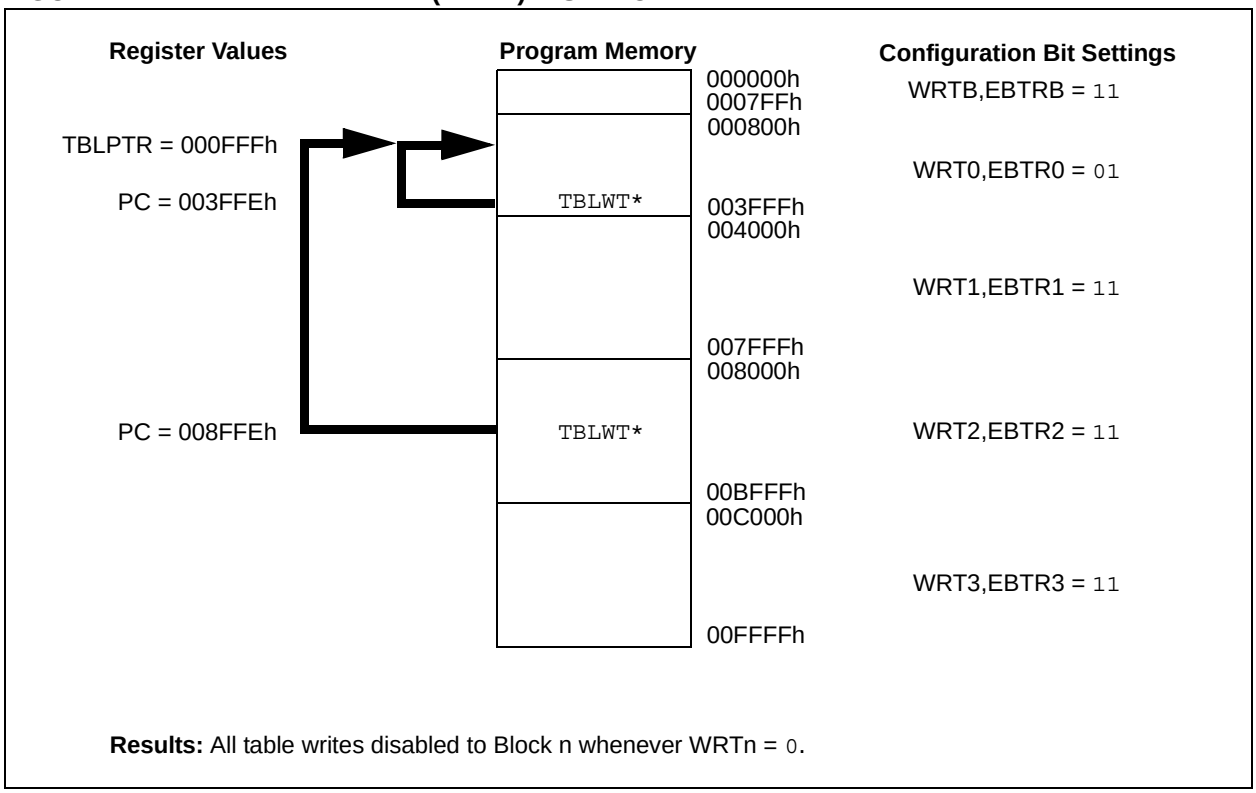
24.4.1 PROGRAM MEMORY CODE PROTECTION

The user memory may be read to or written from any location using the table read and table write instructions. The device ID may be read with table reads. The Configuration registers may be read and written with the table read and table write instructions.

In user mode, the CPn bits have no direct effect. CPn bits inhibit external reads and writes. A block of user memory may be protected from table writes if the WRTn configuration bit is '0'. The EBTRn bits control table reads. For a block of user memory with the EBTRn bit set to '0', a table read instruction that executes from within that block is allowed to read. A table read instruction that executes from a location outside of that block is not allowed to read and will result in reading '0's. Figures 24-4 through 24-6 illustrate table write and table read protection.

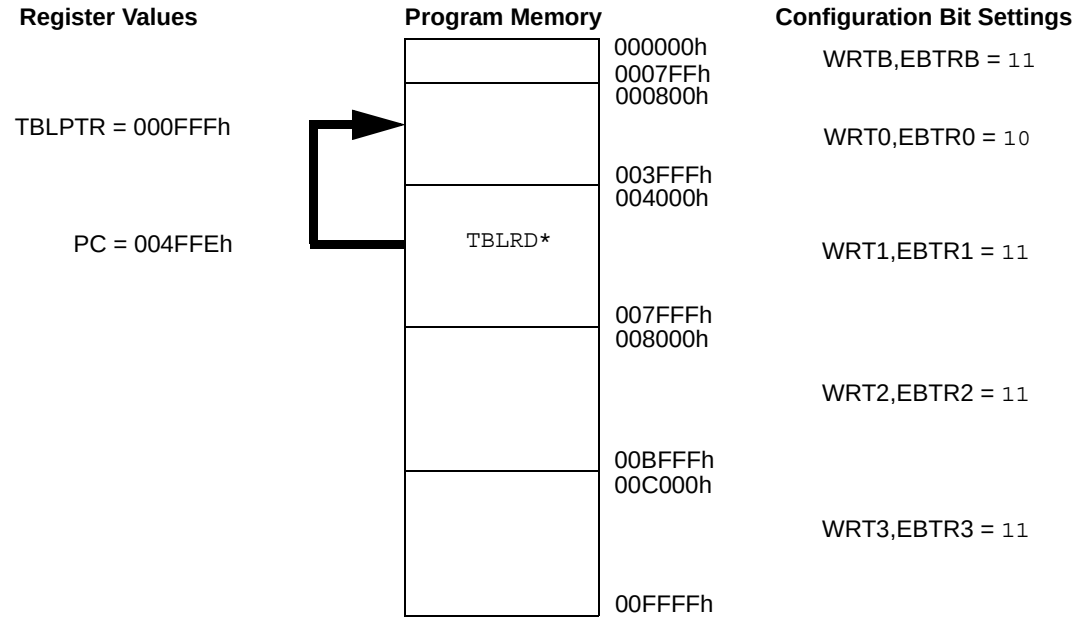
Note:	Code protection bits may only be written to a '0' from a '1' state. It is not possible to write a '1' to a bit in the '0' state. Code protection bits are only set to '1' by a full chip erase or block erase function. The full chip erase and block erase functions can only be initiated via ICSP or an external programmer.

FIGURE 24-4: TABLE WRITE (WRTn) DISALLOWED



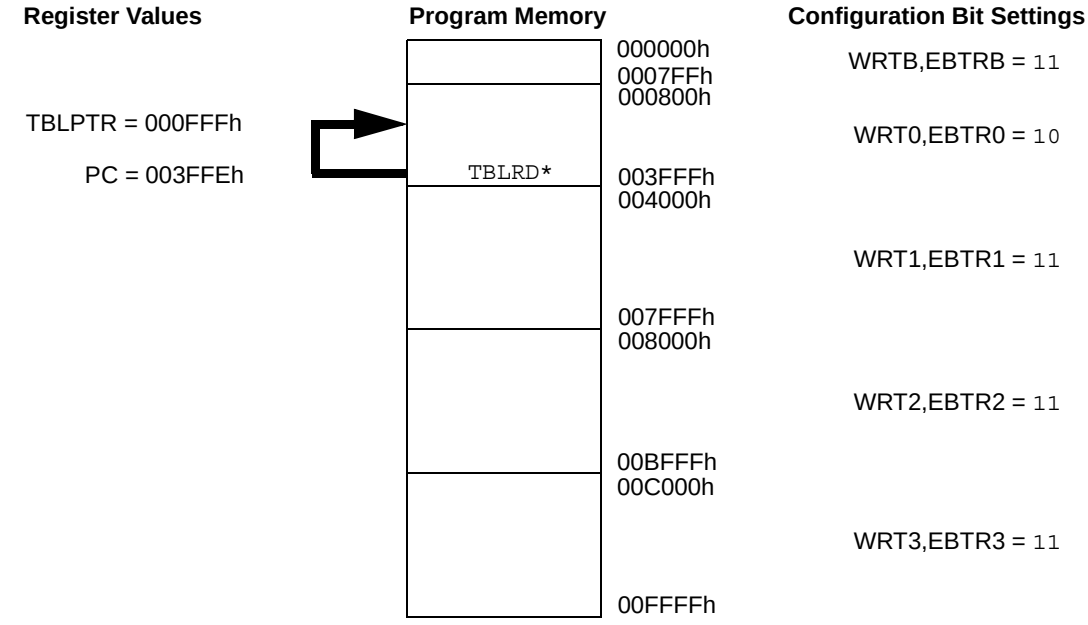
Results: All table writes disabled to Block n whenever WRTn = 0.

FIGURE 24-5: EXTERNAL BLOCK TABLE READ (EBTRn) DISALLOWED



Results: All table reads from external blocks to Block n are disabled whenever EBTRn = 0.
TABLAT register returns a value of '0'.

FIGURE 24-6: EXTERNAL BLOCK TABLE READ (EBTRn) ALLOWED



Results: Table reads permitted within Block n, even when EBTRBn = 0.
TABLAT register returns the value of the data at the location TBLPTR.

24.4.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

24.4.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write protected. The WRTC bit controls protection of the Configuration registers. In user mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

24.5 ID Locations

Eight memory locations (200000h - 200007h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are accessible during normal execution through the TBLRD and TBLWT instructions, or during program/verify. The ID locations can be read when the device is code protected.

24.6 In-Circuit Serial Programming

PIC18F6X2X/8X2X microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

24.7 In-Circuit Debugger

When the DEBUG bit in Configuration register, CONFIG4L, is programmed to a '0', the in-circuit debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some of the resources are not available for general use. Table 24-4 shows which features are consumed by the background debugger.

TABLE 24-4: DEBUGGER RESOURCES

I/O pins	RB6, RB7
Stack	2 levels
Program Memory	512 bytes
Data Memory	10 bytes

To use the in-circuit debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{MCLR}/V_{PP}$, V_{DD}, GND, RB7 and RB6. This will interface to the in-circuit debugger module available from Microchip or one of the third party development tool companies.

24.8 Low Voltage ICSP Programming

The LVP bit Configuration register, CONFIG4L, enables low voltage ICSP programming. This mode allows the microcontroller to be programmed via ICSP using a V_{DD} source in the operating voltage range. This only means that V_{PP} does not have to be brought to V_{IHH}, but can instead be left at the normal operating voltage. In this mode, the RB5/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. During programming, V_{DD} is applied to the $\overline{MCLR}/V_{PP}$ pin. To enter Programming mode, V_{DD} must be applied to the RB5/PGM pin provided the LVP bit is set. The LVP bit defaults to a ('1') from the factory.

Note 1: The High Voltage Programming mode is always available, regardless of the state of the LVP bit, by applying V_{IHH} to the $\overline{MCLR}$ pin.

2: While in Low Voltage ICSP mode, the RB5 pin can no longer be used as a general purpose I/O pin, and should be held low during normal operation.

3: When using low voltage ICSP programming (LVP) and the pull-ups on PORTB are enabled, bit 5 in the TRISB register must be cleared to disable the pull-up on RB5 and ensure the proper operation of the device.

4: If the device Master Clear is disabled, verify that either of the following is done to ensure proper entry into ICSP mode:
a.) disable low voltage programming (CONFIG4L<2> = 0); or
b.) make certain that RB5/PGM is held low during entry into ICSP.

If Low Voltage Programming mode is not used, the LVP bit can be programmed to a '0' and RB5/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed when programming is entered with V_{IHH} on $\overline{MCLR}/V_{PP}$.

It should be noted that once the LVP bit is programmed to '0', only the High Voltage Programming mode is available and only High Voltage Programming mode can be used to program the device.

When using low voltage ICSP, the part must be supplied 4.5V to 5.5V if a bulk erase will be executed. This includes reprogramming of the code protect bits from an on-state to off-state. For all other cases of low voltage ICSP, the part may be programmed at the normal operating voltage. This means unique user IDs or user code can be reprogrammed or added.

25.0 INSTRUCTION SET SUMMARY

The PIC18 instruction set adds many enhancements to the previous PICmicro instruction sets, while maintaining an easy migration from these PICmicro instruction sets.

Most instructions are a single program memory word (16 bits), but there are three instructions that require two program memory locations.

Each single word instruction is a 16-bit word divided into an OPCODE, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 25-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 25-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator ‘f’ specifies which file register is to be used by the instruction.

The destination designator ‘d’ specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator ‘b’ selects the number of the bit affected by the operation, while the file register designator ‘f’ represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the call or return instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for three double-word instructions. These three instructions were made double-word instructions so that all the required information is available in these 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single word instructions are executed in a single instruction cycle unless a conditional test is true, or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 µs. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 µs. Two-word branch instructions (if true) would take 3 µs.

Figure 25-1 shows the general formats that the instructions can have.

All examples use the format ‘nnh’ to represent a hexadecimal number, where ‘h’ signifies a hexadecimal digit.

The Instruction Set Summary, shown in Table 25-2, lists the instructions recognized by the Microchip Assembler (MPASM™).

Section 25.1 provides a description of each instruction.

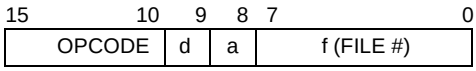
TABLE 25-1: OPCODE FIELD DESCRIPTIONS

Field	Description
a	RAM access bit a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7)
BSR	Bank Select Register. Used to select the current RAM bank.
d	Destination select bit d = 0: store result in WREG d = 1: store result in file register f.
dest	Destination either the WREG register or the specified register file location
f	8-bit register file address (0x00 to 0xFF)
fs	12-bit register file address (0x000 to 0xFFF). This is the source address.
fd	12-bit register file address (0x000 to 0xFFF). This is the destination address.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value)
label	Label name
mm * *+ *- +*	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: No change to register (such as TBLPTR with table reads and writes) Post-Increment register (such as TBLPTR with table reads and writes) Post-Decrement register (such as TBLPTR with table reads and writes) Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions, or the direct address for call/branch and return instructions
PRODH	Product of Multiply High Byte
PRODL	Product of Multiply Low Byte
s	Fast Call/Return mode select bit s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
u	Unused or Unchanged
WREG	Working register (accumulator)
x	Don't care (0 or 1) The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
TBLPTR	21-bit Table Pointer (points to a Program Memory location)
TABLAT	8-bit Table Latch
TOS	Top-of-Stack
PC	Program Counter
PCL	Program Counter Low Byte
PCH	Program Counter High Byte
PCLATH	Program Counter High Byte Latch
PCLATU	Program Counter Upper Byte Latch
GIE	Global Interrupt Enable bit
WDT	Watchdog Timer
$\overline{TO}$	Time-out bit
$\overline{PD}$	Power-down bit
C, DC, Z, OV, N	ALU STATUS bits Carry, Digit Carry, Zero, Overflow, Negative
[ ]	Optional
( )	Contents
→	Assigned to
< >	Register bit field
∈	In the set of
italics	User defined term (font is courier)

FIGURE 25-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations

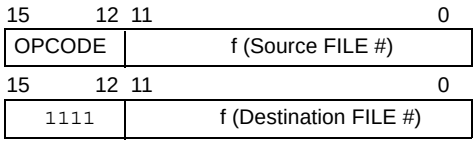
Example Instruction



ADDWF MYREG, W, B

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

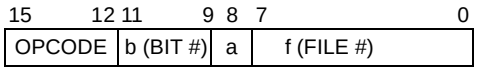
Byte to Byte move operations (2-word)



MOVFF MYREG1, MYREG2

f = 12-bit file register address

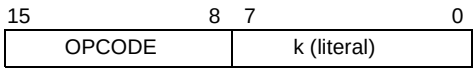
Bit-oriented file register operations



BSF MYREG, bit, B

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Literal operations

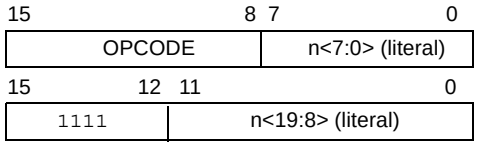


MOVLW 0x7F

k = 8-bit immediate value

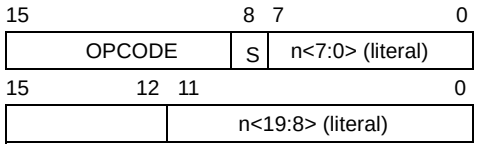
Control operations

CALL, GOTO and Branch operations



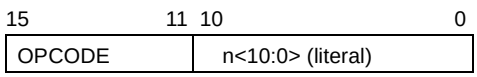
GOTO Label

n = 20-bit immediate value

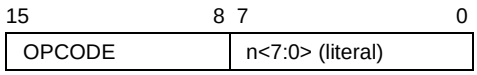


CALL MYFUNC

S = Fast bit



BRA MYFUNC



BC MYFUNC

TABLE 25-2: PIC18FXXX INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d, a	Add WREG and f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	4
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	4
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	None	
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	1, 2
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	1, 2
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	
SUBFWB	f, d, a	Subtract f from WREG with borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Subtract WREG from f with borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	1, 2
SWAPF	f, d, a	Swap nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, d, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2

Note 1: When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

TABLE 25-2: PIC18FXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
CONTROL OPERATIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	2	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	1 (2)	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call subroutine 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C	
GOTO	n	Go to address 1st word	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx	None	4
POP	—	Pop top of return stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push top of return stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software device RESET	1	0000	0000	1111	1111	All	
RETFIE	s	Return from interrupt enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

Note 1: When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

TABLE 25-2: PIC18FXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
LITERAL OPERATIONS									
ADDLW	k	Add literal and WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	AND literal with WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	Inclusive OR literal with WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	Move literal (12-bit) 2nd word	2	1110	1110	00ff	kkkk	None	
		to FSRx 1st word		1111	0000	kkkk	kkkk		
MOVLB	k	Move literal to BSR<3:0>	1	0000	0001	0000	kkkk	None	
MOVLW	k	Move literal to WREG	1	0000	1110	kkkk	kkkk	None	
MULLW	k	Multiply literal with WREG	1	0000	1101	kkkk	kkkk	None	
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	
SUBLW	k	Subtract WREG from literal	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR literal with WREG	1	0000	1010	kkkk	kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS									
TBLRD*		Table Read	2	0000	0000	0000	1000	None	
TBLRD*+		Table Read with post-increment		0000	0000	0000	1001	None	
TBLRD*-		Table Read with post-decrement		0000	0000	0000	1010	None	
TBLRD+*		Table Read with pre-increment		0000	0000	0000	1011	None	
TBLWT*		Table Write	2 (5)	0000	0000	0000	1100	None	
TBLWT*+		Table Write with post-increment		0000	0000	0000	1101	None	
TBLWT*-		Table Write with post-decrement		0000	0000	0000	1110	None	
TBLWT+*		Table Write with pre-increment		0000	0000	0000	1111	None	

Note 1: When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

25.1 Instruction Set

ADDLW **ADD literal to W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal ‘k’ and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: `ADDLW 0x15`

Before Instruction
W = 0x10

After Instruction
W = 0x25

ADDWF **ADD W to f**

Syntax: [*label*] ADDWF f [,d [,a] f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(W) + (f) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	01da	ffff	ffff

Description: Add W to register ‘f’. If ‘d’ is 0, the result is stored in W. If ‘d’ is 1, the result is stored back in register ‘f’ (default). If ‘a’ is 0, the Access Bank will be selected. If ‘a’ is 1, the BSR is used.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `ADDWF REG, 0, 0`

Before Instruction
W = 0x17
REG = 0xC2

After Instruction
W = 0xD9
REG = 0xC2

ADDWFC **ADD W and Carry bit to f**

Syntax: [*label*] ADDWFC f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) + (f) + (C) → dest

Status Affected: N,OV, C, DC, Z

Encoding:

0010	00da	ffff	ffff

Description: Add W, the Carry Flag and data memory location 'f'. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed in data memory location 'f'. If 'a' is 0, the Access Bank will be selected. If 'a' is 1, the BSR will not be overridden.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ADDWFC REG, 0, 1`

Before Instruction
Carry bit = 1
REG = 0x02
W = 0x4D

After Instruction
Carry bit = 0
REG = 0x02
W = 0x50

ANDLW **AND literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N,Z

Encoding:

0000	1011	kkkk	kkkk

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `ANDLW 0x5F`

Before Instruction
W = 0xA3

After Instruction
W = 0x03

ANDWF **AND W with f**

Syntax: [*label*] ANDWF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .AND. (f) → dest

Status Affected: N,Z

Encoding:

0001	01da	ffff	ffff

Description: The contents of W are AND'ed with register 'f'. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'd' (default). If 'a' is 0, the Access Bank will be selected. If 'a' is 1, the BSR will not be overridden (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ANDWF REG, 0, 0`

Before Instruction
W = 0x17
REG = 0xC2
After Instruction
W = 0x02
REG = 0xC2

BC **Branch if Carry**

Syntax: [*label*] BC n

Operands: $-128 \le n \le 127$

Operation: if carry bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0010	nnnn	nnnn

Description: If the Carry bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BC 5`

Before Instruction
PC = address `(HERE)`
After Instruction
If Carry = 1;
PC = address `(HERE+12)`
If Carry = 0;
PC = address `(HERE+2)`

BCF **Bit Clear f**

Syntax: [*label*] BCF f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

1001	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is cleared. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `BCF FLAG_REG, 7, 0`

Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

BN **Branch if Negative**

Syntax: [*label*] BN n

Operands: $-128 \le n \le 127$

Operation: if negative bit is '1'
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0110	nnnn	nnnn

Description: If the Negative bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE    BN   Jump
```

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 1;
PC = address (Jump)
If Negative = 0;
PC = address (HERE+2)

BNC **Branch if Not Carry**

Syntax: [*label*] BNC n

Operands: -128 ≤ n ≤ 127

Operation: if carry bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the Carry bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNC Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Carry = 0;
PC = address `(Jump)`
If Carry = 1;
PC = address `(HERE+2)`

BNN **Branch if Not Negative**

Syntax: [*label*] BNN n

Operands: -128 ≤ n ≤ 127

Operation: if negative bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the Negative bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNN Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Negative = 0;
PC = address `(Jump)`
If Negative = 1;
PC = address `(HERE+2)`

BNOV **Branch if Not Overflow**

Syntax: [*label*] BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if overflow bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0101	nnnn	nnnn

Description: If the Overflow bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNOV Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Overflow = 0;
PC = address `(Jump)`
If Overflow = 1;
PC = address `(HERE+2)`

BNZ **Branch if Not Zero**

Syntax: [*label*] BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if zero bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0001	nnnn	nnnn

Description: If the Zero bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNZ Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Zero = 0;
PC = address `(Jump)`
If Zero = 1;
PC = address `(HERE+2)`

BRA **Unconditional Branch**

Syntax: [*label*] BRA n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	0nnn	nnnn	nnnn

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE BRA Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
PC = address `(Jump)`

BSF **Bit Set f**

Syntax: [*label*] BSF f,b[,a]

Operands: 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

Operation: 1 → f<b>

Status Affected: None

Encoding:

1000	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is set. If 'a' is 0, Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `BSF FLAG_REG, 7, 1`

Before Instruction
FLAG_REG = 0x0A
After Instruction
FLAG_REG = 0x8A

BTFSC **Bit Test File, Skip if Clear**

Syntax: [*label*] BTFSC f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is 0, then the next instruction is skipped.
If bit 'b' is 0, then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

BTFSS **Bit Test File, Skip if Set**

Syntax: [*label*] BTFSS f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is 1, then the next instruction is skipped.
If bit 'b' is 1, then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)

BTG **Bit Toggle f**

Syntax: [*label*] BTG f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit ‘b’ in data memory location ‘f’ is inverted. If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register ‘f’

Example: `BTG PORTC, 4, 0`

Before Instruction:
PORTC = 0111 0101 [0x75]
After Instruction:
PORTC = 0110 0101 [0x65]

BOV **Branch if Overflow**

Syntax: [*label*] BOV n

Operands: $-128 \le n \le 127$

Operation: if overflow bit is ’1’
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the Overflow bit is ‘1’, then the program will branch.
The 2’s complement number ‘2n’ is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	No operation

Example:
```
HERE    BOV   Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 1;
PC = address (Jump)
If Overflow = 0;
PC = address (HERE+2)

BZ **Branch if Zero**

Syntax: [*label*] BZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '1'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the Zero bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BZ Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Zero = 1;
PC = address `(Jump)`
If Zero = 0;
PC = address `(HERE+2)`

CALL **Subroutine Call**

Syntax: [*label*] CALL k [,s]

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>,
if s = 1
(W) → WS,
(STATUS) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2 Mbyte memory range. First, return address (PC+ 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. `CALL` is a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `HERE CALL THERE,1`

Before Instruction
PC = address `(HERE)`
After Instruction
PC = address `(THERE)`
TOS = address `(HERE + 4)`
WS = W
BSRS = BSR
STATUSS= STATUS

CLRF **Clear f**

Syntax: [*label*] CLRF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `CLRF FLAG_REG,1`

Before Instruction
FLAG_REG = 0x5A
After Instruction
FLAG_REG = 0x00

CLRWDT **Clear Watchdog Timer**

Syntax: [*label*] CLRWDT

Operands: None

Operation: 000h → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0100

Description: `CLRWDT` instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. STATUS bits $\overline{TO}$ and $\overline{PD}$ are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: `CLRWDT`

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 0x00
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: [*label*] COMF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(\bar{f}) \rightarrow$ dest

Status Affected: N, Z

Encoding:

0001	11da	ffff	ffff

Description: The contents of register ‘f’ are complemented. If ‘d’ is 0, the result is stored in W. If ‘d’ is 1, the result is stored back in register ‘f’ (default). If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `COMF REG, 0, 0`

Before Instruction
REG = 0x13
After Instruction
REG = 0x13
W = 0xEC

CPFSEQ **Compare f with W, skip if f = W**

Syntax: [*label*] CPFSEQ f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	001a	ffff	ffff

Description: Compares the contents of data memory location ‘f’ to the contents of W by performing an unsigned subtraction.
If ‘f’ = W**,** then the fetched instruction is discarded and a `NOP` is executed instead, making this a two-cycle instruction. If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)

CPFSGT **Compare f with W, skip if f > W**

Syntax: [*label*] CPFSGT f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.
If the contents of 'f' are greater than the contents of WREG**,** then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```

Before Instruction
PC = Address (HERE)
W = ?

After Instruction
If REG > W;
PC = Address (GREATER)
If REG ≤ W;
PC = Address (NGREATER)

CPFSLT **Compare f with W, skip if f < W**

Syntax: [*label*] CPFSLT f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is 0, the Access Bank will be selected. If 'a' is 1, the BSR will not be overridden (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

Before Instruction
PC = Address (HERE)
W = ?

After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

DAW **Decimal Adjust W Register**

Syntax: [*label*] DAW

Operands: None

Operation: If [W<3:0> >9] or [DC = 1] then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>;

If [W<7:4> >9] or [C = 1] then
(W<7:4>) + 6 → W<7:4>;
else
(W<7:4>) → W<7:4>;

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the eight-bit value in W resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example1: DAW

Before Instruction
W = 0xA5
C = 0
DC = 0

After Instruction
W = 0x05
C = 1
DC = 0

Example 2:

Before Instruction
W = 0xCE
C = 0
DC = 0

After Instruction
W = 0x34
C = 1
DC = 0

DECF **Decrement f**

Syntax: [*label*] DECF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0000	01da	ffff	ffff

Description: Decrement register ‘f’. If ‘d’ is 0, the result is stored in W. If ‘d’ is 1, the result is stored back in register ‘f’ (default). If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: DECF CNT, 1, 0

Before Instruction
CNT = 0x01
Z = 0

After Instruction
CNT = 0x00
Z = 1

DECFSZ **Decrement f, skip if 0**

Syntax: [*label*] DECFSZ f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0010	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If the result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    DECFSZ  CNT, 1, 1
        GOTO    LOOP
CONTINUE
```

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT - 1
If CNT = 0;
PC = Address (CONTINUE)
If CNT ≠ 0;
PC = Address (HERE+2)

DCFSNZ **Decrement f, skip if not 0**

Syntax: [*label*] DCFSNZ f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If the result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```

Before Instruction
TEMP = ?

After Instruction
TEMP = TEMP - 1,
If TEMP = 0;
PC = Address (ZERO)
If TEMP ≠ 0;
PC = Address (NZERO)

GOTO **Unconditional Branch**

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	1111	k_7kkk	$kkkk_0$
2nd word(k<19:8>)	1111	$k_{19}kkk$	kkkk	$kkkk_8$

Description: GOTO allows an unconditional branch anywhere within entire 2 Mbyte memory range. The 20-bit value 'k' is loaded into PC<20:1>. GOTO is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: GOTO THERE

After Instruction
PC = Address (THERE)

INCF **Increment f**

Syntax: [*label*] INCF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0010	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: INCF CNT, 1, 0

Before Instruction
CNT = 0xFF
Z = 0
C = ?
DC = ?

After Instruction
CNT = 0x00
Z = 1
C = 1
DC = 1

INCFSZ **Increment f, skip if 0**

Syntax: [*label*] INCFSZ f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0011	11da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If the result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT ≠ 0;
PC = Address (NZERO)

INFSNZ **Increment f, skip if not 0**

Syntax: [*label*] INFSNZ f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If the result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead, making it a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INFSNZ  REG, 1, 0
ZERO
NZERO
```

Before Instruction
PC = Address (HERE)

After Instruction
REG = REG + 1
If REG ≠ 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

IORLW **Inclusive OR literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

0000	1001	kkkk	kkkk

Description: The contents of W are OR'ed with the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `IORLW 0x35`

Before Instruction
W = 0x9A

After Instruction
W = 0xBF

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `IORWF RESULT, 0, 1`

Before Instruction
RESULT = 0x13
W = 0x91

After Instruction
RESULT = 0x13
W = 0x93

LFSR **Load FSR**

Syntax: [*label*] LFSR f,k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 0x3AB`

After Instruction
FSR2H = 0x03
FSR2L = 0xAB

MOVF **Move f**

Syntax: [*label*] MOVF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256-byte bank. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 0x22
W = 0xFF

After Instruction
REG = 0x22
W = 0x22

MOVFF **Move f to f**

Syntax: [*label*] MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register 'f_s' are moved to destination register 'f_d'. Location of source 'f_s' can be anywhere in the 4096-byte data space (000h to FFFh), and location of destination 'f_d' can also be anywhere from 000h to FFFh.
Either source or destination can be W (a useful special situation).
MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).
The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 0x33
REG2 = 0x11

After Instruction
REG1 = 0x33
REG2 = 0x33

MOVLB **Move literal to low nibble in BSR**

Syntax: [*label*] MOVLB k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk

Description: The 8-bit literal 'k' is loaded into the Bank Select Register (BSR).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example: MOVLB 5

Before Instruction
BSR register = 0x02

After Instruction
BSR register = 0x05

MOVLW **Move literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

0000	1110	kkkk	kkkk

Description: The eight-bit literal ‘k’ is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: `MOVLW 0x5A`

After Instruction

W = 0x5A

MOVWF **Move W to f**

Syntax: [*label*] MOVWF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

0110	111a	ffff	ffff

Description: Move data from W to register ‘f’. Location ‘f’ can be anywhere in the 256-byte bank. If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register 'f'

Example: `MOVWF REG, 0`

Before Instruction

W = 0x4F
REG = 0xFF

After Instruction

W = 0x4F
REG = 0x4F

MULLW **Multiply Literal with W**

Syntax: [*label*] MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged. None of the status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0xC4`

Before Instruction

W = 0xE2
PRODH = ?
PRODL = ?

After Instruction

W = 0xE2
PRODH = 0xAD
PRODL = 0x08

MULWF **Multiply W with f**

Syntax: [*label*] MULWF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged. None of the status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a'= 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: `MULWF REG, 1`

Before Instruction

W = 0xC4
REG = 0xB5
PRODH = ?
PRODL = ?

After Instruction

W = 0xC4
REG = 0xB5
PRODH = 0x8A
PRODL = 0x94

NEGF **Negate f**

Syntax: [*label*] NEGF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: $(\overline{f}) + 1 \rightarrow f$

Status Affected: N, OV, C, DC, Z

Encoding:

0110	110a	ffff	ffff

Description: Location ‘f’ is negated using two’s complement. The result is placed in the data memory location ‘f’. If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register ‘f’

Example: `NEGF REG, 1`

Before Instruction
REG = `0011 1010` [0x3A]
After Instruction
REG = `1100 0110` [0xC6]

NOP **No Operation**

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

POP **Pop Top of Return Stack**

Syntax: [*label*] POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

0000	0000	0000	0110

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack.
This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

Example:
```
POP
GOTO    NEW
```
Before Instruction
TOS = 0031A2h
Stack (1 level down) = 014332h

After Instruction
TOS = 014332h
PC = NEW

PUSH **Push Top of Return Stack**

Syntax: [*label*] PUSH

Operands: None

Operation: (PC+2) → TOS

Status Affected: None

Encoding:

0000	0000	0000	0101

Description: The PC+2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows implementing a software stack by modifying TOS, and then pushing it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC+2 onto return stack	No operation	No operation

Example:
```
PUSH
```
Before Instruction
TOS = 00345Ah
PC = 000124h

After Instruction
PC = 000126h
TOS = 000126h
Stack (1 level down) = 00345Ah

RCALL **Relative Call**

Syntax: [*label*] RCALL n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	1nnn	nnnn	nnnn

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC+2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' Push PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE RCALL Jump`

Before Instruction
PC = Address `(HERE)`
After Instruction
PC = Address `(Jump)`
TOS = Address `(HERE+2)`

RESET **Reset**

Syntax: [*label*] RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: `RESET`

After Instruction
Registers = Reset Value
Flags* = Reset Value

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL,
if s = 1
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged.

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from Interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	Pop PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: `RETFIE 1`

After Interrupt
PC = TOS
W = WS
BSR = BSRS
STATUS = STATUSS
GIE/GIEH, PEIE/GIEL = 1

RETLW **Return Literal to W**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	1100	kkkk	kkkk

Description: W is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Pop PC from stack, Write to W
No operation	No operation	No operation	No operation

Example:

```
    CALL TABLE  ; W contains table
                ; offset value
                ; W now has
                ; table value
   :
TABLE
    ADDWF PCL   ; W = offset
    RETLW k0    ; Begin table
    RETLW k1    ;
   :
   :
    RETLW kn    ; End of table
```

Before Instruction
W = 0x07
After Instruction
W = value of kn

RETURN **Return from Subroutine**

Syntax: [*label*] RETURN [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
if s = 1
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	0000	0001	001s

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If 's'= 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Pop PC from stack
No operation	No operation	No operation	No operation

Example: `RETURN`

After Interrupt
PC = TOS

RLCF **Rotate Left f through Carry**

Syntax: [*label*] RLCF f [,d [,a]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

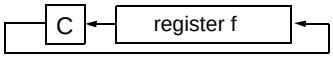
Operation: (f<n>) → dest<n+1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

Encoding:

0011	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is stored back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `RLCF REG, 0, 0`

Before Instruction
REG = 1110 0110
C = 0

After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF **Rotate Left f (no carry)**

Syntax: [*label*] RLNCF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

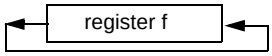
Operation: (f<n>) → dest<n+1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

0100	01da	ffff	ffff

Description: The contents of register ‘f’ are rotated one bit to the left. If ‘d’ is 0, the result is placed in W. If ‘d’ is 1, the result is stored back in register ‘f’ (default). If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ is 1, then the bank will be selected as per the BSR value (default).



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: RLNCF REG, 1, 0

Before Instruction
REG = 1010 1011
After Instruction
REG = 0101 0111

RRCF **Rotate Right f through Carry**

Syntax: [*label*] RRCF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

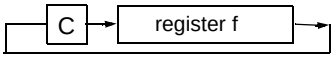
Operation: (f<n>) → dest<n-1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

0011	00da	ffff	ffff

Description: The contents of register ‘f’ are rotated one bit to the right through the Carry flag. If ‘d’ is 0, the result is placed in W. If ‘d’ is 1, the result is placed back in register ‘f’ (default). If ‘a’ is 0, the Access Bank will be selected, overriding the BSR value. If ‘a’ is 1, then the bank will be selected as per the BSR value (default).



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: RRCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 0111 0011
C = 0

RRNCF **Rotate Right f (no carry)**

Syntax: [*label*] RRNCF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

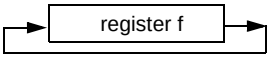
Operation: (f<n>) → dest<n-1>,
(f<0>) → dest<7>

Status Affected: N, Z

Encoding:

0100	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `RRNCF REG, 1, 0`

Before Instruction
REG = 1101 0111
After Instruction
REG = 1110 1011

Example 2: `RRNCF REG, 0, 0`

Before Instruction
W = ?
REG = 1101 0111
After Instruction
W = 1110 1011
REG = 1101 0111

SETF **Set f**

Syntax: [*label*] SETF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `SETF REG,1`

Before Instruction
REG = 0x5A
After Instruction
REG = 0xFF

SLEEP **Enter SLEEP mode**

Syntax: [*label*] SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description: The power-down status bit ($\overline{PD}$) is cleared. The time-out status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared.
The processor is put into SLEEP mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to SLEEP

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?

After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB **Subtract f from W with borrow**

Syntax: [*label*] SUBFWB f [,d [,a]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (W) – (f) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description: Subtract register 'f' and carry flag (borrow) from W (2's complement method). If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: SUBFWB REG, 1, 0

Before Instruction
REG = 3
W = 2
C = 1

After Instruction
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; result is negative

Example 2: SUBFWB REG, 0, 0

Before Instruction
REG = 2
W = 5
C = 1

After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3: SUBFWB REG, 1, 0

Before Instruction
REG = 1
W = 2
C = 0

After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0

SUBLW **Subtract W from literal**

Syntax: [*label*] SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from the eight-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1: `SUBLW 0x02`

Before Instruction
W = 1
C = ?

After Instruction
W = 1
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBLW 0x02`

Before Instruction
W = 2
C = ?

After Instruction
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBLW 0x02`

Before Instruction
W = 3
C = ?

After Instruction
W = FF ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(f) - (W) \rightarrow dest$

Status Affected: N, OV, C, DC, Z

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBWF REG, 1, 0`

Before Instruction
REG = 3
W = 2
C = ?

After Instruction
REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBWF REG, 0, 0`

Before Instruction
REG = 2
W = 2
C = ?

After Instruction
REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBWF REG, 1, 0`

Before Instruction
REG = 1
W = 2
C = ?

After Instruction
REG = FFh ;(2's complement)
W = 2
C = 0 ; result is negative
Z = 0
N = 1

SUBWFB **Subtract W from f with Borrow**

Syntax: [*label*] SUBWFB f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – (W) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	10da	ffff	ffff

Description: Subtract W and the carry flag (borrow) from register 'f' (2's complement method). If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBWFB REG, 1, 0`

Before Instruction
REG = 0x19 (0001 1001)
W = 0x0D (0000 1101)
C = 1

After Instruction
REG = 0x0C (0000 1011)
W = 0x0D (0000 1101)
C = 1
Z = 0
N = 0 ; result is positive

Example 2: `SUBWFB REG, 0, 0`

Before Instruction
REG = 0x1B (0001 1011)
W = 0x1A (0001 1010)
C = 0

After Instruction
REG = 0x1B (0001 1011)
W = 0x00
C = 1
Z = 1 ; result is zero
N = 0

Example 3: `SUBWFB REG, 1, 0`

Before Instruction
REG = 0x03 (0000 0011)
W = 0x0E (0000 1101)
C = 1

After Instruction
REG = 0xF5 (1111 0100)
; [2's comp]
W = 0x0E (0000 1101)
C = 0
Z = 0
N = 1 ; result is negative

SWAPF **Swap f**

Syntax: [*label*] SWAPF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<3:0>) → dest<7:4>,
(f<7:4>) → dest<3:0>

Status Affected: None

Encoding:

0011	10da	ffff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `SWAPF REG, 1, 0`

Before Instruction
REG = 0x53

After Instruction
REG = 0x35

TBLRD **Table Read**

Syntax: [*label*] TBLRD (*; *+; *-; +*)

Operands: None

Operation: if TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT;
TBLPTR - No Change;
if TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) +1 → TBLPTR;
if TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) -1 → TBLPTR;
if TBLRD +*,
(TBLPTR) +1 → TBLPTR;
(Prog Mem (TBLPTR)) → TABLAT;

Status Affected: None

Encoding:

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used. The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read Program Memory)	No operation	No operation (Write TABLAT)

TBLRD **Table Read (cont'd)**

Example1:

```
TBLRD  *+ ;
```

Before Instruction
TABLAT = 0x55
TBLPTR = 0x00A356
MEMORY(0x00A356) = 0x34

After Instruction
TABLAT = 0x34
TBLPTR = 0x00A357

Example2:

```
TBLRD  +* ;
```

Before Instruction
TABLAT = 0xAA
TBLPTR = 0x01A357
MEMORY(0x01A357) = 0x12
MEMORY(0x01A358) = 0x34

After Instruction
TABLAT = 0x34
TBLPTR = 0x01A358

TBLWT **Table Write**

Syntax: [*label*] TBLWT (*; *+; *-; +*)

Operands: None

Operation: if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR - No Change;
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) +1 → TBLPTR;
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) -1 → TBLPTR;
if TBLWT+*,
(TBLPTR) +1 → TBLPTR;
(TABLAT) → Holding Register;

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to Section 5.0 for additional details on programming FLASH memory.)
The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-MByte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The `TBLWT` instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

TBLWT Table Write (Continued)

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

Example1: `TBLWT *+;`

Before Instruction
TABLAT = 0x55
TBLPTR = 0x00A356
HOLDING REGISTER (0x00A356) = 0xFF

After Instructions (table write completion)
TABLAT = 0x55
TBLPTR = 0x00A357
HOLDING REGISTER (0x00A356) = 0x55

Example 2: `TBLWT +*;`

Before Instruction
TABLAT = 0x34
TBLPTR = 0x01389A
HOLDING REGISTER (0x01389A) = 0xFF
HOLDING REGISTER (0x01389B) = 0xFF

After Instruction (table write completion)
TABLAT = 0x34
TBLPTR = 0x01389B
HOLDING REGISTER (0x01389A) = 0xFF
HOLDING REGISTER (0x01389B) = 0x34

TSTFSZ **Test f, skip if 0**

Syntax: [*label*] TSTFSZ f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction, fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
If CNT = 0x00,
PC = Address (ZERO)
If CNT ≠ 0x00,
PC = Address (NZERO)

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: XORLW 0xAF

Before Instruction
W = 0xB5

After Instruction
W = 0x1A

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f [,d [,a]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .XOR. (f) → dest

Status Affected: N, Z

Encoding:

0001	10da	ffff	ffff

Description: Exclusive OR the contents of W with register 'f'. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in the register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `XORWF REG, 1, 0`

Before Instruction
REG = 0xAF
W = 0xB5

After Instruction
REG = 0x1A
W = 0xB5

26.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C17 and MPLAB C18 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB C30 C Compiler
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
  - MPLAB dsPIC30 Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PRO MATE® II Universal Device Programmer
  - PICSTART® Plus Development Programmer
- Low Cost Demonstration Boards
  - PICDEM™ 1 Demonstration Board
  - PICDEM.net™ Demonstration Board
  - PICDEM 2 Plus Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 4 Demonstration Board
  - PICDEM 17 Demonstration Board
  - PICDEM 18R Demonstration Board
  - PICDEM LIN Demonstration Board
  - PICDEM USB Demonstration Board
- Evaluation Kits
  - KEELOQ®
  - PICDEM MSC
  - microID®
  - CAN
  - PowerSmart®
  - Analog

26.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® based application that contains:

- An interface to debugging tools
  - simulator
  - programmer (sold separately)
  - emulator (sold separately)
  - in-circuit debugger (sold separately)
- A full-featured editor with color coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High level source code debugging
- Mouse over variable inspection
- Extensive on-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
  - source files (assembly or C)
  - absolute listing file (mixed assembly and C)
  - machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost effective simulators, through low cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increasing flexibility and power.

26.2 MPASM Assembler

The MPASM assembler is a full-featured, universal macro assembler for all PICmicro MCUs.

The MPASM assembler generates relocatable object files for the MPLINK object linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM assembler features include:

- Integration into MPLAB IDE projects
- User defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

26.3 MPLAB C17 and MPLAB C18 C Compilers

The MPLAB C17 and MPLAB C18 Code Development Systems are complete ANSI C compilers for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

26.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can link relocatable objects from pre-compiled libraries, using directives from a linker script.

The MPLIB object librarian manages the creation and modification of library files of pre-compiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

26.5 MPLAB C30 C Compiler

The MPLAB C30 C compiler is a full-featured, ANSI compliant, optimizing compiler that translates standard ANSI C programs into dsPIC30F assembly language source. The compiler also supports many command-line options and language extensions to take full advantage of the dsPIC30F device hardware capabilities, and afford fine control of the compiler code generator.

MPLAB C30 is distributed with a complete ANSI C standard library. All library functions have been validated and conform to the ANSI C library standard. The library includes functions for string manipulation, dynamic memory allocation, data conversion, time-keeping, and math functions (trigonometric, exponential and hyperbolic). The compiler provides symbolic information for high level source debugging with the MPLAB IDE.

26.6 MPLAB ASM30 Assembler, Linker, and Librarian

MPLAB ASM30 assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 compiler uses the assembler to produce it's object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

26.7 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC hosted environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any pin. The execution can be performed in Single-Step, Execute Until Break, or Trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and MPLAB C18 C Compilers, as well as the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent, economical software development tool.

26.8 MPLAB SIM30 Software Simulator

The MPLAB SIM30 software simulator allows code development in a PC hosted environment by simulating the dsPIC30F series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any of the pins.

The MPLAB SIM30 simulator fully supports symbolic debugging using the MPLAB C30 C Compiler and MPLAB ASM30 assembler. The simulator runs in either a Command Line mode for automated tasks, or from MPLAB IDE. This high speed simulator is designed to debug, analyze and optimize time intensive DSP routines.

26.9 MPLAB ICE 2000 High Performance Universal In-Circuit Emulator

The MPLAB ICE 2000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers. Software control of the MPLAB ICE 2000 in-circuit emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE 2000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

26.10 MPLAB ICE 4000 High Performance Universal In-Circuit Emulator

The MPLAB ICE 4000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PICmicro microcontrollers. Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICD 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, up to 2 Mb of emulation memory, and the ability to view variables in real-time.

The MPLAB ICE 4000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

26.11 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low cost, run-time development tool, connecting to the host PC via an RS-232 or high speed USB interface. This tool is based on the FLASH PICmicro MCUs and can be used to develop for these and other PICmicro microcontrollers. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the FLASH devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost effective in-circuit FLASH debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single-stepping and watching variables, CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real-time. MPLAB ICD 2 also serves as a development programmer for selected PICmicro devices.

26.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features an LCD display for instructions and error messages and a modular detachable socket assembly to support various package types. In Stand-Alone mode, the PRO MATE II device programmer can read, verify, and program PICmicro devices without a PC connection. It can also set code protection in this mode.

26.13 PICSTART Plus Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus development programmer supports most PICmicro devices up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

26.14 PICDEM 1 PICmicro Demonstration Board

The PICDEM 1 demonstration board demonstrates the capabilities of the PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The sample microcontrollers provided with the PICDEM 1 demonstration board can be programmed with a PRO MATE II device programmer, or a PICSTART Plus development programmer. The PICDEM 1 demonstration board can be connected to the MPLAB ICE in-circuit emulator for testing. A prototype area extends the circuitry for additional application components. Features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs.

26.15 PICDEM.net Internet/Ethernet Demonstration Board

The PICDEM.net demonstration board is an Internet/ Ethernet demonstration board using the PIC18F452 microcontroller and TCP/IP firmware. The board supports any 40-pin DIP device that conforms to the standard pinout used by the PIC16F877 or PIC18C452. This kit features a user friendly TCP/IP stack, web server with HTML, a 24L256 Serial EEPROM for Xmodem download to web pages into Serial EEPROM, ICSP/MPLAB ICD 2 interface connector, an Ethernet interface, RS-232 interface, and a 16 x 2 LCD display. Also included is the book and CD-ROM *"TCP/IP Lean, Web Servers for Embedded Systems,"* by Jeremy Bentham

26.16 PICDEM 2 Plus Demonstration Board

The PICDEM 2 Plus demonstration board supports many 18-, 28-, and 40-pin microcontrollers, including PIC16F87X and PIC18FXX2 devices. All the necessary hardware and software is included to run the demonstration programs. The sample microcontrollers provided with the PICDEM 2 demonstration board can be programmed with a PRO MATE II device programmer, PICSTART Plus development programmer, or MPLAB ICD 2 with a Universal Programmer Adapter. The MPLAB ICD 2 and MPLAB ICE in-circuit emulators may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area extends the circuitry for additional application components. Some of the features include an RS-232 interface, a 2 x 16 LCD display, a piezo speaker, an on-board temperature sensor, four LEDs, and sample PIC18F452 and PIC16F877 FLASH microcontrollers.

26.17 PICDEM 3 PIC16C92X Demonstration Board

The PICDEM 3 demonstration board supports the PIC16C923 and PIC16C924 in the PLCC package. All the necessary hardware and software is included to run the demonstration programs.

26.18 PICDEM 4 8/14/18-Pin Demonstration Board

The PICDEM 4 can be used to demonstrate the capabilities of the 8-, 14-, and 18-pin PIC16XXXX and PIC18XXXX MCUs, including the PIC16F818/819, PIC16F87/88, PIC16F62XA and the PIC18F1320 family of microcontrollers. PICDEM 4 is intended to showcase the many features of these low pin count parts, including LIN and Motor Control using ECCP. Special provisions are made for low power operation with the supercapacitor circuit, and jumpers allow on-board hardware to be disabled to eliminate current draw in this mode. Included on the demo board are provisions for Crystal, RC or Canned Oscillator modes, a five volt regulator for use with a nine volt wall adapter or battery, DB-9 RS-232 interface, ICD connector for programming via ICSP and development with MPLAB ICD 2, 2x16 liquid crystal display, PCB footprints for H-Bridge motor driver, LIN transceiver and EEPROM. Also included are: header for expansion, eight LEDs, four potentiometers, three push buttons and a prototyping area. Included with the kit is a PIC16F627A and a PIC18F1320. Tutorial firmware is included along with the User's Guide.

26.19 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. A programmed sample is included. The PRO MATE II device programmer, or the PICSTART Plus development programmer, can be used to reprogram the device for user tailored application development. The PICDEM 17 demonstration board supports program download and execution from external on-board FLASH memory. A generous prototype area is available for user hardware expansion.

26.20 PICDEM 18R PIC18C601/801 Demonstration Board

The PICDEM 18R demonstration board serves to assist development of the PIC18C601/801 family of Microchip microcontrollers. It provides hardware implementation of both 8-bit Multiplexed/De-multiplexed and 16-bit Memory modes. The board includes 2 Mb external FLASH memory and 128 Kb SRAM memory, as well as serial EEPROM, allowing access to the wide range of memory types supported by the PIC18C601/801.

26.21 PICDEM LIN PIC16C43X Demonstration Board

The powerful LIN hardware and software kit includes a series of boards and three PICmicro microcontrollers. The small footprint PIC16C432 and PIC16C433 are used as slaves in the LIN communication and feature on-board LIN transceivers. A PIC16F874 FLASH microcontroller serves as the master. All three microcontrollers are programmed with firmware to provide LIN bus communication.

26.22 PICkit™ 1 FLASH Starter Kit

A complete "development system in a box", the PICkit FLASH Starter Kit includes a convenient multi-section board for programming, evaluation, and development of 8/14-pin FLASH PIC® microcontrollers. Powered via USB, the board operates under a simple Windows GUI. The PICkit 1 Starter Kit includes the user's guide (on CD ROM), PICkit 1 tutorial software and code for various applications. Also included are MPLAB® IDE (Integrated Development Environment) software, software and hardware "Tips 'n Tricks for 8-pin FLASH PIC® Microcontrollers" Handbook and a USB Interface Cable. Supports all current 8/14-pin FLASH PIC microcontrollers, as well as many future planned devices.

26.23 PICDEM USB PIC16C7X5 Demonstration Board

The PICDEM USB Demonstration Board shows off the capabilities of the PIC16C745 and PIC16C765 USB microcontrollers. This board provides the basis for future USB products.

26.24 Evaluation and Programming Tools

In addition to the PICDEM series of circuits, Microchip has a line of evaluation kits and demonstration software for these products.

- KEELOQ evaluation and programming tools for Microchip's HCS Secure Data Products
- CAN developers kit for automotive network applications
- Analog design boards and filter design software
- PowerSmart battery charging evaluation/ calibration kits
- IrDA® development kit
- microID development and rfLab™ development software
- SEEVAL® designer kit for memory evaluation and endurance calculations
- PICDEM MSC demo boards for Switching mode power supply, high power IR driver, delta sigma ADC, and flow rate sensor

Check the Microchip web page and the latest Product Line Card for the complete list of demonstration and evaluation kits.

NOTES:

27.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -55°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$, and RA4) ..... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to +5.5V
Voltage on $\overline{MCLR}$ with respect to V_{SS} **(Note 2)** ..... 0V to +13.25V
Voltage on RA4 with respect to Vss ..... 0V to +8.5V
Total power dissipation **(Note 1)** 1.0W
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin ..... 250 mA
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) ..... ±20 mA
Maximum output current sunk by any I/O pin ..... 25 mA
Maximum output current sourced by any I/O pin ..... 25 mA
Maximum current sunk by all ports ..... 200 mA
Maximum current sourced by all ports ..... 200 mA

Note 1: Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

2: Voltage spikes below V_{SS} at the $\overline{MCLR}/V_{PP}$ pin, inducing currents greater than 80 mA, may cause latchup. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}/V_{PP}$ pin, rather than pulling this pin directly to V_{SS}.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 27-1: PIC18F6X2X/8X2X VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL, EXTENDED)

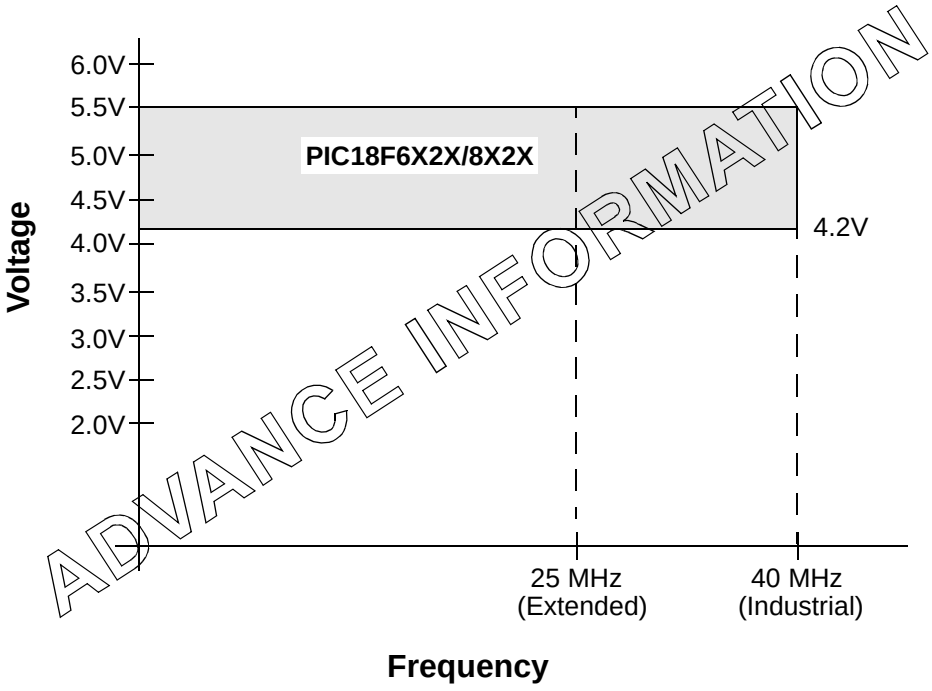
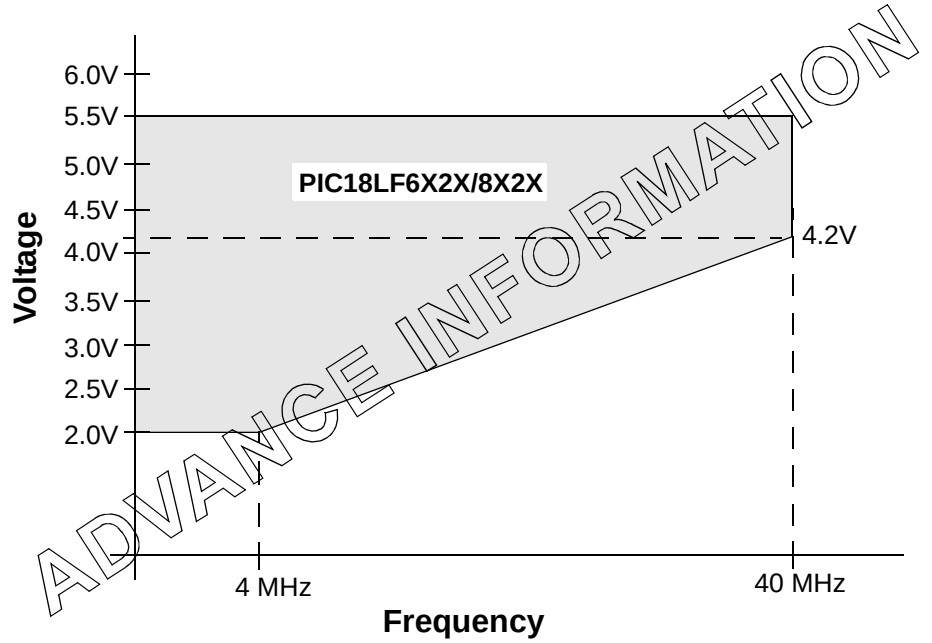


FIGURE 27-2: PIC18LF6X2X/8X2X VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



F_{MAX} = (16.36 MHz/V) ($V_{DDAPPMIN}$ - 2.0V) + 4 MHz

Note: $V_{DDAPPMIN}$ is the minimum voltage of the PICmicro® device in the application.

27.1 DC Characteristics: Supply Voltage PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial)

PIC18LF6X2X/8X2X (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F6X2X/8X2X (Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**					
		PIC18LF6X2X/8X2X	2.0	—	5.5	V	HS, XT, RC and LP Osc mode
		PIC18F6X2X/8X2X	4.2	—	5.5	V	
D002	VDR	**RAM Data Retention Voltage(1)**	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	—	0.7	V	See section on Power-on Reset for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See section on Power-on Reset for details
D005	VBOR	**Brown-out Reset Voltage**					
		BORV1:BORV0 = 11	1.98	—	2.14	V	
		BORV1:BORV0 = 10	2.67	—	2.89	V	
		BORV1:BORV0 = 01	4.16	—	4.5	V	
		BORV1:BORV0 = 00	4.45	—	4.83	V	

Legend: Shading of rows is to assist in readability of the table.

Note 1: This is the limit to which VDD can be lowered in SLEEP mode or during a device RESET without losing RAM data.

27.2 DC Characteristics: Power-Down and Supply Current PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial)

PIC18LF6X2X/8X2X (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F6X2X/8X2X (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**	
	Power-down Current (IPD)(1)					
	PIC18LF6X2X/8X2X	0.2	1	µA	-40°C	VDD = 2.0V, (SLEEP mode)
		0.2	1	µA	25°C	
		1.2	5	µA	85°C	
	PIC18LF6X2X/8X2X	0.4	1	µA	-40°C	VDD = 3.0V, (SLEEP mode)
		0.4	1	µA	25°C	
		1.8	8	µA	85°C	
	All devices	0.7	2	µA	-40°C	VDD = 5.0V, (SLEEP mode)
		0.7	2	µA	25°C	
		3.0	15	µA	85°C	

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

27.2 DC Characteristics: Power-Down and Supply Current PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial) (Continued)

PIC18LF6X2X/8X2X (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F6X2X/8X2X (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC18LF6X2X/8X2X	165	350	µA	-40°C	VDD = 2.0V	FOSC = 1 MHz, EC oscillator
		165	350	µA	25°C		
		170	350	µA	85°C		
	PIC18LF6X2X/8X2X	360	750	µA	-40°C	VDD = 3.0V	
		340	750	µA	25°C		
		300	750	µA	85°C		
	All devices	800	1700	µA	-40°C	VDD = 5.0V	
		730	1700	µA	25°C		
		700	1700	µA	85°C		
	PIC18LF6X2X/8X2X	600	1200	µA	-40°C	VDD = 2.0V	FOSC = 4 MHz, EC oscillator
		600	1200	µA	25°C		
		640	1300	µA	85°C		
	PIC18LF6X2X/8X2X	1000	2500	µA	-40°C	VDD = 3.0V	
		1000	2500	µA	25°C		
		1000	2500	µA	85°C		
	All devices	2.2	5.0	mA	-40°C	VDD = 5.0V	
		2.1	5.0	mA	25°C		
		2.0	5.0	mA	85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

27.2 DC Characteristics: Power-Down and Supply Current PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial) (Continued)

PIC18LF6X2X/8X2X (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F6X2X/8X2X (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	All devices	9.3	15	mA	-40°C	VDD = 4.2V	FOSC = 25 MHz, EC oscillator
		9.5	15	mA	25°C		
		10	15	mA	85°C		
	All devices	11.8	20	mA	-40°C	VDD = 5.0V	
		12	20	mA	25°C		
		12	20	mA	85°C		
	All devices	14	25	mA	-40°C	VDD = 4.2V	FOSC = 40 MHz, EC oscillator
		14	25	mA	25°C		
		15	25	mA	85°C		
	All devices	19	25	mA	-40°C	VDD = 5.0V	
		19	25	mA	25°C		
		20	25	mA	85°C		
	PIC18LF6X2X/8X2X	15	TBD	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz, Timer1 as clock
		15	TBD	µA	25°C		
		15	TBD	µA	70°C		
	PIC18LF6X2X/8X2X	28	TBD	µA	-10°C	VDD = 3.0V	
		28	TBD	µA	25°C		
		28	TBD	µA	70°C		
	All devices	75	TBD	µA	-10°C	VDD = 5.0V	
		75	TBD	µA	25°C		
		75	TBD	µA	70°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

27.2 DC Characteristics: Power-Down and Supply Current PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial) (Continued)

PIC18LF6X2X/8X2X (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F6X2X/8X2X (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)						
D022 (ΔIWDT)	**Watchdog Timer**	<1	2.0	µA	-40°C	VDD = 2.0V	
		<1	1.5	µA	25°C		
		<1	3	µA	85°C		
		3	10	µA	-40°C	VDD = 3.0V	
		2.5	6	µA	25°C		
		3	15	µA	85°C		
		15	25	µA	-40°C	VDD = 5.0V	
		12	20	µA	25°C		
		12	25	µA	85°C		
D022A (ΔIBOR)	**Brown-out Reset**	35	50	µA	-40°C to +85°C	VDD = 3.0V	
		45	65	µA	-40°C to +85°C	VDD = 5.0V	
D022B (ΔILVD)	**Low Voltage Detect**	33	45	µA	-40°C to +85°C	VDD = 2.0V	
		35	50	µA	-40°C to +85°C	VDD = 3.0V	
		45	65	µA	-40°C to +85°C	VDD = 5.0V	
D025 (ΔIOSCB)	**Timer1 Oscillator**	5.2	TBD	µA	-10°C	VDD = 2.0V	32 kHz on Timer1
		5.2	TBD	µA	25°C		
		5.2	TBD	µA	70°C		
		5.8	TBD	µA	-10°C	VDD = 3.0V	32 kHz on Timer1
		5.8	TBD	µA	25°C		
		5.8	TBD	µA	70°C		
		7.2	TBD	µA	-10°C	VDD = 5.0V	32 kHz on Timer1
		7.2	TBD	µA	25°C		
		7.2	TBD	µA	70°C		
D026 (ΔIAD)	**A/D Converter**	<1	2	µA	25°C	VDD = 2.0V	A/D on, not converting
		<1	2	µA	25°C	VDD = 3.0V	
		<1	2	µA	25°C	VDD = 5.0V	

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

27.3 DC Characteristics: PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		I/O ports:				
D030		with TTL buffer	VSS	0.15 VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger buffer	VSS	0.2 VDD	V	
		RC3 and RC4	VSS	0.3 VDD	V	
D032		$\overline{MCLR}$	VSS	0.2 VDD	V	
D032A		OSC1 (in XT, HS and LP modes) and T1OSI	VSS	0.3 VDD	V	
D033		OSC1 (in RC and EC mode)(1)	VSS	0.2 VDD	V	
	VIH	**Input High Voltage**				
		I/O ports:				
D040		with TTL buffer	0.25 VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		with Schmitt Trigger buffer	0.8 VDD	VDD	V	
		RC3 and RC4	0.7 VDD	VDD	V	
D042		$\overline{MCLR}$, OSC1 (EC mode)	0.8 VDD	VDD	V	
D042A		OSC1 (in XT, HS and LP modes) and T1OSI	0.7 VDD	VDD	V	
D043		OSC1 (RC mode)(1)	0.9 VDD	VDD	V	
	IIL	**Input Leakage Current(2,3)**				
D060		I/O ports	—	±1	µA	VSS ≤ VPIN ≤ VDD, Pin at hi-impedance
D061		$\overline{MCLR}$	—	±5	µA	VSS ≤ VPIN ≤ VDD
D063		OSC1	—	±5	µA	VSS ≤ VPIN ≤ VDD
	IPU	**Weak Pull-up Current**				
D070	IPURB	PORTB weak pull-up current	50	400	µA	VDD = 5V, VPIN = VSS

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PICmicro device be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

4: Parameter is characterized but not tested.

27.3 DC Characteristics: PIC18F6X2X/8X2X (Industrial, Extended) PIC18LF6X2X/8X2X (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**				
D080		I/O ports	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C
D080A			—	0.6	V	IOL = 7.0 mA, VDD = 4.5V, -40°C to +125°C
D083		OSC2/CLKO (RC mode)	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C to +85°C
D083A			—	0.6	V	IOL = 1.2 mA, VDD = 4.5V, -40°C to +125°C
	VOH	**Output High Voltage(3)**				
D090		I/O ports	VDD – 0.7	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C to +85°C
D090A			VDD – 0.7	—	V	IOH = -2.5 mA, VDD = 4.5V, -40°C to +125°C
D092		OSC2/CLKO (RC mode)	VDD – 0.7	—	V	IOH = -1.3 mA, VDD = 4.5V, -40°C to +85°C
D092A			VDD – 0.7	—	V	IOH = -1.0 mA, VDD = 4.5V, -40°C to +125°C
D150	VOD	**Open Drain High Voltage**	—	8.5	V	RA4 pin
		Capacitive Loading Specs on Output Pins				
D100(4)	COSC2	OSC2 pin	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1
D101	CIO	All I/O pins and OSC2 (in RC mode)	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	In I^2C mode

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PICmicro device be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

4: Parameter is characterized but not tested.

TABLE 27-1: COMPARATOR SPECIFICATIONS

Operating Conditions: 3.0V < V_{DD} < 5.5V, -40°C < T_A < +125°C (unless otherwise stated)							
Param No.	**Characteristics**	**Sym**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D300	Input Offset Voltage	V_{IOFF}	—	± 5.0	± 10	mV	
D301	Input Common Mode Voltage	V_{ICM}	0	-	V_{DD} – 1.5	V	
D302	Common Mode Rejection Ratio	CMRR	55	-	—	dB	
300 300A	Response Time(1)	T_{RESP}	—	150	400 600	ns ns	PIC18F6X2X/8X2X PIC18LF6X2X/8X2X
301	Comparator Mode Change to Output Valid	T_{MC2OV}	—	—	10	µs	

Note 1: Response time measured with one comparator input at (V_{DD} – 1.5)/2 while the other input transitions from V_{SS} to V_{DD}.

TABLE 27-2: VOLTAGE REFERENCE SPECIFICATIONS

Operating Conditions: 3.0V < V_{DD} < 5.5V, -40°C < T_A < +125°C (unless otherwise stated)							
Spec No.	**Characteristics**	**Sym**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D310	Resolution	V_{RES}	V_{DD}/24	—	V_{DD}/32	LSb	
D311	Absolute Accuracy	V_{RAA}	— —	— —	1/4 1/2	LSb LSb	Low Range (VRR = 1) High Range (VRR = 0)
D312	Unit Resistor Value (R)	V_{RUR}	—	2k	—	Ω	
310	Settling Time(1)	T_{SET}	—	—	10	µs	

Note 1: Settling time measured while VRR = 1 and VR<3:0> transitions from 0000 to 1111.

FIGURE 27-3: LOW VOLTAGE DETECT CHARACTERISTICS

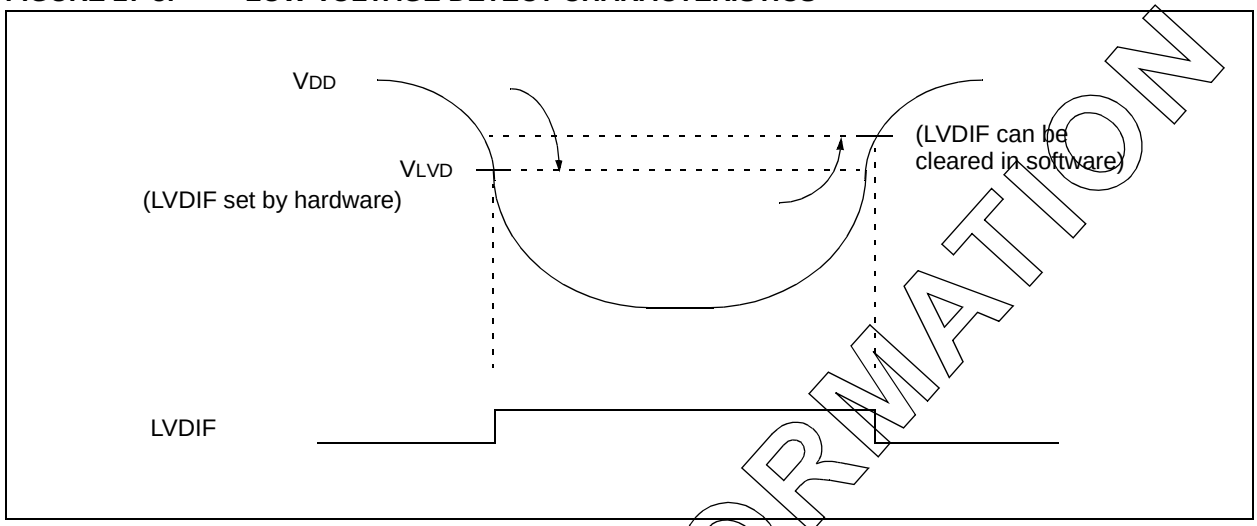


TABLE 27-3: LOW VOLTAGE DETECT CHARACTERISTICS

				Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D420	VLVD	LVD Voltage on VDD transition high to low	LVV = 0001	1.98	2.06	2.14	V	T ≥ 25°C
			LVV = 0010	2.18	2.27	2.36	V	T ≥ 25°C
			LVV = 0011	2.37	2.47	2.57	V	T ≥ 25°C
			LVV = 0100	2.48	2.58	2.68	V	
			LVV = 0101	2.67	2.78	2.89	V	
			LVV = 0110	2.77	2.89	3.01	V	
			LVV = 0111	2.98	3.1	3.22	V	
			LVV = 1000	3.27	3.41	3.55	V	
			LVV = 1001	3.47	3.61	3.75	V	
			LVV = 1010	3.57	3.72	3.87	V	
			LVV = 1011	3.76	3.92	4.08	V	
			LVV = 1100	3.96	4.13	4.3	V	
			LVV = 1101	4.16	4.33	4.5	V	
			LVV = 1110	4.45	4.64	4.83	V	
D423	VBG	Bandgap Reference Voltage Value		—	1.22	—	V	

† Production tested at TAMB = 25°C. Specifications over temp. limits ensured by characterization.

TABLE 27-4: MEMORY PROGRAMMING REQUIREMENTS

DC Characteristics			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Internal Program Memory Programming Specifications (Note 1)					
D110	VPP	Voltage on $\overline{MCLR}$/VPP pin	9.00	—	13.25	V	**(Note 2)**
D112	IPP	Current into $\overline{MCLR}$/VPP pin	—	—	5	µA	
D113	IDDP	Supply Current during Programming	—	—	10	mA	
		Data EEPROM Memory					
D120	ED	Cell Endurance	100K	1M	—	E/W	-40°C to +85°C
D120A	ED	Cell Endurance	10K	100K	—	E/W	+85°C to +125°C
D121	VDRW	VDD for Read/Write	VMIN	—	5.5	V	Using EECON to read/write VMIN = Minimum operating voltage
D122	TDEW	Erase/Write Cycle Time	—	4	—	ms	
D123	TRETD	Characteristic Retention	40	—	—	Year	-40°C to +85°C **(Note 3)**
D123A	TRETD	Characteristic Retention	100	—	—	Year	25°C **(Note3)**
		Program FLASH Memory					
D130	EP	Cell Endurance	10K	100K	—	E/W	-40°C to +85°C
D130A	EP	Cell Endurance	1000	10K	—	E/W	+85°C to +125°C
D131	VPR	VDD for Read	VMIN	—	5.5	V	VMIN = Minimum operating voltage
D132	VIE	VDD for Block Erase	4.5	—	5.5	V	Using ICSP port
D132A	VIW	VDD for Externally Timed Erase or Write	4.5	—	5.5	V	Using ICSP port
D132B	VPEW	VDD for Self-timed Write	VMIN	—	5.5	V	VMIN = Minimum operating voltage
D133	TIE	ICSP Block Erase Cycle Time	—	5	—	ms	VDD > 4.5V
D133A	TIW	ICSP Erase or Write Cycle Time (externally timed)	1	—	—	ms	VDD > 4.5V
D133A	TIW	Self-timed Write Cycle Time	—	2.5	—	ms	
D134	TRETD	Characteristic Retention	40	—	—	Year	-40°C to +85°C **(Note3)**
D134A	TRETD	Characteristic Retention	100	—	—	Year	25°C **(Note3)**

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: These specifications are for programming the on-chip program memory through the use of table write instructions.

2: The pin may be kept in this range at times other than programming but it is not recommended.

3: Retention time is valid provided no other specifications are violated.

27.4 AC (Timing) Characteristics

27.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS		3. TCC:ST	(I^2C specifications only)
2. TppS		4. Ts	(I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I^2C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	STOP condition
STA	START condition		

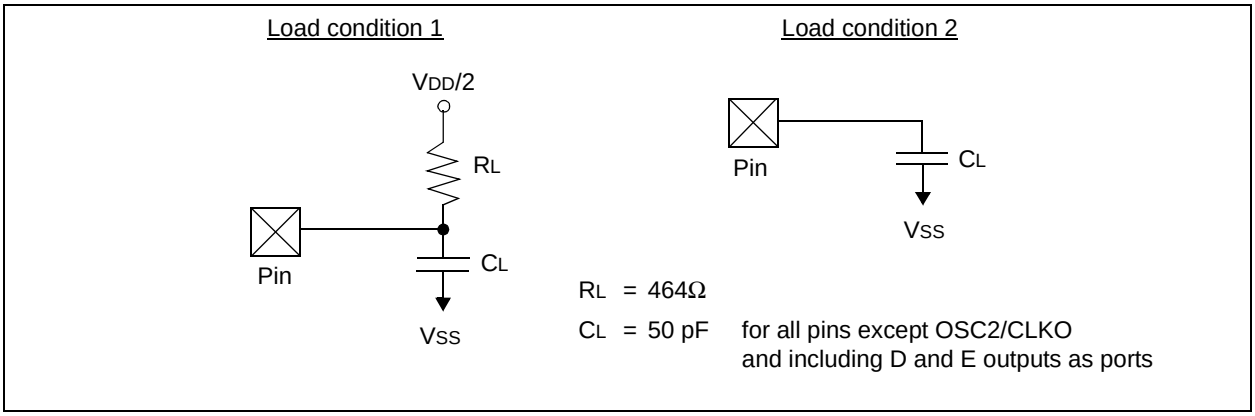
27.4.2 TIMING CONDITIONS

The temperature and voltages specified in Table 27-5 apply to all timing specifications, unless otherwise noted. Figure 27-4 specifies the load conditions for the timing specifications.

TABLE 27-5: TEMPERATURE AND VOLTAGE SPECIFICATIONS - AC

AC CHARACTERISTICS	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended Operating voltage VDD range as described in DC spec Section 27.1 and Section 27.3. LC parts operate for industrial temperatures only.

FIGURE 27-4: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



27.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 27-5: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)

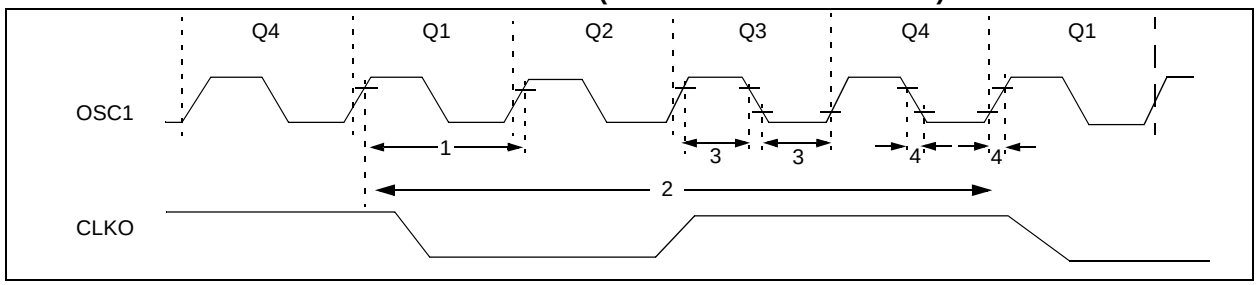


TABLE 27-6: EXTERNAL CLOCK TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	FOSC	External CLKI Frequency(1)	DC	25	MHz	EC, ECIO
		Oscillator Frequency(1)	DC	40	MHz	EC, ECIO
			DC	4	MHz	RC osc
			0.1	4	MHz	XT osc
			4	25	MHz	HS osc
			4	10	MHz	HS + PLL osc
			4	6.25	MHz	HS + PLL osc
			5	200	kHz	LP Osc mode
1	TOSC	External CLKI Period(1)	25	—	ns	EC, ECIO
		Oscillator Period(1)	160	—	ns	EC, ECIO
			250	—	ns	RC osc
			250	10,000	ns	XT osc
			25	250	ns	HS osc
			100	250	ns	HS + PLL osc
			100	160	ns	HS + PLL osc
			25	—	µs	LP osc
2	TCY	Instruction Cycle Time(1)	100	—	ns	TCY = 4/FOSC
3	TosL, TosH	External Clock in (OSC1) High or Low Time	30	—	ns	XT osc
			2.5	—	µs	LP osc
			10	—	ns	HS osc
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	20	ns	XT osc
			—	50	ns	LP osc
			—	7.5	ns	HS osc

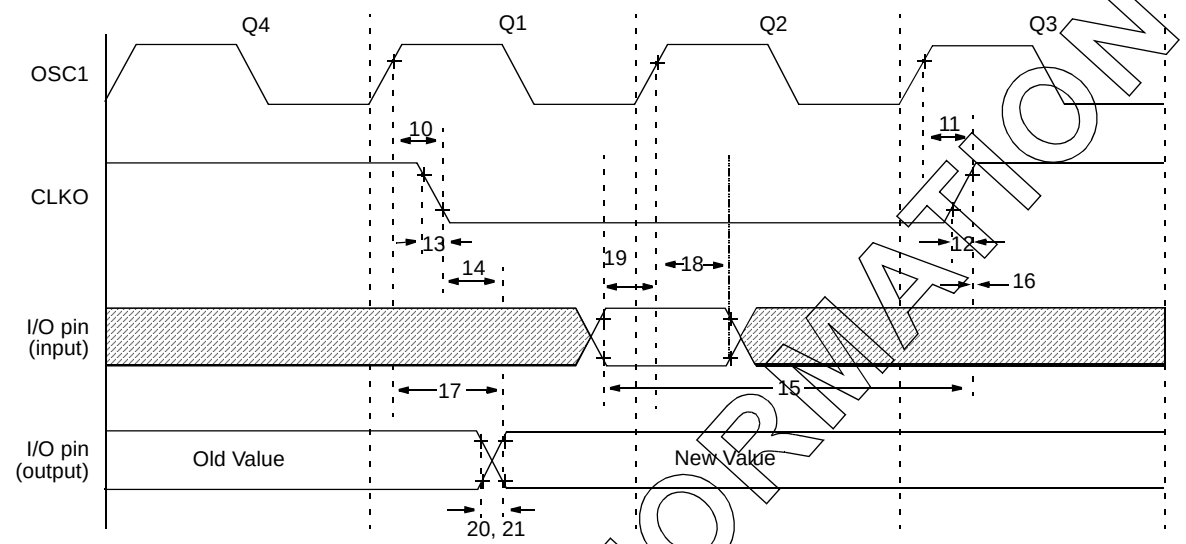
Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the "max." cycle time limit is "DC" (no clock) for all devices.

TABLE 27-7: PLL CLOCK TIMING SPECIFICATIONS (VDD = 4.2 TO 5.5V)

Param. No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	Oscillator Frequency Range	4	—	10	MHz	HS mode
	FSYS	On-chip VCO System Frequency	16	—	40	MHz	HS mode
	t_{rc}	PLL Start-up Time (Lock Time)	—	—	2	ms	
	ΔCLK	CLKO Stability (Jitter)	-2	—	+2	%	

† Data in "Typ" column is at 5V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 27-6: CLKO AND I/O TIMING



Note: Refer to Figure 27-4 for load conditions.

TABLE 27-8: CLKO AND I/O TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
10	TosH2ckL	OSC1 ↑ to CLKO ↓		—	75	200	ns	**(Note 1)**
11	TosH2ckH	OSC1 ↑ to CLKO ↑		—	75	200	ns	**(Note 1)**
12	TckR	CLKO rise time		—	35	100	ns	**(Note 1)**
13	TckF	CLKO fall time		—	35	100	ns	**(Note 1)**
14	TckL2ioV	CLKO ↓ to Port out valid		—	—	0.5 TCY + 20	ns	**(Note 1)**
15	TioV2ckH	Port in valid before CLKO ↑		0.25 TCY + 25	—	—	ns	**(Note 1)**
16	TckH2iol	Port in hold after CLKO ↑		0	—	—	ns	**(Note 1)**
17	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port out valid		—	50	150	ns	
18	TosH2iol	OSC1 ↑ (Q2 cycle) to Port input invalid (I/O in hold time)	PIC18F6X2X/8X2X	100	—	—	ns	
18A			PIC18LF6X2X/8X2X	200	—	—	ns	
19	TioV2osH	Port input valid to OSC1 ↑ (I/O in setup time)		0	—	—	ns	
20	TioR	Port output rise time	PIC18F6X2X/8X2X	—	10	25	ns	
20A			PIC18LF6X2X/8X2X	—	—	60	ns	
21	TioF	Port output fall time	PIC18F6X2X/8X2X	—	10	25	ns	
21A			PIC18LF6X2X/8X2X	—	—	60	ns	
22††	TINP	INT pin high or low time		TCY	—	—	ns	
23††	TRBP	RB7:RB4 change INT high or low time		TCY	—	—	ns	
24††	TRCP	RC7:RC4 change INT high or low time		20			ns	

†† These parameters are asychronous events not related to any internal clock edges.

Note 1: Measurements are taken in RC mode, where CLKO output is 4 x TOSC.

FIGURE 27-7: PROGRAM MEMORY READ TIMING DIAGRAM

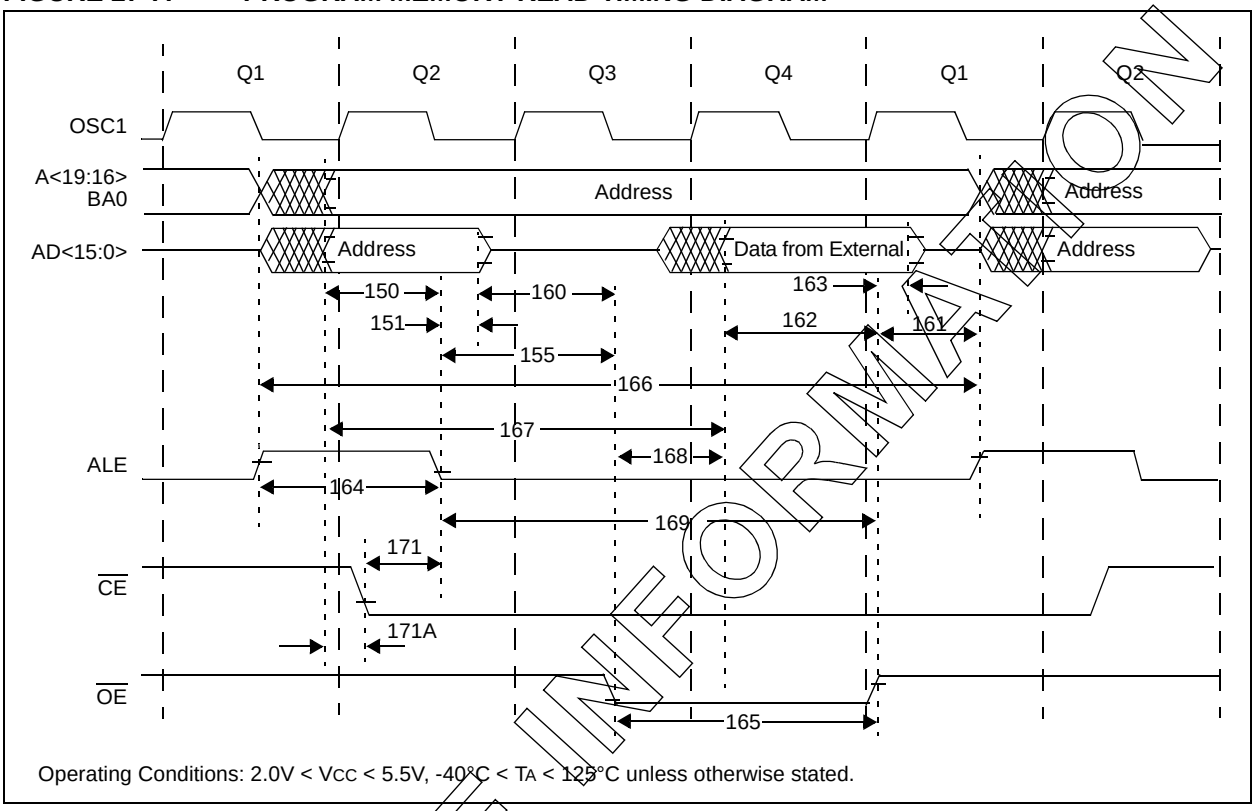


TABLE 27-9: CLKO AND I/O TIMING REQUIREMENTS

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address out valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to address out invalid (address hold time)	5	—	—	ns
155	TalL2oeL	ALE ↓ to $\overline{OE}$ ↓	10	0.125 TCY	—	ns
160	TadZ2oeL	AD high-Z to $\overline{OE}$ ↓ (bus release to $\overline{OE}$)	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ to AD driven	0.125 TCY – 5	—	—	ns
162	TadV2oeH	LS Data valid before $\overline{OE}$ ↑ (data setup time)	20	—	—	ns
163	ToeH2adl	$\overline{OE}$ ↑ to data in invalid (data hold time)	0	—	—	ns
164	TalH2alL	ALE pulse width	—	TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ pulse width	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	0.25 TCY	—	ns
167	Tacc	Address valid to data valid	0.75 TCY – 25	—	—	ns
168	Toe	$\overline{OE}$ ↓ to data valid		—	0.5 TCY – 25	ns
169	TalL2oeH	ALE ↓ to $\overline{OE}$ ↑	0.625 TCY – 10	—	0.625 TCY + 10	ns
171	TalH2csL	Chip Enable active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD valid to Chip Enable active	—	—	10	ns

FIGURE 27-8: PROGRAM MEMORY WRITE TIMING DIAGRAM

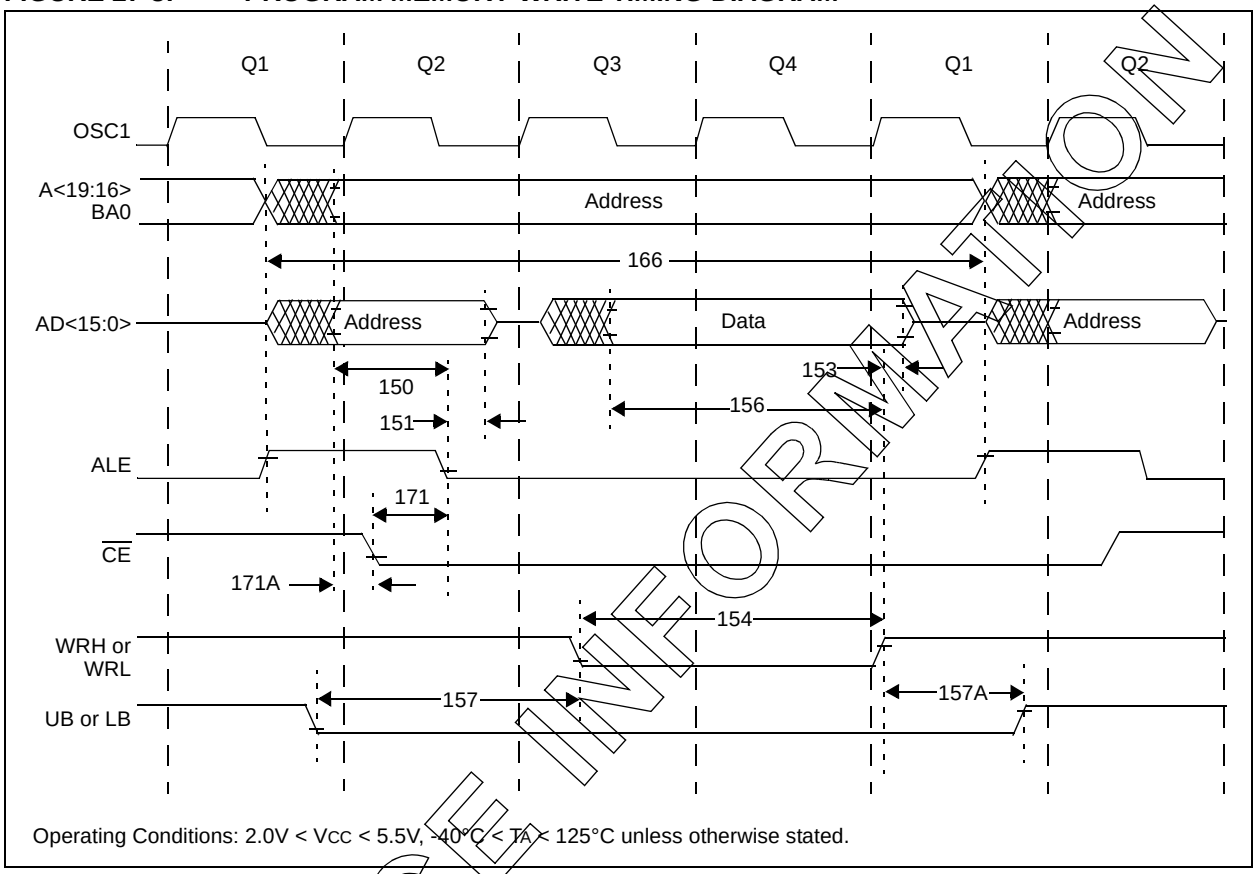


TABLE 27-10: PROGRAM MEMORY WRITE TIMING REQUIREMENTS

Param. No	Symbol	Characteristics	Min	Typ	Max	Units
150	TadV2alL	Address out valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adI	ALE ↓ to address out invalid (address hold time)	5	—	—	ns
153	TwrH2adI	WRn ↑ to data out invalid (data hold time)	5	—	—	ns
154	TwrL	WRn pulse width	0.5 TCY – 5	0.5 TCY	—	ns
156	TadV2wrH	Data valid before WRn ↑ (data setup time)	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	Byte select valid before WRn ↓ (byte select setup time)	0.25 TCY	—	—	ns
157A	TwrH2bsI	WRn ↑ to byte select invalid (byte select hold time)	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	0.25 TCY	—	ns
171	TalH2csL	Chip Enable active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD valid to Chip Enable active	—	—	10	ns

FIGURE 27-9: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

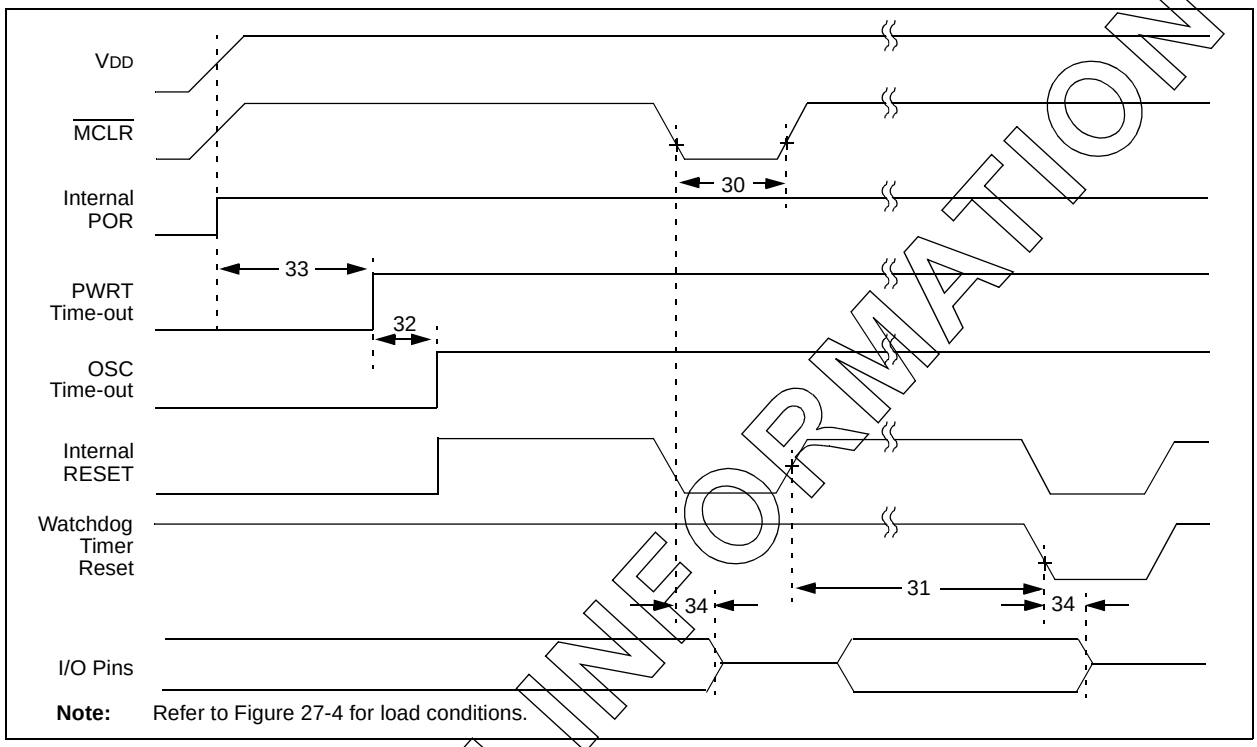


FIGURE 27-10: BROWN-OUT RESET TIMING

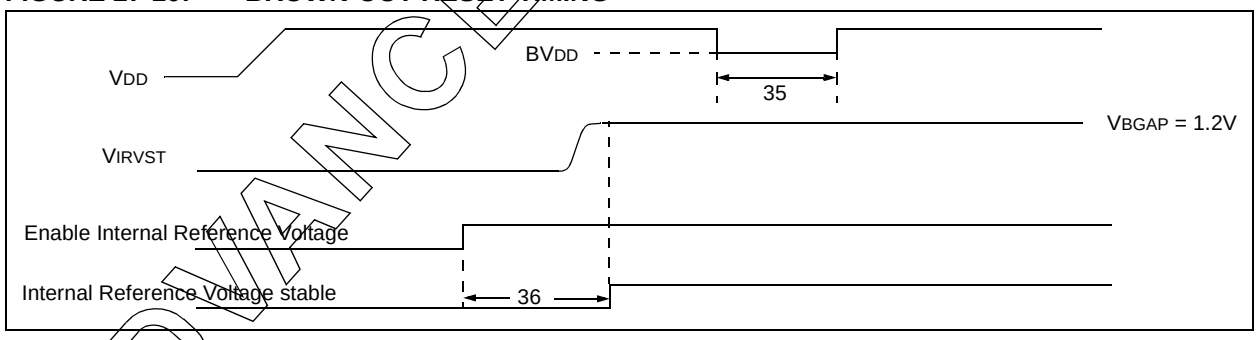
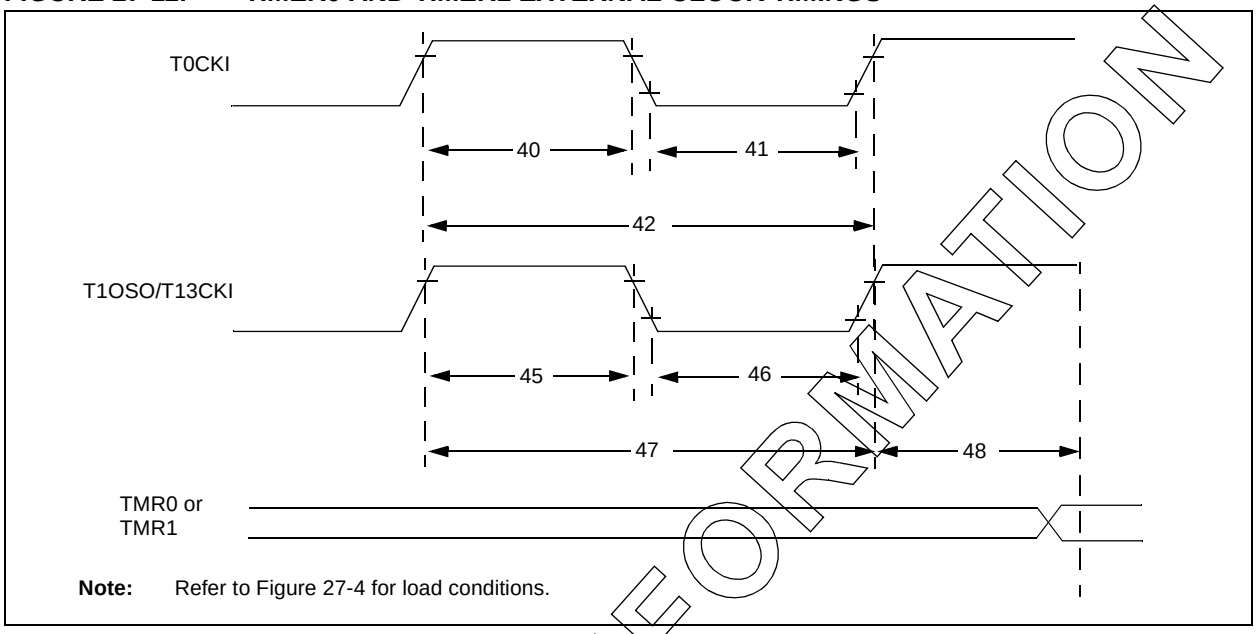


TABLE 27-11: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	TmcL	$\overline{MCLR}$ Pulse Width (low)	2	—	—	µs	
31	TWDT	Watchdog Timer Time-out Period (No Postscaler)	7	18	33	ms	
32	TOST	Oscillation Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	28	72	132	ms	
34	TIOZ	I/O high-impedance from $\overline{MCLR}$ Low or Watchdog Timer Reset	—	2	—	µs	
35	TBOR	Brown-out Reset Pulse Width	200	—	—	µs	VDD ≤ BVDD (see D005)
36	TIVRST	Time for Internal Reference Voltage to become stable	—	20	50	µs	
37	TLVD	Low Voltage Detect Pulse Width	200	—	—	µs	VDD ≤ VLVD

FIGURE 27-11: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

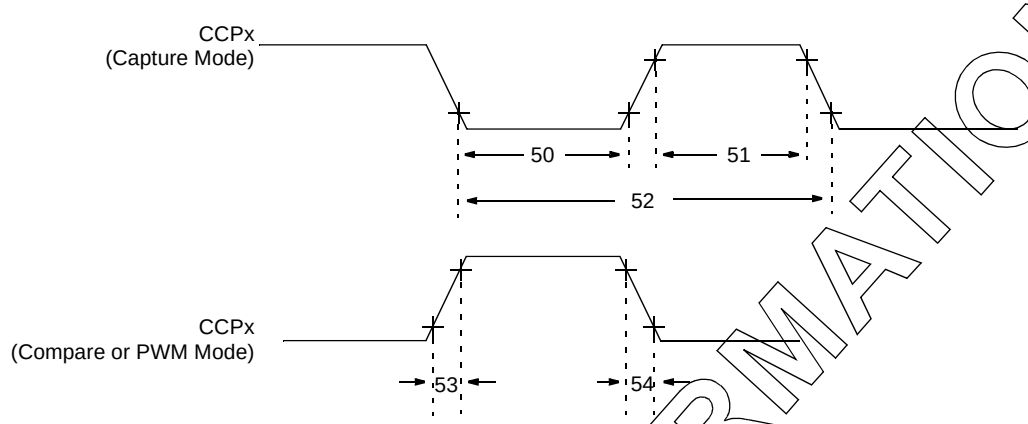


Note: Refer to Figure 27-4 for load conditions.

TABLE 27-12: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param. No.	Symbol	Characteristic			Min	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width		No Prescaler	0.5 TCY + 20	—	ns	
				With Prescaler	10	—	ns	
41	Tt0L	T0CKI Low Pulse Width		No Prescaler	0.5 TCY + 20	—	ns	
				With Prescaler	10	—	ns	
42	Tt0P	T0CKI Period		No Prescaler	TCY + 10	—	ns	
				With Prescaler	Greater of: 20 ns or TCY + 40 / N	—	ns	N = prescale value (1, 2, 4,..., 256)
45	Tt1H	T13CKI High Time	Synchronous, no prescaler		0.5 TCY + 20	—	ns	
			Synchronous, with prescaler	PIC18F6X2X/8X2X	10	—	ns	
				PIC18LF6X2X/8X2X	25	—	ns	
			Asynchronous	PIC18F6X2X/8X2X	30	—	ns	
				PIC18LF6X2X/8X2X	50	—	ns	
46	Tt1L	T13CKI Low Time	Synchronous, no prescaler		0.5 TCY + 5	—	ns	
			Synchronous, with prescaler	PIC18F6X2X/8X2X	10	—	ns	
				PIC18LF6X2X/8X2X	25	—	ns	
			Asynchronous	PIC18F6X2X/8X2X	30	—	ns	
				PIC18LF6X2X/8X2X	TBD	TBD	ns	
47	Tt1P	T13CKI Input Period	Synchronous		Greater of: 20 ns or TCY + 40 / N	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous		60	—	ns	
	Ft1	T13CKI Oscillator Input Frequency Range			DC	50	kHz	
48	Tcke2tmrI	Delay from External T13CKI Clock Edge to Timer Increment			2 TOSC	7 TOSC	—	

FIGURE 27-12: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)

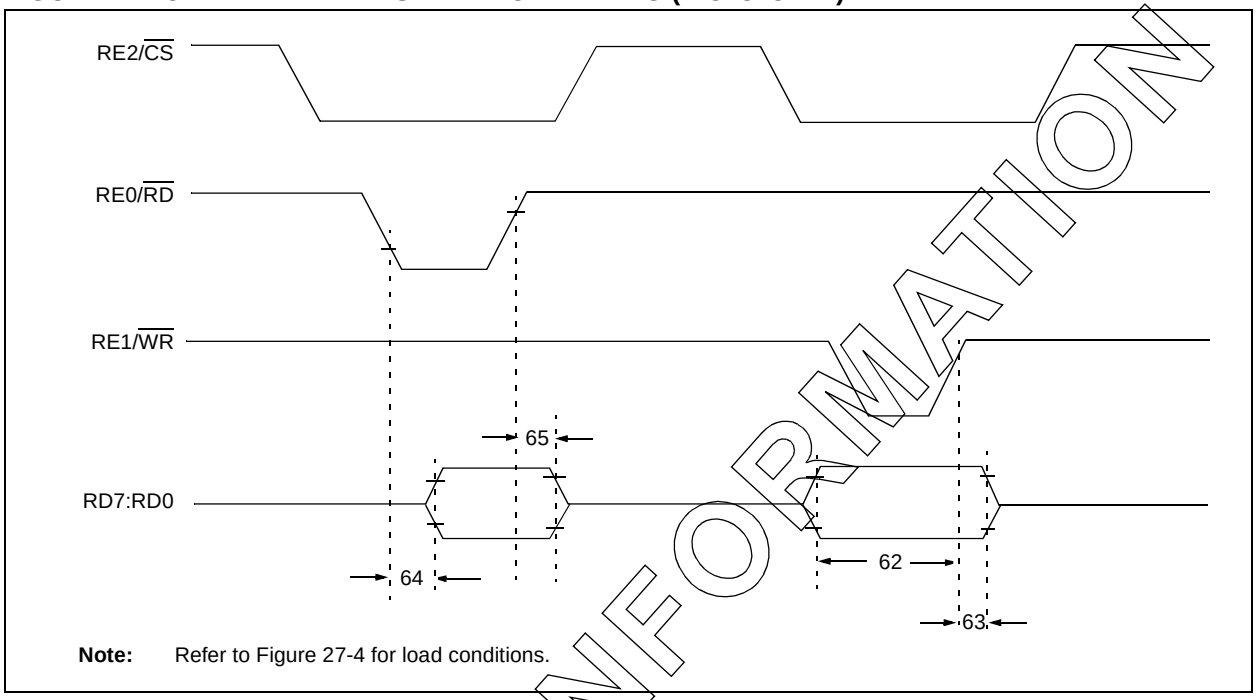


Note: Refer to Figure 27-4 for load conditions.

TABLE 27-13: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)

Param. No.	Symbol	Characteristic			Min	Max	Units	Conditions
50	TccL	CCPx Input Low Time	No Prescaler		0.5 TCY + 20	—	ns	
			With Prescaler	PIC18F6X2X/8X2X	10	—	ns	
				PIC18LF6X2X/8X2X	20	—	ns	
51	TccH	CCPx Input High Time	No Prescaler		0.5 TCY + 20	—	ns	
			With Prescaler	PIC18F6X2X/8X2X	10	—	ns	
				PIC18LF6X2X/8X2X	20	—	ns	
52	TccP	CCPx Input Period			$\frac{3\ T_{CY} + 40}{N}$	—	ns	N = prescale value (1,4 or 16)
53	TccR	CCPx Output Rise Time		PIC18F6X2X/8X2X	—	25	ns	
				PIC18LF6X2X/8X2X	—	45	ns	
54	TccF	CCPx Output Fall Time		PIC18F6X2X/8X2X	—	25	ns	
				PIC18LF6X2X/8X2X	—	45	ns	

FIGURE 27-13: PARALLEL SLAVE PORT TIMING (PIC18F8X2X)

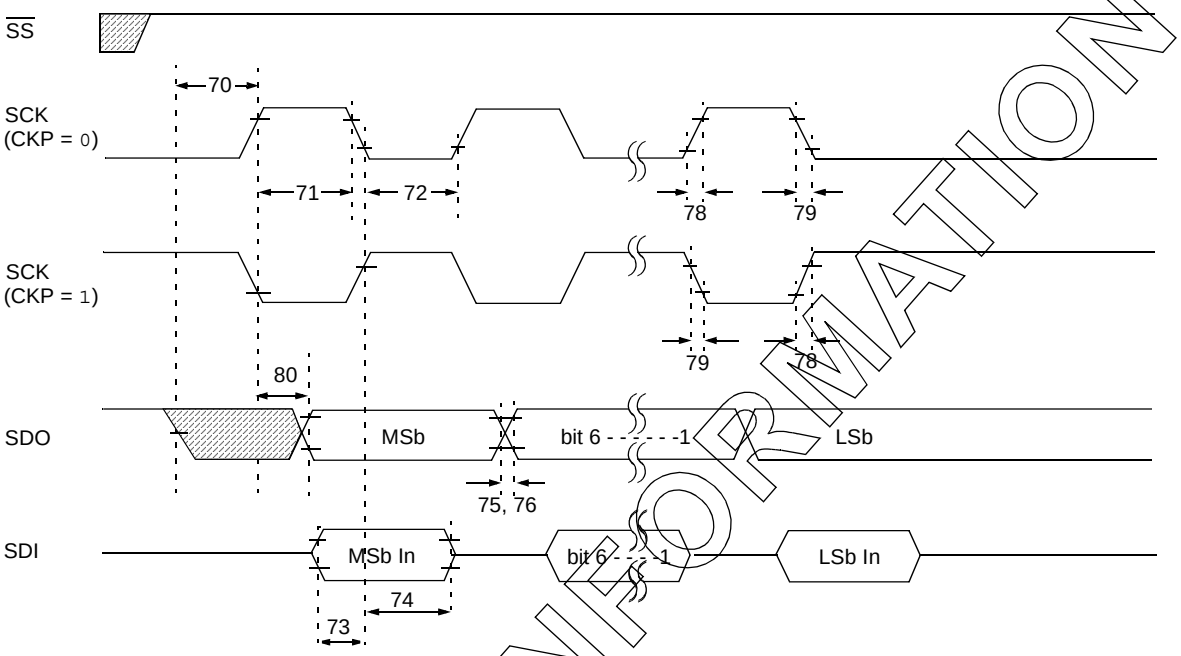


Note: Refer to Figure 27-4 for load conditions.

TABLE 27-14: PARALLEL SLAVE PORT REQUIREMENTS (PIC18F8X2X)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
62	TdtV2wrH	Data in valid before WR ↑ or CS ↑ (setup time)		20 25	— —	ns ns	 Extended Temp. range
63	TwrH2dtI	WR ↑ or CS ↑ to data–in invalid (hold time)	PIC18F6X2X/8X2X	20	—	ns	
			PIC18LF6X2X/8X2X	35	—	ns	
64	TrdL2dtV	RD ↓ and CS ↓ to data–out valid		— —	80 90	ns ns	 Extended Temp. range
65	TrdH2dtI	RD ↑ or CS ↓ to data–out invalid		10	30	ns	
66	TibfINH	Inhibit of the IBF flag bit being cleared from WR ↑ or CS ↑		—	3 TCY		

FIGURE 27-14: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)



Note: Refer to Figure 27-4 for load conditions.

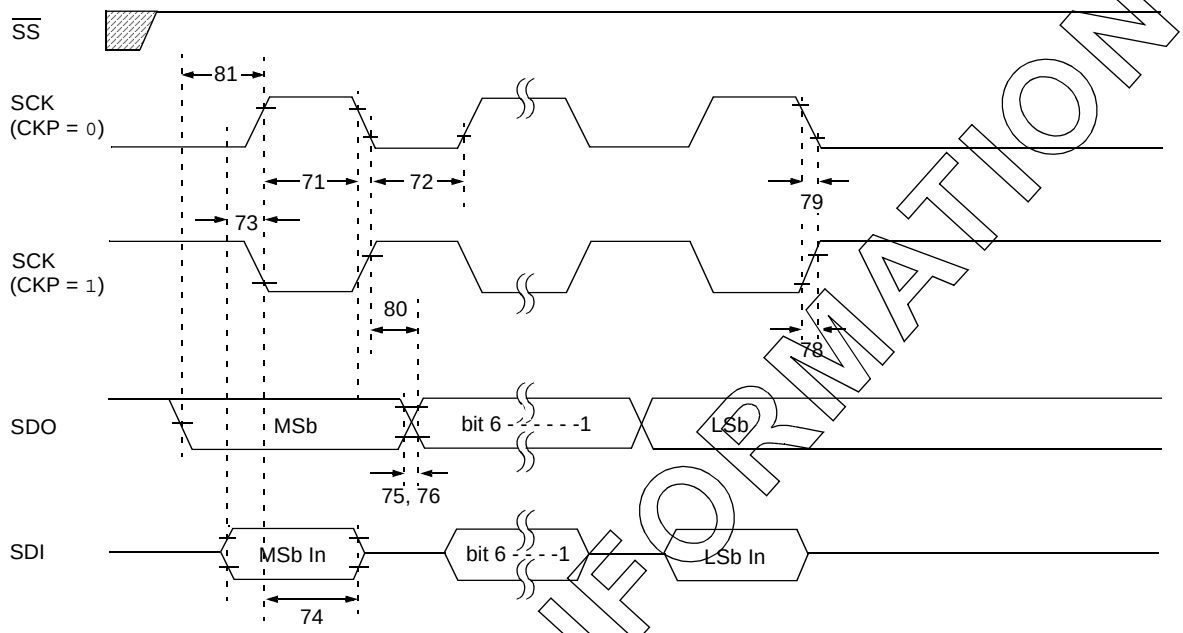
TABLE 27-15: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ input		TCY	—	ns	
71	TscH	SCK input high time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		100	—	ns	
73A	TB2B	Last clock edge of Byte 1 to the 1st clock edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X	—	45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
78	TscR	SCK output rise time (Master mode)	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X	—	45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18F6X2X/8X2X	—	50	ns	
			PIC18LF6X2X/8X2X	—	100	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 27-15: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)



Note: Refer to Figure 27-4 for load conditions.

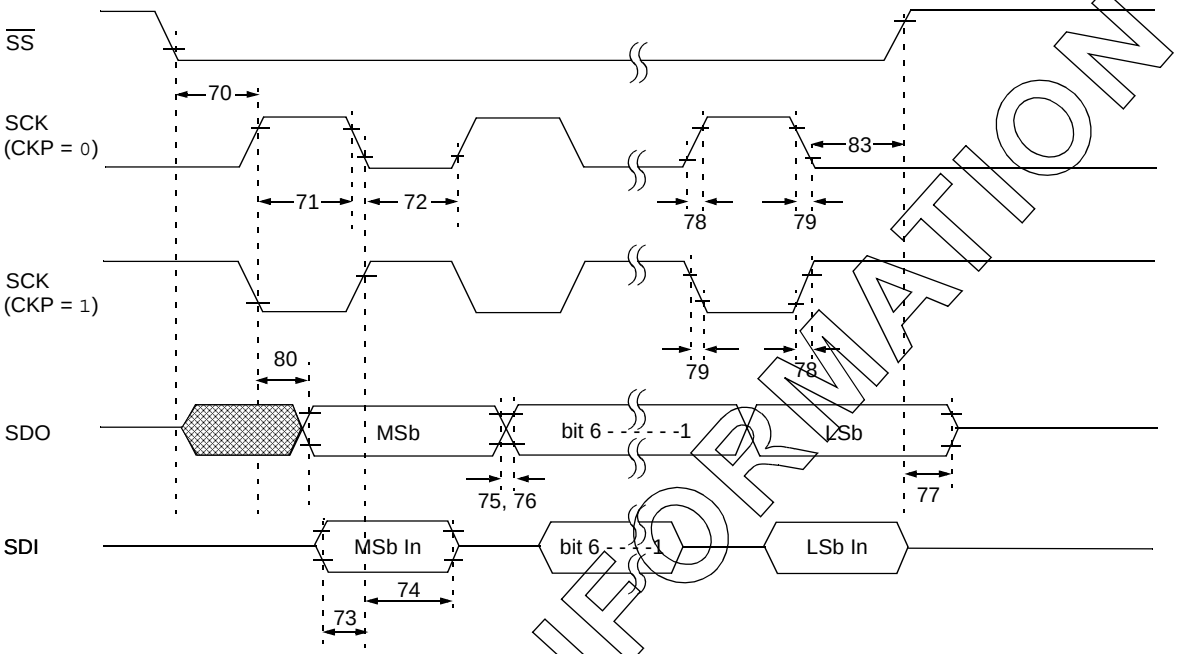
TABLE 27-16: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
71	TscH	SCK input high time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		100	—	ns	
73A	TB2B	Last clock edge of Byte 1 to the 1st clock edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X		45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
78	TscR	SCK output rise time (Master mode)	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X		45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18F6X2X/8X2X	—	50	ns	
			PIC18LF6X2X/8X2X		100	ns	
81	TdoV2scH, TdoV2scL	SDO data output setup to SCK edge		TCY	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 27-16: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)



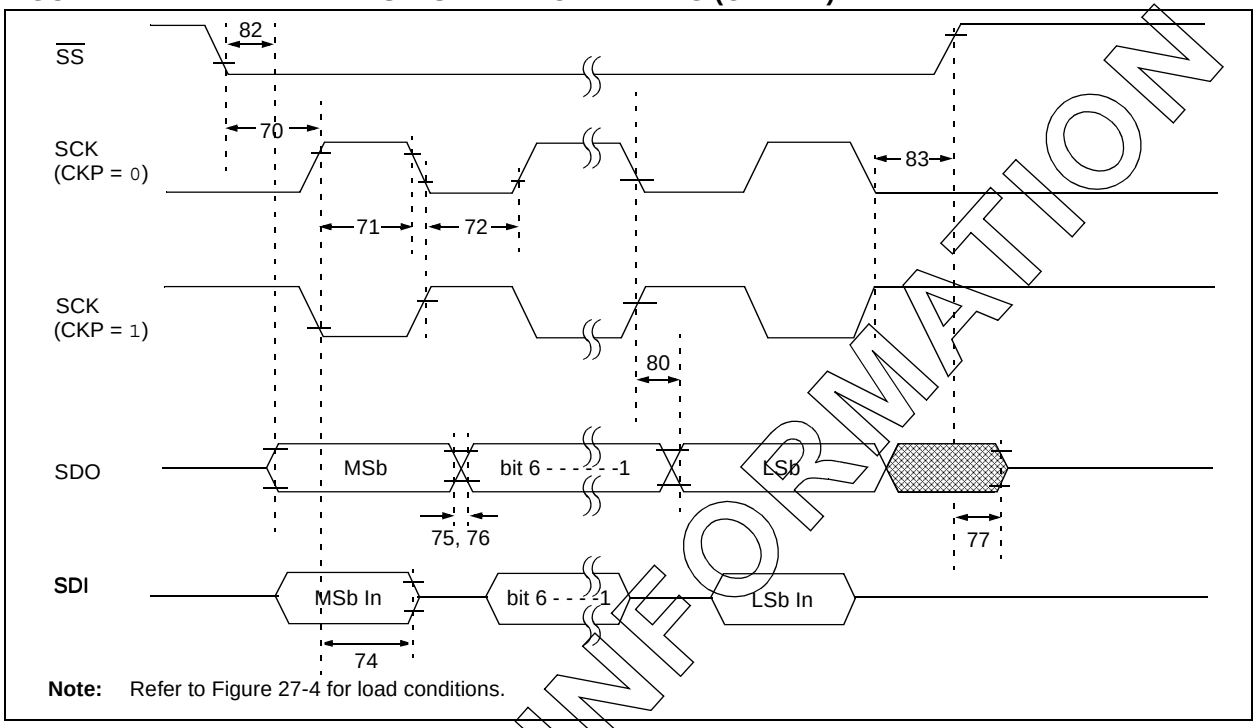
Note: Refer to Figure 27-4 for load conditions.

TABLE 27-17: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ input		TCY	—	ns	
71	TscH	SCK input high time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		100	—	ns	
73A	TB2B	Last clock edge of Byte 1 to the first clock edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18F6X2X/8X2X	—	25	ns	
			PIC18F6X2X/8X2X		45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO output hi-impedance		10	50	ns	
78	TscR	SCK output rise time (Master mode)	PIC18F6X2X/8X2X	—	25	ns	
			PIC18F6X2X/8X2X		45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18F6X2X/8X2X	—	50	ns	
			PIC18F6X2X/8X2X		100	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ after SCK edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter #73A.
2: Only if Parameter #71A and #72A are used.

FIGURE 27-17: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)



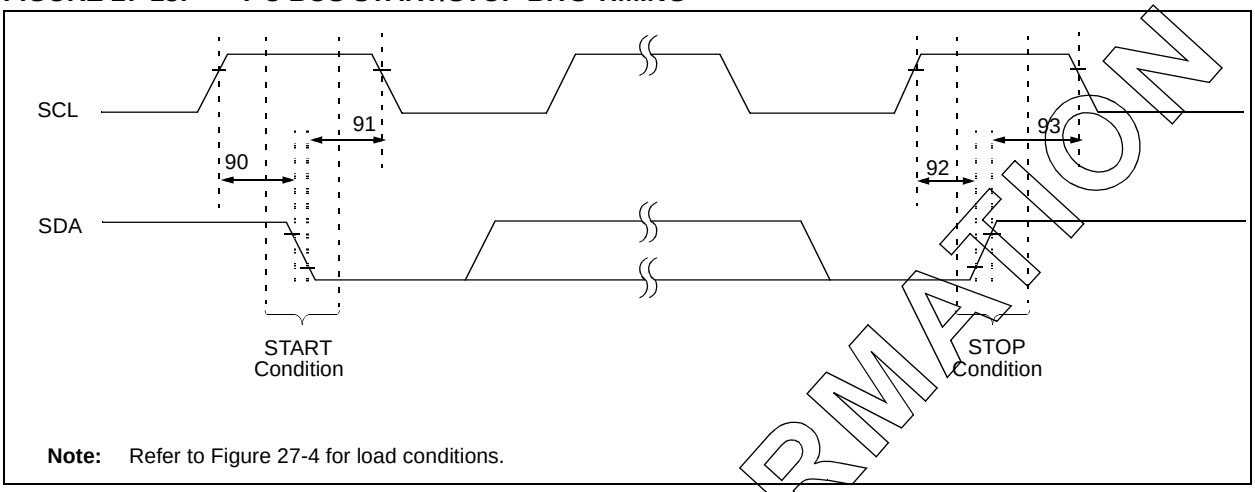
Note: Refer to Figure 27-4 for load conditions.

TABLE 27-18: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ input		TCY	—	ns	
71	TscH	SCK input high time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK input low time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73A	TB2B	Last clock edge of Byte 1 to the first clock edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	ns	
75	TdoR	SDO data output rise time	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X		45	ns	
76	TdoF	SDO data output fall time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO output hi-impedance		10	50	ns	
78	TscR	SCK output rise time (Master mode)	PIC18F6X2X/8X2X	—	25	ns	
			PIC18LF6X2X/8X2X	—	45	ns	
79	TscF	SCK output fall time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge	PIC18F6X2X/8X2X	—	50	ns	
			PIC18LF6X2X/8X2X	—	100	ns	
82	TssL2doV	SDO data output valid after $\overline{SS}$ ↓ edge	PIC18F6X2X/8X2X	—	50	ns	
			PIC18LF6X2X/8X2X	—	100	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ after SCK edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter #73A.
2: Only if Parameter #71A and #72A are used.

FIGURE 27-18: I²C BUS START/STOP BITS TIMING

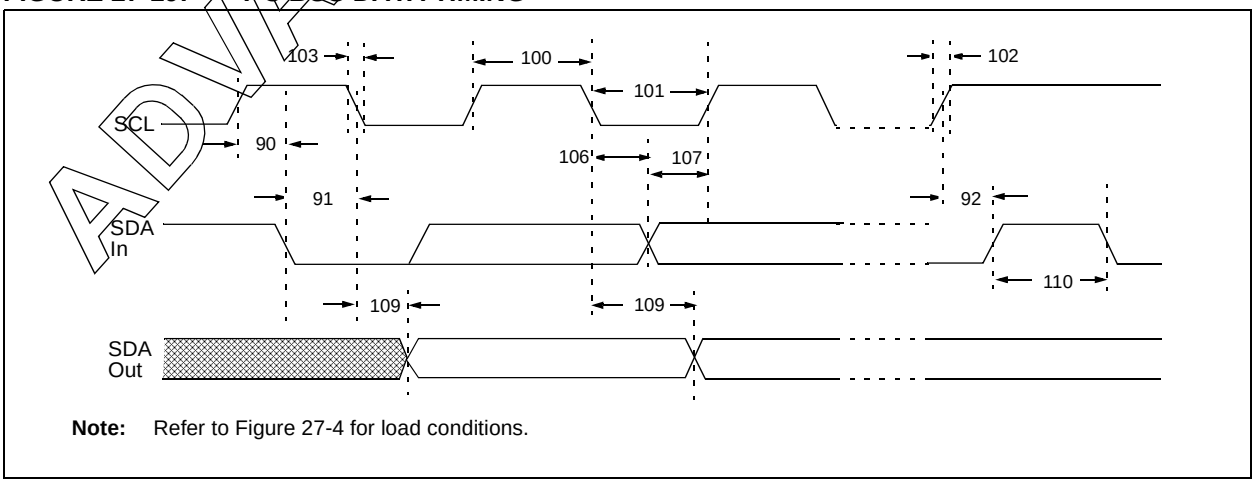


Note: Refer to Figure 27-4 for load conditions.

TABLE 27-19: I²C BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	START condition	100 kHz mode	4700	—	ns	Only relevant for Repeated START condition
		Setup time	400 kHz mode	600	—		
91	THD:STA	START condition	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
		Hold time	400 kHz mode	600	—		
92	TSU:STO	STOP condition	100 kHz mode	4700	—	ns	
		Setup time	400 kHz mode	600	—		
93	THD:STO	STOP condition	100 kHz mode	4000	—	ns	
		Hold time	400 kHz mode	600	—		

FIGURE 27-19: I²C BUS DATA TIMING



Note: Refer to Figure 27-4 for load conditions.

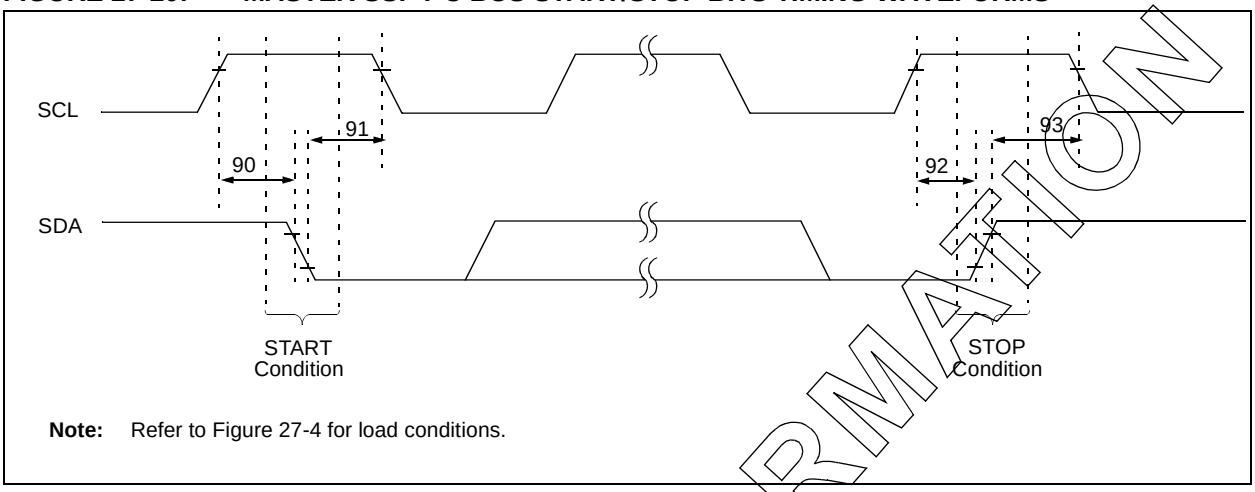
TABLE 27-20: I^2C BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock high time	100 kHz mode	4.0	—	µs	PIC18F6X2X/8X2X must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	µs	PIC18F6X2X/8X2X must operate at a minimum of 10 MHz
			SSP Module	1.5 TCY	—		
101	TLOW	Clock low time	100 kHz mode	4.7	—	µs	PIC18F6X2X/8X2X must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	µs	PIC18F6X2X/8X2X must operate at a minimum of 10 MHz
			SSP Module	1.5 TCY	—		
102	TR	SDA and SCL rise time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
103	TF	SDA and SCL fall time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
90	TSU:STA	START condition setup time	100 kHz mode	4.7	—	µs	Only relevant for Repeated START condition
			400 kHz mode	0.6	—	µs	
91	THD:STA	START condition hold time	100 kHz mode	4.0	—	µs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	µs	
106	THD:DAT	Data input hold time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	µs	
107	TSU:DAT	Data input setup time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
92	TSU:STO	STOP condition setup time	100 kHz mode	4.7	—	µs	
			400 kHz mode	0.6	—	µs	
109	TAA	Output valid from clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110	TBUF	Bus free time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	µs	
D102	CB	Bus capacitive loading		—	400	pF	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of START or STOP conditions.

2: A fast mode I^2C bus device can be used in a standard mode I^2C bus system but the requirement $T_{SU:DAT} \geq 250$ ns must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line.
T_R max. + $T_{SU:DAT}$ = 1000 + 250 = 1250 ns (according to the standard mode I^2C bus specification) before the SCL line is released.

FIGURE 27-20: MASTER SSP I²C BUS START/STOP BITS TIMING WAVEFORMS



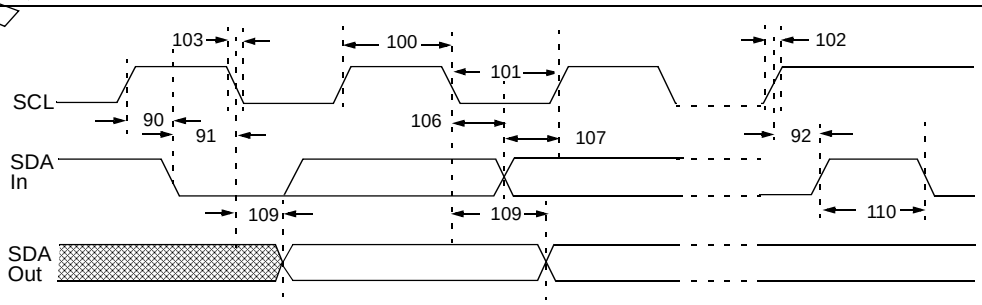
Note: Refer to Figure 27-4 for load conditions.

TABLE 27-21: MASTER SSP I²C BUS START/STOP BITS REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	START condition Setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	Only relevant for Repeated START condition
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	START condition Hold time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	STOP condition Setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	STOP condition Hold time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		

Note 1: Maximum pin capacitance = 10 pF for all I²C pins.

FIGURE 27-21: MASTER SSP I²C BUS DATA TIMING



Note: Refer to Figure 27-4 for load conditions.

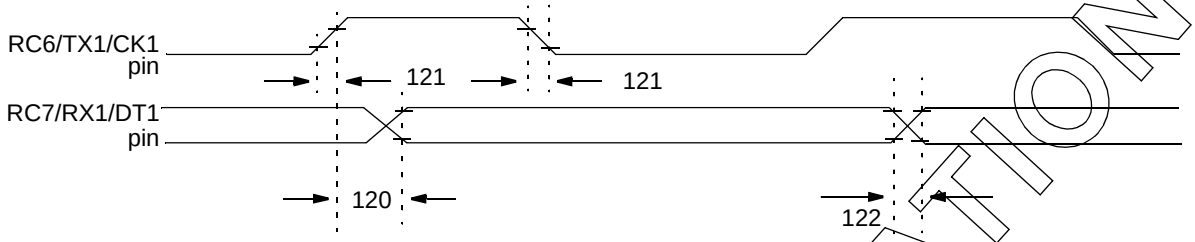
TABLE 27-22: MASTER SSP I²C BUS DATA REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock high time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	Clock low time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDA and SCL rise time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	300	ns	
103	TF	SDA and SCL fall time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	100	ns	
90	TSU:STA	START condition setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	Only relevant for Repeated START condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	START condition hold time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	Data input hold time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	ms	
			1 MHz mode(1)	TBD	—	ns	
107	TSU:DAT	Data input setup time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	TBD	—	ns	
92	TSU:STO	STOP condition setup time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	ms	
109	TAA	Output valid from clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	—	ns	
110	TBUF	Bus free time	100 kHz mode	4.7	—	ms	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	ms	
			1 MHz mode(1)	TBD	—	ms	
D102	CB	Bus capacitive loading		—	400	pF	

Note 1: Maximum pin capacitance = 10 pF for all I²C pins.

2: A fast mode I²C bus device can be used in a standard mode I²C bus system but parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line, parameter #102.+ parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCL line is released.

FIGURE 27-22: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

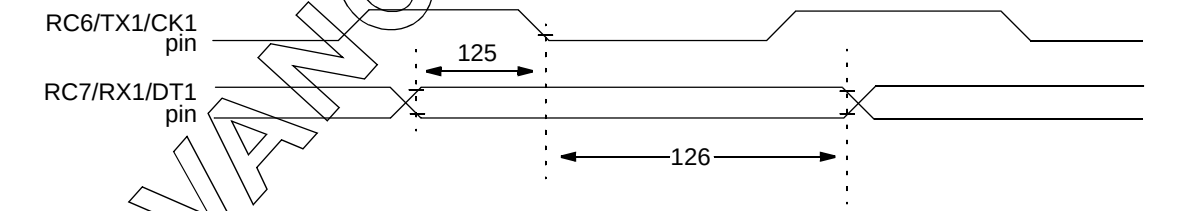


Note: Refer to Figure 27-4 for load conditions.

TABLE 27-23: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) Clock high to data out valid	PIC18F6X2X/8X2X	—	40	ns	
			PIC18LF6X2X/8X2X	—	100	ns	
121	Tckrf	Clock out rise time and fall time (Master mode)	PIC18F6X2X/8X2X	—	20	ns	
			PIC18LF6X2X/8X2X	—	50	ns	
122	Tdtrf	Data out rise time and fall time	PIC18F6X2X/8X2X	—	20	ns	
			PIC18LF6X2X/8X2X	—	50	ns	

FIGURE 27-23: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 27-4 for load conditions.

TABLE 27-24: USART SYNCHRONOUS RECEIVE REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
125	TdtV2ckl	SYNC RCV (MASTER & SLAVE) Data hold before CK ↓ (DT hold time)	10	—	ns	
126	TckL2dtl	Data hold after CK ↓ (DT hold time)	15	—	ns	

TABLE 27-25: A/D CONVERTER CHARACTERISTICS: PIC18F6X2X/8X2X (INDUSTRIAL, EXTENDED) PIC18LF6X2X/8X2X (INDUSTRIAL)

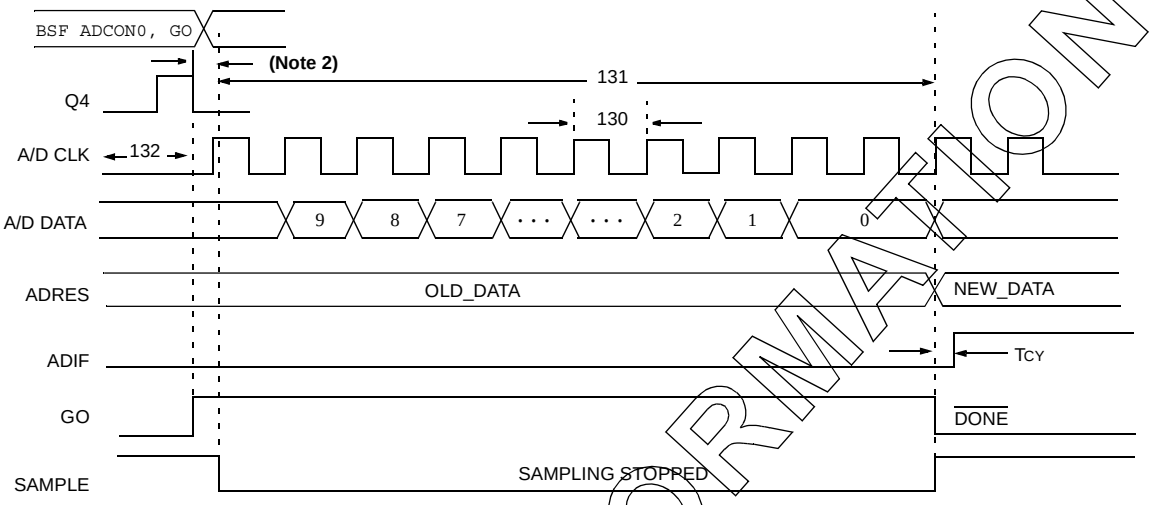
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
A01	NR	Resolution		—	—	10	bit	VREF = VDD ≥ 3.0V
				—	—	TBD	bit	VREF = VDD < 3.0V
A03	EIL	Integral linearity error		—	—	<±1	LSb	VREF = VDD ≥ 3.0V
				—	—	TBD	LSb	VREF = VDD < 3.0V
A04	EDL	Differential linearity error		—	—	<±1	LSb	VREF = VDD ≥ 3.0V
				—	—	TBD	LSb	VREF = VDD < 3.0V
A05	EFS	Full scale error		—	—	<±1	LSb	VREF = VDD ≥ 3.0V
				—	—	TBD	LSb	VREF = VDD < 3.0V
A06	EOFF	Offset error		—	—	<±1	LSb	VREF = VDD ≥ 3.0V
				—	—	TBD	LSb	VREF = VDD < 3.0V
A10	—	Monotonicity		guaranteed(3)			—	VSS ≤ VAIN ≤ VREF
A20	VREF	Reference voltage (VREFH – VREFL)		0V	—	—	V	
A20A				3V	—	—	V	For 10-bit resolution
A21	VREFH	Reference voltage High		AVSS	—	AVDD + 0.3V	V	
A22	VREFL	Reference voltage Low		AVSS – 0.3V	—	AVDD	V	
A25	VAIN	Analog input voltage		AVSS – 0.3V	—	VREF + 0.3V	V	
A30	ZAIN	Recommended impedance of analog voltage source		—	—	10.0	kΩ	
A40	IAD	A/D conversion current (VDD)	PIC18F6X2X/8X2X	—	180	—	μA	Average current consumption when A/D is on **(Note 1)**
			PIC18LF6X2X/8X2X	—	90	—	μA	
A50	IREF	VREF input current **(Note 2)**		10	—	1000	μA	During VAIN acquisition. Based on differential of VHOLD to VAIN. To charge CHOLD, see Section 20.0.
				—	—	10	μA	During A/D conversion cycle.

Note 1: When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.
VREF current is from RA2/AN2/VREF- and RA3/AN3/VREF+ pins or AVDD and AVSS pins, whichever is selected as reference input.

2: VSS ≤ VAIN ≤ VREF

3: The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

FIGURE 27-24: A/D CONVERSION TIMING



Note 1: If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

2: This is a minimal RC delay (typically 100 ns) which also disconnects the holding capacitor from the analog input.

TABLE 27-26: A/D CONVERSION REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
130	TAD	A/D clock period	PIC18F6X2X/8X2X	1.6	20(5)	µs	TOSC based, VREF ≥ 3.0V
			PIC18LF6X2X/8X2X	3.0	20(5)	µs	TOSC based, VREF full range
			PIC18F6X2X/8X2X	2.0	6.0	µs	A/D RC mode
			PIC18LF6X2X/8X2X	3.0	9.0	µs	A/D RC mode
131	TCNV	Conversion time (not including acquisition time) **(Note 1)**		11	12	TAD	
132	TACQ	Acquisition time **(Note 3)**		15 10	— —	µs µs	-40°C ≤ Temp ≤ 125°C 0°C ≤ Temp ≤ 125°C
135	TSWC	Switching time from convert → sample		—	**(Note 4)**		
136	TAMP	Amplifier settling time **(Note 2)**		1	—	µs	This may be used if the "new" input voltage has not changed by more than 1 LSb (i.e., 5 mV @ 5.12V) from the last sampled voltage (as stated on CHOLD).

Note 1: ADRES register may be read on the following TCY cycle.

2: See Section 20.0 for minimum conditions when input voltage has changed more than 1 LSb.

3: The time for the holding capacitor to acquire the "New" input voltage when the voltage changes full scale after the conversion (AVDD to AVSS, or AVSS to AVDD). The source impedance (RS) on the input channels is 50Ω.

4: On the next Q4 cycle of the device clock.

5: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

NOTES:

28.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

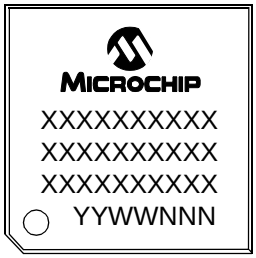
Graphs are not available at this time.

NOTES:

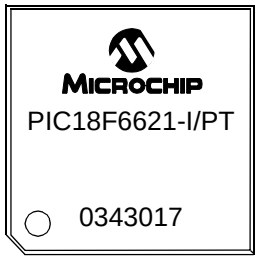
29.0 PACKAGING INFORMATION

29.1 Package Marking Information

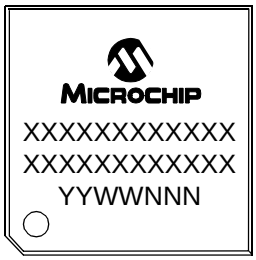
64-Lead TQFP



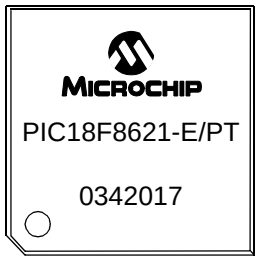
Example



80-Lead TQFP



Example



Legend:	XX...X	Customer specific information*
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code

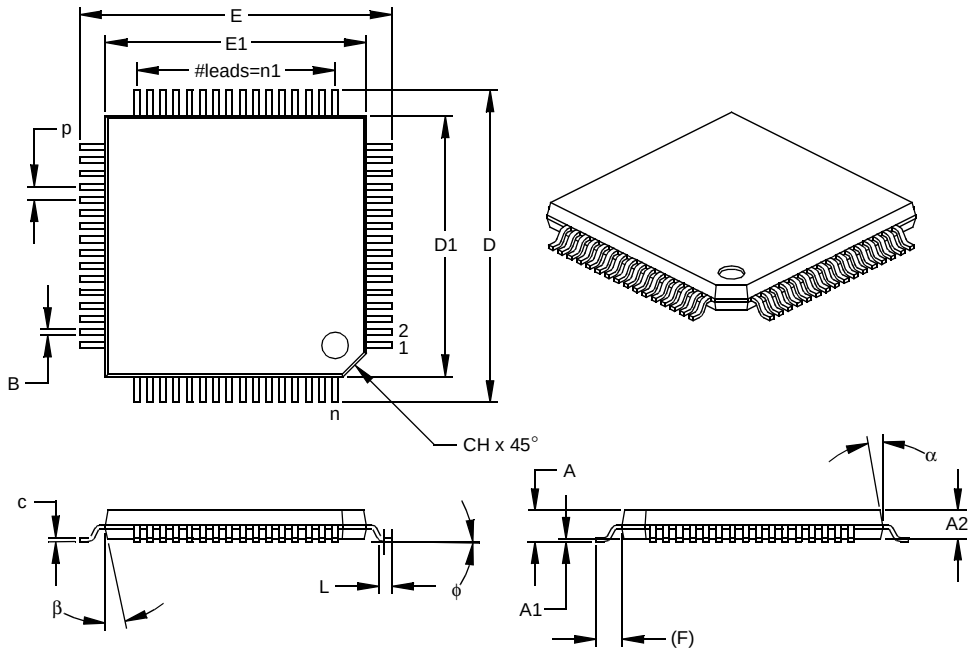
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

***** Standard PICmicro device marking consists of Microchip part number, year code, week code, and traceability code. For PICmicro device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

29.2 Package Details

The following sections give the technical details of the packages.

64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)

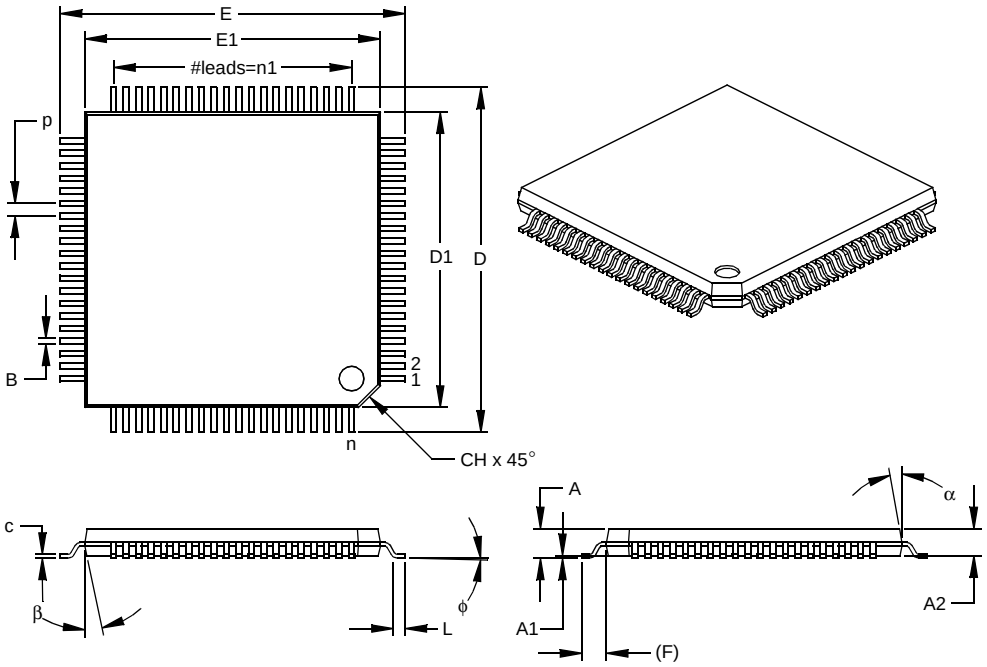


Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		64			64	
Pitch	p		.020			0.50	
Pins per Side	n1		16			16	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002	.006	.010	0.05	0.15	0.25
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039			1.00	
Foot Angle	φ	0	3.5	7	0	3.5	7
Overall Width	E	.463	.472	.482	11.75	12.00	12.25
Overall Length	D	.463	.472	.482	11.75	12.00	12.25
Molded Package Width	E1	.390	.394	.398	9.90	10.00	10.10
Molded Package Length	D1	.390	.394	.398	9.90	10.00	10.10
Lead Thickness	c	.005	.007	.009	0.13	0.18	0.23
Lead Width	B	.007	.009	.011	0.17	0.22	0.27
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-026
Drawing No. C04-085

80-Lead Plastic Thin Quad Flatpack (PT) 12x12x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		80			80	
Pitch	p		.020			0.50	
Pins per Side	n1		20			20	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002	.004	.006	0.05	0.10	0.15
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039			1.00	
Foot Angle	ϕ	0	3.5	7	0	3.5	7
Overall Width	E	.541	.551	.561	13.75	14.00	14.25
Overall Length	D	.541	.551	.561	13.75	14.00	14.25
Molded Package Width	E1	.463	.472	.482	11.75	12.00	12.25
Molded Package Length	D1	.463	.472	.482	11.75	12.00	12.25
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.007	.009	.011	0.17	0.22	0.27
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-026
Drawing No. C04-092

NOTES:

APPENDIX A: REVISION HISTORY

Revision A (July 2003)

Original data sheet for PIC18F6X2X/8X2X family.

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

TABLE B-1: DEVICE DIFFERENCES

Feature	PIC18F6525	PIC18F6621	PIC18F8525	PIC18F8621
On-chip Program Memory (Kbytes)	48K	64K	48K	64K
I/O Ports	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
A/D Channels	12	12	16	16
External Memory Interface	No	No	Yes	Yes
Package Types	64-pin TQFP	64-pin TQFP	80-pin TQFP	80-pin TQFP

APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC17C756 to a PIC18F8720.

Not Applicable

APPENDIX D: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in *AN716, "Migrating Designs from PIC16C74A/74B to PIC18C442*." The changes discussed, while device specific, are generally applicable to all mid-range to enhanced device migrations.

This Application Note is available as Literature Number DS00716.

APPENDIX E: MIGRATION FROM HIGH-END TO ENHANCED DEVICES

A detailed discussion of the migration pathway and differences between the high-end MCU devices (i.e., PIC17CXXX) and the enhanced devices (i.e., PIC18FXXXX) is provided in *AN726, "PIC17CXXX to PIC18CXXX Migration*." This Application Note is available as Literature Number DS00726.

NOTES:

INDEX

A

B

C

D

E

F

G

H

I

Q

R

S

T

U

V

W

X

ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® or Microsoft® Internet Explorer. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

www.microchip.com

The file transfer site is available by using an FTP service to connect to:

ftp://ftp.microchip.com

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

SYSTEMS INFORMATION AND UPGRADE HOT LINE

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits. The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

042003

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC18F6X2X/8X2X Literature Number: DS39612A

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?

PIC18F6X2X/8X2X PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	–	X	/XX	XXX
Device		Temperature Range	Package	Pattern

Device	PIC18F6X2X/8X2X(1), PIC18F6X2X/8X2XT(2); VDD range 4.2V to 5.5V PIC18LF6X2X/8X2X(1), PIC18LF6X2X/8X2XT(2); VDD range 2.0V to 5.5V
Temperature Range	I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)
Package	PT = TQFP (Thin Quad Flatpack)
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)

Examples:

a) PIC18LF6621 - I/PT 301 = Industrial temp., TQFP package, Extended VDD limits, QTP pattern #301.

b) PIC18F8621 - E/PT = Extended temp., TQFP package, standard VDD limits.

Note 1: F = Standard Voltage Range
LF = Extended Voltage Range

2: T = in tape and reel

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Atlanta
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034
Fax: 770-640-0307

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848
Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, IN 46902
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888
Fax: 949-263-1338

Phoenix
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966
Fax: 480-792-4338

San Jose
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950
Fax: 408-436-7955

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100
Fax: 86-10-85282104

China - Chengdu
Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200
Fax: 86-28-86766599

China - Fuzhou
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506
Fax: 86-591-7503521

China - Hong Kong SAR
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen
Rm. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380
Fax: 86-755-8295-1393

China - Shunde
Room 401, Hongjian Building
No. 2 Fengxiangnan Road, Ronggui Town
Shunde City, Guangdong 528303, China
Tel: 86-765-8395507 Fax: 86-765-8395571

China - Qingdao
Rm. B505A, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

India
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan
Kaohsiung Branch
30F - 1 No. 8
Min Chuan 2nd Road
Kaohsiung 806, Taiwan
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan
Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Austria
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45-4420-9895 Fax: 45-4420-9910

France
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy
Via Quasimodo, 12
20025 Legnano (MI)
Milan, Italy
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands
P. A. De Biesbosch 14
NL-5152 SC Drunen, Netherlands
Tel: 31-416-690399
Fax: 31-416-690340

United Kingdom
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-118-921-5869
Fax: 44-118-921-5820

07/28/03

Table of Contents

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC18F6525-E/PT PIC18F6621T-I/PT PIC18LF8525T-I/PT PIC18F8525-I/PT PIC18LF8621T-I/PT PIC18F8621T-I/PT PIC18F6621-I/PT PIC18F8621-E/PT PIC18F8525T-I/PT PIC18LF6525-I/PT PIC18LF8525-I/PT PIC18F8621-I/PT PIC18F6525T-I/PT PIC18F6621-E/PT PIC18F8525-E/PT PIC18LF6525T-I/PT