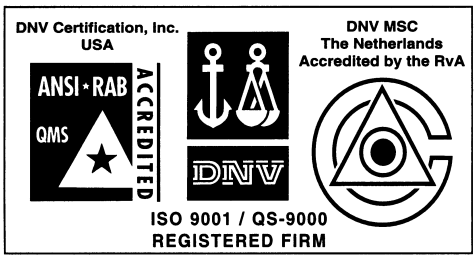
PIC16C925/926
Data Sheet

64/68-Pin CMOS Microcontrollers with LCD Driver

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. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

PIC16C925/926

64/68-Pin CMOS Microcontrollers with LCD Driver

High Performance RISC CPU:

- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14-bit words of EPROM program memory, 336 bytes general purpose registers (SRAM), 60 special function registers
- Pinout compatible with PIC16C923/924

Peripheral Features:

- 25 I/O pins with individual direction control and 25-27 input only pins
- Timer0 module: 8-bit timer/counter with programmable 8-bit prescaler
- Timer1 module: 16-bit timer/counter, can be incremented during SLEEP via external crystal/clock
- Timer2 module: 8-bit timer/counter with 8-bit period register, prescaler, and postscaler
- One Capture, Compare, PWM module
- Synchronous Serial Port (SSP) module with two modes of operation:
  - 3-wire SPI™ (supports all 4 SPI modes)
  - I^2C™ Slave mode
- Programmable LCD timing module:
  - Multiple LCD timing sources available
  - Can drive LCD panel while in SLEEP mode
  - Static, 1/2, 1/3, 1/4 multiplex
  - Static drive and 1/3 bias capability
  - 16 bytes of dedicated LCD RAM
  - Up to 32 segments, up to 4 commons

Common	Segment	Pixels
1	32	32
2	31	62
3	30	90
4	29	116

Analog Features:

- 10-bit 5-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)

Special Microcontroller Features:

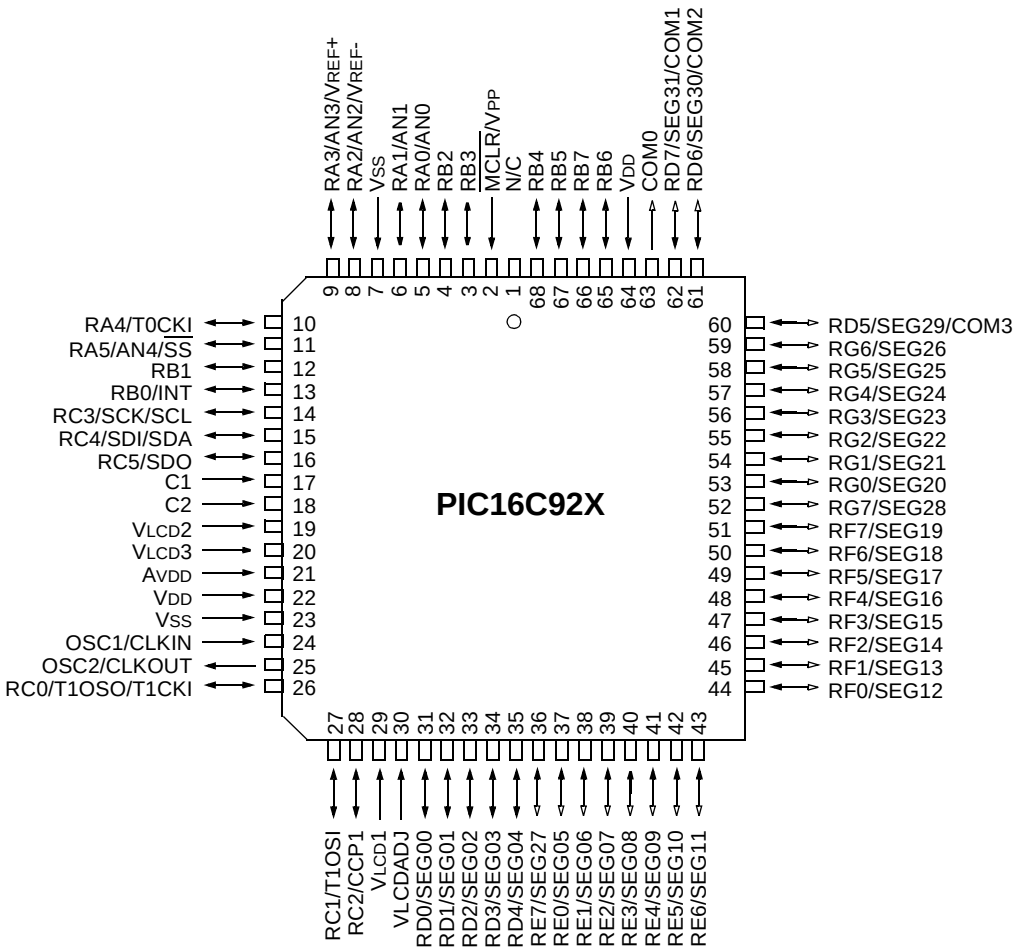
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Selectable oscillator options
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Processor read access to program memory

CMOS Technology:

- Low power, high speed CMOS/EPROM technology
- Fully static design
- Wide operating voltage range: 2.5V to 5.5V
- Commercial and Industrial temperature ranges
- Low power consumption

Pin Diagrams

PLCC, CLCC



LEGEND:

- Input Pin
- Output Pin
- Input/Output Pin
- Digital Input/LCD Output Pin
- LCD Output Pin

Pin Diagrams (Continued)

TQFP

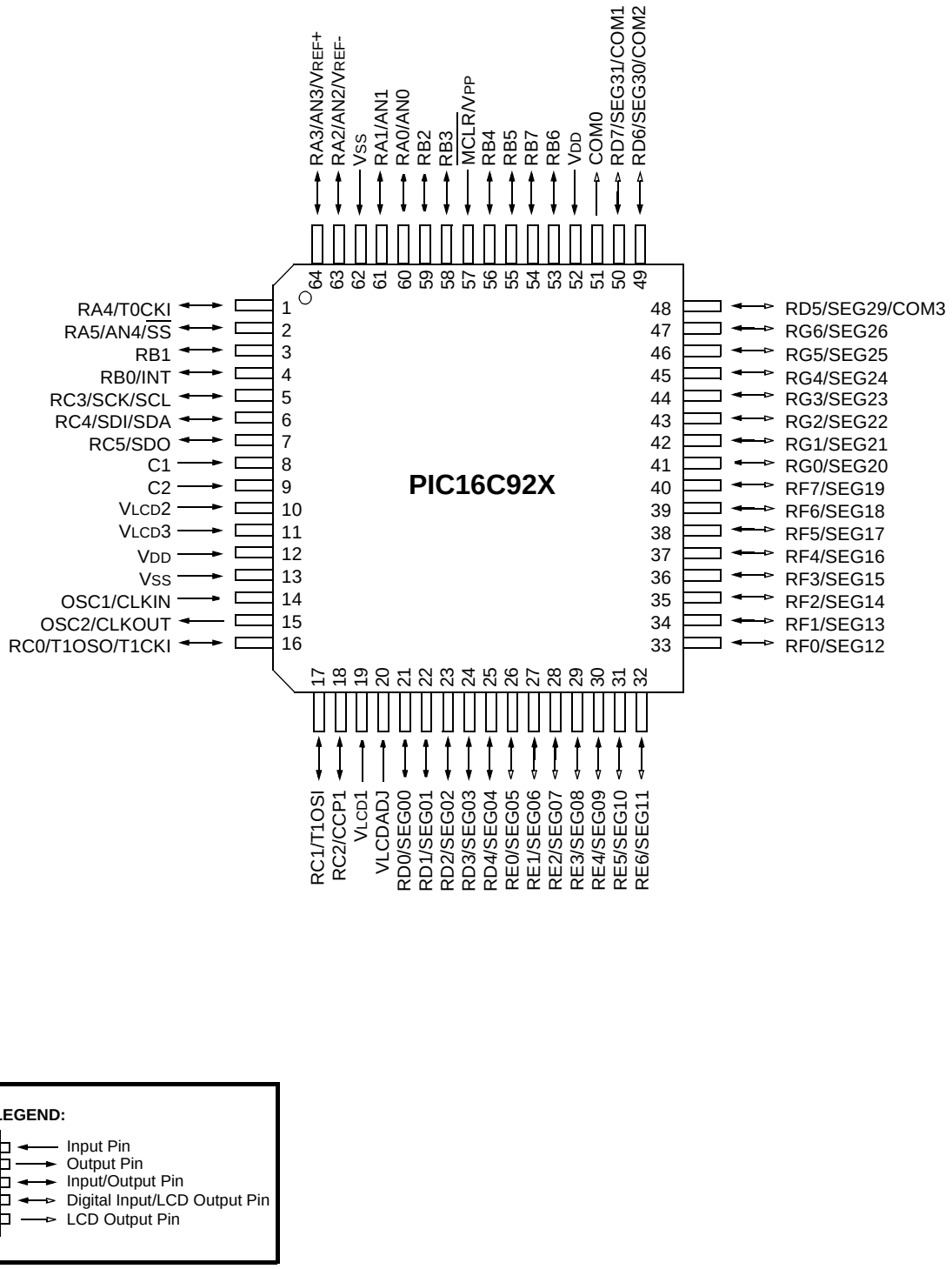


Table of Contents

1.0 Device Overview 5
2.0 Memory Organization 11
3.0 Reading Program Memory 27
4.0 I/O Ports 29
5.0 Timer0 Module 41
6.0 Timer1 Module 47
7.0 Timer2 Module 51
8.0 Capture/Compare/PWM (CCP) Module 53
9.0 Synchronous Serial Port (SSP) Module 59
10.0 Analog-to-Digital Converter (A/D) Module 75
11.0 LCD Module 83
12.0 Special Features of the CPU 97
13.0 Instruction Set Summary 113
14.0 Development Support 133
15.0 Electrical Characteristics 139
16.0 DC and AC Characteristics Graphs and Tables 159
17.0 Packaging Information 161
Appendix A: Revision History 167
Appendix B: Device Differences 167
Appendix C: Conversion Considerations 168
Index 169
On-Line Support 175
Reader Response 176
PIC16C925/926 Product Identification System 177

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.

1.0 DEVICE OVERVIEW

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

1. PIC16C925
2. PIC16C926

The PIC16C925/926 series is a family of low cost, high performance, CMOS, fully static, 8-bit microcontrollers with an integrated LCD Driver module, in the PIC16CXXX mid-range family.

For the PIC16C925/926 family, there are two device "types" as indicated in the device number:

1. **C**, as in PIC16C926. These devices operate over the standard voltage range.
2. **LC**, as in PIC16**LC**926. These devices operate over an extended voltage range.

These devices come in 64-pin and 68-pin packages, as well as die form. Both configurations offer identical peripheral devices and other features. The only difference between the PIC16C925 and PIC16C926 is the additional EPROM and data memory offered in the latter. An overview of features is presented in Table 1-1.

A UV-erasable, CERQUAD packaged version (compatible with PLCC) is also available for both the PIC16C925 and PIC16C926. This version is ideal for cost effective code development.

A block diagram for the PIC16C925/926 family architecture is presented in Figure 1-1.

TABLE 1-1: PIC16C925/926 DEVICE FEATURES

Features	PIC16C925	PIC16C926
Operating Frequency	DC-20 MHz	DC-20 MHz
EPROM Program Memory (words)	4K	8K
Data Memory (bytes)	176	336
Timer Module(s)	TMR0,TMR1,TMR2	TMR0,TMR1,TMR2
Capture/Compare/PWM Module(s)	1	1
Serial Port(s) (SPI/I^2C, USART)	SPI/I^2C	SPI/I^2C
Parallel Slave Port	—	—
A/D Converter (10-bit) Channels	5	5
LCD Module	4 Com, 32 Seg	4 Com, 32 Seg
Interrupt Sources	9	9
I/O Pins	25	25
Input Pins	27	27
Voltage Range (V)	2.5-5.5	2.5-5.5
In-Circuit Serial Programming	Yes	Yes
Brown-out Reset	Yes	Yes
Packages	64-pin TQFP 68-pin PLCC 68-pin CLCC (CERQUAD) Die	64-pin TQFP 68-pin PLCC 68-pin CLCC (CERQUAD) Die

FIGURE 1-1: PIC16C925/926 BLOCK DIAGRAM

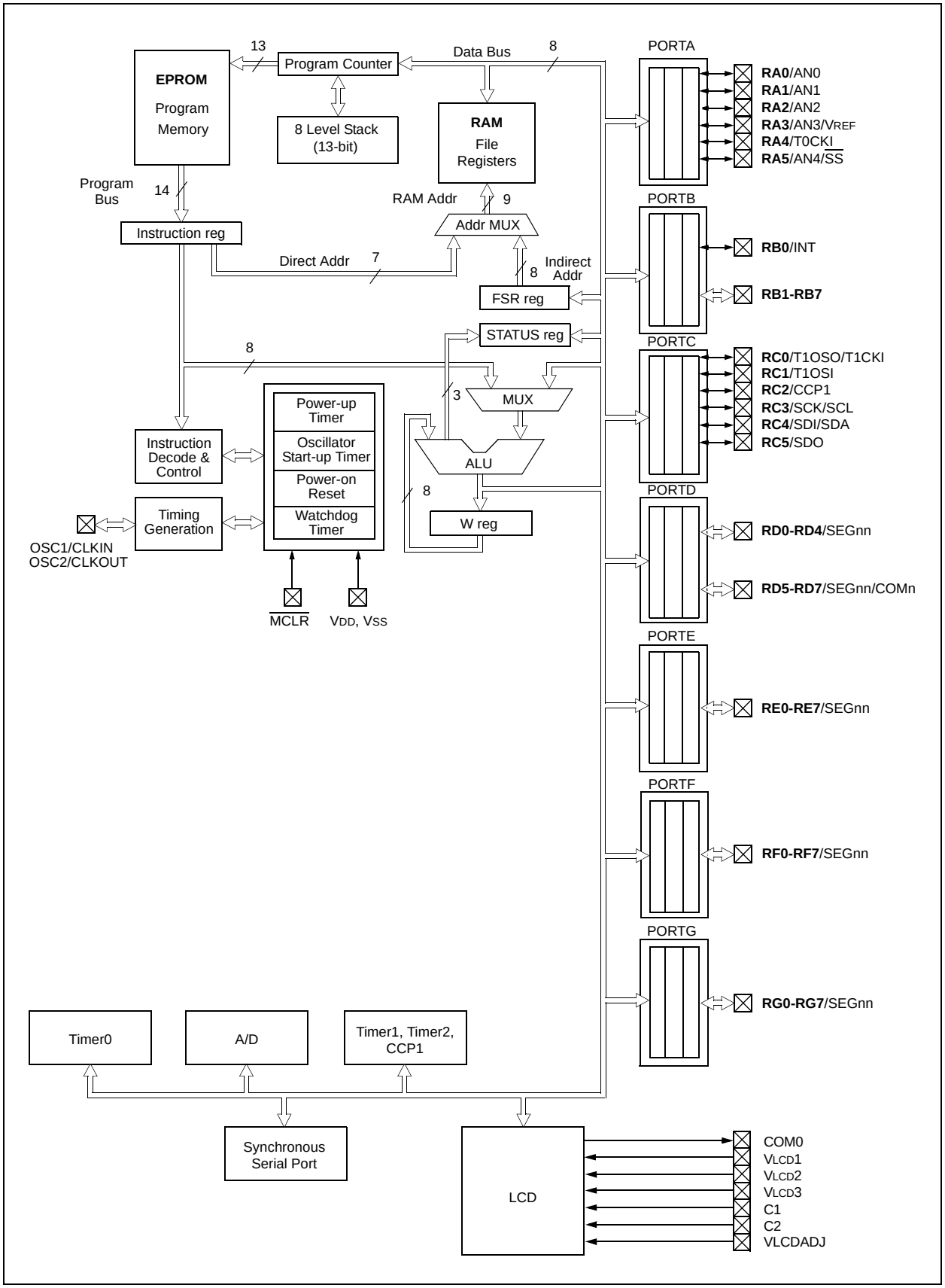


TABLE 1-2: PIC16C925/926 PINOUT DESCRIPTION

Pin Name	PLCC, CLCC Pin#	TQFP Pin#	Pin Type	Buffer Type	Description
OSC1/CLKIN	24	14	I	ST/CMOS	Oscillator crystal input or external clock source input. This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.
OSC2/CLKOUT	25	15	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	2	57	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
					PORTA is a bi-directional I/O port.
RA0/AN0	5	60	I/O	TTL	RA0 can also be Analog input0.
RA1/AN1	6	61	I/O	TTL	RA1 can also be Analog input1.
RA2/AN2	8	63	I/O	TTL	RA2 can also be Analog input2.
RA3/AN3/VREF	9	64	I/O	TTL	RA3 can also be Analog input3 or A/D Voltage Reference.
RA4/T0CKI	10	1	I/O	ST	RA4 can also be the clock input to the Timer0 timer/counter. Output is open drain type.
RA5/AN4/$\overline{SS}$	11	2	I/O	TTL	RA5 can be the slave select for the synchronous serial port or Analog input4.
					PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT	13	4	I/O	TTL/ST	RB0 can also be the external interrupt pin. This buffer is a Schmitt Trigger input when configured as an external interrupt.
RB1	12	3	I/O	TTL	
RB2	4	59	I/O	TTL	
RB3	3	58	I/O	TTL	
RB4	68	56	I/O	TTL	Interrupt-on-change pin.
RB5	67	55	I/O	TTL	Interrupt-on-change pin.
RB6	65	53	I/O	TTL/ST	Interrupt-on-change pin. Serial programming clock. This buffer is a Schmitt Trigger input when used in Serial Programming mode.
RB7	66	54	I/O	TTL/ST	Interrupt-on-change pin. Serial programming data. This buffer is a Schmitt Trigger input when used in Serial Programming mode.
					PORTC is a bi-directional I/O port.
RC0/T1OSO/T1CKI	26	16	I/O	ST	RC0 can also be the Timer1 oscillator output or Timer1 clock input.
RC1/T1OSI	27	17	I/O	ST	RC1 can also be the Timer1 oscillator input.
RC2/CCP1	28	18	I/O	ST	RC2 can also be the Capture1 input/Compare1 output/PWM1 output.
RC3/SCK/SCL	14	5	I/O	ST	RC3 can also be the synchronous serial clock input/ output for both SPI and I^2C modes.
RC4/SDI/SDA	15	6	I/O	ST	RC4 can also be the SPI Data In (SPI mode) or data I/O (I^2C mode).
RC5/SDO	16	7	I/O	ST	RC5 can also be the SPI Data Out (SPI mode).
C1	17	8	P		LCD Voltage Generation.
C2	18	9	P		LCD Voltage Generation.
COM0	63	51	L		Common Driver0.

Legend: I = input O = output P = power L = LCD Driver
— = Not used TTL = TTL input ST = Schmitt Trigger input

TABLE 1-2: PIC16C925/926 PINOUT DESCRIPTION (CONTINUED)

Pin Name	PLCC, CLCC Pin#	TQFP Pin#	Pin Type	Buffer Type	Description
					PORTD is a digital input/output port. These pins are also used as LCD Segment and/or Common Drivers.
RD0/SEG00	31	21	I/O/L	ST	Segment Driver 00/Digital input/output.
RD1/SEG01	32	22	I/O/L	ST	Segment Driver 01/Digital input/output.
RD2/SEG02	33	23	I/O/L	ST	Segment Driver 02/Digital input/output.
RD3/SEG03	34	24	I/O/L	ST	Segment Driver 03/Digital input/output.
RD4/SEG04	35	25	I/O/L	ST	Segment Driver04/Digital input/output.
RD5/SEG29/COM3	60	48	I/L	ST	Segment Driver29/Common Driver 3/Digital input.
RD6/SEG30/COM2	61	49	I/L	ST	Segment Driver30/Common Driver 2/Digital input.
RD7/SEG31/COM1	62	50	I/L	ST	Segment Driver31/Common Driver 1/Digital input.
					PORTE is a Digital input or LCD Segment Driver port.
RE0/SEG05	37	26	I/L	ST	Segment Driver 05.
RE1/SEG06	38	27	I/L	ST	Segment Driver 06.
RE2/SEG07	39	28	I/L	ST	Segment Driver 07.
RE3/SEG08	40	29	I/L	ST	Segment Driver 08.
RE4/SEG09	41	30	I/L	ST	Segment Driver 09.
RE5/SEG10	42	31	I/L	ST	Segment Driver 10.
RE6/SEG11	43	32	I/L	ST	Segment Driver 11.
RE7/SEG27	36	-	I/L	ST	Segment Driver 27 (not available on 64-pin devices).
					PORTF is a Digital input or LCD Segment Driver port.
RF0/SEG12	44	33	I/L	ST	Segment Driver 12.
RF1/SEG13	45	34	I/L	ST	Segment Driver 13.
RF2/SEG14	46	35	I/L	ST	Segment Driver 14.
RF3/SEG15	47	36	I/L	ST	Segment Driver 15.
RF4/SEG16	48	37	I/L	ST	Segment Driver 16.
RF5/SEG17	49	38	I/L	ST	Segment Driver 17.
RF6/SEG18	50	39	I/L	ST	Segment Driver 18.
RF7/SEG19	51	40	I/L	ST	Segment Driver 19.
					PORTG is a Digital input or LCD Segment Driver port.
RG0/SEG20	53	41	I/L	ST	Segment Driver 20.
RG1/SEG21	54	42	I/L	ST	Segment Driver 21.
RG2/SEG22	55	43	I/L	ST	Segment Driver 22.
RG3/SEG23	56	44	I/L	ST	Segment Driver 23.
RG4/SEG24	57	45	I/L	ST	Segment Driver 24.
RG5/SEG25	58	46	I/L	ST	Segment Driver 25.
RG6/SEG26	59	47	I/L	ST	Segment Driver 26.
RG7/SEG28	52	—	I/L	ST	Segment Driver 28 (not available on 64-pin devices).
VLCDADJ	30	20	P	—	LCD Voltage Generation.
AVDD	21	—	P	—	Analog Power (PLCC and CLCC packages only).
VLCD1	29	19	P	—	LCD Voltage.
VLCD2	19	10	P	—	LCD Voltage.
VLCD3	20	11	P	—	LCD Voltage.
VDD	22, 64	12, 52	P	—	Digital power.
VSS	7, 23	13, 62	P	—	Ground reference.
NC	1	—	—	—	These pins are not internally connected. These pins should be left unconnected.

Legend: I = input O = output P = power L = LCD Driver
— = Not used TTL = TTL input ST = Schmitt Trigger input

1.1 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 1-2.

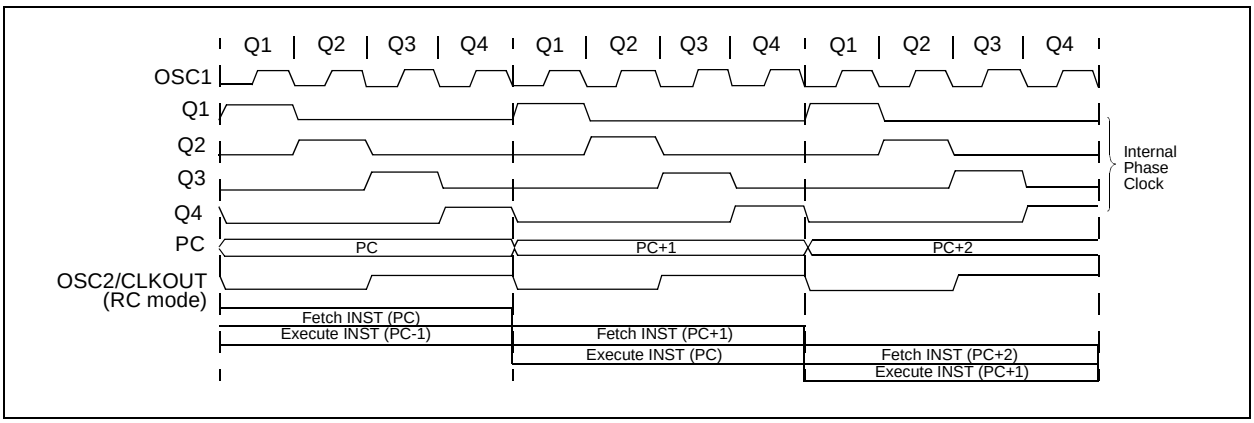
1.2 Instruction Flow/Pipelining

An “Instruction Cycle” consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined, such that fetch takes one instruction cycle, while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g. `GOTO`), then two cycles are required to complete the instruction (Example 1-1).

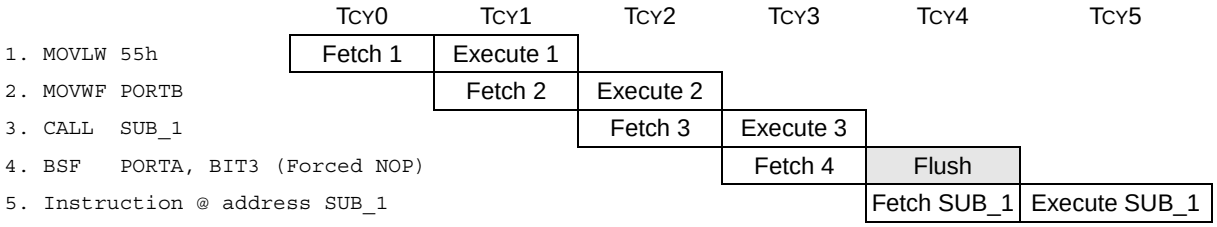
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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

FIGURE 1-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 1-1: INSTRUCTION PIPELINE FLOW



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

NOTES:

2.0 MEMORY ORGANIZATION

2.1 Program Memory Organization

The PIC16C925/926 family has a 13-bit program counter capable of addressing an 8K x 14 program memory space.

For the PIC16C925, only the first 4K x 14 (0000h-0FFFh) are physically implemented. Accessing a location above the physically implemented addresses will cause a wraparound. The RESET vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PROGRAM MEMORY MAP AND STACK FOR PIC16C925

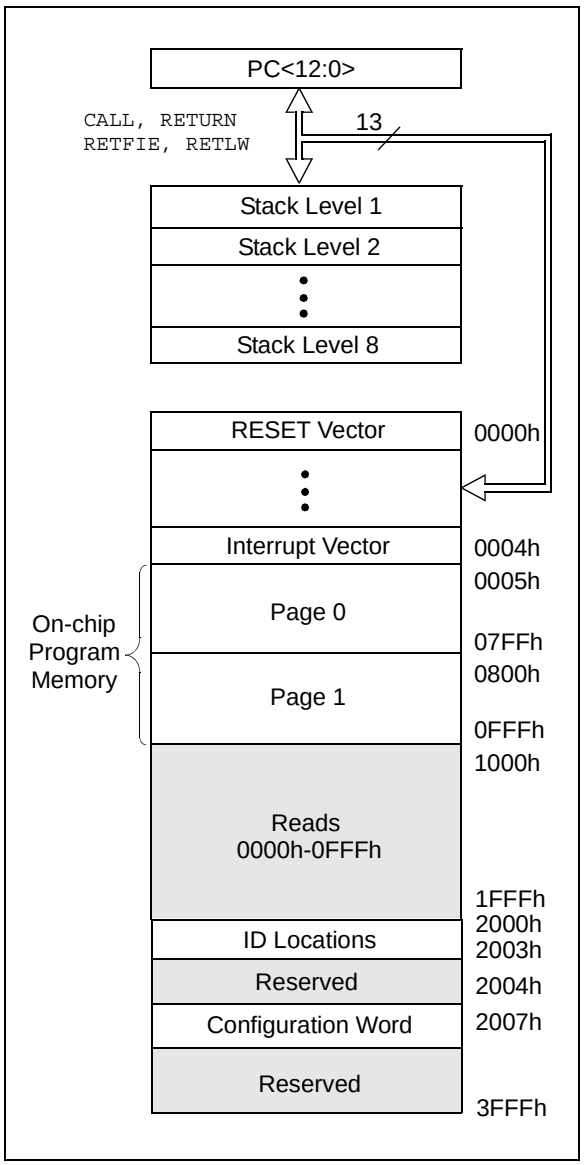
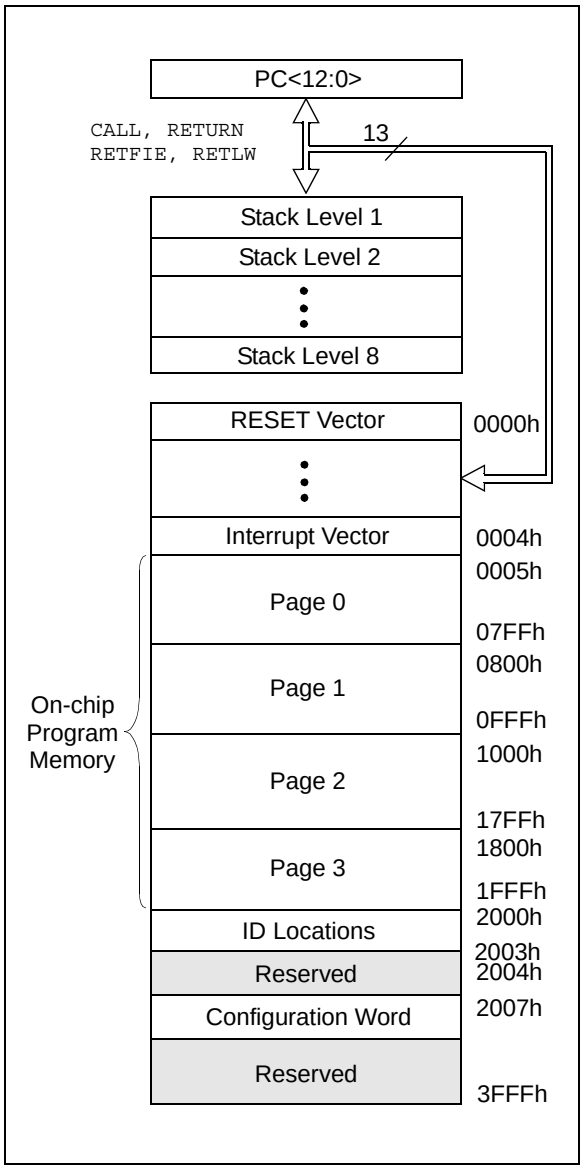


FIGURE 2-2: PROGRAM MEMORY MAP AND STACK FOR PIC16C926



2.2 Data Memory Organization

The data memory is partitioned into four banks which contain the General Purpose Registers and the Special Function Registers. Bits RP1 and RP0 are the bank select bits.

RP1:RP0 (STATUS<6:5>)	Bank
11	3 (180h-1FFh)
10	2 (100h-17Fh)
01	1 (80h-FFh)
00	0 (00h-7Fh)

The lower locations of each Bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers implemented as static RAM. All four banks contain special function registers. Some “high use” special function registers are mirrored in other banks for code reduction and quicker access.

2.2.1 GENERAL PURPOSE REGISTER FILE

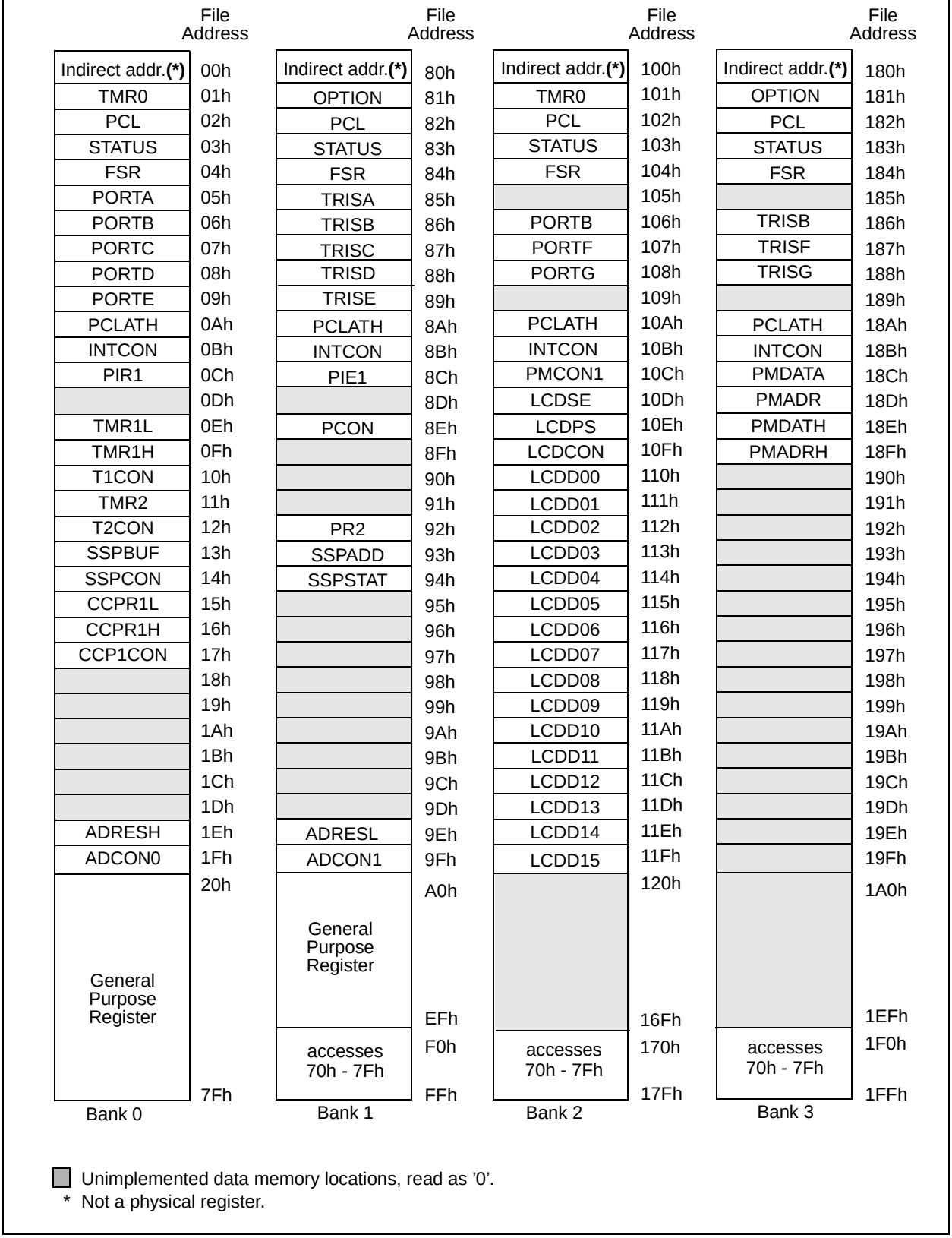
The register file can be accessed either directly, or indirectly through the File Select Register FSR (Section 2.6).

The following General Purpose Registers are not physically implemented:

- F0h-FFh of Bank 1
- 170h-17Fh of Bank 2
- 1F0h-1FFh of Bank 3

These locations are used for common access across banks.

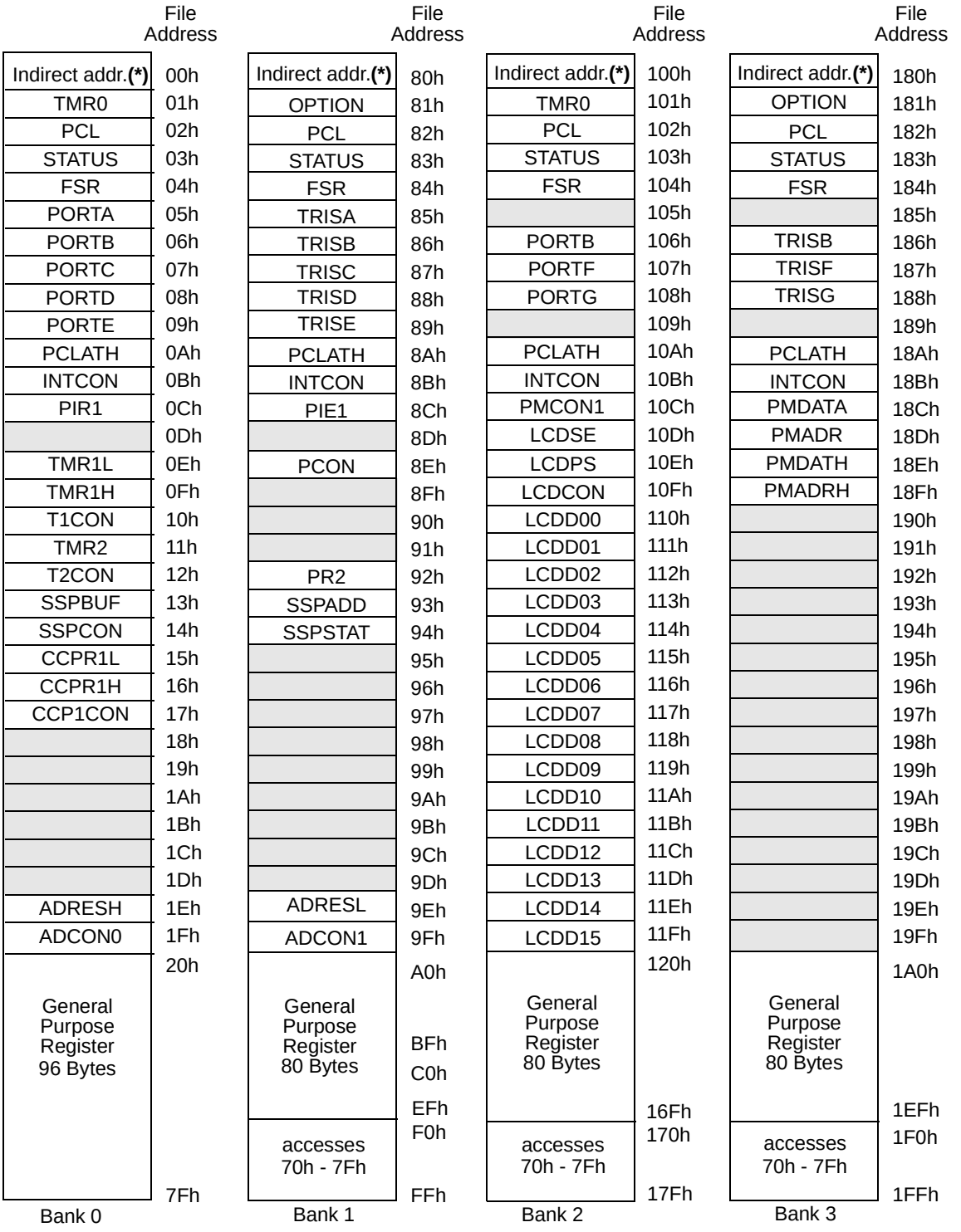
FIGURE 2-3: REGISTER FILE MAP — PIC16C925



Unimplemented data memory locations, read as '0'.

* Not a physical register.

FIGURE 2-4: REGISTER FILE MAP— PIC16C926



Unimplemented data memory locations, read as '0'.

* Not a physical register.

2.3 Special Function Registers

The Special Function Registers (SFRs) are registers used by the CPU and Peripheral Modules for controlling the desired operation of the device. These registers are implemented as static RAM.

The special function registers can be classified into two sets, core and peripheral. Those registers associated with the “core” functions are described in this section. Those related to the operation of the peripheral features are described in the section of that peripheral feature.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Details on page
Bank 0											
00h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26
01h	TMR0	Timer0 Module Register								xxxx xxxx	41
02h	PCL	Program Counter (PC) Least Significant Byte								0000 0000	25
03h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	19
04h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	26
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	29
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	31
07h	PORTC	—	—	PORTC Data Latch when written: PORTC pins when read						--xx xxxx	33
08h	PORTD	PORTD Data Latch when written: PORTD pins when read								0000 0000	34
09h	PORTE	PORTE pins when read								0000 0000	36
0Ah	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	25
0Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	21
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	23
0Dh	—	Unimplemented								—	—
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	47
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	47
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	47
11h	TMR2	Timer2 Module Register								0000 0000	51
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	52
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	64, 72
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	60
15h	CCPR1L	Capture/Compare/PWM Register (LSB)								xxxx xxxx	58
16h	CCPR1H	Capture/Compare/PWM Register (MSB)								xxxx xxxx	58
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	53
18h	—	Unimplemented								—	—
19h	—	Unimplemented								—	—
1Ah	—	Unimplemented								—	—
1Bh	—	Unimplemented								—	—
1Ch	—	Unimplemented								—	—
1Dh	—	Unimplemented								—	—
1Eh	ADRESH	A/D Result Register High								xxxx xxxx	80, 81
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON	0000 0000	75

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0'.
Shaded locations are unimplemented, read as ‘0’.

Note 1: These pixels do not display, but can be used as general purpose RAM.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Details on page
Bank 1											
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26
81h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	20
82h	PCL	Program Counter (PC) Least Significant Byte								0000 0000	25
83h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	19
84h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	26
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	29
86h	TRISB	PORTB Data Direction Register								1111 1111	31
87h	TRISC	—	—	PORTC Data Direction Register						--11 1111	33
88h	TRISD	PORTD Data Direction Register								1111 1111	34
89h	TRISE	PORTE Data Direction Register								1111 1111	36
8Ah	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	25
8Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	21
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	24
8Dh	—	Unimplemented								—	—
8Eh	PCON	—	—	—	—	—	—	POR	$\overline{BOR}$	---- --0-	24
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	—	Unimplemented								—	—
92h	PR2	Timer2 Period Register								1111 1111	51
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	69, 72
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	59
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	—	Unimplemented								—	—
99h	—	Unimplemented								—	—
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	—	Unimplemented								—	—
9Dh	—	Unimplemented								—	—
9Eh	ADRESL	A/D Result Register Low								xxxx xxxx	79
9Fh	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	---- -000	76

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0'.
Shaded locations are unimplemented, read as '0'.

Note 1: These pixels do not display, but can be used as general purpose RAM.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Details on page
Bank 2											
100h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26
101h	TMR0	Timer0 Module Register								xxxx xxxx	41
102h	PCL	Program Counter (PC) Least Significant Byte								0000 0000	25
103h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	19
104h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	26
105h	—	Unimplemented								—	—
106h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	31
107h	PORTF	PORTF pins when read								0000 0000	37
108h	PORTG	PORTG pins when read								0000 0000	38
109h	—	Unimplemented								—	—
10Ah	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	25
10Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	21
10Ch	PMCON1	reserved	—	—	—	—	—	—	RD	1--- ---0	27
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	94
10Eh	LCDPS	—	—	—	—	LP3	LP2	LP1	LP0	---- 0000	84
10Fh	LCDCON	LCDEN	SLPEN	—	VGEN	CS1	CS0	LMUX1	LMUX0	00-0 0000	83
110h	LCDD00	SEG07 COM0	SEG06 COM0	SEG05 COM0	SEG04 COM0	SEG03 COM0	SEG02 COM0	SEG01 COM0	SEG00 COM0	xxxx xxxx	92
111h	LCDD01	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG09 COM0	SEG08 COM0	xxxx xxxx	92
112h	LCDD02	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0	xxxx xxxx	92
113h	LCDD03	SEG31 COM0	SEG30 COM0	SEG29 COM0	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0	xxxx xxxx	92
114h	LCDD04	SEG07 COM1	SEG06 COM1	SEG05 COM1	SEG04 COM1	SEG03 COM1	SEG02 COM1	SEG01 COM1	SEG00 COM1	xxxx xxxx	92
115h	LCDD05	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG09 COM1	SEG08 COM1	xxxx xxxx	92
116h	LCDD06	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1	xxxx xxxx	92
117h	LCDD07	SEG31 COM1(1)	SEG30 COM1	SEG29 COM1	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1	xxxx xxxx	92
118h	LCDD08	SEG07 COM2	SEG06 COM2	SEG05 COM2	SEG04 COM2	SEG03 COM2	SEG02 COM2	SEG01 COM2	SEG00 COM2	xxxx xxxx	92
119h	LCDD09	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG09 COM2	SEG08 COM2	xxxx xxxx	92
11Ah	LCDD10	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2	xxxx xxxx	92
11Bh	LCDD11	SEG31 COM2(1)	SEG30 COM2(1)	SEG29 COM2	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2	xxxx xxxx	92
11Ch	LCDD12	SEG07 COM3	SEG06 COM3	SEG05 COM3	SEG04 COM3	SEG03 COM3	SEG02 COM3	SEG01 COM3	SEG00 COM3	xxxx xxxx	92
11Dh	LCDD13	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG09 COM3	SEG08 COM3	xxxx xxxx	92
11Eh	LCDD14	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3	xxxx xxxx	92
11Fh	LCDD15	SEG31 COM3(1)	SEG30 COM3(1)	SEG29 COM3(1)	SEG28 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3	xxxx xxxx	92

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0'.
Shaded locations are unimplemented, read as '0'.

Note 1: These pixels do not display, but can be used as general purpose RAM.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Details on page
Bank 3											
180h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26
181h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	20
182h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	25
183h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	19
184h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	26
185h	—	Unimplemented								—	—
186h	TRISB	PORTB Data Direction Register								1111 1111	31
187h	TRISF	PORTF Data Direction Register								1111 1111	37
188h	TRISG	PORTG Data Direction Register								1111 1111	38
189h	—	Unimplemented								—	—
18Ah	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	25
18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	21
18Ch	PMDATA	Data Register Low Byte								xxxx xxxx	27
18Dh	PMADR	Address Register Low Byte								xxxx xxxx	27
18Eh	PMDATH	—	—	Data Register High Byte						xxxx xxxx	27
18Fh	PMADRH	—	—	—	Address Register High Byte					xxxx xxxx	27
190h	—	Unimplemented								—	—
191h	—	Unimplemented								—	—
192h	—	Unimplemented								—	—
193h	—	Unimplemented								—	—
194h	—	Unimplemented								—	—
195h	—	Unimplemented								—	—
196h	—	Unimplemented								—	—
197h	—	Unimplemented								—	—
198h	—	Unimplemented								—	—
199h	—	Unimplemented								—	—
19Ah	—	Unimplemented								—	—
19Bh	—	Unimplemented								—	—
19Ch	—	Unimplemented								—	—
19Dh	—	Unimplemented								—	—
19Eh	—	Unimplemented								—	—
19Fh	—	Unimplemented								—	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0'.
Shaded locations are unimplemented, read as '0'.

Note 1: These pixels do not display, but can be used as general purpose RAM.

2.3.1 STATUS REGISTER

The STATUS register, shown in Register 2-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

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 or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. 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` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect the Z, C or DC bits from the STATUS register. For other instructions, not affecting any status bits, see the "Instruction Set Summary."

Note:	The C and DC bits operate as a $\overline{\text{borrow}}$ and digit $\overline{\text{borrow}}$ bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

REGISTER 2-1: STATUS REGISTER (ADDRESS 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP**: Register Bank Select bit (used for indirect addressing)
`1` = Bank 2, 3 (100h - 1FFh)
`0` = Bank 0, 1 (00h - FFh)

bit 6-5 **RP1:RP0**: Register Bank Select bits (used for direct addressing)
`11` = Bank 3 (180h - 1FFh)
`10` = Bank 2 (100h - 17Fh)
`01` = Bank 1 (80h - FFh)
`00` = Bank 0 (00h - 7Fh)

bit 4 **$\overline{TO}$**: Time-out bit
`1` = After power-up, `CLRWDT` instruction, or `SLEEP` instruction
`0` = A WDT time-out occurred

bit 3 **$\overline{PD}$**: Power-down bit
`1` = After power-up or by the `CLRWDT` instruction
`0` = By execution of the `SLEEP` instruction

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 (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
(for $\overline{\text{borrow}}$ the polarity is reversed)
`1` = A carry-out from the 4th low order bit of the result occurred
`0` = No carry-out from the 4th low order bit of the result

bit 0 **C**: Carry/$\overline{\text{borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
(for $\overline{\text{borrow}}$ the polarity is reversed)
`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:	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

2.3.2 OPTION REGISTER

The OPTION register is a readable and writable register, which contains various control bits to configure the TMR0/WDT prescaler, the external RB0/INT pin interrupt, TMR0, and the weak pull-ups on PORTB.

Note:	To achieve a 1:1 prescaler assignment for the TMR0 register, assign the prescaler to the Watchdog Timer.

REGISTER 2-2: OPTION REGISTER (ADDRESS 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

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

bit 6 **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS2:PS0**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

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

2.3.3 INTCON REGISTER

The INTCON Register is a readable and writable register which contains various enable and flag bits for the TMR0 register overflow, RB Port change and external RB0/INT pin interrupts.

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, GIE (INTCON<7>).

REGISTER 2-3: INTCON REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

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	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE/GEIL:** Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

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

bit 4 **INTE:** RB0/INT0 External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT 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 **INTF:** RB0/INT0 External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT 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

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

2.3.4 PIE1 REGISTER

This register contains the individual enable bits for the peripheral interrupts.

Note:	Bit PEIE (INTCON<6>) must be set to enable any peripheral interrupt.

REGISTER 2-4: PIE1 REGISTER (ADDRESS 8Ch)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7 **LCDIE:** LCD Interrupt Enable bit
1 = Enables the LCD interrupt
0 = Disables the LCD interrupt

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

bit 5-4 **Unimplemented:** Read as '0'

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

2.3.5 PIR1 REGISTER

This register contains the individual flag bits for the peripheral interrupts.

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, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-5: PIR1 REGISTER (ADDRESS 0Ch)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit 7 **LCDIF:** LCD Interrupt Flag bit
1 = LCD interrupt has occurred (must be cleared in software)
0 = LCD interrupt did not occur

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 5-4 **Unimplemented:** Read as '0'

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 reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.3.6 PCON REGISTER

The Power Control (PCON) register contains a flag bit to allow differentiation between a Power-on Reset (POR) to an external $\overline{MCLR}$ Reset or WDT Reset.

For various RESET conditions, see Table 12-4 and Table 12-5.

REGISTER 2-6: PCON REGISTER (ADDRESS 8Eh)

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

bit 7-2 **Unimplemented:** Read as '0'

bit 1 **$\overline{POR}$**: Power-on Reset Status bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

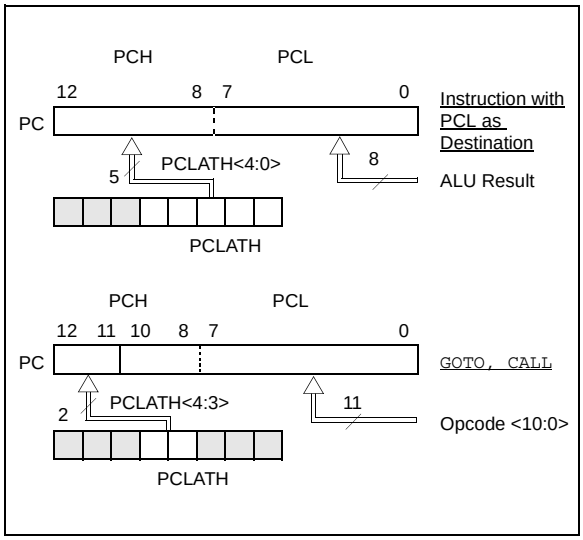
bit 0 **$\overline{BOR}$**: Brown-out Reset Status bit
1 = No Brown-out Reset occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

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

2.4 PCL and PCLATH

The program counter (PC) is 13-bits wide. The low byte comes from the PCL register, which is a readable and writable register. The upper bits (PC<12:8>) are not readable, but are indirectly writable through the PCLATH register. On any RESET, the upper bits of the PC will be cleared. Figure 2-5 shows the two situations for the loading of the PC. The upper example in the figure shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in the figure shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

FIGURE 2-5: LOADING OF PC IN DIFFERENT SITUATIONS



2.4.1 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When doing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256 byte block). Refer to the application note *"Implementing a Table Read"* (AN556).

2.4.2 STACK

The PIC16CXXX family has an 8-level deep x 13-bit wide hardware stack. The stack space is not part of either program or data space and the stack pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed, or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

Note 1: There are no status bits to indicate stack overflow or stack underflow conditions.

2: There are no instructions/mnemonics called PUSH or POP. These are actions that occur from the execution of the CALL, RETURN, RETLW, and RETFIE instructions, or the vectoring to an interrupt address.

2.5 Program Memory Paging

PIC16C925/926 devices are capable of addressing a continuous 8K word block of program memory. The CALL and GOTO instructions provide only 11-bits of address to allow branching within any 2K program memory page. When doing a CALL or GOTO instruction, the upper 2-bits of the address are provided by PCLATH<4:3>. When doing a CALL or GOTO instruction, the user must ensure that the page select bits are programmed so that the desired program memory page is addressed. If a return from a CALL instruction (or interrupt) is executed, the entire 13-bit PC is pushed onto the stack. Therefore, manipulation of the PCLATH<4:3> bits is not required for the RETURN instructions (which POPs the address from the stack).

Note: The contents of the PCLATH register are unchanged after a RETURN or RETFIE instruction is executed. The user must rewrite the PCLATH for any subsequent CALL or GOTO instructions.

Example 2-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that PCLATH is saved and restored by the Interrupt Service Routine (if interrupts are used).

EXAMPLE 2-1: CALL OF A SUBROUTINE IN PAGE 1 FROM PAGE 0

```
ORG 0x500
BCF    PCLATH,4
BSF    PCLATH,3  ;Select page 1 (800h-FFFh)
CALL   SUB1_P1   ;Call subroutine in
       :         ;page 1 (800h-FFFh)
       :
       :
ORG 0x900
SUB1_P1:         ;called subroutine
       :         ;page 1 (800h-FFFh)
       :
RETURN           ;return to Call subroutine
                 ;in page 0 (000h-7FFh)
```

2.6 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

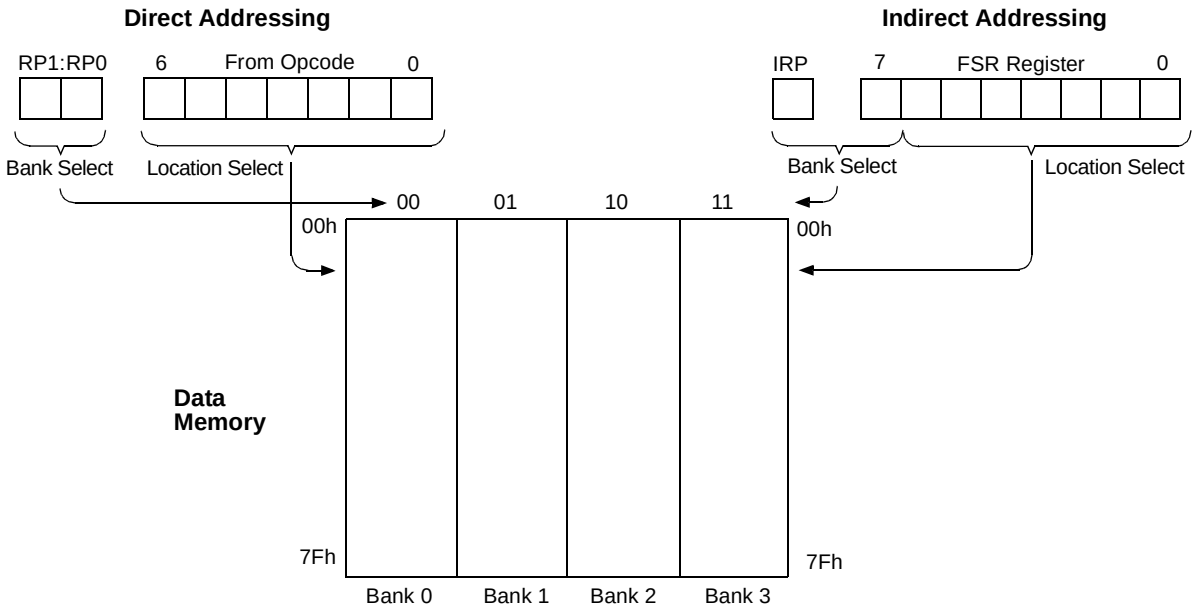
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register (FSR). Reading the INDF register itself, indirectly (FSR = '0'), will produce 00h. Writing to the INDF register indirectly results in a no operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 2-6.

A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 2-2.

EXAMPLE 2-2: INDIRECT ADDRESSING

```
            MOVLW  0x20     ;initialize pointer
            MOVWF  FSR      ;to RAM
NEXT        CLRF   INDF     ;clear INDF register
            INCF   FSR,F    ;inc pointer
            BTFSS  FSR,4    ;all done?
            GOTO   NEXT     ;no clear next
CONTINUE
            :               ;yes continue
```

FIGURE 2-6: DIRECT/INDIRECT ADDRESSING



Note: For memory map detail, see Figure 2-3.

3.0 READING PROGRAM MEMORY

The Program Memory is readable during normal operation over the entire VDD range. It is indirectly addressed through Special Function Registers (SFR). Up to 14-bit numbers can be stored in memory for use as calibration parameters, serial numbers, packed 7-bit ASCII, etc. Executing a program memory location containing data that forms an invalid instruction results in a NOP.

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

- PMCON1
- PMDATA
- PMDATH
- PMADR
- PMADRH

The program memory allows word reads. Program memory access allows for checksum calculation and reading calibration tables.

When interfacing to the program memory block, the PMDATH:PMDATA registers form a two-byte word, which holds the 14-bit data for reads. The PMADRH:PMADR registers form a two-byte word, which holds the 13-bit address of the location being accessed. These devices can have from 4K words to 8K words of program memory, with an address range from 0h to 3FFFh.

The unused upper bits in both the PMDATH and PMADRH registers are not implemented and read as “0’s”.

3.1 PMADR

The address registers can address up to a maximum of 8K words of program memory.

When selecting a program address value, the MSByte of the address is written to the PMADRH register and the LSByte is written to the PMADR register. The upper MSbits of PMADRH must always be clear.

3.2 PMCON1 Register

PMCON1 is the control register for memory accesses.

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

REGISTER 3-1: PMCON1 REGISTER (ADDRESS 10Ch)

R-1	U-0	U-0	U-0	U-x	U-0	U-0	R/S-0
r	—	—	—	—	—	—	RD
bit 7							bit 0

bit 7 **Reserved**: Read as ‘1’

bit 6-1 **Unimplemented:** Read as ‘0’

bit 0 **RD**: Read Control bit

1 = Initiates a read, RD is cleared in hardware. The RD bit can only be set (not cleared) in software.

0 = Does not initiate a read

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

3.3 Reading the Program Memory

A program memory location may be read by writing two bytes of the address to the PMADR and PMADRH registers, and then setting control bit RD (PMCON1<0>). Once the read control bit is set, the microcontroller will use the next two instruction cycles to read the data. The data is available in the PMDATA and PMDATH registers after the NOP instruction. Therefore, it can be read as two bytes in the following instructions. The PMDATA and PMDATH registers will hold this value until another read operation.

EXAMPLE 3-1: PROGRAM READ

```
     BSF     STATUS, RP1             ;
     BSF     STATUS, RP0             ; Bank 3
     MOVLW   MS_PROG_PM_ADDR         ;
     MOVWF   PMADRH                  ; MS Byte of Program Address to read
     MOVLW   LS_PROG_PM_ADDR         ;
     MOVWF   PMADR                   ; LS Byte of Program Address to read
     BCF     STATUS, RP0             ; Bank 2
     BSF     PMCON1, RD              ; PM Read
;
                                     ; First instruction after BSF PMCON1,RD executes normally
     BSF     STATUS, RP0             ; Bank 3
;
     NOP                             ; Any instructions here are ignored as program
                                     ; memory is read in second cycle after BSF PMCON1,RD
;
     MOVF    PMDATA, W               ; W = LS Byte of Program PMDATA
     MOVF    PMDATH, W               ; W = MS Byte of Program PMDATA
```

3.4 Operation During Code Protect

If the program memory is not code protected, the program memory control can read anywhere within the program memory.

If the entire program memory is code protected, the program memory control can read anywhere within the program memory.

If only part of the program memory is code protected, the program memory control can read the unprotected segment and cannot read the protected segment. The protected area cannot be read, because it may be possible to write a downloading routine into the unprotected segment.

TABLE 3-1: REGISTERS ASSOCIATED WITH PROGRAM MEMORY

Address	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
10Ch	PMCON1	**(1)**	—	—	—	—	—	—	RD	1--- ---0	1--- ---0
18Ch	PMDATA	Data Register Low Byte								xxxx xxxx	uuuu uuuu
18Dh	PMADR	Address Register Low Byte								xxxx xxxx	uuuu uuuu
18Eh	PMDATH	—	—	Data Register High Byte						xxxx xxxx	uuuu uuuu
18Fh	PMADRH	—	—	—	Address Register High Byte					xxxx xxxx	uuuu uuuu

Legend: x = unknown, u = unchanged, r = reserved, - = unimplemented, read as '0'.
Shaded cells are not used during FLASH access.

Note 1: This bit always reads as a '1'.

4.0 I/O PORTS

Some pins for these ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

4.1 PORTA and TRISA Register

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. All RA pins have data direction bits (TRISA register), which can configure these pins as output or input.

Setting a bit in the TRISA register puts the corresponding output driver in a Hi-Impedance mode. Clearing a bit in the TRISA register puts 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. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified, and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The other PORTA pins are multiplexed with analog inputs and the analog VREF input. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register1).

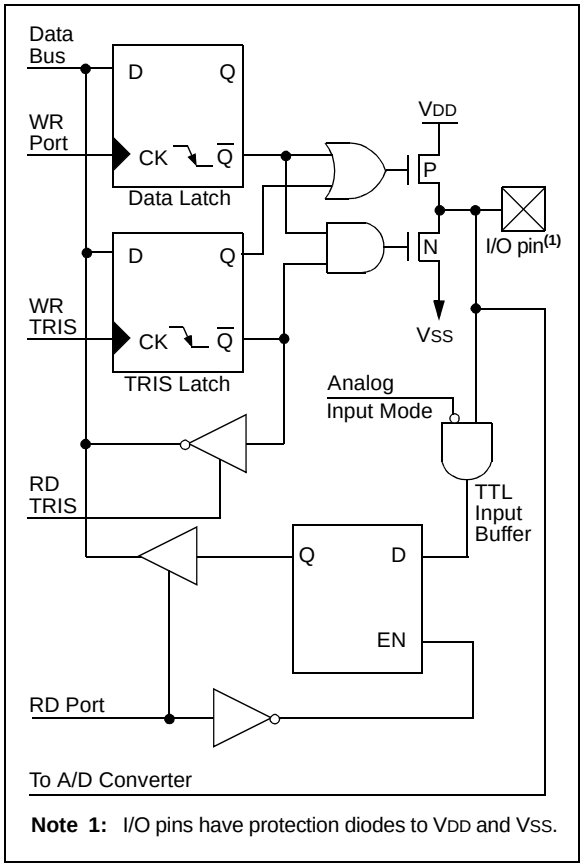
Note:	On a Power-on Reset, these pins are configured as analog inputs and read as '0'.

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 4-1: INITIALIZING PORTA

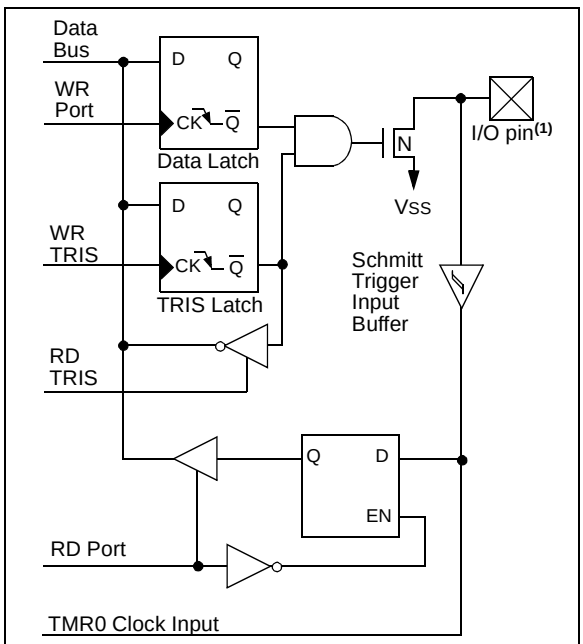
```
BCF    STATUS, RP0   ; Select Bank0
BCF    STATUS, RP1
CLRF   PORTA         ; Initialize PORTA
BSF    STATUS, RP0   ; Select Bank1
MOVLW  0xCF          ; Value used to
                     ; initialize data
                     ; direction
MOVWF  TRISA         ; Set RA<3:0> as inputs
                     ; RA<5:4> as outputs
                     ; RA<7:6> are always
                     ; read as '0'.
```

FIGURE 4-1: BLOCK DIAGRAM OF PINS RA3:RA0 AND RA5



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

FIGURE 4-2: BLOCK DIAGRAM OF RA4/T0CKI PIN



Note 1: I/O pin has protection diodes to VSS only.

TABLE 4-1: PORTA FUNCTIONS

Name	Bit#	Buffer	Function
RA0/AN0	bit0	TTL	Input/output or analog input.
RA1/AN1	bit1	TTL	Input/output or analog input.
RA2/AN2	bit2	TTL	Input/output or analog input.
RA3/AN3/VREF	bit3	TTL	Input/output or analog input or VREF.
RA4/T0CKI	bit4	ST	Input/output or external clock input for Timer0. Output is open drain type.
RA5/AN4/$\overline{SS}$	bit5	TTL	Input/output or analog input or slave select input for synchronous serial port.

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

TABLE 4-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
05h	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--0x 0000	--0x 0000
85h	TRISA	—	—	PORTA Data Direction Control Register						--11 1111	--11 1111
9Fh	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	---- -000	---- -000

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

4.2 PORTB and TRISB Register

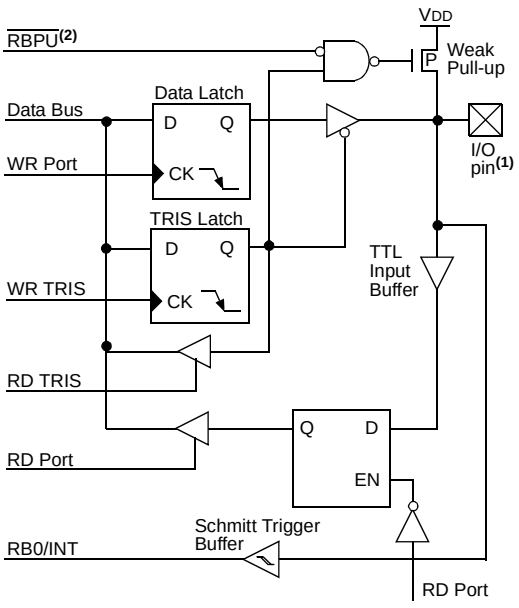
PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a bit in the TRISB register puts the corresponding output driver in a Hi-Impedance Input mode. Clearing a bit in the TRISB register puts the contents of the output latch on the selected pin(s).

EXAMPLE 4-2: INITIALIZING PORTB

```
BCF     STATUS, RP0   ; Select Bank0
BCF     STATUS, RP1
CLRF    PORTB         ; Initialize PORTB
BSF     STATUS, RP0   ; Select Bank1
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 $\overline{\text{RBPU}}$ (OPTION<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are also disabled on a Power-on Reset.

FIGURE 4-3: BLOCK DIAGRAM OF RB3: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<7>).

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

This interrupt can wake the device from SLEEP. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

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

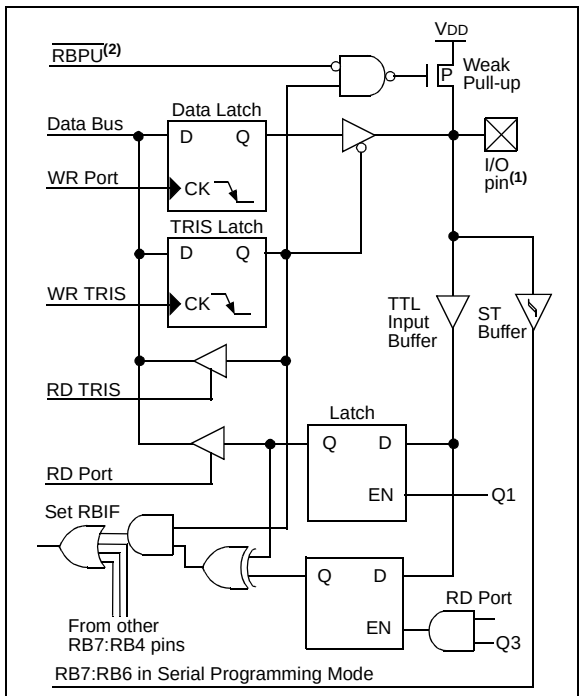
b) Clear flag bit RBIF.

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.

This interrupt-on-mismatch feature, together with software configurable pull-ups on these four pins, allow easy interface to a keypad and make it possible for wake-up on key depression. Refer to the *Embedded Control Handbook, "Implementing Wake-Up on Key Stroke"* (AN552).

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.

FIGURE 4-4: 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 (OPTION<7>).

TABLE 4-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/INT	bit0	TTL/ST	Input/output pin or external interrupt input. Internal software programmable weak pull-up. This buffer is a Schmitt Trigger input when configured as the external interrupt.
RB1	bit1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3	bit3	TTL	Input/output pin. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6	bit6	TTL/ST	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock. This buffer is a Schmitt Trigger input when used in Serial Programming mode.
RB7	bit7	TTL/ST	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data. This buffer is a Schmitt Trigger input when used in Serial Programming mode.

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

TABLE 4-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
06h, 106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h, 186h	TRISB	PORTB Data Direction Control Register								1111 1111	1111 1111
81h, 181h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

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

4.3 PORTC and TRISC Register

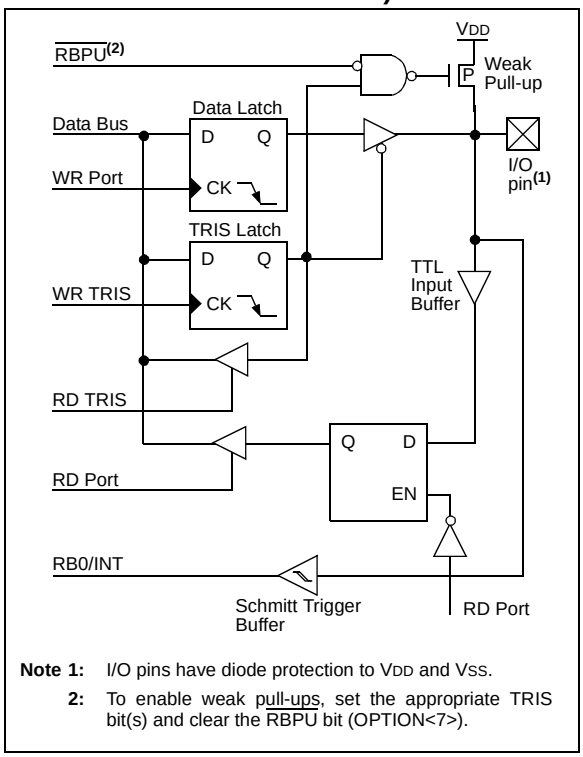
PORTC is a 6-bit, bi-directional port. Each pin is individually configurable as an input or output through the TRISC register. PORTC is multiplexed with several peripheral functions (Table 4-5). PORTC pins have Schmitt Trigger input buffers.

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. Since the TRIS bit override is in effect while the peripheral is enabled, read-modify-write instructions (`BSF, BCF, XORWF`) with TRISC as destination should be avoided. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

EXAMPLE 4-3: INITIALIZING PORTC

```
BCF     STATUS,RP0 ; Select Bank0
BCF     STATUS,RP1
CLRF    PORTC      ; Initialize PORTC
BSF     STATUS,RP0 ; Select Bank1
MOVLW   0xCF       ; Value used to
                   ; initialize data
                   ; direction
MOVWF   TRISC      ; Set RC<3:0> as inputs
                   ; RC<5:4> as outputs
                   ; RC<7:6> always read 0
```

FIGURE 4-5: PORTC BLOCK DIAGRAM (PERIPHERAL OUTPUT OVERRIDE)



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 RBPU bit (OPTION<7>).

TABLE 4-5: PORTC FUNCTIONS

Name	Bit#	Buffer Type	Function
RC0/T1OSO/T1CKI	bit0	ST	Input/output port pin or Timer1 oscillator output or Timer1 clock input.
RC1/T1OSI	bit1	ST	Input/output port pin or Timer1 oscillator input.
RC2/CCP1	bit2	ST	Input/output port pin or Capture input/Compare output/PWM output.
RC3/SCK/SCL	bit3	ST	Input/output port pin or the synchronous serial clock for both SPI and I^2C modes.
RC4/SDI/SDA	bit4	ST	Input/output port pin or the SPI Data In (SPI mode) or data I/O (I^2C mode).
RC5/SDO	bit5	ST	Input/output port pin or Synchronous Serial Port data out.

Legend: ST = Schmitt Trigger input

TABLE 4-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
07h	PORTC	—	—	RC5	RC4	RC3	RC2	RC1	RC0	--xx xxxx	--uu uuuu
87h	TRISC	—	—	PORTC Data Direction Control Register						--11 1111	--11 1111

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

4.4 PORTD and TRISD Registers

PORTD is an 8-bit port with Schmitt Trigger input buffers. The first five pins are configurable as general purpose I/O pins or LCD segment drivers. Pins RD5, RD6 and RD7 can be digital inputs, or LCD segment, or common drivers.

TRISD controls the direction of pins RD0 through RD4 when PORTD is configured as a digital port.

Note 1: On a Power-on Reset, these pins are configured as LCD segment drivers.

2: To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

EXAMPLE 4-4: INITIALIZING PORTD

```
BCF   STATUS,RP0     ;Select Bank2
BSF   STATUS,RP1     ;
BCF   LCDSE, SE29    ;Make RD<7:5> digital
BCF   LCDSE, SE0     ;Make RD<4:0> digital
BSF   STATUS,RP0     ;Select Bank1
BCF   STATUS,RP1     ;
MOVLW 0xE0           ;Make RD<4:0> outputs
MOVWF TRISD          ;Make RD<7:5> inputs
```

FIGURE 4-6: PORTD <4:0> BLOCK DIAGRAM

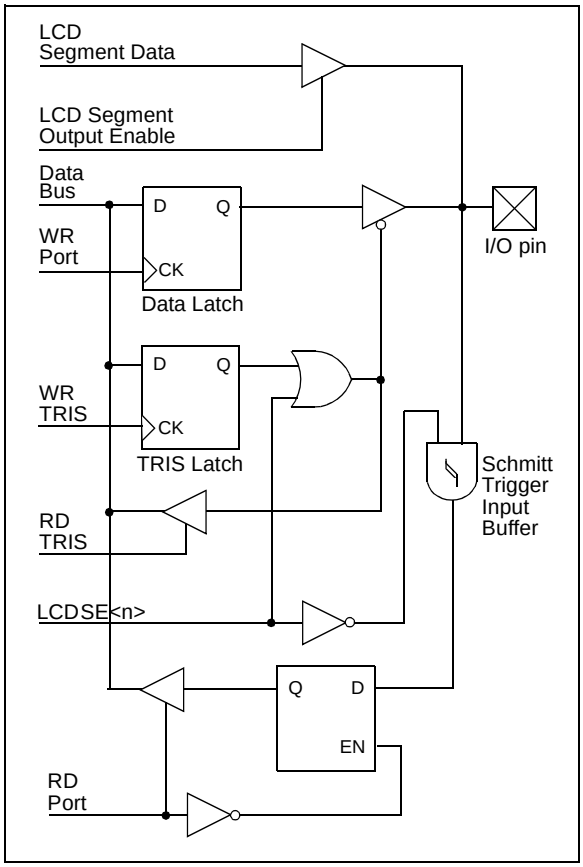


FIGURE 4-7: PORTD<7:5> BLOCK DIAGRAM

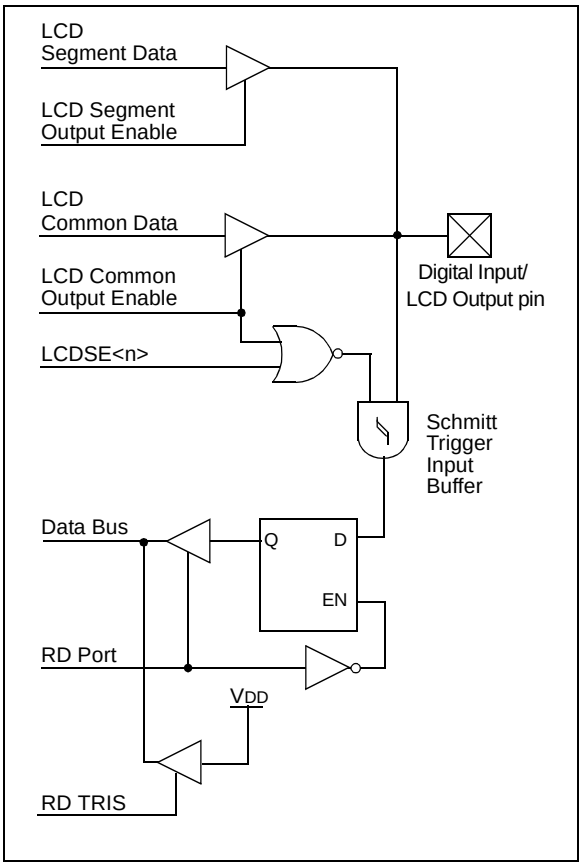


TABLE 4-7: PORTD FUNCTIONS

Name	Bit#	Buffer Type	Function
RD0/SEG00	bit0	ST	Input/output port pin or Segment Driver00.
RD1/SEG01	bit1	ST	Input/output port pin or Segment Driver01.
RD2/SEG02	bit2	ST	Input/output port pin or Segment Driver02.
RD3/SEG03	bit3	ST	Input/output port pin or Segment Driver03.
RD4/SEG04	bit4	ST	Input/output port pin or Segment Driver04.
RD5/SEG29/COM3	bit5	ST	Digital input pin or Segment Driver29 or Common Driver3.
RD6/SEG30/COM2	bit6	ST	Digital input pin or Segment Driver30 or Common Driver2.
RD7/SEG31/COM1	bit7	ST	Digital input pin or Segment Driver31 or Common Driver1.

Legend: ST = Schmitt Trigger input

TABLE 4-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
08h	PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	0000 0000	0000 0000
88h	TRISD	PORTD Data Direction Control Register								1111 1111	1111 1111
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	1111 1111

Legend: Shaded cells are not used by PORTD.

4.5 PORTE and TRISE Register

PORTE is a digital input only port. Each pin is multiplexed with an LCD segment driver. These pins have Schmitt Trigger input buffers.

Note 1: On a Power-on Reset, these pins are configured as LCD segment drivers.

2: To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

EXAMPLE 4-5: INITIALIZING PORTE

```
BCF STATUS,  RP0     ;Select Bank2
BSF STATUS,  RP1     ;
BCF LCDSE,   SE27    ;Make all PORTE
BCF LCDSE,   SE5     ;and PORTG<7>
BCF LCDSE,   SE9     ;digital inputs
```

FIGURE 17-1: PORTE BLOCK DIAGRAM

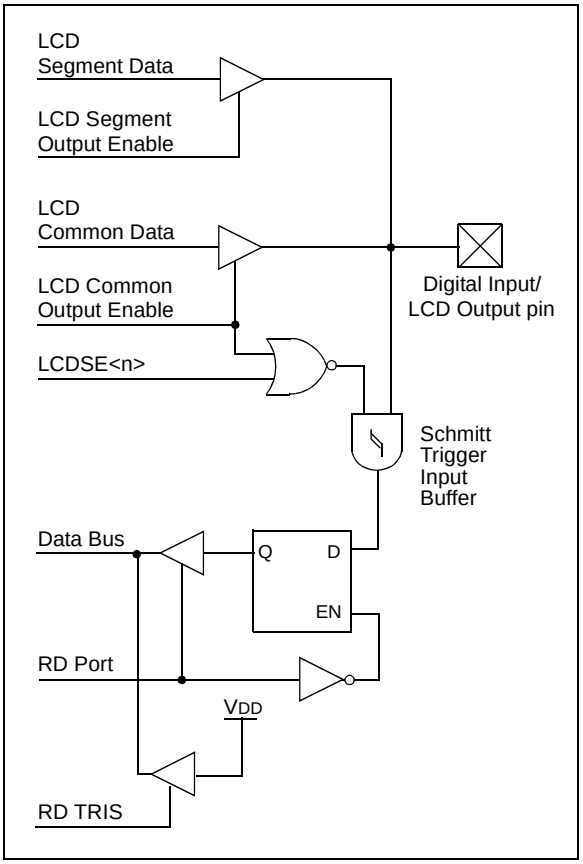


TABLE 4-9: PORTE FUNCTIONS

Name	Bit#	Buffer Type	Function
RE0/SEG05	bit0	ST	Digital input or Segment Driver05.
RE1/SEG06	bit1	ST	Digital input or Segment Driver06.
RE2/SEG07	bit2	ST	Digital input or Segment Driver07.
RE3/SEG08	bit3	ST	Digital input or Segment Driver08.
RE4/SEG09	bit4	ST	Digital input or Segment Driver09.
RE5/SEG10	bit5	ST	Digital input or Segment Driver10.
RE6/SEG11	bit6	ST	Digital input or Segment Driver11.
RE7/SEG27	bit7	ST	Digital input or Segment Driver27 (not available on 64-pin devices).

Legend: ST = Schmitt Trigger input

TABLE 4-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
09h	PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	0000 0000	0000 0000
89h	TRISE	PORTE Data Direction Control Register								1111 1111	1111 1111
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	1111 1111

Legend: Shaded cells are not used by PORTE.

4.6 PORTF and TRISF Register

PORTF is a digital input only port. Each pin is multiplexed with an LCD segment driver. These pins have Schmitt Trigger input buffers.

Note 1: On a Power-on Reset, these pins are configured as LCD segment drivers.

2: To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

EXAMPLE 4-6: INITIALIZING PORTF

```
BCF STATUS,  RP0    ;Select Bank2
BSF STATUS,  RP1    ;
BCF LCDSE,   SE16   ;Make all PORTF
BCF LCDSE,   SE12   ;digital inputs
```

FIGURE 4-8: PORTF BLOCK DIAGRAM

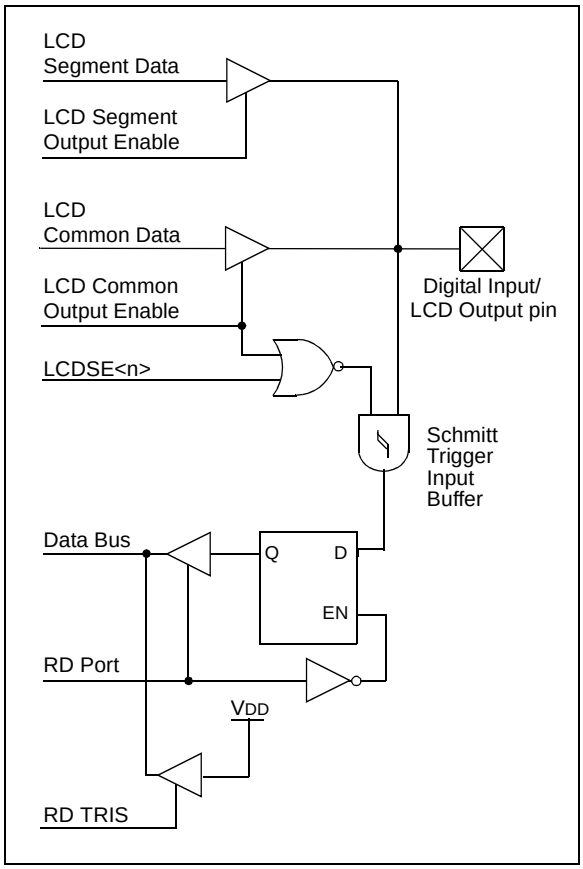


TABLE 4-11: PORTF FUNCTIONS

Name	Bit#	Buffer Type	Function
RF0/SEG12	bit0	ST	Digital input or Segment Driver12.
RF1/SEG13	bit1	ST	Digital input or Segment Driver13.
RF2/SEG14	bit2	ST	Digital input or Segment Driver14.
RF3/SEG15	bit3	ST	Digital input or Segment Driver15.
RF4/SEG16	bit4	ST	Digital input or Segment Driver16.
RF5/SEG17	bit5	ST	Digital input or Segment Driver17.
RF6/SEG18	bit6	ST	Digital input or Segment Driver18.
RF7/SEG19	bit7	ST	Digital input or Segment Driver19.

Legend: ST = Schmitt Trigger input

TABLE 4-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
107h	PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	0000 0000	0000 0000
187h	TRISF	PORTF Data Direction Control Register								1111 1111	1111 1111
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	1111 1111

Legend: Shaded cells are not used by PORTF.

4.7 PORTG and TRISG Register

PORTG is a digital input only port. Each pin is multiplexed with an LCD segment driver. These pins have Schmitt Trigger input buffers.

Note 1: On a Power-on Reset, these pins are configured as LCD segment drivers.

2: To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

EXAMPLE 4-7: INITIALIZING PORTG

```
BCF   STATUS,   RP0    ;Select Bank2
BSF   STATUS,   RP1    ;
BCF   LCDSE,    SE27   ;Make all PORTG
BCF   LCDSE,    SE20   ;and PORTE<7>
                       ;digital inputs
```

FIGURE 4-9: PORTG BLOCK DIAGRAM

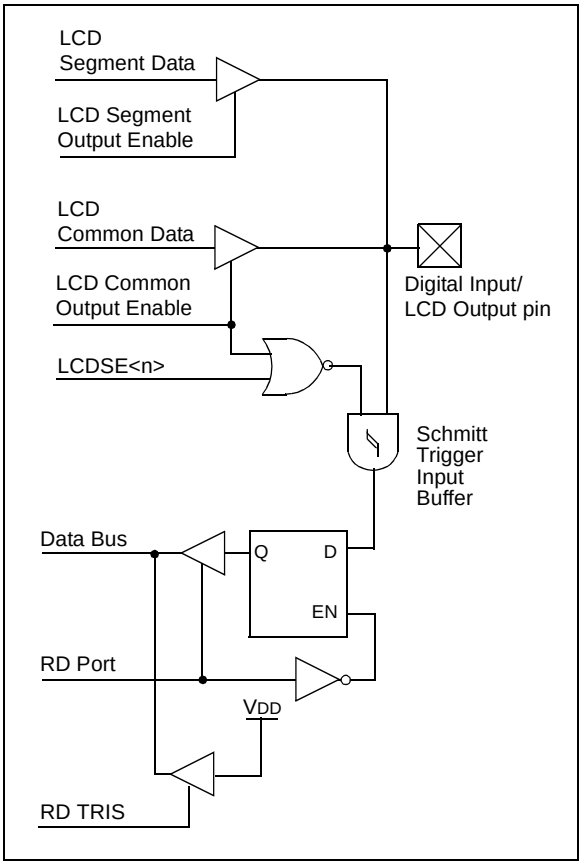


TABLE 4-13: PORTG FUNCTIONS

Name	Bit#	Buffer Type	Function
RG0/SEG20	bit0	ST	Digital input or Segment Driver20.
RG1/SEG21	bit1	ST	Digital input or Segment Driver21.
RG2/SEG22	bit2	ST	Digital input or Segment Driver22.
RG3/SEG23	bit3	ST	Digital input or Segment Driver23.
RG4/SEG24	bit4	ST	Digital input or Segment Driver24.
RG5/SEG25	bit5	ST	Digital input or Segment Driver25.
RG6/SEG26	bit6	ST	Digital input or Segment Driver26.
RG7/SEG28	bit7	ST	Digital input or Segment Driver28 (not available on 64-pin devices).

Legend: ST = Schmitt Trigger input

TABLE 4-14: SUMMARY OF REGISTERS ASSOCIATED WITH PORTG

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
108h	PORTG	RG7	RG6	RG5	RG4	RG3	RG2	RG1	RG0	0000 0000	0000 0000
188h	TRISG	PORTG Data Direction Control Register								1111 1111	1111 1111
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	1111 1111

Legend: Shaded cells are not used by PORTG.

4.8 I/O Programming Considerations

4.8.1 BI-DIRECTIONAL I/O PORTS

Any instruction which writes, operates internally as a read followed by a write operation. The BCF and BSF instructions, for example, read the register into the CPU, execute the bit operation and write the result back to the register. Caution must be used when these instructions are applied to a port with both inputs and outputs defined. For example, a BSF operation on bit5 of PORTB will cause all eight bits of PORTB to be read into the CPU. Then the BSF operation takes place on bit5 and PORTB is written to the output latches. If another bit of PORTB is used as a bi-directional I/O pin (e.g., bit0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and rewritten to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the input mode, no problem occurs. However, if bit0 is switched into output mode later on, the contents of the data latch may now be unknown.

Reading the port register reads the values of the port pins. Writing to the port register, writes the value to the port latch. When using read-modify-write instructions (e.g. BCF, BSF) on a port, the value of the port pins is read, the desired operation is done to this value, and this value is then written to the port latch.

Example 4-8 shows the effect of two sequential read-modify-write instructions on an I/O port. A pin actively outputting a Low or High should not be driven from external devices at the same time, in order to change the level on this pin ("wired-or", "wired-and"). The resulting high output currents may damage the chip.

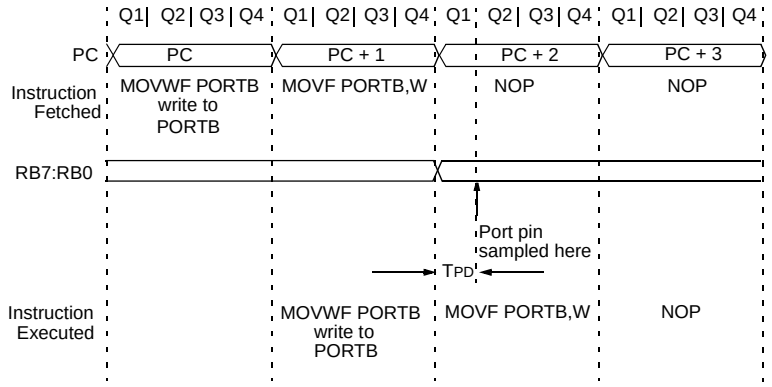
EXAMPLE 4-8: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

```
;Initial PORT settings: PORTB<7:4> Inputs
;                       PORTB<3:0> Outputs
;PORTB<7:6> have external pull-ups and are
;not connected to other circuitry
;
;                    PORT latch   PORT pins
;                    ----------   ---------
  BCF PORTB,  7    ; 01pp pppp    11pp pppp
  BCF PORTB,  6    ; 10pp pppp    11pp pppp
  BCF STATUS, RP1  ; Select Bank1
  BSF STATUS, RP0  ;
  BCF TRISB,  7    ; 10pp pppp    11pp pppp
  BCF TRISB,  6    ; 10pp pppp    10pp pppp
;
;Note that the user may have expected the
;pin values to be 00pp ppp. The 2nd BCF
;caused RB7 to be latched as the pin value
;(high).
```

4.8.2 SUCCESSIVE OPERATIONS ON I/O PORTS

The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle (Figure 4-10). Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such to allow the pin voltage to stabilize (load dependent) before the next instruction, which causes that file to be read into the CPU, is executed. Otherwise, the previous state of that pin may be read into the CPU, rather than the new state. When in doubt, it is better to separate these instructions with a NOP, or another instruction not accessing this I/O port.

FIGURE 4-10: SUCCESSIVE I/O OPERATION



Note:

This example shows a write to PORTB followed by a read from PORTB.

Note that:

data setup time = (0.25TCY - TPD)

where TCY = instruction cycle
TPD = propagation delay

Therefore, at higher clock frequencies, a write followed by a read may be problematic.

NOTES:

5.0 TIMER0 MODULE

The Timer0 module has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt-on-overflow from FFh to 00h
- Edge select for external clock

Figure 5-1 is a simplified block diagram of the Timer0 module.

Timer mode is selected by clearing bit T0CS (OPTION<5>). 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 (Figure 5-2 and Figure 5-3). The user can work around this by writing an adjusted value to the TMR0 register.

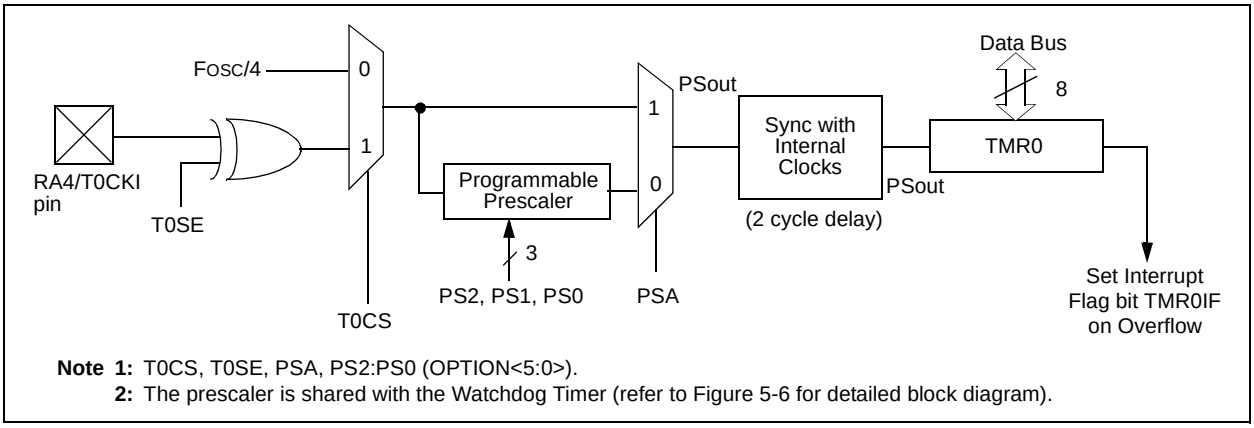
Counter mode is selected by setting bit T0CS (OPTION<5>). In Counter mode, Timer0 will increment either on every rising, or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (OPTION<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 5.2.

The prescaler is mutually exclusively shared between the Timer0 module and the Watchdog Timer. The prescaler assignment is controlled in software by control bit PSA (OPTION<3>). Clearing bit PSA will assign the prescaler to the Timer0 module. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4,..., 1:256 are selectable. Section 5.3 details the operation of the prescaler.

5.1 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit TMR0IF (INTCON<2>). The interrupt can be masked by clearing bit T0IE (INTCON<5>). Bit TMR0IF 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. Figure 5-4 displays the Timer0 interrupt timing.

FIGURE 5-1: TIMER0 BLOCK DIAGRAM



Note 1: T0CS, T0SE, PSA, PS2:PS0 (OPTION<5:0>).
2: The prescaler is shared with the Watchdog Timer (refer to Figure 5-6 for detailed block diagram).

FIGURE 5-2: TIMER0 TIMING: INTERNAL CLOCK/NO PRESCALE

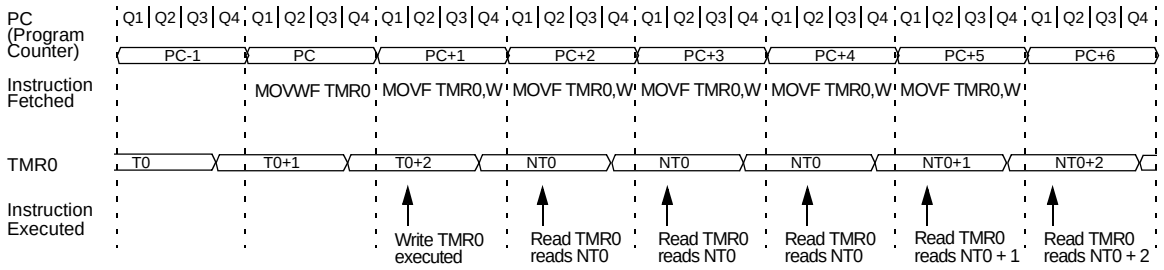


FIGURE 5-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

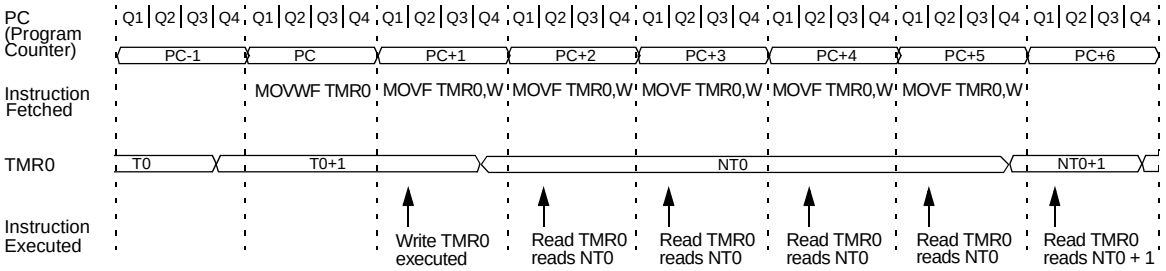
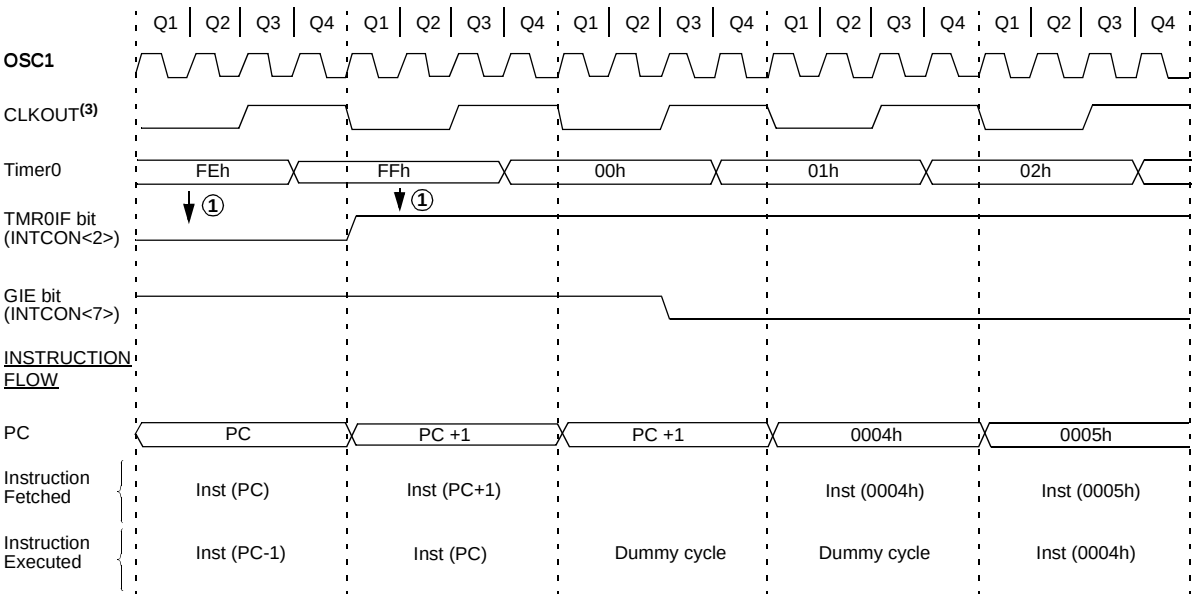


FIGURE 5-4: TIMER0 INTERRUPT TIMING



Note 1: Interrupt flag bit TMR0IF is sampled here (every Q1).
2: Interrupt latency = 4TCY where TCY = instruction cycle time.
3: CLKOUT is available only in RC oscillator mode.

5.2 Using Timer0 with an External Clock

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.

5.2.1 EXTERNAL CLOCK SYNCHRONIZATION

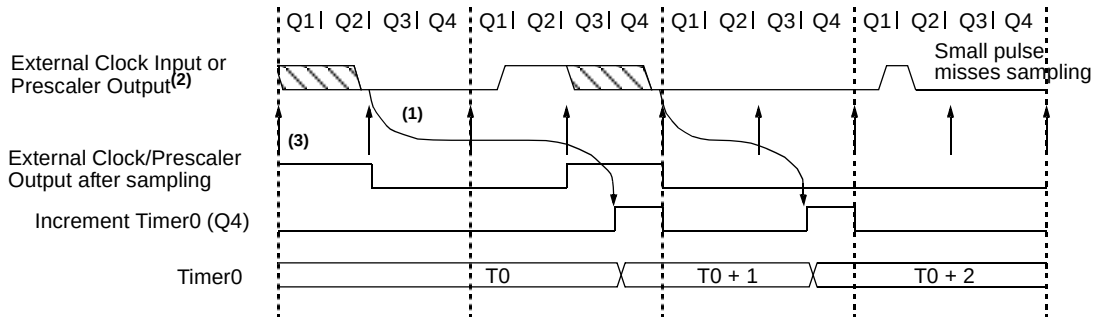
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 5-5). Therefore, it is necessary for T0CKI to be high for at least 2TOSC (and a small RC delay of 20 ns) and low for at least 2TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

When a prescaler is used, the external clock input is divided by the asynchronous ripple counter type prescaler, so that the prescaler output is symmetrical. For the external clock to meet the sampling requirement, the ripple counter must be taken into account. Therefore, it is necessary for T0CKI to have a period of at least 4TOSC (and a small RC delay of 40 ns) divided by the prescaler value. The only requirement on T0CKI high and low time is that they do not violate the minimum pulse width requirement of 10 ns. Refer to parameters 40, 41 and 42 in the electrical specification of the desired device.

5.2.2 TMR0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 module is actually incremented. Figure 5-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 5-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Note 1: Delay from clock input change to Timer0 increment is 3TOSC to 7TOSC. (Duration of Q = TOSC.) Therefore, the error in measuring the interval between two edges on Timer0 input = ±4TOSC max.

2: External clock if no prescaler selected, prescaler output otherwise.

3: The arrows indicate the points in time where sampling occurs.

5.3 Prescaler

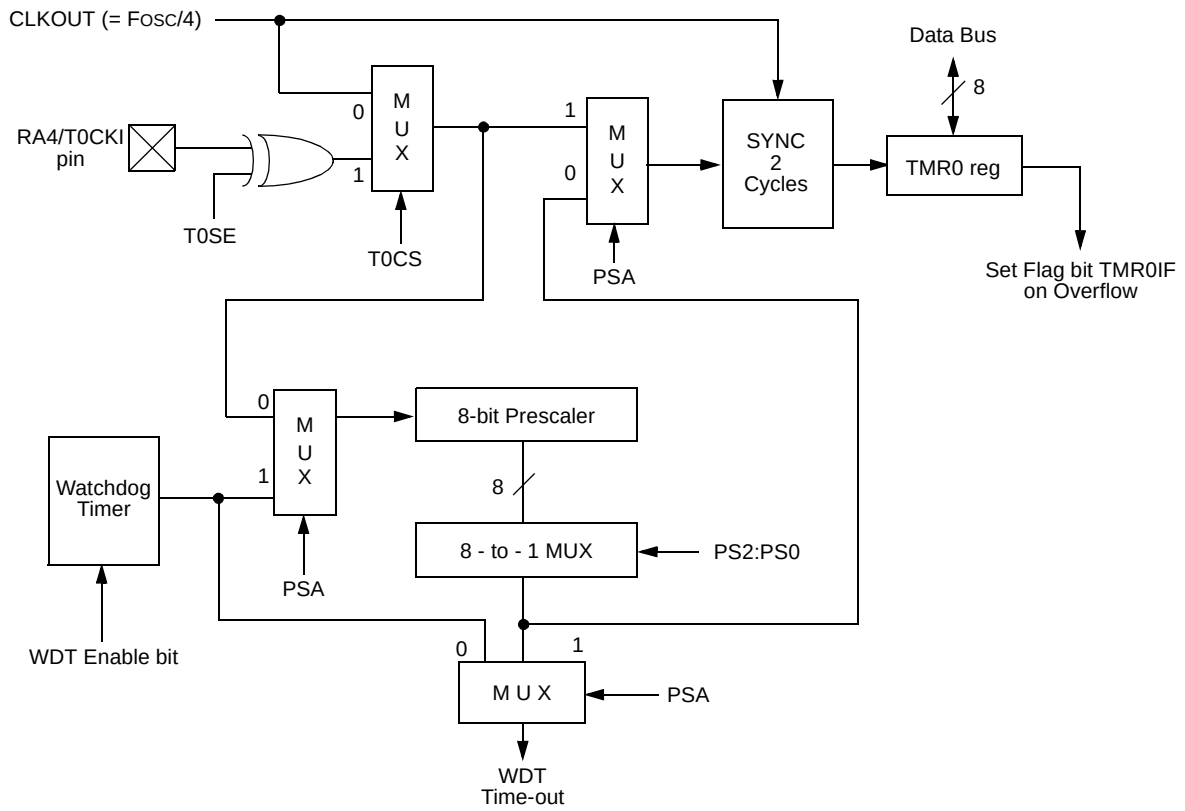
An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer (Figure 5-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that the prescaler may be used by either the Timer0 module or the WDT, but not both. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

The PSA and PS2:PS0 bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g. `CLRF 1, MOVWF 1, BSF 1,x`....etc.) will clear the prescaler count. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler count along with the Watchdog Timer. The prescaler is not readable or writable.

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

FIGURE 5-6: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



Note: T0CS, T0SE, PSA, PS2:PS0 are (OPTION<5:0>).

5.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control, i.e., it can be changed “on the fly” during program execution.

Note: To avoid an unintended device RESET, the following instruction sequence (shown in Example 5-1) must be executed when changing the prescaler assignment from Timer0 to the WDT. This precaution must be followed even if the WDT is disabled.

EXAMPLE 5-1: CHANGING PRESCALER (TIMER0→WDT)

Lines 2 and 3 do NOT have to be included if the final desired prescale value is other than 1:1. If 1:1 is final desired value, then a temporary prescale value is set in lines 2 and 3 and the final prescale value will be set in lines 10 and 11.

```
1)  BSF     STATUS, RP0    ;Select Bank1
2)  MOVLW   b'xx0x0xxx'    ;Select clock source and prescale value of
3)  MOVWF   OPTION_REG     ;other than 1:1
4)  BCF     STATUS, RP0    ;Select Bank0
5)  CLRF    TMR0           ;Clear TMR0 and prescaler
6)  BSF     STATUS, RP1    ;Select Bank1
7)  MOVLW   b'xxxx1xxx'    ;Select WDT, do not change prescale value
8)  MOVWF   OPTION_REG     ;
9)  CLRWDT                 ;Clears WDT and prescaler
10) MOVLW   b'xxxx1xxx'    ;Select new prescale value and WDT
11) MOVWF   OPTION_REG     ;
12) BCF     STATUS, RP0    ;Select Bank0
```

To change prescaler from the WDT to the Timer0 module use the precaution shown in Example 5-2.

EXAMPLE 5-2: CHANGING PRESCALER (WDT→TIMER0)

```
CLRWDT              ;Clear WDT and precaler
BSF     STATUS, RP0 ;Select Bank1
MOVLW   b'xxxx0xxx' ;Select TMR0,
                    ;new prescale value and
MOVWF   OPTION_REG  ;clock source
BCF     STATUS, RP0 ;Select Bank0
```

TABLE 5-1: REGISTERS ASSOCIATED WITH TIMER0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
01h, 101h	TMR0	Timer0 Module Register								xxxx xxxx	uuuu uuuu
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
81h, 181h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	—	—	PORTA Data Direction Control Register						--11 1111	--11 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'.
Shaded cells are not used by Timer0.

NOTES:

6.0 TIMER1 MODULE

Timer1 is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L), which are readable and writable. 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>).

Timer1 can operate in one of two modes:

- As a timer
- As a counter

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

In Timer mode, Timer1 increments every instruction cycle. In Counter mode, it increments on every rising edge of the external clock input.

Timer1 can be turned on and off using the control bit TMR1ON (T1CON<0>).

Timer1 also has an internal “RESET input”. This RESET can be generated by the CCP module (Section 8.0). Register 6-1 shows the Timer1 control register.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI and RC0/T1OSO/T1CKI pins become inputs, regardless of the TRISC<1:0>. RC1 and RC0 will be read as ‘0’.

REGISTER 6-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON
bit 7							bit 0

bit 7-6 **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 Control bit

1 = Oscillator is enabled
0 = Oscillator is shut-off

Note: The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **T1SYNC**: Timer1 External Clock Input Synchronization Control bit

TMR1CS = 1:

1 = Do not synchronize external clock input
0 = Synchronize external clock input

TMR1CS = 0:

This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS**: Timer1 Clock Source Select bit

1 = External clock from pin T1CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON**: Timer1 On bit

1 = Enables Timer1
0 = Stops Timer1

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

6.1 Timer1 Operation in Timer Mode

Timer mode is selected by clearing the TMR1CS (T1CON<1>) bit. In this mode, the input clock to the timer is FOSC/4. The synchronize control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) has no effect since the internal clock is always in sync.

6.2 Timer1 Operation in Synchronized Counter Mode

Counter mode is selected by setting bit TMR1CS. In this mode, the timer increments on every rising edge of clock input on pin RC1/T1OSI when bit T1OSCEN is set, or pin RC0/T1OSO/T1CKI when bit T1OSCEN is cleared.

If $\overline{\text{T1SYNC}}$ is cleared, then the external clock input is synchronized with internal phase clocks. The synchronization is done after the prescaler stage. The prescaler is an asynchronous ripple counter.

In this configuration, during SLEEP mode, Timer1 will not increment even if the external clock is present, since the synchronization circuit is shut-off. The prescaler however will continue to increment.

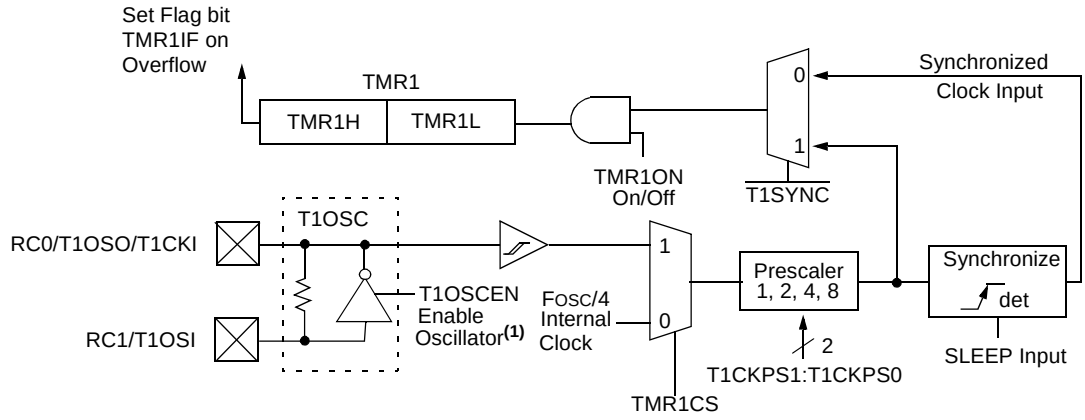
6.2.1 EXTERNAL CLOCK INPUT TIMING FOR SYNCHRONIZED COUNTER MODE

When an external clock input is used for Timer1 in Synchronized Counter mode, it must meet certain requirements. The external clock requirement is due to internal phase clock (TOSC) synchronization. Also, there is a delay in the actual incrementing of TMR1 after synchronization.

When the prescaler is 1:1, the external clock input is the same as the prescaler output. The synchronization of T1CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks. Therefore, it is necessary for T1CKI to be high for at least 2TOSC (and a small RC delay of 20 ns), and low for at least 2TOSC (and a small RC delay of 20 ns). Refer to the appropriate electrical specifications, parameters 45, 46, and 47.

When a prescaler other than 1:1 is used, the external clock input is divided by the asynchronous ripple counter type prescaler, so that the prescaler output is symmetrical. In order for the external clock to meet the sampling requirement, the ripple counter must be taken into account. Therefore, it is necessary for T1CKI to have a period of at least 4TOSC (and a small RC delay of 40 ns), divided by the prescaler value. The only requirement on T1CKI high and low time is that they do not violate the minimum pulse width requirements of 10 ns). Refer to the appropriate electrical specifications, parameters 40, 42, 45, 46, and 47.

FIGURE 6-1: TIMER1 BLOCK DIAGRAM



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

6.3 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during SLEEP and can generate an interrupt-on-overflow which will wake-up the processor. However, special precautions in software are needed to read from, or write to the Timer1 register pair (TMR1H:TMR1L) (Section 6.3.2).

In Asynchronous Counter mode, Timer1 cannot be used as a time-base for capture or compare operations.

6.3.1 EXTERNAL CLOCK INPUT TIMING WITH UNSYNCHRONIZED CLOCK

If control bit $\overline{\text{T1SYNC}}$ is set, the timer will increment completely asynchronously. The input clock must meet certain minimum high time and low time requirements, as specified in timing parameters 45, 46, and 47.

6.3.2 READING AND WRITING TMR1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L, while the timer is running from an external asynchronous clock, will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers while the register is incrementing. This may produce an unpredictable value in the timer register.

Reading the 16-bit value requires some care. Example 6-1 is an example routine to read the 16-bit timer value. This is useful if the timer cannot be stopped.

EXAMPLE 6-1: READING A 16-BIT FREE-RUNNING TIMER

```
; All interrupts are disabled
;
            MOVF   TMR1H, W    ;Read high byte
            MOVWF  TMPH        ;
            MOVF   TMR1L, W    ;Read low byte
            MOVWF  TMPL        ;
            MOVF   TMR1H, W    ;Read high byte
            SUBWF  TMPH,  W    ;Sub 1st read with 2nd read
            BTFSC  STATUS,Z    ;Is result = 0
            GOTO   CONTINUE    ;Good 16-bit read
;
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
;
            MOVF   TMR1H, W    ;Read high byte
            MOVWF  TMPH        ;
            MOVF   TMR1L, W    ;Read low byte
            MOVWF  TMPL        ;
; Re-enable the Interrupt (if required)
;
CONTINUE                       ;Continue with your code
```

6.4 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. Table 6-1 shows the capacitor selection for the Timer1 oscillator.

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

TABLE 6-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	33 pF	33 pF
	100 kHz	15 pF	15 pF
	200 kHz	15 pF	15 pF
These values are for design guidance only.			
Crystals Tested:			
32.768 kHz	Epson C-001R32.768K-A		± 20 PPM
100 kHz	Epson C-2 100.00 KC-P		± 20 PPM
200 kHz	STD XTL 200.000 kHz		± 20 PPM

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

2: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

6.5 Resetting Timer1 Using the CCP Trigger Output

If the CCP1 module is configured in Compare mode to generate a “special event trigger” (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1.

Note: The special event trigger from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode, to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L registers pair effectively become the period register for Timer1.

6.6 Resetting of Timer1 Register Pair (TMR1H:TMR1L)

TMR1H and TMR1L registers are not reset on a POR or any other RESET, except by the CCP1 special event trigger.

T1CON register is reset to 00h on a Power-on Reset. In any other RESET, the register is unaffected.

6.7 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

TABLE 6-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by theTimer1 module.

7.0 TIMER2 MODULE

Timer2 is an 8-bit timer with a prescaler and a postscaler. It can be used as the PWM time-base for the PWM mode of the CCP module. It can also be used as a time-base for the Master mode SPI clock. 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 Timer2 module has an 8-bit period register, PR2. TMR2 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 set during RESET.

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>)).

Timer2 can be shut-off by clearing control bit TMR2ON (T2CON<2>) to minimize power consumption.

Figure 7-1 shows the Timer2 control register.

7.1 Timer2 Prescaler and Postscaler

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

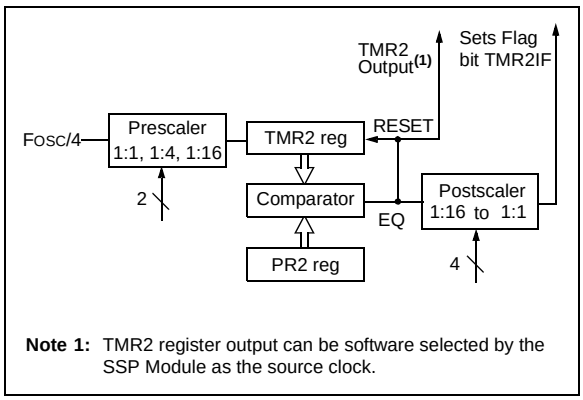
- a write to the TMR2 register
- a write to the T2CON register
- any device RESET (Power-on Reset, $\overline{\text{MCLR}}$ Reset, or Watchdog Timer Reset)

TMR2 will not clear when T2CON is written.

7.2 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 7-1: TIMER2 BLOCK DIAGRAM



Note 1: TMR2 register output can be software selected by the SSP Module as the source clock.

REGISTER 7-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

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

bit 6-3 **TOUTPS3:TOUTPS0**: 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

TABLE 7-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
11h	TMR2	Timer2 Module's Register								0000 0000	0000 0000
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
92h	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.

8.0 CAPTURE/COMPARE/PWM (CCP) MODULE

The CCP (Capture/Compare/PWM) module contains a 16-bit register which can operate as a 16-bit capture register, as a 16-bit compare register, or as a PWM master/slave duty cycle register. Table 8-1 shows the timer resources used by the CCP module.

The Capture/Compare/PWM Register1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. All three are readable and writable.

Register 8-1 shows the CCP1CON register.

For use of the CCP module, refer to the *Embedded Control Handbook,* "Using the CCP Modules" (AN594).

TABLE 8-1: CCP MODE - TIMER RESOURCE

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

REGISTER 8-1: CCP1CON REGISTER (ADDRESS 17h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

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

bit 5-4 **CCP1X:CCP1Y**: PWM Least Significant bits

Capture mode:
Unused

Compare mode:
Unused

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPR1L.

bit 3-0 **CCP1M3:CCP1M0**: CCP1 Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCP1 module)
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, set output on match (bit CCP1IF is set)
1001 = Compare mode, clear output on match (bit CCP1IF is set)
1010 = Compare mode, generate software interrupt-on-match (bit CCP1IF is set, CCP1 pin is unaffected)
1011 = Compare mode, trigger special event (CCP1IF bit is set; CCP1 resets TMR1)
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

8.1 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on pin RC2/CCP1 (Figure 8-1). An event can be selected to be one of the following:

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

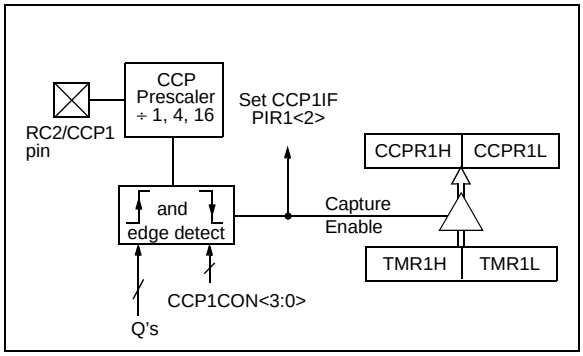
An event is selected by control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit CCP1IF (PIR1<2>) is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten with the new captured value.

8.1.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1 pin should be configured as an input by setting the TRISC<2> bit.

Note:	If the RC2/CCP1 pin is configured as an output, a write to the port can cause a capture condition.

FIGURE 8-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



8.1.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode, or Synchronized Counter mode, for the CCP module to use the capture feature. In Asynchronous Counter mode, the capture operation may not work.

8.1.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep enable bit CCP1IE (PIE1<2>) clear to avoid false interrupts and should clear flag bit CCP1IF following any such change in operating mode.

8.1.4 CCP PRESCALER

There are four prescaler settings, specified by bits CCP1M3:CCP1M0. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any RESET will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 8-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

EXAMPLE 8-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF   CCP1CON     ; Turn CCP module off
MOVLW  NEW_CAPT_PS ; Load the W reg with
                   ; the new prescaler
                   ; mode value and CCP ON
MOVWF  CCP1CON     ; Load CCP1CON with
                   ; this value
```

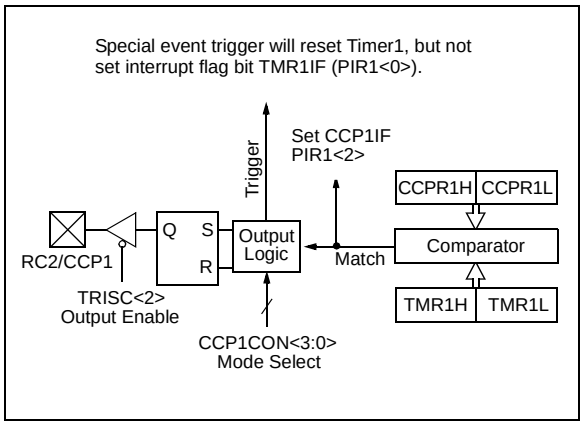

8.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the RC2/CCP1 pin is:

- Driven high
- Driven low
- Remains unchanged

The action on the pin is based on the value of control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). At the same time, a compare interrupt is also generated.

FIGURE 8-2: COMPARE MODE OPERATION BLOCK DIAGRAM



8.2.1 CCP PIN CONFIGURATION

The user must configure the RC2/CCP1 pin as an output by clearing the TRISC<2> bit.

Note:	Clearing the CCP1CON register will force the RC2/CCP1 compare output latch to the default low level. This is not the PORTC I/O data latch.

8.2.2 TIMER1 MODE SELECTION

Timer1 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.

8.2.3 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

8.2.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated which may be used to initiate an action.

The special event trigger output of CCP1 resets the TMR1 register pair and starts an A/D conversion. This allows the CCPR1H:CCPR1L register pair to effectively be a 16-bit programmable period register for Timer1.

Note:	The “special event trigger” from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

8.3 PWM Mode

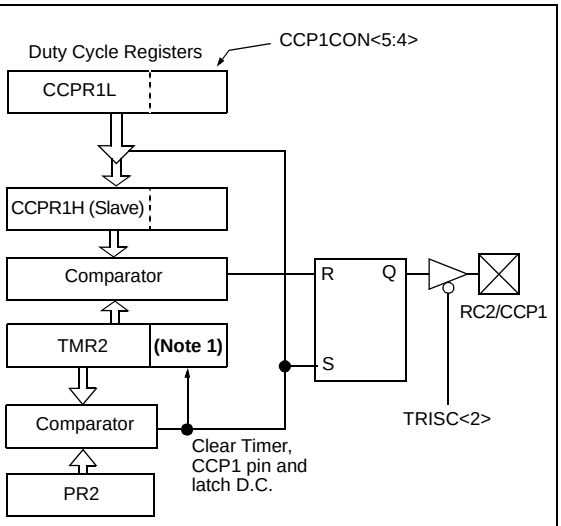
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 8-3 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 8.3.3.

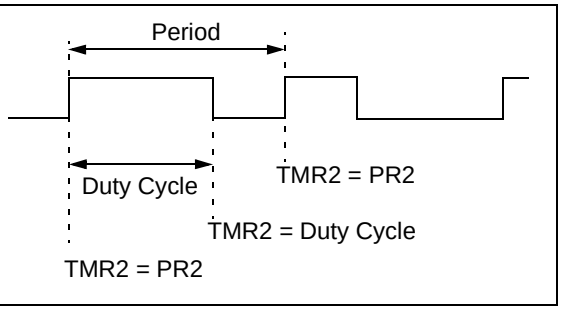
FIGURE 8-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time-base.

A PWM output (Figure 8-4) 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 8-4: PWM OUTPUT



8.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 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 is equal to PR2, the following three events occur on the next increment cycle:

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

Note:	The Timer2 postscaler (Section 7.0) 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.

8.3.2 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 CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (CCPR1L{:}CCP1CON\langle 5{:}4\rangle) \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

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

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This 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 2 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.

EQUATION 8-1: EXAMPLES OF PWM PERIOD AND DUTY CYCLE CALCULATION

1. Find the value of the PR2 register, given:
   - Desired PWM frequency = 31.25 kHz
   - FOSC = 8 MHz
   - TMR2 prescale = 1

   From the equation for PWM period in Section 8.3.1,

   $1 / 31.25\ kHz = [(PR2) + 1] \bullet 4 \bullet 1/8\ MHz \bullet 1$

   or

   $32\ \mu s = [(PR2) + 1] \bullet 4 \bullet 125\ ns \bullet 1 = [(PR2) + 1] \bullet 0.5\ \mu s$

   $PR2 = (32\ \mu s / 0.5\ \mu s) - 1$

   $PR2 = 63$

2. Find the maximum resolution of the duty cycle that can be used with a 31.25 kHz frequency and 8 MHz oscillator.

   From the equation from maximum PWM resolution in Section 8.3.2,

   $1 / 31.25\ kHz = 2^{PWM\ RESOLUTION} \bullet 1 / 8\ MHz \bullet 1$

   or

   $32\ \mu s = 2^{PWM\ RESOLUTION} \bullet 125\ ns \bullet 1$

   $256 = 2^{PWM\ RESOLUTION}$

   $\log(256) = (PWM\ Resolution) \bullet \log(2)$

   $8.0 = PWM\ Resolution$

At most, an 8-bit resolution duty cycle can be obtained from a 31.25 kHz frequency and a 8 MHz oscillator, i.e., $0 \le$ CCPR1L:CCP1CON<5:4> ≤ 255. Any value greater than 255 will result in a 100% duty cycle.

In order to achieve higher resolution, the PWM frequency must be decreased. In order to achieve higher PWM frequency, the resolution must be decreased.

Table 8-2 lists example PWM frequencies and resolutions for FOSC = 8 MHz. TMR2 prescaler and PR2 values are also shown.

8.3.3 SET-UP FOR PWM OPERATION

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

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP module for PWM operation.

TABLE 8-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 8 MHz

PWM Frequency	488 Hz	1.95 kHz	7.81 kHz	31.25 kHz	62.5 kHz	250 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x07
Maximum Resolution (bits)	10	10	10	8	7	5

TABLE 8-3: REGISTERS ASSOCIATED WITH TIMER1, CAPTURE AND COMPARE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
87h	TRISC	—	—	PORTC Data Direction Control Register						--11 1111	--11 1111
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu
15h	CCPR1L	Capture/Compare/PWM1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used in these modes.

TABLE 8-4: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
87h	TRISC	—	—	PORTC Data Direction Control Register						--11 1111	--11 1111
11h	TMR2	Timer2 Module Register								0000 0000	0000 0000
92h	PR2	Timer2 Module Period Register								1111 1111	1111 1111
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
15h	CCPR1L	Capture/Compare/PWM1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used in this mode.

9.0 SYNCHRONOUS SERIAL PORT (SSP) MODULE

The Synchronous Serial Port (SSP) 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 SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI™)
- Inter-Integrated Circuit (I^2C™)

Refer to Application Note AN578, *"Use of the SSP Module in the I^2C Multi-Master Environment."*

REGISTER 9-1: SSPSTAT: SERIAL PORT STATUS REGISTER (ADDRESS 94h)

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:** SPI Data Input Sample Phase 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 (see Figure 9-3, Figure 9-4, and Figure 9-5)
CKP = 0:
1 = Data transmitted on rising edge of SCK
0 = Data transmitted on falling edge of SCK
CKP = 1:
1 = Data transmitted on falling edge of SCK
0 = Data transmitted on rising edge of SCK

bit 5 **D/$\overline{A}$:** Data/$\overline{Address}$ bit (I^2C mode only)
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 (I^2C mode only. This bit is cleared when the SSP module is disabled, or when the START bit was detected last.)
1 = Indicates that a STOP bit has been detected last (this bit is '0' on RESET)
0 = STOP bit was not detected last

bit 3 **S:** START bit (I^2C mode only. This bit is cleared when the SSP module is disabled, or when the STOP bit was detected last.)
1 = Indicates that a START bit has been detected last (this bit is '0' on RESET)
0 = START bit was not detected last

bit 2 **R/$\overline{W}$**: Read/Write bit Information (I^2C mode only)
This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next START bit, STOP bit, or $\overline{ACK}$ bit.
1 = Read
0 = Write

bit 1 **UA**: Update Address (10-bit I^2C 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
Receive (SPI and I^2C modes):
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty
Transmit (I^2C mode only)
1 = Transmit in progress, SSPBUF is full
0 = Transmit 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 9-2: SSPCON: SYNC SERIAL PORT CONTROL REGISTER (ADDRESS 14h)

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
1 = SSPBUF register is written while still transmitting the previous word (must be cleared in software)
0 = No collision

bit 6 **SSPOV**: Receive Overflow Indicator bit
In SPI mode:
1 = A new byte is received while SSPBUF is holding previous data. Data in SSPSR is lost on overflow. Overflow only occurs in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflows. In Master mode, the overflow bit is not set since each operation is initiated by writing to the SSPBUF register. (Must be cleared in software.)
0 = No overflow
In I^2C mode:
1 = A byte is received while the SSPBUF is holding the previous byte. SSPOV is a “don’t care” in transmit mode. (Must be cleared in software.)
0 = No overflow

bit 5 **SSPEN**: Synchronous Serial Port Enable bit
In SPI mode:
When enabled, these pins must be properly configured as input or output.
1 = Enables serial port and configures SCK, SDO, SDI, and $\overline{SS}$ as the source of the serial port pins
0 = Disables serial port and configures these pins as I/O port pins
In I^2C mode:
When enabled, these pins must be properly configured as input or output.
1 = Enables the serial port and configures the SDA and SCL pins as the source of the serial port pins
0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP**: Clock Polarity Select bit
In SPI mode:
1 = Idle state for clock is a high level
0 = Idle state for clock is a low level
In I^2C mode:
SCK release control
1 = Enable clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)

bit 3-0 **SSPM3:SSPM0**: Synchronous Serial Port Mode Select bits
0000 = SPI Master mode, clock = FOSC/4
0001 = SPI Master mode, clock = FOSC/16
0010 = SPI Master mode, clock = FOSC/64
0011 = SPI Master mode, clock = TMR2 output/2
0100 = SPI Slave mode, clock = SCK pin ($\overline{SS}$ pin control enabled)
0101 = SPI Slave mode, clock = SCK pin ($\overline{SS}$ pin control disabled, $\overline{SS}$ can be used as I/O pin)
0110 = I^2C Slave mode, 7-bit address
0111 = I^2C Slave mode, 10-bit address
1011 = I^2C firmware controlled Master mode (slave idle)
1110 = I^2C firmware controlled Master mode, 7-bit address with START and STOP bit interrupts enabled
1111 = I^2C firmware controlled Master mode, 10-bit address with START and STOP bit interrupts enabled
1000, 1001, 1010, 1100, 1101 = reserved

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

9.1 SPI Mode

The SPI mode allows 8-bits of data to be synchronously transmitted and received simultaneously. To accomplish communication, typically three pins are used:

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

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

- Slave Select ($\overline{SS}$) RA5/AN4/$\overline{SS}$

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON register (SSPCON<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)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

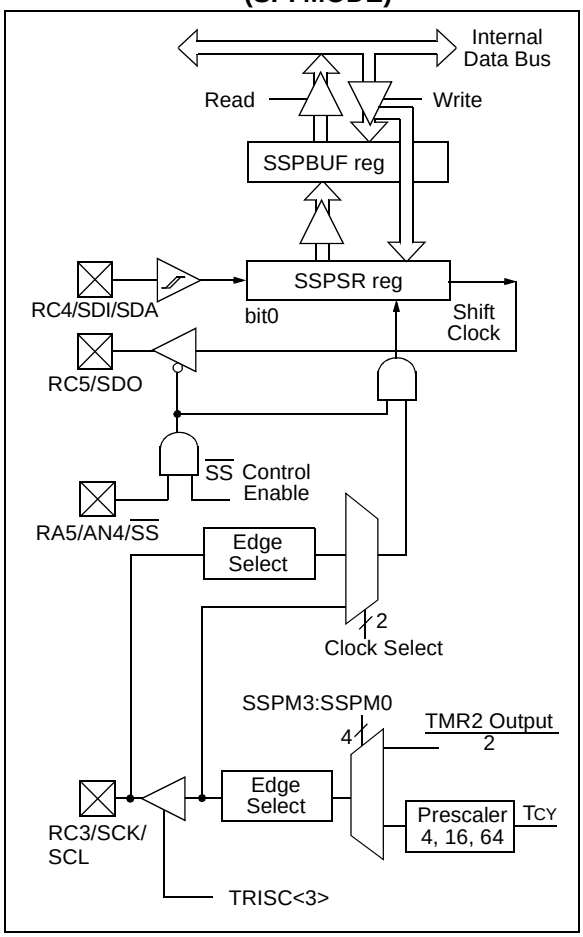
The SSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR, until the received data is ready. Once the 8-bits of data have been received, that byte is moved to the SSPBUF register. Then, the buffer full detect bit, BF (SSPSTAT<0>), and interrupt flag bit, SSPIF (PIR1<3>), 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 (SSPCON<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, bit BF is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the SSP 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 9-1 shows the loading of the SSPBUF (SSPSR) for data transmission. The `MOVWF RXDATA` instruction (shaded) is only required if the received data is meaningful.

EXAMPLE 9-1: LOADING THE SSPBUF (SSPSR) REGISTER

```
        BCF   STATUS, RP1  ;Select Bank1
        BSF   STATUS, RP0  ;
LOOP    BTFSS SSPSTAT, BF  ;Has data been
                           ;received
                           ;(transmit
                           ;complete)?
        GOTO  LOOP         ;No
        BCF   STATUS, RP0  ;Select Bank0
        MOVF  SSPBUF, W    ;W reg = contents
                           ;of SSPBUF
        MOVWF RXDATA       ;Save in user RAM
        MOVF  TXDATA, W    ;W reg = contents
                           ; of TXDATA
        MOVWF SSPBUF       ;New data to xmit
```

The block diagram of the SSP module, when in SPI mode (Figure 9-1), shows that the SSPSR is not directly readable or writable, and can only be accessed from addressing the SSPBUF register. Additionally, the SSP status register (SSPSTAT) indicates the various status conditions.

FIGURE 9-1: SSP BLOCK DIAGRAM (SPI MODE)



To enable the serial port, SSP enable bit, SSPEN (SSPCON<5>) must be set. To reset or reconfigure SPI mode, clear bit SSPEN, re-initialize the SSPCON register, and then set bit SSPEN. This configures the SDI, SDO, SCK, and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, they must have their data direction bits (in the TRISC register) appropriately programmed. That is:

- SDI must have TRISC<4> set
- SDO must have TRISC<5> cleared
- SCK (Master mode) must have TRISC<3> cleared
- SCK (Slave mode) must have TRISC<3> set
- $\overline{SS}$ must have TRISA<5> set and ADCON must be configured such that RA5 is a digital I/O

Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value. An example would be in Master mode, where you are only sending data (to a display driver), then both SDI and $\overline{SS}$ could be used as general purpose outputs by clearing their corresponding TRIS register bits.

Figure 9-2 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 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

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2) is to broadcast data by the firmware 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 SCK 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.

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the interrupt flag bit SSPIF (PIR1<3>) is set.

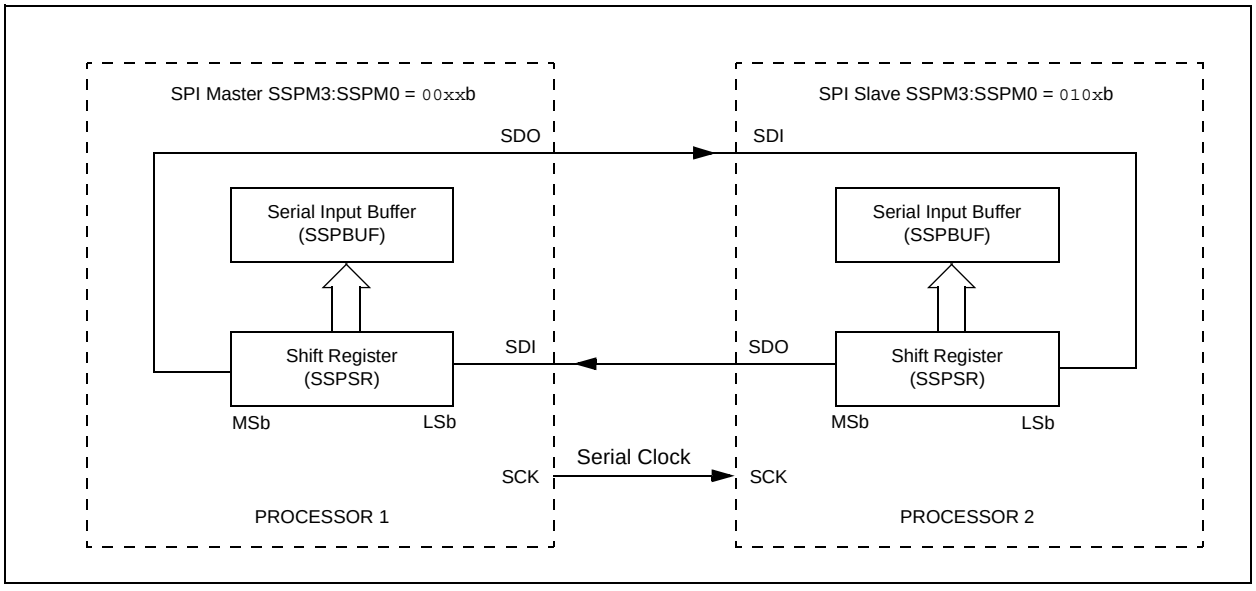
The clock polarity is selected by appropriately programming bit CKP (SSPCON<4>). This then, would give waveforms for SPI communication as shown in Figure 9-3, Figure 9-4, and Figure 9-5, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

This allows a maximum bit clock frequency (at 8 MHz) of 2 MHz. When in Slave mode, the external clock must meet the minimum high and low times.

In SLEEP mode, the slave can transmit and receive data and wake the device from SLEEP.

FIGURE 9-2: SPI MASTER/SLAVE CONNECTION



The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode (SSPCON<3:0> = 04h) and the TRISA<5> bit must be set for the Synchronous Slave mode to be enabled. 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 = '1', then the $\overline{SS}$ pin control must be enabled.

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 9-3: SPI MODE TIMING, MASTER MODE

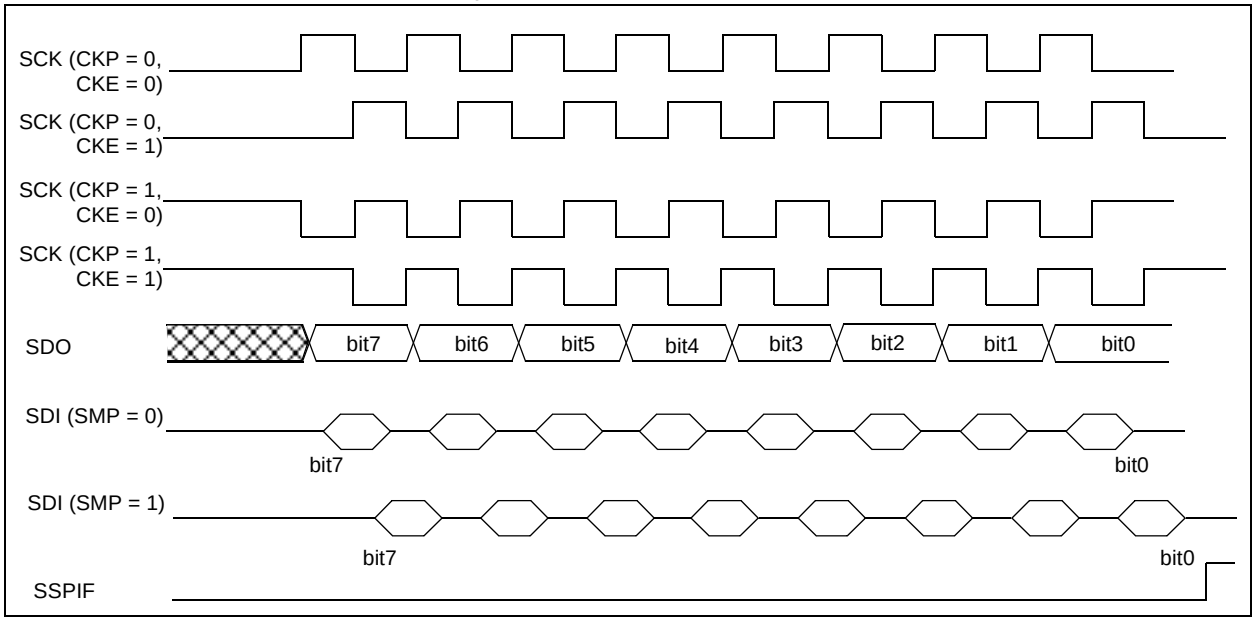


FIGURE 9-4: SPI MODE TIMING (SLAVE MODE WITH CKE = 0)

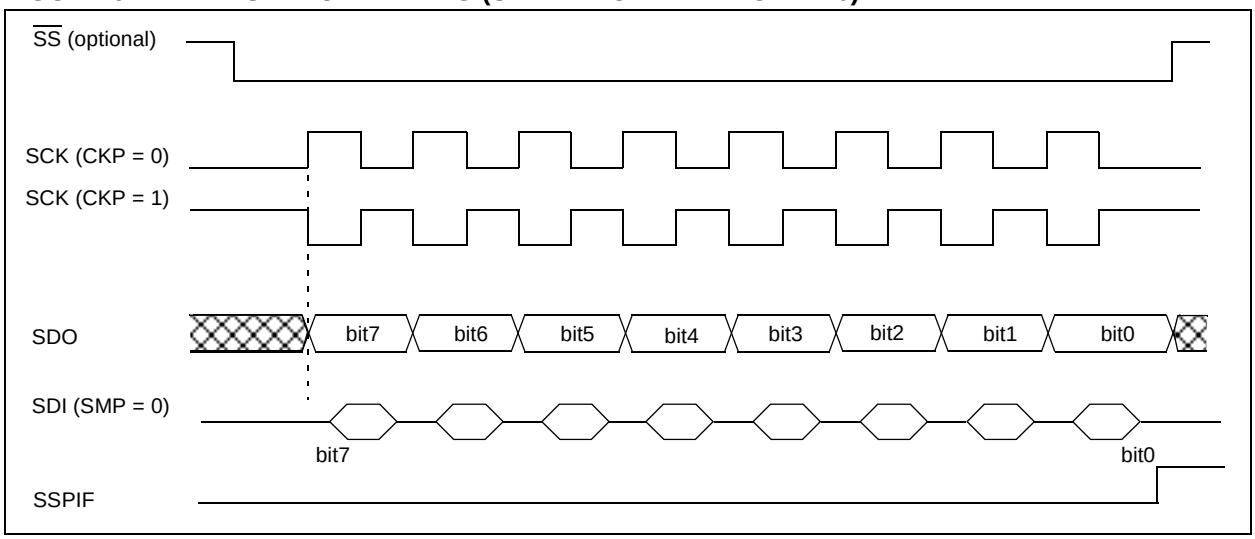


FIGURE 9-5: SPI MODE TIMING (SLAVE MODE WITH CKE = 1)

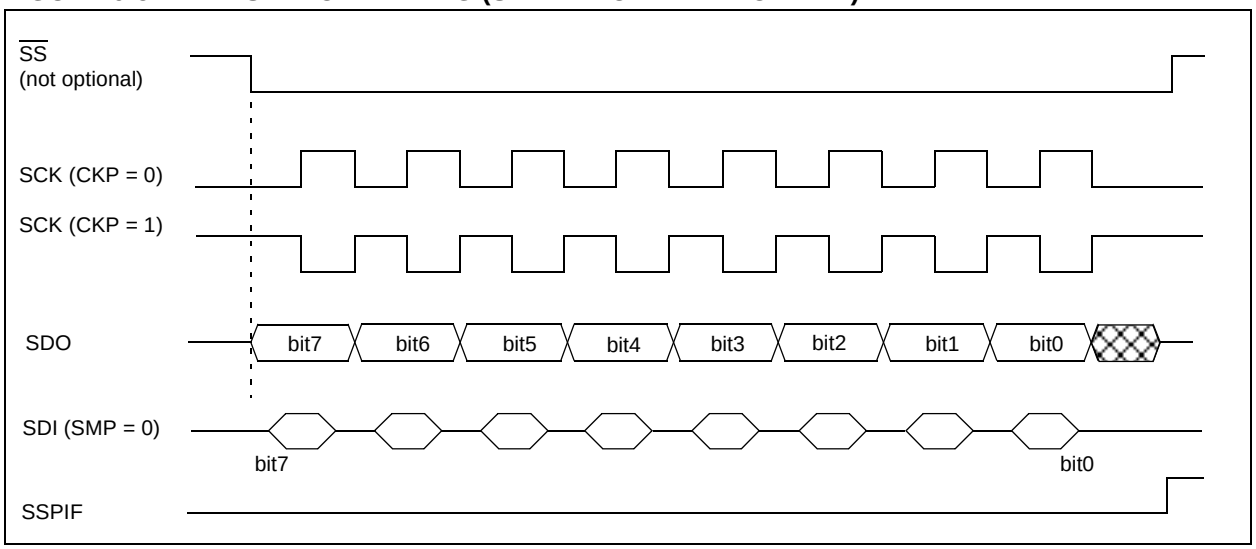


TABLE 9-1: REGISTERS ASSOCIATED WITH SPI OPERATION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
85h	TRISA	—	—	PORTA Data Direction Control Register						--11 1111	--11 1111
87h	TRISC	—	—	PORTC Data Direction Control Register						--11 1111	--11 1111
94h	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 SSP in SPI mode.

9.2 I²C Overview

This section provides an overview of the Inter-Integrated Circuit (I^2C) bus, with Section 9.3 discussing the operation of the SSP module in I^2C mode.

The I^2C bus is a two-wire serial interface developed by the Philips Corporation. The original specification, or standard mode, was for data transfers of up to 100 Kbps. An enhanced specification, or fast mode is not supported. This device will communicate with fast mode devices if attached to the same bus.

The I^2C interface employs a comprehensive protocol to ensure reliable transmission and reception of data. When transmitting data, one device is the "master" which initiates transfer on the bus and generates the clock signals to permit that transfer, while the other device(s) acts as the "slave." All portions of the slave protocol are implemented in the SSP module's hardware, except general call support, while portions of the master protocol need to be addressed in the PIC16CXXX software. Table 9-2 defines some of the I^2C bus terminology. For additional information on the I^2C interface specification, refer to the Philips document #939839340011, "*The I^2C bus and how to use it",* which can be obtained from the Philips Corporation.

In the I^2C interface protocol, each device has an address. When a master wishes to initiate a data transfer, it first transmits the address of the device that it wishes to "talk" to. All devices "listen" to see if this is their address. Within this address, a bit specifies if the master wishes to read from/write to the slave device. The master and slave are always in opposite modes (transmitter/receiver) of operation during a data transfer. That is, they can be thought of as operating in either of these two relations:

- Master-transmitter and Slave-receiver
- Slave-transmitter and Master-receiver

In both cases, the master generates the clock signal.

The output stages of the clock (SCL) and data (SDA) lines must have an open drain or open collector, in order to perform the wired-AND function of the bus. External pull-up resistors are used to ensure a high level when no device is pulling the line down. The number of devices that may be attached to the I^2C bus is limited only by the maximum bus loading specification of 400 pF.

9.2.1 INITIATING AND TERMINATING DATA TRANSFER

During times of no data transfer (idle time), both the clock line (SCL) and the data line (SDA) are pulled high through the external pull-up resistors. The START and STOP conditions determine the start and stop of data transmission. The START condition is defined as a high to low transition of the SDA when the SCL is high. The STOP condition is defined as a low to high transition of the SDA when the SCL is high. Figure 9-6 shows the START and STOP conditions. The master generates these conditions for starting and terminating data transfer. Due to the definition of the START and STOP conditions, when data is being transmitted, the SDA line can only change state when the SCL line is low.

FIGURE 9-6: START AND STOP CONDITIONS

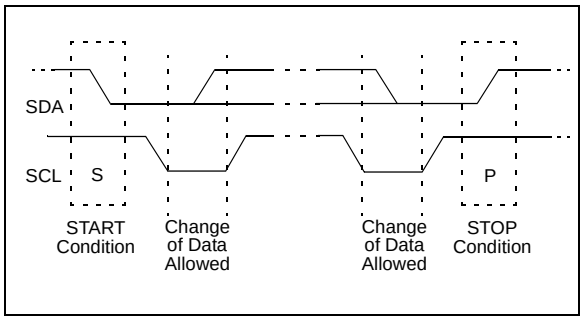


TABLE 9-2: I²C BUS TERMINOLOGY

Term	Description
Transmitter	The device that sends the data to the bus.
Receiver	The device that receives the data from the bus.
Master	The device which initiates the transfer, generates the clock and terminates the transfer.
Slave	The device addressed by a master.
Multi-master	More than one master device in a system. These masters can attempt to control the bus at the same time without corrupting the message.
Arbitration	Procedure that ensures that only one of the master devices will control the bus. This ensures that the transfer data does not get corrupted.
Synchronization	Procedure where the clock signals of two or more devices are synchronized.

9.2.2 ADDRESSING I^2C DEVICES

There are two address formats. The simplest is the 7-bit address format with a R/$\overline{W}$ bit (Figure 9-7). The more complex is the 10-bit address with a R/$\overline{W}$ bit (Figure 9-8). For 10-bit address format, two bytes must be transmitted with the first five bits specifying this to be a 10-bit address.

FIGURE 9-7: 7-BIT ADDRESS FORMAT

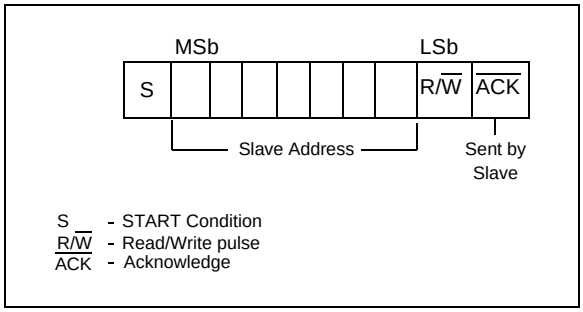
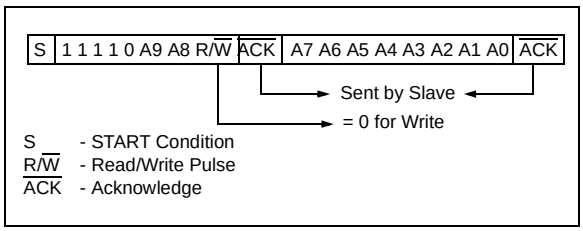


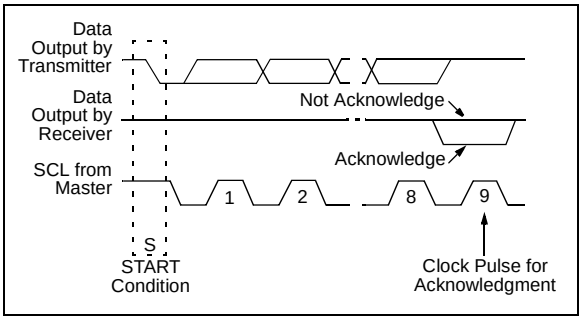
FIGURE 9-8: I^2C 10-BIT ADDRESS FORMAT



9.2.3 TRANSFER ACKNOWLEDGE

All data must be transmitted per byte, with no limit to the number of bytes transmitted per data transfer. After each byte, the slave-receiver generates an Acknowledge bit ($\overline{ACK}$) (see Figure 9-9). When a slave-receiver doesn't acknowledge the slave address or received data, the master must abort the transfer. The slave must leave SDA high so that the master can generate the STOP condition (Figure 9-6).

FIGURE 9-9: SLAVE-RECEIVER ACKNOWLEDGE



If the master is receiving the data (master-receiver), it generates an Acknowledge signal for each received byte of data, except for the last byte. To signal the end of data to the slave-transmitter, the master does not generate an Acknowledge (Not Acknowledge). The slave then releases the SDA line so the master can generate the STOP condition. The master can also generate the STOP condition during the Acknowledge pulse for valid termination of data transfer.

If the slave needs to delay the transmission of the next byte, holding the SCL line low will force the master into a wait state. Data transfer continues when the slave releases the SCL line. This allows the slave to move the received data, or fetch the data it needs to transfer before allowing the clock to start. This wait state technique can also be implemented at the bit level, Figure 9-10. The slave will inherently stretch the clock when it is a transmitter, but will not when it is a receiver. The slave will have to clear the SSPCON<4> bit to enable clock stretching when it is a receiver.

FIGURE 9-10: DATA TRANSFER WAIT STATE

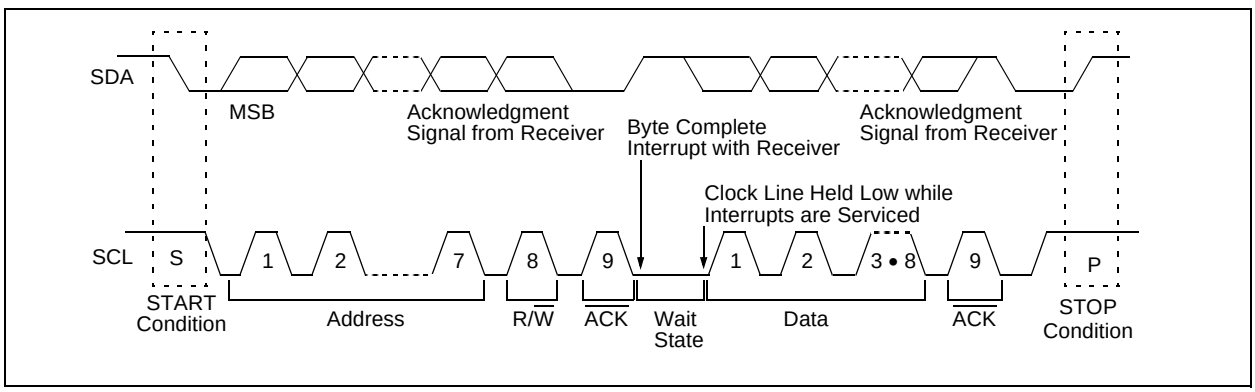


Figure 9-11 and Figure 9-12 show master-transmitter and Master-receiver data transfer sequences.

When a master does not wish to relinquish the bus (by generating a STOP condition), a Repeated START condition (Sr) must be generated. This condition is identical to the START condition (SDA goes high-to-low while SCL is high), but occurs after a data transfer Acknowledge pulse (not the bus-free state). This allows a master to send “commands” to the slave and then receive the requested information, or to address a different slave device. This sequence is shown in Figure 9-13.

FIGURE 9-11: MASTER-TRANSMITTER SEQUENCE

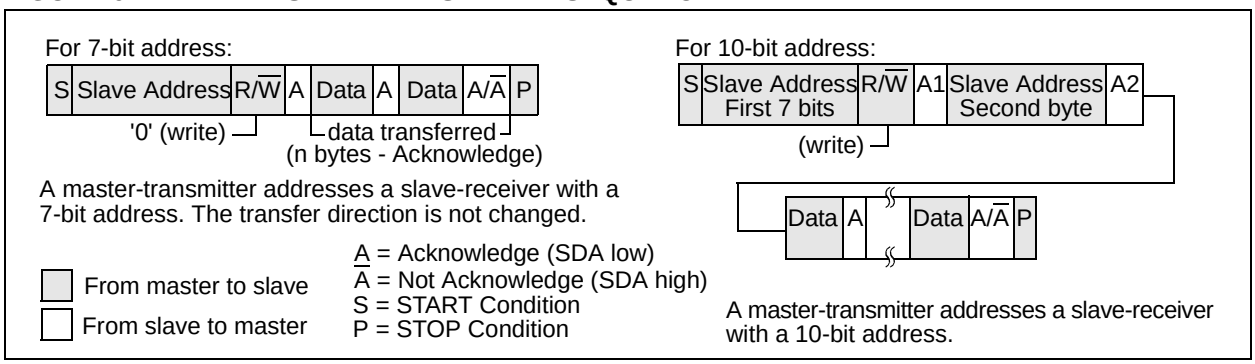


FIGURE 9-12: MASTER-RECEIVER SEQUENCE

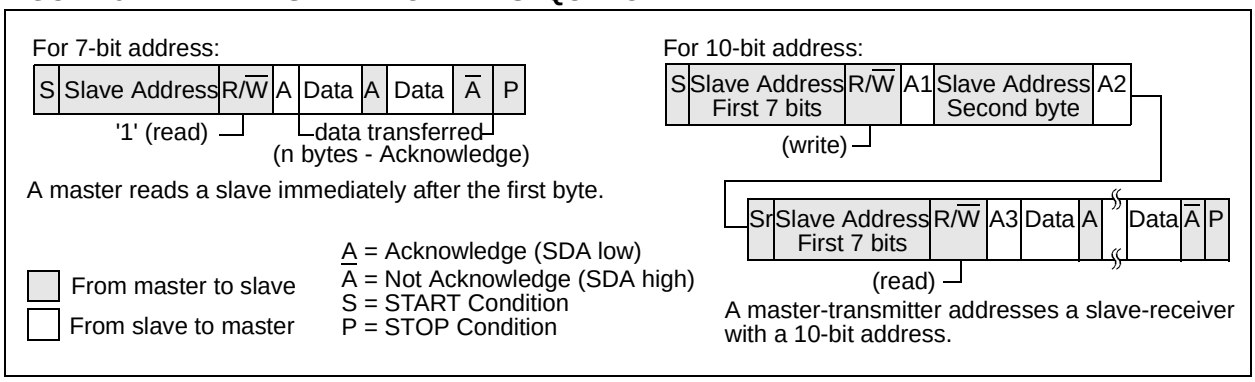
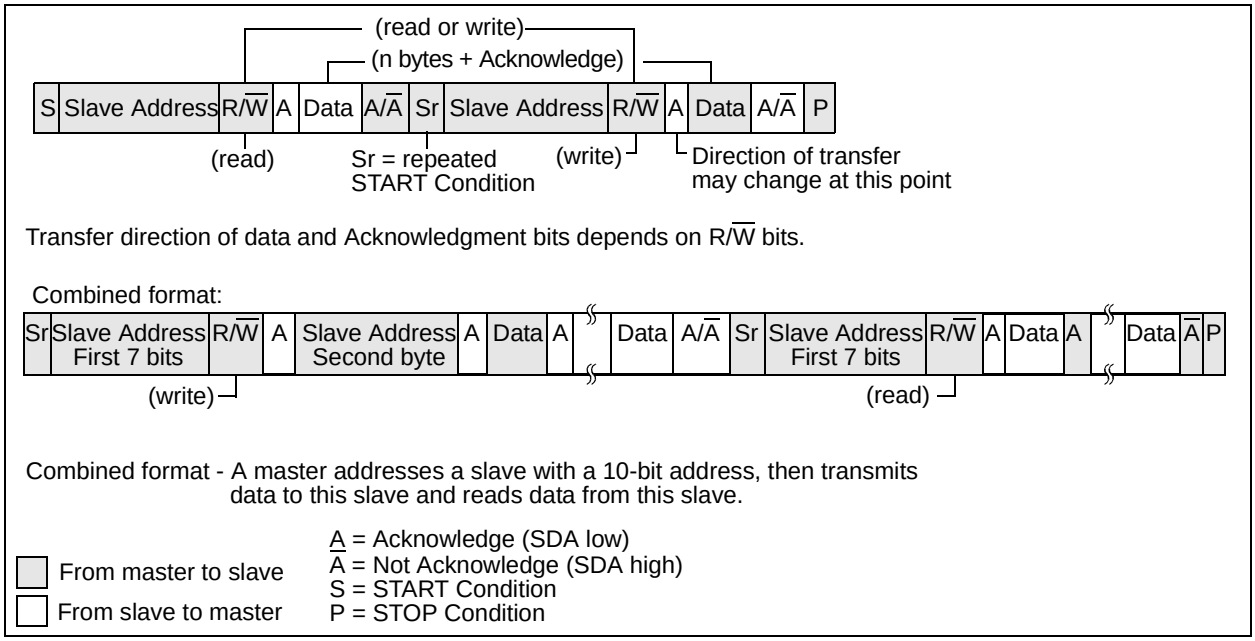


FIGURE 9-13: COMBINED FORMAT



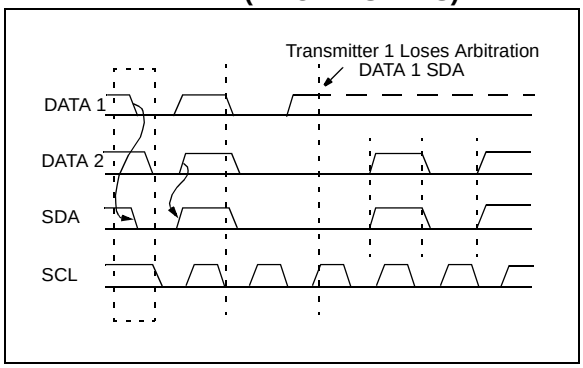
9.2.4 MULTI-MASTER

The I^2C protocol allows a system to have more than one master. This is called multi-master. When two or more masters try to transfer data at the same time, arbitration and synchronization occur.

9.2.4.1 Arbitration

Arbitration takes place on the SDA line, while the SCL line is high. The master, which transmits a high when the other master transmits a low, loses arbitration (Figure 9-14) and turns off its data output stage. A master, which lost arbitration can generate clock pulses until the end of the data byte where it lost arbitration. When the master devices are addressing the same device, arbitration continues into the data.

FIGURE 9-14: MULTI-MASTER ARBITRATION (TWO MASTERS)



Masters that also incorporate the slave function and have lost arbitration, must immediately switch over to Slave-Receiver mode. This is because the winning master-transmitter may be addressing it.

Arbitration is not allowed between:

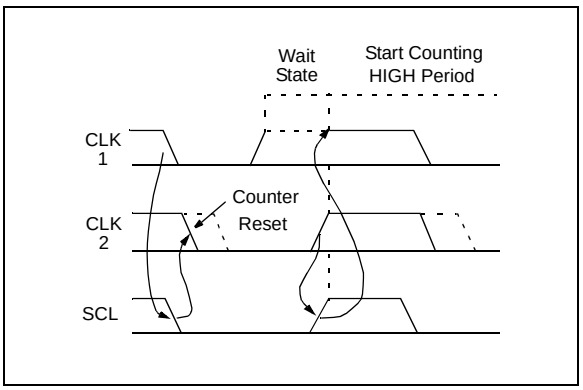
- A Repeated START condition
- A STOP condition and a data bit
- A Repeated START condition and a STOP condition

Care needs to be taken to ensure that these conditions do not occur.

9.2.4.2 Clock Synchronization

Clock synchronization occurs after the devices have started arbitration. This is performed using a wired-AND connection to the SCL line. A high to low transition on the SCL line causes the concerned devices to start counting off their low period. Once a device clock has gone low, it will hold the SCL line low until its SCL high state is reached. The low to high transition of this clock may not change the state of the SCL line, if another device clock is still within its low period. The SCL line is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state, until the SCL line comes high. When the SCL line comes high, all devices start counting off their high periods. The first device to complete its high period will pull the SCL line low. The SCL line high time is determined by the device with the shortest high period, Figure 9-15.

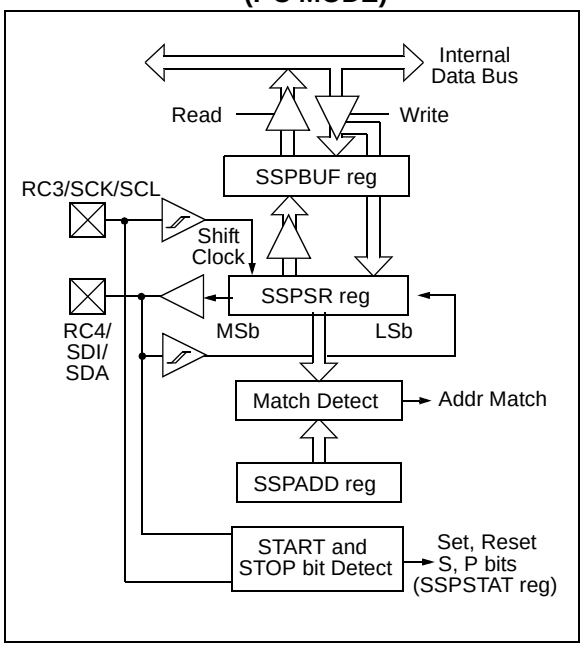
FIGURE 9-15: CLOCK SYNCHRONIZATION



9.3 SSP I²C Operation

The SSP module in I²C mode fully implements all slave functions, except general call support, and provides interrupts on START and STOP bits in hardware to facilitate firmware implementations of the master functions. The SSP module implements the standard mode specifications as well as 7-bit and 10-bit addressing. Two pins are used for data transfer. These are the RC3/SCK/SCL pin, which is the clock (SCL), and the RC4/SDI/SDA pin, which is the data (SDA). The user must configure these pins as inputs or outputs through the TRISC<4:3> bits. The SSP module functions are enabled by setting SSP enable bit, SSPEN (SSPCON<5>).

FIGURE 9-16: SSP BLOCK DIAGRAM (I²C MODE)



The SSP module has five registers for I²C operation. These are the:

- SSP Control Register (SSPCON)
- SSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- SSP Shift Register (SSPSR) - Not directly accessible
- SSP Address Register (SSPADD)

The SSPCON register allows control of the I²C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I²C modes to be selected:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address), with START and STOP bit interrupts enabled
- I²C Slave mode (10-bit address), with START and STOP bit interrupts enabled
- I²C Firmware controlled Master mode, slave is idle

Selection of any I²C 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.

The SSPSTAT register gives the status of the data transfer. This information includes detection of a START or STOP bit, specifies if the received byte was data or address, if the next byte is the completion of 10-bit address, and if this will be a read or write data transfer. The SSPSTAT register is read only.

The SSPBUF is the register to which transfer data is written to or read from. The SSPSR register shifts the data in or out of the device. In receive operations, the SSPBUF and SSPSR create a doubled buffered receiver. This allows reception of the next byte to begin before reading the last byte of received data. When the complete byte is received, it is transferred to the SSPBUF register and flag bit SSPIF is set. If another complete byte is received before the SSPBUF register is read, a receiver overflow has occurred and bit SSPOV (SSPCON<6>) is set and the byte in the SSPSR is lost.

The SSPADD register holds the slave address. In 10-bit mode, the user needs to write the high byte of the address (`1111 0 A9 A8 0`). Following the high byte address match, the low byte of the address needs to be loaded (A7:A0).

9.3.1 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISC<4:3> set). The SSP module will override the input state with the output data when required (slave-transmitter).

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 then load the SSPBUF register with the received value currently in the SSPSR register.

There are certain conditions that will cause the SSP module not to give this $\overline{ACK}$ pulse. These are if either (or both):

a) The buffer full bit BF (SSPSTAT<0>) was set before the transfer was received.

b) 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. Table 9-3 shows what happens when a data transfer byte is received, given the status of bits BF and SSPOV. The shaded cells show the condition where user software did not properly clear the overflow condition. Flag bit BF is cleared by reading the SSPBUF register while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low time for proper operation. The high and low times of the I^2C specification as well as the requirement of the SSP module is shown in timing parameter #100 and parameter #101.

9.3.1.1 Addressing

Once the SSP 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:

a) The SSPSR register value is loaded into the SSPBUF register.

b) The buffer full bit, BF is set.

c) An $\overline{ACK}$ pulse is generated.

d) SSP 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 (Figure 9-8). 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 '1111 0 A9 A8 0', where A9 and A8 are the two MSbs of the address. The sequence of events for a 10-bit address is as follows, with steps 7- 9 for 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.

TABLE 9-3: DATA TRANSFER RECEIVED BYTE ACTIONS

Status Bits as Data Transfer is Received		SSPSR → SSPBUF	Generate $\overline{ACK}$ Pulse	Set bit SSPIF (SSP Interrupt occurs if enabled)
BF	SSPOV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	No	No	Yes

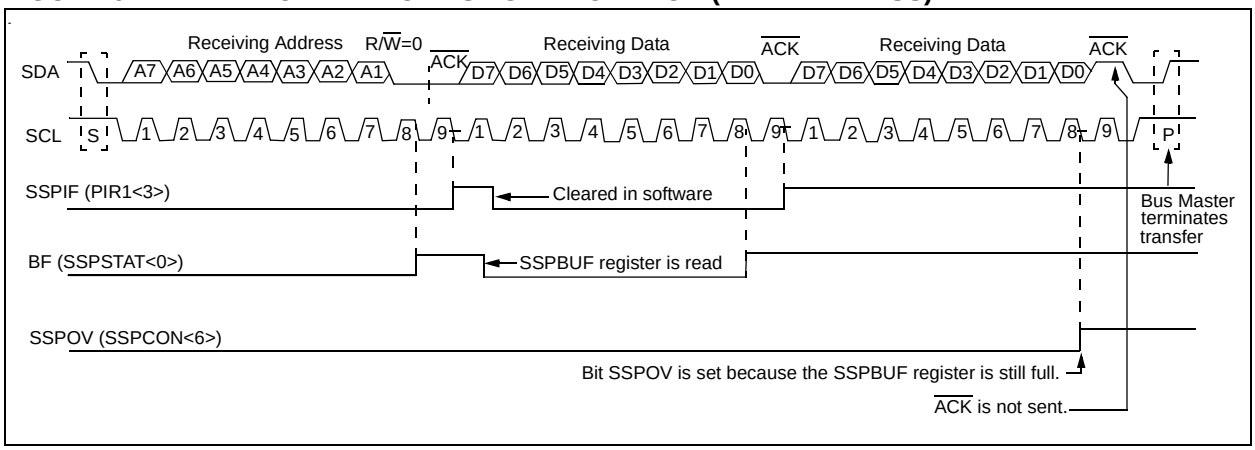
9.3.1.2 Reception

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.

When the address byte overflow condition exists, then no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON<6>) is set.

An SSP 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.

FIGURE 9-17: I²C WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)



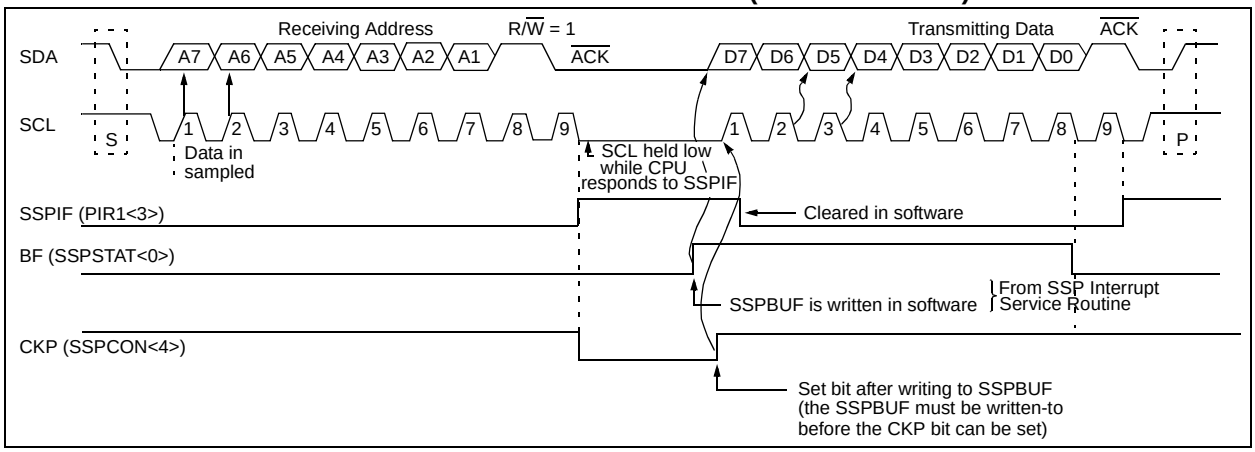
9.3.1.3 Transmission

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. 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 (SSPCON<4>). The master must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. 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 9-18).

An SSP interrupt is generated for each data transfer byte. Flag bit SSPIF must be cleared in software, and the SSPSTAT register is used to determine the status of the byte. Flag bit SSPIF is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. When the $\overline{ACK}$ is latched by the slave, the slave logic is reset and the slave then monitors for another occurrence of the START bit. If the SDA line was low ($\overline{ACK}$), 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.

FIGURE 9-18: I²C WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)



9.3.2 MASTER MODE

Master mode of operation is supported, in firmware, using 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 SSP module is disabled. The STOP and START bits will toggle based on the START and STOP conditions. 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 Master mode, the SCL and SDA lines are manipulated by clearing the corresponding TRISC<4:3> bit(s). The output level is always low, irrespective of the value(s) in PORTC<4:3>. So when transmitting data, a '1' data bit must have the TRISC<4> bit set (input) and a '0' data bit must have the TRISC<4> bit cleared (output). The same scenario is true for the SCL line with the TRISC<3> bit.

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

Master mode of operation can be done with either the Slave mode idle (SSPM3:SSPM0 = 1011), or with the slave active. When both Master and Slave modes are enabled, the software needs to differentiate the source(s) of the interrupt.

9.3.3 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 SSP module is disabled. The STOP and START bits will toggle based on the START and STOP conditions. Control of the I^2C bus may be taken when bit P (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 to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRISC<4:3>). There are two stages where this arbitration can be lost, they are:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to re-transfer the data at a later time.

TABLE 9-4: REGISTERS ASSOCIATED WITH I^2C OPERATION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	0000 0000
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000
87h	TRISC	—	—	PORTC Data Direction Control Register						--11 1111	--11 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by SSP in I^2C mode.

FIGURE 9-19: OPERATION OF THE I²C MODULE IN IDLE_MODE, RCV_MODE OR XMIT_MODE

```
IDLE_MODE (7-bit):
if (Addr_match)                 {       Set interrupt;
                                        if (R/W = 1)    {       Send ACK = 0;
                                                                set XMIT_MODE;
                                                        }
                                        else if (R/W = 0) set RCV_MODE;
                                }
RCV_MODE:
if ((SSPBUF = Full) OR (SSPOV = 1))
        {       Set SSPOV;
                Do not acknowledge;
        }
else    {       transfer SSPSR → SSPBUF;
                send ACK = 0;
        }
Receive 8-bits in SSPSR;
Set interrupt;
XMIT_MODE:
While ((SSPBUF = Empty) AND (CKP=0)) Hold SCL Low;
Send byte;
Set interrupt;
if ( ACK Received = 1)          {       End of transmission;
                                        Go back to IDLE_MODE;
                                }
else if ( ACK Received = 0)   Go back to XMIT_MODE;
IDLE_MODE (10-Bit):
If (High_byte_addr_match AND (R/W = 0))
        {       PRIOR_ADDR_MATCH = FALSE;
                Set interrupt;
                if ((SSPBUF = Full) OR ((SSPOV = 1))
                        {       Set SSPOV;
                                Do not acknowledge;
                        }
                else    {       Set UA = 1;
                                Send ACK = 0;
                                While (SSPADD not updated) Hold SCL low;
                                Clear UA = 0;
                                Receive Low_addr_byte;
                                Set interrupt;
                                Set UA = 1;
                                If (Low_byte_addr_match)
                                        {       PRIOR_ADDR_MATCH = TRUE;
                                                Send ACK = 0;
                                                while (SSPADD not updated) Hold SCL low;
                                                Clear UA = 0;
                                                Set RCV_MODE;
                                        }
                        }
        }
else if (High_byte_addr_match AND (R/W = 1))
        {       if (PRIOR_ADDR_MATCH)
                        {       send ACK = 0;
                                set XMIT_MODE;
                        }
        else PRIOR_ADDR_MATCH = FALSE;
        }
```

NOTES:

10.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs.

The analog input charges a sample and hold capacitor. The output of the sample and hold capacitor is the input into the converter. The converter then generates a digital result of this analog level via successive approximation. The A/D conversion of the analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low voltage reference input, that is software selectable to some combination of VDD, VSS, RA2 or RA3.

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 clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

The ADCON0 register, shown in Register 10-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 10-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference), or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 10-1: ADCON0 REGISTER (ADDRESS: 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS<2:0>**: Analog Channel Select bits
000 = channel 0 (RA0/AN0)
001 = channel 1 (RA1/AN1)
010 = channel 2 (RA2/AN2)
011 = channel 3 (RA3/AN3)
100 = channel 4 (RA5/AN4)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

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

bit 0 **ADON**: A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

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 10-2: ADCON1 REGISTER (ADDRESS 9Fh)

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. 6 Most Significant bits of ADRESH are read as ‘0’.
0 = Left justified. 6 Least Significant bits of ADRESL are read as ‘0’.

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

bit 3-0 **PCFG<3:0>**: A/D Port Configuration Control bits:

PCFG<3:0>	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs(1)
0000	A	A	A	A	A	VDD	VSS	5/0
0001	A	VREF+	A	A	A	RA3	VSS	4/1
0010	A	A	A	A	A	VDD	VSS	5/0
0011	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	A	D	A	A	VDD	VSS	3/0
0101	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	VDD	VSS	0/0
1000	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1001	A	A	A	A	A	VDD	VSS	5/0
1010	A	VREF+	A	A	A	RA3	VSS	4/1
1011	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1100	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	A	VDD	VSS	1/0
1111	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Note 1: This column indicates the number of analog channels available as A/D inputs and the number of analog channels used as voltage reference inputs.

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 ADRESH:ADRESL registers contain the 10-bit result of the A/D conversion. When the A/D conversion is complete, the result is loaded into this A/D result register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared and the A/D interrupt flag bit ADIF is set. The block diagram of the A/D module is shown in Figure 10-1.

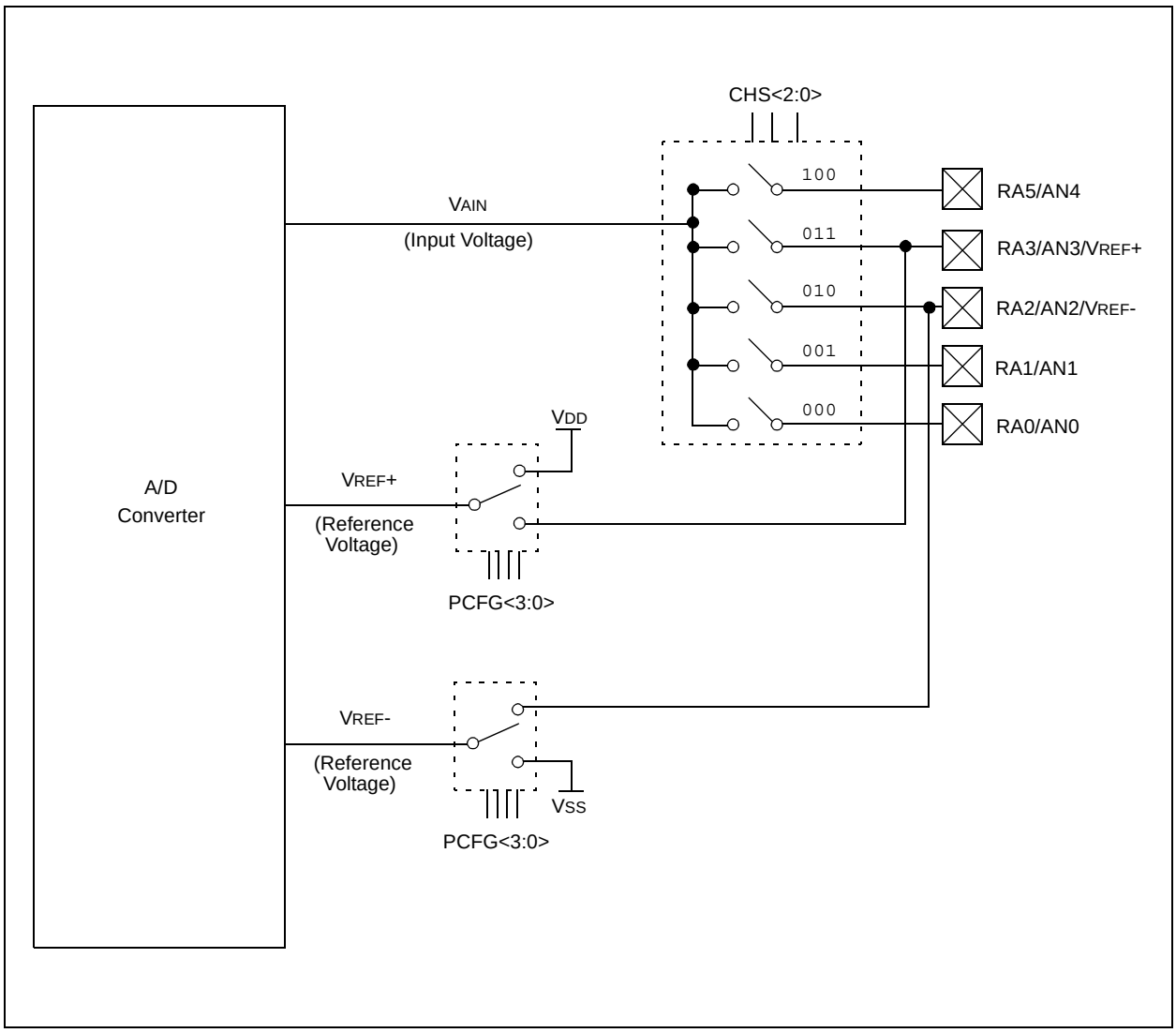
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs. To determine sample time, see Section 10.1. After this acquisition time has elapsed, the A/D conversion can be started.

The following steps should be followed for doing 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 (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set PEIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared (interrupts disabled)

   OR
   - Waiting for the A/D interrupt
6. Read A/D Result register pair (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 2TAD is required before next acquisition starts.

FIGURE 10-1: A/D BLOCK DIAGRAM



10.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 10-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), see Figure 10-2. **The maximum recommended impedance for analog sources is 10 kΩ.** As the impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 10-1 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.

To calculate the minimum acquisition time, TACQ, see the PICmicro™ Mid-Range Reference Manual (DS33023).

EQUATION 10-1: ACQUISITION TIME EXAMPLE

$$
\begin{aligned}
T_{ACQ} &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\
&= T_{AMP} + T_C + T_{COFF} \\
&= 2\mu S + T_C + [(\text{Temperature } -25^\circ C)(0.05\mu S/^\circ C)] \\
T_C &= C_{HOLD}(R_{IC} + R_{SS} + R_S)\ \text{In}(1/2047) \\
&= -120pF(1k\Omega + 7k\Omega + 10k\Omega)\ \text{In}(0.0004885) \\
&= 16.47\mu S \\
T_{ACQ} &= 2\mu S + 16.47\mu S + [(50^\circ C -25^\circ C)(0.05\mu S/^\circ C) \\
&= 19.72\mu S
\end{aligned}
$$

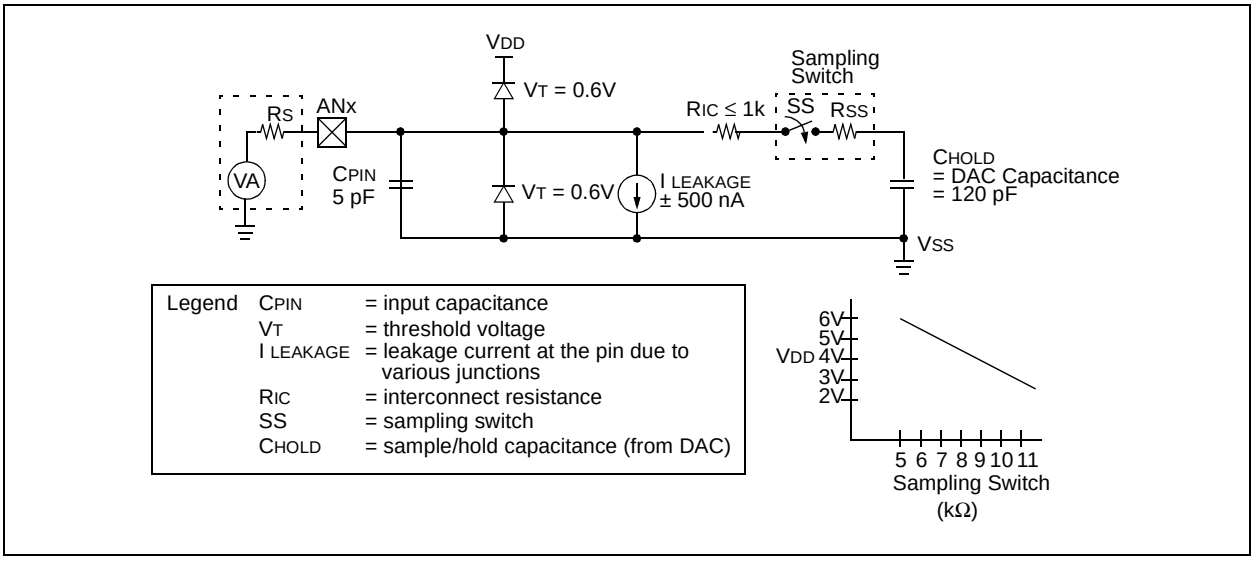
Note 1: The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

2: The charge holding capacitor (CHOLD) is not discharged after each conversion.

3: The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

4: After a conversion has completed, a 2.0TAD delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

FIGURE 10-2: ANALOG INPUT MODEL



10.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires a minimum 12TAD per 10-bit conversion. The source of the A/D conversion clock is software selected. The four possible options for TAD are:

- 2TOSC
- 8TOSC
- 32TOSC
- Internal A/D module 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 10-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 10-1: TAD vs. MAXIMUM DEVICE OPERATING FREQUENCIES (STANDARD DEVICES (C))

AD Clock Source (TAD)		Maximum Device Frequency
Operation	**ADCS<1:0>**	**Max.**
2TOSC	00	1.25 MHz
8TOSC	01	5 MHz
32TOSC	10	20 MHz
RC(1, 2, 3)	11	**(Note 1)**

Note 1: The RC source has a typical TAD time of 4 µs, but can vary between 2-6 µs.

2: When the device frequencies are greater than 1 MHz, the RC A/D conversion clock source is only recommended for SLEEP operation.

3: For extended voltage devices (LC), please refer to the Electrical Specifications section.

10.3 Configuring Analog Port Pins

The ADCON1 and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (V_{OH} or V_{OL}) will be converted.

The A/D operation is independent of the state of the CHS<2:0> bits and the TRIS bits.

Note 1: When reading the port register, any pin configured as an analog input channel 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 that is defined as a digital input (including the AN<4:0> pins), may cause the input buffer to consume current that is out of the device specifications.

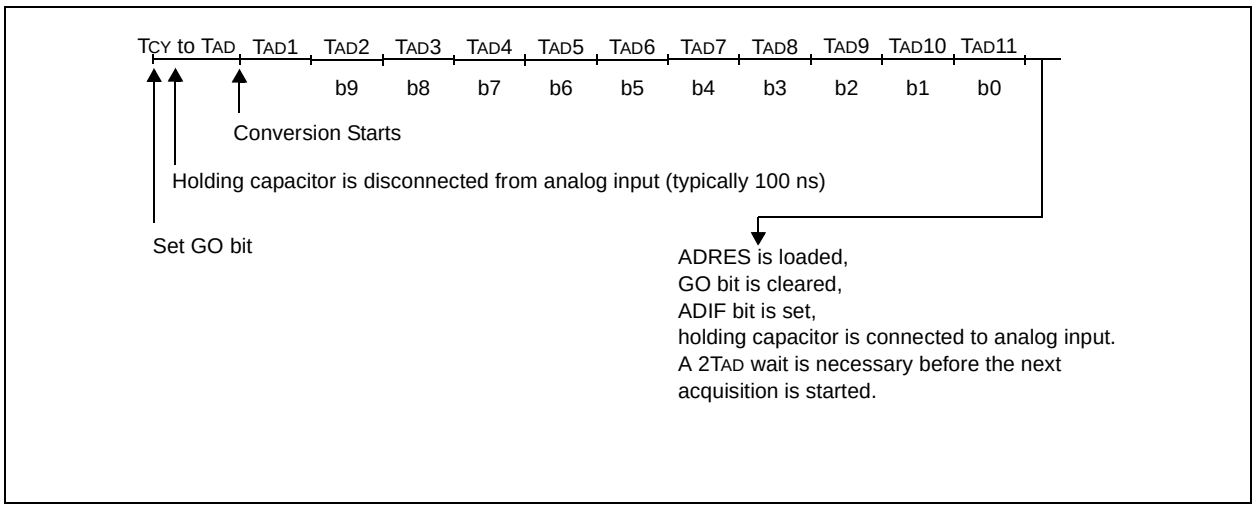
10.4 A/D Conversions

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D result register pair will NOT be updated with the partially completed A/D conversion sample. 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 $2T_{AD}$ wait is required before the next acquisition is started. After this $2T_{AD}$ wait, acquisition on the selected channel is automatically started. After this, the GO/$\overline{\text{DONE}}$ bit can be set to start the conversion.

In Figure 10-3, after the GO bit is set, the first time segment has a minimum of T_{CY} and a maximum of T_{AD}.

Note: The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

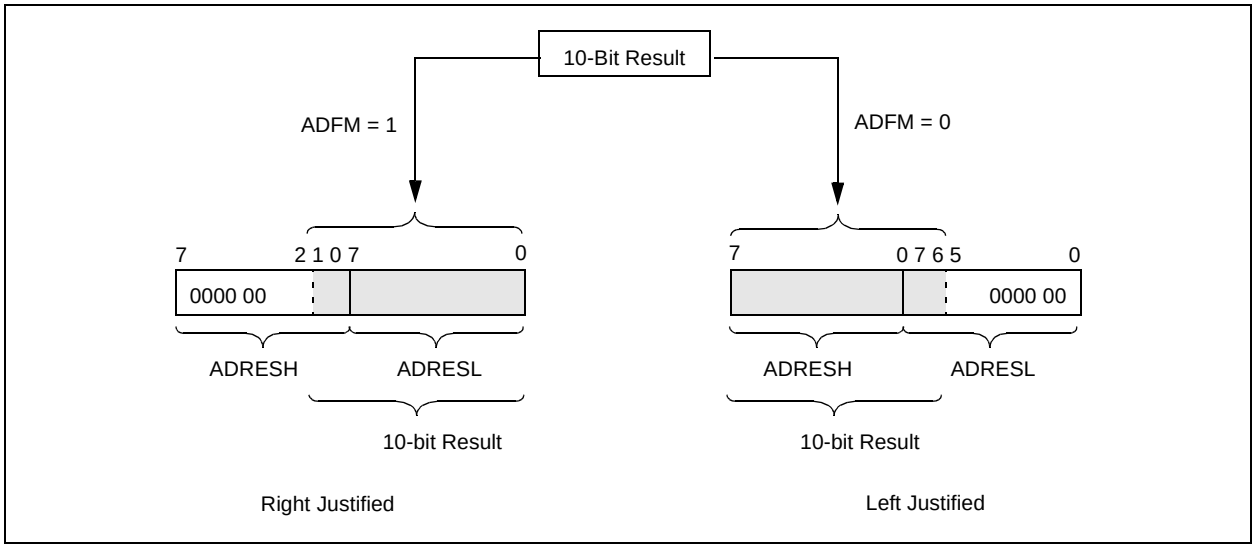
FIGURE 10-3: A/D CONVERSION T_{AD} CYCLES



10.4.1 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16-bits wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D Format Select bit (ADFM) controls this justification. Figure 10-4 shows the operation of the A/D result justification. The extra bits are loaded with '0's'. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

FIGURE 10-4: A/D RESULT JUSTIFICATION



10.5 A/D Operation During SLEEP

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS<1:0> = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/DONE bit will be cleared and the result loaded into the ADRES register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

Note:	For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS<1:0> = 11). To allow the conversion to occur during SLEEP, ensure the SLEEP instruction immediately follows the instruction that sets the GO/DONE bit.

10.6 Effects of a RESET

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. All A/D input pins are configured as analog inputs.

The value that is 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.

TABLE 10-2: REGISTERS/BITS ASSOCIATED WITH A/D

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR, BOR	MCLR, WDT
0Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	**(1)**	**(1)**	SSPIF	CCP1IF	TMR2IF	TMR1IF	r0rr 0000	r0rr 0000
8Ch	PIE1	LCDIE	ADIE	**(1)**	**(1)**	SSPIE	CCP1IE	TMR2IE	TMR1IE	r0rr 0000	r0rr 0000
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON	0000 00-0	0000 00-0
9Fh	ADCON1	ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0	--0- 0000	--0- 0000
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	--0u 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.
Note 1: These bits are reserved; always maintain these bits clear.

NOTES:

11.0 LCD MODULE

The LCD module generates the timing control to drive a static or multiplexed LCD panel, with support for up to 32 segments multiplexed with up to four commons. It also provides control of the LCD pixel data.

The interface to the module consists of 3 control registers (LCDCON, LCDSE, and LCDPS), used to define the timing requirements of the LCD panel and up to 16 LCD data registers (LCD00-LCD15) that represent the array of the pixel data. In normal operation, the control registers are configured to match the LCD panel being used. Primarily, the initialization information consists of selecting the number of commons required by the LCD panel, and then specifying the LCD frame clock rate to be used by the panel.

Once the module is initialized for the LCD panel, the individual bits of the LCD data registers are cleared/set to represent a clear/dark pixel, respectively.

Once the module is configured, the LCDEN (LCDCON<7>) bit is used to enable or disable the LCD module. The LCD panel can also operate during SLEEP by clearing the SLPEN (LCDCON<6>) bit.

Figure 11-2 through Figure 11-5 provides waveforms for static, half-duty cycle, one-third-duty cycle, and quarter-duty cycle drives.

REGISTER 11-1: LCDCON REGISTER (ADDRESS 10Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	SLPEN	WERR	BIAS	CS1	CS0	LMUX1	LMUX0
bit 7							bit 0

bit 7 **LCDEN**: Module Drive Enable bit
1 = LCD drive enabled
0 = LCD drive disabled

bit 6 **SLPEN**: LCD Display Enabled to SLEEP bit
1 = LCD module will stop driving in SLEEP
0 = LCD module will continue driving in SLEEP

bit 5 **WERR:** Write Failed Error bit
1 = System tried to write LCDD register during disallowed time. (Must be reset in software.)
0 = No error

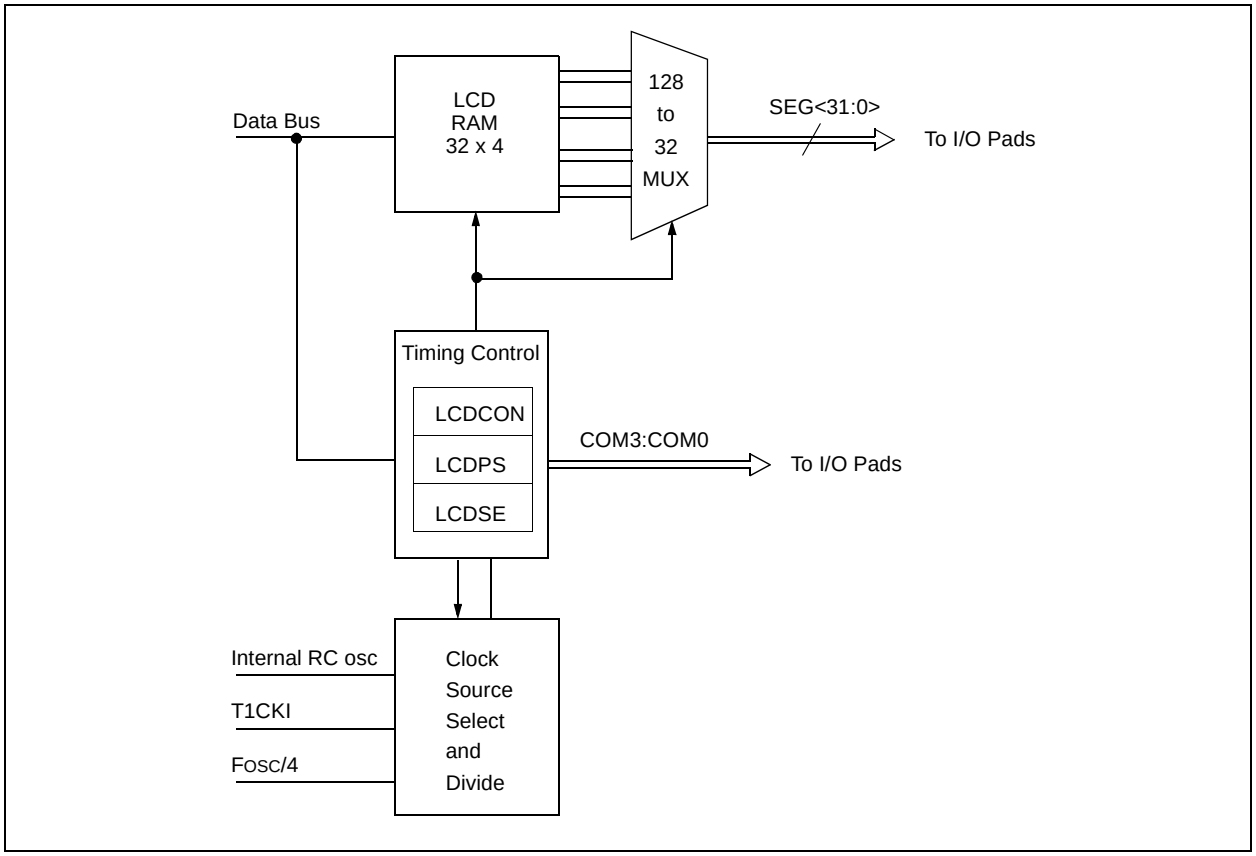
bit 4 **BIAS**: Bias Generator Enable bit
0 = Internal bias generator powered down, bias is expected to be provided externally
1 = Internal bias generator enabled, powered up

bit 3-2 **CS<1:0>:** Clock Source bits
00 = FOSC/256
01 = T1CKI (Timer1)
1x = Internal RC oscillator

bit 1-0 **LMUX<1:0>:** Common Selection bits
Specifies the number of commons
00 = Static(COM0)
01 = 1/2 (COM0, 1)
10 = 1/3 (COM0, 1, 2)
11 = 1/4 (COM0, 1, 2, 3)

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: LCD MODULE BLOCK DIAGRAM



REGISTER 11-2: LCDPS REGISTER (ADDRESS 10Eh)

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	LP3	LP2	LP1	LP0
bit 7							bit 0

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

bit 3-0 **LP<3:0>:** Frame Clock Prescale Selection bits (see Section 11.1.2)

LMUX1:LMUX0	Multiplex	Frame Frequency
00	Static	Clock source/(128 * (LP3:LP0 + 1))
01	1/2	Clock source/(128 * (LP3:LP0 + 1))
10	1/3	Clock source/(96 * (LP3:LP0 + 1))
11	1/4	Clock source/(128 * (LP3:LP0 + 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

FIGURE 11-2: WAVEFORMS IN STATIC DRIVE

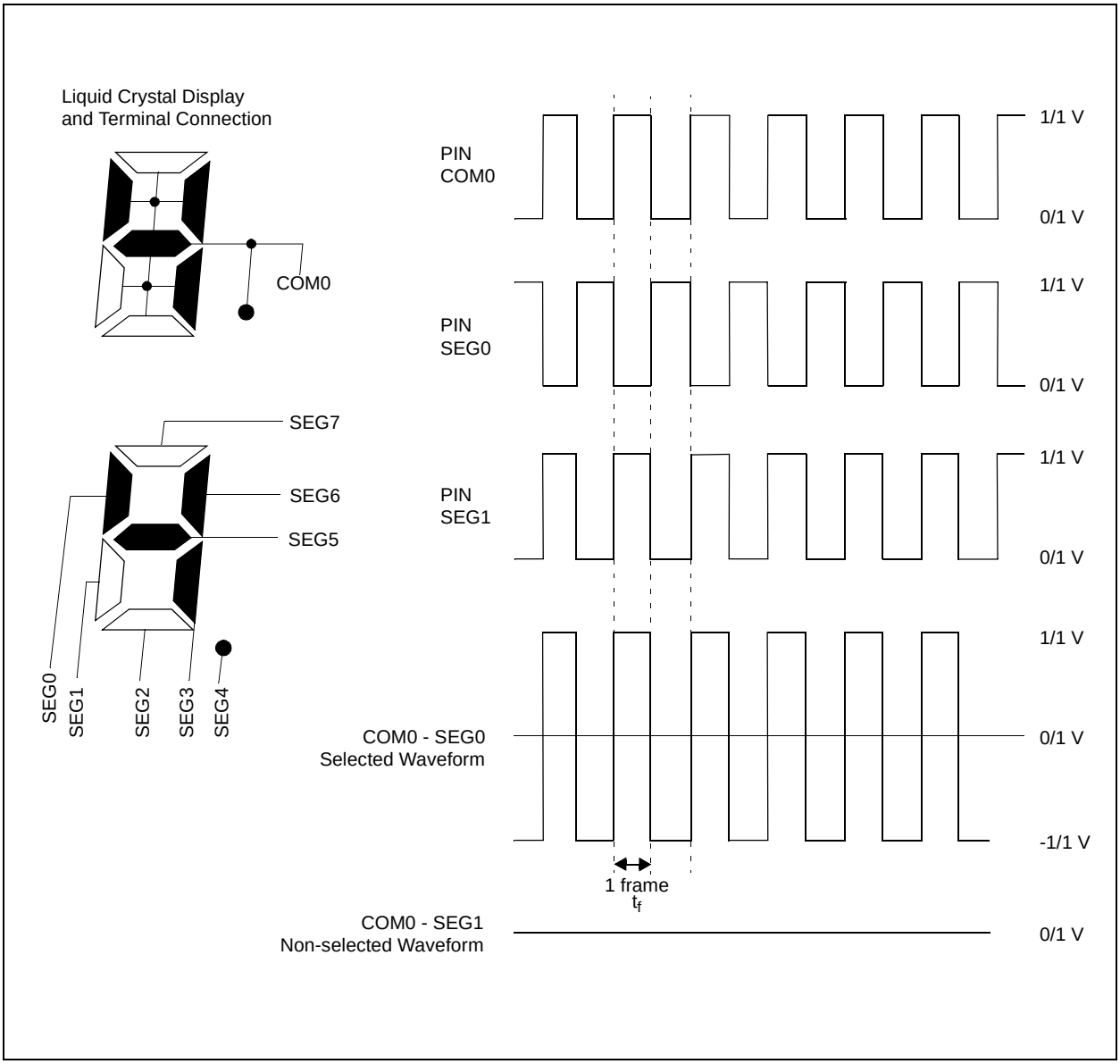


FIGURE 11-3: WAVEFORMS IN HALF-DUTY CYCLE DRIVE (B TYPE)

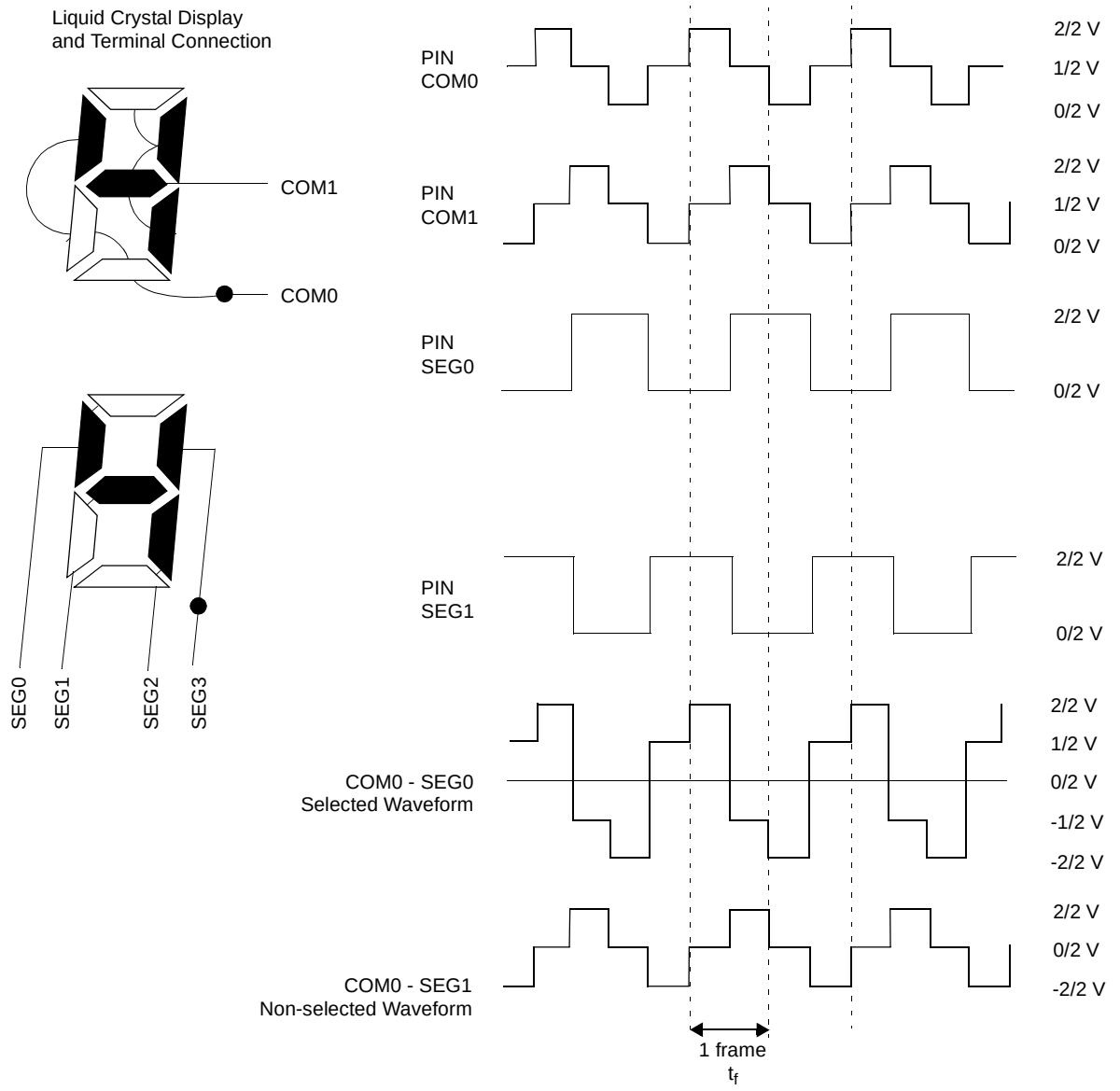


FIGURE 11-4: WAVEFORMS IN ONE-THIRD DUTY CYCLE DRIVE (B TYPE)

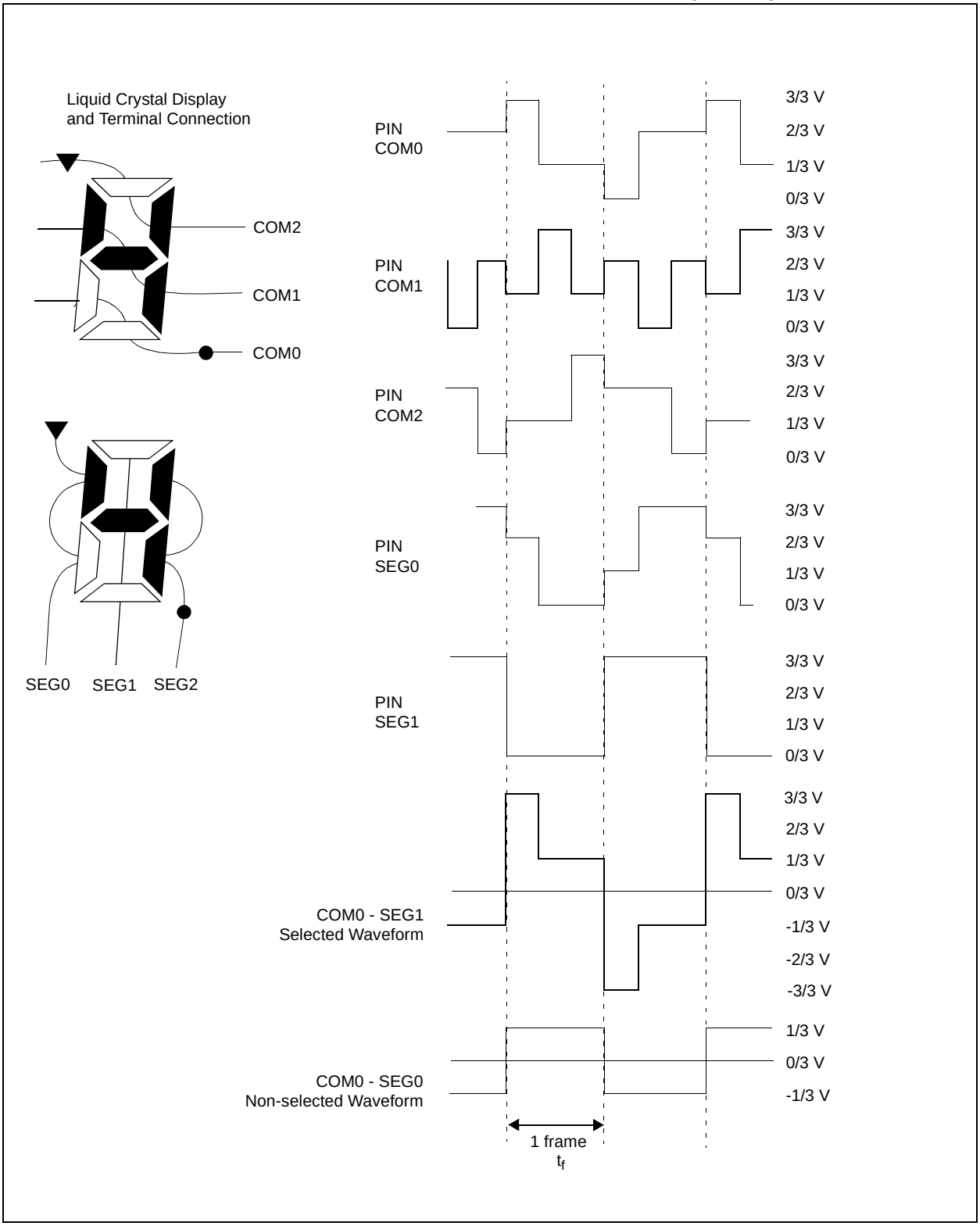
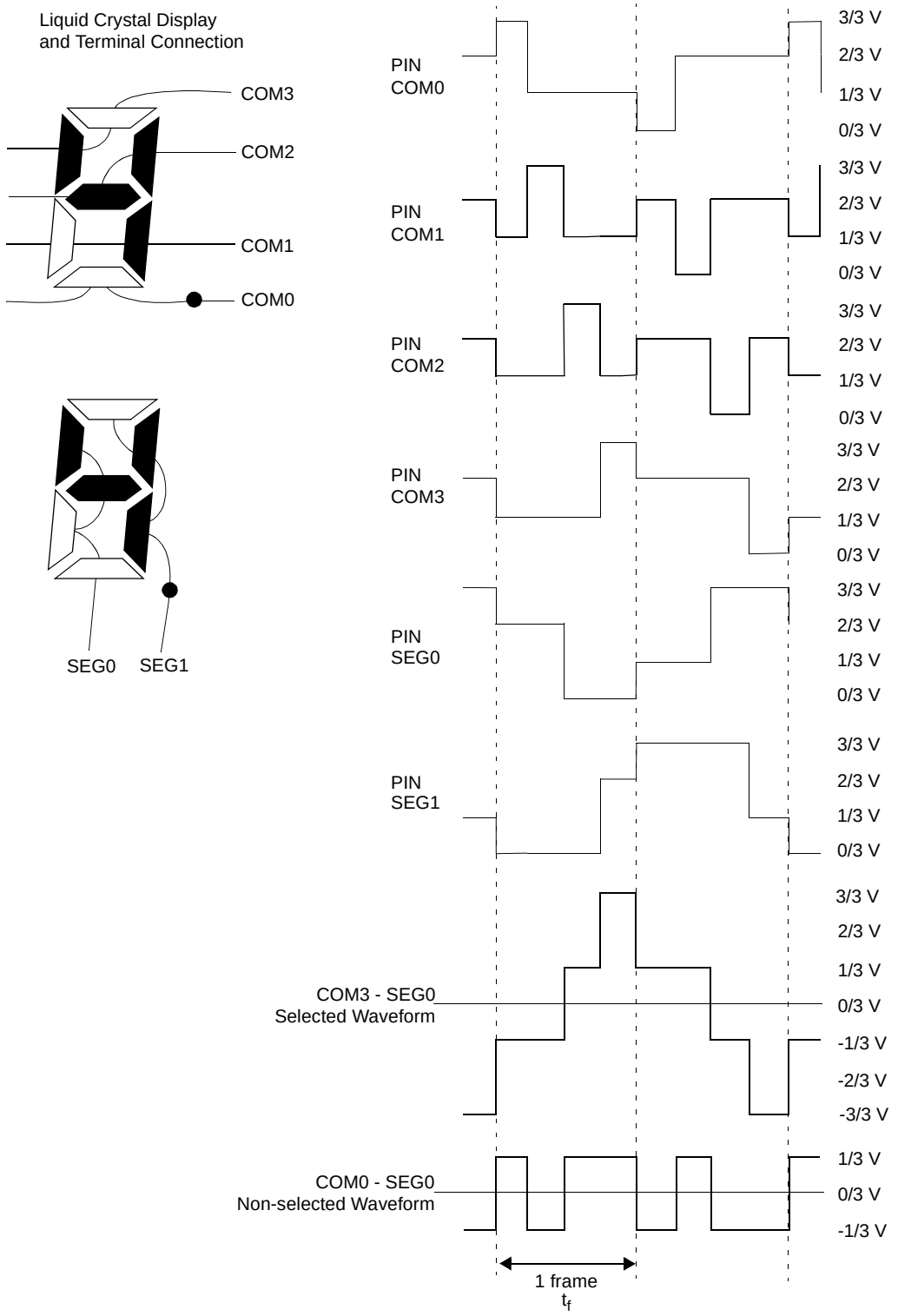


FIGURE 11-5: WAVEFORMS IN QUARTER-DUTY CYCLE DRIVE (B TYPE)



11.1 LCD Timing

The LCD module has 3 possible clock source inputs and supports static, 1/2, 1/3, and 1/4 multiplexing.

11.1.1 TIMING CLOCK SOURCE SELECTION

The clock sources for the LCD timing generation are:

- Internal RC oscillator
- Timer1 oscillator
- System clock divided by 256

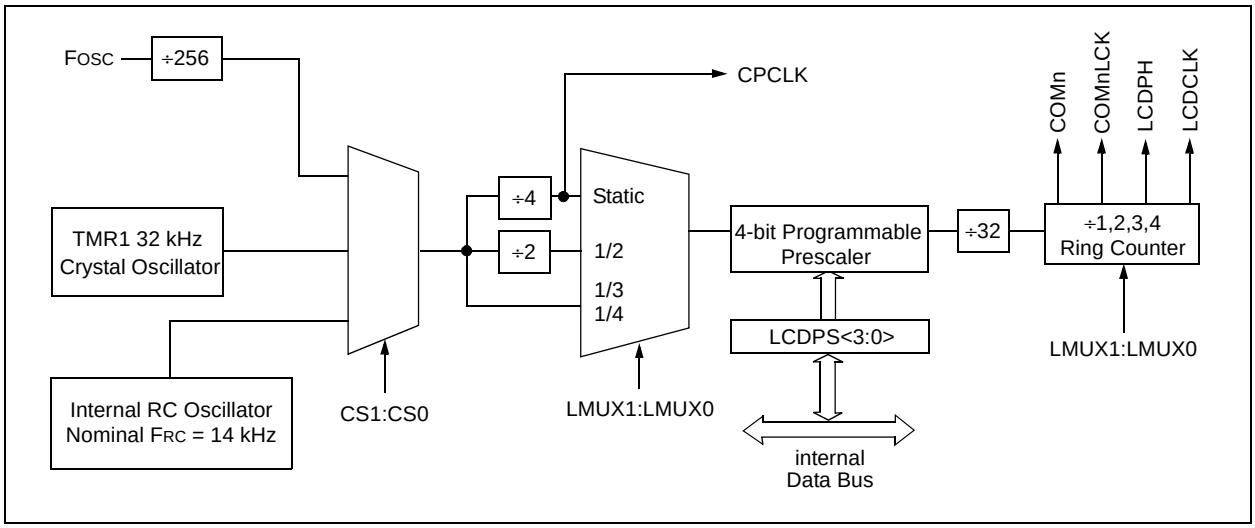
The first timing source is an internal RC oscillator which runs at a nominal frequency of 14 kHz. This oscillator provides a lower speed clock which may be used to continue running the LCD while the processor is in SLEEP. The RC oscillator will power-down when it is not selected or when the LCD module is disabled.

The second source is the Timer1 external oscillator. This oscillator provides a lower speed clock which may be used to continue running the LCD while the processor is in SLEEP. It is assumed that the frequency provided on this oscillator will be 32 kHz. To use the Timer1 oscillator as a LCD module clock source, it is only necessary to set the T1OSCEN (T1CON<3>) bit.

The third source is the system clock divided by 256. This divider ratio is chosen to provide about 32 kHz output when the external oscillator is 8 MHz. The divider is not programmable. Instead the LCDPS register is used to set the LCD frame clock rate.

All of the clock sources are selected with bits CS1:CS0 (LCDCON<3:2>). Refer to Register 11-1 for details of the register programming.

FIGURE 11-6: LCD CLOCK GENERATION



11.1.2 MULTIPLEX TIMING GENERATION

The timing generation circuitry will generate one to four common clocks based on the display mode selected. The mode is specified by bits LMUX1:LMUX0 (LCDCON<1:0>). Table 11-1 shows the formulas for calculating the frame frequency.

TABLE 11-1: FRAME FREQUENCY FORMULAS

Multiplex	Frame Frequency =
Static	Clock source/(128 * (LP3:LP0 + 1))
1/2	Clock source/(128 * (LP3:LP0 + 1))
1/3	Clock source/(96 * (LP3:LP0 + 1))
1/4	Clock source/(128 * (LP3:LP0 + 1))

TABLE 11-2: APPROXIMATE FRAME FREQUENCY (IN Hz) USING TIMER1 @ 32.768 kHz OR FOSC @ 8 MHz

LP3:LP0	Static	1/2	1/3	1/4
2	85	85	114	85
3	64	64	85	64
4	51	51	68	51
5	43	43	57	43
6	37	37	49	37
7	32	32	43	32

TABLE 11-3: APPROXIMATE FRAME FREQUENCY (IN Hz) USING INTERNAL RC OSC @ 14 kHz

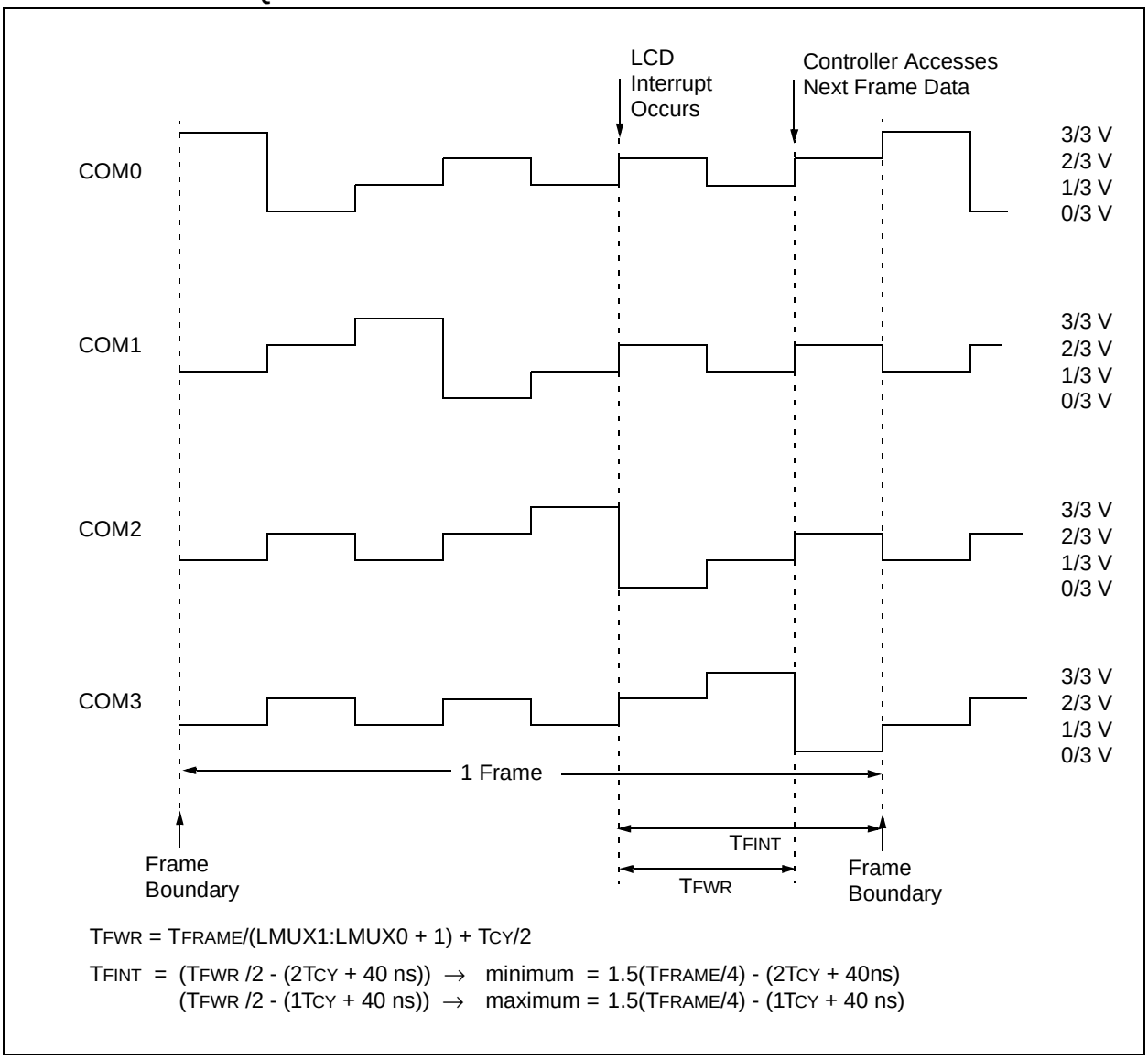
LP3:LP0	Static	1/2	1/3	1/4
0	109	109	146	109
1	55	55	73	55
2	36	36	49	36
3	27	27	36	27

11.2 LCD Interrupts

The LCD timing generation provides an interrupt that defines the LCD frame timing. This interrupt can be used to coordinate the writing of the pixel data with the start of a new frame. Writing pixel data at the frame boundary allows a visually crisp transition of the image. This interrupt can also be used to synchronize external events to the LCD. For example, the interface to an external segment driver, such as a Microchip AY0438, can be synchronized for segment data update to the LCD frame.

A new frame is defined to begin at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (T_{FINT}), as shown in Figure 11-7. The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins to access data after the interrupt (T_{FWR}). New data must be written within T_{FWR}, as this is when the LCD controller will begin to access the data for the next frame.

FIGURE 11-7: EXAMPLE WAVEFORMS AND INTERRUPT TIMING IN QUARTER-DUTY CYCLE DRIVE



$T_{FWR} = T_{FRAME}/(LMUX1:LMUX0 + 1) + T_{CY}/2$

T_{FINT} = (T_{FWR} /2 - (2T_{CY} + 40 ns)) → minimum = 1.5(T_{FRAME}/4) - (2T_{CY} + 40ns)
(T_{FWR} /2 - (1T_{CY} + 40 ns)) → maximum = 1.5(T_{FRAME}/4) - (1T_{CY} + 40 ns)

11.3 Pixel Control

11.3.1 LCDD (PIXEL DATA) REGISTERS

The pixel registers contain bits which define the state of each pixel. Each bit defines one unique pixel.

Table 11-4 shows the correlation of each bit in the LCDD registers to the respective common and segment signals.

Any LCD pixel location not being used for display can be used as general purpose RAM.

REGISTER 11-3: GENERIC LCDD REGISTER LAYOUT

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc
bit 7							bit 0

bit 7-0 **SEGsCOMc**: Pixel Data bit for Segment S and Common C

1 = Pixel on (dark)
0 = Pixel off (clear)

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

11.4 Operation During SLEEP

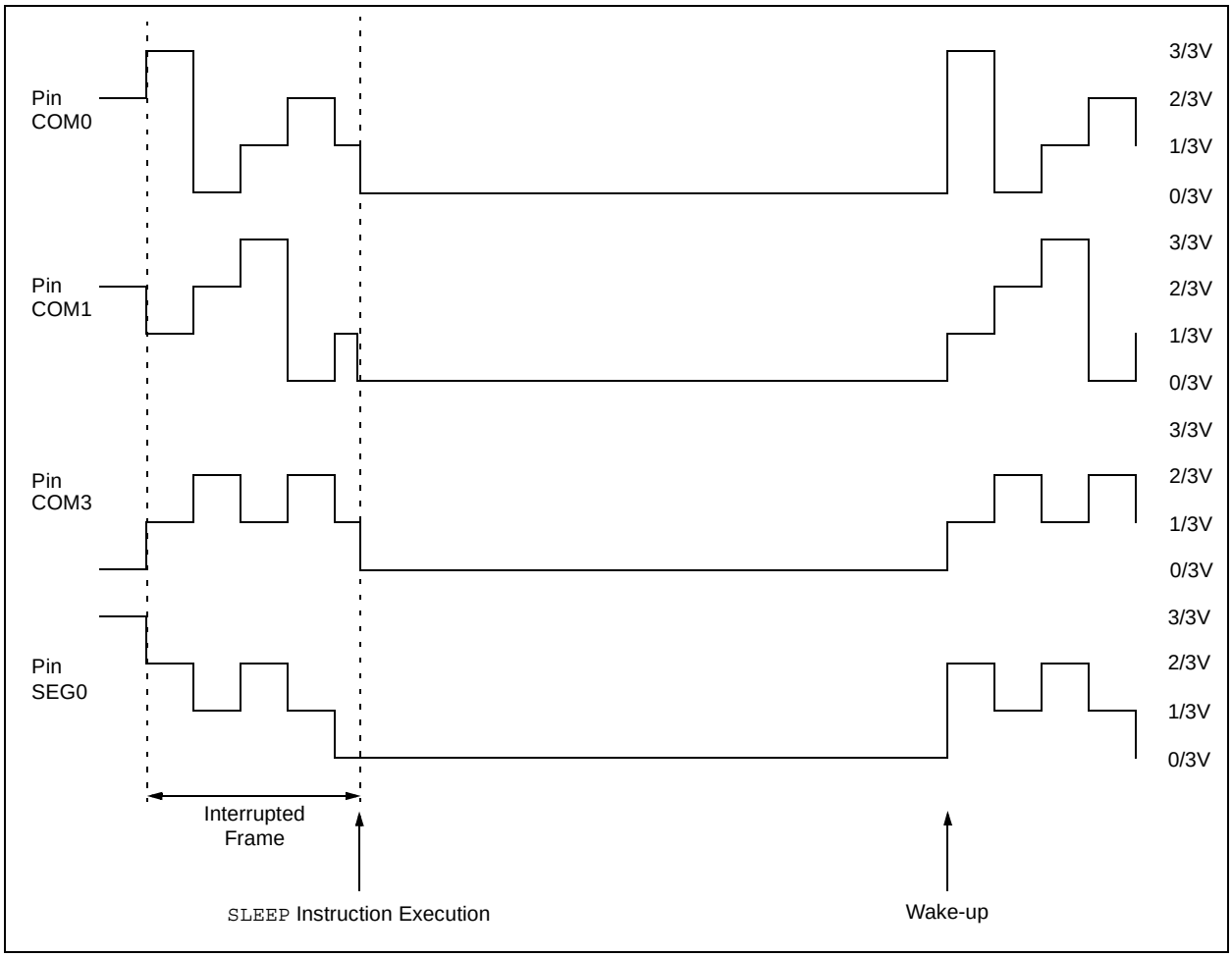
The LCD module can operate during SLEEP. The selection is controlled by bit SLPEN (LCDCON<6>). Setting the SLPEN bit allows the LCD module to go to SLEEP. Clearing the SLPEN bit allows the module to continue to operate during SLEEP.

If a SLEEP instruction is executed and SLPEN = '1', the LCD module will cease all functions and go into a very low current consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 11-8 shows this operation. To ensure that the LCD completes the frame, the SLEEP instruction should be executed immediately after a LCD frame boundary. The LCD interrupt can be used to determine the frame boundary. See Section 11.2 for the formulas to calculate the delay.

If a SLEEP instruction is executed and SLPEN = '0', the module will continue to display the current contents of the LCDD registers. To allow the module to continue operation while in SLEEP, the clock source must be either the internal RC oscillator or Timer1 external oscillator. While in SLEEP, the LCD data cannot be changed. The LCD module current consumption will not decrease in this mode, however, the overall consumption of the device will be lower due to shut-down of the core and other peripheral functions.

Note:	The internal RC oscillator or external Timer1 oscillator must be used to operate the LCD module during SLEEP.

FIGURE 11-8: SLEEP ENTRY/EXIT WHEN SLPEN = 1 OR CS1:CS0 = 00



11.4.1 SEGMENT ENABLES

The LCDSE register is used to select the pin function for groups of pins. The selection allows each group of pins to operate as either LCD drivers or digital only pins. To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared.

If the pin is a digital I/O the corresponding TRIS bit controls the data direction. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

Note 1: On a Power-on Reset, these pins are configured as LCD drivers.

2: The LMUX1:LMUX0 takes precedence over the LCDSE bit settings for pins RD7, RD6 and RD5.

EXAMPLE 11-1: STATIC MUX WITH 32 SEGMENTS

```
BCF   STATUS,RP0    ;Select Bank 2
BSF   STATUS,RP1    ;
BCF   LCDCON,LMUX1 ;Select Static MUX
BCF   LCDCON,LMUX0 ;
MOVLW 0xFF          ;Make PortD,E,F,G
MOVWF LCDSE         ;LCD pins
. . .               ;configure rest of LCD
```

EXAMPLE 11-2: ONE-THIRD DUTY CYCLE WITH 13 SEGMENTS

```
BCF   STATUS,RP0    ;Select Bank 2
BSF   STATUS,RP1    ;
BSF   LCDCON,LMUX1 ;Select 1/3 MUX
BCF   LCDCON,LMUX0 ;
MOVLW 0x87          ;Make PORTD<7:0> &
MOVWF LCDSE         ;PORTE<6:0> LCD pins
. . .               ;configure rest of LCD
```

REGISTER 11-4: LCDSE REGISTER (ADDRESS 10Dh)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0
bit 7							bit 0

bit 7 **SE29**: Pin Function Select RD7/COM1/SEG31 - RD5/COM3/SEG29
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 6 **SE27**: Pin Function Select RG7/SEG28 and RE7/SEG27
1 = Pins have LCD drive function
0 = Pins have LCD drive function

bit 5 **SE20**: Pin Function Select RG6/SEG26 - RG0/SEG20
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 4 **SE16**: Pin Function Select RF7/SEG19 - RF4/SEG16
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 3 **SE12**: Pin Function Select RF3/SEG15 - RF0/SEG12
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 2 **SE9**: Pin Function Select RE6/SEG11 - RE4/SEG09
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 1 **SE5**: Pin Function Select RE3/SEG08 - RE0/SEG05
1 = Pins have LCD drive function
0 = Pins have digital Input function

bit 0 **SE0**: Pin Function Select RD4/SEG04 - RD0/SEG00
1 = Pins have LCD drive function
0 = Pins have digital Input function

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

11.5 Voltage Generation

There are two methods for LCD voltage generation: internal charge pump, or external resistor ladder.

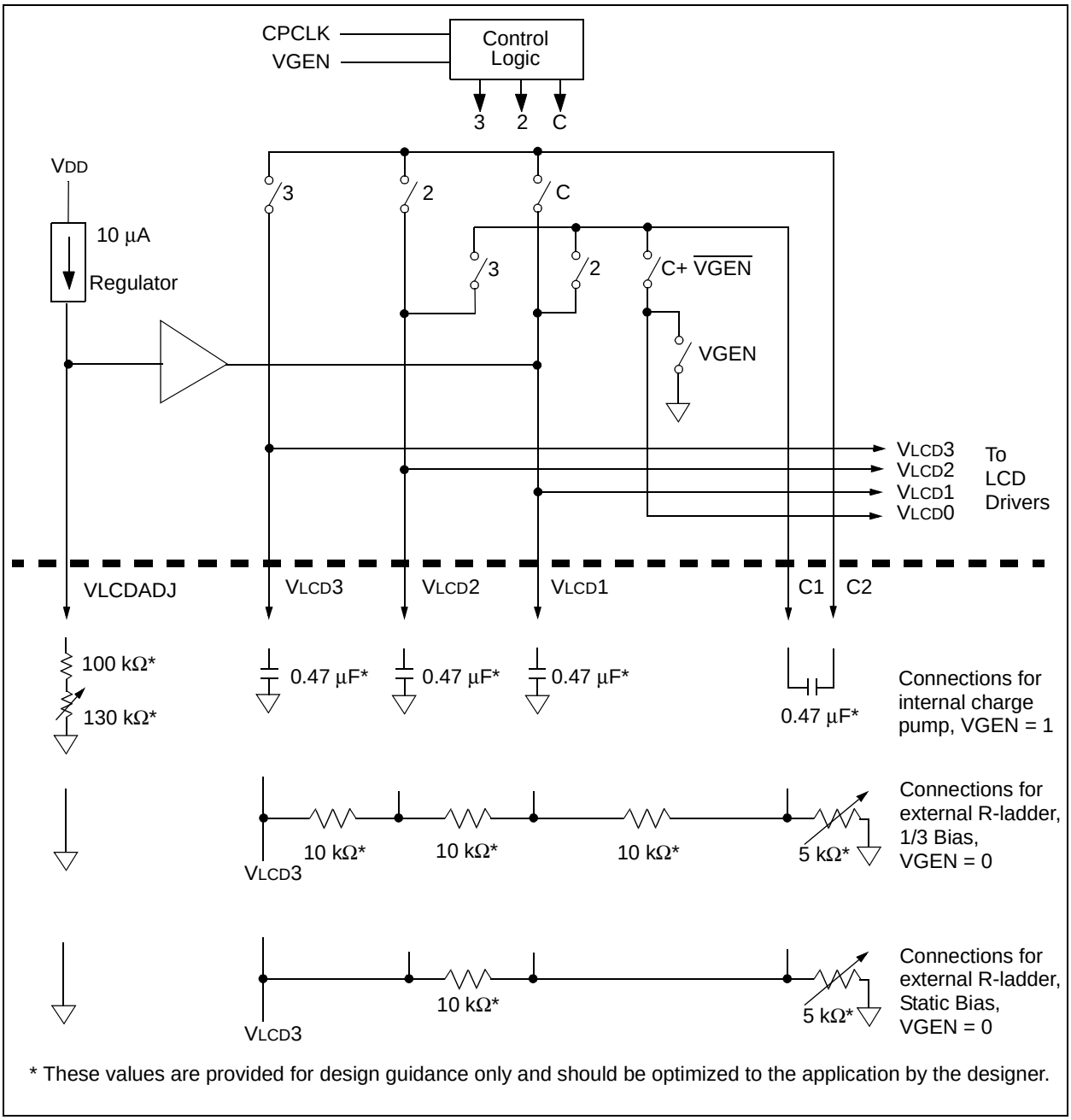
11.5.1 CHARGE PUMP

The LCD charge pump is shown in Figure 11-9. The 1.0V - 2.3V regulator will establish a stable base voltage from the varying battery voltage. This regulator is adjustable through the range by connecting a variable external resistor from VLCDADJ to ground. The potentiometer provides contrast adjustment for the LCD. This base voltage is connected to VLCD1 on the charge pump. The charge pump boosts VLCD1 into VLCD2 = 2*VLCD1 and VLCD3 = 3 * VLCD1. When the charge pump is not operating, Vlcd3 will be internally tied to VDD. See the Electrical Specifications section for charge pump capacitor and potentiometer values.

11.5.2 EXTERNAL R-LADDER

The LCD module can also use an external resistor ladder (R-Ladder) to generate the LCD voltages. Figure 11-9 shows external connections for static and 1/3 bias. The VGEN (LCDCON<4>) bit must be cleared to use an external R-Ladder.

FIGURE 11-9: CHARGE PUMP AND RESISTOR LADDER



* These values are provided for design guidance only and should be optimized to the application by the designer.

11.6 Configuring the LCD Module

The following is the sequence of steps to follow to configure the LCD module.

1. Select the frame clock prescale using bits LP3:LP0 (LCDPS<3:0>).
2. Configure the appropriate pins to function as segment drivers using the LCDSE register.
3. Configure the LCD module for the following using the LCDCON register:
   - Multiplex mode and Bias, bits LMUX1:LMUX0
   - Timing source, bits CS1:CS0
   - Voltage generation, bit VGEN
   - SLEEP mode, bit SLPEN
4. Write initial values to pixel data registers, LCDD00 through LCDD15.
5. Clear LCD interrupt flag, LCDIF (PIR1<7>), and if desired, enable the interrupt by setting bit LCDIE (PIE1<7>).
6. Enable the LCD module, by setting bit LCDEN (LCDCON<7>).

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH THE LCD MODULE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu
110h	LCDD00	SEG07 COM0	SEG06 COM0	SEG05 COM0	SEG04 COM0	SEG03 COM0	SEG02 COM0	SEG01 COM0	SEG00 COM0	xxxx xxxx	uuuu uuuu
111h	LCDD01	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG09 COM0	SEG08 COM0	xxxx xxxx	uuuu uuuu
112h	LCDD02	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0	xxxx xxxx	uuuu uuuu
113h	LCDD03	SEG31 COM0	SEG30 COM0	SEG29 COM0	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0	xxxx xxxx	uuuu uuuu
114h	LCDD04	SEG07 COM1	SEG06 COM1	SEG05 COM1	SEG04 COM1	SEG03 COM1	SEG02 COM1	SEG01 COM1	SEG00 COM1	xxxx xxxx	uuuu uuuu
115h	LCDD05	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG09 COM1	SEG08 COM1	xxxx xxxx	uuuu uuuu
116h	LCDD06	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1	xxxx xxxx	uuuu uuuu
117h	LCDD07	SEG31 COM1(1)	SEG30 COM1	SEG29 COM1	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1	xxxx xxxx	uuuu uuuu
118h	LCDD08	SEG07 COM2	SEG06 COM2	SEG05 COM2	SEG04 COM2	SEG03 COM2	SEG02 COM2	SEG01 COM2	SEG00 COM2	xxxx xxxx	uuuu uuuu
119h	LCDD09	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG09 COM2	SEG08 COM2	xxxx xxxx	uuuu uuuu
11Ah	LCDD10	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2	xxxx xxxx	uuuu uuuu
11Bh	LCDD11	SEG31 COM2(1)	SEG30 COM2(1)	SEG29 COM2	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2	xxxx xxxx	uuuu uuuu
11Ch	LCDD12	SEG07 COM3	SEG06 COM3	SEG05 COM3	SEG04 COM3	SEG03 COM3	SEG02 COM3	SEG01 COM3	SEG00 COM3	xxxx xxxx	uuuu uuuu
11Dh	LCDD13	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG09 COM3	SEG08 COM3	xxxx xxxx	uuuu uuuu
11Eh	LCDD14	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3	xxxx xxxx	uuuu uuuu
11Fh	LCDD15	SEG31 COM3(1)	SEG30 COM3(1)	SEG29 COM3(1)	SEG28 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3	xxxx xxxx	uuuu uuuu
10Dh	LCDSE	SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0	1111 1111	1111 1111
10Eh	LCDPS	—	—	—	—	LP3	LP2	LP1	LP0	---- 0000	---- 0000
10Fh	LCDCON	LCDEN	SLPEN	—	VGEN	CS1	CS0	LMUX1	LMUX0	00-0 0000	00-0 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the LCD module.
Note 1: These pixels do not display, but can be used as general purpose RAM.

12.0 SPECIAL FEATURES OF THE CPU

What sets a microcontroller apart from other processors are special circuits to deal with the needs of real time applications. The PIC16CXXX family has a host of such features, intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These are:

- Oscillator 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

The PIC16CXXX has a Watchdog Timer which can be shut-off only through configuration bits. 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 of 72 ms (nominal) 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.

12.1 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 in program memory location 2007h.

The user will note that address 2007h is beyond the user program memory space and can be accessed only during programming.

REGISTER 12-1: CONFIGURATION WORD (ADDRESS 2007h)

—	—	—	—	—	—	—	BOREN	CP1	CP0	$\overline{\text{PWRTE}}$	WDTE	F0SC1	F0SC0

bit13 bit0

bit 13-7 **Unimplemented**

bit 6 **BOREN**: Brown-out Reset Enable bit
1 = BOR enabled
0 = BOR disabled

bit 5-4 **CP1:CP0:** Program Memory Code Protection bits
PIC16C926 (8K program memory):
11 = Code protection off
10 = 0000h to 0FFFh code protected (1/2 protected)
01 = 0000h to 1EFFh code protected (all but last 256 protected)
00 = 0000h to 1FFFh code protected (all protected)
PIC16C925 (4K program memory):
11 = Code protection off
10 = 0000h to 07FFh code protected (1/2 protected)
01 = 0000h to 0EFFh code protected (all but last 256 protected)
00 = 0000h to 0FFFh code protected (all protected)
1000h to 1FFFh wraps around to 0000h to 0FFFh

bit 3 **$\overline{\text{PWRTE}}$**: Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 2 **WDTE**: Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 1-0 **FOSC1:FOSC0**: Oscillator Selection bits
11 = RC oscillator
10 = HS oscillator
01 = XT oscillator
00 = LP oscillator

12.2 Oscillator Configurations

12.2.1 OSCILLATOR TYPES

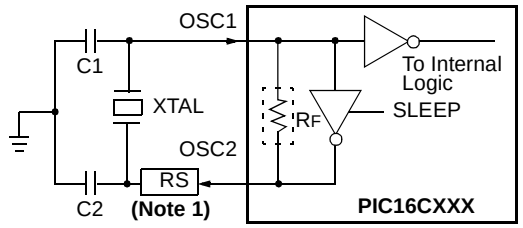
The PIC16CXXX can be operated in four different oscillator modes. The user can program two configuration bits (FOSC1 and FOSC0) to select one of these four modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- RC Resistor/Capacitor

12.2.2 CRYSTAL OSCILLATOR/CERAMIC RESONATORS

In XT, LP, or HS modes a crystal or ceramic resonator is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 12-1). The PIC16CXXX oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, LP, or HS modes, the device can have an external clock source to drive the OSC1/CLKIN pin (Figure 12-2).

FIGURE 12-1: CRYSTAL/CERAMIC RESONATOR OPERATION (HS, XT OR LP OSC CONFIGURATION)



See Table 12-1 and Table 12-2 for recommended values of C1 and C2.

Note 1: A series resistor may be required for AT strip cut crystals.

FIGURE 12-2: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

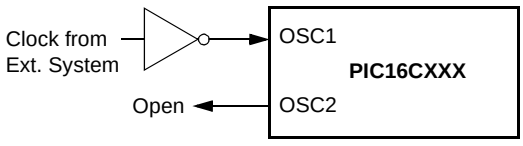


TABLE 12-1: CERAMIC RESONATORS

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

These values are for design guidance only.
See notes following Table 12-2.

TABLE 12-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq.	Cap. Range C1	Cap. Range C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15-33 pF	15-33 pF

These values are for design guidance only.
See notes following this table.

Note 1: Recommended ranges of C1 and C2 are depicted in Table 12-1.

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: Rs may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification.

12.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

Either a prepackaged oscillator can be used, or a simple oscillator circuit with TTL gates can be built. Prepackaged oscillators provide a wide operating range and better stability. A well designed crystal oscillator will provide good performance with TTL gates. Two types of crystal oscillator circuits can be used: one with series resonance, or one with parallel resonance.

Figure 12-3 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180-degree phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides the negative feedback for stability. The 10 kΩ potentiometer biases the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 12-3: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

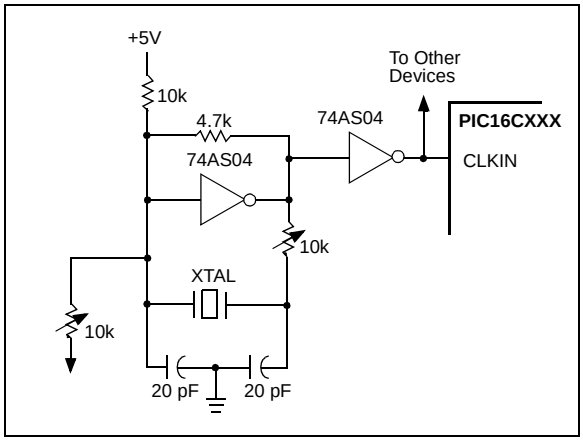
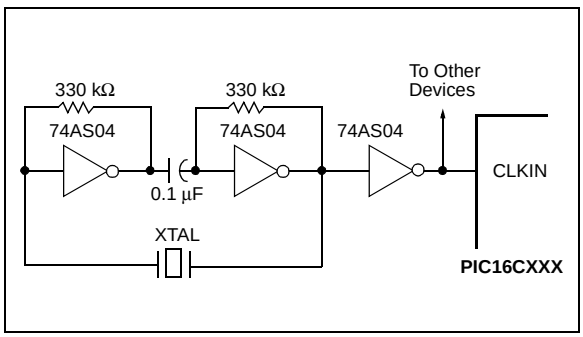


Figure 12-4 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 12-4: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



12.2.4 RC OSCILLATOR

For timing insensitive applications, the "RC" device option offers 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 12-5 shows how the R/C combination is connected to the PIC16CXXX. For REXT values below 2.2 kΩ, the oscillator operation may become unstable, or stop completely. For very high REXT values (e.g. 1 MΩ), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend to keep REXT between 3 kΩ and 100 kΩ.

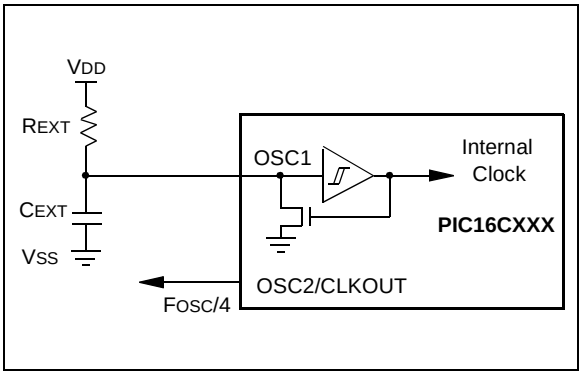
Although the oscillator will operate with no external capacitor (CEXT = 0 pF), we recommend using values above 20 pF for noise and stability reasons. With no or small external capacitance, the oscillation frequency can vary dramatically due to changes in external capacitances, such as PCB trace capacitance, or package lead frame capacitance.

See characterization data for desired device for RC frequency variation from part to part, due to normal process variation. The variation is larger for larger R (since leakage current variation will affect RC frequency more for large R) and for smaller C (since variation of input capacitance will affect RC frequency more).

See characterization data for desired device for variation of oscillator frequency, due to VDD for given REXT/CEXT values, as well as frequency variation due to operating temperature for given R, C, and VDD values.

The oscillator frequency, divided by 4, is available on the OSC2/CLKOUT pin, and can be used for test purposes or to synchronize other logic (see Figure 1-2 for waveform).

FIGURE 12-5: RC OSCILLATOR MODE



12.3 RESET

The PIC16C9XX differentiates between various kinds of RESET:

- Power-on Reset (POR)
- $\overline{MCLR}$ Reset during normal operation
- $\overline{MCLR}$ Reset during SLEEP
- WDT Reset (normal operation)
- Brown-out Reset (BOR)

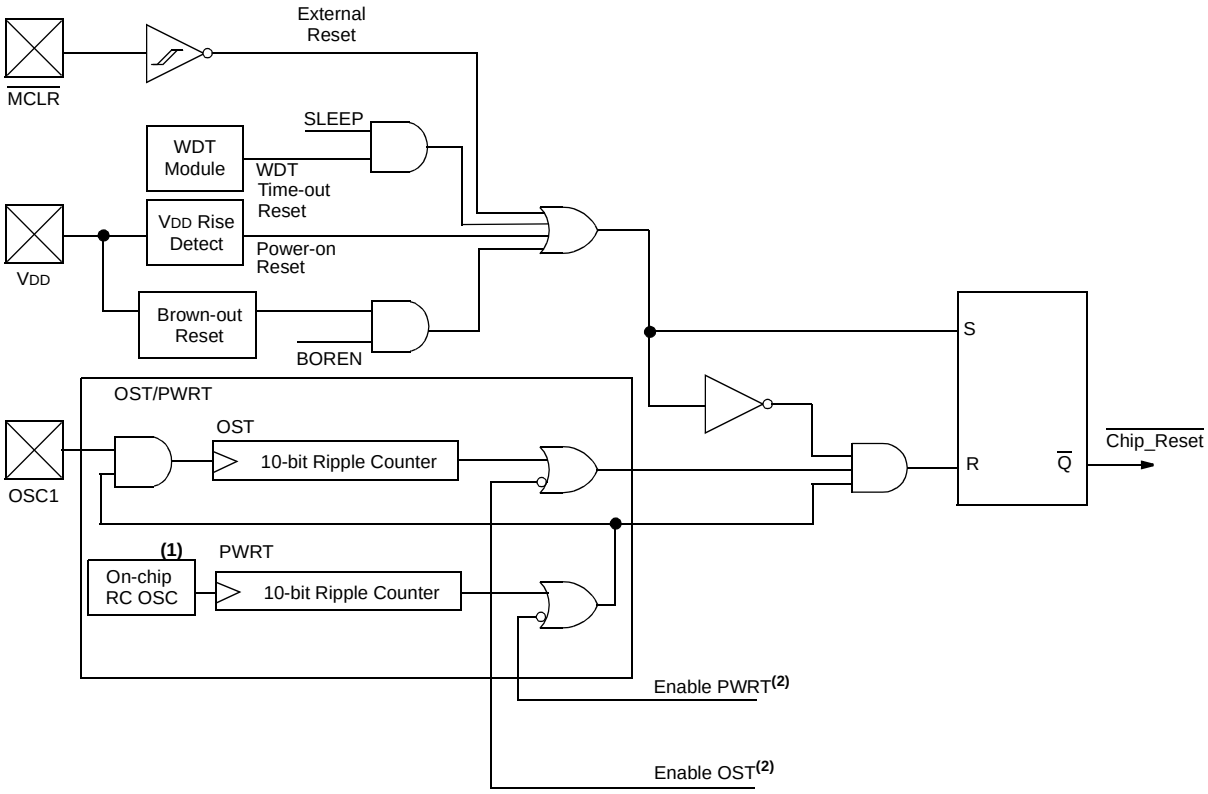
Some registers are not affected in any RESET condition; their status is unknown on POR and unchanged in any other RESET. Most other registers are reset to a "RESET state" on Power-on Reset (POR), on the $\overline{MCLR}$ and WDT Reset, and on $\overline{MCLR}$ Reset during SLEEP. They are not affected by a WDT Wake-up, which is viewed as the resumption of normal operation. The $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different RESET situations, as indicated in Table 12-4. These bits are used in software to determine the nature of the RESET. See Table 12-6 for a full description of RESET states of all registers.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 12-6.

The devices all have a $\overline{MCLR}$ noise filter in the $\overline{MCLR}$ Reset path. The filter will detect and ignore small pulses.

It should be noted that a WDT Reset does not drive $\overline{MCLR}$ pin low.

FIGURE 12-6: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is a separate oscillator from the RC oscillator of the CLKIN pin.

2: See Table 12-3 for various time-out situations.

12.4 Power-on Reset (POR), Power-up Timer (PWRT), Brown-out Reset (BOR) and Oscillator Start-up Timer (OST)

12.4.1 POWER-ON RESET (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.5V - 2.1V). To take advantage of the POR, just tie the MCLR pin directly (or through a resistor) to VDD. This will eliminate external RC components usually needed to create a Power-on Reset. A maximum rise time for VDD is specified. See Electrical Specifications for details.

When the device starts normal operation (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.

For additional information, refer to Application Note AN607, "*Power-up Trouble Shooting*."

12.4.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 72 ms nominal time-out on power-up only, 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 parameters for details.

12.4.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST), if enabled, provides 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay (if the PWRT is enabled). This helps to ensure 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.

12.4.4 BROWN-OUT RESET (BOR)

The configuration bit, BOREN, can enable or disable the Brown-out Reset circuit. If VDD falls below VBOR (parameter D005, about 4V) for longer than TBOR (parameter #35, about 100µS), the brown-out situation will reset the device. If VDD falls below VBOR for less than TBOR, a RESET may not occur.

Once the brown-out occurs, the device will remain in Brown-out Reset until VDD rises above VBOR. The Power-up Timer, if enabled, then keeps the device in RESET for TPWRT (parameter #33, about 72mS). If VDD should fall below VBOR during TPWRT, the Brown-out Reset process will restart when VDD rises above VBOR with the Power-up Timer Reset. The Power-up Timer is enabled separately from Brown-out Reset.

12.4.5 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 12-7, Figure 12-8, and Figure 12-9 depict time-out sequences on power-up.

Since the time-outs occur from the POR pulse, if MCLR is kept low long enough, the time-outs will expire. Then bringing MCLR high will begin execution immediately (Figure 12-8). This is useful for testing purposes or to synchronize more than one PIC16CXXX device operating in parallel.

Table 12-5 shows the RESET conditions for some special function registers, while Table 12-6 shows the RESET conditions for all the registers.

12.4.6 POWER CONTROL/STATUS REGISTER (PCON)

The Power Control/Status Register, PCON, has up to two bits depending upon the device.

Bit0 is Brown-out Reset Status bit, BOR. Bit BOR is unknown on a Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if bit BOR cleared, indicating a BOR occurred. When the Brown-out Reset is disabled, the state of the BOR bit is unpredictable and is, therefore, not valid at any time.

Bit1 is Power-on Reset Status bit POR. It is cleared on a Power-on Reset and unaffected otherwise. The user must set this bit following a Power-on Reset.

TABLE 12-3: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Wake-up from SLEEP
	PWRTE = 1	PWRTE = 0	
XT, HS, LP	1024TOSC	72 ms + 1024TOSC	1024 TOSC
RC	—	72 ms	—

TABLE 12-4: STATUS BITS AND THEIR SIGNIFICANCE

$\overline{POR}$	$\overline{BOR}$	$\overline{TO}$	$\overline{PD}$	Condition
0	x	1	1	Power-on Reset
0	x	0	x	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	x	x	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
1	0	1	1	Brown-out Reset
1	1	0	1	WDT Reset
1	1	0	0	WDT Wake-up
1	1	u	u	$\overline{MCLR}$ Reset during normal operation
1	1	1	0	$\overline{MCLR}$ Reset during SLEEP or interrupt wake-up from SLEEP

TABLE 12-5: RESET CONDITION FOR SPECIAL REGISTERS

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0x
$\overline{MCLR}$ Reset during normal operation	000h	000u uuuu	---- --uu
$\overline{MCLR}$ Reset during SLEEP	000h	0001 0uuu	---- --uu
WDT Reset	000h	0000 1uuu	---- --uu
WDT Wake-up	PC + 1	uuu0 0uuu	---- --uu
Brown-out Reset	000h	0001 1uuu	---- --u0
Interrupt wake-up from SLEEP	PC + 1**(1)**	uuu1 0uuu	---- --uu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'.

Note 1: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

TABLE 12-6: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Power-on Reset	$\overline{\text{MCLR}}$ Resets WDT Reset	Wake-up via WDT or Interrupt
W	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	N/A	N/A	N/A
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000h	0000h	PC + 1(2)
STATUS	0001 1xxx	000q quuu(3)	uuuq quuu(3)
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	--0x 0000	--0u 0000	--uu uuuu
PORTB	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	--xx xxxx	--uu uuuu	--uu uuuu
PORTD	0000 0000	0000 0000	uuuu uuuu
PORTE	0000 0000	0000 0000	uuuu uuuu
PCLATH	---0 0000	---0 0000	---u uuuu
INTCON	0000 000x	0000 000u	uuuu uuuu(1)
PIR1	00-- 0000	00-- 0000	uu-- uuuu(1)
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	--00 0000	--uu uuuu	--uu uuuu
TMR2	0000 0000	0000 0000	uuuu uuuu
T2CON	-000 0000	-000 0000	-uuu uuuu
SSPBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	0000 0000	0000 0000	uuuu uuuu
CCPR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	--00 0000	--00 0000	--uu uuuu
ADRES	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	0000 00-0	0000 00-0	uuuu uu-u
OPTION_REG	1111 1111	1111 1111	uuuu uuuu
TRISA	--11 1111	--11 1111	--uu uuuu
TRISB	1111 1111	1111 1111	uuuu uuuu
TRISC	--11 1111	--11 1111	--uu uuuu
TRISD	1111 1111	1111 1111	uuuu uuuu
TRISE	1111 1111	1111 1111	uuuu uuuu
PIE1	00-- 0000	00-- 0000	uu-- uuuu
PCON	---- --0-	---- --u-	---- --u-

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

Note 1: One or more bits in INTCON and/or PIR1 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 12-5 for RESET value for specific condition.

TABLE 12-6: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Power-on Reset	$\overline{\text{MCLR}}$ Resets WDT Reset	Wake-up via WDT or Interrupt
PR2	1111 1111	1111 1111	1111 1111
SSPADD	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	0000 0000	0000 0000	uuuu uuuu
ADCON1	---- -000	---- -000	---- -uuu
PORTF	0000 0000	0000 0000	uuuu uuuu
PORTG	0000 0000	0000 0000	uuuu uuuu
LCDSE	1111 1111	1111 1111	uuuu uuuu
LCDPS	---- 0000	---- 0000	---- uuuu
LCDCON	00-0 0000	00-0 0000	uu-u uuuu
LCDD00 to LCDD15	xxxx xxxx	uuuu uuuu	uuuu uuuu
TRISF	1111 1111	1111 1111	uuuu uuuu
TRISG	1111 1111	1111 1111	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

Note 1: One or more bits in INTCON and/or PIR1 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 12-5 for RESET value for specific condition.

FIGURE 12-7: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 1

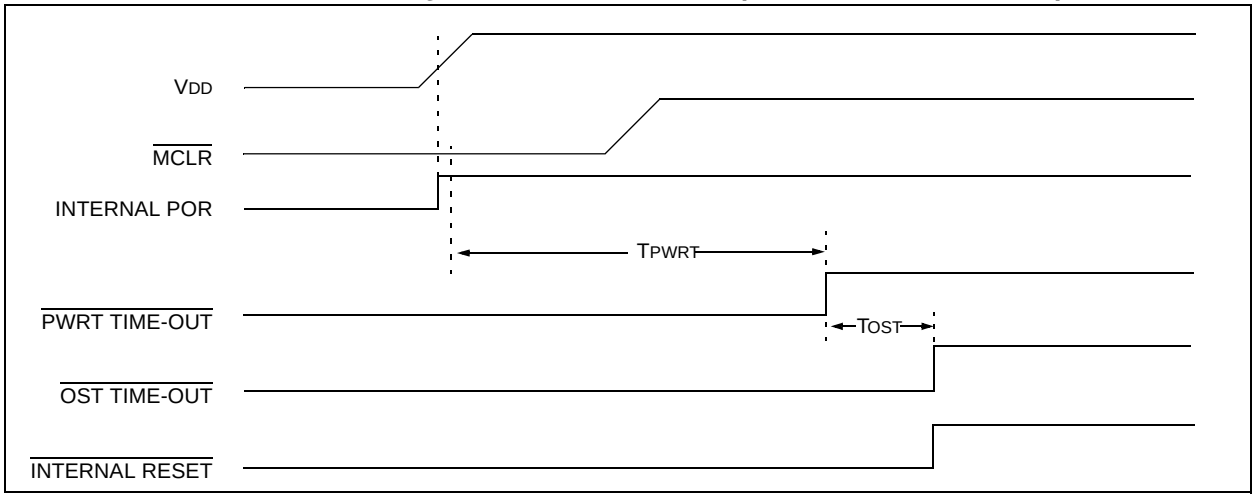


FIGURE 12-8: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 2

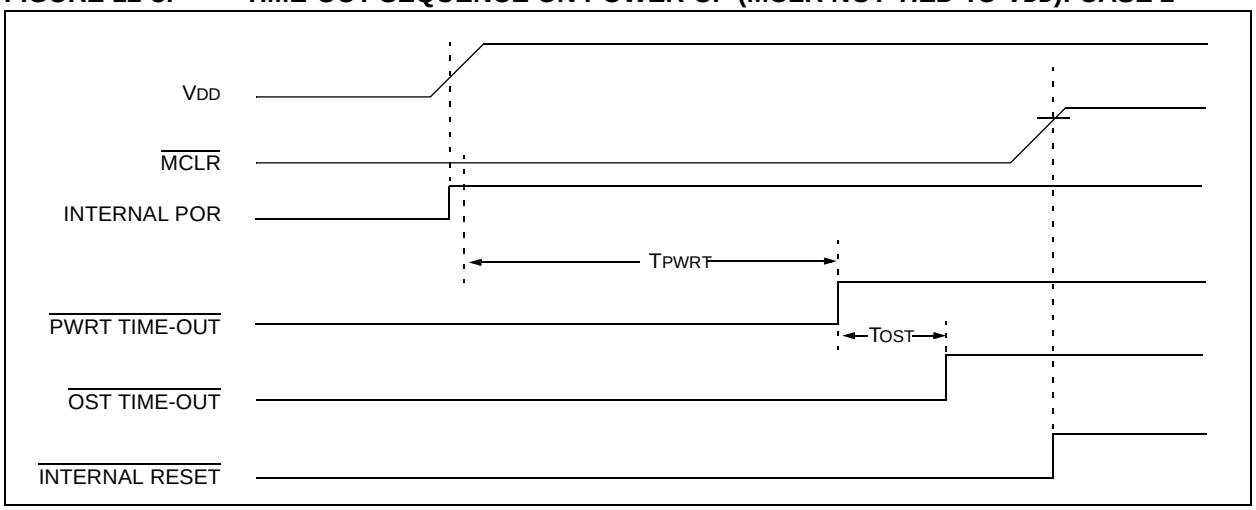
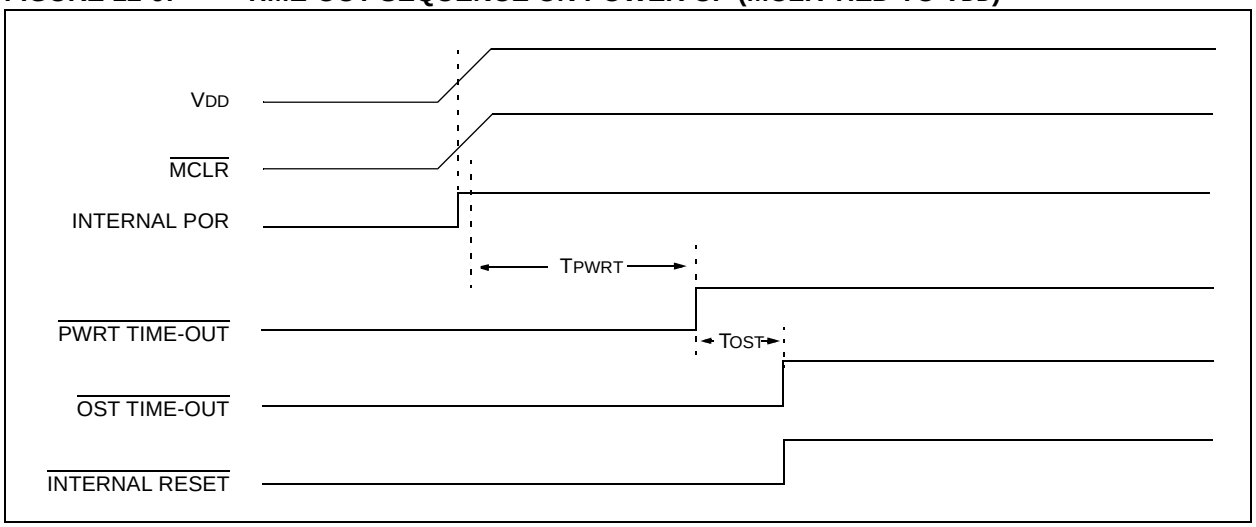


FIGURE 12-9: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO V_{DD})



12.5 Interrupts

The PIC16C925/926 family has nine sources of interrupt:

- External interrupt RB0/INT
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB7:RB4)
- A/D Interrupt
- TMR1 overflow interrupt
- TMR2 matches period interrupt
- CCP1 interrupt
- Synchronous serial port interrupt
- LCD module interrupt

The interrupt control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

Note:	Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit, or the GIE bit.

A global interrupt enable bit, GIE (INTCON<7>), enables (if set) all unmasked interrupts, or disables (if cleared) all interrupts. When bit GIE is enabled, and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt bits are set, regardless of the status of the GIE bit. The GIE bit is cleared on RESET.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine as well as sets the GIE bit, which re-enables interrupts.

The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flags are contained in the special function register, PIR1. The corresponding interrupt enable bits are contained in special function register, PIE1, and the peripheral interrupt enable bit is contained in special function register, INTCON.

When an interrupt is responded to, the GIE bit is cleared to disable any further interrupts, the return address is pushed onto the stack and the PC is loaded with 0004h. Once in the Interrupt Service Routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

For external interrupt events, such as the RB0/INT pin or RB Port change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 12-11). The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit, PEIE bit, or the GIE bit.

FIGURE 12-10: INTERRUPT LOGIC

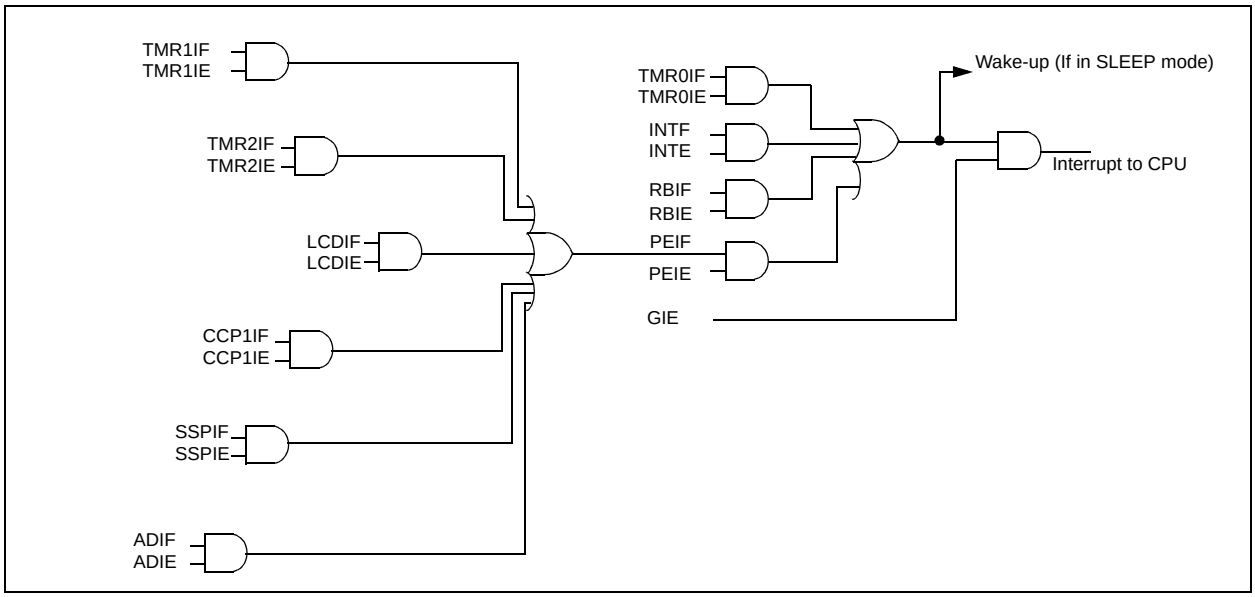
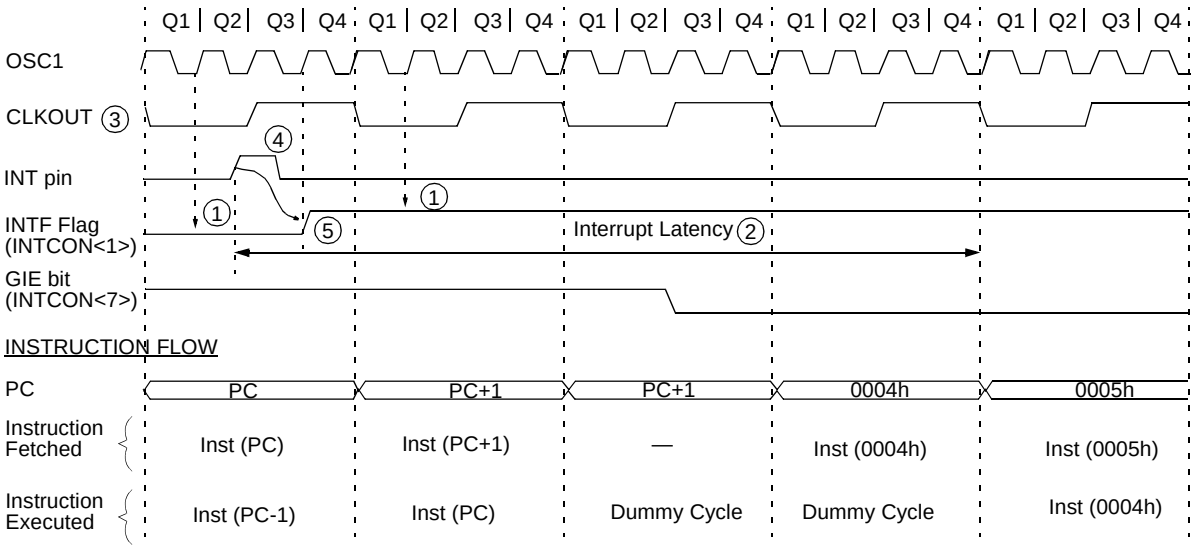


FIGURE 12-11: INT PIN INTERRUPT TIMING



Note 1: INTF flag is sampled here (every Q1).
2: Interrupt latency = 3-4 TCY where TCY = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.
3: CLKOUT is available only in RC oscillator mode.
4: For minimum width of INT pulse, refer to AC specs.
5: INTF can be set any time during the Q4-Q1 cycles.

12.5.1 INT INTERRUPT

External interrupt on RB0/INT pin is edge triggered: either rising if bit INTEDG (OPTION_REG<6>) is set, or falling, if the INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, flag bit INTF (INTCON<1>) is set. This interrupt can be disabled by clearing enable bit INTE (INTCON<4>). Flag bit INTF must be cleared in software in the Interrupt Service Routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from SLEEP, if bit INTE was set prior to going into SLEEP. The status of global interrupt enable bit, GIE, decides whether or not the processor branches to the interrupt vector following wake-up. See Section 12.8 for details on SLEEP mode.

12.5.2 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set flag bit, TMR0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>) (Section 5.0).

12.5.3 PORTB INTCON 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<4>) (Section 4.2).

12.6 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers during an interrupt, i.e., the W and STATUS registers. This will have to be implemented in software.

Example 12-1 stores and restores the STATUS, W, and PCLATH registers. The register, W_TEMP, must be defined in each bank and must be defined at the same offset from the bank base address (i.e., if W_TEMP is defined at 0x20 in bank 0, it must also be defined at 0xA0 in bank 1).

The code in the example:

e) Stores the W register.
f) Stores the STATUS register in bank 0.
g) Stores the PCLATH register.
h) Executes the ISR code.
i) Restores the STATUS register (and bank select bit).
j) Restores the W and PCLATH registers.

EXAMPLE 12-1: SAVING STATUS, W, AND PCLATH REGISTERS IN RAM

```
MOVWF     W_TEMP              ;Copy W to TEMP register, could be bank one or zero
SWAPF     STATUS,W            ;Swap status to be saved into W
CLRF      STATUS              ;bank 0, regardless of current bank, Clears IRP,RP1,RP0
MOVWF     STATUS_TEMP         ;Save status to bank zero STATUS_TEMP register
MOVF      PCLATH, W           ;Only required if using pages 1, 2 and/or 3
MOVWF     PCLATH_TEMP         ;Save PCLATH into W
CLRF      PCLATH              ;Page zero, regardless of current page
BCF       STATUS, IRP         ;Return to Bank 0
MOVF      FSR, W              ;Copy FSR to W
MOVWF     FSR_TEMP            ;Copy FSR from W to FSR_TEMP
    :
    :(ISR)                    ;Insert user code here
    :
MOVF      PCLATH_TEMP, W      ;Restore PCLATH
MOVWF     PCLATH              ;Move W into PCLATH
SWAPF     STATUS_TEMP,W       ;Swap STATUS_TEMP register into W
                              ;(sets bank to original state)
MOVWF     STATUS              ;Move W into STATUS register
SWAPF     W_TEMP,F            ;Swap W_TEMP
SWAPF     W_TEMP,W            ;Swap W_TEMP into W
```

12.7 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/CLKIN pin. That means that the WDT will run, even if the clock on the OSC1/CLKIN and OSC2/CLKOUT 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 WDT can be permanently disabled by clearing configuration bit WDTE (Section 12.1).

12.7.1 WDT PERIOD

The WDT has a nominal time-out period of 18 ms (with no prescaler). The time-out periods vary with temperature, VDD and process variations from part to part (see DC specs). If longer time-out periods are desired, a prescaler with a division ratio of up to 1:128 can be assigned to the WDT under software control, by writing to the OPTION register. Thus, time-out periods up to 2.3 seconds can be realized.

The CLRWDT and SLEEP instructions clear the WDT and the postscaler, if assigned to the WDT, prevent it from timing out and generating a device RESET condition.

The $\overline{TO}$ bit in the STATUS register will be cleared upon a Watchdog Timer time-out.

12.7.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken into account that under worst case conditions (VDD = Min., Temperature = Max., and Max. WDT prescaler) it may take several seconds before a WDT time-out occurs.

Note:	When a CLRWDT instruction is executed and the prescaler is assigned to the WDT, the prescaler count will be cleared, but the prescaler assignment is not changed.

FIGURE 12-12: WATCHDOG TIMER BLOCK DIAGRAM

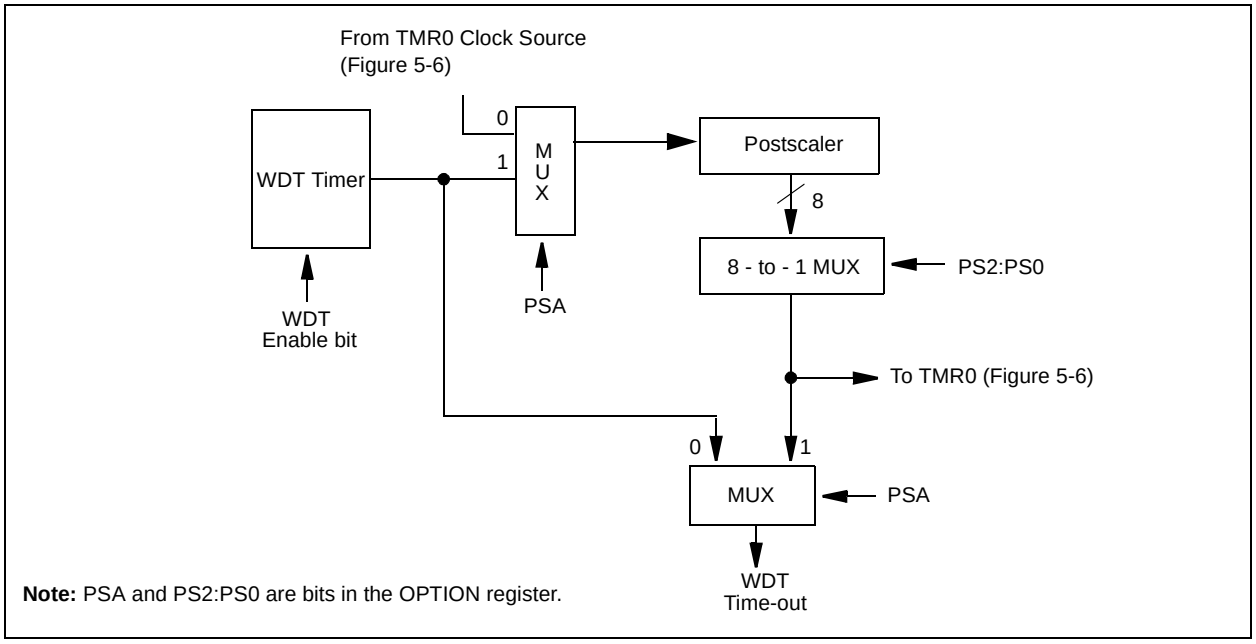


FIGURE 12-13: SUMMARY OF WATCHDOG TIMER REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
2007h	Config. bits	**(1)**	BOREN(1)	CP1	CP0	$\overline{PWRTE}$(1)	WDTE	FOSC1	FOSC0
81h, 181h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

Legend: Shaded cells are not used by the Watchdog Timer.
Note 1: See Register 12-1 for operation of these bits.

12.8 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 (STATUS<3>) is cleared, the $\overline{TO}$ (STATUS<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, 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 also be considered.

The $\overline{MCLR}$ pin must be at a logic high level (V_{IHMC}).

12.8.1 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 RB0/INT pin, RB port change, or peripheral interrupt.

External $\overline{MCLR}$ Reset will cause a device RESET. All other events are considered a continuation of program execution and cause a “wake-up”. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of 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).

The following peripheral interrupts can wake the device from SLEEP:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. SSP (START/STOP) bit detect interrupt.
3. SSP transmit or receive in Slave mode (SPI/I^2C).
4. CCP Capture mode interrupt.
5. A/D conversion (when A/D clock source is RC).
6. Special event trigger (Timer1 in Asynchronous mode using an external clock).
7. LCD module.

Other peripherals can not generate interrupts since during SLEEP, no on-chip Q clocks are present.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. 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 (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

12.8.2 WAKE-UP USING INTERRUPTS

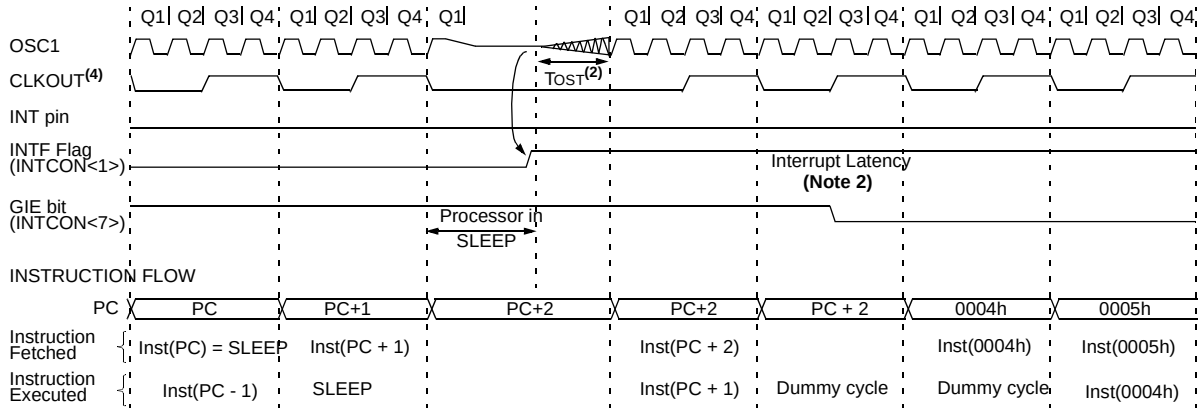
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt 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 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 12-14: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: XT, HS or LP oscillator mode assumed.
2: TOST = 1024TOSC (drawing not to scale) This delay will not be there for RC osc mode.
3: GIE = '1' assumed. In this case after wake-up, the processor jumps to the interrupt routine. If GIE = '0', execution will continue in-line.
4: CLKOUT is not available in these osc modes, but shown here for timing reference.

12.9 Program Verification/Code Protection

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

Note:	Microchip does not recommend code protecting windowed devices.

12.10 ID Locations

Four memory locations (2000h - 2003h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are not accessible during normal execution, but are readable and writable during program/verify. It is recommended that only the four Least Significant bits of the ID location are used.

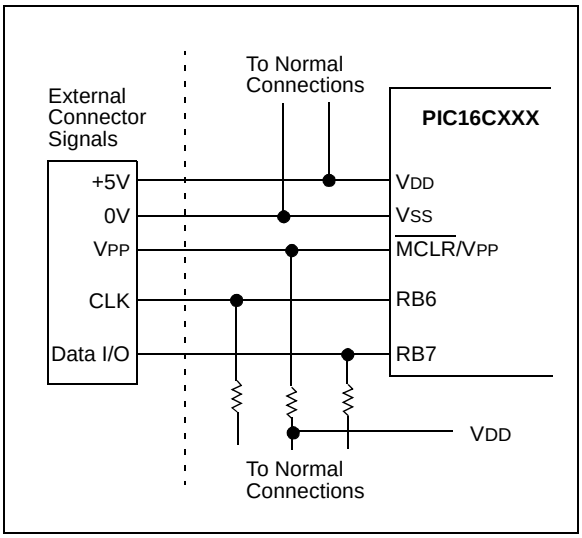
12.11 In-Circuit Serial Programming

PIC16CXXX 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.

The device is placed into a Program/Verify mode by holding the RB6 and RB7 pins low, while raising the $\overline{MCLR}$ (V_{PP}) pin from V_{IL} to V_{IHH} (see programming specification). RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode.

After RESET, to place the device into Program/Verify mode, the program counter (PC) is at location 00h. A 6-bit command is then supplied to the device. Depending on the command, 14-bits of program data are then supplied to or from the device, depending if the command was a load or a read. For complete details of serial programming, please refer to the PIC16C6X/7X Programming Specifications (Literature #DS30228).

FIGURE 12-15: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION



13.0 INSTRUCTION SET SUMMARY

Each PIC16CXXX instruction is a 14-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 PIC16CXXX instruction set summary in Table 13-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 13-1 shows the opcode field descriptions.

The instruction set is highly orthogonal and is grouped into three basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal and control** operations

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator which selects the number of the bit affected by the operation, while 'f' represents the address of the file in which the bit is located.

For **literal and control** operations, 'k' represents an eight or eleven bit constant or literal value.

FIGURE 13-1: GENERAL FORMAT FOR INSTRUCTIONS

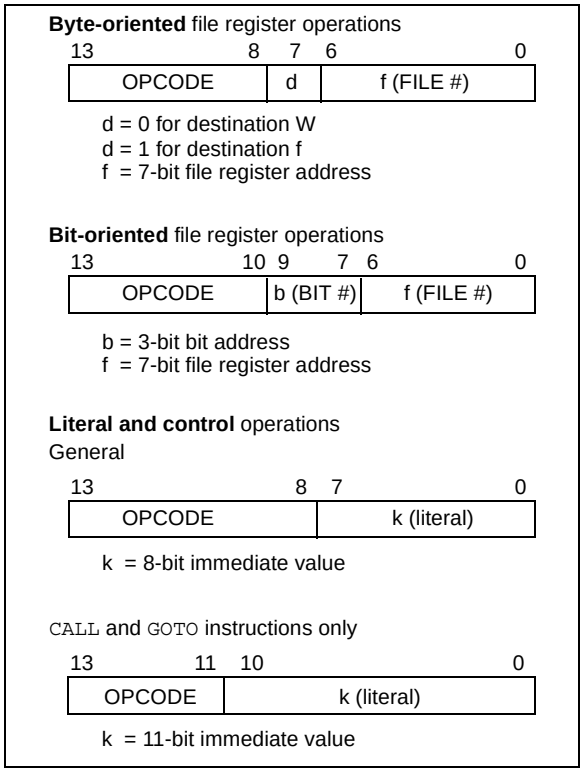


TABLE 13-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 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.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1.
label	Label name
TOS	Top-of-Stack
PC	Program Counter
PCLATH	Program Counter High Latch
GIE	Global Interrupt Enable bit
WDT	Watchdog Timer/Counter
$\overline{TO}$	Time-out bit
$\overline{PD}$	Power-down bit
dest	Destination either the W register or the specified register file location
[ ]	Options
( )	Contents
→	Assigned to
< >	Register bit field
∈	In the set of
italics	User defined term (font is courier)

All instructions are executed within one single instruction cycle, unless a conditional test is true, or the program counter is changed, as a result of an instruction. In this case, the execution takes two instruction cycles, with the second cycle executed as a NOP. 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.

Table 13-2 lists the instructions recognized by the MPASM™ assembler.

Figure 13-1 shows the general formats that the instructions can have.

Note:	To maintain upward compatibility with future PIC16CXXX products, do not use the OPTION and TRIS instructions.

All examples use the format '0xnn' to represent a hexadecimal number.

TABLE 13-2: PIC16CXXX INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from interrupt	2	00 0000 0000 1001		
RETLW	k	Return with literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

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`.

13.1 Instruction Descriptions

ADDLW **Add Literal and W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → (W)

Status Affected: C, DC, Z

Encoding:

11	111x	kkkk	kkkk

Description: The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register.

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 and f**

Syntax: [*label*] ADDWF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W) + (f) → (destination)

Status Affected: C, DC, Z

Encoding:

00	0111	dfff	ffff

Description: Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `ADDWF FSR, 0`

Before Instruction:
W = 0x17
FSR = 0xC2

After Instruction:
W = 0xD9
FSR = 0xC2

ANDLW **AND Literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. (k) → (W)

Status Affected: Z

Encoding:

11	1001	kkkk	kkkk

Description: The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.

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]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W).AND. (f) → (destination)

Status Affected: Z

Encoding:

00	0101	dfff	ffff

Description: AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `ANDWF FSR, 1`

Before Instruction:
W = 0x17
FSR = 0xC2
After Instruction
W = 0x17
FSR = 0x02

BCF **Bit Clear f**

Syntax: [*label*] BCF f [,b]

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 0 → (f<b>)

Status Affected: None

Encoding:

01	00bb	bfff	ffff

Description: Bit 'b' in register 'f' is cleared.

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
```
Before Instruction:
FLAG_REG = 0xC7
After Instruction:
FLAG_REG = 0x47

BSF **Bit Set f**

Syntax: [*label*] BSF f [,b]

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 1 → (f<b>)

Status Affected: None

Encoding:

01	01bb	bfff	ffff

Description: Bit 'b' in register 'f' is set.

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
```
Before Instruction:
FLAG_REG = 0x0A
After Instruction:
FLAG_REG = 0x8A

BTFSC **Bit Test, Skip if Clear**

Syntax: [*label*] BTFSC f [,b]

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

01	10bb	bfff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is executed.
If bit 'b' in register 'f' is '0', then the next instruction is discarded, and a NOP is executed instead, making this a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	No Operation

If Skip: (2nd Cycle)

Q1	Q2	Q3	Q4
No Operation	No Operation	No Operation	No Operation

Example
```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CODE
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 f, Skip if Set**

Syntax: [*label*] BTFSS f [,b]

Operands: 0 ≤ f ≤ 127
0 ≤ b < 7

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

01	11bb	bfff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is executed.
If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	No Operation

If Skip: (2nd Cycle)

Q1	Q2	Q3	Q4
No Operation	No Operation	No Operation	No Operation

Example

```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

Before Instruction:
PC = address HERE

After Instruction:
if FLAG<1> = 0,
PC = address FALSE
if FLAG<1> = 1,
PC = address TRUE

CALL **Call Subroutine**

Syntax: [*label*] CALL k

Operands: 0 ≤ k ≤ 2047

Operation: (PC)+ 1→ TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Status Affected: None

Encoding:

10	0kkk	kkkk	kkkk

Description: Call Subroutine. First, return address (PC+1) is pushed onto the stack. The eleven-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	Read literal 'k', Push PC to Stack	Process data	Write to PC
2nd Cycle	No Operation	No Operation	No Operation	No Operation

Example

```
HERE    CALL    THERE
```

Before Instruction:
PC = Address HERE

After Instruction:
PC = Address THERE
TOS = Address HERE+1

CLRF **Clear f**

Syntax: [*label*] CLRF f

Operands: $0 \le f \le 127$

Operation: $00h \rightarrow (f)$
$1 \rightarrow Z$

Status Affected: Z

Encoding:

00	0001	1fff	ffff

Description: The contents of register 'f' are cleared and the Z bit is set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example `CLRF FLAG_REG`

Before Instruction:
FLAG_REG = 0x5A

After Instruction:
FLAG_REG = 0x00
Z = 1

CLRW **Clear W**

Syntax: [*label*] CLRW

Operands: None

Operation: $00h \rightarrow (W)$
$1 \rightarrow Z$

Status Affected: Z

Encoding:

00	0001	0xxx	xxxx

Description: W register is cleared. Zero bit (Z) is set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No Operation	Process data	Write to W

Example `CLRW`

Before Instruction:
W = 0x5A

After Instruction:
W = 0x00
Z = 1

CLRWDT **Clear Watchdog Timer**

Syntax: [*label*] CLRWDT

Operands: None

Operation: 00h → WDT
0 → WDT prescaler,
1 → $\overline{TO}$
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

00	0000	0110	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler 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	Clear WDT Counter

Example CLRWDT

Before Instruction:
WDT counter = ?

After Instruction:
WDT counter = 0x00
WDT prescaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: [*label*] COMF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: $(\bar{f}) \rightarrow$ (destination)

Status Affected: Z

Encoding:

00	1001	dfff	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'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example COMF REG1,0

Before Instruction:
REG1 = 0x13

After Instruction:
REG1 = 0x13
W = 0xEC

DECF **Decrement f**

Syntax: [*label*] DECF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) - 1 → (destination)

Status Affected: Z

Encoding:

00	0011	dfff	ffff

Description: Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example
```
DECF    CNT,  1
```
Before Instruction:
CNT = 0x01
Z = 0

After Instruction:
CNT = 0x00
Z = 1

DECFSZ **Decrement f, Skip if 0**

Syntax: [*label*] DECFSZ f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) - 1 → (destination); skip if result = 0

Status Affected: None

Encoding:

00	1011	dfff	ffff

Description: The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

If Skip: (2nd Cycle)

Q1	Q2	Q3	Q4
No Operation	No Operation	No Operation	No Operation

Example
```
HERE     DECFSZ   CNT, 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+1

GOTO **Unconditional Branch**

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 2047$

Operation: $k \rightarrow PC<10:0>$
$PCLATH<4:3> \rightarrow PC<12:11>$

Status Affected: None

Encoding:

10	1kkk	kkkk	kkkk

Description: GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	Read literal 'k'	Process data	Write to PC
2nd Cycle	No Operation	No Operation	No Operation	No Operation

Example GOTO THERE

After Instruction:
PC = Address THERE

INCF **Increment f**

Syntax: [*label*] INCF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) + 1 → (destination)

Status Affected: Z

Encoding:

00	1010	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example INCF CNT, 1

Before Instruction:
CNT = 0xFF
Z = 0

After Instruction:
CNT = 0x00
Z = 1

INCFSZ **Increment f, Skip if 0**

Syntax: [*label*] INCFSZ f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) + 1 → (destination), skip if result = 0

Status Affected: None

Encoding:

00	1111	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

If Skip: (2nd Cycle)

Q1	Q2	Q3	Q4
No Operation	No Operation	No Operation	No Operation

Example
```
HERE     INCFSZ CNT,   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 +1

IORLW **Inclusive OR Literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → (W)

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

Description: The contents of the W register is OR'ed with the eight-bit literal 'k'. The result is placed in the W register.

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
Z = 0

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W).OR. (f) → (destination)

Status Affected: $\overline{Z}$

Encoding:

00	0100	dfff	ffff

Description: Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `IORWF RESULT, 0`

Before Instruction:
RESULT = 0x13
W = 0x91

After Instruction:
RESULT = 0x13
W = 0x93
Z = 0

MOVF **Move f**

Syntax: [*label*] MOVF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) → (destination)

Status Affected: Z

Encoding:

00	1000	dfff	ffff

Description: The contents of register f are moved to a destination dependant upon the status of d. If d = 0, the destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register, since status flag Z is affected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `MOVF FSR, 0`

After Instruction:
W = value in FSR register
Z = 1 if W = 0

MOVLW **Move Literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: k → (W)

Status Affected: None

Encoding:

11	00xx	kkkk	kkkk

Description: The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

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

Operands: $0 \le f \le 127$

Operation: (W) → (f)

Status Affected: None

Encoding:

00	0000	1fff	ffff

Description: Move data from W register to register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example

```
MOVWF     OPTION_REG
```

Before Instruction:
OPTION = 0xFF
W = 0x4F

After Instruction:
OPTION = 0x4F
W = 0x4F

NOP **No Operation**

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

00	0000	0xx0	0000

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No Operation	No Operation	No Operation

Example

```
NOP
```

OPTION **Load Option Register**

Syntax: [*label*] OPTION

Operands: None

Operation: (W) → OPTION

Status Affected: None

Encoding:

00	0000	0110	0010

Description: The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it.

Words: 1

Cycles: 1

Example

To maintain upward compatibility with future PIC16CXXX products, do not use this instruction.

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE

Operands: None

Operation: TOS → PC,
1 → GIE

Status Affected: None

Encoding:

00	0000	0000	1001

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	No Operation	Set the GIE bit	Pop from the Stack
2nd Cycle	No Operation	No Operation	No Operation	No Operation

Example `RETFIE`

After Interrupt:
PC = TOS
GIE = 1

RETLW **Return with Literal in W**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W);
TOS → PC

Status Affected: None

Encoding:

11	01xx	kkkk	kkkk

Description: The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	Read literal 'k'	No Operation	Write to W, Pop from the Stack
2nd Cycle	No Operation	No Operation	No Operation	No Operation

Example

```
CALL TABLE ;W contains table
           ;offset value
           ;W now has table value
           •
           •
TABLE ADDWF PC   ;W = offset
      RETLW k1   ;Begin table
      RETLW k2   ;
      •
      •
      •
      RETLW kn   ; End of table
```

Before Instruction:
W = 0x07

After Instruction:
W = value of k8

RETURN **Return from Subroutine**

Syntax: [*label*] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Encoding:

00	0000	0000	1000

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	No Operation	No Operation	Pop from the Stack
2nd Cycle	No Operation	No Operation	No Operation	No Operation

Example `RETURN`

After Interrupt:
PC = TOS

RLF **Rotate Left f through Carry**

Syntax: [*label*] RLF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

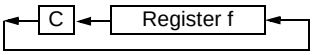
Operation: See description below

Status Affected: C

Encoding:

00	1101	dfff	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 the W register. If 'd' is 1, the result is stored back in register 'f'.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `RLF REG1,0`

Before Instruction:
REG1 = 1110 0110
C = 0

After Instruction:
REG1 = 1110 0110
W = 1100 1100
C = 1

RRF **Rotate Right f through Carry**

Syntax: [*label*] RRF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

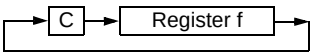
Operation: See description below

Status Affected: C

Encoding:

00	1100	dfff	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 the W register. If 'd' is 1, the result is placed back in register 'f'.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example
```
RRF             REG1,0
```
Before Instruction:
```
REG1  =  1110 0110
C     =  0
```
After Instruction:
```
REG1  =  1110 0110
W     =  0111 0011
C     =  0
```

SLEEP

Syntax: [*label*] SLEEP

Operands: None

Operation: 00h → WDT,
0 → $\overline{\text{WDT}}$ prescaler,
1 → $\overline{\text{TO}}$,
0 → $\overline{\text{PD}}$

Status Affected: $\overline{\text{TO}}$, $\overline{\text{PD}}$

Encoding:

00	0000	0110	0011

Description: The power-down status bit, $\overline{\text{PD}}$ is cleared. Time-out status bit, $\overline{\text{TO}}$ is set. Watchdog Timer and its prescaler are cleared.
The processor is put into SLEEP mode with the oscillator stopped. See Section 12.8 for more details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No Operation	No Operation	Go to Sleep

Example:
```
SLEEP
```


SUBLW **Subtract W from Literal**

Syntax: [*label*] SUBLW k

Operands: $0 \le k \le 255$

Operation: k - (W) → (W)

Status Affected: C, DC, Z

Encoding:

11	110x	kkkk	kkkk

Description: The W register is subtracted (2's complement method) from the eight-bit literal 'k'. The result is placed in the W register.

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 = ?
Z = ?

After Instruction:
W = 1
C = 1; result is positive
Z = 0

Example 2:

Before Instruction:
W = 2
C = ?
Z = ?

After Instruction:
W = 0
C = 1; result is zero
Z = 1

Example 3:

Before Instruction:
W = 3
C = ?
Z = ?

After Instruction:
W = 0xFF
C = 0; result is negative
Z = 0

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) - (W) → (destination)

Status Affected: C, DC, Z

Encoding:

00	0010	dfff	ffff

Description: Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example 1: `SUBWF REG1,1`

Before Instruction:
REG1 = 3
W = 2
C = ?
Z = ?

After Instruction:
REG1 = 1
W = 2
C = 1; result is positive
Z = 0

Example 2:

Before Instruction:
REG1 = 2
W = 2
C = ?
Z = ?

After Instruction:
REG1 = 0
W = 2
C = 1; result is zero
Z = 1

Example 3:

Before Instruction:
REG1 = 1
W = 2
C = ?
Z = ?

After Instruction:
REG1 = 0xFF
W = 2
C = 0; result is negative
Z = 0

SWAPF **Swap Nibbles in f**

Syntax: [*label*] SWAPF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f<3:0>) → (destination<7:4>),
(f<7:4>) → (destination<3:0>)

Status Affected: None

Encoding:

00	1110	dfff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example
```
SWAPF   REG,    0
```
Before Instruction:
REG1 = 0xA5

After Instruction:
REG1 = 0xA5
W = 0x5A

TRIS **Load TRIS Register**

Syntax: [*label*] TRIS f

Operands: $5 \le f \le 7$

Operation: (W) → TRIS register f;

Status Affected: None

Encoding:

00	0000	0110	0fff

Description: The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.

Words: 1

Cycles: 1

Example

To maintain upward compatibility with future PIC16CXXX products, do not use this instruction.

XORLW **Exclusive OR Literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → (W)

Status Affected: Z

Encoding:

11	1010	kkkk	kkkk

Description: The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.

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]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W) .XOR. (f) → (destination)

Status Affected: Z

Encoding:

00	0110	dfff	ffff

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `XORWF REG 1`

Before Instruction:
REG = 0xAF
W = 0xB5

After Instruction:
REG = 0x1A
W = 0xB5

NOTES:

14.0 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
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - ICEPIC™ In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD for PIC16F87X
- Device Programmers
  - PRO MATE® II Universal Device Programmer
  - PICSTART® Plus Entry-Level Development Programmer
- Low Cost Demonstration Boards
  - PICDEM™ 1 Demonstration Board
  - PICDEM 2 Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 17 Demonstration Board
  - KEELOQ® Demonstration Board

14.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8-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
- A project manager
- Customizable toolbar and key mapping
- A status bar
- 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
  - absolute listing file
  - machine code

The ability to use MPLAB IDE with multiple debugging tools allows users to easily switch from the cost-effective simulator to a full-featured emulator with minimal retraining.

14.2 MPASM Assembler

The MPASM assembler is a full-featured universal macro assembler for all PICmicro MCU’s.

The MPASM assembler has a command line interface and a Windows shell. It can be used as a stand-alone application on a Windows 3.x or greater system, or it can be used through MPLAB IDE. 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, an absolute LST file that contains source lines and generated machine code, and a COD file 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.

14.3 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, respectively. These compilers provide powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compilers provide symbol information that is compatible with the MPLAB IDE memory display.

14.4 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 also link relocatable objects from pre-compiled libraries, using directives from a linker script.

The MPLIB object librarian is a librarian for pre-compiled code to be used with the MPLINK object linker. When a routine from a library is called from another 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 MPLIB object librarian manages the creation and modification of library files.

The MPLINK object linker features include:

- Integration with MPASM assembler and MPLAB C17 and MPLAB C18 C compilers.
- Allows all memory areas to be defined as sections to provide link-time flexibility.

The MPLIB object librarian features include:

- Easier linking because single libraries can be included instead of many smaller files.
- Helps keep code maintainable by grouping related modules together.
- Allows libraries to be created and modules to be added, listed, replaced, deleted or extracted.

14.5 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 of the pins. 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 the MPLAB C18 C compilers and the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent multi-project software development tool.

14.6 MPLAB ICE High Performance Universal In-Circuit Emulator with MPLAB IDE

The MPLAB ICE universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers (MCUs). Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment (IDE), 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 in-circuit emulator system has been designed as a real-time emulation system, with advanced features that are generally found on more expensive development tools. The PC platform and Microsoft® Windows environment were chosen to best make these features available to you, the end user.

14.7 ICEPIC In-Circuit Emulator

The ICEPIC low cost, in-circuit emulator is a solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X and PIC16CXXX families of 8-bit One-Time-Programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules, or daughter boards. The emulator is capable of emulating without target application circuitry being present.

14.8 MPLAB ICD In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD, is a powerful, low cost, run-time development tool. This tool is based on the FLASH PIC16F87X and can be used to develop for this and other PICmicro microcontrollers from the PIC16CXXX family. The MPLAB ICD utilizes the in-circuit debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming™ 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 watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time.

14.9 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable V_{DD} and V_{PP} supplies, which allow it to verify programmed memory at V_{DD} min and V_{DD} max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In stand-alone mode, the PRO MATE II device programmer can read, verify, or program PICmicro devices. It can also set code protection in this mode.

14.10 PICSTART Plus Entry Level 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 all PICmicro devices with 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.

14.11 PICDEM 1 Low Cost PICmicro Demonstration Board

The PICDEM 1 demonstration board is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: 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 user can program the sample microcontrollers provided with the PICDEM 1 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The user can also connect the PICDEM 1 demonstration board to the MPLAB ICE in-circuit emulator and download the firmware to the emulator for testing. A prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs connected to PORTB.

14.12 PICDEM 2 Low Cost PIC16CXX Demonstration Board

The PICDEM 2 demonstration board is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 2 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a serial EEPROM to demonstrate usage of the I^2C™ bus and separate headers for connection to an LCD module and a keypad.

14.13 PICDEM 3 Low Cost PIC16CXXX Demonstration Board

The PICDEM 3 demonstration board is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with an LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 3 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer with an adapter socket, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 3 demonstration board to test firmware. A prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM 3 demonstration board is a LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM 3 demonstration board provides an additional RS-232 interface and Windows software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals.

14.14 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. All necessary hardware is included to run basic demo programs, which are supplied on a 3.5-inch disk. A programmed sample is included and the user may erase it and program it with the other sample programs using the PRO MATE II device programmer, or the PICSTART Plus development programmer, and easily debug and test the sample code. In addition, the PICDEM 17 demonstration board supports downloading of programs to and executing out of external FLASH memory on board. The PICDEM 17 demonstration board is also usable with the MPLAB ICE in-circuit emulator, or the PICMASTER emulator and all of the sample programs can be run and modified using either emulator. Additionally, a generous prototype area is available for user hardware.

14.15 KEELOQ Evaluation and Programming Tools

KEELOQ evaluation and programming tools support Microchip's HCS Secure Data Products. The HCS evaluation kit includes a LCD display to show changing codes, a decoder to decode transmissions and a programming interface to program test transmitters.

TABLE 14-1: DEVELOPMENT TOOLS FROM MICROCHIP

		PIC12CXXX	PIC14000	PIC16C5X	PIC16C6X	PIC16CXXX	PIC16F62X	PIC16C7X	PIC16C7XX	PIC16C8X	PIC16F8XX	PIC16C9XX	PIC17C4X	PIC17C7XX	PIC18CXX2	24CXX/ 25CXX/ 93CXX	HCSXXX	MCRFXXX	MCP2510
Software Tools	MPLAB® Integrated Development Environment	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				
	MPLAB® C17 C Compiler												✓	✓					
	MPLAB® C18 C Compiler														✓				
	MPASM™ Assembler/ MPLINK™ Object Linker	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		
Emulators	MPLAB® ICE In-Circuit Emulator	✓	✓	✓	✓	✓	✓**	✓	✓	✓	✓	✓	✓	✓	✓				
	ICEPIC™ In-Circuit Emulator	✓		✓	✓	✓		✓	✓			✓							
Debugger	MPLAB® ICD In-Circuit Debugger				✓*			✓*			✓								
Programmers	PICSTART® Plus Entry Level Development Programmer	✓	✓	✓	✓	✓	✓**	✓	✓	✓	✓	✓	✓	✓	✓				
	PRO MATE® II Universal Device Programmer	✓	✓	✓	✓	✓	✓**	✓	✓	✓	✓	✓	✓	✓	✓				
Demo Boards and Eval Kits	PICDEM™ 1 Demonstration Board			✓		✓		✓†		✓			✓						
	PICDEM™ 2 Demonstration Board				✓†			✓†							✓				
	PICDEM™ 3 Demonstration Board											✓							
	PICDEM™ 14A Demonstration Board		✓																
	PICDEM™ 17 Demonstration Board													✓					
	KEELOQ® Evaluation Kit																✓		
	KEELOQ® Transponder Kit																✓		
	microID™ Programmer's Kit																	✓	
	125 kHz microID™ Developer's Kit																	✓	
	125 kHz Anticollision microID™ Developer's Kit																	✓	
	13.56 MHz Anticollision microID™ Developer's Kit																	✓	
	MCP2510 CAN Developer's Kit																		✓

* Contact the Microchip Technology Inc. web site at www.microchip.com for information on how to use the MPLAB® ICD In-Circuit Debugger (DV164001) with PIC16C62, 63, 64, 65, 72, 73, 74, 76, 77.

** Contact Microchip Technology Inc. for availability date.

† Development tool is available on select devices.

NOTES:

15.0 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} ..... 0V to +7.5V
Voltage on $\overline{MCLR}$ with respect to V_{SS} ..... 0V to +13.25V
Voltage on RA4 with respect to V_{SS} ..... 0V to +8.5V
Voltage on VLCD2, VLCD3 with respect to V_{SS} ..... 0V to +10V
Total power dissipation **(Note 1)** ..... 1.0 W
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 combined ..... 200 mA
Maximum current sourced by all Ports combined ..... 200 mA

Note 1: Power dissipation is calculated as follows: $P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OI} \times I_{OL})$

† NOTICE: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 15-1: PIC16C925/926 VOLTAGE-FREQUENCY GRAPH

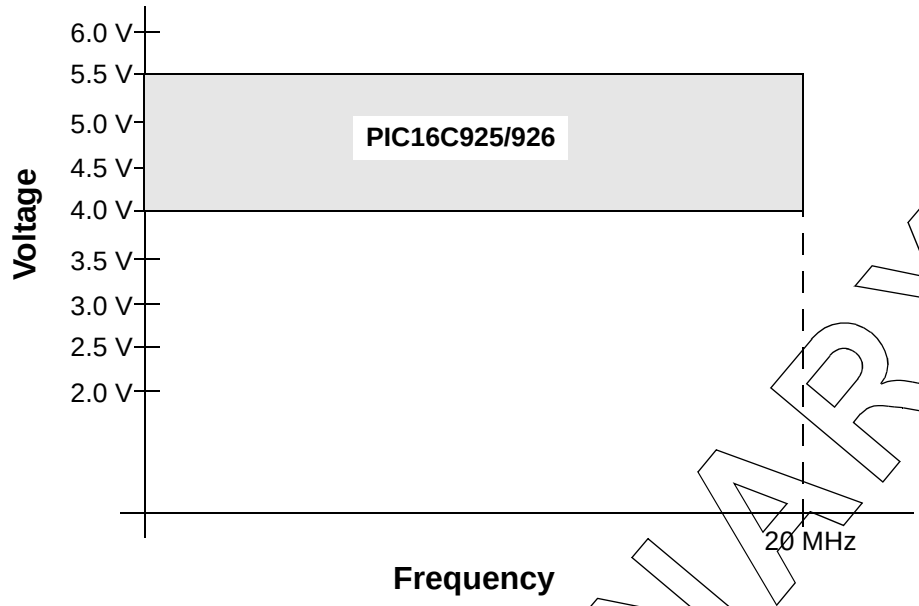
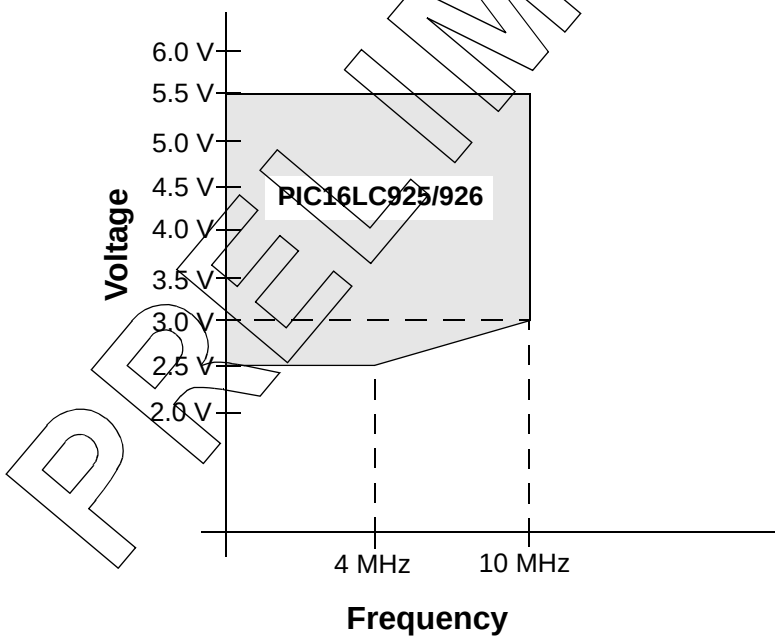


FIGURE 15-2: PIC16LC925/926 VOLTAGE-FREQUENCY GRAPH



FMAX = (6.0 MHz/V) (VDDAPPMIN - 2.0 V) + 4 MHz

Note 1: VDDAPPMIN is the minimum voltage of the PICmicro® device in the application.

2: FMAX has a maximum frequency of 10MHz.

15.1 DC Characteristics

PIC16LC925/926 (Commercial, Industrial)			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
PIC16C925/926 (Commercial, Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VDD	**Supply Voltage**					
D001 D001A		PIC16LC925/926	2.5 4.5	— —	5.5 5.5	V V	LP, XT and RC osc configuration HS osc configuration
D001 D001A		PIC16C925/926	4.0 4.5	— —	5.5 5.5	V V	XT, RC and LP osc configuration HS osc configuration
D002	VDR	**RAM Data Retention Voltage (Note 1)**	—	1.5	—	V	Device in SLEEP mode
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	VSS	—	V	See Power-on Reset section for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See Power-on Reset section for details **(Note 6)**
D005	VBOR	**Brown-out Reset** voltage trip point	3.65	—	4.35	V	BODEN bit set
	IDD	**Supply Current (Note 2)**					
D010 D011		PIC16LC925/926	— —	.6 225	2.0 48	mA µA	XT and RC osc configuration FOSC = 4 MHz, VDD = 3.0V **(Note 4)** LP osc configuration FOSC = 32 kHz, VDD = 3.0V, WDT disabled
D010 D011 D012		PIC16C925/926	— — —	2.7 35 7	5 70 10	mA µA mA	XT and RC osc configuration FOSC = 4 MHz, VDD = 5.5V **(Note 4)** LP osc configuration FOSC = 32 kHz, VDD = 4.0V HS osc configuration FOSC = 20 MHz, VDD = 5.5V

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, 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.

3: 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 hi-impedance state and tied to VDD and VSS.

4: For RC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kOhm.

5: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

6: PWRT must be enabled for slow ramps.

7: ΔILCDT1 and ΔILCDRC includes the current consumed by the LCD Module and the voltage generation circuitry. This does not include current dissipated by the LCD panel.

15.1 DC Characteristics (Continued)

PIC16LC925/926 (Commercial, Industrial)	Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial
PIC16C925/926 (Commercial, Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	IPD	**Power-down Current (Note 3)**					
D020		PIC16LC925/926	—	0.9	5	µA	VDD = 3.0V
D020		PIC16C925/926	—	1.5	21	µA	VDD = 4.0V
		Module Differential Current (Note 5)					
D021	ΔIWDT	**Watchdog Timer** PIC16LC925/926	—	6.0	20	µA	VDD = 3.0V
D021		**Watchdog Timer** PIC16C925/926	—	9.0	25	µA	VDD = 4.0V
D022	ΔILCDT1	**LCD Voltage Generation** with internal RC osc enabled PIC16LC925/926	—	36	50	µA	VDD = 3.0V **(Note 7)**
D022		**LCD Voltage Generation** with internal RC osc enabled PIC16C925/926	—	40	55	µA	VDD = 4.0V **(Note 7)**
D022A	ΔIBOR	**Brown-out Reset**	—	100	150	µA	BODEN bit set, VDD = 5.0
D024	ΔILCDT1	**LCD Voltage Generation** with Timer1 @ 32.768 kHz PIC16LC925/926	—	15	29	µA	VDD = 3.0V **(Note 7)**
D024		**LCD Voltage Generation** with Timer1 @ 32.768 kHz PIC16C925/926	—	33	60	µA	VDD = 4.0V **(Note 7)**

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, 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.

3: 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 hi-impedance state and tied to VDD and VSS.

4: For RC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kOhm.

5: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

6: PWRT must be enabled for slow ramps.

7: ΔILCDT1 and ΔILCDRC includes the current consumed by the LCD Module and the voltage generation circuitry. This does not include current dissipated by the LCD panel.

15.1 DC Characteristics (Continued)

PIC16LC925/926 (Commercial, Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
PIC16C925/926 (Commercial, Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D025	ΔIT1OSC	**Timer1 Oscillator** PIC16LC925/926	—	—	50	µA	VDD = 3.0V
D025		**Timer1 Oscillator** PIC16C925/926	—	—	50	µA	VDD = 4.0V
D026	ΔIAD	**A/D Converter** PIC16LC925/926	—	1.0	—	µA	A/D on, not converting
D026		**A/D Converter** PIC16C925/926	—	1.0	—	µA	A/D on, not converting

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, 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.

3: 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 hi-impedance state and tied to VDD and VSS.

4: For RC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula IR = VDD/2REXT (mA) with REXT in kOhm.

5: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

6: PWRT must be enabled for slow ramps.

7: ΔILCDT1 and ΔILCDRC includes the current consumed by the LCD Module and the voltage generation circuitry. This does not include current dissipated by the LCD panel.

15.2 DC Characteristics: PIC16C925/926 (Commercial, Industrial) PIC16LC925/926 (Commercial, Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial Operating voltage VDD range as described in DC spec				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Input Low Voltage					
	VIL	I/O ports					
D030		with TTL buffer	VSS	—	0.15VDD	V	For entire VDD range
			Vss	—	0.8V	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger buffer	VSS	—	0.2VDD	V	
D032		MCLR, OSC1 (in RC mode)	VSS	—	0.2VDD	V	
D033		OSC1 (in XT, HS and LP)	VSS	—	0.3VDD	V	**(Note 1)**
		Input High Voltage					
	VIH	I/O ports		—			
D040		with TTL buffer	2.0	—	VDD	V	4.5V ≤ VDD ≤ 5.5V
D040A			0.25VDD + 0.8V	—	VDD	V	For entire VDD range
D041		with Schmitt Trigger buffer	0.8VDD	—	VDD	V	
D042		MCLR	0.8VDD	—	VDD	V	
D042A		OSC1 (XT, HS and LP)	0.7VDD	—	VDD	V	**(Note 1)**
D043		OSC1 (in RC mode)	0.9VDD	—	VDD	V	
D070	IPURB	**PORTB Weak Pull-up Current**	50	250	400	µA	VDD = 5V, VPIN = VSS
		Input Leakage Current (Notes 2, 3)					
D060	IIL	I/O ports	—	—	±1.0	µA	Vss ≤ VPIN ≤ VDD, Pin at hi-Z
D061		MCLR, RA4/T0CKI	—	—	±5	µA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	—	±5	µA	Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
		Output Low Voltage					
D080	VOL	I/O ports	—	—	0.6	V	IOL = 4.0 mA, VDD = 4.5V
D083		OSC2/CLKOUT (RC osc mode)	—	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V
		Output High Voltage					
D090	VOH	I/O ports **(Note 3)**	VDD - 0.7	—	—	V	IOH = -3.0 mA, VDD = 4.5V
D092		OSC2/CLKOUT (RC osc mode)	VDD - 0.7	—	—	V	IOH = -1.3 mA, VDD = 4.5V
		Capacitive Loading Specs on Output Pins					
D100	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)	—	—	50	pF	
D102	CB	SCL, SDA in I2C mode	—	—	400	pF	
D150	VDD	**Open Drain High Voltage**	—	—	8.5	V	RA4 pin

† Data in "Typ" column is at 5 V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C925/926 be driven with external clock in RC mode.

2: The leakage current on the 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.

FIGURE 15-3: LCD VOLTAGE WAVEFORM

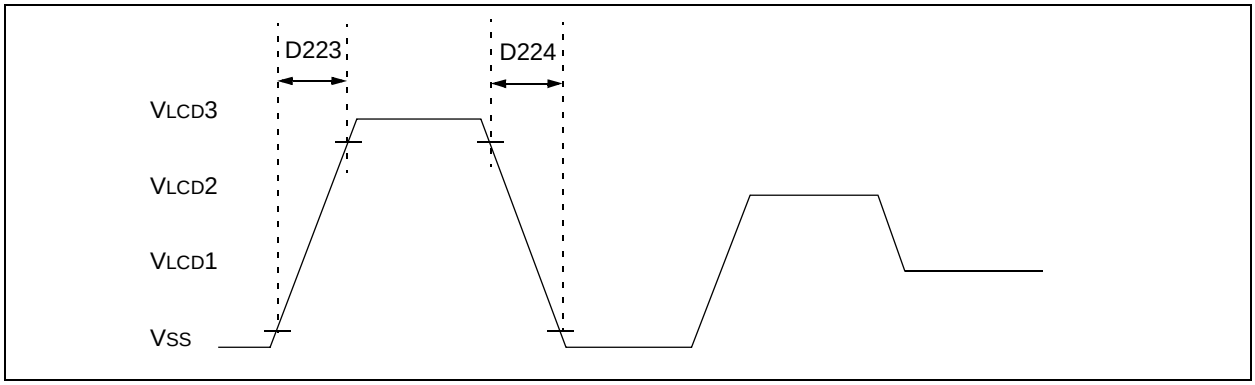


TABLE 15-1: LCD MODULE ELECTRICAL SPECIFICATIONS

Parameter No.	Symbol	Characteristic	Min	Typ†	Max	Units	Conditions
D200	VLCD3	LCD Voltage on pin VLCD3	VDD - 0.3	—	Vss + 7.0	V	
D201	VLCD2	LCD Voltage on pin VLCD2	Vss - 0.3	—	VLCD3	V	
D202	VLCD1	LCD Voltage on pin VLCD1	Vss - 0.3	—	VLCD3	V	
D220	VOH	Output High Voltage	Max VLCDN - 0.1	—	Max VLCDN	V	COM outputs IOH = 25 µA SEG outputs IOH = 3 µA
D221	VOL	Output Low Voltage	Min VLCDN	—	Min VLCDN + 0.1	V	COM outputs IOL = 25 µA SEG outputs IOL = 3 µA
D222	FLCDRC	LCDRC Oscillator Frequency	5	14	22	kHz	VDD = 5V, -40°C to +85°C
D223	TrLCD	Output Rise Time	—	—	200	µs	COM outputs Cload = 5,000 pF SEG outputs Cload = 500 pF VDD = 5.0V, T = 25°C
D224	TfLCD	Output Fall Time(1)	TrLCD - 0.05 TrLCD	—	TrLCD + 0.05 TrLCD	µs	COM outputs Cload = 5,000 pF SEG outputs Cload = 500 pF VDD = 5.0V, T = 25°C

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: 0 ohm source impedance at VLCD.

TABLE 15-2: VLCD CHARGE PUMP ELECTRICAL SPECIFICATIONS

Parameter No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
D250	IVADJ	VLCDADJ Regulated Current Output		—	10	—	µA	
D252	Δ IVADJ/Δ VDD	VLCDADJ Current VDD Rejection		—	—	0.1	µA/V	
D265	VVADJ	VLCDADJ Voltage Limits	PIC16**C**925/926	1.0	—	2.3	V	
			PIC16**LC**925/926	1.0		VDD - 0.7V	V	VDD < 3V

Note 1: For design guidance only.

15.3 Timing Parameter Symbology

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I^2C specifications only)
4. Ts (I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	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	$\overline{WR}$

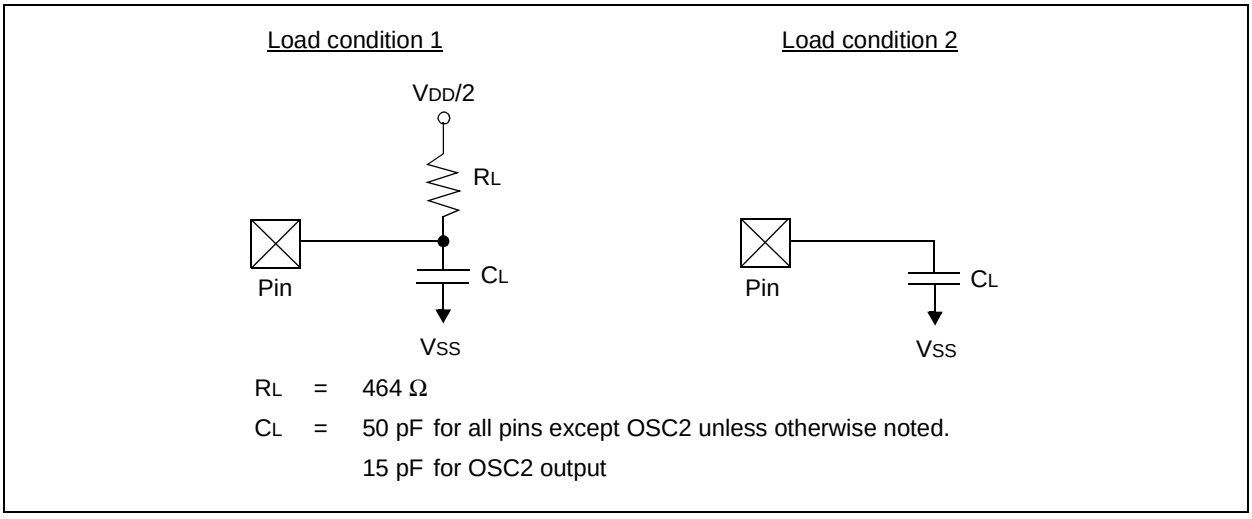
Uppercase letters and their meanings:

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		

FIGURE 15-4: LOAD CONDITIONS



R_L = 464 Ω

C_L = 50 pF for all pins except OSC2 unless otherwise noted.
15 pF for OSC2 output

15.4 Timing Diagrams and Specifications

FIGURE 15-5: EXTERNAL CLOCK TIMING

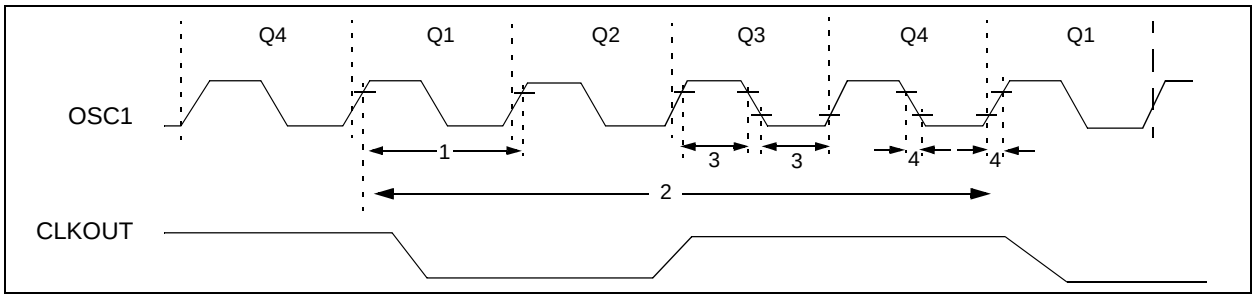


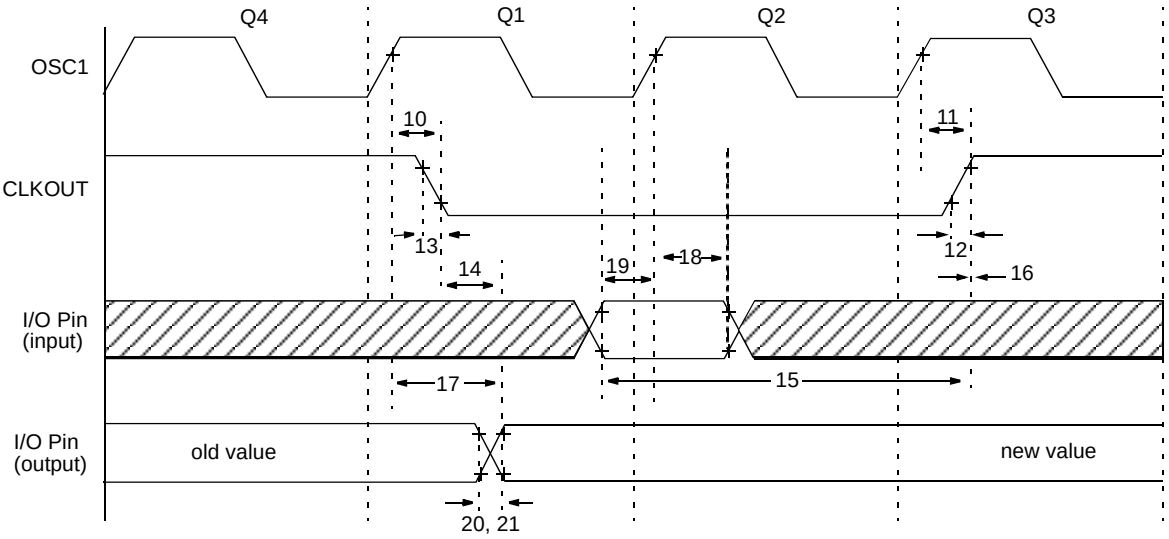
TABLE 15-3: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	External CLKIN Frequency **(Note 1)**	DC	—	4	MHz	XT and RC osc mode
			DC	—	20	MHz	HS osc mode
			DC	—	200	kHz	LP osc mode
		Oscillator Frequency **(Note 1)**	DC	—	4	MHz	RC osc mode
			0.1	—	4	MHz	XT osc mode
			4	—	20	MHz	HS osc mode
			5	—	200	kHz	LP osc mode
1	TOSC	External CLKIN Period **(Note 1)**	250	—	—	ns	XT and RC osc mode
			125	—	—	ns	HS osc mode
			5	—	—	µs	LP osc mode
		Oscillator Period **(Note 1)**	250	—	—	ns	RC osc mode
			250	—	10,000	ns	XT osc mode
			125	—	250	ns	HS osc mode
			5	—	—	µs	LP osc mode
2	TCY	Instruction Cycle Time **(Note 1)**	500	—	DC	ns	TCY = 4/FOSC
3	TosL, TosH	External Clock in (OSC1) High or Low Time	50	—	—	ns	XT oscillator
			2.5	—	—	µs	LP oscillator
			10	—	—	ns	HS oscillator
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	25	ns	XT oscillator
			—	—	50	ns	LP oscillator
			—	—	15	ns	HS oscillator

† Data in "Typ" column is at 5 V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time-base period. 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/CLKIN pin.
When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

FIGURE 15-6: CLKOUT AND I/O TIMING



Note: Refer to Figure 15-4 for load conditions.

TABLE 15-4: CLKOUT AND I/O TIMING REQUIREMENTS

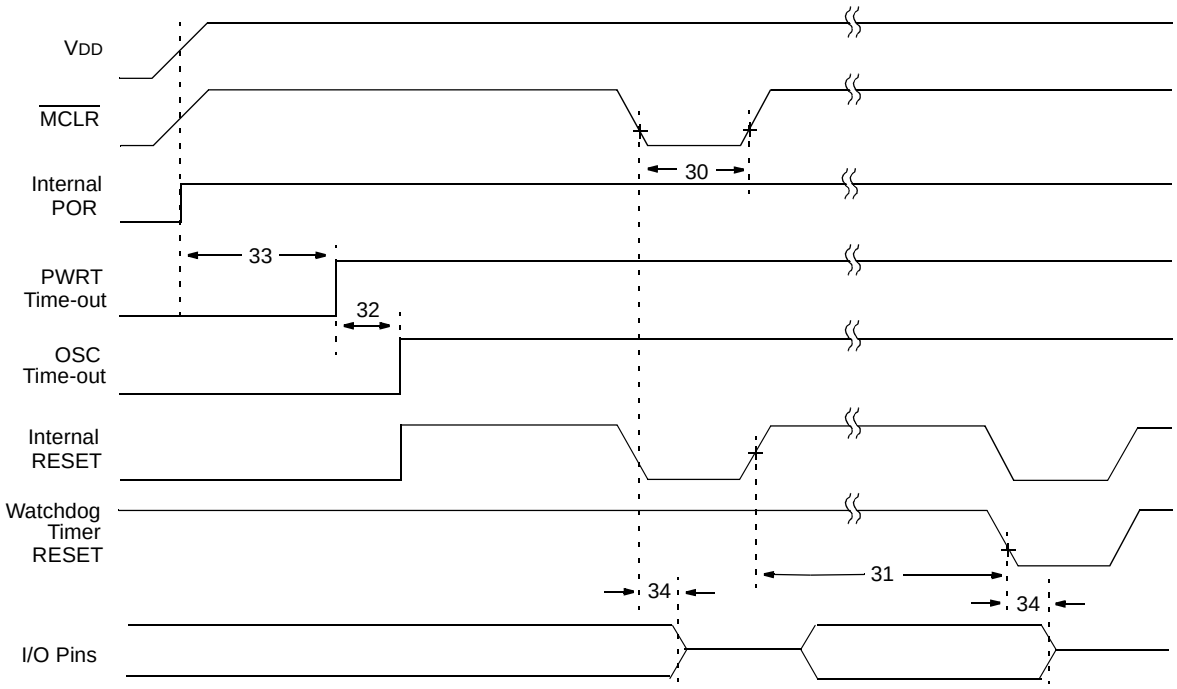
Parameter No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
10	TosH2ckL	OSC1↑ to CLKOUT↓		—	75	200	ns	**(Note 1)**
11	TosH2ckH	OSC1↑ to CLKOUT↑		—	75	200	ns	**(Note 1)**
12	TckR	CLKOUT rise time		—	35	100	ns	**(Note 1)**
13	TckF	CLKOUT fall time		—	35	100	ns	**(Note 1)**
14	TckL2ioV	CLKOUT ↓ to Port out valid		—	—	0.5TCY + 20	ns	**(Note 1)**
15	TioV2ckH	Port in valid before CLKOUT ↑		Tosc + 200	—	—	ns	**(Note 1)**
16	TckH2ioI	Port in hold after CLKOUT ↑		0	—	—	ns	**(Note 1)**
17	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid		—	50	150	ns	
18	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	PIC16**C**925/926	100	—	—	ns	
			PIC16**LC**925/926	200	—	—	ns	
19	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)		0	—	—	ns	
20	TioR	Port output rise time	PIC16**C**925/926	—	10	40	ns	
			PIC16**LC**925/926	—	—	80	ns	
21	TioF	Port output fall time	PIC16**C**925/926	—	10	40	ns	
			PIC16**LC**925/926	—	—	80	ns	
22††	Tinp	INT pin high or low time		TCY	—	—	ns	
23††	Trbp	RB7:RB4 change INT high or low time		TCY	—	—	ns	

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

†† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in RC mode where CLKOUT output is 4 x TOSC.

FIGURE 15-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



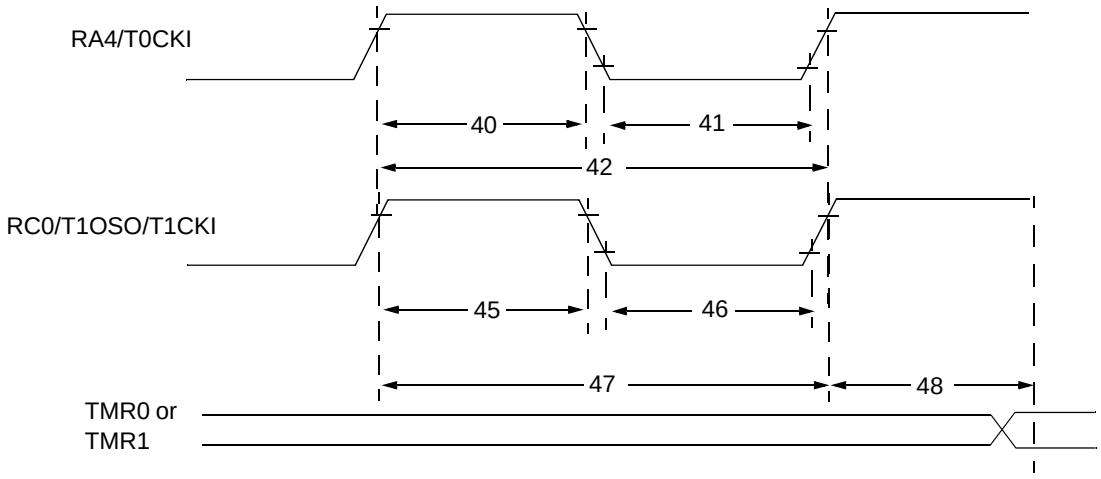
Note: Refer to Figure 15-4 for load conditions.

TABLE 15-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter 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 Prescaler)	7	18	33	ms	VDD = 5V, -40°C to +85°C
32	TOST	Oscillation Start-up Timer Period	—	1024TOSC	—	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	28	72	132	ms	VDD = 5V, -40°C to +85°C
34	TIOZ	I/O Hi-impedance from MCLR Low or Watchdog Timer Reset	—	—	2.1	µs	

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 15-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



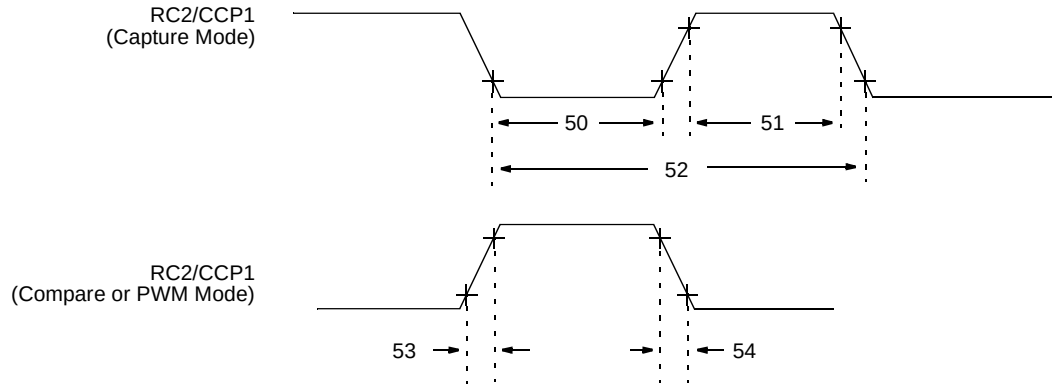
Note: Refer to Figure 15-4 for load conditions.

TABLE 15-6: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Symbol	Characteristic			Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width		No Prescaler	0.5TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width		No Prescaler	0.5TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
42	Tt0P	T0CKI Period		No Prescaler	TCY + 40	—	—	ns	
				With Prescaler	Greater of: 20 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (2, 4,..., 256)
45	Tt1H	T1CKI High Time	Synchronous, Prescaler = 1		0.5TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2,4,8	PIC16**C**925/926	15	—	—	ns	
				PIC16**LC**925/926	25	—	—	ns	
			Asynchronous	PIC16**C**925/926	30	—	—	ns	
				PIC16**LC**925/926	50	—	—	ns	
46	Tt1L	T1CKI Low Time	Synchronous, Prescaler = 1		0.5TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2,4,8	PIC16**C**925/926	15	—	—	ns	
				PIC16**LC**925/926	25	—	—	ns	
			Asynchronous	PIC16**C**925/926	30	—	—	ns	
				PIC16**LC**925/926	50	—	—	ns	
47	Tt1P	T1CKI Input Period	Synchronous	PIC16**C**925/926	Greater of: 30 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16**LC**925/926	Greater of: 50 or $\frac{T_{CY} + 40}{N}$				N = prescale value (1, 2, 4, 8)
			Asynchronous	PIC16**C**925/926	60	—	—	ns	
				PIC16**LC**925/926	100	—	—	ns	
	Ft1	Timer1 oscillator input frequency range (oscillator enabled by setting bit T1OSCEN)			DC	—	200	kHz	
48	TCKEZtmr1	Delay from external clock edge to timer increment			2Tosc	—	7Tosc	—	

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 15-9: CAPTURE/COMPARE/PWM TIMINGS



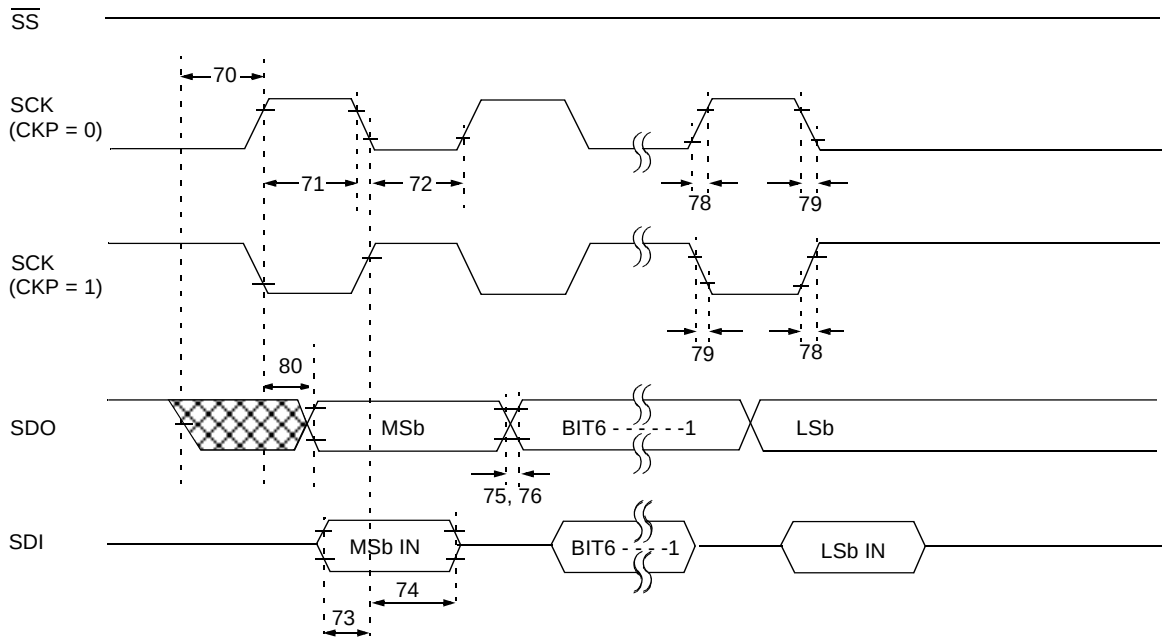
Note: Refer to Figure 15-4 for load conditions.

TABLE 15-7: CAPTURE/COMPARE/PWM REQUIREMENTS

Parameter No.	Symbol	Characteristic			Min	Typ†	Max	Units	Conditions
50	TccL	Input Low Time	No Prescaler		0.5TCY + 20	—	—	ns	
			With Prescaler	PIC16**C**925/926	10	—	—	ns	
				PIC16**LC**925/926	20	—	—	ns	
51	TccH	Input High Time	No Prescaler		0.5TCY + 20	—	—	ns	
			With Prescaler	PIC16**C**925/926	10	—	—	ns	
				PIC16**LC**925/926	20	—	—	ns	
52	TccP	Input Period			3TCY + 40 N	—	—	ns	N = prescale value (1,4 or 16)
53	TccR	Output Rise Time		PIC16**C**925/926	—	10	25	ns	
				PIC16**LC**925/926	—	25	45	ns	
54	TccF	Output Fall Time		PIC16**C**925/926	—	10	25	ns	
				PIC16**LC**925/926	—	25	45	ns	

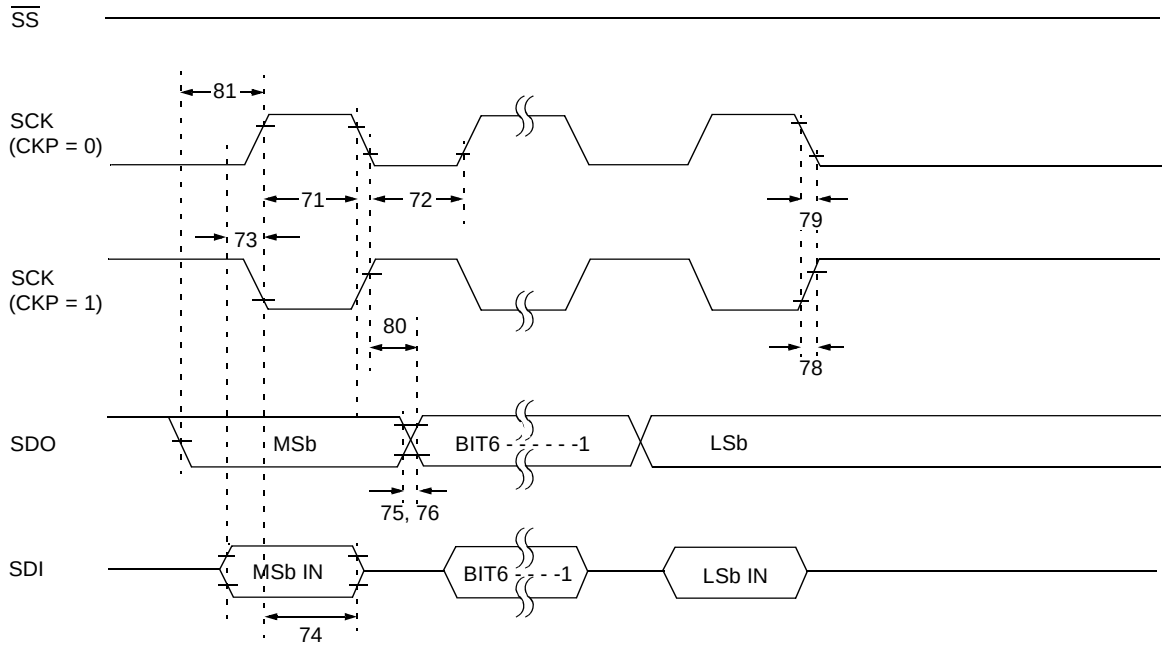
† Data in “Typ” column is at 5 V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 15-10: SPI MASTER MODE TIMING (CKE = 0)



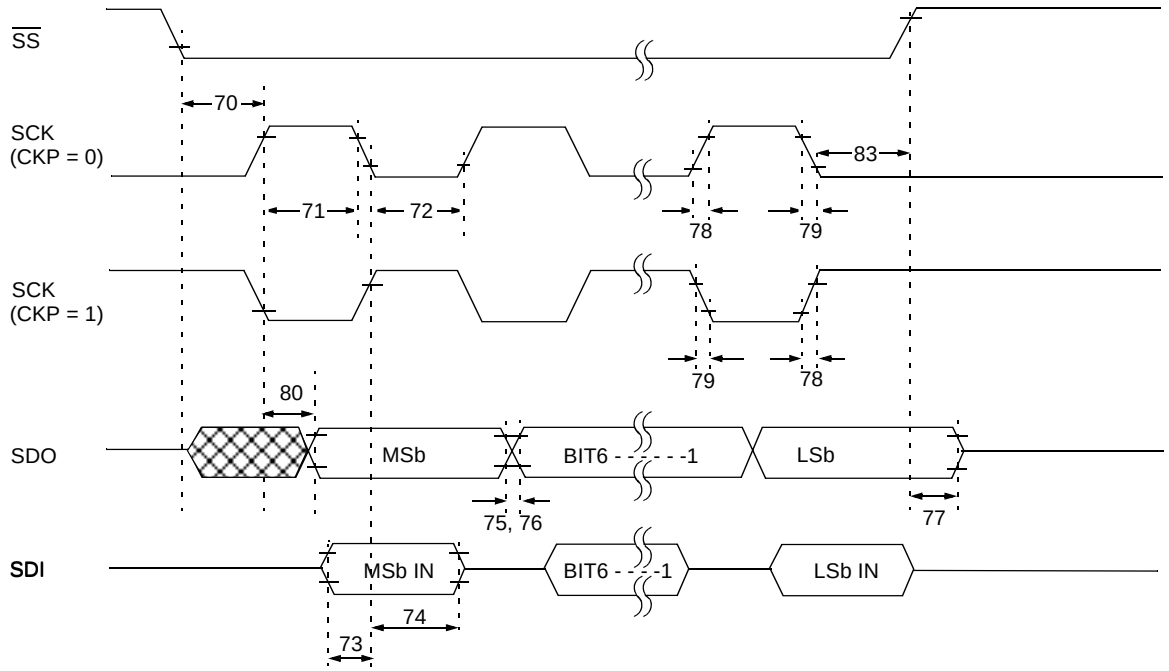
Note: Refer to Figure 15-4 for load conditions.

FIGURE 15-11: SPI MASTER MODE TIMING (CKE = 1)



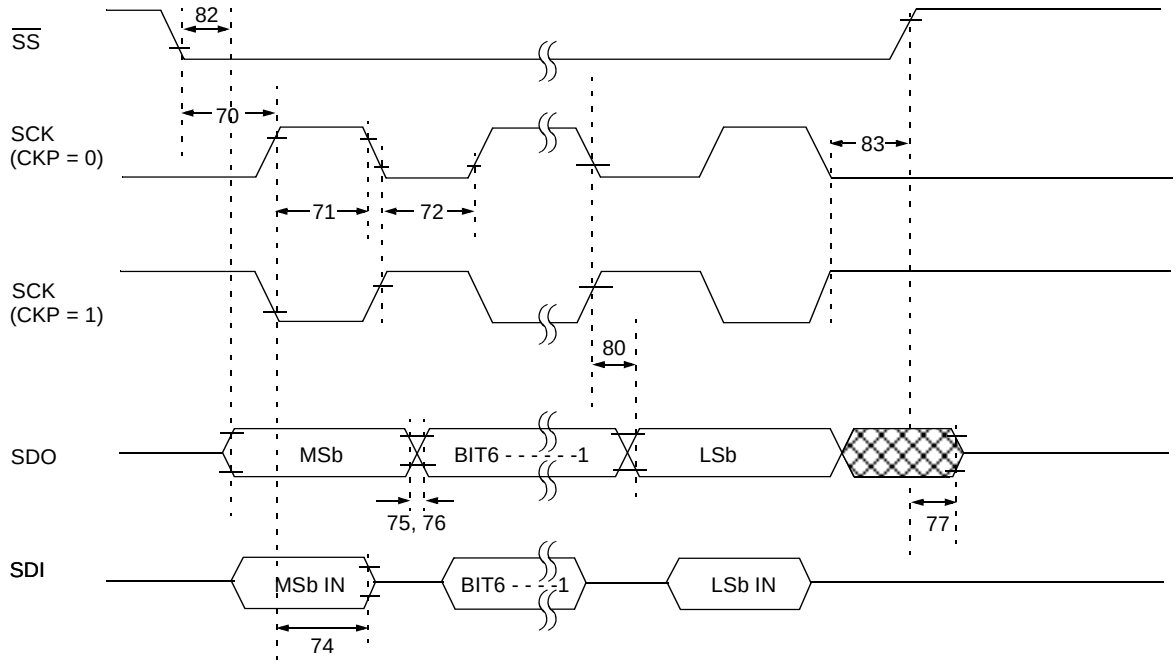
Note: Refer to Figure 15-4 for load conditions.

FIGURE 15-12: SPI SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 15-4 for load conditions.

FIGURE 15-13: SPI SLAVE MODE TIMING (CKE = 1)



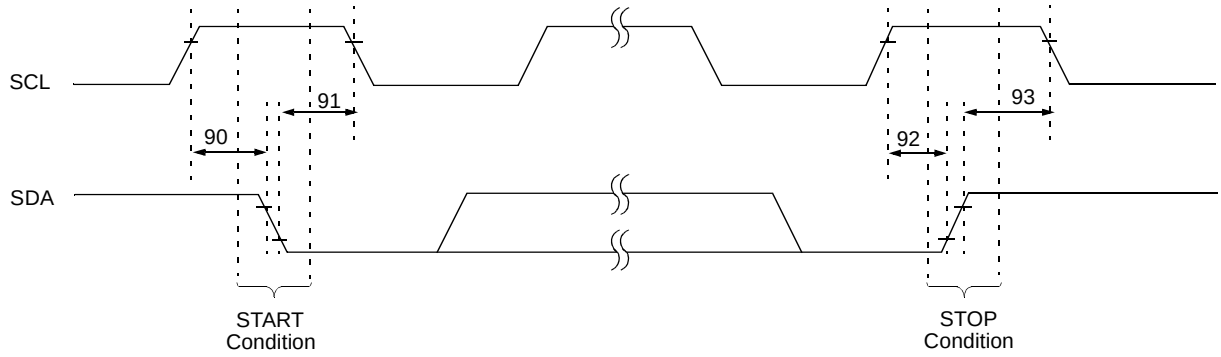
Note: Refer to Figure 15-4 for load conditions.

TABLE 15-8: SPI MODE REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}\downarrow$ to SCK↓ or SCK↑ input		TCY	—	—	ns	
71	TscH	SCK input high time (Slave mode)	Continuous	1.25TCY + 30	—	—	ns	
71A			Single Byte	40	—	—	ns	
72	TscL	SCK input low time (Slave mode)	Continuous	1.25TCY + 30	—	—	ns	
72A			Single Byte	40				
73	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		50	—	—	ns	
74	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		50	—	—	ns	
75	TdoR	SDO data output rise time		—	10	25	ns	
76	TdoF	SDO data output fall time		—	10	25	ns	
77	TssH2doZ	$\overline{SS}\uparrow$ to SDO output hi-impedance		10	—	50	ns	
78	TscR	SCK output rise time (Master mode)		—	10	25	ns	
79	TscF	SCK output fall time (Master mode)		—	10	25	ns	
80	TscH2doV, TscL2doV	SDO data output valid after SCK edge		—	—	50	ns	
81	TdoV2scH, TdoV2scL	SDO data output setup to SCK edge		TCY	—	—	ns	
82	TssL2doV	SDO data output valid after $\overline{SS}\downarrow$ edge		—	—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}\uparrow$ after SCK edge		1.5TCY + 40	—	—	ns	
84	Tb2b	Delay between consecutive bytes		1.5TCY + 40	—	—	ns	

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 15-14: I²C BUS START/STOP BITS TIMING

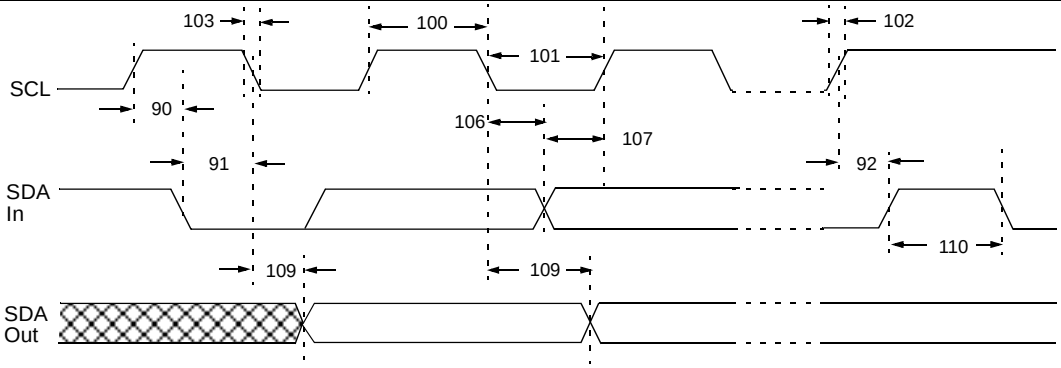


Note: Refer to Figure 15-4 for load conditions.

TABLE 15-9: I²C BUS START/STOP BITS REQUIREMENTS

Parameter No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
90	TSU:STA	START condition Setup time	100 kHz mode	4700	—	—	ns	Only relevant for Repeated START condition
91	THD:STA	START condition Hold time	100 kHz mode	4000	—	—	ns	After this period the first clock pulse is generated
92	TSU:STO	STOP condition Setup time	100 kHz mode	4700	—	—	ns	
93	THD:STO	STOP condition Hold time	100 kHz mode	4000	—	—	ns	

FIGURE 15-15: I²C BUS DATA TIMING



Note: Refer to Figure 15-4 for load conditions.

TABLE 15-10: I²C BUS DATA REQUIREMENTS

Parameter No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock high time	100 kHz mode	4.0	—	µs	Device must operate at a minimum of 1.5 MHz
			SSP Module	1.5TCY	—		
101	TLOW	Clock low time	100 kHz mode	4.7	—	µs	Device must operate at a minimum of 1.5 MHz
			SSP Module	1.5TCY	—		
102	TR	SDA and SCL rise time	100 kHz mode	—	1000	ns	
103	TF	SDA and SCL fall time	100 kHz mode	—	300	ns	
90	TSU:STA	START condition setup time	100 kHz mode	4.7	—	µs	Only relevant for Repeated START condition
91	THD:STA	START condition hold time	100 kHz mode	4.0	—	µs	After this period the first clock pulse is generated
106	THD:DAT	Data input hold time	100 kHz mode	0	—	ns	
107	TSU:DAT	Data input setup time	100 kHz mode	250	—	ns	
92	TSU:STO	STOP condition setup time	100 kHz mode	4.7	—	µs	
109	TAA	Output valid from clock	100 kHz mode	—	3500	ns	**(Note 1)**
110	TBUF	Bus free time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
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.

TABLE 15-11: A/D CONVERTER CHARACTERISTICS: PIC16C925/926 (COMMERCIAL, INDUSTRIAL) PIC16LC925/926 (COMMERCIAL, INDUSTRIAL)

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
A01	NR	Resolution		—	—	10-bits	bit	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A02	EABS	Total Absolute error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A03	EIL	Integral linearity error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A04	EDL	Differential linearity error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A05	EFS	Full scale error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A06	EOFF	Offset error		—	—	< ± 2	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A07	EGN	Gain error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A10	—	Monotonicity		—	guaranteed	—	—	VSS ≤ VAIN ≤ VREF
A20	VREF	Reference voltage		AVDD - 2.5V	—	AVDD + 0.3	V	
A25	VAIN	Analog input voltage		VSS - 0.3	—	VREF + 0.3	V	
A30	ZAIN	Recommended impedance of analog voltage source		—	—	10.0	kΩ	
A40	IAD	A/D conversion current (VDD)	PIC16**C**925/926	—	220	—	µA	Average current consumption when A/D is on. **(Note 1)**
			PIC16**LC**925/926	—	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.
				—	—	10	µA	During A/D Conversion cycle

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

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.

2: VREF current is from RA3 pin or VDD pin, whichever is selected as reference input.

FIGURE 15-16: A/D CONVERSION TIMING

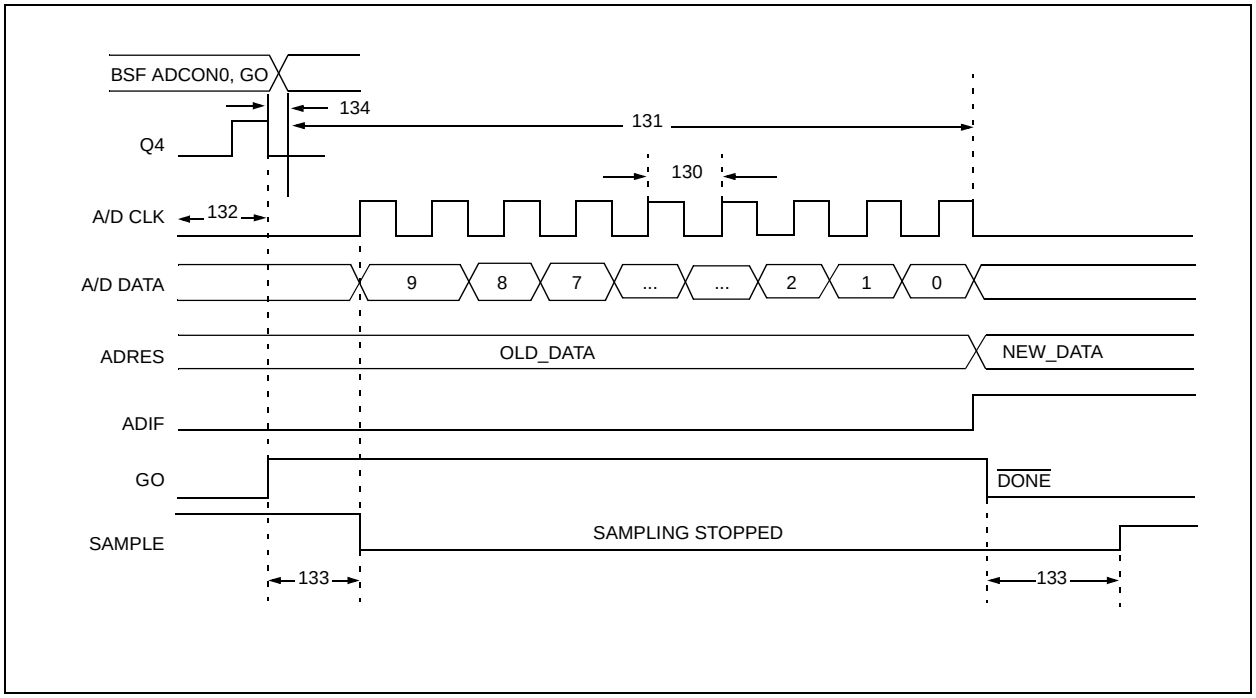


TABLE 15-12: A/D CONVERSION REQUIREMENTS

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
130	TAD	A/D clock period	PIC16**C**925/926	1.6	—	—	µs	TOSC based, VREF ≥ 3.0V
			PIC16**LC**925/926	3.0	—	—	µs	TOSC based, VREF ≥ 2.0V
			PIC16**C**925/926	2.0	4.0	6.0	µs	A/D RC Mode
			PIC16**LC**925/926	3.0	6.0	9.0	µs	A/D RC Mode
131	TCNV	Conversion time (not including S/H time) **(Note 1)**		—	—	12	TAD	
132	TACQ	Acquisition time		**(Note 2)**	40	—	µs	
				10	—	—	µs	The minimum time is the amplifier settling time. 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).
134	TGO	Q4 to A/D clock start		—	TOSC/2	—	—	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.
135	TSWC	Switching from convert → sample time		1.5	—	—	TAD	

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: ADRES register may be read on the following TCY cycle.

2: See Section 10.1 for min. conditions.

16.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

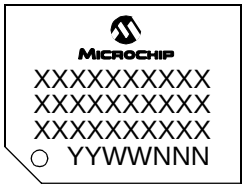
Graphs and Tables are not available at this time.

NOTES:

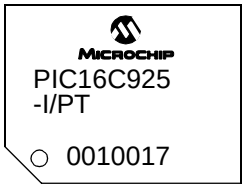
17.0 PACKAGING INFORMATION

17.1 Package Marking Information

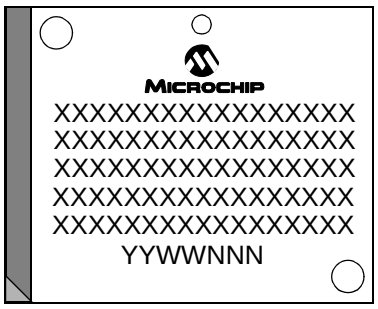
64-Lead TQFP



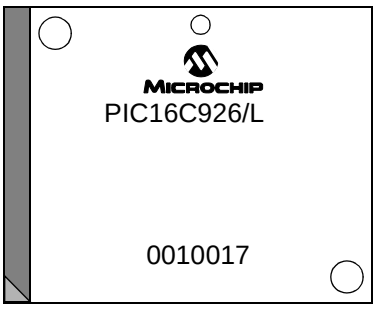
Example



68-Lead PLCC



Example



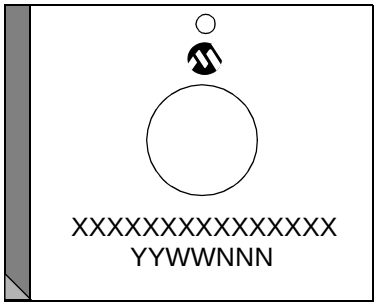
Legend:	XX...X	Customer specific information*
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code

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.

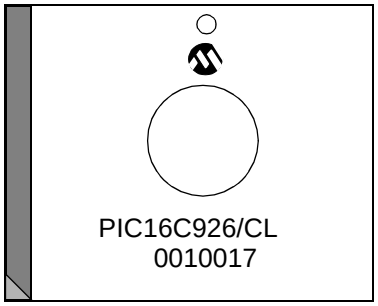
***** Standard OTP marking consists of Microchip part number, year code, week code and traceability code. For OTP marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

Package Marking Information (Continued)

68-Lead CERQUAD Windowed

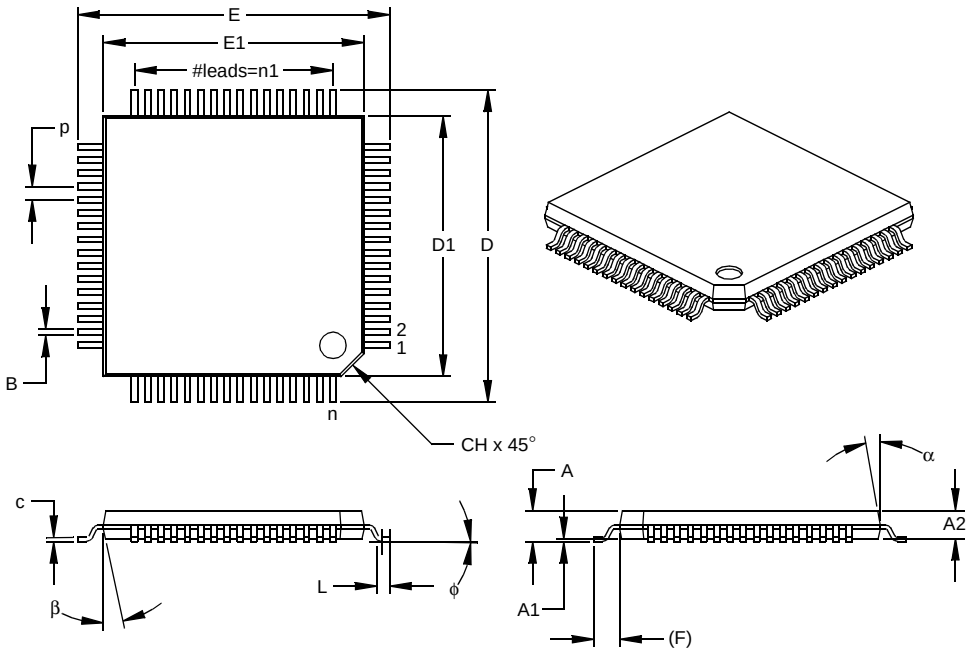


Example



17.2 Package Details

64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)

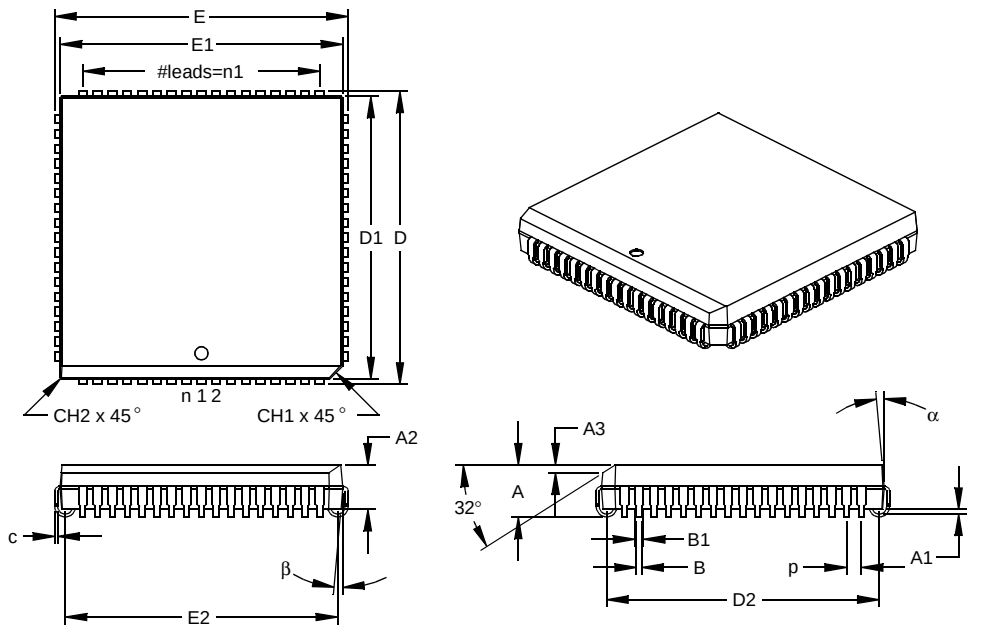


Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		64			64	
Pitch	p		.020			0.50	
Pins per Side	n1		16			16	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002	.006	.010	0.05	0.15	0.25
Foot 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

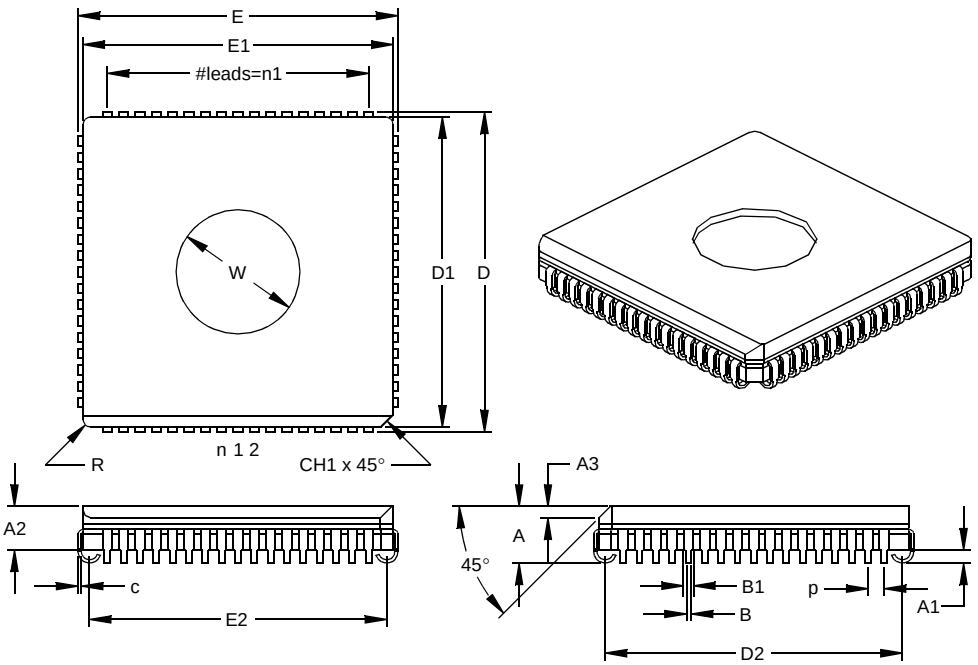
68-Lead Plastic Leaded Chip Carrier (L) – Square (PLCC)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		68			68	
Pitch	p		.050			1.27	
Pins per Side	n1		17			17	
Overall Height	A	.165	.173	.180	4.19	4.39	4.57
Molded Package Thickness	A2	.145	.153	.160	3.68	3.87	4.06
Standoff §	A1	.020	.028	.035	0.51	0.71	0.89
Side 1 Chamfer Height	A3	.024	.029	.034	0.61	0.74	0.86
Corner Chamfer 1	CH1	.040	.045	.050	1.02	1.14	1.27
Corner Chamfer (others)	CH2	.000	.005	.010	0.00	0.13	0.25
Overall Width	E	.985	.990	.995	25.02	25.15	25.27
Overall Length	D	.985	.990	.995	25.02	25.15	25.27
Molded Package Width	E1	.950	.954	.958	24.13	24.23	24.33
Molded Package Length	D1	.950	.954	.958	24.13	24.23	24.33
Footprint Width	E2	.890	.920	.930	22.61	23.37	23.62
Footprint Length	D2	.890	.920	.930	22.61	23.37	23.62
Lead Thickness	c	.008	.011	.013	0.20	0.27	0.33
Upper Lead Width	B1	.026	.029	.032	0.66	0.74	0.81
Lower Lead Width	B	.013	.020	.021	0.33	0.51	0.53
Mold Draft Angle Top	α	0	5	10	0	5	10
Mold Draft Angle Bottom	β	0	5	10	0	5	10

* Controlling Parameter
§ Significant Characteristic
Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MO-047
Drawing No. C04-049

68-Lead Ceramic Leaded (CL) Chip Carrier with Window – Square (CERQUAD)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		68			68	
Pitch	p		.050			1.27	
Overall Height	A	.165	.175	.185	4.19	4.45	4.70
Package Thickness	A2	.118	.137	.155	3.00	3.48	3.94
Standoff §	A1	.030	.040	.050	0.76	1.02	1.27
Side One Chamfer Dim.	A3	.030	.035	.040	0.76	0.89	1.02
Corner Chamfer (1)	CH1	.030	.040	.050	0.76	1.02	1.27
Corner Radius (Others)	R	.020	.025	.030	0.51	0.64	0.76
Overall Package Width	E	.983	.988	.993	24.97	25.10	25.22
Overall Package Length	D	.983	.988	.993	24.97	25.10	25.22
Ceramic Package Width	E1	.942	.950	.958	23.93	24.13	24.33
Ceramic Package Length	D1	.942	.950	.958	23.93	24.13	24.33
Footprint Width	E2	.890	.910	.930	22.61	23.11	23.62
Footprint Length	D2	.890	.910	.930	22.61	23.11	23.62
Pins each side	n1		17			17	
Lead Thickness	c	.008	.010	.012	0.20	0.25	0.30
Upper Lead Width	B1	.026	.029	.031	0.66	0.72	0.79
Lower Lead Width	B	.015	.018	.021	0.38	0.46	0.53
Window Diameter	W	.370	.380	.390	9.40	9.65	9.91

* Controlling Parameter
§ Significant Characteristic
JEDEC Equivalent: MO-087
Drawing No. C04-097

NOTES:

APPENDIX A: REVISION HISTORY

Version	Date	Description
A	February 2001	This is a new data sheet. However, these devices are similar to those described in the PIC16C923/924 data sheet (DS30444).

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are listed in Table B-1.

TABLE B-1: DEVICE DIFFERENCES

Feature	PIC16C925	PIC16C926
EPROM Program Memory (words)	4K	8K
Data Memory (bytes)	176	336

Note: On 64-pin TQFP, pins RG7 and RE7 are not available.

APPENDIX C: CONVERSION CONSIDERATIONS

Considerations for converting to the devices listed in this data sheet from previous device types are summarized in Table C-1.

TABLE C-1: CONVERSION CONSIDERATIONS

Feature	PIC16C923/924	PIC16C925/926
Operating Frequency	DC - 8 MHz	DC - 20 MHz
EPROM Program Memory (words)	4K	4K (925) 8K (926)
Data Memory (bytes)	176	176 (925) 336 (926)
A/D Converter Resolution	8-bit (924 only)	10-bit
A/D Converter Channels	none (923) 5 (924)	5
Interrupt Sources	8 (923) 9 (924)	9
Brown-out Reset	No	Yes

INDEX

A

A/D 75
ADCON0 Register 75
ADCON1 Register 76
ADIF bit 76
Block Diagrams
Analog Input Model 78
Converter 77
Configuring Analog Port Pins 80
Configuring the Interrupt 77
Configuring the Module 77
Conversion Clock 79
Conversions 80
Converter Characteristics 157
Delays 78
Effects of a RESET 81
GO/$\overline{\text{DONE}}$ bit 76
Internal Sampling Switch (Rss) Impedence 78
Operation During SLEEP 81
Register Initialization States 104, 105
Sampling Requirements 78
Source Impedence 78
Time Delays 78
Absolute Maximum Ratings 139
$\overline{\text{ACK}}$ Pulse 66
$\overline{\text{ACK}}$ pulse 70, 71, 72
Analog-to-Digital Converter. *See* A/D
Appendic C
Conversion Considerations 168
Appendix A
Revision History 167
Appendix B
Device Differences 167
Application Notes
AN552 31
AN556 25
AN578 59
AN594 53
AN607 102
Assembler
MPASM Assembler 133
Associated Registers 81

B

BF bit 70
Block Diagrams
A/D Converter 77
Analog Input Model 78
Capture Mode 54
Compare Mode 55
External Parallel Cystal Oscillator 100
External Series Crystal Oscillator 100
Interrupt Logic 107
LCD Charge Pump 95
LCD Module 84
LCD Resistor Ladder 95
On-Chip Reset Circuit 101
PIC16C925/926 Architecture 6
PORTA
RA3:RA0 and RA5 Port Pins 29
RA4/T0CKI Pin 29
PORTB
RB3:RB0 Port Pins 31
RB7:RB4 Port Pins 31
PORTC 33
PORTD
Pins <4:0> 34
Pins <7:5> 34
PORTE 36
PORTF 37
PORTG 38
PWM Mode 56
RC Oscillator 100
SSP
I^2C Mode 69
SPI Mode 61
Timer0 41
Timer0/WDT Prescaler 44
Timer1 48
Timer2 51
Watchdog Timer 110
BOR. *See* Brown-out Reset.
Brown-out Reset (BOR) 97, 102, 103
BOR Status ($\overline{\text{BOR}}$ Bit) 24

C

C (Carry) bit 19
Capture Mode (CCP)
Associated Registers 58
Block Diagram 54
Changing Between Prescalers 54
Pin Configuration 54
Prescaler 54
Software Interrupt 54
Capture/Compare/PWM (CCP)
CCP1CON Register 53
CCPR1 Register 53
CCPR1H Register 53
CCPR1L Register 53
Register Initialization States 104
Timer Resources 53
CCP. *See* Capture/Compare/PWM (CCP).
Charge Pump (LCD) 95
CKP (Clock Polarity Select) bit 60
Clocking Scheme 9
Code Examples
Call of a Subroutine in Page 1 from Page 0 25
Changing Between Capture Prescalers 54
Changing Prescaler (Timer0 to WDT) 45
Changing Prescaler (WDT to Timer0) 45
I/O Programming 39
I^2C Module Operation 73
Indirect Addressing 26
Initializing PORTA 29
Initializing PORTB 31
Initializing PORTC 33
Initializing PORTD 34
Initializing PORTE 36
Initializing PORTF 37
Initializing PORTG 38
Loading the SSPBUF Register 61
Program Read 28
Reading a 16-bit Free-running Timer 49
Saving STATUS, W and PCLATH Registers in RAM 109
Segment Enable
One-Third-Duty with 13 Segments 94
Static MUX with 32 Segments 94

Code Protection 97, 112
Compare Mode (CCP)
Associated Registers 58
Block Diagram 55
Pin Configuration 55
Software Interrupt Mode 55
Special Event Trigger 55
Timer1 Mode 55
Computed GOTO 25
Configuration Bits 97
Configuration Word 98

D

DC and AC Characteristics Graphs and Tables 159
DC bit 19
Development Support 133
Device DC Characteristics 141–145
LC Devices 144
Direct Addressing 26

E

Errata 4

F

FSR Register 26
Initialization States 104

G

GIE bit 107

I

I/O Programming Considerations 39
Read-Modify-Write Example 39
I^2C
Addressing I^2C Devices 66
Arbitration 68
BF 70, 71
CKP 71
Clock Synchronization 68
Combined Format 67
Initiating and Terminating Data Transfer 65
Master-Receiver Sequence 67
Master-Transmitter Sequence 67
Multi-Master 68
Overview 65
START 65
STOP 65, 66
Transfer Acknowledge 66
ICEPIC In-Circuit Emulator 134
IDLE_MODE 73
In-Circuit Serial Programming 97, 112
INDF Register 26
Initialization States 104
Indirect Addressing 26
Instruction Cycle 9
Instruction Flow/Pipelining 9
Instruction Format 113
Instruction Set
ADDLW 115
ADDWF 115
ANDLW 116
ANDWF 116
BCF 117
BSF 117
BTFSC 117
BTFSS 118
CALL 118
CLRF 119
CLRW 119
CLRWDT 120
COMF 120
DECF 121
DECFSZ 121
GOTO 122
INCF 122
INCFSZ 123
IORLW 123
IORWF 124
MOVF 124
MOVLW 124
MOVWF 125
NOP 125
OPTION 125
RETFIE 126
RETLW 126
RETURN 127
RLF 127
RRF 128
SLEEP 128
SUBLW 129
SUBWF 129
SWAPF 130
TRIS 130
XORLW 131
XORWF 131
Instruction Set Summary 113–131
INT Interrupt 108
INTCON Register 21, 107
Initialization States 104
Inter-Integrated Circuit (I^2C). *See* I^2C.
Internal Sampling Switch (Rss) Impedence 78
Interrupt Flag 107
Interrupts 97, 107
RB7:RB4 Port Change 31
IRP bit 19

K

KEELOQ Evaluation and Programming Tools 136

L

LCD Module
Associated Registers 96
Block Diagram 84
Charge Pump 95
Block Diagram 95
Electrical Specifications 145
External R-Ladder 95
Block Diagram 95
Generic LCDD Register 92
LCDCON Register 83
LCDPS Register 84
LCDSE Register 94
Register Initialization States 105
Voltage Generation 95
Loading PC Register (Diagram) 25

M

Master Clear (MCLR) 101
    MCLR Initialization Condition for Registers 104
    MCLR Reset, Normal Operation 103
    MCLR Reset, SLEEP 103
MCLR. *See* Master Clear.
Memory
    Data Memory 12
    Maps, PIC16C9XX 11
    Program Memory 11
MPLAB C17 and MPLAB C18 C Compilers 133
MPLAB ICD In-Circuit Debugger 135
MPLAB ICE High Performance Universal
    In-Circuit Emulator with MPLAB IDE 134
MPLAB Integrated Development
    Environment Software 133
MPLINK Object Linker/MPLIB Object Librarian 134

O

OPCODE 113
OPTION_REG Register 20
    Initialization States 104
    INTEDG Bit 20
    PS2:PS0 Bits 20
    PSA Bit 20
    T0CS Bit 20
    T0SE Bit 20
OSC selection 97
Oscillator
    HS 99, 102
    LP 99, 102
Oscillator Configurations 99

P

Package Details 163
Package Marking Information 161
Packaging Information 161
Paging, Program Memory 25
PCL Register 25
    Initialization States 104
PCLATH Register 25
    Initialization States 104
PCON Register 24
    BOR Bit 24
    Initialization States 104
    POR Bit 24
PD bit 19, 101
PICDEM 1 Low Cost PICmicro
    Demonstration Board 135
PICDEM 17 Demonstration Board 136
PICDEM 2 Low Cost PIC16CXX
    Demonstration Board 135
PICDEM 3 Low Cost PIC16CXXX
    Demonstration Board 136
PICSTART Plus Entry Level
    Development Programmer 135
PIE1 Register 22, 107
    Initialization States 104
Pin Functions
    MCLR/VPP 7
    OSC1/CLKIN 7
    OSC2/CLKOUT 7
    RA0/AN0 7
    RA1/AN1 7
    RA2/AN2 7
    RA3/AN3/VREF 7
    RA4/T0CKI 7
    RA5/AN4/SS 7
    RB0/INT 7
    RB1 7
    RB2 7
    RB3 7
    RB4 7
    RB5 7
    RB6 7
    RB7 7
    RC0/T1OSO/T1CKI 7
    RC1/T1OSI 7
    RC2/CCP1 7
    RC3/SCK/SCL 7
    RC4/SDI/SDA 7
    RC5/SDO 7
    RD0/SEG00 8
    RD1/SEG01 8
    RD2/SEG02 8
    RD3/SEG03 8
    RD4/SEG04 8
    RD5/SEG29/COM3 8
    RD6/SEG30/COM2 8
    RD7/SEG31/COM1 8
    RE0/SEG05 8
    RE1/SEG06 8
    RE2/SEG07 8
    RE3/SEG08 8
    RE4/SEG09 8
    RE5/SEG10 8
    RE6/SEG11 8
    RE7/SEG27 8
    RF0/SEG12 8
    RF1/SEG13 8
    RF2/SEG14 8
    RF3/SEG15 8
    RF4/SEG16 8
    RF5/SEG17 8
    RF6/SEG18 8
    RF7/SEG19 8
    RG0/SEG20 8
    RG1/SEG21 8
    RG2/SEG22 8
    RG3/SEG23 8
    RG4/SEG24 8
    RG5/SEG25 8
    RG6/SEG26 8
    RG7/SEG28 8
    VDD 8
    VSS 8
PIR1 Register 23, 107
    Initialization States 104
POP 25

POR 102
Oscillator Start-up Timer (OST) 97, 102
POR Status (POR Bit) 24
Power Control Register (PCON) 102
Power-on Reset (POR) 97, 102, 104
Power-up Timer (PWRT) 97, 102
RESET Condition for Special Registers 103
Time-out Sequence 102
Time-out Sequence on Power-up 106
TO 101
Port RB Interrupt 108
PORTA
Associated Registers 30
Initialization 29
Initialization States 104
Pin Functions 30
RA3:RA0 and RA5 Port Pins 29
RA4/T0CKI Pin 29
Register 29
TRISA Register 29
PORTB
Associated Registers 32
Initialization 31
Initialization States 104
Pin Functions 32
RB0/INT Edge Select (INTEDG Bit) 20
RB3:RB0 Port Pins 31
RB7:RB4 Port Pins 31
Register 31
TRISB Register 31
PORTC
Associated Registers 33
Block Diagram (Peripheral Output Override) 33
Initialization 33
Initialization States 104
Pin Functions 33
Register 33
TRISC Register 33
PORTD
Associated Registers 35
Initialization 34
Initialization States 104
Pin Functions 35
Pins <4:0> 34
Pins <7:5> 34
Register 34
TRISD Register 34
PORTE
Associated Registers 36
Block Diagram 36
Initialization 36
Initialization States 104
Pin Functions 36
Register 36
TRISE Register 36
PORTF
Associated Registers 37
Block Diagram 37
Initialization 37
Initialization States 105
Pin Functions 37
Register 37
TRISF Register 37
PORTG
Associated Registers 38
Block Diagram 38
Initialization 38
Initialization States 105
Pin Functions 38
Register 38
TRISG Register 38
Postscaler, WDT
Assignment (PSA Bit) 20
Rate Select (PS2:PS0 Bits) 20
Power-down Mode (SLEEP) 111
Power-on Reset. *See* POR.
PR2 105
Prescaler, Timer0
Assignment (PSA Bit) 20
Rate Select (PS2:PS0 Bits) 20
Switching Between Timer0 and WDT 45
PRO MATE II Universal Device Programmer 135
Product Identification System 177
Program Counter
RESET Conditions 103
Program Memory
Associated Registers 28
Operation During Code Protect 28
PMADR Register 27
PMCON1 Register 27
Program Read (Code Example) 28
Read 28
Program Memory and Stack Maps 11
PUSH 25
PWM Mode (CCP) 56
Associated Registers 58
Block Diagram 56
Example Frequencies/Resolutions 57
Example Period and Duty Cycle Calculations 57

R

R/W bit 66, 70, 71
RBIF bit 31, 108
RC Oscillator 99, 100, 102
RCV_MODE 73
Read-Modify-Write 39
Register File 12
Register File Map
PIC16C925 13
PIC16C926 14
Registers
ADCON0 (A/D Control 0) 75
ADCON1 (A/D Control 1) 76
CCP1CON (CCP Control) 53
Flag 23
Initialization Conditions 104–105
INTCON (Interrupt Control) 21
LCDCON (LCD Control) 83
LCDD (LCD Pixel Data, General Format) 92
LCDPS (LCD Prescale) 84
LCDSE (LCD Segment Enable) 94
OPTION_REG 20
PCON (Power Control) 24
PIE2 (Peripheral Interrupt Enable 1) 22
PIR1 (Peripheral Interrupt Request) 23
PMCON1 (Program Memory Control) 27
SSPCON (Sync Serial Port Control) 60
SSPSTAT (Sync Serial Port Status) 59
STATUS 19

T1CON (Timer1 Control) 47
T2CON (Timer2 Control) 52
RESET 97, 101
Block Diagram 101
RESET Conditions for PCON Register 103
RESET Conditions for Program Counter 103
RESET Conditions for STATUS Register 103
Resistor Ladder (LCD) 95
RP1:RP0 (Bank Select) bits 12, 19

S

SCL 70, 71, 72
SDA 71, 72
Slave Mode
SCL pin 70
SDA pin 70
SLEEP 97, 101
Software Simulator (MPLAB SIM) 134
Special Features of the CPU 97
Special Function Registers, Summary 15
SPI
Associated Registers 64
Master Mode 62
Serial Clock 61
Serial Data In 61
Serial Data Out 61
Serial Peripheral Interface (SPI) 59
Slave Select 61
SPI Clock 62
SPI Mode 61
SSP
Block Diagrams
I^2C Mode 69
SPI Mode 61
Register Initialization States 104, 105
SSPADD Register 69, 70
SSPBUF Register 62, 69, 70, 71
SSPCON Register 60, 69
SSPIF bit 70, 71, 72
SSPOV bit 70
SSPSR 62
SSPSR Register 70, 71
SSPSTAT 71
SSPSTAT Register 59, 69, 71
SSP I^2C
Addressing 70
Associated Registers 72
Multi-Master Mode 72
Reception 71
SSP I^2C Operation 69
START 71
START (S) 72
STOP (P) 72
Transmission 71
SSPEN (Sync Serial Port Enable) bit 60
SSPM3:SSPM0 60
SSPOV (Receive Overflow Indicator) bit 60
SSPOV bit 70
Stack 25
Overflows 25
Underflow 25
STATUS Register 19
Initialization States 104
Synchronous Serial Port Mode Select bits,
SSPM3:SSPM0 60

T

TAD 79
Timer0
Associated Registers 45
Block Diagram 41
Clock Source Edge Select (T0SE Bit) 20
Clock Source Select (T0CS Bit) 20
External Clock 43
Synchronization 43
Timing 43
Increment Delay 43
Initialization States 104
Interrupt 41
Interrupt Timing 42
Prescaler 44
Block Diagram 44
Timing 42
TMR0 Interrupt 108
Timer1
Associated Registers 50
Asynchronous Counter Mode 49
Block Diagram 48
Capacitor Selection 50
External Clock Input
Synchronized Counter Mode 48
Timing with Unsynchronized Clock 49
Unsynchronized Clock Timing 49
Oscillator 50
Prescaler 50
Reading a Free-running Timer 49
Register Initialization States 104
Resetting Register Pair 50
Resetting with a CCP Trigger Output 50
Switching Prescaler Assignment 45
Synchronized Counter Mode 48
T1CON Register 47
Timer Mode 48
Timer2
Block Diagram 51
Output 51
Register Initialization States 104
T2CON Register 52
Timing Diagrams (Operational)
Clock/Instruction Cycle 9
I^2C Clock Synchronization 68
I^2C Data Transfer Wait State 66
I^2C Multi-Master Arbitration 68
I^2C Reception (7-bit address) 71
I^2C Slave-Receiver Acknowledge 66
I^2C STARTand STOP Conditions 65
I^2C Transmission (7-bit address) 71
INT Pin Interrupt Timing 108
LCD Half-Duty Cycle Drive 86
LCD Interrupt Timing in Quarter-Duty Cycle Drive 91
LCD One-Third Duty Cycle Drive 87
LCD Quarter-Duty Cycle Drive 88
LCD SLEEP Entry/Exit (SLPEN=1) 93
LCD Static Drive 85
SPI (Master Mode) 63
SPI (Slave Mode, CKE = 0) 63
SPI (Slave Mode, CKE = 1) 64
Successive I/O Operation 39
Time-out Sequences on Power-up 106
Timer0 Interrupt Timing 42
Timer0 with External Clock 43

Timer0,Internal Timing 42
Wake-up from SLEEP through Interrupt 112
Timing Diagrams and Specifications 147
Timing Parameter Symbology 146
$\overline{TO}$ bit 19
TRISA Register 29
Initialization State 104
TRISB Register 31
Initialization State 104
TRISC Register 33
Initialization State 104
TRISD Register 34
Initialization State 104
TRISE Register 36
Initialization State 104
TRISF Register 37
Initialization States 105
TRISG Register 38
Initialization States 105

W

W Register
Initialization States 104
Wake-up from SLEEP 111
Interrupts 103
Watchdog Timer (WDT) 97, 101, 110
Associated Registers 110
WDT Reset, Normal Operation 103
WDT Reset, SLEEP 103
WCOL 60
WDT
Period 110
Programming Considerations 110
Timeout 104
Write Collision Detect bit, WCOL 60
WWW, On-Line Support 4, 175

X

XMIT_MODE 73
XT 99, 102

Z

Z (Zero) bit 19

ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web (WWW) 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 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 by using your favorite Internet browser to attach to:

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 any currently available 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.

013001

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 Data Sheet.

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: **PIC16C925/926** Literature Number: **DS39544A**

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 data sheet easy to follow? If not, why?

4. What additions to the data sheet do you think would enhance the structure and subject?

5. What deletions from the data sheet 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?

8. How would you improve our software, systems, and silicon products?

PIC16C925/926 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	PIC16C92X(1), PIC16C92XT(2); VDD range 4.0V to 5.5V PIC16LC92X(1), PIC16LC92XT(2); VDD range 2.5V to 5.5V			
Temperature Range	I	=	-40°C to +85°C	(Industrial)
	S	=	-40°C to +85°C	(Industrial, tape/reel)
	-	=	0°C to +70°C	(Commercial)
	T	=	0°C to +70°C	(Commercial, tape/reel)
Package	CL	=	Windowed CERQUAD(3)	
	PT	=	TQFP (Thin Quad Flatpack)	
	L	=	PLCC	
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)			

Examples:

a) PIC16C926/P 301 = Commercial Temp., normal VDD limits, QTP pattern #301

b) PIC16LC925/PT = Commercial Temp., TQFP package, extended VDD limits

c) PIC16C925-I/CL = Industrial Temp., windowed CERQUAD package, normal VDD limits

Note 1: C = Standard Voltage range
LC = Wide Voltage Range

2: T = in tape and reel - PLCC and TQFP packages only.

3: CL Devices are UV erasable and can be programmed to any device configuration. CL devices meet the electrical requirement of each oscillator type (including LC devices).

Sales and Support

Data Sheets
Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 792-7277
3. The Microchip Worldwide Site (www.microchip.com)

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

New Customer Notification System
Register on our web site (www.microchip.com/cn) to receive the most current information on our products.

NOTES:

NOTES:

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

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

Atlanta
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Austin
Analog Product Sales
8303 MoPac Expressway North
Suite A-201
Austin, TX 78759
Tel: 512-345-2030 Fax: 512-345-6085

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Boston
Analog Product Sales
Unit A-8-1 Millbrook Tarry Condominium
97 Lowell Road
Concord, MA 01742
Tel: 978-371-6400 Fax: 978-371-0050

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

Dayton
Two Prestige Place, Suite 130
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

Mountain View
Analog Product Sales
1300 Terra Bella Avenue
Mountain View, CA 94043-1836
Tel: 650-968-9241 Fax: 650-967-1590

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose
Microchip Technology Inc.
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
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Beijing Office
Unit 915
New China Hong Kong Manhattan Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Shanghai
Microchip Technology Shanghai Office
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

Hong Kong
Microchip Asia Pacific
RM 2101, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
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
Microchip Technology Intl. Inc.
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

ASIA/PACIFIC (continued)

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Denmark ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Arizona Microchip Technology SARL
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
Arizona Microchip Technology GmbH
Gustav-Heinemann Ring 125
D-81739 Munich, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Germany
Analog Product Sales
Lochhamer Strasse 13
D-82152 Martinsried, Germany
Tel: 49-89-895650-0 Fax: 49-89-895650-22

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/30/01

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC16C926-I/PT PIC16LC925-I/L PIC16LC925-I/PT PIC16C926-I/L PIC16LC926-I/PT PIC16C925-I/PT PIC16C926T-I/PT PIC16C926T-I/L PIC16C925T-I/L PIC16LC926T-I/PT PIC16C925-I/L PIC16LC926-I/L