



Welcome to [E-XFL.COM](https://www.e-xfl.com)

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	4MHz
Connectivity	I ² C, SPI
Peripherals	LCD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	7KB (4K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	176 x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	68-PLCC (24.23x24.23)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lc923t-04i-l

PIC16C9XX

NOTES:

6.0 OVERVIEW OF TIMER MODULES

Each module can generate an interrupt to indicate that an event has occurred (e.g. timer overflow). Each of these modules is explained in full detail in the following sections. The timer modules are:

- Timer0 Module (Section 7.0)
- Timer1 Module (Section 8.0)
- Timer2 Module (Section 9.0)

6.1 Timer0 Overview

The Timer0 module is a simple 8-bit timer/counter. The clock source can be either the internal system clock ($F_{osc}/4$) or an external clock. When the clock source is an external clock, the Timer0 module can be selected to increment on either the rising or falling edge.

The Timer0 module also has a programmable prescaler option. This prescaler can be assigned to either the Timer0 module or the Watchdog Timer. Bit PSA ($OPTION<3>$) assigns the prescaler, and bits PS2:PS0 ($OPTION<2:0>$) determine the prescaler value. Timer0 can increment at the following rates: 1:1 when prescaler assigned to Watchdog timer, 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128, and 1:256.

Synchronization of the external clock occurs after the prescaler. When the prescaler is used, the external clock frequency may be higher than the device's frequency. The maximum frequency is 50 MHz, given the high and low time requirements of the clock.

6.2 Timer1 Overview

Timer1 is a 16-bit timer/counter. The clock source can be either the internal system clock ($F_{osc}/4$), an external clock, or an external crystal. Timer1 can operate as either a timer or a counter. When operating as a counter (external clock source), the counter can either operate synchronized to the device or asynchronously to the device. Asynchronous operation allows Timer1 to operate during sleep, which is useful for applications that require a real-time clock as well as the power savings of SLEEP mode.

Timer1 also has a prescaler option which allows Timer1 to increment at the following rates: 1:1, 1:2, 1:4, and 1:8. Timer1 can be used in conjunction with the Capture/Compare/PWM module. When used with a CCP module, Timer1 is the time-base for 16-bit capture or the 16-bit compare and must be synchronized to the device. Timer1 oscillator is also one of the clock sources for the LCD module.

6.3 Timer2 Overview

Timer2 is an 8-bit timer with a programmable prescaler and postscaler, as well as an 8-bit period register (PR2). Timer2 can be used with the CCP1 module (in PWM mode) as well as the clock source for the Syn-

chronous Serial Port (SSP). The prescaler option allows Timer2 to increment at the following rates: 1:1, 1:4, 1:16.

The postscaler allows the TMR2 register to match the period register (PR2) a programmable number of times before generating an interrupt. The postscaler can be programmed from 1:1 to 1:16 (inclusive).

6.4 CCP Overview

The CCP module can operate in one of these three modes: 16-bit capture, 16-bit compare, or up to 10-bit Pulse Width Modulation (PWM).

Capture mode captures the 16-bit value of TMR1 into the CCPR1H:CCPR1L register pair. The capture event can be programmed for either the falling edge, rising edge, fourth rising edge, or the sixteenth rising edge of the CCP1 pin.

Compare mode compares the TMR1H:TMR1L register pair to the CCPR1H:CCPR1L register pair. When a match occurs an interrupt can be generated, and the output pin CCP1 can be forced to given state (High or Low), TMR1 can be reset and start A/D conversion. This depends on the control bits CCP1M3:CCP1M0.

PWM mode compares the TMR2 register to a 10-bit duty cycle register (CCPR1H:CCPR1L<5:4>) as well as to an 8-bit period register (PR2). When the TMR2 register = Duty Cycle register, the CCP1 pin will be forced low. When $TMR2 = PR2$, TMR2 is cleared to 00h, an interrupt can be generated, and the CCP1 pin (if an output) will be forced high.

7.2 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.

7.2.1 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 7-5). 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.

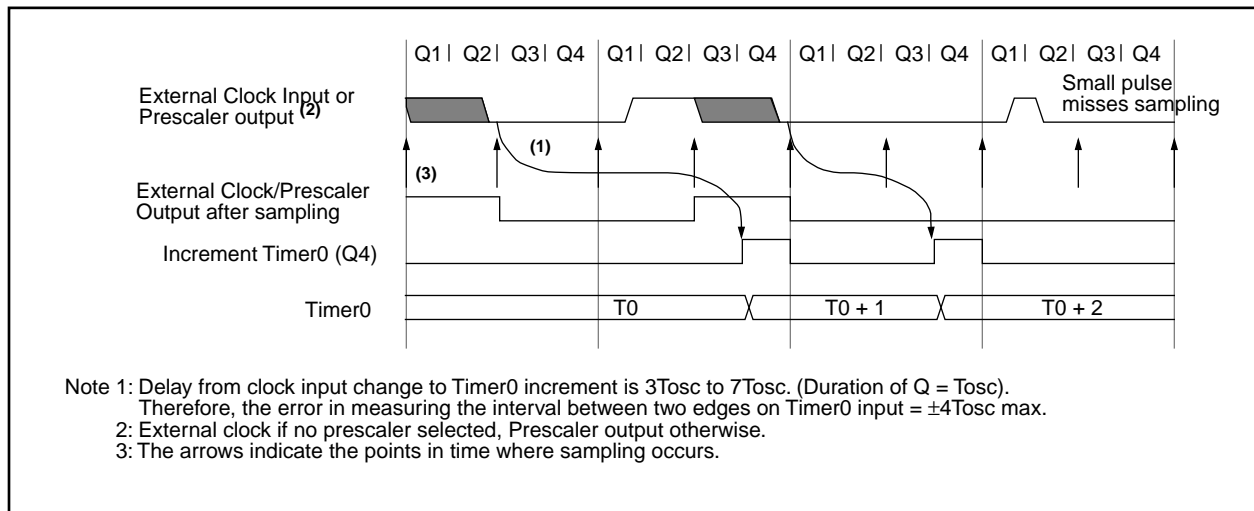
When a prescaler is used, the external clock input is divided by the asynchronous ripple-counter type pres-

caler 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.

7.2.2 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 7-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 7-5: TIMER0 TIMING WITH EXTERNAL CLOCK



8.3 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{T1SYNC}$ ($T1CON<2>$) 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 8.3.2).

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

8.3.1 EXTERNAL CLOCK INPUT TIMING WITH UNSYNCHRONIZED CLOCK

If control bit $\overline{T1SYNC}$ 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.

8.3.2 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 8-1 is an example routine to read the 16-bit timer value. This is useful if the timer cannot be stopped.

EXAMPLE 8-1: 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
```

8.4 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 8-1 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 8-1: 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 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.			

11.3.1.2 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 11-19: I²C WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)

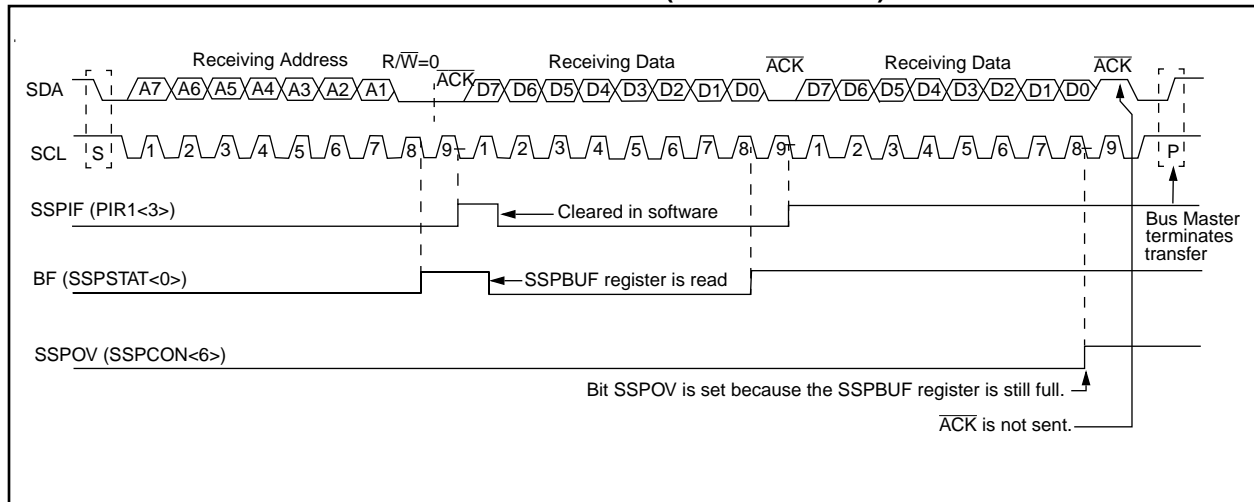


TABLE 14-6: INITIALIZATION CONDITIONS FOR ALL REGISTERS (Cont.'d)

Register	Applicable Devices		Power-on Reset	MCLR Resets WDT Reset	Wake-up via WDT or Interrupt
PORTD	923	924	0000 0000	0000 0000	uuuu uuuu
PORTE	923	924	0000 0000	0000 0000	uuuu uuuu
PCLATH	923	924	---0 0000	---0 0000	---u uuuu
INTCON	923	924	0000 000x	0000 000u	uuuu uuuu ⁽¹⁾
PIR1 ⁽⁴⁾	923	924	00-- 0000	00-- 0000	uu-- uuuu ⁽¹⁾
TMR1L	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	923	924	--00 0000	--uu uuuu	--uu uuuu
TMR2	923	924	0000 0000	0000 0000	uuuu uuuu
T2CON	923	924	-000 0000	-000 0000	-uuu uuuu
SSPBUF	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	923	924	0000 0000	0000 0000	uuuu uuuu
CCPR1L	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	923	924	--00 0000	--00 0000	--uu uuuu
ADRES	923	924	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	923	924	0000 00-0	0000 00-0	uuuu uu-u
OPTION	923	924	1111 1111	1111 1111	uuuu uuuu
TRISA	923	924	--11 1111	--11 1111	--uu uuuu
TRISB	923	924	1111 1111	1111 1111	uuuu uuuu
TRISC	923	924	--11 1111	--11 1111	--uu uuuu
TRISD	923	924	1111 1111	1111 1111	uuuu uuuu
TRISE	923	924	1111 1111	1111 1111	uuuu uuuu
PIE1 ⁽⁴⁾	923	924	00-- 0000	00-- 0000	uu-- uuuu
PCON	923	924	---- --0-	---- --u-	---- --u-
PR2	923	924	1111 1111	1111 1111	1111 1111
SSPADD	923	924	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	923	924	0000 0000	0000 0000	uuuu uuuu
ADCON1	923	924	---- -000	---- -000	---- -uuu
PORTF	923	924	0000 0000	0000 0000	uuuu uuuu
PORTG	923	924	0000 0000	0000 0000	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 14-5 for reset value for specific condition.

4: Bits PIE1<6> and PIR1<6> are reserved on the PIC16C923, always maintain these bits clear.

5: PORTA values when read.

FIGURE 14-8: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 1

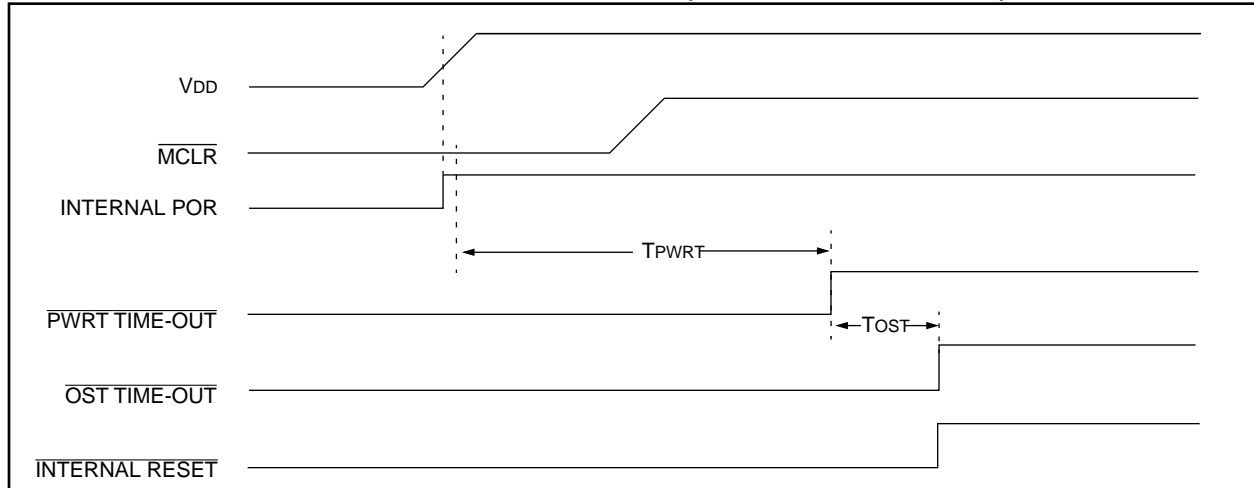


FIGURE 14-9: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 2

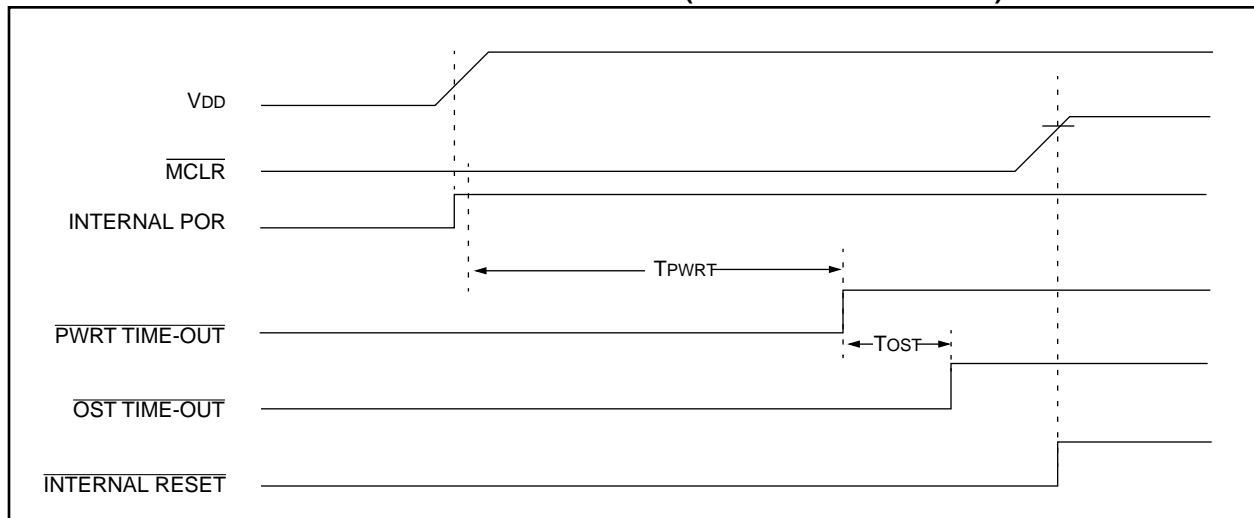
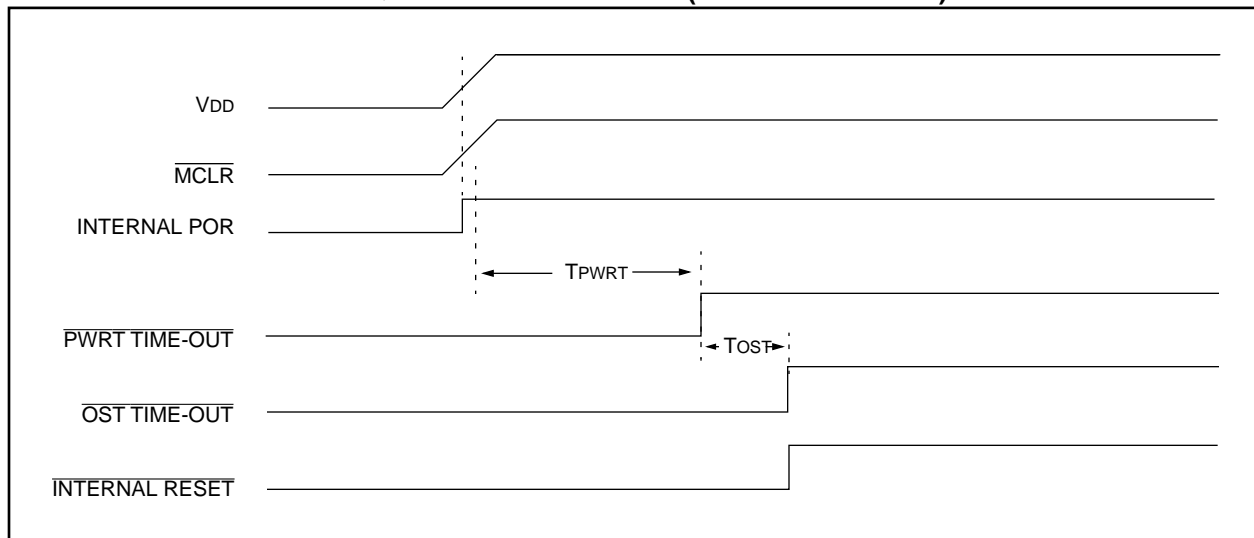


FIGURE 14-10: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO V_{DD})



15.1 Instruction Descriptions

ADDLW Add Literal and W

Syntax: `[label] ADDLW k`

Operands: $0 \leq k \leq 255$

Operation: $(W) + k \rightarrow (W)$

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

Operation: $(W) + (f) \rightarrow (\text{destination})$

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

PIC16C9XX

BTFSS Bit Test f, Skip if Set

Syntax: `[label] BTFSS f,b`

Operands: $0 \leq f \leq 127$
 $0 \leq b < 7$

Operation: skip if $(f \ll b) = 1$

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: $0 \leq k \leq 2047$

Operation: $(PC)+1 \rightarrow TOS$,
 $k \rightarrow PC \ll 10:0$,
 $(PCLATH \ll 4:3) \rightarrow PC \ll 12:11$

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 $\ll 10:0$. 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

DECFSZ Decrement f, Skip if 0

Syntax: [*label*] DECFSZ f,d

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

Operation: $(f) - 1 \rightarrow (\text{destination});$
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: $0 \leq k \leq 2047$

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:

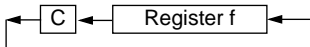
	Q1	Q2	Q3	Q4
1st Cycle	Decode	Read literal 'k'	Process data	Write to PC
2nd Cycle	No-Operation	No-Operation	No-Operation	No-Operation

Example

GOTO THERE

After Instruction

PC = Address THERE

RLF		Rotate Left f through Carry						
Syntax:	[<i>label</i>] RLF f,d							
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$							
Operation:	See description below							
Status Affected:	C							
Encoding:	<table><tr><td>00</td><td>1101</td><td>dfff</td><td>ffff</td></tr></table>				00	1101	dfff	ffff
00	1101	dfff	ffff					
Description:	<p>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'.</p> 							
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

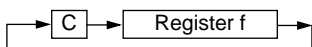
C = 0

After Instruction

REG1 = 1110 0110

W = 1100 1100

C = 1

RRF		Rotate Right f through Carry							
Syntax:	[<i>label</i>] RRF f,d								
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$								
Operation:	See description below								
Status Affected:	C								
Encoding:	<table><tr><td>00</td><td>1100</td><td>dfff</td><td>ffff</td></tr></table>					00	1100	dfff	ffff
00	1100	dfff	ffff						
Description:	<p>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'.</p> 								
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

PIC16C9XX

XORLW Exclusive OR Literal with W

Syntax: `[label] XORLW k`

Operands: $0 \leq k \leq 255$

Operation: $(W) \text{ .XOR. } k \rightarrow (W)$

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

Operation: $(W) \text{ .XOR. } (f) \rightarrow (\text{destination})$

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
```

16.0 DEVELOPMENT SUPPORT

16.1 Development Tools

The PICmicro™ microcontrollers are supported with a full range of hardware and software development tools:

- PICMASTER/PICMASTER CE Real-Time In-Circuit Emulator
- ICEPIC Low-Cost PIC16C5X and PIC16CXXX In-Circuit Emulator
- PRO MATE® II Universal Programmer
- PICSTART® Plus Entry-Level Prototype Programmer
- PICDEM-1 Low-Cost Demonstration Board
- PICDEM-2 Low-Cost Demonstration Board
- PICDEM-3 Low-Cost Demonstration Board
- MPASM Assembler
- MPLAB™ SIM Software Simulator
- MPLAB-C (C Compiler)
- Fuzzy Logic Development System (*fuzzyTECH*®-MP)

16.2 PICMASTER: High Performance Universal In-Circuit Emulator with MPLAB IDE

The PICMASTER Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for all microcontrollers in the PIC12CXXX, PIC14C000, PIC16C5X, PIC16CXXX and PIC17CXX families. PICMASTER is supplied with the MPLAB™ Integrated Development Environment (IDE), which allows editing, “make” and download, and source debugging from a single environment.

Interchangeable target probes allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the PICMASTER allows expansion to support all new Microchip microcontrollers.

The PICMASTER 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 compatible 386 (and higher) machine platform and Microsoft Windows® 3.x environment were chosen to best make these features available to you, the end user.

A CE compliant version of PICMASTER is available for European Union (EU) countries.

16.3 ICEPIC: Low-Cost PIC16CXXX In-Circuit Emulator

ICEPIC is a low-cost in-circuit emulator solution for the Microchip PIC16C5X and PIC16CXXX families of 8-bit OTP microcontrollers.

ICEPIC is designed to operate on PC-compatible machines ranging from 286-AT® through Pentium™ based machines under Windows 3.x environment. ICEPIC features real time, non-intrusive emulation.

16.4 PRO MATE II: Universal Programmer

The PRO MATE II Universal Programmer is a full-featured programmer capable of operating in stand-alone mode as well as PC-hosted mode.

The PRO MATE II has programmable VDD and VPP supplies which allows it to verify programmed memory at VDD min and VDD max for maximum reliability. It has an LCD display for displaying 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 can read, verify or program PIC12CXXX, PIC14C000, PIC16C5X, PIC16CXXX and PIC17CXX devices. It can also set configuration and code-protect bits in this mode.

16.5 PICSTART Plus Entry Level Development System

The PICSTART programmer is an easy-to-use, low-cost prototype programmer. It connects to the PC via one of the COM (RS-232) ports. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. PICSTART Plus is not recommended for production programming.

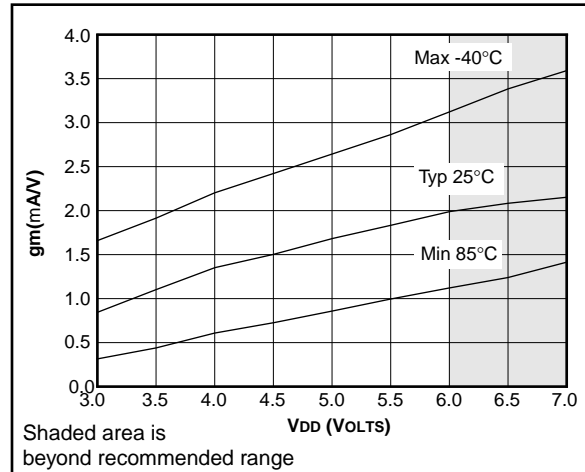
PICSTART Plus supports all PIC12CXXX, PIC14C000, PIC16C5X, PIC16CXXX and PIC17CXX devices with up to 40 pins. Larger pin count devices such as the PIC16C923 and PIC16C924 may be supported with an adapter socket.

**TABLE 18-1: RC OSCILLATOR
FREQUENCIES**

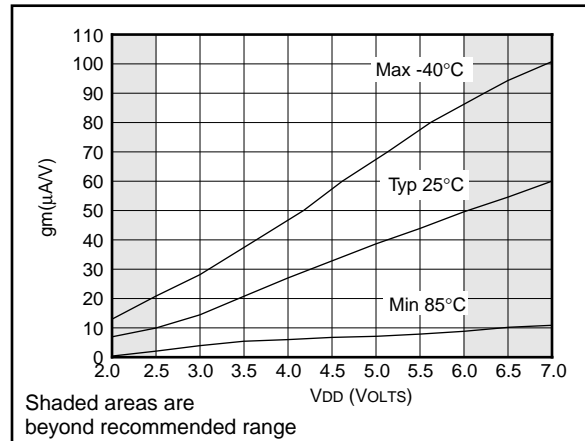
Cext	Rext	Average	
		Fosc @ 5V, 25°C	
22 pF	5k	4.12 MHz	± 1.4%
	10k	2.35 MHz	± 1.4%
	100k	268 kHz	± 1.1%
100 pF	3.3k	1.80 MHz	± 1.0%
	5k	1.27 MHz	± 1.0%
	10k	688 kHz	± 1.2%
	100k	77.2 kHz	± 1.0%
300 pF	3.3k	707 kHz	± 1.4%
	5k	501 kHz	± 1.2%
	10k	269 kHz	± 1.6%
	100k	28.3 kHz	± 1.1%

The percentage variation indicated here is part to part variation due to normal process distribution. The variation indicated is ± 3 standard deviation from average value for $V_{DD} = 5V$.

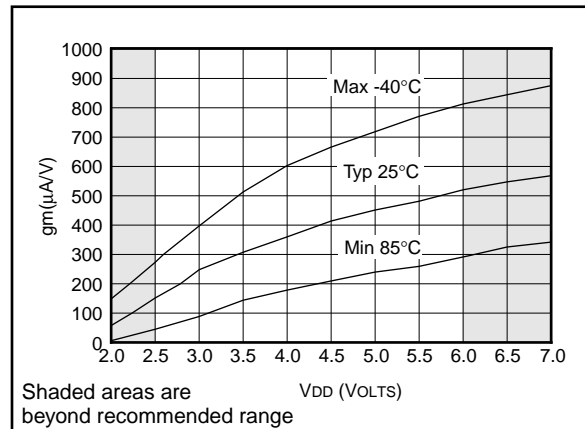
**FIGURE 18-20: TRANSCONDUCTANCE(gm)
OF HS OSCILLATOR vs. VDD**



**FIGURE 18-21: TRANSCONDUCTANCE(gm)
OF LP OSCILLATOR vs. VDD**



**FIGURE 18-22: TRANSCONDUCTANCE(gm)
OF XT OSCILLATOR vs. VDD**



Data based on process characterization samples. See first page of this section for details.

FIGURE 18-28: TYPICAL I_{DD} vs. V_{DD}
(XT MODE @ 25°C)

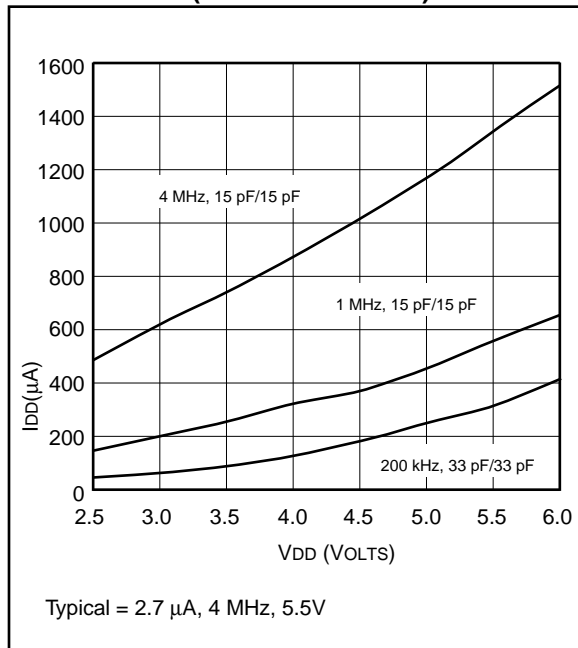


FIGURE 18-30: TYPICAL I_{DD} vs. V_{DD}
(HS MODE @ 25°C)

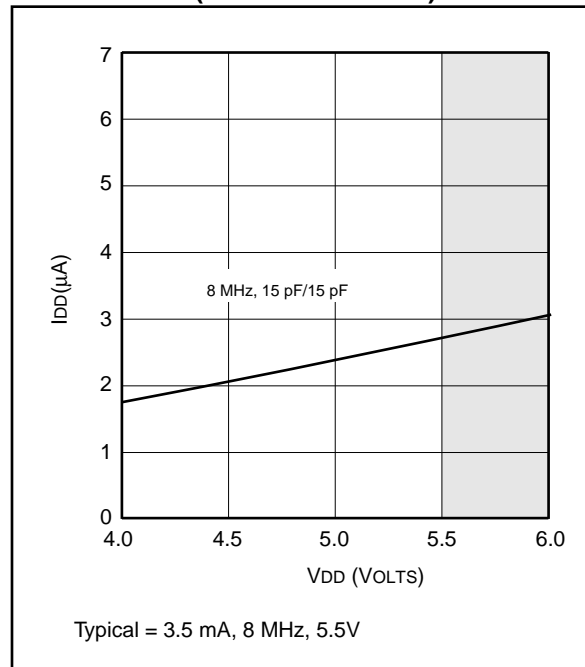


FIGURE 18-29: MAXIMUM I_{DD} vs. V_{DD}
(XT MODE -40°C TO +85°C)

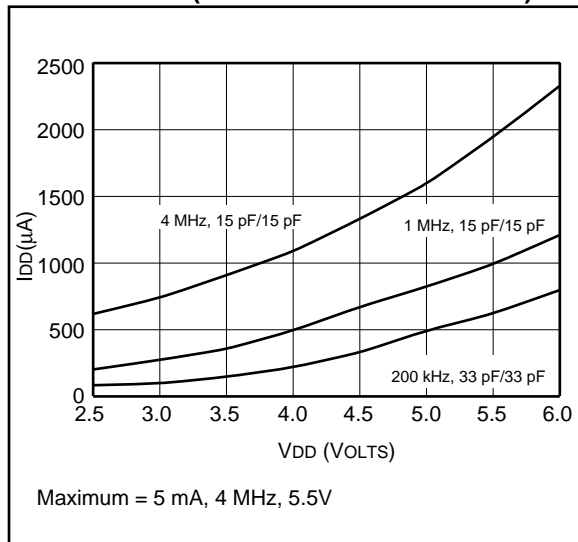
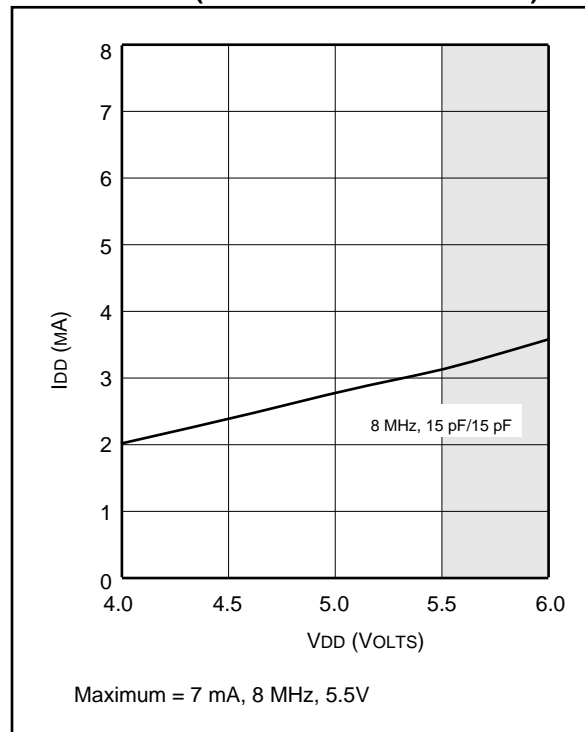


FIGURE 18-31: MAXIMUM I_{DD} vs. V_{DD}
(HS MODE -40°C TO +85°C)

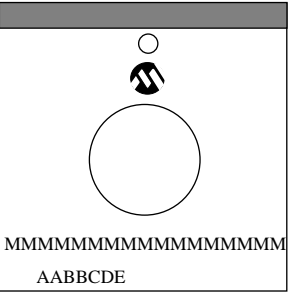


Data based on process characterization samples. See first page of this section for details.

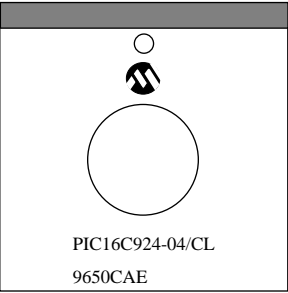
PIC16C9XX

19.4 Package Marking Information

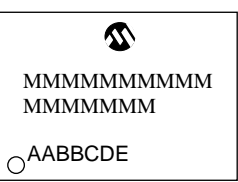
68-Lead CERQUAD Windowed



Example



64-Lead TQFP



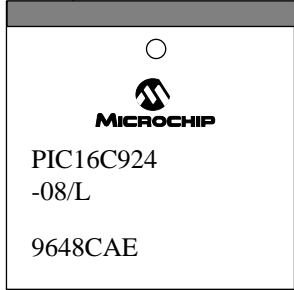
Example



68-Lead PLCC



Example



64-Lead SDIP (Shrink DIP)



Example



Legend:	MM...M	Microchip part number information
	XX...X	Customer specific information*
	AA	Year code (last 2 digits of calendar year)
	BB	Week code (week of January 1 is week '01')
	C	Facility code of the plant at which wafer is manufactured. C = Chandler, Arizona, U.S.A. S = Tempe, Arizona, U.S.A.
	D ₁	Mask revision number for microcontroller
	E	Assembly code of the plant or country of origin in which part was assembled.
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, facility code, mask revision number, and assembly 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.

INDEX

A

A/D

Accuracy/Error	86
ADCON0	79, 80
ADCON1	79, 80
ADIF	80
Analog-to-Digital Converter	79
Configuring Analog Port	83
Connection Considerations	87
Conversion time	85
Conversions	84
Converter Characteristics	158
Faster Conversion - Lower Resolution Tradeoff	85
GO/DONE	80
Internal Sampling Switch (Rss) Impedance	82
Operation During Sleep	86
Sampling Requirements	82
Sampling Time	82
Source Impedance	82
Transfer Function	87

A/D Conversion Clock	83
----------------------------	----

Registers

Section	19
Absolute Maximum Ratings	141
ACK	70, 74, 75, 76, 77
ADCON0 Register	19
ADCON1 Register	20
ADIE bit	26
ADIF bit	27
ADRES	19, 79, 80, 109
ALU	9

Application Notes

AN546	79
AN552	33
AN556	29
AN578	63
AN594	57
AN607	107

Architecture

Harvard	9
Overview	9
von Neumann	9

Assembler

MPASM Assembler	138
-----------------------	-----

B

BF	74
----------	----

Block Diagrams

A/D	81
Capture Mode	58
Compare Mode	58
External Brown-out1	112
External Brown-out2	112
External Parallel Crystal Oscillator	105
External Power-on Reset	112
External Series Crystal Oscillator	105
Interrupt Logic	114
LCD Module	90
On-Chip Reset Circuit	106
PIC16C923	10
PIC16C924	11
PORTC	35
PORTD	36, 37
PORTE	38

PORTF	39
PORTG	40
PWM	59
RA3:RA0 and RA5 Port Pins	31
RA4/T0CKI Pin	31
RB3:RB0 Port Pins	33
RB7:RB4 Port Pins	33
RC Oscillator	105
SSP (I ² C Mode)	73
SSP (SPI Mode)	65
Timer0	45
Timer0/WDT Prescaler	48
Timer1	52
Timer2	55
Watchdog Timer	116
Brown-out Protection Circuit	112

C

C bit	23
-------------	----

Capture/Compare/PWM (CCP)

Capture Mode	58
CCP1	57
CCP1CON	109
CCPR1H	109
CCPR1L	109
Compare Mode	58
Compare Mode Block Diagram	58
Prescaler	58
PWM Block Diagram	59
PWM Mode	59
PWM, Example Frequencies/Resolutions	60
Section	57

Carry bit	9
-----------------	---

CCP1CON Register	19
------------------------	----

CCP1IE bit	26
------------------	----

CCP1IF bit	27
------------------	----

CCPR1H Register	19
-----------------------	----

CCPR1L Register	19
-----------------------	----

Clocking Scheme	15
-----------------------	----

Code Examples

Call of a Subroutine in Page 1 from Page 0	30
Changing Between Capture Prescalers	58
Changing Prescaler (Timer0 to WDT)	49
Changing Prescaler (WDT to Timer0)	49
Doing an A/D Conversion	84
I/O Programming	41
I ² C Module Operation	78
Indirect Addressing	30
Initializing PORTA	31
Initializing PORTB	33
Initializing PORTC	35
Initializing PORTD	36
Initializing PORTE	38
Initializing PORTF	39
Initializing PORTG	40
Loading the SSPBUF register	65
Reading a 16-bit Free-running Timer	53

Code Protection	103, 118
-----------------------	----------

Computed GOTO	29
---------------------	----

Configuration Bits	103
--------------------------	-----

D

DC bit	23
--------------	----

DC Characteristics	142, 143
--------------------------	----------

Development Support	137
---------------------------	-----

Development Tools	137
-------------------------	-----

Digit Carry bit	9
-----------------------	---

PIC16C9XX

Direct Addressing.....	30
E	
Electrical Characteristics.....	141
External Power-on Reset Circuit.....	112
F	
Family of Devices	
PIC16C9XX.....	6
FSR.....	108
FSR Register.....	19, 20, 21, 22, 30
Fuzzy Logic Dev. System (<i>fuzzyTECH®</i> -MP).....	139
G	
GIE.....	113
I	
I/O Ports	
Section.....	31
I/O Programming Considerations.....	41
I ² C	
Addressing I ² C Devices.....	70
Arbitration.....	72
BF.....	74, 75
CKP.....	76
Clock Synchronization.....	72
Combined Format.....	71
I ² C Overview.....	69
Initiating and Terminating Data Transfer.....	69
Master-Receiver Sequence.....	71
Master-Transmitter Sequence.....	71
Multi-master.....	72
START.....	69
STOP.....	69, 70
Transfer Acknowledge.....	70
ICEPIC Low-Cost PIC16CXXX In-Circuit Emulator.....	137
IDLE_MODE.....	78
In-Circuit Serial Programming.....	103, 118
INDF.....	108
INDF Register.....	19, 20, 21, 22, 30
Indirect Addressing.....	30
Instruction Cycle.....	15
Instruction Flow/Pipelining.....	15
Instruction Format.....	119
Instruction Set	
ADDLW.....	121
ADDWF.....	121
ANDLW.....	122
ANDWF.....	122
BCF.....	122
BSF.....	123
BTFSC.....	123
BTFSS.....	124
CALL.....	124
CLRF.....	125
CLRW.....	125
CLRWDI.....	126
COMF.....	126
DECF.....	126
DECFSZ.....	127
GOTO.....	127
INCF.....	128
INCFSZ.....	128
IORLW.....	129
IORWF.....	129
MOVF.....	130
MOVLW.....	130
MOVWF.....	130

NOP.....	131
OPTION.....	131
RETFIE.....	131
RETLW.....	132
RETURN.....	132
RLF.....	133
RRF.....	133
SLEEP.....	134
SUBLW.....	134
SUBWF.....	135
SWAPF.....	135
TRIS.....	135
XORLW.....	136
XORWF.....	136
Section.....	119
INT Interrupt.....	115
INTCON.....	109, 113, 115
INTCON Register.....	19, 20, 21, 22, 25, 102
INTEDG.....	115
INTEDG bit.....	24
Inter-Integrated Circuit (I ² C).....	63
Internal Sampling Switch (R _{ss}) Impedance.....	82
Interrupt Flag.....	113
Interrupts.....	103, 113
RB7:RB4 Port Change.....	33
IRP bit.....	23
K	
KeeLoq® Evaluation and Programming Tools.....	139
L	
Loading of PC.....	29
M	
MCLR.....	106, 108
Memory	
Data Memory.....	17
Maps, PIC16C9XX.....	17
Program Memory.....	17
MP-DriveWay™ - Application Code Generator.....	139
MPLAB C.....	139
MPLAB Integrated Development Environment Software..	138
O	
One-Time-Programmable Devices.....	7
OPCODE.....	119
OPTION.....	109, 115
OPTION Register.....	20, 22, 24
Orthogonal.....	9
OSC selection.....	103
Oscillator	
HS.....	104, 107
LP.....	104, 107
Oscillator Configurations.....	104
Output of TMR2.....	55
P	
Paging, Program Memory.....	29
PC.....	108
PCL Register.....	19, 20, 21, 22, 29
PCLATH.....	109
PCLATH Register.....	19, 20, 21, 22, 29
PCON.....	109
PCON Register.....	28
PD.....	106, 108
PD bit.....	23
PICDEM-1 Low-Cost PICmicro Demo Board.....	138
PICDEM-2 Low-Cost PIC16CXX Demo Board.....	138
PICDEM-3 Low-Cost PIC16CXXX Demo Board.....	138

List of Equations And Examples

Example 3-1: Instruction Pipeline Flow	15
Example 4-1: Call of a Subroutine in Page 1 from Page 030	30
Example 4-2: Indirect Addressing	30
Example 5-1: Initializing PORTA	31
Example 5-2: Initializing PORTB	33
Example 5-3: Initializing PORTC	35
Example 5-4: Initializing PORTD	36
Example 5-5: Initializing PORTE	38
Example 5-6: Initializing PORTF	39
Example 5-7: Initializing PORTG	40
Example 5-8: Read-Modify-Write Instructions on an I/O Port	41
Example 7-1: Changing Prescaler (Timer0→WDT)	49
Example 7-2: Changing Prescaler (WDT→Timer0)	49
Example 8-1: Reading a 16-bit Free-Running Timer	53
Example 10-1: Changing Between Capture Prescalers	58
Example 10-2: PWM Period and Duty Cycle Calculation ...	60
Example 11-1: Loading the SSPBUF (SSPSR) Register....	65
Equation 12-1: A/D Minimum Charging Time.....	82
Example 12-1: Calculating the Minimum Required Sample Time.....	82
Example 12-2: Doing an A/D Conversion	84
Example 12-3: 4-bit vs. 8-bit Conversion Times	85
Example 13-1: Static MUX with 32 Segments	100
Example 13-2: 1/3 MUX with 13 Segments	100
Example 14-1: Saving STATUS, W, and PCLATH Registers in RAM	115

List of Figures

Figure 3-1: PIC16C923 Block Diagram	10
Figure 3-2: PIC16C924 Block Diagram	11
Figure 3-3: Clock/Instruction Cycle	15
Figure 4-1: Program Memory Map and Stack	17
Figure 4-2: Register File Map	18
Figure 4-3: Status Register (Address 03h, 83h, 103h, 183h).....	23
Figure 4-4: OPTION Register (Address 81h, 181h)	24
Figure 4-5: INTCON Register (Address 0Bh, 8Bh, 10Bh, 18Bh).....	25
Figure 4-6: PIE1 Register (Address 8Ch)	26
Figure 4-7: PIR1 Register (Address 0Ch)	27
Figure 4-8: PCON Register (Address 8Eh)	28
Figure 4-9: Loading of PC In Different Situations.....	29
Figure 4-10: Direct/Indirect Addressing.....	30
Figure 5-1: Block Diagram of pins RA3:RA0 and RA5..	31
Figure 5-2: Block Diagram of RA4/T0CKI Pin	31
Figure 5-3: Block Diagram of RB3:RB0 Pins	33
Figure 5-4: Block Diagram of RB7:RB4 Pins	33
Figure 5-5: PORTC Block Diagram (Peripheral Output Override).....	35
Figure 5-6: PORTD<4:0> Block Diagram.....	36
Figure 5-7: PORTD<7:5> Block Diagram.....	37
Figure 5-8: PORTE Block Diagram	38
Figure 5-9: PORTF Block Diagram	39
Figure 5-10: PORTG Block Diagram.....	40
Figure 5-11: Successive I/O Operation.....	41
Figure 7-1: Timer0 Block Diagram	45
Figure 7-2: Timer0 Timing: Internal Clock/No Prescale	45
Figure 7-3: Timer0 Timing: Internal Clock/Prescale 1:2	46
Figure 7-4: Timer0 Interrupt Timing	46
Figure 7-5: Timer0 Timing with External Clock	47
Figure 7-6: Block Diagram of the Timer0/WDT Prescaler.....	48

Figure 8-1: T1CON: Timer1 Control Register (Address 10h)	51
Figure 8-2: Timer1 Block Diagram.....	52
Figure 9-1: Timer2 Block Diagram	55
Figure 9-2: T2CON: Timer2 Control Register (Address 12h)	55
Figure 10-1: CCP1CON Register (Address 17h).....	57
Figure 10-2: Capture Mode Operation Block Diagram	58
Figure 10-3: Compare Mode Operation Block Diagram ..	58
Figure 10-4: Simplified PWM Block Diagram.....	59
Figure 10-5: PWM Output.....	59
Figure 11-1: SSPSTAT: Sync Serial Port Status Register (Address 94h)	63
Figure 11-2: SSPCON: Sync Serial Port Control Register (Address 14h)	64
Figure 11-3: SSP Block Diagram (SPI Mode).....	65
Figure 11-4: SPI Master/Slave Connection	66
Figure 11-5: SPI Mode Timing, Master Mode	67
Figure 11-6: SPI Mode Timing (Slave Mode With CKE = 0)	67
Figure 11-7: SPI Mode Timing (Slave Mode With CKE = 1)	68
Figure 11-8: Start and Stop Conditions	69
Figure 11-9: 7-bit Address Format.....	70
Figure 11-10: I ² C 10-bit Address Format.....	70
Figure 11-11: Slave-receiver Acknowledge	70
Figure 11-12: Data Transfer Wait State	70
Figure 11-13: Master-transmitter Sequence	71
Figure 11-14: Master-receiver Sequence	71
Figure 11-15: Combined Format.....	71
Figure 11-16: Multi-master Arbitration (Two Masters)	72
Figure 11-17: Clock Synchronization.....	72
Figure 11-18: SSP Block Diagram (I ² C Mode).....	73
Figure 11-19: I ² C Waveforms for Reception (7-bit Address).....	75
Figure 11-20: I ² C Waveforms for Transmission (7-bit Address).....	76
Figure 11-21: Operation of the I ² C Module in IDLE_MODE, RCV_MODE or XMIT_MODE	78
Figure 12-1: ADCON0 Register (Address 1Fh)	79
Figure 12-2: ADCON1 Register (Address 9Fh)	80
Figure 12-3: A/D Block Diagram	81
Figure 12-4: Analog Input Model	82
Figure 12-5: A/D Transfer Function	87
Figure 12-6: Flowchart of A/D Operation	88
Figure 13-1: LCDCON Register (Address 10Fh)	89
Figure 13-2: LCD Module Block Diagram	90
Figure 13-3: LCDPS Register (Address 10Eh)	90
Figure 13-4: Waveforms in Static Drive	91
Figure 13-5: Waveforms in 1/2 MUX, 1/3 Bias Drive	92
Figure 13-6: Waveforms in 1/3 MUX, 1/3 Bias	93
Figure 13-7: Waveforms in 1/4 MUX, 1/3 Bias	94
Figure 13-8: LCD Clock Generation	95
Figure 13-9: Example Waveforms in 1/4 MUX Drive	97
Figure 13-10: Generic LCDD Register Layout.....	98
Figure 13-11: Sleep Entry/exit When SLPEN = 1 or CS1:CS0 = 00	99
Figure 13-12: LCDSE Register (Address 10Dh)	100
Figure 13-13: Charge Pump and Resistor Ladder.....	101
Figure 14-1: Configuration Word	103
Figure 14-2: Crystal/Ceramic Resonator Operation (HS, XT or LP OSC Configuration).....	104
Figure 14-3: External Clock Input Operation (HS, XT or LP OSC Configuration).....	104

