Welcome to 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	Active
Core Processor	PIC
Core Size	8-Bit
Speed	4MHz
Connectivity	I²C, 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/pic16lc923-04i-l

TABLE 4-1: SPECIAL FUNCTION REGISTER SUMMARY (Cont.'d)

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
Bank 1											
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	0000 0000
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
83h	STATUS	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	000q quuu
84h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
86h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
87h	TRISC	—	—	PORTC Data Direction Register						--11 1111	--11 1111
88h	TRISD	PORTD Data Direction Register								1111 1111	1111 1111
89h	TRISE	PORTE Data Direction Register								1111 1111	1111 1111
8Ah	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	---0 0000
8Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
8Ch	PIE1	LCDIE	ADIE(2)	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
8Dh	—	Unimplemented								—	—
8Eh	PCON	—	—	—	—	—	—	$\overline{\text{POR}}$	—	---- --0-	---- --u-
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	—	Unimplemented								—	—
92h	PR2	Timer2 Period Register								1111 1111	1111 1111
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	0000 0000
94h	SSPSTAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	0000 0000	0000 0000
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	—	Unimplemented								—	—
99h	—	Unimplemented								—	—
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	—	Unimplemented								—	—
9Dh	—	Unimplemented								—	—
9Eh	—	Unimplemented								—	—
9Fh(1)	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	---- -000	---- -000

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

Note 1: Registers ADRES, ADCON0, and ADCON1 are not implemented in the PIC16C923, read as '0'.
2: These bits are reserved on the PIC16C923, always maintain these bits clear.
3: These pixels do not display, but can be used as general purpose RAM.
4: PIC16C923 reset values for PORTA: `--xx xxxx` for a POR, and `--uu uuuu` for all other resets, PIC16C924 reset values for PORTA: `--0x 0000` when read.
5: Bit1 of ADCON0 is reserved on the PIC16C924, always maintain this bit clear.

4.2.2.2 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.

FIGURE 4-4: 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
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR reset

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

5.8 I/O Programming Considerations

5.8.1 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 (ex. 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 5-8 shows the effect of two sequential read-modify-write instructions on an I/O port.

EXAMPLE 5-8: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

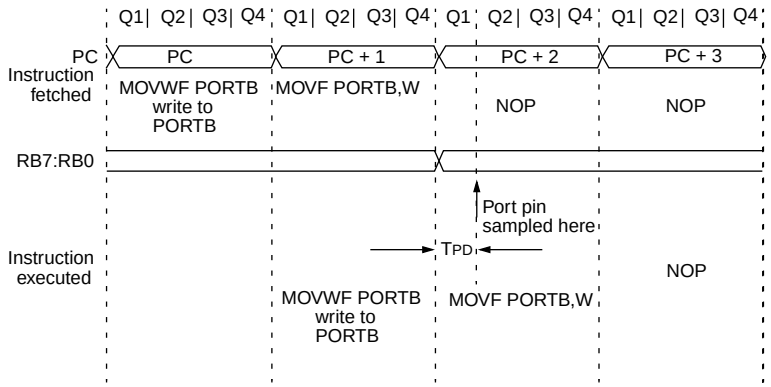
```
;Initial PORT settings: PORTB<7:4> Inputs
;                       PORTB<3:0> Outputs
;PORTB<7:6> have external pull-ups and are
;not connected to other circuitry
;
;                    PORT latch  PORT pins
;                    ----------  ---------
  BCF PORTB, 7    ; 01pp pppp    11pp pppp
  BCF PORTB, 6    ; 10pp pppp    11pp pppp
  BCF STATUS, RP1 ;
  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).
```

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.

5.8.2 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 5-11). 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 5-11: 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.

FIGURE 7-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

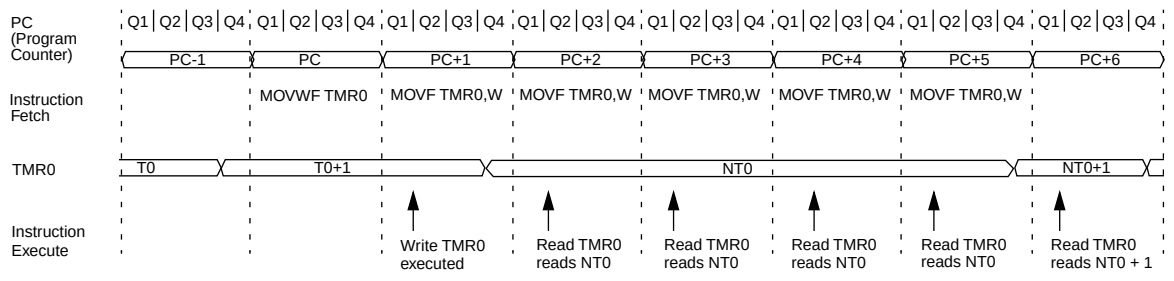
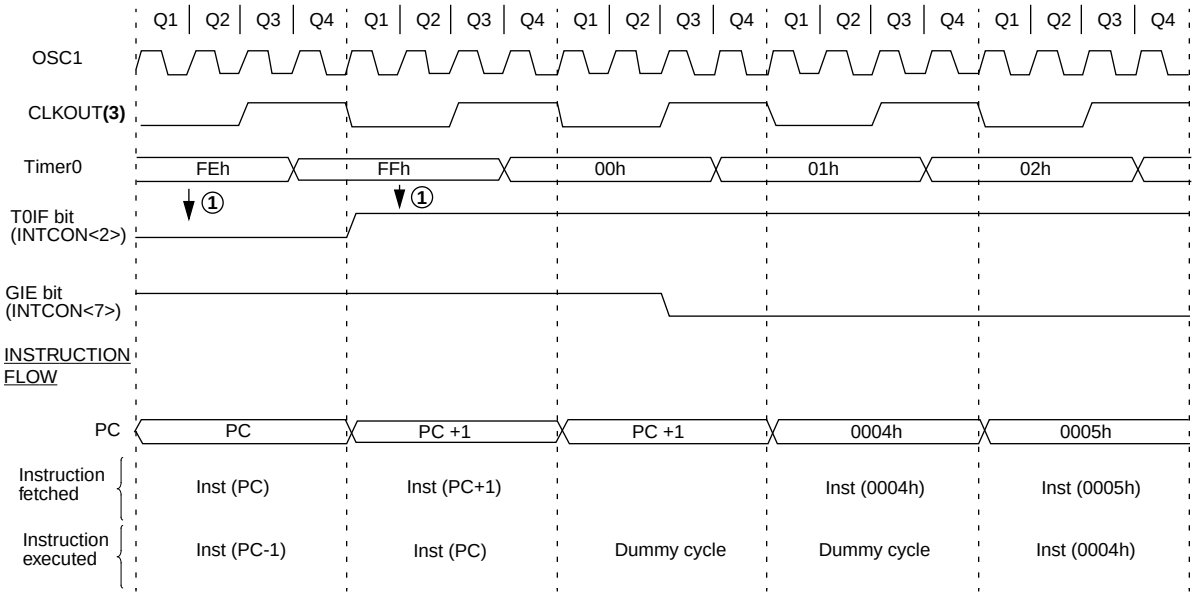


FIGURE 7-4: TIMER0 INTERRUPT TIMING



Note 1: Interrupt flag bit T0IF is sampled here (every Q1).
2: Interrupt latency = 4TCY where TCY = instruction cycle time.
3: CLKOUT is available only in RC oscillator mode.

8.3 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 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 $\overline{\text{T1SYNC}}$ 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.

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.

NOTES:

11.2 I²C™ Overview

This section provides an overview of the Inter-Integrated Circuit (I^2C) bus, with Section 11.3 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 11-2 defines some of the I^2C bus terminology. For additional information on the I^2C interface specification, refer to the Philips document "*The I^2C bus and how to use it.*" #939839340011, 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.

11.2.1 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 11-8 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 11-8: START AND STOP CONDITIONS

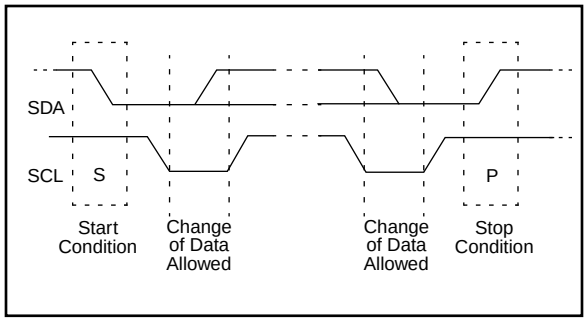


TABLE 11-2: I^2C BUS TERMINOLOGY

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 ensure that the transfer data does not get corrupted.
Synchronization	Procedure where the clock signals of two or more devices are synchronized.

11.3.1 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 11-3 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.

11.3.1.1 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 11-10). 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 11-3: DATA TRANSFER RECEIVED BYTE ACTIONS

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

FIGURE 11-21: 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;
        }
```

12.4 A/D Conversions

Example 12-2 show how to perform an A/D conversion. The RA pins are configured as analog inputs. The analog reference (V_{REF}) is the device V_{DD}. The A/D interrupt is enabled, and the A/D conversion clock is FRC. The conversion is performed on the RA0 pin (channel0).

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

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The ADRES register will NOT be updated with the partially completed A/D conversion sample. That is, the ADRES register will continue to contain the value of the last completed conversion (or the last value written to the ADRES register). After the A/D conversion is aborted, a $2T_{AD}$ wait is required before the next acquisition is started. After this $2T_{AD}$ wait, an acquisition is automatically started on the selected channel.

EXAMPLE 12-2: DOING AN A/D CONVERSION

```
     BCF     STATUS, RP1         ; Select Bank1
     BSF     STATUS, RP0         ;
     CLRF    ADCON1              ; Configure A/D inputs
     BSF     PIE1,   ADIE        ; Enable A/D interrupts
     BCF     STATUS, RP0         ; Select Bank0
     MOVLW   0xC1                ; RC Clock, A/D is on, Channel 0 is selected
     MOVWF   ADCON0              ;
     BCF     PIR1,   ADIF        ; Clear A/D interrupt flag bit
     BSF     INTCON, PEIE        ; Enable peripheral interrupts
     BSF     INTCON, GIE         ; Enable all interrupts
;
;  Ensure that the required acquisition time for the selected input channel has elapsed.
;  Then the conversion may be started.
;
     BSF     ADCON0, GO          ; Start A/D Conversion
       :                         ; The ADIF bit will be set and the GO/DONE bit
       :                         ;  is cleared upon completion of the A/D Conversion.
```

12.4.1 FASTER CONVERSION - LOWER RESOLUTION TRADE-OFF

Not all applications require a result with 8-bits of resolution, but may instead require a faster conversion time. The A/D module allows users to make the trade-off of conversion speed to resolution. Regardless of the resolution required, the acquisition time is the same. To speed up the conversion, the clock source of the A/D module may be switched so that the T_{AD} time violates the minimum specified time (see the applicable electrical specification). Once the T_{AD} time violates the minimum specified time, all the following A/D result bits are not valid (see A/D Conversion Timing in the Electrical Specifications section.) The clock sources may only be switched between the three oscillator versions (cannot be switched from/to RC). The equation to determine the time before the oscillator can be switched is as follows:

Conversion time = $2T_{AD} + N \bullet T_{AD} + (8 - N)(2T_{OSC})$
Where: N = number of bits of resolution required.

Since the T_{AD} is based from the device oscillator, the user must use some method (a timer, software loop, etc.) to determine when the A/D oscillator may be changed. Example 12-3 shows a comparison of time required for a conversion with 4-bits of resolution, versus the 8-bit resolution conversion. The example is for devices operating at 8 MHz (The A/D clock is programmed for $32T_{OSC}$), and assumes that immediately after $6T_{AD}$, the A/D clock is programmed for $2T_{OSC}$.

The $2T_{OSC}$ violates the minimum T_{AD} time, therefore the last 4-bits will not be converted to correct values.

EXAMPLE 12-3: 4-BIT vs. 8-BIT CONVERSION TIMES

	Freq. (MHz)	Resolution	
		4-bit	8-bit
T_{AD}	8	1.6 µs	1.6 µs
T_{OSC}	8	12.5 ns	125 ns
$2T_{AD} + N \bullet T_{AD} + (8 - N)(2T_{OSC})$	8	10.6 µs	16 µs

13.4.1 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.
Note 2:	The LMUX1:LMUX0 takes precedence over the LCDSE bit settings for pins RD7, RD6 and RD5.

EXAMPLE 13-1: 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 13-2: 1/3 MUX 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
```

FIGURE 13-12: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
bit7							bit0

R =Readable bit
W =Writable bit
U =Unimplemented bit, Read as ‘0’
-n =Value at POR reset

bit 7: **SE29**: Pin function select RD7/COM1/SEG31 - RD5/COM3/SEG29
1 = pins have LCD drive function
0 = pins have digital Input function
The LMUX1:LMUX0 setting takes precedence over the LCDSE register.

bit 6: **SE27**: Pin function select RG7/SEG28 and RE7/SEG27
1 = pins have LCD drive function
0 = pins have digital Input 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 I/O function

13.5 Voltage Generation

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

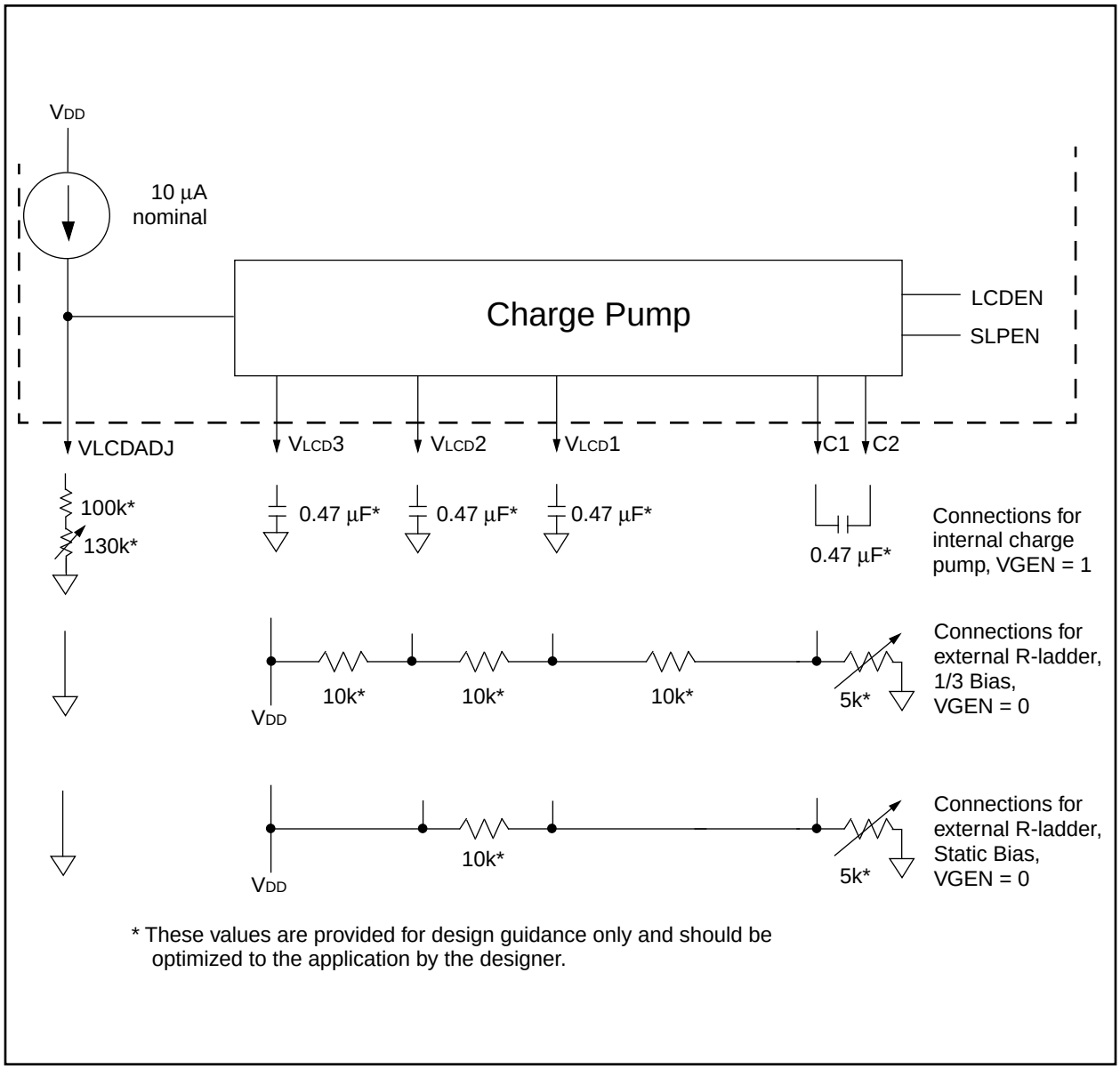
13.5.1 CHARGE PUMP

The LCD charge pump is shown in Figure 13-13. 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 $V_{LCD}1$ on the charge pump. The charge pump boosts $V_{LCD}1$ into $V_{LCD}2$ = 2*$V_{LCD}1$ and $V_{LCD}3$ = 3 * $V_{LCD}1$. When the charge pump is not operating, Vlcd3 will be internally tied to V_{DD}. See the Electrical Specifications section for charge pump capacitor and potentiometer values.

13.5.2 EXTERNAL R-LADDER

The LCD module can also use an external resistor ladder (R-Ladder) to generate the LCD voltages. Figure 13-13 shows external connections for static and 1/3 bias. The VGEN (LCDCON<4>) bit must be cleared to use an external R-Ladder.

FIGURE 13-13:CHARGE PUMP AND RESISTOR LADDER



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

TABLE 15-2: 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.

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

TABLE 16-1: DEVELOPMENT TOOLS FROM MICROCHIP

		PIC12C5XX	PIC14000	PIC16C5X	PIC16CXXX	PIC16C6X	PIC16C7XX	PIC16C8X	PIC16C9XX	PIC17C4X	PIC17C75X	24CXX 25CXX 93CXX	HCS200 HCS300 HCS301
Emulator Products	PICMASTER®/ PICMASTER-CE In-Circuit Emulator	✔	✔	✔	✔	✔	✔	✔	✔	✔	Available 3Q97		
	ICEPIC Low-Cost In-Circuit Emulator	✔		✔	✔	✔	✔	✔					
Software Tools	MPLAB™ Integrated Development Environment	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔		
	MPLAB™ C Compiler	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔		
	*fuzzy*TECH®-MP Explorer/Edition Fuzzy Logic Dev. Tool	✔	✔	✔	✔	✔	✔	✔	✔	✔			
	MP-DriveWay™ Applications Code Generator			✔	✔	✔	✔	✔		✔			
	Total Endurance™ Software Model											✔	
Programmers	PICSTART® Lite Ultra Low-Cost Dev. Kit			✔		✔	✔	✔					
	PICSTART® Plus Low-Cost Universal Dev. Kit	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔		
	PRO MATE® II Universal Programmer	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
	KEELOQ® Programmer												✔
Demo Boards	SEEVAL® Designers Kit											✔	
	PICDEM-1			✔	✔			✔		✔			
	PICDEM-2					✔	✔						
	PICDEM-3								✔				
	KEELOQ® Evaluation Kit												✔

17.3 DC Characteristics: PIC16C923/924-04 (Commercial, Industrial) PIC16C923/924-08 (Commercial, Industrial) PIC16LC923/924-04 (Commercial, Industrial)

DC CHARACTERISTICS							Standard Operating Conditions (unless otherwise stated) Operating temperature -40˚C ≤ TA ≤ +85˚C for industrial and 0˚C ≤ TA ≤ +70˚C for commercial Operating voltage VDD range as described in DC spec
Param No.	**Characteristic**	**Sym**	**Min**	**Typ †**	**Max**	**Units**	**Conditions**
	Input Low Voltage						
	I/O ports	VIL					
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	$\overline{\text{MCLR}}$, OSC1 (in RC mode)		VSS	-	0.2VDD	V	
D033	OSC1 (in XT, HS and LP)		VSS	-	0.3VDD	V	Note1
	Input High Voltage						
	I/O ports	VIH		-			
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	$\overline{\text{MCLR}}$		0.8VDD	-	VDD	V	
D042A	OSC1 (XT, HS and LP)		0.7VDD	-	VDD	V	Note1
D043	OSC1 (in RC mode)		0.9VDD	-	VDD	V	
D070	PORTB weak pull-up current	IPURB	50	250	400	μA	VDD = 5V, VPIN = VSS
	Input Leakage Current (Notes 2, 3)						
D060	I/O ports	IIL	-	-	±1.0	μA	Vss ≤ VPIN ≤ VDD, Pin at hi-Z
D061	$\overline{\text{MCLR}}$, RA4/T0CKI		-	-	±5	μA	Vss ≤ VPIN ≤ VDD
D063	OSC1		-	-	±5	μA	Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
	Output Low Voltage						
D080	I/O ports	VOL	-	-	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	I/O ports (Note 3)	VOH	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*	OSC2 pin	COSC2	-	-	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1.
D101*	All I/O pins and OSC2 (in RC)	CIO	-	-	50	pF	
D102*	SCL, SDA in I^2C mode	CB	-	-	400	pF	
D150*	Open -Drain High Voltage	VDD	-	-	14	V	RA4 pin

* These parameters are characterized but not tested.

† 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: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C9XX be driven with external clock in RC mode.

2: The leakage current on the $\overline{\text{MCLR}}$ 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 18-18:TYPICAL IDD vs. FREQUENCY (RC MODE @ 300 pF, 25°C)

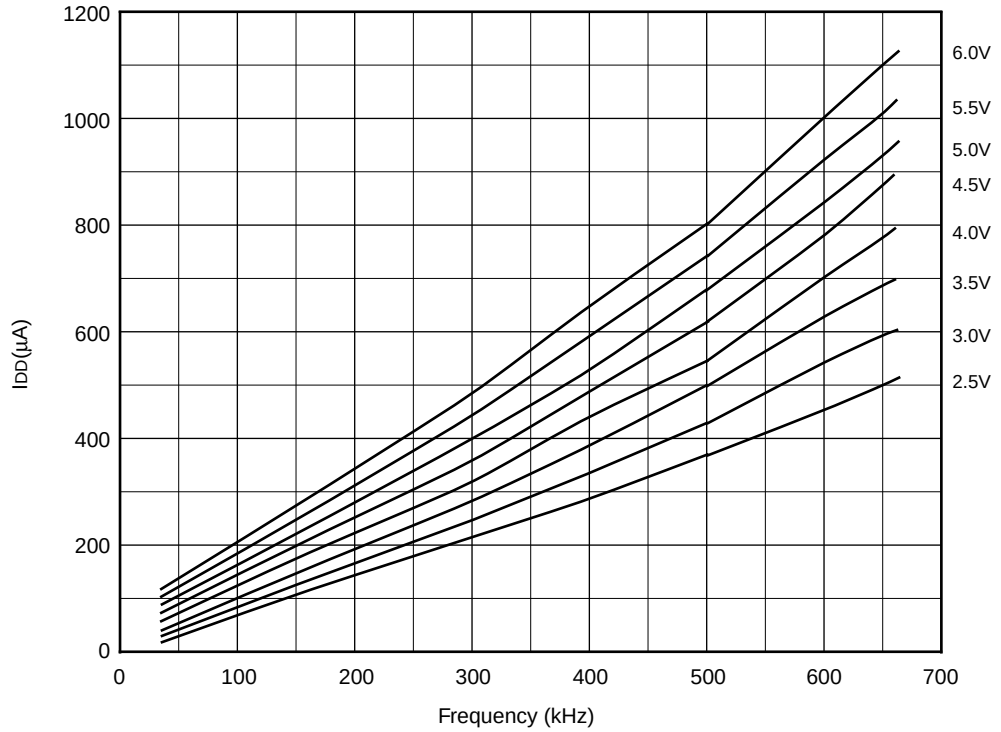
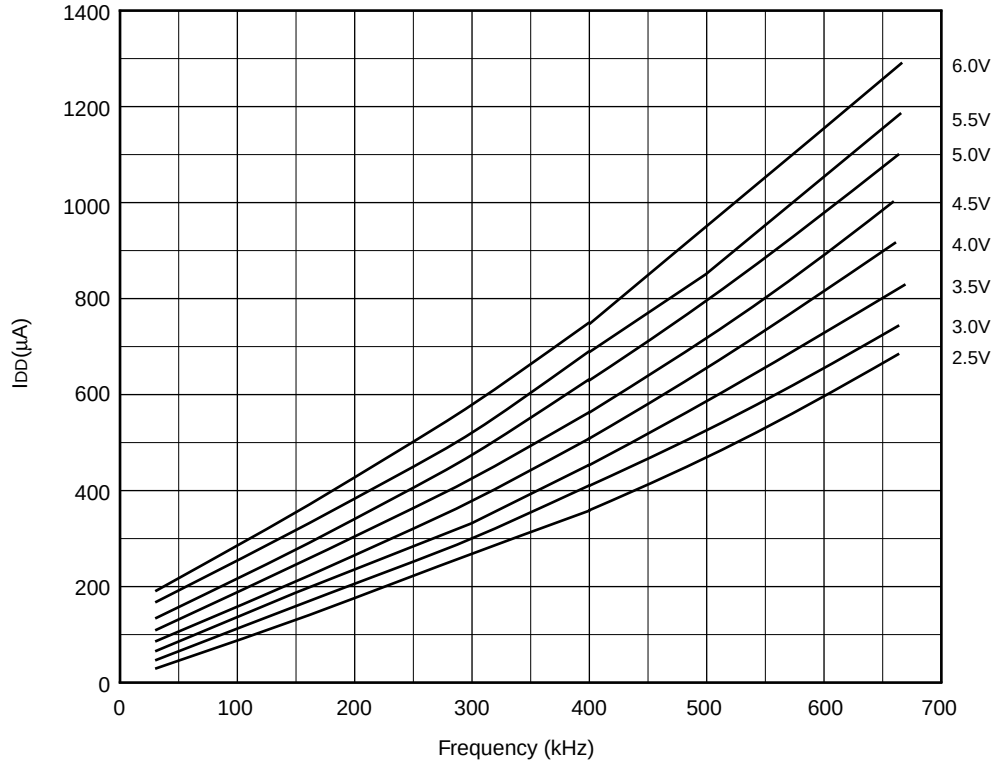
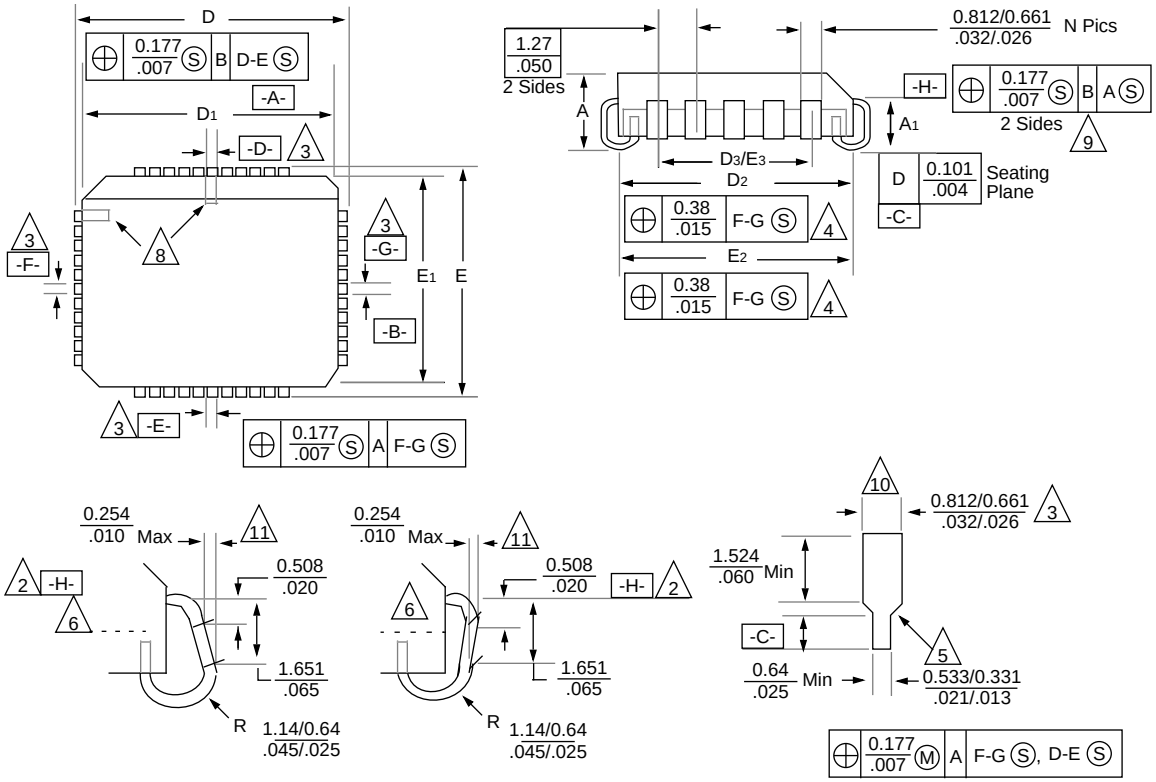


FIGURE 18-19:MAXIMUM IDD vs. FREQUENCY (RC MODE @ 300 pF, -40°C TO +85°C)



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

19.3 68-Lead Plastic Leaded Chip Carrier (Square)



Package Group: Plastic Leaded Chip Carrier (PLCC)						
	Millimeters			Inches		
Symbol	Min	Max	Notes	Min	Max	Notes
A	4.191	4.699		0.165	0.185	
A1	2.286	2.794		0.090	0.110	
D	25.019	25.273		0.985	0.995	
D1	24.130	24.334		0.950	0.958	
D2	22.860	23.622		0.900	0.930	
D3	20.320	-	**Reference**	0.800	-	**Reference**
E	25.019	25.273		0.985	0.995	
E1	24.130	24.334		0.950	0.958	
E2	22.860	23.622		0.900	0.930	
E3	20.320	-	**Reference**	0.800	-	**Reference**
N	68	-		68	-	
CP	-	0.102		-	0.004	
LT	0.203	0.254		0.008	0.010	

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Trademarks

Printed on recycled paper.

DNV Certification, Inc. USA
ANSI • RAB QMS ACCREDITED
DNV MSC The Netherlands Accredited by the RvA
DNV
ISO 9001 / QS-9000 REGISTERED FIRM

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.