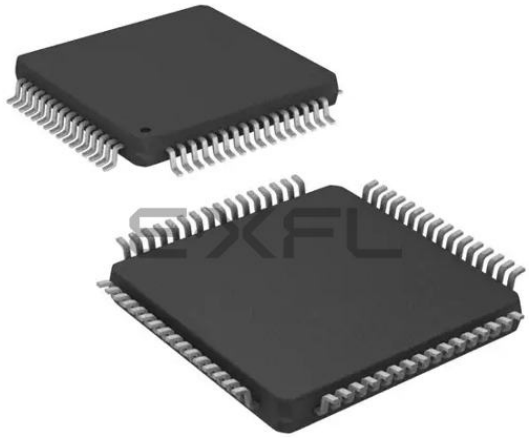


Welcome to E-XFL.COM

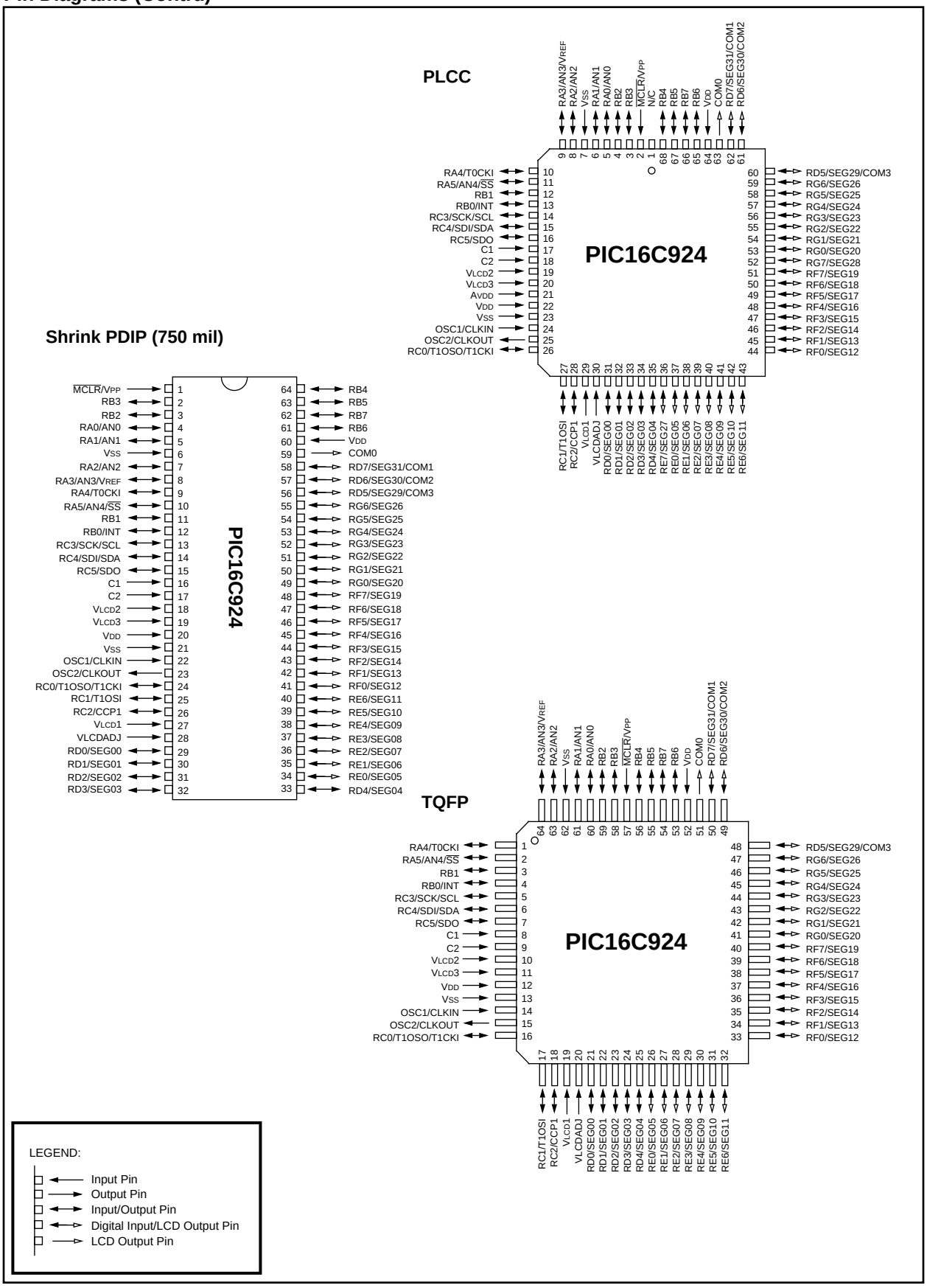
What is "Embedded - Microcontrollers"?

"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	Obsolete
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	A/D 5x8b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lc924t-04i-pt

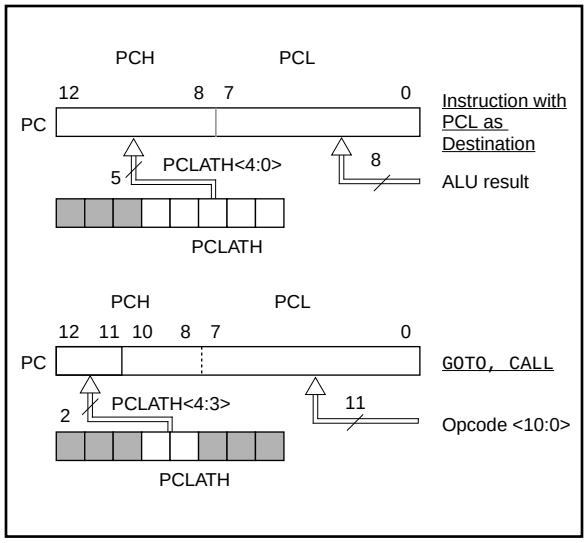
Pin Diagrams (Cont.'d)



4.3 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 4-9 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 4-9: LOADING OF PC IN DIFFERENT SITUATIONS



4.3.1 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).

4.3.2 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.

Note 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.

4.4 Program Memory Paging

PIC16C9XX 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 are not required for the return instructions (which POPs the address from the stack).

Note: The PIC16C9XX ignores paging bit PCLATH<4>, which is used to access program memory pages 2 and 3. The use of PCLATH<4> as a general purpose read/write bit is not recommended since this may affect upward compatibility with future products.

5.0 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.

5.1 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.

For the PIC16C924 only, 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).

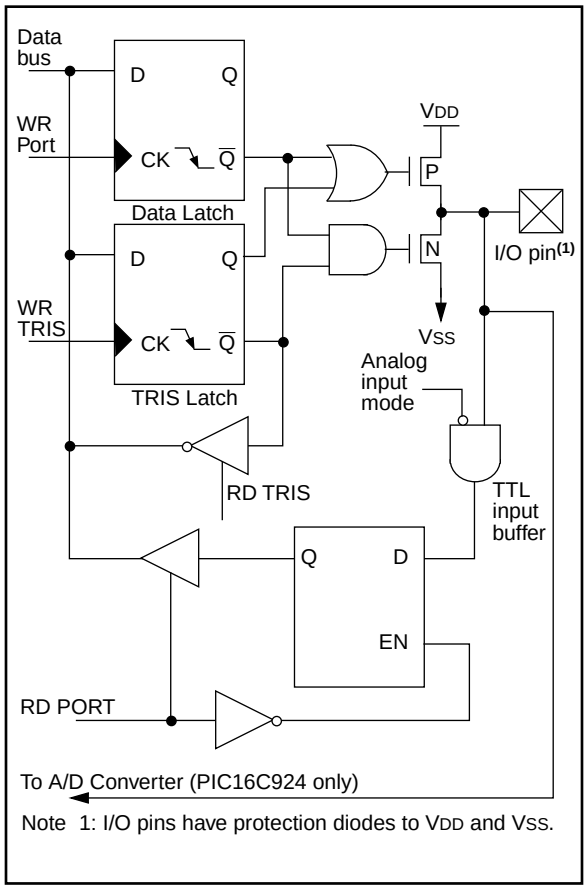
Note:	On a Power-on Reset, these pins are configured as analog inputs and read as '0'.

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 5-1: INITIALIZING PORTA

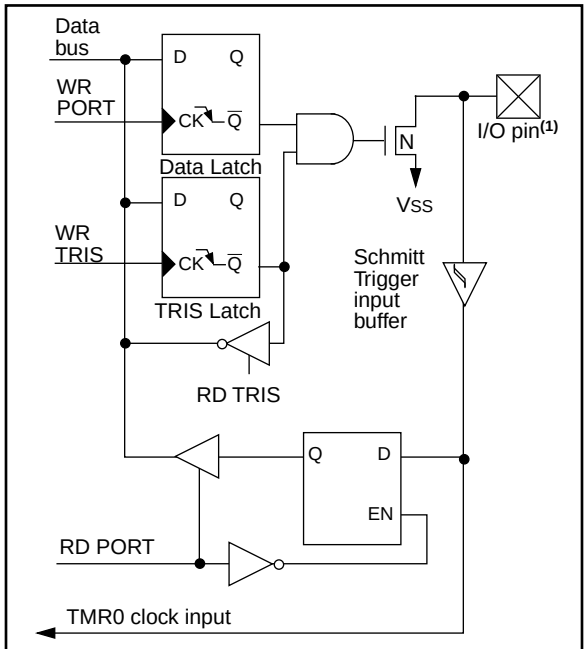
```
BCF     STATUS, RP0  ; Select Bank0
BCF     STATUS, RP1
CLRF    PORTA        ; Initialize PORTA
BSF     STATUS, RP0  ;
MOVLW   0xCF         ; Value used to
                     ; initialize data
                     ; direction
MOVWF   TRISA        ; Set RA<3:0> as inputs
                     ; RA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

FIGURE 5-1: BLOCK DIAGRAM OF PINS RA3:RA0 AND RA5



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

FIGURE 5-2: BLOCK DIAGRAM OF RA4/T0CKI PIN



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

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

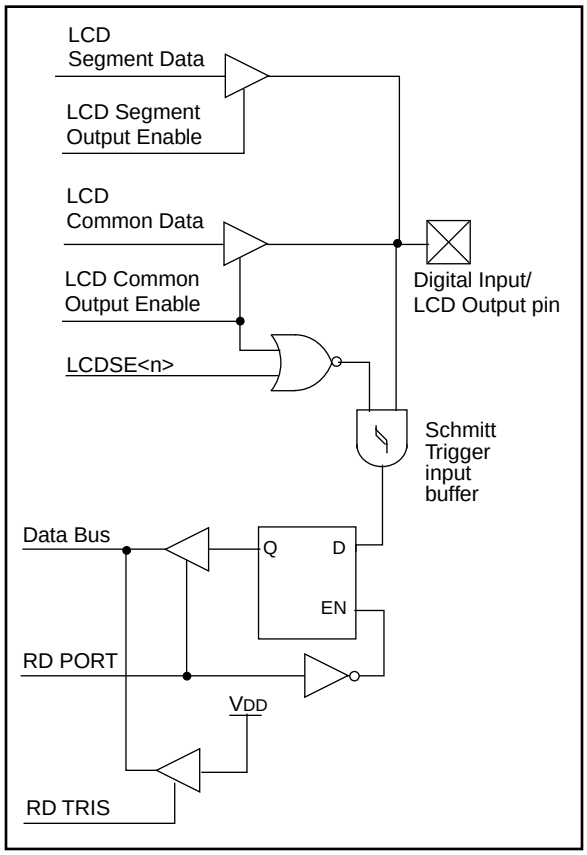


TABLE 5-7: PORTD FUNCTIONS

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 5-8: 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.

5.6 PORTF and TRISF Register

PORTF is an 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.

Note 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 5-6: INITIALIZING PORTF

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

FIGURE 5-9: PORTF BLOCK DIAGRAM

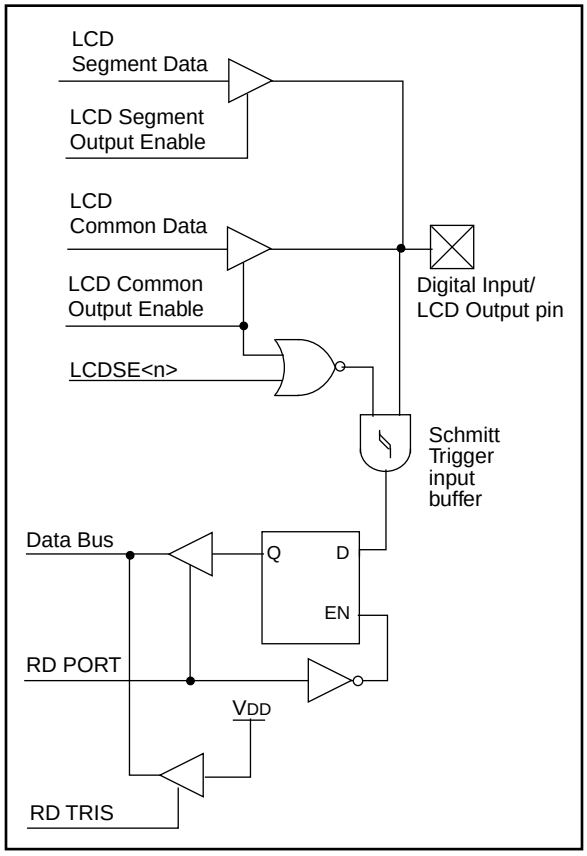


TABLE 5-11: PORTF FUNCTIONS

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 5-12: 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.

7.0 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 7-1 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 7-2 and Figure 7-3). The user can work around this by writing an adjusted value to the TMR0 register.

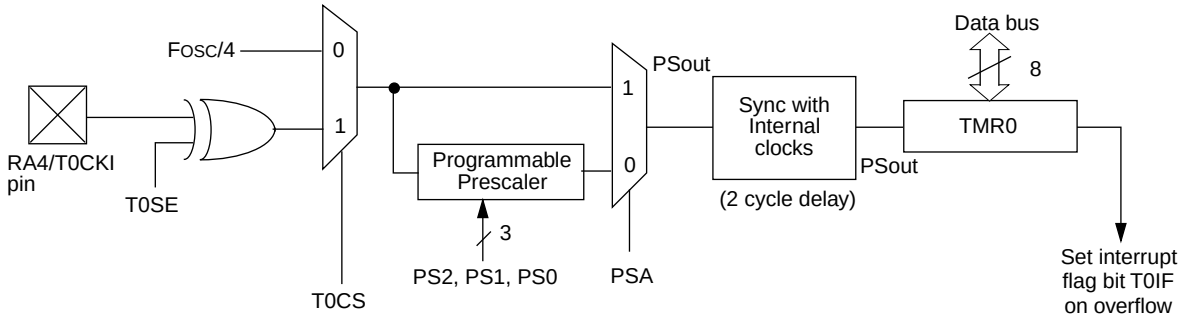
Counter mode is selected by setting bit T0CS (OPTION<5>). In counter mode Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (OPTION<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 7.2.

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 7.3 details the operation of the prescaler.

7.1 Timer0 Interrupt

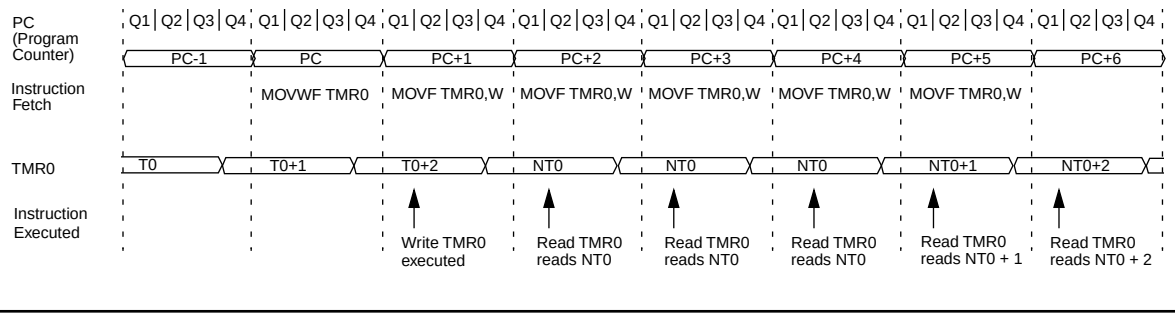
The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit T0IF (INTCON<2>). The interrupt can be masked by clearing bit T0IE (INTCON<5>). Bit T0IF must be cleared in software by the Timer0 module interrupt service routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from SLEEP since the timer is shut off during SLEEP. Figure 7-4 displays the Timer0 interrupt timing.

FIGURE 7-1: TIMER0 BLOCK DIAGRAM



Note 1: T0CS, T0SE, PSA, PS2:PS0 (OPTION<5:0>).
2: The prescaler is shared with Watchdog Timer (refer to Figure 7-6 for detailed block diagram).

FIGURE 7-2: TIMER0 TIMING: INTERNAL CLOCK/NO PRESCALE



8.0 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 10.0). Figure 8-1 shows the Timer1 control register.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI and RC0/T1OSO/T1CKI pins become inputs.

FIGURE 8-1: 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	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit7							bit0

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

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: **$\overline{\text{T1SYNC}}$**: 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.

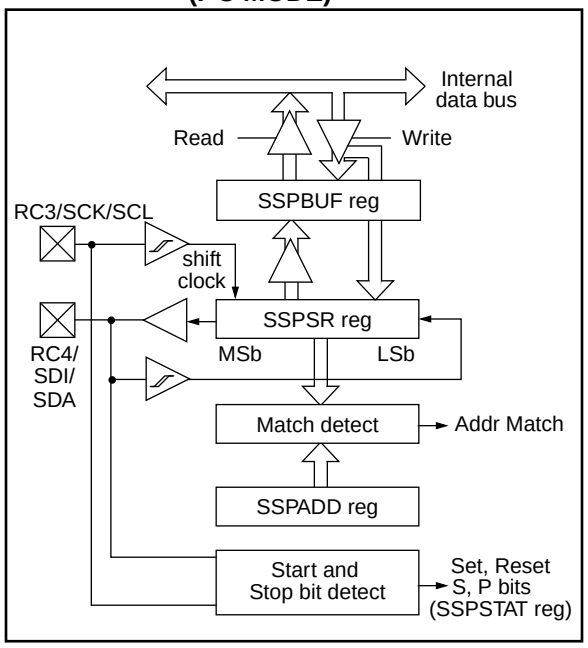
bit 1: **TMR1CS**: Timer1 Clock Source Select bit
1 = External clock from pin T1CKI (on the rising edge)
0 = Internal clock (Fosc/4)

bit 0: **TMR1ON**: Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

11.3 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 11-18: 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).

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 TAD time violates the minimum specified time (see the applicable electrical specification). Once the TAD 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 TAD 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 TAD 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

13.1 LCD Timing

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

13.1.1 TIMING CLOCK SOURCE SELECTION

The clock sources for the LCD timing generation are:

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

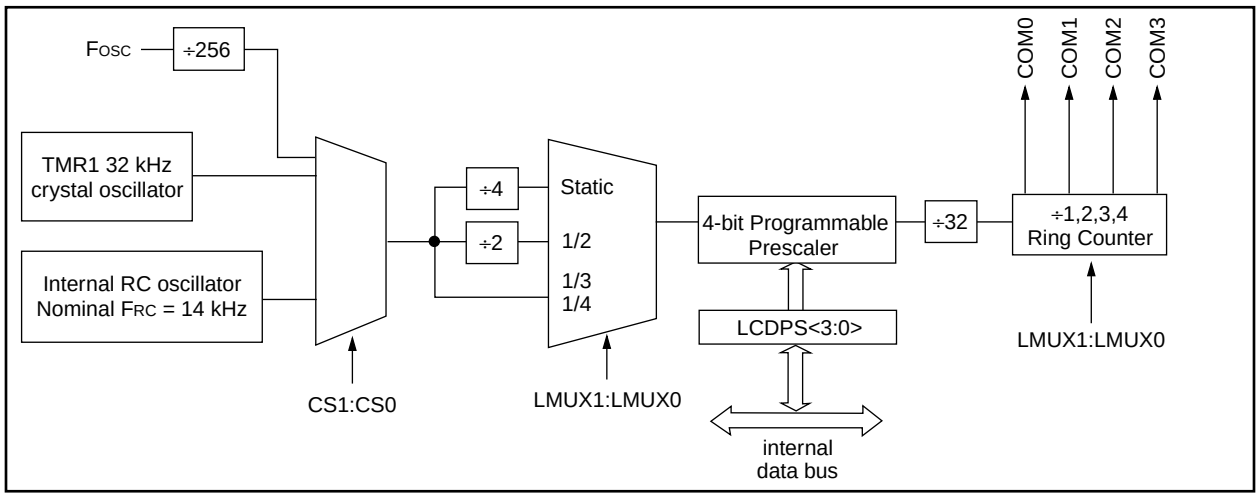
The first timing source is an internal RC oscillator which runs at a nominal frequency of 14 kHz. This oscillator provides a lower speed clock which may be used to continue running the LCD while the processor is in sleep. The RC oscillator will power-down when it is not selected or when the LCD module 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 Figure 13-1 for details of the register programming.

FIGURE 13-8: LCD CLOCK GENERATION



13.5 Voltage Generation

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

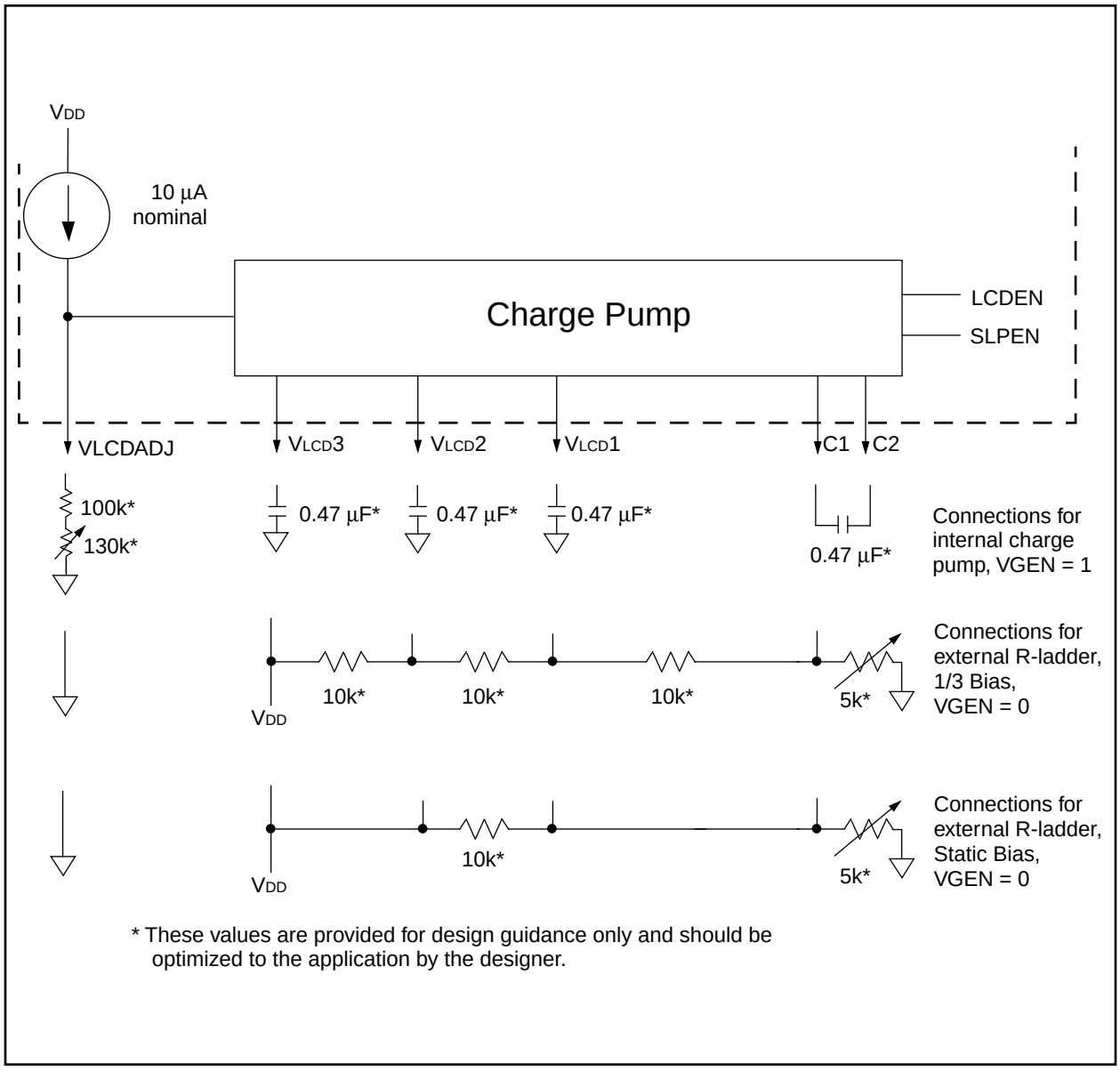
13.5.1 CHARGE PUMP

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.

14.4 Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)

14.4.1 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 $\overline{MCLR}$ 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, ...) 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*."

14.4.2 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.

14.4.3 OSCILLATOR START-UP TIMER (OST)

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

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

14.4.4 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 14-8, Figure 14-9, and Figure 14-10 depict time-out sequences on power-up.

Since the time-outs occur from the POR pulse, if $\overline{MCLR}$ is kept low long enough, the time-outs will expire. Then bringing $\overline{MCLR}$ high will begin execution immediately (Figure 14-9). This is useful for testing purposes or to synchronize more than one PIC16CXXX device operating in parallel.

Table 14-5 shows the reset conditions for some special function registers, while Table 14-6 shows the reset conditions for all the registers.

14.4.5 POWER CONTROL/STATUS REGISTER (PCON)

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

TABLE 14-3: TIME-OUT IN VARIOUS SITUATIONS

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

14.5 Interrupts

The PIC16C9XX family has up to 9 sources of interrupt:

Interrupt Sources	Applicable Devices	
External interrupt RB0/INT	923	924
TMR0 overflow interrupt	923	924
PORTB change interrupts (pins RB7:RB4)	923	924
A/D Interrupt	923	924
TMR1 overflow interrupt	923	924
TMR2 matches period interrupt	923	924
CCP1 interrupt	923	924
Synchronous serial port interrupt	923	924
LCD Module interrupt	923	924

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 un-masked 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 14-15). 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 or the GIE bit.

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

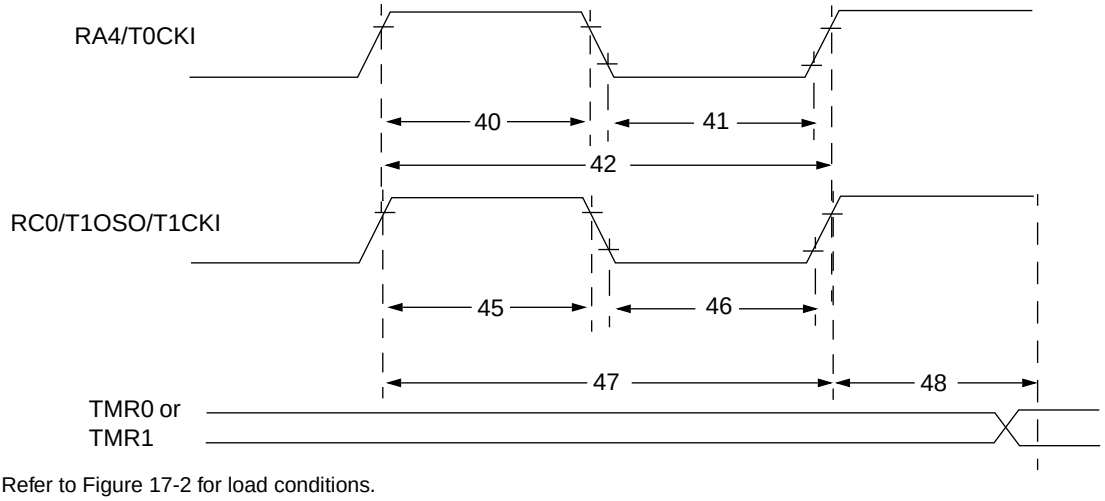
```
HERE    INCFSZ   CNT, 1
        GOTO     LOOP
CONTINUE •
         •
         •
```

Before Instruction
PC = address HERE
After Instruction
CNT = CNT + 1
if CNT= 0,
PC = address CONTINUE
if CNT≠ 0,
PC = address HERE +1

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												✓

FIGURE 17-6: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



Refer to Figure 17-2 for load conditions.

TABLE 17-7: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Sym	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 TCY + 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**923/924	15	—	—	ns	
				PIC16**LC**923/924	25	—	—	ns	
			Asynchronous	PIC16**C**923/924	30	—	—	ns	
				PIC16**LC**923/924	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**923/924	15	—	—	ns	
				PIC16**LC**923/924	25	—	—	ns	
			Asynchronous	PIC16**C**923/924	30	—	—	ns	
				PIC16**LC**923/924	50	—	—	ns	
47*	Tt1P	T1CKI input period	Synchronous	PIC16**C**923/924	Greater of: 30 OR TCY + 40 N	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16**LC**923/924	Greater of: 50 OR TCY + 40 N				N = prescale value (1, 2, 4, 8)
			Asynchronous	PIC16**C**923/924	60	—	—	ns	
				PIC16**LC**923/924	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	—	

* 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.

FIGURE 18-14:TYPICAL IDD vs. FREQUENCY (RC MODE @ 20 pF, 25°C)

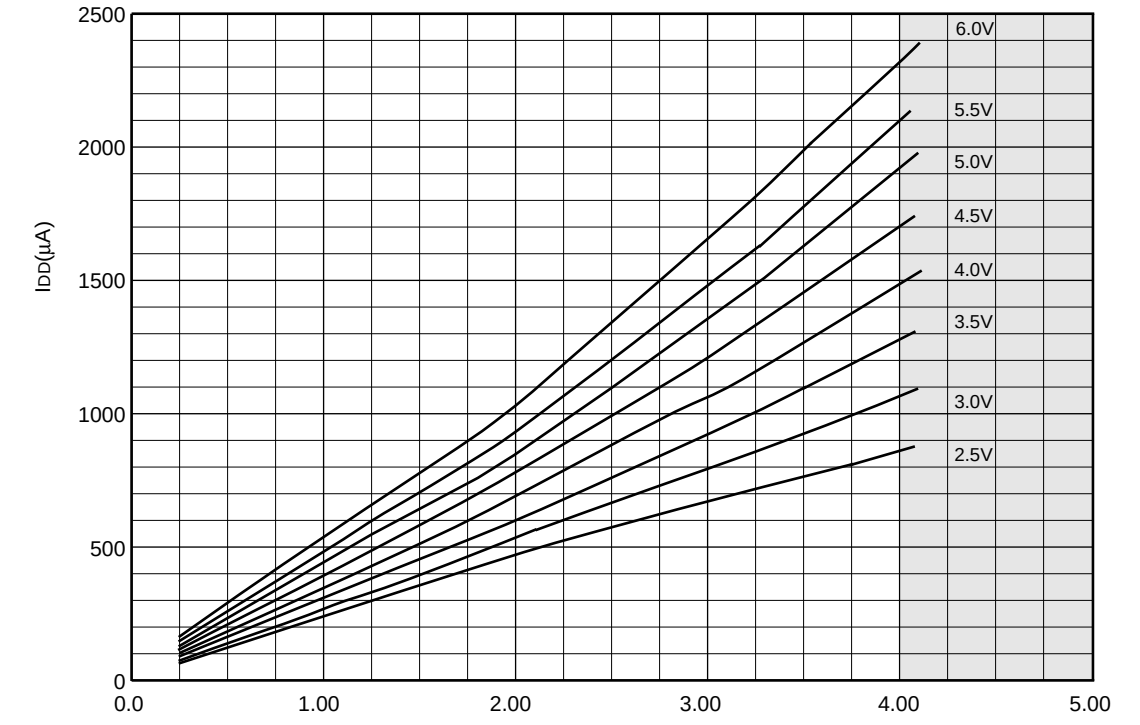
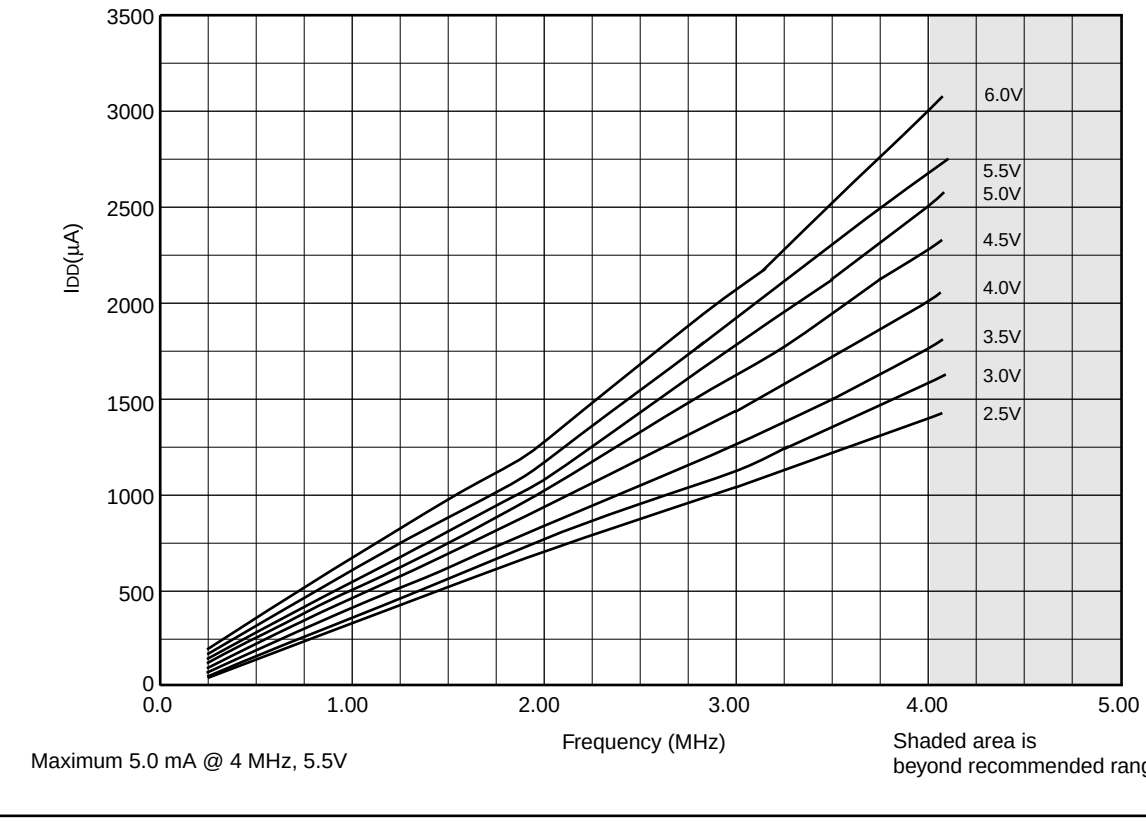


FIGURE 18-15:MAXIMUM IDD vs. FREQUENCY (RC MODE @ 20 pF, -40°C TO +85°C)



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

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 VDD = 5V.

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

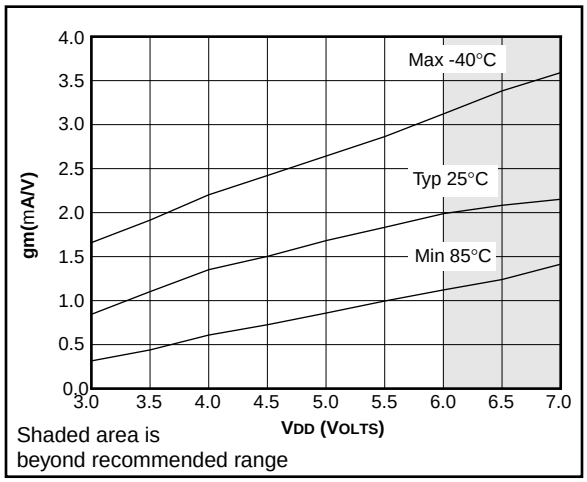


FIGURE 18-21:TRANSCONDUCTANCE(gm) OF LP OSCILLATOR vs. VDD

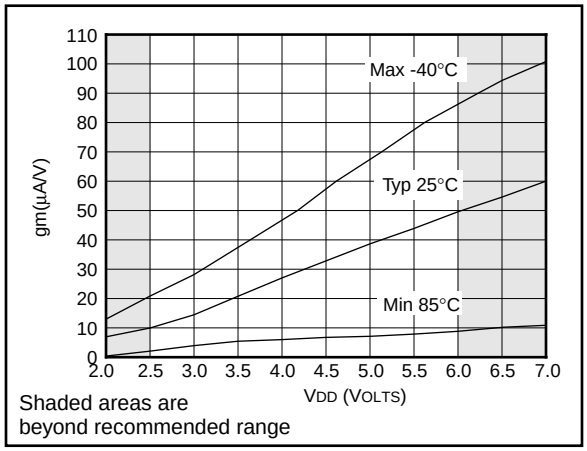
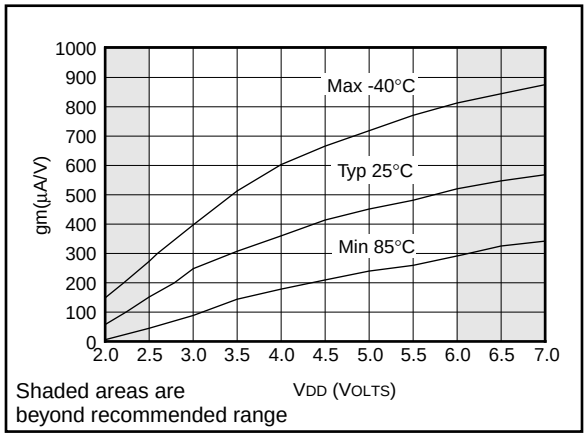


FIGURE 18-22:TRANSCONDUCTANCE(gm) OF XT OSCILLATOR vs. VDD



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

APPENDIX A:

The following are the list of modifications over the PIC16C5X microcontroller family:

1. Instruction word length is increased to 14-bits. This allows larger page sizes both in program memory (4K now as opposed to 512 before) and register file (192 bytes now versus 32 bytes before).
2. A PC high latch register (PCLATH) is added to handle program memory paging. Bits PA2, PA1, PA0 are removed from STATUS register.
3. Data memory paging is redefined slightly. STATUS register is modified.
4. Four new instructions have been added: RETURN, RETFIE, ADDLW, and SUBLW. Two instructions TRIS and OPTION are being phased out although they are kept for compatibility with PIC16C5X.
5. OPTION and TRIS registers are made addressable.
6. Interrupt capability is added. Interrupt vector is at 0004h.
7. Stack size is increased to 8 deep.
8. Reset vector is changed to 0000h.
9. Reset of all registers is revisited. Five different reset (and wake-up) types are recognized. Registers are reset differently.
10. Wake up from SLEEP through interrupt is added.
11. Two separate timers, Oscillator Start-up Timer (OST) and Power-up Timer (PWRT) are included for more reliable power-up. These timers are invoked selectively to avoid unnecessary delays on power-up and wake-up.
12. PORTB has weak pull-ups and interrupt on change feature.
13. T0CKI pin is also a port pin (RA4) now.
14. FSR is made a full eight bit register.
15. “In-circuit programming” is made possible. The user can program PIC16CXX devices using only five pins: V_{DD}, V_{SS}, $\overline{MCLR}/V_{PP}$, RB6 (clock) and RB7 (data in/out).
16. PCON status register is added with a Power-on Reset status bit ($\overline{POR}$).
17. Code protection scheme is enhanced such that portions of the program memory can be protected, while the remainder is unprotected.

APPENDIX B: COMPATIBILITY

To convert code written for PIC16C5X to PIC16CXX, the user should take the following steps:

1. Remove any program memory page select operations (PA2, PA1, PA0 bits) for CALL, GOTO.
2. Revisit any computed jump operations (write to PC or add to PC, etc.) to make sure page bits are set properly under the new scheme.
3. Eliminate any data memory page switching. Redefine data variables to reallocate them.
4. Verify all writes to STATUS, OPTION, and FSR registers since these have changed.
5. Change reset vector to 0000h.

INDEX