



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	Active
Core Processor	PIC
Core Size	8-Bit
Speed	20MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, POR, WDT
Number of I/O	13
Program Memory Size	896B (512 x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	96 x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 5.5V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	18-DIP (0.300", 7.62mm)
Supplier Device Package	18-PDIP
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic16c620a-20-p">https://www.e-xfl.com/product-detail/microchip-technology/pic16c620a-20-p</a>

# PIC16C62X

---

## Device Differences

Device	Voltage Range	Oscillator	Process Technology (Microns)
PIC16C620 <sup>(3)</sup>	2.5 - 6.0	See <b>Note 1</b>	0.9
PIC16C621 <sup>(3)</sup>	2.5 - 6.0	See <b>Note 1</b>	0.9
PIC16C622 <sup>(3)</sup>	2.5 - 6.0	See <b>Note 1</b>	0.9
PIC16C620A <sup>(4)</sup>	2.7 - 5.5	See <b>Note 1</b>	0.7
PIC16CR620A <sup>(2)</sup>	2.5 - 5.5	See <b>Note 1</b>	0.7
PIC16C621A <sup>(4)</sup>	2.7 - 5.5	See <b>Note 1</b>	0.7
PIC16C622A <sup>(4)</sup>	2.7 - 5.5	See <b>Note 1</b>	0.7

**Note 1:** If you change from this device to another device, please verify oscillator characteristics in your application.

**2:** For ROM parts, operation from 2.5V - 3.0V will require the PIC16LCR62X parts.

**3:** For OTP parts, operation from 2.5V - 3.0V will require the PIC16LC62X parts.

**4:** For OTP parts, operations from 2.7V - 3.0V will require the PIC16LC62XA parts.

## 3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16C62X family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16C62X uses a Harvard architecture, in which, program and data are accessed from separate memories using separate busses. This improves bandwidth over traditional von Neumann architecture, where program and data are fetched from the same memory. Separating program and data memory further allows instructions to be sized differently than 8-bit wide data word. Instruction opcodes are 14-bits wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions. Consequently, all instructions (35) execute in a single cycle (200 ns @ 20 MHz) except for program branches.

The PIC16C620(A) and PIC16CR620A address 512 x 14 on-chip program memory. The PIC16C621(A) addresses 1K x 14 program memory. The PIC16C622(A) addresses 2K x 14 program memory. All program memory is internal.

The PIC16C62X can directly or indirectly address its register files or data memory. All special function registers including the program counter are mapped in the data memory. The PIC16C62X has an orthogonal (symmetrical) instruction set that makes it possible to carry out any operation on any register using any Addressing mode. This symmetrical nature and lack of 'special optimal situations' make programming with the PIC16C62X simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16C62X devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

The W register is an 8-bit working register used for ALU operations. It is not an addressable register.

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, bit in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

A simplified block diagram is shown in Figure 3-1, with a description of the device pins in Table 3-1.

# PIC16C62X

---

## 4.2 Data Memory Organization

The data memory (Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7) is partitioned into two banks, which contain the General Purpose Registers and the Special Function Registers. Bank 0 is selected when the RP0 bit is cleared. Bank 1 is selected when the RP0 bit (STATUS <5>) is set. The Special Function Registers are located in the first 32 locations of each bank. Register locations 20-7Fh (Bank0) on the PIC16C620A/CR620A/621A and 20-7Fh (Bank0) and A0-BFh (Bank1) on the PIC16C622 and PIC16C622A are General Purpose Registers implemented as static RAM. Some Special Purpose Registers are mapped in Bank 1.

Addresses F0h-FFh of bank1 are implemented as common ram and mapped back to addresses 70h-7Fh in bank0 on the PIC16C620A/621A/622A/CR620A.

### 4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 80 x 8 in the PIC16C620/621, 96 x 8 in the PIC16C620A/621A/CR620A and 128 x 8 in the PIC16C622(A). Each is accessed either directly or indirectly through the File Select Register FSR (Section 4.4).

# PIC16C62X

**FIGURE 4-6: DATA MEMORY MAP FOR THE PIC16C620A/CR620A/621A**

File Address			File Address
00h	INDF <sup>(1)</sup>	INDF <sup>(1)</sup>	80h
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h			87h
08h			88h
09h			89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	PIR1	PIE1	8Ch
0Dh			8Dh
0Eh		PCON	8Eh
0Fh			8Fh
10h			90h
11h			91h
12h			92h
13h			93h
14h			94h
15h			95h
16h			96h
17h			97h
18h			98h
19h			99h
1Ah			9Ah
1Bh			9Bh
1Ch			9Ch
1Dh			9Dh
1Eh			9Eh
1Fh	CMCON	VRCON	9Fh
20h	General Purpose Register		A0h
6Fh			F0h
70h	General Purpose Register	Accesses 70h-7Fh	F0h
7Fh			FFh
Bank 0		Bank 1	

☐ Unimplemented data memory locations, read as '0'.  
**Note 1:** Not a physical register.

**FIGURE 4-7: DATA MEMORY MAP FOR THE PIC16C622A**

File Address			File Address
00h	INDF <sup>(1)</sup>	INDF <sup>(1)</sup>	80h
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h			87h
08h			88h
09h			89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	PIR1	PIE1	8Ch
0Dh			8Dh
0Eh		PCON	8Eh
0Fh			8Fh
10h			90h
11h			91h
12h			92h
13h			93h
14h			94h
15h			95h
16h			96h
17h			97h
18h			98h
19h			99h
1Ah			9Ah
1Bh			9Bh
1Ch			9Ch
1Dh			9Dh
1Eh			9Eh
1Fh	CMCON	VRCON	9Fh
20h	General Purpose Register	General Purpose Register	A0h
			BFh
			C0h
6Fh			F0h
70h	General Purpose Register	Accesses 70h-7Fh	F0h
7Fh			FFh
Bank 0		Bank 1	

☐ Unimplemented data memory locations, read as '0'.  
**Note 1:** Not a physical register.

## 4.2.2.4 PIE1 Register

This register contains the individual enable bit for the comparator interrupt.

### REGISTER 4-4: PIE1 REGISTER (ADDRESS 8CH)

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	CMIE	—	—	—	—	—	—
bit 7							bit 0

- bit 7 **Unimplemented:** Read as '0'
- bit 6 **CMIE:** Comparator Interrupt Enable bit  
1 = Enables the Comparator interrupt  
0 = Disables the Comparator interrupt
- bit 5-0 **Unimplemented:** Read as '0'

#### Legend:

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

## 4.2.2.5 PIR1 Register

This register contains the individual flag bit for the comparator interrupt.

**Note:** Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

### REGISTER 4-5: PIR1 REGISTER (ADDRESS 0CH)

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	CMIF	—	—	—	—	—	—
bit 7							bit 0

- bit 7 **Unimplemented:** Read as '0'
- bit 6 **CMIF:** Comparator Interrupt Flag bit  
1 = Comparator input has changed  
0 = Comparator input has not changed
- bit 5-0 **Unimplemented:** Read as '0'

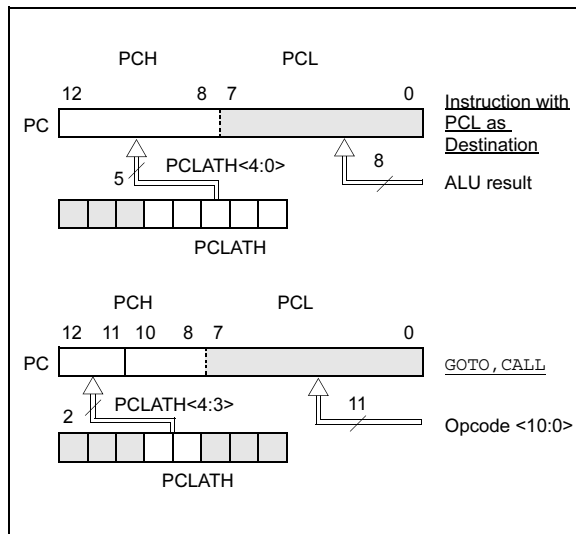
#### Legend:

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

## 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 high byte (PC<12:8>) is not directly readable or writable and comes from PCLATH. On any RESET, the PC is cleared. Figure 4-8 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 4-8: 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 PIC16C62X family has an 8-level deep x 13-bit wide hardware stack (Figure 4-2 and Figure 4-3). The stack space is not part of either program or data space and the stack pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

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

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

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

## 5.0 I/O PORTS

The PIC16C62X have two ports, PORTA and PORTB. Some pins for these I/O 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 Registers

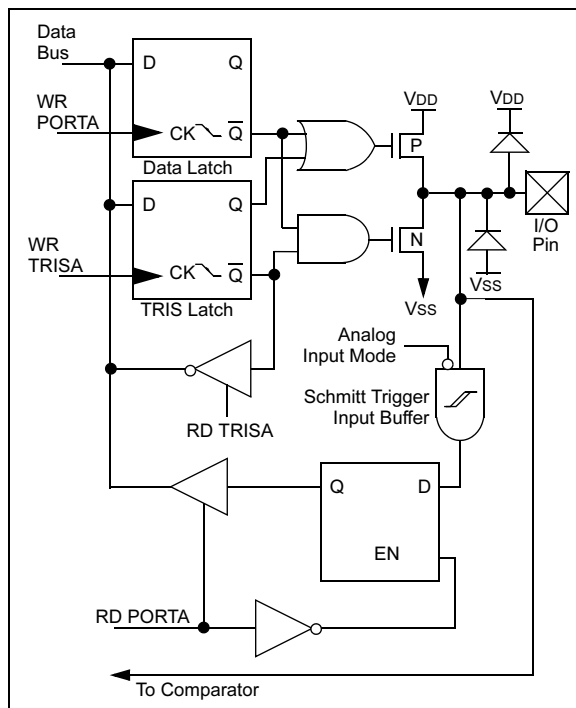
PORTA is a 5-bit wide latch. RA4 is a Schmitt Trigger input and an open drain output. Port RA4 is multiplexed with the T0CKI clock input. All other RA port pins have Schmitt Trigger input levels and full CMOS output drivers. All pins have data direction bits (TRIS registers), which can configure these pins as input or output.

A '1' in the TRISA register puts the corresponding output driver in a Hi-impedance mode. A '0' in the TRISA register puts the contents of the output latch on the selected pin(s).

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. So a write to a port implies that the port pins are first read, then this value is modified and written to the port data latch.

The PORTA pins are multiplexed with comparator and voltage reference functions. The operation of these pins are selected by control bits in the CMCON (comparator control register) register and the VRCON (voltage reference control register) register. When selected as a comparator input, these pins will read as '0's.

**FIGURE 5-1: BLOCK DIAGRAM OF RA1:RA0 PINS**



**Note:** On RESET, the TRISA register is set to all inputs. The digital inputs are disabled and the comparator inputs are forced to ground to reduce excess current consumption.

TRISA controls the direction of the RA pins, even when they are being used as comparator inputs. The user must make sure to keep the pins configured as inputs when using them as comparator inputs.

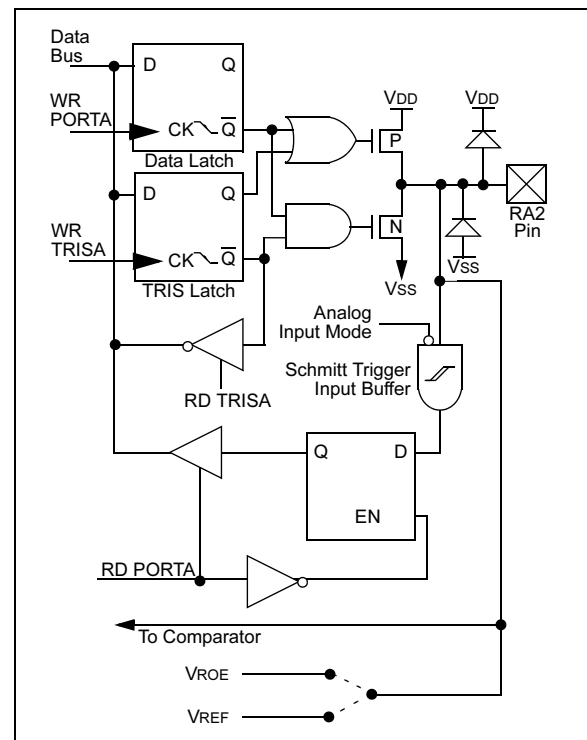
The RA2 pin will also function as the output for the voltage reference. When in this mode, the VREF pin is a very high impedance output and must be buffered prior to any external load. The user must configure TRISA<2> bit as an input and use high impedance loads.

In one of the Comparator modes defined by the CMCON register, pins RA3 and RA4 become outputs of the comparators. The TRISA<4:3> bits must be cleared to enable outputs to use this function.

**EXAMPLE 5-1: INITIALIZING PORTA**

```
CLRF    PORTA      ;Initialize PORTA by setting
                   ;output data latches
MOVLW   0X07       ;Turn comparators off and
MOVWF   CMCON      ;enable pins for I/O
                   ;functions
BSF     STATUS, RP0 ;Select Bank1
MOVLW   0x1F       ;Value used to initialize
                   ;data direction
MOVWF   TRISA      ;Set RA<4:0> as inputs
                   ;TRISA<7:5> are always
                   ;read as '0'.
```

**FIGURE 5-2: BLOCK DIAGRAM OF RA2 PIN**





# PIC16C62X

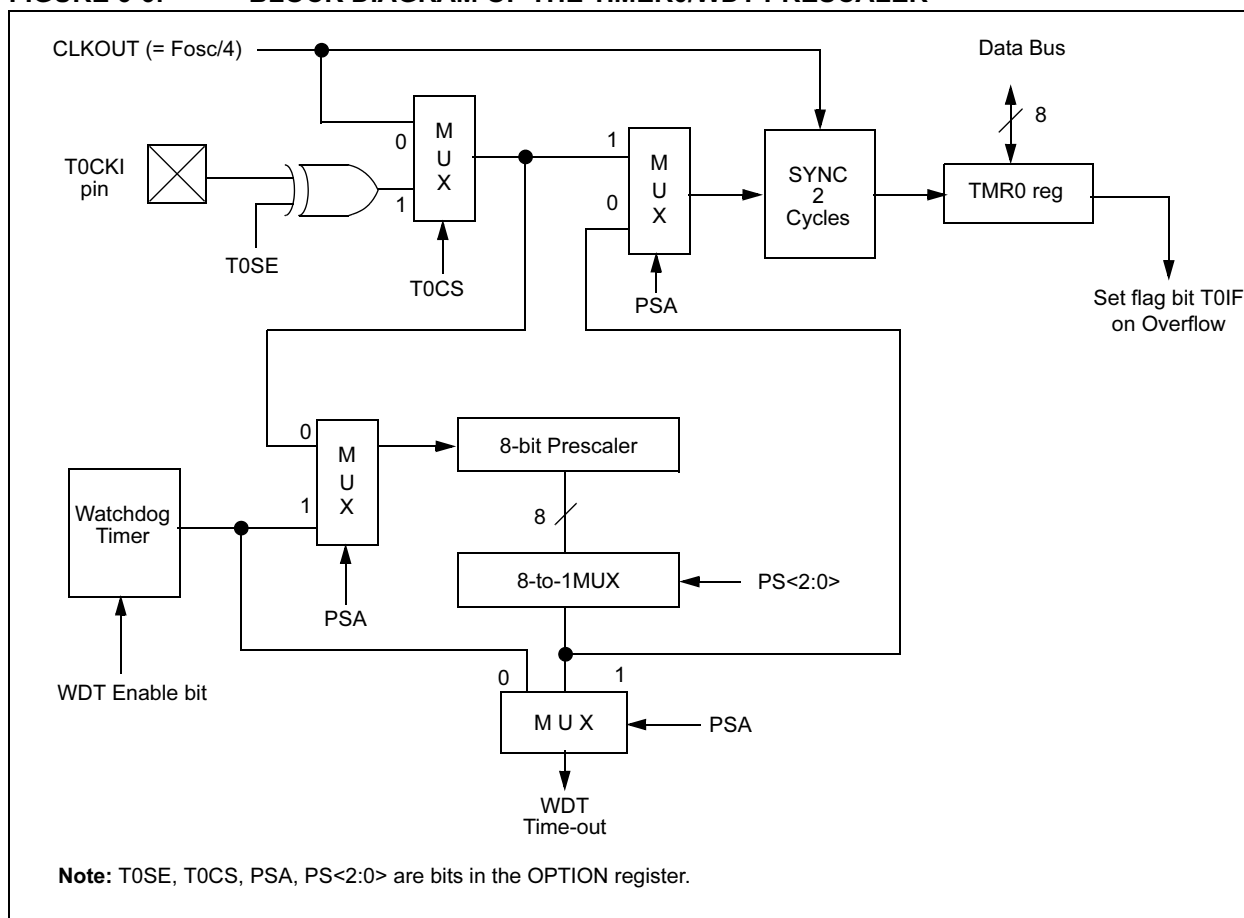
## 6.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer, respectively (Figure 6-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that there is only one prescaler available which is mutually exclusive between the Timer0 module and the Watchdog Timer. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer and vice-versa.

The PSA and PS<2:0> bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

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

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



# PIC16C62X

---

NOTES:

## 7.0 COMPARATOR MODULE

The comparator module contains two analog comparators. The inputs to the comparators are multiplexed with the RA0 through RA3 pins. The On-Chip Voltage Reference (Section 8.0) can also be an input to the comparators.

The CMCON register, shown in Register 7-1, controls the comparator input and output multiplexers. A block diagram of the comparator is shown in Figure 7-1.

### REGISTER 7-1: CMCON REGISTER (ADDRESS 1Fh)

R-0	R-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	—	—	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT**: Comparator 2 output

1 = C2 VIN+ > C2 VIN-

0 = C2 VIN+ < C2 VIN-

bit 6 **C1OUT**: Comparator 1 output

1 = C1 VIN+ > C1 VIN-

0 = C1 VIN+ < C1 VIN-

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

bit 3 **CIS**: Comparator Input Switch

When CM<2:0> = 001:

1 = C1 VIN- connects to RA3

0 = C1 VIN- connects to RA0

When CM<2:0> = 010:

1 = C1 VIN- connects to RA3

C2 VIN- connects to RA2

0 = C1 VIN- connects to RA0

C2 VIN- connects to RA1

bit 2-0 **CM<2:0>**: Comparator mode.

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

- n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

## 7.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that has occurred. The CMIF bit, PIR1<6>, is the comparator interrupt flag. The CMIF bit must be RESET by clearing '0'. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

The CMIE bit (PIE1<6>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

**Note:** If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR1<6>) interrupt flag may not get set.

The user, in the interrupt service routine, can clear the interrupt in the following manner:

- a) Any read or write of CMCON. This will end the mismatch condition.
- b) Clear flag bit CMIF.

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

## 7.7 Comparator Operation During SLEEP

When a comparator is active and the device is placed in SLEEP mode, the comparator remains active and the interrupt is functional if enabled. This interrupt will

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

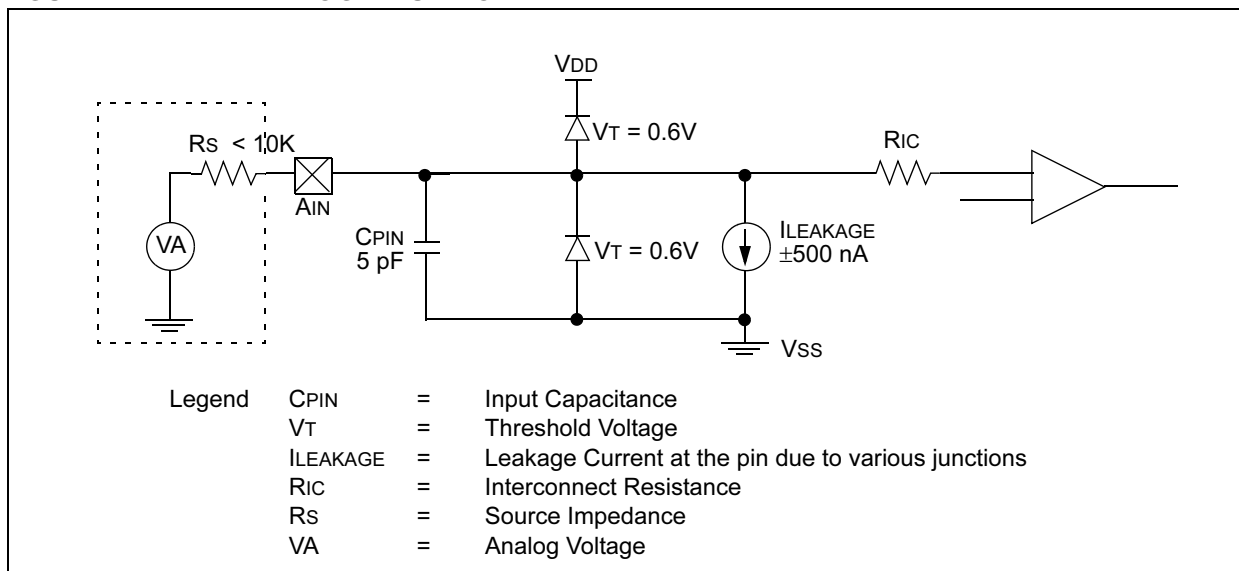
## 7.8 Effects of a RESET

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

## 7.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 7-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latchup may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

**FIGURE 7-4: ANALOG INPUT MODEL**



# PIC16C62X

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

### 9.4.1 POWER-ON RESET (POR)

The on-chip POR circuit holds the chip in RESET until VDD has reached a high enough level for proper operation. To take advantage of the POR, just tie the MCLR pin through a resistor to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A maximum rise time for VDD is required. See Electrical Specifications for details.

The POR circuit does not produce an internal RESET when VDD declines.

When the device starts normal operation (exits the RESET condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not met, the device must be held in RESET until the operating conditions are met.

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

### 9.4.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 72 ms (nominal) time-out on power-up only, from POR or Brown-out Reset. The Power-up Timer operates on an internal RC oscillator. The chip is kept in RESET as long as PWRT is active. The PWRT delay allows the VDD to rise to an acceptable level. A configuration bit, **PWRT**, can disable (if set) or enable (if cleared or programmed) the Power-up Timer. The Power-up Timer should always be enabled when Brown-out Reset is enabled.

The Power-up Time delay will vary from chip-to-chip and due to VDD, temperature and process variation. See DC parameters for details.

### 9.4.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-Up Timer (OST) provides a 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.

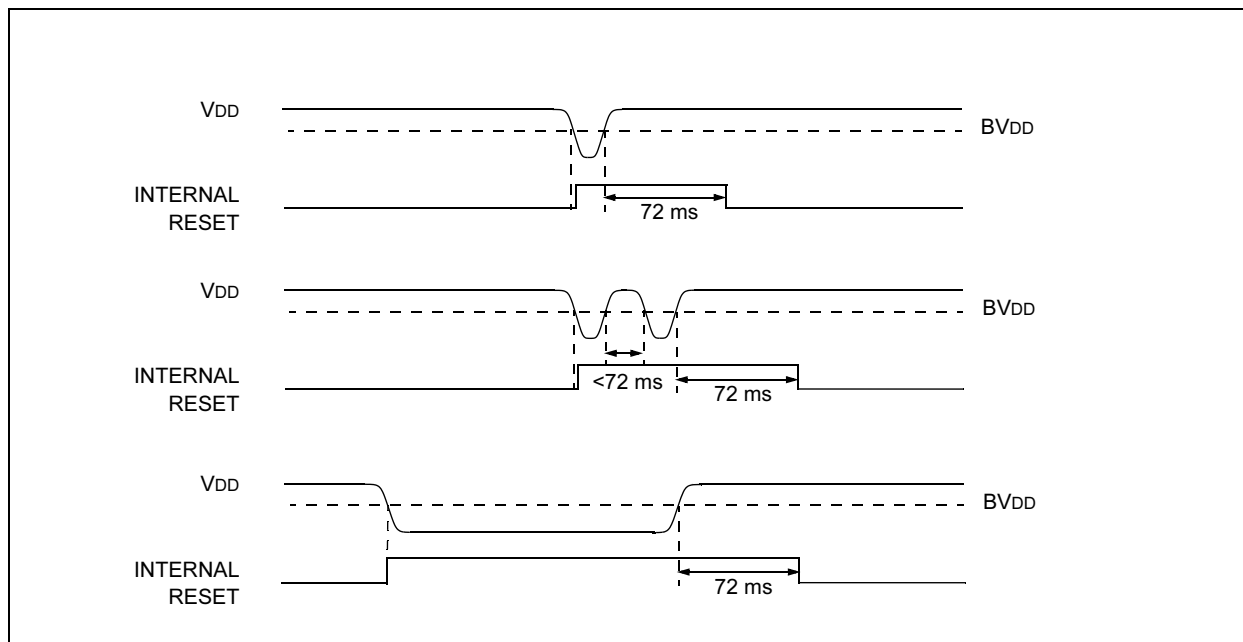
### 9.4.4 BROWN-OUT RESET (BOR)

The PIC16C62X members have on-chip Brown-out Reset circuitry. A configuration bit, **BODEN**, can disable (if clear/programmed) or enable (if set) the Brown-out Reset circuitry. If VDD falls below 4.0V refer to VBOR parameter D005 (VBOR) for greater than parameter (TBOR) in Table 12-5. The brown-out situation will RESET the chip. A RESET won't occur if VDD falls below 4.0V for less than parameter (TBOR).

On any RESET (Power-on, Brown-out, Watchdog, etc.) the chip will remain in RESET until VDD rises above BVDD. The Power-up Timer will now be invoked and will keep the chip in RESET an additional 72 ms.

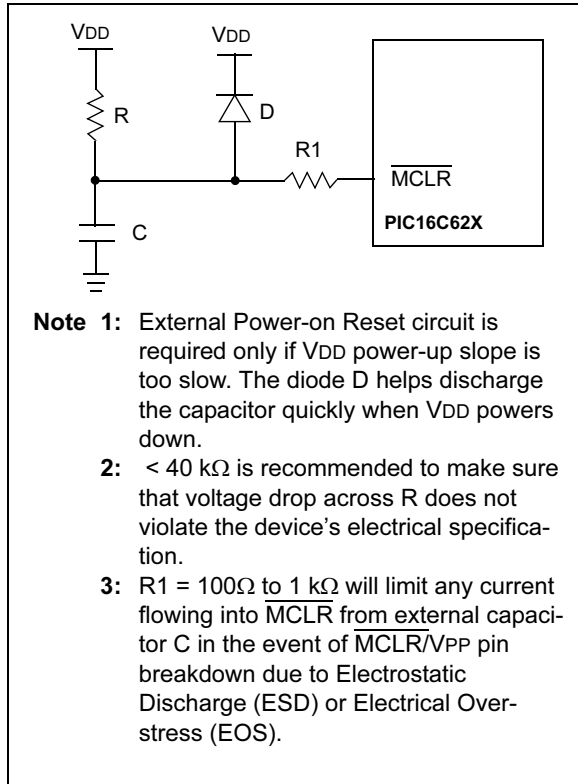
If VDD drops below BVDD while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be re-initialized. Once VDD rises above BVDD, the Power-Up Timer will execute a 72 ms RESET. The Power-up Timer should always be enabled when Brown-out Reset is enabled. Figure 9-7 shows typical Brown-out situations.

**FIGURE 9-7: BROWN-OUT SITUATIONS**

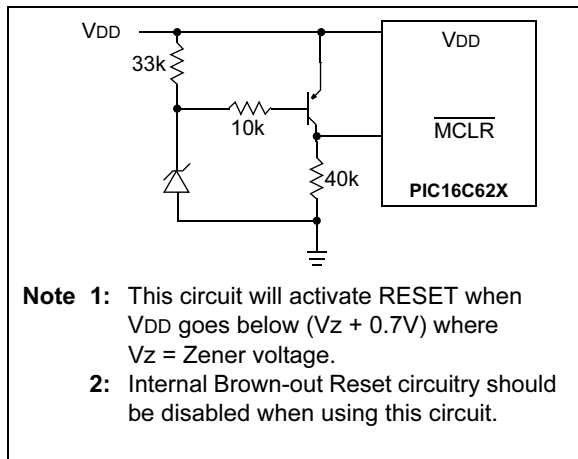


# PIC16C62X

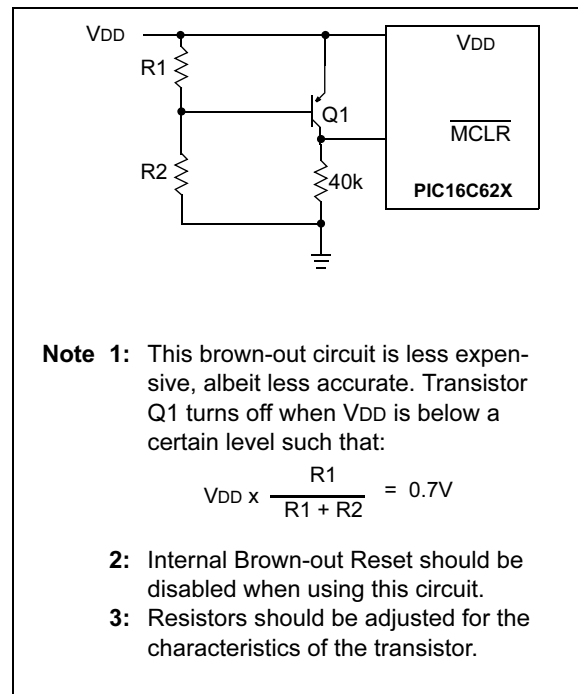
**FIGURE 9-11: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)**



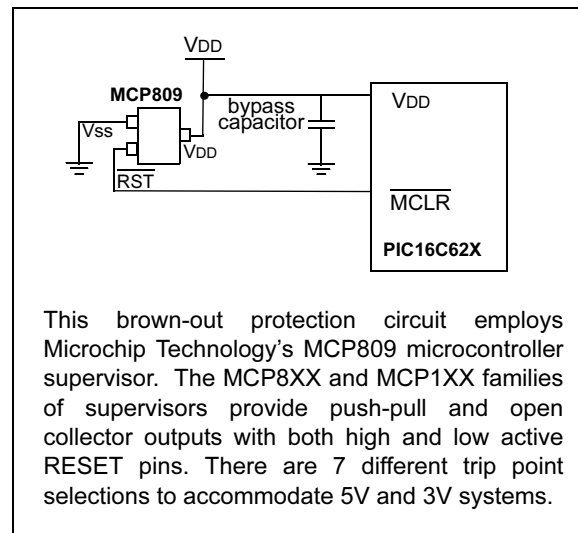
**FIGURE 9-12: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 1**



**FIGURE 9-13: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 2**



**FIGURE 9-14: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 3**



## 9.5 Interrupts

The PIC16C62X has 4 sources of interrupt:

- External interrupt RB0/INT
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB<7:4>)
- Comparator interrupt

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

A global interrupt enable bit, GIE (INTCON<7>) enables (if set) all un-masked interrupts or disables (if cleared) all interrupts. Individual interrupts can be disabled through their corresponding enable bits in INTCON register. GIE is cleared on RESET.

The “return from interrupt” instruction, RETFIE, exits interrupt routine, as well as sets the GIE bit, which re-enable RB0/INT interrupts.

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

The peripheral interrupt flag is contained in the special register PIR1. The corresponding interrupt enable bit is contained in special registers PIE1.

When an interrupt is responded to, the GIE is cleared to disable any further interrupt, the return address is pushed into the stack and the PC is loaded with 0004h.

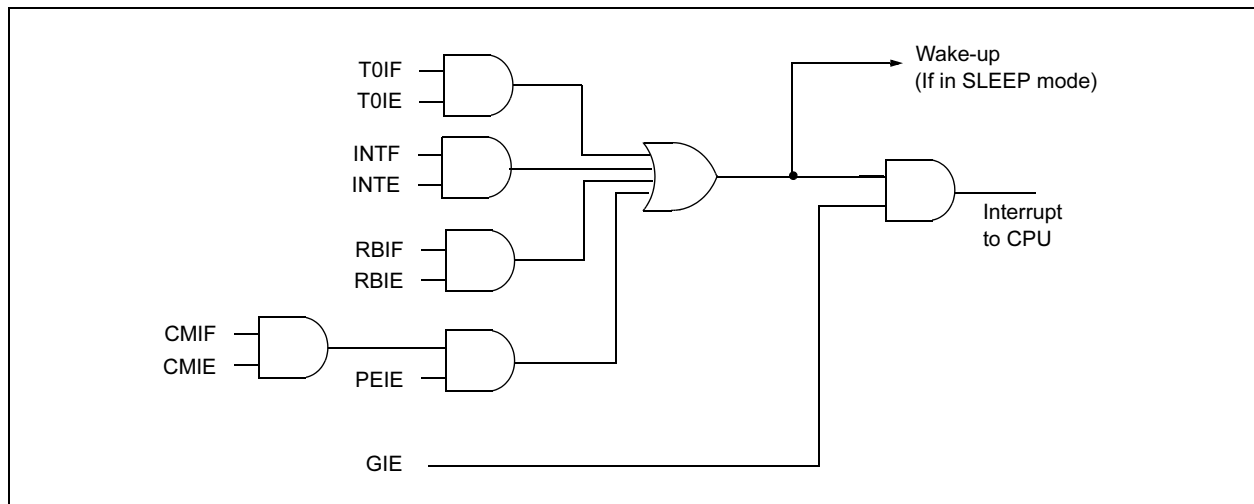
Once in the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid RB0/INT recursive interrupts.

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 9-16). The latency is the same for one or two cycle instructions. 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 multiple interrupt requests.

**Note 1:** Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

**2:** When an instruction that clears the GIE bit is executed, any interrupts that were pending for execution in the next cycle are ignored. The CPU will execute a NOP in the cycle immediately following the instruction which clears the GIE bit. The interrupts which were ignored are still pending to be serviced when the GIE bit is set again.

**FIGURE 9-15: INTERRUPT LOGIC**



BTFSS		Bit Test f, Skip if Set						
Syntax:	[ <i>label</i> ] BTFSS f,b							
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$							
Operation:	skip if (f<b>) = 1							
Status Affected:	None							
Encoding:	<table border="1"><tr><td>01</td><td>11bb</td><td>bfff</td><td>ffff</td></tr></table>				01	11bb	bfff	ffff
01	11bb	bfff	ffff					
Description:	<p>If bit 'b' in register 'f' is '1', then the next instruction is skipped.</p> <p>If bit 'b' is '1', then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a two-cycle instruction.</p>							
Words:	1							
Cycles:	1(2)							
Example	HERE	BTFSS	FLAG,1					
	FALSE	GOTO	PROCESS_CO					
	TRUE	•	DE					
		•						
		•						
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:	[ <i>label</i> ] CALL k								
Operands:	$0 \leq k \leq 2047$								
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<4:3>) → PC<12:11>								
Status Affected:	None								
Encoding:	<table border="1"><tr><td>10</td><td>0kkk</td><td>kkkk</td><td>kkkk</td></tr></table>					10	0kkk	kkkk	kkkk
10	0kkk	kkkk	kkkk						
Description:	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The eleven bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle instruction.								
Words:	1								
Cycles:	2								
Example	<pre>HERE    CALL   THER                                 E</pre> <p>Before Instruction PC = Address HERE</p> <p>After Instruction PC = Address THERE TOS = Address HERE+1</p>								

CLRF		Clear f						
Syntax:	[ <i>label</i> ] CLRF f							
Operands:	$0 \leq f \leq 127$							
Operation:	00h → (f) 1 → Z							
Status Affected:	Z							
Encoding:	<table border="1"><tr><td>00</td><td>0001</td><td>1fff</td><td>ffff</td></tr></table>				00	0001	1fff	ffff
00	0001	1fff	ffff					
Description:	The contents of register 'f' are cleared and the Z bit is set.							
Words:	1							
Cycles:	1							
Example	<pre>CLRF    FLAG_REG</pre> <p>Before Instruction</p> <p>FLAG_REG = 0x5A</p> <p>After Instruction</p> <p>FLAG_REG = 0x00</p> <p>Z = 1</p>							



# PIC16C62X

## CLRW Clear W

Syntax: [ *label* ] CLRW

Operands: None

Operation: 00h → (W)  
1 → Z

Status Affected: Z

Encoding: 

00	0001	0000	0011
----	------	------	------

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

Words: 1

Cycles: 1

Example CLRW

Before Instruction  
W = 0x5A  
After Instruction  
W = 0x00  
Z = 1

## CLRWDT Clear Watchdog Timer

Syntax: [ *label* ] CLRWDT

Operands: None

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

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

Encoding: 

00	0000	0110	0100
----	------	------	------

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. STATUS bits  $\overline{TO}$  and  $\overline{PD}$  are set.

Words: 1

Cycles: 1

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

Operation:  $(\bar{f}) \rightarrow (\text{dest})$

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

Example COMF REG1,0

Before Instruction  
REG1 = 0x13  
After Instruction  
REG1 = 0x13  
W = 0xEC

## DECF Decrement f

Syntax: [ *label* ] DECF f,d

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

Operation:  $(f) - 1 \rightarrow (\text{dest})$

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

Example DECF CNT, 1

Before Instruction  
CNT = 0x01  
Z = 0  
After Instruction  
CNT = 0x00  
Z = 1

SWAPF		Swap Nibbles in f							
Syntax:	[ <i>label</i> ] SWAPF f,d								
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$								
Operation:	(f<3:0>) → (dest<7:4>), (f<7:4>) → (dest<3:0>)								
Status Affected:	None								
Encoding:	<table><tr><td>00</td><td>1110</td><td>dfff</td><td>ffff</td></tr></table>					00	1110	dfff	ffff
00	1110	dfff	ffff						
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.								
Words:	1								
Cycles:	1								
Example	SWAPF REG, 0								
	Before Instruction								
	REG1 = 0xA5								
	After Instruction								
	REG1 = 0xA5								
	W = 0x5A								

TRIS	Load TRIS Register				
Syntax:	[ <i>label</i> ] TRIS f				
Operands:	$5 \leq f \leq 7$				
Operation:	(W) → TRIS register f;				
Status Affected:	None				
Encoding:	<table><tr><td>00</td><td>0000</td><td>0110</td><td>0fff</td></tr></table>	00	0000	0110	0fff
00	0000	0110	0fff		
Description:	The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.				
Words:	1				
Cycles:	1				
Example	<div><b>To maintain upward compatibility with future PICmicro<sup>®</sup> products, do not use this instruction.</b></div>				

XORLW		Exclusive OR Literal with W			
Syntax:	[ <i>label</i> XORLW   k 				

XORWF

Exclusive OR W with f

Syntax:

[ *label* ] XORWF f,d

Operands:

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

Operation:

(W) .XOR. (f) → (dest)

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

Example

XORWF REG 1

Before Instruction

REG = 0xAF

W = 0xB5

After Instruction

REG = 0x1A

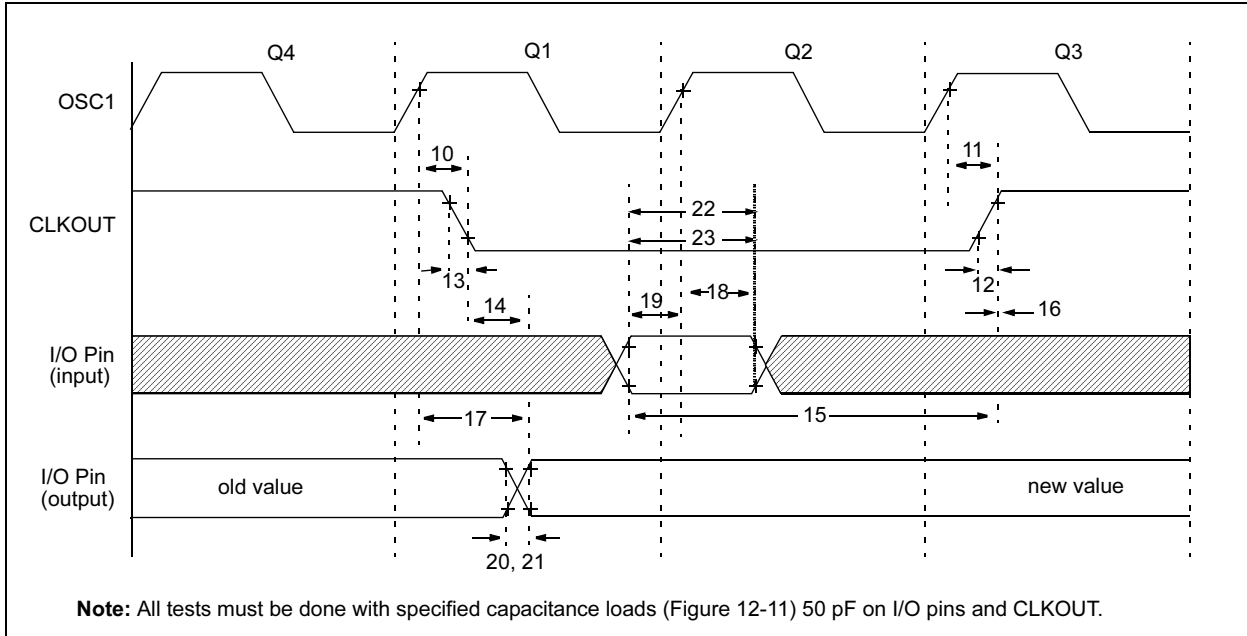
W = 0xB5

# PIC16C62X

---

NOTES:

**FIGURE 12-13: CLKOUT AND I/O TIMING**



## ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® or Microsoft® Internet Explorer. Files are also available for FTP download from our FTP site.

### Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

**[www.microchip.com](http://www.microchip.com)**

The file transfer site is available by using an FTP service to connect to:

**<ftp://ftp.microchip.com>**

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

## SYSTEMS INFORMATION AND UPGRADE HOT LINE

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits. The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

092002