



# PIC16F62X

## FLASH-Based 8-Bit CMOS Microcontrollers

### Devices included in this data sheet:

- PIC16F627
- PIC16F628

Referred to collectively as PIC16F62X .

### High Performance RISC CPU:

- Only 35 instructions to learn
- All single-cycle instructions (200 ns), except for program branches which are two-cycle
- Operating speed:
  - DC - 20 MHz clock input
  - DC - 200 ns instruction cycle

Device	Memory		
	FLASH Program	RAM Data	EEPROM Data
PIC16F627	1024 x 14	224 x 8	128 x 8
PIC16F628	2048 x 14	224 x 8	128 x 8

- Interrupt capability
- 16 special function hardware registers
- 8-level deep hardware stack
- Direct, Indirect and Relative addressing modes

### Peripheral Features:

- 15 I/O pins with individual direction control
- High current sink/source for direct LED drive
- Analog comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (VREF) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler
- Timer1: 16-bit timer/counter with external crystal/clock capability
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Capture, Compare, PWM (CCP) module
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Universal Synchronous/Asynchronous Receiver/Transmitter USART/SCI
- 16 Bytes of common RAM

### Special Microcontroller Features:

- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Detect (BOD)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Multiplexed  $\overline{\text{MCLR}}$ -pin
- Programmable weak pull-ups on PORTB
- Programmable code protection
- Low voltage programming
- Power saving SLEEP mode
- Selectable oscillator options
  - FLASH configuration bits for oscillator options
  - ER (External Resistor) oscillator
    - Reduced part count
  - Dual speed INTRC
    - Lower current consumption
  - EC External Clock input
  - XT oscillator mode
  - HS oscillator mode
  - LP oscillator mode
- Serial in-circuit programming (via two pins)
- Four user programmable ID locations

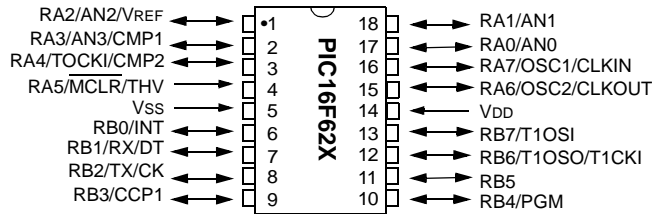
### CMOS Technology:

- Low-power, high-speed CMOS FLASH technology
- Fully static design
- Wide operating voltage range
  - PIC16F627 - 3.0V to 5.5V
  - PIC16F628 - 3.0V to 5.5V
  - PIC16LF627 - 2.0V to 5.5V
  - PIC16LF628 - 2.0V to 5.5V
- Commercial, industrial and extended temperature range
- Low power consumption
  - < 2.0 mA @ 5.0V, 4.0 MHz
  - 15  $\mu\text{A}$  typical @ 3.0V, 32 kHz
  - < 1.0  $\mu\text{A}$  typical standby current @ 3.0V

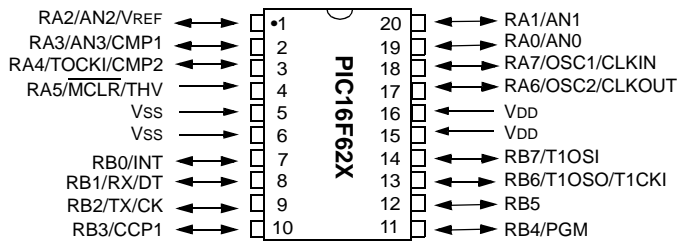
# PIC16F62X

## Pin Diagrams

### PDIP, SOIC



### SSOP



## Device Differences

Device	Voltage Range	Oscillator	Process Technology (Microns)
PIC16F627	3.0 - 5.5	See Note 1	0.7
PIC16F628	3.0 - 5.5	See Note 1	0.7
PIC16LF627	2.0 - 5.5	See Note 1	0.7
PIC16LF628	2.0 - 5.5	See Note 1	0.7

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

## Table of Contents

1.0	General Description.....	5
2.0	PIC16F62X Device Varieties.....	7
3.0	Architectural Overview.....	9
4.0	Memory Organization.....	13
5.0	I/O Ports.....	27
6.0	Timer0 Module.....	45
7.0	Timer1 Module.....	50
8.0	Timer2 Module.....	54
9.0	Comparator Module.....	57
10.0	Capture/Compare/PWM (CCP) Module.....	63
11.0	Voltage Reference Module.....	69
12.0	Universal Synchronous Asynchronous Receiver Transmitter (USART).....	71
13.0	Data EEPROM Memory.....	91
14.0	Special Features of the CPU.....	95
15.0	Instruction Set Summary.....	113
16.0	Development Support.....	125
17.0	Electrical Specifications.....	131
18.0	Device Characterization Information.....	145
19.0	Packaging Information.....	147
	Index.....	151
	On-Line Support.....	155
	Reader Response.....	156
	PIC16F62X Product Identification System.....	157

### *To Our Valued Customers*

#### **Most Current Data Sheet**

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

<http://www.microchip.com>

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number. e.g., DS30000A is version A of document DS30000.

#### **New Customer Notification System**

Register on our web site ([www.microchip.com/cn](http://www.microchip.com/cn)) to receive the most current information on our products.

#### **Errata**

An errata sheet may exist for current devices, describing minor operational differences (from the data sheet) and recommended workarounds. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; <http://www.microchip.com>
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (602) 786-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

#### **Corrections to this Data Sheet**

We constantly strive to improve the quality of all our products and documentation. We have spent a great deal of time to ensure that this document is correct. However, we realize that we may have missed a few things. If you find any information that is missing or appears in error, please:

- Fill out and mail in the reader response form in the back of this data sheet.
- E-mail us at [webmaster@microchip.com](mailto:webmaster@microchip.com).

We appreciate your assistance in making this a better document.

# PIC16F62X

---

NOTES:

## 1.0 GENERAL DESCRIPTION

The PIC16F62X are 18-Pin FLASH-based members of the versatile PIC16CXX family of low-cost, high-performance, CMOS, fully-static, 8-bit microcontrollers.

All PICmicro<sup>®</sup> microcontrollers employ an advanced RISC architecture. The PIC16F62X have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with the separate 8-bit wide data. The two-stage instruction pipeline allows all instructions to execute in a single-cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance.

PIC16F62X microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

PIC16F62X devices have special features to reduce external components, thus reducing system cost, enhancing system reliability and reducing power consumption. There are eight oscillator configurations, of which the single pin ER oscillator provides a low-cost solution. The LP oscillator minimizes power consumption, XT is a standard crystal, INTRC is a self-contained internal oscillator and the HS is for High Speed crystals. The SLEEP (power-down) mode offers power savings. The user can wake up the chip from SLEEP through several external and internal interrupts and reset.

A highly reliable Watchdog Timer with its own on-chip RC oscillator provides protection against software lock-up.

Table 1-1 shows the features of the PIC16F62X mid-range microcontroller families.

A simplified block diagram of the PIC16F62X is shown in Figure 3-1.

The PIC16F62X series fits in applications ranging from battery chargers to low-power remote sensors. The FLASH technology makes customization of application programs (detection levels, pulse generation, timers, etc.) extremely fast and convenient. The small footprint packages make this microcontroller series ideal for all applications with space limitations. Low-cost, low-power, high-performance, ease of use and I/O flexibility make the PIC16F62X very versatile.

## 1.1 Development Support

The PIC16F62X family is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a low-cost development programmer and a full-featured programmer. A Third Party "C" compiler support tool is also available.

# PIC16F62X

**TABLE 1-1: PIC16F62X FAMILY OF DEVICES**

		PIC16F627	PIC16F628	PIC16LF627	PIC16LF628
<b>Clock</b>	Maximum Frequency of Operation (MHz)	20	20	20	20
	FLASH Program Memory (words)	1024	2048	1024	2048
<b>Memory</b>	RAM Data Memory (bytes)	224	224	224	224
	EEPROM Data Memory (bytes)	128	128	128	128
	Timer Module(s)	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2
<b>Peripherals</b>	Comparators(s)	2	2	2	2
	Capture/Compare/PWM modules	1	1	1	1
	Serial Communications	USART	USART	USART	USART
	Internal Voltage Reference	Yes	Yes	Yes	Yes
	Interrupt Sources	10	10	10	10
<b>Features</b>	I/O Pins	16	16	16	16
	Voltage Range (Volts)	3.0-5.5	3.0-5.5	2.0-5.5	2.0-5.5
	Brown-out Detect	Yes	Yes	Yes	Yes
	Packages	18-pin DIP, SOIC; 20-pin SSOP	18-pin DIP, SOIC; 20-pin SSOP	18-pin DIP, SOIC; 20-pin SSOP	18-pin DIP, SOIC; 20-pin SSOP

All PICmicro<sup>®</sup> Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability. All PIC16F62X Family devices use serial programming with clock pin RB6 and data pin RB7.

## 2.0 PIC16F62X DEVICE VARIETIES

A variety of frequency ranges and packaging options are available. Depending on application and production requirements the proper device option can be selected using the information in the PIC16F62X Product Identification System section at the end of this data sheet. When placing orders, please use this page of the data sheet to specify the correct part number.

### 2.1 Flash Devices

These devices are offered in the lower cost plastic package, even though the device can be erased and reprogrammed. This allows the same device to be used for prototype development and pilot programs as well as production.

A further advantage of the electrically-erasable Flash version is that it can be erased and reprogrammed in-circuit, or by device programmers, such as Microchip's PICSTART<sup>®</sup> Plus or PRO MATE<sup>®</sup> II programmers.

### 2.2 Quick-Turnaround-Production (QTP) Devices

Microchip offers a QTP Programming Service for factory production orders. This service is made available for users who chose not to program a medium to high quantity of units and whose code patterns have stabilized. The devices are standard FLASH devices but with all program locations and configuration options already programmed by the factory. Certain code and prototype verification procedures apply before production shipments are available. Please contact your Microchip Technology sales office for more details.

### 2.3 Serialized Quick-Turnaround-Production (SQTP<sup>SM</sup>) Devices

Microchip offers a unique programming service where a few user-defined locations in each device are programmed with different serial numbers. The serial numbers may be random, pseudo-random or sequential.

Serial programming allows each device to have a unique number which can serve as an entry-code, password or ID number.

# PIC16F62X

---

NOTES:



## 3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16F62X family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16F62X 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 Table below lists program memory (Flash, Data and EEPROM).

Device	Memory		
	FLASH Program	RAM Data	EEPROM Data
PIC16F627	1024 x 14	224 x 8	128 x 8
PIC16F628	2048 x 14	224 x 8	128 x 8
PIC16LF627	1024 x 14	224 x 8	128 x 8
PIC16LF628	2048 x 14	224 x 8	128 x 8

The PIC16F62X 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 PIC16F62X have 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 PIC16F62X simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16F62X 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-bit 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.

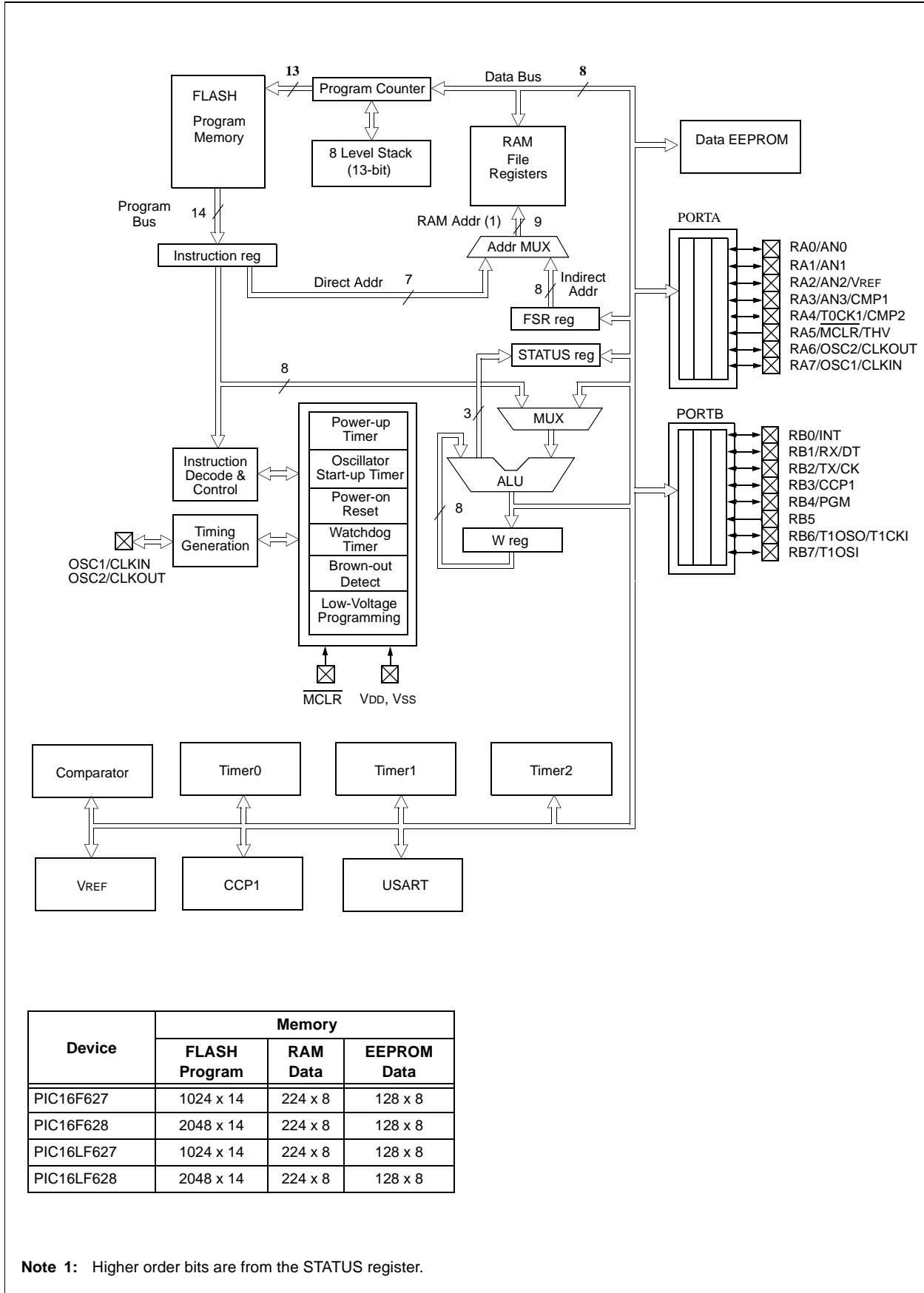
Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, bit in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

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

Two types of data memory are provided on the PIC16F62X devices. Non-volatile EEPROM data memory is provided for long term storage of data such as calibration values, look up table data, and any other data which may require periodic updating in the field. This data is not lost when power is removed. The other data memory provided is regular RAM data memory. Regular RAM data memory is provided for temporary storage of data during normal operation. It is lost when power is removed.

# PIC16F62X

**FIGURE 3-1: BLOCK DIAGRAM**



**TABLE 3-1: PIC16F62X PINOUT DESCRIPTION**

Name	DIP/ SOIC Pin #	SSOP Pin #	I/O/P Type	Buffer Type	Description
RA0/AN0	17	19	I/O	ST	Bi-directional I/O port/Analog comparator input
RA1/AN1	18	20	I/O	ST	Bi-directional I/O port/Analog comparator input
RA2/AN2/VREF	1	1	I/O	ST	Bi-directional I/O port/Analog comparator input/VREF output
RA3/AN3/CMP1	2	2	I/O	ST	Bi-directional I/O port/Analog comparator input/comparator output
RA4/T0CKI/CMP2	3	3	I/O	ST	Bi-directional I/O port/Can be configured as T0CKI/comparator output
RA5/ $\overline{\text{MCLR}}$ /THV	4	4	I	ST	Input port/master clear (reset input/programming voltage input. When configured as $\overline{\text{MCLR}}$ , this pin is an active low reset to the device. Voltage on $\overline{\text{MCLR}}$ /THV must not exceed VDD during normal device operation.
RA6/OSC2/CLKOUT	15	17	I/O	ST	Bi-directional I/O port/Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In ER mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
RA7/OSC1/CLKIN	16	18	I/O	ST	Bi-directional I/O port/Oscillator crystal input/external clock source input. ER biasing pin.
RB0/INT	6	7	I/O	TTL/ST <sup>(1)</sup>	Bi-directional I/O port/external interrupt. Can be software programmed for internal weak pull-up.
RB1/RX/DT	7	8	I/O	TTL/ST <sup>(3)</sup>	Bi-directional I/O port/ USART receive pin/synchronous data I/O. Can be software programmed for internal weak pull-up.
RB2/TX/CK	8	9	I/O	TTL/ST <sup>(3)</sup>	Bi-directional I/O port/ USART transmit pin/synchronous clock I/O. Can be software programmed for internal weak pull-up.
RB3/CCP1	9	10	I/O	TTL/ST <sup>(4)</sup>	Bi-directional I/O port/Capture/Compare/PWM I/O. Can be software programmed for internal weak pull-up.
RB4/PGM	10	11	I/O	TTL/ST <sup>(5)</sup>	Bi-directional I/O port/Low voltage programming input pin. Wake-up from SLEEP on pin change. Can be software programmed for internal weak pull-up. When low voltage programming is enabled, the interrupt on pin change and weak pull-up resistor are disabled.
RB5	11	12	I/O	TTL	Bi-directional I/O port/Wake-up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
RB6/T1OSO/T1CKI	12	13	I/O	TTL/ST <sup>(2)</sup>	Bi-directional I/O port/Timer1 oscillator output/Timer1 clock input. Wake up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
RB7/T1OSI	13	14	I/O	TTL/ST <sup>(2)</sup>	Bi-directional I/O port/Timer1 oscillator input. Wake up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
Vss	5	5,6	P	—	Ground reference for logic and I/O pins.
VDD	14	15,16	P	—	Positive supply for logic and I/O pins.

Legend:            O = output                            I/O = input/output            P = power  
                       — = Not used                        I = Input                        ST = Schmitt Trigger input  
                       TTL = TTL input                        I/OD =input/open drain output

- Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.  
**Note 2:** This buffer is a Schmitt Trigger input when used in serial programming mode.  
**Note 3:** This buffer is a Schmitt Trigger I/O when used in USART/Synchronous mode.  
**Note 4:** This buffer is a Schmitt Trigger I/O when used in CCP mode.  
**Note 5:** This buffer is a Schmitt Trigger input when used in low voltage program mode.

# PIC16F62X

## 3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN/RA7 pin) is internally divided by four to generate four non-overlapping quadrature clocks namely Q1, Q2, Q3 and Q4. Internally, the program counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2.

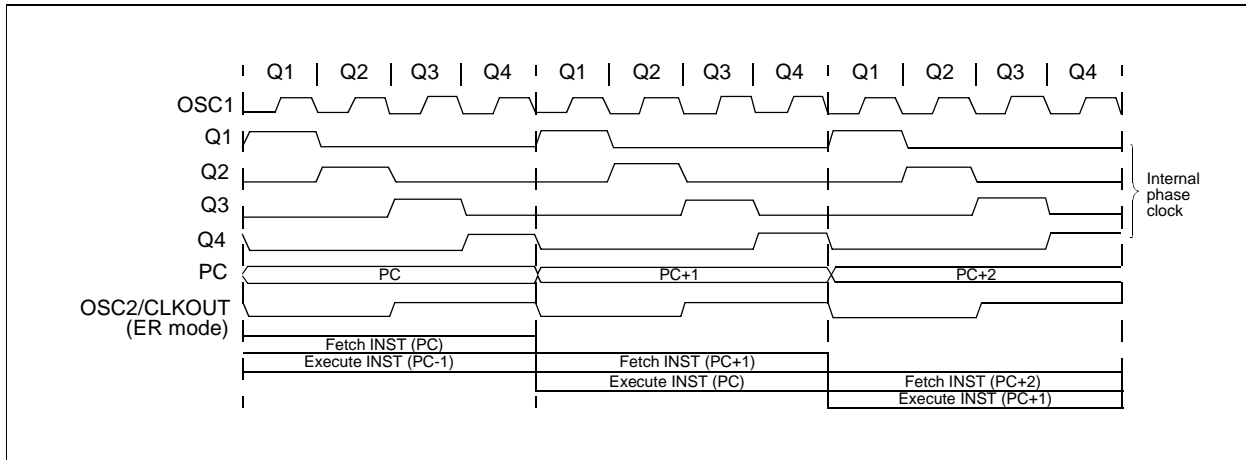
## 3.2 Instruction Flow/Pipelining

An "Instruction Cycle" consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO) then two cycles are required to complete the instruction (Example 3-1).

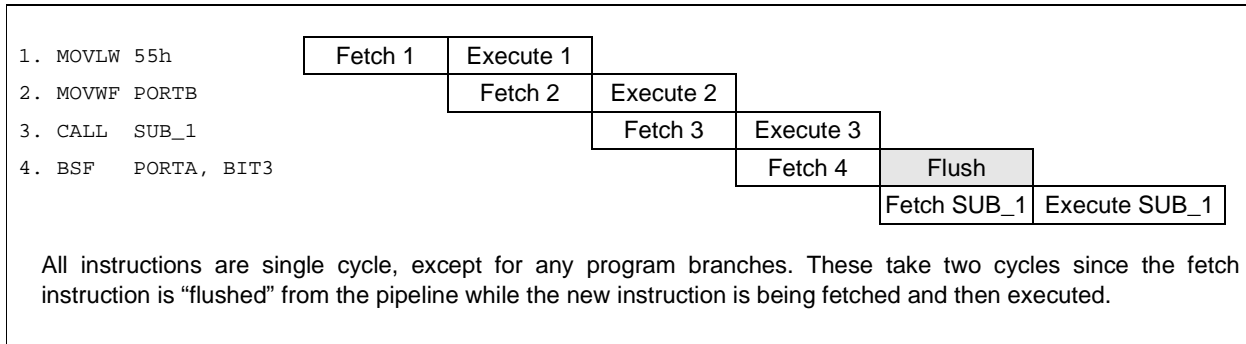
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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

**FIGURE 3-2: CLOCK/INSTRUCTION CYCLE**



**EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW**

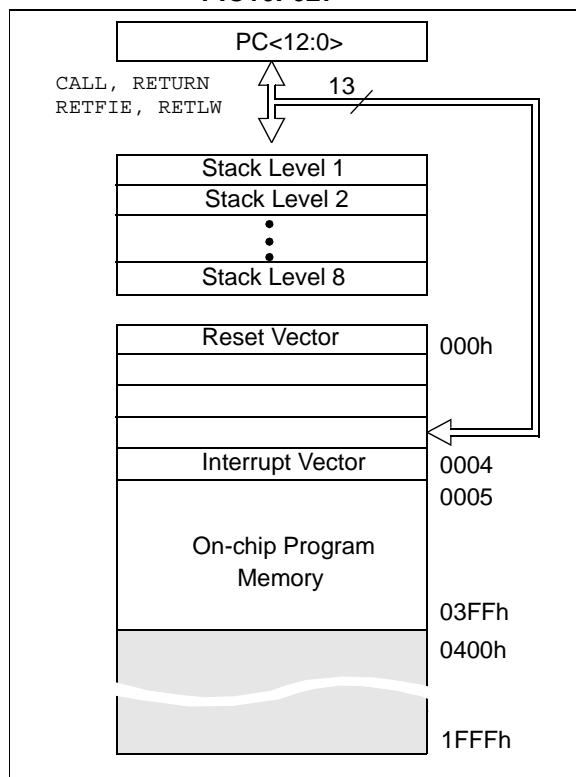


## 4.0 MEMORY ORGANIZATION

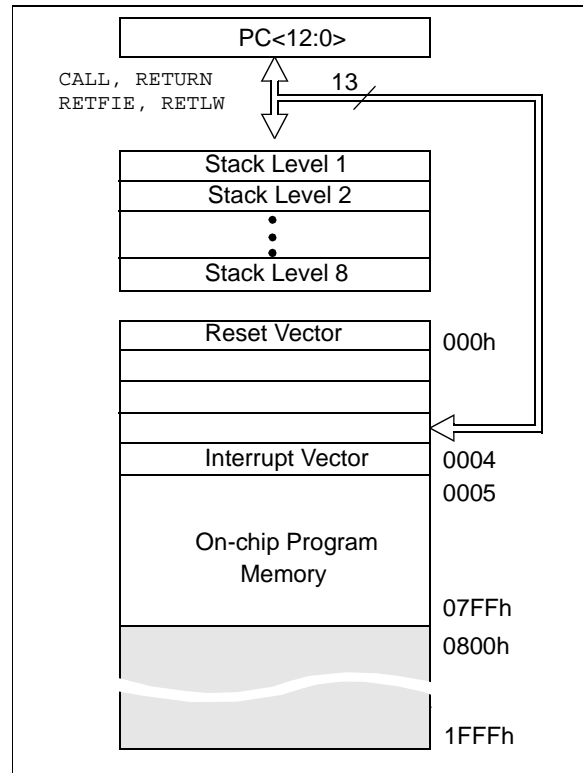
### 4.1 Program Memory Organization

The PIC16F62X has a 13-bit program counter capable of addressing an 8K x 14 program memory space. Only the first 1K x 14 (0000h - 03FFh) for the PIC16F627 and 2K x 14 (0000h - 07FFh) for the PIC16F628 are physically implemented. Accessing a location above these boundaries will cause a wrap-around within the first 1K x 14 space (PIC16F627) or 2K x 14 space (PIC16F628). The reset vector is at 0000h and the interrupt vector is at 0004h (Figure 4-1 and Figure 4-2).

**FIGURE 4-1: PROGRAM MEMORY MAP AND STACK FOR THE PIC16F627**



**FIGURE 4-2: PROGRAM MEMORY MAP AND STACK FOR THE PIC16F628**



### 4.2 Data Memory Organization

The data memory (Figure 4-3) is partitioned into four Banks which contain the general purpose registers and the special function registers. The Special Function Registers are located in the first 32 locations of each Bank. Register locations 20h-7Fh, A0h-FFh, 120h-14Fh, 170h-17Fh and 1F0h-1FFh are general purpose registers implemented as static RAM.

The Table below lists how to access the four banks of registers:

	RP1	RP0
Bank0	0	0
Bank1	0	1
Bank2	1	0
Bank3	1	1

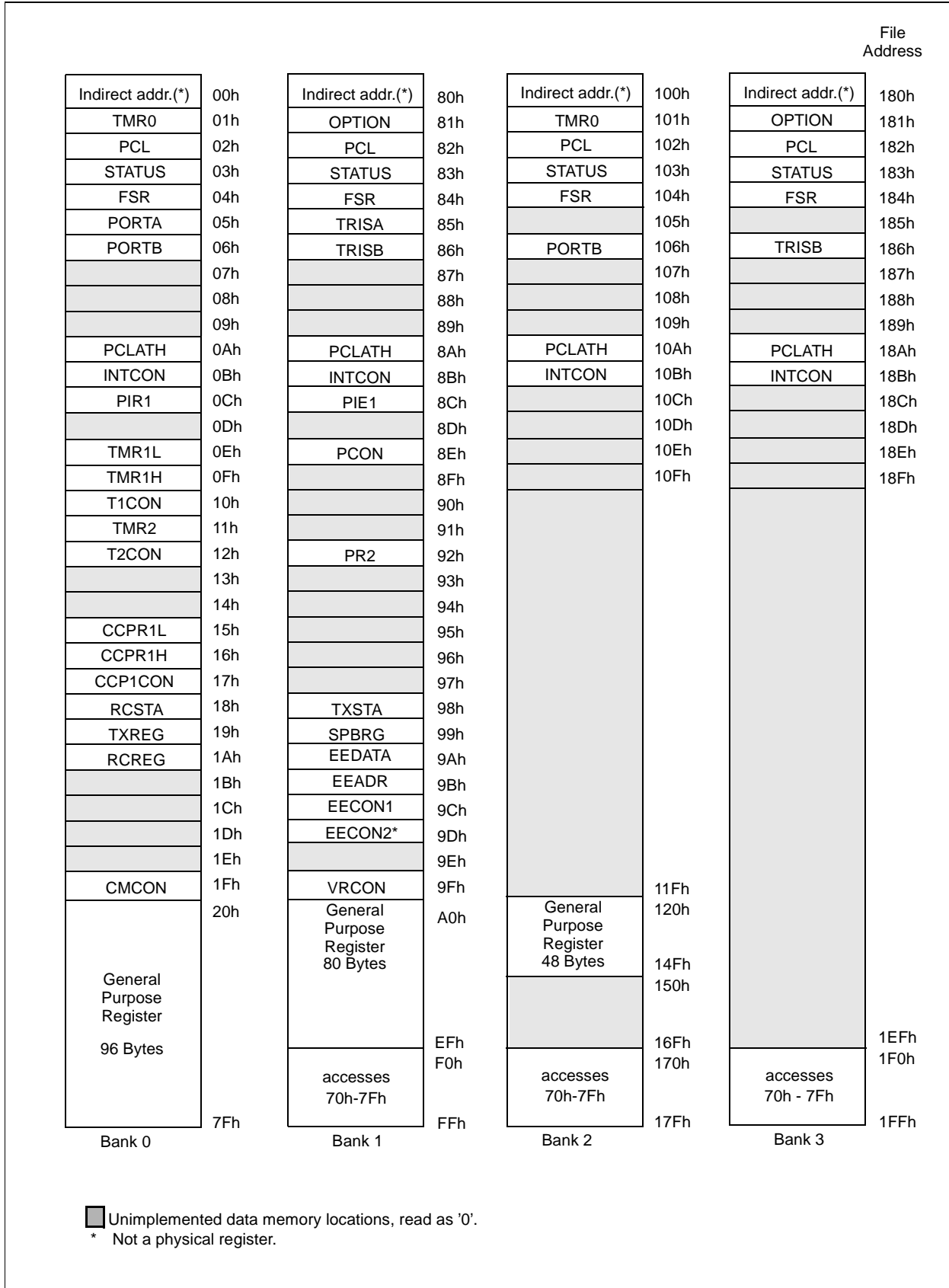
Addresses F0h-FFh, 170h-17Fh and 1F0h-1FFh are implemented as common RAM and mapped back to addresses 70h-7Fh.

#### 4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 224 x 8 in the PIC16F62X. Each is accessed either directly or indirectly through the File Select Register FSR (Section 4.4).

# PIC16F62X

**FIGURE 4-3: DATA MEMORY MAP OF THE PIC16F627 AND PIC16F628**



## 4.2.2 SPECIAL FUNCTION REGISTERS

The special function registers are registers used by the CPU and Peripheral functions for controlling the desired operation of the device (Table 4-1). These registers are static RAM.

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

**TABLE 4-1: SPECIAL REGISTERS SUMMARY BANK0**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other Resets <sup>(1)</sup>		
<b>Bank 0</b>													
00h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx		
01h	TMR0	Timer0 Module's Register								xxxx xxxx	uuuu uuuu		
02h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000		
03h	STATUS	IRP	RP1	RP0	T0	PD	Z	DC	C	0001 1xxx	000q quuu		
04h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu		
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000	xxxx 0000		
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu		
07h	Unimplemented									—	—		
08h	Unimplemented									—	—		
09h	Unimplemented									—	—		
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---	0 0000	---	0 0000
0Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u		
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000		
0Dh	Unimplemented									—	—		
0Eh	TMR1L	Holding register for the least significant byte of the 16-bit TMR1								xxxx xxxx	uuuu uuuu		
0Fh	TMR1H	Holding register for the most significant byte of the 16-bit TMR1								xxxx xxxx	uuuu uuuu		
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu		
11h	TMR2	TMR2 module's register								0000 0000	0000 0000		
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-uuu uuuu		
13h	Unimplemented									—	—		
14h	Unimplemented									—	—		
15h	CCPR1L	Capture/Compare/PWM register (LSB)								xxxx xxxx	uuuu uuuu		
16h	CCPR1H	Capture/Compare/PWM register (MSB)								xxxx xxxx	uuuu uuuu		
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000		
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x		
19h	TXREG	USART Transmit data register								0000 0000	0000 0000		
1Ah	RCREG	USART Receive data register								0000 0000	0000 0000		
1Bh	Unimplemented									—	—		
1Ch	Unimplemented									—	—		
1Dh	Unimplemented									—	—		
1Eh	Unimplemented									—	—		
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000		

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

**Note 1:** Other (non power-up) resets include MCLR Reset, Brown-out Detect and Watchdog Timer Reset during normal operation.

# PIC16F62X

**TABLE 4-2: SPECIAL FUNCTION REGISTERS SUMMARY BANK1**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets <sup>(1)</sup>	
Bank 1												
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx	
81h	OPTION	RBPV	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	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu	
84h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu	
85h	TRISA	TRISA7	TRISA6	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11-1 1111	11-1 1111	
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111	
87h	Unimplemented									—	—	
88h	Unimplemented									—	—	
89h	Unimplemented									—	—	
8Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter				---	0 0000	---	0 0000
8Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u	
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000	
8Dh	Unimplemented									—	—	
8Eh	PCON	—	—	—	—	OSCF	—	$\overline{POR}$	$\overline{BOD}$	---- 1-0x	---- 1-uq	
8Fh	Unimplemented									—	—	
90h	Unimplemented									—	—	
91h	Unimplemented									—	—	
92h	PR2	Timer2 Period Register								11111111	11111111	
93h	Unimplemented									—	—	
94h	Unimplemented									—	—	
95h	Unimplemented									—	—	
96h	Unimplemented									—	—	
97h	Unimplemented									—	—	
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010	
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000	
9Ah	EEDATA	EEPROM data register								xxxx xxxx	uuuu uuuu	
9Bh	EEADR	—	EEPROM address register								xxxx xxxx	uuuu uuuu
9Ch	EECON1	—	—	—	—	WRERR	WREN	WR	RD	---- x000	---- q000	
9Dh	EECON2	EEPROM control register 2 (not a physical register)								-----	-----	
9Eh	Unimplemented									—	—	
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	000- 0000	

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

**Note 1:** Other (non power-up) resets include  $\overline{MCLR}$  Reset, Brown-out Detect and Watchdog Timer Reset during normal operation.



**TABLE 4-3: SPECIAL FUNCTION REGISTERS SUMMARY BANK2**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets <sup>(1)</sup>
Bank 1											
100h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx
101h	TMR0	RBP $\overline{U}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
102h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
103h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
104h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
105h	Unimplemented									—	—
106h	PORTB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
107h	Unimplemented									—	—
108h	Unimplemented									—	—
109h	Unimplemented									—	—
10Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter				---0 0000	---0 0000	
10Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
10Ch										—	—
10Dh	Unimplemented									—	—
10Eh										—	—
10Fh	Unimplemented									—	—
110h	Unimplemented									—	—
111h	Unimplemented									—	—
112h										—	—
113h	Unimplemented									—	—
114h	Unimplemented									—	—
115h	Unimplemented									—	—
116h	Unimplemented									—	—
117h	Unimplemented									—	—
118h										—	—
119h										—	—
11Ah										—	—
11Bh										—	—
11Ch										—	—
11Dh										—	—
11Eh	Unimplemented									—	—
11Fh										—	—

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

**Note 1:** Other (non power-up) resets include  $\overline{MCLR}$  Reset, Brown-out Detect and Watchdog Timer Reset during normal operation.

# PIC16F62X

**TABLE 4-4: SPECIAL FUNCTION REGISTERS SUMMARY BANK3**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets <sup>(1)</sup>
Bank 1											
180h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx
181h	OPTION	RBP $\overline{U}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
182h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
183h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
184h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
185h	Unimplemented									—	—
186h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
187h	Unimplemented									—	—
188h	Unimplemented									—	—
189h	Unimplemented									—	—
18Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---0 0000	---0 0000
18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
18Ch											
18Dh											
18Eh											
18Fh											
190h											
191h											
192h											
193h											
194h											
195h											
196h											
197h											
198h											
199h											
19Ah											
19Bh											
19Ch											
19Dh											
19Eh											
19Fh											

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

**Note 1:** Other (non power-up) resets include  $\overline{MCLR}$  Reset, Brown-out Detect and Watchdog Timer Reset during normal operation.

## 4.2.2.1 STATUS REGISTER

The STATUS register, shown in Register 4-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory (SRAM).

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the  $\overline{TO}$  and  $\overline{PD}$  bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the status register as `000uu1uu` (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect any status bit. For other instructions, not affecting any status bits, see the "Instruction Set Summary".

**Note 1:** The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

### REGISTER 4-1: STATUS REGISTER (ADDRESS 03H OR 83H)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
			bit7				bit0

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

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

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

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

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

bit 2: **Z:** Zero bit  
 1 = The result of an arithmetic or logic operation is zero  
 0 = The result of an arithmetic or logic operation is not zero

bit 1: **DC:** Digit carry/borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)(for borrow the polarity is reversed)  
 1 = A carry-out from the 4th low order bit of the result occurred  
 0 = No carry-out from the 4th low order bit of the result

bit 0: **C:** Carry/borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)  
 1 = A carry-out from the most significant bit of the result occurred  
 0 = No carry-out from the most significant bit of the result occurred

**Note:** For borrow the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

# PIC16F62X

## 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 interrupt, TMR0, and the weak pull-ups on PORTB.

**Note:** To achieve a 1:1 prescaler assignment for TMR0, assign the prescaler to the WDT (PSA = 1). See Section 6.3.1

### REGISTER 4-2: OPTION REGISTER (ADDRESS 81H)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
<b>RBPU</b>	<b>INTEDG</b>	<b>T0CS</b>	<b>T0SE</b>	<b>PSA</b>	<b>PS2</b>	<b>PS1</b>	<b>PS0</b>
bit7						bit0	

R = Readable bit  
W = Writable bit  
-n = Value at POR reset

bit 7: **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

## 4.2.2.3 INTCON REGISTER

The INTCON register is a readable and writable register which contains the various enable and flag bits for all interrupt sources except the comparator module. See Section 4.2.2.4 and Section 4.2.2.5 for a description of the comparator enable and flag bits.

**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>).

### REGISTER 4-3: INTCON REGISTER (ADDRESS 0BH OR 8BH)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF
bit7							bit0

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

bit 7: **GIE:** Global Interrupt Enable bit  
 1 = Enables all un-masked interrupts  
 0 = Disables all interrupts

bit 6: **PEIE:** Peripheral Interrupt Enable bit  
 1 = Enables all un-masked peripheral interrupts  
 0 = Disables all peripheral interrupts

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

bit 4: **INTE:** RB0/INT External Interrupt Enable bit  
 1 = Enables the RB0/INT external interrupt  
 0 = Disables the RB0/INT external interrupt

bit 3: **RBIE:** RB Port Change Interrupt Enable bit  
 1 = Enables the RB port change interrupt  
 0 = Disables the RB port change interrupt

bit 2: **TOIF:** TMR0 Overflow Interrupt Flag bit  
 1 = TMR0 register has overflowed (must be cleared in software)  
 0 = TMR0 register did not overflow

bit 1: **INTF:** RB0/INT External Interrupt Flag bit  
 1 = The RB0/INT external interrupt occurred (must be cleared in software)  
 0 = The RB0/INT external interrupt did not occur

bit 0: **RBIF:** RB Port Change Interrupt Flag bit  
 1 = When at least one of the RB7:RB4 pins changed state (must be cleared in software)  
 0 = None of the RB7:RB4 pins have changed state

# PIC16F62X

## 4.2.2.4 PIE1 REGISTER

This register contains interrupt enable bits.

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

R/W-0	R/W-0	R/W-0	R/W-0	U	R/W-0	R/W-0	R/W-0
EEIE	CMIE	RCIE	TXIE	-	CCP1IE	TMR2IE	TMR1IE
bit7					bit0		

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

bit 7: **EEIE:** EE Write Complete Interrupt Enable Bit  
1 = Enables the EE write complete interrupt  
0 = Disables the EE write complete interrupt

bit 6: **CMIE:** Comparator Interrupt Enable bit  
1 = Enables the comparator interrupt  
0 = Disables the comparator interrupt

bit 5: **RCIE:** USART Receive Interrupt Enable bit  
1 = Enables the USART receive interrupt  
0 = Disables the USART receive interrupt

bit 4: **TXIE:** USART Transmit Interrupt Enable bit  
1 = Enables the USART transmit interrupt  
0 = Disables the USART transmit interrupt

bit 3: **Unimplemented:** Read as '0'

bit 2: **CCP1IE:** CCP1 Interrupt Enable bit  
1 = Enables the CCP1 interrupt  
0 = Disables the CCP1 interrupt

bit 1: **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit  
1 = Enables the TMR2 to PR2 match interrupt  
0 = Disables the TMR2 to PR2 match interrupt

bit 0: **TMR1IE:** TMR1 Overflow Interrupt Enable bit  
1 = Enables the TMR1 overflow interrupt  
0 = Disables the TMR1 overflow interrupt

## 4.2.2.5 PIR1 REGISTER

This register contains interrupt flag bits.

**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)

R/W-0	R/W-0	R-0	R-0	U	R/W-0	R/W-0	R/W-0
EEIF	CMIF	RCIF	TXIF	-	CCP1IF	TMR2IF	TMR1IF
bit7					bit0		

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

bit 7: **EEIF:** EEPROM Write Operation Interrupt Flag bit  
 1 = The write operation completed (must be cleared in software)  
 0 = The write operation has not completed or has not been started

bit 6: **CMIF:** Comparator Interrupt Flag bit  
 1 = Comparator input has changed  
 0 = Comparator input has not changed

bit 5: **RCIF:** USART Receive Interrupt Flag bit  
 1 = The USART receive buffer is full  
 0 = The USART receive buffer is empty

bit 4: **TXIF:** USART Transmit Interrupt Flag bit  
 1 = The USART transmit buffer is empty  
 0 = The USART transmit buffer is full

bit 3: **Unimplemented:** Read as '0'

bit 2: **CCP1IF:** CCP1 Interrupt Flag bit  
Capture Mode  
 1 = A TMR1 register capture occurred (must be cleared in software)  
 0 = No TMR1 register capture occurred  
Compare Mode  
 1 = A TMR1 register compare match occurred (must be cleared in software)  
 0 = No TMR1 register compare match occurred  
PWM Mode  
 Unused in this mode

bit 1: **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit  
 1 = TMR2 to PR2 match occurred (must be cleared in software)  
 0 = No TMR2 to PR2 match occurred

bit 0: **TMR1IF:** TMR1 Overflow Interrupt Flag bit  
 1 = TMR1 register overflowed (must be cleared in software)  
 0 = TMR1 register did not overflow

# PIC16F62X

## 4.2.2.6 PCON REGISTER

The PCON register contains flag bits to differentiate between a Power-on Reset, an external MCLR reset, WDT reset or a Brown-out Detect.

**Note:**  $\overline{\text{BOD}}$  is unknown on Power-on Reset. It must then be set by the user and checked on subsequent resets to see if  $\overline{\text{BOD}}$  is cleared, indicating a brown-out has occurred. The  $\overline{\text{BOD}}$  status bit is a "don't care" and is not necessarily predictable if the brown-out circuit is disabled (by programming BOREN bit in the Configuration word).

### REGISTER 4-6: PCON REGISTER (ADDRESS 8Eh)

U-0	U-0	U-0	U-0	R/W-1	U-0	R/W-q	R/W-q
—	—	—	—	OSCF	—	$\overline{\text{POR}}$	$\overline{\text{BOD}}$
bit7						bit0	

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

bit 7-4,2: **Unimplemented:** Read as '0'

bit 3: **OSCF:** INTRC/ER oscillator speed  
 1 = 4 MHz typical<sup>(1)</sup>  
 0 = 37 KHz typical

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

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

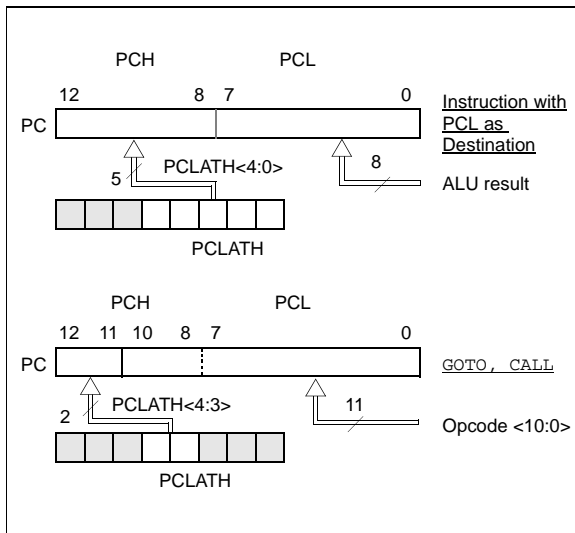
**Note 1:** When in ER oscillator mode, setting OSCF = 1 will cause the oscillator speed to change to the speed specified by the external resistor.



## 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 4-7 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 4-7: 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 PIC16F62X family has an 8 level deep x 13-bit wide hardware stack (Figure 4-1 and Figure 4-2). 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.

# PIC16F62X

## 4.4 Indirect Addressing, INDF and FSR Registers

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

Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the file select register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-8.

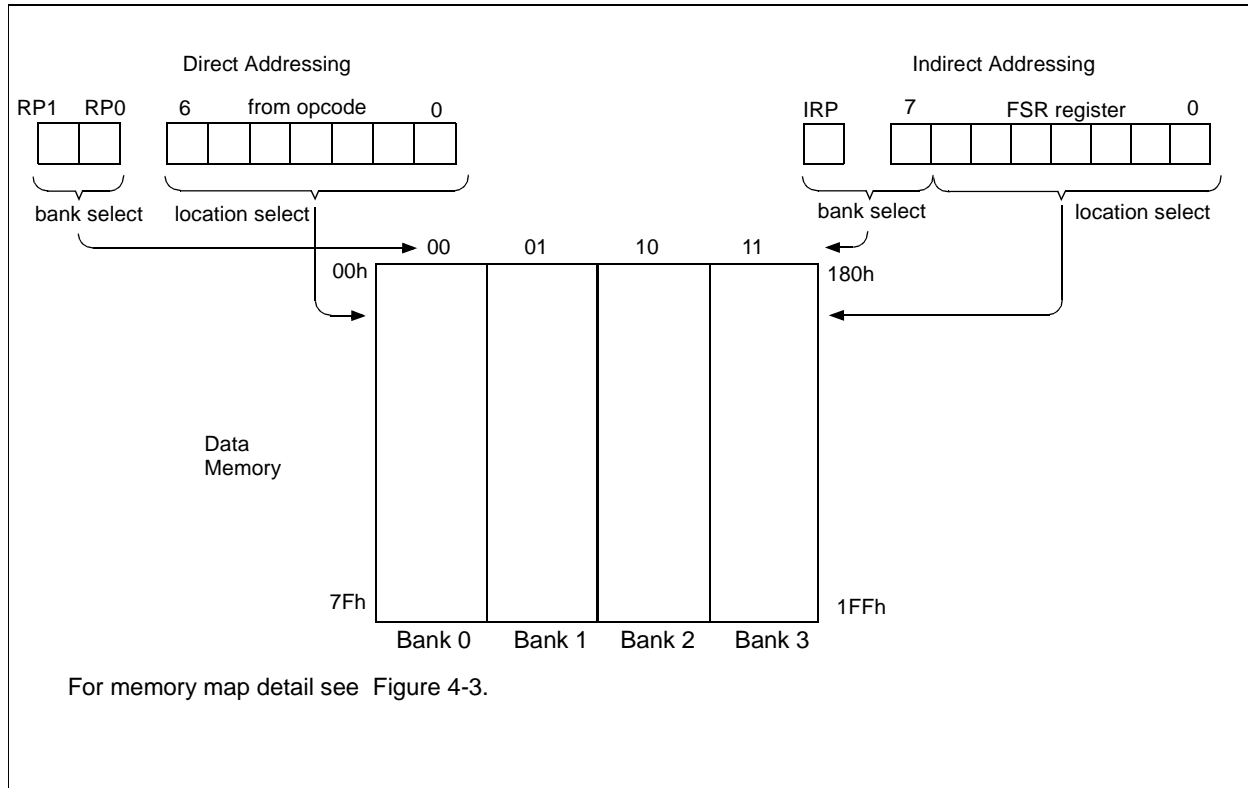
A simple program to clear RAM location 20h-2Fh using indirect addressing is shown in Example 4-1.

### EXAMPLE 4-1: INDIRECT ADDRESSING

```

movlw 0x20 ;initialize pointer
movwf FSR ;to RAM
NEXT   clrf INDF ;clear INDF register
       incf FSR ;inc pointer
       btfss FSR,4 ;all done?
       goto NEXT ;no clear next
                               ;yes continue
CONTINUE:
    
```

FIGURE 4-8: DIRECT/INDIRECT ADDRESSING PIC16F62X



## 5.0 I/O PORTS

The PIC16F62X 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 an 8-bit wide latch. RA4 is a Schmitt Trigger input and an open drain output. Port RA4 is multiplexed with the T0CKI clock input. RA5 is a Schmitt Trigger input only and has no output drivers. 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.

**Note 1:** 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.

**Note 2:** When RA6/OSC2/CLKOUT is configured as CLKOUT, the corresponding TRIS bit is overridden and the pin is configured as an output. The PORTA data bit reads 0, and the PORTA TRIS bit reads 0.

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. 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.

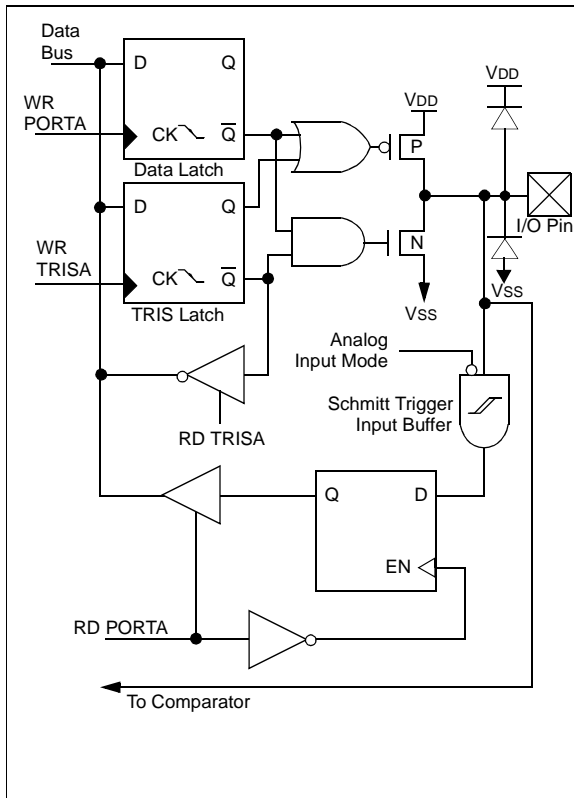
#### EXAMPLE 5-1: INITIALIZING PORTA

```
CLRF  PORTA      ;Initialize PORTA by setting
                  ;output data latches
MOVLW 0X07      ;Turn comparators off and
MOVWF  CMCON     ;enable pins for I/O
                  ;functions

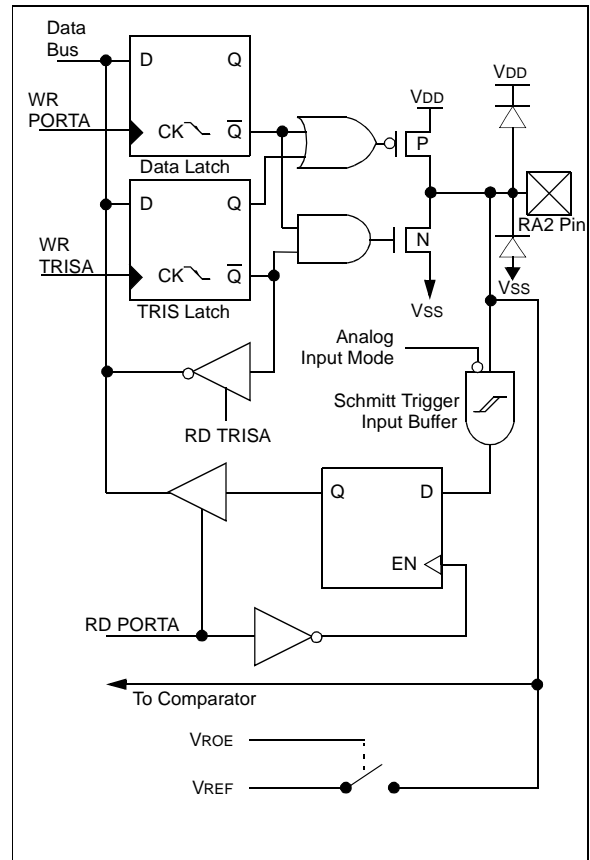
BCF   STATUS, RP1
BSF   STATUS, RP0 ;Select Bank1
MOVLW 0x1F      ;Value used to initialize
                  ;data direction
MOVWF TRISA     ;Set RA<4:0> as inputs
                  ;TRISA<7:5> are always
                  ;read as '0'.
```

# PIC16F62X

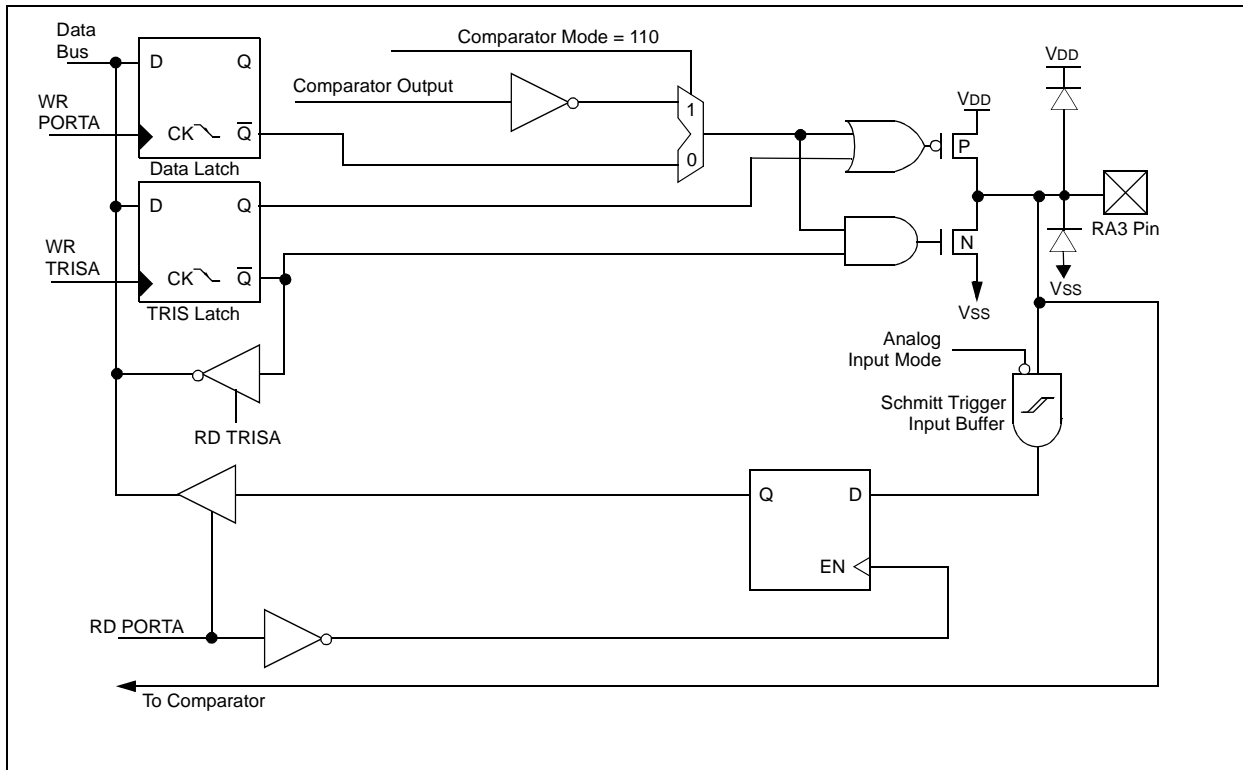
**FIGURE 5-1: BLOCK DIAGRAM OF RA0/AN0:RA1/AN1 PINS**



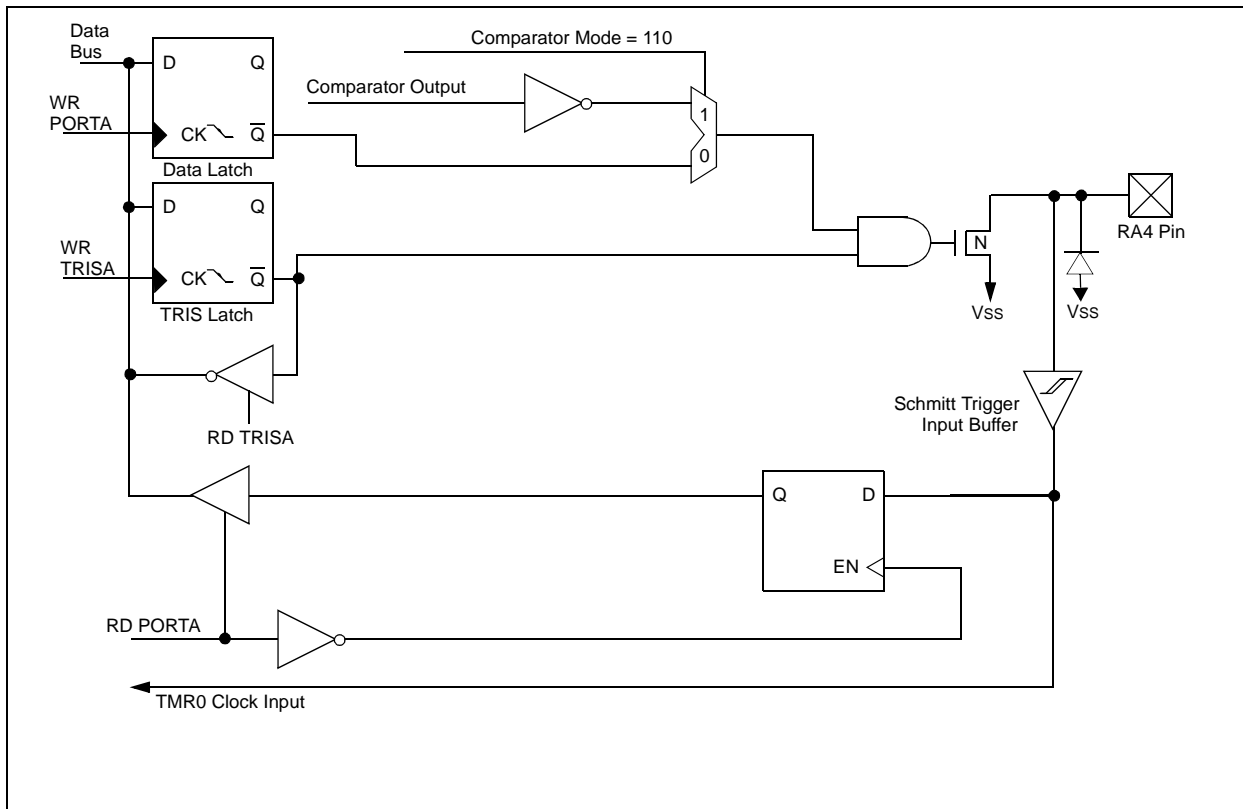
**FIGURE 5-2: BLOCK DIAGRAM OF RA2/VREF PIN**



**FIGURE 5-3: BLOCK DIAGRAM OF THE RA3/AN3 PIN**

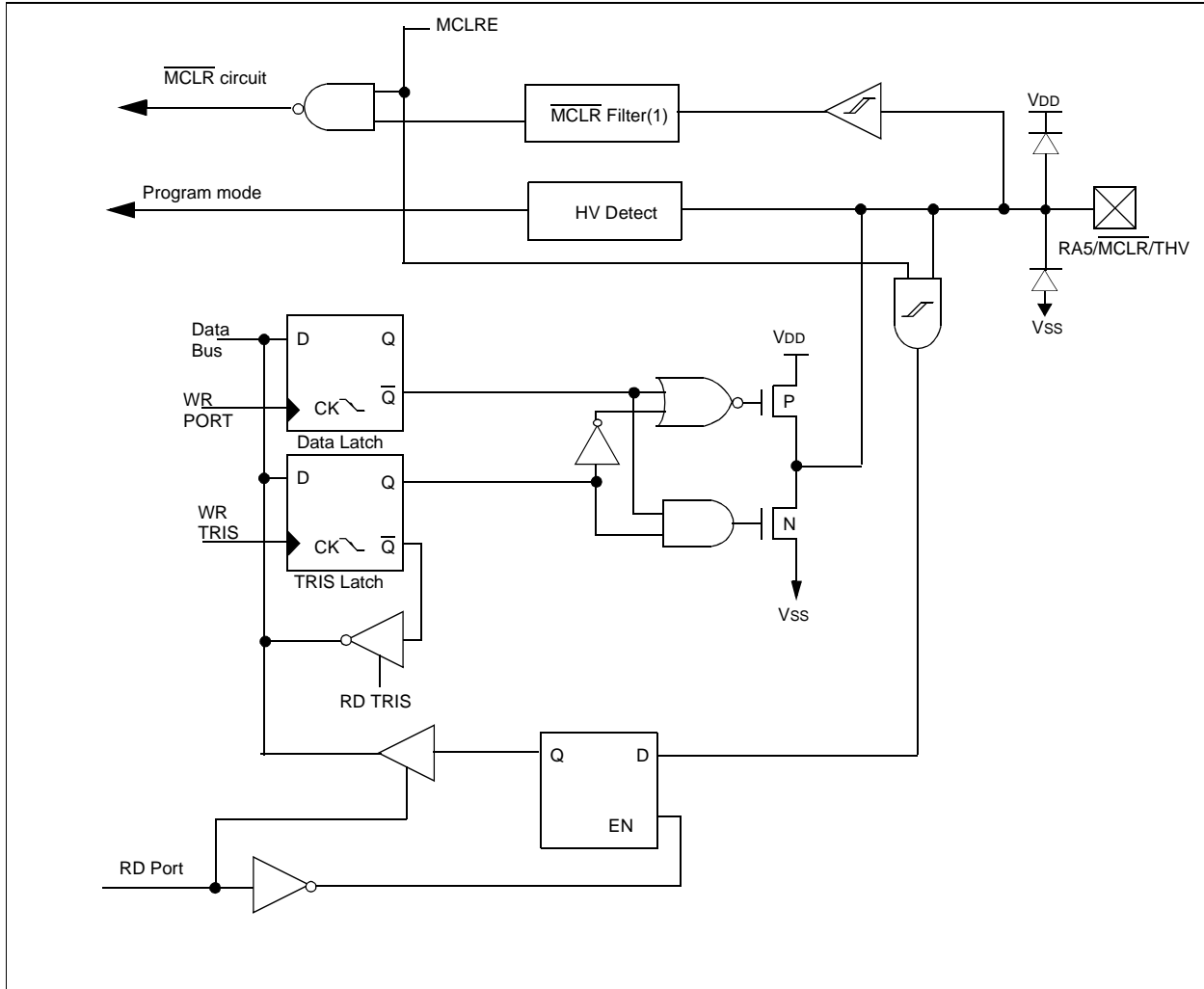


**FIGURE 5-4: BLOCK DIAGRAM OF RA4/T0CKI PIN**

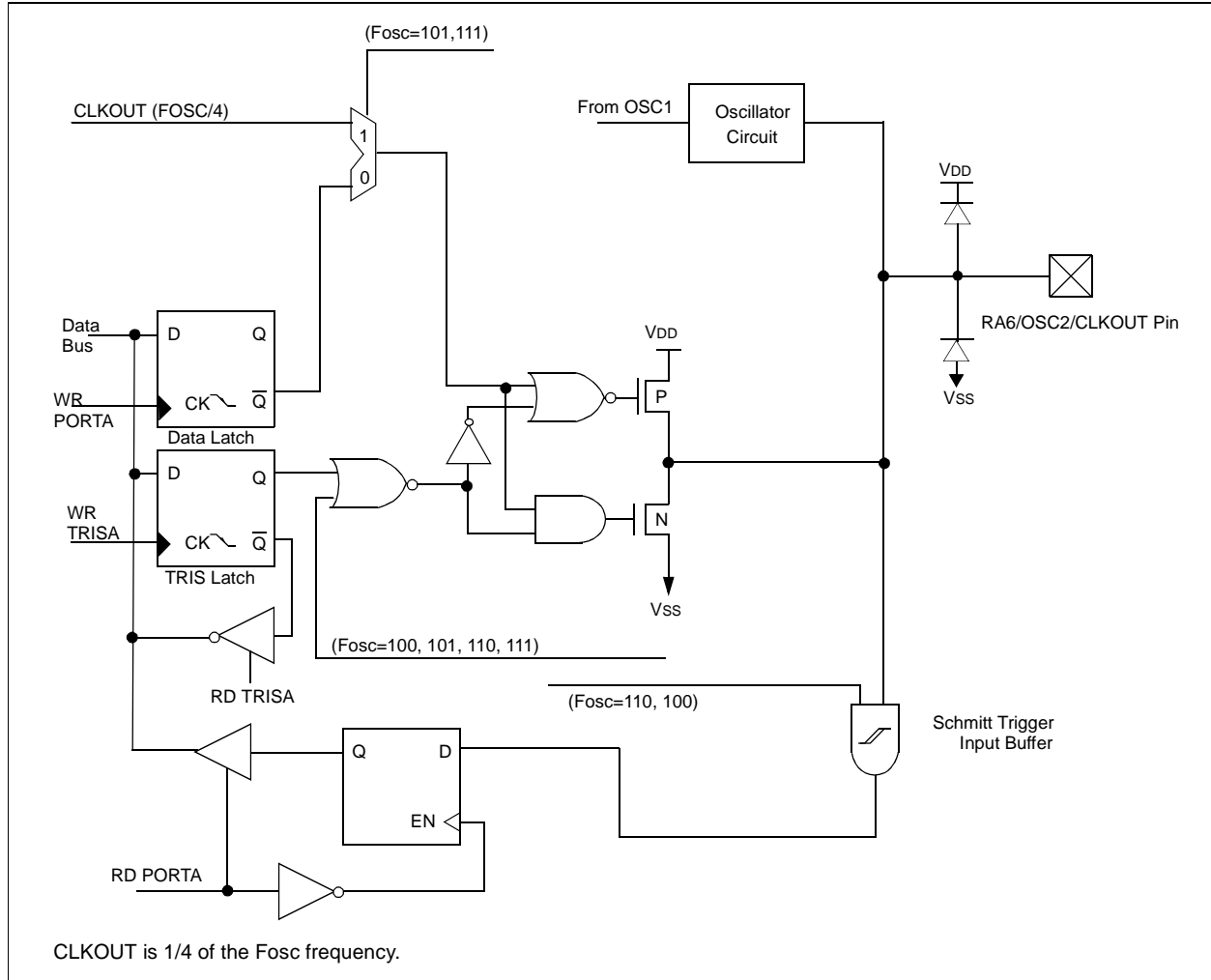


# PIC16F62X

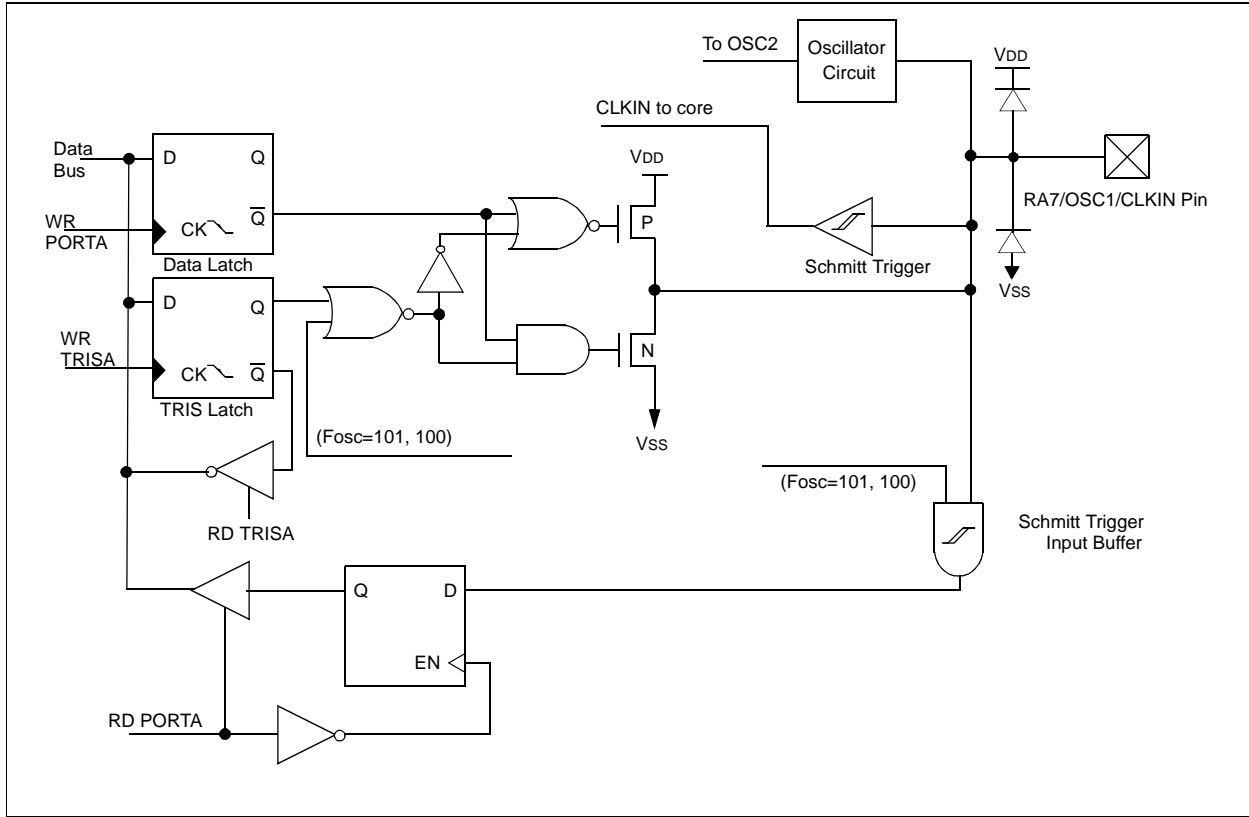
**FIGURE 5-5: BLOCK DIAGRAM OF THE RA5/MCLR/THV PIN**



**FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKOUT PIN**



**FIGURE 5-7: BLOCK DIAGRAM OF RA7/OSC1/CLKIN PIN**





**TABLE 5-1: PORTA FUNCTIONS**

Name	Bit #	Buffer Type	Function
RA0/AN0	bit0	ST	Bi-directional I/O port/comparator input
RA1/AN1	bit1	ST	Bi-directional I/O port/comparator input
RA2/AN2/VREF	bit2	ST	Bi-directional I/O port/analog/comparator input or VREF output
RA3/AN3	bit3	ST	Bi-directional I/O port/analog/comparator input/comparator output
RA4/T0CKI	bit4	ST	Bi-directional I/O port/external clock input for TMR0 or comparator output. Output is open drain type.
RA5/ $\overline{\text{MCLR}}$ /THV	bit5	ST	Input port/master clear (reset input/programming voltage input. When configured as $\overline{\text{MCLR}}$ , this pin is an active low reset to the device. Voltage on $\overline{\text{MCLR}}$ /THV must not exceed VDD during normal device operation.
RA6/OSC2/CLK-OUT	bit6	ST	Bi-directional I/O port/Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In ER mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
RA7/OSC1/CLKIN	bit7	ST	Bi-directional I/O port/oscillator crystal input/external clock source input.

Legend: ST = Schmitt Trigger input

**TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000	xxux 0000
85h	TRISA	TRISA7	TRISA6	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11-1 1111	11-1 1111
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	000- 0000

Legend: — = Unimplemented locations, read as '0', u = unchanged, x = unknown

**Note:** Shaded bits are not used by PORTA.

## 5.2 PORTB and TRISB Registers

PORTB is an 8-bit wide bi-directional port. The corresponding data direction register is TRISB. A '1' in the TRISB register puts the corresponding output driver in a high impedance mode. A '0' in the TRISB register puts the contents of the output latch on the selected pin(s).

PORTB is multiplexed with the interrupt, USART, CCP module and the TMR1 clock input/output. The standard port functions and the alternate port functions are shown in Table 5-3.

Reading PORTB 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.

Each of the PORTB pins has a weak internal pull-up ( $\approx 200 \mu\text{A}$  typical). A single control bit can turn on all the pull-ups. This is done by clearing the  $\overline{\text{RBPU}}$  (OPTION<7>) bit. The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on Power-on Reset.

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

This interrupt can wake the device from SLEEP. The user, in the interrupt service routine, can clear the interrupt in the following manner:

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

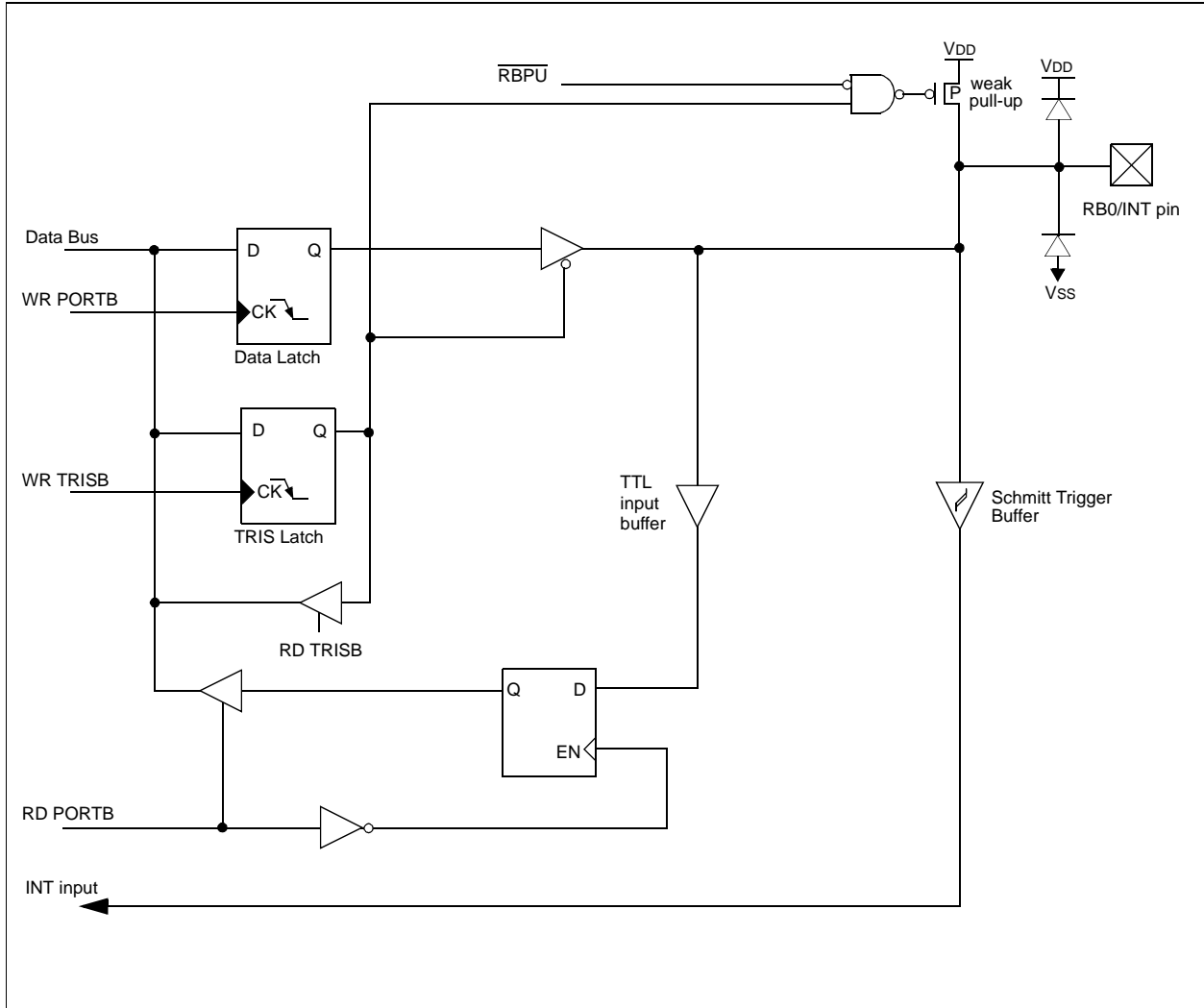
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition, and allow flag bit RBIF to be cleared.

This interrupt on mismatch feature, together with software configurable pull-ups on these four pins allow easy interface to a key pad and make it possible for wake-up on key-depression. (See AN552 in the Microchip *Embedded Control Handbook*.)

**Note:** If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

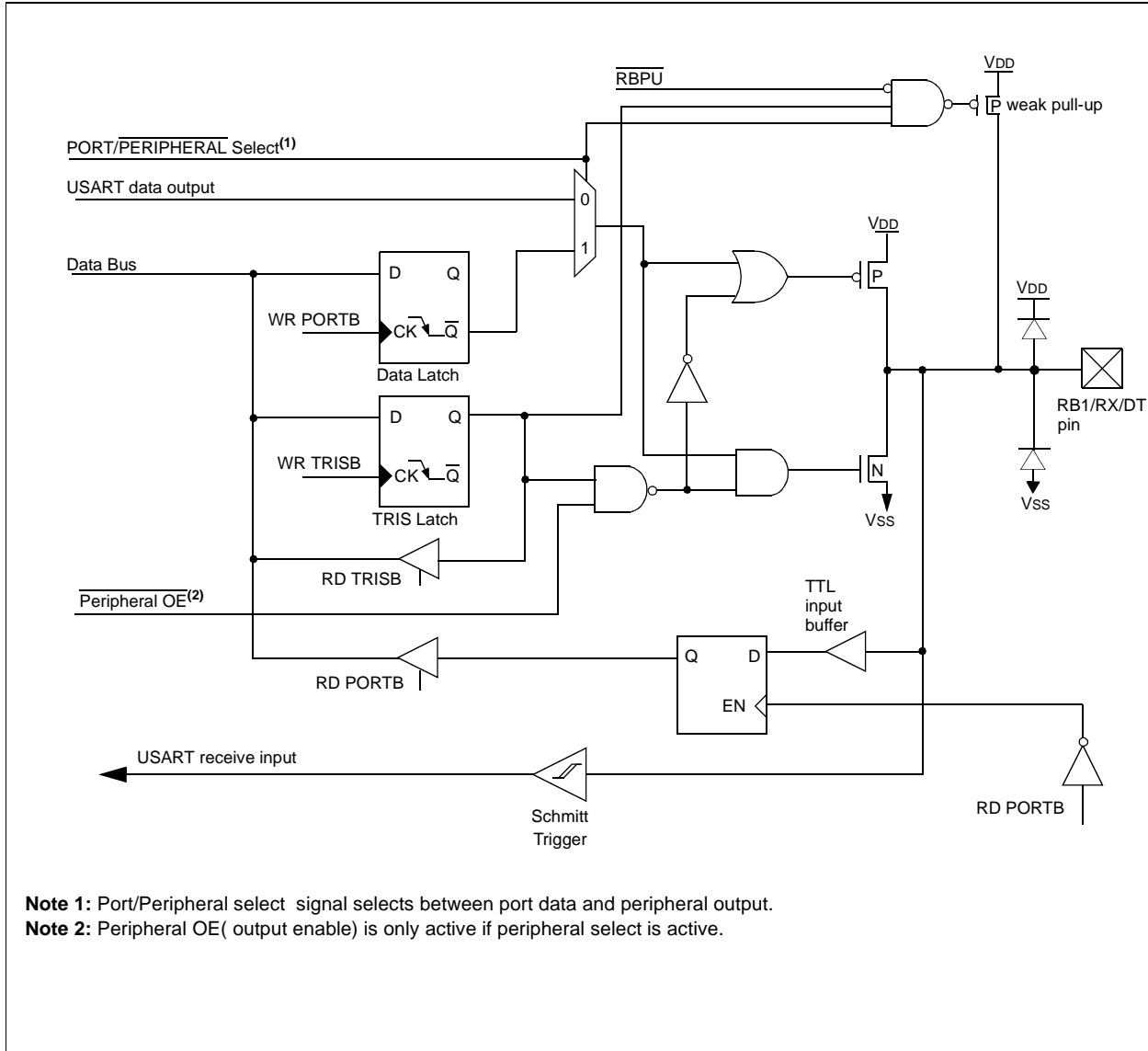
The interrupt on change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt on change feature. Polling of PORTB is not recommended while using the interrupt on change feature.

**FIGURE 5-8: BLOCK DIAGRAM OF RB0/INT PIN**

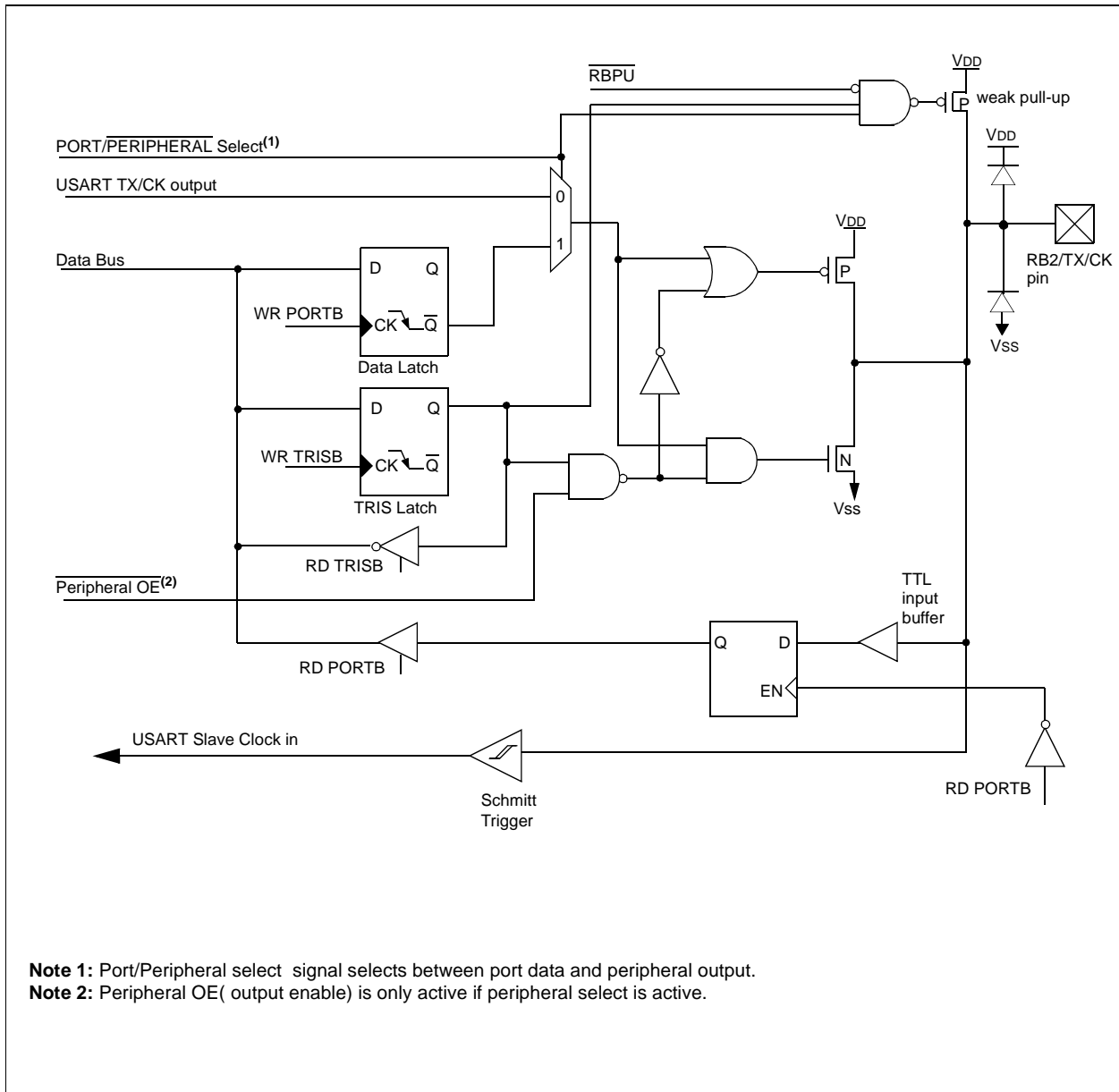


# PIC16F62X

**FIGURE 5-9: BLOCK DIAGRAM OF RB1/TX/DT PIN**

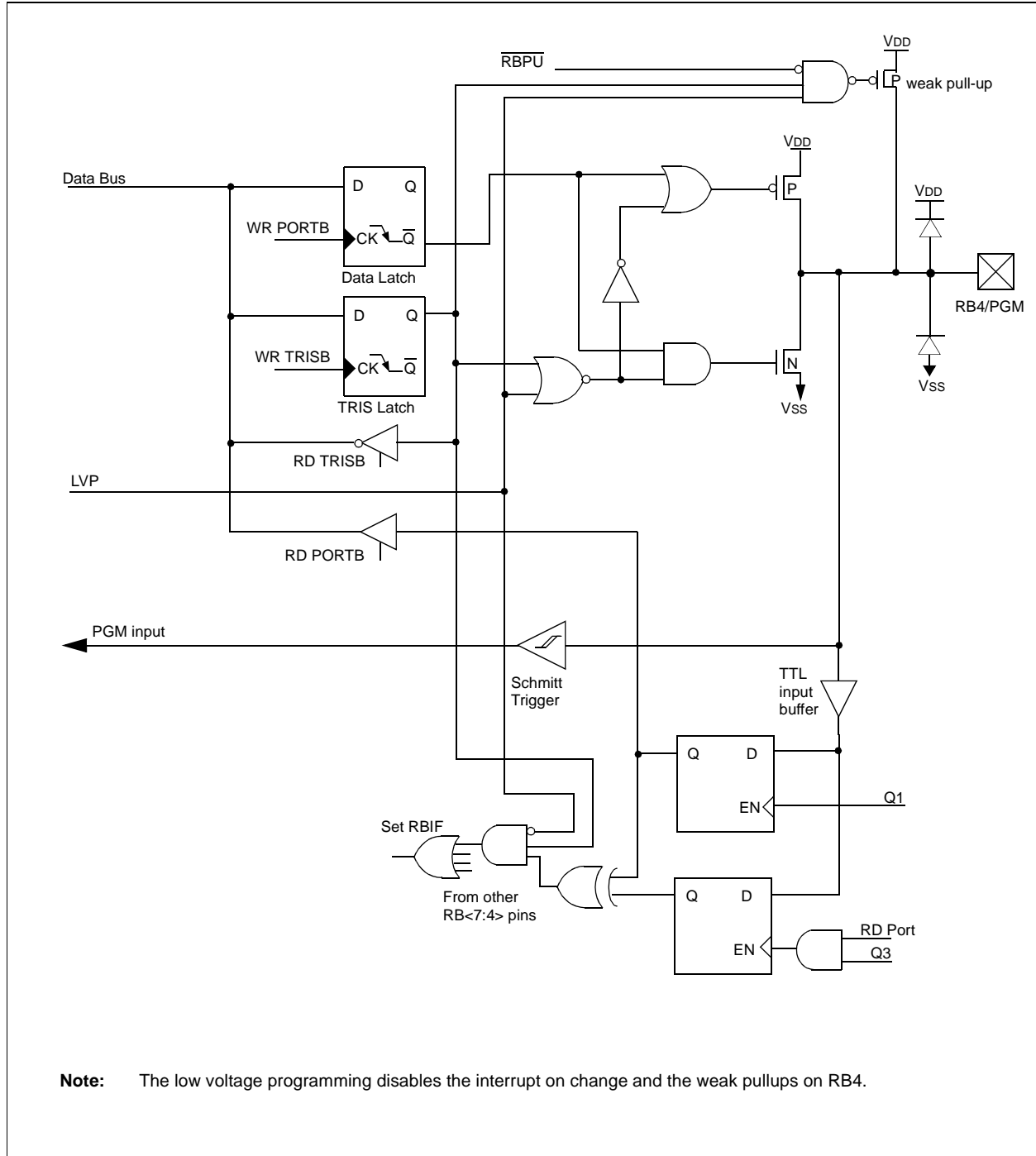


**FIGURE 5-10: BLOCK DIAGRAM OF RB2/TX/CK PIN**



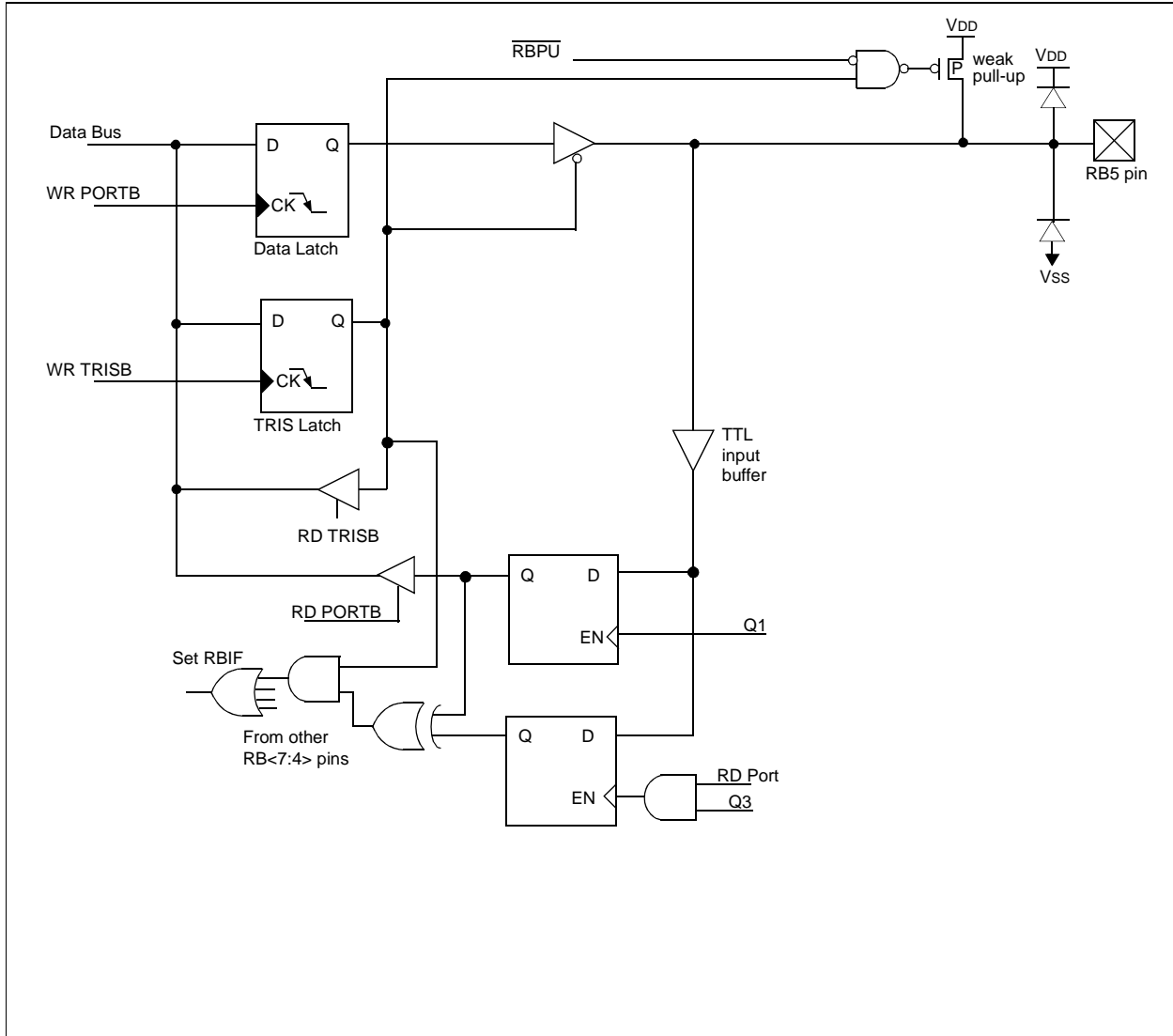


**FIGURE 5-12: BLOCK DIAGRAM OF RB4/PGM PIN**



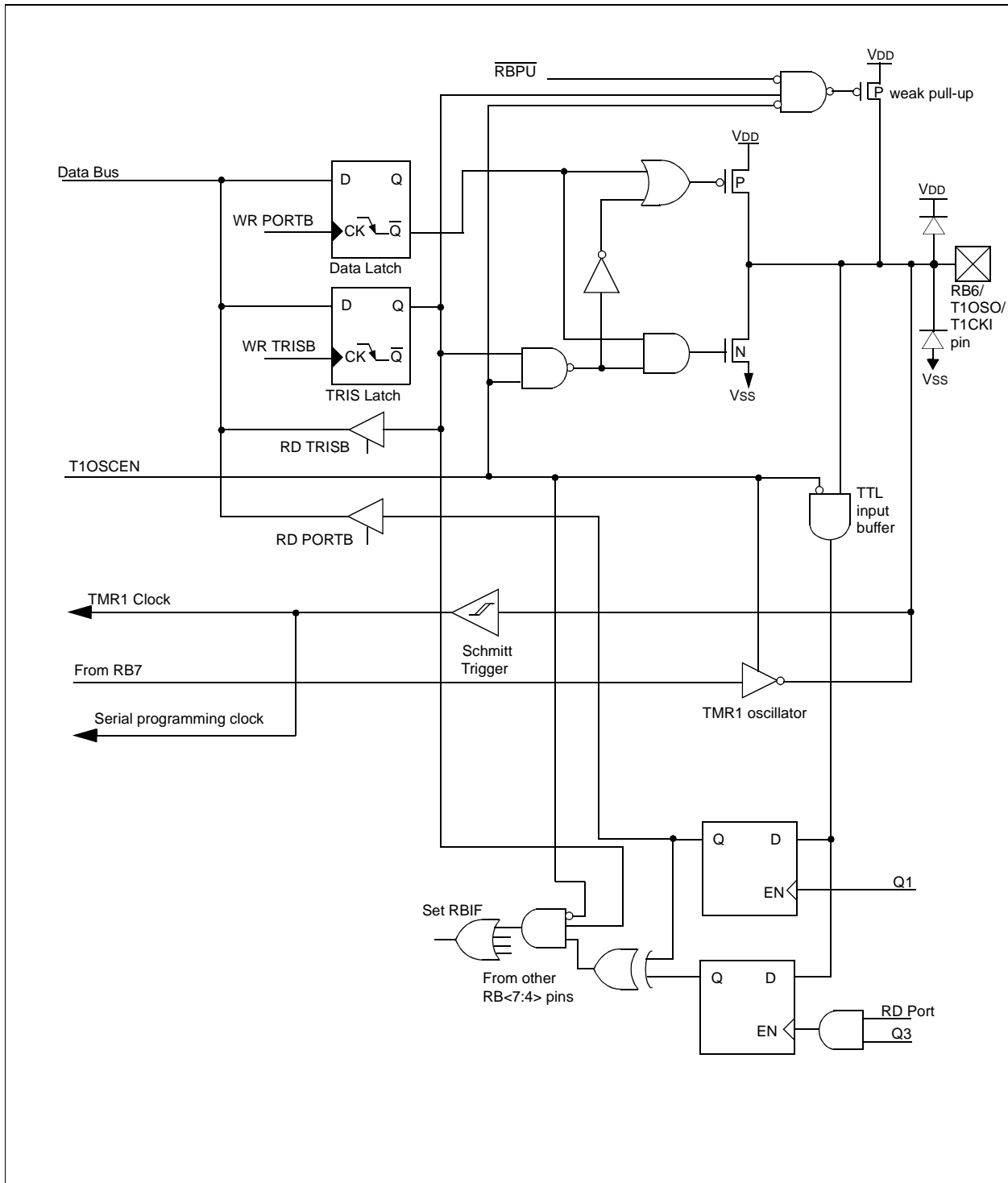
# PIC16F62X

**FIGURE 5-13: BLOCK DIAGRAM OF RB5 PIN**



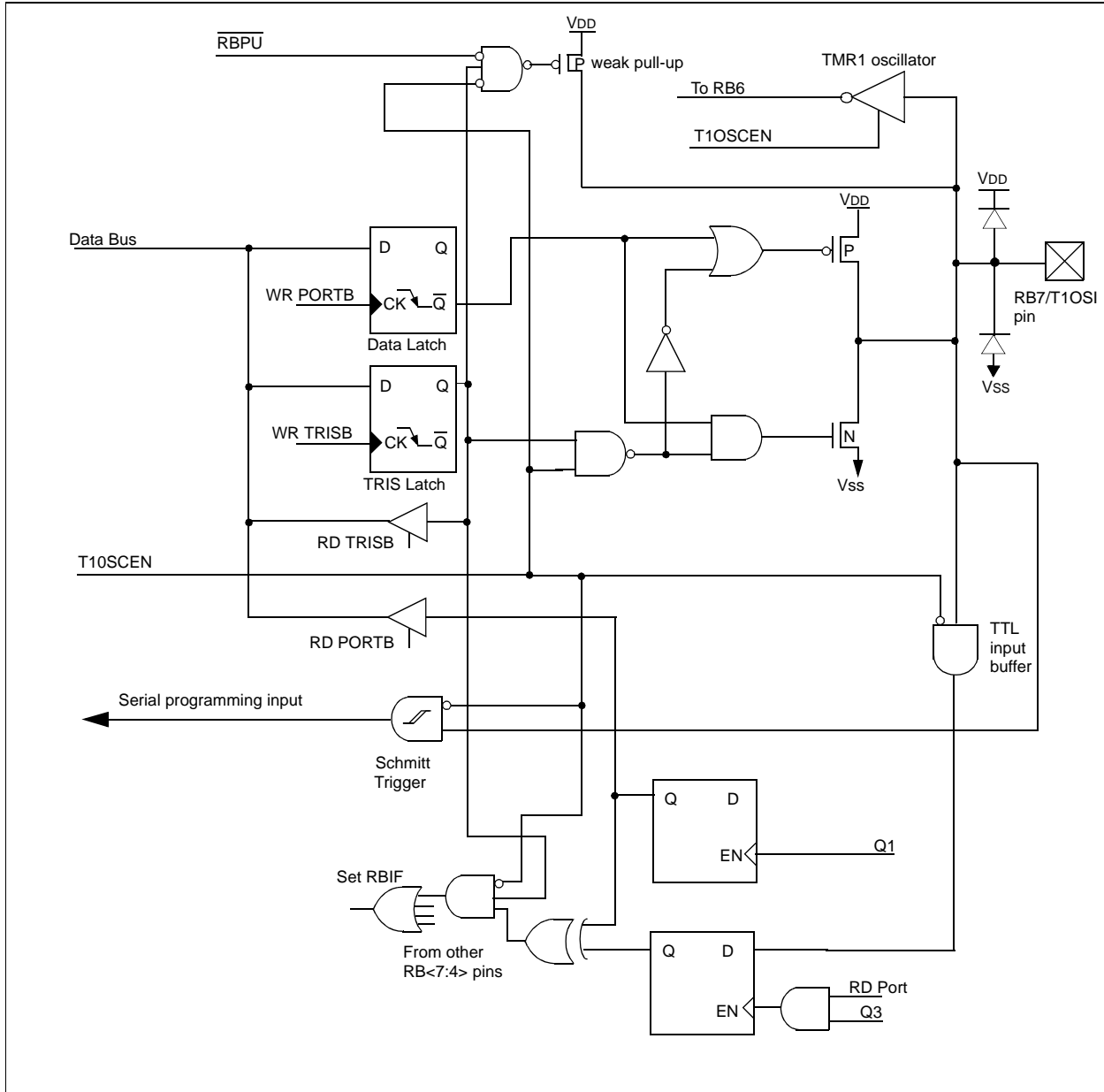


**FIGURE 5-14: BLOCK DIAGRAM OF RB6/T1OSO/T1CKI PIN**



# PIC16F62X

**FIGURE 5-15: BLOCK DIAGRAM OF THE RB7/T10SI PIN**



**TABLE 5-3: PORTB FUNCTIONS**

Name	Bit #	Buffer Type	Function
RB0/INT	bit0	TTL/ST <sup>(1)</sup>	Bi-directional I/O port/external interrupt. Can be software programmed for internal weak pull-up.
RB1/RX/DT	bit1	TTL/ST <sup>(3)</sup>	Bi-directional I/O port/ USART receive pin/synchronous data I/O. Can be software programmed for internal weak pull-up.
RB2/TX/CK	bit2	TTL/ST <sup>(3)</sup>	Bi-directional I/O port/ USART transmit pin/synchronous clock I/O. Can be software programmed for internal weak pull-up.
RB3/CCP1	bit3	TTL/ST <sup>(4)</sup>	Bi-directional I/O port/Capture/Compare/PWM I/O. Can be software programmed for internal weak pull-up.
RB4/PGM	bit4	TTL/ST <sup>(5)</sup>	Bi-directional I/O port/Low voltage programming input pin. Wake-up from SLEEP on pin change. Can be software programmed for internal weak pull-up. When low voltage programming is enabled, the interrupt on pin change and weak pull-up resistor are disabled.
RB5	bit5	TTL	Bi-directional I/O port/Wake-up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
RB6/T1OSO/T1CKI	bit6	TTL/ST <sup>(2)</sup>	Bi-directional I/O port/Timer1 oscillator output/Timer1 clock input. Wake up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
RB7/T1OSI	bit7	TTL/ST <sup>(2)</sup>	Bi-directional I/O port/Timer1 oscillator input. Wake up from SLEEP on pin change. Can be software programmed for internal weak pull-up.

Legend: ST = Schmitt Trigger, TTL = TTL input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.

**Note 2:** This buffer is a Schmitt Trigger input when used in serial programming mode.

**Note 3:** This buffer is a Schmitt Trigger I/O when used in USART/synchronous mode.

**Note 4:** This buffer is a Schmitt Trigger I/O when used in CCP mode.

**Note 5:** This buffer is a Schmitt Trigger input when used in low voltage program mode.

**TABLE 5-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORT**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
81h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: u = unchanged, x = unknown

**Note:** Shaded bits are not used by PORTB.

## 5.3 I/O Programming Considerations

### 5.3.1 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 bidirectional 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 re-written 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 content of the data latch may now be unknown.

Reading a 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, etc.) 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 5-2 shows the effect of two sequential read-modify-write instructions (ex., BCF, BSF, etc.) on an I/O port.

A pin actively outputting a Low or High should not be driven from external devices at the same time in order to change the level on this pin ("wired-or", "wired-and"). The resulting high output currents may damage the chip.

### EXAMPLE 5-2: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

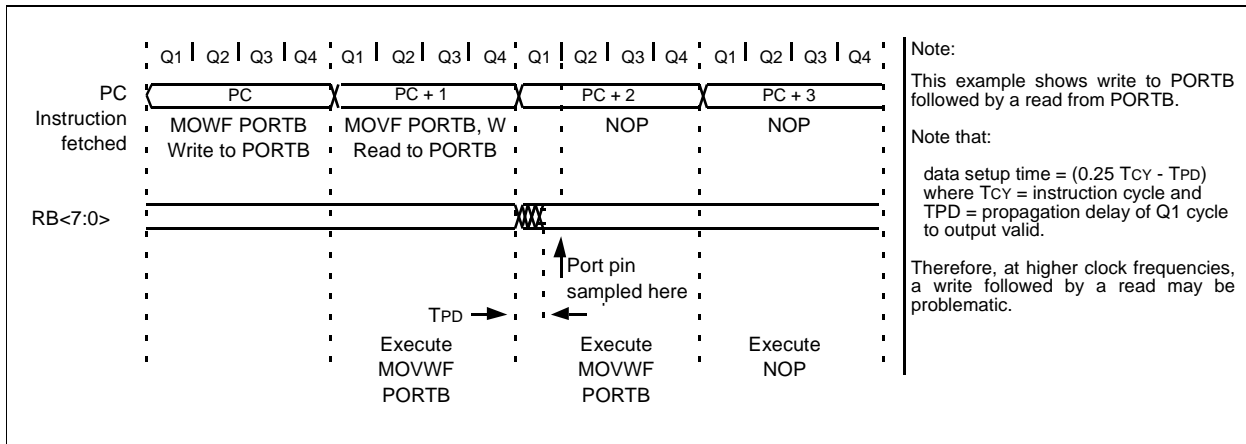
```

; Initial PORT settings:  PORTB<7:4> Inputs
;
;                               PORTB<3:0> Outputs
; PORTB<7:6> have external pull-up and are not
; connected to other circuitry
;
;                               PORT latch  PORT pins
;                               -----  -----
;
; BDF STATUS,RPO                ;
BCF PORTB, 7                    ; 01pp pppp  11pp pppp
BCF PORTB, 6                    ; 10pp pppp  11pp pppp
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 pppp. The 2nd BCF caused
; RB7 to be latched as the pin value (High).
    
```

### 5.3.2 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 5-16). 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 5-16: SUCCESSIVE I/O OPERATION**



## 6.0 TIMER0 MODULE

The Timer0 module timer/counter 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 6-1 is a simplified block diagram of the Timer0 module.

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In timer mode, the TMR0 will increment every instruction cycle (without prescaler). If Timer0 is written, the increment is inhibited for the following two cycles (Figure 6-2 and Figure 6-3). The user can work around this by writing an adjusted value to TMR0.

Counter mode is selected by setting the T0CS bit. In this mode Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the source edge (T0SE) control

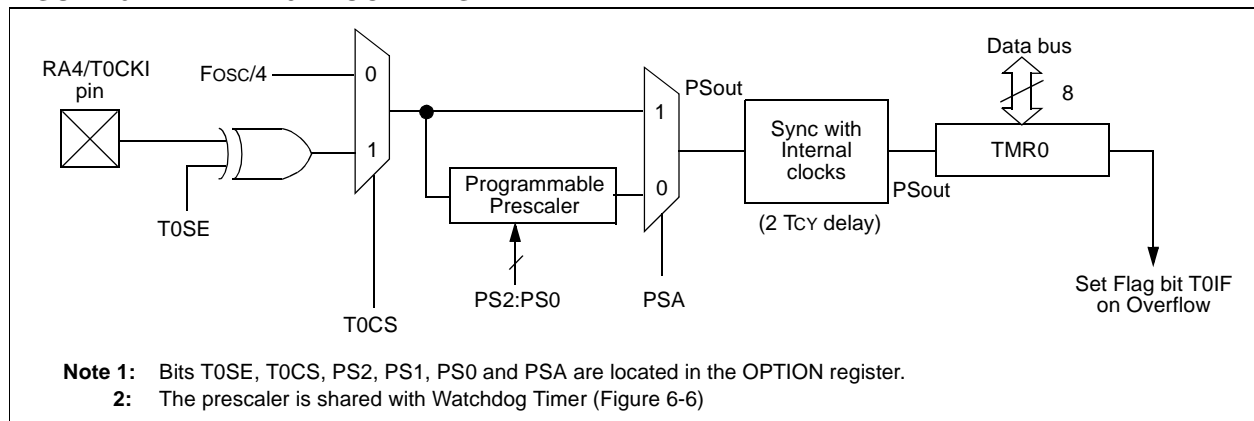
bit (OPTION<4>). Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.2.

The prescaler is shared between the Timer0 module and the Watchdog Timer. The prescaler assignment is controlled in software by the control bit PSA (OPTION<3>). Clearing the PSA bit will assign the prescaler to Timer0. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale value of 1:2, 1:4, ..., 1:256 are selectable. Section 6.3 details the operation of the prescaler.

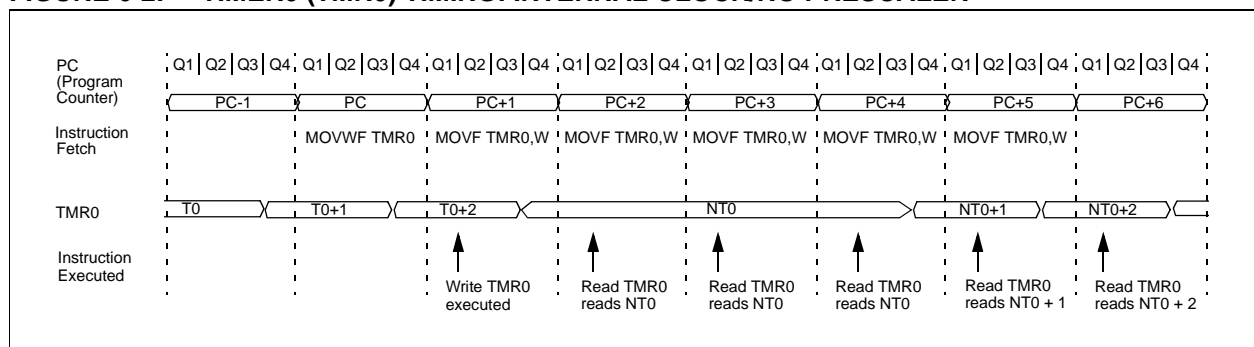
### 6.1 TIMER0 Interrupt

Timer0 interrupt is generated when the TMR0 register timer/counter overflows from FFh to 00h. This overflow sets the TOIF bit. The interrupt can be masked by clearing the TOIE bit (INTCON<5>). The TOIF bit (INTCON<2>) must be cleared in software by the Timer0 module interrupt service routine before re-enabling this interrupt. The Timer0 interrupt cannot wake the processor from SLEEP since the timer is shut off during SLEEP. See Figure 6-4 for Timer0 interrupt timing.

**FIGURE 6-1: TIMER0 BLOCK DIAGRAM**

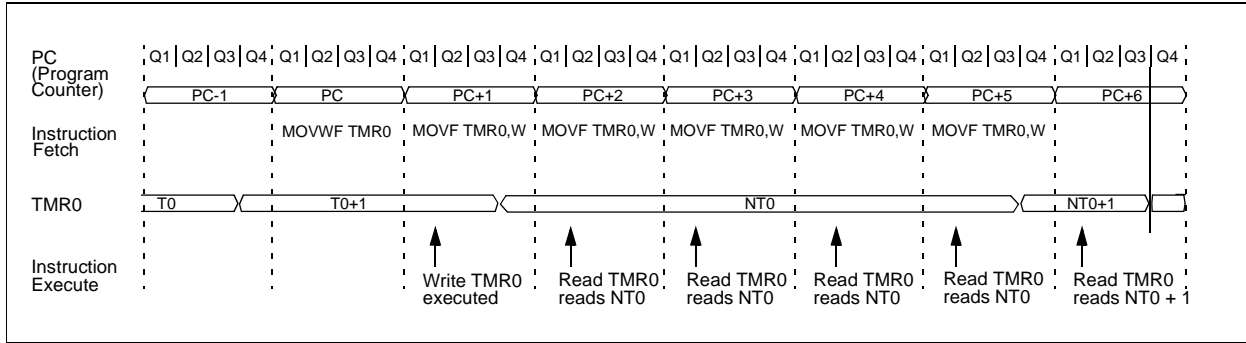


**FIGURE 6-2: TIMER0 (TMR0) TIMING: INTERNAL CLOCK/NO PRESCALER**

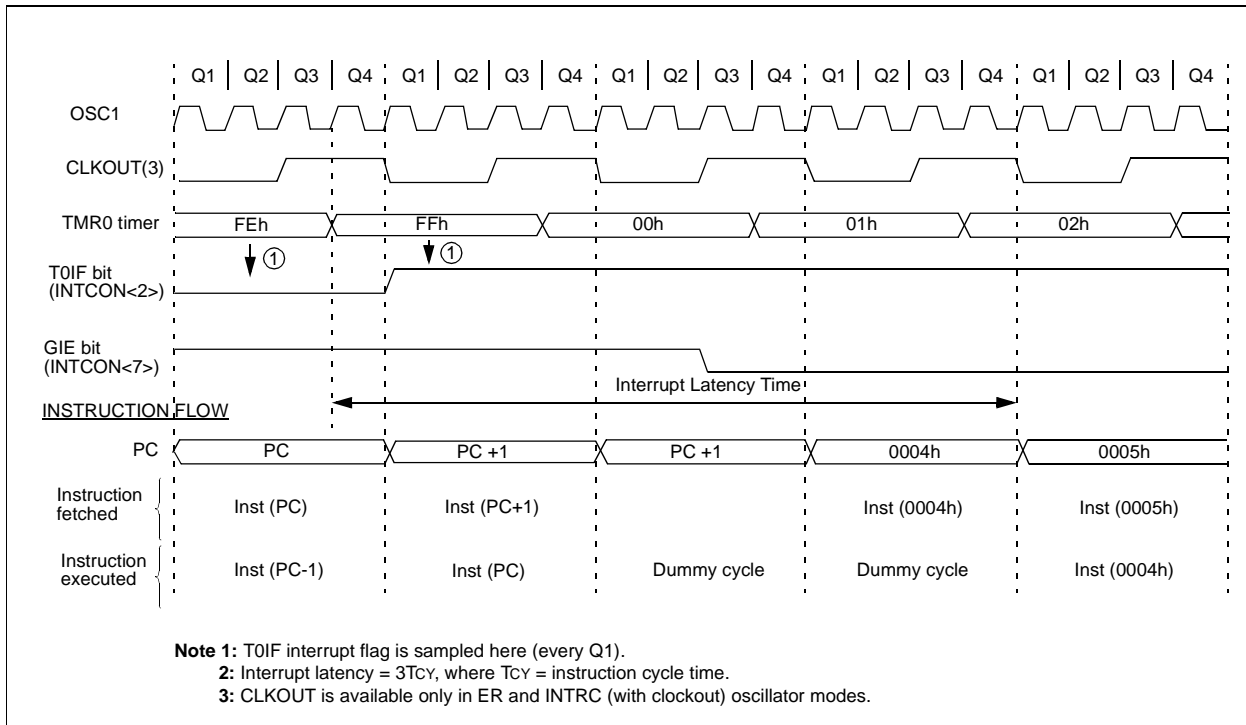


# PIC16F62X

**FIGURE 6-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2**



**FIGURE 6-4: TIMER0 INTERRUPT TIMING**



## 6.2 Using Timer0 with External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The external clock requirement is due to internal phase clock (TOSC) synchronization. Also, there is a delay in the actual incrementing of Timer0 after synchronization.

### 6.2.1 EXTERNAL CLOCK SYNCHRONIZATION

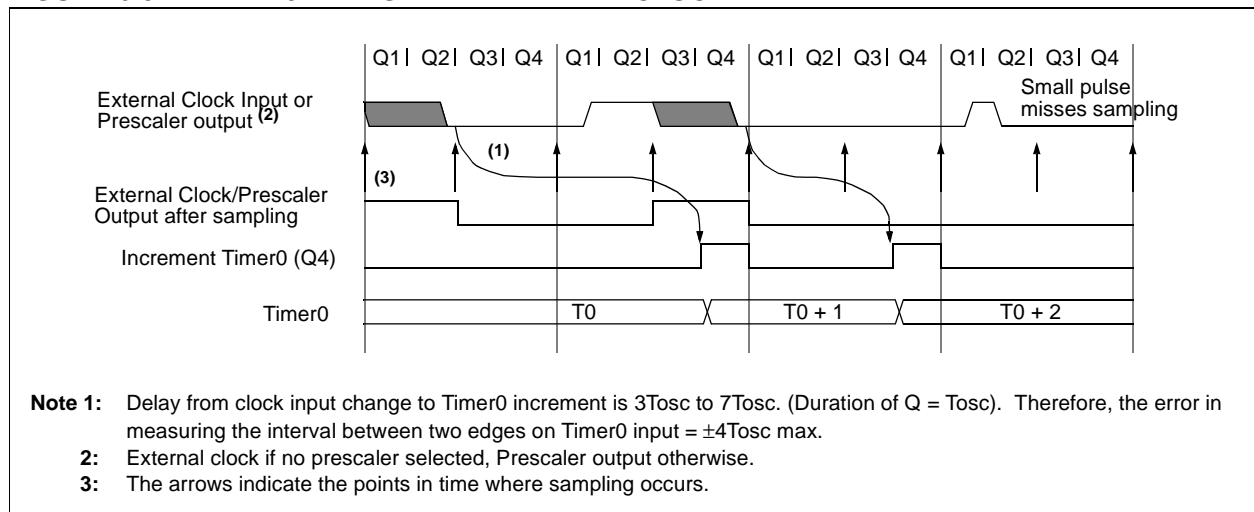
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 6-5). Therefore, it is necessary for T0CKI to be high for at least 2TOSC (and a small RC delay of 20 ns) and low for at least 2TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

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

### 6.2.2 TIMER0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the TMR0 is actually incremented. Figure 6-5 shows the delay from the external clock edge to the timer incrementing.

**FIGURE 6-5: TIMER0 TIMING WITH EXTERNAL CLOCK**



# PIC16F62X

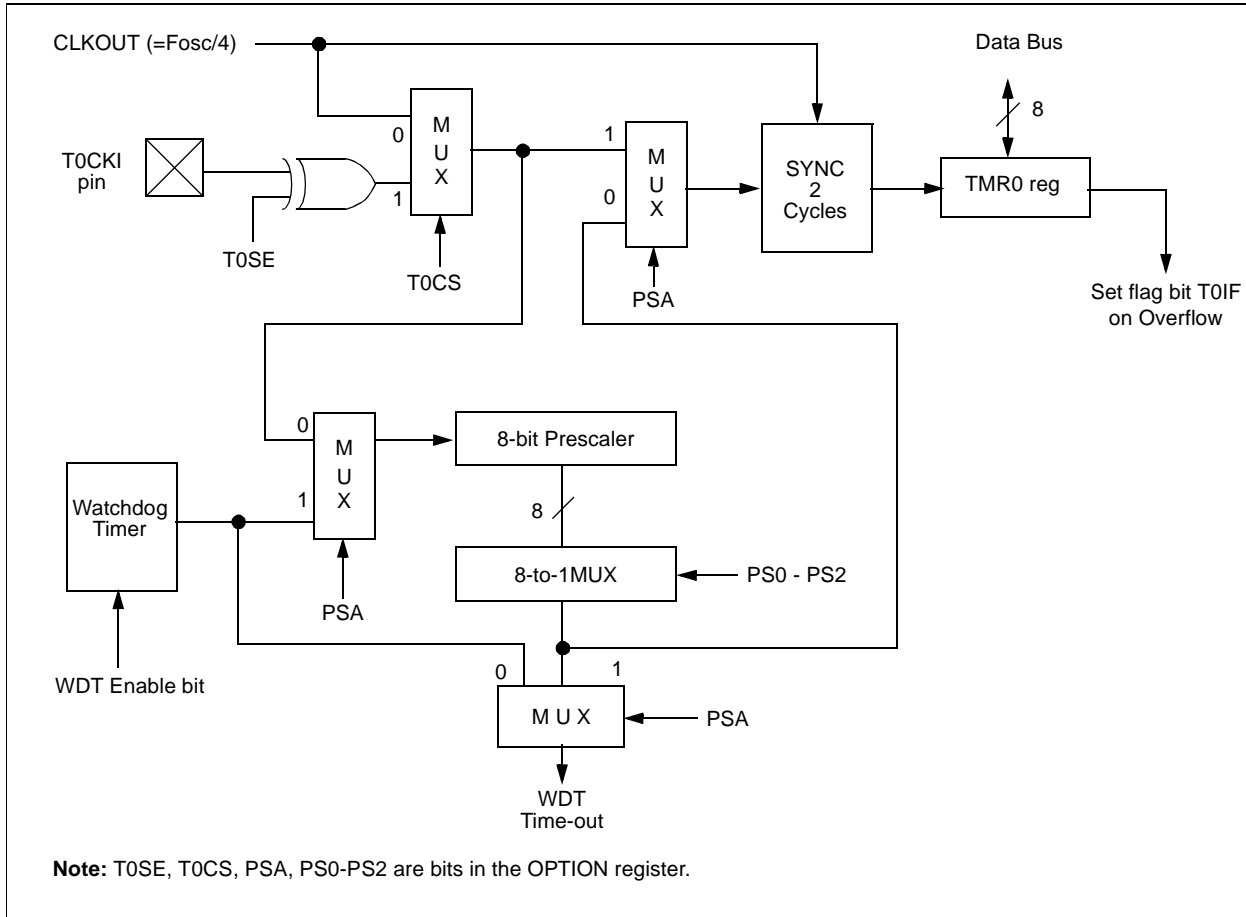
## 6.3 Prescaler

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

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF 1`, `MOVWF 1`, `BSF 1, x...` etc.) will clear the prescaler. 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**





## 6.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed “on the fly” during program execution). To avoid an unintended device RESET, the following instruction sequence (Example 6-1) must be executed when changing the prescaler assignment from Timer0 to WDT.

### EXAMPLE 6-1: CHANGING PRESCALER (TIMER0→WDT)

```

1.BCF STATUS, RP0 ;Skip if already in
; Bank 0
2.CLRWDT ;Clear WDT
3.CLRF TMR0 ;Clear TMR0 & Prescaler
4.BSF STATUS, RP0 ;Bank 1
5.MOVLW '00101111'b ;These 3 lines (5, 6, 7)
6.MOVWF OPTION ; are required only if
; desired PS<2:0> are
7.CLRWDT ; 000 or 001
8.MOVLW '00101xxx'b ;Set Postscaler to
9.MOVWF OPTION ; desired WDT rate
10.BCF STATUS, RP0 ;Return to Bank 0
    
```

To change prescaler from the WDT to the TMR0 module use the sequence shown in Example 6-2. This precaution must be taken even if the WDT is disabled.

### EXAMPLE 6-2: CHANGING PRESCALER (WDT→TIMER0)

```

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

**TABLE 6-1: REGISTERS ASSOCIATED WITH TIMER0**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
01h	TMR0	Timer0 module register								xxxx xxxx	uuuu uuuu
0Bh/8Bh/ 10Bh/18Bh	INTCON	GIE	—	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	0000 000u
81h	OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	TRISA7	TRISA6	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11-1 1111	11-1 1111

Legend: — = Unimplemented locations, read as '0', u = unchanged, x = unknown

**Note:** Shaded bits are not used by TMR0 module.

# PIC16F62X

## 7.0 TIMER1 MODULE

The Timer1 module 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 enabled/disabled by setting/clearing control bit TMR1ON (T1CON<0>).

Timer1 also has an internal "reset input". This reset can be generated by the CCP module (Section 10.0). Register 7-1 shows the Timer1 control register.

For the PIC16F627 and PIC16F628, when the Timer1 oscillator is enabled (T1OSCEN is set), the RB7/T1OSI and RB6/T1OSO/T1CKI pins become inputs. That is, the TRISB<7:6> value is ignored.

**REGISTER 7-1: 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	$\overline{T1SYNC}$	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:  **$\overline{T1SYNC}$ :** 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 RB6/T1OSO/T1CKI (on the rising edge)  
0 = Internal clock (FOSC/4)

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

## 7.1 Timer1 Operation in Timer Mode

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

## 7.2 Timer1 Operation in Synchronized Counter Mode

Counter mode is selected by setting bit TMR1CS. In this mode the timer increments on every rising edge of clock input on pin RB7/T1OSI when bit T1OSCEN is set or pin RB6/T1OSO/T1CKI when bit T1OSCEN is cleared.

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

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

### 7.2.1 EXTERNAL CLOCK INPUT TIMING FOR SYNCHRONIZED COUNTER MODE

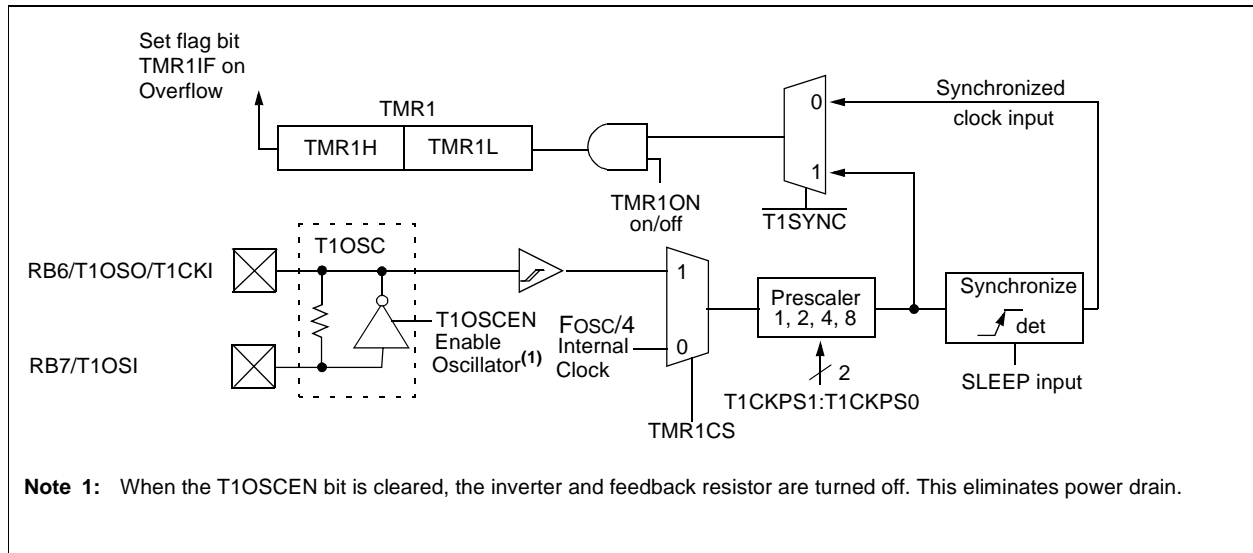
When an external clock input is used for Timer1 in synchronized counter mode, it must meet certain requirements. The external clock requirement is due to

internal phase clock ( $T_{osc}$ ) synchronization. Also, there is a delay in the actual incrementing of TMR1 after synchronization.

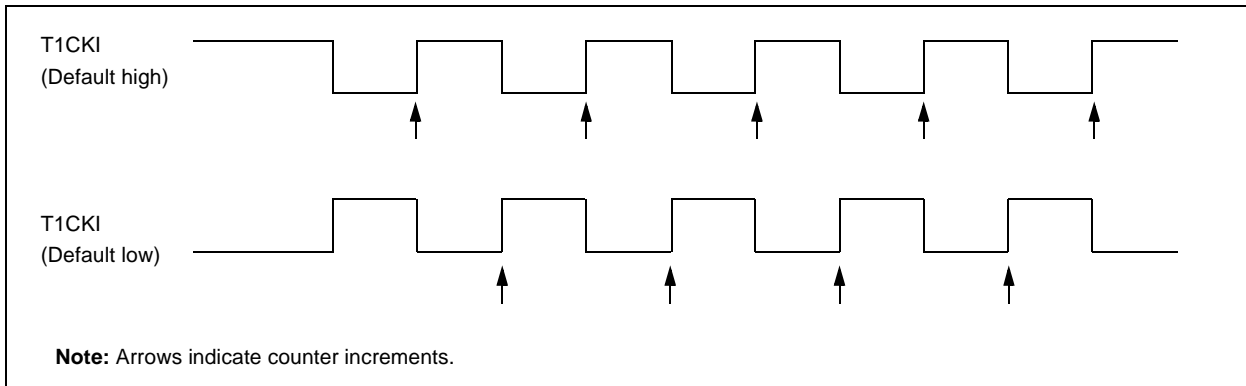
When the prescaler is 1:1, the external clock input is the same as the prescaler output. The synchronization of T1CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks. Therefore, it is necessary for T1CKI to be high for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

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

**FIGURE 7-1: TIMER1 BLOCK DIAGRAM**



**FIGURE 7-2: TIMER1 INCREMENTING EDGE**



### 7.3 Timer1 Operation in Asynchronous Counter Mode

If control bit  $\overline{T1SYNC}$  (T1CON<2>) is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during SLEEP and can generate an interrupt on overflow which will wake-up the processor. However, special precautions in software are needed to read/write the timer (Section 7.3.2).

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

#### 7.3.1 EXTERNAL CLOCK INPUT TIMING WITH UNSYNCHRONIZED CLOCK

If control bit  $\overline{T1SYNC}$  is set, the timer will increment completely asynchronously. The input clock must meet certain minimum high time and low time requirements. Refer to the appropriate Electrical Specifications Section, timing parameters 45, 46, and 47.

#### 7.3.2 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running, from an external asynchronous clock, will guarantee 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 7-1 is an example routine to read the 16-bit timer value. This is useful if the timer cannot be stopped.

### EXAMPLE 7-1: READING A 16-BIT FREE-RUNNING TIMER

```

; All interrupts are disabled
MOVWF TMR1H, W ;Read high byte
MOVWF TMPH ;
MOVWF TMR1L, W ;Read low byte
MOVWF TMPL ;
MOVWF 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.
;
MOVWF TMR1H, W ;Read high byte
MOVWF TMPH ;
MOVWF TMR1L, W ;Read low byte
MOVWF TMPL ;
; Re-enable the Interrupt (if required)
CONTINUE ;Continue with your code
    
```

### 7.4 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 7-1 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 7-1: 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
<b>These values are for design guidance only.</b>			

## 7.5 Resetting Timer1 using a CCP Trigger Output

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

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

Timer1 must be configured for either timer or synchronized counter mode to take advantage of this feature. If Timer1 is running in asynchronous counter mode, this reset operation may not work.

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

In this mode of operation, the CCPRxH:CCPRxL registers pair effectively becomes the period register for Timer1.

## 7.6 Resetting of Timer1 Register Pair (TMR1H, TMR1L)

TMR1H and TMR1L registers are not reset to 00h on a POR or any other reset except by the CCP1 special event triggers.

T1CON register is reset to 00h on a Power-on Reset or a Brown-out Reset, which shuts off the timer and leaves a 1:1 prescale. In all other resets, the register is unaffected.

## 7.7 Timer1 Prescaler

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

**TABLE 7-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other resets
0Bh/8Bh/ 10Bh/18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	--00 0000	--uu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by the Timer1 module.

# PIC16F62X

## 8.0 TIMER2 MODULE

Timer2 is an 8-bit timer with a prescaler and a postscaler. It can be used as the PWM time-base for PWM mode of the CCP module. The TMR2 register is readable and writable, and is cleared on any device reset.

The input clock ( $F_{osc}/4$ ) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits T2CKPS1:T2CKPS0 (T2CON<1:0>).

The Timer2 module has an 8-bit period register PR2. Timer2 increments from 00h until it matches PR2 and then resets to 00h on the next increment cycle. PR2 is a readable and writable register. The PR2 register is initialized to FFh upon reset.

The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit TMR2IF, (PIR1<1>)).

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

Register 8-1 shows the Timer2 control register.

## 8.1 Timer2 Prescaler and Postscaler

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

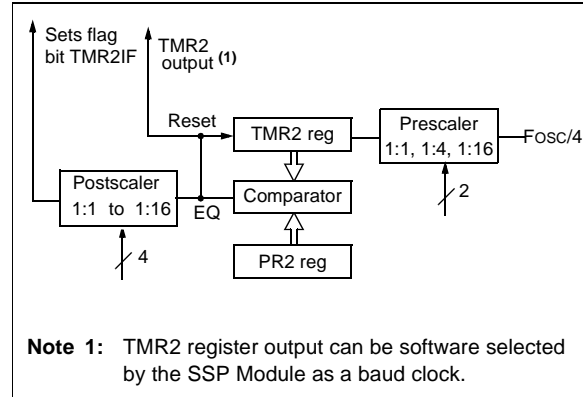
- a write to the TMR2 register
- a write to the T2CON register
- any device reset (Power-on Reset,  $\overline{MCLR}$  reset, Watchdog Timer reset, or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

## 8.2 Output of TMR2

The output of TMR2 (before the postscaler) is fed to the Synchronous Serial Port module which optionally uses it to generate shift clock.

**FIGURE 8-1: TIMER2 BLOCK DIAGRAM**



## REGISTER 8-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0		
bit7							bit0		

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

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

bit 6-3: **TOUTPS3:TOUTPS0:** Timer2 Output Postscale Select bits  
 0000 = 1:1 Postscale  
 0001 = 1:2 Postscale  
 •  
 •  
 •  
 1111 = 1:16 Postscale

bit 2: **TMR2ON:** Timer2 On bit  
 1 = Timer2 is on  
 0 = Timer2 is off

bit 1-0: **T2CKPS1:T2CKPS0:** Timer2 Clock Prescale Select bits  
 00 = Prescaler is 1  
 01 = Prescaler is 4  
 1x = Prescaler is 16

**TABLE 8-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other resets
0Bh/8Bh/ 10Bh/18Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
11h	TMR2	Timer2 module's register								0000 0000	0000 0000
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
92h	PR2	Timer2 Period Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by the Timer2 module.

# PIC16F62X

---

NOTES:



## 9.0 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 11.0) can also be an input to the comparators.

The CMCON register, shown in Register 9-1, controls the comparator input and output multiplexers. A block diagram of the comparator is shown in Figure 9-1.

### REGISTER 9-1: CMCON REGISTER (ADDRESS 01Fh)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit7							bit0

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

bit 7: **C2OUT**: Comparator 2 output  
 When C2INV=0;  
 1 = C2 VIN+ > C2 VIN-  
 0 = C2 VIN+ < C2 VIN-  
  
 When C2INV=1;  
 0 = C2 VIN+ > C2 VIN-  
 1 = C2 VIN+ < C2 VIN-

bit 6: **C1OUT**: Comparator 1 output  
 When C1INV=0;  
 1 = C1 VIN+ > C1 VIN-  
 0 = C1 VIN+ < C1 VIN-  
  
 When C1INV=1;  
 0 = C1 VIN+ > C1 VIN-  
 1 = C1 VIN+ < C1 VIN-

bit 5: **C2INV**: Comparator 2 output inversion  
 1 = C2 Output inverted  
 0 = C2 Output not inverted

bit 4: **C1INV**: Comparator 1 output inversion  
 1 = C1 Output inverted  
 0 = C1 Output not inverted

bit 3: **CIS**: Comparator Input Switch  
  
 When CM2:CM0 = 001:  
 Then:  
 1 = C1 VIN- connects to RA3  
 0 = C1 VIN- connects to RA0  
  
 When CM2:CM0 = 010:  
 Then:  
 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: **CM2:CM0**: Comparator mode  
 Figure 9-1 shows the comparator modes and CM2:CM0 bit settings.

# PIC16F62X

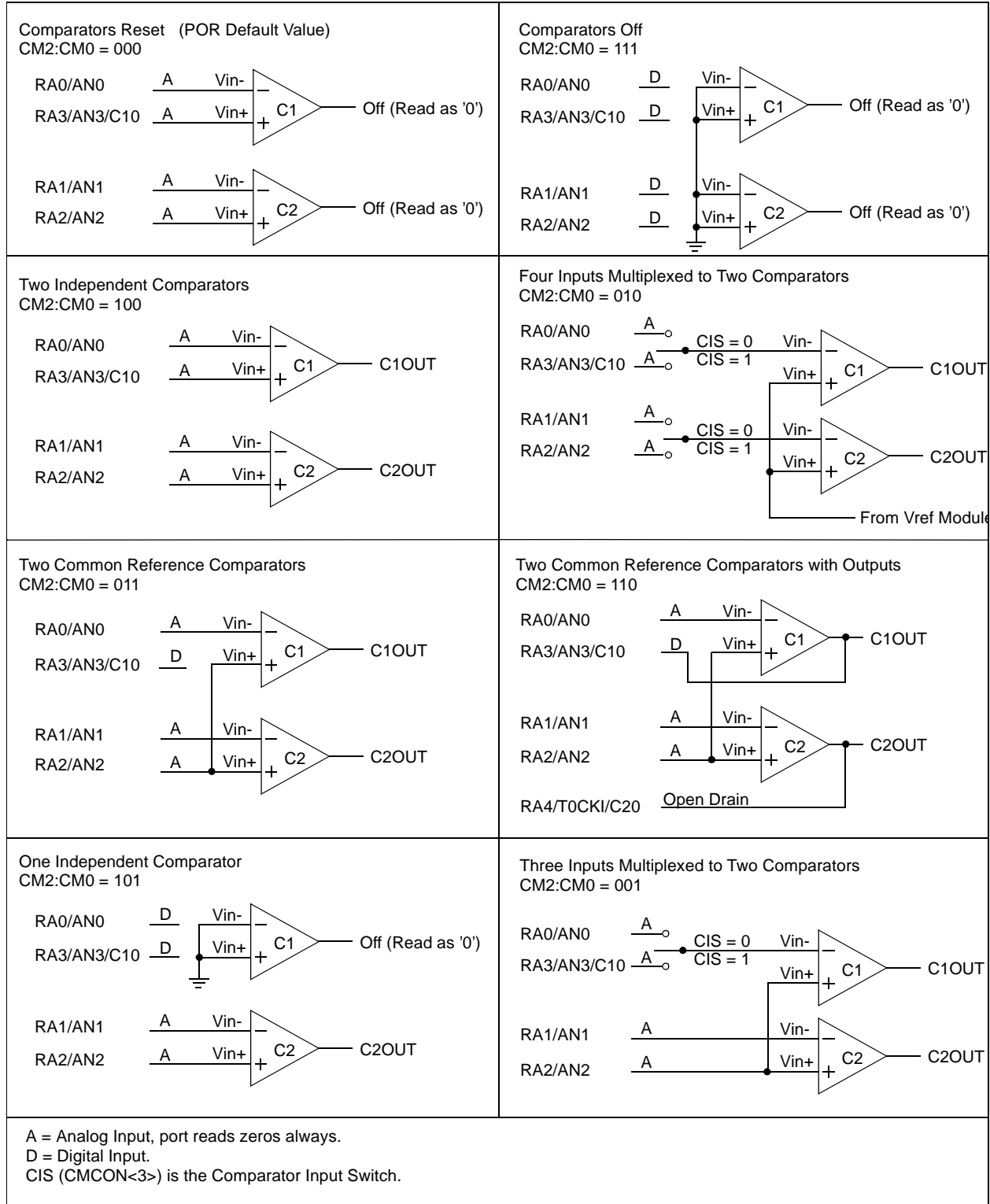
## 9.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select the mode. Figure 9-1 shows the eight possible modes. The TRISA register controls the data direction of the comparator pins for each mode. If the comparator

mode is changed, the comparator output level may not be valid for the specified mode change delay shown in Table 12-2.

**Note:** Comparator interrupts should be disabled during a comparator mode change otherwise a false interrupt may occur.

**FIGURE 9-1: COMPARATOR I/O OPERATING MODES**



The code example in Example 9-1 depicts the steps required to configure the comparator module. RA3 and RA4 are configured as digital output. RA0 and RA1 are configured as the V- inputs and RA2 as the V+ input to both comparators.

## EXAMPLE 9-1: INITIALIZING COMPARATOR MODULE

```

FLAG_REG EQU      0X20
CLRF   FLAG_REG   ;Init flag register
CLRF   PORTA      ;Init PORTA
MOVF   CMCON, W   ;Load comparator bits
ANDLW  0xC0       ;Mask comparator bits
IORWF  FLAG_REG,F ;Store bits in flag register
MOVLW  0x03       ;Init comparator mode
MOVWF  CMCON      ;CM<2:0> = 011
BSF    STATUS,RP0 ;Select Bank1
MOVLW  0x07       ;Initialize data direction
MOVWF  TRISA      ;Set RA<2:0> as inputs
                          ;RA<4:3> as outputs
                          ;TRISA<7:5> always read '0'

BCF    STATUS,RP0 ;Select Bank 0
CALL   DELAY 10   ;10µs delay
MOVF   CMCON,F   ;Read CMCON to end change condition
BCF    PIR1,CMIF ;Clear pending interrupts
BSF    STATUS,RP0 ;Select Bank 1
BSF    PIE1,CMIE ;Enable comparator interrupts
BCF    STATUS,RP0 ;Select Bank 0
BSF    INTCON,PEIE ;Enable peripheral interrupts
BSF    INTCON,GIE ;Global interrupt enable
    
```

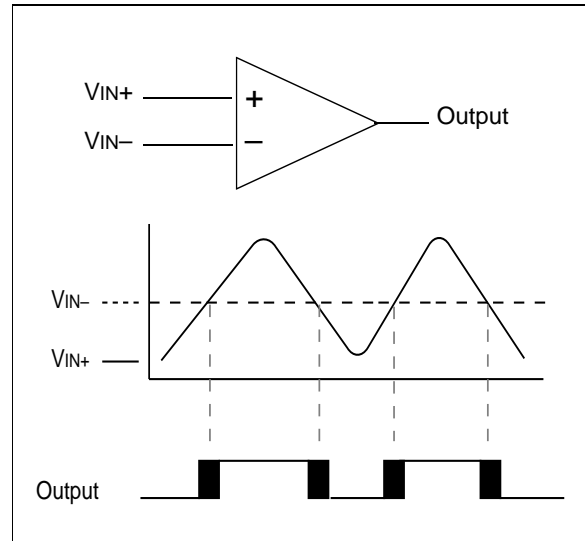
## 9.2 Comparator Operation

A single comparator is shown in Figure 9-2 along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 9-2 represent the uncertainty due to input offsets and response time.

## 9.3 Comparator Reference

An external or internal reference signal may be used depending on the comparator operating mode. The analog signal that is present at VIN- is compared to the signal at VIN+, and the digital output of the comparator is adjusted accordingly (Figure 9-2).

FIGURE 9-2: SINGLE COMPARATOR



### 9.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD, and can be applied to either pin of the comparator(s).

### 9.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference for the comparators. Section 13, Instruction Sets, contains a detailed description of the Voltage Reference Module that provides this signal. The internal reference signal is used when the comparators are in mode CM<2:0>=010 (Figure 9-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

# PIC16F62X

## 9.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output is guaranteed to have a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise the maximum delay of the comparators should be used (Table 12-2).

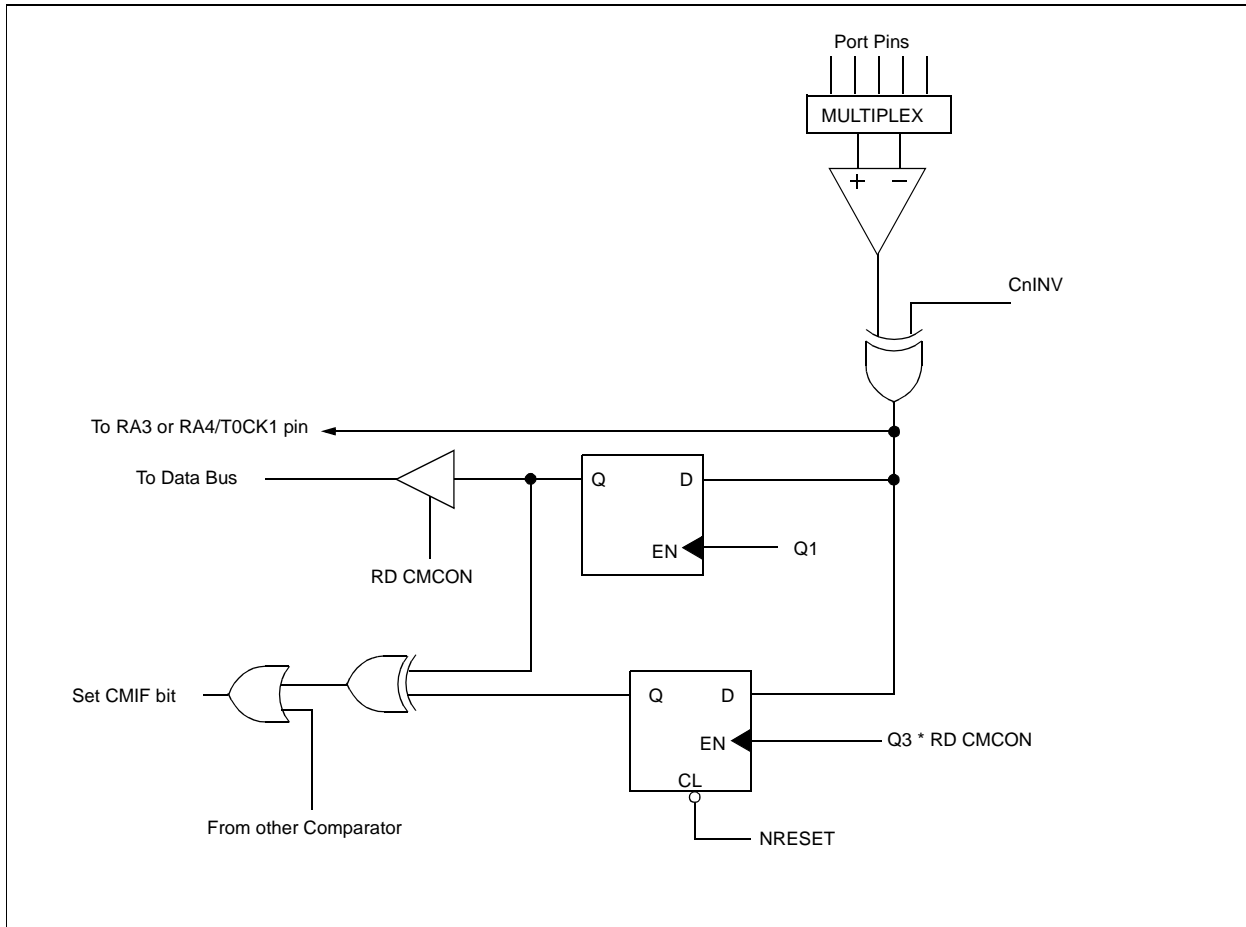
## 9.5 Comparator Outputs

The comparator outputs are read through the CMCON register. These bits are read only. The comparator outputs may also be directly output to the RA3 and RA4 I/O pins. When the CM<2:0> = 110 or 001, multiplexors in the output path of the RA3 and RA4/T0CK1 pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 9-3 shows the comparator output block diagram.

The TRISA bits will still function as an output enable/disable for the RA3 and RA4/T0CK1 pins while in this mode.

- Note 1:** When reading the PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert an analog input according to the Schmitt Trigger input specification.
- 2:** Analog levels on any pin that is defined as a digital input may cause the input buffer to consume more current than is specified.

**FIGURE 9-3: MODIFIED COMPARATOR OUTPUT BLOCK DIAGRAM**



## 9.6 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.

## 9.7 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.

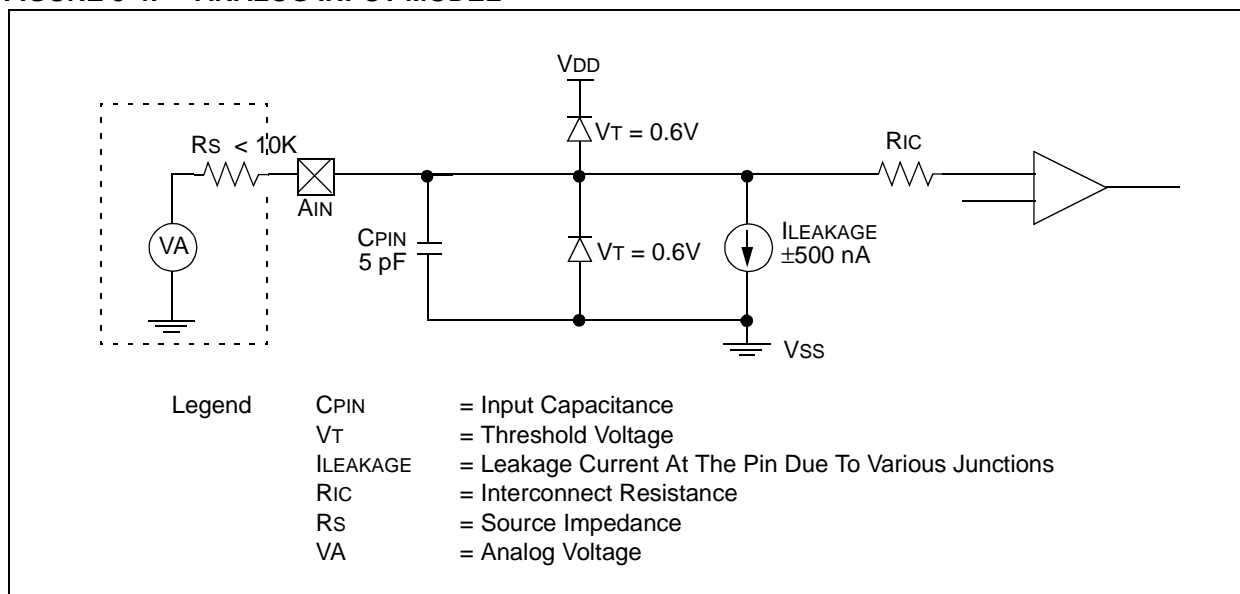
## 9.8 Effects of a RESET

A device reset forces the CMCON register to its reset state. This forces the comparator module to be in the comparator reset mode, CM2:CM0 = 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.

## 9.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 9-4. 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 latch-up 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 9-4: ANALOG INPUT MODEL**



# PIC16F62X

**TABLE 9-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1NV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	000- 0000
0Bh/8Bh/ 10Bh/18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
85h	TRISA	TRISA7	TRISA6	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11-1 1111	11-1 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as "0"

## 10.0 CAPTURE/COMPARE/PWM (CCP) MODULE

The CCP (Capture/Compare/PWM) module contains a 16-bit register which can operate as a 16-bit capture register, as a 16-bit compare register or as a PWM master/slave Duty Cycle register. Table 10-1 shows the timer resources of the CCP module modes.

### CCP1 Module

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

Additional information on the CCP module is available in the PICmicro™ Mid-Range Reference Manual, (DS33023).

**TABLE 10-1 CCP MODE - TIMER RESOURCE**

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

### REGISTER 10-1: CCP1CON REGISTER (ADDRESS 17h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	
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: **CCP1X:CCP1Y:** PWM Least Significant bits  
 Capture Mode: Unused  
 Compare Mode: Unused  
 PWM Mode: These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0: **CCP1M3:CCP1M0:** CCPx Mode Select bits  
 0000 = Capture/Compare/PWM off (resets CCP1 module)  
 0100 = Capture mode, every falling edge  
 0101 = Capture mode, every rising edge  
 0110 = Capture mode, every 4th rising edge  
 0111 = Capture mode, every 16th rising edge  
 1000 = Compare mode, set output on match (CCP1IF bit is set)  
 1001 = Compare mode, clear output on match (CCP1IF bit is set)  
 1010 = Compare mode, generate software interrupt on match (CCP1IF bit is set, CCP1 pin is unaffected)  
 1011 = Compare mode, trigger special event (CCP1IF bit is set; CCP1 resets TMR1)  
 11xx = PWM mode

# PIC16F62X

## 10.1 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on pin RB3/CCP1. An event is defined as:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

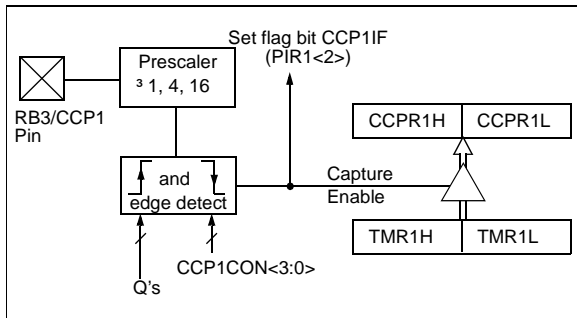
An event is selected by control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit CCP1IF (PIR1<2>) is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value will be lost.

### 10.1.1 CCP PIN CONFIGURATION

In Capture mode, the RB3/CCP1 pin should be configured as an input by setting the TRISB<3> bit.

**Note:** If the RB3/CCP1 is configured as an output, a write to the port can cause a capture condition.

**FIGURE 10-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**



### 10.1.2 TIMER1 MODE SELECTION

Timer1 must be running in timer mode or synchronized counter mode for the CCP module to use the capture feature. In asynchronous counter mode, the capture operation may not work.

### 10.1.3 SOFTWARE INTERRUPT

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

### 10.1.4 CCP PRESCALER

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

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore the first capture may be from a non-zero prescaler. Example 10-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

### EXAMPLE 10-1: CHANGING BETWEEN CAPTURE PRESCALERS

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



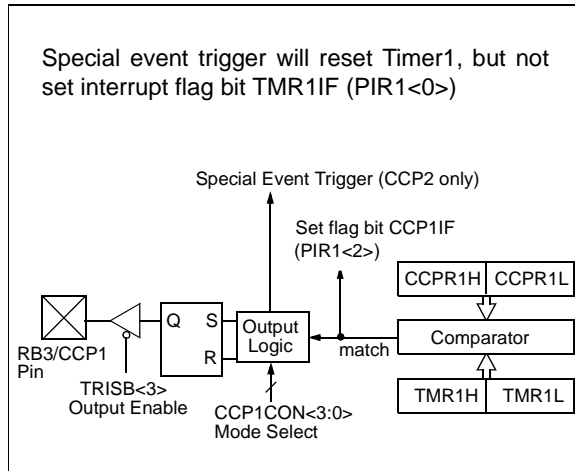
## 10.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the RB3/CCP1 pin is:

- driven High
- driven Low
- remains Unchanged

The action on the pin is based on the value of control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). At the same time, interrupt flag bit CCP1IF is set.

**FIGURE 10-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



### 10.2.1 CCP PIN CONFIGURATION

The user must configure the RB3/CCP1 pin as an output by clearing the TRISB<3> bit.

**Note:** Clearing the CCP1CON register will force the RB3/CCP1 compare output latch to the default low level. This is not the data latch.

### 10.2.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

### 10.2.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

### 10.2.4 SPECIAL EVENT TRIGGER

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

The special event trigger output of CCP1 resets the TMR1 register pair. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

**TABLE 10-2 REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, AND TIMER1**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other resets
0Bh/8Bh/10Bh/18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
87h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	--uu uuuu
15h	CCPR1L	Capture/Compare/PWM register1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM register1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by Capture and Timer1.

# PIC16F62X

## 10.3 PWM Mode

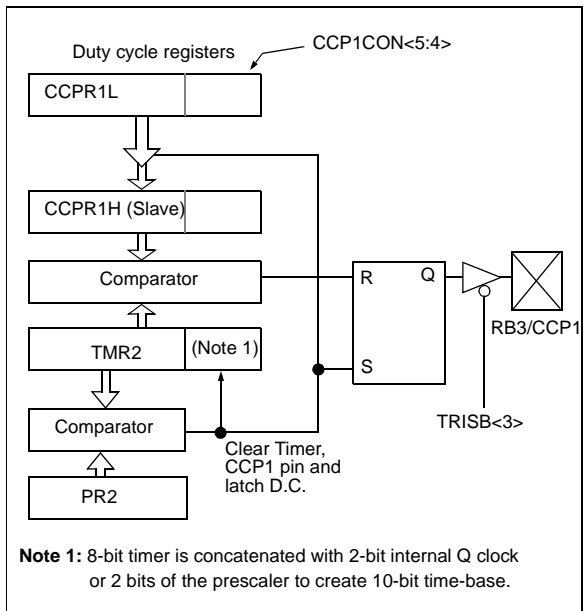
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISB<3> bit must be cleared to make the CCP1 pin an output.

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

Figure 10-3 shows a simplified block diagram of the CCP module in PWM mode.

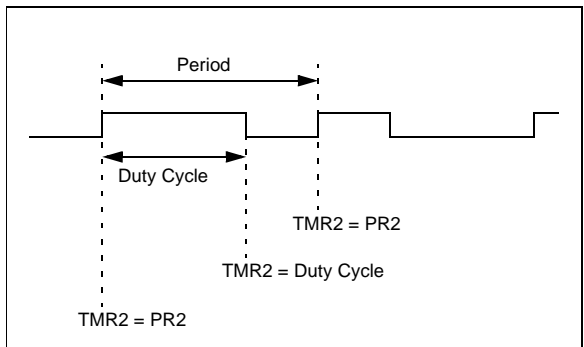
For a step by step procedure on how to set up the CCP module for PWM operation, see Section 10.3.3.

**FIGURE 10-3: SIMPLIFIED PWM BLOCK DIAGRAM**



A PWM output (Figure 10-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

**FIGURE 10-4: PWM OUTPUT**



### 10.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

$$\text{PWM period} = [(PR2) + 1] \cdot 4 \cdot T_{osc} \cdot (\text{TMR2 prescale value})$$

PWM frequency is defined as  $1 / [\text{PWM period}]$ .

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

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

**Note:** The Timer2 postscaler (see Section 8.0) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

### 10.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available: the CCPR1L contains the eight MSBs and the CCP1CON<5:4> contains the two LSBs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{osc} \cdot (\text{TMR2 prescale value})$$

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

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2 concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

Maximum PWM resolution (bits) for a given PWM frequency:

$$= \frac{\log \left( \frac{F_{osc}}{F_{pwm}} \right)}{\log (2)} \text{ bits}$$

**Note:** If the PWM duty cycle value is longer than the PWM period the CCP1 pin will not be cleared.

For an example PWM period and duty cycle calculation, see the PICmicro™ Mid-Range Reference Manual (DS33023).

## 10.3.3 SET-UP FOR PWM OPERATION

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

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

**TABLE 10-3 EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 20 MHz**

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	5.5

**TABLE 10-4 REGISTERS ASSOCIATED WITH PWM AND TIMER2**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other resets
0Bh/8Bh/10Bh/18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
87h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
11h	TMR2	Timer2 module's register								0000 0000	0000 0000
92h	PR2	Timer2 module's period register								1111 1111	1111 1111
12h	T2CON	—	TOUTPS 3	TOUTPS 2	TOUTPS 1	TOUTPS 0	TMR2ON	T2CKPS 1	T2CKPS 0	-000 0000	uuuu uuuu
15h	CCPR1L	Capture/Compare/PWM register1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM register1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by PWM and Timer2.

# PIC16F62X

---

NOTES:

## 11.0 VOLTAGE REFERENCE MODULE

The Voltage Reference is a 16-tap resistor ladder network that provides a selectable voltage reference. The resistor ladder is segmented to provide two ranges of VREF values and has a power-down function to conserve power when the reference is not being used. The VRCON register controls the operation of the reference as shown in Figure 11-1. The block diagram is given in Figure 11-2.

### 11.1 Configuring the Voltage Reference

The Voltage Reference can output 16 distinct voltage levels for each range.

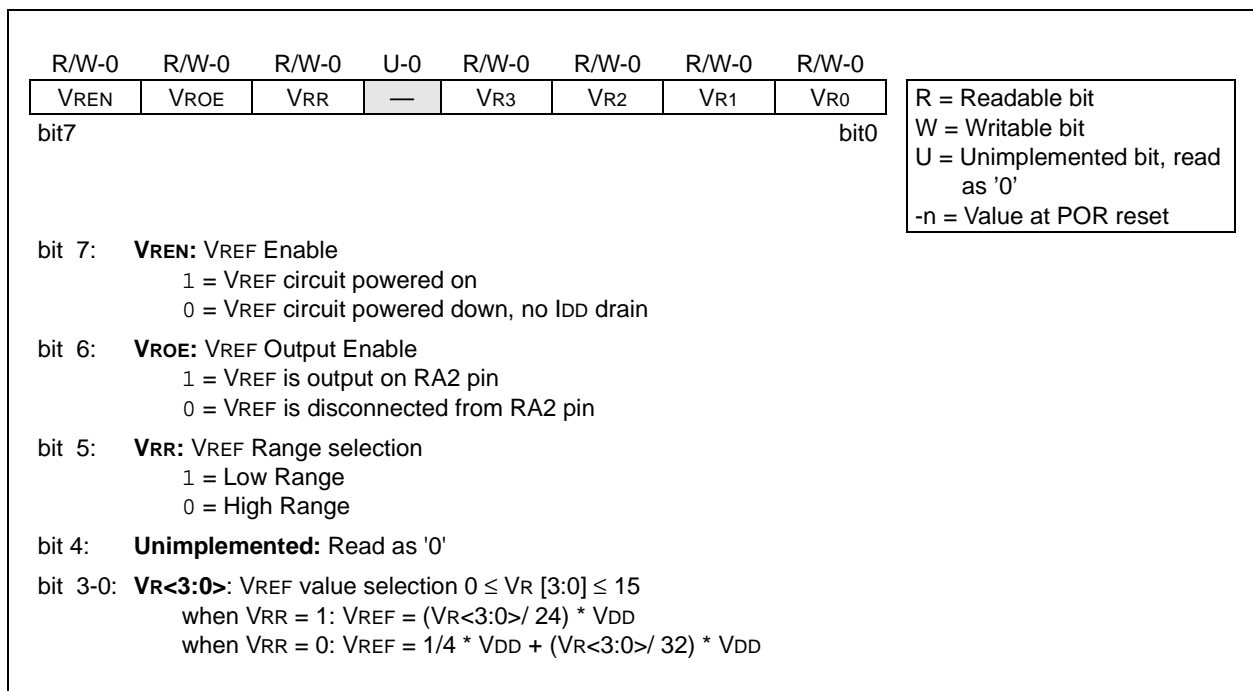
The equations used to calculate the output of the Voltage Reference are as follows:

$$\text{if } VRR = 1: VREF = (VR<3:0>/24) \times VDD$$

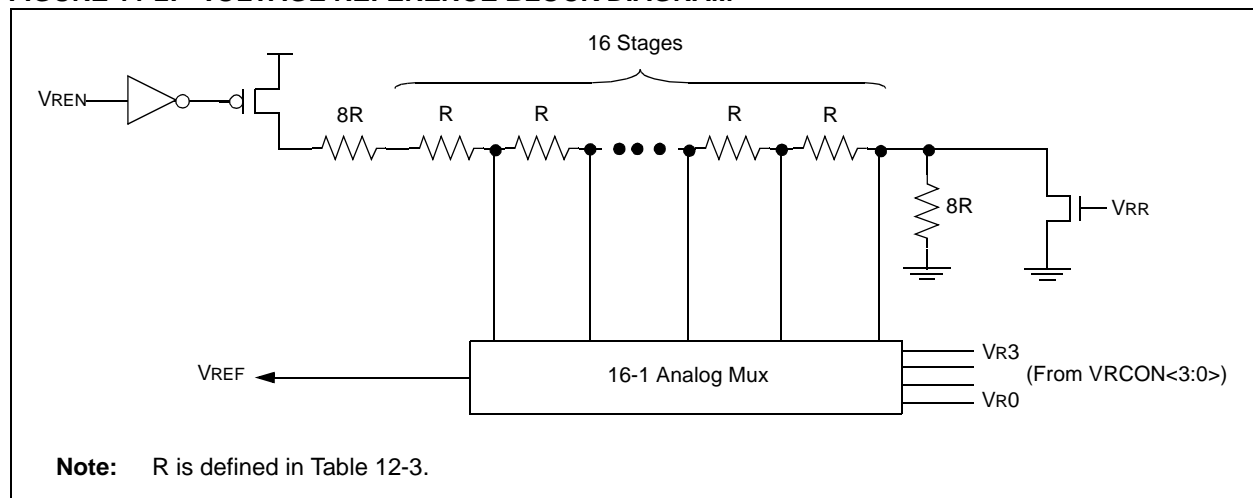
$$\text{if } VRR = 0: VREF = (VDD \times 1/4) + (VR<3:0>/32) \times VDD$$

The setting time of the Voltage Reference must be considered when changing the VREF output (Table 12-2). Example 11-1 shows an example of how to configure the Voltage Reference for an output voltage of 1.25V with VDD = 5.0V.

**FIGURE 11-1: VRCON REGISTER (ADDRESS 9Fh)**



**FIGURE 11-2: VOLTAGE REFERENCE BLOCK DIAGRAM**



## EXAMPLE 11-1: VOLTAGE REFERENCE CONFIGURATION

```

MOVLW    0x02           ; 4 Inputs Muxed
MOVWF    CMCON          ; to 2 comps.
BSF      STATUS,RP0    ; go to Bank 1
MOVLW    0x07           ; RA3-RA0 are
MOVWF    TRISA          ; outputs
MOVLW    0xA6           ; enable VREF
MOVWF    VRCON          ; low range
                                ; set VR<3:0>=6
BCF      STATUS,RP0    ; go to Bank 0
CALL     DELAY10        ; 10µs delay
    
```

### 11.2 Voltage Reference Accuracy/Error

The full range of VSS to VDD cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 11-2) keep VREF from approaching VSS or VDD. The Voltage Reference is VDD derived and therefore, the VREF output changes with fluctuations in VDD. The tested absolute accuracy of the Voltage Reference can be found in Table 17-2.

### 11.3 Operation During Sleep

When the device wakes up from sleep through an interrupt or a Watchdog Timer time-out, the contents of the VRCON register are not affected. To minimize current consumption in SLEEP mode, the Voltage Reference should be disabled.

### 11.4 Effects of a Reset

A device reset disables the Voltage Reference by clearing bit VREN (VRCON<7>). This reset also disconnects the reference from the RA2 pin by clearing bit VROE (VRCON<6>) and selects the high voltage range by clearing bit VRR (VRCON<5>). The VREF value select bits, VRCON<3:0>, are also cleared.

### 11.5 Connection Considerations

The Voltage Reference Module operates independently of the comparator module. The output of the reference generator may be connected to the RA2 pin if the TRISA<2> bit is set and the VROE bit, VRCON<6>, is set. Enabling the Voltage Reference output onto the RA2 pin with an input signal present will increase current consumption. Connecting RA2 as a digital output with VREF enabled will also increase current consumption.

The RA2 pin can be used as a simple D/A output with limited drive capability. Due to the limited drive capability, a buffer must be used in conjunction with the Voltage Reference output for external connections to VREF. Figure 11-3 shows an example buffering technique.

FIGURE 11-3: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE

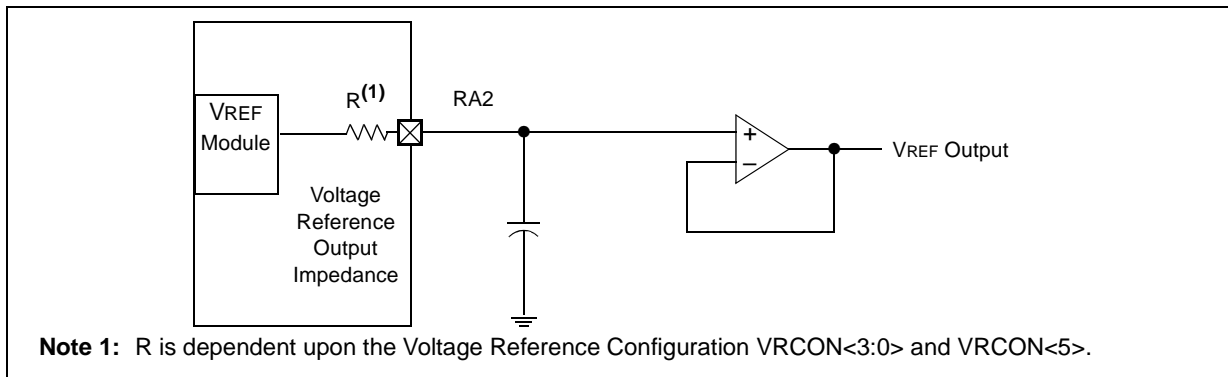


TABLE 11-1: REGISTERS ASSOCIATED WITH VOLTAGE REFERENCE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value On POR	Value On All Other Resets
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	000- 0000
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
85h	TRISA	TRISA7	TRISA6	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11-1 1111	11-1 1111

**Note:** — = Unimplemented, read as "0"

## 12.0 UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI). The USART can be configured as a full duplex asynchronous system that can communicate with peripheral devices such as CRT terminals and personal computers, or it can be configured

as a half duplex synchronous system that can communicate with peripheral devices such as A/D or D/A integrated circuits, Serial EEPROMs etc.

The USART can be configured in the following modes:

- Asynchronous (full duplex)
- Synchronous - Master (half duplex)
- Synchronous - Slave (half duplex)

Bit SPEN (RCSTA<7>), and bits TRISB<2:1>, have to be set in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter.

### REGISTER 12-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0	
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	
bit7								bit0
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin-left: auto; margin-right: auto;">                     R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'                      -n = Value at POR reset                 </div>								
bit 7:	<b>CSRC:</b> Clock Source Select bit <u>Asynchronous mode</u> Don't care  <u>Synchronous mode</u> 1 = Master mode (Clock generated internally from BRG) 0 = Slave mode (Clock from external source)							
bit 6:	<b>TX9:</b> 9-bit Transmit Enable bit 1 = Selects 9-bit transmission 0 = Selects 8-bit transmission							
bit 5:	<b>TXEN:</b> Transmit Enable bit 1 = Transmit enabled 0 = Transmit disabled Note: SREN/CREN overrides TXEN in SYNC mode.							
bit 4:	<b>SYNC:</b> USART Mode Select bit 1 = Synchronous mode 0 = Asynchronous mode							
bit 3:	<b>Unimplemented:</b> Read as '0'							
bit 2:	<b>BRGH:</b> High Baud Rate Select bit  <u>Asynchronous mode</u> 1 = High speed 0 = Low speed  <u>Synchronous mode</u> Unused in this mode							
bit 1:	<b>TRMT:</b> Transmit Shift Register Status bit 1 = TSR empty 0 = TSR full							
bit 0:	<b>TX9D:</b> 9th bit of transmit data. Can be parity bit.							

# PIC16F62X

## REGISTER 12-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D
bit7				bit0			

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

bit 7: **SPEN**: Serial Port Enable bit  
(Configures RB1/RX/DT and RB2/TX/CK pins as serial port pins when bits TRISB<2:17> are set)  
1 = Serial port enabled  
0 = Serial port disabled

bit 6: **RX9**: 9-bit Receive Enable bit  
1 = Selects 9-bit reception  
0 = Selects 8-bit reception

bit 5: **SREN**: Single Receive Enable bit  
Asynchronous mode:  
Don't care  
Synchronous mode - master:  
1 = Enables single receive  
0 = Disables single receive  
This bit is cleared after reception is complete.  
Synchronous mode - slave:  
Unused in this mode

bit 4: **CREN**: Continuous Receive Enable bit  
Asynchronous mode:  
1 = Enables continuous receive  
0 = Disables continuous receive  
Synchronous mode:  
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)  
0 = Disables continuous receive

bit 3: **ADEN**: Address Detect Enable bit  
Asynchronous mode 9-bit (RX9 = 1):  
1 = Enables address detection, enable interrupt and load of the receive buffer when RSR<8> is set  
0 = Disables address detection, all bytes are received, and ninth bit can be used as parity bit  
Asynchronous mode 8-bit (RX9=0):  
Unused in this mode  
Synchronous mode  
Unused in this mode

bit 2: **FERR**: Framing Error bit  
1 = Framing error (Can be updated by reading RCREG register and receive next valid byte)  
0 = No framing error

bit 1: **OERR**: Overrun Error bit  
1 = Overrun error (Can be cleared by clearing bit CREN)  
0 = No overrun error

bit 0: **RX9D**: 9th bit of received data (Can be parity bit)



## 12.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free running 8-bit timer. In asynchronous mode bit BRGH (TXSTA<2>) also controls the baud rate. In synchronous mode bit BRGH is ignored. Table 12-1 shows the formula for computation of the baud rate for different USART modes which only apply in master mode (internal clock).

Given the desired baud rate and Fosc, the nearest integer value for the SPBRG register can be calculated using the formula in Table 12-1. From this, the error in baud rate can be determined.

Example 12-1 shows the calculation of the baud rate error for the following conditions:

Fosc = 16 MHz  
 Desired Baud Rate = 9600  
 BRGH = 0  
 SYNC = 0

### EXAMPLE 12-1: CALCULATING BAUD RATE ERROR

$$\begin{aligned} \text{Desired Baud rate} &= F_{osc} / (64 (X + 1)) \\ 9600 &= 16000000 / (64 (X + 1)) \\ X &= \hat{25.042} = 25 \\ \text{Calculated Baud Rate} &= 16000000 / (64 (25 + 1)) \\ &= 9615 \\ \text{Error} &= \frac{(\text{Calculated Baud Rate} - \text{Desired Baud Rate})}{\text{Desired Baud Rate}} \\ &= (9615 - 9600) / 9600 \\ &= 0.16\% \end{aligned}$$

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the  $F_{osc}/(16(X + 1))$  equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register, causes the BRG timer to be reset (or cleared), this ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

**TABLE 12-1: BAUD RATE FORMULA**

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = $F_{osc}/(64(X+1))$	Baud Rate = $F_{osc}/(16(X+1))$
1	(Synchronous) Baud Rate = $F_{osc}/(4(X+1))$	NA

X = value in SPBRG (0 to 255)

**TABLE 12-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other resets	
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010	
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x	
99h	SPBRG	Baud Rate Generator Register									0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used by the BRG.

# PIC16F62X

**TABLE 12-3: BAUD RATES FOR SYNCHRONOUS MODE**

BAUD RATE (K)	FOSC = 20 MHz			16 MHz			10 MHz			7.15909 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	NA	-	-	NA	-	-
9.6	NA	-	-	NA	-	-	9.766	+1.73	255	9.622	+0.23	185
19.2	19.53	+1.73	255	19.23	+0.16	207	19.23	+0.16	129	19.24	+0.23	92
76.8	76.92	+0.16	64	76.92	+0.16	51	75.76	-1.36	32	77.82	+1.32	22
96	96.15	+0.16	51	95.24	-0.79	41	96.15	+0.16	25	94.20	-1.88	18
300	294.1	-1.96	16	307.69	+2.56	12	312.5	+4.17	7	298.3	-0.57	5
500	500	0	9	500	0	7	500	0	4	NA	-	-
HIGH	5000	-	0	4000	-	0	2500	-	0	1789.8	-	0
LOW	19.53	-	255	15.625	-	255	9.766	-	255	6.991	-	255

BAUD RATE (K)	FOSC = 5.0688 MHz			4 MHz			3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-	0.303	+1.14	26
1.2	NA	-	-	NA	-	-	NA	-	-	1.202	+0.16	207	1.170	-2.48	6
2.4	NA	-	-	NA	-	-	NA	-	-	2.404	+0.16	103	NA	-	-
9.6	9.6	0	131	9.615	+0.16	103	9.622	+0.23	92	9.615	+0.16	25	NA	-	-
19.2	19.2	0	65	19.231	+0.16	51	19.04	-0.83	46	19.24	+0.16	12	NA	-	-
76.8	79.2	+3.13	15	76.923	+0.16	12	74.57	-2.90	11	83.34	+8.51	2	NA	-	-
96	97.48	+1.54	12	1000	+4.17	9	99.43	+3.57	8	NA	-	-	NA	-	-
300	316.8	+5.60	3	NA	-	-	298.3	-0.57	2	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	1267	-	0	100	-	0	894.9	-	0	250	-	0	8.192	-	0
LOW	4.950	-	255	3.906	-	255	3.496	-	255	0.9766	-	255	0.032	-	255

**TABLE 12-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)**

BAUD RATE (K)	FOSC = 20 MHz			16 MHz			10 MHz			7.15909 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	1.221	+1.73	255	1.202	+0.16	207	1.202	+0.16	129	1.203	+0.23	92
2.4	2.404	+0.16	129	2.404	+0.16	103	2.404	+0.16	64	2.380	-0.83	46
9.6	9.469	-1.36	32	9.615	+0.16	25	9.766	+1.73	15	9.322	-2.90	11
19.2	19.53	+1.73	15	19.23	+0.16	12	19.53	+1.73	7	18.64	-2.90	5
76.8	78.13	+1.73	3	83.33	+8.51	2	78.13	+1.73	1	NA	-	-
96	104.2	+8.51	2	NA	-	-	NA	-	-	NA	-	-
300	312.5	+4.17	0	NA	-	-	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	312.5	-	0	250	-	0	156.3	-	0	111.9	-	0
LOW	1.221	-	255	0.977	-	255	0.6104	-	255	0.437	-	255

BAUD RATE (K)	FOSC = 5.0688 MHz			4 MHz			3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	0.31	+3.13	255	0.3005	-0.17	207	0.301	+0.23	185	0.300	+0.16	51	0.256	-14.67	1
1.2	1.2	0	65	1.202	+1.67	51	1.190	-0.83	46	1.202	+0.16	12	NA	-	-
2.4	2.4	0	32	2.404	+1.67	25	2.432	+1.32	22	2.232	-6.99	6	NA	-	-
9.6	9.9	+3.13	7	NA	-	-	9.322	-2.90	5	NA	-	-	NA	-	-
19.2	19.8	+3.13	3	NA	-	-	18.64	-2.90	2	NA	-	-	NA	-	-
76.8	79.2	+3.13	0	NA	-	-	NA	-	-	NA	-	-	NA	-	-
96	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
300	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
HIGH	79.2	-	0	62.500	-	0	55.93	-	0	15.63	-	0	0.512	-	0
LOW	0.3094	-	255	3.906	-	255	0.2185	-	255	0.0610	-	255	0.0020	-	255

**TABLE 12-5: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)**

BAUD RATE (K)	FOSC = 20 MHz			16 MHz			10 MHz			7.16 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
9.6	9.615	+0.16	129	9.615	+0.16	103	9.615	+0.16	64	9.520	-0.83	46
19.2	19.230	+0.16	64	19.230	+0.16	51	18.939	-1.36	32	19.454	+1.32	22
38.4	37.878	-1.36	32	38.461	+0.16	25	39.062	+1.7	15	37.286	-2.90	11
57.6	56.818	-1.36	21	58.823	+2.12	16	56.818	-1.36	10	55.930	-2.90	7
115.2	113.636	-1.36	10	111.111	-3.55	8	125	+8.51	4	111.860	-2.90	3
250	250	0	4	250	0	3	NA	-	-	NA	-	-
625	625	0	1	NA	-	-	625	0	0	NA	-	-
1250	1250	0	0	NA	-	-	NA	-	-	NA	-	-

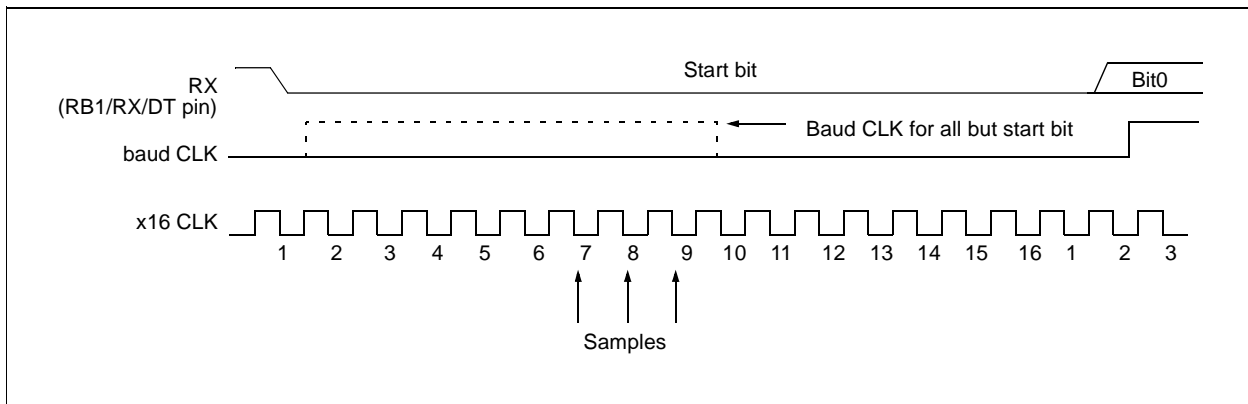
BAUD RATE (K)	FOSC = 5.068 MHz			4 MHz			3.579 MHz			1 MHz			32.768 kHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
9.6	9.6	0	32	NA	-	-	9.727	+1.32	22	8.928	-6.99	6	NA	-	-
19.2	18.645	-2.94	16	1.202	+0.17	207	18.643	-2.90	11	20.833	+8.51	2	NA	-	-
38.4	39.6	+3.12	7	2.403	+0.13	103	37.286	-2.90	5	31.25	-18.61	1	NA	-	-
57.6	52.8	-8.33	5	9.615	+0.16	25	55.930	-2.90	3	62.5	+8.51	0	NA	-	-
115.2	105.6	-8.33	2	19.231	+0.16	12	111.860	-2.90	1	NA	-	-	NA	-	-
250	NA	-	-	NA	-	-	223.721	-10.51	0	NA	-	-	NA	-	-
625	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1250	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-

# PIC16F62X

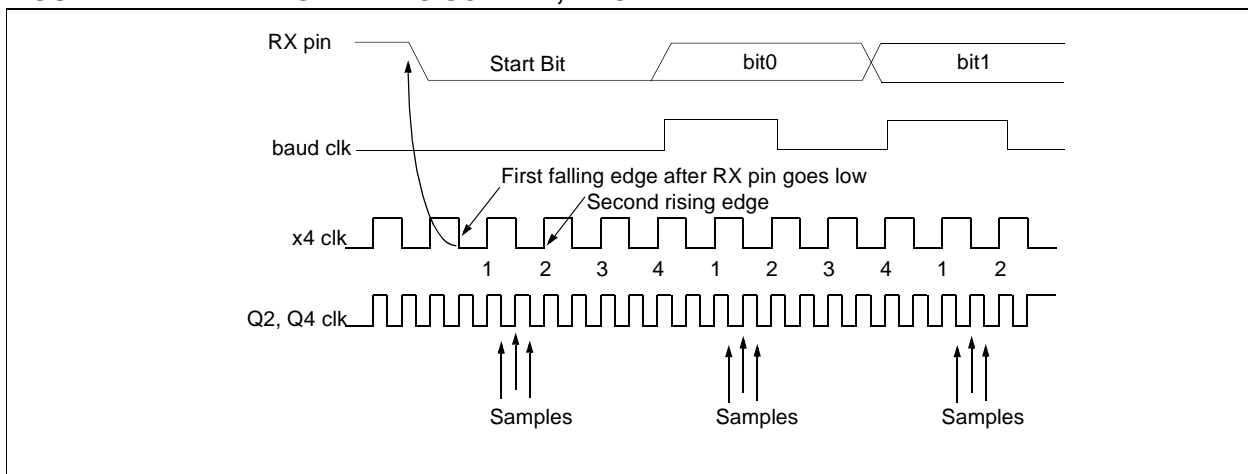
## 12.1.1 SAMPLING

The data on the RB1/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin. If bit BRGH (TXSTA<2>) is clear (i.e., at the low baud rates), the sampling is done on the seventh, eighth and ninth falling edges of a x16 clock (Figure 12-3). If bit BRGH is set (i.e., at the high baud rates), the sampling is done on the 3 clock edges preceding the second rising edge after the first falling edge of a x4 clock (Figure 12-4 and Figure 12-5).

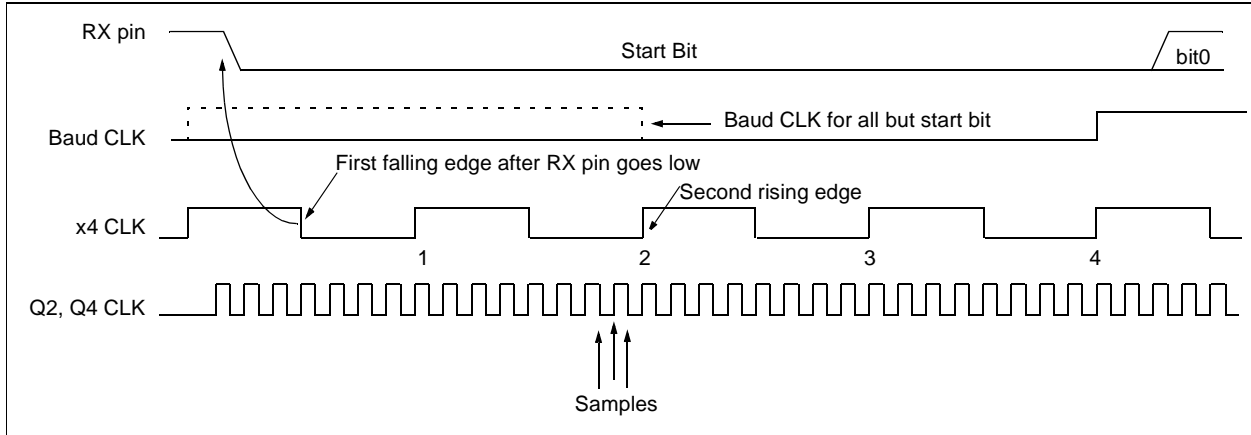
**FIGURE 12-1: RX PIN SAMPLING SCHEME. BRGH = 0**



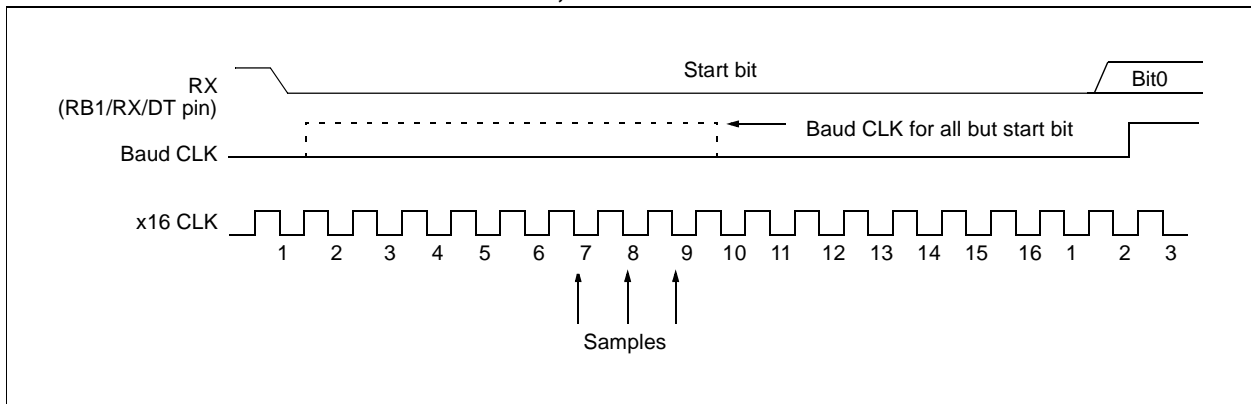
**FIGURE 12-2: RX PIN SAMPLING SCHEME, BRGH = 1**



**FIGURE 12-3: RX PIN SAMPLING SCHEME, BRGH = 1**



**FIGURE 12-4: RX PIN SAMPLING SCHEME, BRGH = 0 OR BRGH = 1**



# PIC16F62X

## 12.2 USART Asynchronous Mode

In this mode, the USART uses standard nonreturn-to-zero (NRZ) format (one start bit, eight or nine data bits and one stop bit). The most common data format is 8-bits. An on-chip dedicated 8-bit baud rate generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The USART's transmitter and receiver are functionally independent but use the same data format and baud rate. The baud rate generator produces a clock either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during SLEEP.

Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

### 12.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 12-5. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the STOP bit has been transmitted from the previous load. As soon as the STOP bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one  $T_{cy}$ ), the TXREG register is empty and flag bit TXIF (PIR1<4>) is set. This interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the

state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicated the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. Status bit TRMT is a read only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

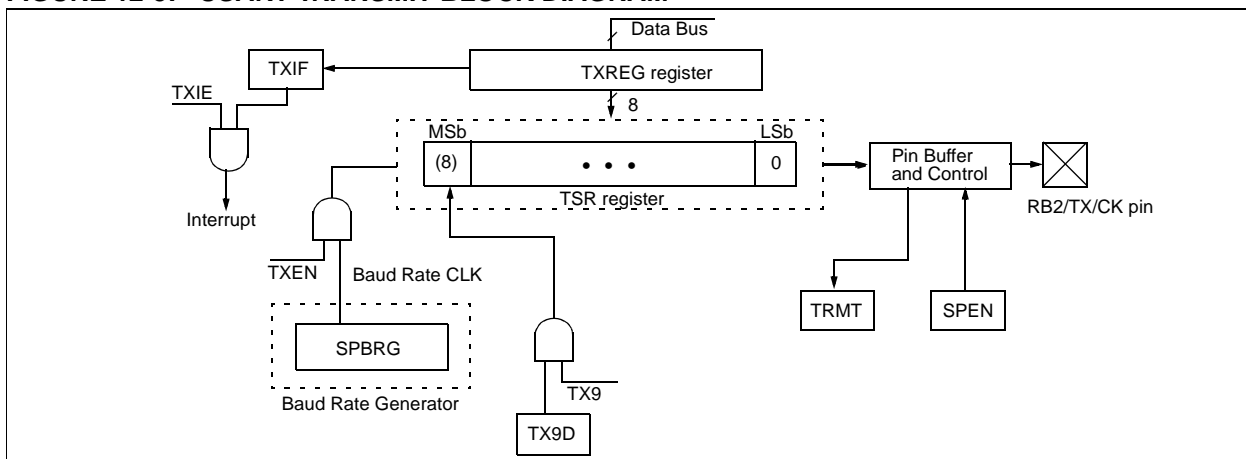
**Note 1:** The TSR register is not mapped in data memory so it is not available to the user.

**Note 2:** Flag bit TXIF is set when enable bit TXEN is set.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the baud rate generator (BRG) has produced a shift clock (Figure 12-5). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 12-7). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result the RB2/TX/CK pin will revert to hi-impedance.

In order to select 9-bit transmission, transmit bit TX9 (TXSTA<6>) should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit maybe loaded in the TSR register.

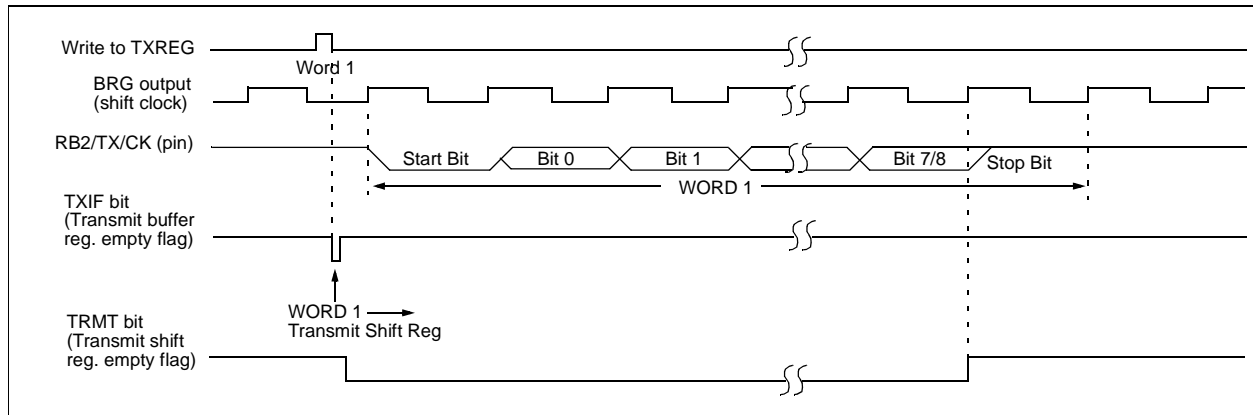
**FIGURE 12-5: USART TRANSMIT BLOCK DIAGRAM**



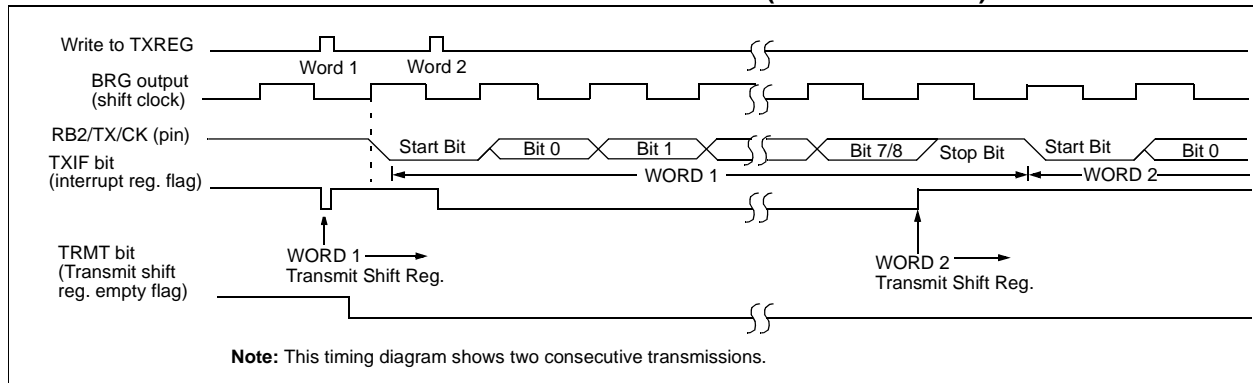
Steps to follow when setting up an Asynchronous Transmission:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH. (Section 12.1)
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set transmit bit TX9.
5. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).

**FIGURE 12-6: ASYNCHRONOUS MASTER TRANSMISSION**



**FIGURE 12-7: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)**



**TABLE 12-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for Asynchronous Transmission.

# PIC16F62X

## 12.2.2 USART ASYNCHRONOUS RECEIVER

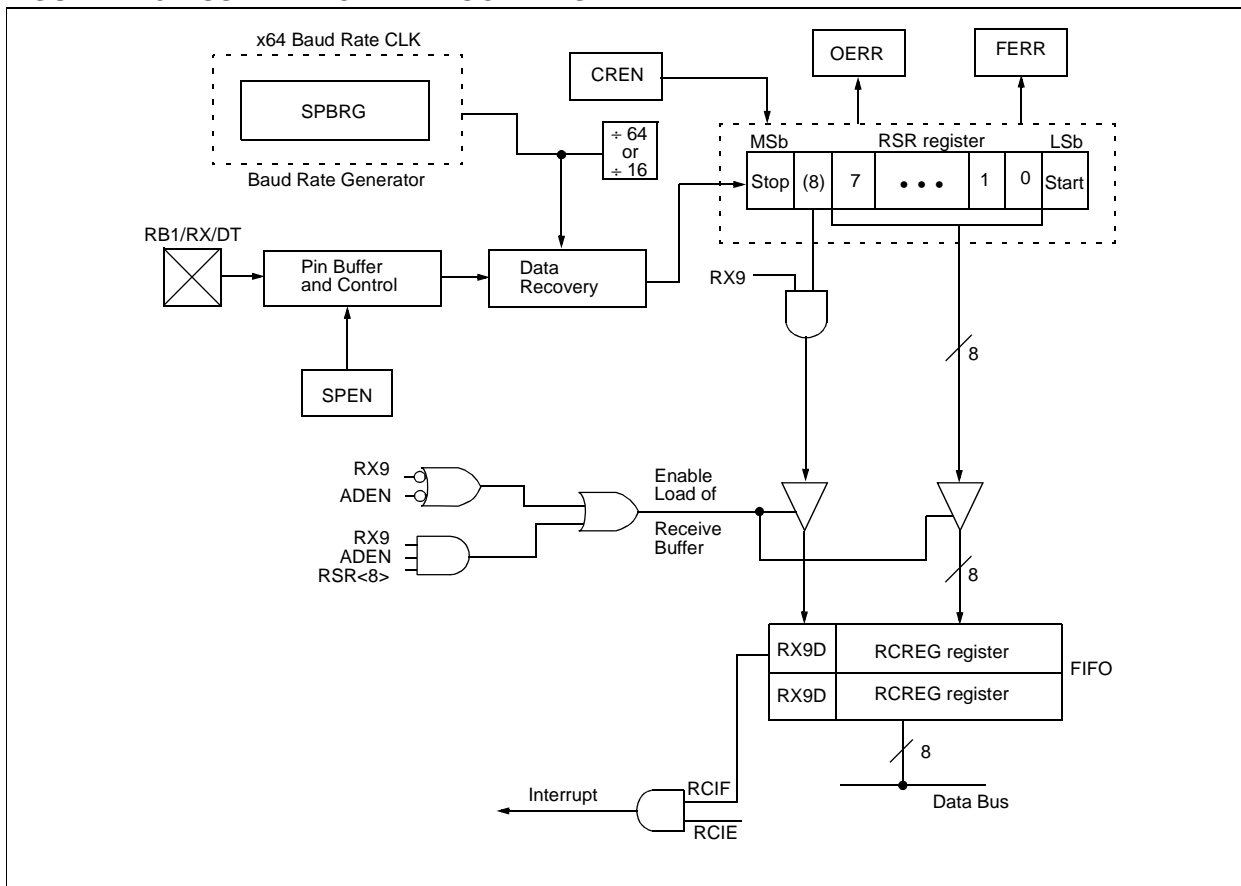
The receiver block diagram is shown in Figure 12-8. The data is received on the RB1/RX/DT pin and drives the data recovery block. The data recovery block is actually a high speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC.

Once Asynchronous mode is selected, reception is enabled by setting bit CREN (RCSTA<4>).

The heart of the receiver is the receive (serial) shift register (RSR). After sampling the STOP bit, the received data in the RSR is transferred to the RCREG register (if it is empty). If the transfer is complete, flag bit RCIF (PIR1<5>) is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>). Flag bit RCIF is a read only bit which is cleared by the hardware. It is cleared when the RCREG register has been read and is empty. The RCREG is a double buff-

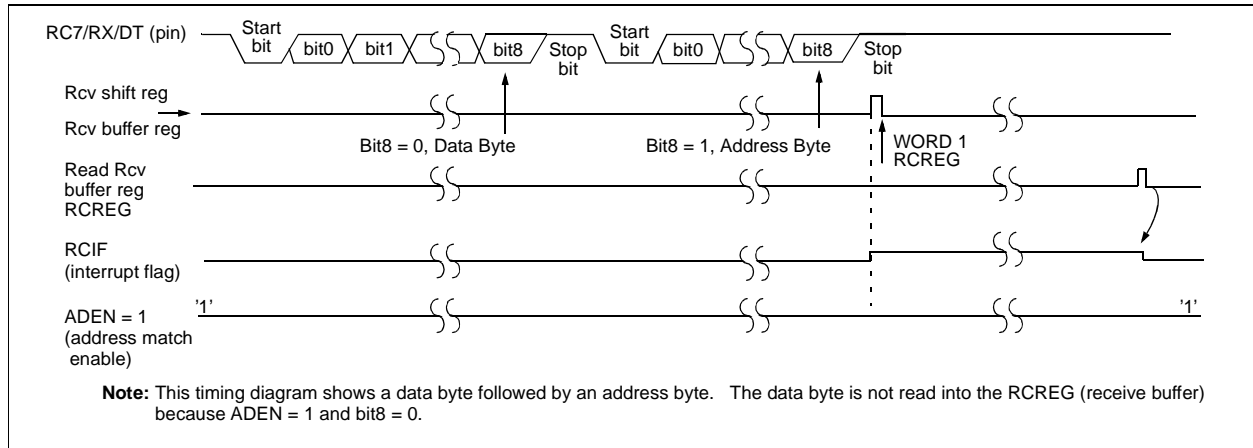
ered register, i.e. it is a two deep FIFO. It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte begin shifting to the RSR register. On the detection of the STOP bit of the third byte, if the RCREG register is still full then overrun error bit OERR (RCSTA<1>) will be set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Overrun bit OERR has to be cleared in software. This is done by resetting the receive logic (CREN is cleared and then set). If bit OERR is set, transfers from the RSR register to the RCREG register are inhibited, so it is essential to clear error bit OERR if it is set. Framing error bit FERR (RCSTA<2>) is set if a stop bit is detected as clear. Bit FERR and the 9th receive bit are buffered the same way as the receive data. Reading the RCREG, will load bits RX9D and FERR with new values, therefore it is essential for the user to read the RCSTA register before reading RCREG register in order not to lose the old FERR and RX9D information.

**FIGURE 12-8: USART RECEIVE BLOCK DIAGRAM**

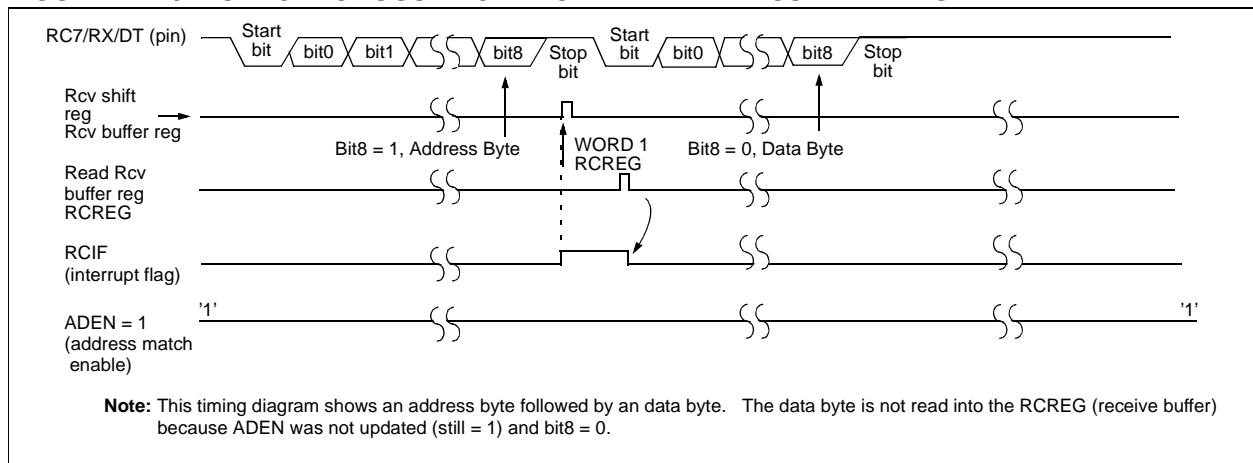




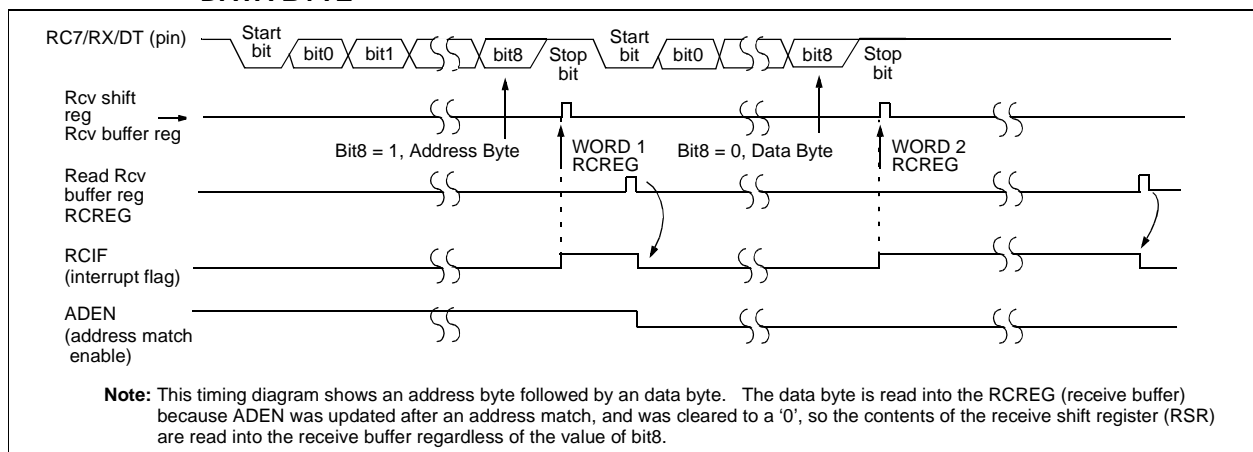
**FIGURE 12-9: ASYNCHRONOUS RECEPTION WITH ADDRESS DETECT**



**FIGURE 12-10: ASYNCHRONOUS RECEPTION WITH ADDRESS BYTE FIRST**



**FIGURE 12-11: ASYNCHRONOUS RECEPTION WITH ADDRESS BYTE FIRST FOLLOWED BY VALID DATA BYTE**



# PIC16F62X

Steps to follow when setting up an Asynchronous Reception:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH. (Section 12.1).
2. Enable the asynchronous serial port by clearing bit SYNC, and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.

**TABLE 12-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for Asynchronous Reception.

## 12.3 USART Function

The USART function is similar to that on the PIC16C74B, which includes the BRGH = 1 fix.

### 12.3.1 USART 9-BIT RECEIVER WITH ADDRESS DETECT

When the RX9 bit is set in the RCSTA register, 9-bits are received and the ninth bit is placed in the RX9D bit of the RCSTA register. The USART module has a special provision for multi-processor communication. Multiprocessor communication is enabled by setting the ADEN bit (RCSTA<3>) along with the RX9 bit. The port is now programmed such that when the last bit is received, the contents of the receive shift register (RSR) are transferred to the receive buffer, the ninth bit of the RSR (RSR<8>) is transferred to RX9D, and the receive interrupt is set if and only if RSR<8> = 1. This feature can be used in a multi-processor system as follows:

A master processor intends to transmit a block of data to one of many slaves. It must first send out an address byte that identifies the target slave. An address byte is identified by setting the ninth bit (RSR<8>) to a '1' (instead of a '0' for a data byte). If the ADEN and RX9 bits are set in the slave's RCSTA register, enabling multiprocessor communication, all data bytes will be ignored. However, if the ninth received bit is equal to a '1', indicating that the received byte is an address, the slave will be interrupted and the contents of the RSR register will be transferred into the receive buffer. This allows the slave to be interrupted only by addresses, so that the slave can examine the received byte to see if it is being addressed. The addressed slave will then clear its ADEN bit and prepare to receive data bytes from the master.

When ADEN is enabled (= '1'), all data bytes are ignored. Following the STOP bit, the data will not be loaded into the receive buffer, and no interrupt will occur. If another byte is shifted into the RSR register, the previous data byte will be lost.

The ADEN bit will only take effect when the receiver is configured in 9-bit mode (RX9 = '1'). When ADEN is disabled (= '0'), all data bytes are received and the 9th bit can be used as the parity bit.

The receive block diagram is shown in Figure 12-8.

Reception is enabled by setting bit CREN (RCSTA<4>).

#### 12.3.1.1 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

Steps to follow when setting up an Asynchronous or Synchronous Reception with Address Detect Enabled:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH.
2. Enable asynchronous or synchronous communication by setting or clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. Set bit RX9 to enable 9-bit reception.
5. Set ADEN to enable address detect.
6. Enable the reception by setting enable bit CREN or SREN.
7. Flag bit RCIF will be set when reception is complete, and an interrupt will be generated if enable bit RCIE was set.
8. Read the 8-bit received data by reading the RCREG register to determine if the device is being addressed.
9. If any error occurred, clear the error by clearing enable bit CREN if it was already set.
10. If the device has been addressed (RSR<8> = 1 with address match enabled), clear the ADEN and RCIF bits to allow data bytes and address bytes to be read into the receive buffer and interrupt the CPU.

# PIC16F62X

**TABLE 12-1: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for Asynchronous Reception.

## 12.4 USART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner, i.e. transmission and reception do not occur at the same time. When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition enable bit SPEN (RCSTA<7>) is set in order to configure the RB2/TX/CK and RB1/RX/DT I/O pins to CK (clock) and DT (data) lines respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTA<7>).

### 12.4.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 12-5. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer register TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one Tcycle), the TXREG is empty and interrupt bit, TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. TRMT is a read only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the CK line. Data out is sta-

ble around the falling edge of the synchronous clock (Figure 12-12). The transmission can also be started by first loading the TXREG register and then setting bit TXEN (Figure 12-13). This is advantageous when slow baud rates are selected, since the BRG is kept in reset when bits TXEN, CREN, and SREN are clear. Setting enable bit TXEN will start the BRG, creating a shift clock immediately. Normally when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing enable bit TXEN, during a transmission, will cause the transmission to be aborted and will reset the transmitter. The DT and CK pins will revert to hi-impedance. If either bit CREN or bit SREN is set, during a transmission, the transmission is aborted and the DT pin reverts to a hi-impedance state (for a reception). The CK pin will remain an output if bit CSRC is set (internal clock). The transmitter logic however is not reset although it is disconnected from the pins. In order to reset the transmitter, the user has to clear bit TXEN. If bit SREN is set (to interrupt an on-going transmission and receive a single word), then after the single word is received, bit SREN will be cleared and the serial port will revert back to transmitting since bit TXEN is still set. The DT line will immediately switch from hi-impedance receive mode to transmit and start driving. To avoid this, bit TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit should be set and the ninth bit should be written to bit TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG can result in an immediate transfer of the data to the TSR register (if the TSR is empty). If the TSR was empty and the TXREG was written before writing the "new" TX9D, the "present" value of bit TX9D is loaded.

Steps to follow when setting up a Synchronous Master Transmission:

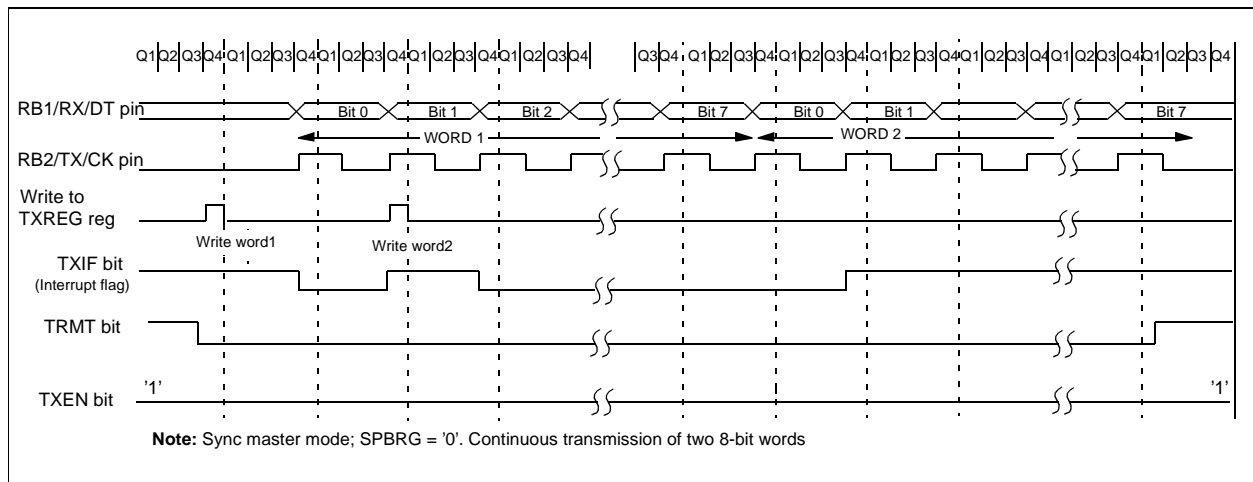
1. Initialize the SPBRG register for the appropriate baud rate (Section 12.1).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.

**TABLE 12-2: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

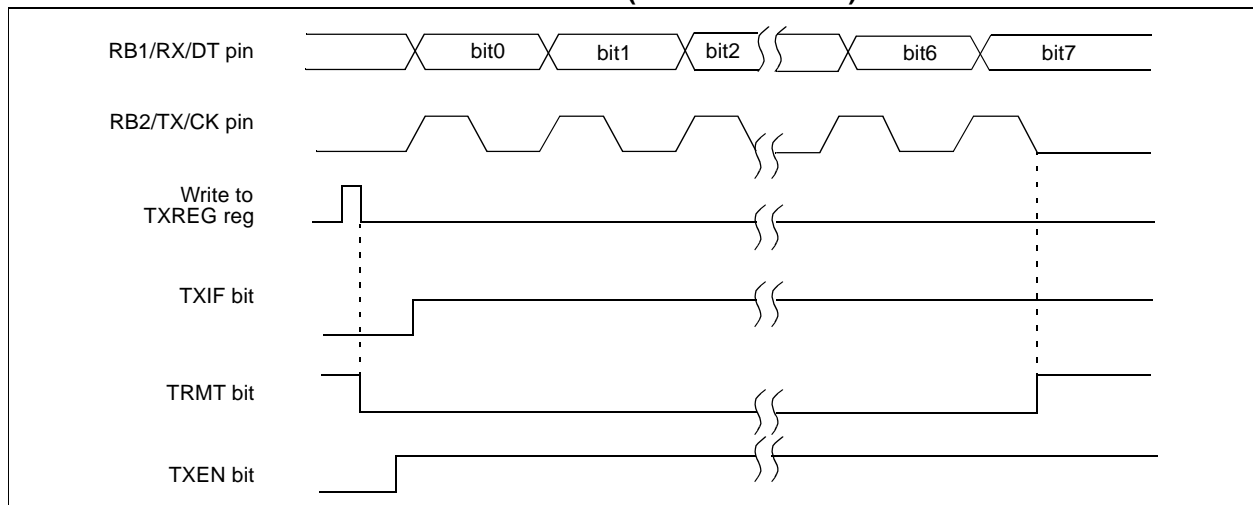
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for Synchronous Master Transmission.

**FIGURE 12-12: SYNCHRONOUS TRANSMISSION**



**FIGURE 12-13: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



# PIC16F62X

## 12.4.2 USART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either enable bit SREN (RCSTA<5>) or enable bit CREN (RCSTA<4>). Data is sampled on the RB1/RX/DT pin on the falling edge of the clock. If enable bit SREN is set, then only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set then CREN takes precedence. After clocking the last bit, the received data in the Receive Shift Register (RSR) is transferred to the RCREG register (if it is empty). When the transfer is complete, interrupt flag bit RCIF (PIR1<5>) is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>). Flag bit RCIF is a read only bit which is reset by the hardware. In this case it is reset when the RCREG register has been read and is empty. The RCREG is a double buffered register, i.e. it is a two deep FIFO. It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting into the RSR register. On the clocking of the last bit of the third byte, if the RCREG register is still full then overrun error bit OERR (RCSTA<1>) is set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Bit OERR has to be cleared in software (by clearing bit CREN). If bit OERR is set, transfers from the RSR to the RCREG are inhibited, so it is essential to clear bit OERR if it is set. The 9th

receive bit is buffered the same way as the receive data. Reading the RCREG register, will load bit RX9D with a new value, therefore it is essential for the user to read the RCSTA register before reading RCREG in order not to lose the old RX9D information.

Steps to follow when setting up a Synchronous Master Reception:

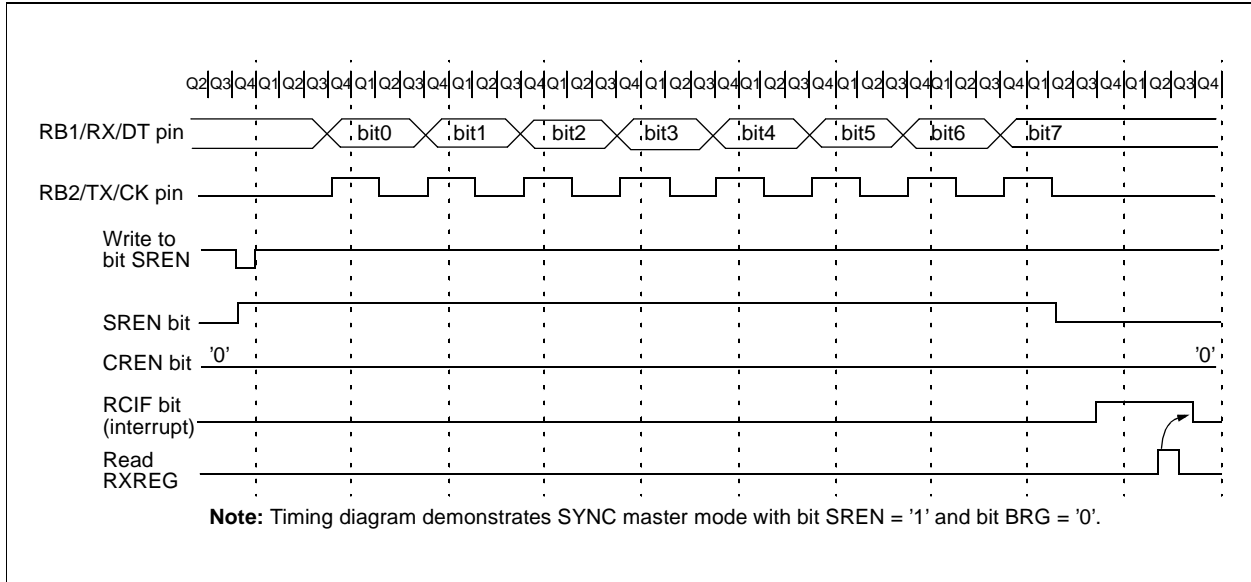
1. Initialize the SPBRG register for the appropriate baud rate. (Section 12.1)
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, then set enable bit RCIE.
5. If 9-bit reception is desired, then set bit RX9.
6. If a single reception is required, set bit SREN. For continuous reception set bit CREN.
7. Interrupt flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
8. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing bit CREN.

**TABLE 12-3: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	EEPIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used for Synchronous Master Reception.

**FIGURE 12-14: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**



## 12.5 USART Synchronous Slave Mode

Synchronous slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the RB2/TX/CK pin (instead of being supplied internally in master mode). This allows the device to transfer or receive data while in SLEEP mode. Slave mode is entered by clearing bit CSRC (TXSTA<7>).

### 12.5.1 USART SYNCHRONOUS SLAVE TRANSMIT

The operation of the synchronous master and slave modes are identical except in the case of the SLEEP mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

- a) The first word will immediately transfer to the TSR register and transmit.
- b) The second word will remain in TXREG register.
- c) Flag bit TXIF will not be set.
- d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will now be set.
- e) If enable bit TXIE is set, the interrupt will wake the chip from SLEEP and if the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

Steps to follow when setting up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.

### 12.5.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the synchronous master and slave modes is identical except in the case of the SLEEP mode. Also, bit SREN is a don't care in slave mode.

If receive is enabled, by setting bit CREN, prior to the SLEEP instruction, then a word may be received during SLEEP. On completely receiving the word, the RSR register will transfer the data to the RCREG register and if enable bit RCIE bit is set, the interrupt generated will wake the chip from SLEEP. If the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

Steps to follow when setting up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, then set enable bit RCIE.
3. If 9-bit reception is desired, then set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete and an interrupt will be generated, if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.



**TABLE 12-4: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used for Synchronous Slave Transmission.

**TABLE 12-5: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 -00x	0000 -00x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented read as '0'. Shaded cells are not used for Synchronous Slave Reception.

# PIC16F62X

---

NOTES:

## 13.0 DATA EEPROM MEMORY

The EEPROM data memory is readable and writable during normal operation (full VDD range). This memory is not directly mapped in the register file space. Instead it is indirectly addressed through the Special Function Registers. There are four SFRs used to read and write this memory. These registers are:

- EECON1
- EECON2 (Not a physically implemented register)
- EEDATA
- EEADR

EEDATA holds the 8-bit data for read/write, and EEADR holds the address of the EEPROM location being accessed. PIC16F62X devices have 128 bytes of data EEPROM with an address range from 0h to 7Fh.

The EEPROM data memory allows byte read and write. A byte write automatically erases the location and writes the new data (erase before write). The EEPROM data memory is rated for high erase/write cycles. The write time is controlled by an on-chip timer. The write-time will vary with voltage and temperature as well as from chip to chip. Please refer to AC specifications for exact limits.

When the device is code protected, the CPU may continue to read and write the data EEPROM memory. The device programmer can no longer access this memory.

Additional information on the Data EEPROM is available in the PICmicro™ Mid-Range Reference Manual, (DS33023).

### REGISTER 13-1: EEADR REGISTER (ADDRESS 9Bh)

U	R/W	R/W	R/W	R/W	R/W	R/W	R/W
—	EADR6	EADR5	EADR4	EADR3	EADR2	EADR1	EADR0
bit7							bit0

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

bit 7     **Unimplemented Address:** Must be set to '0'

bit 6:0   **EEADR:** Specifies one of 128 locations for EEPROM Read/Write Operation

### 13.1 EEADR

The EEADR register can address up to a maximum of 256 bytes of data EEPROM. Only the first 128 bytes of data EEPROM are implemented and only seven of the eight bits in the register (EEADR<6:0>) are required.

The upper bit is address decoded. This means that this bit should always be '0' to ensure that the address is in the 128 byte memory space.

# PIC16F62X

## 13.2 EECON1 AND EECON2 REGISTERS

EECON1 is the control register with five low order bits physically implemented. The upper-three bits are non-existent and read as '0's.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a MCLR reset or a WDT time-out reset during normal operation. In these situations, following reset, the user can check the WRERR bit and rewrite the location. The data and address will be unchanged in the EEDATA and EEADR registers.

Interrupt flag bit EEIF in the PIR1 register is set when write is complete. This bit must be cleared in software.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the Data EEPROM write sequence.

### REGISTER 13-2: EECON1 REGISTER (ADDRESS 9Ch) DEVICES

U	U	U	U	R/W-x	R/W-0	R/S-0	R/S-x
—	—	—	—	WRERR	WREN	WR	RD
bit7				bit0			

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

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

bit 3 **WRERR:** EEPROM Error Flag bit  
1 = A write operation is prematurely terminated (any MCLR reset, any WDT reset during normal operation or BOD detect)  
0 = The write operation completed

bit 2 **WREN:** EEPROM Write Enable bit  
1 = Allows write cycles  
0 = Inhibits write to the data EEPROM

bit 1 **WR:** Write Control bit  
1 = initiates a write cycle. (The bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)  
0 = Write cycle to the data EEPROM is complete

bit 0 **RD:** Read Control bit  
1 = Initiates an EEPROM read (read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software).  
0 = Does not initiate an EEPROM read

## 13.3 READING THE EEPROM DATA MEMORY

To read a data memory location, the user must write the address to the EEADR register and then set control bit RD (EECON1<0>). The data is available, in the very next cycle, in the EEDATA register; therefore it can be read in the next instruction. EEDATA will hold this value until another read or until it is written to by the user (during a write operation).

### EXAMPLE 13-1: DATA EEPROM READ

```
BCF    STATUS, RP0 ; Bank 0
MOVLW CONFIG_ADDR ;
MOVWF  EEADR      ; Address to read
BSF    STATUS, RP0 ; Bank 1
BSF    EECON1, RD ; EE Read
BCF    STATUS, RP0 ; Bank 0
MOVF   EEDATA, W  ; W = EEDATA
```

## 13.4 WRITING TO THE EEPROM DATA MEMORY

To write an EEPROM data location, the user must first write the address to the EEADR register and the data to the EEDATA register. Then the user must follow a specific sequence to initiate the write for each byte.

### EXAMPLE 13-2: DATA EEPROM WRITE

Required Sequence	BSF	STATUS, RP0	; Bank 1
	BSF	EECON1, WREN	; Enable write
	BCF	INTCON, GIE	; Disable INTs.
	MOVLW	55h	;
	MOVWF	EECON2	; Write 55h
	MOVLW	AAh	;
	MOVWF	EECON2	; Write AAh
	BSF	EECON1, WR	; Set WR bit
			; begin write
	BSF	INTCON, GIE	; Enable INTs.

The write will not initiate if the above sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. We strongly recommend that interrupts be disabled during this code segment. A cycle count is executed during the required sequence. Any number what is not equal to the required cycles to execute the required sequence will cause the data not to be written into the EEPROM.

Additionally, the WREN bit in EECON1 must be set to enable write. This mechanism prevents accidental writes to data EEPROM due to errant (unexpected) code execution (i.e., lost programs). The user should keep the WREN bit clear at all times, except when updating EEPROM. The WREN bit is not cleared by hardware

After a write sequence has been initiated, clearing the WREN bit will not affect this write cycle. The WR bit will be inhibited from being set unless the WREN bit is set.

At the completion of the write cycle, the WR bit is cleared in hardware and the EE Write Complete Interrupt Flag bit (EEIF) is set. The user can either enable this interrupt or poll this bit. The EEIF bit in the PIR1 registers must be cleared by software.

## 13.5 WRITE VERIFY

Depending on the application, good programming practice may dictate that the value written to the Data EEPROM should be verified (Example 13-3) to the desired value to be written. This should be used in applications where an EEPROM bit will be stressed near the specification limit.

### EXAMPLE 13-3: WRITE VERIFY

```
BCF    STATUS, RP0 ; Bank 0
:      ; Any code can go here
:      ;
MOVF   EEDATA, W   ; Must be in Bank 0
BSF    STATUS, RP0 ; Bank 1 READ
BSF    EECON1, RD  ; YES, Read the
:      ; value written
BCF    STATUS, RP0 ; Bank 0
;
; Is the value written (in W reg) and
; read (in EEDATA) the same?
;
SUBWF  EEDATA, W   ;
BTFSZ  STATUS, Z   ; Is difference 0?
GOTO   WRITE_ERR  ; NO, Write error
:      ; YES, Good write
:      ; Continue program
```

## 13.6 PROTECTION AGAINST SPURIOUS WRITE

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built in. On power-up, WREN is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

## 13.7 DATA EEPROM OPERATION DURING CODE PROTECT

When the device is code protected, the CPU is able to read and write unscrambled data to the Data EEPROM.

# PIC16F62X

**TABLE 13-1 REGISTERS/BITS ASSOCIATED WITH DATA EEPROM**

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
9Ah	EEDATA	EEPROM data register								xxxx xxxx	uuuu uuuu
9Bh	EEADR	EEPROM address register								xxxx xxxx	uuuu uuuu
9Ch	EECON1	—	—	—	—	WRERR	WREN	WR	RD	---- x000	---- q000
9Dh	EECON2 <sup>(1)</sup>	EEPROM control register 2								---- ----	---- ----

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', q = value depends upon condition. Shaded cells are not used by data EEPROM.

**Note 1:** EECON2 is not a physical register

## 14.0 SPECIAL FEATURES OF THE CPU

Special circuits to deal with the needs of real time applications are what sets a microcontroller apart from other processors. The PIC16F62X family has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection.

These are:

1. OSC selection
2. Reset
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-Up Timer (OST)
  - Brown-out Reset (BOD)
3. Interrupts
4. Watchdog Timer (WDT)
5. SLEEP
6. Code protection
7. ID Locations
8. In-circuit serial programming

The PIC16F62X has a Watchdog Timer which is controlled by configuration bits. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in reset until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only, designed to keep the part in reset while the power supply stabilizes. There is also circuitry to reset the device if a brown-out occurs which provides at least a 72 ms reset. With these three functions on-chip, most applications need no external reset circuitry.

The SLEEP mode is designed to offer a very low current power-down mode. The user can wake-up from SLEEP through external reset, Watchdog Timer wake-up or through an interrupt. Several oscillator options are also made available to allow the part to fit the application. The ER oscillator option saves system cost while the LP crystal option saves power. A set of configuration bits are used to select various options.

# PIC16F62X

## 14.1 Configuration Bits

The configuration bits can be programmed (read as '0') or left unprogrammed (read as '1') to select various device configurations. These bits are mapped in program memory location 2007h.

The user will note that address 2007h is beyond the user program memory space. In fact, it belongs to the special configuration memory space (2000h – 3FFFh), which can be accessed only during programming.

**FIGURE 14-1: CONFIGURATION WORD**

CP1	CP0	CP1	CP0	-	CPD	LVP	BODEN	MCLR $\overline{E}$	FOSC2	$\overline{PWRT\overline{E}}$	WDTE	FOSC1	FOSC0	Register: CONFIG Address: 2007h
bit13													bit0	
<b>bit 13-10: CP1:CP0: Code Protection bits<sup>(2)</sup></b> Code protection for 2K program memory 11 = Program memory code protection off 10 = 0400h-07FFh code protected 01 = 0200h-07FFh code protected 00 = 0000h-07FFh code protected Code protection for 1K program memory 11 = Program memory code protection off 10 = Program memory code protection off 01 = 0200h-03FFh code protected 00 = 0000h-03FFh code protected														
<b>bit 8: CPD: Data Code Protection bit<sup>(3)</sup></b> 1 = Data memory code protection off 0 = Data memory code protected														
<b>bit 7: LVP: Low Voltage Programming Enable</b> 1 = RB4/PGM pin has PGM function, low voltage programming enabled 0 = RB4/PGM is digital I/O, HV on $\overline{MCLR}$ must be used for programming														
<b>bit 6: BODEN: Brown-out Detect Enable bit<sup>(1)</sup></b> 1 = BOD enabled 0 = BOD disabled														
<b>bit 5: <math>\overline{MCLR\overline{E}}</math>: RA5/<math>\overline{MCLR}</math> pin function select</b> 1 = RA5/ $\overline{MCLR}$ pin function is $\overline{MCLR}$ 0 = RA5/ $\overline{MCLR}$ pin function is digital I/O, $\overline{MCLR}$ internally tied to VDD														
<b>bit 3: <math>\overline{PWRT\overline{E}}</math>: Power-up Timer Enable bit<sup>(1)</sup></b> 1 = PWRT disabled 0 = PWRT enabled														
<b>bit 2: WDTE: Watchdog Timer Enable bit</b> 1 = WDT enabled 0 = WDT disabled														
<b>bit 4,1-0: FOSC2:FOSC0: Oscillator Selection bits<sup>(4)</sup></b> 111 = ER oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, Resistor on RA7/OSC1/CLKIN 110 = ER oscillator: I/O function on RA6/OSC2/CLKOUT pin, Resistor on RA7/OSC1/CLKIN 101 = INTRC oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN 100 = INTRC oscillator: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN 011 = EC: I/O function on RA6/OSC2/CLKOUT pin, CLKIN on RA7/OSC1/CLKIN 010 = HS oscillator: High speed crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN 001 = XT oscillator: Crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN 000 = LP oscillator: Low power crystal on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN														
<b>Note 1:</b> Enabling Brown-out Reset automatically enables Power-up Timer (PWRT) regardless of the value of bit $\overline{PWRT\overline{E}}$ . Ensure the Power-up Timer is enabled anytime Brown-out Reset is enabled. <b>2:</b> All of the CP1:CP0 pairs have to be given the same value to enable the code protection scheme listed. <b>3:</b> The entire data EEPROM will be erased when the code protection is turned off. <b>4:</b> When $\overline{MCLR}$ is asserted in INTRC or ER mode, the internal clock oscillator is disabled.														



## 14.2 Oscillator Configurations

### 14.2.1 OSCILLATOR TYPES

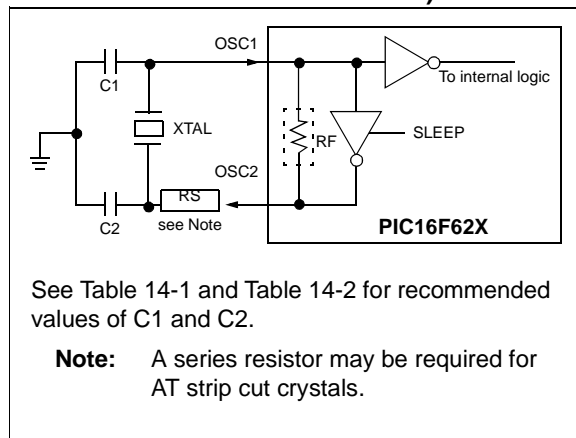
The PIC16F62X can be operated in eight different oscillator options. The user can program three configuration bits (FOSC2 thru FOSC0) to select one of these eight modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- ER External Resistor (2 modes)
- INTRC Internal Resistor/Capacitor (2 modes)
- EC External Clock In

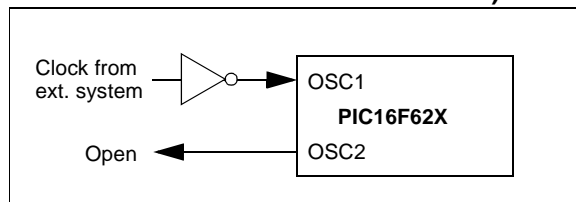
### 14.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, LP or HS modes a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation (Figure 14-2). The PIC16F62X oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, LP or HS modes, the device can have an external clock source to drive the OSC1 pin (Figure 14-3).

**FIGURE 14-2: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (HS, XT OR LP OSC CONFIGURATION)**



**FIGURE 14-3: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)**



**TABLE 14-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS**

Ranges Characterized:			
Mode	Freq	OSC1(C1)	OSC2(C2)
XT	455 kHz	22 - 100 pF	22 - 100 pF
	2.0 MHz	15 - 68 pF	15 - 68 pF
	4.0 MHz	15 - 68 pF	15 - 68 pF
HS	8.0 MHz	10 - 68 pF	10 - 68 pF
	16.0 MHz	10 - 22 pF	10 - 22 pF

Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.

**TABLE 14-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR**

Mode	Freq	OSC1(C1)	OSC2(C2)
LP	32 kHz	68 - 100 pF	68 - 100 pF
	200 kHz	15 - 30 pF	15 - 30 pF
XT	100 kHz	68 - 150 pF	150 - 200 pF
	2 MHz	15 - 30 pF	15 - 30 pF
	4 MHz	15 - 30 pF	15 - 30 pF
HS	8 MHz	15 - 30 pF	15 - 30 pF
	10 MHz	15 - 30 pF	15 - 30 pF
	20 MHz	15 - 30 pF	15 - 30 pF

Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Rs may be required in HS mode as well as XT mode to avoid overdriving crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

# PIC16F62X

## 14.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

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

Figure 14-4 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180° phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides the negative feedback for stability. The 10 kΩ potentiometers bias the 74AS04 in the linear region. This could be used for external oscillator designs.

**FIGURE 14-4: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT**

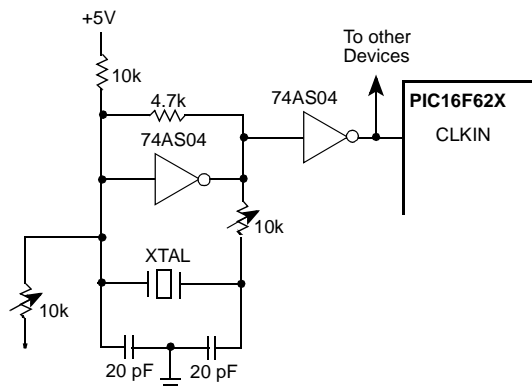
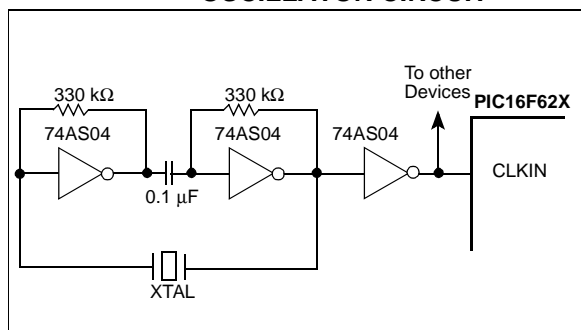


Figure 14-5 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180° phase shift in a series resonant oscillator circuit. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

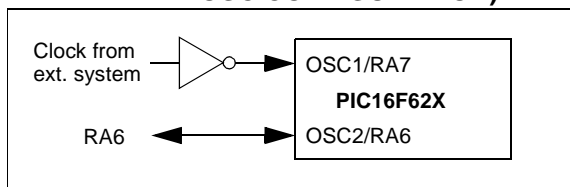
**FIGURE 14-5: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT**



## 14.2.4 EXTERNAL CLOCK IN

For applications where a clock is already available elsewhere, users may directly drive the PIC16F62X provided that this external clock source meets the AC/DC timing requirements listed in Section 17.4. Figure 14-6 below shows how an external clock circuit should be configured.

**FIGURE 14-6: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)**

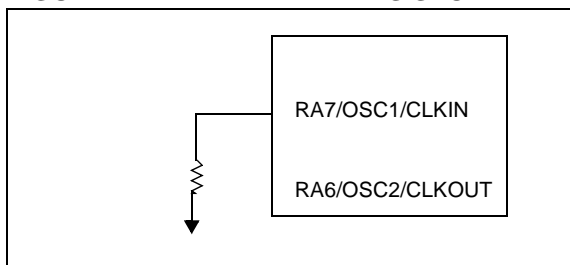


## 14.2.5 ER OSCILLATOR

For timing insensitive applications, the ER (External Resistor) clock mode offers additional cost savings. Only one external component, a resistor to VSS, is needed to set the operating frequency of the internal oscillator. The resistor draws a DC bias current which controls the oscillation frequency. In addition to the resistance value, the oscillator frequency will vary from unit to unit, and as a function of supply voltage and temperature. Since the controlling parameter is a DC current and not a capacitance, the particular package type and lead frame will not have a significant effect on the resultant frequency.

Figure 14-7 shows how the controlling resistor is connected to the PIC16F62X. For Rext values below 38k, the oscillator operation may become unstable, or stop completely. For very high Rext values (e.g. 1M), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend keeping Rext between 38k and 1M.

**FIGURE 14-7: EXTERNAL RESISTOR**



The Electrical Specification section shows the relationship between the resistance value and the operating frequency as well as frequency variations due to operating temperature for given R and VDD values.

The ER oscillator mode has two options that control the unused OSC2 pin. The first allows it to be used as a general purpose I/O port. The other configures the pin as an output providing the Fosc signal (internal clock divided by 4) for test or external synchronization purposes.

## 14.2.6 INTERNAL 4 MHz OSCILLATOR

The internal RC oscillator provides a fixed 4 MHz (nominal) system clock at  $V_{DD} = 5V$  and  $25^{\circ}C$ , see “Electrical Specifications” section for information on variation over voltage and temperature.

## 14.2.7 CLKOUT

The PIC16F62X can be configured to provide a clock out signal by programming the configuration word. The oscillator frequency, divided by 4 can be used for test purposes or to synchronize other logic.

## 14.3 Special Feature: Dual Speed Oscillator Modes

A software programmable dual speed oscillator mode is provided when the PIC16F62X is configured in either ER or INTRC oscillator modes. This feature allows users to dynamically toggle the oscillator speed between 4MHz and 37kHz. In ER mode, the 4MHz setting will vary depending on the size of the external resistor. Also in ER mode, the 37kHz operation is fixed and does not vary with resistor size. Applications that require low current power savings, but cannot tolerate putting the part into sleep, may use this mode.

The OSCF bit in the PCON register is used to control dual speed mode. See Section 4.2.2.6, Figure 4-9.

## 14.4 Reset

The PIC16F62X differentiates between various kinds of reset:

- a) Power-on reset (POR)
- b)  $\overline{MCLR}$  reset during normal operation
- c)  $\overline{MCLR}$  reset during SLEEP
- d) WDT reset (normal operation)
- e) WDT wake-up (SLEEP)
- f) Brown-out Detect (BOD)

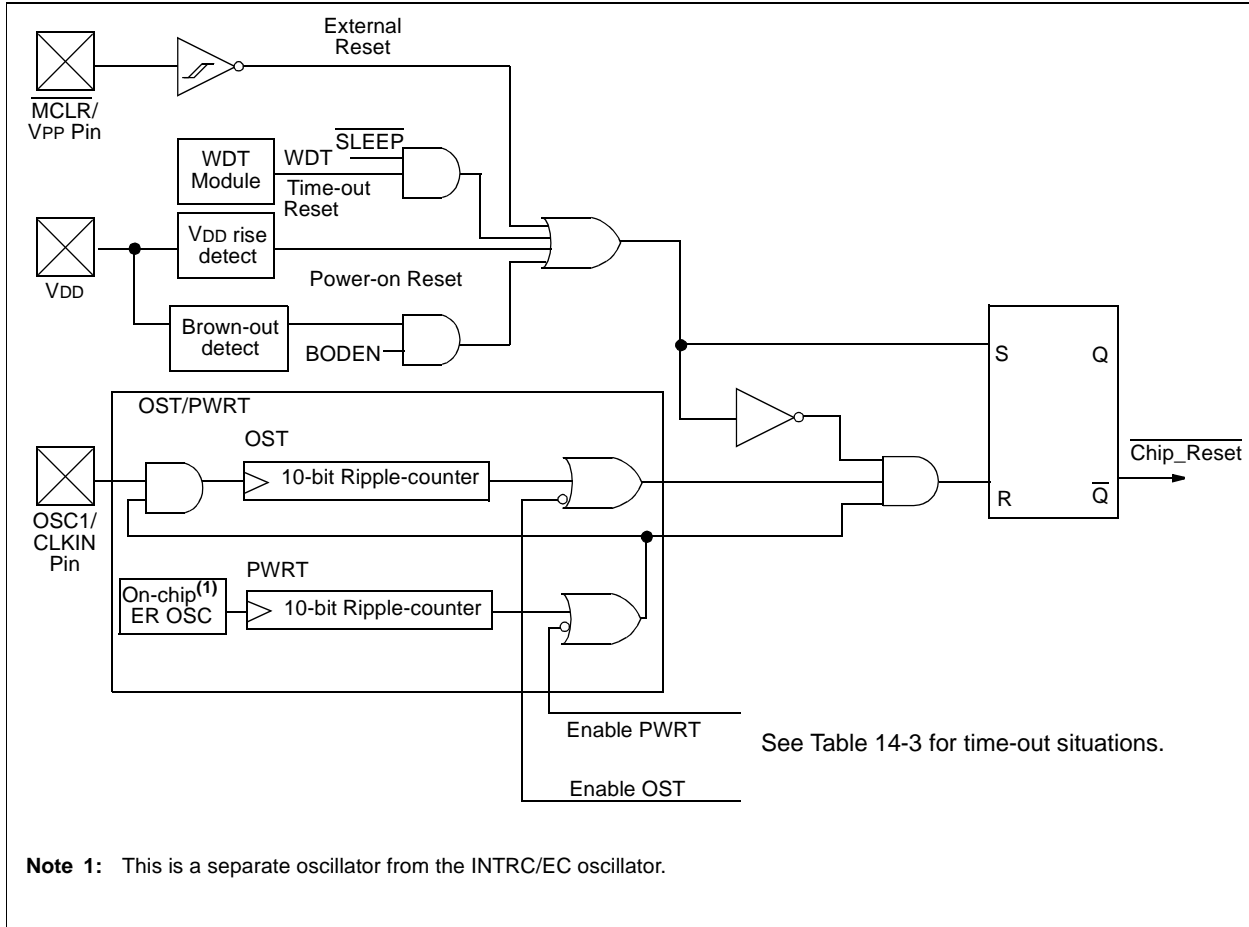
Some registers are not affected in any reset condition; their status is unknown on POR and unchanged in any other reset. Most other registers are reset to a “reset state” on Power-on reset,  $\overline{MCLR}$  reset, WDT reset and  $\overline{MCLR}$  reset during SLEEP. They are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation.  $\overline{TO}$  and  $\overline{PD}$  bits are set or cleared differently in different reset situations as indicated in Table 14-4. These bits are used in software to determine the nature of the reset. See Table 14-7 for a full description of reset states of all registers.

A simplified block diagram of the on-chip reset circuit is shown in Figure 14-8.

The  $\overline{MCLR}$  reset path has a noise filter to detect and ignore small pulses. See Table 12-6 for pulse width specification.

# PIC16F62X

**FIGURE 14-8: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



## 14.5 Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST) and Brown-out Detect (BOD)

### 14.5.1 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".

### 14.5.2 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,  $\overline{\text{PWRT}}\text{E}$  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.

### 14.5.3 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.

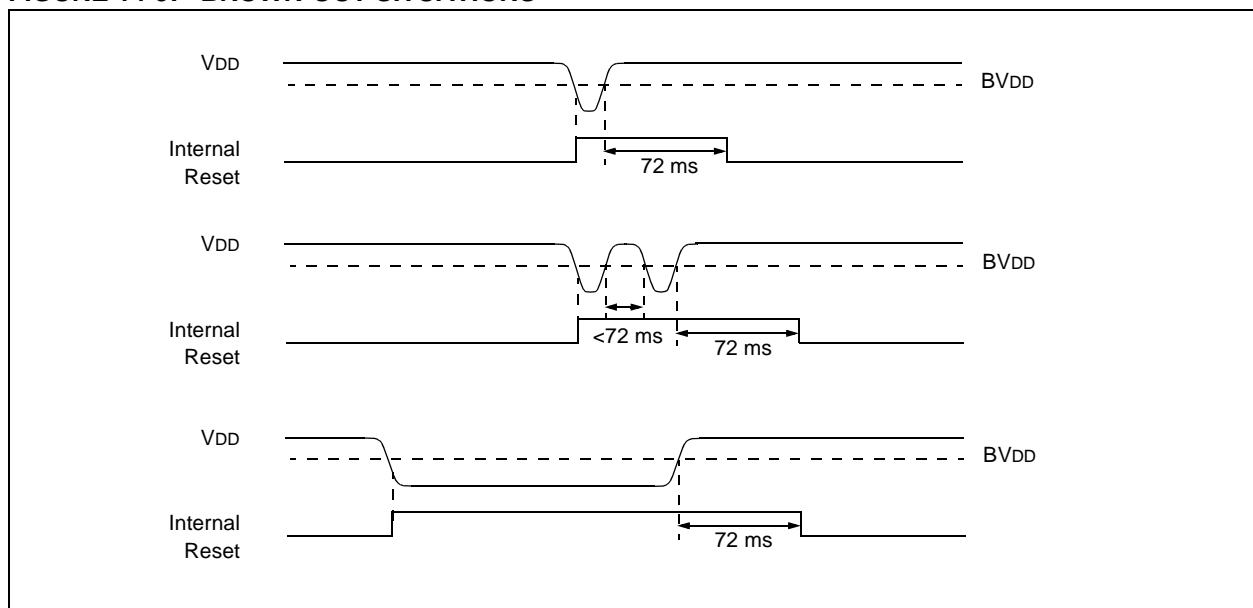
### 14.5.4 BROWN-OUT DETECT (BOD)

The PIC16F62X members have on-chip Brown-out Detect circuitry. A configuration bit, BODEN, can disable (if clear/programmed) or enable (if set) the Brown-out Detect circuitry. If VDD falls below 4.0V, refer to  $V_{\text{BOD}}$  parameter D005( $V_{\text{BOD}}$ ) for greater than parameter (TBOD) in Table 17.1, the brown-out situation will reset the chip. A reset is not guaranteed to occur if VDD falls below 4.0V for less than parameter (TBOD).

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 Detect is enabled. Figure 14-9 shows typical Brown-out situations.

**FIGURE 14-9: BROWN-OUT SITUATIONS**



# PIC16F62X

## 14.5.5 TIME-OUT SEQUENCE

On power-up the time-out sequence is as follows: First PWRT time-out is invoked after POR has expired. Then OST is activated. The total time-out will vary based on oscillator configuration and  $\overline{\text{PWRTE}}$  bit status. For example, in ER mode with  $\overline{\text{PWRTE}}$  bit erased (PWRT disabled), there will be no time-out at all. Figure 14-10, Figure 14-11 and Figure 14-12 depict time-out sequences.

Since the time-outs occur from the POR pulse, if  $\overline{\text{MCLR}}$  is kept low long enough, the time-outs will expire. Then bringing  $\overline{\text{MCLR}}$  high will begin execution immediately (see Figure 14-11). This is useful for testing purposes or to synchronize more than one PIC16F62X device operating in parallel.

Table 14-6 shows the reset conditions for some special registers, while Table 14-7 shows the reset conditions for all the registers.

## 14.5.6 POWER CONTROL (PCON)/ STATUS REGISTER

The power control/status register, PCON (address 8Eh) has two bits.

Bit0 is  $\overline{\text{BOD}}$  (Brown-out).  $\overline{\text{BOD}}$  is unknown on power-on-reset. It must then be set by the user and checked on subsequent resets to see if  $\overline{\text{BOD}} = 0$  indicating that a brown-out has occurred. The  $\overline{\text{BOD}}$  status bit is a don't care and is not necessarily predictable if the brown-out circuit is disabled (by setting BODEN bit = 0 in the Configuration word).

Bit1 is  $\overline{\text{POR}}$  (Power-on-reset). It is a '0' on power-on-reset and unaffected otherwise. The user must write a '1' to this bit following a power-on-reset. On a subsequent reset if  $\overline{\text{POR}}$  is '0', it will indicate that a power-on-reset must have occurred (VDD may have gone too low).

**TABLE 14-3: TIME-OUT IN VARIOUS SITUATIONS**

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

**TABLE 14-4: STATUS/PCON BITS AND THEIR SIGNIFICANCE**

$\overline{\text{POR}}$	$\overline{\text{BOD}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	
0	X	1	1	Power-on-reset
0	X	0	X	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
0	X	X	0	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
1	0	X	X	Brown-out Detect
1	1	0	u	WDT Reset
1	1	0	0	WDT Wake-up
1	1	u	u	$\overline{\text{MCLR}}$ reset during normal operation
1	1	1	0	$\overline{\text{MCLR}}$ reset during SLEEP

Legend: u = unchanged, x = unknown

**TABLE 14-5: SUMMARY OF REGISTERS ASSOCIATED WITH BROWN-OUT**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets <sup>(1)</sup>
03h	STATUS	IRP	RP1	RPO	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	000q qruu
8Eh	PCON	—	—	—	—	OSCF	—	POR	$\overline{\text{BOD}}$	---- 1-0x	---- u-ug

**Note 1:** Other (non power-up) resets include  $\overline{\text{MCLR}}$  reset, Brown-out Detect and Watchdog Timer Reset during normal operation.

**TABLE 14-6: INITIALIZATION CONDITION FOR SPECIAL REGISTERS**

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- 1-0x
MCLR reset during normal operation	000h	000u uuuu	---- 1-uu
MCLR reset during SLEEP	000h	0001 0uuu	---- 1-uu
WDT reset	000h	0000 uuuu	---- 1-uu
WDT Wake-up	PC + 1	uuu0 0uuu	---- --uu
Brown-out Detect	000h	000x xuuu	---- 1-u0
Interrupt Wake-up from SLEEP	PC + 1 <sup>(1)</sup>	uuu1 0uuu	---- --uu

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

**Note 1:** When the wake-up is due to an interrupt and global enable bit, GIE is set, the PC is loaded with the interrupt vector (0004h) after execution of PC+1.

# PIC16F62X

TABLE 14-7: INITIALIZATION CONDITION FOR REGISTERS

Register	Address	Power-on Reset	<ul style="list-style-type: none"> <li>• MCLR Reset during normal operation</li> <li>• MCLR Reset during SLEEP</li> <li>• WDT Reset</li> <li>• Brown-out Detect <sup>(1)</sup></li> </ul>	<ul style="list-style-type: none"> <li>• Wake up from SLEEP through interrupt</li> <li>• Wake up from SLEEP through WDT time-out</li> </ul>
W	-	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	00h	-	-	-
TMR0	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h	0000 0000	0000 0000	PC + 1 <sup>(3)</sup>
STATUS	03h	0001 1xxx	000q quuu <sup>(4)</sup>	uuuq quuu <sup>(4)</sup>
FSR	04h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	xxxx 0000	xxxx u000	xxxx 0000
PORTB	06h	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	10h	--00 0000	--uu uuuu	
T2CON	12h	-000 0000	-000 0000	
CCP1CON	17h	--00 0000	--00 0000	
RCSTA	18h	0000 -00x	0000 -00x	
CMCON	1Fh	0000 0000	0000 0000	uu-- uuuu
PCLATH	0Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh	0000 000x	0000 000u	uuuu u <u>q</u> q <sup>(2)</sup>
PIR1	0Ch	0000 -000	0000 -000	-q-- ---- <sup>(2,5)</sup>
OPTION	81h	1111 1111	1111 1111	uuuu uuuu
TRISA	85h	11-1 1111	11-- 1111	uu-u uuuu
TRISB	86h	1111 1111	1111 1111	uuuu uuuu
PIE1	8Ch	0000 -000	0000 -000	uuuu -uuu
PCON	8Eh	---- 1-0x	---- 1-uq <sup>(1,6)</sup>	---- --uu
TXSTA	98h	0000 -010	0000 -010	
EECON1	9Ch	---- x000	---- q000	
VRCON	9Fh	000- 0000	000- 0000	uuu- uuuu

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

**Note 1:** If VDD goes too low, Power-on Reset will be activated and registers will be affected differently.

**2:** One or more bits in INTCON, PIR1 and/or PIR2 will be affected (to cause wake-up).

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

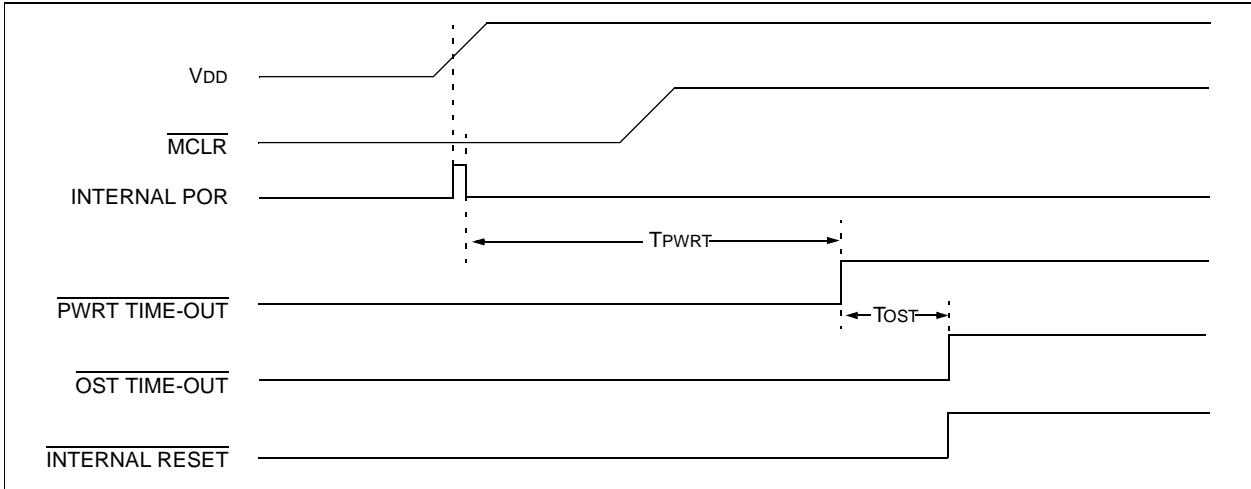
**4:** See Table 14-6 for reset value for specific condition.

**5:** If wake-up was due to comparator input changing, then bit 6 = 1. All other interrupts generating a wake-up will cause bit 6 = u.

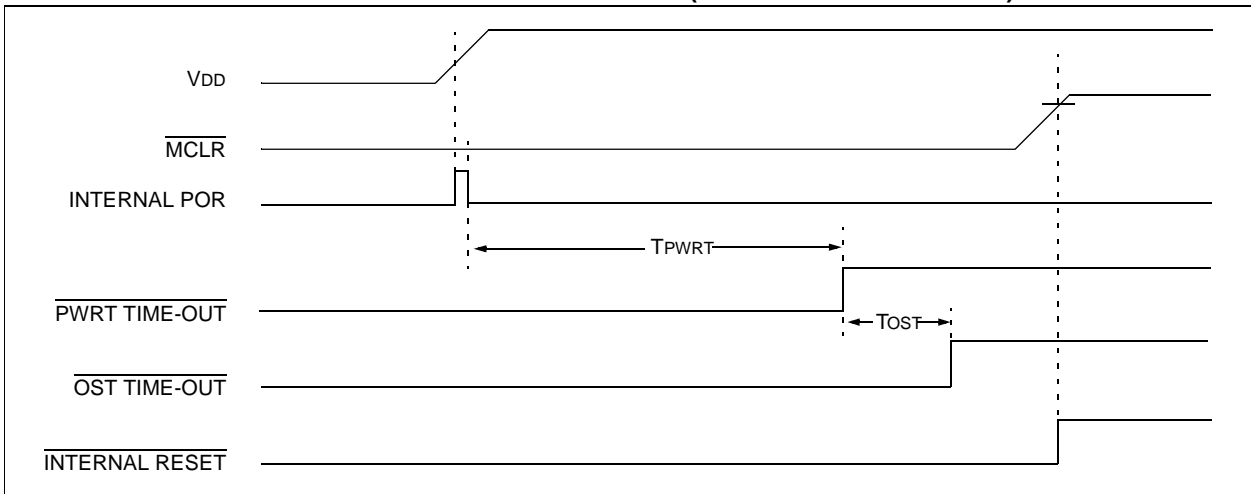
**6:** If reset was due to brown-out, then bit 0 = 0. All other resets will cause bit 0 = u.



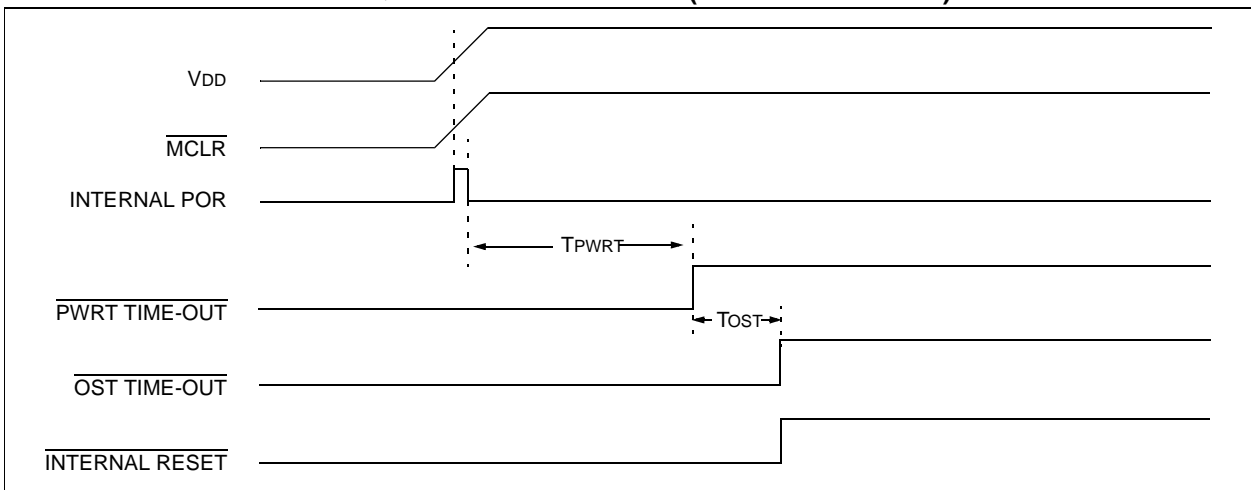
**FIGURE 14-10: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{\text{DD}}$ ): CASE 1**



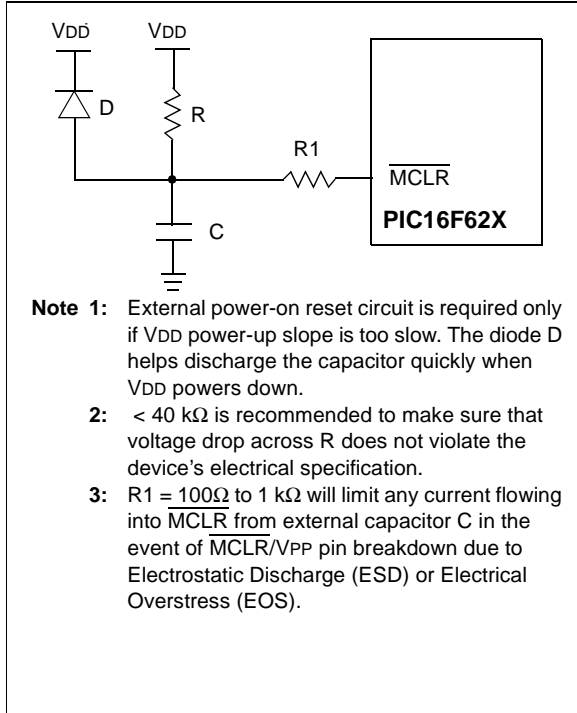
**FIGURE 14-11: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{\text{DD}}$ ): CASE 2**



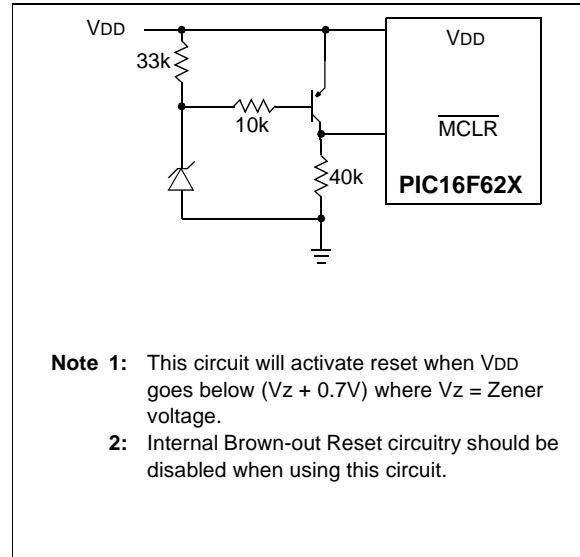
**FIGURE 14-12: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{\text{DD}}$ )**



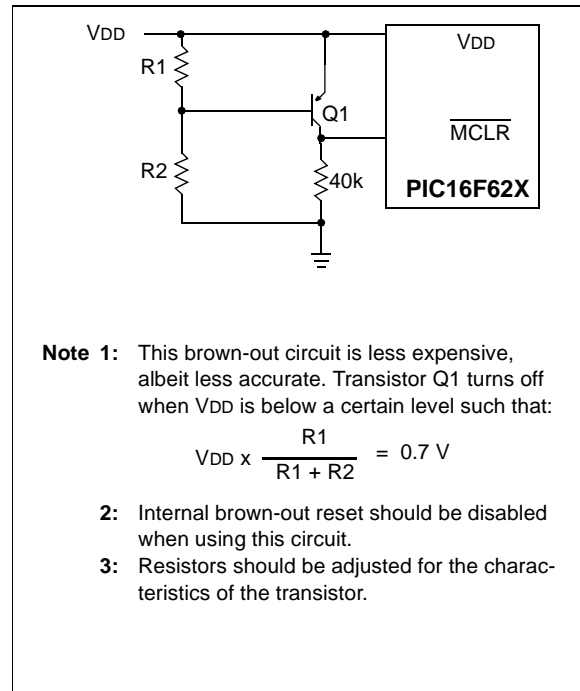
**FIGURE 14-13: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW V<sub>DD</sub> POWER-UP)**



**FIGURE 14-14: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 1**



**FIGURE 14-15: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 2**



## 14.6 Interrupts

The PIC16F62X has 10 sources of interrupt:

- External Interrupt RB0/INT
- TMR0 Overflow Interrupt
- PortB Change Interrupts (pins RB7:RB4)
- Comparator Interrupt
- USART Interrupt
- CCP Interrupt
- TMR1 Overflow Interrupt
- TMR2 Match 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.

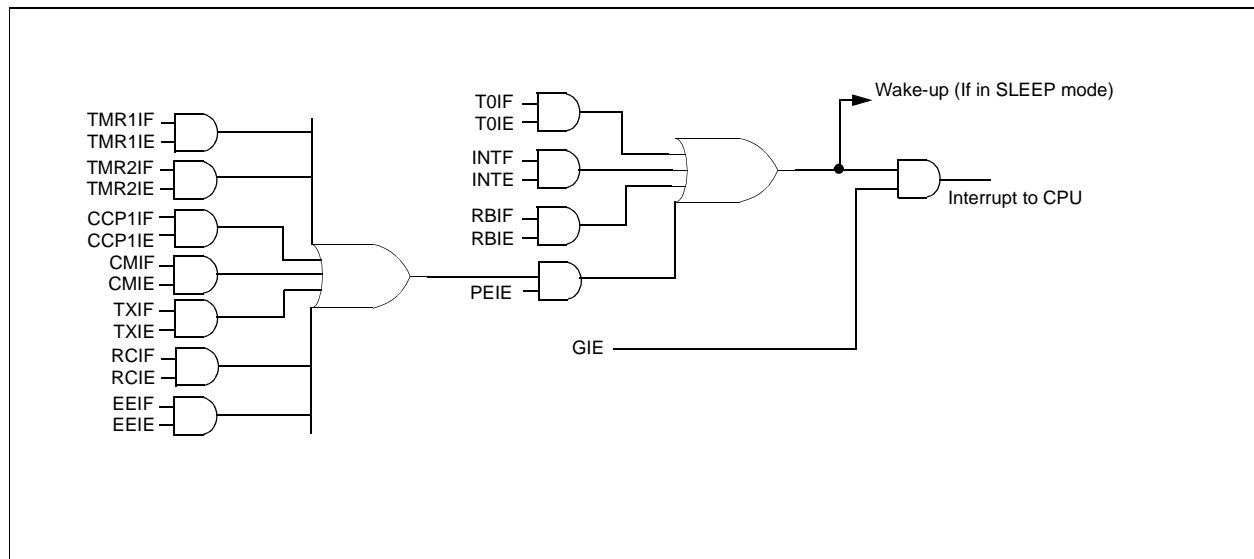
When an interrupt is responded to, the GIE is cleared to disable any further interrupt, the return address is pushed into the stack and the PC is loaded with 0004h. Once in the interrupt service routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt 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 14-17). 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. Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

**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 14-16: INTERRUPT LOGIC**



# PIC16F62X

## 14.6.1 RB0/INT INTERRUPT

External interrupt on RB0/INT pin is edge triggered: either rising if INTEDG bit (OPTION<6>) is set, or falling, if INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, the INTF bit (INTCON<1>) is set. This interrupt can be disabled by clearing the INTE control bit (INTCON<4>). The INTF bit must be cleared in software in the interrupt service routine before re-enabling this interrupt. The RB0/INT interrupt can wake-up the processor from SLEEP, if the INTE bit was set prior to going into SLEEP. The status of the GIE bit decides whether or not the processor branches to the interrupt vector following wake-up. See Section 14.9 for details on SLEEP and Figure 14-19 for timing of wake-up from SLEEP through RB0/INT interrupt.

## 14.6.2 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set the T0IF (INTCON<2>) bit. The interrupt can be enabled/disabled by setting/clearing T0IE (INTCON<5>) bit. For operation of the Timer0 module, see Section 6.0.

## 14.6.3 PORTB INTERRUPT

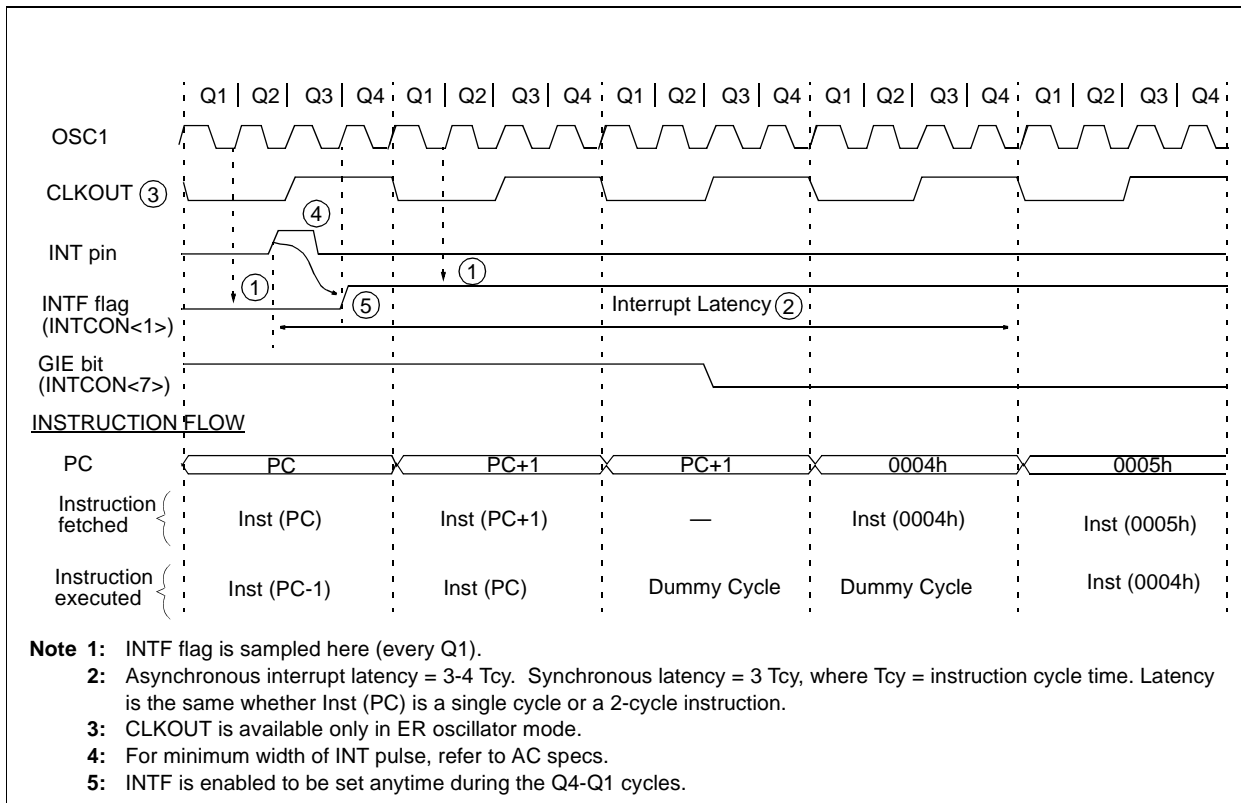
An input change on PORTB <7:4> sets the RBIF (INTCON<0>) bit. The interrupt can be enabled/disabled by setting/clearing the RBIE (INTCON<4>) bit. For operation of PORTB (Section 5.2).

**Note:** If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

## 14.6.4 COMPARATOR INTERRUPT

See Section 9.6 for complete description of comparator interrupts.

**FIGURE 14-17: INT PIN INTERRUPT TIMING**



**TABLE 14-8: SUMMARY OF INTERRUPT REGISTERS**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets <sup>(1)</sup>
0Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000

**Note 1:** Other (non power-up) resets include MCLR reset, Brown-out Reset and Watchdog Timer Reset during normal operation.

## 14.7 Context Saving During Interrupts

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

Example 14-1 stores and restores the STATUS and W registers. The user register, W\_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., W\_TEMP is defined at 0x20 in Bank 0 and it must also be defined at 0xA0 in Bank 1). The user register, STATUS\_TEMP, must be defined in Bank 0. The Example 14-1:

- Stores the W register
- Stores the STATUS register in Bank 0
- Executes the ISR code
- Restores the STATUS (and bank select bit register)
- Restores the W register

### EXAMPLE 14-1: SAVING THE STATUS AND W REGISTERS IN RAM

```

MOVWF    W_TEMP        ;copy W to temp register,
                       ;could be in either bank

SWAPF    STATUS,W      ;swap status to be saved into W

BCF      STATUS,RP0    ;change to bank 0 regardless
                       ;of current bank

MOVWF    STATUS_TEMP   ;save status to bank 0
                       ;register

:
:   (ISR)
:

SWAPF    STATUS_TEMP,W ;swap STATUS_TEMP register
                       ;into W, sets bank to original
                       ;state

MOVWF    STATUS        ;move W into STATUS register

SWAPF    W_TEMP,F      ;swap W_TEMP

SWAPF    W_TEMP,W      ;swap W_TEMP into W
    
```

## 14.8 Watchdog Timer (WDT)

The watchdog timer is a free running on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the ER oscillator of the CLKIN pin. That means that the WDT will run, even if the clock on the OSC1 and OSC2 pins of the device has been stopped, for example, by execution of a SLEEP instruction. During normal operation, a WDT time-out generates a device RESET. If the device is in SLEEP mode, a WDT time-out causes the device to wake-up and continue with normal operation. The WDT can be permanently disabled by programming the configuration bit WDTE as clear (Section 14.1).

### 14.8.1 WDT PERIOD

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

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

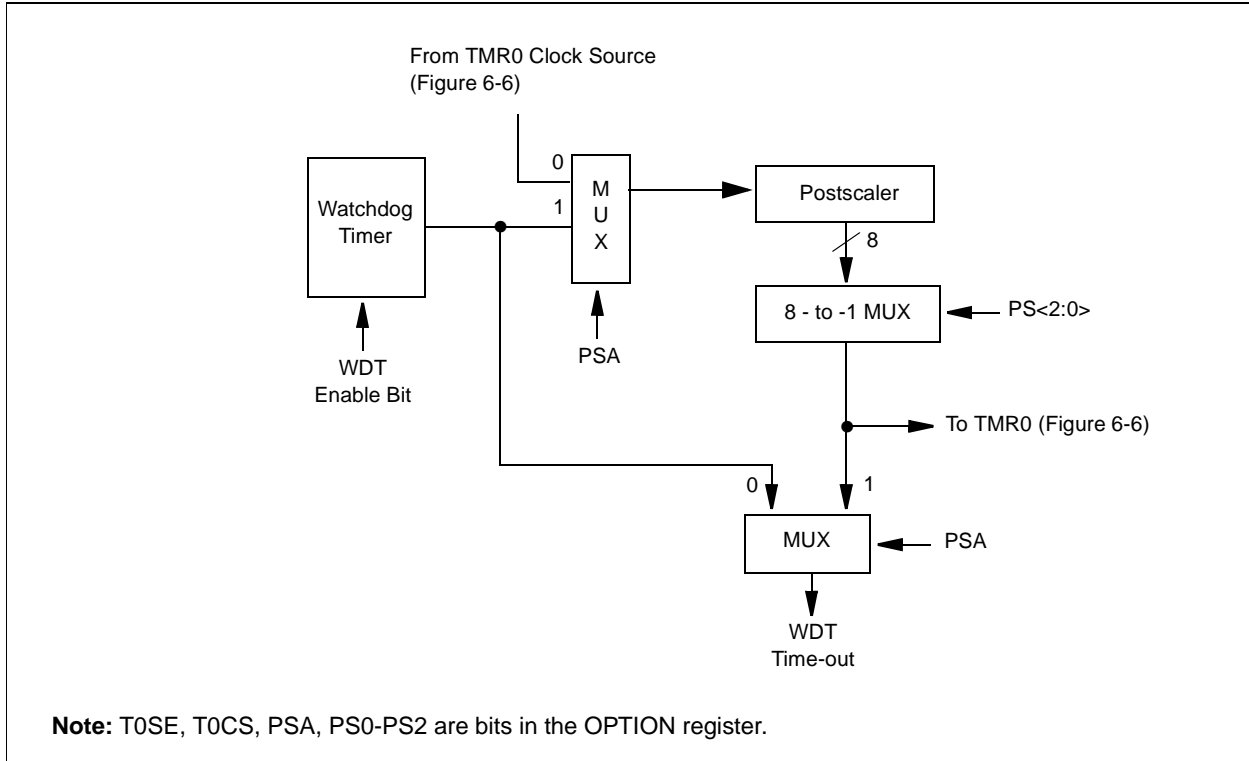
The  $\overline{\text{TO}}$  bit in the STATUS register will be cleared upon a Watchdog Timer time-out.

### 14.8.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken in account that under worst case conditions (V<sup>DD</sup> = Min., Temperature = Max., max. WDT prescaler) it may take several seconds before a WDT time-out occurs.

# PIC16F62X

**FIGURE 14-18: WATCHDOG TIMER BLOCK DIAGRAM**



**TABLE 14-9: SUMMARY OF WATCHDOG TIMER REGISTERS**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other Resets
2007h	Config. bits	LVP	BOREN	MCLRE	FOSC2	PWRTE	WDTE	FOSC1	FOSC0	uuuu uuuu	uuuu uuuu
81h	OPTION	RBP $\bar{U}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: Shaded cells are not used by the Watchdog Timer.

**Note:** - = Unimplemented location, read as "0"  
 + = Reserved for future use

## 14.9 Power-Down Mode (SLEEP)

The Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the  $\overline{PD}$  bit in the STATUS register is cleared, the  $\overline{TO}$  bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had, before SLEEP was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, all I/O pins should be either at VDD, or VSS, with no external circuitry drawing current from the I/O pin and the comparators and VREF should be disabled. I/O pins that are hi-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at VDD or VSS for lowest current consumption. The contribution from on chip pull-ups on PORTB should be considered.

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

**Note:** It should be noted that a RESET generated by a WDT time-out does not drive  $\overline{MCLR}$  pin low.

The first event will cause a device reset. The two latter events are considered a continuation of program execution. The  $\overline{TO}$  and  $\overline{PD}$  bits in the STATUS register can be used to determine the cause of device reset.  $\overline{PD}$  bit, which is set on power-up is cleared when SLEEP is invoked.  $\overline{TO}$  bit is cleared if WDT Wake-up occurred.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

**Note:** If the global interrupts are disabled (GIE is cleared), but any interrupt source has both its interrupt enable bit and the corresponding interrupt flag bits set, the device will immediately wakeup from sleep. The sleep instruction is completely executed.

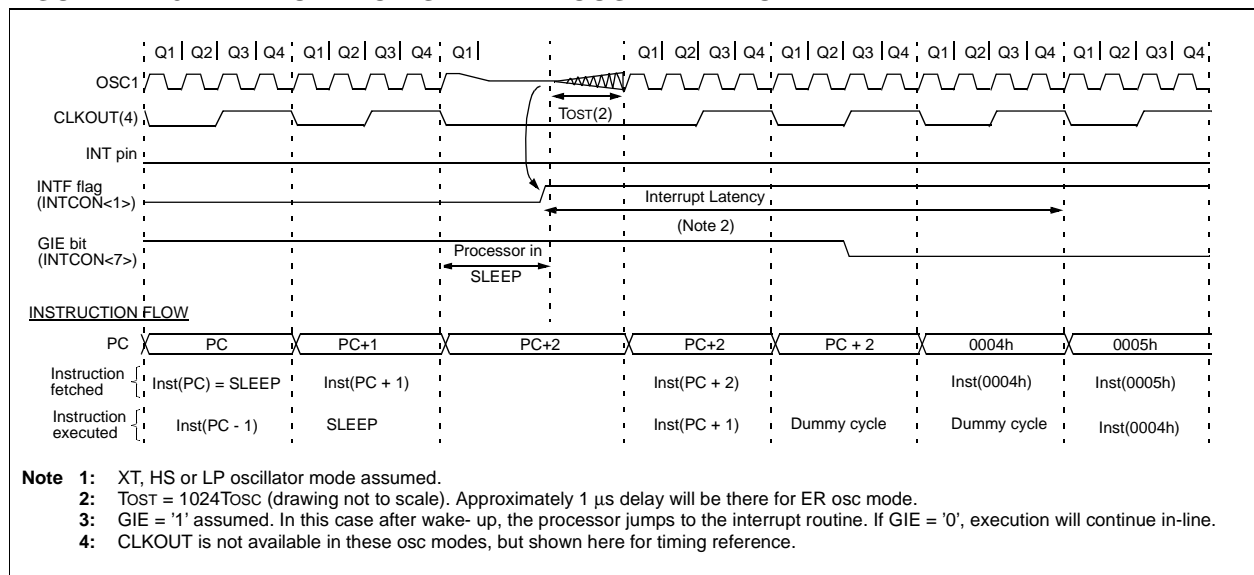
### 14.9.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

1. External reset input on  $\overline{MCLR}$  pin
2. Watchdog Timer Wake-up (if WDT was enabled)
3. Interrupt from RB0/INT pin, RB Port change, or the Peripheral Interrupt (Comparator).

The WDT is cleared when the device wakes-up from sleep, regardless of the source of wake-up.

**FIGURE 14-19: WAKE-UP FROM SLEEP THROUGH INTERRUPT**



# PIC16F62X

## 14.10 Code Protection

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

**Note:** The entire data EEPROM and FLASH program memory will be erased when the code protection is turned off. The INTRC calibration data is not erased.

## 14.11 ID Locations

Four memory locations (2000h-2003h) are designated as ID locations where the user can store checksum or other code-identification numbers. These locations are not accessible during normal execution but are readable and writable during program/verify. Only the least significant 4 bits of the ID locations are used.

## 14.12 In-Circuit Serial Programming

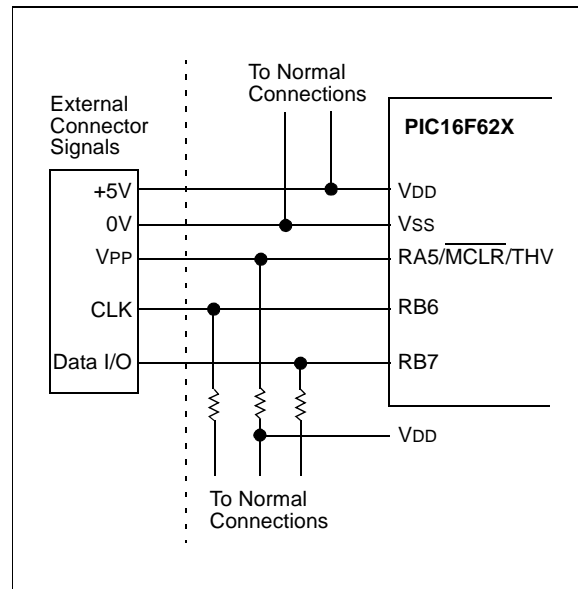
The PIC16F62X microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground, and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

The device is placed into a program/verify mode by holding the RB6 and RB7 pins low while raising the MCLR (VPP) pin from VIL to VIH (see programming specification). RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode.

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

A typical in-circuit serial programming connection is shown in Figure 14-20.

**FIGURE 14-20: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION**



## 14.13 Low Voltage Programming

The LVP bit of the configuration word, enables the low voltage programming. This mode allows the microcontroller to be programmed via ICSP using only a 5V source. This mode removes the requirement of VIH on the MCLR pin. The LVP bit is normally erased to '1' which enables the low voltage programming. In this mode, the RB4/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. The device will enter programming mode when a '1' is placed on the RB4/PGM pin. The HV programming mode is still available by placing VIH on the MCLR pin.

- Note 1:** While in this mode the RB4 pin can no longer be used as a general purpose I/O pin.
- 2:** VDD must be 5.0V  $\pm$ 10% during erase/program operations while in low voltage programming mode.

If Low-voltage programming mode is not used, the LVP bit can be programmed to a '0' and RB4/PGM becomes a digital I/O pin. To program the device, VIH must be placed onto MCLR during programming. The LVP bit may only be programmed when programming is entered with VIH on MCLR. The LVP bit cannot be programmed when programming is entered with RB4/PGM.

It should be noted, that once the LVP bit is programmed to 0, only the high voltage programming mode is available and only high voltage programming mode can be used to program the device.



## 15.0 INSTRUCTION SET SUMMARY

Each PIC16F62X instruction is a 14-bit word divided into an OPCODE which specifies the instruction type and one or more operands which further specify the operation of the instruction. The PIC16F62X instruction set summary in Table 15-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 15-1 shows the opcode field descriptions.

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

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

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

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

**TABLE 15-1: OPCODE FIELD DESCRIPTIONS**

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

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

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

All instructions are executed within one single instruction cycle, unless a conditional test is true or the program counter is changed as a result of an instruction. In this case, the execution takes two instruction cycles with the second cycle executed as a NOP. One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs.

Table 15-1 lists the instructions recognized by the MPASM assembler.

Figure 15-1 shows the three general formats that the instructions can have.

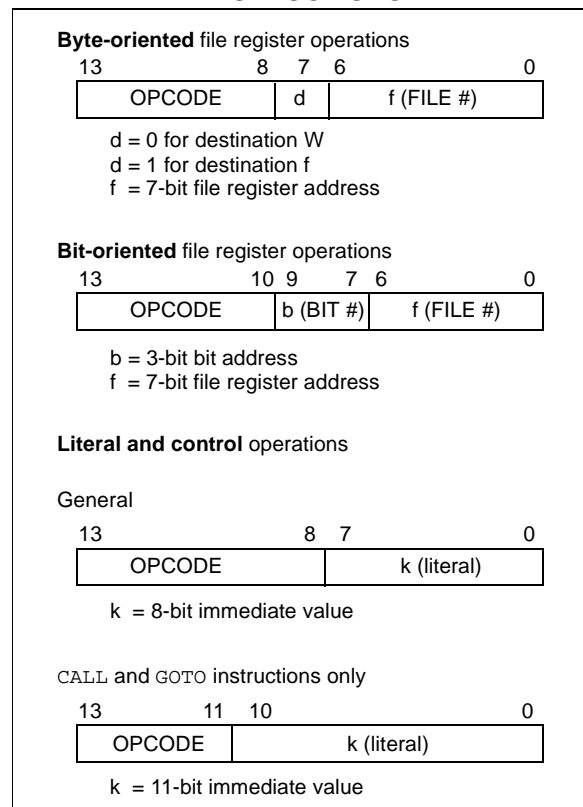
**Note:** To maintain upward compatibility with future PICmicro® products, do not use the OPTION and TRIS instructions.

All examples use the following format to represent a hexadecimal number:

0xhh

where h signifies a hexadecimal digit.

**FIGURE 15-1: GENERAL FORMAT FOR INSTRUCTIONS**



# PIC16F62X

**TABLE 15-2: PIC16F62X INSTRUCTION SET**

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

## 15.1 Instruction Descriptions

<b>ADDLW</b>	<b>Add Literal and W</b>				
Syntax:	[ <i>label</i> ] ADDLW <i>k</i>				
Operands:	$0 \leq k \leq 255$				
Operation:	$(W) + k \rightarrow (W)$				
Status Affected:	C, DC, Z				
Encoding:	<table border="1"> <tr> <td>11</td> <td>111x</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	111x	kkkk	kkkk
11	111x	kkkk	kkkk		
Description:	The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.				
Words:	1				
Cycles:	1				
Example	<pre>ADDLW    0x15 Before Instruction       W = 0x10 After Instruction       W = 0x25</pre>				

<b>ANDLW</b>	<b>AND Literal with W</b>				
Syntax:	[ <i>label</i> ] ANDLW <i>k</i>				
Operands:	$0 \leq k \leq 255$				
Operation:	$(W) .AND. (k) \rightarrow (W)$				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>11</td> <td>1001</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	1001	kkkk	kkkk
11	1001	kkkk	kkkk		
Description:	The contents of W register are AND'ed with the eight bit literal 'k'. The result is placed in the W register.				
Words:	1				
Cycles:	1				
Example	<pre>ANDLW    0x5F Before Instruction       W = 0xA3 After Instruction       W = 0x03</pre>				

<b>ADDWF</b>	<b>Add W and f</b>				
Syntax:	[ <i>label</i> ] ADDWF <i>f,d</i>				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	$(W) + (f) \rightarrow (dest)$				
Status Affected:	C, DC, Z				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0111</td> <td>dfff</td> <td>ffff</td> </tr> </table>	00	0111	dfff	ffff
00	0111	dfff	ffff		
Description:	Add the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.				
Words:	1				
Cycles:	1				
Example	<pre>ADDWF    FSR, 0 Before Instruction       W = 0x17       FSR = 0xC2 After Instruction       W = 0xD9       FSR = 0xC2</pre>				

<b>ANDWF</b>	<b>AND W with f</b>				
Syntax:	[ <i>label</i> ] ANDWF <i>f,d</i>				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	$(W) .AND. (f) \rightarrow (dest)$				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0101</td> <td>dfff</td> <td>ffff</td> </tr> </table>	00	0101	dfff	ffff
00	0101	dfff	ffff		
Description:	AND the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.				
Words:	1				
Cycles:	1				
Example	<pre>ANDWF    FSR, 1 Before Instruction       W = 0x17       FSR = 0xC2 After Instruction       W = 0x17       FSR = 0x02</pre>				

# PIC16F62X

## BCF Bit Clear f

Syntax: [ *label* ] BCF f,b  
 Operands:  $0 \leq f \leq 127$   
 $0 \leq b \leq 7$   
 Operation:  $0 \rightarrow (f<b>)$   
 Status Affected: None  
 Encoding: 

01	00bb	bfff	ffff
----	------	------	------

  
 Description: Bit 'b' in register 'f' is cleared.  
 Words: 1  
 Cycles: 1  
 Example      BCF      FLAG\_REG, 7

Before Instruction  
                   FLAG\_REG = 0xC7  
 After Instruction  
                   FLAG\_REG = 0x47

## BTFSC Bit Test, Skip if Clear

Syntax: [ *label* ] BTFSC f,b  
 Operands:  $0 \leq f \leq 127$   
 $0 \leq b \leq 7$   
 Operation: skip if (f<b>) = 0  
 Status Affected: None  
 Encoding: 

01	10bb	bfff	ffff
----	------	------	------

  
 Description: If bit 'b' in register 'f' is '0' then the next instruction is skipped.  
 If bit 'b' is '0' then the next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead, making this a two-cycle instruction.  
 Words: 1  
 Cycles: 1(2)  
 Example      HERE      BTFSC    FLAG, 1  
                   FALSE    GOTO    PROCESS\_CODE  
                   TRUE      •  
                               •  
                               •

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

## BSF Bit Set f

Syntax: [ *label* ] BSF f,b  
 Operands:  $0 \leq f \leq 127$   
 $0 \leq b \leq 7$   
 Operation:  $1 \rightarrow (f<b>)$   
 Status Affected: None  
 Encoding: 

01	01bb	bfff	ffff
----	------	------	------

  
 Description: Bit 'b' in register 'f' is set.  
 Words: 1  
 Cycles: 1  
 Example      BSF      FLAG\_REG, 7

Before Instruction  
                   FLAG\_REG = 0x0A  
 After Instruction  
                   FLAG\_REG = 0x8A

## **BTFSS**      **Bit Test f, Skip if Set**

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

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

**Operation:**    skip if (f<b>) = 1

**Status Affected:** None

**Encoding:**

01	11bb	bfff	ffff
----	------	------	------

**Description:** If bit 'b' in register 'f' is '1' then the next instruction is skipped. 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.

**Words:**        1

**Cycles:**       1(2)

**Example**

```

HERE   BTFSS  FLAG,1
FALSE  GOTO  PROCESS_CODE
TRUE   .
      .
      .

```

**Before Instruction**  
PC = address HERE

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

## **CALL**            **Call Subroutine**

**Syntax:**      [ *label* ] CALL k

**Operands:**     $0 \leq k \leq 2047$

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

**Status Affected:** None

**Encoding:**

10	0kkk	kkkk	kkkk
----	------	------	------

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

**Words:**        1

**Cycles:**       2

**Example**

```

HERE   CALL  THERE

```

**Before Instruction**  
PC = Address HERE

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

## **CLRF**            **Clear f**

**Syntax:**      [ *label* ] CLRF f

**Operands:**     $0 \leq f \leq 127$

**Operation:**    00h → (f)  
1 → Z

**Status Affected:** Z

**Encoding:**

00	0001	1fff	ffff
----	------	------	------

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

**Words:**        1

**Cycles:**       1

**Example**

```

CLRF   FLAG_REG

```

**Before Instruction**  
FLAG\_REG = 0x5A

**After Instruction**  
FLAG\_REG = 0x00  
Z = 1

## **CLRW**            **Clear W**

**Syntax:**      [ *label* ] CLRW

**Operands:**    None

**Operation:**    00h → (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

# PIC16F62X

## 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

## DECFSZ Decrement f, Skip if 0

Syntax: [ *label* ] DECFSZ f,d

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

Operation:  $(f) - 1 \rightarrow (\text{dest});$  skip if result = 0

Status Affected: None

Encoding:	00	1011	dfff	ffff
-----------	----	------	------	------

Description: The contents of register 'f' are decremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'. If the result is 0, the next instruction, which is already fetched, is discarded. A NOP is executed instead making it a two-cycle instruction.

Words: 1

Cycles: 1(2)

Example

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

Before Instruction  
PC = address HERE

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

**GOTO Unconditional Branch**

Syntax: `[label] GOTO k`

Operands:  $0 \leq k \leq 2047$

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

Status Affected: None

Encoding: 

10	1kkk	kkkk	kkkk
----	------	------	------

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

Words: 1

Cycles: 2

Example `GOTO THERE`

After Instruction  
 $PC = \text{Address } THERE$

**INCFSZ Increment f, Skip if 0**

Syntax: `[label] INCFSZ f,d`

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

Operation:  $(f) + 1 \rightarrow (\text{dest})$ , 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 0, the next instruction, which is already fetched, is discarded. A NOP is executed instead making it a two-cycle instruction.

Words: 1

Cycles: 1(2)

Example

```

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

Before Instruction  
 $PC = \text{address } HERE$

After Instruction  
 $CNT = CNT + 1$   
 if  $CNT = 0$ ,  
 $PC = \text{address } CONTINUE$   
 if  $CNT \neq 0$ ,  
 $PC = \text{address } HERE + 1$

**INCF Increment f**

Syntax: `[label] INCF f,d`

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

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

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

Example `INCF CNT, 1`

Before Instruction  
 $CNT = 0xFF$   
 $Z = 0$

After Instruction  
 $CNT = 0x00$   
 $Z = 1$

**IORLW Inclusive OR Literal with W**

Syntax: `[label] IORLW k`

Operands:  $0 \leq k \leq 255$

Operation:  $(W) .OR. k \rightarrow (W)$

Status Affected: Z

Encoding: 

11	1000	kkkk	kkkk
----	------	------	------

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

Words: 1

Cycles: 1

Example `IORLW 0x35`

Before Instruction  
 $W = 0x9A$

After Instruction  
 $W = 0xBF$   
 $Z = 1$

# PIC16F62X

**IORWF**                    **Inclusive OR W with f**

---

Syntax:                    [ *label* ] IORWF f,d

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

Operation:                (W) .OR. (f)  $\rightarrow$  (dest)

Status Affected:        Z

Encoding:                

00	0100	dfff	ffff
----	------	------	------

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

Words:                    1

Cycles:                    1

Example                    IORWF                RESULT, 0

                              Before Instruction

                              RESULT = 0x13

                              W = 0x91

                              After Instruction

                              RESULT = 0x13

                              W = 0x93

                              Z = 1

**MOVF**                    **Move f**

---

Syntax:                    [ *label* ] MOVF f,d

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

Operation:                (f)  $\rightarrow$  (dest)

Status Affected:        Z

Encoding:                

00	1000	dfff	ffff
----	------	------	------

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

Words:                    1

Cycles:                    1

Example                    MOVF                FSR, 0

                              After Instruction

                              W = value in FSR register

                              Z = 1

**MOVLW**                   **Move Literal to W**

---

Syntax:                    [ *label* ] MOVLW k

Operands:                 $0 \leq k \leq 255$

Operation:                 $k \rightarrow$  (W)

Status Affected:        None

Encoding:                

11	00xx	kkkk	kkkk
----	------	------	------

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

Words:                    1

Cycles:                    1

Example                    MOVLW              0x5A

                              After Instruction

                              W = 0x5A

**MOVWF**                   **Move W to f**

---

Syntax:                    [ *label* ] MOVWF f

Operands:                 $0 \leq f \leq 127$

Operation:                (W)  $\rightarrow$  (f)

Status Affected:        None

Encoding:                

00	0000	1fff	ffff
----	------	------	------

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

Words:                    1

Cycles:                    1

Example                    MOVWF                OPTION

                              Before Instruction

                              OPTION = 0xFF

                              W = 0x4F

                              After Instruction

                              OPTION = 0x4F

                              W = 0x4F



<b>NOP</b>	<b>No Operation</b>				
Syntax:	[ <i>label</i> ] NOP				
Operands:	None				
Operation:	No operation				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0xx0</td> <td>0000</td> </tr> </table>	00	0000	0xx0	0000
00	0000	0xx0	0000		
Description:	No operation.				
Words:	1				
Cycles:	1				
Example	NOP				

<b>RETFIE</b>	<b>Return from Interrupt</b>				
Syntax:	[ <i>label</i> ] RETFIE				
Operands:	None				
Operation:	TOS → PC, 1 → GIE				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0000</td> <td>1001</td> </tr> </table>	00	0000	0000	1001
00	0000	0000	1001		
Description:	Return from Interrupt. Stack is POPed and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a two-cycle instruction.				
Words:	1				
Cycles:	2				
Example	RETFIE				

After Interrupt  
 PC = TOS  
 GIE = 1

<b>OPTION</b>	<b>Load Option Register</b>				
Syntax:	[ <i>label</i> ] OPTION				
Operands:	None				
Operation:	(W) → OPTION				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0110</td> <td>0010</td> </tr> </table>	00	0000	0110	0010
00	0000	0110	0010		
Description:	The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it.				
Words:	1				
Cycles:	1				
Example	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <p><b>To maintain upward compatibility with future PICmicro® products, do not use this instruction.</b></p> </div>				

<b>RETLW</b>	<b>Return with Literal in W</b>				
Syntax:	[ <i>label</i> ] RETLW k				
Operands:	0 ≤ k ≤ 255				
Operation:	k → (W); TOS → PC				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>11</td> <td>01xx</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	01xx	kkkk	kkkk
11	01xx	kkkk	kkkk		
Description:	The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.				

Words: 1  
 Cycles: 2

Example

```
CALL TABLE ;W contains table
                ;offset value
                ;W now has table
value
.
.
TABLE
ADDWF PC ;W = offset
RETLW k1 ;Begin table
RETLW k2 ;
.
.
RETLW kn ; End of table
```

Before Instruction  
 W = 0x07  
 After Instruction  
 W = value of k8

# PIC16F62X

## RETURN Return from Subroutine

Syntax: `[label] RETURN`

Operands: None

Operation: TOS → PC

Status Affected: None

Encoding: 

00	0000	0000	1000
----	------	------	------

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two cycle instruction.

Words: 1

Cycles: 2

Example

```
RETURN
After Interrupt
      PC = TOS
```

## RRF Rotate Right f through Carry

Syntax: `[label] RRF f,d`

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

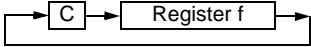
Operation: See description below

Status Affected: C

Encoding: 

00	1100	dfff	ffff
----	------	------	------

Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.



Words: 1

Cycles: 1

Example

```
RRF      REG1,0
```

Before Instruction

```
REG1 = 1110 0110
C     = 0
```

After Instruction

```
REG1 = 1110 0110
W     = 0111 0011
C     = 0
```

## RLF Rotate Left f through Carry

Syntax: `[label] RLF f,d`

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

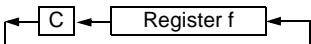
Operation: See description below

Status Affected: C

Encoding: 

00	1101	dfff	ffff
----	------	------	------

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.



Words: 1

Cycles: 1

Example

```
RLF      REG1,0
```

Before Instruction

```
REG1 = 1110 0110
C     = 0
```

After Instruction

```
REG1 = 1110 0110
W     = 1100 1100
C     = 1
```

## SLEEP

Syntax: `[label] SLEEP`

Operands: None

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

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

Encoding: 

00	0000	0110	0011
----	------	------	------

Description: The power-down status bit,  $\overline{PD}$  is cleared. Time-out status bit,  $\overline{TO}$  is set. Watchdog Timer and its prescaler are cleared. The processor is put into SLEEP mode with the oscillator stopped. See Section 14.9 for more details.

Words: 1

Cycles: 1

Example: SLEEP

## SUBLW Subtract W from Literal

Syntax: [ *label* ] SUBLW *k*

Operands:  $0 \leq k \leq 255$

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

Status: C, DC, Z

Affected:

Encoding: 

11	110x	kkkk	kkkk
----	------	------	------

Description: The W register is subtracted (2's complement method) from the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Example 1: SUBLW 0x02

Before Instruction

W = 1  
C = ?

After Instruction

W = 1  
C = 1; result is positive

Example 2: Before Instruction

W = 2  
C = ?

After Instruction

W = 0  
C = 1; result is zero

Example 3: Before Instruction

W = 3  
C = ?

After Instruction

W = 0xFF  
C = 0; result is negative

## SUBWF Subtract W from f

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

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

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

Status: C, DC, Z

Affected:

Encoding: 

00	0010	dfff	ffff
----	------	------	------

Description: Subtract (2's complement method) W register from register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example 1: SUBWF REG1, 1

Before Instruction

REG1 = 3  
W = 2  
C = ?

After Instruction

REG1 = 1  
W = 2  
C = 1; result is positive

Example 2: Before Instruction

REG1 = 2  
W = 2  
C = ?

After Instruction

REG1 = 0  
W = 2  
C = 1; result is zero

Example 3: Before Instruction

REG1 = 1  
W = 2  
C = ?

After Instruction

REG1 = 0xFF  
W = 2  
C = 0; result is negative

# PIC16F62X

SWAPF	Swap Nibbles in f				
Syntax:	[ <i>label</i> ] SWAPF f,d				
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]				
Operation:	(f<3:0>) → (dest<7:4>), (f<7:4>) → (dest<3:0>)				
Status Affected:	None				
Encoding:	<table border="1"> <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				

XORLW	Exclusive OR Literal with W				
Syntax:	[ <i>label</i> ] XORLW k				
Operands:	0 ≤ k ≤ 255				
Operation:	(W) .XOR. k → (W)				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>11</td> <td>1010</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	1010	kkkk	kkkk
11	1010	kkkk	kkkk		
Description:	The contents of the W register are XOR'ed with the eight bit literal 'k'. The result is placed in the W register.				
Words:	1				
Cycles:	1				
Example:	XORLW 0xAF  Before Instruction W = 0xB5  After Instruction W = 0x1A				

TRIS	Load TRIS Register				
Syntax:	[ <i>label</i> ] TRIS f				
Operands:	5 ≤ f ≤ 7				
Operation:	(W) → TRIS register f;				
Status Affected:	None				
Encoding:	<table border="1"> <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	<table border="1"> <tr> <td><b>To maintain upward compatibility with future PICmicro® products, do not use this instruction.</b></td> </tr> </table>	<b>To maintain upward compatibility with future PICmicro® products, do not use this instruction.</b>			
<b>To maintain upward compatibility with future PICmicro® products, do not use this instruction.</b>					

XORWF	Exclusive OR W with f				
Syntax:	[ <i>label</i> ] XORWF f,d				
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]				
Operation:	(W) .XOR. (f) → (dest)				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0110</td> <td>dfff</td> <td>ffff</td> </tr> </table>	00	0110	dfff	ffff
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				

## 16.0 DEVELOPMENT SUPPORT

The PICmicro<sup>®</sup> microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB™ IDE Software
- Assemblers/Compilers/Linkers
  - MPASM Assembler
  - MPLAB-C17 and MPLAB-C18 C Compilers
  - MPLINK/MPLIB Linker/Librarian
- Simulators
  - MPLAB-SIM Software Simulator
- Emulators
  - MPLAB-ICE Real-Time In-Circuit Emulator
  - PICMASTER<sup>®</sup>/PICMASTER-CE In-Circuit Emulator
  - ICEPIC™
- In-Circuit Debugger
  - MPLAB-ICD for PIC16F877
- Device Programmers
  - PRO MATE<sup>®</sup> II Universal Programmer
  - PICSTART<sup>®</sup> Plus Entry-Level Prototype Programmer
- Low-Cost Demonstration Boards
  - SIMICE
  - PICDEM-1
  - PICDEM-2
  - PICDEM-3
  - PICDEM-17
  - SEEVAL<sup>®</sup>
  - KEELOQ<sup>®</sup>

### 16.1 MPLAB Integrated Development Environment Software

- The MPLAB IDE software brings an ease of software development previously unseen in the 8-bit microcontroller market. MPLAB is a Windows<sup>®</sup>-based application which contains:
  - Multiple functionality
    - editor
    - simulator
    - programmer (sold separately)
    - emulator (sold separately)
  - A full featured editor
  - A project manager
  - Customizable tool bar and key mapping
  - A status bar
  - On-line help

MPLAB allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PICmicro tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
  - object code

The ability to use MPLAB with Microchip's simulator, MPLAB-SIM, allows a consistent platform and the ability to easily switch from the cost-effective simulator to the full featured emulator with minimal retraining.

### 16.2 MPASM Assembler

MPASM is a full featured universal macro assembler for all PICmicro MCU's. It can produce absolute code directly in the form of HEX files for device programmers, or it can generate relocatable objects for MPLINK.

MPASM has a command line interface and a Windows shell and can be used as a standalone application on a Windows 3.x or greater system. MPASM generates relocatable object files, Intel standard HEX files, MAP files to detail memory usage and symbol reference, an absolute LST file which contains source lines and generated machine code, and a COD file for MPLAB debugging.

MPASM features include:

- MPASM and MPLINK are integrated into MPLAB projects.
- MPASM allows user defined macros to be created for streamlined assembly.
- MPASM allows conditional assembly for multi purpose source files.
- MPASM directives allow complete control over the assembly process.

### 16.3 MPLAB-C17 and MPLAB-C18 C Compilers

The MPLAB-C17 and MPLAB-C18 Code Development Systems are complete ANSI 'C' compilers and integrated development environments for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers, respectively. These compilers provide powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compilers provide symbol information that is compatible with the MPLAB IDE memory display.

## 16.4 MPLINK/MPLIB Linker/Librarian

MPLINK is a relocatable linker for MPASM and MPLAB-C17 and MPLAB-C18. It can link relocatable objects from assembly or C source files along with pre-compiled libraries using directives from a linker script.

MPLIB is a librarian for pre-compiled code to be used with MPLINK. When a routine from a library is called from another source file, only the modules that contains that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. MPLIB manages the creation and modification of library files.

MPLINK features include:

- MPLINK works with MPASM and MPLAB-C17 and MPLAB-C18.
- MPLINK allows all memory areas to be defined as sections to provide link-time flexibility.

MPLIB features include:

- MPLIB makes linking easier because single libraries can be included instead of many smaller files.
- MPLIB helps keep code maintainable by grouping related modules together.
- MPLIB commands allow libraries to be created and modules to be added, listed, replaced, deleted, or extracted.

## 16.5 MPLAB-SIM Software Simulator

The MPLAB-SIM Software Simulator allows code development in a PC host environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file or user-defined key press to any of the pins. The execution can be performed in single step, execute until break, or trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C17 and MPLAB-C18 and MPASM. The Software Simulator offers the flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

## 16.6 MPLAB-ICE High Performance Universal In-Circuit Emulator with MPLAB IDE

The MPLAB-ICE Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers (MCUs). Software control of MPLAB-ICE is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, "make" and download, and source debugging from a single environment.

Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB-ICE allows expansion to support new PICmicro microcontrollers.

The MPLAB-ICE Emulator System has been designed as a real-time emulation system with advanced features that are generally found on more expensive development tools. The PC platform and Microsoft® Windows 3.x/95/98 environment were chosen to best make these features available to you, the end user.

MPLAB-ICE 2000 is a full-featured emulator system with enhanced trace, trigger, and data monitoring features. Both systems use the same processor modules and will operate across the full operating speed range of the PICmicro MCU.

## 16.7 PICMASTER/PICMASTER CE

The PICMASTER system from Microchip Technology is a full-featured, professional quality emulator system. This flexible in-circuit emulator provides a high-quality, universal platform for emulating Microchip 8-bit PICmicro microcontrollers (MCUs). PICMASTER systems are sold worldwide, with a CE compliant model available for European Union (EU) countries.

## 16.8 ICEPIC

ICEPIC is a low-cost in-circuit emulation solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X, and PIC16CXXX families of 8-bit one-time-programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules or daughter boards. The emulator is capable of emulating without target application circuitry being present.

## 16.9 MPLAB-ICD In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB-ICD, is a powerful, low-cost run-time development tool. This tool is based on the flash PIC16F877 and can be used to develop for this and other PICmicro microcontrollers from the PIC16CXXX family. MPLAB-ICD utilizes the In-Circuit Debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming protocol, offers cost-effective in-circuit flash programming and debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time. The MPLAB-ICD is also a programmer for the flash PIC16F87X family.

## **16.10 PRO MATE II Universal Programmer**

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

The PRO MATE II has programmable VDD and VPP supplies which allows it to verify programmed memory at VDD min and VDD max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In stand-alone mode the PRO MATE II can read, verify or program PICmicro devices. It can also set code-protect bits in this mode.

## **16.11 PICSTART Plus Entry Level Development System**

The PICSTART programmer is an easy-to-use, low-cost prototype programmer. It connects to the PC via one of the COM (RS-232) ports. MPLAB Integrated Development Environment software makes using the programmer simple and efficient.

PICSTART Plus supports all PICmicro devices with up to 40 pins. Larger pin count devices such as the PIC16C92X, and PIC17C76X may be supported with an adapter socket. PICSTART Plus is CE compliant.

## **16.12 SIMICE Entry-Level Hardware Simulator**

SIMICE is an entry-level hardware development system designed to operate in a PC-based environment with Microchip's simulator MPLAB-SIM. Both SIMICE and MPLAB-SIM run under Microchip Technology's MPLAB Integrated Development Environment (IDE) software. Specifically, SIMICE provides hardware simulation for Microchip's PIC12C5XX, PIC12CE5XX, and PIC16C5X families of PICmicro 8-bit microcontrollers. SIMICE works in conjunction with MPLAB-SIM to provide non-real-time I/O port emulation. SIMICE enables a developer to run simulator code for driving the target system. In addition, the target system can provide input to the simulator code. This capability allows for simple and interactive debugging without having to manually generate MPLAB-SIM stimulus files. SIMICE is a valuable debugging tool for entry-level system development.

## **16.13 PICDEM-1 Low-Cost PICmicro Demonstration Board**

The PICDEM-1 is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The users can program the sample microcontrollers provided with

the PICDEM-1 board, on a PRO MATE II or PICSTART-Plus programmer, and easily test firmware. The user can also connect the PICDEM-1 board to the MPLAB-ICE emulator and download the firmware to the emulator for testing. Additional prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push-button switches and eight LEDs connected to PORTB.

## **16.14 PICDEM-2 Low-Cost PIC16CXX Demonstration Board**

The PICDEM-2 is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-2 board, on a PRO MATE II programmer or PICSTART-Plus, and easily test firmware. The MPLAB-ICE emulator may also be used with the PICDEM-2 board to test firmware. Additional prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push-button switches, a potentiometer for simulated analog input, a Serial EEPROM to demonstrate usage of the I<sup>2</sup>C bus and separate headers for connection to an LCD module and a keypad.

## **16.15 PICDEM-3 Low-Cost PIC16CXXX Demonstration Board**

The PICDEM-3 is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with a LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-3 board, on a PRO MATE II programmer or PICSTART Plus with an adapter socket, and easily test firmware. The MPLAB-ICE emulator may also be used with the PICDEM-3 board to test firmware. Additional prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include an RS-232 interface, push-button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM-3 board is an LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM-3 provides an additional RS-232 interface and Windows 3.1 software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals.

## **16.16 PICDEM-17**

The PICDEM-17 is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756, PIC17C762, and PIC17C766. All necessary hardware is included to run basic demo programs, which are supplied on a 3.5-inch disk. A programmed sample is included, and the user may erase it and program it with the other sample programs using the PRO MATE II or PICSTART Plus device programmers and easily debug and test the sample code. In addition, PICDEM-17 supports down-loading of programs to and executing out of external FLASH memory on board. The PICDEM-17 is also usable with the MPLAB-ICE or PICMASTER emulator, and all of the sample programs can be run and modified using either emulator. Additionally, a generous prototype area is available for user hardware.

## **16.17 SEEVAL Evaluation and Programming System**

The SEEVAL SEEPROM Designer's Kit supports all Microchip 2-wire and 3-wire Serial EEPROMs. The kit includes everything necessary to read, write, erase or program special features of any Microchip SEEPROM product including Smart Serials™ and secure serials. The Total Endurance™ Disk is included to aid in trade-off analysis and reliability calculations. The total kit can significantly reduce time-to-market and result in an optimized system.

## **16.18 KEELOQ Evaluation and Programming Tools**

KEELOQ evaluation and programming tools support Microchips HCS Secure Data Products. The HCS evaluation kit includes an LCD display to show changing codes, a decoder to decode transmissions, and a programming interface to program test transmitters.



**TABLE 16-1: DEVELOPMENT TOOLS FROM MICROCHIP**

Development Tools	PIC12CXX	PIC14000	PIC16C5X	PIC16C6X	PIC16CXX	PIC16F62X	PIC16C7X	PIC16C7XX	PIC16C8X	PIC16F88X	PIC16C9XX	PIC17C4X	PIC17C7XX	PIC18CXX2	24CXX/ 25CXX/ 93CXX	HCSXX	MCRFXX	MCP2510
MPLAB™ Integrated Development Environment	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				
MPLAB™ C17 Compiler												✓	✓	✓				
MPLAB™ C18 Compiler												✓	✓	✓				
MPASM/MPLINK	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			
MPLAB™-ICE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			
PICMASTER/PICMASTER-CE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			
ICEPIC™ Low-Cost In-Circuit Emulator	✓		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓			
MPLAB-ICD In-Circuit Debugger				✓*			✓*			✓								
PICSTART® Plus Low-Cost Universal Dev. Kit	✓	✓	✓	✓	✓	✓**	✓	✓	✓	✓	✓	✓	✓	✓	✓			
PRO MATE® II Universal Programmer	✓	✓	✓	✓	✓	✓**	✓	✓	✓	✓	✓	✓	✓	✓	✓			
SIMICE	✓		✓															
PICDEM-1			✓				†		✓									
PICDEM-2							†							✓				
PICDEM-3											✓							
PICDEM-14A																		
PICDEM-17		✓										✓						
KEELOQ® Evaluation Kit																✓		
KEELOQ Transponder Kit																✓		
microID™ Programmer's Kit																	✓	
125 kHz microID Developer's Kit																	✓	
125 kHz Anticollision microID Developer's Kit																	✓	
13.56 MHz Anticollision microID Developer's Kit																	✓	
MCP2510 CAN Developer's Kit																	✓	✓

\* Contact the Microchip Technology Inc. web site at [www.microchip.com](http://www.microchip.com) for information on how to use the MPLAB-ICD In-Circuit Debugger (DV164001) with PIC16C62, 63, 64, 65, 72, 73, 74, 76, 77

\*\* Contact Microchip Technology Inc. for availability date.

† Development tool is available on select devices.

# PIC16F62X

---

NOTES:

## 17.0 ELECTRICAL SPECIFICATIONS

### Absolute Maximum Ratings †

Ambient temperature under bias .....	-40 to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on VDD with respect to VSS .....	-0.3 to +6.5V
Voltage on $\overline{\text{MCLR}}$ and RA4 with respect to VSS .....	-0.3 to +14V
Voltage on all other pins with respect to VSS .....	-0.3V to VDD + 0.3V
Total power dissipation (Note 1).....	800 mW
Maximum current out of VSS pin .....	300 mA
Maximum current into VDD pin .....	250 mA
Input clamp current, I <sub>IK</sub> (V <sub>I</sub> < 0 or V <sub>I</sub> > VDD).....	± 20 mA
Output clamp current, I <sub>OK</sub> (V <sub>O</sub> < 0 or V <sub>O</sub> > VDD).....	± 20 mA
Maximum output current sunk by any I/O pin.....	25 mA
Maximum output current sourced by any I/O pin .....	25 mA
Maximum current sunk by PORTA and PORTB.....	200 mA
Maximum current sourced by PORTA and PORTB .....	200 mA

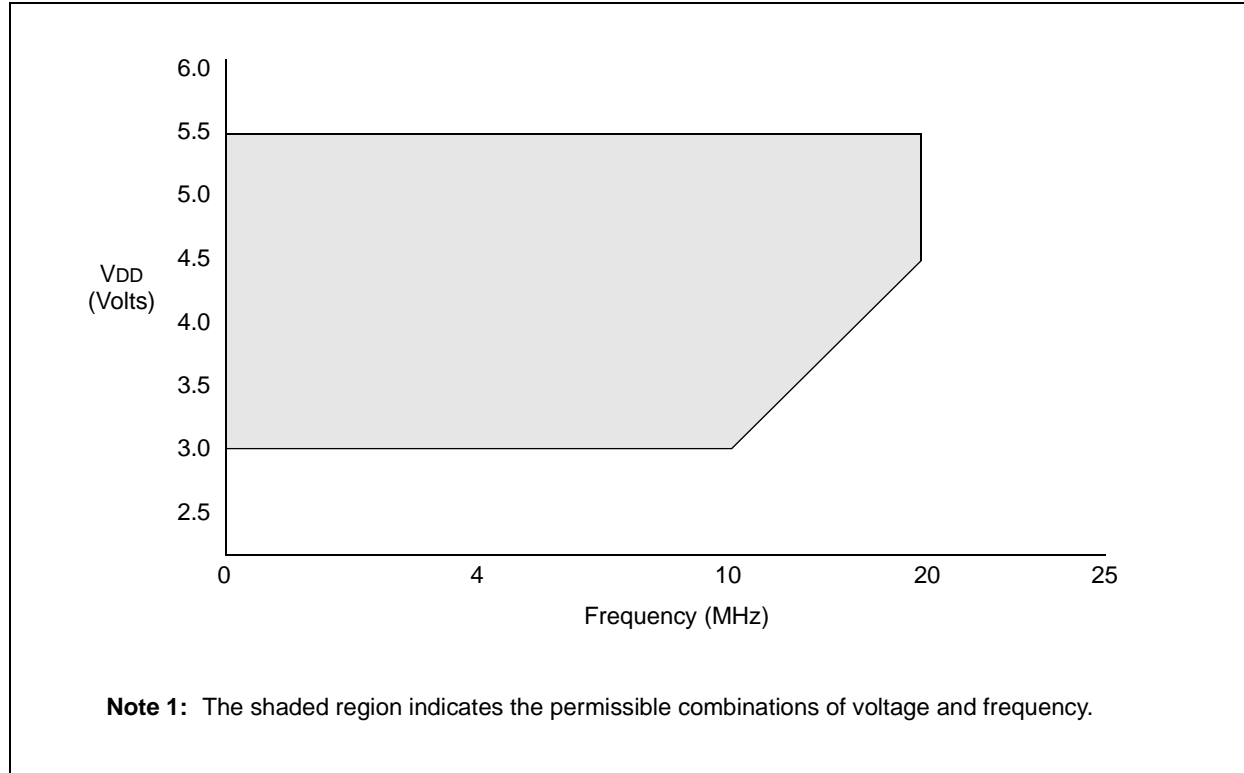
**Note 1:** Power dissipation is calculated as follows:  $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$

† **NOTICE:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

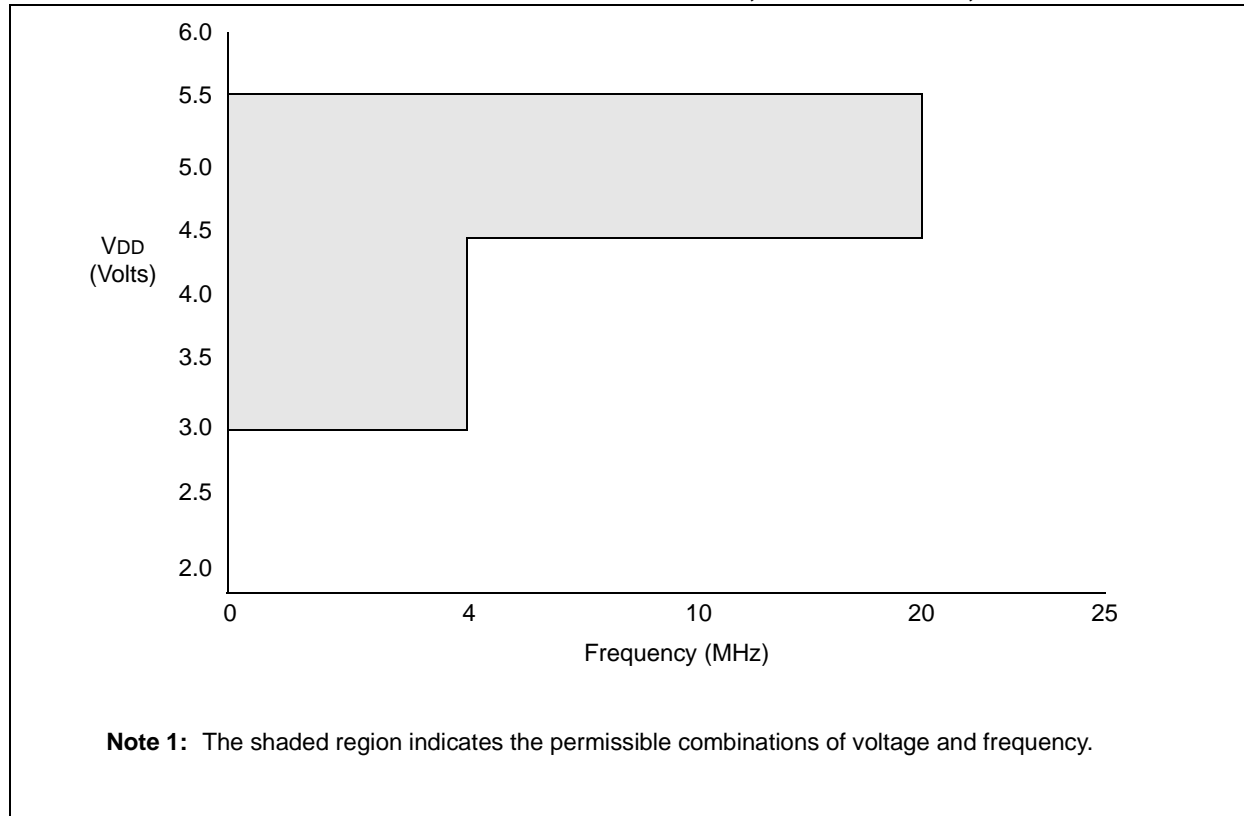
**Note:** Voltage spikes below VSS at the  $\overline{\text{MCLR}}$  pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the  $\overline{\text{MCLR}}$  pin rather than pulling this pin directly to VSS.

# PIC16F62X

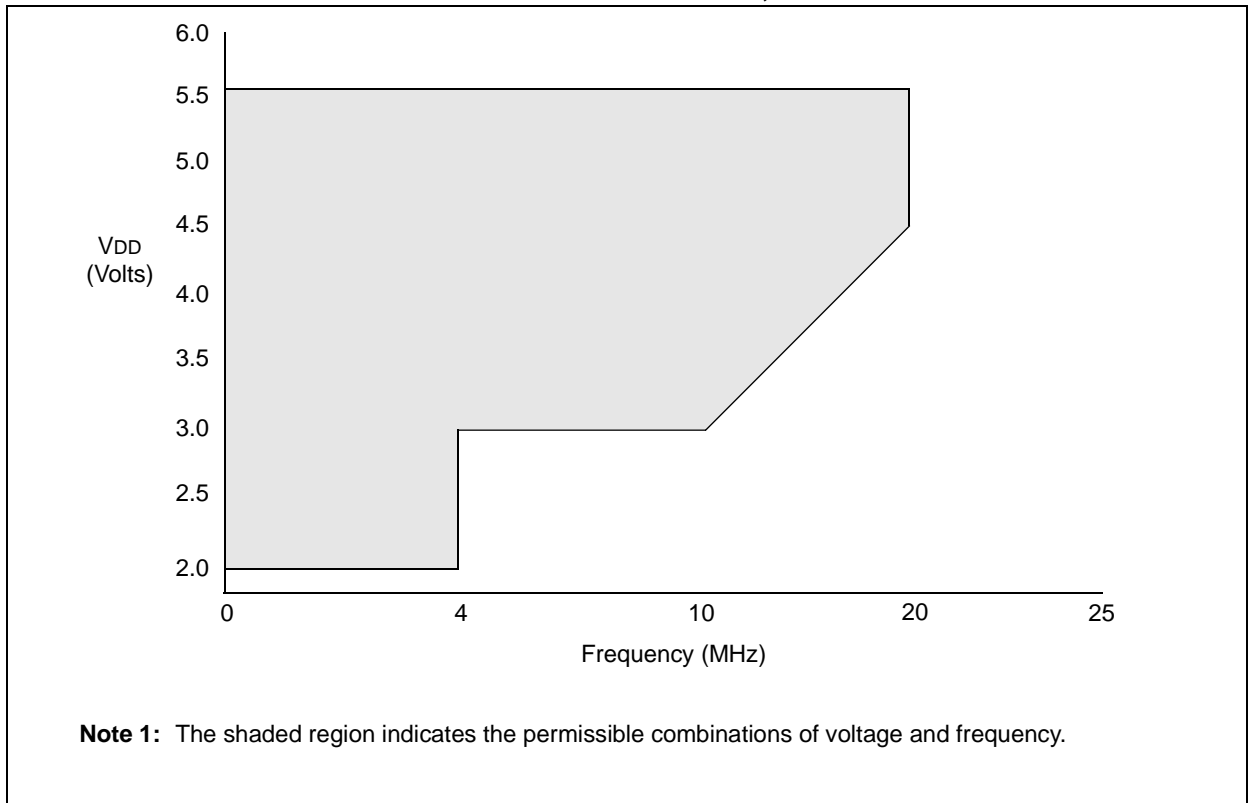
**FIGURE 17-1: PIC16F62X VOLTAGE-FREQUENCY GRAPH,  $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$**



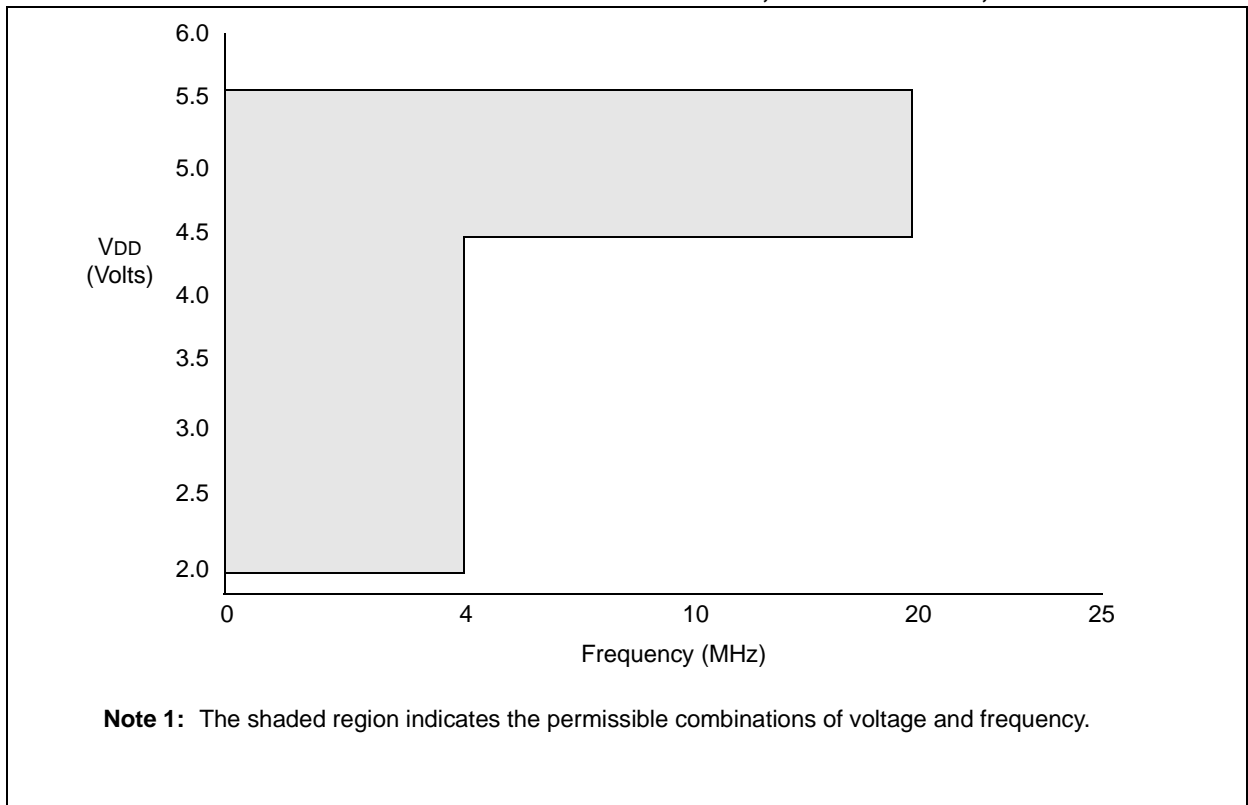
**FIGURE 17-2: PIC16F62X VOLTAGE-FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A < 0^{\circ}\text{C}$ ,  $+70^{\circ}\text{C} < T_A \leq 85^{\circ}\text{C}$**



**FIGURE 17-3: PIC16LF62X VOLTAGE-FREQUENCY GRAPH,  $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$**



**FIGURE 17-4: PIC16LF62X VOLTAGE-FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A < 0^{\circ}\text{C}$ ,  $+70^{\circ}\text{C} < T_A \leq 85^{\circ}\text{C}$**



# PIC16F62X

## 17.1 DC CHARACTERISTICS: PIC16F62X-04 (Commercial, Industrial, Extended) PIC16F62X-20 (Commercial, Industrial, Extended)

		Standard Operating Conditions (unless otherwise stated)					
		Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial and $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ for commercial and $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for extended					
Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
D001	VDD	Supply Voltage	3.0	-	5.5	V	
D002	VDR	RAM Data Retention Voltage (Note 1)	-	1.5*	-	V	Device in SLEEP mode
D003	VPOR	VDD start voltage to ensure Power-on Reset	-	Vss	-	V	See section on power-on reset for details
D004	SVDD	VDD rise rate to ensure Power-on Reset	0.05*	-	-	V/ms	See section on power-on reset for details
D005	VBOD	Brown-out Detect Voltage	3.7 3.7	4.0 4.0	4.3 4.4	V	BODEN configuration bit is cleared (Extended)
D010	IDD	Supply Current (Note 2, 5)	-	-	0.7	mA	FOSC = 4.0 MHz, VDD = 3.0
D013			-	4.0	7.0	mA	FOSC = 20.0 MHz, VDD = 5.5
			-	-	6.0	mA	FOSC = 20.0 MHz, VDD = 4.5
			-	-	2.0	mA	FOSC = 10.0 MHz, VDD = 3.0
D020	IPD	Power Down Current (Note 3)	-	-	2.2	$\mu\text{A}$	VDD = 3.0
			-	-	5.0	$\mu\text{A}$	VDD = 4.5
			-	-	9.0	$\mu\text{A}$	VDD = 5.5
			-	-	15.0	$\mu\text{A}$	VDD = 5.5 Extended
D023	$\Delta\text{IWDT}$	WDT Current (Note 4)	-	6.0	20	$\mu\text{A}$	VDD=4.0V (125°C)
	$\Delta\text{IBOD}$	Brown-out Detect Current (Note 4)	-	75	125	$\mu\text{A}$	BOD enabled, VDD = 5.0V
	$\Delta\text{ICOMP}$	Comparator Current for each Comparator (Note 4)	-	30	50	$\mu\text{A}$	VDD = 4.0V
	$\Delta\text{VREF}$	VREF Current (Note 4)	-	-	135	$\mu\text{A}$	VDD = 4.0V
1A	FOSC	LP Oscillator Operating Frequency	0	-	200	KHz	All temperatures
		INTRC Oscillator Operating Frequency	-	-	4	MHz	All temperatures
		XT Oscillator Operating Frequency	0	-	4	MHz	All temperatures
		HS Oscillator Operating Frequency	0	-	20	MHz	All temperatures

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

**2:** The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail to rail; all I/O pins tri-stated, pulled to VDD,

MCLR = VDD; WDT enabled/disabled as specified.

**3:** The power down current in SLEEP mode does not depend on the oscillator type. Power down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD or VSS.

**4:** The  $\Delta$  current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

**5:** For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula  $I_r = V_{DD}/2R_{ext}$  (mA) with Rext in k $\Omega$ .

## 17.2 DC CHARACTERISTICS: PIC16LF62X-04 (Commercial, Industrial, Extended)

Standard Operating Conditions (unless otherwise stated)							
Operating temperature $-40^{\circ}\text{C} \leq \text{TA} \leq +85^{\circ}\text{C}$ for industrial and $0^{\circ}\text{C} \leq \text{TA} \leq +70^{\circ}\text{C}$ for commercial and $-40^{\circ}\text{C} \leq \text{TA} \leq +125^{\circ}\text{C}$ for extended							
Operating voltage VDD range as described in DC spec Table 17.1 and Table 12-2							
Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
D001	VDD	Supply Voltage	2.0	-	5.5	V	
D002	VDR	RAM Data Retention Voltage (Note 1)	-	1.5*	-	V	Device in SLEEP mode
D003	VPOR	VDD start voltage to ensure Power-on Reset	-	VSS	-	V	See section on Power-on Reset for details
D004	SVDD	VDD rise rate to ensure Power-on Reset	0.05*	-	-	V/ms	See section on Power-on Reset for details
D005	VBOD	Brown-out Detect Voltage	3.7	4.0	4.3	V	BODEN configuration bit is cleared
D010	IDD	Supply Current (Note 2, 5)	-	-	0.6	mA	FOSC = 4.0 MHz, VDD = 2.5
-				4.0	7.0	mA	FOSC = 20.0 MHz, VDD = 5.5
-				-	6.0	mA	FOSC = 20.0 MHz, VDD = 4.5
D013			-	-	2.0	mA	FOSC = 10.0 MHz, VDD = 3.0
D020	IPD	Power Down Current (Note 2)	-	-	2.0	$\mu\text{A}$	VDD = 2.5
-				2.2	$\mu\text{A}$	VDD = 3.0	
-				5.0	$\mu\text{A}$	VDD = 4.5	
-				9.0	$\mu\text{A}$	VDD = 5.5	
-				15.0	$\mu\text{A}$	VDD = 5.5 Extended	
D023	$\Delta\text{IWDT}$	WDT Current (Note 4)	-	6.0	15	$\mu\text{A}$	VDD=3.0V
	$\Delta\text{IBOD}$	Brown-out Detect Current (Note 4)	-	75	125	$\mu\text{A}$	BOD enabled, VDD = 5.0V
	$\Delta\text{ICOMP}$	Comparator Current for each Comparator (Note 4)	-	30	50	$\mu\text{A}$	VDD = 3.0V
	$\Delta\text{IVREF}$	VREF Current (Note 4)	-	-	135	$\mu\text{A}$	VDD = 3.0V
1A	FOSC	LP Oscillator Operating Frequency	0	-	200	KHz	All temperatures
		INTRC Oscillator Operating Frequency	-	-	4	MHz	All temperatures
		XT Oscillator Operating Frequency	0	-	4	MHz	All temperatures
		HS Oscillator Operating Frequency	0	-	20	MHz	All temperatures

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

**2:** The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1=external square wave, from rail to rail; all I/O pins tristated, pulled to VDD,

MCLR = VDD; WDT enabled/disabled as specified.

**3:** The power down current in SLEEP mode does not depend on the oscillator type. Power down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD to VSS.

**4:** The  $\Delta$  current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

**5:** For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula  $I_r = \text{VDD}/2\text{Rext}$  (mA) with Rext in k $\Omega$ .

# PIC16F62X

## 17.3 DC CHARACTERISTICS: PIC16F62X (Commercial, Industrial, Extended) PIC16LF62X (Commercial, Industrial, Extended)

Standard Operating Conditions (unless otherwise stated)							
Operating temperature $-40^{\circ}\text{C} \leq \text{TA} \leq +85^{\circ}\text{C}$ for industrial and $0^{\circ}\text{C} \leq \text{TA} \leq +70^{\circ}\text{C}$ for commercial and $-40^{\circ}\text{C} \leq \text{TA} \leq +125^{\circ}\text{C}$ for extended							
Operating voltage $V_{DD}$ range as described in DC spec Table 17.1 and Table 12-2							
Param. No.	Sym	Characteristic	Min	Typ†	Max	Unit	Conditions
D030	$V_{IL}$	<b>Input Low Voltage</b> I/O ports with TTL buffer	$V_{SS}$	-	0.8V 0.15V <sub>DD</sub>	V	$V_{DD} = 4.5\text{V to } 5.5\text{V}$ otherwise
D031		with Schmitt Trigger input	$V_{SS}$	-	0.2V <sub>DD</sub>	V	Note1
D032		MCLR, RA4/T0CKI, OSC1 (in ER mode)	$V_{SS}$	-	0.2V <sub>DD</sub>	V	Note1
D033		OSC1 (in XT and HS)	$V_{SS}$	-	0.3V <sub>DD</sub>	V	
		OSC1 (in LP)	$V_{SS}$	-	0.6V <sub>DD</sub> -1.0	V	
D040	$V_{IH}$	<b>Input High Voltage</b> I/O ports with TTL buffer	2.0V .25V <sub>DD</sub> + 0.8V	-	V <sub>DD</sub> V <sub>DD</sub>	V	$V_{DD} = 4.5\text{V to } 5.5\text{V}$ otherwise
D041		with Schmitt Trigger input	0.8V <sub>DD</sub>	-	V <sub>DD</sub>	V	
D042		MCLR RA4/T0CKI	0.8V <sub>DD</sub>	-	V <sub>DD</sub>	V	
D043		OSC1 (XT, HS and LP)	0.7V <sub>DD</sub>	-	V <sub>DD</sub>	V	
D043A		OSC1 (in ER mode)	0.9V <sub>DD</sub>	-		V	Note1
D070	IPURB	PORTB weak pull-up current	50	200	400	μA	$V_{DD} = 5.0\text{V}$ , $V_{PIN} = V_{SS}$
D060	$I_{IL}$	<b>Input Leakage Current</b> (Notes 2, 3) I/O ports (Except PORTA)	-	-	±1.0	μA	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , pin at hi-impedance
D061		PORTA	-	-	±0.5	μA	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , pin at hi-impedance
D063		RA4/T0CKI	-	-	±1.0	μA	$V_{SS} \leq V_{PIN} \leq V_{DD}$
		OSC1, MCLR	-	-	±5.0	μA	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , XT, HS and LP osc configuration
D080	$V_{OL}$	<b>Output Low Voltage</b> I/O ports	-	-	0.6	V	$I_{OL}=8.5\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $-40^{\circ}$ to $+85^{\circ}\text{C}$
D083		OSC2/CLKOUT (ER only)	-	-	0.6	V	$I_{OL}=7.0\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $+125^{\circ}\text{C}$
			-	-	0.6	V	$I_{OL}=1.6\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $-40^{\circ}$ to $+85^{\circ}\text{C}$
			-	-	0.6	V	$I_{OL}=1.2\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $+125^{\circ}\text{C}$
D090	$V_{OH}$	<b>Output High Voltage</b> (Note 3) I/O ports (Except RA4)	$V_{DD}-0.7$	-	-	V	$I_{OH}=-3.0\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $-40^{\circ}$ to $+85^{\circ}\text{C}$
D092		OSC2/CLKOUT (ER only)	$V_{DD}-0.7$	-	-	V	$I_{OH}=-2.5\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $+125^{\circ}\text{C}$
			$V_{DD}-0.7$	-	-	V	$I_{OH}=-1.3\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $-40^{\circ}$ to $+85^{\circ}\text{C}$
			$V_{DD}-0.7$	-	-	V	$I_{OH}=-1.0\text{ mA}$ , $V_{DD}=4.5\text{V}$ , $+125^{\circ}\text{C}$
*D150	$V_{OD}$	<b>Open-Drain High Voltage</b>		-	8.5*	V	RA4 pin PIC16F62X, PIC16LF62X
D100	COSC2	<b>Capacitive Loading Specs on Output Pins</b> OSC2 pin		-	15	pF	In XT, HS and LP modes when external clock used to drive OSC1.
D101	Cio	All I/O pins/OSC2 (in ER mode)		-	50	pF	

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** In ER oscillator configuration, the OSC1 pin is a Schmitt Trigger input. It is not recommended that the PIC16F62X be driven with external clock in ER mode.
- 2:** The leakage current on the MCLR pin is strongly dependent on 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 coming out of the pin.



**TABLE 17-1: COMPARATOR SPECIFICATIONS**

Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +125°C, unless otherwise stated.							
Param No.	Characteristics	Sym	Min	Typ	Max	Units	Comments
D300	Input offset voltage	VIOFF	-	± 5.0	± 10	mV	
D301	Input common mode voltage*	VICM	0	-	VDD - 1.5	V	
D302	Common Mode Rejection Ratio*	CMRR	55	-	-	db	
300 300A	Response Time <sup>(1)</sup> *	TRESP	-	150	400 600	ns ns	16F62X 16LF62X
301	Comparator Mode Change to Output Valid*	TMC2OV	-	-	10	µs	

\* These parameters are characterized but not tested.

Response time measured with one comparator input at (VDD - 1.5)/2 while the other input transitions from VSS to VDD.

**TABLE 17-2: VOLTAGE REFERENCE SPECIFICATIONS**

Operating Conditions: 3.0V < VDD < 5.5V, -40°C < TA < +125°C, unless otherwise stated.							
Spec No.	Characteristics	Sym	Min	Typ	Max	Units	Comments
D310	Resolution	VRES	VDD/24	-	VDD/32	LSb	
D311	Absolute Accuracy	VRAA	-	-	1/4 1/2	LSb LSb	Low Range (VRR = 1) High Range (VRR = 0)
D312	Unit Resistor Value (R)*	VRUR	-	2k	-	Ω	
310	Settling Time <sup>(1)</sup> *	TSET	-	-	10	µs	

\* These parameters are characterized but not tested.

Settling time measured while VRR = 1 and VR<3:0> transitions from 0000 to 1111.

# PIC16F62X

## 17.4 Timing Parameter Symbology

The timing parameter symbols have been created with one of the following formats:

1. TppS2ppS
2. TppS

<b>T</b>			
F	Frequency	T	Time

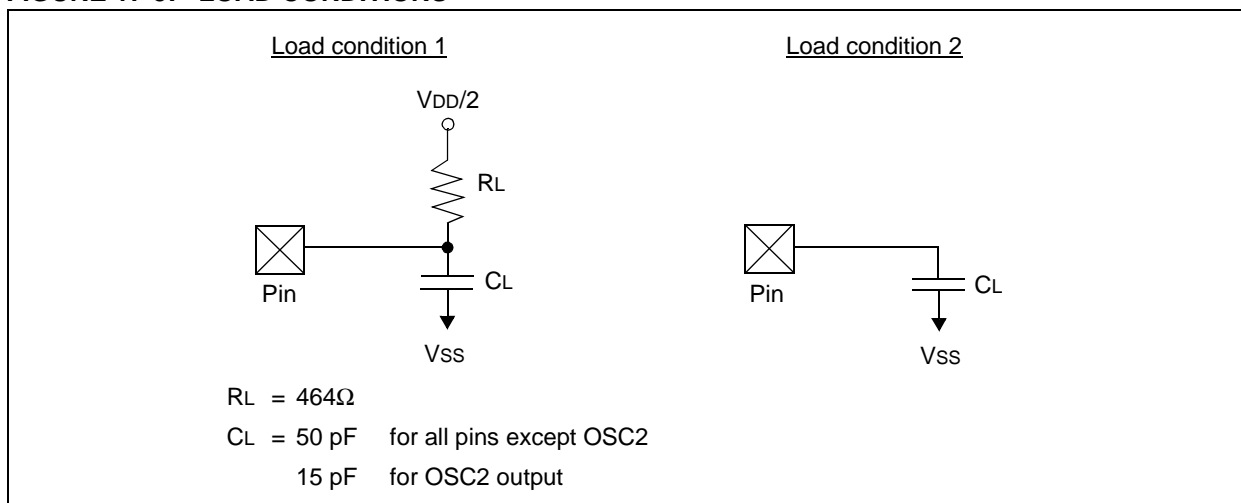
Lowercase subscripts (pp) and their meanings:

<b>pp</b>			
ck	CLKOUT	osc	OSC1
io	I/O port	t0	T0CKI
mc	MCLR		

Uppercase letters and their meanings:

<b>S</b>			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-Impedance

**FIGURE 17-5: LOAD CONDITIONS**



**TABLE 17-3: DC CHARACTERISTICS: PIC16F62X, PIC16LF62X**

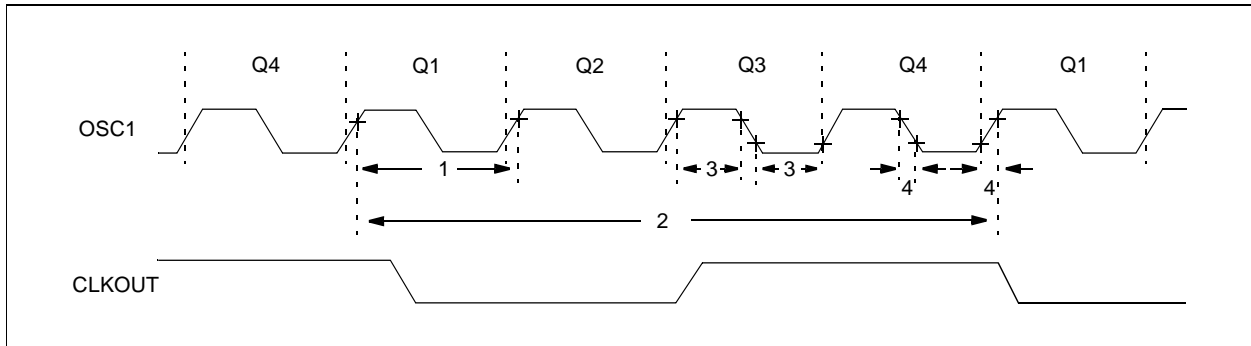
DC Characteristics		Standard Operating Conditions (unless otherwise stated)					
Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
<b>Data EEPROM Memory</b>							
D120	Ed	Endurance	1M*	10M	—	E/W	25°C at 5V V <sub>MIN</sub> = Minimum operating voltage
D121	Vdrw	V <sub>DD</sub> for read/write	V <sub>MIN</sub>	—	5.5	V	
D122	Tdew	Erase/Write cycle time	—	4	8*	ms	
<b>Program Flash Memory</b>							
D130	Ep	Endurance	1000*	10000	—	E/W	V <sub>MIN</sub> = Minimum operating voltage
D131	Vpr	V <sub>DD</sub> for read	V <sub>min</sub>	—	5.5	V	
D132	Vpew	V <sub>DD</sub> for erase/write	4.5	—	5.5	V	
D133	Tpew	Erase/Write cycle time	—	4	8*	ms	

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

## 17.5 Timing Diagrams and Specifications

**FIGURE 17-6: EXTERNAL CLOCK TIMING**

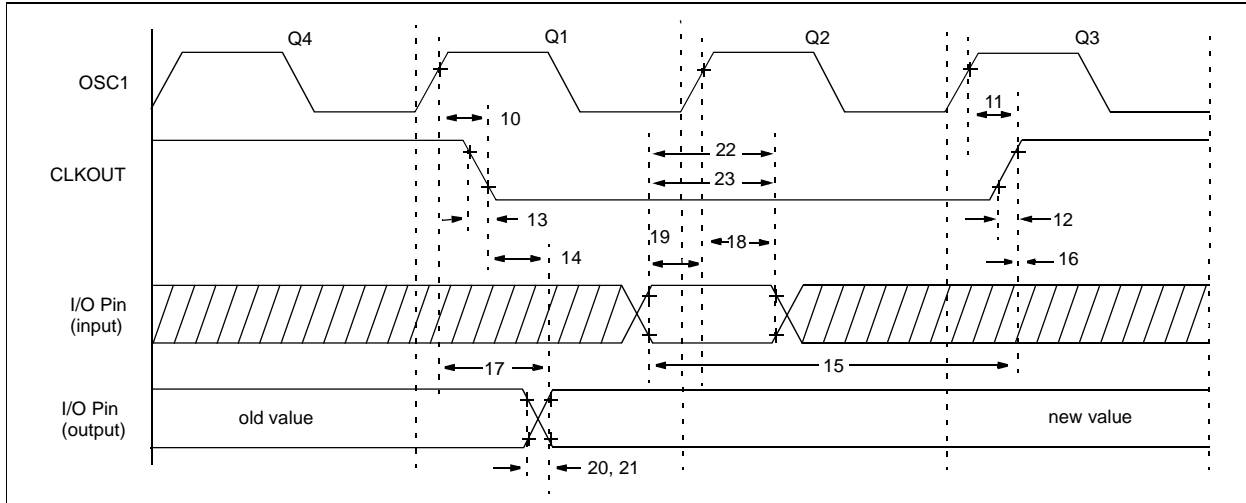


**TABLE 17-4: EXTERNAL CLOCK TIMING REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	External CLKIN Frequency (Note 1)	DC	—	4	MHz	XT and ER osc mode, VDD=5.0V
			DC	—	20	MHz	HS osc mode
DC	—		200	kHz	LP osc mode		
	Oscillator Frequency (Note 1)	Oscillator Frequency (Note 1)	0.1	—	4	MHz	ER osc mode, VDD=5.0V
			1	—	4	MHz	XT osc mode
			—	—	20	MHz	HS osc mode
			—	—	200	kHz	LP osc mode
			4	—	—	MHz	INTRC mode (fast)
			37	—	—	kHz	INTRC mode (slow)
1	Tosc	External CLKIN Period (Note 1)	250	—	—	ns	XT and ER osc mode
			50	—	—	ns	HS osc mode
5	—		—	μs	LP osc mode		
	Oscillator Period (Note 1)	Oscillator Period (Note 1)	250	—	—	ns	ER osc mode
			250	—	10,000	ns	XT osc mode
			50	—	1,000	ns	HS osc mode
			5	—	—	μs	LP osc mode
			250	—	—	ns	INTRC mode (fast)
			27	—	—	μs	INTRC mode (slow)
2	Tcy	Instruction Cycle Time (Note 1)	1.0	T <sub>CY</sub>	DC	ns	T <sub>CY</sub> = 4/Fosc
3	TosL, TosH	External CLKIN (OSC1) High External CLKIN Low	100 *	—	—	ns	XT oscillator, T <sub>osc</sub> L/H duty cycle
4	INTRC	Internal Calibrated ER	3.65	4.00	4.28	MHz	VDD = 5.0V
5	ER	External Biased ER Frequency	10kHz	—	8MHz	—	VDD = 5.0V

# PIC16F62X

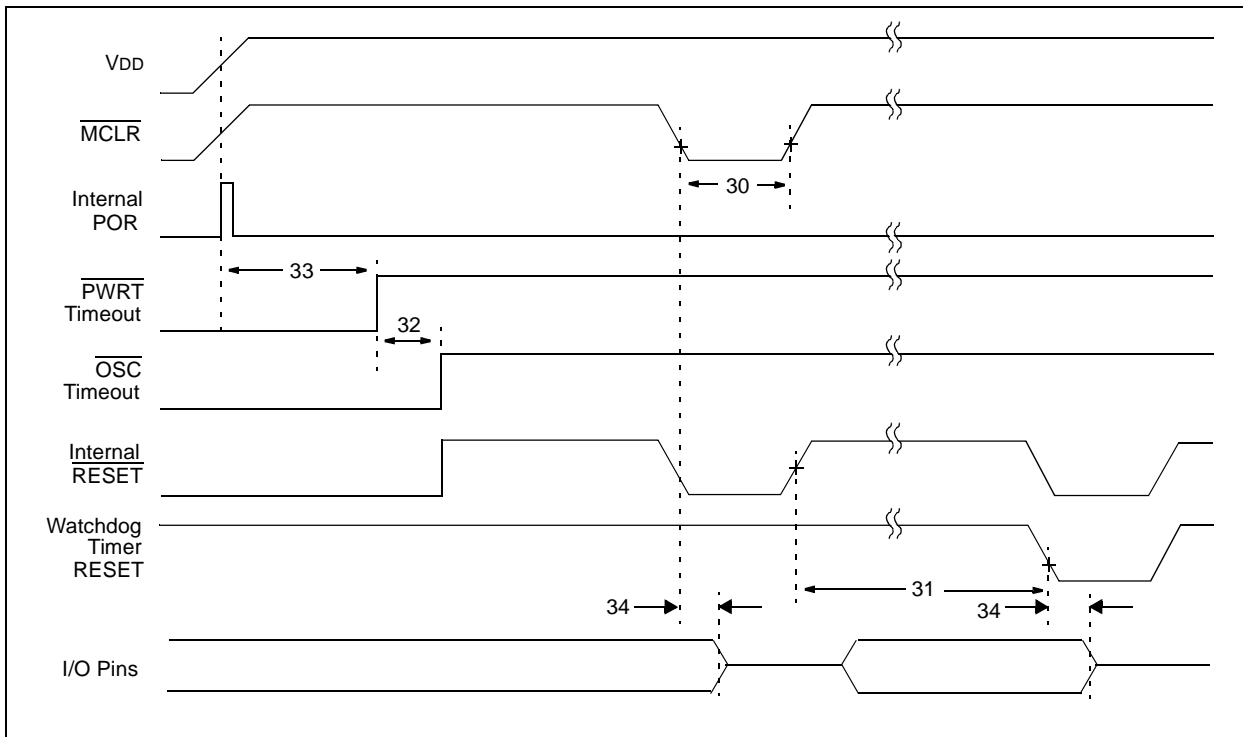
**FIGURE 17-7: CLKOUT AND I/O TIMING**



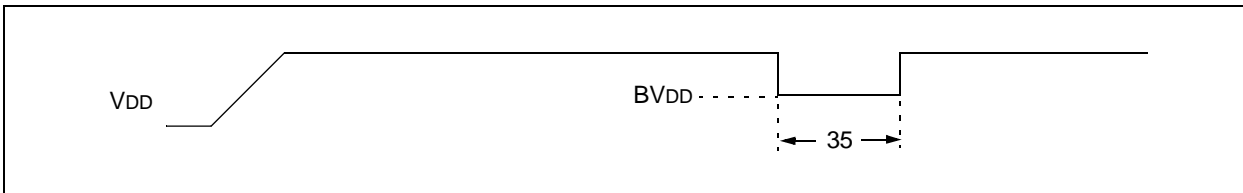
**TABLE 17-5: CLKOUT AND I/O TIMING REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	
10	TosH2ckL	OSC1↑ to CLKOUT↓	—	75	200	ns	
10A			—	—	400	ns	
11	TosH2ckH	OSC1↑ to CLKOUT↑	—	75	200	ns	
11A			—	—	400	ns	
12	TckR	CLKOUT rise time	—	35	100	ns	
12A			—	—	200	ns	
13	TckF	CLKOUT fall time	—	35	100	ns	
13A			—	—	200	ns	
14	TckL2ioV	CLKOUT ↓ to Port out valid	—	—	20	ns	
15	TioV2ckH	Port in valid before CLKOUT ↑	16F62X	Tosc +200 ns	—	—	ns
			16LF62X	Tosc =400 ns	—	—	ns
16	TckH2iol	Port in hold after CLKOUT ↑	0	—	—	ns	
17	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid	16F62X	—	50	150 *	ns
			16LF62X	—	—	300	ns
18	TosH2iol	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	100 200	—	—	ns	

**FIGURE 17-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**FIGURE 17-9: BROWN-OUT DETECT TIMING**

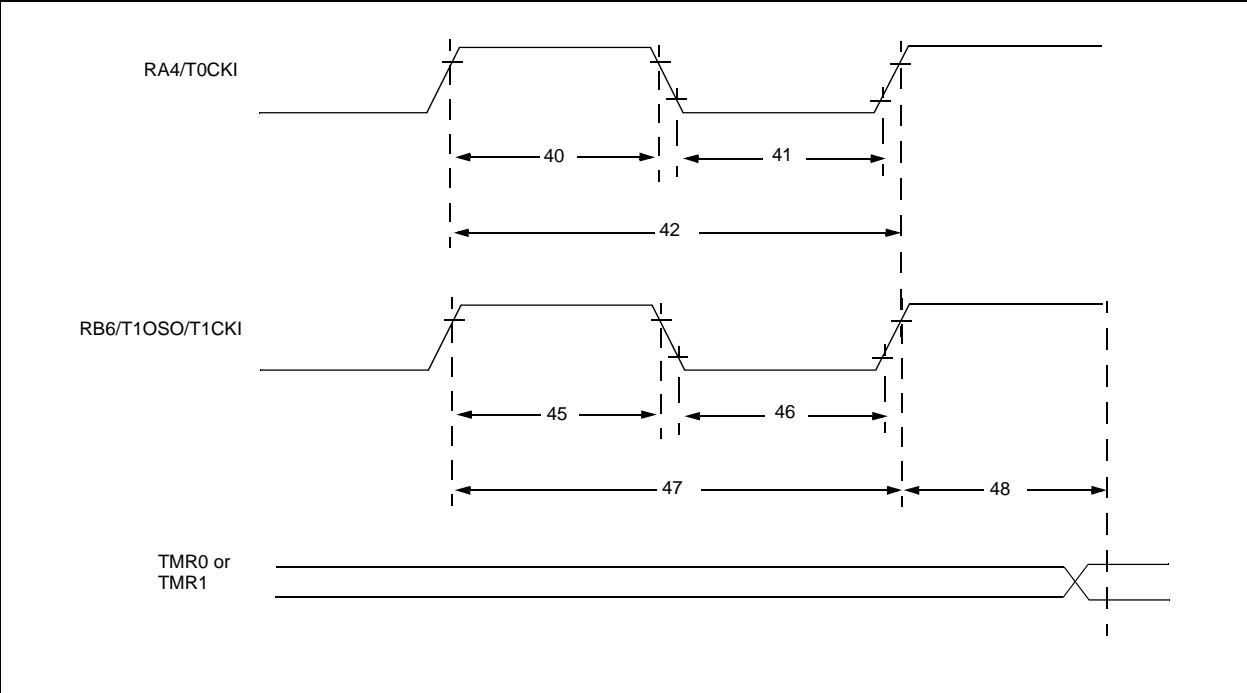


**TABLE 17-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Unit s	Conditions
30	Tmcl	MCLR Pulse Width (low)	2000 TBD	— TBD	— TBD	ns ms	VDD = 5V, -40°C to +85°C Extended temperature
31	Twdt	Watchdog Timer Time-out Period (No Prescaler)	7 TBD	18 TBD	33 TBD	ms ms	VDD = 5V, -40°C to +85°C Extended temperature
32	Tost	Oscillation Start-up Timer Period	—	1024Tosc	—	—	Tosc = OSC1 period
33*	Tpwrt	Power up Timer Period	28 TBD	72 TBD	132 TBD	ms ms	VDD = 5V, -40°C to +85°C
34	Tioz	I/O Hi-impedance from MCLR Low or Watchdog Timer Reset	—	—	2.0	μs	
35	TBOD	Brown-out Detect pulse width	100	—	—	μs	VDD ≤ BVDD (D005)

# PIC16F62X

FIGURE 17-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



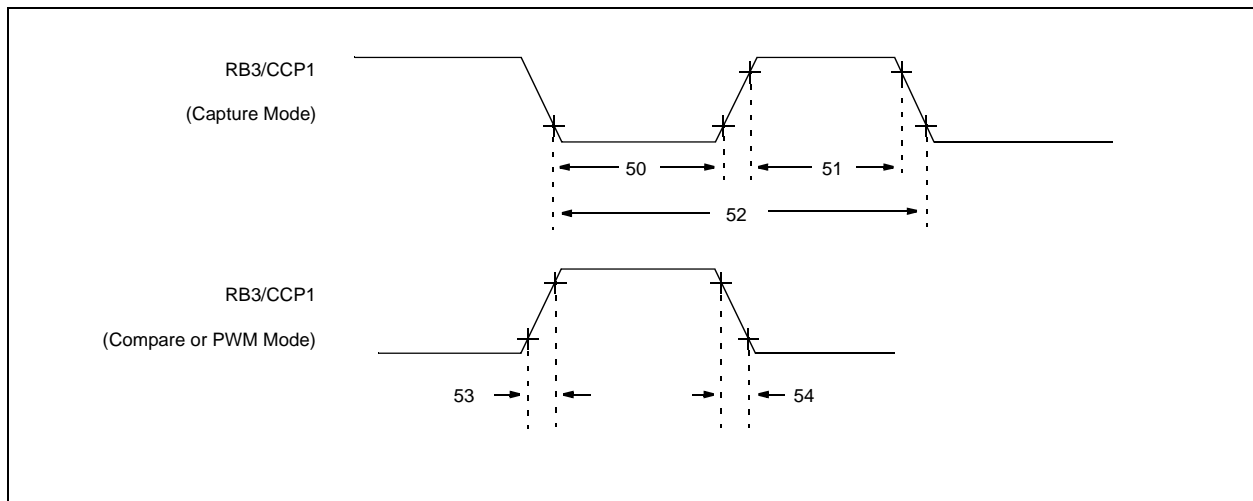
**TABLE 17-7: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions	
40*	Tt0H	T0CKI High Pulse Width	No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			With Prescaler	10	—	—	ns		
41*	Tt0L	T0CKI Low Pulse Width	No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			With Prescaler	10	—	—	ns		
42*	Tt0P	T0CKI Period		Greater of: $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (2, 4, ..., 256)	
45*	Tt1H	T1CKI High Time	Synchronous, No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			Synchronous, 16F62X with Prescaler	15	—	—	ns		
			16LF62X	25	—	—	ns		
			Asynchronous 16F62X	30	—	—	ns		
		16LF62X	50	—	—	ns			
46*	Tt1L	T1CKI Low Time	Synchronous, No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			Synchronous, 16F62X with Prescaler	15	—	—	ns		
			16LF62X	25	—	—	ns		
			Asynchronous 16F62X	30	—	—	ns		
		16LF62X	50	—	—	ns			
47*	Tt1P	T1CKI input period	Synchronous	16F62X	Greater of: $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (1, 2, 4, 8)
				16LF62X	Greater of: $\frac{T_{CY} + 40}{N}$	—	—	—	
			Asynchronous	16F62X	60	—	—	ns	
				16LF62X	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		$2T_{osc}$	—	$7T_{osc}$	—		

\* 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 17-11: CAPTURE/COMPARE/PWM TIMINGS**



# PIC16F62X

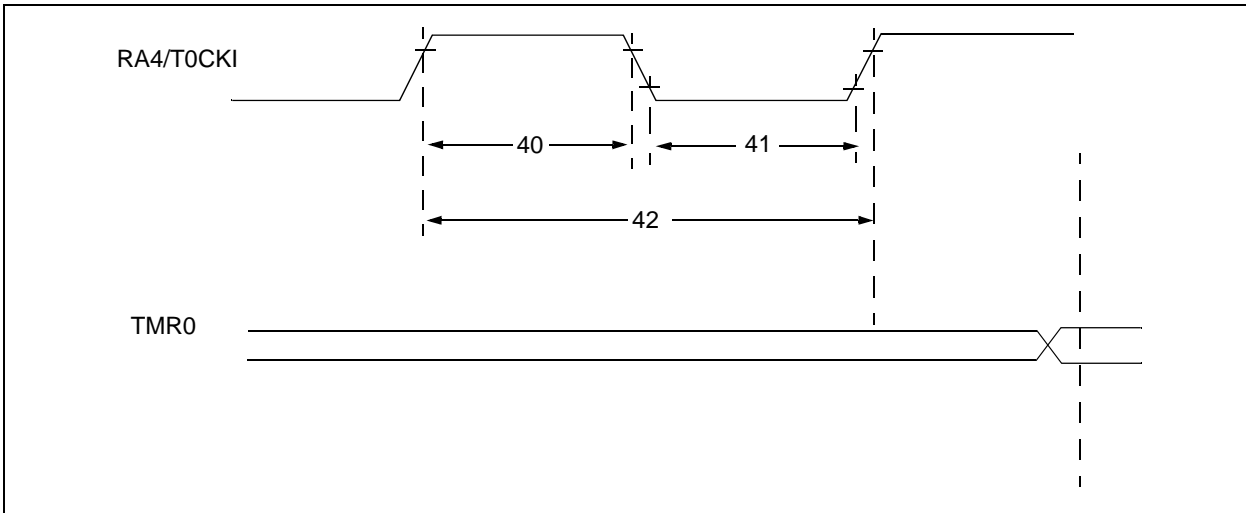
**TABLE 17-8: CAPTURE/COMPARE/PWM REQUIREMENTS**

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions	
50*	TccL	CCP input low time	No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			With Prescaler	16F62X	10	—	—		ns
				16LF62X	20	—	—		ns
51*	TccH	CCP input high time	No Prescaler	$0.5T_{CY} + 20$	—	—	ns		
			With Prescaler	16F62X	10	—	—		ns
				16LF62X	20	—	—		ns
52*	TccP	CCP input period		$\frac{3T_{CY} + 40}{N}$	—	—	ns	N = prescale value (1, 4 or 16)	
53*	TccR	CCP output rise time	16F62X		10	25	ns		
			16LF62X		25	45	ns		
54*	TccF	CCP output fall time	16F62X		10	25	ns		
			16LF62X		25	45	ns		

\* 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 17-12: TIMER0 CLOCK TIMING**



**TABLE 17-9: TIMER0 CLOCK REQUIREMENTS**

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	$0.5 T_{CY} + 20^*$	—	—	ns	
			With Prescaler	10*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	$0.5 T_{CY} + 20^*$	—	—	ns	
			With Prescaler	10*	—	—	ns	
42	Tt0P	T0CKI Period		$\frac{T_{CY} + 40^*}{N}$	—	—	ns	N = prescale value (1, 2, 4, ..., 256)

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.



## 18.0 DEVICE CHARACTERIZATION INFORMATION

Not Available at this time.

# PIC16F62X

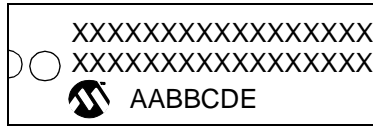
---

NOTES:

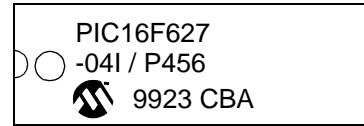
## 19.0 PACKAGING INFORMATION

### 19.1 Package Marking Information

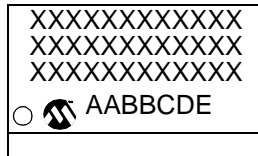
#### 18-Lead PDIP



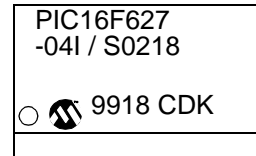
#### Example



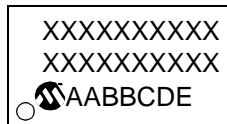
#### 18-Lead SOIC (.300")



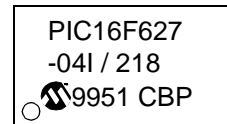
#### Example



#### 20-Lead SSOP



#### Example

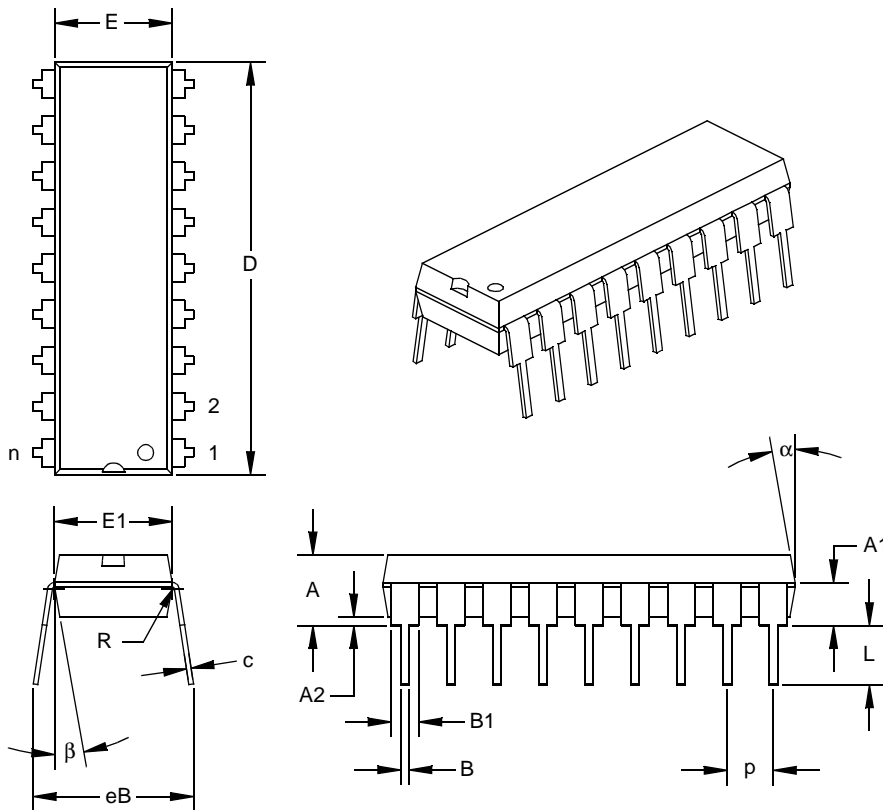


<b>Legend:</b>	MM...M	Microchip part number information
	XX...X	Customer specific information*
	AA	Year code (last 2 digits of calendar year)
	BB	Week code (week of January 1 is week '01')
	C	Facility code of the plant at which wafer is manufactured
		O = Outside Vendor
		C = 5" Line
		S = 6" Line
		H = 8" Line
	D	Mask revision number
	E	Assembly code of the plant or country of origin in which part was assembled
<b>Note:</b>	In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.	

\* Standard OTP marking consists of Microchip part number, year code, week code, facility code, mask rev#, and assembly code. For OTP marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

# PIC16F62X

Package Type: K04-007 18-Lead Plastic Dual In-line (P) – 300 mil



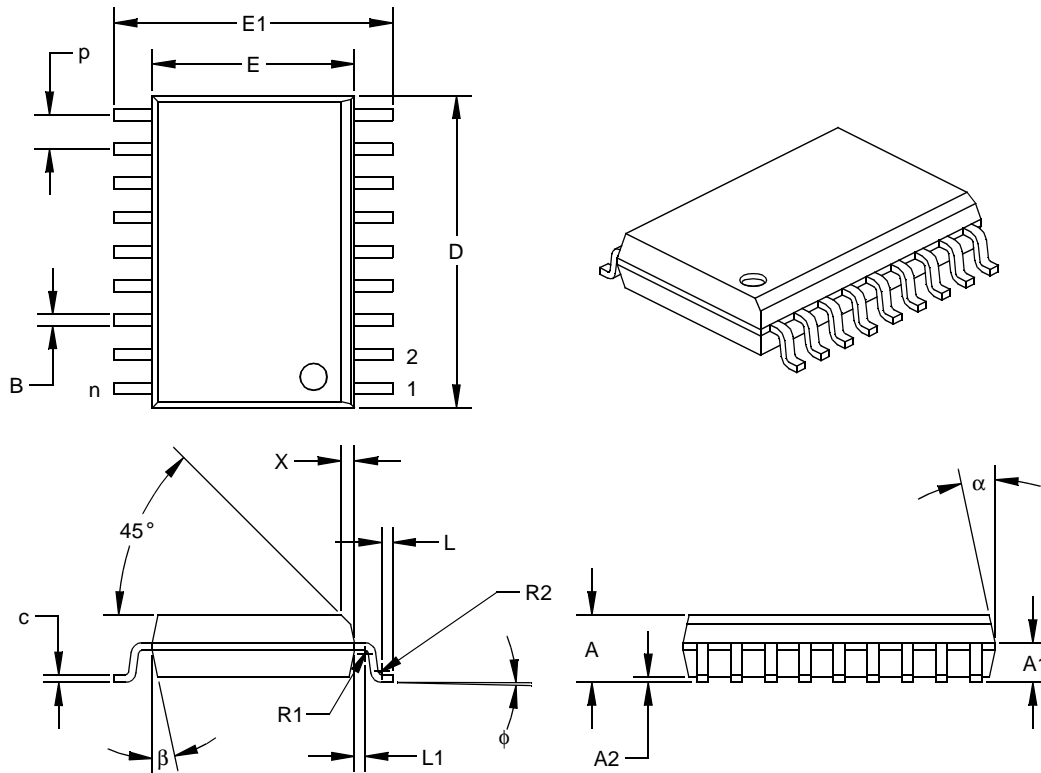
Units		INCHES*			MILLIMETERS		
		MIN	NOM	MAX	MIN	NOM	MAX
Dimension Limits							
PCB Row Spacing			0.300			7.62	
Number of Pins	n		18			18	
Pitch	p		0.100			2.54	
Lower Lead Width	B	0.013	0.018	0.023	0.33	0.46	0.58
Upper Lead Width	B1 <sup>†</sup>	0.055	0.060	0.065	1.40	1.52	1.65
Shoulder Radius	R	0.000	0.005	0.010	0.00	0.13	0.25
Lead Thickness	c	0.005	0.010	0.015	0.13	0.25	0.38
Top to Seating Plane	A	0.110	0.155	0.155	2.79	3.94	3.94
Top of Lead to Seating Plane	A1	0.075	0.095	0.115	1.91	2.41	2.92
Base to Seating Plane	A2	0.000	0.020	0.020	0.00	0.51	0.51
Tip to Seating Plane	L	0.125	0.130	0.135	3.18	3.30	3.43
Package Length	D <sup>‡</sup>	0.890	0.895	0.900	22.61	22.73	22.86
Molded Package Width	E <sup>‡</sup>	0.245	0.255	0.265	6.22	6.48	6.73
Radius to Radius Width	E1	0.230	0.250	0.270	5.84	6.35	6.86
Overall Row Spacing	eB	0.310	0.349	0.387	7.87	8.85	9.83
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

\* Controlling Parameter.

<sup>†</sup> Dimension "B1" does not include dam-bar protrusions. Dam-bar protrusions shall not exceed 0.003" (0.076 mm) per side or 0.006" (0.152 mm) more than dimension "B1."

<sup>‡</sup> Dimensions "D" and "E" do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010" (0.254 mm) per side or 0.020" (0.508 mm) more than dimensions "D" or "E."

Package Type: K04-051 18-Lead Plastic Small Outline (SO) – Wide, 300 mil



Units		INCHES*			MILLIMETERS		
		MIN	NOM	MAX	MIN	NOM	MAX
Dimension Limits							
Pitch	p		0.050			1.27	
Number of Pins	n		18			18	
Overall Pack. Height	A	0.093	0.099	0.104	2.36	2.50	2.64
Shoulder Height	A1	0.048	0.058	0.068	1.22	1.47	1.73
Standoff	A2	0.004	0.008	0.011	0.10	0.19	0.28
Molded Package Length	D <sup>‡</sup>	0.450	0.456	0.462	11.43	11.58	11.73
Molded Package Width	E <sup>‡</sup>	0.292	0.296	0.299	7.42	7.51	7.59
Outside Dimension	E1	0.394	0.407	0.419	10.01	10.33	10.64
Chamfer Distance	X	0.010	0.020	0.029	0.25	0.50	0.74
Shoulder Radius	R1	0.005	0.005	0.010	0.13	0.13	0.25
Gull Wing Radius	R2	0.005	0.005	0.010	0.13	0.13	0.25
Foot Length	L	0.011	0.016	0.021	0.28	0.41	0.53
Foot Angle	φ	0	4	8	0	4	8
Radius Centerline	L1	0.010	0.015	0.020	0.25	0.38	0.51
Lead Thickness	c	0.009	0.011	0.012	0.23	0.27	0.30
Lower Lead Width	B <sup>†</sup>	0.014	0.017	0.019	0.36	0.42	0.48
Mold Draft Angle Top	α	0	12	15	0	12	15
Mold Draft Angle Bottom	β	0	12	15	0	12	15

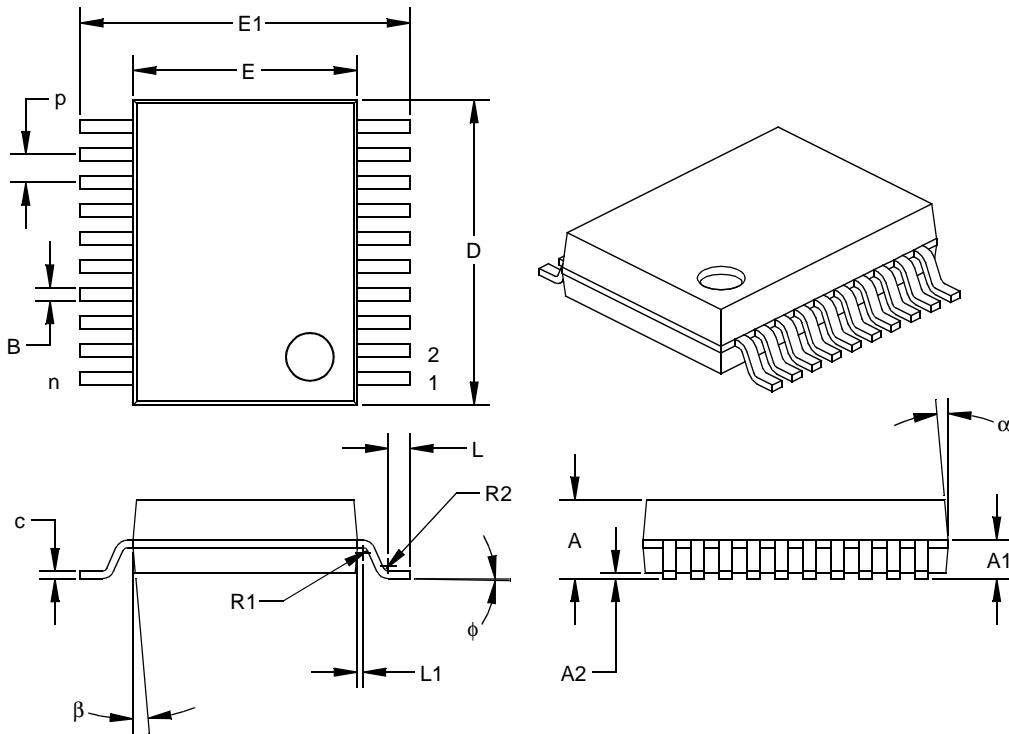
\* Controlling Parameter.

† Dimension "B" does not include dam-bar protrusions. Dam-bar protrusions shall not exceed 0.003" (0.076 mm) per side or 0.006" (0.152 mm) more than dimension "B."

‡ Dimensions "D" and "E" do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010" (0.254 mm) per side or 0.020" (0.508 mm) more than dimensions "D" or "E."

# PIC16F62X

Package Type: K04-072 20-Lead Plastic Shrink Small Outline (SS) – 5.30 mm



Units		INCHES			MILLIMETERS*		
		MIN	NOM	MAX	MIN	NOM	MAX
Dimension Limits							
Pitch	p		0.026			0.65	
Number of Pins	n		20			20	
Overall Pack. Height	A	0.068	0.073	0.078	1.73	1.86	1.99
Shoulder Height	A1	0.026	0.036	0.046	0.66	0.91	1.17
Standoff	A2	0.002	0.005	0.008	0.05	0.13	0.21
Molded Package Length	D <sup>†</sup>	0.278	0.283	0.289	7.07	7.20	7.33
Molded Package Width	E <sup>‡</sup>	0.205	0.208	0.212	5.20	5.29	5.38
Outside Dimension	E1	0.301	0.306	0.311	7.65	7.78	7.90
Shoulder Radius	R1	0.005	0.005	0.010	0.13	0.13	0.25
Gull Wing Radius	R2	0.005	0.005	0.010	0.13	0.13	0.25
Foot Length	L	0.015	0.020	0.025	0.38	0.51	0.64
Foot Angle	φ	0	4	8	0	4	8
Radius Centerline	L1	0.000	0.005	0.010	0.00	0.13	0.25
Lead Thickness	c	0.005	0.007	0.009	0.13	0.18	0.22
Lower Lead Width	B <sup>†</sup>	0.010	0.012	0.015	0.25	0.32	0.38
Mold Draft Angle Top	α	0	5	10	0	5	10
Mold Draft Angle Bottom	β	0	5	10	0	5	10

\* Controlling Parameter.

† Dimension "B" does not include dam-bar protrusions. Dam-bar protrusions shall not exceed 0.003" (0.076 mm) per side or 0.006" (0.152 mm) more than dimension "B."

‡ Dimensions "D" and "E" do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010" (0.254 mm) per side or 0.020" (0.508 mm) more than dimensions "D" or "E."

## INDEX

### A

A/D	
Special Event Trigger (CCP)	65
Absolute Maximum Ratings	131
ADDLW Instruction	115
ADDWF Instruction	115
ANDLW Instruction	115
ANDWF Instruction	115
Architectural Overview	9
Assembler	
MPASM Assembler	125

### B

Baud Rate Error	73
Baud Rate Formula	73
Baud Rates	
Asynchronous Mode	74
Synchronous Mode	74
BCF Instruction	116
Block Diagram	
TIMER0	45
TMR0/WDT PRESCALER	48
Block Diagrams	
Comparator I/O Operating Modes	58
Comparator Output	60
RA3:RA0 and RA5 Port Pins	35
Timer1	51
Timer2	54
USART Receive	80
USART Transmit	78
BRGH bit	73
Brown-Out Detect (BOD)	101
BSF Instruction	116
BTFSC Instruction	116
BTFSS Instruction	117

### C

CALL Instruction	117
Capture (CCP Module)	64
Block Diagram	64
CCP Pin Configuration	64
CCPR1H:CCPR1L Registers	64
Changing Between Capture Prescalers	64
Software Interrupt	64
Timer1 Mode Selection	64
Capture/Compare/PWM (CCP)	63
CCP1	63
CCP1CON Register	63
CCPR1H Register	63
CCPR1L Register	63
CCP2	63
Timer Resources	63
CCP1CON Register	63
CCP1M3:CCP1M0 Bits	63
CCP1X:CCP1Y Bits	63
CCP2CON Register	
CCP2M3:CCP2M0 Bits	63
CCP2X:CCP2Y Bits	63
Clocking Scheme/Instruction Cycle	12
CLRF Instruction	117
CLRWF Instruction	117
CLRWDW Instruction	118
CMCON Register	57
Code Protection	112
COMF Instruction	118
Comparator Configuration	58

Comparator Interrupts	61
Comparator Module	57
Comparator Operation	59
Comparator Reference	59
Compare (CCP Module)	65
Block Diagram	65
CCP Pin Configuration	65
CCPR1H:CCPR1L Registers	65
Software Interrupt	65
Special Event Trigger	65
Timer1 Mode Selection	65
Configuration Bits	96
Configuring the Voltage Reference	69
Crystal Operation	97

### D

DATA	93
Data	93
Data EEPROM Memory	91
EECON1 Register	91
EECON2 Register	91
Data Memory Organization	13
DECf Instruction	118
DECFSZ Instruction	118
Development Support	125

### E

EECON1	92
Errata	3
External Crystal Oscillator Circuit	98

### G

General purpose Register File	13
GOTO Instruction	119

### I

I/O Ports	27
I/O Programming Considerations	44
ID Locations	112
INCF Instruction	119
INCFSZ Instruction	119
In-Circuit Serial Programming	112
Indirect Addressing, INDF and FSR Registers	26
Instruction Flow/Pipelining	12
Instruction Set	
ADDLW	115
ADDWF	115
ANDLW	115
ANDWF	115
BCF	116
BSF	116
BTFSC	116
BTFSS	117
CALL	117
CLRF	117
CLRWF	117
CLRWDW	118
COMF	118
DECf	118
DECFSZ	118
GOTO	119
INCF	119
INCFSZ	119
IORLW	119
IORWF	120
MOVF	120
MOVLW	120
MOVWF	120

# PIC16F62X

NOP .....	121	Power-Down Mode (SLEEP) .....	111
OPTION .....	121	Power-On Reset (POR) .....	101
RETFIE .....	121	Power-up Timer (PWRT) .....	101
RETLW .....	121	PR2 Register .....	54
RETURN .....	122	Prescaler .....	48
RLF .....	122	Prescaler, Capture .....	64
RRF .....	122	Prescaler, Timer2 .....	66
SLEEP .....	122	PRO MATE® II Universal Programmer .....	127
SUBLW .....	123	Program Memory Organization .....	13
SUBWF .....	123	PROTECTION .....	93
SWAPF .....	124	PWM (CCP Module) .....	66
TRIS .....	124	Block Diagram .....	66
XORLW .....	124	CCPR1H:CCPR1L Registers .....	66
XORWF .....	124	Duty Cycle .....	66
Instruction Set Summary .....	113	Example Frequencies/Resolutions .....	67
INT Interrupt .....	108	Output Diagram .....	66
INTCON Register .....	21	Period .....	66
Interrupt Sources .....		Set-Up for PWM Operation .....	67
Capture Complete (CCP) .....	64	TMR2 to PR2 Match .....	66
Compare Complete (CCP) .....	65	<b>Q</b>	
TMR2 to PR2 Match (PWM) .....	66	Q-Clock .....	66
Interrupts .....	107	Quick-Turnaround-Production (QTP) Devices .....	7
Interrupts, Enable Bits .....		<b>R</b>	
CCP1 Enable (CCP1IE Bit) .....	64	RC Oscillator .....	98
Interrupts, Flag Bits .....		READING .....	93
CCP1 Flag (CCP1IF Bit) .....	64, 65	Registers .....	
IORLW Instruction .....	119	Maps .....	
IORWF Instruction .....	120	PIC16C76 .....	14
<b>K</b>		PIC16C77 .....	14
KeeLoq® Evaluation and Programming Tools .....	128	RCSTA .....	
<b>M</b>		Diagram .....	72
Memory Organization .....		Reset .....	99
Data EEPROM Memory .....	91	RETFIE Instruction .....	121
MOVF Instruction .....	120	RETLW Instruction .....	121
MOVLW Instruction .....	120	RETURN Instruction .....	122
MOVWF Instruction .....	120	RLF Instruction .....	122
MPLAB Integrated Development Environment Software ..	125	RRF Instruction .....	122
<b>N</b>		<b>S</b>	
NOP Instruction .....	121	SEEVAL® Evaluation and Programming System .....	128
<b>O</b>		Serialized Quick-Turnaround-Production	
OPTION Instruction .....	121	(SQTP) Devices .....	7
OPTION Register .....	20	SLEEP Instruction .....	122
Oscillator Configurations .....	97	Software Simulator (MPLAB-SIM) .....	126
Oscillator Start-up Timer (OST) .....	101	Special .....	99
Output of TMR2 .....	54	Special Features of the CPU .....	95
<b>P</b>		Special Function Registers .....	15
Package Marking Information .....	147	Stack .....	25
Packaging Information .....	147	Status Register .....	19
PCL and PCLATH .....	25	SUBLW Instruction .....	123
PCON Register .....	24	SUBWF Instruction .....	123
PICDEM-1 Low-Cost PICmicro Demo Board .....	127	SWAPF Instruction .....	124
PICDEM-2 Low-Cost PIC16CXX Demo Board .....	127	<b>T</b>	
PICDEM-3 Low-Cost PIC16CXXX Demo Board .....	127	T1CKPS0 bit .....	50
PICSTART® Plus Entry Level Development System .....	127	T1CKPS1 bit .....	50
PIE1 Register .....	22	T1CON Register .....	50
Pin Functions .....		T1OSCEN bit .....	50
RC6/TX/CK .....	71–88	T1SYNC bit .....	50
RC7/RX/DT .....	71–88	T2CKPS0 bit .....	55
Pinout Description .....	11	T2CKPS1 bit .....	55
PIR1 Register .....	23	T2CON Register .....	55
Port RB Interrupt .....	108		
PORTA .....	27		
PORTB .....	34		
Power Control/Status Register (PCON) .....	102		



Timer0		USART	
TIMER0.....	45	Asynchronous Mode.....	78
TIMER0 (TMR0) Interrupt.....	45	Asynchronous Receiver.....	80
TIMER0 (TMR0) Module.....	45	Asynchronous Reception.....	82
TMR0 with External Clock.....	47	Asynchronous Transmission.....	79
Timer1		Asynchronous Transmitter.....	78
Special Event Trigger (CCP).....	65	Baud Rate Generator (BRG).....	73
Switching Prescaler Assignment.....	49	Sampling.....	76
Timer2		Synchronous Master Mode.....	84
PR2 Register.....	66	Synchronous Master Reception.....	86
TMR2 to PR2 Match Interrupt.....	66	Synchronous Master Transmission.....	84
Timers		Synchronous Slave Mode.....	88
Timer1		Synchronous Slave Reception.....	88
Asynchronous Counter Mode.....	52	Synchronous Slave Transmit.....	88
Block Diagram.....	51	Transmit Block Diagram.....	78
Capacitor Selection.....	52	<b>V</b>	
External Clock Input.....	51	Voltage Reference Module.....	69
External Clock Input Timing.....	52	VRCON Register.....	69
Operation in Timer Mode.....	51	<b>W</b>	
Oscillator.....	52	Watchdog Timer (WDT).....	109
Prescaler.....	51, 53	WRITE.....	93
Resetting of Timer1 Registers.....	53	WRITING.....	93
Resetting Timer1 using a CCP Trigger Output...	53	WWW, On-Line Support.....	3
Synchronized Counter Mode.....	51	<b>X</b>	
T1CON.....	50	XORLW Instruction.....	124
TMR1H.....	52	XORWF Instruction.....	124
TMR1L.....	52		
Timer2			
Block Diagram.....	54		
Module.....	54		
Postscaler.....	54		
Prescaler.....	54		
T2CON.....	55		
Timing Diagrams			
Timer0.....	142		
Timer1.....	142		
USART Asynchronous Master Transmission.....	79		
USART RX Pin Sampling.....	76, 77		
USART Synchronous Reception.....	87		
USART Synchronous Transmission.....	85		
USART, Asynchronous Reception.....	81		
Timing Diagrams and Specifications.....	139		
TMR0 Interrupt.....	108		
TMR1CS bit.....	50		
TMR1ON bit.....	50		
TMR2ON bit.....	55		
TOUTPS0 bit.....	55		
TOUTPS1 bit.....	55		
TOUTPS2 bit.....	55		
TOUTPS3 bit.....	55		
TRIS Instruction.....	124		
TRISA.....	27		
TRISB.....	34		
TXSTA Register.....	71		
<b>U</b>			
Universal Synchronous Asynchronous Receiver			
Transmitter (USART).....	71		
Asynchronous Receiver			
Setting Up Reception.....	83		
Timing Diagram.....	81		
Asynchronous Receiver Mode			
Block Diagram.....	83		
Section.....	83		



## ON-LINE SUPPORT

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

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

### Connecting to the Microchip Internet Web Site

The Microchip web site is available by using your favorite Internet browser to attach to:

**[www.microchip.com](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 any currently available upgrade kits. The Hot Line Numbers are:

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

1-602-786-7302 for the rest of the world.

981103

**Trademarks:** The Microchip name, logo, PIC, PICmicro, PICSTART, PICMASTER and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. *FlexROM*, *MPLAB* and *fuzzy-LAB* are trademarks and *SQTP* is a service mark of Microchip in the U.S.A.

All other trademarks mentioned herein are the property of their respective companies.

# PIC16F62X

---

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (602) 786-7578.

Please list the following information, and use this outline to provide us with your comments about this Data Sheet.

To: Technical Publications Manager Total Pages Sent  
RE: Reader Response  
From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: **PIC16F62X** Literature Number: **DS40300B**

Questions:

1. What are the best features of this document?  
\_\_\_\_\_  
\_\_\_\_\_
2. How does this document meet your hardware and software development needs?  
\_\_\_\_\_  
\_\_\_\_\_
3. Do you find the organization of this data sheet easy to follow? If not, why?  
\_\_\_\_\_  
\_\_\_\_\_
4. What additions to the data sheet do you think would enhance the structure and subject?  
\_\_\_\_\_  
\_\_\_\_\_
5. What deletions from the data sheet could be made without affecting the overall usefulness?  
\_\_\_\_\_  
\_\_\_\_\_
6. Is there any incorrect or misleading information (what and where)?  
\_\_\_\_\_  
\_\_\_\_\_
7. How would you improve this document?  
\_\_\_\_\_  
\_\_\_\_\_
8. How would you improve our software, systems, and silicon products?  
\_\_\_\_\_  
\_\_\_\_\_

## PIC16F62X PRODUCT IDENTIFICATION SYSTEM

To order or to obtain information, e.g., on pricing or delivery, please use the listed part numbers, and refer to the factory or the listed sales offices.

PART NO.	-XX	X	/XX	XXX	
					<b>Pattern:</b> 3-Digit Pattern Code for QTP (blank otherwise)
					<b>Package:</b> P = PDIP SO = SOIC (Gull Wing, 300 mil body) SS = SSOP (209 mil)
					<b>Temperature Range:</b> - = 0°C to +70°C I = -40°C to +85°C E = -40°C to +125°C
					<b>Frequency Range:</b> 04 = 200kHz (LP osc) 04 = 4 MHz (XT and ER osc) 20 = 20 MHz (HS osc)
					<b>Device:</b> PIC16F62X :VDD range 3.0V to 5.5V PIC16F62XT:VDD range 3.0V to 5.5V (Tape and Reel) PIC16LF62X:VDD range 2.0V to 5.5V PIC16LF62XT:VDD range 2.0V to 5.5V (Tape and Reel)

**Examples:**

g) PIC16F627 - 04/P 301 = Commercial temp., PDIP package, 4 MHz, normal VDD limits, QTP pattern #301.

h) PIC16LF627- 04I/SO = Industrial temp., SOIC package, 200kHz, extended VDD limits.

## Sales and Support

### Data Sheets

Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (602) 786-7277
3. The Microchip Worldwide Site ([www.microchip.com](http://www.microchip.com))

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

### New Customer Notification System

Register on our web site ([www.microchip.com/cn](http://www.microchip.com/cn)) to receive the most current information on our products.

# PIC16F62X

---

NOTES:

NOTES:



## WORLDWIDE SALES AND SERVICE

### AMERICAS

#### Corporate Office

Microchip Technology Inc.  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-786-7200 Fax: 480-786-7277  
Technical Support: 480-786-7627  
Web Address: <http://www.microchip.com>

#### Atlanta

Microchip Technology Inc.  
500 Sugar Mill Road, Suite 200B  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0307

#### Boston

Microchip Technology Inc.  
5 Mount Royal Avenue  
Marlborough, MA 01752  
Tel: 508-480-9990 Fax: 508-480-8575

#### Chicago

Microchip Technology Inc.  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

#### Dallas

Microchip Technology Inc.  
4570 Westgrove Drive, Suite 160  
Addison, TX 75248  
Tel: 972-818-7423 Fax: 972-818-2924

#### Dayton

Microchip Technology Inc.  
Two Prestige Place, Suite 150  
Miamisburg, OH 45342  
Tel: 937-291-1654 Fax: 937-291-9175

#### Detroit

Microchip Technology Inc.  
Tri-Atria Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48334  
Tel: 248-538-2250 Fax: 248-538-2260

#### Los Angeles

Microchip Technology Inc.  
18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 949-263-1888 Fax: 949-263-1338

#### New York

Microchip Technology Inc.  
150 Motor Parkway, Suite 202  
Hauppauge, NY 11788  
Tel: 631-273-5305 Fax: 631-273-5335

#### San Jose

Microchip Technology Inc.  
2107 North First Street, Suite 590  
San Jose, CA 95131  
Tel: 408-436-7950 Fax: 408-436-7955

### AMERICAS (continued)

#### Toronto

Microchip Technology Inc.  
5925 Airport Road, Suite 200  
Mississauga, Ontario L4V 1W1, Canada  
Tel: 905-405-6279 Fax: 905-405-6253

### ASIA/PACIFIC

#### Hong Kong

Microchip Asia Pacific  
Unit 2101, Tower 2  
Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2-401-1200 Fax: 852-2-401-3431

#### Beijing

Microchip Technology, Beijing  
Unit 915, 6 Chaoyangmen Bei Dajie  
Dong Erhuan Road, Dongcheng District  
New China Hong Kong Manhattan Building  
Beijing 100027 PRC  
Tel: 86-10-85282100 Fax: 86-10-85282104

#### India

Microchip Technology Inc.  
India Liaison Office  
No. 6, Legacy, Convent Road  
Bangalore 560 025, India  
Tel: 91-80-229-0061 Fax: 91-80-229-0062

#### Japan

Microchip Technology Intl. Inc.  
Benex S-1 6F  
3-18-20, Shinyokohama  
Kohoku-Ku, Yokohama-shi  
Kanagawa 222-0033 Japan  
Tel: 81-45-471-6166 Fax: 81-45-471-6122

#### Korea

Microchip Technology Korea  
168-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea  
Tel: 82-2-554-7200 Fax: 82-2-558-5934

#### Shanghai

Microchip Technology  
RM 406 Shanghai Golden Bridge Bldg.  
2077 Yan'an Road West, Hong Qiao District  
Shanghai, PRC 200335  
Tel: 86-21-6275-5700 Fax: 86 21-6275-5060

### ASIA/PACIFIC (continued)

#### Singapore

Microchip Technology Singapore Pte Ltd.  
200 Middle Road  
#07-02 Prime Centre  
Singapore 188980  
Tel: 65-334-8870 Fax: 65-334-8850

#### Taiwan, R.O.C

Microchip Technology Taiwan  
10F-1C 207  
Tung Hua North Road  
Taipei, Taiwan, ROC  
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

### EUROPE

#### United Kingdom

Arizona Microchip Technology Ltd.  
505 Eskdale Road  
Winkers Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44 118 921 5858 Fax: 44-118 921-5835

#### Denmark

Microchip Technology Denmark ApS  
Regus Business Centre  
Lautrup hof 1-3  
Ballerup DK-2750 Denmark  
Tel: 45 4420 9895 Fax: 45 4420 9910

#### France

Arizona Microchip Technology SARL  
Parc d'Activite du Moulin de Massy  
43 Rue du Saule Trapu  
Batiment A - 1er Etage  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

#### Germany

Arizona Microchip Technology GmbH  
Gustav-Heinemann-Ring 125  
D-81739 München, Germany  
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

#### Italy

Arizona Microchip Technology SRL  
Centro Direzionale Colleoni  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-039-65791-1 Fax: 39-039-6899883

11/15/99



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and water 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, KEELOC® 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.

All rights reserved. © 1999 Microchip Technology Incorporated. Printed in the USA. 11/99 Printed on recycled paper.

Information contained in this publication regarding device applications and the like is intended for suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.