



# **PIC16F627A/628A/648A**

## **Data Sheet**

FLASH-Based

8-Bit CMOS Microcontrollers

---

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products.

---

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. 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.

#### Trademarks

The Microchip name and logo, the Microchip logo, KEELOQ, MPLAB, PIC, PICmicro, PICSTART and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

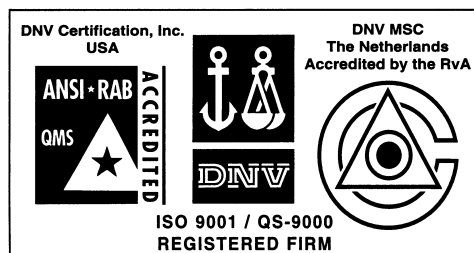
dsPIC, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2002, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



*Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.*



# PIC16F627A/628A/648A

## 18-pin FLASH-Based 8-Bit CMOS Microcontrollers

### High Performance RISC CPU:

- Operating speeds from DC - 20 MHz
- Interrupt capability
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes
- 35 single word instructions
  - All instructions single cycle except branches

### Special Microcontroller Features:

- Internal and external oscillator options
  - Precision Internal 4 MHz oscillator factory calibrated to  $\pm 1\%$
  - Low Power Internal 37 kHz oscillator
  - External Oscillator support for crystals and resonators.
- Power saving SLEEP mode
- Programmable weak pull-ups on PORTB
- Multiplexed Master Clear/Input-pin
- Watchdog Timer with independent oscillator for reliable operation
- Low voltage programming
- In-Circuit Serial Programming™ (via two pins)
- Programmable code protection
- Brown-out Reset
- Power-on Reset
- Power-up Timer and Oscillator Start-up Timer
- Wide operating voltage range. (2.0 - 5.5V)
- Industrial and extended temperature range
- High Endurance FLASH/EEPROM Cell
  - 100,000 write FLASH endurance
  - 1,000,000 write EEPROM endurance
  - 100 year data retention

### Low Power Features:

- Standby Current:
  - 100 nA @ 2.0V, typical
- Operating Current:
  - 12  $\mu$ A @ 32 kHz, 2.0V, typical
  - 120  $\mu$ A @ 1 MHz, 2.0V, typical
- Watchdog Timer Current
  - 1  $\mu$ A @ 2.0V, typical
- Timer1 oscillator current:
  - 1.2  $\mu$ A @ 32 kHz, 2.0V, typical
- Dual Speed Internal Oscillator:
  - Run-time selectable between 4 MHz and 37 kHz
  - 4  $\mu$ s wake-up from SLEEP, 3.0V, typical

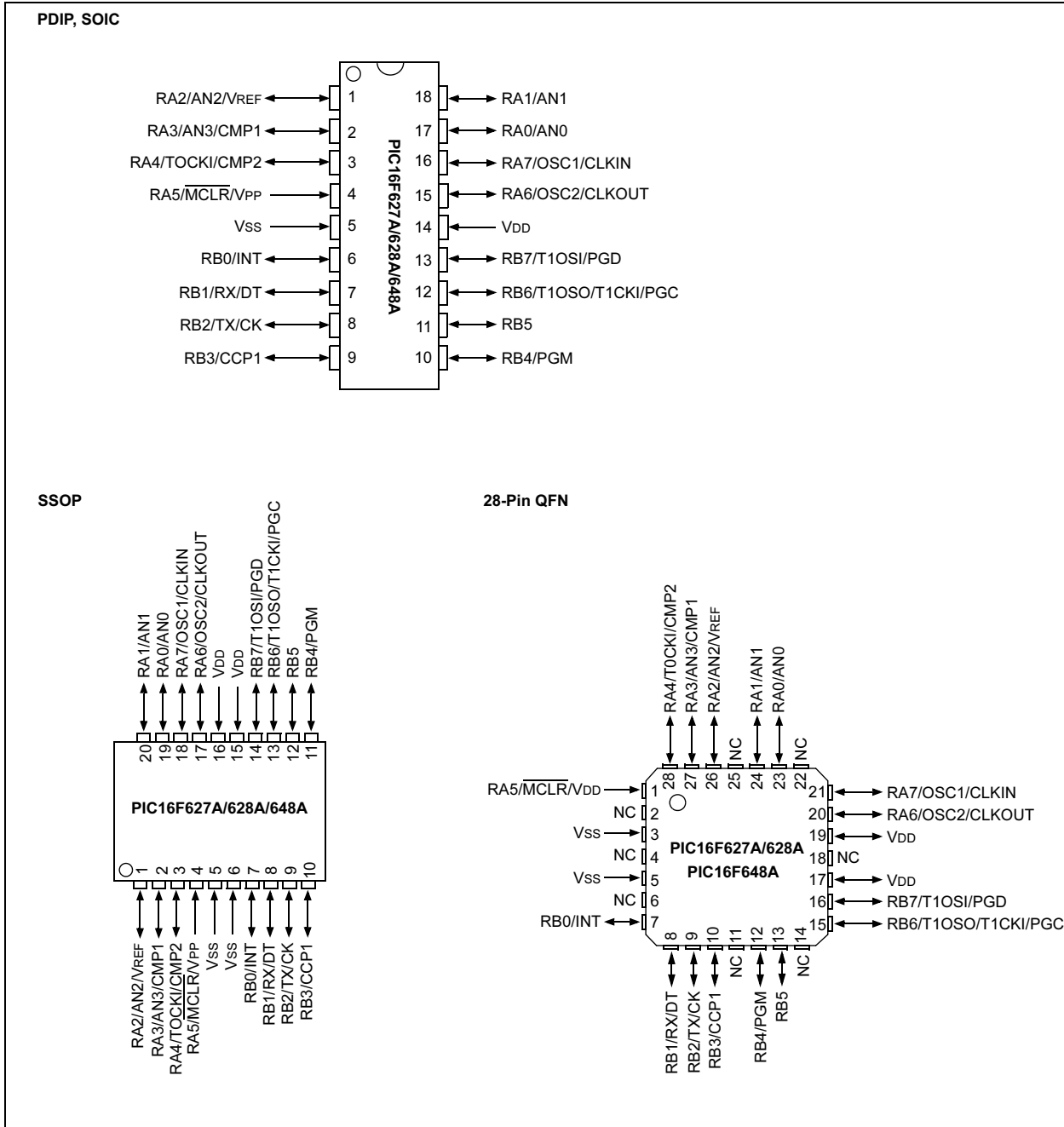
### Peripheral Features:

- 16 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
  - Selectable internal or external 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 module
  - 16-bit Capture/Compare
  - 10-bit PWM
- Addressable Universal Synchronous/Asynchronous Receiver/Transmitter USART/SCI

Device	Program Memory	Data Memory		I/O	CCP (PWM)	USART	Comparators	Timers 8/16-bit
	FLASH (words)	SRAM (bytes)	EEPROM (bytes)					
PIC16F627A	1024	224	128	16	1	Y	2	2/1
PIC16F628A	2048	224	128	16	1	Y	2	2/1
PIC16F648A	4096	256	256	16	1	Y	2	2/1

# PIC16F627A/628A/648A

## Pin Diagrams



## Table of Contents

1.0	General Description.....	5
2.0	PIC16F627A/628A/648A Device Varieties .....	7
3.0	Architectural Overview .....	9
4.0	Memory Organization .....	15
5.0	I/O Ports .....	31
6.0	Timer0 Module .....	45
7.0	Timer1 Module .....	48
8.0	Timer2 Module .....	52
9.0	Capture/Compare/PWM (CCP) Module .....	55
10.0	Comparator Module.....	61
11.0	Voltage Reference Module.....	67
12.0	Universal Synchronous Asynchronous Receiver Transmitter (USART) Module.....	69
13.0	Data EEPROM Memory .....	89
14.0	Special Features of the CPU.....	93
15.0	Instruction Set Summary .....	111
16.0	Development Support.....	125
17.0	Electrical Specifications.....	131
18.0	DC and AC Characteristics Graphs and Tables.....	147
19.0	Packaging Information.....	149

## TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at [docerrors@mail.microchip.com](mailto:docerrors@mail.microchip.com) or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

### 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).

### Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. 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: (480) 792-7277

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.

### Customer Notification System

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

# PIC16F627A/628A/648A

---

---

NOTES:

# PIC16F627A/628A/648A

## 1.0 GENERAL DESCRIPTION

The PIC16F627A/628A/648A are 18-Pin FLASH-based members of the versatile PIC16CXX family of low cost, high performance, CMOS, fully-static, 8-bit microcontrollers.

All PICmicro® microcontrollers employ an advanced RISC architecture. The PIC16F627A/628A/648A 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, complemented by a large register set.

PIC16F627A/628A/648A microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

PIC16F627A/628A/648A devices have integrated features to reduce external components, thus reducing system cost, enhancing system reliability and reducing power consumption.

The PIC16F627A/628A/648A has 8 oscillator configurations. The single-pin RC oscillator provides a low cost solution. The LP oscillator minimizes power consumption, XT is a standard crystal, and INTOSC is a self-contained precision two-speed internal oscillator. The

HS is for High-Speed crystals. The EC mode is for an external clock source.

The SLEEP (Power-down) mode offers power savings. Users can wake-up the chip from SLEEP through several external interrupts, internal interrupts and RESETS.

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 PIC16F627A/628A/648A mid-range microcontroller families.

A simplified block diagram of the PIC16F627A/628A/648A is shown in Figure 3-1.

The PIC16F627A/628A/648A series fits in applications ranging from battery chargers to low power remote sensors. The FLASH technology makes customizing application programs (detection levels, pulse generation, timers, etc.) extremely fast and convenient. The small footprint packages makes 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 PIC16F627A/628A/648A very versatile.

### 1.1 Development Support

The PIC16F627A/628A/648A family is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a low cost in-circuit debugger, a low cost development programmer and a full-featured programmer. A Third Party "C" compiler support tool is also available.

**TABLE 1-1: PIC16F627A/628A/648A FAMILY OF DEVICES**

		PIC16F627A	PIC16F628A	PIC16F648A	PIC16LF627A	PIC16LF628A	PIC16LF648A
Clock	Maximum Frequency of Operation (MHz)	20	20	20	4	4	4
Memory	FLASH Program Memory (words)	1024	2048	4096	1024	2048	4096
	RAM Data Memory (bytes)	224	224	256	224	224	256
	EEPROM Data Memory (bytes)	128	128	256	128	128	256
Peripherals	Timer module(s)	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2
	Comparator(s)	2	2	2	2	2	2
	Capture/Compare/PWM modules	1	1	1	1	1	1
	Serial Communications	USART	USART	USART	USART	USART	USART
	Internal Voltage Reference	Yes	Yes	Yes	Yes	Yes	Yes
Features	Interrupt Sources	10	10	10	10	10	10
	I/O Pins	16	16	16	16	16	16
	Voltage Range (Volts)	3.0-5.5	3.0-5.5	3.0-5.5	2.0-5.5	2.0-5.5	2.0-5.5
	Brown-out Reset	Yes	Yes	Yes	Yes	Yes	Yes
	Packages	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN	18-pin DIP, SOIC, 20-pin SSOP, 28-pin QFN

All PICmicro® Family devices have Power-on Reset, selectable Watchdog Timer, selectable Code Protect and high I/O current capability. All PIC16F627A/628A/648A Family devices use serial programming with clock pin RB6 and data pin RB7.

# PIC16F627A/628A/648A

---

---

NOTES:



## 2.0 PIC16F627A/628A/648A 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 PIC16F627A/628A/648A Product Identification System, 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

FLASH devices can be erased and re-programmed electrically. This allows the same device to be used for prototype development, pilot programs and production.

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

# PIC16F627A/628A/648A

---

NOTES:

## 3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16F627A/628A/648A family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16F627A/628A/648A uses a Harvard architecture, in which program and data are accessed from separate memories using separate buses. 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.

Table 3-1 lists device memory sizes (FLASH, Data and EEPROM).

**TABLE 3-1: DEVICE MEMORY LIST**

Device	Memory		
	FLASH Program	RAM Data	EEPROM Data
PIC16F627A	1024 x 14	224 x 8	128 x 8
PIC16F628A	2048 x 14	224 x 8	128 x 8
PIC16F648A	4096 x 14	256 x 8	256 x 8
PIC16LF627A	1024 x 14	224 x 8	128 x 8
PIC16LF628A	2048 x 14	224 x 8	128 x 8
PIC16LF648A	4096 x 14	256 x 8	256 x 8

The PIC16F627A/628A/648A 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 PIC16F627A/628A/648A 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 PIC16F627A/628A/648A simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16F627A/628A/648A 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.

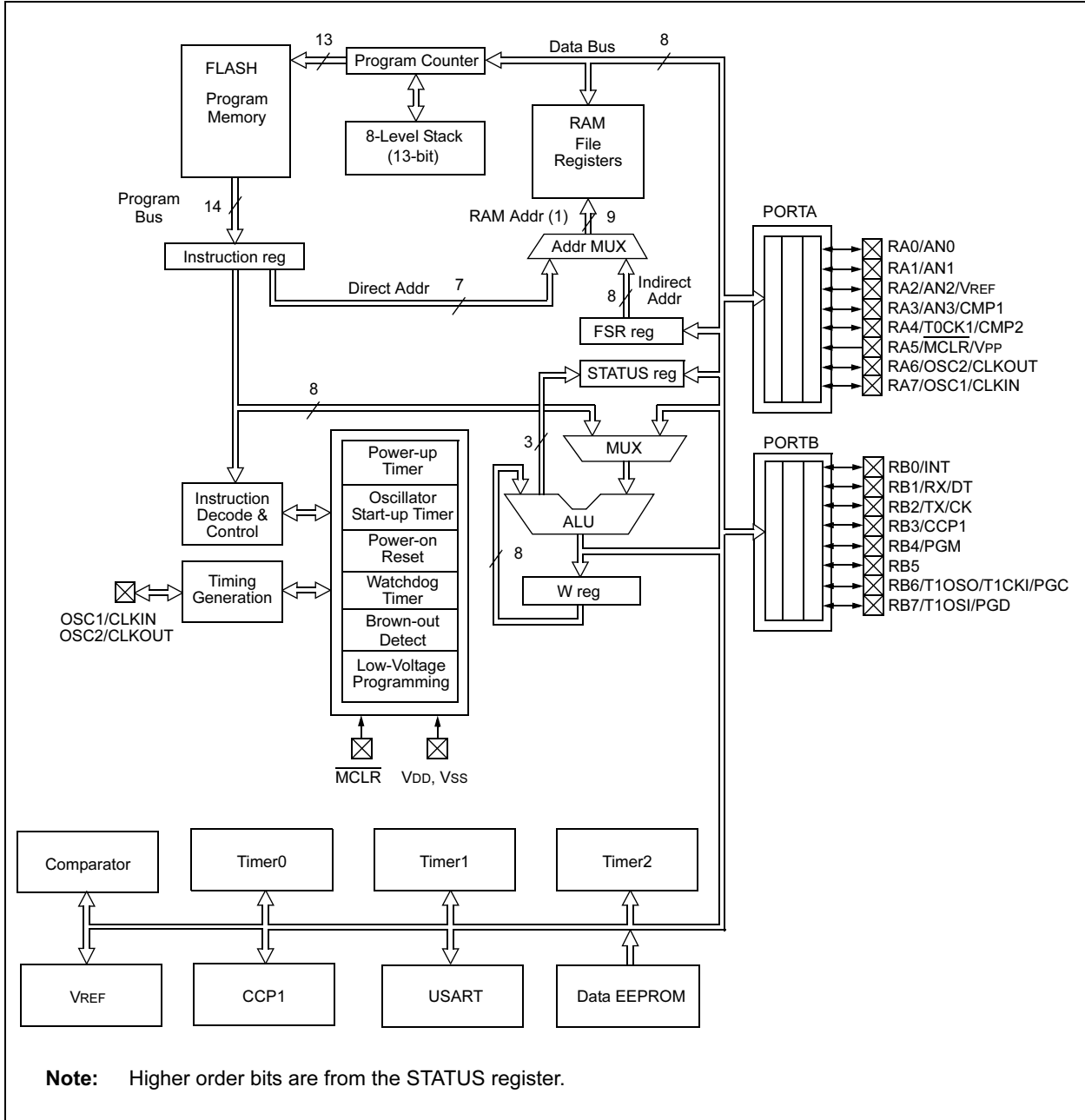
Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, bit in subtraction. See the *SUBLW* and *SUBWF* instructions for examples.

A simplified block diagram is shown in Figure 3-1, and a description of the device pins in Table 3-2.

Two types of data memory are provided on the PIC16F627A/628A/648A 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. These data are 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. Data are lost when power is removed.

# PIC16F627A/628A/648A

FIGURE 3-1: BLOCK DIAGRAM



# PIC16F627A/628A/648A

**TABLE 3-2: PIC16F627A/628A/648A PINOUT DESCRIPTION**

Name	Function	Input Type	Output Type	Description
RA0/AN0	RA0	ST	CMOS	Bi-directional I/O port
	AN0	AN	—	Analog comparator input
RA1/AN1	RA1	ST	CMOS	Bi-directional I/O port
	AN1	AN	—	Analog comparator input
RA2/AN2/VREF	RA2	ST	CMOS	Bi-directional I/O port
	AN2	AN	—	Analog comparator input
	VREF	—	AN	VREF output
RA3/AN3/CMP1	RA3	ST	CMOS	Bi-directional I/O port
	AN3	AN	—	Analog comparator input
	CMP1	—	CMOS	Comparator 1 output
RA4/T0CKI/CMP2	RA4	ST	OD	Bi-directional I/O port
	T0CKI	ST	—	Timer0 clock input
	CMP2	—	OD	Comparator 2 output
RA5/MCLR/VPP	RA5	ST	—	Input port
	MCLR	ST	—	Master clear. When configured as MCLR, this pin is an active low RESET to the device. Voltage on MCLR/VPP must not exceed VDD during normal device operation.
	VPP	—	—	Programming voltage input.
RA6/OSC2/CLKOUT	RA6	ST	CMOS	Bi-directional I/O port
	OSC2	—	XTAL	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
	CLKOUT	—	CMOS	In RC/INTOSC mode, OSC2 pin can output CLKOUT, which has 1/4 the frequency of OSC1
RA7/OSC1/CLKIN	RA7	ST	CMOS	Bi-directional I/O port
	OSC1	XTAL	—	Oscillator crystal input
	CLKIN	ST	—	External clock source input. RC biasing pin.
RB0/INT	RB0	TTL	CMOS	Bi-directional I/O port. Can be software programmed for internal weak pull-up.
	INT	ST	—	External interrupt.
RB1/RX/DT	RB1	TTL	CMOS	Bi-directional I/O port. Can be software programmed for internal weak pull-up.
	RX	ST	—	USART receive pin
	DT	ST	CMOS	Synchronous data I/O.
RB2/TX/CK	RB2	TTL	CMOS	Bi-directional I/O port. Can be software programmed for internal weak pull-up.
	TX	—	CMOS	USART transmit pin
	CK	ST	CMOS	Synchronous clock I/O.
RB3/CCP1	RB3	TTL	CMOS	Bi-directional I/O port. Can be software programmed for internal weak pull-up.
	CCP1	ST	CMOS	Capture/Compare/PWM I/O

Legend: O = Output  
 — = Not used  
 TTL = TTL Input

CMOS = CMOS Output  
 I = Input  
 OD = Open Drain Output

P = Power  
 ST = Schmitt Trigger Input  
 AN = Analog

# PIC16F627A/628A/648A

**TABLE 3-2: PIC16F627A/628A/648A PINOUT DESCRIPTION**

Name	Function	Input Type	Output Type	Description
RB4/PGM	RB4	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
	PGM	ST	—	Low voltage programming input pin. When low voltage programming is enabled, the interrupt-on-pin change and weak pull-up resistor are disabled.
RB5	RB5	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
RB6/T1OSO/T1CKI/PGC	RB6	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
	T1OSO	—	XTAL	Timer1 oscillator output.
	T1CKI	ST	—	Timer1 clock input.
	PGC	ST	—	ICSP Programming Clock.
RB7/T1OSI/PGD	RB7	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
	T1OSI	XTAL	—	Timer1 oscillator input.
	PGD	ST	CMOS	ICSP Data I/O
VSS	VSS	Power	—	Ground reference for logic and I/O pins
VDD	VDD	Power	—	Positive supply for logic and I/O pins

Legend: O = Output  
 — = Not used  
 TTL = TTL Input

CMOS = CMOS Output  
 I = Input  
 OD = Open Drain Output

P = Power  
 ST = Schmitt Trigger Input  
 AN = Analog

## 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.

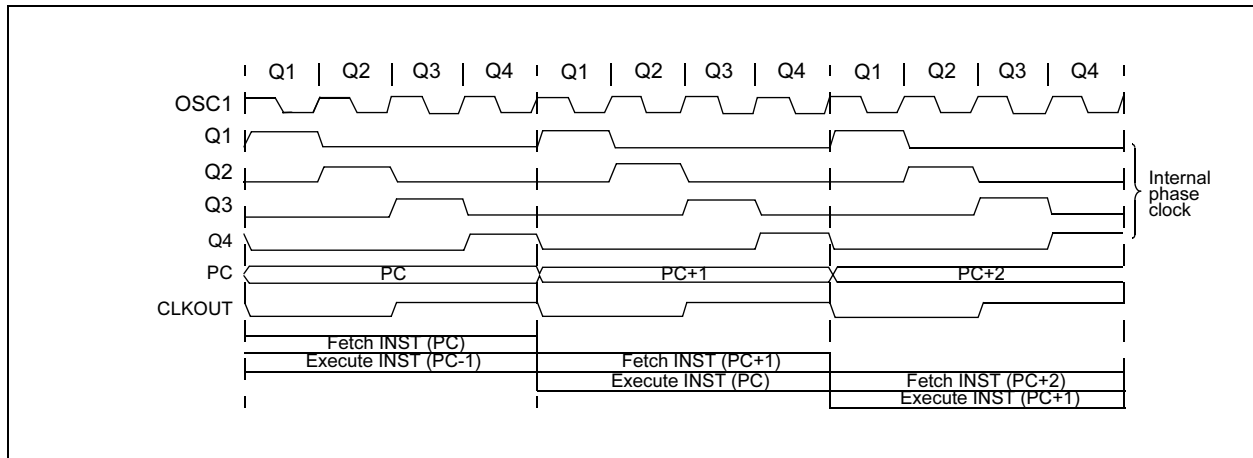
## 3.2 Instruction Flow/Pipelining

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).

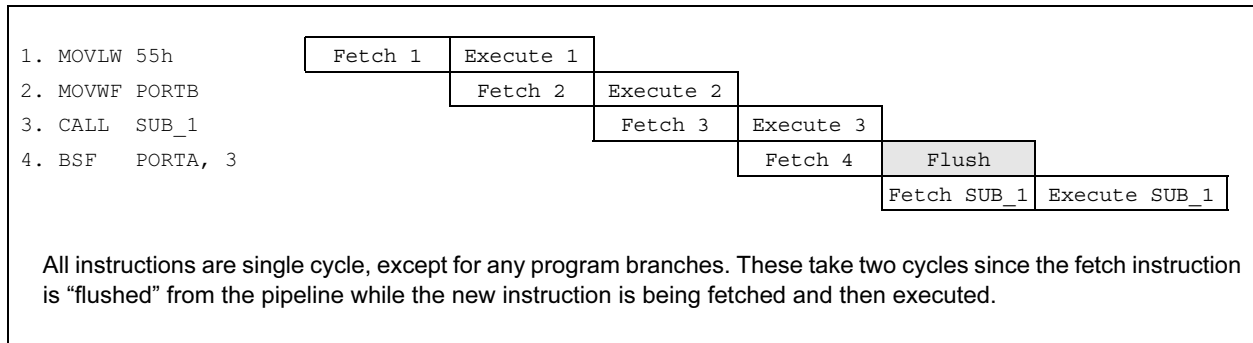
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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**



# PIC16F627A/628A/648A

---

---

NOTES:



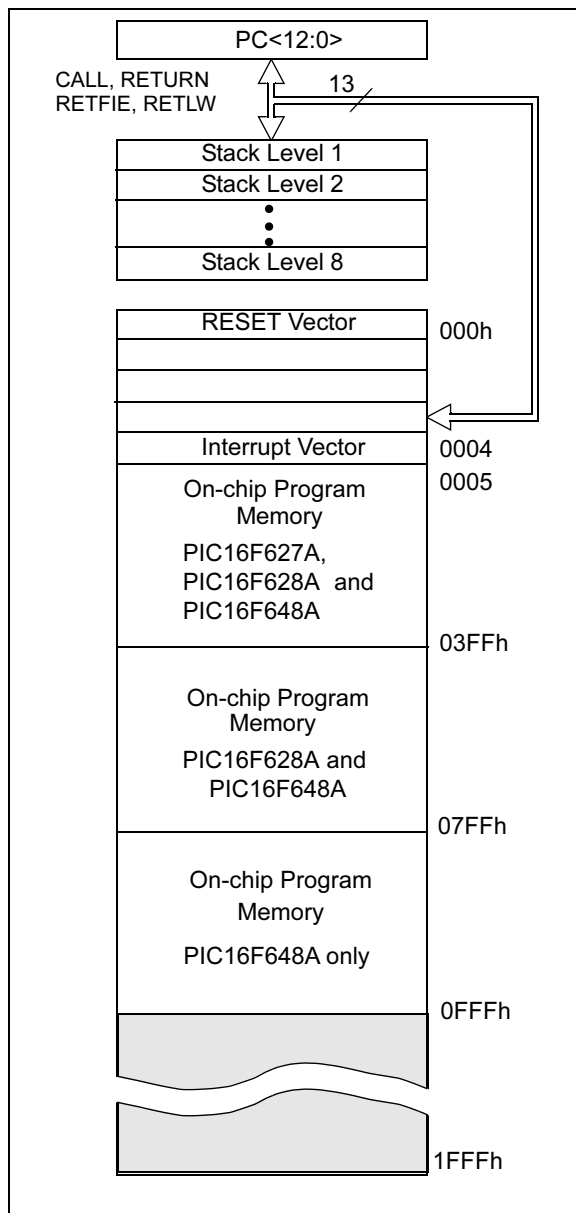
# PIC16F627A/628A/648A

## 4.0 MEMORY ORGANIZATION

### 4.1 Program Memory Organization

The PIC16F627A/628A/648A 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 PIC16F627A, 2K x 14 (0000h - 07FFh) for the PIC16F628A and 4K x 14 (0000h - 0FFFh) for the PIC16F648A are physically implemented. Accessing a location above these boundaries will cause a wrap-around within the first 1K x 14 space (PIC16F627A), 2K x 14 space (PIC16F628A) or 4K x 14 space (PIC16F648A). The RESET vector is at 0000h and the interrupt vector is at 0004h (Figure 4-1).

**FIGURE 4-1: PROGRAM MEMORY MAP AND STACK**



### 4.2 Data Memory Organization

The data memory (Figure 4-2 and Figure 4-3) is partitioned into four banks, which contain the general purpose registers and the Special Function Registers (SFR). The SFR's are located in the first 32 locations of each Bank. There are general purpose registers implemented as static RAM in each Bank. Table 4-1 lists the general purpose register available in each of the four banks.

**TABLE 4-1: GENERAL PURPOSE STATIC RAM REGISTERS**

	PIC16F627A/628A	PIC16F648A
Bank0	20-7Fh	20-7Fh
Bank1	A0h-FF	A0h-FF
Bank2	120h-14Fh, 170h-17Fh	120h-17Fh
Bank3	1F0h-1FFh	1F0h-1FFh

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

Table 4-2 lists how to access the four banks of registers via the STATUS Register bits RP1 and RP0.

**TABLE 4-2: ACCESS TO BANKS OF REGISTERS**

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

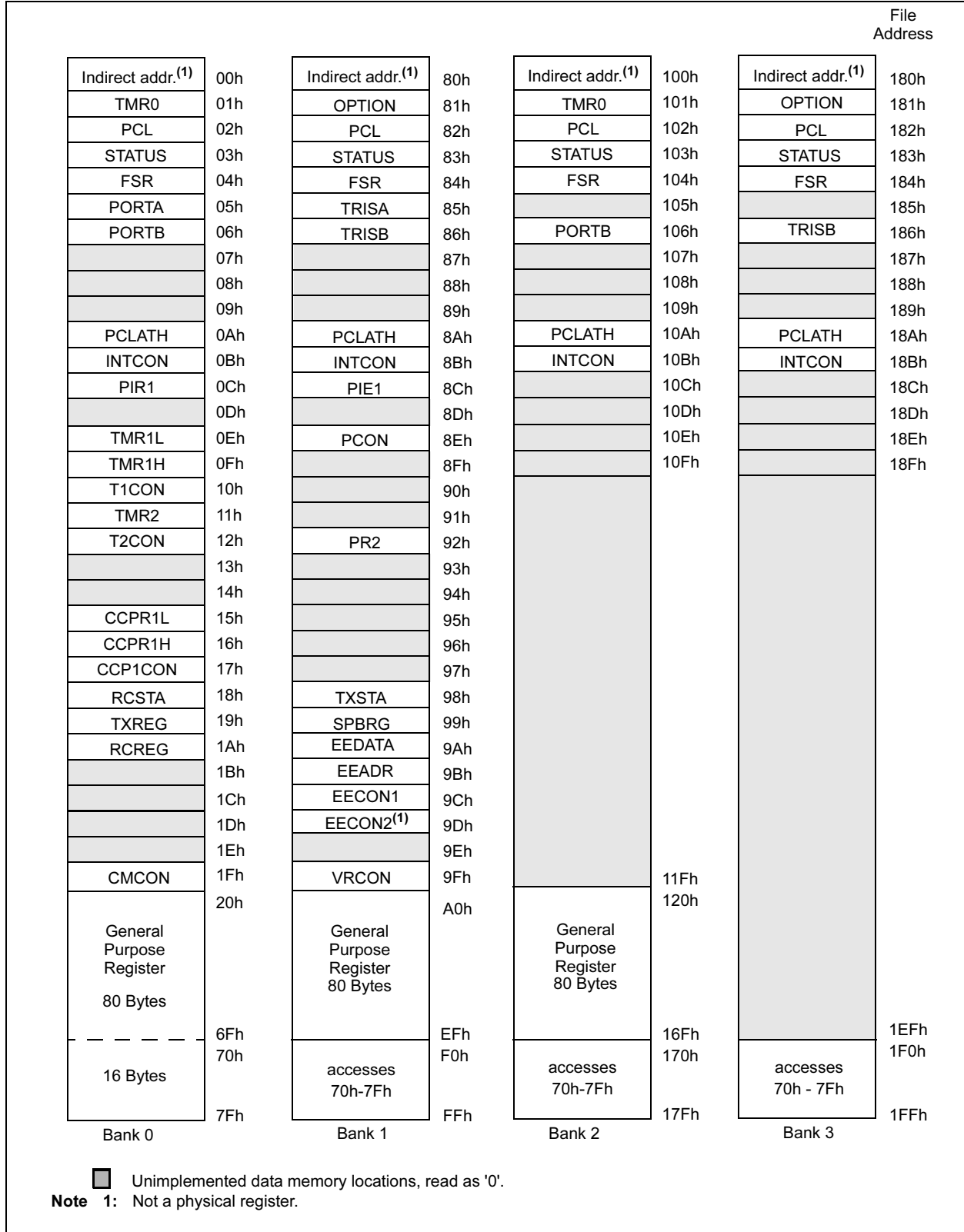
#### 4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 224 x 8 in the PIC16F627A/628A and 256 x 8 in the PIC16F648A. Each is accessed either directly or indirectly through the File Select Register (FSR), See Section 4.4.



# PIC16F627A/628A/648A

**FIGURE 4-3: DATA MEMORY MAP OF THE PIC16F648A**



# PIC16F627A/628A/648A

## 4.2.2 SPECIAL FUNCTION REGISTERS

The SFRs are registers used by the CPU and Peripheral functions for controlling the desired operation of the device (Table 4-3). These registers are static RAM.

The special registers can be classified into two sets (core and peripheral). The SFRs 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 4-3: 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 <sup>(1)</sup>	Details on Page
<b>Bank 0</b>											
00h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	28
01h	TMR0	Timer0 module's Register								xxxx xxxx	45
02h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	28
03h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22
04h	FSR	Indirect data memory address pointer								xxxx xxxx	28
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000	31
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	36
07h	—	Unimplemented								—	—
08h	—	Unimplemented								—	—
09h	—	Unimplemented								—	—
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter				---	0000	28
0Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	24
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	26
0Dh	—	Unimplemented								—	—
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1								xxxx xxxx	48
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1								xxxx xxxx	48
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYN\overline{C}}$	TMR1CS	TMR1ON	--00 0000	48
11h	TMR2	TMR2 module's register								0000 0000	52
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	52
13h	—	Unimplemented								—	—
14h	—	Unimplemented								—	—
15h	CCPR1L	Capture/Compare/PWM register (LSB)								xxxx xxxx	55
16h	CCPR1H	Capture/Compare/PWM register (MSB)								xxxx xxxx	55
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	55
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 000x	69
19h	TXREG	USART Transmit data register								0000 0000	76
1Ah	RCREG	USART Receive data register								0000 0000	79
1Bh	—	Unimplemented								—	—
1Ch	—	Unimplemented								—	—
1Dh	—	Unimplemented								—	—
1Eh	—	Unimplemented								—	—
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	61

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

**Note 1:** For the Initialization Condition for Registers Tables, refer to Table 14-6 and Table 14-7.

# PIC16F627A/628A/648A

**TABLE 4-4: 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 <sup>(1)</sup>	Details on Page
<b>Bank 1</b>											
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	28
81h	OPTION	$\overline{\text{BPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	23
82h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	28
83h	STATUS	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	22
84h	FSR	Indirect data memory address pointer								xxxx xxxx	28
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	31
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	36
87h	—	Unimplemented								—	—
88h	—	Unimplemented								—	—
89h	—	Unimplemented								—	—
8Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---0 0000	28
8Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	24
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	0000 -000	25
8Dh	—	Unimplemented								—	—
8Eh	PCON	—	—	—	—	OSCF	—	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	---- 1-0x	27
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	—	Unimplemented								—	—
92h	PR2	Timer2 Period Register								1111 1111	52
93h	—	Unimplemented								—	—
94h	—	Unimplemented								—	—
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	71
99h	SPBRG	Baud Rate Generator Register								0000 0000	71
9Ah	EEDATA	EEPROM data register								xxxx xxxx	89
9Bh	EEADR	EEPROM address register								xxxx xxxx	90
9Ch	EECON1	—	—	—	—	WRERR	WREN	WR	RD	---- x000	90
9Dh	EECON2	EEPROM control register 2 (not a physical register)								---- ----	90
9Eh	—	Unimplemented								—	—
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	67

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

**Note 1:** For the Initialization Condition for Registers Tables, refer to Table 14-6 and Table 14-7.

# PIC16F627A/628A/648A

**TABLE 4-5: 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 <sup>(1)</sup>	Details on Page
<b>Bank 2</b>											
100h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	28
101h	TMR0	Timer0 module's Register								xxxx xxxx	45
102h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	28
103h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22
104h	FSR	Indirect data memory address pointer								xxxx xxxx	28
105h	—	Unimplemented								—	—
106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	36
107h	—	Unimplemented								—	—
108h	—	Unimplemented								—	—
109h	—	Unimplemented								—	—
10Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter				---0 0000	28	
10Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	24
10Ch	—	Unimplemented								—	—
10Dh	—	Unimplemented								—	—
10Eh	—	Unimplemented								—	—
10Fh	—	Unimplemented								—	—
110h	—	Unimplemented								—	—
111h	—	Unimplemented								—	—
112h	—	Unimplemented								—	—
113h	—	Unimplemented								—	—
114h	—	Unimplemented								—	—
115h	—	Unimplemented								—	—
116h	—	Unimplemented								—	—
117h	—	Unimplemented								—	—
118h	—	Unimplemented								—	—
119h	—	Unimplemented								—	—
11Ah	—	Unimplemented								—	—
11Bh	—	Unimplemented								—	—
11Ch	—	Unimplemented								—	—
11Dh	—	Unimplemented								—	—
11Eh	—	Unimplemented								—	—
11Fh	—	Unimplemented								—	—

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

**Note 1:** For the Initialization Condition for Registers Tables, refer to Table 14-6 and Table 14-7.

# PIC16F627A/628A/648A

**TABLE 4-6: 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 <sup>(1)</sup>	Details on Page
<b>Bank 3</b>											
180h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	28
181h	OPTION	RBP $\bar{U}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	23
182h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	28
183h	STATUS	IRP	RP1	RP0	$\bar{T}O$	$\bar{P}D$	Z	DC	C	0001 1xxx	22
184h	FSR	Indirect data memory address pointer								xxxx xxxx	28
185h	—	Unimplemented								—	—
186h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	36
187h	—	Unimplemented								—	—
188h	—	Unimplemented								—	—
189h	—	Unimplemented								—	—
18Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---0 0000	28
18Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	24
18Ch	—	Unimplemented								—	—
18Dh	—	Unimplemented								—	—
18Eh	—	Unimplemented								—	—
18Fh	—	Unimplemented								—	—
190h	—	Unimplemented								—	—
191h	—	Unimplemented								—	—
192h	—	Unimplemented								—	—
193h	—	Unimplemented								—	—
194h	—	Unimplemented								—	—
195h	—	Unimplemented								—	—
196h	—	Unimplemented								—	—
197h	—	Unimplemented								—	—
198h	—	Unimplemented								—	—
199h	—	Unimplemented								—	—
19Ah	—	Unimplemented								—	—
19Bh	—	Unimplemented								—	—
19Ch	—	Unimplemented								—	—
19Dh	—	Unimplemented								—	—
19Eh	—	Unimplemented								—	—
19Fh	—	Unimplemented								—	—

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

**Note 1:** For the Initialization Condition for Registers Tables, refer to Table 14-6 and Table 14-7.

# PIC16F627A/628A/648A

## 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  $\overline{\text{TO}}$  and  $\overline{\text{PD}}$  bits are non-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  $\overline{\text{C}}$  and  $\overline{\text{DC}}$  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, 83h, 103h, 183h)

	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	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C
bit 7								bit 0

- 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)  
 00 = Bank 0 (00h - 7Fh)  
 01 = Bank 1 (80h - FFh)  
 10 = Bank 2 (100h - 17Fh)  
 11 = Bank 3 (180h - 1FFh)
  - bit 4  **$\overline{\text{TO}}$ :** Time out bit  
 1 = After power-up, `CLRWDT` instruction, or `SLEEP` instruction  
 0 = A WDT time out occurred
  - bit 3  **$\overline{\text{PD}}$ :** Power-down bit  
 1 = After power-up or by the `CLRWDT` 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.

Legend:

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



# PIC16F627A/628A/648A

## 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, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
RBP $\bar{U}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	
bit 7								bit 0

- bit 7 **RBP $\bar{U}$** : 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

**Legend:**

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

# PIC16F627A/628A/648A

## 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, 8Bh, 10Bh, 18Bh)

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
							bit 0
							bit 7

- 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

Legend:

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

# PIC16F627A/628A/648A

## 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-0	R/W-0	R/W-0	R/W-0
EEIE	CMIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

- 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

**Legend:**

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

# PIC16F627A/628A/648A

## 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-0	R/W-0	R/W-0	R/W-0	
EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	
bit 7								bit 0

- 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 output has changed  
           0 = Comparator output 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

Legend:

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

# PIC16F627A/628A/648A

## 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 Reset.

**Note:** BOR is unknown on Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if BOR is cleared, indicating a brown-out has occurred. The BOR STATUS bit is a “don't care” and is not necessarily predictable if the brown-out circuit is disabled (by clearing the 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-0	R/W-x
	—	—	—	—	OSCF	—	POR	BOR
bit 7								bit 0

- bit 7-4    **Unimplemented:** Read as '0'
- bit 3     **OSCF:** INTOSC oscillator frequency  
1 = 4 MHz typical  
0 = 37 kHz typical
- bit 2     **Unimplemented:** Read as '0'
- bit 1     **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     **BOR:** Brown-out Reset 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)

**Legend:**

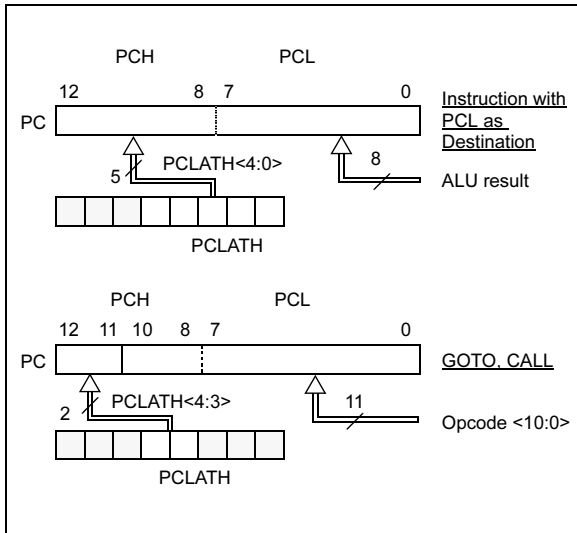
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared    x = Bit is unknown

# PIC16F627A/628A/648A

## 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 4-4 shows the two situations for loading the PC. The upper example in Figure 4-4 shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in Figure 4-4 shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

**FIGURE 4-4: 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 PIC16F627A/628A/648A family has an 8-level deep x 13-bit wide hardware stack (Figure 4-1). 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 RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

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

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

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

## 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 4-5.

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

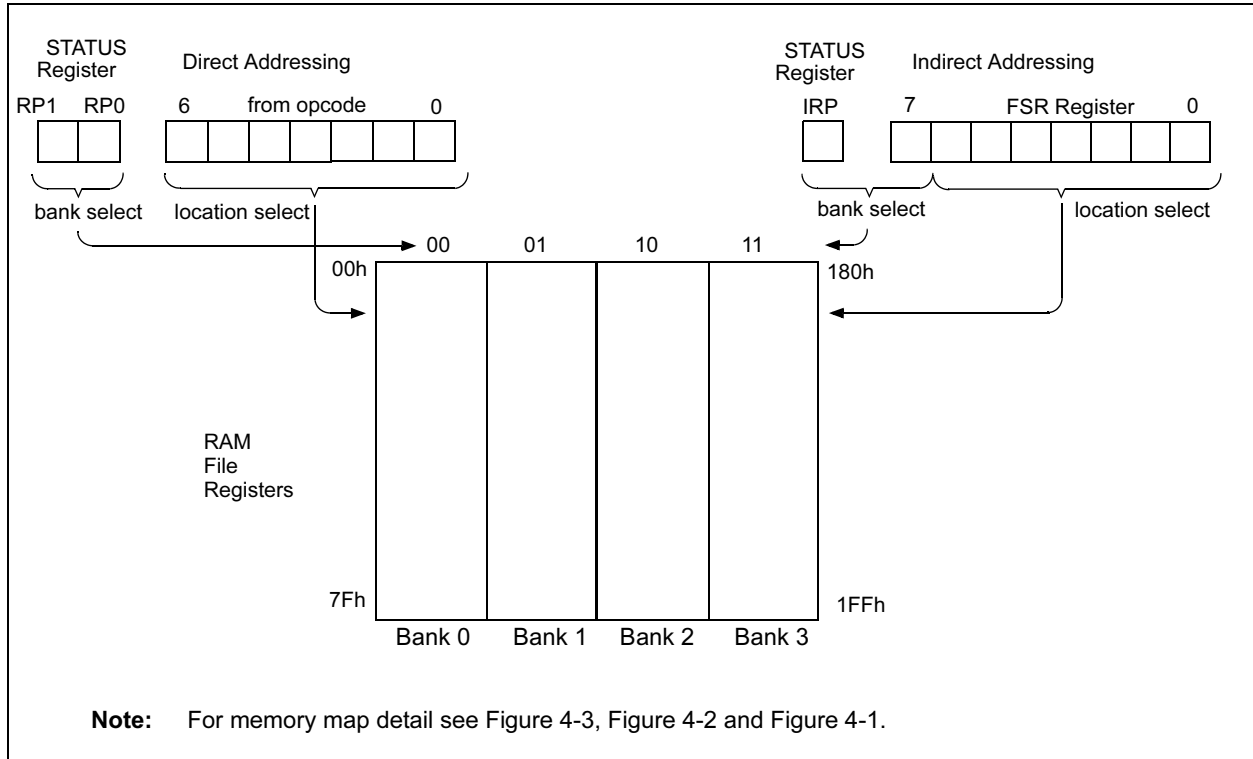
### EXAMPLE 4-1: Indirect Addressing

```

MOV LW 0x20 ;initialize pointer
MOV WF FSR ;to RAM
NEXT   CLRF INDF ;clear INDF register
       INCF FSR ;inc pointer
       BTFS FSR,4 ;all done?
       GOTO NEXT ;no clear next
                          ;yes continue
    
```

# PIC16F627A/628A/648A

**FIGURE 4-5: DIRECT/INDIRECT ADDRESSING PIC16F627A/628A/648A**



# PIC16F627A/628A/648A

---

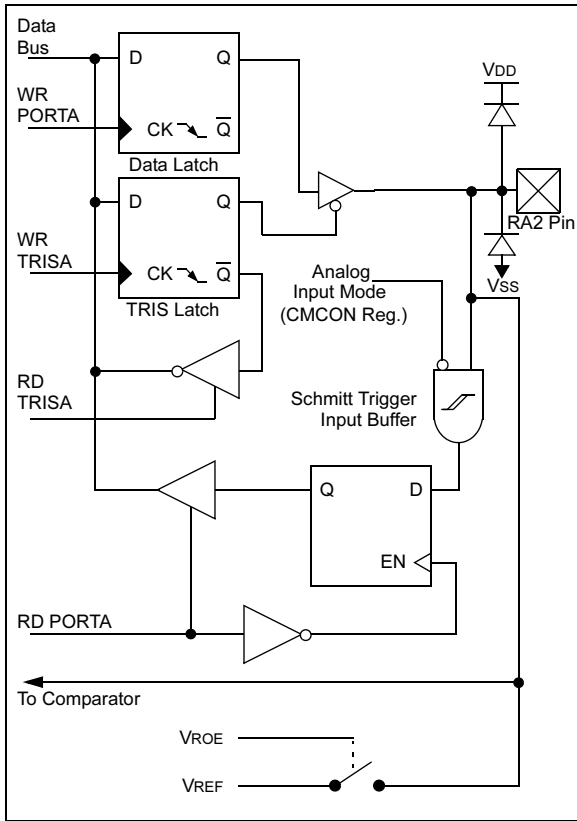
NOTES:



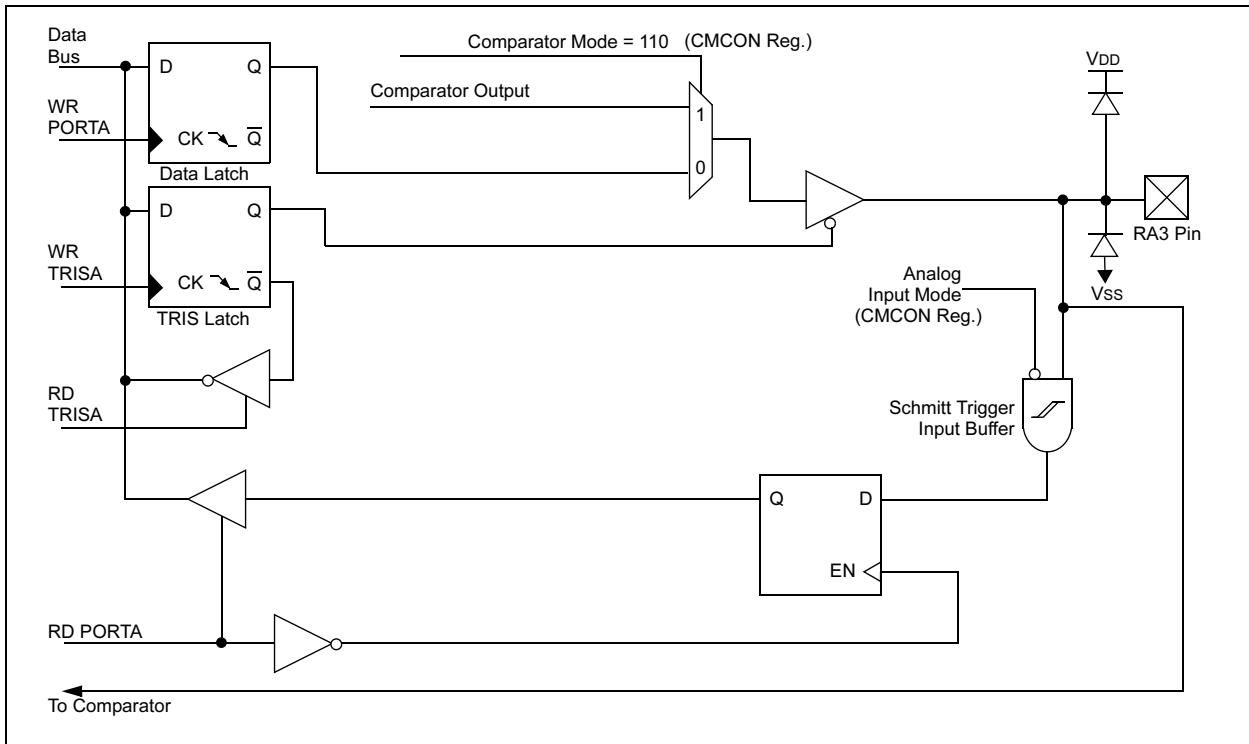


# PIC16F627A/628A/648A

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

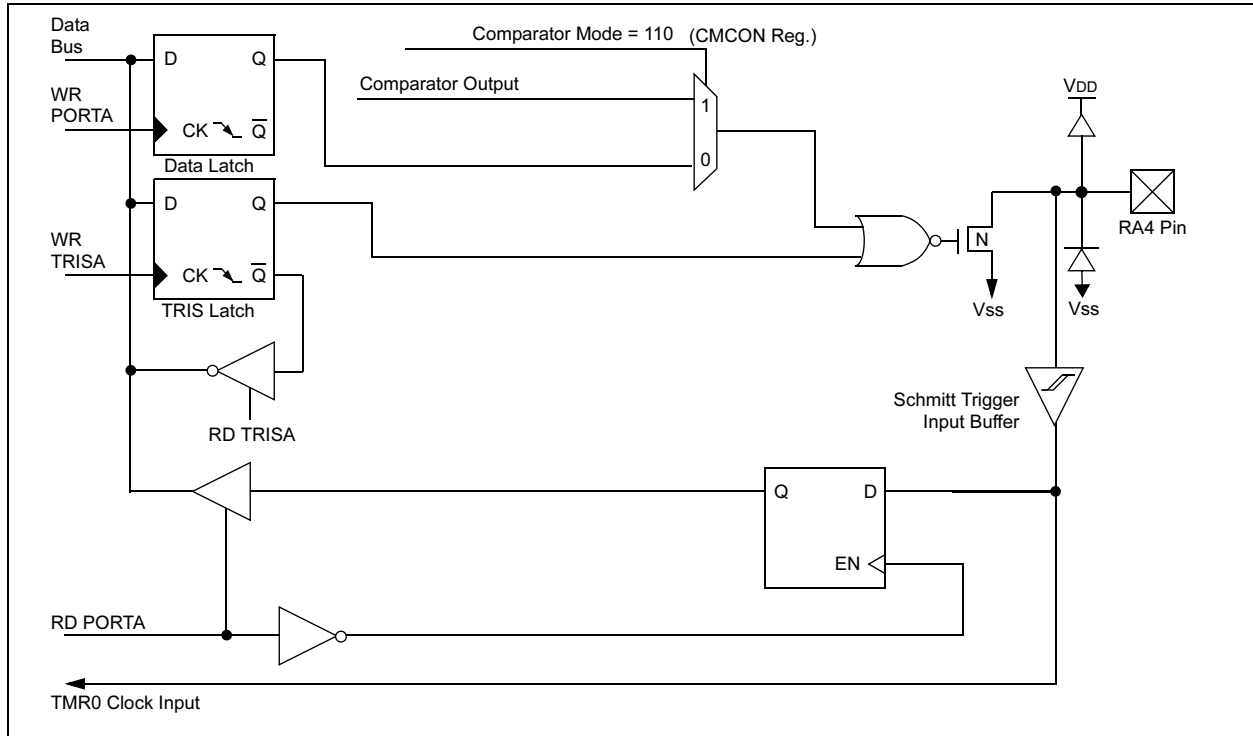


**FIGURE 5-3: BLOCK DIAGRAM OF THE RA3/AN3 PIN**

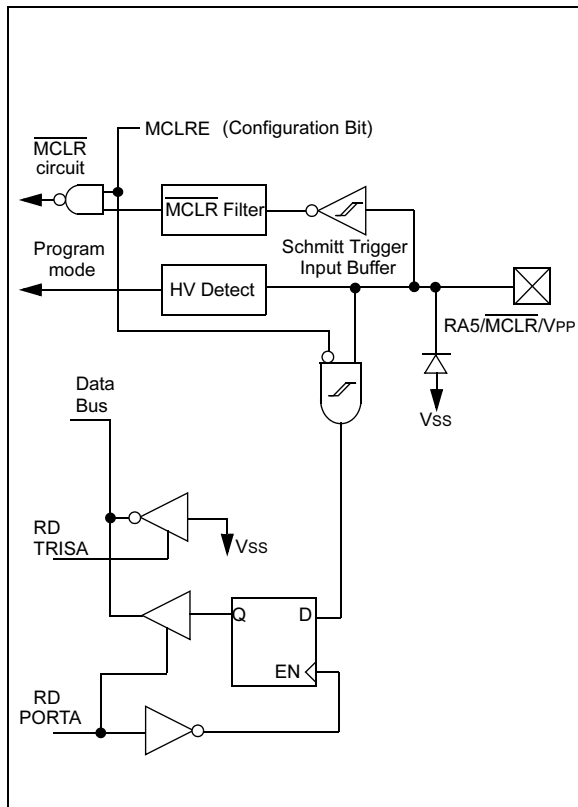


# PIC16F627A/628A/648A

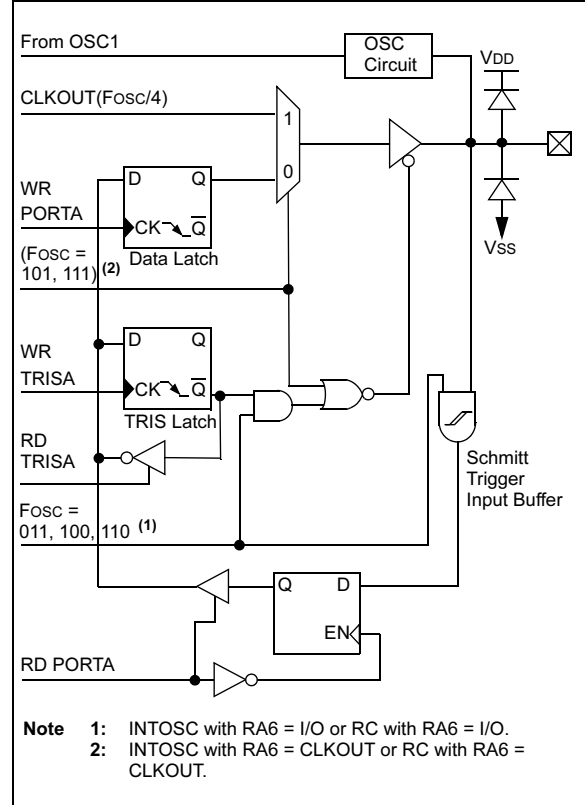
**FIGURE 5-4: BLOCK DIAGRAM OF RA4/T0CKI PIN**



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

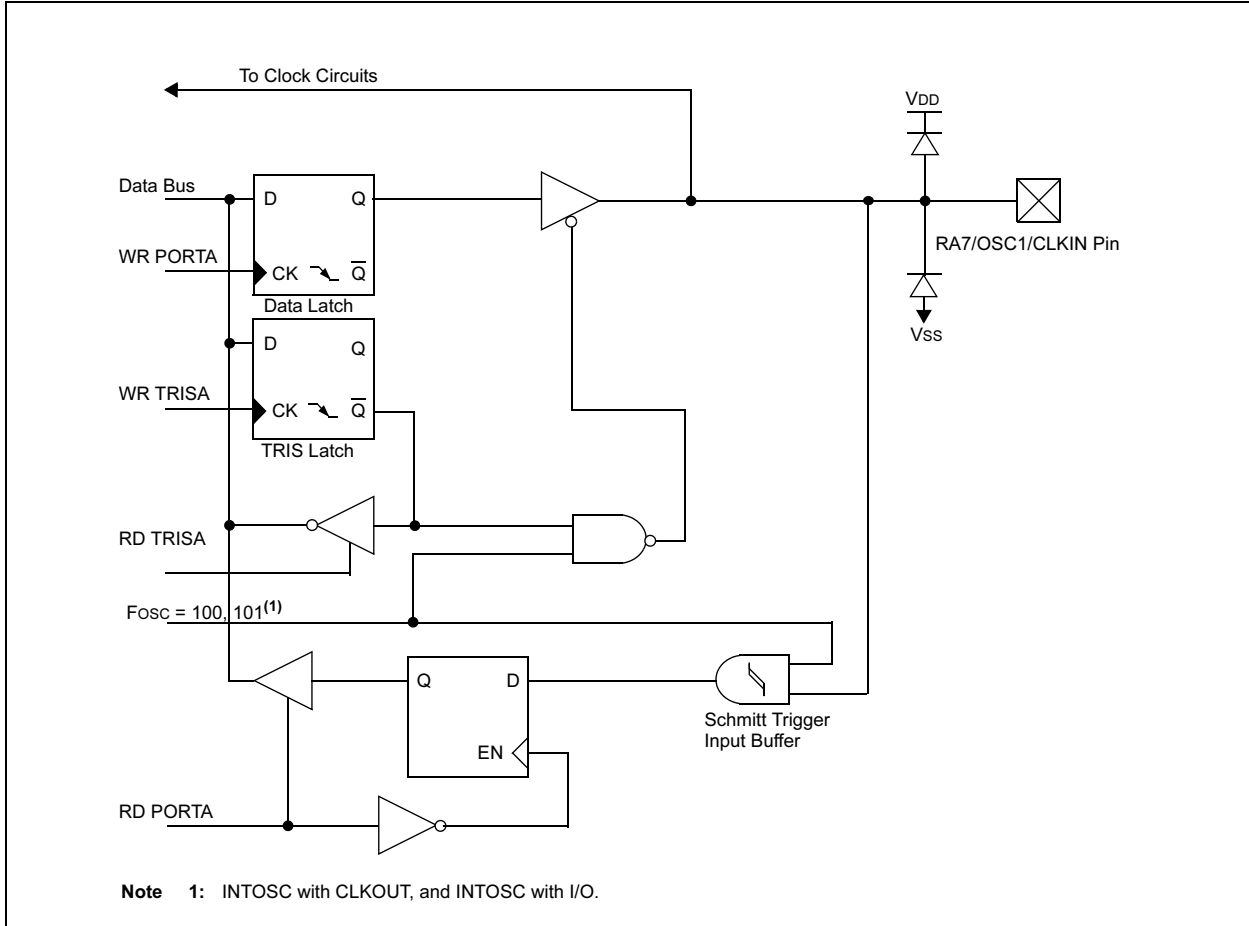


**FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKOUT PIN**



# PIC16F627A/628A/648A

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





# PIC16F627A/628A/648A

**TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA<sup>(1)</sup>**

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 <sup>(2)</sup>	RA4	RA3	RA2	RA1	RA0	xxxx 0000	qqqu 0000
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 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, q = value depends on condition, shaded = unimplemented

- Note 1:** Shaded bits are not used by PORTA.  
**Note 2:** MCLRE Configuration Bit sets RA5 functionality.

## 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 external 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. Alternate port functions may override TRIS setting when enabled.

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 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, RB<7:4>, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB<7:4> 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:

- Any read or write of PORTB. This will end the mismatch condition.
- 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.

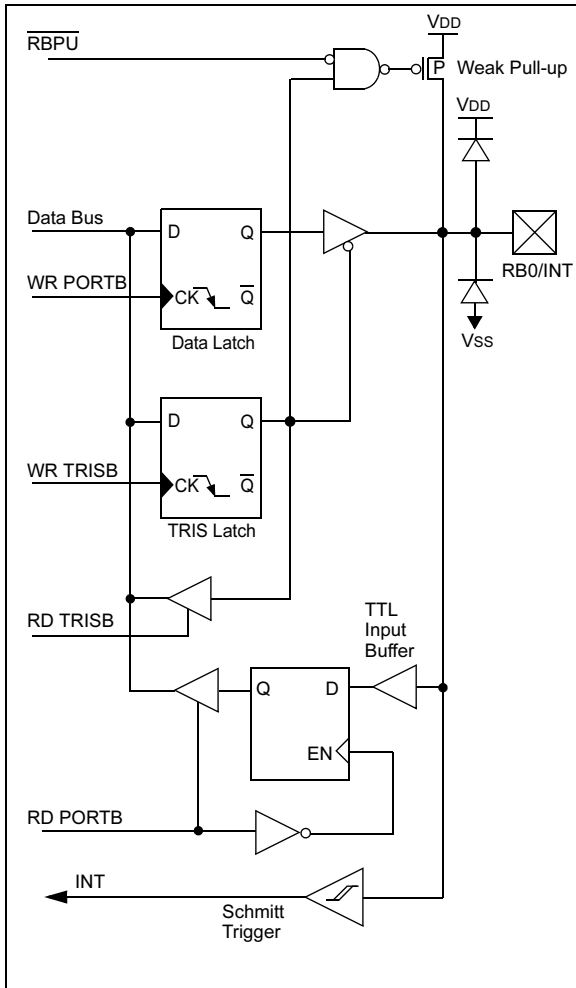
This interrupt on mismatch feature, together with software configurable pull-ups on these four pins allow easy interface to a key pad and make it possible for wake-up on key-depression. (See AN552)

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

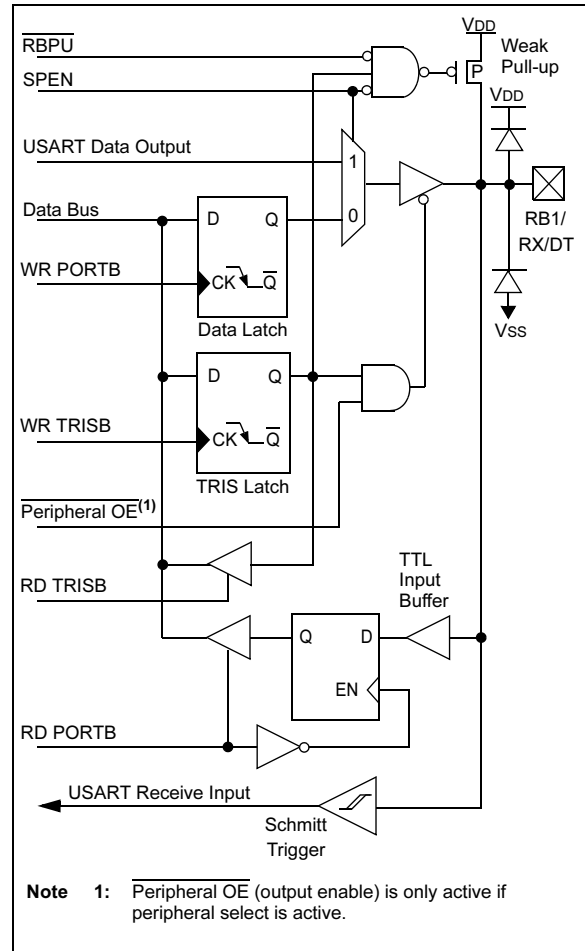
The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

# PIC16F627A/628A/648A

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

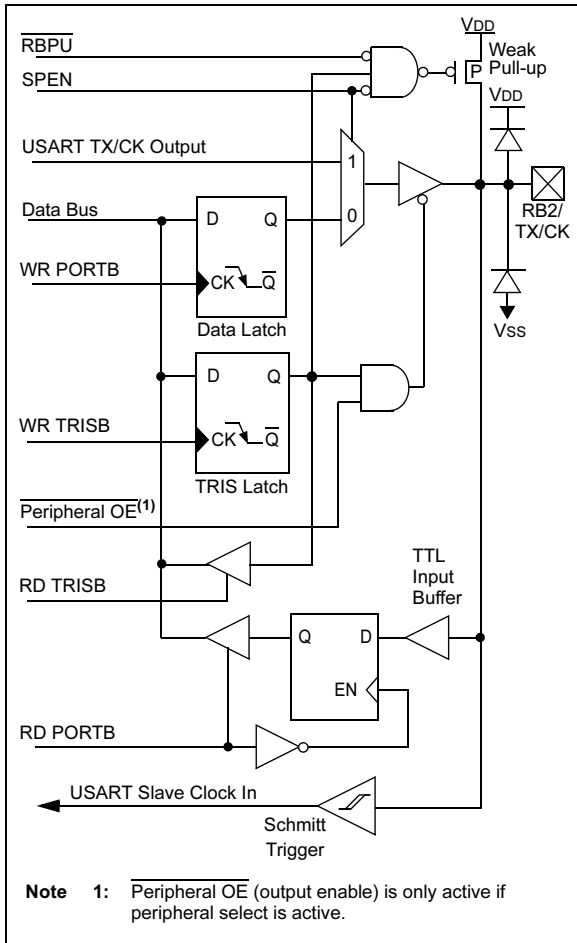


**FIGURE 5-9: BLOCK DIAGRAM OF RB1/RX/DT PIN**

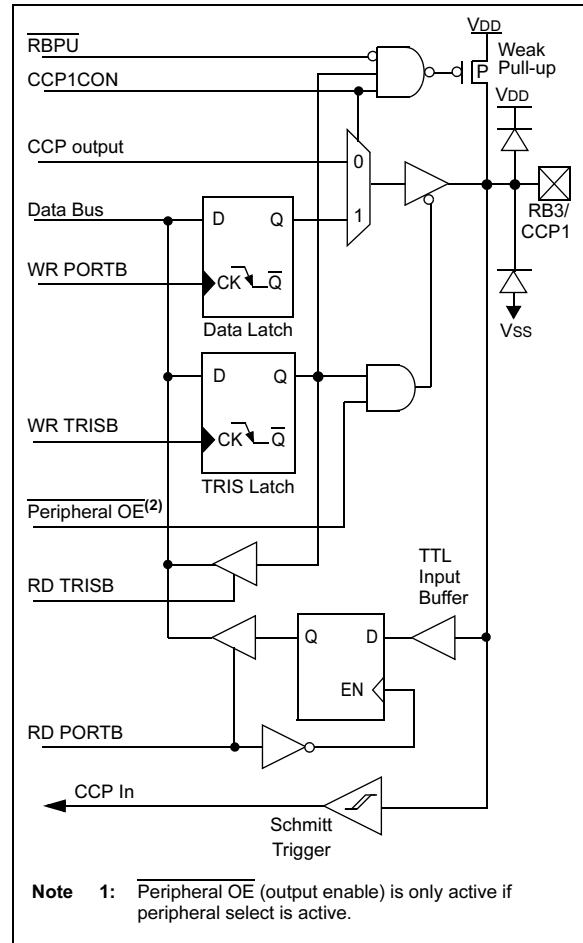


# PIC16F627A/628A/648A

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



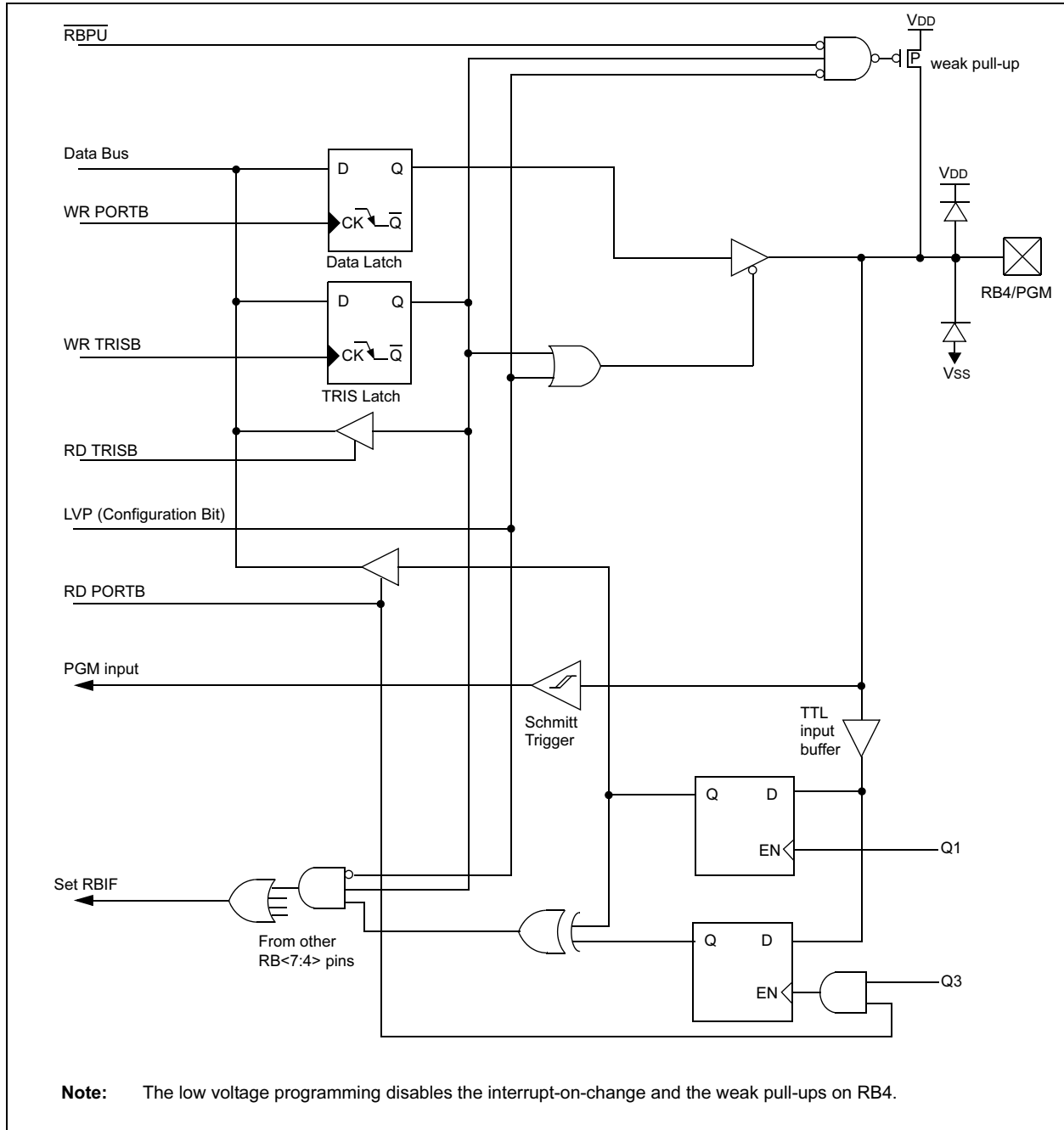
**FIGURE 5-11: BLOCK DIAGRAM OF RB3/CCP1 PIN**





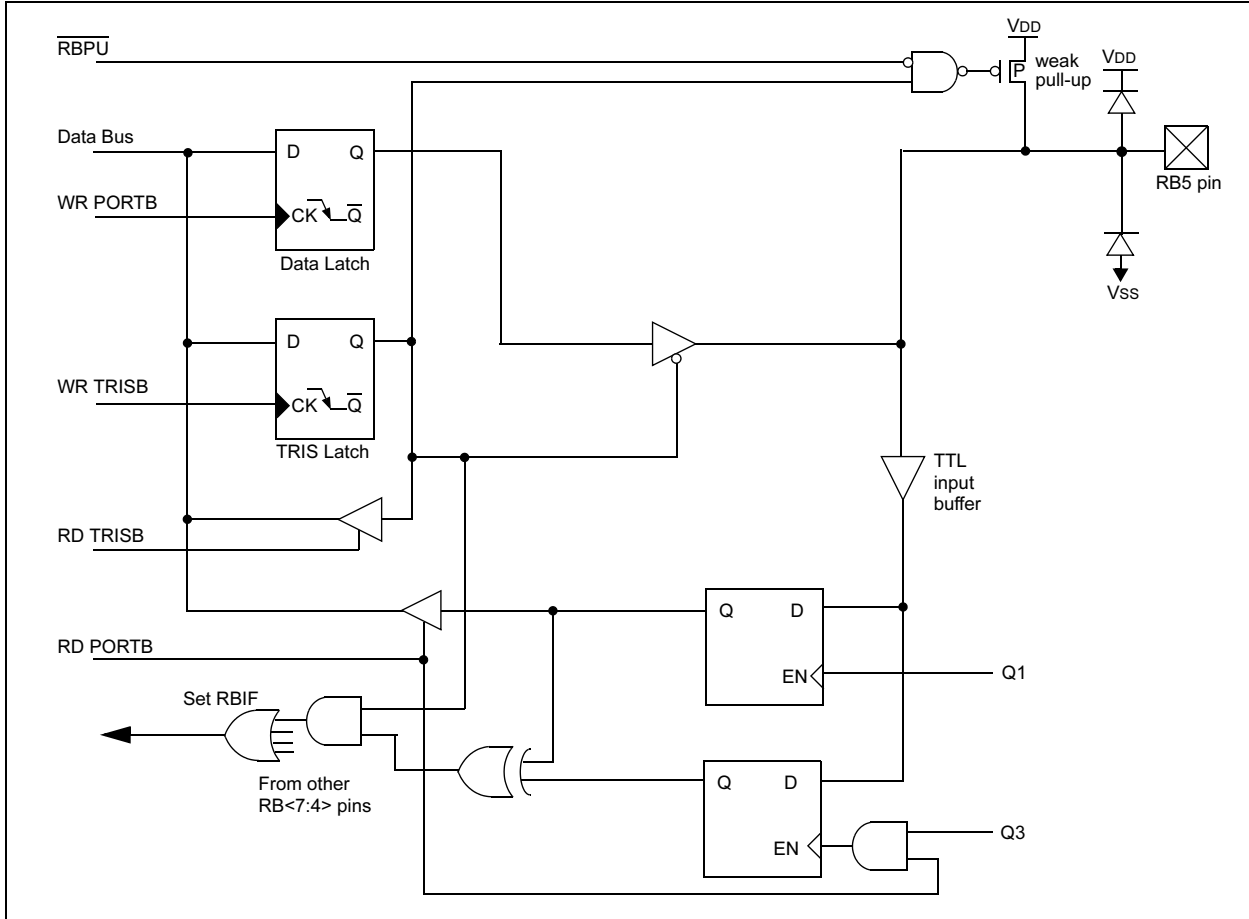
# PIC16F627A/628A/648A

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



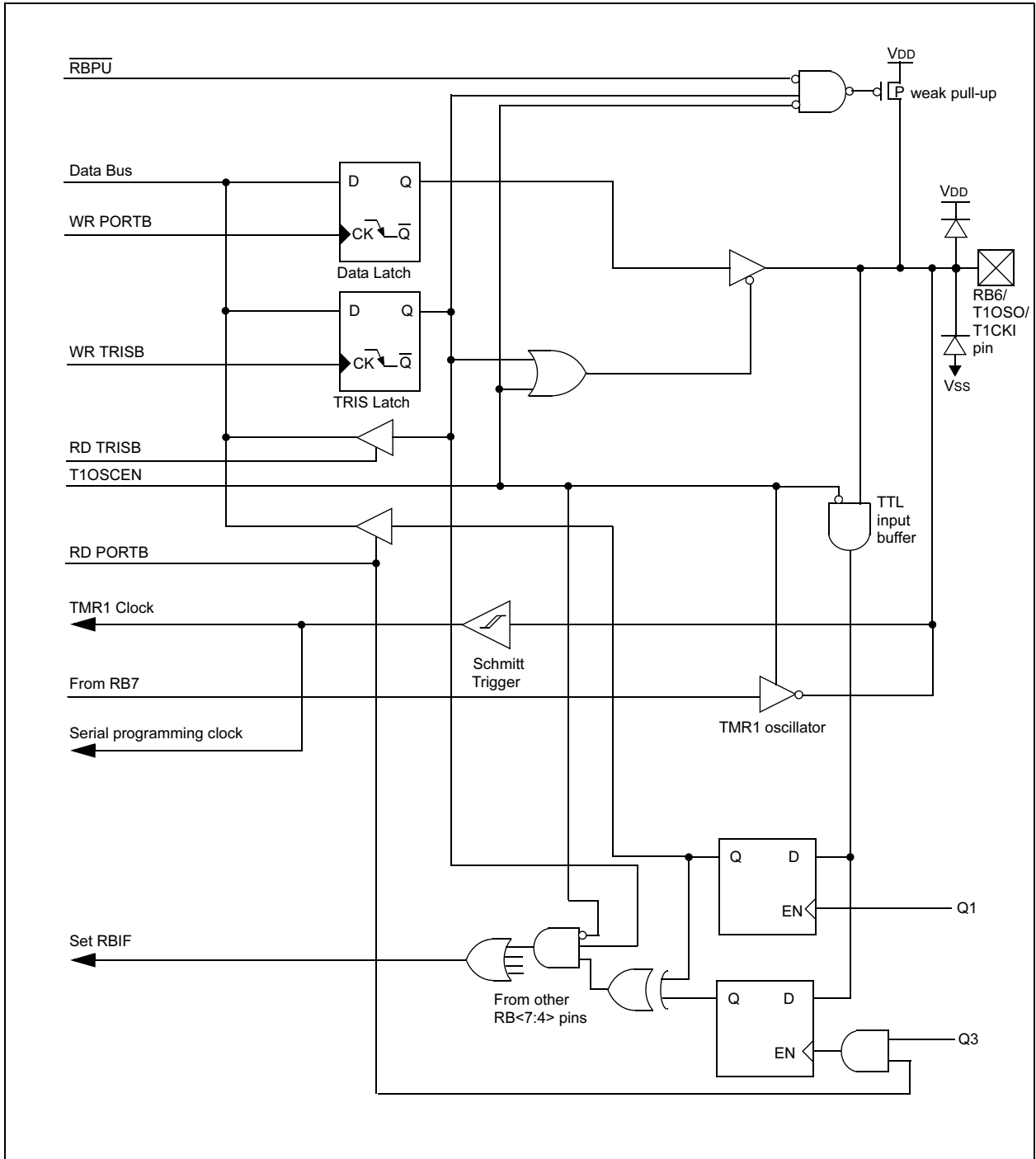
# PIC16F627A/628A/648A

FIGURE 5-13: BLOCK DIAGRAM OF RB5 PIN



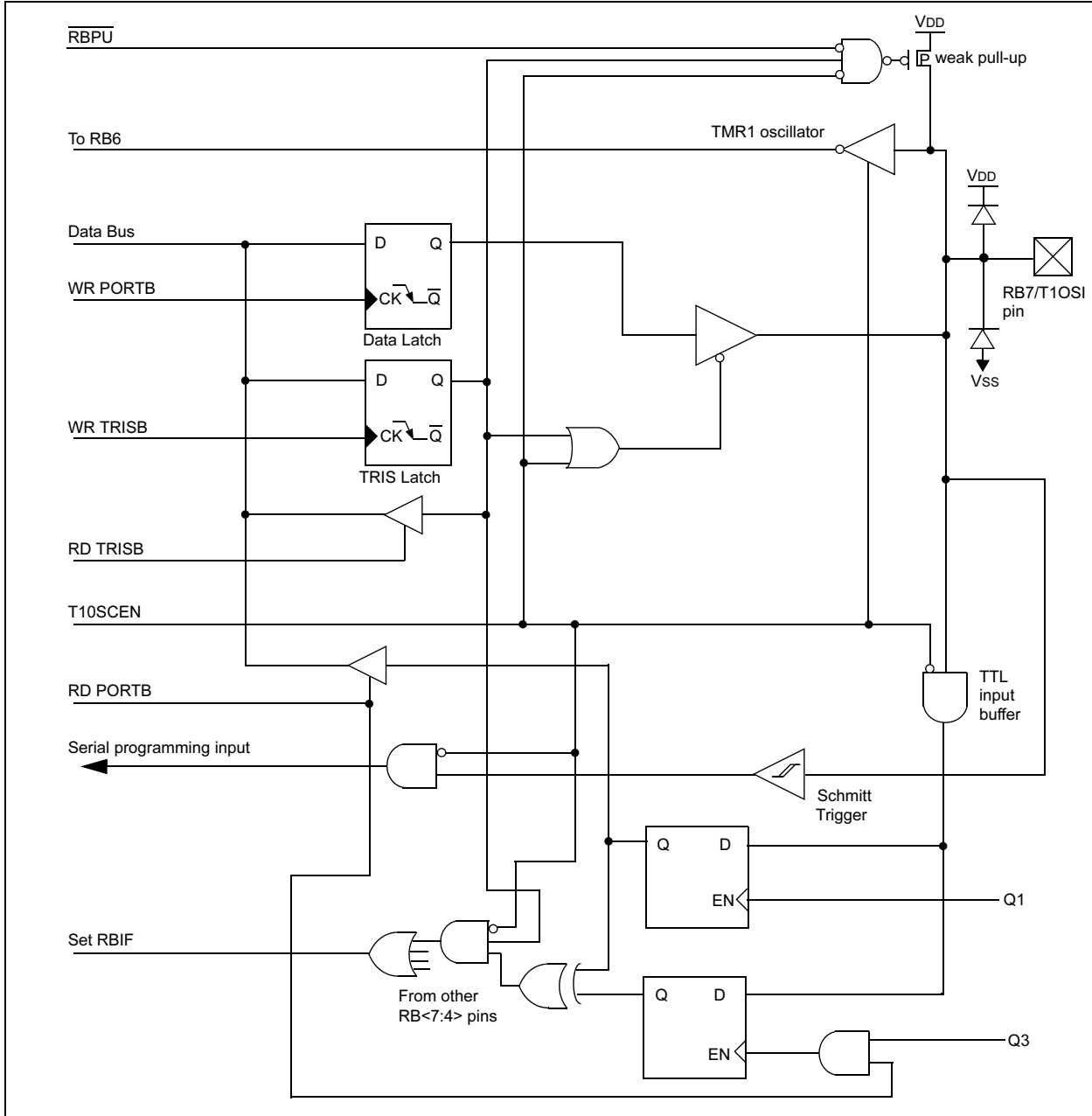
# PIC16F627A/628A/648A

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



# PIC16F627A/628A/648A

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





# PIC16F627A/628A/648A

## 5.3 I/O Programming Considerations

### 5.3.1 BI-DIRECTIONAL I/O PORTS

Any instruction that writes, operates internally as a read followed by a write operation. The `BCF` and `BSF` instructions, for example, read the register into the CPU, execute the bit operation and write the result back to the register. Caution must be used when these instructions are applied to a port with both inputs and outputs defined. For example, a `BSF` operation on bit5 of `PORTB` will cause all eight bits of `PORTB` to be read into the CPU. Then the `BSF` operation takes place on bit5 and `PORTB` is written to the output latches. If another bit of `PORTB` is used as a bi-directional I/O pin (e.g., bit0) and is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and rewritten to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the Input mode, no problem occurs. However, if bit0 is switched into Output mode later on, the 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`, `BSE`, 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`, `BSE`, 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

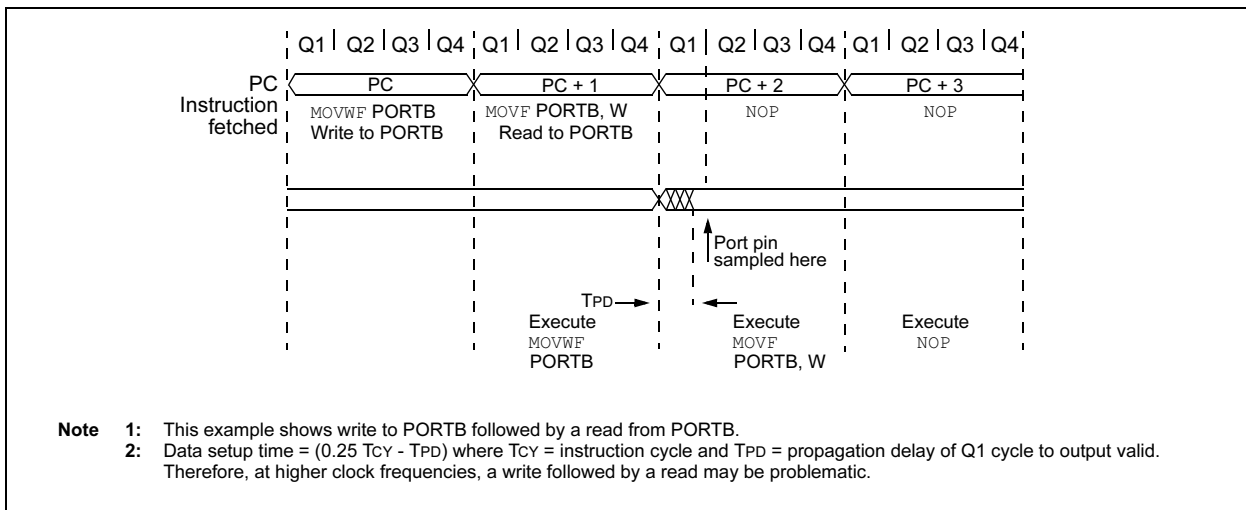
```

;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 latchPORT Pins
;                          -----
BCF STATUS, RP0           ;
BCF PORTB, 7              ;01pp 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
- Read/Write capabilities
- 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. Additional information is available in the PICmicro™ Mid-Range MCU Family Reference Manual, DS33023.

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

Counter mode is selected by setting the T0CS bit. In this mode the TMR0 register value 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 T0IF bit. The interrupt can be masked by clearing the T0IE bit (INTCON<5>). The T0IF 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.

### 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 6-1). 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. See Table 17-9.

# PIC16F627A/628A/648A

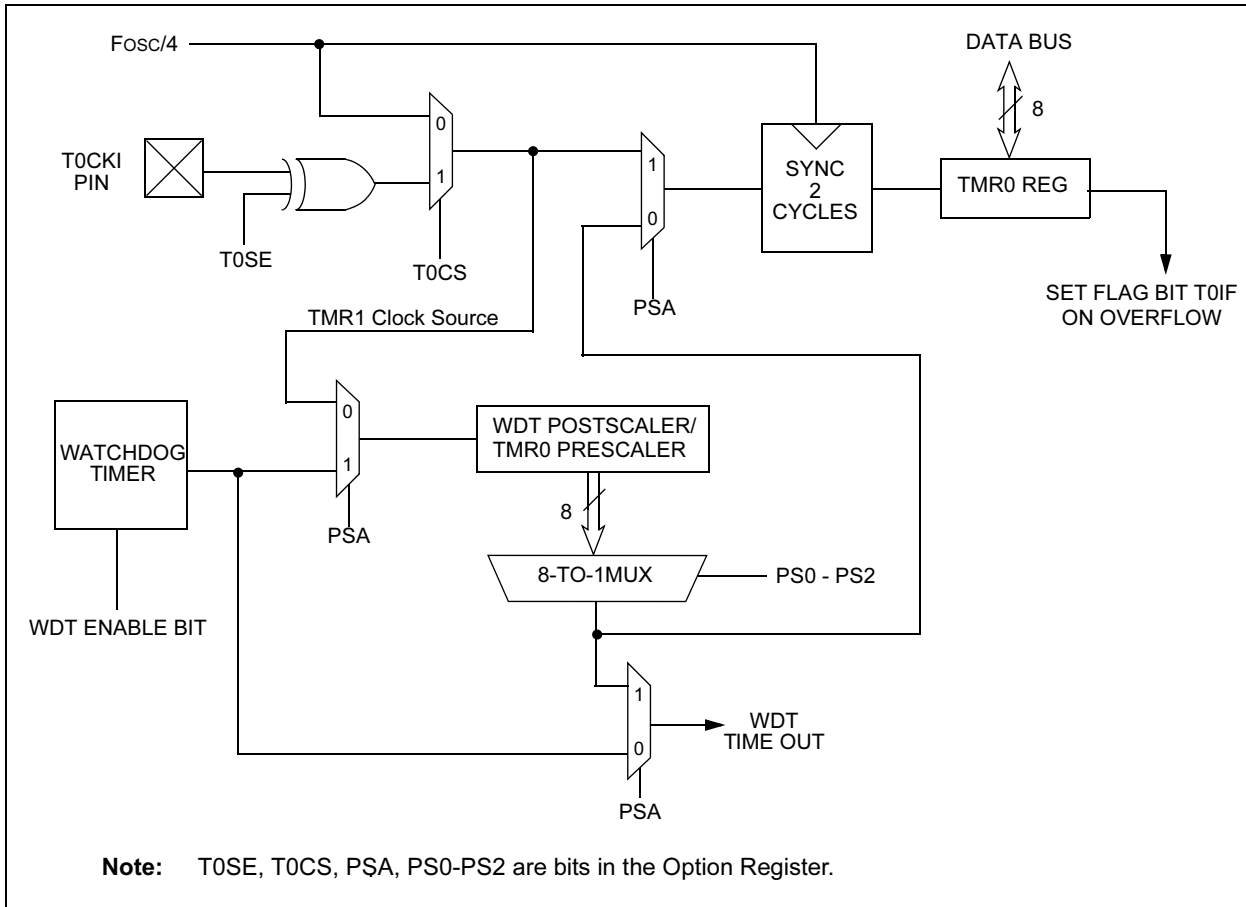
## 6.3 Timer0 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer. A prescaler assignment for the Timer0 module means that there is no postscaler 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., CLRWF 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 6-1: BLOCK DIAGRAM OF THE TIMER0/WDT**





# PIC16F627A/628A/648A

## 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). Use the instruction sequences shown in Example 6-1 when changing the prescaler assignment from Timer0 to WDT, to avoid an unintended device RESET.

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

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

To change prescaler from the WDT to the Timer0 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, 101h	TMR0	Timer0 module register								xxxx xxxx	uuuu uuuu
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	0000 000u
81h, 181h	OPTION <sup>(2)</sup>	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111

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

- Note 1:** Shaded bits are not used by Timer0 module.  
**Note 2:** Option is referred by OPTION\_REG in MPLAB®.

# PIC16F627A/628A/648A

## 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 Timer1 Interrupt, if enabled, is generated on overflow of the TMR1 register pair which latches the interrupt flag bit TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing the Timer1 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, the TMR1 register pair value 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 9.0). Register 7-1 shows the Timer1 control register.

For the PIC16F627A/628A/648A, 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	T1SYNC	TMR1CS	TMR1ON	
bit 7								bit 0

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<sup>(1)</sup>

bit 2 **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 = Disables Timer1  
 0 = Stops Timer1

**Note 1:** The oscillator inverter and feedback resistor are turned off to eliminate power drain.

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

## 7.1 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 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 TMR1 register pair value 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 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, the TMR1 register pair value will not increment even if the external clock is present, since the synchronization circuit is shut off. The prescaler however will continue to increment.

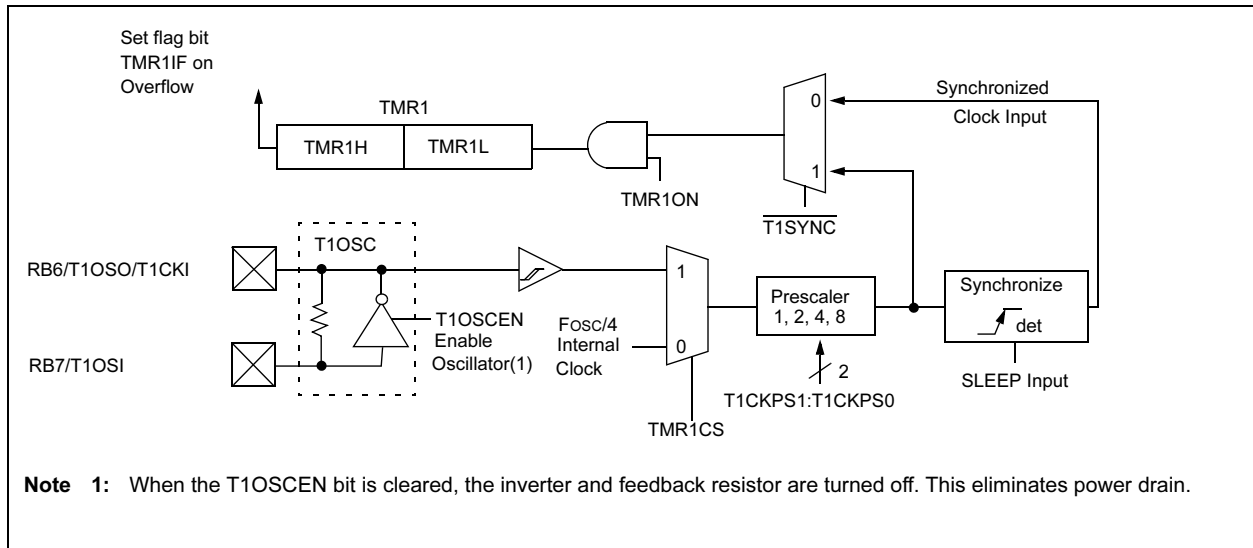
## 7.2.1 EXTERNAL CLOCK INPUT TIMING FOR SYNCHRONIZED COUNTER MODE

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 the TMR1 register pair value 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 45, 46, and 47.

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



# PIC16F627A/628A/648A

## 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).

**Note:** In Asynchronous Counter mode, Timer1 cannot 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 and low time requirements. Refer to Table 17-9 in the Electrical Specifications Section, timing parameters 45, 46, and 47.

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

Reading the TMR1H or TMR1L register while the timer is running, from an external asynchronous clock, will produce 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
MOVF   TMR1H, W   ;Read high byte
MOVWF  TMPH      ;
MOVF   TMR1L, W   ;Read low byte
MOVWF  TMPL      ;
MOVF   TMR1H, W   ;Read high byte
SUBWF  TMPH, W    ;Sub 1st read with
                        ;2nd read
BTFSC  STATUS,Z   ;Is result = 0
GOTO   CONTINUE   ;Good 16-bit read
;
; TMR1L may have rolled over between the
; read of the high and low bytes. Reading
; the high and low bytes now will read a good
; value.
;
MOVF   TMR1H, W   ;Read high byte
MOVWF  TMPH      ;
MOVF   TMR1L, W   ;Read low byte
MOVWF  TMPL      ;
; Re-enable the Interrupts (if required)
CONTINUE      ;Continue with your
                        ;code
```

# PIC16F627A/628A/648A

## 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>). It will continue to run during SLEEP. It is primarily intended for a 32.768 kHz watch crystal. Table 7-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper oscillator start-up.

**TABLE 7-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR**

Freq	C1	C2
32.768 kHz	15 pF	15 pF
<b>These values are for design guidance only. Consult AN826 (DS00826) for further information on Crystal/Capacitor Selection.</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 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	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.

# PIC16F627A/628A/648A

## 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. The TMR2 register value increments from 00h until it matches the PR2 register value and then resets to 00h on the next increment cycle. The PR2 register is a readable and writable register. The PR2 register is initialized to FFh upon RESET.

The match output of Timer2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a Timer2 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:

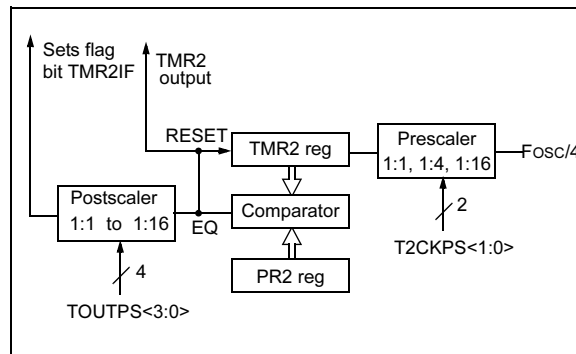
- a write to the TMR2 register
- a write to the T2CON register
- any device RESET (Power-on Reset,  $\overline{MCLR}$  Reset, Watchdog Timer Reset, or Brown-out Reset)

The TMR2 register is not cleared when T2CON is written.

## 8.2 TMR2 Output

The TMR2 output (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



# PIC16F627A/628A/648A

## 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
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

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

bit 6-3 **TOUTPS3:TOUTPS0:** Timer2 Output Postscale Select bits

0000 = 1:1 Postscale Value

0001 = 1:2 Postscale Value

•

•

•

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 = 1:1 Prescaler Value

01 = 1:4 Prescaler Value

1x = 1:16 Prescaler Value

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

**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	TOIF	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.

# PIC16F627A/628A/648A

---

---

NOTES:



## 9.0 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 9-1 shows the timer resources of the CCP module modes.

TABLE 9-1: CCP MODE - TIMER RESOURCE

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

### 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).

### REGISTER 9-1: 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	
bit 7								bit 0

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

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

# PIC16F627A/628A/648A

## 9.1 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

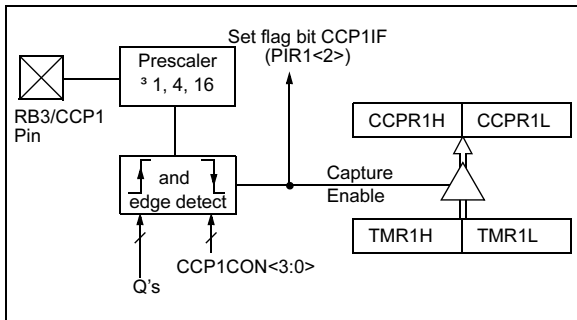
An event is selected by control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit CCP1IF (PIR1<2>) is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value will be lost.

### 9.1.1 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 9-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**



### 9.1.2 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.

### 9.1.3 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.

### 9.1.4 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 9-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the "false" interrupt.

**EXAMPLE 9-1: CHANGING BETWEEN CAPTURE PRESCALERS**

```
CLRF    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
```

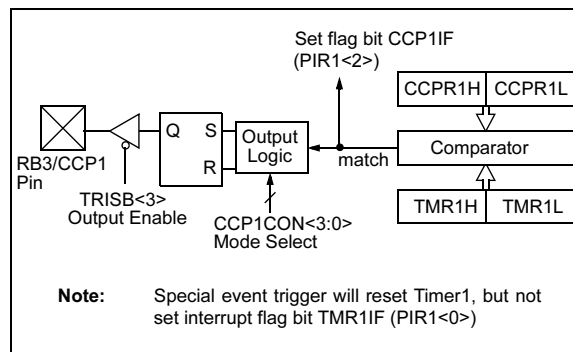
## 9.2 Compare Mode

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 9-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



# PIC16F627A/628A/648A

## 9.2.1 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.

## 9.2.2 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.

## 9.2.3 SOFTWARE INTERRUPT MODE

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

## 9.2.4 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 9-2: 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	TOIE	INTE	RBIE	TOIF	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
86h, 186h	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.

# PIC16F627A/628A/648A

## 9.3 PWM Mode

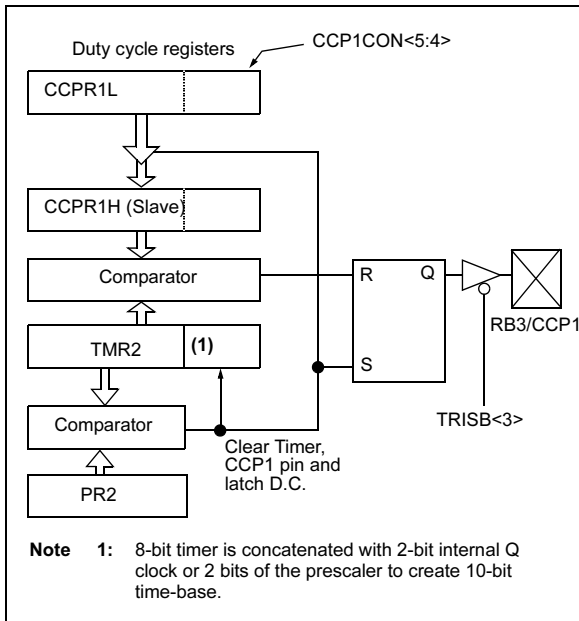
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTB data latch, the TRISB<3> bit must be cleared to make the CCP1 pin an output.

**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 9-3 shows a simplified block diagram of the CCP module in PWM mode.

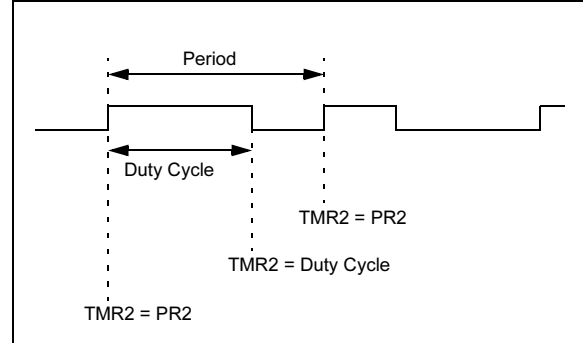
For a step by step procedure on how to set up the CCP module for PWM operation, see Section 9.3.3.

**FIGURE 9-3: SIMPLIFIED PWM BLOCK DIAGRAM**



A PWM output (Figure 9-4) 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 (frequency = 1/period).

**FIGURE 9-4: PWM OUTPUT**



### 9.3.1 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.

# PIC16F627A/628A/648A

## 9.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCP1R1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available: the CCP1R1L contains the eight MSBs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCP1R1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCP1R1L:CCP1CON<5:4>}) \cdot T_{osc} \cdot \text{TMR2 prescale value}$$

CCP1R1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCP1R1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCP1R1H is a read-only register.

The CCP1R1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitch less PWM operation.

When the CCP1R1H 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:

$$\text{PWM Resolution} = \frac{\log\left(\frac{F_{osc}}{F_{pwm} \times \text{TMR2 Prescaler}}\right)}{\log(2)} \text{ bits}$$

**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).

## 9.3.3 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 CCP1R1L 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 9-3: 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	6.5

**TABLE 9-4: 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
86h, 186h	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	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	uuuu uuuu
15h	CCP1R1L	Capture/Compare/PWM register1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCP1R1H	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.

# PIC16F627A/628A/648A

---

---

NOTES:

# PIC16F627A/628A/648A

## 10.0 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 10-1, controls the comparator input and output multiplexers. A block diagram of the comparator is shown in Figure 10-1.

**REGISTER 10-1: 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
bit 7				bit 0			

- bit 7      **C2OUT:** Comparator 2 Output  
When C2INV = 0:  
 1 = C2 VIN+ > C2 VIN-  
 0 = C2 VIN+ < C2 VIN-  
  
When C2INV = 1:  
 1 = C2 VIN+ < C2 VIN-  
 0 = 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:  
 1 = C1 VIN+ < C1 VIN-  
 0 = 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 10-1 shows the Comparator modes and CM2:CM0 bit settings

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

# PIC16F627A/628A/648A

## 10.1 Comparator Configuration

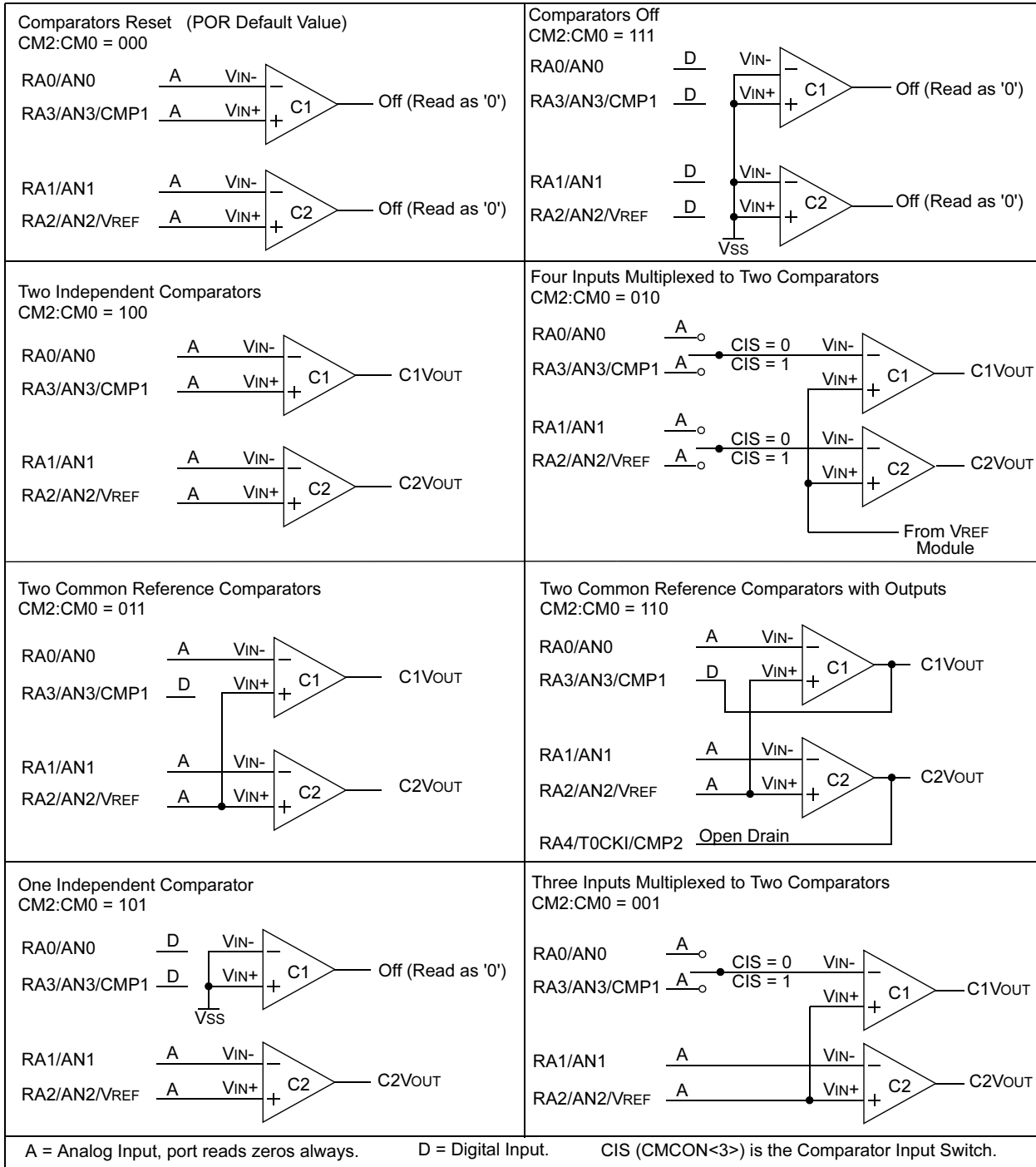
There are eight modes of operation for the comparators. The CMCON register is used to select the mode. Figure 10-1 shows the eight possible modes. The TRISA register controls the data direction of the comparator pins for each mode. If the Compara-

tor mode is changed, the comparator output level may not be valid for the specified mode change delay shown in Table 17-2.

**Note 1:** Comparator interrupts should be disabled during a Comparator mode change, otherwise a false interrupt may occur.

**Note 2:** Comparators can have an inverted output. See Figure 10-3.

**FIGURE 10-1: COMPARATOR I/O OPERATING MODES**





The code example in Example 10-1 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 10-1: 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 Bank 1
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 DELAY10 ;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
    
```

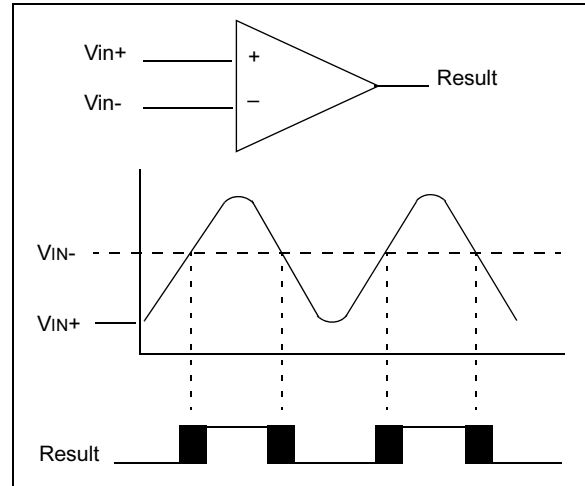
## 10.2 Comparator Operation

A single comparator is shown in Figure 10-2 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 10-2 represent the uncertainty due to input offsets and response time. See Table 17-2 for Common Mode Voltage.

## 10.3 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 10-2).

FIGURE 10-2: SINGLE COMPARATOR



### 10.3.1 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).

### 10.3.2 INTERNAL REFERENCE SIGNAL

The Comparator module also allows the selection of an internally generated voltage reference for the comparators. Section 11.0, Voltage Reference Manual, 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 10-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

## 10.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output is 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 17-2).



## 10.6 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 write or read 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.

## 10.7 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.

## 10.8 Effects of a RESET

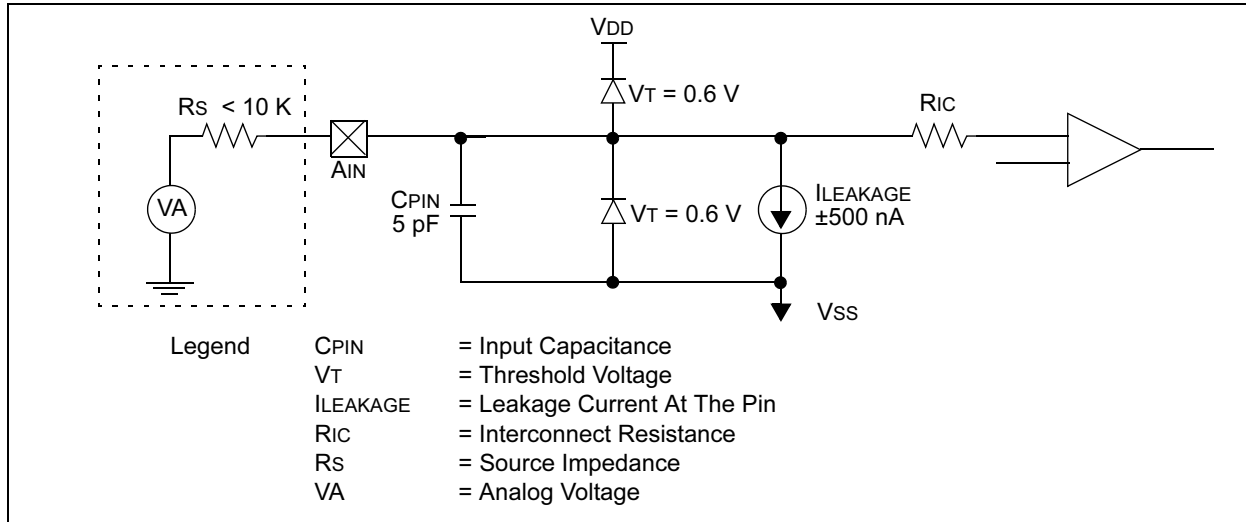
A device RESET forces the CMCON register to its RESET state. This forces the Comparator module to be in the comparator RESET mode, CM2:CM0 = 000. This ensures that all potential inputs are analog inputs. Device current is minimized when analog inputs are present at RESET time. The comparators will be powered-down during the RESET interval.

## 10.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 10-4. 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.

# PIC16F627A/628A/648A

**FIGURE 10-4: ANALOG INPUT MODE**



**TABLE 10-1: 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
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	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111

Legend: x = Unknown, u = Unchanged, - = Unimplemented, read as '0'

## 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 11-1.

### 11.1 Voltage Reference Configuration

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:

if VRR = 1:

$$V_{REF} = \frac{VR_{<3:0>}}{24} \times V_{DD}$$

if VRR = 0:

$$V_{REF} = \left( V_{DD} \times \frac{1}{4} \right) + \frac{VR_{<3:0>}}{32} \times V_{DD}$$

The setting time of the Voltage Reference must be considered when changing the VREF output (Table 17-3). Example 11-1 demonstrates how Voltage Reference is configured for an output voltage of 1.25V with VDD = 5.0V.

#### REGISTER 11-1: VRCON REGISTER (ADDRESS: 9Fh)

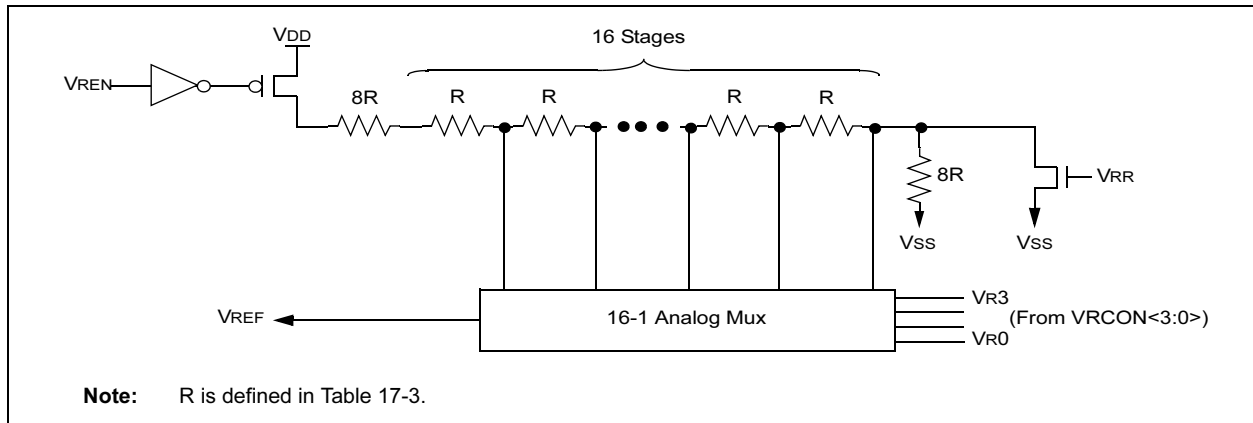
R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	
VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	
			bit 7					bit 0

- bit 7 **VREN:** VREF Enable  
1 = VREF circuit powered on  
0 = VREF circuit powered down, no IDD drain
- bit 6 **VROE:** VREF Output Enable  
1 = VREF is output on RA2 pin  
0 = VREF is disconnected from RA2 pin
- bit 5 **VRR:** VREF Range selection  
1 = Low Range  
0 = High Range
- bit 4 **Unimplemented:** Read as '0'
- bit 3-0 **VR<3:0>:** VREF value selection  $0 \leq VR [3:0] \leq 15$   
When VRR = 1:  $V_{REF} = (VR_{<3:0>} / 24) * V_{DD}$   
When VRR = 0:  $V_{REF} = 1/4 * V_{DD} + (VR_{<3:0>} / 32) * V_{DD}$

Legend:

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

FIGURE 11-1: VOLTAGE REFERENCE BLOCK DIAGRAM



# PIC16F627A/628A/648A

## 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  $V_{SS}$  to  $V_{DD}$  cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 11-1) keep  $V_{REF}$  from approaching  $V_{SS}$  or  $V_{DD}$ . The Voltage Reference is  $V_{DD}$  derived and therefore, the  $V_{REF}$  output changes with fluctuations in  $V_{DD}$ . The tested absolute accuracy of the Voltage Reference can be found in Table 17-3.

## 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  $V_{REF}$  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  $V_{REF}$  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  $V_{REF}$ . Figure 11-2 shows an example buffering technique.

FIGURE 11-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE

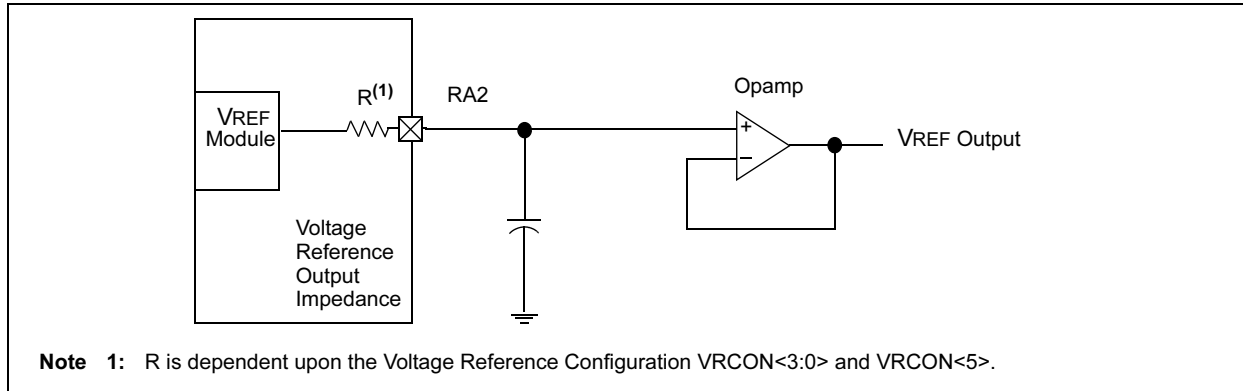


TABLE 11-1: REGISTERS ASSOCIATED WITH VOLTAGE REFERENCE

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	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111

**Note:** — = Unimplemented, read as '0'.

# PIC16F627A/628A/648A

## 12.0 UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART) MODULE

The Universal Synchronous Asynchronous Receiver Transmitter (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 shows the Transmit Status and Control Register (TXSTA) and Register 12-2 shows the Receive Status and Control Register (RCSTA).

### 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
						bit 7	bit 0

- bit 7 **CSRC:** Clock Source Select bit  
Asynchronous mode  
 Don't care  
Synchronous mode  
 1 = Master mode (Clock generated internally from BRG)  
 0 = Slave mode (Clock from external source)
- bit 6 **TX9:** 9-bit Transmit Enable bit  
 1 = Selects 9-bit transmission  
 0 = Selects 8-bit transmission
- bit 5 **TXEN:** Transmit Enable bit<sup>(1)</sup>  
 1 = Transmit enabled  
 0 = Transmit disabled
- bit 4 **SYNC:** USART Mode Select bit  
 1 = Synchronous mode  
 0 = Asynchronous mode
- bit 3 **Unimplemented:** Read as '0'
- bit 2 **BRGH:** High Baud Rate Select bit  
Asynchronous mode  
 1 = High speed  
 0 = Low speed  
Synchronous mode  
 Unused in this mode
- bit 1 **TRMT:** Transmit Shift Register STATUS bit  
 1 = TSR empty  
 0 = TSR full
- bit 0 **TX9D:** 9th bit of transmit data. Can be parity bit.  
**Note:** SREN/CREN overrides TXEN in SYNC mode.

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

# PIC16F627A/628A/648A

## 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
bit 7							bit 0

- bit 7 **SPEN:** Serial Port Enable bit  
(Configures RB1/RX/DT and RB2/TX/CK pins as serial port pins when bits TRISB<2:1> 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)

**Legend:**

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



## 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

$$\text{Desired Baud Rate} = \frac{F_{osc}}{64(x+1)}$$

$$9600 = \frac{16000000}{64(x+1)}$$

$$x = 25.042$$

$$\text{Calculated Baud Rate} = \frac{16000000}{64(25+1)} = 9615$$

$$\text{Error} = \frac{(\text{Calculated Baud Rate} - \text{Desired Baud Rate})}{\text{Desired Baud Rate}}$$

$$= \frac{9615 - 9600}{9600} = 0.16\%$$

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

Legend: 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 000x	0000 000x
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.

# PIC16F627A/628A/648A

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

BAUD RATE (K)	Fosc = 20 MHz			16 MHz			10 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	NA	—	—
1.2	NA	—	—	NA	—	—	NA	—	—
2.4	NA	—	—	NA	—	—	NA	—	—
9.6	NA	—	—	NA	—	—	9.766	+1.73%	255
19.2	19.53	+1.73%	255	19.23	+0.16%	207	19.23	+0.16%	129
76.8	76.92	+0.16%	64	76.92	+0.16%	51	75.76	-1.36%	32
96	96.15	+0.16%	51	95.24	-0.79%	41	96.15	+0.16%	25
300	294.1	-1.96	16	307.69	+2.56%	12	312.5	+4.17%	7
500	500	0	9	500	0	7	500	0	4
HIGH	5000	—	0	4000	—	0	2500	—	0
LOW	19.53	—	255	15.625	—	255	9.766	—	255

BAUD RATE (K)	Fosc = 7.15909 MHz			5.0688 MHz			4 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	NA	—	—
1.2	NA	—	—	NA	—	—	NA	—	—
2.4	NA	—	—	NA	—	—	NA	—	—
9.6	9.622	+0.23%	185	9.6	0	131	9.615	+0.16%	103
19.2	19.24	+0.23%	92	19.2	0	65	19.231	+0.16%	51
76.8	77.82	+1.32	22	79.2	+3.13%	15	75.923	+0.16%	12
96	94.20	-1.88	18	97.48	+1.54%	12	1000	+4.17%	9
300	298.3	-0.57	5	316.8	5.60%	3	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	1789.8	—	0	1267	—	0	100	—	0
LOW	6.991	—	255	4.950	—	255	3.906	—	255

BAUD RATE (K)	Fosc = 3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	0.303	+1.14%	26
1.2	NA	—	—	1.202	+0.16%	207	1.170	-2.48%	6
2.4	NA	—	—	2.404	+0.16%	103	NA	—	—
9.6	9.622	+0.23%	92	9.615	+0.16%	25	NA	—	—
19.2	19.04	-0.83%	46	19.24	+0.16%	12	NA	—	—
76.8	74.57	-2.90%	11	83.34	+8.51%	2	NA	—	—
96	99.43	+3.57%	8	NA	—	—	NA	—	—
300	298.3	0.57%	2	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	—	—	—
HIGH	894.9	—	0	250	—	0	8.192	—	0
LOW	3.496	—	255	0.9766	—	255	0.032	—	255

# PIC16F627A/628A/648A

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

BAUD RATE (K)	Fosc = 20 MHz			16 MHz			10 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	NA	—	—
1.2	1.221	+1.73%	255	1.202	+0.16%	207	1.202	+0.16%	129
2.4	2.404	+0.16%	129	2.404	+0.16%	103	2.404	+0.16%	64
9.6	9.469	-1.36%	32	9.615	+0.16%	25	9.766	+1.73%	15
19.2	19.53	+1.73%	15	19.23	+0.16%	12	19.53	+1.73V	7
76.8	78.13	+1.73%	3	83.33	+8.51%	2	78.13	+1.73%	1
96	104.2	+8.51%	2	NA	—	—	NA	—	—
300	312.5	+4.17%	0	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	312.5	—	0	250	—	0	156.3	—	0
LOW	1.221	—	255	0.977	—	255	0.6104	—	255

BAUD RATE (K)	Fosc = 7.15909 MHz			5.0688 MHz			4 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	NA	—	—	0.31	+3.13%	255	0.3005	-0.17%	207
1.2	1.203	+0.23%	92	1.2	0	65	1.202	+1.67%	51
2.4	2.380	-0.83%	46	2.4	0	32	2.404	+1.67%	25
9.6	9.322	-2.90%	11	9.9	+3.13%	7	NA	—	—
19.2	18.64	-2.90%	5	19.8	+3.13%	3	NA	—	—
76.8	NA	—	—	79.2	+3.13%	0	NA	—	—
96	NA	—	—	NA	—	—	NA	—	—
300	NA	—	—	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	111.9	—	0	79.2	—	0	62.500	—	0
LOW	0.437	—	255	0.3094	—	255	3.906	—	255

BAUD RATE (K)	Fosc = 3.579545 MHz			1 MHz			32.768 kHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
0.3	0.301	+0.23%	185	0.300	+0.16%	51	0.256	-14.67%	1
1.2	1.190	-0.83%	46	1.202	+0.16%	12	NA	—	—
2.4	2.432	+1.32%	22	2.232	-6.99%	6	NA	—	—
9.6	9.322	-2.90%	5	NA	—	—	NA	—	—
19.2	18.64	-2.90%	2	NA	—	—	NA	—	—
76.8	NA	—	—	NA	—	—	NA	—	—
96	NA	—	—	NA	—	—	NA	—	—
300	NA	—	—	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	55.93	—	0	15.63	—	0	0.512	—	0
LOW	0.2185	—	255	0.0610	—	255	0.0020	—	255

# PIC16F627A/628A/648A

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

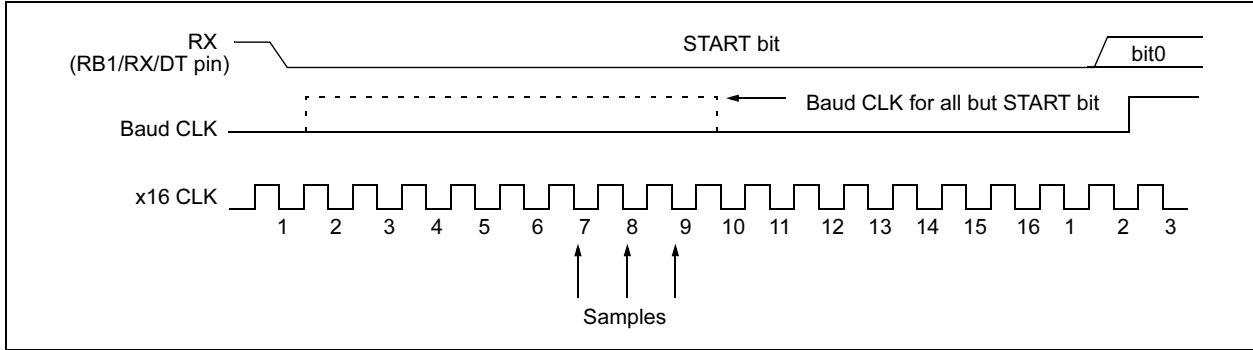
BAUD RATE (K)	Fosc = 20 MHz			16 MHz			10 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
9600	9.615	+0.16%	129	9.615	+0.16%	103	9.615	+0.16%	64
19200	19.230	+0.16%	64	19.230	+0.16%	51	18.939	-1.36%	32
38400	37.878	-1.36%	32	38.461	+0.16%	25	39.062	+1.7%	15
57600	56.818	-1.36%	21	58.823	+2.12%	16	56.818	-1.36%	10
115200	113.636	-1.36%	10	111.111	-3.55%	8	125	+8.51%	4
250000	250	0	4	250	0	3	NA	—	—
625000	625	0	1	NA	—	—	625	0	0
1250000	1250	0	0	NA	—	—	NA	—	—

BAUD RATE (K)	Fosc = 7.16 MHz			5.068 MHz			4 MHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
9600	9.520	-0.83%	46	9598.485	0.016%	32	9615.385	0.160%	25
19200	19.454	+1.32%	22	18632.35	-2.956%	16	19230.77	0.160%	12
38400	37.286	-2.90%	11	39593.75	3.109%	7	35714.29	-6.994%	6
57600	55.930	-2.90%	7	52791.67	-8.348%	5	62500	8.507%	3
115200	111.860	-2.90%	3	105583.3	-8.348%	2	125000	8.507%	1
250000	NA	—	—	316750	26.700%	0	250000	0.000%	0
625000	NA	—	—	NA	—	—	NA	—	—
1250000	NA	—	—	NA	—	—	NA	—	—

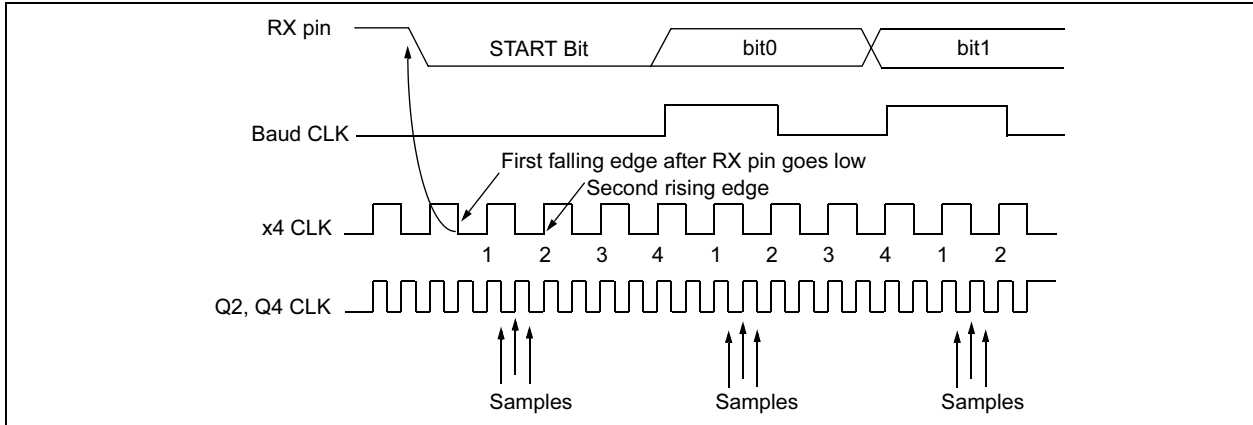
BAUD RATE (K)	Fosc = 3.579 MHz			1 MHz			32.768 kHz		
	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)	KBAUD	ERROR	SPBRG value (decimal)
9600	9725.543	1.308%	22	8.928	-6.994%	6	NA	NA	NA
19200	18640.63	-2.913%	11	20833.3	8.507%	2	NA	NA	NA
38400	37281.25	-2.913%	5	31250	-18.620%	1	NA	NA	NA
57600	55921.88	-2.913%	3	62500	+8.507	0	NA	NA	NA
115200	111243.8	-2.913%	1	NA	—	—	NA	NA	NA
250000	223687.5	-10.525%	0	NA	—	—	NA	NA	NA
625000	NA	—	—	NA	—	—	NA	NA	NA
1250000	NA	—	—	NA	—	—	NA	NA	NA

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).

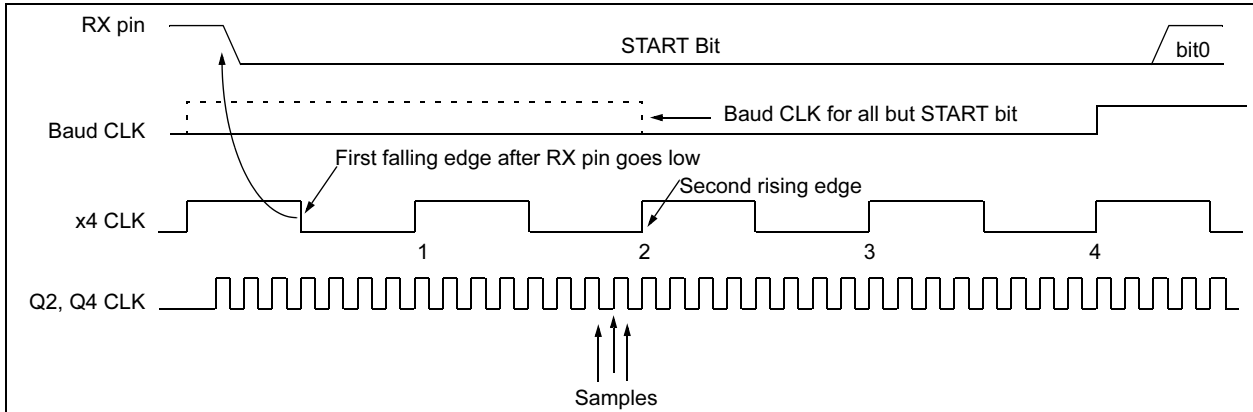
**FIGURE 12-1: RX PIN SAMPLING SCHEME, BRGH = 0**



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

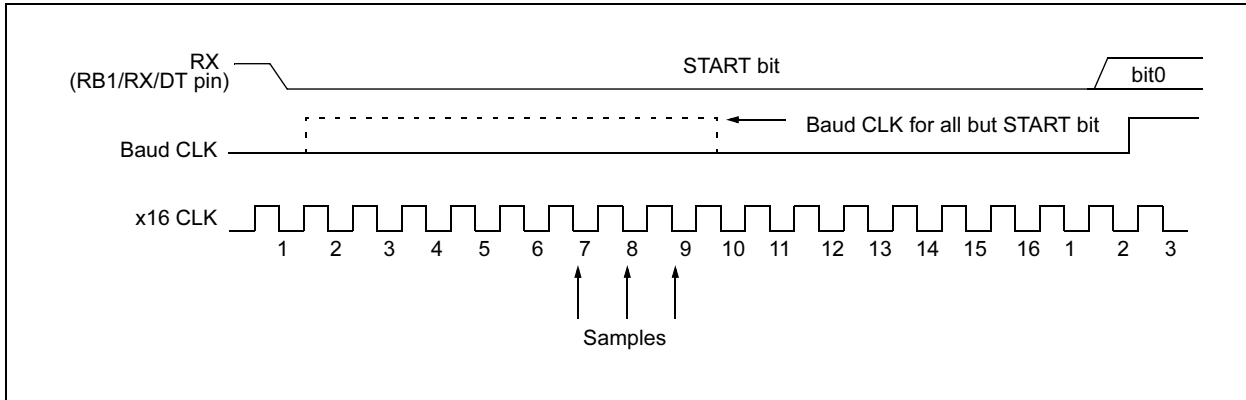


**FIGURE 12-3: RX PIN SAMPLING SCHEME, BRGH = 1**



# PIC16F627A/628A/648A

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



## 12.2 USART Asynchronous Mode

In this mode, the USART uses standard non-return-to-zero (NRZ) format (one START bit, eight or nine data bits and one STOP bit). The most common data format is 8-bit. A dedicated 8-bit baud rate generator is used to derive 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 soft-

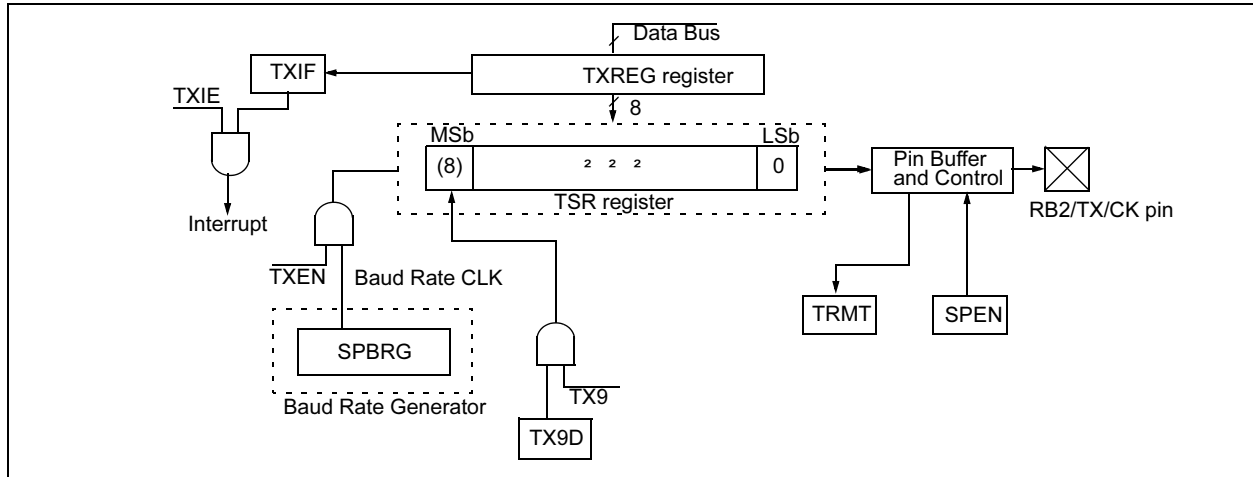
ware. It will RESET only when new data is loaded into the TXREG register. While flag bit TXIF indicated the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. STATUS bit TRMT is a read only bit which is set when the TSR register is empty. No interrupt logic 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.
- 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.

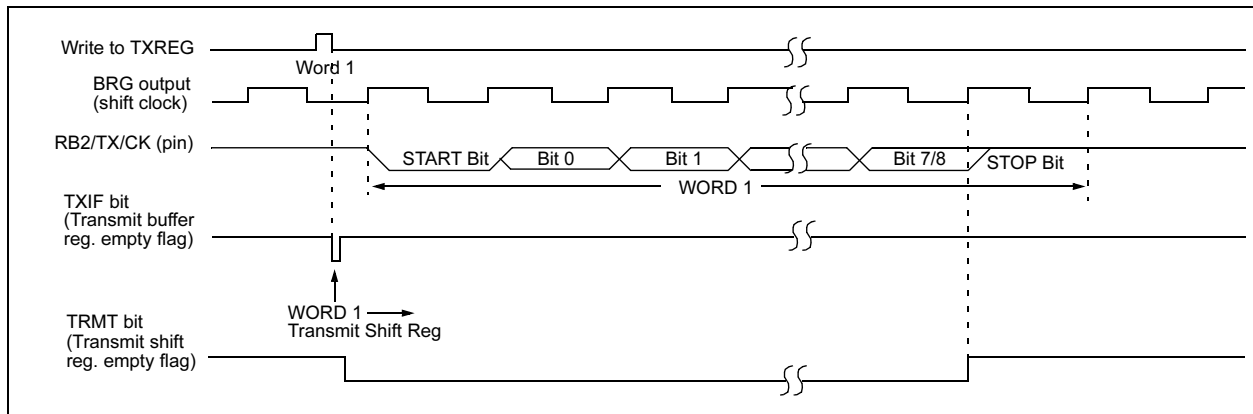
**FIGURE 12-5: USART TRANSMIT BLOCK DIAGRAM**



Follow these steps when setting up an Asynchronous Transmission:

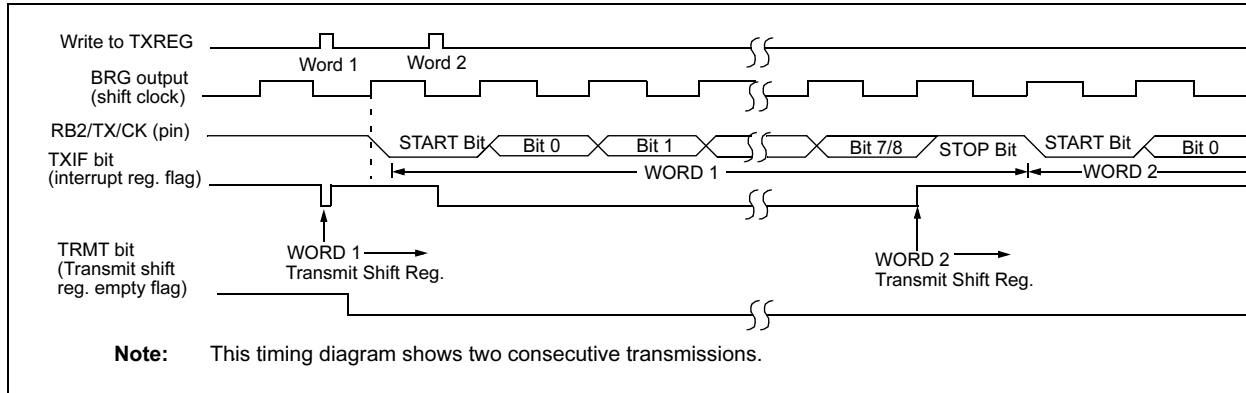
1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
2. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH. (Section 12.1)
3. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
4. If interrupts are desired, then set enable bit TXIE.
5. If 9-bit transmission is desired, then set transmit bit TX9.
6. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
8. Load data to the TXREG register (starts transmission).

**FIGURE 12-6: ASYNCHRONOUS TRANSMISSION**



# PIC16F627A/628A/648A

**FIGURE 12-7: ASYNCHRONOUS 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 000x	0000 000x
19h	TXREG	USART Transmit data 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.



## 12.2.2 USART ASYNCHRONOUS RECEIVER

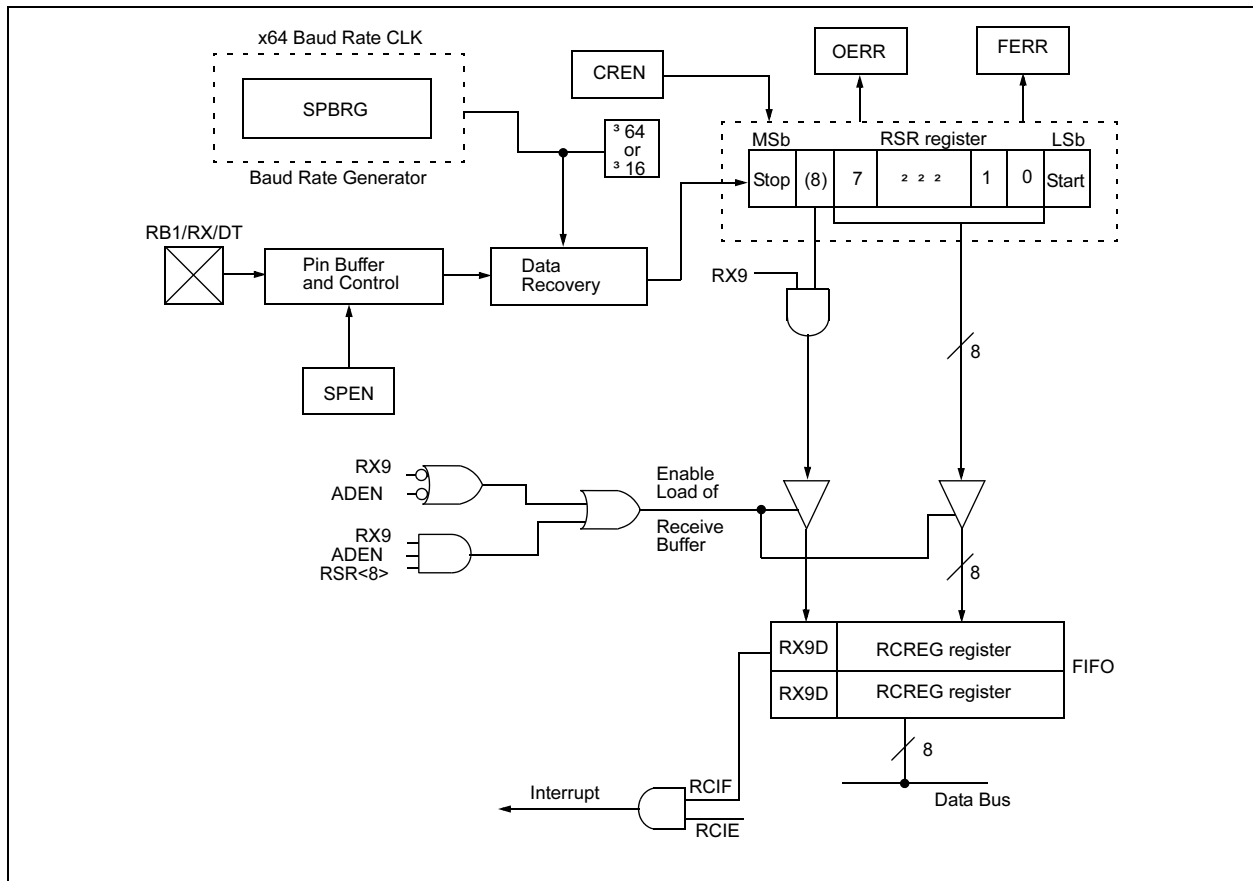
The receiver block diagram is shown in Figure 12-8. The data is received on the RB1/RX/DT pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC.

When 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

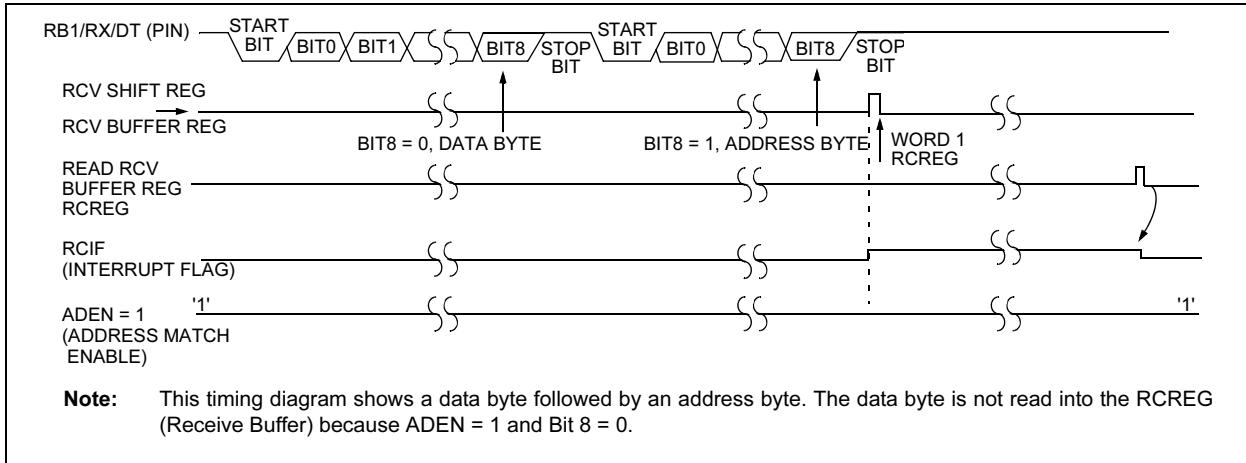
double buffered register, (i.e., it is a two deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte begin shifting to the RSR register. On the detection of the STOP bit of the third byte, if the RCREG register is still full then overrun error 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**

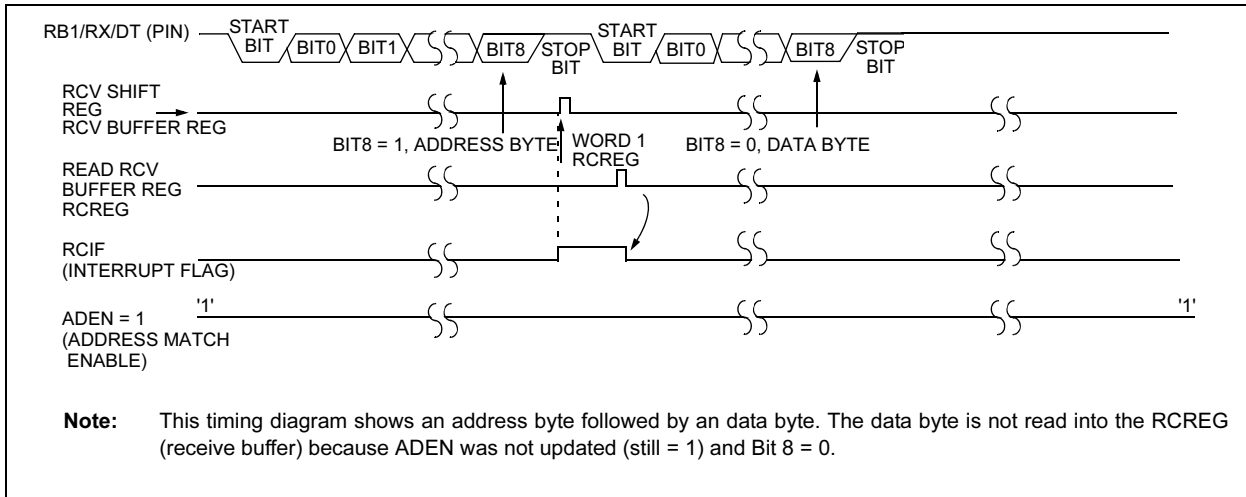


# PIC16F627A/628A/648A

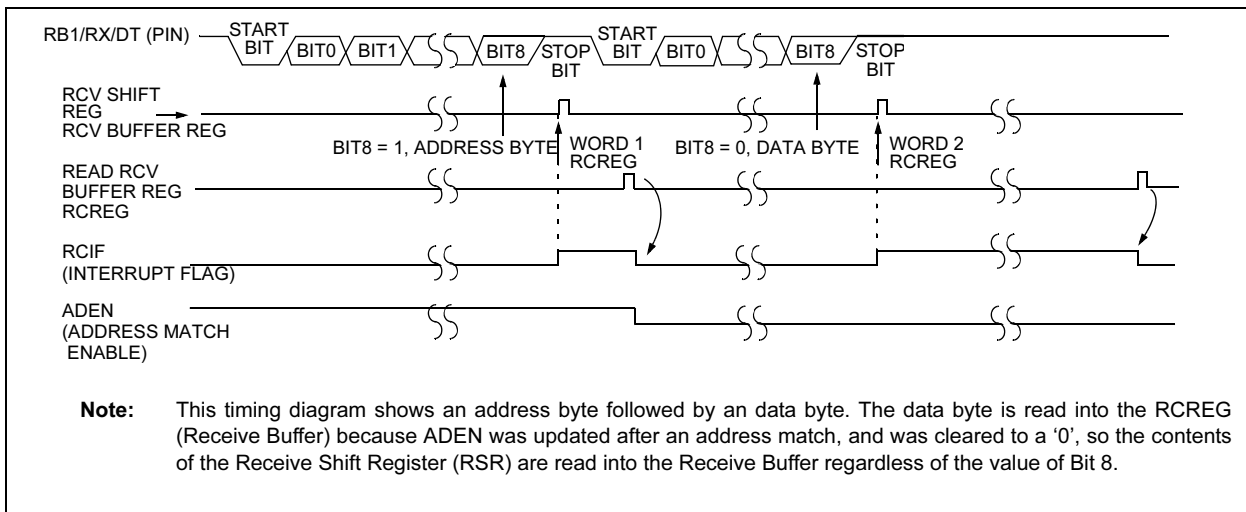
**FIGURE 12-9: ASYNCHRONOUS RECEPTION WITH ADDRESS DETECT**



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



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



# PIC16F627A/628A/648A

Follow these steps when setting up an Asynchronous Reception:

1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
2. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH. (Section 12.1).
3. Enable the asynchronous serial port by clearing bit SYNC, and setting bit SPEN.
4. If interrupts are desired, then set enable bit RCIE.
5. If 9-bit reception is desired, then set bit RX9.
6. Enable the reception by setting bit CREN.
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 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 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 000x	0000 000x
1Ah	RCREG	USART Receive data 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.

# PIC16F627A/628A/648A

## 12.3 USART Address Detect Function

### 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

Follow these steps when setting up Asynchronous Reception with Address Detect Enabled:

1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
2. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH.
3. Enable asynchronous communication by setting or clearing bit SYNC and setting bit SPEN.
4. If interrupts are desired, then set enable bit RCIE.
5. Set bit RX9 to enable 9-bit reception.
6. Set ADEN to enable address detect.
7. Enable the reception by setting enable bit CREN or SREN.
8. Flag bit RCIF will be set when reception is complete, and an interrupt will be generated if enable bit RCIE was set.
9. Read the 8-bit received data by reading the RCREG register to determine if the device is being addressed.
10. If any error occurred, clear the error by clearing enable bit CREN if it was already set.
11. 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.

**TABLE 12-8: 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 000x	0000 000x
1Ah	RCREG	USART Receive data 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.

## 12.4 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 stable 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.

Follow these steps when setting up a Synchronous Master Transmission:

1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
2. Initialize the SPBRG register for the appropriate baud rate (Section 12.1).
3. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
4. If interrupts are desired, then set enable bit TXIE.
5. If 9-bit transmission is desired, then set bit TX9.
6. Enable the transmission by setting bit TXEN.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
8. Start transmission by loading data to the TXREG register.

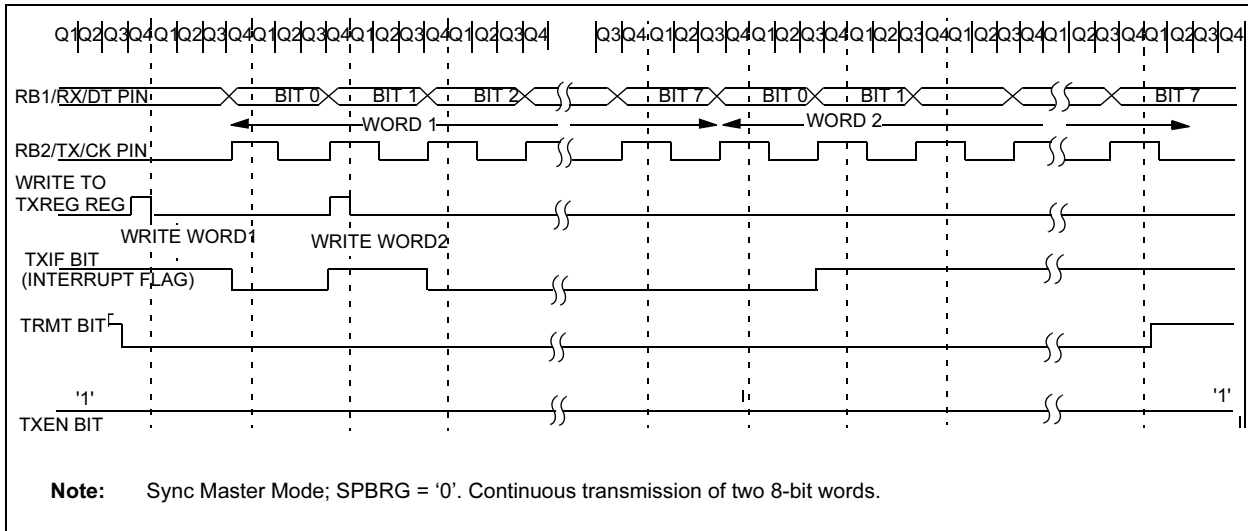
# PIC16F627A/628A/648A

**TABLE 12-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

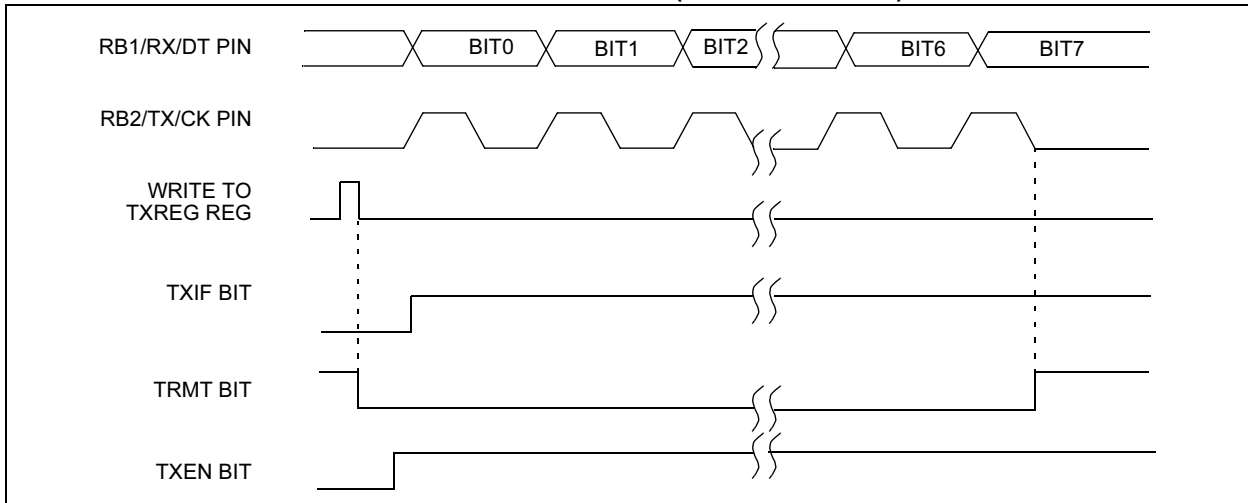
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other RESETS
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 000x	0000 000x
19h	TXREG	USART Transmit data register								0000 0000	0000 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)**



## 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.

Follow these steps when setting up a Synchronous Master Reception:

1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
2. Initialize the SPBRG register for the appropriate baud rate. (Section 12.1)
3. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
4. Ensure bits CREN and SREN are clear.
5. If interrupts are desired, then set enable bit RCIE.
6. If 9-bit reception is desired, then set bit RX9.
7. If a single reception is required, set bit SREN. For continuous reception set bit CREN.
8. Interrupt flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
9. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
10. Read the 8-bit received data by reading the RCREG register.
11. If any error occurred, clear the error by clearing bit CREN.

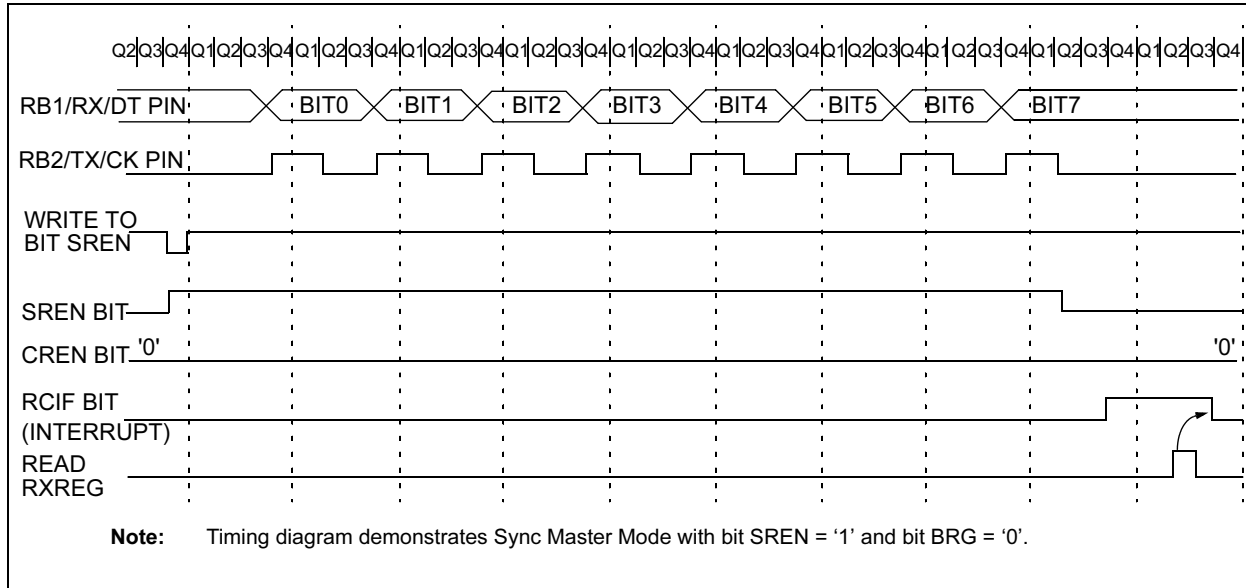
**TABLE 12-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

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 000x	0000 000x
1Ah	RCREG	USART Receive data 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.

# PIC16F627A/628A/648A

**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:

- The first word will immediately transfer to the TSR register and transmit.
- The second word will remain in TXREG register.
- Flag bit TXIF will not be set.
- 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.
- 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).

Follow these steps when setting up a Synchronous Slave Transmission:

- TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.
- Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
- Clear bits CREN and SREN.
- If interrupts are desired, then set enable bit TXIE.
- If 9-bit transmission is desired, then set bit TX9.
- Enable the transmission by setting enable bit TXEN.
- If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
- Start transmission by loading data to the TXREG register.



# PIC16F627A/628A/648A

## 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).

Follow these steps when setting up a Synchronous Slave Reception:

1. TRISB<1> bit needs to be set and TRISB<2> bit cleared in order to configure pins RB2/TX/CK and RB1/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter pins.

2. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. To enable reception, set enable 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 bit CREN.

**TABLE 12-11: 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 000x	0000 000x
19h	TXREG	USART Transmit data 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 12-12: 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 000x	0000 000x
1Ah	RCREG	USART Receive data 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.

# PIC16F627A/628A/648A

---

NOTES:

# PIC16F627A/628A/648A

## 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 (SFRs). 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. PIC16F627A/628A devices have 128 bytes of data EEPROM with an address range from 0h to 7Fh. PIC16F648A device has 256 bytes of data EEPROM with an address range from 0h to FFh.

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 can continue to read and write the data EEPROM memory. A 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: EEDATA REGISTER (ADDRESS: 9Ah)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	
EEDAT7	EEDAT6	EEDAT5	EEDAT4	EEDAT3	EEDAT2	EEDAT1	EEDAT0	
bit 7								bit 0

bit 7-0 **EEDATn**: Byte value to write to or read from Data EEPROM memory location.

#### Legend:

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

### REGISTER 13-2: EEADR REGISTER (ADDRESS: 9Bh)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	
EADR7	EADR6	EADR5	EADR4	EADR3	EADR2	EADR1	EADR0	
bit 7								bit 0

bit 7 **PIC16F627A/628A - Unimplemented Address: Must be set to '0'**  
**PIC16F648A - EEADR**: Set to '1' specifies top 128 locations (128-256) of EEPROM Read/Write Operation

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

#### Legend:

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

# PIC16F627A/628A/648A

## 13.1 EEADR

The PIC16F648A EEADR register addresses 256 bytes of data EEPROM. All eight bits in the register (EEADR<7:0>) are required.

The PIC16F627A/628A EEADR register addresses only the first 128 bytes of data EEPROM so 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.

## 13.2 EECON1 AND EECON2 REGISTERS

EECON1 is the control register with four low order bits physically implemented. The upper-four 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 13-3: EECON1 REGISTER (ADDRESS: 9Ch) DEVICES

U-0	U-0	U-0	U-0	R/W-x	R/W-0	R/S-0	R/S-0
—	—	—	—	WRERR	WREN	WR	RD
bit 7				bit 0			

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 BOR Reset)

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

**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

## 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

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

## 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

```
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

```
BSF    STATUS, RP0 ;Bank 1
MOVF   EEDATA, W
BSF    EECON1, RD ;Read the
                          ;value written
;
;Is the value written (in W reg) and
;read (in EEDATA) the same?
;
SUBWF  EEDATA, W ;
BTFS   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 when enabled, 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.

# PIC16F627A/628A/648A

## 13.7 Using the Data EEPROM

The data EEPROM is a high endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124. If this is not the case, an array refresh must be performed. For this reason, variables

that change infrequently (such as constants, IDs, calibration, etc.) should be stored in FLASH program memory.

A simple data EEPROM refresh routine is shown in Example 13-4.

**Note:** If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124.

### EXAMPLE 13-4: DATA EEPROM REFRESH ROUTINE

```

    clrf    EEADR                ; Start at address 0
    bcf    EECON1,CFGSR        ; Set for memory
    bcf    EECON1,EEPGD        ; Set for Data EEPROM
    bcf    INTCON,GIE          ; Disable interrupts
    bsf    EECON1,WREN         ; Enable writes
Loop
    bsf    EECON1,RD           ; Loop to refresh array
    movlw  55h                 ; Read current address
    movwf  EECON2              ; Write 55h
    movlw  AAh                 ;
    movwf  EECON2              ; Write AAh
    bsf    EECON1,WR           ; Set WR bit to begin write
    btfsc  EECON1,WR          ; Wait for write to complete
    bra    $-2
    incfsz EEADR,F             ; Increment address
    bra    Loop                ; Not zero, do it again

    bcf    EECON1,WREN         ; Disable writes
    bsf    INTCON,GIE          ; Enable interrupts
    
```

## 13.8 DATA EEPROM OPERATION DURING CODE PROTECT

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

**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 PIC16F627A/628A/648A 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
3. Power-on Reset (POR)
4. Power-up Timer (PWRT)
5. Oscillator Start-Up Timer (OST)
6. Brown-out Reset (BOR)
7. Interrupts
8. Watchdog Timer (WDT)
9. SLEEP
10. Code protection
11. ID Locations
12. In-Circuit Serial Programming™ (ICSP™)

The PIC16F627A/628A/648A 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. 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 RC oscillator option saves system cost while the LP crystal option saves power. A set of configuration bits are used to select various options.

## 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. See Programming Specification (DS41196) for additional information.

# PIC16F627A/628A/648A

## REGISTER 14-1: CONFIGURATION WORD

CP	—	—	—	—	CPD	LVP	BOREN	MCLRE	FOSC2	PWRTE	WDTE	FOSC1	FOSC0
bit 13													bit 0

bit 13: **CP**: FLASH Program Memory Code Protection bit<sup>(2)</sup>  
**(PIC16F648A)**  
 1 = Code protection off  
 0 = 0000h to 0FFFh code protected  
**(PIC16F628A)**  
 1 = Code protection off  
 0 = 0000h to 07FFh code protected  
**(PIC16F627A)**  
 1 = Code protection off  
 0 = 0000h to 03FFh code protected

bit 12-9: **Unimplemented**: Read as '0'

bit 8: **CPD**: Data Code Protection bit<sup>(3)</sup>  
 1 = Data memory code protection off  
 0 = Data memory code protected

bit 7: **LVP**: Low Voltage Programming Enable  
 1 = RB4/PGM pin has PGM function, low voltage programming enabled  
 0 = RB4/PGM is digital I/O, HV on MCLR must be used for programming

bit 6: **BOREN**: Brown-out Reset Enable bit <sup>(1)</sup>  
 1 = BOR Reset enabled  
 0 = BOR Reset disabled

bit 5: **MCLRE**: RA5/MCLR pin function select  
 1 = RA5/MCLR pin function is MCLR  
 0 = RA5/MCLR pin function is digital Input, MCLR internally tied to VDD

bit 3: **PWRTE**: Power-up Timer Enable bit <sup>(1)</sup>  
 1 = PWRT disabled  
 0 = PWRT enabled

bit 2: **WDTEN**: Watchdog Timer Enable bit  
 1 = WDT enabled  
 0 = WDT disabled

bit 4, 1-0: **FOSC2:FOSC0**: Oscillator Selection bits<sup>(4)</sup>  
 111 = RC oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, Resistor and Capacitor on RA7/OSC1/CLKIN  
 110 = RC oscillator: I/O function on RA6/OSC2/CLKOUT pin, Resistor and Capacitor on RA7/OSC1/CLKIN  
 101 = INTOSC oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN  
 100 = INTOSC 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

- Note**
- 1: Enabling Brown-out Reset does not automatically enable the Power-up Timer (PWRT) the way it did in the PIC16F627/628.
  - 2: The code protection scheme has changed from the code protection scheme used in the PIC16F627/628. The entire FLASH program memory needs to be bulk erased to set the CP bit, turning the code protection off. See Programming Specification DS41196 for details.
  - 3: The entire data EEPROM needs to be bulk erased to set the CPD bit, turning the code protection off. See Programming Specification DS41196 for details.
  - 4: When MCLR is asserted in INTOSC mode, the internal clock oscillator is disabled.

### Legend

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



## 14.2 Oscillator Configurations

### 14.2.1 OSCILLATOR TYPES

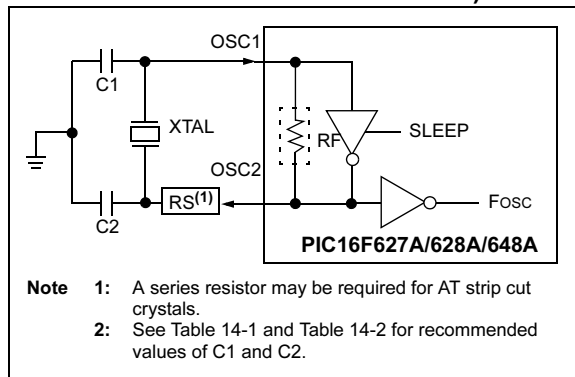
The PIC16F627A/628A/648A can be operated in eight different oscillator options. The user can program three configuration bits (FOSC2 through FOSC0) to select one of these eight modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- RC External Resistor/Capacitor (2 modes)
- INTOSC Internal Precision Oscillator (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 14-1). The PIC16F627A/628A/648A 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 14-4).

**FIGURE 14-1: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (HS, XT OR LP OSC CONFIGURATION)**



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

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

**Note:** 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	15 - 30 pF	15 - 30 pF
	200 kHz	0 - 15 pF	0 - 15 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

**Note:** Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. A series resistor (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.

### 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 14-2 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.

# PIC16F627A/628A/648A

**FIGURE 14-2: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT**

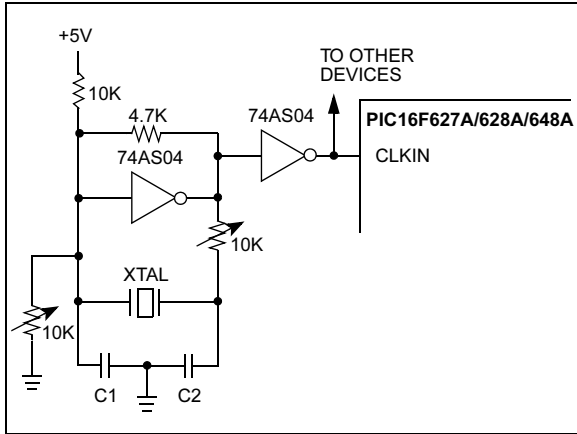
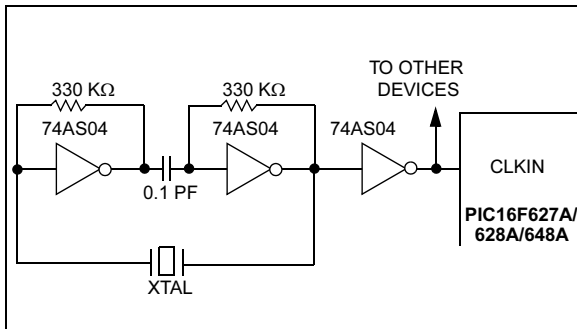


Figure 14-3 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180° phase shift in a series resonant oscillator circuit. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

**FIGURE 14-3: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT**



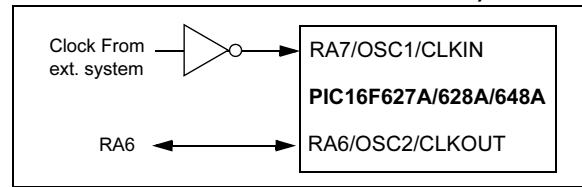
## 14.2.4 PRECISION INTERNAL 4 MHz OSCILLATOR

The internal precision oscillator provides a fixed 4 MHz (nominal) system clock at  $V_{DD} = 5V$  and 25°C. See Section 17.0, Electrical Specifications, for information on variation over voltage and temperature.

## 14.2.5 EXTERNAL CLOCK IN

For applications where a clock is already available elsewhere, users may directly drive the PIC16F627A/628A/648A provided that this external clock source meets the AC/DC timing requirements listed in Section 17.6. Figure 14-4 below shows how an external clock circuit should be configured.

**FIGURE 14-4: EXTERNAL CLOCK INPUT OPERATION (EC, HS, XT OR LP OSC CONFIGURATION)**



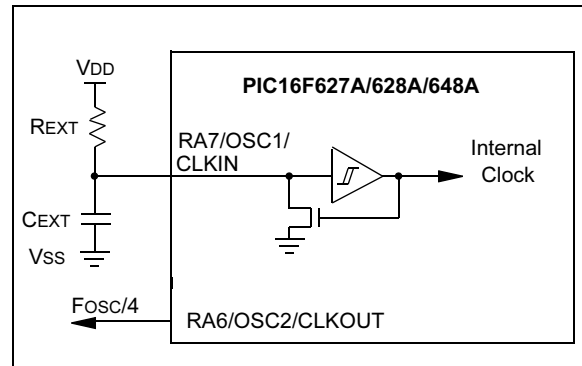
## 14.2.6 RC OSCILLATOR

For applications where precise timing is not a requirement, the RC oscillator option is available. The operation and functionality of the RC oscillator is dependent upon a number of variables. The RC oscillator frequency is a function of:

- Supply voltage
- Resistor ( $R_{EXT}$ ) and capacitor ( $C_{EXT}$ ) values
- Operating temperature.

The oscillator frequency will vary from unit to unit due to normal process parameter variation. The difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low  $C_{EXT}$  values. The user also needs to account for the tolerance of the external R and C components. Figure 14-5 shows how the R/C combination is connected.

**FIGURE 14-5: RC OSCILLATOR MODE**



The RC 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.

## 14.2.7 CLKOUT

The PIC16F627A/628A/648A 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.

# PIC16F627A/628A/648A

## 14.2.8 SPECIAL FEATURE: DUAL SPEED OSCILLATOR MODES

A software programmable dual speed Oscillator mode is provided when the PIC16F627A/628A/648A is configured in the INTOSC Oscillator mode. This feature allows users to dynamically toggle the oscillator speed between 4 MHz and 37 kHz nominal in the INTOSC mode. Applications that require low current power savings, but cannot tolerate putting the part into SLEEP, may use this mode.

There is a time delay associated with the transition between Fast and Slow oscillator speeds. This Oscillator Speed Transition delay consists of two existing clock pulses and eight new speed clock pulses. During this Clock Speed Transition Delay the System Clock is halted causing the processor to be frozen in time. During this delay the Program Counter and the Clock Out stop.

The OSCF bit in the PCON register is used to control Dual Speed mode. See Section 4.2.2.6, Register 4-6.

## 14.3 RESET

The PIC16F627A/628A/648A differentiates between various kinds of RESET:

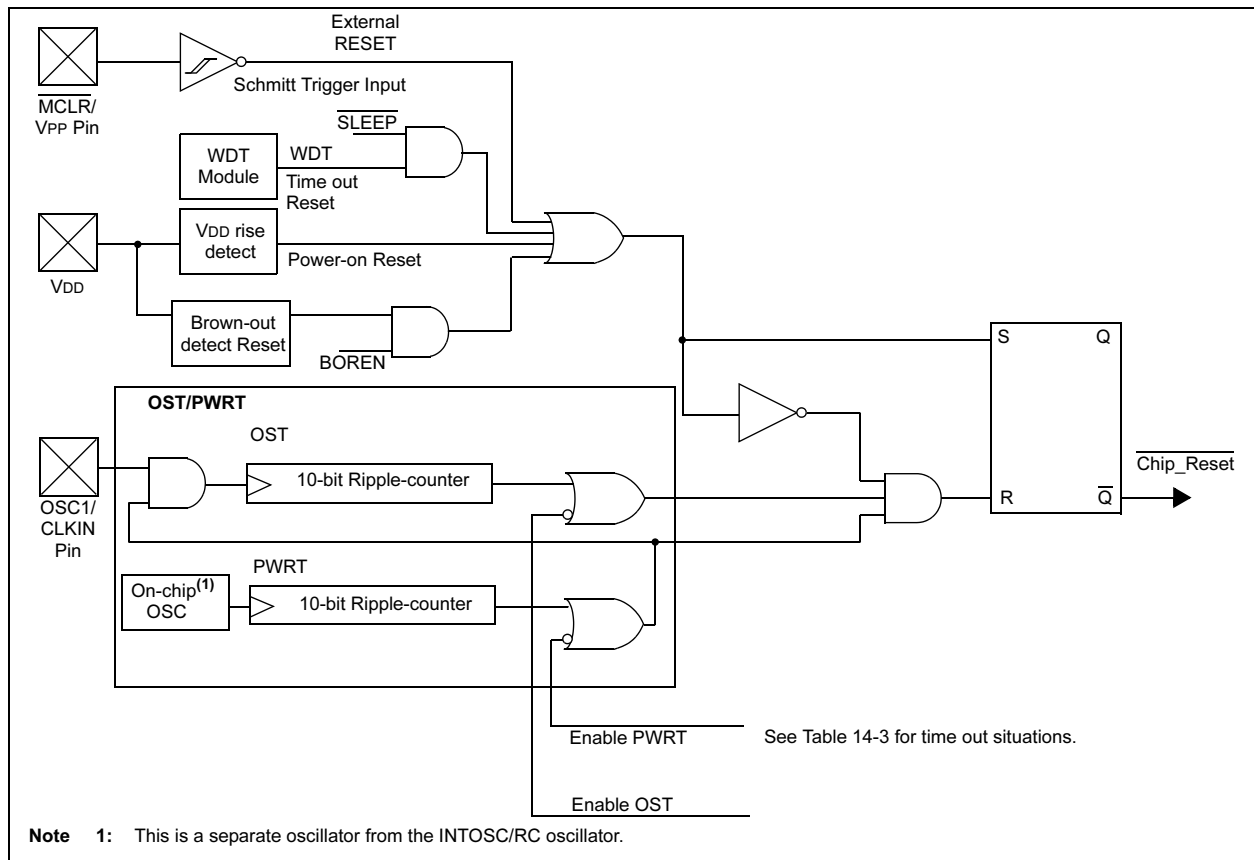
- Power-on Reset (POR)
- $\overline{\text{MCLR}}$  Reset during normal operation
- $\overline{\text{MCLR}}$  Reset during SLEEP
- WDT Reset (normal operation)
- WDT wake-up (SLEEP)
- Brown-out Reset (BOR)

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, Brown-out Reset,  $\overline{\text{MCLR}}$  Reset, WDT Reset and  $\overline{\text{MCLR}}$  Reset during SLEEP. They are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation.  $\overline{\text{TO}}$  and  $\overline{\text{PD}}$  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 14-6.

The  $\overline{\text{MCLR}}$  Reset path has a noise filter to detect and ignore small pulses. See Table 17-7 for pulse width specification.

**FIGURE 14-6: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



# PIC16F627A/628A/648A

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

### 14.4.1 POWER-ON RESET (POR)

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

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

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

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

### 14.4.2 POWER-UP TIMER (PWRT)

The PWRT provides a fixed 72 ms (nominal) time out on power-up (POR) or if enabled from a Brown-out Reset. The PWRT 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, PWRTE can disable (if set) or enable (if cleared or programmed) the PWRT. It is recommended that the PWRT 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 Table 17-7 for details.

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

The OST provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over. Program execution will not start until the OST time out is complete. 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. See Table 17-7.

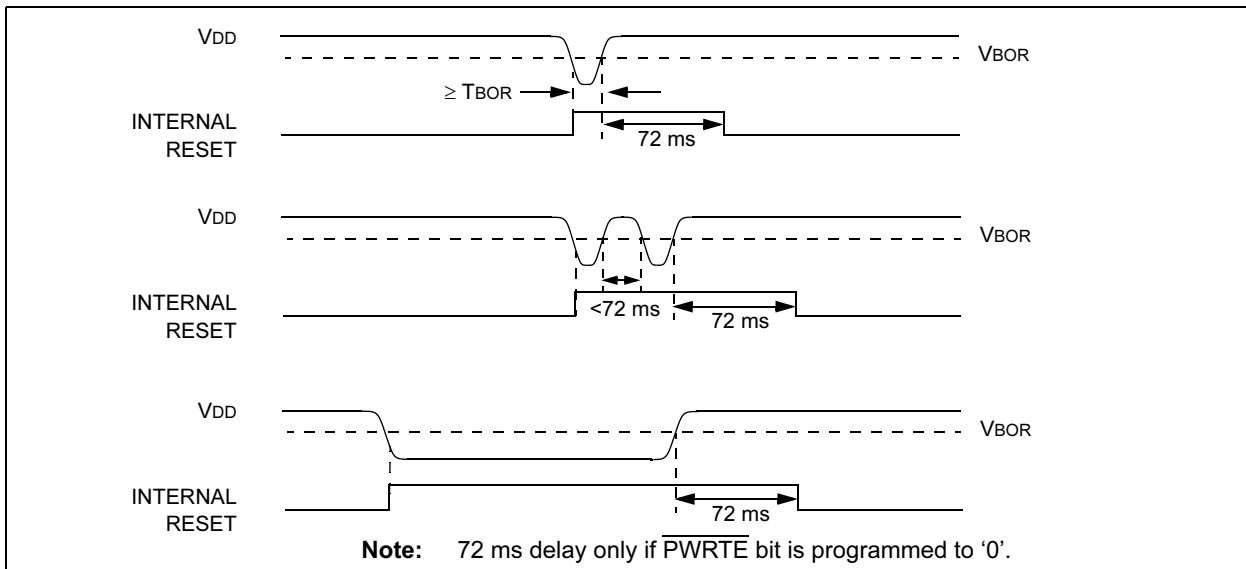
### 14.4.4 BROWN-OUT RESET (BOR)

The PIC16F627A/628A/648A have on-chip BOR circuitry. A configuration bit, BOREN, can disable (if clear/programmed) or enable (if set) the BOR Reset circuitry. If VDD falls below VBOR for longer than TBOR, the brown-out situation will RESET the chip. A RESET is not guaranteed to occur if VDD falls below VBOR for shorter than TBOR. VBOR and TBOR are defined in Table 17-2 and Table 17-7, respectively.

On any RESET (Power-on, Brown-out, Watchdog, etc.), the chip will remain in RESET until VDD rises above BVDD (see Figure 14-7). The Power-up Timer will now be invoked, if enabled, and will keep the chip in RESET an additional 72 ms.

If VDD drops below VBOR 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 VBOR, the Power-Up Timer will execute a 72 ms RESET. Figure 14-7 shows typical Brown-out situations.

FIGURE 14-7: BROWN-OUT SITUATIONS WITH PWRT ENABLED



# PIC16F627A/628A/648A

## 14.4.5 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 RC mode with  $\overline{\text{PWRTE}}$  bit set (PWRT disabled), there will be no time out at all. Figure 14-8, Figure 14-9 and Figure 14-10 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 14-9). This is useful for testing purposes or to synchronize more than one PIC16F627A/628A/648A 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.

## 14.4.6 POWER CONTROL (PCON) STATUS REGISTER

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

Bit0 is  $\overline{\text{BOR}}$  (Brown-out Reset).  $\overline{\text{BOR}}$  is unknown on Power-on-Reset. It must then be set by the user and checked on subsequent RESETS to see if  $\overline{\text{BOR}} = 0$  indicating that a brown-out has occurred. The  $\overline{\text{BOR}}$  STATUS bit is a don't care and is not necessarily predictable if the brown-out circuit is disabled (by setting BOREN 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 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
	$\overline{\text{PWRTE}} = 0$	$\overline{\text{PWRTE}} = 1$	$\overline{\text{PWRTE}} = 0$	$\overline{\text{PWRTE}} = 1$	
XT, HS, LP	72 ms + 1024•Tosc	1024•Tosc	72 ms + 1024•Tosc	1024•Tosc	1024•Tosc
RC, EC	72 ms	—	72 ms	—	—
INTOSC	72 ms	—	72 ms	—	6 $\mu$ s

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

POR	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Condition
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 Reset
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.

# PIC16F627A/628A/648A

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

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, 83h, 103h, 183h	STATUS	IRP	RP1	RPO	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
8Eh	PCON	—	—	—	—	OSCF	—	$\overline{POR}$	$\overline{BOR}$	---- 1-0x	---- u-uq

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

**Note 1:** Other (non Power-up) Resets include  $\overline{MCLR}$  Reset, Brown-out Reset 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
$\overline{MCLR}$ Reset during normal operation	000h	000u uuuu	---- 1-uu
$\overline{MCLR}$ Reset during SLEEP	000h	0001 0uuu	---- 1-uu
WDT Reset	000h	0000 uuuu	---- 1-uu
WDT Wake-up	PC + 1	uuu0 0uuu	---- u-uu
Brown-out Reset	000h	000x xuuu	---- 1-u0
Interrupt Wake-up from SLEEP	PC + 1 <sup>(1)</sup>	uuu1 0uuu	---- u-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.

# PIC16F627A/628A/648A

**TABLE 14-7: INITIALIZATION CONDITION FOR REGISTERS**

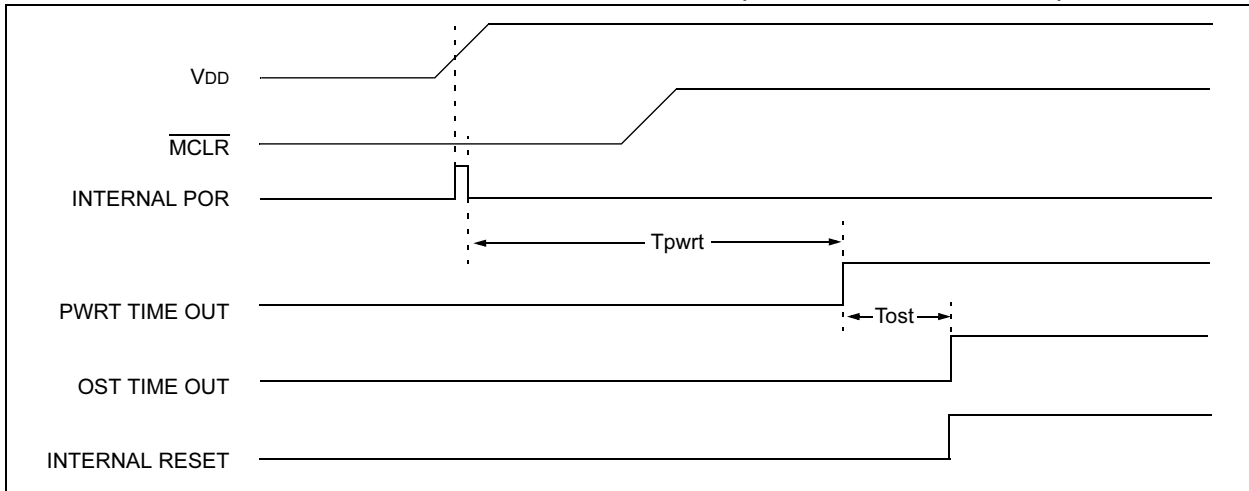
Register	Address	Power-on Reset	<ul style="list-style-type: none"> <li>• MCLR Reset during normal operation</li> <li>• MCLR Reset during SLEEP</li> <li>• WDT Reset</li> <li>• Brown-out Reset <sup>(1)</sup></li> </ul>	<ul style="list-style-type: none"> <li>• Wake-up from SLEEP<sup>(7)</sup> through interrupt</li> <li>• Wake-up from SLEEP<sup>(7)</sup> through WDT time out</li> </ul>
W	—	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	00h	—	—	—
TMR0	01h, 101h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h, 82h, 102h, 182h	0000 0000	0000 0000	PC + 1 <sup>(3)</sup>
STATUS	03h, 83h, 103h, 183h	0001 1xxx	000q quuu <sup>(4)</sup>	uuuq 0uuu <sup>(4)</sup>
FSR	04h, 84h, 104h, 184h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	xxxx 0000	xxxx 0000	uuuu uuuu
PORTB	06h, 106h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCLATH	0Ah, 8Ah, 10Ah, 18Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh, 8Bh, 10Bh, 18Bh	0000 000x	0000 000u	uuuu uqqq <sup>(2)</sup>
PIR1	0Ch	0000 -000	0000 -000	qqqq -qqq <sup>(2)</sup>
TMR1L	0Eh	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	0Fh	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	10h	--00 0000	--uu uuuu <sup>(6)</sup>	--uu uuuu
TMR2	11h	0000 0000	0000 0000	uuuu uuuu
T2CON	12h	-000 0000	-000 0000	-uuu uuuu
CCPR1L	15h	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	16h	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	17h	--00 0000	--00 0000	--uu uuuu
RCSTA	18h	0000 000x	0000 000x	uuuu uuuu
TXREG	19h	0000 0000	0000 0000	uuuu uuuu
RCREG	1Ah	0000 0000	0000 0000	uuuu uuuu
CMCON	1Fh	0000 0000	0000 0000	uu-- uuuu
OPTION	81h, 181h	1111 1111	1111 1111	uuuu uuuu
TRISA	85h	1111 1111	1111 1111	uuuu uuuu
TRISB	86h, 186h	1111 1111	1111 1111	uuuu uuuu
PIE1	8Ch	0000 -000	0000 -000	uuuu -uuu
PCON	8Eh	---- 1-0x	---- 1-uq <sup>(1,5)</sup>	---- u-uu
PR2	92h	1111 1111	1111 1111	uuuu uuuu
TXSTA	98h	0000 -010	0000 -010	uuuu -uuu
SPBRG	99h	0000 0000	0000 0000	uuuu uuuu
EEDATA	9Ah	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEADR	9Bh	xxxx xxxx	uuuu uuuu	uuuu uuuu
EECON1	9Ch	---- x000	---- q000	---- uuuu
EECON2	9Dh	—	—	—
VRCON	9Fh	000- 0000	000- 0000	uuu- uuuu

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

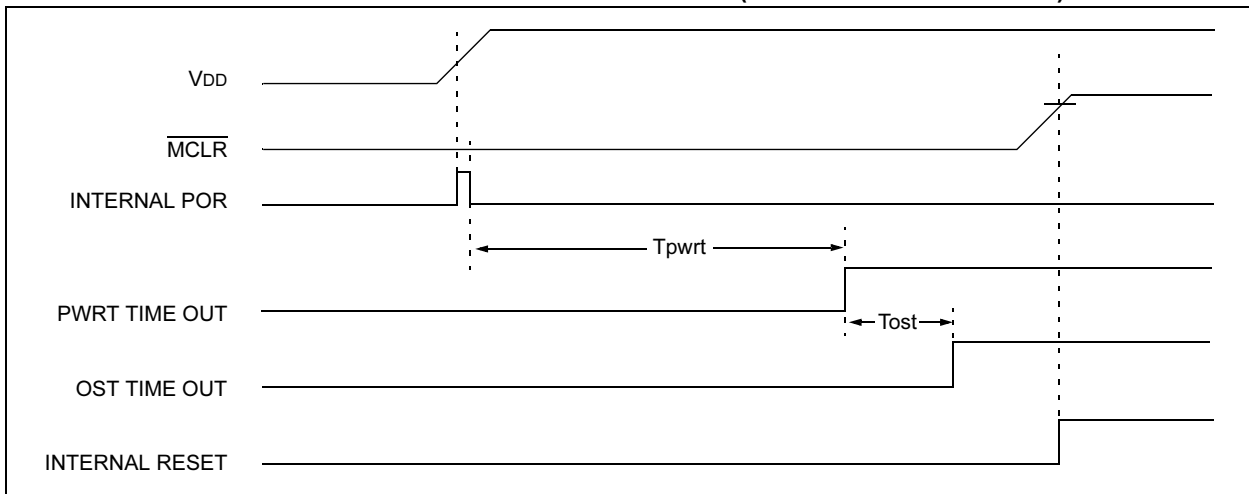
- Note**
- 1: If VDD goes too low, Power-on Reset will be activated and registers will be affected differently.
  - 2: One or more bits in INTCON and PIR1 will be affected (to cause wake-up).
  - 3: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).
  - 4: See Table 14-6 for RESET value for specific condition.
  - 5: If RESET was due to brown-out, then bit 0 = 0. All other RESETS will cause bit 0 = u.
  - 6: Reset to '--00 0000' on a Brown-out Reset (BOR).
  - 7: Peripherals generating interrupts for wake-up from SLEEP will change the resulting bits in the associated registers.

# PIC16F627A/628A/648A

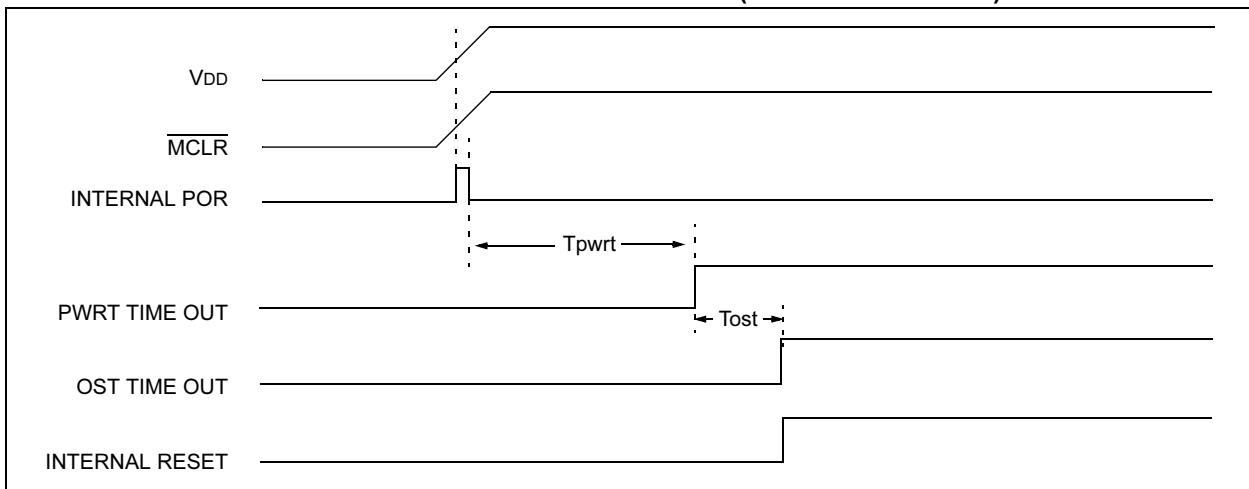
**FIGURE 14-8: TIME OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{DD}$ ): CASE**



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

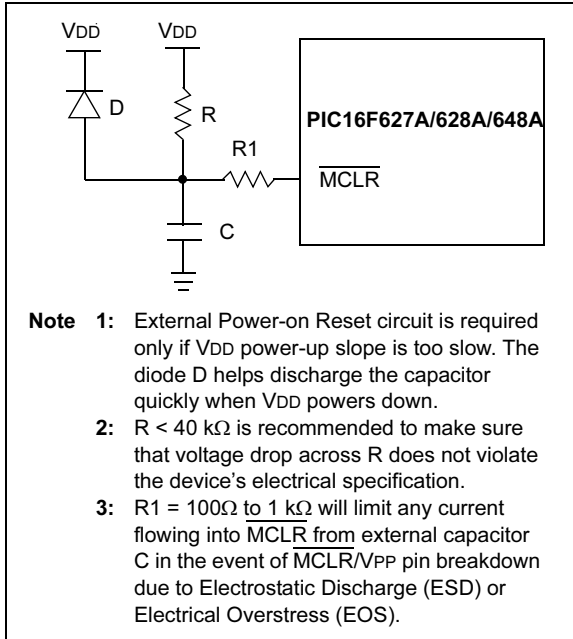


**FIGURE 14-10: TIME OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{DD}$ )**

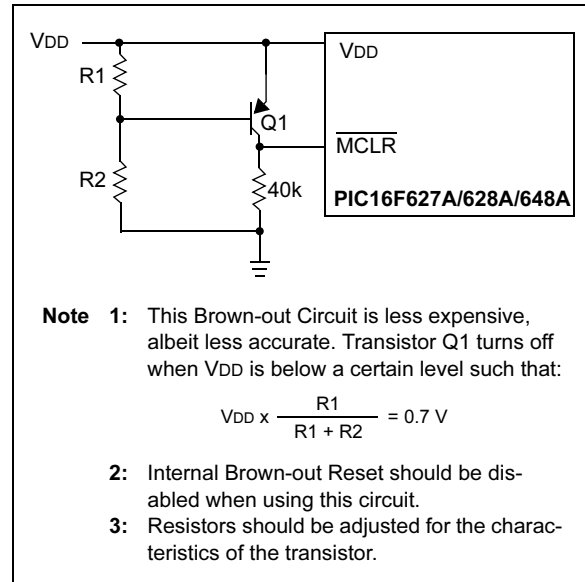




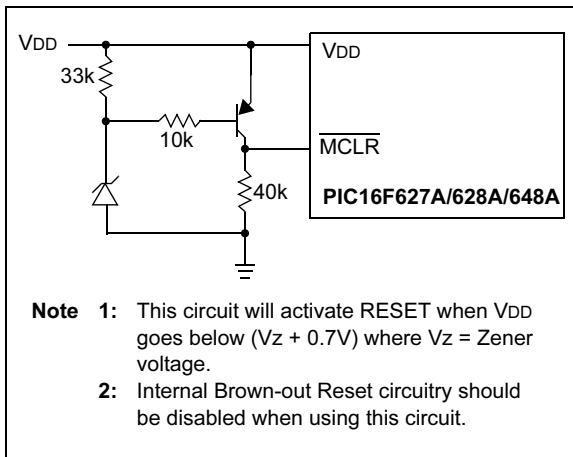
**FIGURE 14-11: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)**



**FIGURE 14-13: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 2**



**FIGURE 14-12: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 1**



# PIC16F627A/628A/648A

## 14.5 Interrupts

The PIC16F627A/628A/648A has 10 sources of interrupt:

- External Interrupt RB0/INT
- TMR0 Overflow Interrupt
- PORTB Change Interrupts (pins RB7:RB4)
- Comparator Interrupt
- USART Interrupt TX
- USART Interrupt RX
- CCP Interrupt
- TMR1 Overflow Interrupt
- TMR2 Match Interrupt
- Data EEPROM 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-enables 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.

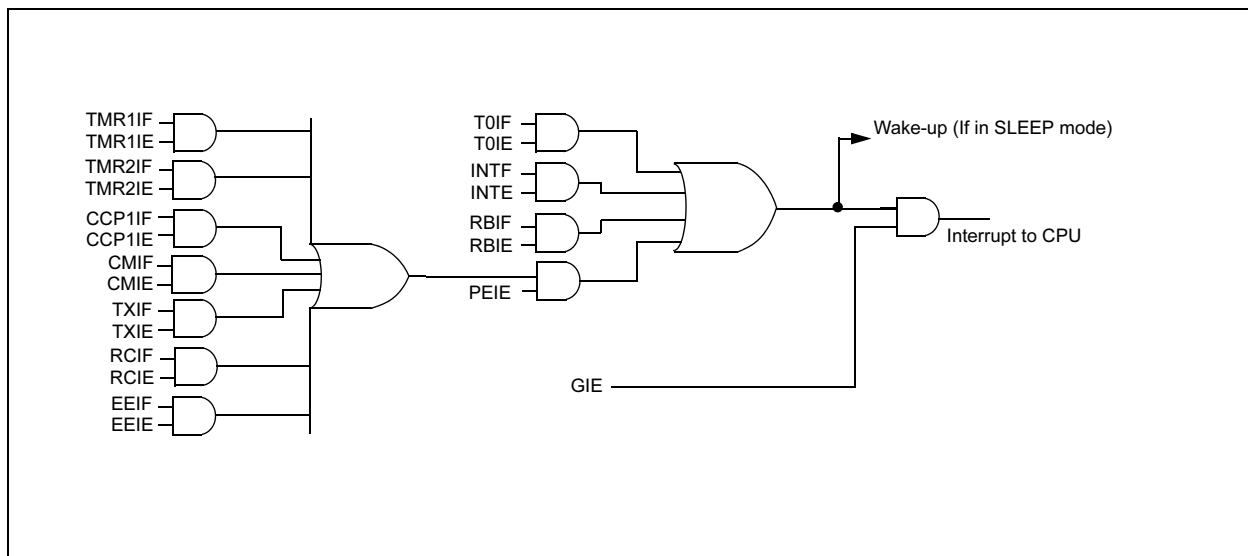
When an interrupt is responded to, the GIE is cleared to disable any further interrupt, the return address is pushed into the stack and the PC is loaded with 0004h. Once in the interrupt service routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt 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 14-15). 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 14-14: INTERRUPT LOGIC



## 14.5.1 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 14.8 for details on SLEEP, and Figure 14-17 for timing of wake-up from SLEEP through RB0/INT interrupt.

## 14.5.2 TMR0 INTERRUPT

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

## 14.5.3 PORTB INTERRUPT

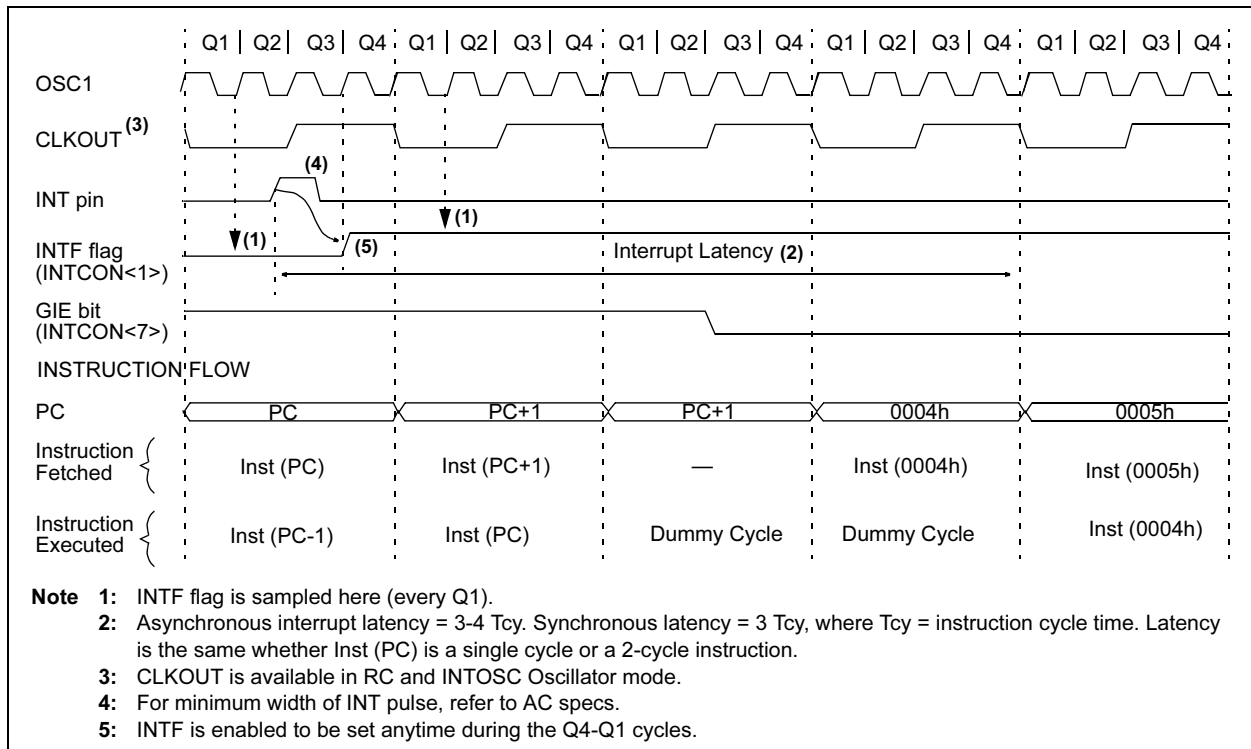
An input change on PORTB <7:4> sets the RBIF (INTCON<0>) bit. The interrupt can be enabled/disabled by setting/clearing the RBIE (INTCON<4>) bit. For operation of PORTB (Section 5.2).

**Note:** If a change on the I/O pin should occur when the read operation is being executed (starts during the Q2 cycle and ends before the start of the Q3 cycle), then the RBIF interrupt flag may not get set.

## 14.5.4 COMPARATOR INTERRUPT

See Section 10.6 for complete description of comparator interrupts.

**FIGURE 14-15: INT PIN INTERRUPT TIMING**



# PIC16F627A/628A/648A

**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, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	TOIF	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.

## 14.6 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 must be implemented in software.

Example 14-2 stores and restores the STATUS and W registers. The user register, W\_TEMP, must be defined in a common memory location (i.e., W\_TEMP is defined at 0x70 in Bank 0 and is therefore, accessible at 0xF0, 0x170 and 0x1F0). The Example 14-2:

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

### EXAMPLE 14-2: SAVING THE STATUS AND W REGISTERS IN RAM

```

MOVWF W_TEMP      ;copy W to temp register,
                  ;could be in any 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
    
```

## 14.7 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 RC 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).

### 14.7.1 WDT PERIOD

The WDT has a nominal time out period of 18 ms (with no prescaler). The time out periods vary with temperature, VDD and process variations from part to part (see DC Specifications, Table 17-7). If longer time out periods are desired, a postscaler 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.

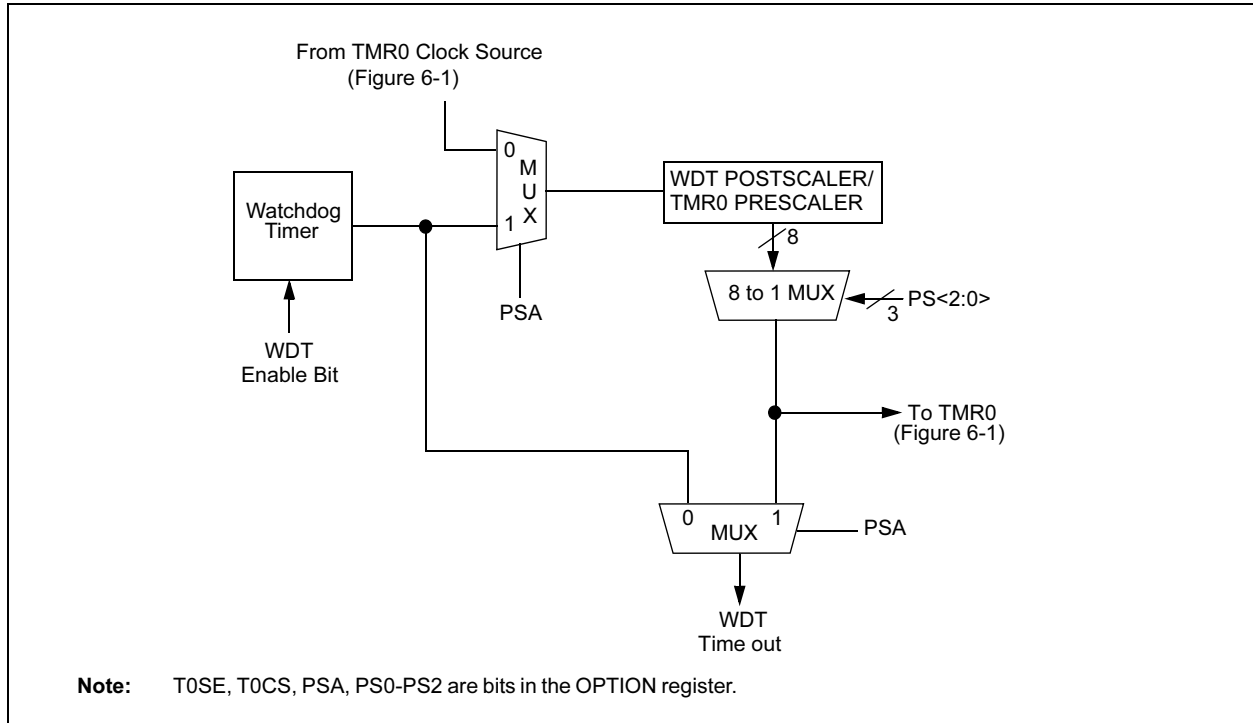
The CLRWDT and SLEEP instructions clear the WDT and the postscaler, if assigned to the WDT, and prevent it from timing out and generating a device RESET.

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

### 14.7.2 WDT PROGRAMMING CONSIDERATIONS

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

**FIGURE 14-16: 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	MCLR $\overline{\text{RE}}$	FOSC2	$\overline{\text{PWRTE}}$	WDTE	FOSC1	FOSC0	uuuu uuuu	uuuu uuuu
81h, 181h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', q = value depends upon condition.

**Note:** Shaded cells are not used by the Watchdog Timer.

## 14.8 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  $\overline{\text{PD}}$  bit in the STATUS register is cleared, the 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  $\overline{\text{MCLR}}$  pin must be at a logic high level (VIHMC).

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

# PIC16F627A/628A/648A

## 14.8.1 WAKE-UP FROM SLEEP

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

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

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

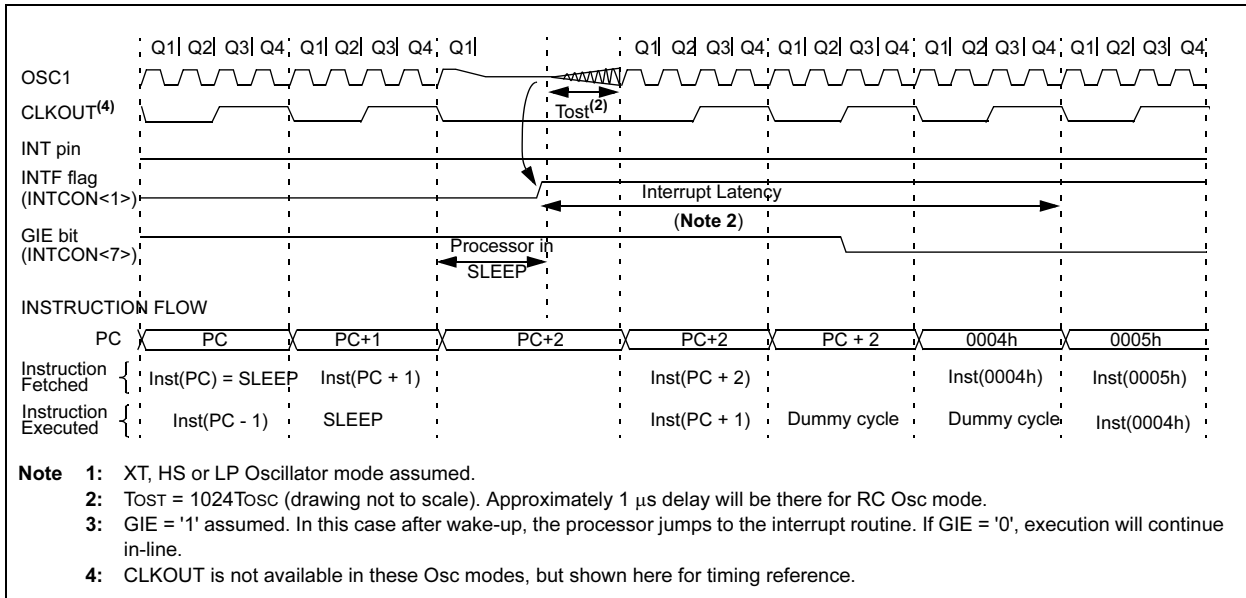
When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the correspond-

ing 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 an 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 not enter SLEEP. The SLEEP instruction is executed as a NOP instruction.

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

**FIGURE 14-17: WAKE-UP FROM SLEEP THROUGH INTERRUPT**



## 14.9 Code Protection

With the Code Protect bit is cleared (Code Protect enabled) the contents of the program memory locations are read out as "00". See Programing Specification, DS41196, for details.

**Note:** Only a Bulk Erase function can set the CP and CPD bits by turning off the code protection. The entire data EEPROM and FLASH program memory will be erased to turn the code protection off.

## 14.10 User ID Locations

Four memory locations (2000h-2003h) are designated as user 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 user ID locations are used.

## 14.11 In-Circuit Serial Programming

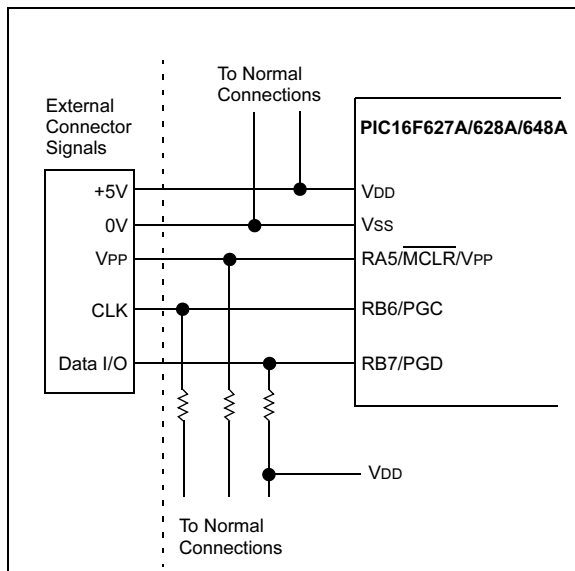
The PIC16F627A/628A/648A microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground, and the programming voltage. This allows customers to manufacture boards with unprogrammed devices 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 (DS41196).

A typical In-Circuit Serial Programming connection is shown in Figure 14-18.

**FIGURE 14-18: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION**



## 14.12 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 operations.

If Low-voltage Programming mode is not used, the LVP bit should be programmed to a '0' so that 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 high voltage Programming mode can be used to program the device.

# PIC16F627A/628A/648A

---

---

NOTES:





# PIC16F627A/628A/648A

**TABLE 15-2: PIC16F627A/628A/648A INSTRUCTION SET**

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

- Note 1:** When an I/O register is modified as a function of itself ( e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
- Note 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.
- Note 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

### ADDLW Add Literal and W

Syntax:	[ <i>label</i> ] ADDLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	$(W) + k \rightarrow (W)$				
Status Affected:	C, DC, Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 20px; text-align: center;">11</td> <td style="width: 20px; text-align: center;">111x</td> <td style="width: 20px; text-align: center;">kkkk</td> <td style="width: 20px; text-align: center;">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>				

### ANDLW AND Literal with W

Syntax:	[ <i>label</i> ] ANDLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	$(W) .AND. (k) \rightarrow (W)$				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 20px; text-align: center;">11</td> <td style="width: 20px; text-align: center;">1001</td> <td style="width: 20px; text-align: center;">kkkk</td> <td style="width: 20px; text-align: center;">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>				

### ADDWF Add W and f

Syntax:	[ <i>label</i> ] ADDWF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	$(W) + (f) \rightarrow (dest)$				
Status Affected:	C, DC, Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 20px; text-align: center;">00</td> <td style="width: 20px; text-align: center;">0111</td> <td style="width: 20px; text-align: center;">dfff</td> <td style="width: 20px; text-align: center;">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 REG1, 0 Before Instruction W = 0x17 REG1 = 0xC2 After Instruction W = 0xD9 REG1 = 0xC2 Z = 0 C = 0 DC = 0</pre>				

### ANDWF AND W with f

Syntax:	[ <i>label</i> ] ANDWF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	$(W) .AND. (f) \rightarrow (dest)$				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 20px; text-align: center;">00</td> <td style="width: 20px; text-align: center;">0101</td> <td style="width: 20px; text-align: center;">dfff</td> <td style="width: 20px; text-align: center;">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 REG1, 1 Before Instruction W = 0x17 REG1 = 0xC2 After Instruction W = 0x17 REG1 = 0x02</pre>				

# PIC16F627A/628A/648A

## 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 REG1, 7`

Before Instruction  
`REG1 = 0xC7`  
 After Instruction  
`REG1 = 0x47`

## 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 REG1, 7`

Before Instruction  
`REG1 = 0x0A`  
 After Instruction  
`REG1 = 0x8A`

## BTFSC Bit Test f, 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   REG1
FALSE   GOTO   PROCESS_CODE
TRUE    .
        .
        .
```

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

# PIC16F627A/628A/648A

## 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   REG1
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   REG1

Before Instruction
      REG1 = 0x5A
After Instruction
      REG1 = 0x00
      Z    = 1
    
```

# PIC16F627A/628A/648A

## CLRW Clear W

Syntax:	[ <i>label</i> ] CLRW				
Operands:	None				
Operation:	00h → (W) 1 → Z				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>00</td><td>0001</td><td>0000</td><td>0011</td></tr></table>	00	0001	0000	0011
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				

## COMF Complement f

Syntax:	[ <i>label</i> ] COMF f,d				
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]				
Operation:	(f) → (dest)				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>00</td><td>1001</td><td>dfff</td><td>ffff</td></tr></table>	00	1001	dfff	ffff
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				

## CLRWDT Clear Watchdog Timer

Syntax:	[ <i>label</i> ] CLRWDT				
Operands:	None				
Operation:	00h → WDT 0 → WDT prescaler, 1 → $\overline{TO}$ 1 → $\overline{PD}$				
Status Affected:	$\overline{TO}$ , $\overline{PD}$				
Encoding:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>00</td><td>0000</td><td>0110</td><td>0100</td></tr></table>	00	0000	0110	0100
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				

## DECF Decrement f

Syntax:	[ <i>label</i> ] DECF f,d				
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]				
Operation:	(f) - 1 → (dest)				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>00</td><td>0011</td><td>dfff</td><td>ffff</td></tr></table>	00	0011	dfff	ffff
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				

# PIC16F627A/628A/648A

<b>DECFSZ</b>	<b>Decrement f, Skip if 0</b>				
Syntax:	[ <i>label</i> ] 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:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">00</td> <td style="padding: 2px 10px;">1011</td> <td style="padding: 2px 10px;">dfff</td> <td style="padding: 2px 10px;">ffff</td> </tr> </table>	00	1011	dfff	ffff
00	1011	dfff	ffff		
Description:	<p>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'.</p> <p>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.</p>				
Words:	1				
Cycles:	1(2)				
Example	<pre> HERE    DECFSZ  REG1, 1         GOTO    LOOP CONTINUE •         •         • </pre> <p>Before Instruction PC = address HERE</p> <p>After Instruction REG1 = REG1 - 1 if REG1 = 0, PC = address CONTINUE if REG1 <math>\neq</math> 0, PC = address HERE+1</p>				

<b>GOTO</b>	<b>Unconditional Branch</b>				
Syntax:	[ <i>label</i> ] GOTO k				
Operands:	$0 \leq k \leq 2047$				
Operation:	$k \rightarrow \text{PC}<10:0>$ $\text{PCLATH}<4:3> \rightarrow \text{PC}<12:11>$				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">10</td> <td style="padding: 2px 10px;">1kkk</td> <td style="padding: 2px 10px;">kkkk</td> <td style="padding: 2px 10px;">kkkk</td> </tr> </table>	10	1kkk	kkkk	kkkk
10	1kkk	kkkk	kkkk		
Description:	<p>GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits &lt;10:0&gt;. The upper bits of PC are loaded from PCLATH&lt;4:3&gt;. GOTO is a two-cycle instruction.</p>				
Words:	1				
Cycles:	2				
Example	<pre> GOTO THERE </pre> <p>After Instruction PC = Address THERE</p>				

# PIC16F627A/628A/648A

## 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 REG1, 1`

Before Instruction

`REG1 = 0xFF`  
`Z = 0`

After Instruction

`REG1 = 0x00`  
`Z = 1`

## 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 REG1, 1`

`GOTO LOOP`

`CONTINUE` •  
 •  
 •

Before Instruction

`PC = address HERE`

After Instruction

`REG1 = REG1 + 1`  
 if `CNT = 0`,  
`PC = address CONTINUE`  
 if `REG1 ≠ 0`,  
`PC = address HERE + 1`



# PIC16F627A/628A/648A

## IORLW Inclusive OR Literal with W

Syntax:	[ <i>label</i> ] IORLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	(W) .OR. k $\rightarrow$ (W)				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">11</td><td style="padding: 2px 10px;">1000</td><td style="padding: 2px 10px;">kkkk</td><td style="padding: 2px 10px;">kkkk</td></tr></table>	11	1000	kkkk	kkkk
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	<pre>IORLW    0x35 Before Instruction     W = 0x9A After Instruction     W = 0xBF     Z = 0</pre>				

## MOVLW Move Literal to W

Syntax:	[ <i>label</i> ] MOVLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	k $\rightarrow$ (W)				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">11</td><td style="padding: 2px 10px;">00xx</td><td style="padding: 2px 10px;">kkkk</td><td style="padding: 2px 10px;">kkkk</td></tr></table>	11	00xx	kkkk	kkkk
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	<pre>MOVLW   0x5A After Instruction     W = 0x5A</pre>				

## IORWF Inclusive OR W with f

Syntax:	[ <i>label</i> ] IORWF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	(W) .OR. (f) $\rightarrow$ (dest)				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">00</td><td style="padding: 2px 10px;">0100</td><td style="padding: 2px 10px;">dfff</td><td style="padding: 2px 10px;">ffff</td></tr></table>	00	0100	dfff	ffff
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	<pre>IORWF   REG1, 0 Before Instruction     REG1 = 0x13     W    = 0x91 After Instruction     REG1 = 0x13     W    = 0x93     Z    = 1</pre>				

## MOVF Move f

Syntax:	[ <i>label</i> ] MOVF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	(f) $\rightarrow$ (dest)				
Status Affected:	Z				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">00</td><td style="padding: 2px 10px;">1000</td><td style="padding: 2px 10px;">dfff</td><td style="padding: 2px 10px;">ffff</td></tr></table>	00	1000	dfff	ffff
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	<pre>MOVF   REG1, 0 After Instruction     W = value in REG1 register     Z = 1</pre>				

# PIC16F627A/628A/648A

## MOVWF Move W to f

Syntax:	[ <i>label</i> ] MOVWF f				
Operands:	$0 \leq f \leq 127$				
Operation:	(W) → (f)				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">00</td> <td style="padding: 2px 10px;">0000</td> <td style="padding: 2px 10px;">1fff</td> <td style="padding: 2px 10px;">ffff</td> </tr> </table>	00	0000	1fff	ffff
00	0000	1fff	ffff		
Description:	Move data from W register to register 'f'.				
Words:	1				
Cycles:	1				
Example	MOVWF REG1				

Before Instruction  
 REG1 = 0xFF  
 W = 0x4F  
 After Instruction  
 REG1 = 0x4F  
 W = 0x4F

## OPTION Load Option Register

Syntax:	[ <i>label</i> ] OPTION				
Operands:	None				
Operation:	(W) → OPTION				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">00</td> <td style="padding: 2px 10px;">0000</td> <td style="padding: 2px 10px;">0110</td> <td style="padding: 2px 10px;">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. Using only register instruction such as MOVWF.				
Words:	1				
Cycles:	1				
Example					

**To maintain upward compatibility with future PICmicro® products, do not use this instruction.**

## NOP No Operation

Syntax:	[ <i>label</i> ] NOP				
Operands:	None				
Operation:	No operation				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">00</td> <td style="padding: 2px 10px;">0000</td> <td style="padding: 2px 10px;">0xx0</td> <td style="padding: 2px 10px;">0000</td> </tr> </table>	00	0000	0xx0	0000
00	0000	0xx0	0000		
Description:	No operation.				
Words:	1				
Cycles:	1				
Example	NOP				

## RETIE Return from Interrupt

Syntax:	[ <i>label</i> ] RETIE				
Operands:	None				
Operation:	TOS → PC, 1 → GIE				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">00</td> <td style="padding: 2px 10px;">0000</td> <td style="padding: 2px 10px;">0000</td> <td style="padding: 2px 10px;">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	RETIE After Interrupt PC = TOS GIE = 1				

# PIC16F627A/628A/648A


## RETLW Return with Literal in W

Syntax:	[ <i>label</i> ] RETLW <i>k</i>				
Operands:	$0 \leq k \leq 255$				
Operation:	$k \rightarrow (W)$ ; TOS $\rightarrow$ PC				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">11</td><td style="padding: 2px 10px;">01xx</td><td style="padding: 2px 10px;">kkkk</td><td style="padding: 2px 10px;">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	<pre>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</pre>				

## RETURN Return from Subroutine

Syntax:	[ <i>label</i> ] RETURN				
Operands:	None				
Operation:	TOS $\rightarrow$ PC				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">00</td><td style="padding: 2px 10px;">0000</td><td style="padding: 2px 10px;">0000</td><td style="padding: 2px 10px;">1000</td></tr></table>	00	0000	0000	1000
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	<pre>RETURN  After Interrupt     PC = TOS</pre>				

## RLF Rotate Left f through Carry

Syntax:	[ <i>label</i> ] RLF <i>f,d</i>				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	See description below				
Status Affected:	C				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">00</td><td style="padding: 2px 10px;">1101</td><td style="padding: 2px 10px;">dfff</td><td style="padding: 2px 10px;">ffff</td></tr></table>	00	1101	dfff	ffff
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	<pre>RLF    REG1, 0  Before Instruction     REG1=1110 0110     C = 0 After Instruction     REG1=1110 0110     W  = 1100 1100     C  = 1</pre>				

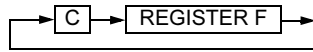
# PIC16F627A/628A/648A

## 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

## 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 14.8 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 Affected: C, DC, Z  
 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

# PIC16F627A/628A/648A

## 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 Affected: C, DC, Z

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  
Z = DC = 1

Example 2: Before Instruction

REG1 = 2  
W = 2  
C = ?

After Instruction

REG1 = 0  
W = 2  
C = 1; result is zero  
Z = DC = 1

Example 3: Before Instruction

REG1 = 1  
W = 2  
C = ?

After Instruction

REG1 = 0xFF  
W = 2  
C = 0; result is negative  
Z = DC = 0

## SWAPF Swap Nibbles in f

Syntax: [label] SWAPF f,d

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

Operation:  $(f<3:0>) \rightarrow (\text{dest}<7:4>)$ ,  
 $(f<7:4>) \rightarrow (\text{dest}<3:0>)$

Status Affected: None

Encoding: 

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 REG1, 0

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5  
W = 0x5A

TRIS	Load TRIS Register				
Syntax:	[label] TRIS f				
Operands:	$5 \leq f \leq 7$				
Operation:	$(W) \rightarrow \text{TRIS register } f$ ;				
Status Affected:	None				
Encoding:	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td style="padding: 2px 10px;">00</td><td style="padding: 2px 10px;">0000</td><td style="padding: 2px 10px;">0110</td><td style="padding: 2px 10px;">0fff</td></tr></table>	00	0000	0110	0fff
00	0000	0110	0fff		
Description:	The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.				
Words:	1				
Cycles:	1				
Example	<div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>To maintain upward compatibility with future PICmicro® products, do not use this instruction.</b></p> </div>				

# PIC16F627A/628A/648A

---

---

## XORLW Exclusive OR Literal with W

---

Syntax: [label] XORLW k

Operands:  $0 \leq k \leq 255$

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

Status Affected: Z

Encoding: 

11	1010	kkkk	kkkk
----	------	------	------

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

Words: 1

Cycles: 1

Example: XORLW 0xAF

Before Instruction

W = 0xB5

After Instruction

W = 0x1A

## XORWF Exclusive OR W with f

---

Syntax: [label] XORWF f,d

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

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

Status Affected: Z

Encoding: 

00	0110	dfff	ffff
----	------	------	------

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example: XORWF REG1, 1

Before Instruction

REG1 = 0xAF

W = 0xB5

After Instruction

REG1 = 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<sup>®</sup> IDE Software
- Assemblers/Compilers/Linkers
  - MPASM<sup>™</sup> Assembler
  - MPLAB C17 and MPLAB C18 C Compilers
  - MPLINK<sup>™</sup> Object Linker/  
MPLIB<sup>™</sup> Object Librarian
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - ICEPIC<sup>™</sup> In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD
- Device Programmers
  - PRO MATE<sup>®</sup> II Universal Device Programmer
  - PICSTART<sup>®</sup> Plus Entry-Level Development Programmer
- Low Cost Demonstration Boards
  - PICDEM<sup>™</sup> 1 Demonstration Board
  - PICDEM 2 Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 17 Demonstration Board
  - KEELOQ<sup>®</sup> Demonstration Board

### 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. The MPLAB IDE is a Windows<sup>®</sup>-based application that contains:

- An interface to debugging tools
  - simulator
  - programmer (sold separately)
  - emulator (sold separately)
  - in-circuit debugger (sold separately)
- A full-featured editor
- A project manager
- Customizable toolbar and key mapping
- A status bar
- On-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
  - machine code

The ability to use MPLAB IDE with multiple debugging tools allows users to easily switch from the cost-effective simulator to a full-featured emulator with minimal retraining.

### 16.2 MPASM Assembler

The MPASM assembler is a full-featured universal macro assembler for all PICmicro MCU's.

The MPASM assembler has a command line interface and a Windows shell. It can be used as a stand-alone application on a Windows 3.x or greater system, or it can be used through MPLAB IDE. The MPASM assembler generates relocatable object files for the MPLINK object linker, Intel<sup>®</sup> standard HEX files, MAP files to detail memory usage and symbol reference, an absolute LST file that contains source lines and generated machine code, and a COD file for debugging.

The MPASM assembler features include:

- Integration into MPLAB IDE projects.
- User-defined macros to streamline assembly code.
- Conditional assembly for multi-purpose source files.
- Directives that 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 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.

# PIC16F627A/628A/648A

---

## 16.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can also link relocatable objects from pre-compiled libraries, using directives from a linker script.

The MPLIB object librarian is a librarian for pre-compiled code to be used with the MPLINK object linker. When a routine from a library is called from another source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. The MPLIB object librarian manages the creation and modification of library files.

The MPLINK object linker features include:

- Integration with MPASM assembler and MPLAB C17 and MPLAB C18 C compilers.
- Allows all memory areas to be defined as sections to provide link-time flexibility.

The MPLIB object librarian features include:

- Easier linking because single libraries can be included instead of many smaller files.
- Helps keep code maintainable by grouping related modules together.
- Allows 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-hosted 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.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and the MPLAB C18 C compilers and the MPASM assembler. 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 the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE in-circuit 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 environment were chosen to best make these features available to you, the end user.

## 16.7 ICEPIC In-Circuit Emulator

The ICEPIC low cost, in-circuit emulator is a 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.



## 16.8 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 PICmicro MCUs and can be used to develop for this and other PICmicro microcontrollers. The MPLAB ICD utilizes the in-circuit debugging capability built into the FLASH devices. This feature, along with Microchip's In-Circuit Serial Programming™ protocol, offers cost-effective in-circuit FLASH 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.

## 16.9 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in Stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable VDD and VPP supplies, which allow 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 device programmer can read, verify, or program PICmicro devices. It can also set code protection in this mode.

## 16.10 PICSTART Plus Entry Level Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient.

The PICSTART Plus development programmer 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. The PICSTART Plus development programmer is CE compliant.

## 16.11 PICDEM 1 Low Cost PICmicro Demonstration Board

The PICDEM 1 demonstration board 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 user can program the sample microcontrollers provided with the PICDEM 1 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The user can also connect the PICDEM 1 demonstration board to the MPLAB ICE in-circuit emulator and download the firmware to the emulator for testing. A 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.

## 16.12 PICDEM 2 Low Cost PIC16CXX Demonstration Board

The PICDEM 2 demonstration board 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 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 2 demonstration board to test firmware. A 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.

# PIC16F627A/628A/648A

---

## 16.13 PICDEM 3 Low Cost PIC16CXXX Demonstration Board

The PICDEM 3 demonstration board 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 an 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 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer with an adapter socket, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 3 demonstration board to test firmware. A prototype area has been provided to the user for adding 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 thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM 3 demonstration board is a LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM 3 demonstration board provides an additional RS-232 interface and Windows 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.

## 16.14 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, 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 device programmer, or the PICSTART Plus development programmer, and easily debug and test the sample code. In addition, the PICDEM 17 demonstration board supports downloading of programs to and executing out of external FLASH memory on board. The PICDEM 17 demonstration board is also usable with the MPLAB ICE in-circuit emulator, or the 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.

## 16.15 KEELOQ Evaluation and Programming Tools

KEELOQ evaluation and programming tools support Microchip's HCS Secure Data Products. The HCS evaluation kit includes a LCD display to show changing codes, a decoder to decode transmissions and a programming interface to program test transmitters.

# PIC16F627A/628A/648A

TABLE 16-1: DEVELOPMENT TOOLS FROM MICROCHIP

Tool	PIC12CXX	PIC14000	PIC16C5X	PIC16C6X	PIC16CXX	PIC16C7X	PIC16C7XX	PIC16C8X	PIC16F8XX	PIC16G9XX	PIC17C4X	PIC17C7XX	PIC18CXX2	PIC18FXX	24CXX/ 25CXX/ 93CXX	HC5XX	MCRFXXX	MCP2510
<b>Software Tools</b>																		
MPLAB® Integrated Development Environment	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MPLAB® C17 C Compiler																		
MPLAB® C18 C Compiler																		
MPASM™ Assembler/ MPLINK™ Object Linker	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MPLAB® ICE In-Circuit Emulator	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ICEPIC™ In-Circuit Emulator	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<b>Debugger</b>																		
MPLAB® ICD In-Circuit Debugger				✓*					✓									
<b>Programmers</b>																		
PICSTART® Plus Entry Level Development Programmer	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PRO MATE® II Universal Device Programmer	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<b>Demo Boards and Eval Kits</b>																		
PICDEM™ 1 Demonstration Board			✓								✓							
PICDEM™ 2 Demonstration Board				✓†									✓					
PICDEM™ 3 Demonstration Board										✓								
PICDEM™ 14A Demonstration Board		✓																
PICDEM™ 17 Demonstration Board											✓							
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.

# PIC16F627A/628A/648A

---

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 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 <sup>(1)</sup> .....	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 (Combined).....	200 mA
Maximum current sourced by PORTA and PORTB (Combined).....	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 MCLR pin, inducing currents greater than 80 mA, may cause latchup. Thus, a series resistor of 50-100 Ω should be used when applying a “low” level to the MCLR pin rather than pulling this pin directly to VSS.

# PIC16F627A/628A/648A

FIGURE 17-1: PIC16F627A/628A/648A VOLTAGE-FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq \text{TA} \leq +125^{\circ}\text{C}$

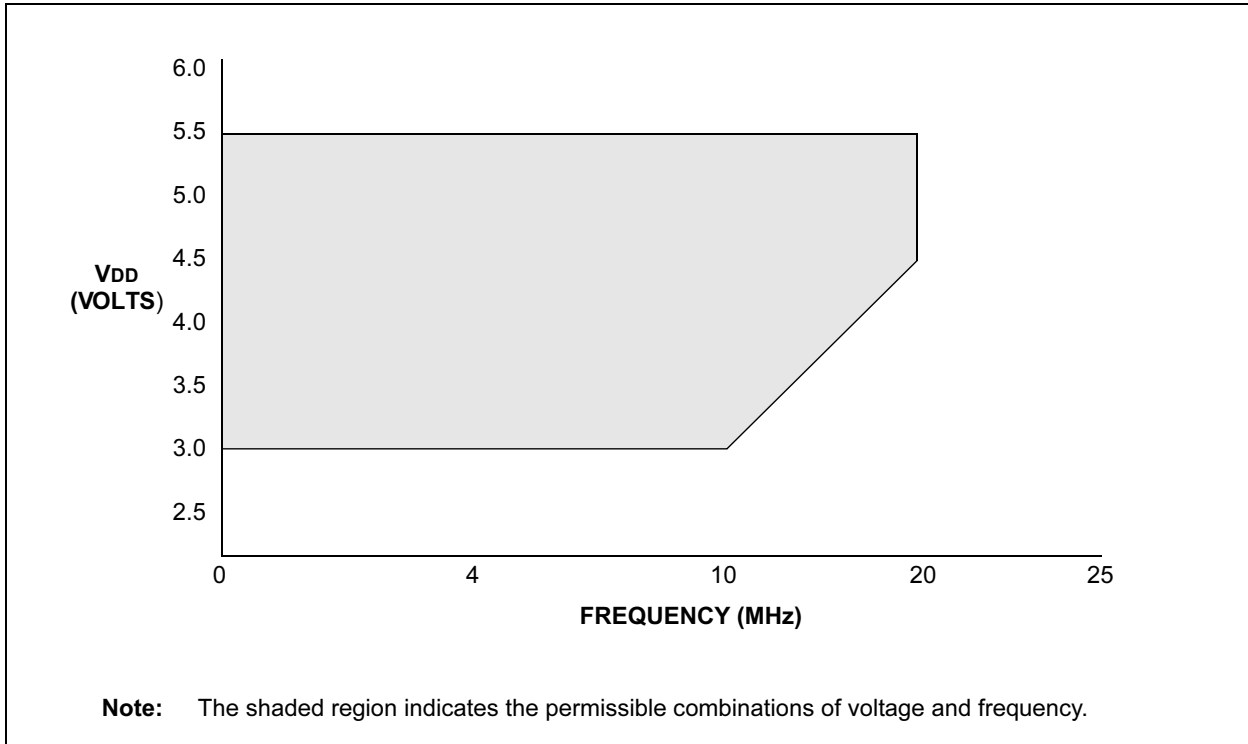
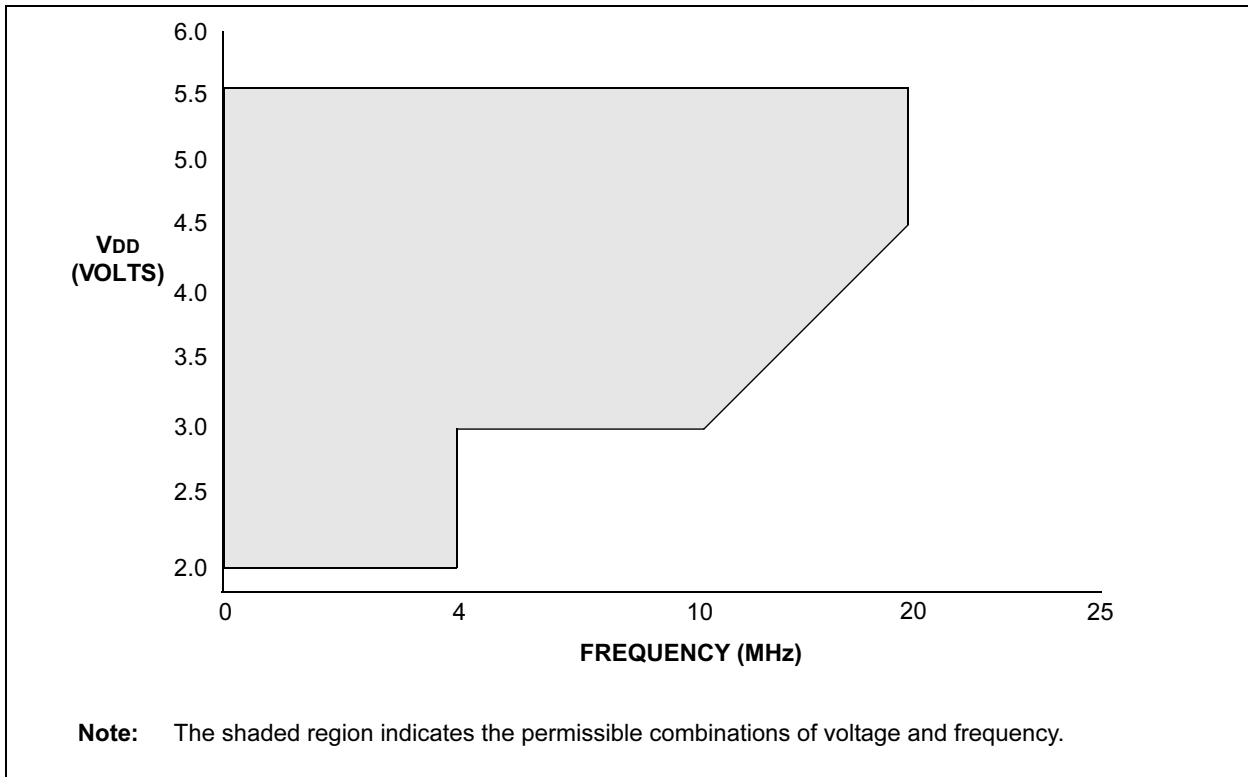


FIGURE 17-2: PIC16LF627A/628A/648A VOLTAGE-FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq \text{TA} \leq +85^{\circ}\text{C}$



# PIC16F627A/628A/648A

## 17.1 DC Characteristics: PIC16F627A/628A/648A (Industrial, Extended) PIC16LF627A/628A/648A (Industrial)

<b>PIC16LF627A/628A/648A</b> (Industrial)		<b>Standard Operating Conditions (unless otherwise stated)</b> Operating temperature $-40^{\circ}\text{C} \leq T_a \leq +85^{\circ}\text{C}$ for industrial					
<b>PIC16F627A/628A/648A</b> (Industrial, Extended)		<b>Standard Operating Conditions (unless otherwise stated)</b> Operating temperature $-40^{\circ}\text{C} \leq T_a \leq +85^{\circ}\text{C}$ for industrial and $-40^{\circ}\text{C} \leq T_a \leq +125^{\circ}\text{C}$ for extended					
Param No.	Sym	Characteristic/Device	Min	Typ†	Max	Units	Conditions
D001	VDD	<b>Supply Voltage</b>					
		PIC16LF627A/628A/648A	2.0	—	5.5	V	
		PIC16F627A/628A/648A	3.0	—	5.5	V	
D002	VDR	<b>RAM Data Retention Voltage<sup>(1)</sup></b>	—	1.5*	—	V	Device in SLEEP mode
D003	VPOR	<b>VDD Start Voltage</b> to ensure Power-on Reset	—	VSS	—	V	See Section 14.4 on Power-on Reset for details
D004	SVDD	<b>VDD Rise Rate</b> to ensure Power-on Reset	0.05*	—	—	V/ms	See Section 14.4 on Power-on Reset for details
D005	VBOR	<b>Brown-out Reset Voltage</b>	3.65	4.0	4.35	V	BOREN configuration bit is set
			3.65	4.0	4.4	V	

Legend: Rows with standard voltage device data only are shaded for improved readability.

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0 V, 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.

# PIC16F627A/628A/648A

## 17.2 DC Characteristics: PIC16F627A/628A/648A (Industrial) PIC16LF627A/628A/648A (Industrial)

		Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_a \leq +85^{\circ}\text{C}$ for industrial					
Param No.	LF and F Device Characteristics	Min†	Typ	Max	Units	Conditions	
						VDD	Note
<b>Supply Voltage (VDD)</b>							
D001	LF	2.0	—	5.5	V	—	
	LF/F	3.0	—	5.5	V	—	
<b>Power-down Base Current (IPD)</b>							
D020	LF	—	0.1	0.80	$\mu\text{A}$	2.0	WDT, BOR, Comparators, VREF, and T1OSC: disabled
	LF/F	—	0.1	0.85	$\mu\text{A}$	3.0	
		—	0.2	0.95	$\mu\text{A}$	5.0	
<b>Peripheral Module Current (<math>\Delta\text{IMOD}</math>)<sup>(1)</sup></b>							
D021	LF	—	1	2.0	$\mu\text{A}$	2.0	WDT Current
	LF/F	—	2	3.4	$\mu\text{A}$	3.0	
		—	9	12.5	$\mu\text{A}$	5.0	
D022	LF/F	—	32	TBD	$\mu\text{A}$	4.5	BOR Current
		—	33	TBD	$\mu\text{A}$	5.0	
D023	LF	—	15	TBD	$\mu\text{A}$	2.0	Comparator Current
	LF/F	—	27	TBD	$\mu\text{A}$	3.0	
		—	49	TBD	$\mu\text{A}$	5.0	
D024	LF	—	34	TBD	$\mu\text{A}$	2.0	VREF Current
	LF/F	—	50	TBD	$\mu\text{A}$	3.0	
		—	80	TBD	$\mu\text{A}$	5.0	
D025	LF	—	1.2	2.0	$\mu\text{A}$	2.0	T1Osc Current
	LF/F	—	1.3	2.2	$\mu\text{A}$	3.0	
		—	1.8	2.9	$\mu\text{A}$	5.0	
<b>Supply Current (IDD)</b>							
D010	LF	—	12	15	$\mu\text{A}$	2.0	Fosc = 32 kHz LP Oscillator Mode
	LF/F	—	21	25	$\mu\text{A}$	3.0	
		—	38	48	$\mu\text{A}$	5.0	
D011	LF	—	120	160	$\mu\text{A}$	2.0	Fosc = 1 MHz XT Oscillator Mode
	LF/F	—	180	250	$\mu\text{A}$	3.0	
		—	290	370	$\mu\text{A}$	5.0	
D012	LF	—	240	300	$\mu\text{A}$	2.0	Fosc = 4 MHz XT Oscillator Mode
	LF/F	—	370	470	$\mu\text{A}$	3.0	
		—	670	780	$\mu\text{A}$	5.0	
D013	LF/F	—	2.6	2.9	mA	4.5	Fosc = 20 MHz
		—	3	3.3	mA	5.0	HS Oscillator Mode

**Note 1:** 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. Max values should be used when calculating total current consumption.



# PIC16F627A/628A/648A

## 17.3 DC Characteristics: PIC16F627A/628A/648A (Extended)

		Standard Operating Conditions (unless otherwise stated)					
		Operating temperature $-40^{\circ}\text{C} \leq T_a \leq +125^{\circ}\text{C}$ for extended					
Param No.	Device Characteristics	Min†	Typ	Max	Units	Conditions	
						VDD	Note
<b>Supply Voltage (VDD)</b>							
D001	—	3.0	—	5.5	V	—	
<b>Power-down Base Current (IPD)</b>							
D020E	—	—	0.1	TBD	$\mu\text{A}$	3.0	WDT, BOR, Comparators, VREF, and T1OSC: disabled
		—	0.2	TBD	$\mu\text{A}$	5.0	
<b>Peripheral Module Current (<math>\Delta\text{I}_{\text{MOD}}</math>)<sup>(1)</sup></b>							
D021E	—	—	2	TBD	$\mu\text{A}$	3.0	WDT Current
		—	9	TBD	$\mu\text{A}$	5.0	
D022E	—	—	32	TBD	$\mu\text{A}$	4.5	BOR Current
		—	33	TBD	$\mu\text{A}$	5.0	
D023E	—	—	27	TBD	$\mu\text{A}$	3.0	Comparator Current
		—	49	TBD	$\mu\text{A}$	5.0	
D024E	—	—	50	TBD	$\mu\text{A}$	3.0	VREF Current
		—	83	TBD	$\mu\text{A}$	5.0	
D025E	—	—	1.3	TBD	$\mu\text{A}$	3.0	T1Osc Current
		—	1.8	TBD	$\mu\text{A}$	5.0	
<b>Supply Current (IDD)</b>							
D010E	—	—	21	TBD	$\mu\text{A}$	3.0	Fosc = 32 kHz LP Oscillator Mode
		—	38	TBD	$\mu\text{A}$	5.0	
D011E	—	—	182	TBD	$\mu\text{A}$	3.0	Fosc = 1 MHz XT Oscillator Mode
		—	293	TBD	$\mu\text{A}$	5.0	
D012E	—	—	371	TBD	$\mu\text{A}$	3.0	Fosc = 4 MHz XT Oscillator Mode
		—	668	TBD	$\mu\text{A}$	5.0	
D013E	—	—	2.6	TBD	$\text{mA}$	4.5	Fosc = 20 MHz HS Oscillator Mode
		—	3	TBD	$\text{mA}$	5.0	

**Note 1:** 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. Max values should be used when calculating total current consumption.

# PIC16F627A/628A/648A

## 17.4 DC Characteristics: PIC16F627A/628A/648A (Industrial, Extended) PIC16LF627A/628A/648A (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated)				
			Operating temperature -40°C ≤ TA ≤ +85°C for industrial and -40°C ≤ TA ≤ +125°C for extended				
			Operating voltage VDD range as described in DC spec Table 17-2 and Table 17-3				
Param. No.	Sym	Characteristic/Device	Min	Typ†	Max	Unit	Conditions
<b>Input Low Voltage</b>							
D030	VIL	I/O ports with TTL buffer	VSS	—	0.8	V	VDD = 4.5V to 5.5V otherwise <b>(Note1)</b>
			VSS	—	0.15 VDD	V	
		with Schmitt Trigger input <sup>(4)</sup>	VSS	—	0.2 VDD	V	
		MCLR, RA4/T0CKI, OSC1 (in RC mode)	VSS	—	0.2 VDD	V	
D033		OSC1 (in HS)	VSS	—	0.3 VDD	V	
		OSC1 (in LP and XT)	VSS	—	0.8	V	
<b>Input High Voltage</b>							
D040	VIH	I/O ports with TTL buffer	2.0 V	—	VDD	V	VDD = 4.5V to 5.5V otherwise <b>(Note1)</b>
			.25 VDD + 0.8 V	—	VDD	V	
		with Schmitt Trigger input <sup>(4)</sup>	0.8 VDD	—	VDD	V	
		MCLR RA4/T0CKI	0.8 VDD	—	VDD	V	
		OSC1 (XT, HS and LP)	0.7 VDD	—	VDD	V	
D043A		OSC1 (in RC mode)	0.9 VDD	—	VDD	V	
D070	IPURB	<b>PORTB weak pull-up current</b>	50	200	400	μA	VDD = 5.0V, VPIN = VSS
<b>Input Leakage Current<sup>(2), (3)</sup></b>							
D060	IIL	I/O ports (Except PORTA)	—	—	±1.0	μA	VSS ≤ VPIN ≤ VDD, pin at hi-impedance VSS ≤ VPIN ≤ VDD, pin at hi-impedance VSS ≤ VPIN ≤ VDD VSS ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
		PORTA <sup>(4)</sup>	—	—	±0.5	μA	
		RA4/T0CKI	—	—	±1.0	μA	
		OSC1, MCLR	—	—	±5.0	μA	
<b>Output Low Voltage</b>							
D080	VOL	I/O ports <sup>(4)</sup>	—	—	0.6	V	IOL=8.5 mA, VDD=4.5 V, -40° to +85°C IOL=7.0 mA, VDD=4.5 V, +85° to +125°C
			—	—	0.6	V	
<b>Output High Voltage<sup>(3)</sup></b>							
D090	VOH	I/O ports (Except RA4 <sup>(4)</sup> )	VDD-0.7	—	—	V	IOH=-3.0 mA, VDD=4.5 V, -40° to +85°C IOH=-2.5 mA, VDD=4.5 V, +85° to +125°C
			VDD-0.7	—	—	V	
D150	VOd	<b>Open-Drain High Voltage</b>	—	—	8.5*	V	RA4 pin PIC16F627A/628A/648A, PIC16LF627A/628A/648A
<b>Capacitive Loading Specs on Output Pins</b>							
D100*	COSC2	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 RC 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 RC oscillator configuration, the OSC1 pin is a Schmitt Trigger input. It is not recommended that the PIC16F627A/628A/648A be driven with external clock in RC 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.
- 4: Includes OSC1 and OSC2 when configured as I/O pins, CLKIN, or CLKOUT.

# PIC16F627A/628A/648A

**TABLE 17-1: DC Characteristics: PIC16F627A/628A/648A (Industrial, Extended)  
PIC16LF627A/628A/648A (Industrial)**

DC Characteristics			Standard Operating Conditions (unless otherwise stated)				
			Operating temperature $-40^{\circ}\text{C} \leq \text{TA} \leq +85^{\circ}\text{C}$ for industrial and $-40^{\circ}\text{C} \leq \text{TA} \leq +125^{\circ}\text{C}$ for extended				
			Operating voltage $V_{\text{DD}}$ range as described in DC spec Table 17-2 and Table 17-3				
Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
<b>Data EEPROM Memory</b>							
D120	ED	Endurance	100K	1M	—	E/W	$-40^{\circ}\text{C} \leq \text{TA} \leq 85^{\circ}\text{C}$
D120A	ED	Endurance	10K	100K	—	E/W	$85^{\circ}\text{C} \leq \text{TA} \leq 125^{\circ}\text{C}$
D121	VDRW	$V_{\text{DD}}$ for read/write	$V_{\text{MIN}}$	—	5.5	V	$V_{\text{MIN}}$ = Minimum operating voltage
D122	TDEW	Erase/Write cycle time	—	4	8*	ms	
D123	TRETD	Characteristic Retention	100	—	—	Year	Provided no other specifications are violated
D124	TREF	Number of Total Erase/Write Cycles before Refresh <sup>(1)</sup>	1M	10M	—	E/W	$-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
<b>Program FLASH Memory</b>							
D130	EP	Endurance	10K	100K	—	E/W	$-40^{\circ}\text{C} \leq \text{TA} \leq 85^{\circ}\text{C}$
D130A	EP	Endurance	1000	10K	—	E/W	$85^{\circ}\text{C} \leq \text{TA} \leq 125^{\circ}\text{C}$
D131	VPR	$V_{\text{DD}}$ for read	$V_{\text{MIN}}$	—	5.5	V	$V_{\text{MIN}}$ = Minimum operating voltage
D132	VIE	$V_{\text{DD}}$ for Block erase	4.5	—	5.5	V	
D132A	VPEW	$V_{\text{DD}}$ for write	$V_{\text{MIN}}$	—	5.5	V	$V_{\text{MIN}}$ = Minimum operating voltage
D133	TIE	Block Erase cycle time	—	4	8*	ms	$V_{\text{DD}} > 4.5\text{V}$
D133A	TPEW	Write cycle time	—	2	4*	ms	
D134	TRETP	Characteristic Retention	100	—	—	year	Provided no other specifications are violated

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0 V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Refer to Section 13.7 for a more detailed discussion on data EEPROM endurance.

# PIC16F627A/628A/648A

**TABLE 17-2: COMPARATOR SPECIFICATIONS**

Operating Conditions: 2.0V < 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	
D303	Response Time <sup>(1)</sup>	TRESP	—	300	400*	ns	VDD = 3.0V to 5.5V -40° to +85°C VDD = 3.0V to 5.5V -85° to +125°C VDD = 2.0V to 3.0V -40° to +85°C
			—	400	600*	ns	
			—	400	600*	ns	
D304	Comparator Mode Change to Output Valid	TMC2OV	—	300	10*	µs	

\* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at (VDD - 1.5)/2 while the other input transitions from VSS to VDD.

**TABLE 17-3: VOLTAGE REFERENCE SPECIFICATIONS**

Operating Conditions: 2.0V < 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	LSb	Low Range (VRR = 1) High Range (VRR = 0)
					VDD/32	LSb	
D311	Absolute Accuracy	VRAA	—	—	1/4 <sup>(2)*</sup>	LSb	Low Range (VRR = 1) High Range (VRR = 0)
					1/2 <sup>(2)*</sup>	LSb	
D312	Unit Resistor Value (R)	VRUR	—	2k*	—	Ω	
D313	Settling Time <sup>(1)</sup>	TSET	—	—	10*	µs	

\* These parameters are characterized but not tested.

**Note 1:** Settling time measured while VRR = 1 and VR<3:0> transitions from 0000 to 1111.

**Note 2:** When VDD is between 2.0V and 3.0V the VREF output voltage levels on RA2 described by the equation: [VDD/2 ± (3-VDD)/2] may cause the Absolute Accuracy (VRAA) of the VREF output signal on RA2 to be greater than the stated max.

## 17.5 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

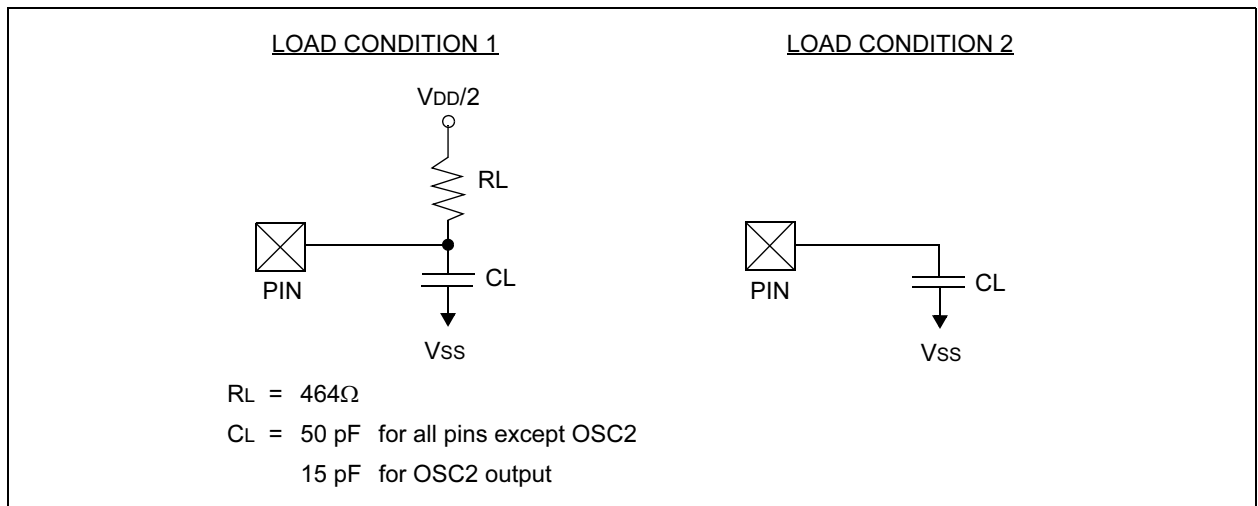
Lowercase subscripts (pp) and their meanings:

<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 17-3: LOAD CONDITIONS**



# PIC16F627A/628A/648A

## 17.6 Timing Diagrams and Specifications

FIGURE 17-4: EXTERNAL CLOCK TIMING

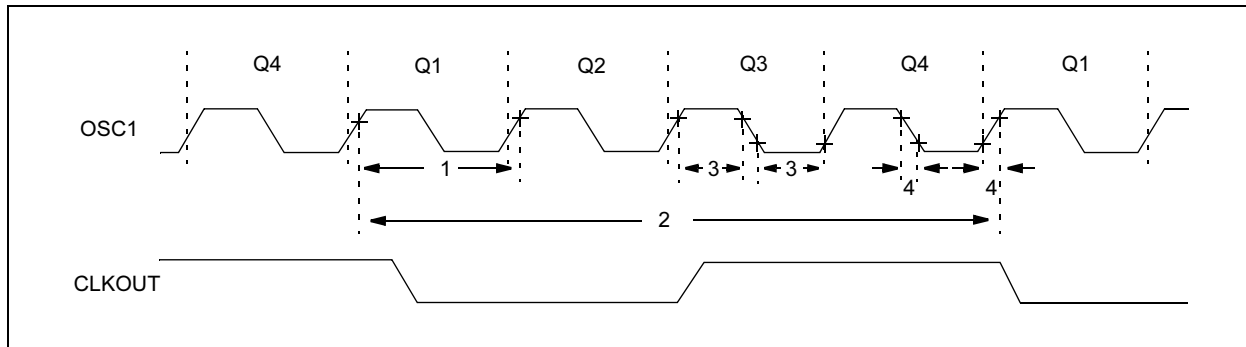


TABLE 17-4: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	External CLKIN Frequency <sup>(1)</sup>	DC	—	4	MHz	XT and RC Osc mode, VDD = 5.0 V
			DC	—	20	MHz	HS Osc mode
			DC	—	200	kHz	LP Osc mode
		Oscillator Frequency <sup>(1)</sup>	—	—	4	MHz	RC Osc mode, VDD = 5.0V
			0.1	—	4	MHz	XT Osc mode
			1	—	20	MHz	HS Osc mode
			—	—	200	kHz	LP Osc mode
			—	4	—	MHz	INTOSC mode (fast)
			—	37	—	kHz	INTOSC mode (slow)
1	Tosc	External CLKIN Period <sup>(1)</sup>	250	—	—	ns	XT and RC Osc mode
			50	—	—	ns	HS Osc mode
			5	—	—	μs	LP Osc mode
		Oscillator Period <sup>(1)</sup>	250	—	—	ns	RC Osc mode
			250	—	10,000	ns	XT Osc mode
			50	—	1,000	ns	HS Osc mode
			5	—	—	μs	LP Osc mode
			—	250	—	ns	INTOSC mode (fast)
—	27	—	μs	INTOSC mode (slow)			
2	Tcy	Instruction Cycle Time	1.0	Tcy	DC	ns	Tcy = 4/Fosc
3	TosL, TosH	External CLKIN (OSC1) High External CLKIN Low	100*	—	—	ns	XT oscillator, TOSC L/H duty cycle
4	RC	External Biased RC Fre- quency	10 kHz*	—	4 MHz	—	VDD = 5.0V

\* 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.

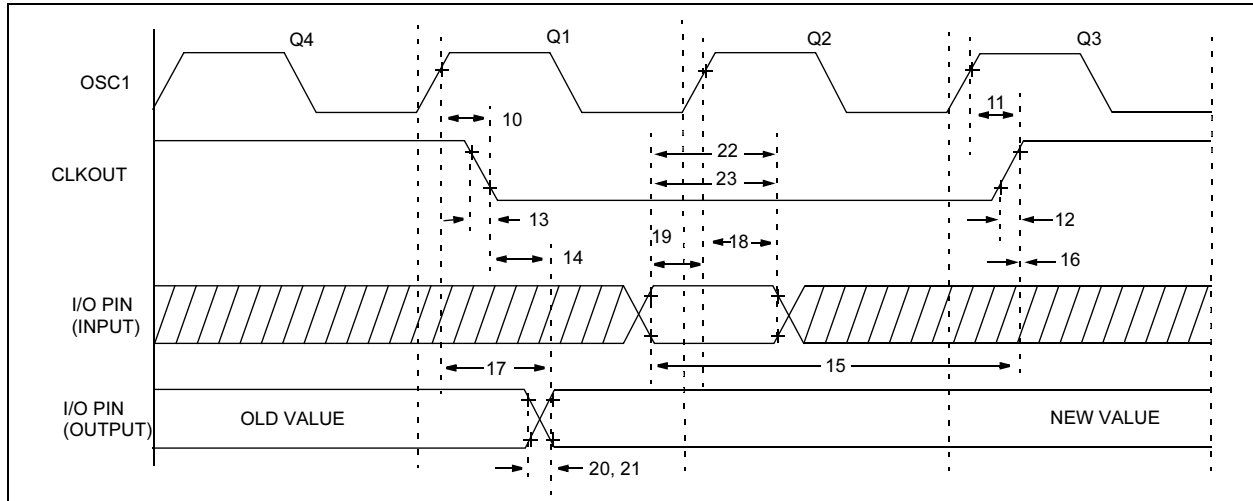
**Note:** Instruction cycle period (Tcy) equals four times the input oscillator time-based period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "Min" values with an external clock applied to the OSC1 pin. When an external clock input is used, the "Max" cycle time limit is "DC" (no clock) for all devices.

# PIC16F627A/628A/648A

**TABLE 17-5: PRECISION INTERNAL OSCILLATOR PARAMETERS**

Parameter No.	Sym	Characteristic	Min	Typ	Max	Units	Conditions
F10	F1OSC	Oscillator Center frequency	—	4	—	MHz	
F13	$\Delta$ Iosc	Oscillator Stability (jitter)	—	—	$\pm 1$	%	$V_{DD} = 3.5\text{ V}$ , $25^\circ\text{C}$
			—	—	$\pm 2$	%	$2.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ $0^\circ\text{C} \leq T_A \leq +85^\circ\text{C}$
			—	—	$\pm 5$	%	$2.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ $-40^\circ\text{C} \leq T_A \leq +85^\circ\text{C}$ (IND) $-40^\circ\text{C} \leq T_A \leq +125^\circ\text{C}$ (EXT)
F14	T1OSCST	Oscillator Wake-up from SLEEP start-up time	—	6	TBD	$\mu\text{s}$	$V_{DD} = 2.0\text{V}$ , $-40^\circ\text{C}$ to $+85^\circ\text{C}$
			—	4	TBD	$\mu\text{s}$	$V_{DD} = 3.0\text{V}$ , $-40^\circ\text{C}$ to $+85^\circ\text{C}$
			—	3	TBD	$\mu\text{s}$	$V_{DD} = 5.0\text{V}$ , $-40^\circ\text{C}$ to $+85^\circ\text{C}$

**FIGURE 17-5: CLKOUT AND I/O TIMING**



**TABLE 17-6: CLKOUT AND I/O TIMING REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	
10 10A	TosH2ckL	OSC1 $\uparrow$ to CLKOUT $\downarrow$	PIC16F62X	—	75	200*	ns
			PIC16LF62X	—	—	400*	ns
11 11A	TosH2ckH	OSC1 $\uparrow$ to CLKOUT $\uparrow$	PIC16F62X	—	75	200*	ns
			PIC16LF62X	—	—	400*	ns
12 12A	TckR	CLKOUT rise time	PIC16F62X	—	35	100*	ns
			PIC16LF62X	—	—	200*	ns
13 13A	TckF	CLKOUT fall time	PIC16F62X	—	35	100*	ns
			PIC16LF62X	—	—	200*	ns
14	TckL2ioV	CLKOUT $\downarrow$ to Port out valid	—	—	20*	ns	
15	TioV2ckH	Port in valid before	PIC16F62X	Tosc+200 ns*	—	—	ns
18	TosH2ioI	OSC1 $\uparrow$ (Q2 cycle) to Port input invalid (I/O in hold time)	—	100*	—	—	ns
			—	200*	—	—	ns

\* 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.

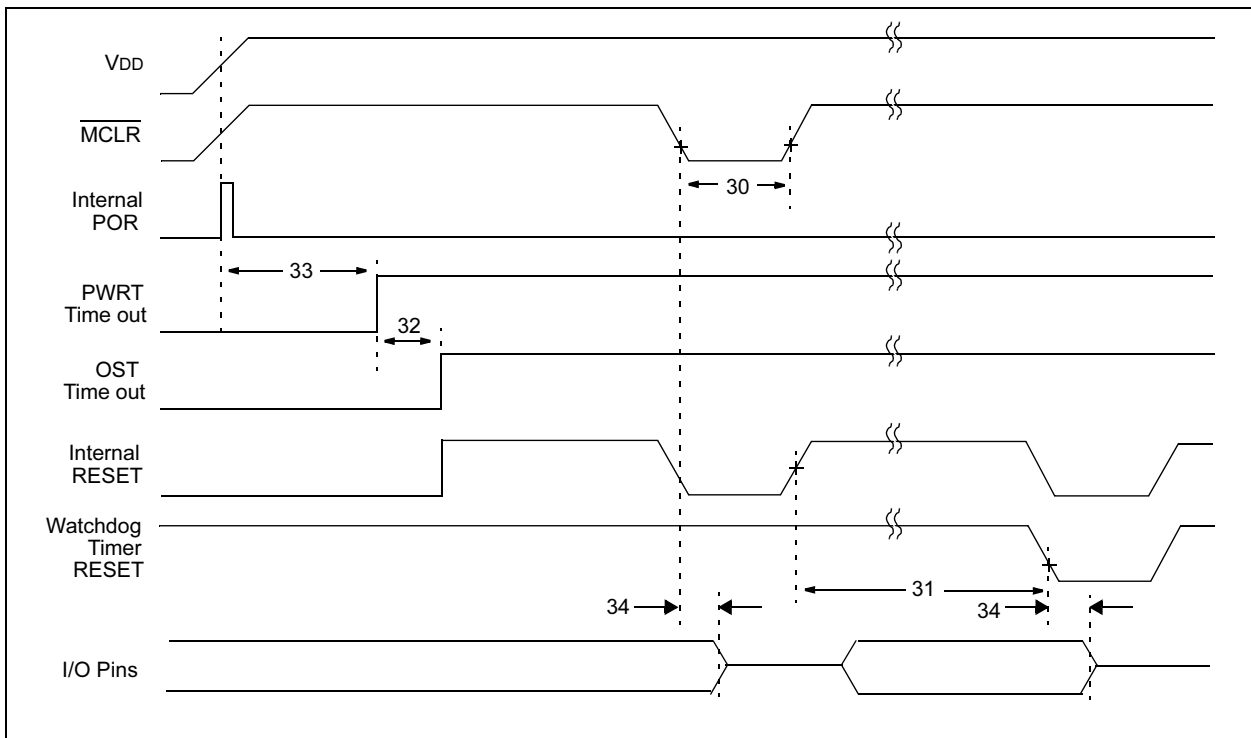
# PIC16F627A/628A/648A

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units
		CLKOUT ↑	PIC16LF62X	$T_{osc}+400$ ns*	—	—	ns
16	TckH2iol	Port in hold after CLKOUT ↑		0	—	—	ns
17	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid	PIC16F62X	—	50	150*	ns
			PIC16LF62X	—	—	300*	ns
18	TosH2iol	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)		100* 200*	—	—	ns

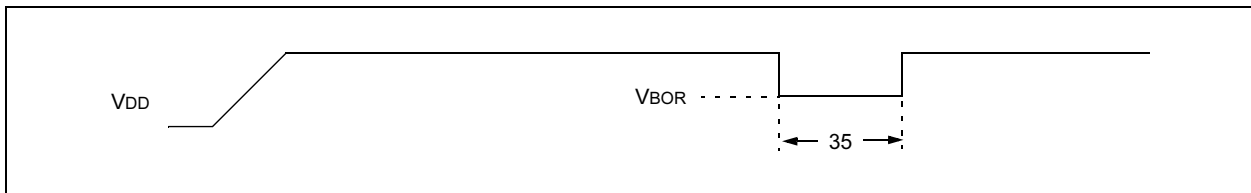
\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 17-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**FIGURE 17-7: BROWN-OUT DETECT TIMING**





# PIC16F627A/628A/648A

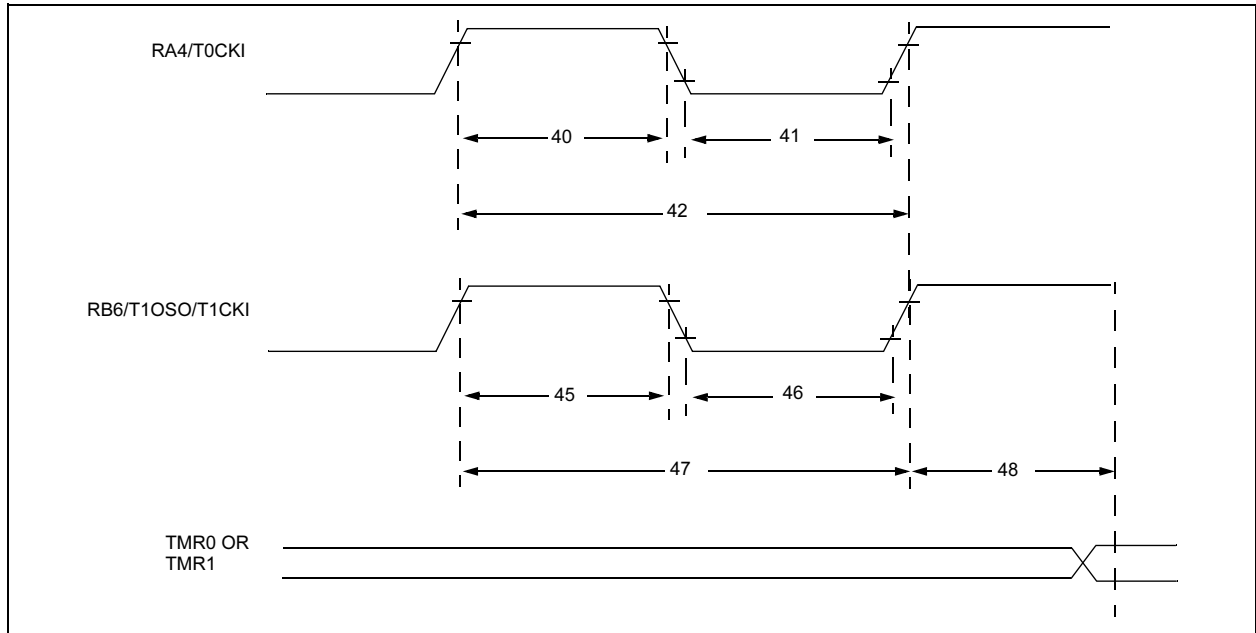
**TABLE 17-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	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 Extended temperature
34	TIOZ	I/O Hi-impedance from MCLR Low or Watchdog Timer Reset	—	—	2.0*	μs	
35	TBOR	Brown-out Reset pulse width	100*	—	—	μs	VDD ≤ VBOR (D005)

\* 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.

**FIGURE 17-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**



# PIC16F627A/628A/648A

**TABLE 17-9: 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, with Prescaler	PIC16F62X	$15^*$	—	—	ns	
				PIC16LF62X	$25^*$	—	—	ns	
			Asynchronous	PIC16F62X	$30^*$	—	—	ns	
			PIC16LF62X	$50^*$	—	—	ns		
46	Tt1L	T1CKI Low Time	Synchronous, No Prescaler	$0.5T_{CY} + 20^*$	—	—	ns		
			Synchronous, with Prescaler	PIC16F62X	$15^*$	—	—	ns	
				PIC16LF62X	$25^*$	—	—	ns	
			Asynchronous	PIC16F62X	$30^*$	—	—	ns	
			PIC16LF62X	$50^*$	—	—	ns		
47	Tt1P	T1CKI input period	Synchronous	PIC16F62X	Greater of: $\frac{T_{CY} + 40^*}{N}$	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16LF62X	Greater of: $\frac{T_{CY} + 40^*}{N}$	—	—	—	
			Asynchronous	PIC16F62X	$60^*$	—	—	ns	
				PIC16LF62X	$100^*$	—	—	ns	
	Ft1	Timer1 oscillator input frequency range (oscillator enabled by setting bit T1OSCEN)		—	$32.7^{(1)}$	—	kHz		
48	TCKEZt mr1	Delay from external clock edge to timer increment		$2T_{osc}$	—	$7T_{osc}$	—		

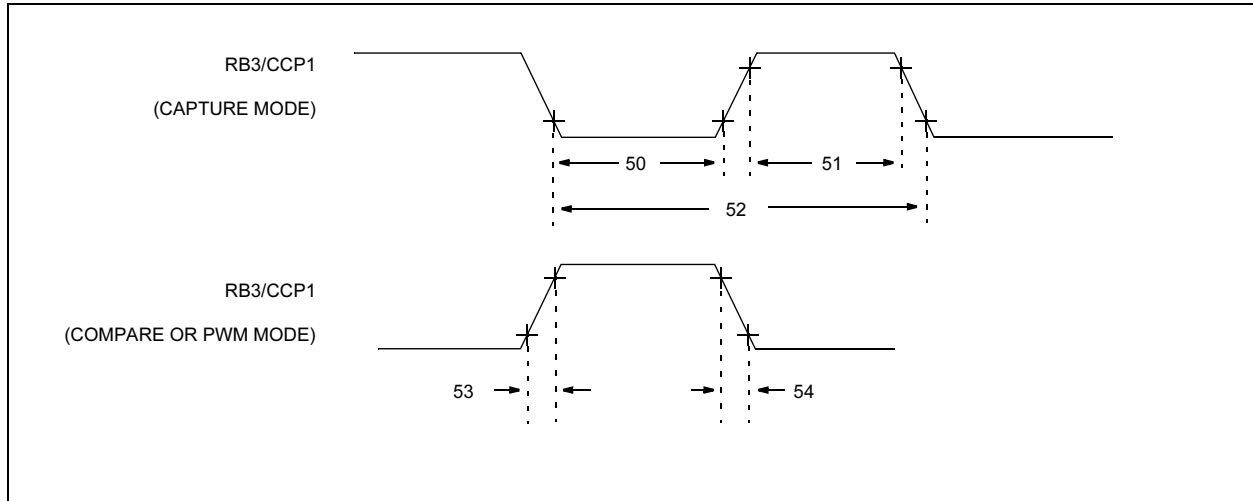
\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** This oscillator is intended to work only with 32.768 kHz watch crystals and their manufactured tolerances. Higher value crystal frequencies may not be compatible with this crystal driver.

# PIC16F627A/628A/648A

**FIGURE 17-10: CAPTURE/COMPARE/PWM TIMINGS**



**TABLE 17-8: CAPTURE/COMPARE/PWM REQUIREMENTS**

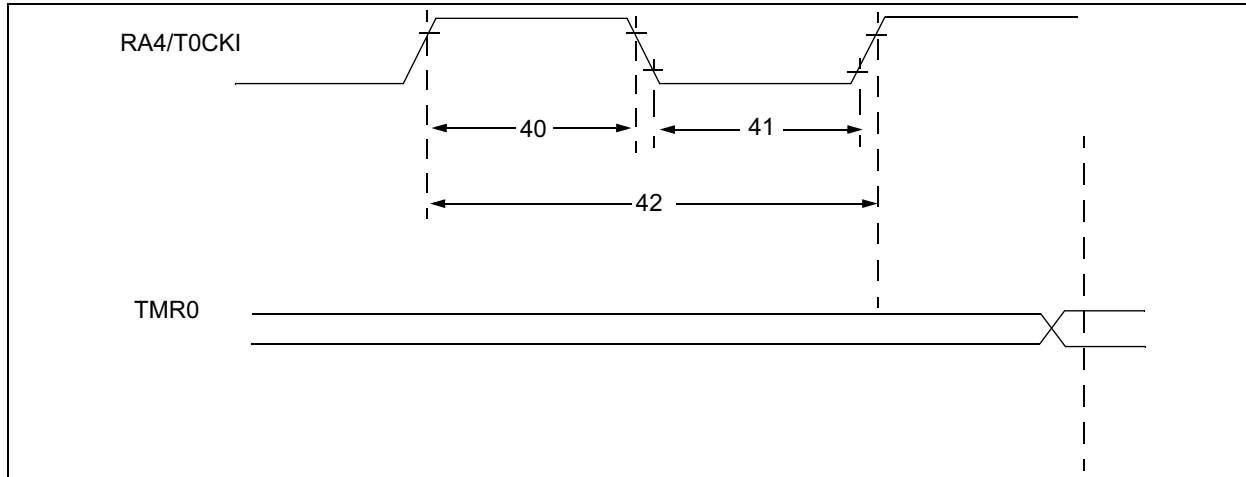
Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
50	TccL	CCP input low time	No Prescaler	$0.5T_{CY} + 20^*$	—	—	ns	
			With Prescaler	PIC16F62X	10*	—	—	
51	TccH	CCP input high time	No Prescaler	$0.5T_{CY} + 20^*$	—	—	ns	
			With Prescaler	PIC16F62X	10*	—	—	
52	TccP	CCP input period		$\frac{3T_{CY} + 40^*}{N}$	—	—	ns	N = prescale value (1,4 or 16)
53	TccR	CCP output rise time	PIC16F62X		10	25*	ns	
			PIC16LF62X		25	45*	ns	
54	TccF	CCP output fall time	PIC16F62X		10	25*	ns	
			PIC16LF62X		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.

# PIC16F627A/628A/648A

**FIGURE 17-11: 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 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Not Available at this time.

# PIC16F627A/628A/648A

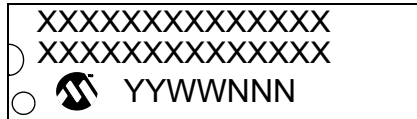
---

NOTES:

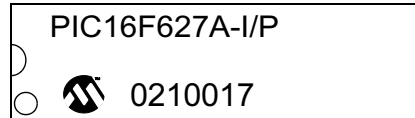
## 19.0 PACKAGING INFORMATION

### 19.1 Package Marking Information

#### 18-LEAD PDIP (.300")



#### EXAMPLE



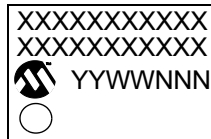
#### 18-LEAD SOIC (.300")



#### EXAMPLE



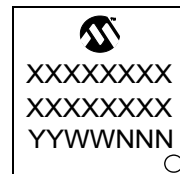
#### 20-LEAD SSOP



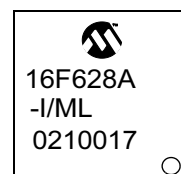
#### EXAMPLE



#### 28-LEAD QFN



#### EXAMPLE



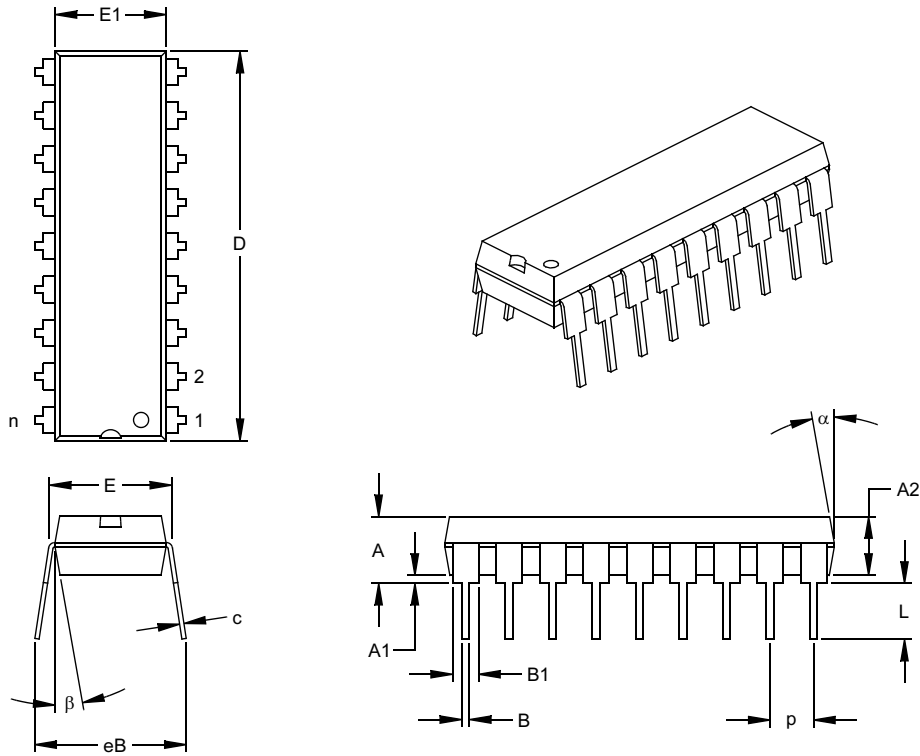
Legend: XX...X	Customer specific information*
YY	Year code (last 2 digits of calendar year)
WW	Week code (week of January 1 is week '01')
NNN	Alphanumeric traceability code

**Note:** In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

\* Standard PICmicro device marking consists of Microchip part number, year code, week code, and traceability code. For PICmicro device 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.

# PIC16F627A/628A/648A

## 18-Lead Plastic Dual In-line (P) – 300 mil (PDIP)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.100			2.54	
Top to Seating Plane	A	.140	.155	.170	3.56	3.94	4.32
Molded Package Thickness	A2	.115	.130	.145	2.92	3.30	3.68
Base to Seating Plane	A1	.015			0.38		
Shoulder to Shoulder Width	E	.300	.313	.325	7.62	7.94	8.26
Molded Package Width	E1	.240	.250	.260	6.10	6.35	6.60
Overall Length	D	.890	.898	.905	22.61	22.80	22.99
Tip to Seating Plane	L	.125	.130	.135	3.18	3.30	3.43
Lead Thickness	c	.008	.012	.015	0.20	0.29	0.38
Upper Lead Width	B1	.045	.058	.070	1.14	1.46	1.78
Lower Lead Width	B	.014	.018	.022	0.36	0.46	0.56
Overall Row Spacing	§ eB	.310	.370	.430	7.87	9.40	10.92
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

\* Controlling Parameter

§ Significant Characteristic

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

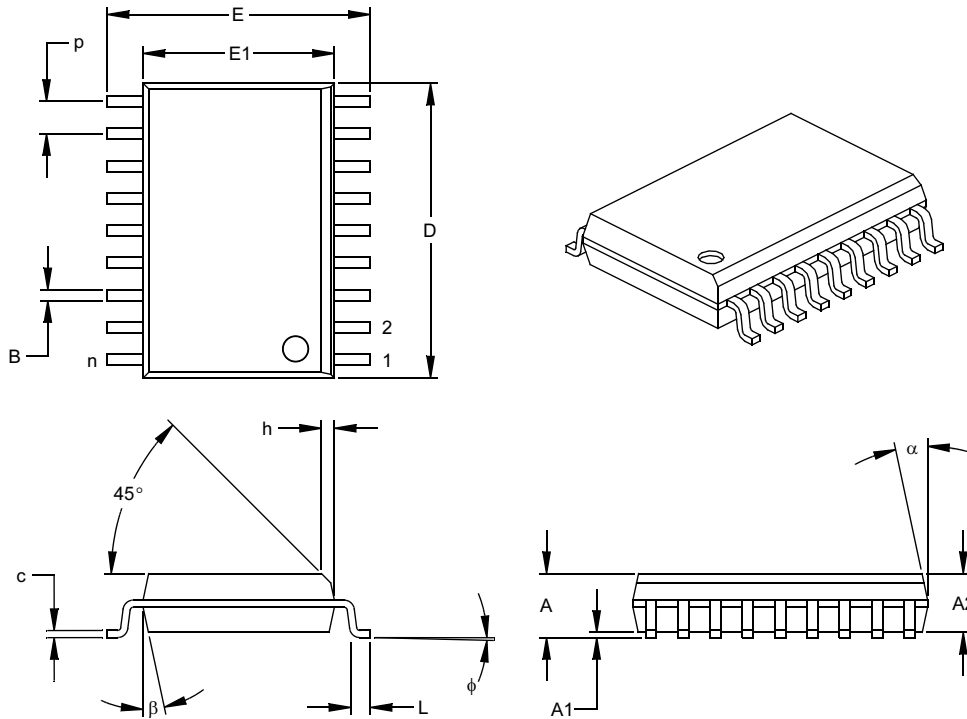
JEDEC Equivalent: MS-001

Drawing No. C04-007



# PIC16F627A/628A/648A

## 18-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.050			1.27	
Overall Height	A	.093	.099	.104	2.36	2.50	2.64
Molded Package Thickness	A2	.088	.091	.094	2.24	2.31	2.39
Standoff §	A1	.004	.008	.012	0.10	0.20	0.30
Overall Width	E	.394	.407	.420	10.01	10.34	10.67
Molded Package Width	E1	.291	.295	.299	7.39	7.49	7.59
Overall Length	D	.446	.454	.462	11.33	11.53	11.73
Chamfer Distance	h	.010	.020	.029	0.25	0.50	0.74
Foot Length	L	.016	.033	.050	0.41	0.84	1.27
Foot Angle	φ	0	4	8	0	4	8
Lead Thickness	c	.009	.011	.012	0.23	0.27	0.30
Lead Width	B	.014	.017	.020	0.36	0.42	0.51
Mold Draft Angle Top	α	0	12	15	0	12	15
Mold Draft Angle Bottom	β	0	12	15	0	12	15

\* Controlling Parameter

§ Significant Characteristic

Notes:

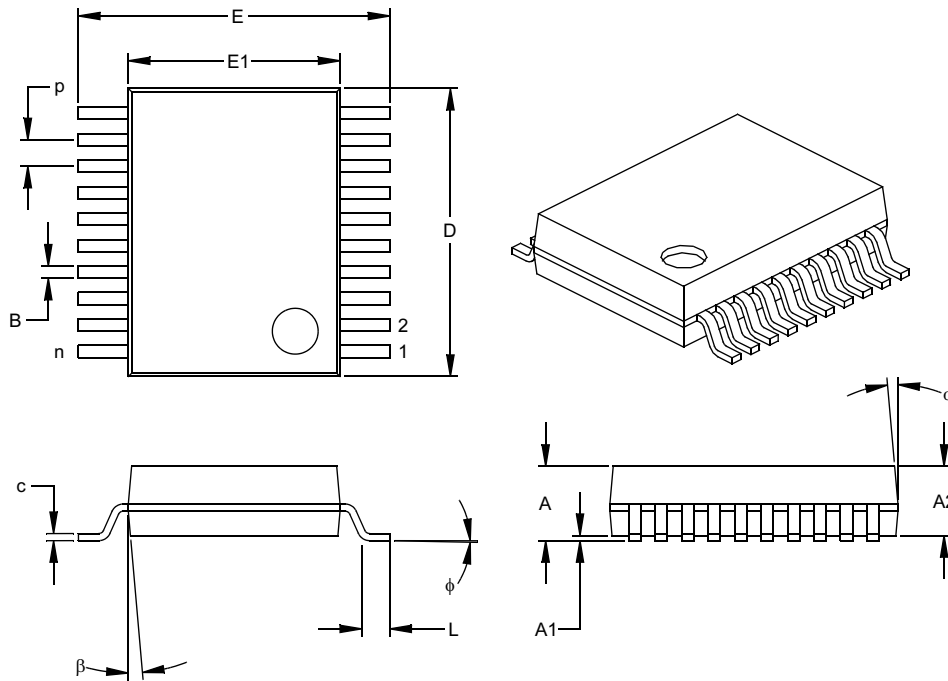
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-013

Drawing No. C04-051

# PIC16F627A/628A/648A

20-Lead Plastic Shrink Small Outline (SS) – 209 mil, 5.30 mm (SSOP)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		20			20	
Pitch	p		.026			0.65	
Overall Height	A	.068	.073	.078	1.73	1.85	1.98
Molded Package Thickness	A2	.064	.068	.072	1.63	1.73	1.83
Standoff §	A1	.002	.006	.010	0.05	0.15	0.25
Overall Width	E	.299	.309	.322	7.59	7.85	8.18
Molded Package Width	E1	.201	.207	.212	5.11	5.25	5.38
Overall Length	D	.278	.284	.289	7.06	7.20	7.34
Foot Length	L	.022	.030	.037	0.56	0.75	0.94
Lead Thickness	c	.004	.007	.010	0.10	0.18	0.25
Foot Angle	φ	0	4	8	0.00	101.60	203.20
Lead Width	B	.010	.013	.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

§ Significant Characteristic

Notes:

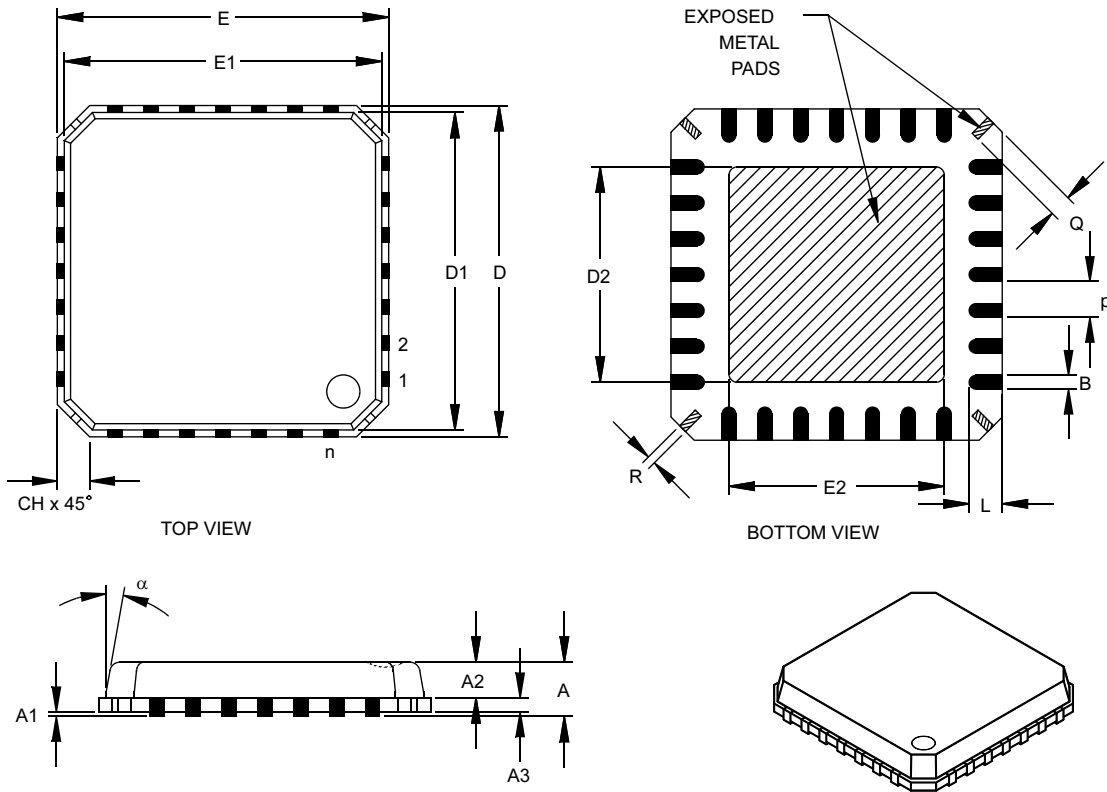
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-150

Drawing No. C04-072

# PIC16F627A/628A/648A

## 28-Lead Plastic Quad Flat No Lead Package (ML) 6x6 mm Body (QFN)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		28			28	
Pitch	p	.026 BSC			0.65 BSC		
Overall Height	A		.033	.039		0.85	1.00
Molded Package Thickness	A2		.026	.031		0.65	0.80
Standoff	A1	.000	.0004	.002	0.00	0.01	0.05
Base Thickness	A3	.008 REF.			0.20 REF.		
Overall Width	E	.236 BSC			6.00 BSC		
Molded Package Width	E1	.226 BSC			5.75 BSC		
Exposed Pad Width	E2	.140	.146	.152	3.55	3.70	3.85
Overall Length	D	.236 BSC			6.00 BSC		
Molded Package Length	D1	.226 BSC			5.75 BSC		
Exposed Pad Length	D2	.140	.146	.152	3.55	3.70	3.85
Lead Width	B	.009	.011	.014	0.23	0.28	0.35
Lead Length	L	.020	.024	.030	0.50	0.60	0.75
Tie Bar Width	R	.005	.007	.010	0.13	0.17	0.23
Tie Bar Length	Q	.012	.016	.026	0.30	0.40	0.65
Chamfer	CH	.009	.017	.024	0.24	0.42	0.60
Mold Draft Angle Top	α			12°			12°

\*Controlling Parameter

**Notes:**

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC equivalent: M0-220

Drawing No. C04-114

# PIC16F627A/628A/648A

---

---

NOTES:

# PIC16F627A/628A/648A

---

## APPENDIX A: DATA SHEET REVISION HISTORY

### Revision A

This is a new data sheet.

## APPENDIX B: DEVICE DIFFERENCES

The differences between the PIC16F627A/628A/648A devices listed in this data sheet are shown in Table B-1.

**TABLE B-1: DEVICE DIFFERENCES**

Device	Memory		
	FLASH Program	RAM Data	EEPROM Data
PIC16F627A	1024 x 14	224 x 8	128 x 8
PIC16F628A	2048 x 14	224 x 8	128 x 8
PIC16F648A	4096 x 14	256 x 8	256 x 8

# PIC16F627A/628A/648A

---

## APPENDIX C: DEVICE MIGRATIONS

This section describes the functional and electrical specification differences when migrating between functionally similar devices. (such as from a PIC16F627 to a PIC16F627A).

### C.1 PIC16F627/628 to a PIC16F627A/628A

1. ER mode is now RC mode.
2. Code Protection for the Program Memory has changed from Code Protect sections of memory to Code Protect of the whole memory. The Configuration bits CP0 and CP1 in the PIC16F627/628 do not exist in the PIC16F627A/628A. They have been replaced with one Configuration bit <13> CP.
3. “Brown-out Detect (BOD)” terminology has changed to “Brown-out Reset (BOR)” to better represent the function of the Brown-out circuitry.
4. Enabling Brown-out Reset (BOR) does not automatically enable the Power-up Timer (PWRT) the way it did in the PIC16F627/628.
5. INTRC is now called INTOSC.
6. Timer1 Oscillator is now designed for 32.768 kHz operation. In the PIC16F627/628 the Timer1 Oscillator was designed to run up to 200 kHz.
7. The Dual Speed Oscillator mode only works in the INTOSC Oscillator mode. In the PIC16F627/628 the Dual Speed Oscillator mode worked in both the INTRC and ER Oscillator modes.

## APPENDIX D: MIGRATING FROM OTHER PICmicro DEVICES

This discusses some of the issues in migrating from other PICmicro devices to the PIC16F627A/628A/648A family of devices.

### D.1 PIC16C62X/CE62X to PIC16F627A/628A/648A Migration

See Microchip web site for availability ([www.microchip.com](http://www.microchip.com)).

### D.2 PIC16C622A to PIC16F627A/628A/648A Migration

See Microchip web site for availability ([www.microchip.com](http://www.microchip.com)).

**Note:** This device has been designed to perform to the parameters of its data sheet. It has been tested to an electrical specification designed to determine its conformance with these parameters. Due to process differences in the manufacture of this device, this device may have different performance characteristics than its earlier version. These differences may cause this device to perform differently in your application than the earlier version of this device.

## APPENDIX E: DEVELOPMENT TOOL VERSION REQUIREMENTS

This lists the minimum requirements (software/firmware) of the specified development tool to support the devices listed in this data sheet.

**MPLAB® IDE:** TBD

**MPLAB® SIMULATOR:** TBD

**MPLAB® ICE 3000:**

PIC16F627A/628A/648A Processor Module:

Part Number - TBD

PIC16F627A/628A/648A Device Adapter:

Socket Part Number

18-pin PDIP TBD

18-pin SOIC TBD

20-pin SSOP TBD

28-pin QFN TBD

**MPLAB® ICD:** TBD

**PRO MATE® II:** TBD

**PICSTART® Plus:** TBD

**MPASM™ Assembler:** TBD

**MPLAB® C18 C Compiler:** TBD

<p><b>Note:</b> Please read all associated README.TXT files that are supplied with the development tools. These "read me" files will discuss product support and any known limitations.</p>
---

# PIC16F627A/628A/648A

---

---

NOTES:



## ON-LINE SUPPORT

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

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

### Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

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

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

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

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

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

## SYSTEMS INFORMATION AND UPGRADE HOT LINE

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

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

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

# PIC16F627A/628A/648A

---

---

## 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 (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager  
RE: Reader Response  
From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: PIC16F627A/628A/648A Literature Number: DS40044A

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 document easy to follow? If not, why?

---

---

4. What additions to the document do you think would enhance the structure and subject?

---

---

5. What deletions from the document 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?

---

---

# PIC16F627A/628A/648A

## INDEX

### A

A/D	
Special Event Trigger (CCP)	57
Absolute Maximum Ratings	131
ADDLW Instruction	113
ADDWF Instruction	113
ANDLW Instruction	113
ANDWF Instruction	113
Architectural Overview	9
Assembler	
MPASM Assembler	125

### B

Baud Rate Error	71
Baud Rate Formula	71
BCF Instruction	114
Block Diagrams	
Comparator	
I/O Operating Modes	62
Modified Comparator Output	64
I/O Ports	
RB0/INT Pin	37
RB1/RX/DT Pin	37
RB2/TX/CK Pin	38
RB3/CCP1 Pin	38
RB4/PGM Pin	39
RB5 Pin	40
RB6/T1OSO/T1CKI Pin	41
RB7/T1OSI Pin	42
RC Oscillator Mode	96
USART Receive	79
USART Transmit	77
BRGH bit	71
Brown-Out Reset (BOR)	98
BSF Instruction	114
BTFSC Instruction	114
BTFSS Instruction	115

### C

CALL Instruction	115
Capture (CCP Module)	56
Block Diagram	56
CCP Pin Configuration	56
CCPR1H:CCPR1L Registers	56
Changing Between Capture Prescalers	56
Prescaler	56
Software Interrupt	56
Timer1 Mode Selection	56
Capture/Compare/PWM (CCP)	55
Capture Mode. See Capture	
CCP1	55
CCPR1H Register	55
CCPR1L Register	55
CCP2	55
Compare Mode. See Compare	
PWM Mode. See PWM	
Timer Resources	55
CCP1CON Register	
CCP1M3:CCP1M0 Bits	55
CCP1X:CCP1Y Bits	55
CCP2CON Register	
CCP2M3:CCP2M0 Bits	55
CCP2X:CCP2Y Bits	55

Clocking Scheme/Instruction Cycle	13
CLRF Instruction	115
CLRWF Instruction	116
CLRWDI Instruction	116
Code Examples	
Data EEPROM Refresh Routine	92
Code Protection	108
COMF Instruction	116
Comparator	
Block Diagrams	
I/O Operating Modes	62
Modified Comparator Output	64
Comparator Module	61
Configuration	62
Interrupts	65
Operation	63
Reference	63
Compare (CCP Module)	56
Block Diagram	56
CCP Pin Configuration	57
CCPR1H:CCPR1L Registers	56
Software Interrupt	57
Special Event Trigger	57
Timer1 Mode Selection	57
Configuration Bits	93
Crystal Operation	95

### D

Data EEPROM Memory	89
EECON1 Register	89
EECON2 Register	89
Operation During Code Protection	92
Reading	91
Spurious Write Protection	91
Using	92
Write Verify	91
Writing to	91
Data Memory Organization	15
DECIF Instruction	116
DECFSZ Instruction	117
Development Support	125
Development Tool Version Requirements	157
Device Differences	155
Device Migrations	156
Dual-speed Oscillator Modes	97

### E

EECON1 register	90
EECON2 register	90
Errata	3
External Crystal Oscillator Circuit	95

### G

General-Purpose Register File	15
GOTO Instruction	117

### I

I/O Ports	31
Bi-Directional	44
Block Diagrams	
RB0/INT Pin	37
RB1/RX/DT Pin	37
RB2/TX/CK Pin	38
RB3/CCP1 Pin	38
RB4/PGM Pin	39

# PIC16F627A/628A/648A

RB5 Pin.....	40	CCP1 Flag (CCP1IF Bit).....	56
RB6/T1OSO/T1CKI Pin .....	41	IORLW Instruction .....	119
RB7/T1OSI Pin .....	42	IORWF Instruction .....	119
PORTA .....	31	<b>K</b>	
PORTB.....	36	KeelQ Evaluation and Programming Tools .....	128
Programming Considerations .....	44	<b>M</b>	
Successive Operations .....	44	Memory Organization	
TRISA .....	31	Data EEPROM Memory.....	89, 91, 92
TRISB .....	36	Migrating from other PICmicro Devices .....	156
ICEPIC In-Circuit Emulator .....	126	MOVF Instruction.....	119
ID Locations .....	108	MOVLW Instruction.....	119
INCF Instruction .....	118	MOVWF Instruction .....	120
INCFSZ Instruction.....	118	MPLAB C17 and MPLAB C18 C Compilers .....	125
In-Circuit Serial Programming.....	109	MPLAB ICD In-Circuit Debugger .....	127
Indirect Addressing, INDF and FSR Registers.....	28	MPLAB ICE High Performance Universal In-Circuit Emulator with MPLAB IDE .....	126
Instruction Flow/Pipelining .....	13	MPLAB Integrated Development Environment Software..	125
Instruction Set		MPLINK Object Linker/MPLIB Object Librarian .....	126
ADDLW .....	113	<b>N</b>	
ADDWF .....	113	NOP Instruction .....	120
ANDLW .....	113	<b>O</b>	
ANDWF .....	113	OPTION Instruction .....	120
BCF .....	114	OPTION Register.....	23
BSF .....	114	Oscillator Configurations.....	95
BTFSC .....	114	Oscillator Start-up Timer (OST).....	98
BTFSS .....	115	<b>P</b>	
CALL .....	115	Package Marking Information .....	149
CLRF .....	115	Packaging Information .....	149
CLRW .....	116	PCL and PCLATH.....	28
CLRWDT.....	116	Stack.....	28
COMF .....	116	PCON Register .....	27
DECF .....	116	PICDEM 1 Low Cost PICmicro Demonstration Board .....	127
DECFSZ.....	117	PICDEM 17 Demonstration Board.....	128
GOTO.....	117	PICDEM 2 Low Cost PIC16CXX Demonstration Board ...	127
INCF.....	118	PICDEM 3 Low Cost PIC16CXXX Demonstration Board .	128
INCFSZ .....	118	PICSTART Plus Entry Level Development Programmer..	127
IORLW.....	119	PIE1 Register.....	25
IORWF .....	119	Pin Functions	
MOVF .....	119	RC6/TX/CK.....	69–86
MOVLW.....	119	RC7/RX/DT.....	69–86
MOVWF .....	120	PIR1 Register .....	26
NOP .....	120	Port RB Interrupt.....	105
OPTION .....	120	PORTA .....	31
RETFIE .....	120	PORTB .....	36
RETLW .....	121	Power Control/Status Register (PCON).....	99
RETURN .....	121	Power-Down Mode (SLEEP) .....	107
RLF .....	121	Power-On Reset (POR).....	98
RRF.....	122	Power-up Timer (PWRT) .....	98
SLEEP .....	122	PR2 Register .....	52, 58
SUBLW.....	122	PRO MATE II Universal Device Programmer .....	127
SUBWF .....	123	Program Memory Organization.....	15
SWAPF .....	123	PWM (CCP Module) .....	58
TRIS .....	123	Block Diagram .....	58
XORLW .....	124	Simplified PWM .....	58
XORWF .....	124	CCPR1H:CCPR1L Registers.....	58
Instruction Set Summary.....	111	Duty Cycle .....	59
INT Interrupt.....	105	Example Frequencies/Resolutions .....	59
INTCON Register .....	24	Period .....	58
Interrupt Sources		Set-Up for PWM Operation.....	59
Capture Complete (CCP).....	56	TMR2 to PR2 Match .....	58
Compare Complete (CCP).....	57		
TMR2 to PR2 Match (PWM) .....	58		
Interrupts .....	104		
Interrupts, Enable Bits			
CCP1 Enable (CCP1IE Bit).....	56		
Interrupts, Flag Bits			

# PIC16F627A/628A/648A

## Q

Q-Clock .....	59
Quick-Turnaround-Production (QTP) Devices .....	7

## R

RC Oscillator .....	96
RC Oscillator Mode	
Block Diagram .....	96
Registers	
Maps	
PIC16F627A .....	16, 17
PIC16F628A .....	16, 17
Reset .....	97
RETFIE Instruction .....	120
RETLW Instruction .....	121
RETURN Instruction .....	121
Revision History .....	155
RLF Instruction .....	121
RRF Instruction .....	122

## S

Serial Communication Interface (SCI) Module, See USART	
Serialized Quick-Turnaround-Production (SQTP) Devices ...	7
SLEEP Instruction .....	122
Software Simulator (MPLAB SIM) .....	126
Special Event Trigger. See Compare	
Special Features of the CPU .....	93
Special Function Registers .....	18
Status Register .....	22
SUBLW Instruction .....	122
SUBWF Instruction .....	123
SWAPF Instruction .....	123

## T

T1CKPS0 bit .....	48
T1CKPS1 bit .....	48
T1OSCEN bit .....	48
T1SYNC bit .....	48
T2CKPS0 bit .....	53
T2CKPS1 bit .....	53
Timer0	
Block Diagrams	
Timer0/WDT .....	46
External Clock Input .....	45
Interrupt .....	45
Prescaler .....	46
Switching Prescaler Assignment .....	47
Timer0 Module .....	45
Timer1	
Asynchronous Counter Mode .....	50
Capacitor Selection .....	51
External Clock Input .....	49
External Clock Input Timing .....	50
Oscillator .....	51
Prescaler .....	49, 51
Resetting Timer1 .....	51
Resetting Timer1 Registers .....	51
Special Event Trigger (CCP) .....	57
Synchronized Counter Mode .....	49
Timer Mode .....	49
TMR1H .....	50
TMR1L .....	50
Timer2	
Block Diagram .....	52
Postscaler .....	52

PR2 register .....	52
Prescaler .....	52, 59
Timer2 Module .....	52
TMR2 output .....	52
TMR2 to PR2 Match Interrupt .....	58

## Timing Diagrams

Timer0 .....	143
Timer1 .....	143
USART	
Asynchronous Receiver .....	80
USART Asynchronous Master Transmission .....	77
USART Asynchronous Reception .....	80
USART RX Pin Sampling .....	75, 76
USART Synchronous Reception .....	86
USART Synchronous Transmission .....	84
Timing Diagrams and Specifications .....	140
TMR0 Interrupt .....	105
TMR1CS bit .....	48
TMR1ON bit .....	48
TMR2ON bit .....	53
TOUTPS0 bit .....	53
TOUTPS1 bit .....	53
TOUTPS2 bit .....	53
TOUTPS3 bit .....	53
TRIS Instruction .....	123
TRISA .....	31
TRISB .....	36

## U

Universal Synchronous Asynchronous Receiver Transmitter (USART) .....	69
Asynchronous Receiver	
Setting Up Reception .....	82
Asynchronous Receiver Mode	
Address Detect .....	82
Block Diagram .....	82
USART	
Asynchronous Mode .....	76
Asynchronous Receiver .....	79
Asynchronous Reception .....	81
Asynchronous Transmission .....	77
Asynchronous Transmitter .....	76
Baud Rate Generator (BRG) .....	71
Block Diagrams	
Transmit .....	77
USART Receive .....	79
BRGH bit .....	71
Sampling .....	72, 73, 74
Synchronous Master Mode .....	83
Synchronous Master Reception .....	85
Synchronous Master Transmission .....	83
Synchronous Slave Mode .....	86
Synchronous Slave Reception .....	87
Synchronous Slave Transmit .....	86

## V

Voltage Reference	
Configuration .....	67
Voltage Reference Module .....	67

## W

Watchdog Timer (WDT) .....	106
WWW, On-Line Support .....	3

# PIC16F627A/628A/648A

---

## X

XORLW Instruction .....	124
XORWF Instruction .....	124

# PIC16F627A/628A/648A

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

<b>PART NO.</b>   <b>Device</b>	<b>-X</b>   <b>Temperature Range</b>	<b>/XX</b>   <b>Package</b>	<b>XXX</b>   <b>Pattern</b>	<b>Examples:</b>
Device	PIC16F627A/628A/648A: Standard V <sub>DD</sub> range 3.0V to 5.5V PIC16F627A/628A/648AT: V <sub>DD</sub> range 3.0V to 5.5V (Tape and Reel) PIC16LF627A/628A/648A: V <sub>DD</sub> range 2.0V to 5.5V PIC16LF627A/628A/648AT: V <sub>DD</sub> range 2.0V to 5.5V (Tape and Reel)			a) PIC16F627A - E/P 301 = Extended Temp., PDIP package, 20 MHz, normal V <sub>DD</sub> limits, QTP pattern #301. b) PIC16LF627A - I/SO = Industrial Temp., SOIC package, 20 MHz, extended V <sub>DD</sub> limits.
Temperature Range	I = -40°C to +85°C E = -40°C to +125°C			
Package	P = PDIP SO = SOIC (Gull Wing, 300 mil body) SS = SSOP (209 mil) ML = QFN (28 Lead)			
Pattern	3-Digit Pattern Code for QTP (blank otherwise).			

\* JW Devices are UV erasable and can be programmed to any device configuration. JW Devices meet the electrical requirement of each oscillator type.

## 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: (480) 792-7277
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.