PIC16C84

8-Bit CMOS EEPROM Microcontroller

High Performance RISC CPU Features

- Only 35 single word instructions to learn
- All instructions single cycle (400 ns @ 10MHz) except for program branches which are two-cycle
- Operating speed: DC - 10MHz clock input
  DC - 400 ns instruction cycle
- 14-bit wide instructions
- 8-bit wide data path
- 1K x 14 EEPROM program memory
- 36 x 8 general purpose registers (SRAM)
- 64 x 8 on-chip EEPROM data memory
- 15 special function hardware registers
- Eight-level deep hardware stack
- Direct, indirect and relative addressing modes
- Four interrupt sources:
  - External RB0/INT pin
  - TMR0 timer overflow
  - PORTB<7:4> interrupt on change
  - Data EEPROM write complete
- 1,000,000 data memory EEPROM ERASE/WRITE cycles
- EEPROM Data Retention > 40 years

Peripheral Features

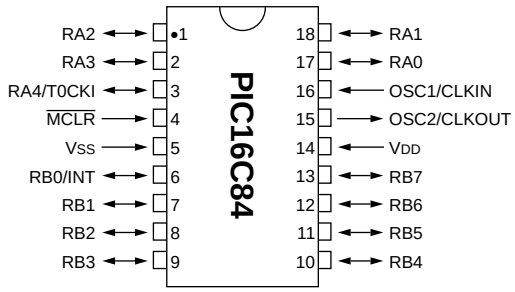
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 20 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features

- Power-on Reset (POR)
- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Code protection
- Power saving SLEEP mode
- Selectable oscillator options
- Serial In-System Programming - via two pins

Pin Diagram

PDIP, SOIC



CMOS Technology

- Low-power, high-speed CMOS EEPROM technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.0V to 6.0V
  - Industrial: 2.0V to 6.0V
- Low power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 60 μA typical @ 2V, 32 kHz
  - 26 μA typical standby current @ 2V

Table of Contents

1.0 General Description 3
2.0 PIC16C84 Device Varieties 5
3.0 Architectural Overview 7
4.0 Memory Organization 11
5.0 I/O Ports 19
6.0 Timer0 Module and TMR0 Register 25
7.0 Data EEPROM Memory 31
8.0 Special Features of the CPU 35
9.0 Instruction Set Summary 51
10.0 Development Support 63
11.0 Electrical Characteristics for PIC16C84 67
12.0 DC & AC Characteristics Graphs/Tables for PIC16C84 77
13.0 Packaging Information 91
Appendix A: Feature Improvements 95
Appendix B: Compatibility 95
Appendix C: What's New 96
Appendix D: What's Changed 96
Appendix E: PIC16C84 to PIC16F83/F84/CR83/ CR84 Conversion Considerations 97
Appendix F: PIC16/17 Microcontrollers 99
Index 111
On-Line Support 109
Reader Response 110
PIC16C84 Product Identification System 115

To Our Valued Customers

We constantly strive to improve the quality of all our products and documentation. We have spent a great deal of time to ensure that these documents are correct. However, we realize that we may have missed a few things. If you find any information that is missing or appears in error, please use the reader response form in the back of this data sheet to inform us. We appreciate your assistance in making this a better document.

1.0 GENERAL DESCRIPTION

The PIC16C84 is a low-cost, high-performance, CMOS, fully-static, 8-bit microcontroller.

All PIC16/17 microcontrollers employ an advanced RISC architecture. PIC16CXX devices 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 a separate 8-bit wide data bus. The two stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set is used to achieve a very high performance level.

PIC16CXX microcontrollers typically achieve a 2:1 code compression and up to a 2:1 speed improvement (at 10 MHz) over other 8-bit microcontrollers in their class.

The PIC16C84 has 36 bytes of RAM, 64 bytes of Data EEPROM memory, and 13 I/O pins. A timer/counter is also available.

The PIC16CXX family has special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. There are four oscillator options, of which the single pin RC oscillator provides a low-cost solution, the LP oscillator minimizes power consumption, XT is a standard crystal, and the HS is for High Speed crystals. The SLEEP (power-down) mode offers power saving. The user can wake the chip from sleep through several external and internal interrupts and resets.

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

The PIC16C84 EEPROM program memory allows the same device package to be used for prototyping and production. In-circuit reprogrammability allows the code to be updated without the device being removed from the end application. This is useful in the development of many applications where the device may not be easily accessible, but the prototypes may require code updates. This is also useful for remote applications where the code may need to be updated (such as rate information).

Table 1-1 lists the features of the PIC16C84, and Appendix F: lists the features of all of the Microchip microcontrollers.

A simplified block diagram of the PIC16C84 is shown in Figure 3-1.

The PIC16C84 fits perfectly in applications ranging from high speed automotive and appliance motor control to low-power remote sensors, electronic locks, security devices and smart cards. The EEPROM technology makes customization of application programs (transmitter codes, motor speeds, receiver frequencies, security codes, etc.) extremely fast and convenient. The small footprint packages make this microcontroller series perfect for all applications with space limitations. Low-cost, low-power, high performance, ease of use and I/O flexibility make the PIC16C84 very versatile even in areas where no microcontroller use has been considered before (e.g., timer functions, serial communication, capture and compare, PWM functions and co-processor applications).

The serial in-system programming feature (via two pins) offers flexibility of customizing the product after complete assembly and testing. This feature can be used to serialize a product, store calibration data, or program the device with the current firmware before shipping.

1.1 Family and Upward Compatibility

Those users familiar with the PIC16C5X family of microcontrollers will realize that this is an enhanced version of the PIC16C5X architecture. Please refer to Appendix A: for a detailed list of enhancements. Code written for PIC16C5X can be easily ported to the PIC16C84 (Appendix B:).

1.2 Development Support

The PIC16CXX family is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a low-cost development programmer and a full-featured programmer. A “C” compiler and fuzzy logic support tools are also available.

TABLE 1-1: PIC16C8X FAMILY OF DEVICES

	Clock	Memory					Peripherals		Features	
	Maximum Frequency of Operation (MHz)	Program Memory			Data Memory (bytes)	Data EEPROM (bytes)	Timer Module(s)	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Packages
		Flash	EEPROM	ROM							
PIC16C84	10	—	1K	—	36	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16F84(1)	10	1K	—	—	68	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16CR84(1)	10	—	—	1K	68	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16F83(1)	10	512	—	—	36	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16CR83(1)	10	—	—	512	36	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC

All PIC16/17 family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect, and high I/O current capability.

All PIC16C8X family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local sales office for availability of these devices.

2.0 PIC16C84 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 this section. When placing orders, please use the "PIC16C84 Product Identification System" at the back of this data sheet to specify the correct part number.

There are two device "types" as indicated in the device number.

1. **C**, as in PIC16**C**84. These devices have EEPROM program memory and operate over the standard voltage range.
2. **LC**, as in PIC16**LC**84. These devices have EEPROM program memory and operate over an extended voltage range.

When discussing memory maps and other architectural features, the use of **C** also implies the **LC** versions.

2.1 Electrically Erasable Devices

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

A further advantage of the electrically erasable version is that they 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 choose not to program a medium to high quantity of units and whose code patterns have stabilized. The devices have all EEPROM locations and configuration options already programmed by the factory. Certain code and prototype verification procedures do apply before production shipments are available.

For information on submitting a QTP code, please contact your Microchip Regional Sales Office.

2.3 Serialized Quick-Turnaround-Production (SQTP℠) Devices

Microchip offers the 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.

For information on submitting a SQTP code, please contact your Microchip Regional Sales Office.

NOTES:

3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16CXX family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16CXX uses a Harvard architecture. This architecture has the program and data accessed from separate memories. So the device has a program memory bus and a data memory bus. This improves bandwidth over traditional von Neumann architecture where program and data are fetched from the same memory (accesses over the same bus). Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word. PIC16CXX opcodes are 14-bits wide, enabling single word instructions. The full 14-bit wide program memory bus fetches a 14-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions (Example 3-1). Consequently, all instructions execute in a single cycle (400 ns @ 10 MHz) except for program branches.

The PIC16C84 addresses 1K x 14 program memory. All program memory is internal.

PIC16CXX devices 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. 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 PIC16CXX simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16C84 has 36 x 8 SRAM and 64 x 8 EEPROM data memory.

PIC16CXX devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register), and 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 $\overline{\text{borrow}}$ and $\overline{\text{digit borrow}}$ out bit, respectively, in subtraction. See the SUBLW and SUBWF instructions for examples.

A simplified block diagram for the PIC16C84 is shown in Figure 3-1, its corresponding pin description is shown in Table 3-1.

FIGURE 3-1: PIC16C84 BLOCK DIAGRAM

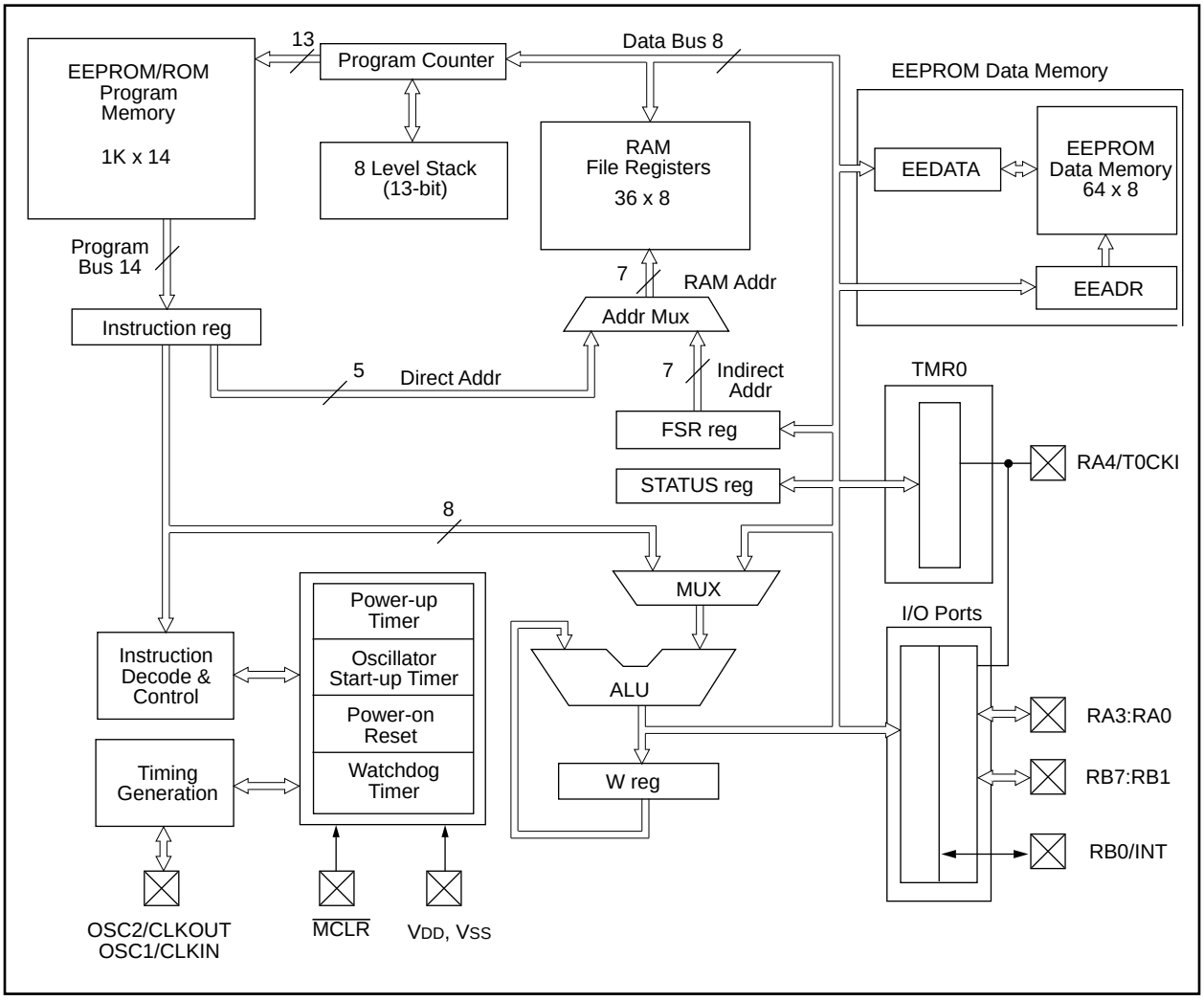


TABLE 3-1: PIC16C8X PINOUT DESCRIPTION

Pin Name	DIP No.	SOIC No.	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	16	16	I	ST/CMOS (1)	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	15	15	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
$\overline{MCLR}$	4	4	I/P	ST	Master clear (reset) input/programming voltage input. This pin is an active low reset to the device.
					PORTA is a bi-directional I/O port.
RA0	17	17	I/O	TTL	
RA1	18	18	I/O	TTL	
RA2	1	1	I/O	TTL	
RA3	2	2	I/O	TTL	
RA4/T0CKI	3	3	I/O	ST	Can also be selected to be the clock input to the TMR0 timer/counter. Output is open drain type.
					PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	6	6	I/O	TTL	RB0/INT can also be selected as an external interrupt pin.
RB1	7	7	I/O	TTL	
RB2	8	8	I/O	TTL	
RB3	9	9	I/O	TTL	
RB4	10	10	I/O	TTL	Interrupt on change pin.
RB5	11	11	I/O	TTL	Interrupt on change pin.
RB6	12	12	I/O	TTL/ST (2)	Interrupt on change pin. Serial programming clock.
RB7	13	13	I/O	TTL/ST (2)	Interrupt on change pin. Serial programming data.
VSS	5	5	P	—	Ground reference for logic and I/O pins.
VDD	14	14	P	—	Positive supply for logic and I/O pins.

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

Note 1: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.
2: This buffer is a Schmitt Trigger input when used in serial programming mode.

3.1 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) 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.

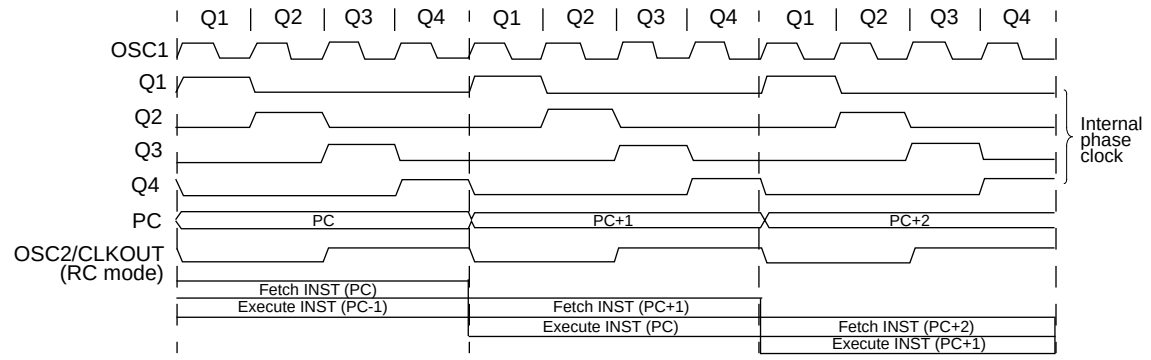
3.2 Instruction Flow/Pipelining

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).

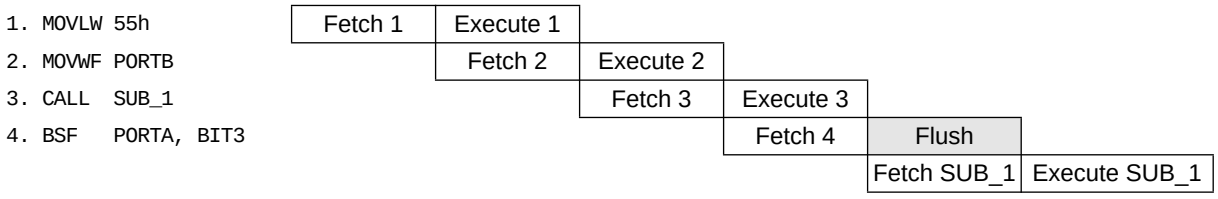
A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the "Instruction Register" 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



All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline while the new instruction is being fetched and then executed.

4.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16C84. These are the program memory and the data memory. Each block has its own bus, so that access to each block can occur during the same oscillator cycle.

The data memory can further be broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

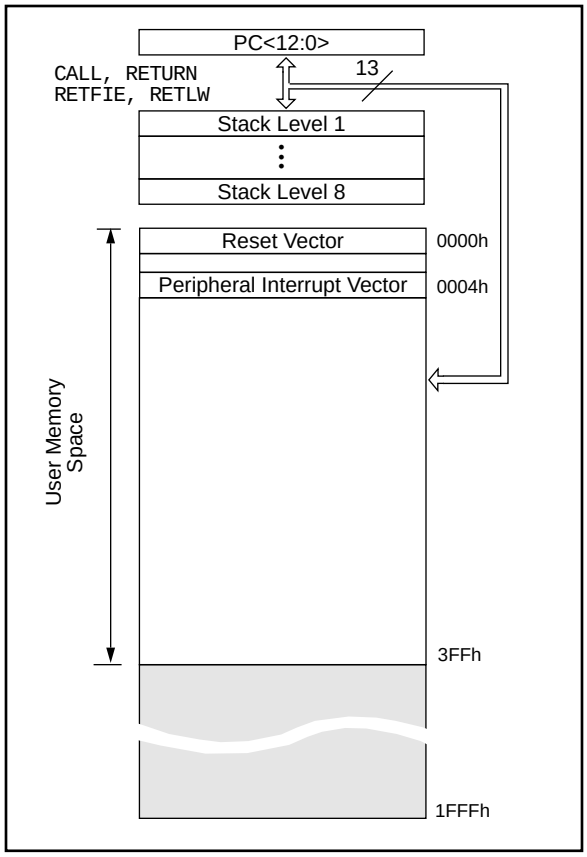
The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is an indirect address pointer specifies the address of the data EEPROM memory to read/write. The 64 bytes of data EEPROM memory have the address range 0h-3Fh. More details on the EEPROM memory can be found in Section 7.0.

4.1 Program Memory Organization

The PIC16CXX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16C84, only the first 1K x 14 (0000h-03FFh) are physically implemented (Figure 4-1). Accessing a location above the physically implemented address will cause a wraparound. For example, locations 20h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h will be the same instruction.

The reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 4-1: PROGRAM MEMORY MAP AND STACK



4.2 Data Memory Organization

The data memory is partitioned into two areas. The first is the Special Function Registers (SFR) area, while the second is the General Purpose Registers (GPR) area. The SFRs control the operation of the device.

Portions of data memory are banked. This is for both the SFR area and the GPR area. The GPR area is banked to allow greater than 116 bytes of general purpose RAM. The banked areas of the SFR are for the registers that control the peripheral functions. Banking requires the use of control bits for bank selection. These control bits are located in the STATUS Register. Figure 4-2 shows the data memory map organization.

Instructions `MOVWF` and `MOVF` can move values from the W register to any location in the register file ("F"), and vice-versa.

The entire data memory can be accessed either directly using the absolute address of each register file or indirectly through the File Select Register (FSR) (Section 4.5). Indirect addressing uses the present value of the RP1:RP0 bits for access into the banked areas of data memory.

Data memory is partitioned into two banks which contain the general purpose registers and the special function registers. Bank 0 is selected by clearing the RP0 bit (STATUS<5>). Setting the RP0 bit selects Bank 1. Each Bank extends up to 7Fh (128 bytes). The first twelve locations of each Bank are reserved for the Special Function Registers. The remainder are General Purpose Registers implemented as static RAM.

4.2.1 GENERAL PURPOSE REGISTER FILE

All devices have some amount of General Purpose Register (GPR) area. Each GPR is 8 bits wide and is accessed either directly or indirectly through the FSR (Section 4.5).

The GPR addresses in bank 1 are mapped to addresses in bank 0. As an example, addressing location 0Ch or 08h will access the same GPR.

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers (Figure 4-2 and Table 4-1) are used by the CPU and Peripheral functions to control the device operation. These registers are static RAM.

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

FIGURE 4-2: REGISTER FILE MAP

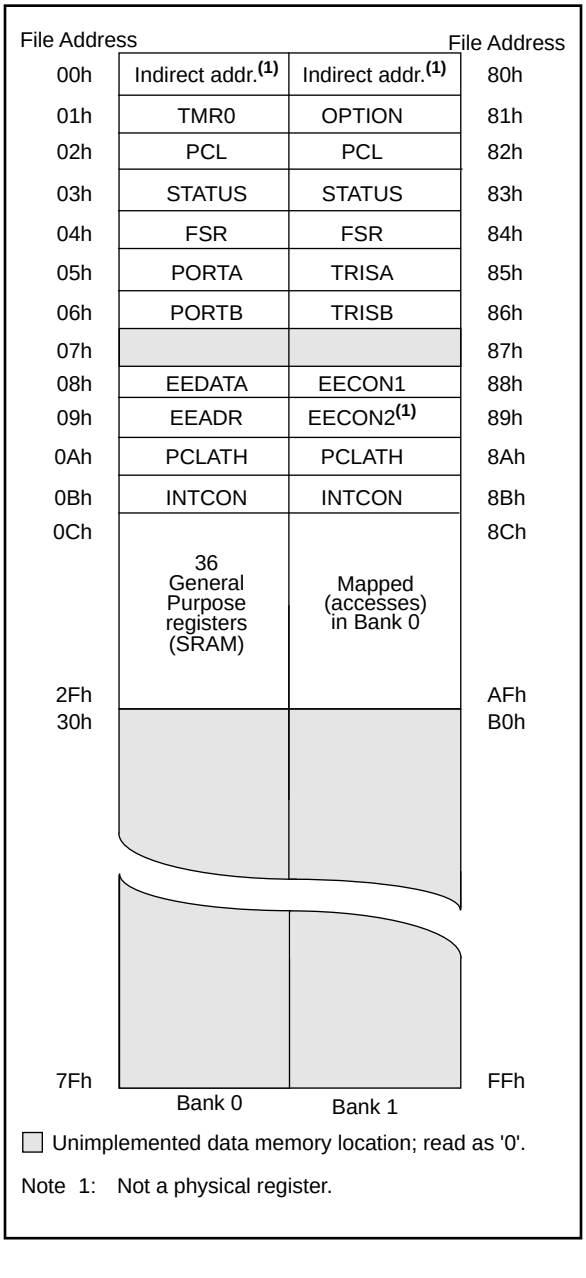


TABLE 4-1: REGISTER FILE SUMMARY

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 (Note3)
Bank 0											
00h	INDF	Uses contents of FSR to address data memory (not a physical register)								---- ----	---- ----
01h	TMR0	8-bit real-time clock/counter								xxxx xxxx	uuuu uuuu
02h	PCL	Low order 8 bits of the Program Counter (PC)								0000 0000	0000 0000
03h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
04h	FSR	Indirect data memory address pointer 0								xxxx xxxx	uuuu uuuu
05h	PORTA	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	---u uuuu
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	uuuu uuuu
07h		Unimplemented location, read as '0'								---- ----	---- ----
08h	EEDATA	EEPROM data register								xxxx xxxx	uuuu uuuu
09h	EEADR	EEPROM address register								xxxx xxxx	uuuu uuuu
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC (1)					---0 0000	---0 0000
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
Bank 1											
80h	INDF	Uses contents of FSR to address data memory (not a physical register)								---- ----	---- ----
81h	OPTION	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	0000 0000
83h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	uuuu uuuu
85h	TRISA	—	—	—	PORTA data direction register					---1 1111	---1 1111
86h	TRISB	PORTB data direction register								1111 1111	1111 1111
87h		Unimplemented location, read as '0'								---- ----	---- ----
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	---0 q000
89h	EECON2	EEPROM control register 2 (not a physical register)								---- ----	---- ----
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC (1)					---0 0000	---0 0000
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u

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

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a slave register for PC<12:8>. The contents of PCLATH can be transferred to the upper byte of the program counter, but the contents of PC<12:8> is never transferred to PCLATH.

2: The $\overline{TO}$ and $\overline{PD}$ status bits in the STATUS register are not affected by a $\overline{MCLR}$ reset.

3: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and the Watchdog Timer Reset.

4.2.2.1 STATUS REGISTER

The STATUS register contains the arithmetic status of the ALU, the RESET status and the bank select bit for data memory.

As with any register, the STATUS register can be the destination for any instruction. 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 device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, CLRF STATUS will clear the upper-three bits and set the Z bit. This leaves the STATUS register as 000u u1uu (where u = unchanged).

Only the BCF, BSF, SWAPF and MOVWF instructions should be used to alter the STATUS register (Table 9-2) because these instructions do not affect any status bit.

Note 1: The IRP and RP1 bits (STATUS<7:6>) are not used by the PIC16C84 and should be programmed as cleared. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

Note 2: The C and DC bits operate as a $\overline{borrow}$ and digit borrow out bit, respectively, in subtraction. See the SUBLW and SUBWF instructions for examples.

Note 3: When 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. The specified bit(s) will be updated according to device logic

FIGURE 4-3: STATUS REGISTER (ADDRESS 03h, 83h)

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

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

bit 7: **IRP**: Register Bank Select bit (used for indirect addressing)
0 = Bank 0, 1 (00h - FFh)
1 = Bank 2, 3 (100h - 1FFh)
The IRP bit is not used by the PIC16C8X. IRP should be maintained clear.

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)
Each bank is 128 bytes. Only bit RP0 is used by the PIC16C8X. RP1 should be maintained clear.

bit 4: **TO**: Time-out bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3: **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/$\overline{borrow}$ bit (for ADDWF and ADDLW instructions) (For $\overline{borrow}$ 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/$\overline{borrow}$ bit (for ADDWF and ADDLW 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 $\overline{borrow}$ 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.

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

Note: When the prescaler is assigned to the WDT (PSA = '1'), TMR0 has a 1:1 prescaler assignment.

FIGURE 4-4: OPTION REGISTER (ADDRESS 81h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit7							bit0

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

bit 7: **$\overline{\text{RBPU}}$**: PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled (by individual port latch values)

bit 6: **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5: **T0CS**: TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4: **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin

bit 3: **PSA**: Prescaler Assignment bit
1 = Prescaler assigned to the WDT
0 = Prescaler assigned to TMR0

bit 2-0: **PS2:PS0**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

4.2.2.3 INTCON REGISTER

The INTCON register is a readable and writable register which contains the various enable bits for all interrupt sources.

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>).

FIGURE 4-5: INTCON REGISTER (ADDRESS 0Bh, 8Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit7							bit0

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

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

Note: For the operation of the interrupt structure, please refer to Section 8.5.

bit 6: **EEIE**: EE Write Complete Interrupt Enable bit
1 = Enables the EE write complete interrupt
0 = Disables the EE write complete interrupt

bit 5: **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4: **INTE**: RB0/INT Interrupt Enable bit
1 = Enables the RB0/INT interrupt
0 = Disables the RB0/INT 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: **T0IF**: TMR0 overflow interrupt flag bit
1 = TMR0 has overflowed (must be cleared in software)
0 = TMR0 did not overflow

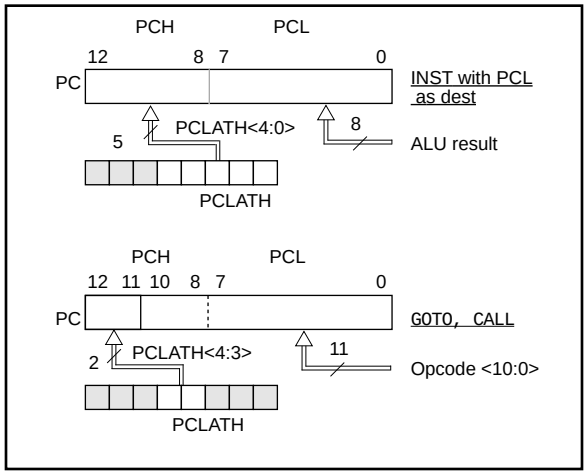
bit 1: **INTF**: RB0/INT Interrupt Flag bit
1 = The RB0/INT interrupt occurred
0 = The RB0/INT interrupt did not occur

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

4.3 Program Counter: PCL and PCLATH

The Program Counter (PC) is 13-bits wide. The low byte is the PCL register, which is a readable and writable register. The high byte of the PC (PC<12:8>) is not directly readable nor writable and comes from the PCLATH register. The PCLATH (PC latch high) register is a holding register for PC<12:8>. The contents of PCLATH are transferred to the upper byte of the program counter when the PC is loaded with a new value. This occurs during a CALL, GOTO or a write to PCL. The high bits of PC are loaded from PCLATH as shown in Figure 4-6.

FIGURE 4-6: 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 PROGRAM MEMORY PAGING

The PIC16C84 has 1K of program memory. The CALL and GOTO instructions have an 11-bit address range. This 11-bit address range allows a branch within a 2K program memory page size. For future PIC16CXX program memory expansion, there must be another two bits to specify the program memory page. These paging bits come from the PCLATH<4:3> bits (Figure 4-6). When doing a CALL or a GOTO instruction, the user must ensure that these page bits (PCLATH<4:3>) are programmed to the desired program memory page. If a CALL instruction (or interrupt) is executed, the entire 13-bit PC is "pushed" onto the stack (see next section). Therefore, manipulation of the PCLATH<4:3> is not required for the return instructions (which "pops" the PC from the stack)

Note:	The PIC16C84 ignores the PCLATH<4:3> bits, which are used for program memory pages 1, 2 and 3 (0800h - 1FFFh). The use of PCLATH<4:3> as general purpose R/W bits is not recommended since this may affect upward compatibility with future products.

4.4 Stack

The PIC16C84 has an 8 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 entire 13-bit PC is "pushed" onto the stack when a CALL instruction is executed or an interrupt is acknowledged. The stack is "popped" in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a push or a pop operation.

Note:	There are no instruction 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.

The stack operates as a circular buffer. That is, 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).

If the stack is effectively popped nine times, the PC value is the same as the value from the first pop.

Note:	There are no status bits to indicate stack overflow or stack underflow conditions.

4.5 Indirect Addressing; INDF and FSR Registers

The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a *pointer*). This is indirect addressing.

EXAMPLE 4-1: INDIRECT ADDRESSING

- Register file 05 contains the value 10h
- Register file 06 contains the value 0Ah
- Load the value 05 into the FSR register
- A read of the INDF register will return the value of 10h
- Increment the value of the FSR register by one (FSR = 06)
- A read of the INDR register now will return the value of 0Ah.

Reading INDF itself indirectly (FSR = 0) will produce 00h. Writing to the INDF register indirectly results in a no-operation (although STATUS bits may be affected).

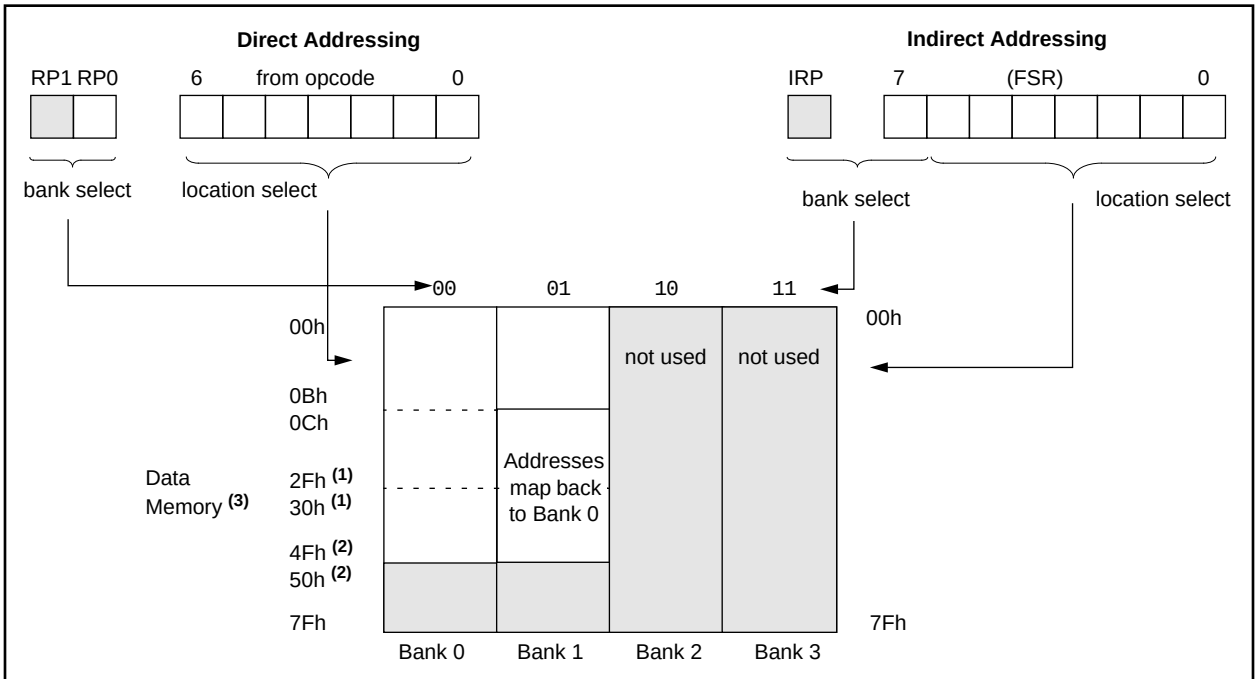
A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 4-2.

EXAMPLE 4-2: HOW TO CLEAR RAM USING INDIRECT ADDRESSING

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

An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-7. However, IRP is not used in the PIC16C84.

FIGURE 4-7: DIRECT/INDIRECT ADDRESSING



Note 1: PIC16C83, PIC16CR83, and PIC16C84 devices.
2: PIC16C84A and PIC16CR84 devices
3: For memory map detail see Figure 4-1.

5.0 I/O PORTS

The PIC16C84 has two ports, PORTA and PORTB. Some port pins are multiplexed with an alternate function for other features on the device.

5.1 PORTA and TRISA Registers

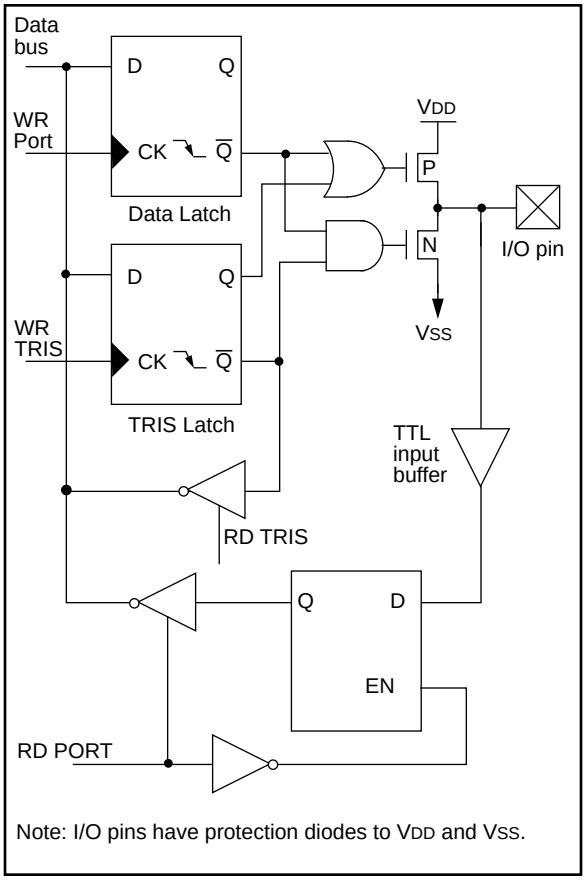
PORTA is a 5-bit wide latch. RA4 is a Schmitt Trigger input and an open drain output. All other RA port pins have TTL input levels and full CMOS output drivers. All pins have data direction bits (TRIS registers) which can configure these pins as output or input.

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

Reading the PORTA register reads the status of the pins whereas writing to it will write to the port latch. All write operations are read-modify-write operations. So a write to a port implies that the port pins are first read, then this value is modified and written to the port data latch.

The RA4 pin is multiplexed with the TMR0 clock input.

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

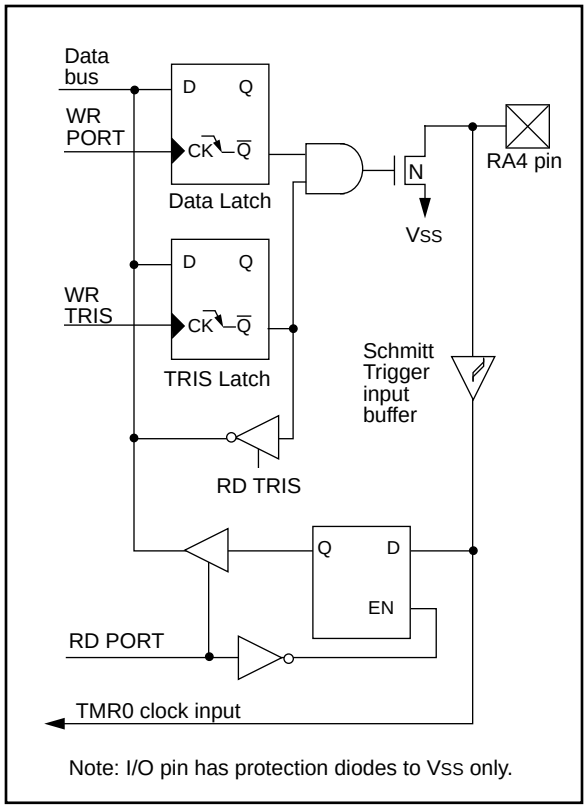


Note: I/O pins have protection diodes to VDD and VSS.

EXAMPLE 5-1: INITIALIZING PORTA

```
CLRF    PORTA       ; Initialize PORTA by
                    ; setting output
                    ; data latches
BSF     STATUS, RP0 ; Select Bank 1
MOVLW   0x0F        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISA       ; Set RA<3:0> as inputs
                    ; RA4 as outputs
                    ; TRISA<7:5> are always
                    ; read as '0'.
```

FIGURE 5-2: BLOCK DIAGRAM OF PIN RA4



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

Note: For crystal oscillator configurations operating below 500 kHz, the device may generate a spurious internal Q-clock when PORTA<0> switches state. This does not occur with an external clock in RC mode. To avoid this, the RA0 pin should be kept static, i.e. in input/output mode, pin RA0 should not be toggled.

TABLE 5-1: PORTA FUNCTIONS

Name	Bit0	Buffer Type	Function
RA0	bit0	TTL	Input/output
RA1	bit1	TTL	Input/output
RA2	bit2	TTL	Input/output
RA3	bit3	TTL	Input/output
RA4/T0CKI	bit4	ST	Input/output or external clock input for TMR0. Output is open drain type.

Legend: TTL = TTL input, ST = Schmitt Trigger input

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets
05h	PORTA	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	---u uuuu
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are unimplemented, read as '0'

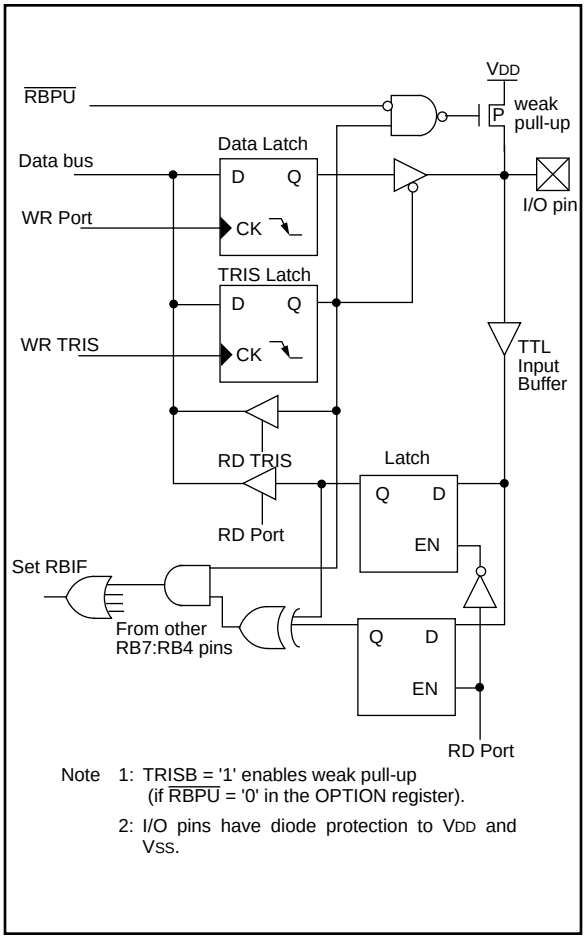
5.2 PORTB and TRISB Registers

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

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

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

FIGURE 5-3: BLOCK DIAGRAM OF PINS RB7:RB4



Note 1: TRISB = '1' enables weak pull-up (if $\overline{\text{RBPU}}$ = '0' in the OPTION register).

2: I/O pins have diode protection to VDD and VSS.

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

a) Read (or write) PORTB. This will end the mismatch condition.

b) Clear flag bit RBIF.

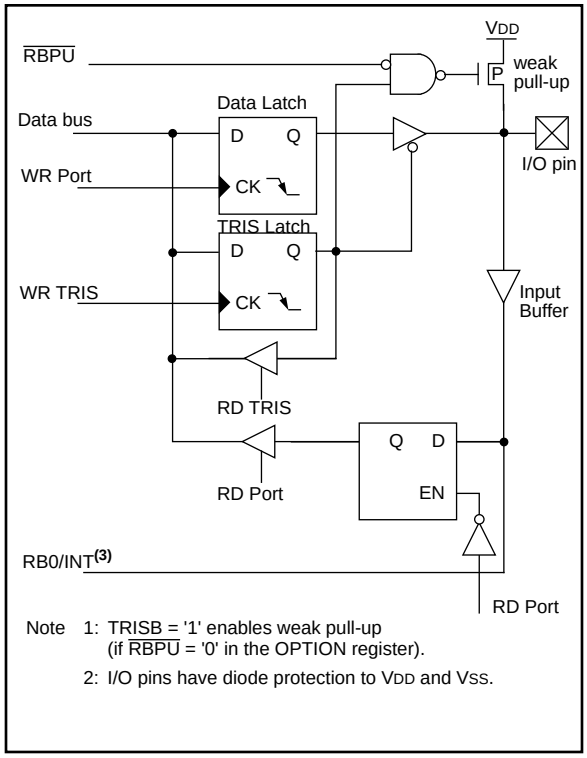
A mismatch condition will continue to set the RBIF bit. Reading PORTB will end the mismatch condition, and allow the RBIF bit to be cleared.

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

Note 1: If a change on the I/O pin should occur when a read operation of PORTB is being executed (start of the Q2 cycle), the RBIF interrupt flag bit may not be set.

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

FIGURE 5-4: BLOCK DIAGRAM OF PINS RB3:RB0



Note 1: TRISB = '1' enables weak pull-up (if $\overline{\text{RBPU}}$ = '0' in the OPTION register).

2: I/O pins have diode protection to VDD and VSS.

EXAMPLE 5-2: INITIALIZING PORTB

```
CLRF    PORTB        ; Initialize PORTB by
                     ; setting output
                     ; data latches
BSF     STATUS, RP0  ; Select Bank 1
MOVLW   0xCF         ; Value used to
                     ; initialize data
                     ; direction
MOVWF   TRISB        ; Set RB<3:0> as inputs
                     ; RB<5:4> as outputs
                     ; RB<7:6> as inputs
```

TABLE 5-3: PORTB FUNCTIONS

Name	Bit	Buffer Type	I/O Consistency Function
RB0/INT	bit0	TTL/ST(1)	Input/output pin or external interrupt input. Internal software programmable weak pull-up.
RB1	bit1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3	bit3	TTL	Input/output pin. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt on change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt on change). Internal software programmable weak pull-up.
RB6	bit6	TTL/ST(2)	Input/output pin (with interrupt on change). Internal software programmable weak pull-up. Serial programming clock.
RB7	bit7	TTL/ST(2)	Input/output pin (with interrupt on change). Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger.

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt, except for the PIC16C84, which remains TTL.

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

TABLE 5-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

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
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	uuuu uuuu
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

5.3 I/O Programming Considerations

5.3.1 BI-DIRECTIONAL I/O PORTS

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

Reading the port register, reads the values of the port pins. Writing to the port register writes the value to the port latch. When using read-modify-write instructions (i.e., BCF, BSF, etc.) on a port, the value of the port pins is read, the desired operation is done to this value, and this value is then written to the port latch.

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 current may damage the chip.

5.3.2 SUCCESSIVE OPERATIONS ON I/O PORTS

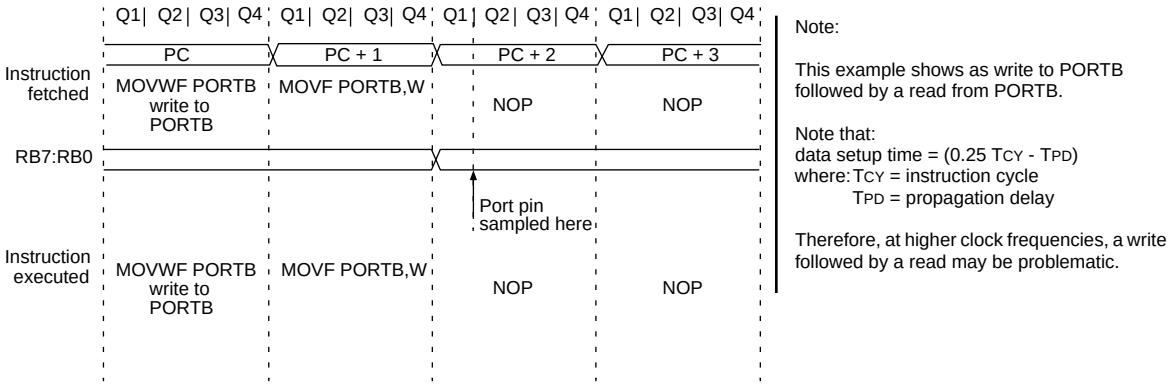
The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle (Figure 5-5). Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such that the pin voltage stabilizes (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.

Example 5-3 shows the effect of two sequential read-modify-write instructions (e.g., BCF, BSF, etc.) on an I/O port.

EXAMPLE 5-3: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

```
;Initial PORT settings: PORTB<7:4> Inputs
;                       PORTB<3:0> Outputs
;PORTB<7:6> have external pull-ups and are
;not connected to other circuitry
;
;                     PORT latch  PORT pins
;                     ----------  ---------
  BCF PORTB, 7      ; 01pp ppp    11pp ppp
  BCF PORTB, 6      ; 10pp ppp    11pp ppp
  BSF STATUS, RP0   ;
  BCF TRISB, 7      ; 10pp ppp    11pp ppp
  BCF TRISB, 6      ; 10pp ppp    10pp ppp
;
;Note that the user may have expected the
;pin values to be 00pp ppp. The 2nd BCF
;caused RB7 to be latched as the pin value
;(high).
```

FIGURE 5-5: SUCCESSIVE I/O OPERATION



NOTES:

6.0 TIMER0 MODULE AND TMR0 REGISTER

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt on overflow from FFh to 00h
- Edge select for external clock

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In timer mode, the Timer0 module (Figure 6-1) will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two cycles (Figure 6-2 and Figure 6-3). The user can work around this by writing an adjusted value to the TMR0 register.

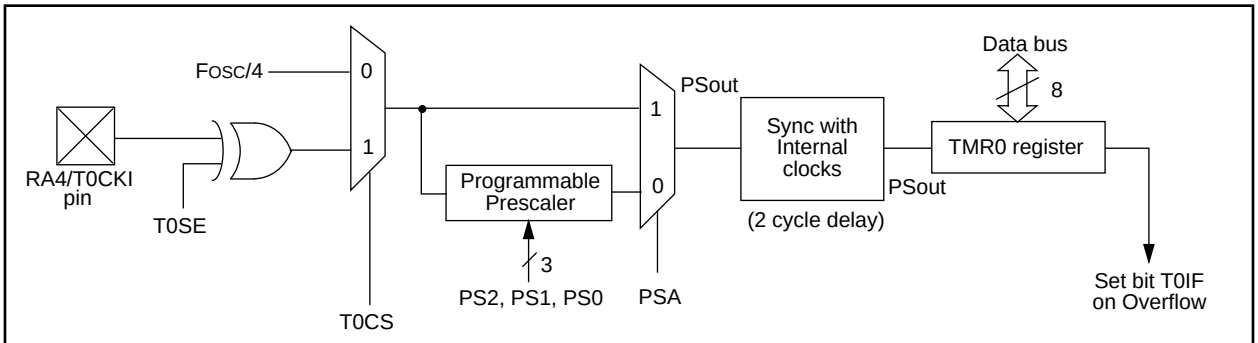
Counter mode is selected by setting the T0CS bit (OPTION<5>). In this mode TMR0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the T0 source edge select bit, T0SE (OPTION<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.2.

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

6.1 TMR0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets the T0IF bit (INTCON<2>). The interrupt can be masked by clearing enable bit T0IE (INTCON<5>). The T0IF bit must be cleared in software by the Timer0 Module interrupt service routine before re-enabling this interrupt. The TMR0 interrupt (Figure 6-4) cannot wake the processor from SLEEP since the timer is shut off during SLEEP.

FIGURE 6-1: TMR0 BLOCK DIAGRAM



Note 1: Bits T0CS, T0SE, PS2, PS1, PS0 and PSA are located in the OPTION register.
2: The prescaler is shared with the Watchdog Timer (Figure 6-6)

FIGURE 6-2: TMR0 TIMING: INTERNAL CLOCK/NO PRESCALER

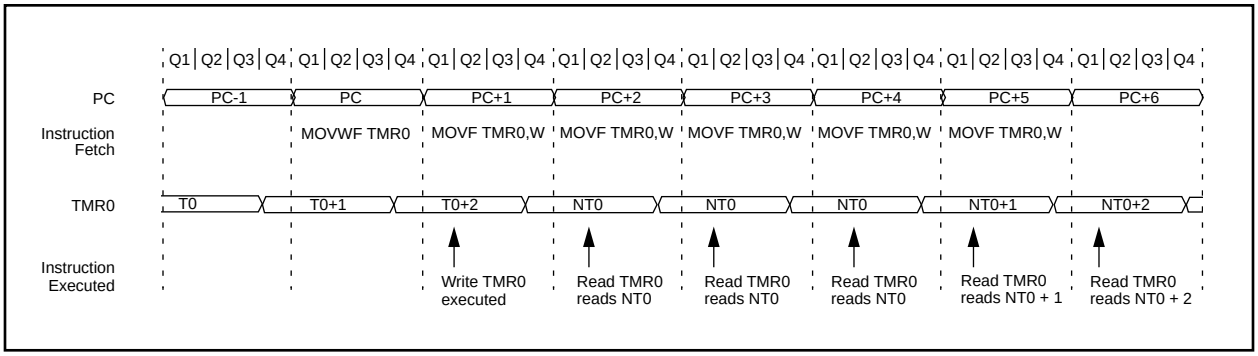


FIGURE 6-3: TMR0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

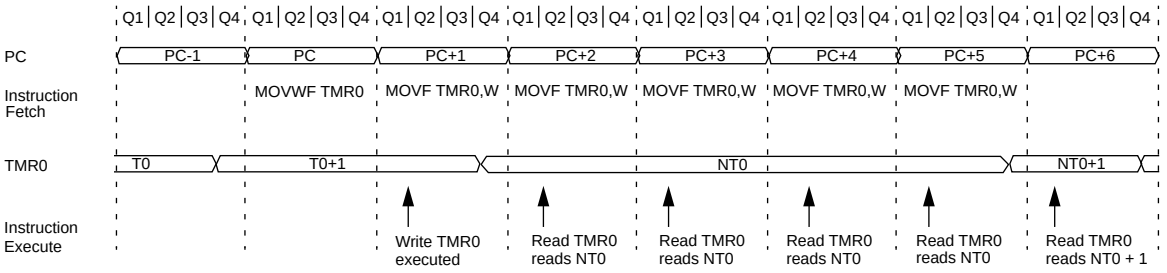
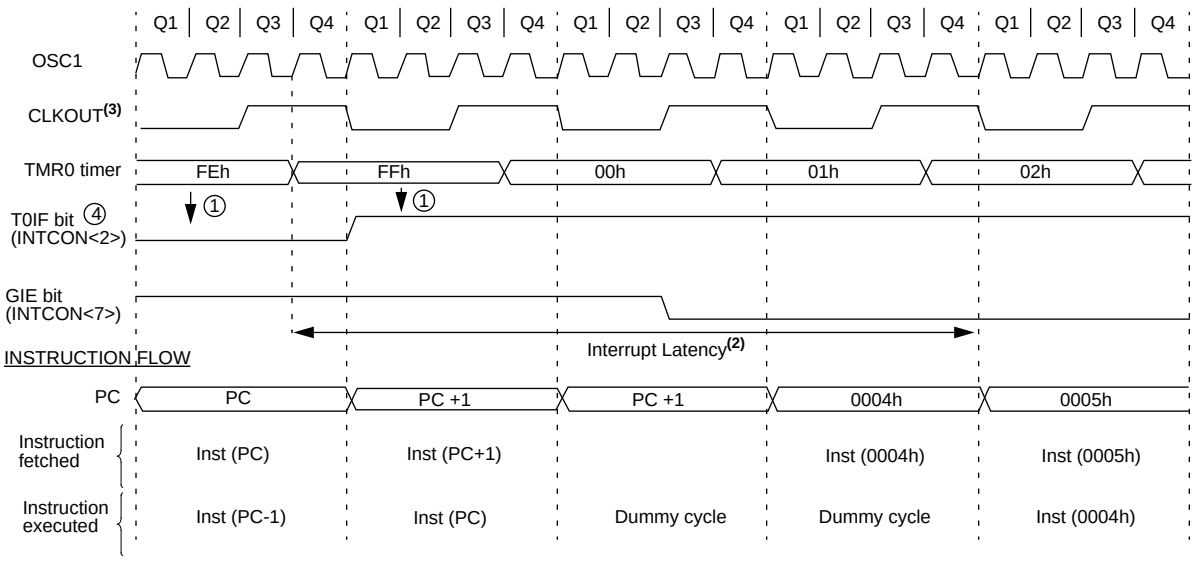


FIGURE 6-4: TMR0 INTERRUPT TIMING



Note 1: T0IF interrupt flag is sampled here (every Q1).
2: Interrupt latency = 3.25Tcy, where Tcy = instruction cycle time.
3: CLKOUT is available only in RC oscillator mode.
4: The timer clock (after the synchronizer circuit) which increments the timer from FFh to 00h immediately sets the T0IF bit. The TMR0 register will roll over 3 Tosc cycles later.

6.2 Using TMR0 with External Clock

When an external clock input is used for TMR0, 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 the TMR0 register 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 pin RA4/T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 6-5). Therefore, it is necessary for T0CKI to be high for at least 2Tosc (plus a small RC delay) and low for at least 2Tosc (plus a small RC delay). Refer to the electrical specification of the desired device.

When a prescaler is used, the external clock input is divided by an 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 (plus a small RC delay) 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 AC Electrical Specifications of the desired device.

6.2.2 TMR0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 Module is actually incremented. Figure 6-5 shows the delay from the external clock edge to the timer incrementing.

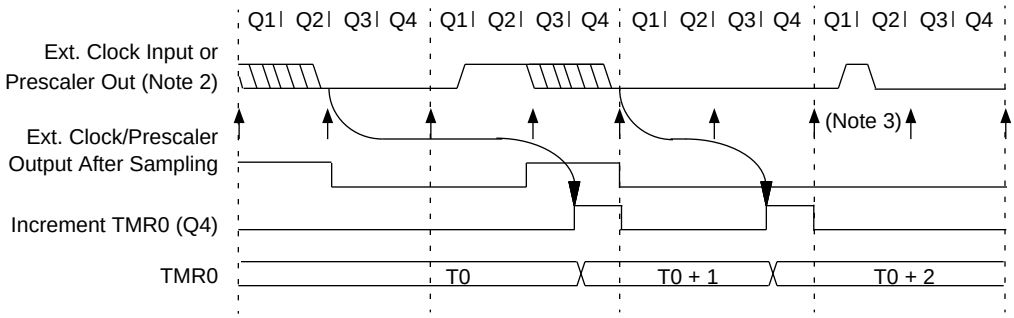
6.3 Prescaler

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

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

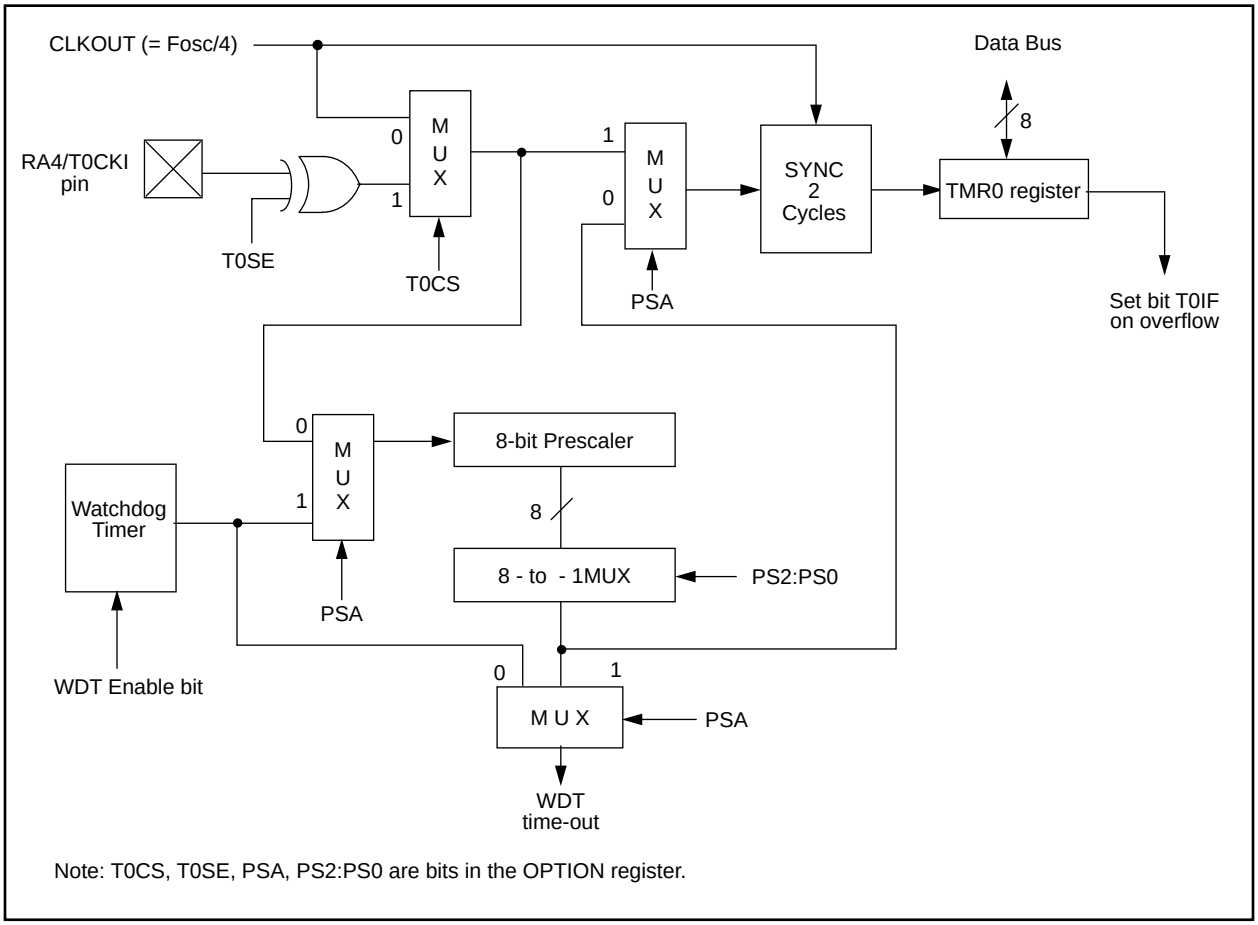
When assigned to the Timer0 Module, all instructions writing to the Timer0 Module (e.g., `CLRF 1`, `MOVWF 1`, `BSF 1,x` ....etc.) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

FIGURE 6-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Note 1: Delay from clock input change to TMR0 increment is 3Tosc to 7Tosc. (Duration of Q = Tosc). Therefore, the error in measuring the interval between two edges on TMR0 input = ± 4Tosc max.
2: External clock if no prescaler selected, Prescaler output otherwise.
3: The arrows ↑ indicate where sampling occurs. A small clock pulse may be missed by sampling.

FIGURE 6-6: BLOCK DIAGRAM OF THE TMR0/WDT PRESCALER



Note: T0CS, T0SE, PSA, PS2:PS0 are bits in the OPTION register.

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).

Note: To avoid an unintended device RESET, the following instruction sequence (Example 6-1) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be taken even if the WDT is disabled. To change prescaler from the WDT to the Timer0 module use the sequence shown in Example 6-2.

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

```
BCF     STATUS, RP0  ;Bank 0
CLRF    TMR0         ;Clear TMR0
                     ; and Prescaler
BSF     STATUS, RP0  ;Bank 1
CLRWDT               ;Clears WDT
MOVLW   b'xxxx1xxx'  ;Select new
MOVWF   OPTION       ; prescale value
BCF     STATUS, RP0  ;Bank 0
```

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

```
CLRWDT               ;Clear WDT and
                     ; prescaler
BSF     STATUS, RP0  ;Bank 1
MOVLW   b'xxxx0xxx'  ;Select TMR0, new
                     ; prescale value
                     ' and clock source
MOVWF   OPTION       ;
BCF     STATUS, RP0  ;Bank 0
```

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 Power-on Reset	Value on all other resets
01h	TMR0	Timer0 module’s register								xxxx xxxx	uuuu uuuu
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 0000
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: x = unknown, u = unchanged. - = unimplemented read as '0'. Shaded cells are not associated with Timer0.

NOTES:

7.0 DATA EEPROM MEMORY

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

- EECON1
- EECON2
- EEDATA
- EEADR

EEDATA holds the 8-bit data for read/write, and EEADR holds the address of the EEPROM location being accessed. PIC16C84 devices have 64 bytes of data EEPROM with an address range from 0h to 3Fh.

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

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

7.1 EEADR

The EEADR register can address up to a maximum of 256 bytes of data EEPROM. Only the first 64 bytes of data EEPROM are implemented.

The upper two bits are address decoded. This means that these two bits must always be '0' to ensure that the address is in the 64 byte memory space.

FIGURE 7-1: EECON1 REGISTER (ADDRESS 88h)

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

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

bit 7:5 **Unimplemented:** Read as '0'

bit 4 **EEIF**: EEPROM Write Operation Interrupt Flag bit
1 = The write operation completed (must be cleared in software)
0 = The write operation is not complete or has not been started

bit 3 **WRERR**: EEPROM Error Flag bit
1 = A write operation is prematurely terminated (any $\overline{MCLR}$ reset or any WDT reset during normal operation)
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

7.2 EECON1 and EECON2 Registers

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

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

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a $\overline{\text{MCLR}}$ 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 is set when write is complete. It 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.

7.3 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 7-1: DATA EEPROM READ

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

7.4 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 7-2: DATA EEPROM WRITE

Required Sequence

```
        BSF     STATUS, RP0  ; Bank 1
        BCF     INTCON, GIE  ; Disable INTs.
        BSF     EECON1, WREN ; Enable Write
        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.

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. EEIF must be cleared by software.

Note:	The data EEPROM memory E/W cycle time may occasionally exceed the 10 ms specification (typical). To ensure that the write cycle is complete, use the EE interrupt or poll the WR bit (EECON1<1>). Both these events signify the completion of the write cycle.

7.5 Write Verify

Depending on the application, good programming practice may dictate that the value written to the Data EEPROM should be verified (Example 7-3) to the desired value to be written. This should be used in applications where an EEPROM bit will be stressed near the specification limit. The Total Endurance disk will help determine your comfort level.

Generally the EEPROM write failure will be a bit which was written as a '1', but reads back as a '0' (due to leakage off the bit).

EXAMPLE 7-3: WRITE VERIFY

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

7.6 Protection Against Spurious Writes

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

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

7.7 Data EEPROM Operation during Code Protect

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

For ROM devices, there are two code protection bits (Section 8.1). One for the ROM program memory and one for the Data EEPROM memory.

7.8 Power Consumption Considerations

Note: It is recommended that the EEADR<7:6> bits be cleared. When either of these bits is set, the maximum IDD for the device is higher than when both are cleared. The specification is 400 μA. With EEADR<7:6> cleared, the maximum is approximately 150 μA.

TABLE 7-1: 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
08h	EEDATA	EEPROM data register								xxxx xxxx	uuuu uuuu
09h	EEADR	EEPROM address register								xxxx xxxx	uuuu uuuu
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	---0 q000
89h	EECON2	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.

NOTES:

8.0 SPECIAL FEATURES OF THE CPU

What sets a microcontroller apart from other processors are special circuits to deal with the needs of real time applications. The PIC16C84 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 features are:

- OSC selection
- Reset
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code protection
- ID locations
- In-circuit serial programming

The PIC16C84 has a Watchdog Timer which can be shut off only through 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. This design keeps the device in reset while the power supply stabilizes. With these two timers on-chip, most applications need no external reset circuitry.

SLEEP mode offers a very low current power-down mode. The user can wake-up from SLEEP through external reset, Watchdog Timer time-out or through an interrupt. Several oscillator options are provided 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 the various options.

8.1 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.

Address 2007h is beyond the user program memory space and it belongs to the special test/configuration memory space (2000h - 3FFFh). This space can only be accessed during programming.

FIGURE 8-1: CONFIGURATION WORD

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	R/P-u	R/P-u	R/P-u	R/P-u	R/P-u
—	—	—	—	—	—	—	—	—	CP	PWRTE	WDTE	FOSC1	FOSC0

bit13 bit0

R = Readable bit
P = Programmable bit
U = Unimplemented bit, read as '1'
- n = Value at POR reset
u = unchanged

bit 13:5 **Unimplemented:** Read as '1'

bit 4 **CP**: Code Protection bit
1 = Code protection off
0 = All memory is code protected

bit 3 **PWRTE**: Power-up Timer Enable bit
1 = Power-up timer is enabled
0 = Power-up timer is disabled

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

bit 1:0 **FOSC1:FOSC0**: Oscillator Selection bits
11 =RC oscillator
10 = HS oscillator
01 = XT oscillator
00 = LP oscillator

8.2 Oscillator Configurations

8.2.1 OSCILLATOR TYPES

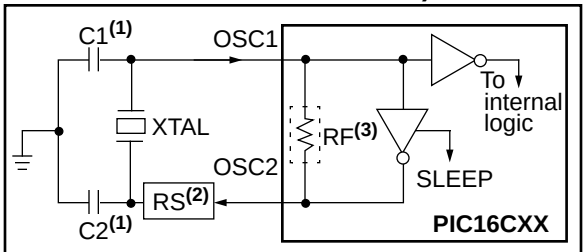
The PIC16C84 can be operated in four different oscillator modes. The user can program two configuration bits (FOSC1 and FOSC0) to select one of these four modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- RC Resistor/Capacitor

8.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, LP or HS modes a crystal or ceramic resonator is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 8-2).

FIGURE 8-2: CRYSTAL/CERAMIC RESONATOR OPERATION (HS, XT OR LP OSC CONFIGURATION)



Note1: See Table 8-1 and Table 8-2 for recommended values of C1 and C2.
2: A series resistor (RS) may be required for AT strip cut crystals.
3: RF varies with the crystal chosen.

The PIC16C84 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/CLKIN pin (Figure 8-3).

FIGURE 8-3: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

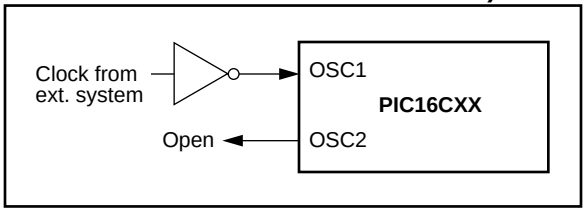


TABLE 8-1: PIC16C84 CAPACITOR SELECTION FOR CERAMIC RESONATORS

Ranges Tested:			
Mode	**Freq**	**OSC1/C1**	**OSC2/C2**
XT	455 kHz	47 - 100 pF	47 - 100 pF
	2.0 MHz	15 - 33 pF	15 - 33 pF
	4.0 MHz	15 - 33 pF	15 - 33 pF
HS	8.0 MHz	15 - 33 pF	15 - 33 pF
	10.0 MHz	15 - 33 pF	15 - 33 pF

Note: Recommended values of C1 and C2 are identical to the ranges tested table.
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 the appropriate values of external components.

Resonators Tested:		
455 kHz	Panasonic EFO-A455K04B	± 0.3%
2.0 MHz	Murata Erie CSA2.00MG	± 0.5%
4.0 MHz	Murata Erie CSA4.00MG	± 0.5%
8.0 MHz	Murata Erie CSA8.00MT	± 0.5%
10.0 MHz	Murata Erie CSA10.00MTZ	± 0.5%

None of the resonators had built-in capacitors.

TABLE 8-2: PIC16C84 CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Mode	Freq	OSC1/C1	OSC2/C2
LP	32 kHz	68 - 100 pF	68 - 100 pF
	200 kHz	15 - 33 pF	15 - 33 pF
XT	100 kHz	100 - 150 pF	100 - 150 pF
	2 MHz	15 - 33 pF	15 - 33 pF
	4 MHz	15 - 33 pF	15 - 33 pF
HS	4 MHz	15 - 33 pF	15 - 33 pF
	10 MHz	15 - 33 pF	15 - 33 pF

Note: Higher capacitance increases the stability of oscillator but also increases the start-up time. These values are for design guidance only. Rs may be required in HS mode as well as XT mode to avoid overdriving crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.
For V_{DD} > 4.5V, C1 = C2 ≈ 30 pF is recommended.

Crystals Tested:		
32.768 kHz	Epson C-001R32.768K-A	± 20 PPM
100 kHz	Epson C-2 100.00 KC-P	± 20 PPM
200 kHz	STD XTL 200.000 KHz	± 20 PPM
1.0 MHz	ECS ECS-10-13-2	± 50 PPM
2.0 MHz	ECS ECS-20-S-2	± 50 PPM
4.0 MHz	ECS ECS-40-S-4	± 50 PPM
10.0 MHz	ECS ECS-100-S-4	± 50 PPM

8.2.3 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 are available; one with series resonance, and one with parallel resonance.

Figure 8-4 shows a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180-degree phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides negative feedback for stability. The 10 kΩ potentiometer biases the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 8-4: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

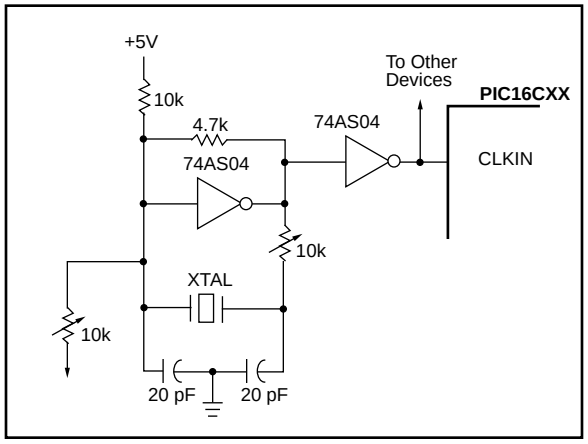
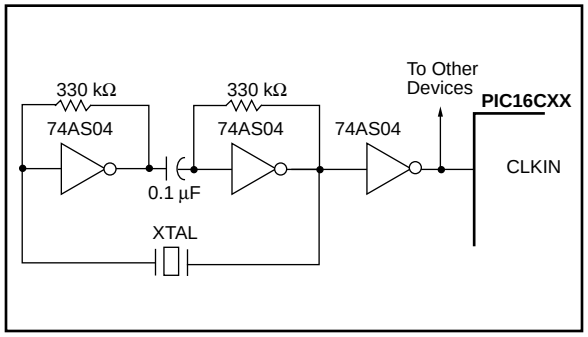


Figure 8-5 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 8-5: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



8.2.4 RC OSCILLATOR

For timing insensitive applications the RC device option offers additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (Rext) values, capacitor (Cext) values, and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal process parameter variation. Furthermore, the difference in lead frame capacitance between package types also affects the oscillation frequency, especially for low Cext values. The user needs to take into account variation due to tolerance of the external R and C components. Figure 8-6 shows how an R/C combination is connected to the PIC16C84. For Rext values below 2.2 kΩ, the oscillator operation may become unstable, or stop completely. For very high Rext values (e.g., 1 MΩ), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend keeping Rext between 3 kΩ and 100 kΩ.

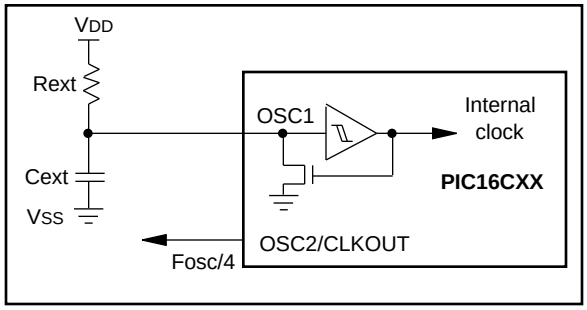
Although the oscillator will operate with no external capacitor (Cext = 0 pF), we recommend using values above 20 pF for noise and stability reasons. With little or no external capacitance, the oscillation frequency can vary dramatically due to changes in external capacitances, such as PCB trace capacitance or package lead frame capacitance.

See the electrical specification section for RC frequency variation from part to part due to normal process variation. The variation is larger for larger R (since leakage current variation will affect RC frequency more for large R) and for smaller C (since variation of input capacitance has a greater affect on RC frequency).

See the electrical specification section for variation of oscillator frequency due to VDD for given Rext/Cext values as well as frequency variation due to operating temperature.

The oscillator frequency, divided by 4, is available on the OSC2/CLKOUT pin, and can be used for test purposes or to synchronize other logic (see Figure 3-2 for waveform).

FIGURE 8-6: RC OSCILLATOR MODE



Note:	When the device oscillator is in RC mode, do not drive the OSC1 pin with an external clock or you may damage the device.

8.3 Reset

The PIC16C84 differentiates between various kinds of reset:

- Power-on Reset (POR)
- $\overline{MCLR}$ reset during normal operation
- $\overline{MCLR}$ reset during SLEEP
- WDT Reset (during normal operation)
- WDT Wake-up (during SLEEP)

Figure 8-7 shows a simplified block diagram of the on-chip reset circuit. The electrical specifications state the pulse width requirements for the $\overline{MCLR}$ pin.

Some registers are not affected in any reset condition; their status is unknown on a POR reset and unchanged in any other reset. Most other registers are reset to a "reset state" on POR, $\overline{MCLR}$ or WDT reset during normal operation and on $\overline{MCLR}$ reset during SLEEP. They are not affected by a WDT reset during SLEEP, since this reset is viewed as the resumption of normal operation.

Table 8-3 gives a description of reset conditions for the program counter (PC) and the STATUS register. Table 8-4 gives a full description of reset states for all registers.

The $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different reset situations (Section 8.7). These bits are used in software to determine the nature of the reset.

FIGURE 8-7: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT

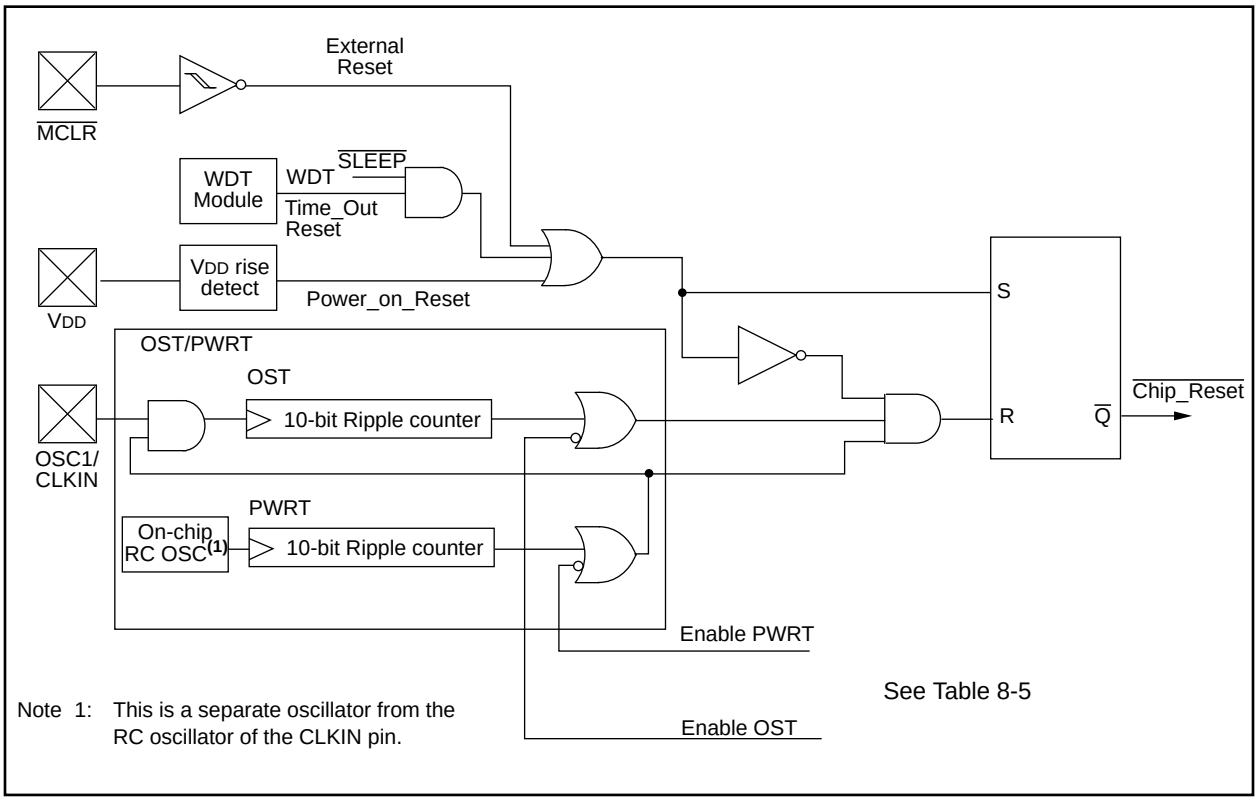


TABLE 8-3: RESET CONDITION FOR PROGRAM COUNTER AND THE STATUS REGISTER

Condition	Program Counter	STATUS Register
Power-on Reset	000h	0001 1xxx
$\overline{MCLR}$ Reset during normal operation	000h	000u uuuu
$\overline{MCLR}$ Reset during SLEEP	000h	0001 0uuu
WDT Reset (during normal operation)	000h	0000 1uuu
WDT Wake-up	PC + 1	uuu0 0uuu
Interrupt wake-up from SLEEP	PC + 1 (1)	uuu1 0uuu

Legend: u = unchanged, x = unknown.
Note 1: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

TABLE 8-4: RESET CONDITIONS FOR ALL REGISTERS

Register	Address	Power-on Reset	$\overline{MCLR}$ Reset during: – normal operation – SLEEP WDT Reset during normal operation	Wake-up from SLEEP: – through interrupt – through WDT time-out
W	—	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	00h	---- ----	---- ----	---- ----
TMR0	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h	0000h	0000h	PC + 1(2)
STATUS	03h	0001 1xxx	000q quuu(3)	uuuq quuu(3)
FSR	04h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	---x xxxx	---u uuuu	---u uuuu
PORTB	06h	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEDATA	08h	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEADR	09h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCLATH	0Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh	0000 000x	0000 000u	uuuu uuuu(1)
INDF	80h	---- ----	---- ----	---- ----
OPTION	81h	1111 1111	1111 1111	uuuu uuuu
PCL	82h	0000h	0000h	PC + 1
STATUS	83h	0001 1xxx	000q quuu(3)	uuuq quuu(3)
FSR	84h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TRISA	85h	---1 1111	---1 1111	---u uuuu
TRISB	86h	1111 1111	1111 1111	uuuu uuuu
EECON1	88h	---0 x000	---0 q000	---0 uuuu
EECON2	89h	---- ----	---- ----	---- ----
PCLATH	8Ah	---0 0000	---0 0000	---u uuuu
INTCON	8Bh	0000 000x	0000 000u	uuuu uuuu(1)

Legend: u = unchanged, x = unknown, - = unimplemented bit read as '0', q = value depends on condition.
Note 1: One or more bits in INTCON will be affected (to cause wake-up).
2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).
3: Table 8-3 lists the reset value for each specific condition.

8.4 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.2V - 1.7V). To take advantage of the POR, just tie the $\overline{MCLR}$ pin directly (or through a resistor) to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A minimum rise time for VDD must be met for this to operate properly. See Electrical Specifications for details.

When the device starts normal operation (exits the reset condition), device operating parameters (voltage, frequency, temperature, ...) must be meet 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*."

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

8.5 Power-up Timer (PWRT)

The Power-up Timer (PWRT) provides a fixed 72 ms nominal time-out (TPWRT) from POR (Figure 8-9, Figure 8-10, Figure 8-11 and Figure 8-12). The Power-up Timer operates on an internal RC oscillator. The chip is kept in reset as long as the PWRT is active. The PWRT delay allows the VDD to rise to an acceptable level (Possible exception shown in Figure 8-12).

A configuration bit, PWRTE, can enable/disable the PWRT (Figure 8-1).

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

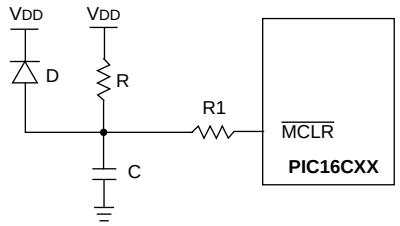
8.6 Oscillator Start-up Timer (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle delay (from OSC1 input) after the PWRT delay ends (Figure 8-9, Figure 8-10, Figure 8-11 and Figure 8-12). This ensures the crystal oscillator or resonator has started and stabilized.

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

When VDD rises very slowly, it is possible that the TPWRT time-out and TOST time-out will expire before VDD has reached its final value. In this case (Figure 8-12), an external power-on reset circuit may be necessary (Figure 8-8).

FIGURE 8-8: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External Power-on Reset circuit is required only if VDD power-up rate is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to make sure that voltage drop across R does not exceed 0.2V (max leakage current spec on $\overline{MCLR}$ pin is 5 μA). A larger voltage drop will degrade VIH level on the $\overline{MCLR}$ pin.

3: R1 = 100Ω to 1 kΩ will limit any current flowing into $\overline{MCLR}$ from external capacitor C in the event of an $\overline{MCLR}$ pin breakdown due to ESD or EOS.

FIGURE 8-9: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 1

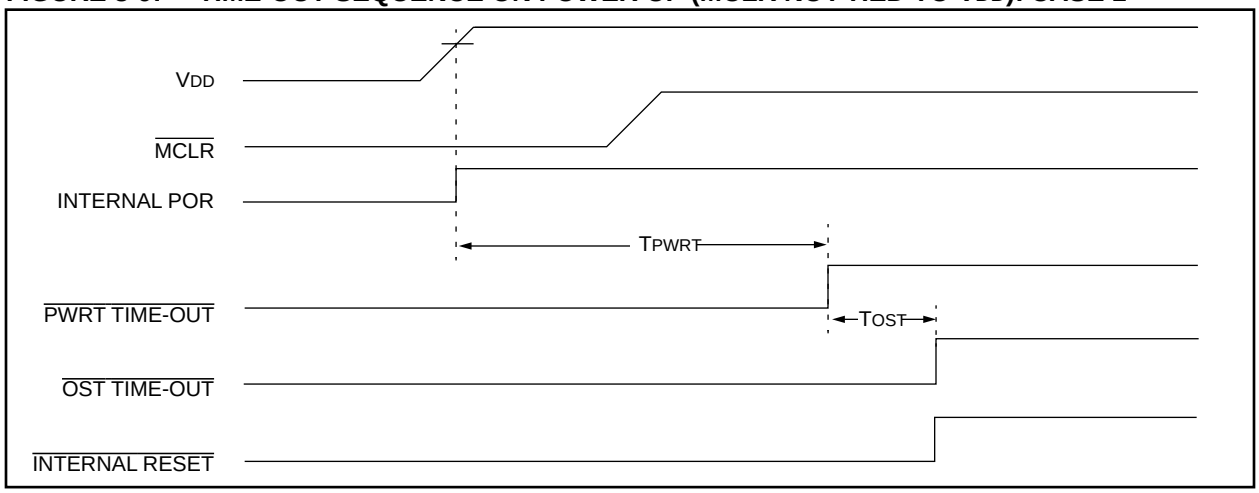


FIGURE 8-10: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 2

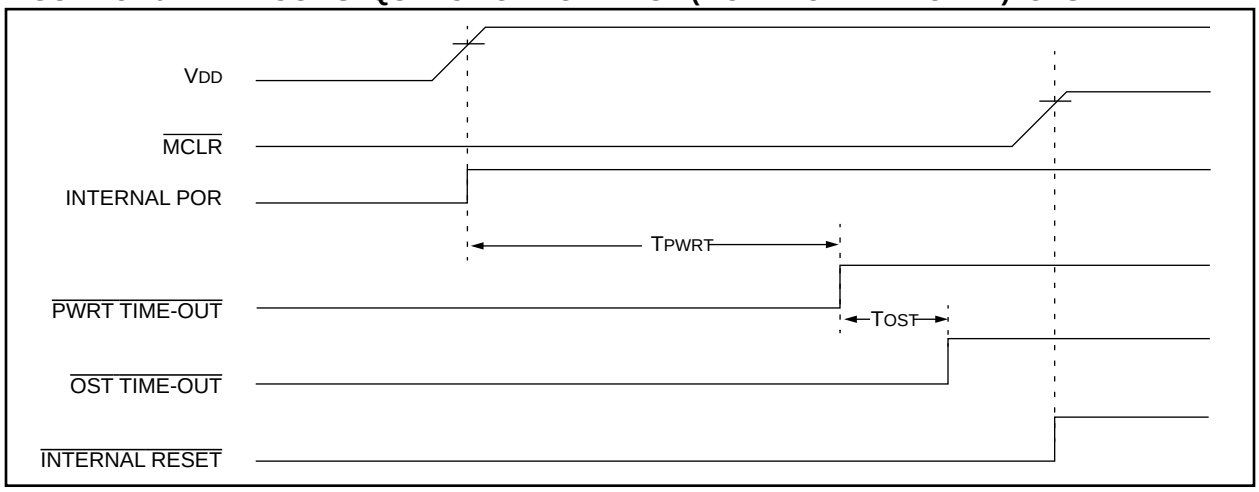


FIGURE 8-11: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO V_{DD}): FAST V_{DD} RISE TIME

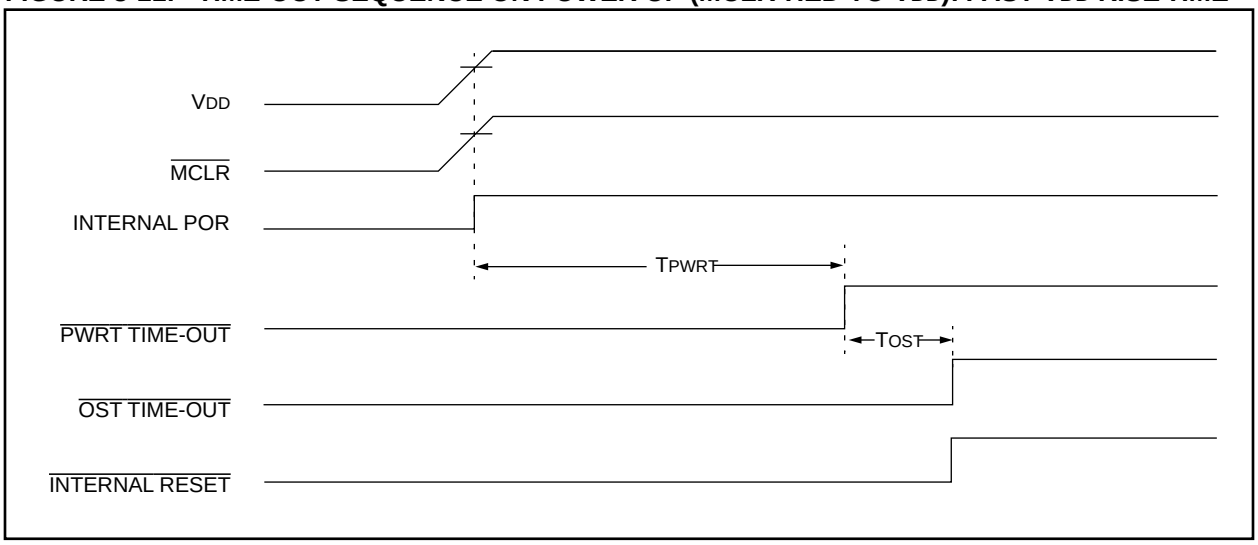
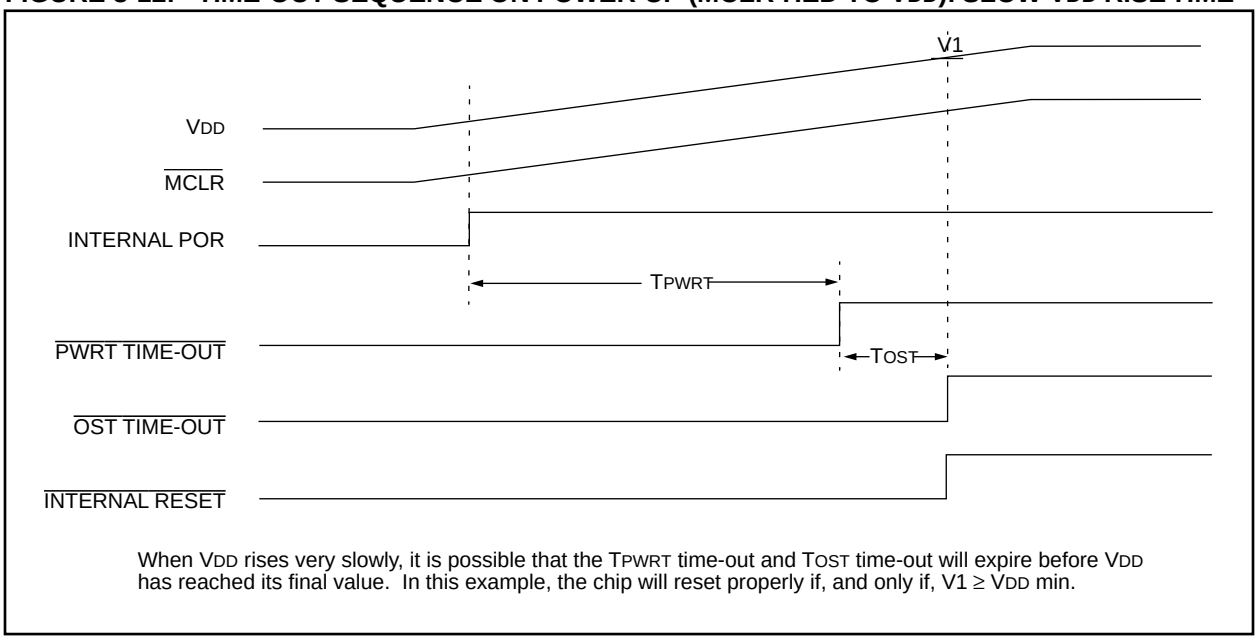


FIGURE 8-12: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO V_{DD}): SLOW V_{DD} RISE TIME



When V_{DD} rises very slowly, it is possible that the T_{PWRT} time-out and T_{OST} time-out will expire before V_{DD} has reached its final value. In this example, the chip will reset properly if, and only if, V1 ≥ V_{DD} min.

8.7 Time-out Sequence and Power Down Status Bits ($\overline{TO}$/$\overline{PD}$)

On power-up (Figure 8-9, Figure 8-10, Figure 8-11 and Figure 8-12) the time-out sequence is as follows: First PWRT time-out is invoked after a POR has expired. Then the OST is activated. The total time-out will vary based on oscillator configuration and PWRTE configuration bit status. For example, in RC mode with the PWRT disabled, there will be no time-out at all.

TABLE 8-5: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Wake-up from SLEEP
	PWRT Enabled	PWRT Disabled	
XT, HS, LP	72 ms + 1024TOSC	1024TOSC	1024TOSC
RC	72 ms	—	—

Since the time-outs occur from the POR reset pulse, if $\overline{MCLR}$ is kept low long enough, the time-outs will expire. Then bringing $\overline{MCLR}$ high, execution will begin immediately (Figure 8-9). This is useful for testing purposes or to synchronize more than one PIC16CXX device when operating in parallel.

Table 8-6 shows the significance of the $\overline{TO}$ and $\overline{PD}$ bits. Table 8-3 lists the reset conditions for some special registers, while Table 8-4 lists the reset conditions for all the registers.

TABLE 8-6: STATUS BITS AND THEIR SIGNIFICANCE

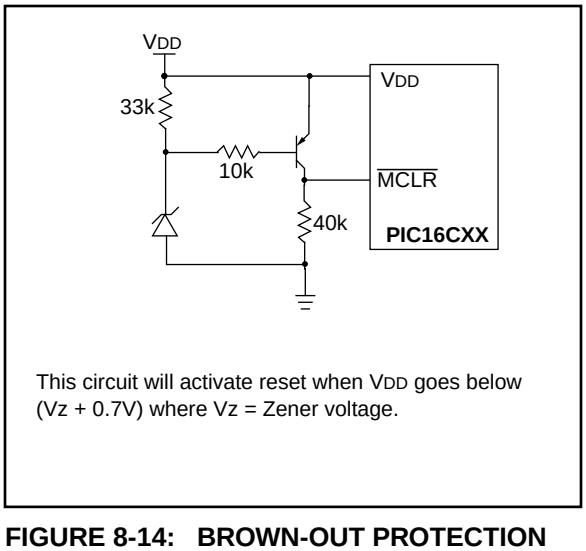
$\overline{TO}$	$\overline{PD}$	Condition
1	1	Power-on Reset
0	x	Illegal, $\overline{TO}$ is set on $\overline{POR}$
x	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
0	1	WDT Reset (during normal operation)
0	0	WDT Wake-up
1	1	$\overline{MCLR}$ Reset during normal operation
1	0	$\overline{MCLR}$ Reset during SLEEP or interrupt wake-up from SLEEP

8.8 Reset on Brown-Out

A brown-out is a condition where device power (VDD) dips below its minimum value, but not to zero, and then recovers. The device should be reset in the event of a brown-out.

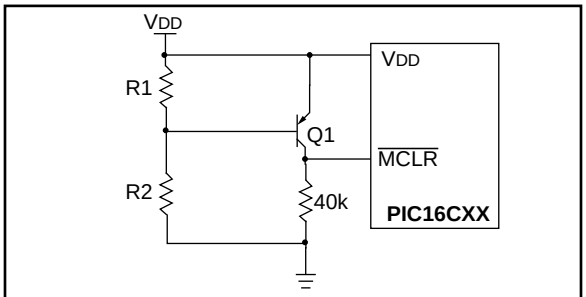
To reset PIC16C84 devices when a brown-out occurs, external brown-out protection circuits may be built, as shown in Figure 8-13 and Figure 8-14.

FIGURE 8-13: BROWN-OUT PROTECTION CIRCUIT 1



This circuit will activate reset when VDD goes below (Vz + 0.7V) where Vz = Zener voltage.

FIGURE 8-14: BROWN-OUT PROTECTION CIRCUIT 2



This brown-out circuit is less expensive, although less accurate. Transistor Q1 turns off when VDD is below a certain level such that:

$$V_{DD} \bullet \frac{R1}{R1 + R2} = 0.7V$$

8.9 Interrupts

The PIC16C84 has 4 sources of interrupt:

- External interrupt RB0/INT pin
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB7:RB4)
- EEPROM write complete interrupt

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

The 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. Bit GIE is cleared on reset.

The "return from interrupt" instruction, RETFIE, exits interrupt routine as well as sets the GIE bit, which re-enable interrupts.

The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

When an interrupt is responded to; the GIE bit is cleared to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with 0004h. For external interrupt events, such as the RB0/INT pin or PORTB change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 8-16). The latency is the same for both one and 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 infinite interrupt requests.

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

Note 2: If an interrupt occurs while the Global Interrupt Enable (GIE) bit is being cleared, the GIE bit may unintentionally be re-enabled by the user's Interrupt Service Routine (the RETFIE instruction). The events that would cause this to occur are:

1. An instruction clears the GIE bit while an interrupt is acknowledged
2. The program branches to the Interrupt vector and executes the Interrupt Service Routine.
3. The Interrupt Service Routine completes with the execution of the RETFIE instruction. This causes the GIE bit to be set (enables interrupts), and the program returns to the instruction after the one which was meant to disable interrupts.

The method to ensure that interrupts are globally disabled is:

1. Ensure that the GIE bit is cleared by the instruction, as shown in the following code:

```
LOOP  BCF   INTCON,GIE  ;Disable All
                        ;  Interrupts
      BTFSC INTCON,GIE  ;All Interrupts
                        ;  Disabled?
      GOTO  LOOP        ;NO, try again
                        ;  Yes, continue
                        ;  with program
                        ;  flow
```

FIGURE 8-15: INTERRUPT LOGIC

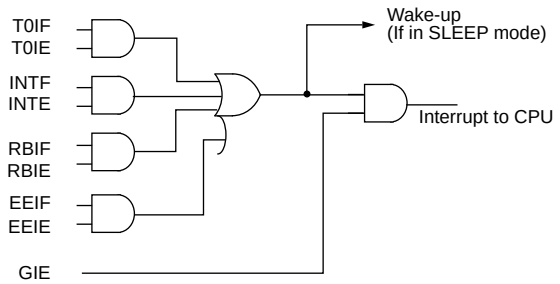
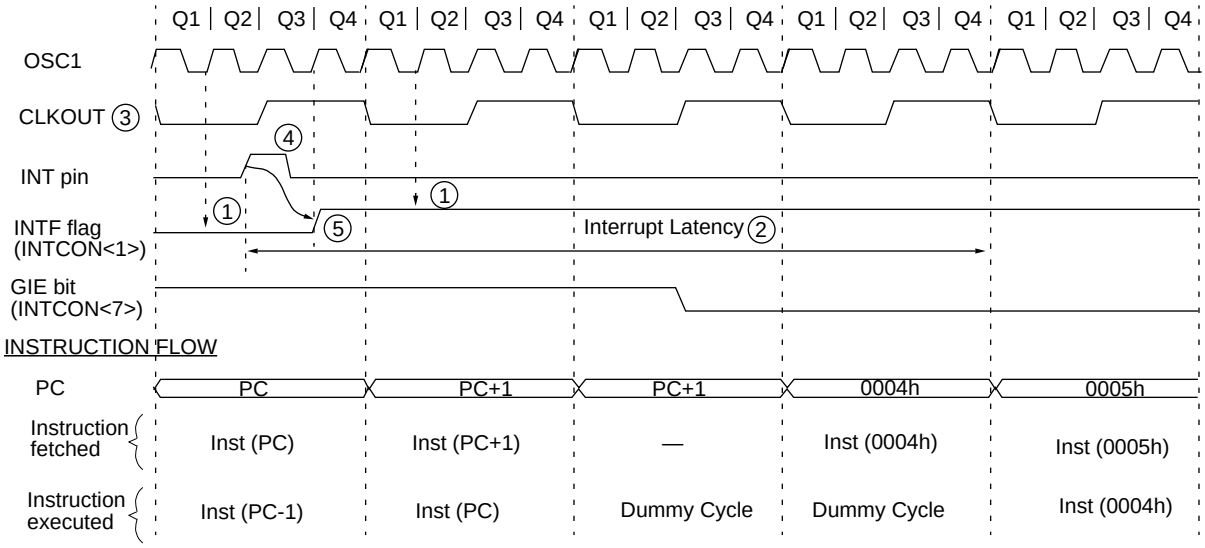


FIGURE 8-16: INT PIN INTERRUPT TIMING



Note 1: INTF flag is sampled here (every Q1).
2: Interrupt latency = 3-4Tcy where Tcy = instruction cycle time.
Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.
3: CLKOUT is available only in RC oscillator mode.
4: For minimum width of INT pulse, refer to AC specs.
5: INTF is enabled to be set anytime during the Q4-Q1 cycles.

8.9.1 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 control bit INTE (INTCON<4>). Flag bit INTF must be cleared in software via the interrupt service routine before re-enabling this interrupt. The INT interrupt can wake the processor from SLEEP (Section 8.12) only if the INTE bit was set prior to going into SLEEP. The status of the GIE bit decides whether the processor branches to the interrupt vector following wake-up.

8.9.2 TMR0 INTERRUPT

An overflow (FFh → 00h) in TMR0 will set flag bit T0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>) (Section 6.0).

8.9.3 PORT RB INTERRUPT

An input change on PORTB<7:4> sets flag bit RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit RBIE (INTCON<3>) (Section 5.2).

Note 1:	If a change on an I/O pin should occur when a read operation of PORTB is being executed (start of the Q2 cycle), the RBIF interrupt flag bit may not get set.

8.10 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users wish to save key register values during an interrupt (e.g., W register and STATUS register). This is implemented in software.

Example 8-1 stores and restores the STATUS and W register's values. The User defined registers, W_TEMP and STATUS_TEMP are the temporary storage locations for the W and STATUS registers values.

Example 8-1 does the following:

a) Stores the W register.
b) Stores the STATUS register in STATUS_TEMP.
c) Executes the Interrupt Service Routine code.
d) Restores the STATUS (and bank select bit) register.
e) Restores the W register.

EXAMPLE 8-1: SAVING STATUS AND W REGISTERS IN RAM

```
PUSH    MOVWF   W_TEMP          ; Copy W to TEMP register,
        SWAPF   STATUS, W       ; Swap status to be saved into W
        MOVWF   STATUS_TEMP     ; Save status to STATUS_TEMP register
ISR     :                       :
        :                       ; Interrupt Service Routine
        :                       ;  should configure Bank as required
        :                       ;
POP     SWAPF   STATUS_TEMP, W  ; Swap nibbles in STATUS_TEMP register
                                ; and place result into W
        MOVWF   STATUS          ; Move W into STATUS register
                                ;   (sets bank to original state)
        SWAPF   W_TEMP, F       ; Swap nibbles in W_TEMP and place result in W_TEMP
        SWAPF   W_TEMP, W       ; Swap nibbles in W_TEMP and place result into W
```

8.11 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 OSC1/CLKIN pin. That means that the WDT will run even if the clock on the OSC1/CLKIN and OSC2/CLKOUT 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 Wake-up causes the device to wake-up and continue with normal operation. The WDT can be permanently disabled by programming configuration bit WDTE as a '0' (Section 8.1).

8.11.1 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 specs). If longer time-out periods are desired, a prescaler with a division ratio of up to 1:128 can be assigned to the WDT under software control by writing to the OPTION register. Thus, time-out periods up to 2.3 seconds can be realized.

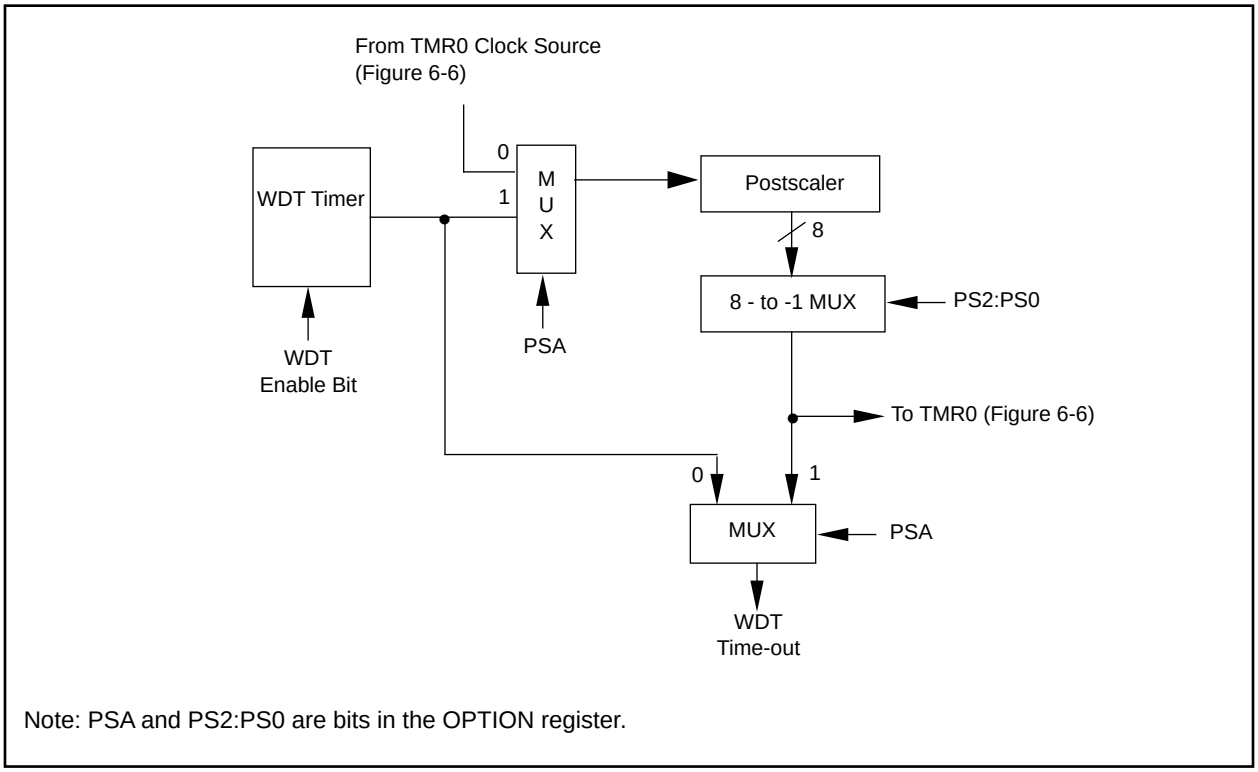
The CLRWDT and SLEEP instructions clear the WDT and the postscaler (if assigned to the WDT) and prevent it from timing out and generating a device RESET condition.

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

8.11.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken into 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 8-17: WATCHDOG TIMER BLOCK DIAGRAM



Note: PSA and PS2:PS0 are bits in the OPTION register.

TABLE 8-7: SUMMARY OF REGISTERS ASSOCIATED WITH THE WATCHDOG TIMER

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
2007h	Config. bits	(2)	(2)	(2)	(2)	PWRTE(1)	WDTE	FOSC1	FOSC0	(2)	
81h	OPTION	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown. Shaded cells are not used by the WDT.

Note 1: See Figure 8-1 for operation of the PWRTE bit.
2: See Figure 8-1 and Section 8.13 for operation of the Code and Data protection bits.

8.12 Power-down Mode (SLEEP)

A device may be powered down (SLEEP) and later powered up (Wake-up from SLEEP).

8.12.1 SLEEP

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

If enabled, the Watchdog Timer is cleared (but keeps running), the $\overline{PD}$ bit (STATUS<3>) is cleared, the $\overline{TO}$ bit (STATUS<4>) is set, and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, low, or hi-impedance).

For the lowest current consumption in SLEEP mode, place all I/O pins at either at V_{DD} or V_{SS}, with no external circuitry drawing current from the I/O pins, and disable external clocks. 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 V_{DD} or V_{SS}. The contribution from on-chip pull-ups on PORTB should be considered.

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

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

8.12.2 WAKE-UP FROM SLEEP

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

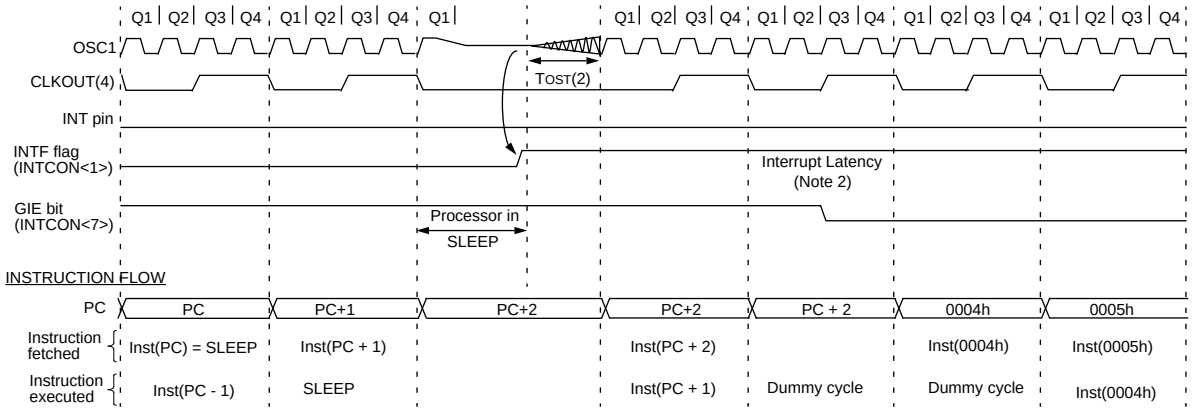
1. External reset input on $\overline{MCLR}$ pin.
2. WDT Wake-up (if WDT was enabled).
3. Interrupt from RB0/INT pin, RB port change, or data EEPROM write complete.

Peripherals cannot generate interrupts during SLEEP, since no on-chip Q clocks are present.

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

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

FIGURE 8-18: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: XT, HS or LP oscillator mode assumed.
2: T_{OST} = 1024T_{OSC} (drawing not to scale) This delay will not be there for RC osc mode.
3: GIE = '1' assumed. In this case after wake- up, the processor jumps to the interrupt routine. If GIE = '0', execution will continue in-line.
4: CLKOUT is not available in these osc modes, but shown here for timing reference.

8.12.3 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT postscaler will not be cleared, the $\overline{TO}$ bit will not be set and $\overline{PD}$ bits will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake up from sleep . The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

8.13 Program Verification/Code Protection

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

Note:	Microchip does not recommend code protecting windowed devices..

8.14 ID Locations

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

For ROM devices, these values are submitted along with the ROM code.

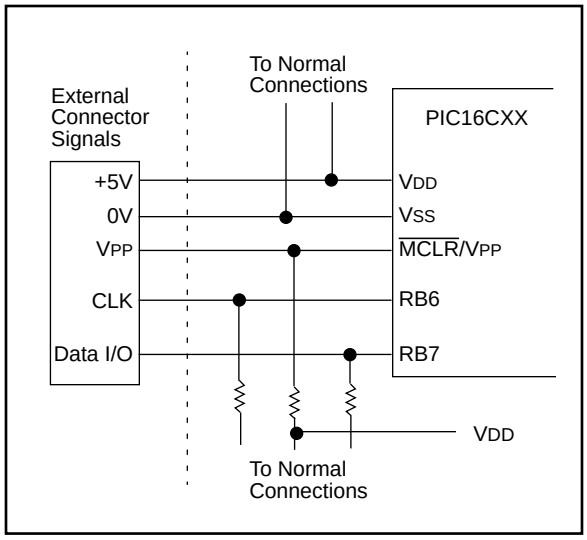
8.15 In-Circuit Serial Programming

PIC16C84 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. Customers can manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product, allowing the most recent firmware or 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 $\overline{MCLR}$ pin from V_{IL} to V_{IHH} (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) points to location 00h. A 6-bit command is then supplied to the device, 14-bits of program data is then supplied to or from the device, using load or read-type instructions. For complete details of serial programming, please refer to the PIC16CXX Programming Specifications (Literature #DS30189).

FIGURE 8-19: TYPICAL IN-SYSTEM SERIAL PROGRAMMING CONNECTION



For ROM devices, both the program memory and Data EEPROM memory may be read, but only the Data EEPROM memory may be programmed.

NOTES:

9.0 INSTRUCTION SET SUMMARY

Each PIC16CXX instruction is a 14-bit word divided into an OPCODE which specifies the instruction type and one or more operands which further specify the operation of the instruction. The PIC16CXX instruction set summary in Table 9-2 lists byte-oriented, bit-oriented, and literal and control operations. Table 9-1 shows the opcode field descriptions.

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

The destination designator specifies where the result of the operation is to be placed. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed in the file register specified by the instruction.

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

Literal and control operations: 'k' represents an eight or eleven bit constant or literal value.

TABLE 9-1: OPCODE FIELD DESCRIPTIONS

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

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

- Byte-oriented
- Bit-oriented
- Literal and control

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

Table 9-2 lists the instructions recognized by Microchip's assembler (MPASM).

Figure 9-1 shows the three general formats of instructions.

Note:	To maintain upward compatibility with future PIC16CXX products, do not use the OPTION and TRIS instructions.

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

0xhh

where h signifies a hexadecimal digit.

FIGURE 9-1: GENERAL FORMAT FOR INSTRUCTIONS

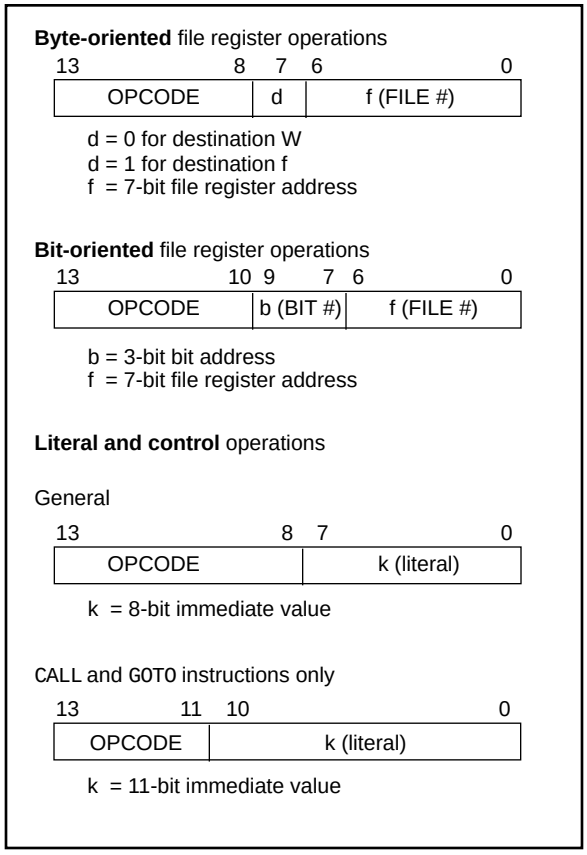


TABLE 9-2: INSTRUCTION SET SUMMARY

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb			LSb	Status Affected	Notes
ADDWF	**f, d**	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	**f, d**	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	**f**	Clear f	1	00	0001	lfff	ffff	Z	2
CLRW	**-**	Clear W	1	00	0001	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(2)	00	1011	dfff	ffff	None	1,2,3
INCF	**f, d**	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	**f, d**	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff	None	1,2,3
IORWF	**f, d**	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	**f, d**	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	**f**	Move W to f	1	00	0000	lfff	ffff	None	
NOP	**-**	No Operation	1	00	0000	0xx0	0000	None	
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	None	1,2
XORWF	**f, d**	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	**f, b**	Bit Clear f	1	01	00bb	bfff	ffff	None	1,2
BSF	**f, b**	Bit Set f	1	01	01bb	bfff	ffff	None	1,2
BTFSC	**f, b**	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff	None	3
BTFSS	**f, b**	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff	None	3
LITERAL AND CONTROL OPERATIONS									
ADDLW	**k**	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	**k**	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	**k**	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	**-**	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	**k**	Go to address	2	10	1kkk	kkkk	kkkk	None	
IORLW	**k**	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	**k**	Move literal to W	1	11	00xx	kkkk	kkkk	None	
RETFIE	**-**	Return from interrupt	2	00	0000	0000	1001	None	
RETLW	**k**	Return with literal in W	2	11	01xx	kkkk	kkkk	None	
RETURN	**-**	Return from subroutine	2	00	0000	0000	1000	None	
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 (i.e., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the TMR0.

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.

9.1 Instruction Descriptions

ADDLW **Add Literal and W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → (W)

Status Affected: C, DC, Z

Encoding:

11	111x	kkkk	kkkk

Description: The contents of the W register are added to the eight bit literal 'k' and the result is placed back in the W register.

Words: 1

Cycles: 1

Example
```
ADDLW   0x15
```
Before Instruction
W = 0x10
After Instruction
W = 0x25

ANDLW **AND Literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. (k) → (W)

Status Affected: Z

Encoding:

11	1001	kkkk	kkkk

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

Words: 1

Cycles: 1

Example
```
ANDLW   0x5F
```
Before Instruction
W = 0xA3
After Instruction
W = 0x03

ADDWF **Add W and f**

Syntax: [*label*] ADDWF f,d

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

Operation: (W) + (f) → (dest)

Status Affected: C, DC, Z

Encoding:

00	0111	dfff	ffff

Description: Add the contents of the W register to 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
```
ADDWF   FSR, 0
```
Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0xD9
FSR = 0xC2

ANDWF **AND W with f**

Syntax: [*label*] ANDWF f,d

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

Operation: (W) .AND. (f) → (dest)

Status Affected: Z

Encoding:

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
```
ANDWF   FSR, 1
```
Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0x17
FSR = 0x02

BCF **Bit Clear f**

Syntax: [*label*] BCF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 0 → (f<b>)

Status Affected: None

Encoding:

01	00bb	bfff	ffff

Description: Bit 'b' in register 'f' is cleared.

Words: 1

Cycles: 1

Example

```
BCF     FLAG_REG,7
```

Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

BSF **Bit Set f**

Syntax: [*label*] BSF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 1 → (f<b>)

Status Affected: None

Encoding:

01	01bb	bfff	ffff

Description: Bit 'b' in register 'f' is set.

Words: 1

Cycles: 1

Example

```
BSF     FLAG_REG,  7
```

Before Instruction
FLAG_REG= 0x0A
After Instruction
FLAG_REG= 0x8A

BTFSC **Bit Test f, Skip if Clear**

Syntax: [*label*] BTFSC f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 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 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example

```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

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

BTFSS **Bit Test f, skip if Set**

Syntax: [*label*] BTFSS f,b

Operands: 0 ≤ f ≤ 127
0 ≤ 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 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example

```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

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

CALL **Subroutine Call**

Syntax: [*label*] CALL k

Operands: 0 ≤ k ≤ 2047

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

Status Affected: None

Encoding:

10	0kkk	kkkk	kkkk

Description: Subroutine call. 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

CLRF **Clear f**

Syntax: [*label*] CLRF f

Operands: 0 ≤ f ≤ 127

Operation: 00h → (f)
1 → Z

Status Affected: Z

Encoding:

00	0001	1fff	ffff

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

Words: 1

Cycles: 1

Example

```
CLRF    FLAG_REG
```

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

CLRW **Clear W Register**

Syntax: [*label*] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Encoding:

00	0001	0000	0011

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

Words: 1

Cycles: 1

Example

```
CLRW
```

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

CLRWDT **Clear Watchdog Timer**

Syntax: [*label*] CLRWDT

Operands: None

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

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

Encoding:

00	0000	0110	0100

Description: The 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 prescale = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: [*label*] COMF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: ($\overline{f}$) → (dest)

Status Affected: Z

Encoding:

00	1001	dfff	ffff

Description: The contents of register 'f' are complemented. If 'd' is 0 the result is stored in W. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
COMF    REG1,0
```
Before Instruction
REG1 = 0x13
After Instruction
REG1 = 0x13
W = 0xEC

DECF **Decrement f**

Syntax: [*label*] DECF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) – 1 → (dest)

Status Affected: Z

Encoding:

00	0011	dfff	ffff

Description: Decrement register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
DECF    CNT, 1
```
Before Instruction
CNT = 0x01
Z = 0
After Instruction
CNT = 0x00
Z = 1

DECFSZ **Decrement f, Skip if 0**

Syntax: [*label*] DECFSZ f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) – 1 → (dest); skip if result = 0

Status Affected: None

Encoding:

00	1011	dfff	ffff

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

Words: 1

Cycles: 1(2)

Example
```
HERE     DECFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```
Before Instruction
PC = address HERE
After Instruction
CNT = CNT - 1
if CNT = 0,
PC = address CONTINUE
if CNT ≠ 0,
PC = address HERE+1

GOTO **Go to address**

Syntax: [*label*] GOTO k

Operands: 0 ≤ k ≤ 2047

Operation: k → (PC<10:0>)
(PCLATH<4:3>) → (PC<12:11>)

Status Affected: None

Encoding:

10	1kkk	kkkk	kkkk

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

Words: 1

Cycles: 2

Example
```
GOTO THERE
```
After Instruction
PC = Address THERE

INCF **Increment f**

Syntax: [*label*] INCF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) + 1 → (dest)

Status Affected: Z

Encoding:

00	1010	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Example
```
INCF CNT, 1
```
Before Instruction
CNT = 0xFF
Z = 0
After Instruction
CNT = 0x00
Z = 1

INCFSZ **Increment f, Skip if 0**

Syntax: [*label*] INCFSZ f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) + 1 → (dest), skip if result = 0

Status Affected: None

Encoding:

00	1111	dfff	ffff

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

Words: 1

Cycles: 1(2)

Example
```
HERE    INCFSZ   CNT,
1
        GOTO     LOOP
CONTINUE •
         •
         •
```
Before Instruction
PC = address HERE
After Instruction
CNT = CNT + 1
if CNT = 0,
PC = address CONTINUE
if CNT ≠ 0,
PC = address HERE +1

IORLW **Inclusive OR Literal with W**

Syntax: [*label*] IORLW k

Operands: 0 ≤ k ≤ 255

Operation: (W) .OR. (k) → (W)

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

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

Words: 1

Cycles: 1

Example
```
IORLW   0x35
```
Before Instruction
W = 0x9A
After Instruction
W = 0xBF

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f,d

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

Operation: (W) .OR. (f) → (W)

Status Affected: $\overline{Z}$

Encoding:

00	0100	dfff	ffff

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

Words: 1

Cycles: 1

Example
```
IORWF          RESULT, 0
```
Before Instruction
RESULT = 0x13
W = 0x91
After Instruction
RESULT = 0x13
W = 0x93

MOVF **Move f**

Syntax: [*label*] MOVF f,d

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

Operation: (f) → (dest)

Status Affected: Z

Encoding:

00	1000	dfff	ffff

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

Words: 1

Cycles: 1

Example
```
MOVF      FSR, 0
```
After Instruction
W =value in FSR register

MOVLW **Move literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: k → (W)

Status Affected: None

Encoding:

11	00XX	kkkk	kkkk

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

Words: 1

Cycles: 1

Example
```
MOVLW    0x5A
```
After Instruction
W = 0x5A

MOVWF **Move W to f**

Syntax: [*label*] MOVWF f

Operands: $0 \le f \le 127$

Operation: (W) → (f)

Status Affected: None

Encoding:

00	0000	1fff	ffff

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

Words: 1

Cycles: 1

Example
```
MOVWF       OPTION
```
Before Instruction
OPTION = 0xFF
W = 0x4F
After Instruction
OPTION = 0x4F
W = 0x4F

NOP — **No Operation**

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

00	0000	0xx0	0000

Description: No operation.

Words: 1

Cycles: 1

Example

```
NOP
```

RETFIE — **Return from Interrupt**

Syntax: [*label*] RETFIE

Operands: None

Operation: TOS → (PC),
1 → GIE

Status Affected: None

Encoding:

00	0000	0000	1001

Description: The Stack is popped and Top of Stack (TOS) is loaded into the PC. Interrupts are enabled by setting the Global Interrupt Enable bit. This is a two cycle instruction.

Words: 1

Cycles: 2

Example

```
RETFIE
```

After Interrupt
PC = TOS
GIE = 1

OPTION — **Load Option Register**

Syntax: [*label*] OPTION

Operands: None

Operation: (W) → OPTION

Status Affected: None

Encoding:

00	0000	0110	0010

Description: The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it.

Words: 1

Cycles: 1

Example

Note: To maintain upward compatibility with future PIC16CXX products, do not use this instruction.

RETLW — **Return Literal to W**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W),
TOS → (PC)

Status Affected: None

Encoding:

11	01xx	kkkk	kkkk

Description: The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two cycle instruction.

Words: 1

Cycles: 2

Example

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

Before Instruction
W = 0x07
After Instruction
W = value of k7

RETURN **Return from Subroutine**

Syntax: [*label*] RETURN

Operands: None

Operation: TOS → (PC)

Status Affected: None

Encoding:

00	0000	0000	1000

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

Words: 1

Cycles: 2

Example

```
RETURN
```

After Interrupt

```
PC  =  TOS
```

RRF **Rotate Right f through Carry**

Syntax: [*label*] RRF f,d

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

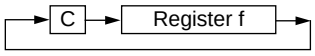
Operation: See description below

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       =   1
```

RLF **Rotate Left f through Carry**

Syntax: [*label*] RLF f,d

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

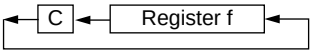
Operation: See description below

Status Affected: C

Encoding:

00	1101	dfff	ffff

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



Words: 1

Cycles: 1

Example

```
RLF             REG1,0
```

Before Instruction

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

After Instruction

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

SLEEP **Go into Standby Mode**

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.

Words: 1

Cycles: 1

Example: SLEEP

SUBLW **Subtract W from Literal**

Syntax: [*label*] SUBLW k

Operands: $0 \le k \le 255$

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 = FF
C = 0; result is negative

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f,d

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

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

Status Affected: C, DC, Z

Encoding:

00	0010	dfff	ffff

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

Words: 1

Cycles: 1

Example 1: `SUBWF REG1,1`

Before Instruction

REG1 = 3
W = 2
C = ?

After Instruction

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

Example 2: Before Instruction

REG1 = 2
W = 2
C = ?

After Instruction

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

Example 3: Before Instruction

REG1 = 1
W = 2
C = ?

After Instruction

REG1 = FF
W = 2
C = 0; result is negative

SWAPF **Swap f**

Syntax: [*label* SWAPF f,d]

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

Operation: (f<3:0>) → (dest<7:4>),
(f<7:4>) → (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

```
SWAP F REG,   0
```

Before Instruction

```
REG1   =   0xA5
```

After Instruction

```
REG1   =   0xA5
W      =   0x5A
```

TRIS **Load TRIS Register**

Syntax: [*label*] TRIS f

Operands: $5 \le f \le 7$

Operation: (W) → TRIS register (f)

Status Affected: None

Encoding:

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

Note: To maintain upward compatibility with future PIC16CXX products, do not use this instruction.

XORLW **Exclusive OR Literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → (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: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W) .XOR. (f) → (dest)

Status Affected: Z

Encoding:

00	0110	dfff	ffff

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

Words: 1

Cycles: 1

Example

```
XORWF     REG  1
```

Before Instruction

```
REG    =   0xAF
W      =   0xB5
```

After Instruction

```
REG    =   0x1A
W      =   0xB5
```

10.0 DEVELOPMENT SUPPORT

10.1 Development Tools

The PIC16/17 microcontrollers are supported with a full range of hardware and software development tools:

- PICMASTER/PICMASTER CE Real-Time In-Circuit Emulator
- ICEPIC Low-Cost PIC16C5X and PIC16CXX In-Circuit Emulator
- PRO MATE™ II Universal Programmer
- PICSTART® Plus Entry-Level Prototype Programmer
- PICDEM-1 Low-Cost Demonstration Board
- PICDEM-2 Low-Cost Demonstration Board
- PICDEM-3 Low-Cost Demonstration Board
- MPASM Assembler
- MPLAB-SIM Software Simulator
- MPLAB-C (C Compiler)
- Fuzzy logic development system (fuzzyTECH®–MP)

10.2 PICMASTER: High Performance Universal In-Circuit Emulator with MPLAB IDE

The PICMASTER Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for all microcontrollers in the PIC12C5XX, PIC16C5X, PIC16CXX and PIC17CXX families. PICMASTER is supplied with the MPLAB™ Integrated Development Environment (IDE), which allows editing, “make” and download, and source debugging from a single environment.

Interchangeable target probes allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the PICMASTER allows expansion to support all new Microchip microcontrollers.

The PICMASTER 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 compatible 386 (and higher) machine platform and Microsoft Windows® 3.x environment were chosen to best make these features available to you, the end user.

A CE compliant version of PICMASTER is available for European Union (EU) countries.

10.3 ICEPIC: Low-cost PIC16CXX In-Circuit Emulator

ICEPIC is a low-cost in-circuit emulator solution for the Microchip PIC16C5X and PIC16CXX families of 8-bit OTP microcontrollers.

ICEPIC is designed to operate on PC-compatible machines ranging from 286-AT® through Pentium™ based machines under Windows 3.x environment. ICEPIC features real time, non-intrusive emulation.

10.4 PRO MATE II: Universal Programmer

The PRO MATE II Universal Programmer is a full-featured programmer capable of operating in stand-alone mode as well as PC-hosted mode.

The PRO MATE II has programmable V_{DD} and V_{PP} supplies which allows it to verify programmed memory at V_{DD} min and V_{DD} max for maximum reliability. It has an LCD display for displaying error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In stand-alone mode the PRO MATE II can read, verify or program PIC16C5X, PIC16CXX, PIC17CXX and PIC14000 devices. It can also set configuration and code-protect bits in this mode.

10.5 PICSTART Plus Entry Level Development System

The PICSTART programmer is an easy-to-use, low-cost prototype programmer. It connects to the PC via one of the COM (RS-232) ports. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. PICSTART Plus is not recommended for production programming.

PICSTART Plus supports all PIC16/17 devices with up to 40 pins. Larger pin count devices such as the PIC16C923 and PIC16C924 may be supported with an adapter socket.

10.6 PICDEM-1 Low-Cost PIC16/17 Demonstration Board

The PICDEM-1 is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The users can program the sample microcontrollers provided with the PICDEM-1 board, on a PRO MATE II or PICSTART-16B programmer, and easily test firmware. The user can also connect the PICDEM-1 board to the PICMASTER emulator and download the firmware to the emulator for testing. Additional prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push-button switches and eight LEDs connected to PORTB.

10.7 PICDEM-2 Low-Cost PIC16CXX Demonstration Board

The PICDEM-2 is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-2 board, on a PRO MATE II programmer or PICSTART-16C, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-2 board to test firmware. Additional prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push-button switches, a potentiometer for simulated analog input, a Serial EEPROM to demonstrate usage of the I^2C bus and separate headers for connection to an LCD module and a keypad.

10.8 PICDEM-3 Low-Cost PIC16CXX Demonstration Board

The PICDEM-3 is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with a LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-3 board, on a PRO MATE II programmer or PICSTART Plus with an adapter socket, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-3 board to test firmware. Additional prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include an RS-232 interface, push-button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM-3 board is an LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM-3 provides an additional RS-232 interface and Windows 3.1 software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals. PICDEM-3 will be available in the 3rd quarter of 1996.

10.9 MPLAB Integrated Development Environment Software

The MPLAB IDE Software brings an ease of software development previously unseen in the 8-bit microcontroller market. MPLAB is a windows based application which contains:

- A full featured editor
- Three operating modes
  - editor
  - emulator
  - simulator
- A project manager
- Customizable tool bar and key mapping
- A status bar with project information
- Extensive on-line help

MPLAB allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PIC16/17 tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
- Transfer data dynamically via DDE (soon to be replaced by OLE)
- Run up to four emulators on the same PC

The ability to use MPLAB with Microchip's simulator allows a consistent platform and the ability to easily switch from the low cost simulator to the full featured emulator with minimal retraining due to development tools.

10.10 Assembler (MPASM)

The MPASM Universal Macro Assembler is a PC-hosted symbolic assembler. It supports all microcontroller series including the PIC16C5X, PIC16CXX, and PIC17CXX families.

MPASM offers full featured Macro capabilities, conditional assembly, and several source and listing formats. It generates various object code formats to support Microchip's development tools as well as third party programmers.

MPASM allows full symbolic debugging from the Microchip Universal Emulator System (PICMASTER).

MPASM has the following features to assist in developing software for specific use applications.

- Provides translation of Assembler source code to object code for all Microchip microcontrollers.
- Macro assembly capability.
- Produces all the files (Object, Listing, Symbol, and special) required for symbolic debug with Microchip's emulator systems.
- Supports Hex (default), Decimal and Octal source and listing formats.

MPASM provides a rich directive language to support programming of the PIC16/17. Directives are helpful in making the development of your assemble source code shorter and more maintainable.

10.11 Software Simulator (MPLAB-SIM)

The MPLAB-SIM Software Simulator allows code development in a PC host environment. It allows the user to simulate the PIC16/17 series microcontrollers on an instruction level. On any given instruction, the user may examine or modify any of the data areas or provide external stimulus to any of the pins. The input/output radix can be set by the user and the execution can be performed in; single step, execute until break, or in a trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C and MPASM. The Software Simulator offers the low cost flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

10.12 C Compiler (MPLAB-C)

The MPLAB-C Code Development System is a complete 'C' compiler and integrated development environment for Microchip's PIC16/17 family of microcontrollers. The compiler provides powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compiler provides symbol information that is compatible with the MPLAB IDE memory display (PICMASTER emulator software versions 1.13 and later).

10.13 Fuzzy Logic Development System (*fuzzy*TECH-MP)

*fuzzy*TECH-MP fuzzy logic development tool is available in two versions - a low cost introductory version, MP Explorer, for designers to gain a comprehensive working knowledge of fuzzy logic system design; and a full-featured version, *fuzzy*TECH-MP, edition for implementing more complex systems.

Both versions include Microchip's *fuzzy*LAB™ demonstration board for hands-on experience with fuzzy logic systems implementation.

10.14 MP-DriveWay™ – Application Code Generator

MP-DriveWay is an easy-to-use Windows-based Application Code Generator. With MP-DriveWay you can visually configure all the peripherals in a PIC16/17 device and, with a click of the mouse, generate all the initialization and many functional code modules in C language. The output is fully compatible with Microchip's MPLAB-C C compiler. The code produced is highly modular and allows easy integration of your own code. MP-DriveWay is intelligent enough to maintain your code through subsequent code generation.

10.15 SEEVAL® Evaluation and Programming System

The SEEVAL SEEPROM Designer's Kit supports all Microchip 2-wire and 3-wire Serial EEPROMs. The kit includes everything necessary to read, write, erase or program special features of any Microchip SEEPROM product including Smart Serials™ and secure serials. The Total Endurance™ Disk is included to aid in trade-off analysis and reliability calculations. The total kit can significantly reduce time-to-market and result in an optimized system.

10.16 TrueGauge® Intelligent Battery Management

The TrueGauge development tool supports system development with the MTAXXXXX TrueGauge Intelligent Battery Management IC. System design verification can be accomplished before hardware prototypes are built. User interface is graphically-oriented and measured data can be saved in a file for exporting to Microsoft Excel.

TABLE 10-1: DEVELOPMENT TOOLS FROM MICROCHIP

Product	** MPLAB™ Integrated Development Environment	MPLAB™ C Compiler	MP-DriveWay Applications Code Generator	fuzzyTECH®-MP Explorer/Edition Fuzzy Logic Dev. Tool	*** PICMASTER® PICMASTER-CE In-Circuit Emulator	ICEPIC Low-Cost In-Circuit Emulator	****PRO MATE™ II Universal Microchip Programmer	PICSTART® Lite Ultra Low-Cost Dev. Kit	PICSTART® Plus Low-Cost Universal Dev. Kit
PIC12C508, 509	SW007002	SW006005	—	—	EM167015/ EM167101	—	DV007003	—	DV003001
PIC14000	SW007002	SW006005	—	—	EM147001/ EM147101	—	DV007003	—	DV003001
PIC16C52, 54, 54A, 55, 56, 57, 58A	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167015/ EM167101	EM167201	DV007003	DV162003	DV003001
PIC16C554, 556, 558	SW007002	SW006005	—	DV005001/ DV005002	EM167033/ EM167113	—-	DV007003	—	DV003001
PIC16C61	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167021/ N/A	EM167205	DV007003	DV162003	DV003001
PIC16C62, 62A, 64, 64A	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167025/ EM167103	EM167203	DV007003	DV162002	DV003001
PIC16C620, 621, 622	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167023/ EM167109	EM167202	DV007003	DV162003	DV003001
PIC16C63, 65, 65A, 73, 73A, 74, 74A	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167025/ EM167103	EM167204	DV007003	DV162002	DV003001
PIC16C642, 662*	SW007002	SW006005	—	—	EM167035/ EM167105	—-	DV007003	DV162002	DV003001
PIC16C71	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167027/ EM167105	EM167205	DV007003	DV162003	DV003001
PIC16C710, 711	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167027/ EM167105	—	DV007003	DV162003	DV003001
PIC16C72	SW007002	SW006005	SW006006	—	EM167025/ EM167103	—	DV007003	DV162002	DV003001
PIC16F83	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167029/ EM167107	—	DV007003	DV162003	DV003001
PIC16C84	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167029/ EM167107	EM167206	DV007003	DV162003	DV003001
PIC16F84	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167029/ EM167107	—	DV007003	DV162003	DV003001
PIC16C923, 924*	SW007002	SW006005	SW006006	DV005001/ DV005002	EM167031/ EM167111	—	DV007003	—	DV003001
PIC17C42, 42A, 43, 44	SW007002	SW006005	SW006006	DV005001/ DV005002	EM177007/ EM177107	—	DV007003	—	DV003001

*Contact Microchip Technology for availability date
**MPLAB Integrated Development Environment includes MPLAB-SIM Simulator and MPASM Assembler

***All PICMASTER and PICMASTER-CE ordering part numbers above include PRO MATE II programmer
****PRO MATE socket modules are ordered separately. See development systems ordering guide for specific ordering part numbers

Product	TRUEGAUGE® Development Kit	SEEVAL® Designers Kit	Hopping Code Security Programmer Kit	Hopping Code Security Eval/Demo Kit
All 2 wire and 3 wire Serial EEPROM's	N/A	DV243001	N/A	N/A
MTA11200B	DV114001	N/A	N/A	N/A
HCS200, 300, 301 *	N/A	N/A	PG306001	DM303001

11.0 ELECTRICAL CHARACTERISTICS FOR PIC16C84

Absolute Maximum Ratings †

Ambient temperature under bias ..... -55°C to +125°C

Storage temperature ..... -65°C to +150°C

Voltage on V_{DD} with respect to V_{SS} ..... -0.3 to +7.5V

Voltage on $\overline{MCLR}$ with respect to V_{SS} (2) ..... -0.3 to +14V

Voltage on all other pins with respect to V_{SS} ..... -0.6V to (V_{DD} + 0.6V)

Total power dissipation (1) ..... 800 mW

Maximum current out of V_{SS} pin ..... 150 mA

Maximum current into V_{DD} pin ..... 100 mA

Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ±20 mA

Output clamp current, I_{OK} (V0 < 0 or V0 > V_{DD}) ..... ±20 mA

Maximum output current sunk by any I/O pin ..... 25 mA

Maximum output current sourced by any I/O pin ..... 20 mA

Maximum current sunk by PORTA ..... 80 mA

Maximum current sourced by PORTA ..... 50 mA

Maximum current sunk by PORTB ..... 150 mA

Maximum current sourced by PORTB ..... 100 mA

Note 1: Power dissipation is calculated as follows: Pdis = V_{DD} x {I_{DD} - $\sum I_{OH}$} + $\sum$ {(V_{DD}-V_{OH}) x I_{OH}} + $\sum$(V_{OL} x I_{OL})

Note 2: Voltage spikes below V_{SS} at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin rather than pulling this pin directly to V_{SS}.

† 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.

TABLE 11-1: CROSS REFERENCE OF DEVICE SPECS FOR OSCILLATOR CONFIGURATIONS AND FREQUENCIES OF OPERATION (COMMERCIAL DEVICES)

OSC	PIC16C84-04	PIC16C84-10	PIC16LC84-04
RC	V_{DD}: 4.0V to 6.0V I_{DD}: 4.5 mA max. at 5.5V I_{PD}: 100 µA max. at 4.0V WDT dis Freq: 4.0 MHz max.	V_{DD}: 4.5V to 5.5V I_{DD}: 1.8 mA typ. at 5.5V I_{PD}: 40.0 µA typ. at 4.5V WDT dis Freq: 4.0 MHz max.	V_{DD}: 2.0V to 6.0V I_{DD}: 4.5 mA max. at 5.5V I_{PD}: 100 µA max. at 4V WDT dis Freq: 2.0 MHz max.
XT	V_{DD}: 4.0V to 6.0V I_{DD}: 4.5 mA max. at 5.5V I_{PD}: 100 µA max. at 4.0V WDT dis Freq: 4.0 MHz max.	V_{DD}: 4.5V to 5.5V I_{DD}: 1.8 mA typ. at 5.5V I_{PD}: 40.0 µA typ. at 4.5V WDT dis Freq: 4.0 MHz max.	V_{DD}: 2.0V to 6.0V I_{DD}: 4.5 mA max. at 5.5V I_{PD}: 100 µA max. at 4V WDT dis Freq: 2.0 MHz max.
HS	V_{DD}: 4.5V to 5.5V I_{DD}: 4.5 mA typ. at 5.5V I_{PD}: 40.0 µA typ. at 4.5V WDT dis Freq: 4.0 MHz max.	V_{DD}: 4.5V to 5.5V I_{DD}: 10 mA max. at 5.5V typ. I_{PD}: 40.0 µA typ. at 4.5V WDT dis Freq: 10 MHz max.	Do not use in HS mode
LP	V_{DD}: 4.0V to 6.0V I_{DD}: 60 µA typ. at 32 kHz, 2.0V I_{PD}: 26 µA typ. at 2.0V WDT dis Freq: 200 kHz max.	Do not use in LP mode	V_{DD}: 2.0V to 6.0V I_{DD}: 400 µA max. at 32 kHz, 2.0V I_{PD}: 100 µA max. at 4.0V WDT dis Freq: 200 kHz max.

The shaded sections indicate oscillator selections which are tested for functionality, but not for MIN/MAX specifications. It is recommended that the user select the device type that ensures the specifications required.

11.1 DC CHARACTERISTICS: PIC16C84-04 (Commercial, Industrial) PIC16C84-10 (Commercial, Industrial)

DC Characteristics Power Supply Pins			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial)				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D001	VDD	Supply Voltage	4.0	—	6.0	V	XT, RC and LP osc configuration
D001A			4.5	—	5.5	V	HS osc configuration
D002	VDR	RAM Data Retention Voltage(1)	1.5*	—	—	V	Device in SLEEP mode
D003	VPOR	VDD start voltage to ensure internal Power-on Reset signal	—	VSS	—	V	See section on Power-on Reset for details
D004	SVDD	VDD rise rate to ensure internal Power-on Reset signal	0.05*	—	—	V/ms	See section on Power-on Reset for details
	IDD	Supply Current (2)					RC and XT osc configuration(4)
D010			—	1.8	4.5	mA	FOSC = 4 MHz, VDD = 5.5V
D010A			—	7.3	10	mA	FOSC = 4 MHz, VDD = 5.5V (During EEPROM programming)
D013			—	5.0	10	mA	HS osc configuration (PIC16C84-10) FOSC = 10 MHz, VDD = 5.5V
D020	IPD	Power-down Current(3)	—	40	100	µA	VDD = 4.0V, WDT enabled, industrial
D021			—	38	100	µA	VDD = 4.0V, WDT disabled, commercial
D021A			—	38	100	µA	VDD = 4.0V, WDT disabled, industrial

* These parameters are characterized but not tested.

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

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail to rail; all I/O pins tristated, pulled to VDD, T0CKI = VDD, $\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: The power down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD or VSS.

4: For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula IR = VDD/2Rext (mA) with Rext in kOhm.

11.2 DC CHARACTERISTICS: PIC16LC84-04 (Commercial, Industrial)

DC Characteristics Power Supply Pins			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial)				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D001	VDD	Supply Voltage	2.0	—	6.0	V	XT, RC, and LP osc configuration
D002	VDR	RAM Data Retention Voltage(1)	1.5 *	—	—	V	Device in SLEEP mode
D003	VPOR	VDD start voltage to ensure internal Power-on Reset signal	—	VSS	—	V	See section on Power-on Reset for details
D004	SVDD	VDD rise rate to ensure internal Power-on Reset signal	0.05*	—	—	V/ms	See section on Power-on Reset for details
	IDD	Supply Current(2)					RC and XT osc configuration(4)
D010			—	1	4	mA	FOSC = 2 MHz, VDD = 5.5V
D010A			—	7.3	10	mA	FOSC = 2 MHz, VDD = 5.5V (During EEPROM programming)
							LP osc configuration
D014			—	60	400	µA	FOSC = 32 kHz, VDD = 2.0V WDT disabled
D020	IPD	Power-down Current(3)	—	26	100	µA	VDD = 2.0V, WDT enabled, industrial
D021			—	26	100	µA	VDD = 2.0V, WDT disabled, commercial
D021A			—	26	100	µA	VDD = 2.0V, WDT disabled, industrial

* These parameters are characterized but not tested.

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

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1=external square wave, from rail to rail; all I/O pins tristated, pulled to VDD, T0CKI = VDD, $\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: The power down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD or VSS.

4: For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula IR = VDD/2Rext (mA) with Rext in kOhm.

11.3 DC CHARACTERISTICS: PIC16C84-04 (Commercial, Industrial) PIC16C84-10 (Commercial, Industrial) PIC16LC84-04 (Commercial, Industrial)

DC Characteristics All Pins Except Power Supply Pins			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) Operating voltage VDD range as described in DC spec Section 11-1 and Section 11.2.				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage** I/O ports					
D030		with TTL buffer	VSS	—	0.8	V	4.5 ≤ VDD ≤ 5.5V
D030A			VSS	—	0.16VDD	V	entire range(4)
D031		with Schmitt Trigger buffer	VSS	—	0.2VDD	V	entire range
D032		MCLR, RA4/T0CKI, OSC1 (RC mode)	Vss	—	0.2VDD	V	
D033		OSC1 (XT, HS and LP modes)(1)	Vss	—	0.3VDD	V	
	VIH	**Input High Voltage** I/O ports					
D040		with TTL buffer	0.36VDD	—	VDD	V	4.5 ≤ VDD ≤ 5.5V
D040A			0.48VDD				entire range(4)
D041		with Schmitt Trigger buffer	0.45VDD	—	VDD		entire range
D042		MCLR, RA4/T0CKI, OSC1 (RC mode)	0.85VDD	—	VDD	V	
D043		OSC1 (XT, HS and LP modes)(1)	0.7VDD	—	VDD	V	
D050	VHYS	Hysteresis of Schmitt Trigger inputs	TBD	—	—	V	
D070	IPURB	PORTB weak pull-up current	50*	250*	400*	µA	VDD = 5V, VPIN = VSS
		Input Leakage Current(2,3)					
D060	IIL	I/O ports	—	—	±1	µA	Vss ≤ VPIN ≤ VDD, Pin at hi-impedance
D061		MCLR, RA4/T0CKI	—	—	±5	µA	Vss ≤ VPIN ≤ VDD
D063		OSC1/CLKIN	—	—	±5	µA	Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
		Output Low Voltage					
D080	VOL	I/O ports	—	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V
D083		OSC2/CLKOUT (RC osc configuration)	—	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V
		Output High Voltage					
D090	VOH	I/O ports(3)	VDD - 0.7	—	—	V	IOH = -3.0 mA, VDD = 4.5V
D093		OSC2/CLKOUT (RC osc configuration)	VDD - 0.7	—	—	V	IOH = -1.3 mA, VDD = 4.5V

* 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/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C84 be driven with external clock in RC mode.

2: The leakage current on the MCLR pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as coming out of the pin.

4: The user may use better of the two specs.

11.4 DC CHARACTERISTICS: PIC16C84-04 (Commercial, Industrial) PIC16C84-10 (Commercial, Industrial) PIC16LC84-04 (Commercial, Industrial)

DC Characteristics All Pins Except Power Supply Pins			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) Operating voltage VDD range as described in DC spec Section 11-1 and Section 11.2.				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Capacitive Loading Specs on Output Pins					
D100	COSC2	OSC2/CLKOUT pin	—	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1.
D101	CIO	All I/O pins and OSC2 (RC mode)	—	—	50	pF	
		Data EEPROM Memory					
D120	ED	Endurance	1M	10M	—	E/W	25°C at 5V
D121	VDRW	VDD for read/write	VMIN	—	6.0	V	VMIN = Minimum operating voltage
D122	TDEW	Erase/Write cycle time(1)	—	—	10	ms	
		Program EEPROM Memory					
D130	EP	Endurance	100	1000	—	E/W	
D131	VPR	VDD for read	VMIN	—	6.0	V	VMIN = Minimum operating voltage
D132	VPEW	VDD for erase/write	4.5	—	5.5	V	
D133	TPEW	Erase/Write cycle time(1)	—	10	—	ms	

† 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: The user should use interrupts or poll the EEIF or WR bits to ensure the write cycle has completed.

TABLE 11-2: TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

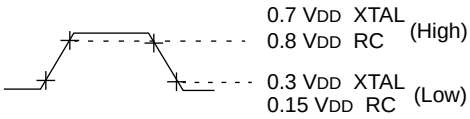
Lowercase symbols (pp) and their meanings:

pp			
2	to	os,osc	OSC1
ck	CLKOUT	ost	oscillator start-up timer
cy	cycle time	pwrt	power-up timer
io	I/O port	rbt	RBx pins
inp	INT pin	t0	T0CKI
mc	$\overline{\text{MCLR}}$	wdt	watchdog timer

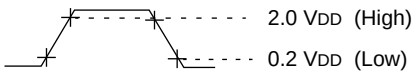
Uppercase symbols and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance

FIGURE 11-1: PARAMETER MEASUREMENT INFORMATION



OSC1 Measurement Points

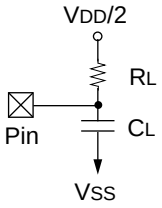


I/O Port Measurement Points

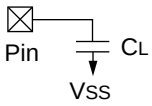
All timings are measured between high and low measurement points as indicated in the figure.

FIGURE 11-2: LOAD CONDITIONS

Load Condition 1



Load Condition 2



R_L = 464Ω

C_L = 50 pF for all pins except OSC2.

15 pF for OSC2 output.

11.5 Timing Diagrams and Specifications

FIGURE 11-3: EXTERNAL CLOCK TIMING

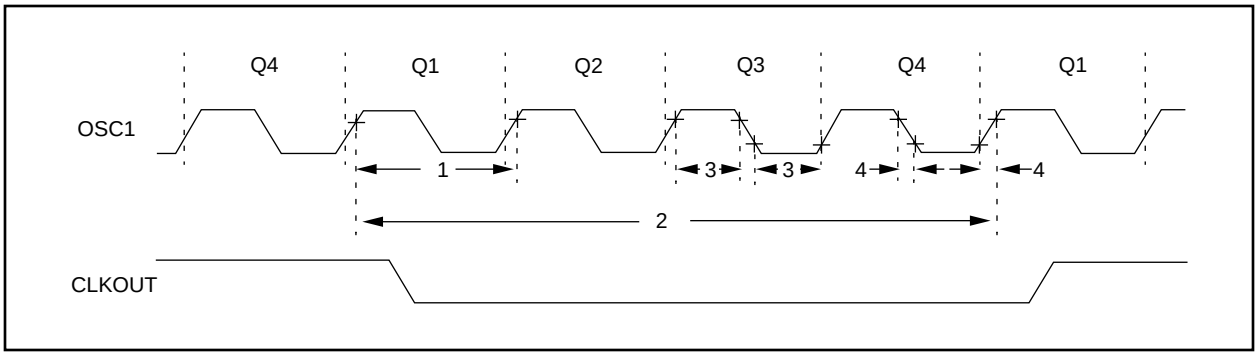


TABLE 11-3: EXTERNAL CLOCK TIMING REQUIREMENTS

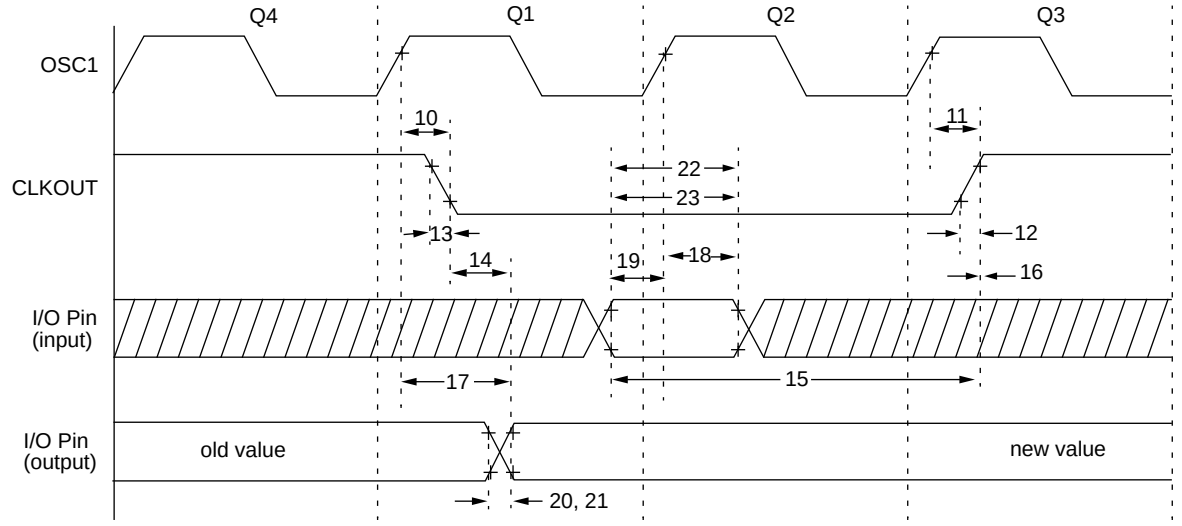
Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions	
	FOSC	**External CLKIN Frequency(1)**	DC	—	2	MHz	XT, RC osc	PIC16LC84-04
			DC	—	4	MHz	XT, RC osc	PIC16C84-04
			DC	—	10	MHz	HS osc	PIC16C84-10
			DC	—	200	kHz	LP osc	PIC16LC84-04
		Oscillator Frequency(1)	DC	—	2	MHz	RC osc	PIC16LC84-04
			DC	—	4	MHz	RC osc	PIC16C84-04
			0.1	—	2	MHz	XT osc	PIC16LC84-04
			0.1	—	4	MHz	XT osc	PIC16C84-04
			1	—	10	MHz	HS osc	PIC16C84-10
			DC	—	200	kHz	LP osc	PIC16LC84-04
1	Tosc	**External CLKIN Period(1)**	500	—	—	ns	XT, RC osc	PIC16LC84-04
			250	—	—	ns	XT, RC osc	PIC16C84-04
			100	—	—	ns	HS osc	PIC16C84-10
			5	—	—	µs	LP osc	PIC16LC84-04
		Oscillator Period(1)	500	—	—	ns	RC osc	PIC16LC84-04
			250	—	—	ns	RC osc	PIC16C84-04
			500	—	10,000	ns	XT osc	PIC16LC84-04
			250	—	10,000	ns	XT osc	PIC16C84-04
			100	—	1,000	ns	HS osc	PIC16C84-10
			5	—	—	µs	LP osc	PIC16LC84-04
2	TCY	**Instruction Cycle Time(1)**	0.4	4/Fosc	DC	µs		
3	TosL, TosH	**Clock in (OSC1) High or Low Time**	60 *	—	—	ns	XT osc	PIC16LC84-04
			50 *	—	—	ns	XT osc	PIC16C84-04
			2 *	—	—	µs	LP osc	PIC16LC84-04
			35 *	—	—	ns	HS osc	PIC16C84-10
4	TosR, TosF	**Clock in (OSC1) Rise or Fall Time**	25 *	—	—	ns	XT osc	PIC16C84-04
			50 *	—	—	ns	LP osc	PIC16LC84-04
			15 *	—	—	ns	HS osc	PIC16C84-10

* 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: Instruction cycle period (TCY) equals four times the input oscillator time base 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.

FIGURE 11-4: CLKOUT AND I/O TIMING



Note: All tests must be done with specified capacitive loads (Figure 11-2) 50 pF on I/O pins and CLKOUT.

TABLE 11-4: CLKOUT AND I/O TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
10	TosH2ckL	OSC1↑ to CLKOUT↓	PIC16C84	—	15	30 *	ns	Note 1
10A			PIC16LC84	—	15	120 *	ns	Note 1
11	TosH2ckH	OSC1↑ to CLKOUT↑	PIC16C84	—	15	30 *	ns	Note 1
11A			PIC16LC84	—	15	120 *	ns	Note 1
12	TckR	CLKOUT rise time	PIC16C84	—	15	30 *	ns	Note 1
12A			PIC16LC84	—	15	100 *	ns	Note 1
13	TckF	CLKOUT fall time	PIC16C84	—	15	30 *	ns	Note 1
13A			PIC16LC84	—	15	100 *	ns	Note 1
14	TckL2ioV	CLKOUT ↓ to Port out valid		—	—	0.5TCY +20 *	ns	Note 1
15	TioV2ckH	Port in valid before	PIC16C84	0.30TCY + 30 *	—	—	ns	Note 1
		CLKOUT ↑	PIC16LC84	0.30TCY + 80 *	—	—	ns	Note 1
16	TckH2ioI	Port in hold after CLKOUT ↑		0 *	—	—	ns	Note 1
17	TosH2ioV	OSC1↑ (Q1 cycle) to	PIC16C84	—	—	125 *	ns	
		Port out valid	PIC16LC84	—	—	250 *	ns	
18	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)		TBD	—	—	ns	
19	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)		TBD	—	—	ns	
20	TioR	Port output rise time	PIC16C84	—	10	25 *	ns	
20A			PIC16LC84	—	10	60 *	ns	
21	TioF	Port output fall time	PIC16C84	—	10	25 *	ns	
21A			PIC16LC84	—	10	60 *	ns	
22	Tinp	INT pin high	PIC16C84	20 *	—	—	ns	
22A		or low time	PIC16LC84	55 *	—	—	ns	
23	Trbp	RB7:RB4 change INT	PIC16C84	20 *	—	—	ns	
23A		high or low time	PIC16LC84	55 *	—	—	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.

Note 1: Measurements are taken in RC Mode where CLKOUT output is 4 x TOSC.

FIGURE 11-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

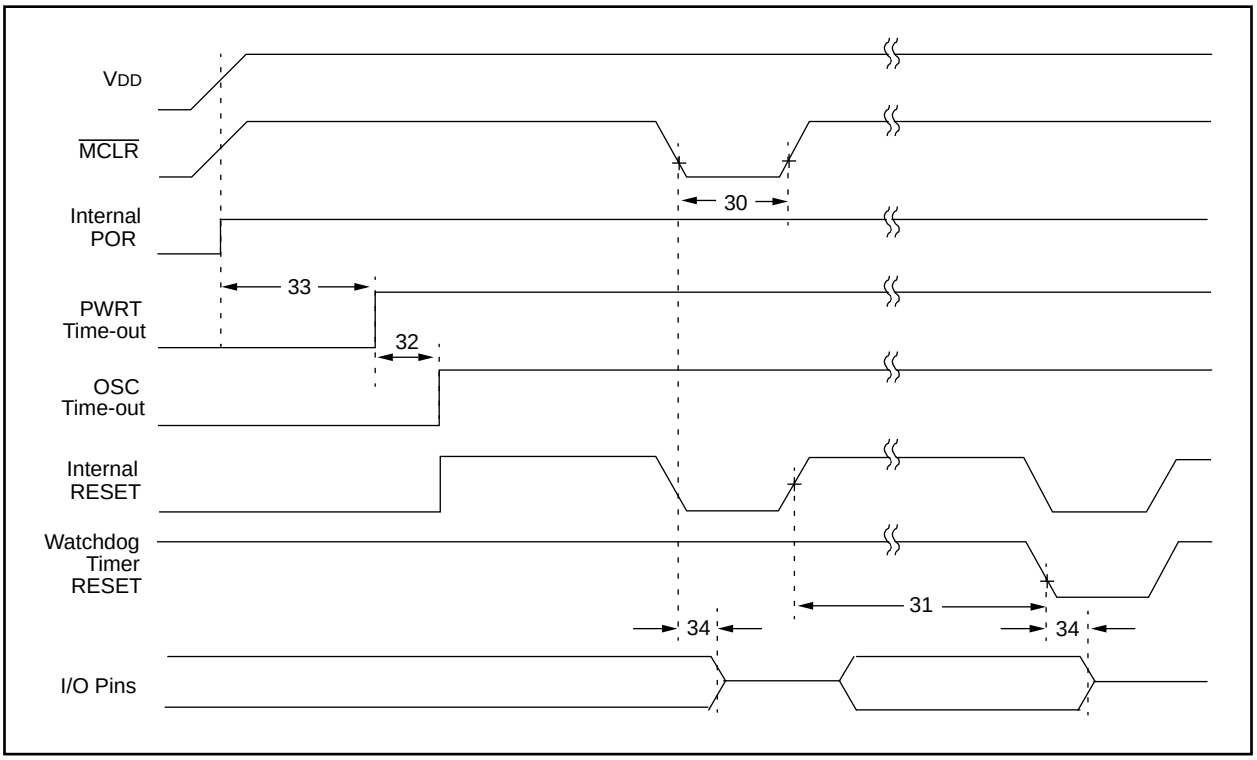


TABLE 11-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{MCLR}$ Pulse Width (low)	350 * 150 *	— —	— —	ns ns	2.0V ≤ VDD ≤ 3.0V 3.0V ≤ VDD ≤ 6.0V
31	Twdt	Watchdog Timer Time-out Period (No Prescaler)	7 *	18	33 *	ms	VDD = 5V
32	Tost	Oscillation Start-up Timer Period	—	1024TOSC	—	ms	TOSC = OSC1 period
33	Tpwrt	Power-up Timer Period	28 *	72	132 *	ms	VDD = 5.0V
34	TIOZ	I/O Hi-impedance from $\overline{MCLR}$ Low or reset	—	—	100 *	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 11-6: TIMER0 CLOCK TIMINGS

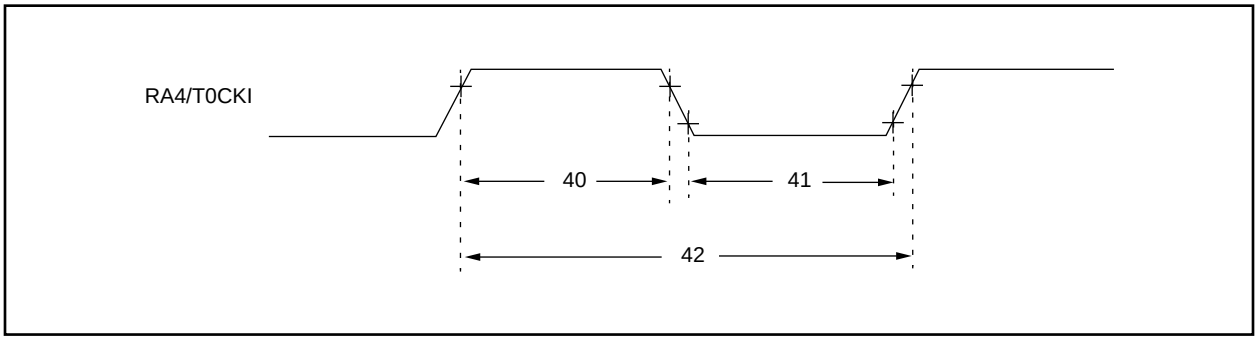


TABLE 11-6: TIMER0 CLOCK REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	0.5TCY + 20 *	—	—	ns	
			With Prescaler	50 *	—	—	ns	2.0V ≤ VDD ≤ 3.0V
				30 *	—	—	ns	3.0V ≤ VDD ≤ 6.0V
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	0.5TCY + 20 *	—	—	ns	
			With Prescaler	50 *	—	—	ns	2.0V ≤ VDD ≤ 3.0V
				20 *	—	—	ns	3.0V ≤ VDD ≤ 6.0V
42	Tt0P	T0CKI Period		$\frac{T_{CY} + 40}{N}$ *	—	—	ns	N = prescale value (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.

12.0 DC & AC CHARACTERISTICS GRAPHS/TABLES FOR PIC16C84

The data graphs and tables provided in this section are for design guidance and are not tested or guaranteed. In some graphs or tables the data presented is outside specified operating range (e.g., outside specified VDD range). This is for information only and devices are guaranteed to operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. “Typical” represents the mean of the distribution while 'max' or 'Min' represents (mean + 3σ) and (mean - 3σ) respectively where σ is standard deviation.

FIGURE 12-1: TYPICAL RC OSCILLATOR FREQUENCY vs. TEMPERATURE

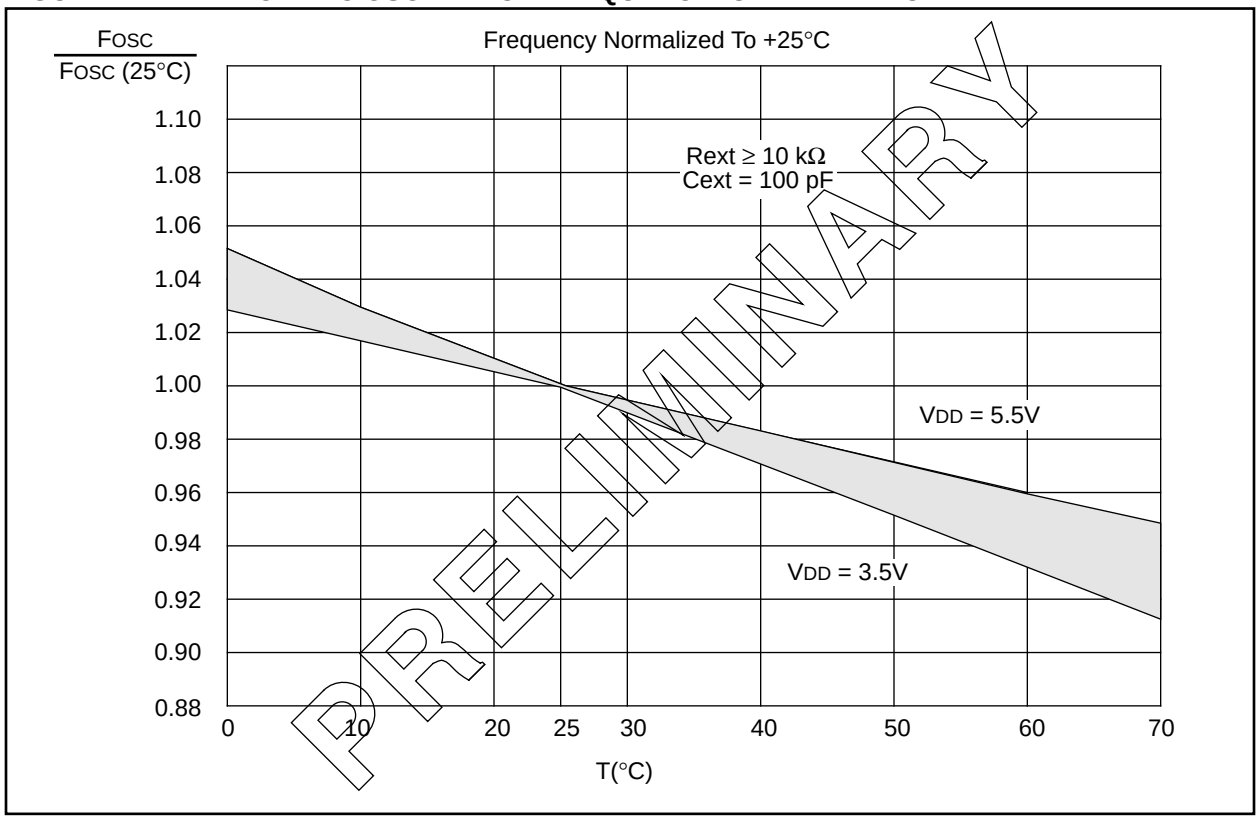


TABLE 12-1: RC OSCILLATOR FREQUENCIES *

Cext	Rext	Average Fosc @ 5V, 25°C	
20 pF	3.3k	4.68 MHz	± 27%
	5.1k	3.94 MHz	± 25%
	10k	2.34 MHz	± 29%
	100k	250.16 kHz	± 33%
100 pF	3.3k	1.49 MHz	± 25%
	5.1k	1.12 MHz	± 25%
	10k	620.31 kHz	± 30%
	100k	90.25 kHz	± 26%
300 pF	3.3k	524.24 kHz	± 28%
	5.1k	415.52 kHz	± 30%
	10k	270.33 kHz	± 26%
	100k	25.37 kHz	± 25%

*Measured in PDIP Packages.The percentage variation indicated here is part to part variation due to normal process distribution. The variation indicated is ±3 standard deviation from average value.

FIGURE 12-2: TYPICAL RC OSCILLATOR FREQUENCY vs. V_{DD} (Cext = 20 pF)

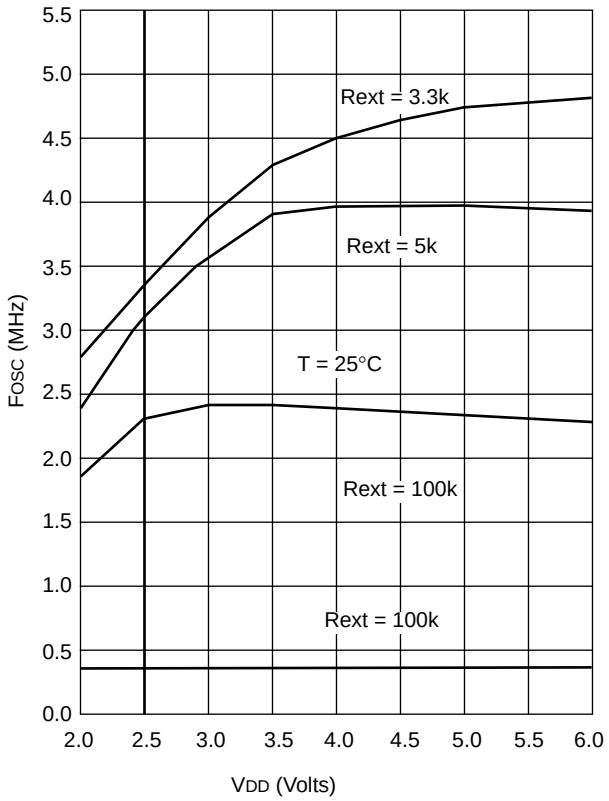


FIGURE 12-3: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD (Cext = 100 pF)

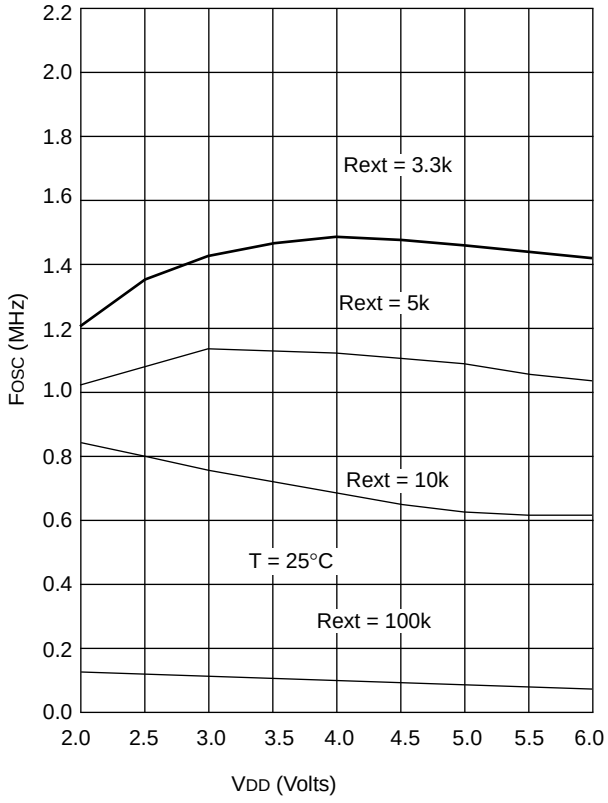


FIGURE 12-4: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD (Cext = 300 pF)

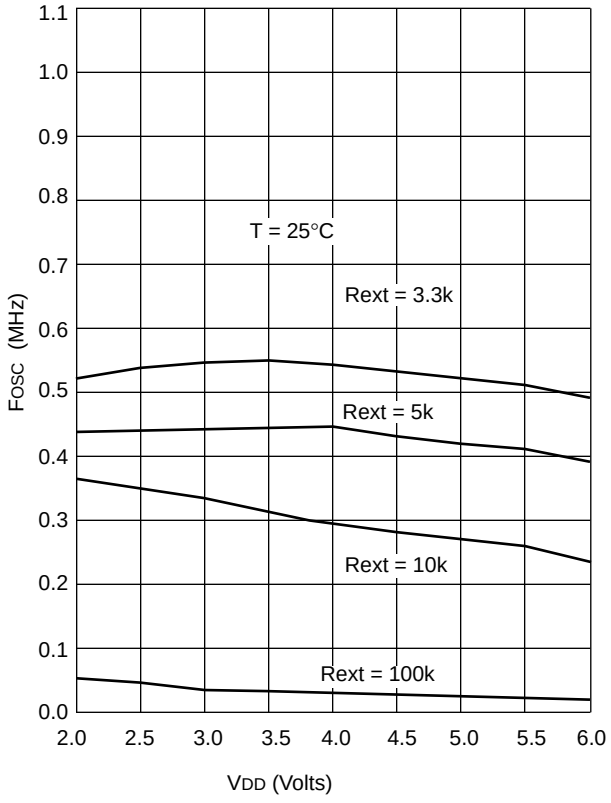


FIGURE 12-5: TYPICAL IPD VS. VDD WATCHDOG DISABLED (25˚C)

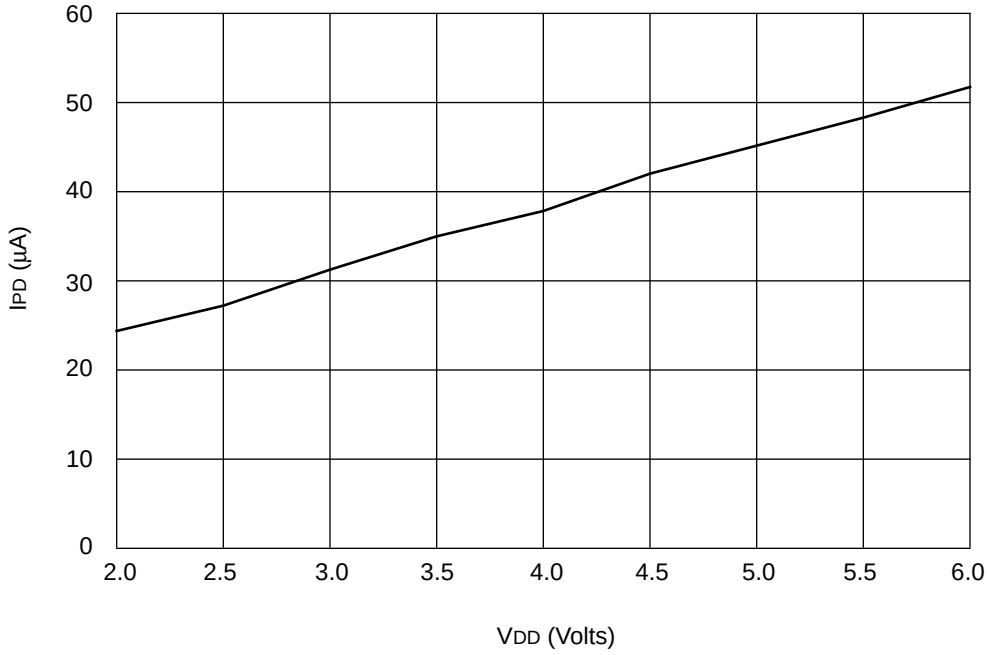


FIGURE 12-6: TYPICAL IPD VS. VDD WATCHDOG ENABLED (25˚C)

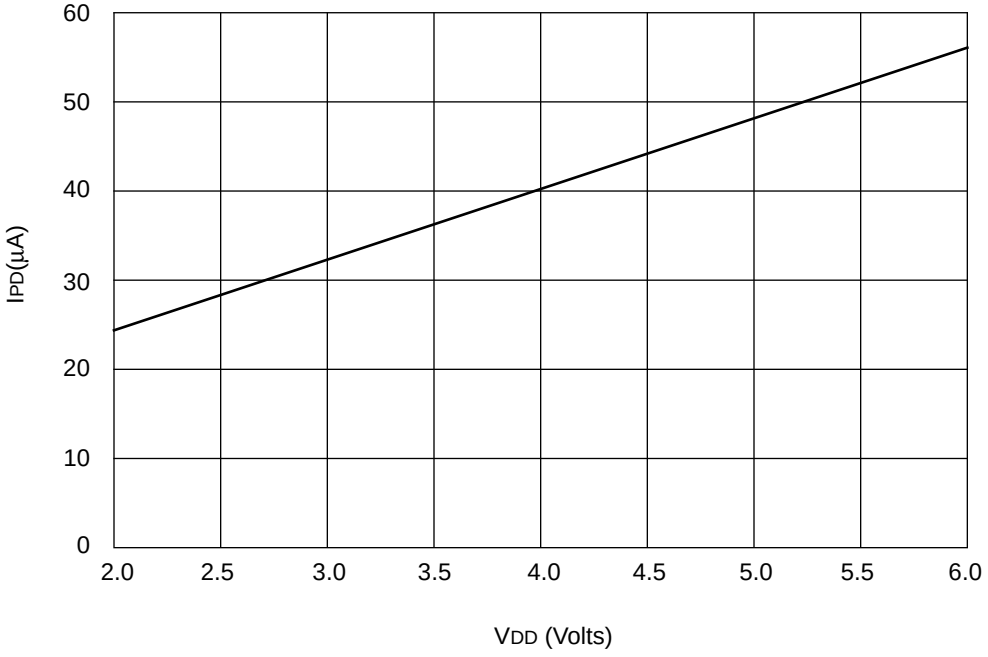


FIGURE 12-7: MAXIMUM IPD vs. VDD WATCHDOG DISABLED

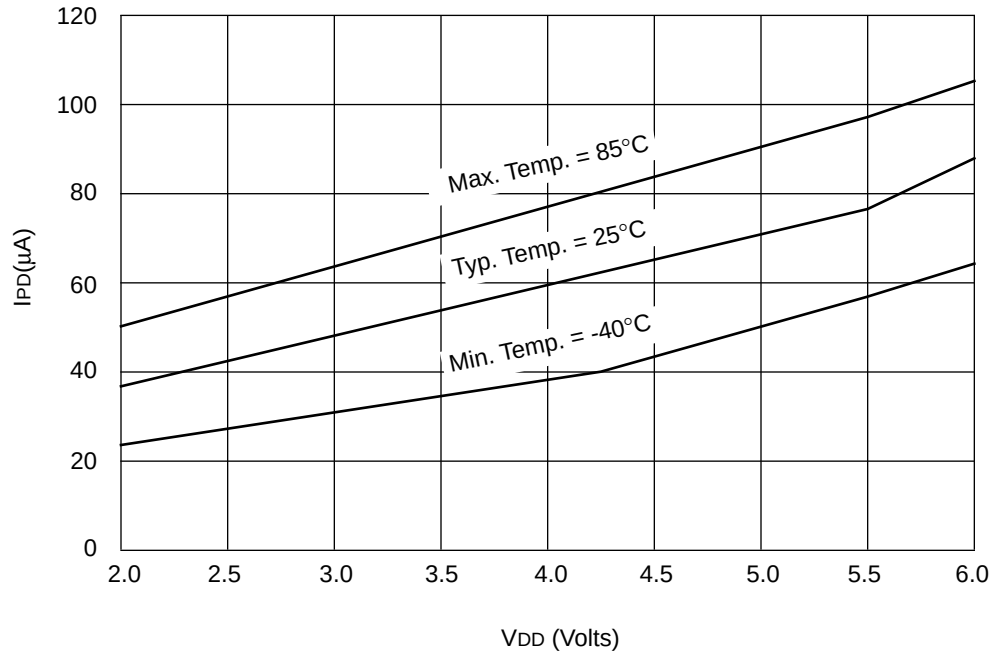
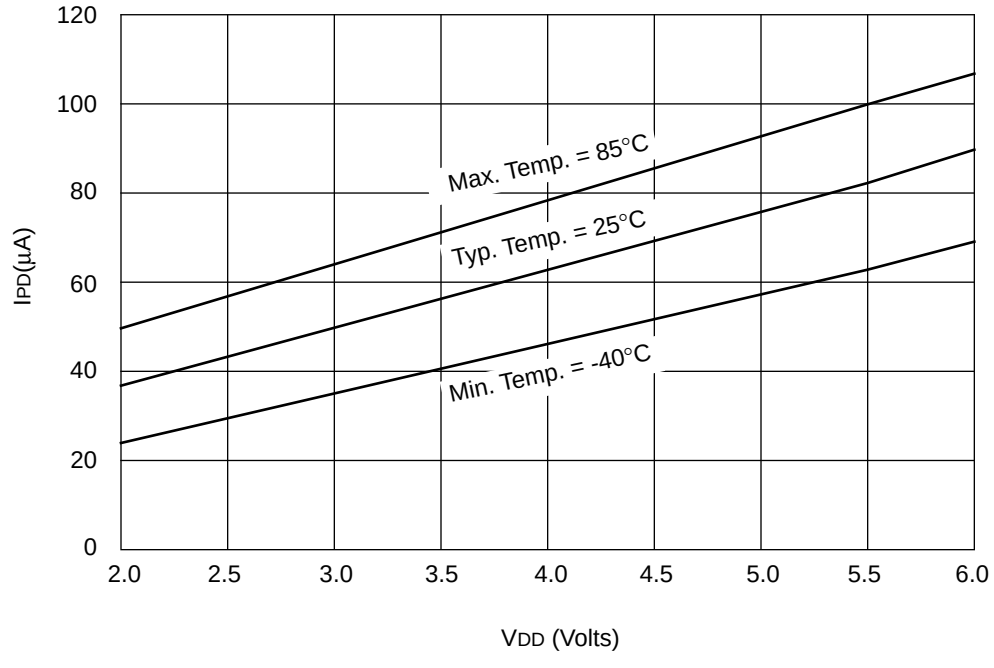


FIGURE 12-8: MAXIMUM IPD vs. VDD WATCHDOG ENABLED*



* IPD, with Watchdog Timer enabled, has two components: The leakage current which increases with higher temperature and the operating current of the Watchdog Timer logic which increases with lower temperature. At -40°C, the latter dominates explaining the apparently anomalous behavior.

FIGURE 12-9: VTH (INPUT THRESHOLD VOLTAGE) OF I/O PINS vs. VDD

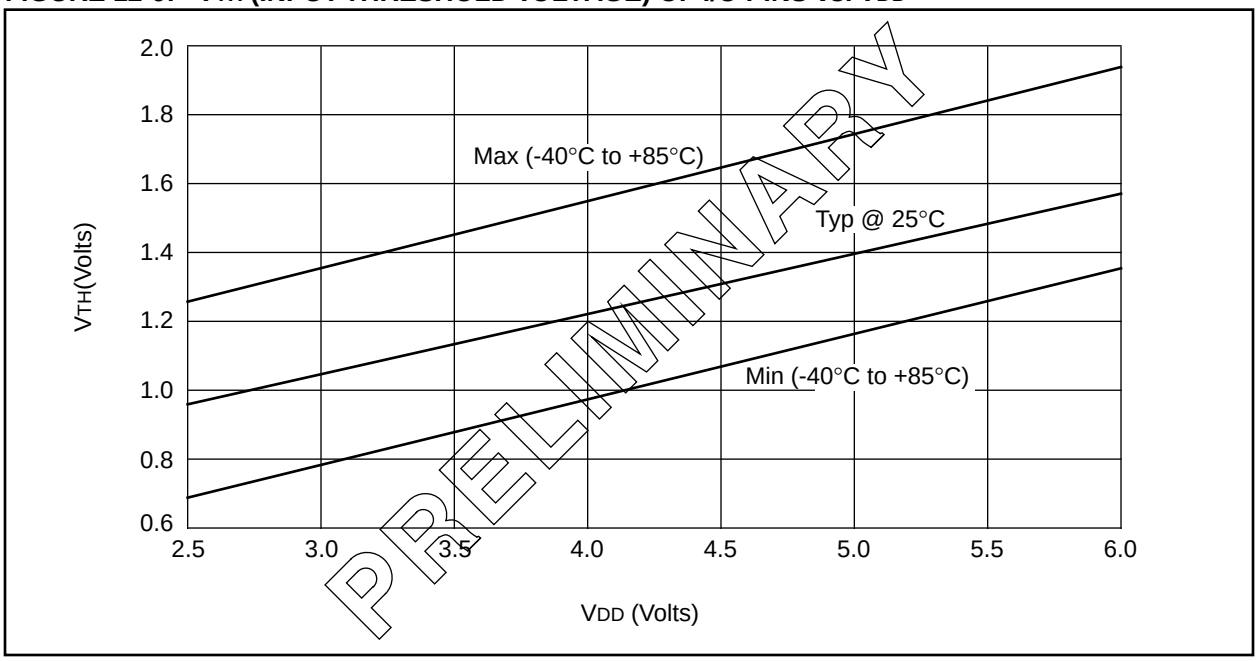


FIGURE 12-10: VTH (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN XT, HS, AND LP MODES) vs. VDD

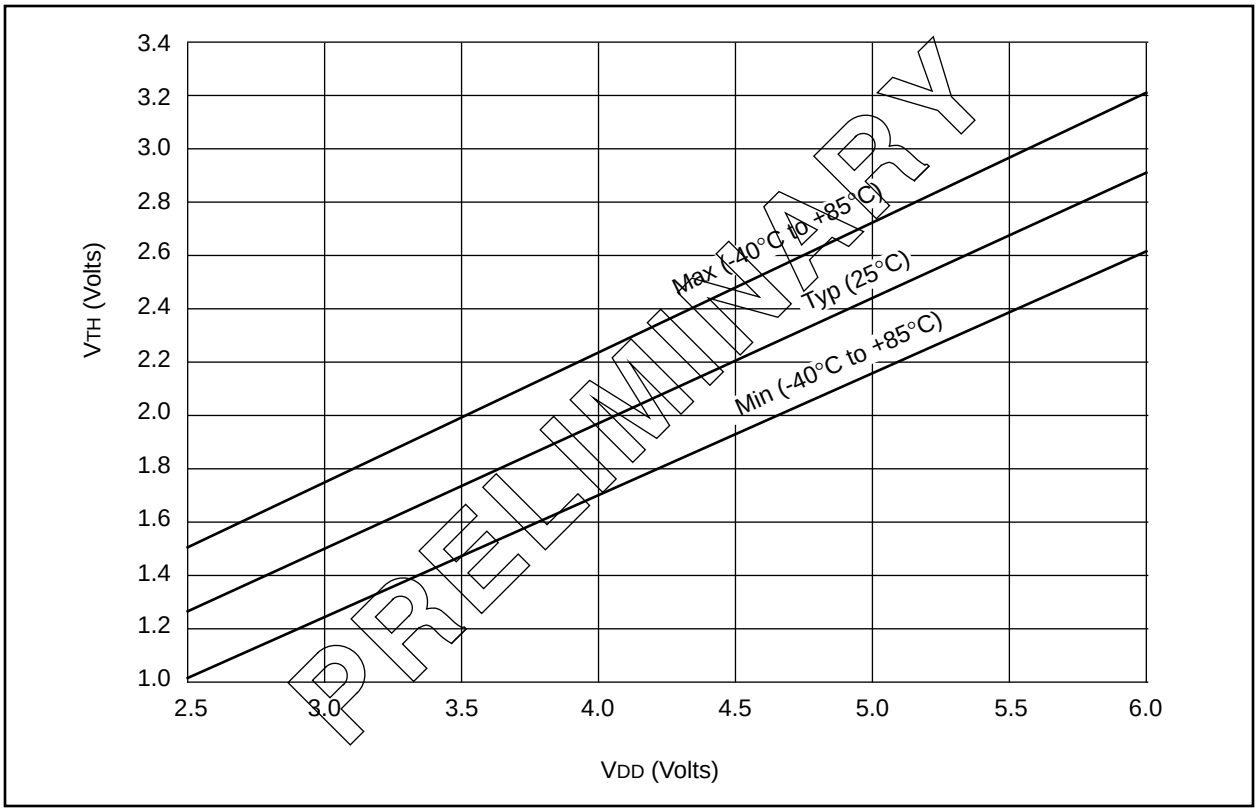


FIGURE 12-11: VIH, VIL OF $\overline{\text{MCLR}}$, T0CKI and OSC1 (IN RC MODE) vs. VDD

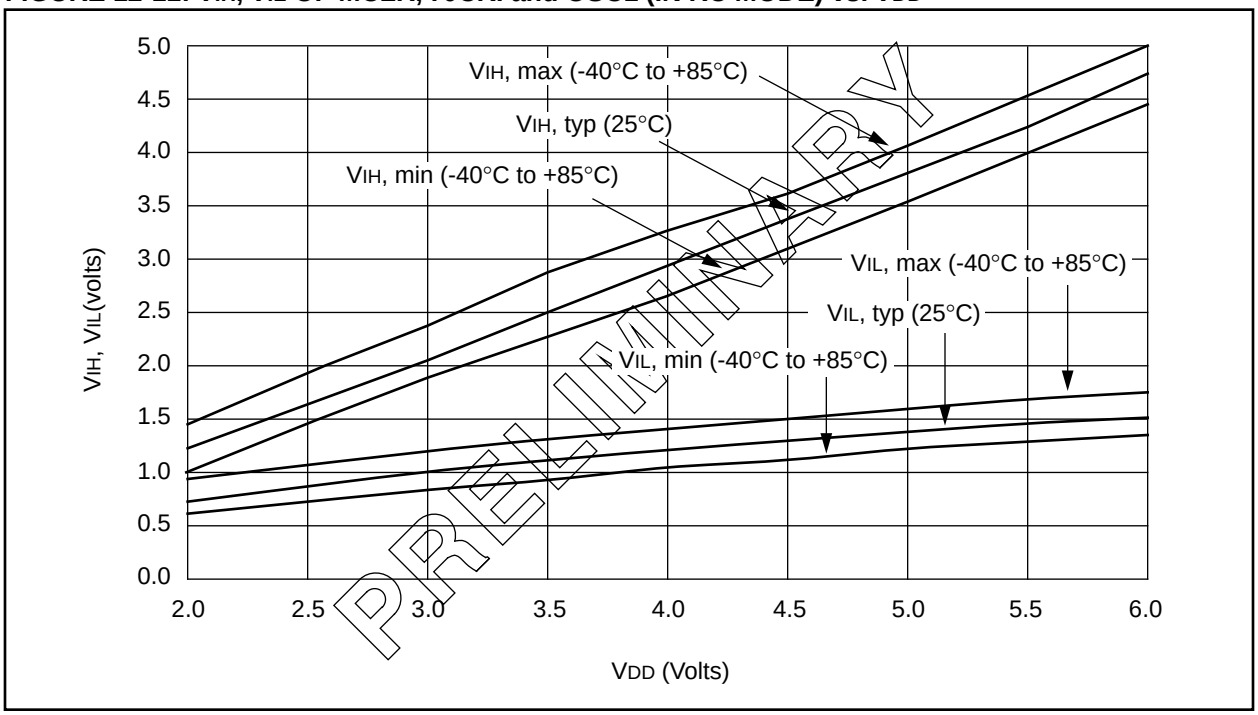


FIGURE 12-12: TYPICAL IDD vs. FREQ (EXT CLOCK, 25˚C)

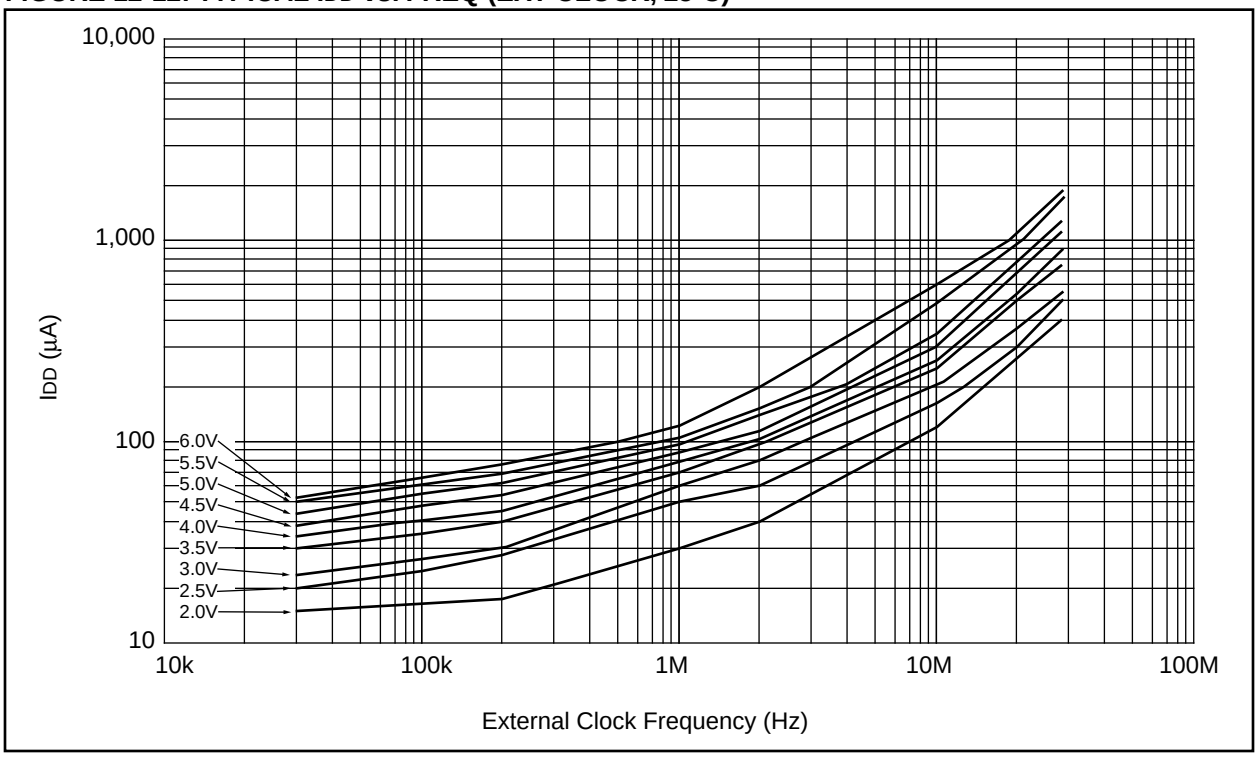


FIGURE 12-13: MAXIMUM IDD vs. FREQ (EXT CLOCK, -40˚ TO +85˚C)

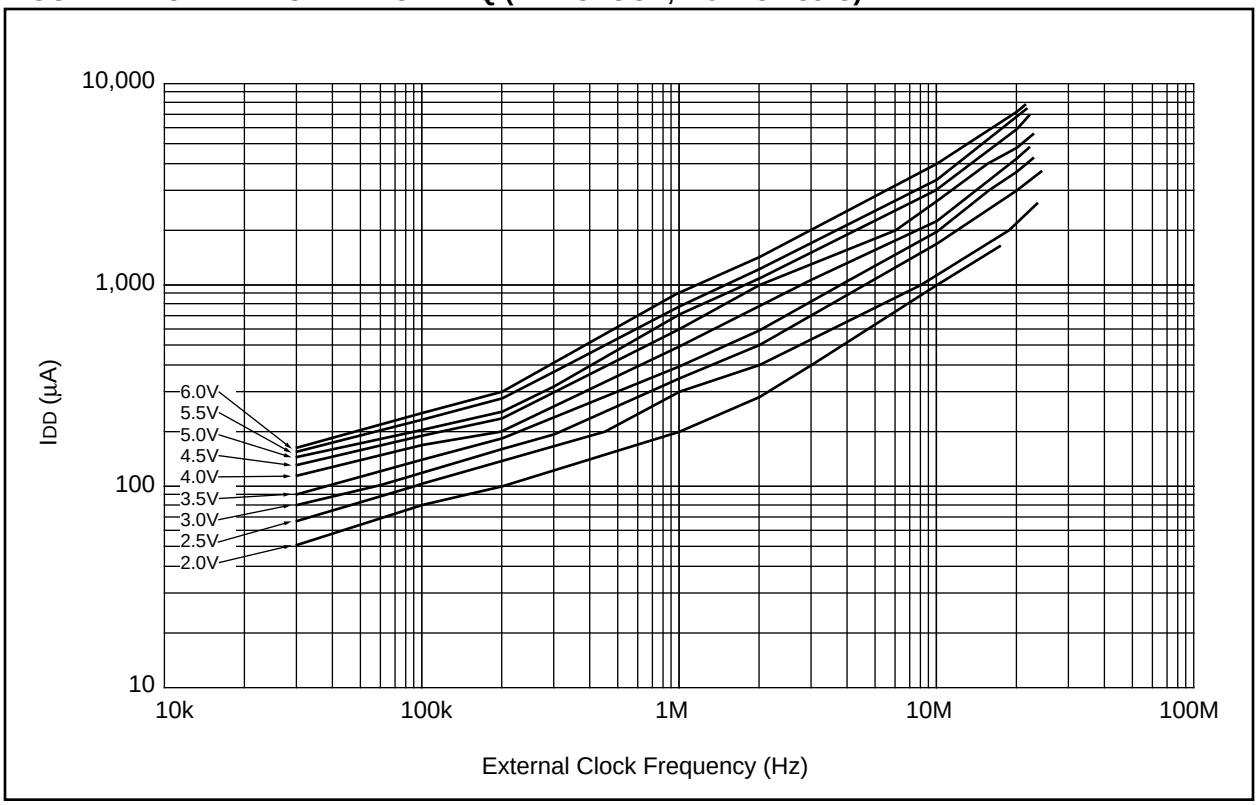


FIGURE 12-14: WDT TIME-OUT PERIOD vs. VDD

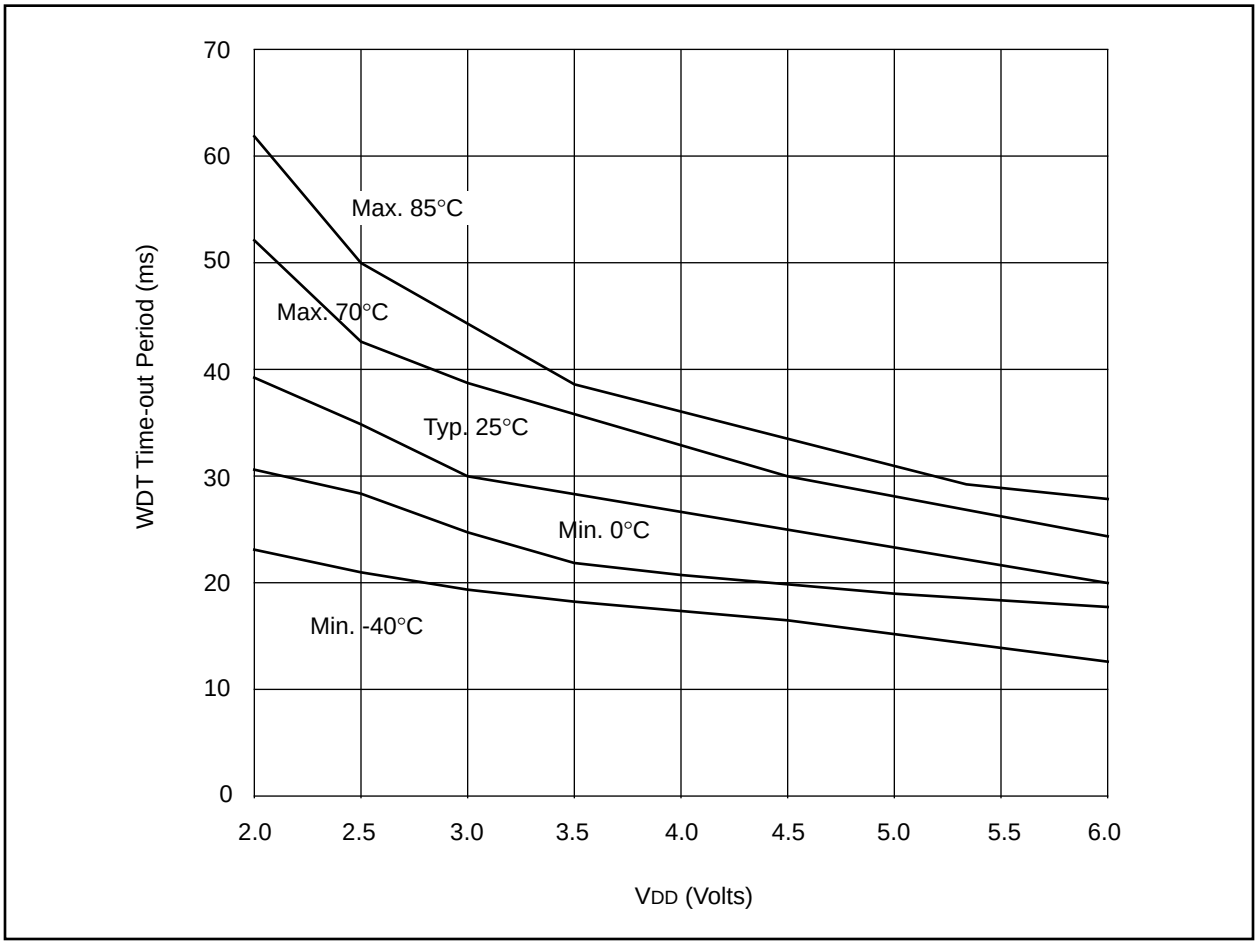


FIGURE 12-15: TRANSCONDUCTANCE (gm) OF HS OSCILLATOR vs. VDD

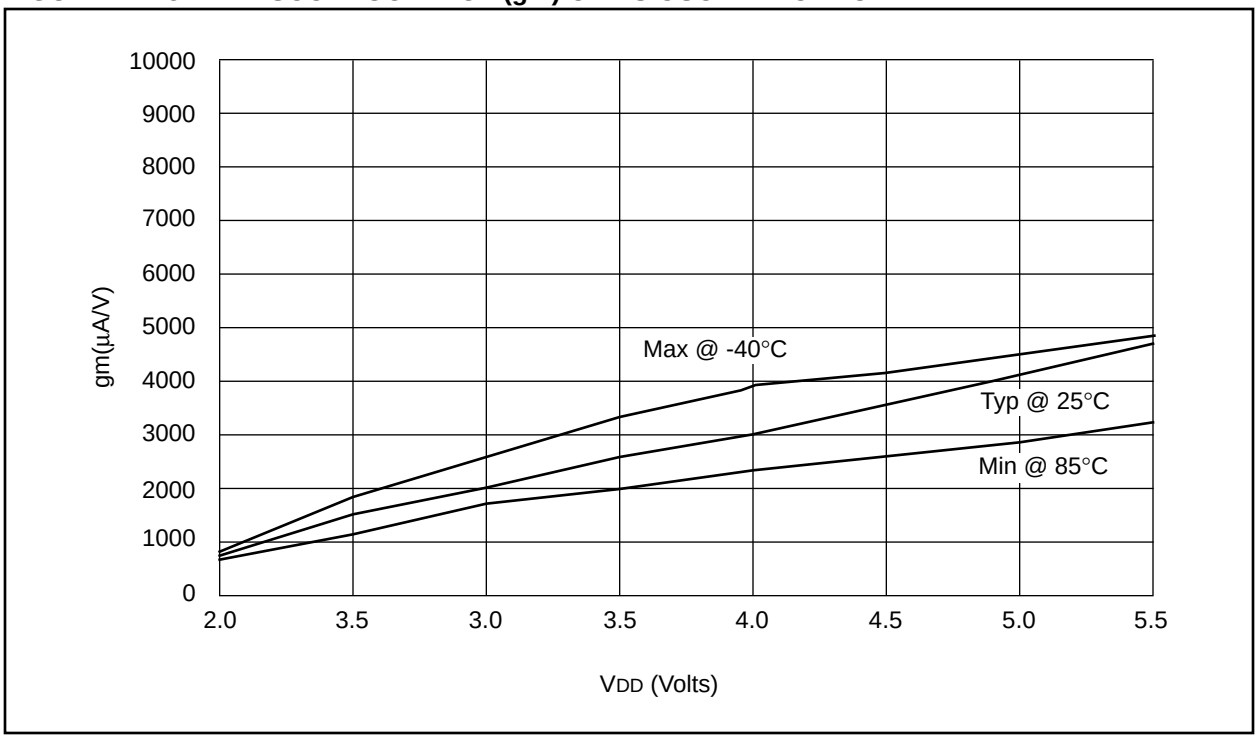


FIGURE 12-16: TRANSCONDUCTANCE (gm) OF LP OSCILLATOR vs. VDD

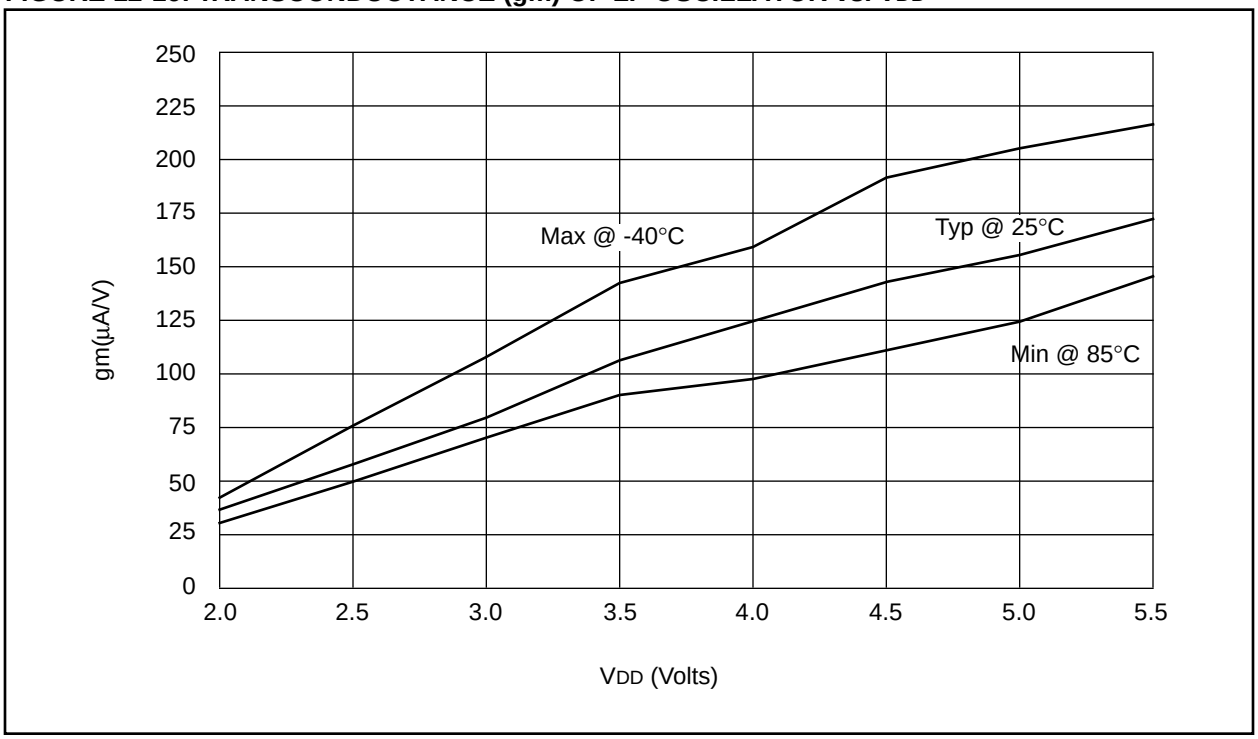


FIGURE 12-17: TRANSCONDUCTANCE (gm) OF XT OSCILLATOR vs. VDD

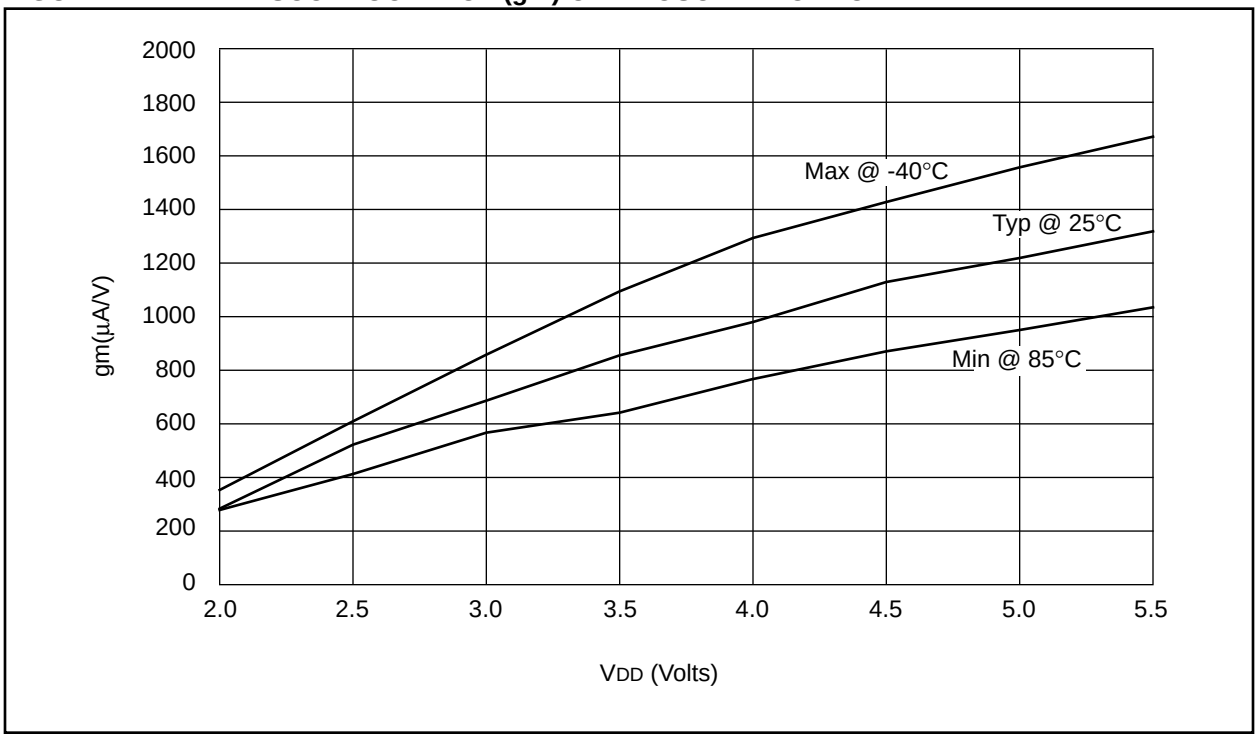


FIGURE 12-18: IOH vs. VOH, VDD = 3V

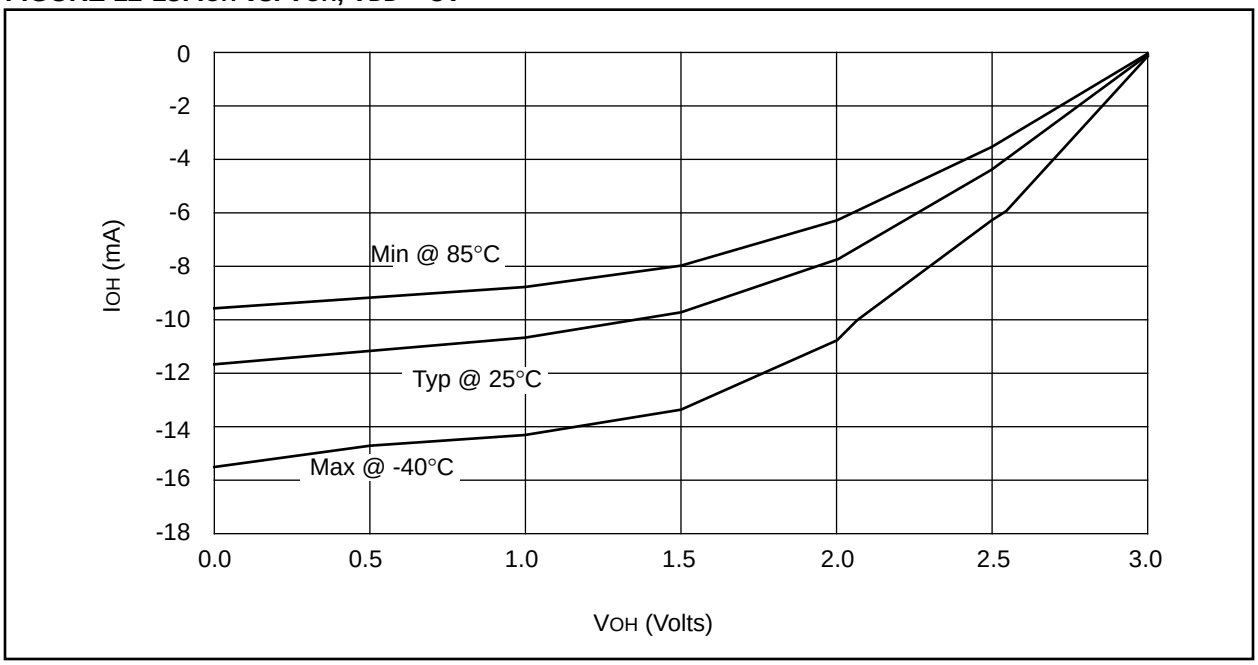


FIGURE 12-19: IOH vs. VOH, VDD = 5V

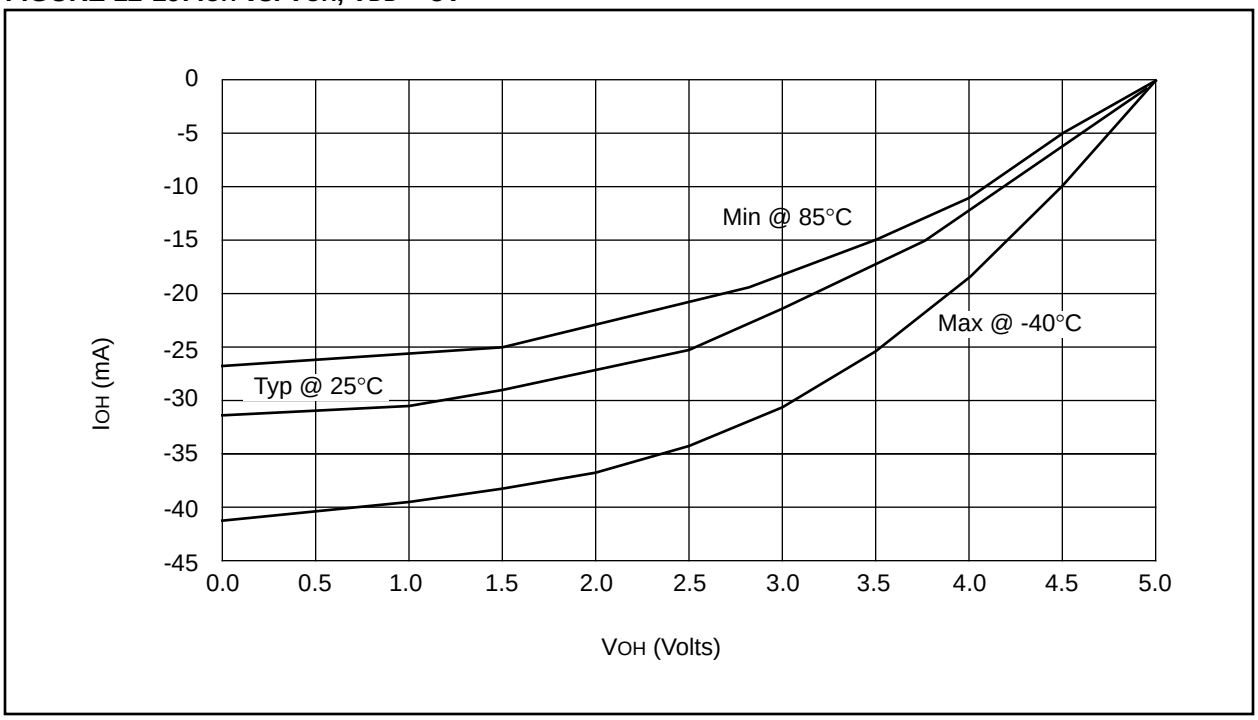


FIGURE 12-20: IOL vs. VOL, VDD = 3V

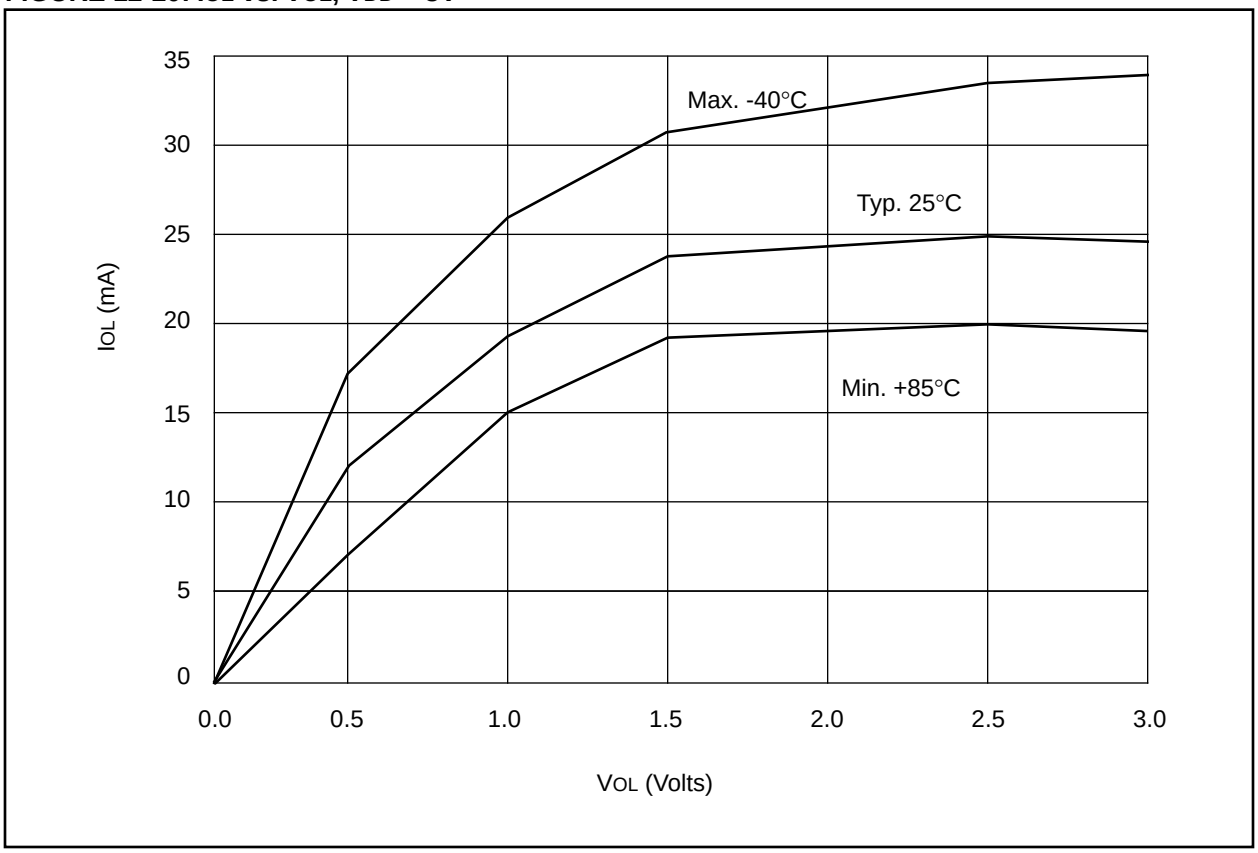


FIGURE 12-21: IOL vs. VOL, VDD = 5V

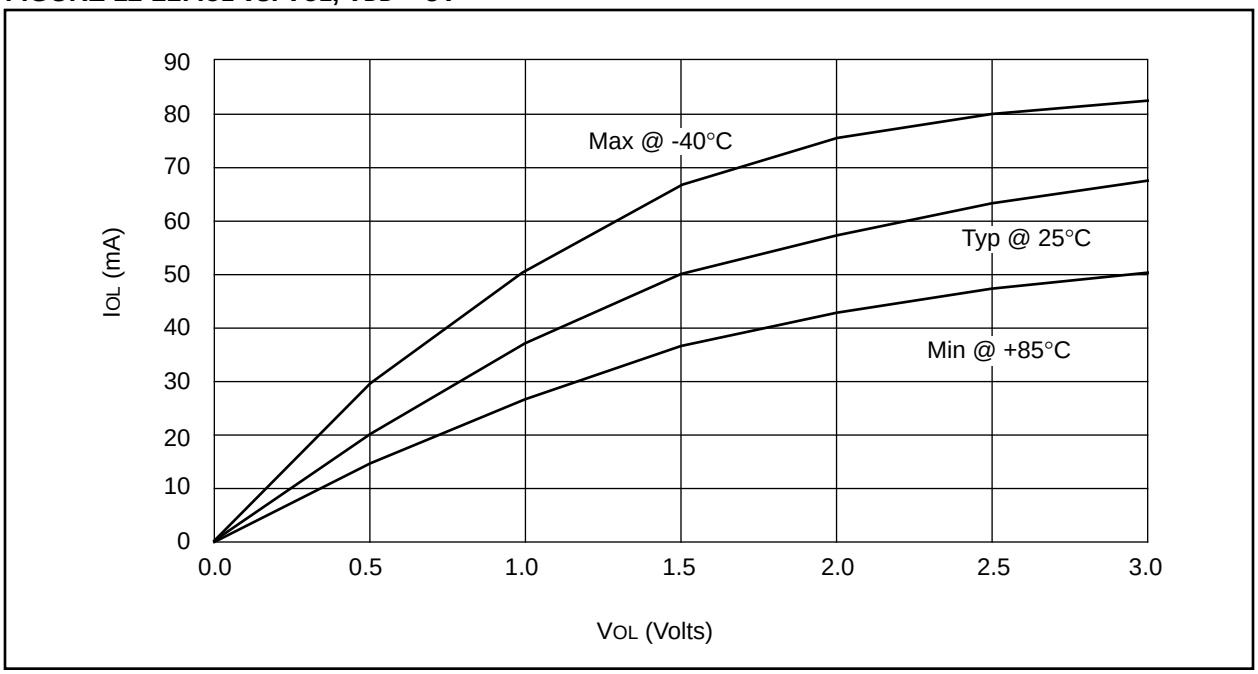


TABLE 12-2: INPUT CAPACITANCE *

Pin Name	Typical Capacitance (pF)	
	18L PDIP	18L SOIC
PORTA	5.0	4.3
PORTB	5.0	4.3
$\overline{MCLR}$	17.0	17.0
OSC1/CLKIN	4.0	3.5
OSC2/CLKOUT	4.3	3.5
T0CKI	3.2	2.8

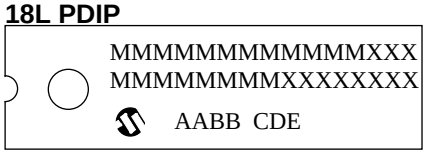
* All capacitance values are typical at 25°C. A part to part variation of ±25% (three standard deviations) should be taken into account.

NOTES:

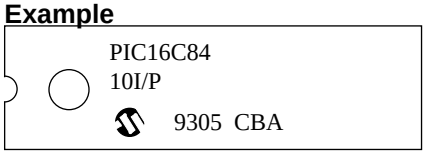
13.0 PACKAGING INFORMATION

13.1 Package Marking Information

18L PDIP

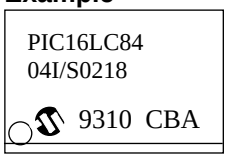


Example



18L SOIC

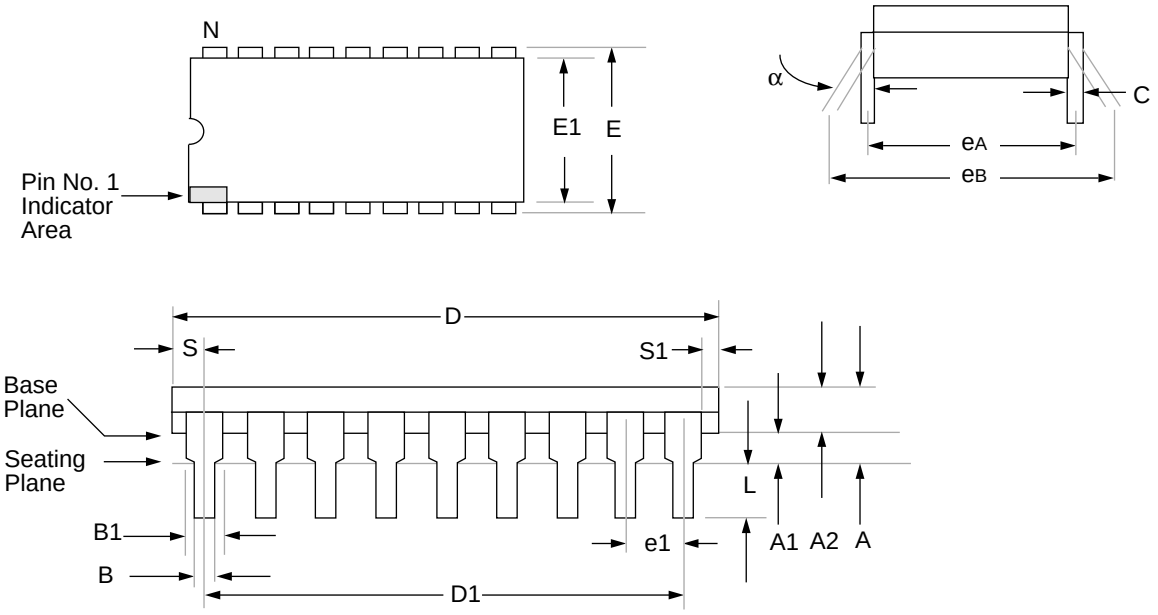
Example



Legend:	MM...M	Microchip part number information
	XX...X	Customer specific information*
	AA	Year code (last two digits of calendar year)
	BB	Week code (week of January 1 is week '01')
	C	Facility code of the plant at which wafer is manufactured C = Chandler, Arizona, U.S.A., S = Tempe, Arizona, U.S.A.
	D	Mask revision number
	E	Assembly code of the plant or country of origin in which part was assembled
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.	

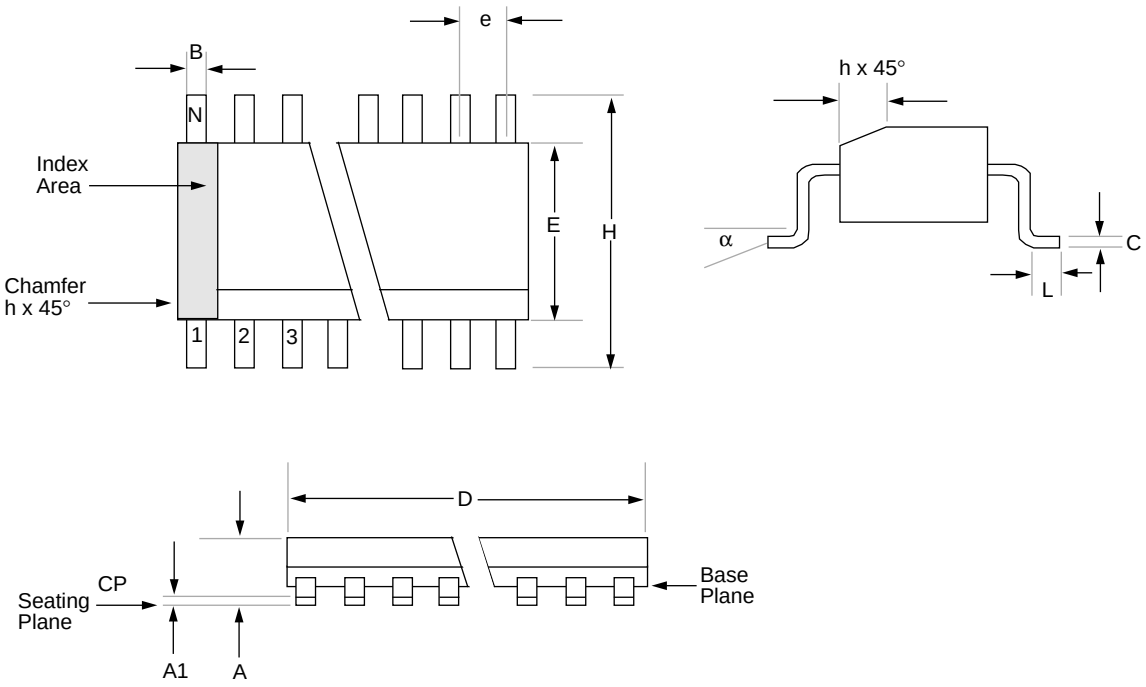
***** Standard OTP marking consists of Microchip part number, year code, week code, facility code, mask rev#, and assembly code. For OTP marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

13.2 18-Lead Plastic Dual In-line (PDIP) - 300 mil



Package Group: Plastic Dual In-Line (PLA)						
	Millimeters			Inches		
Symbol	Min	Max	Notes	Min	Max	Notes
α	0°	10°		0°	10°	
A	–	4.064		–	0.160	
A1	0.381	–		0.015	–	
A2	3.048	3.810		0.120	0.150	
B	0.355	0.559		0.014	0.022	
B1	1.524	1.524	**Reference**	0.060	0.060	**Reference**
C	0.203	0.381	**Typical**	0.008	0.015	**Typical**
D	22.479	23.495		0.885	0.925	
D1	20.320	20.320	**Reference**	0.800	0.800	**Reference**
E	7.620	8.255		0.300	0.325	
E1	6.096	7.112		0.240	0.280	
e1	2.489	2.591	**Typical**	0.098	0.102	**Typical**
eA	7.620	7.620	**Reference**	0.300	0.300	**Reference**
eB	7.874	9.906		0.310	0.390	
L	3.048	3.556		0.120	0.140	
N	18	18		18	18	
S	0.889	–		0.035	–	
S1	0.127	–		0.005	–	

13.3 18-Lead Plastic Surface Mount (SOIC) - 300 mil



Package Group: Plastic SOIC (SO)						
	Millimeters			Inches		
Symbol	Min	Max	Notes	Min	Max	Notes
α	0°	8°		0°	8°	
A	2.362	2.642		0.093	0.104	
A1	0.101	0.300		0.004	0.012	
B	0.355	0.483		0.014	0.019	
C	0.241	0.318		0.009	0.013	
D	11.353	11.735		0.447	0.462	
E	7.416	7.595		0.292	0.299	
e	1.270	1.270	**Reference**	0.050	0.050	**Reference**
H	10.007	10.643		0.394	0.419	
h	0.381	0.762		0.015	0.030	
L	0.406	1.143		0.016	0.045	
N	18	18		18	18	
CP	–	0.102		–	0.004	

NOTES:

APPENDIX A: FEATURE IMPROVEMENTS

The following is the list of feature improvements over the PIC16C5X microcontroller family:

1. Instruction word length is increased to 14 bits. This allows larger page sizes both in program memory (2K now as opposed to 512 before) and the register file (128 bytes now versus 32 bytes before).
2. A PC latch register (PCLATH) is added to handle program memory paging. PA2, PA1 and PA0 bits are removed from the status register and placed in the option register.
3. Data memory paging is redefined slightly. The STATUS register is modified.
4. Four new instructions have been added: RETURN, RETFIE, ADDLW, and SUBLW. Two instructions, TRIS and OPTION, are being phased out although they are kept for compatibility with PIC16C5X.
5. OPTION and TRIS registers are made addressable.
6. Interrupt capability is added. Interrupt vector is at 0004h.
7. Stack size is increased to 8 deep.
8. Reset vector is changed to 0000h.
9. Reset of all registers is revisited. Five different reset (and wake-up) types are recognized. Registers are reset differently.
10. Wake up from SLEEP through interrupt is added.
11. Two separate timers, the Oscillator Start-up Timer (OST) and Power-up Timer (PWRT), are included for more reliable power-up. These timers are invoked selectively to avoid unnecessary delays on power-up and wake-up.
12. PORTB has weak pull-ups and interrupt on change features.
13. T0CKI pin is also a port pin (RA4/T0CKI).
14. FSR is a full 8-bit register.
15. "In system programming" is made possible. The user can program PIC16CXX devices using only five pins: V_{DD}, V_{SS}, V_{PP}, RB6 (clock) and RB7 (data in/out).

APPENDIX B: COMPATIBILITY

To convert code written for PIC16C5X to PIC16C84, the user should take the following steps:

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

APPENDIX C: WHAT'S NEW

No new information has been added to this data sheet.

For information on upgrade devices from the PIC16C84, please refer to the PIC16F8X data sheet.

APPENDIX D: WHAT'S CHANGED

Here's what's changed in this data sheet:

1. Some sections have been rearranged for clarity and consistency.
2. Time-out Sequence on Power-up figures in the Special Features of the CPU section have been updated.

APPENDIX E: PIC16C84 TO PIC16F83/F84/CR83/ CR84 CONVERSION CONSIDERATIONS

This appendix discusses some of the issues that you may encounter as you convert your design from a PIC16C84 to any of the newly introduced devices. These new devices are:

- PIC16F83
- PIC16CR83
- PIC16F84
- PIC16CR84

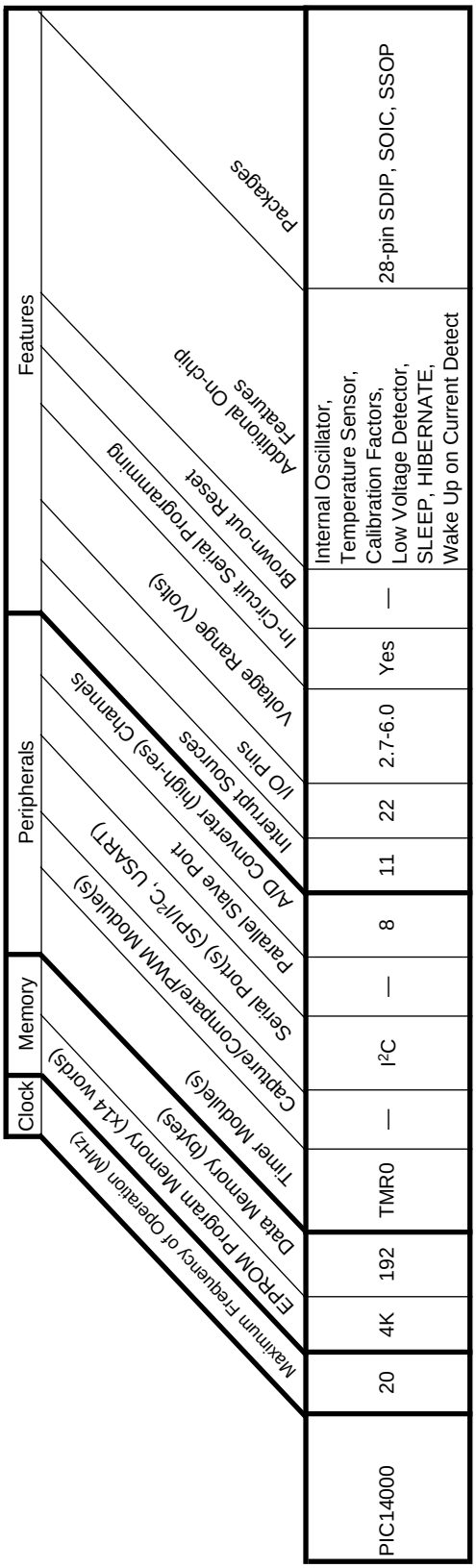
Some of the issues that may be encountered are:

1. The polarity of the PWRTE configuration bit has been reversed. Ensure that the programmer has this bit correctly set before programming.
2. The PIC16F84 and PIC16CR84 have larger RAM sizes. Ensure that this does not cause an issue with your program.
3. The $\overline{\text{MCLR}}$ pin now has an on-chip filter. The input signal on the MCLR pin will require a longer low pulse to generate an interrupt.
4. Many electrical specifications have been improved. Compare the electrical specifications of the two devices to ensure that this will not cause a compatibility issue.

NOTES:

APPENDIX F: PIC16/17 MICROCONTROLLERS

F.1 PIC14000 Devices



F.2 PIC16C5X Family of Devices

	Clock	Memory				Peripherals		Features	
	Maximum Frequency of Operation (MHz)	Program Memory (x12 words)	EPROM	ROM	RAM Data Memory (bytes)	Timer Module(s)	I/O Pins	Voltage Range (Volts)	Number of Instructions	Packages
PIC16C52	4	384	—	25	TMR0	12	2.5-6.25	33	18-pin DIP, SOIC
PIC16C54	20	512	—	25	TMR0	12	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16C54A	20	512	—	25	TMR0	12	2.0-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16CR54A	20	—	512	25	TMR0	12	2.0-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16C55	20	512	—	24	TMR0	20	2.5-6.25	33	28-pin DIP, SOIC, SSOP
PIC16C56	20	1K	—	25	TMR0	12	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16C57	20	2K	—	72	TMR0	20	2.5-6.25	33	28-pin DIP, SOIC, SSOP
PIC16CR57B	20	—	2K	72	TMR0	20	2.5-6.25	33	28-pin DIP, SOIC, SSOP
PIC16C58A	20	2K	—	73	TMR0	12	2.0-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16CR58A	20	—	2K	73	TMR0	12	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP

All PIC16/17 Family devices have Power-On Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.

F.3 PIC16CXXX Family of Devices

	Clock	Memory			Peripherals			Features		
	Maximum Frequency of Operation (MHz)	EPROM	Program Memory (x14 words)	Data Memory (bytes)	Timer Module(s)	Comparator(s)	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Brown-out Reset	Packages
PIC16C554	20	512	80	TMR0	—	—	3	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C556	20	1K	80	TMR0	—	—	3	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C558	20	2K	128	TMR0	—	—	3	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C620	20	512	80	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C621	20	1K	80	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C622	20	2K	128	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.

All PIC16C62X Family devices use serial programming with clock pin RB6 and data pin RB7.

F.4 PIC16C6X Family of Devices

	Clock	Memory				Peripherals					Features			
	Maximum Frequency of Operation (MHz)	EPROM	ROM	Data Memory (bytes)	Timer Module(s)	Capture/Compare/PWM Module(s)	Serial Port(s) (SPI/I²C, USART)	Parallel Slave Port	Interrupt Sources	I/O Pins	Voltage Range (Volts)	In-Circuit Serial Programming	Brown-out Reset	Packages
PIC16C62	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	2.5-6.0	Yes	—	28-pin SDIP, SOIC, SSOP
PIC16C62A(1)	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP
PIC16CR62(1)	20	—	2K	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP
PIC16C63	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	10	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC
PIC16CR63(1)	20	—	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	10	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC
PIC16C64	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	2.5-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP
PIC16C64A(1)	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16CR64(1)	20	—	2K	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16C65	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	2.5-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP
PIC16C65A(1)	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16CR65(1)	20	—	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP

All PIC16/17 family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect, and high I/O current capability.
All PIC16C6X family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local sales office for availability of these devices.

F.5 PIC16C7X Family of Devices

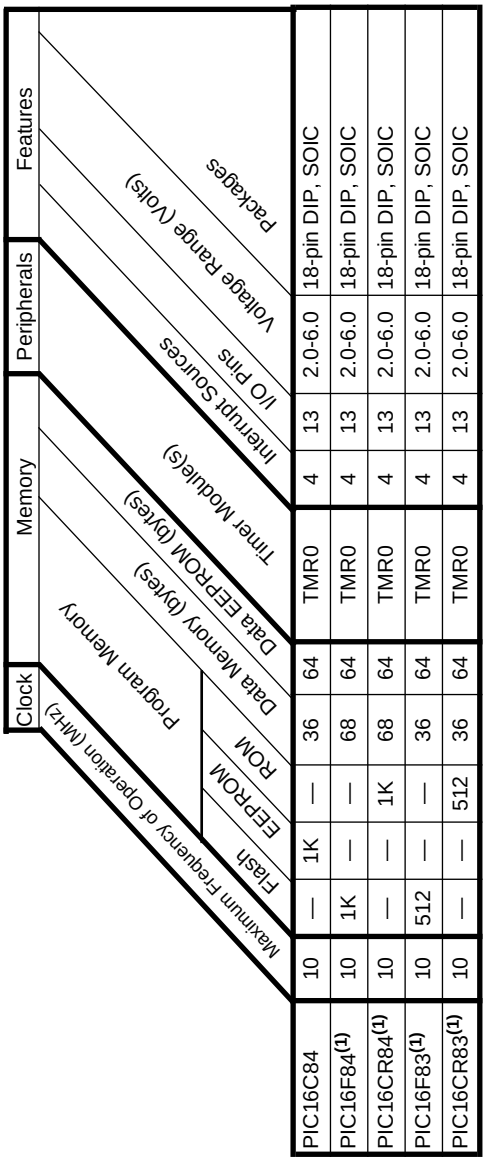
	Clock	Memory			Peripherals					Features			
	Maximum Frequency of Operation (MHz)	EPROM Program Memory (x14 words)	Data Memory (bytes)	Timer Module(s)	Capture/Compare/PWM Module(s)	Serial Port(s) (SPI/I²C, USART)	Parallel Slave Port	A/D Converter (8-bit) Channels	Interrupt Sources	I/O Pins	Voltage Range (Volts)	In-Circuit Serial Programming	Brown-out Reset	Packages
PIC16C710	20	512	36	TMR0	—	—	—	4	4	13	2.5-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C71	20	1K	36	TMR0	—	—	—	4	4	13	2.5-6.0	Yes	—	18-pin DIP, SOIC
PIC16C711	20	1K	68	TMR0	—	—	—	4	4	13	2.5-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C72	20	2K	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	5	8	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP
PIC16C73	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	5	11	22	2.5-6.0	Yes	—	28-pin SDIP, SOIC
PIC16C73A(1)	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	5	11	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC
PIC16C74	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	8	12	33	2.5-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP
PIC16C74A(1)	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	8	12	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.

All PIC16C7X Family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local sales office for availability of these devices.

F.6 PIC16C8X Family of Devices



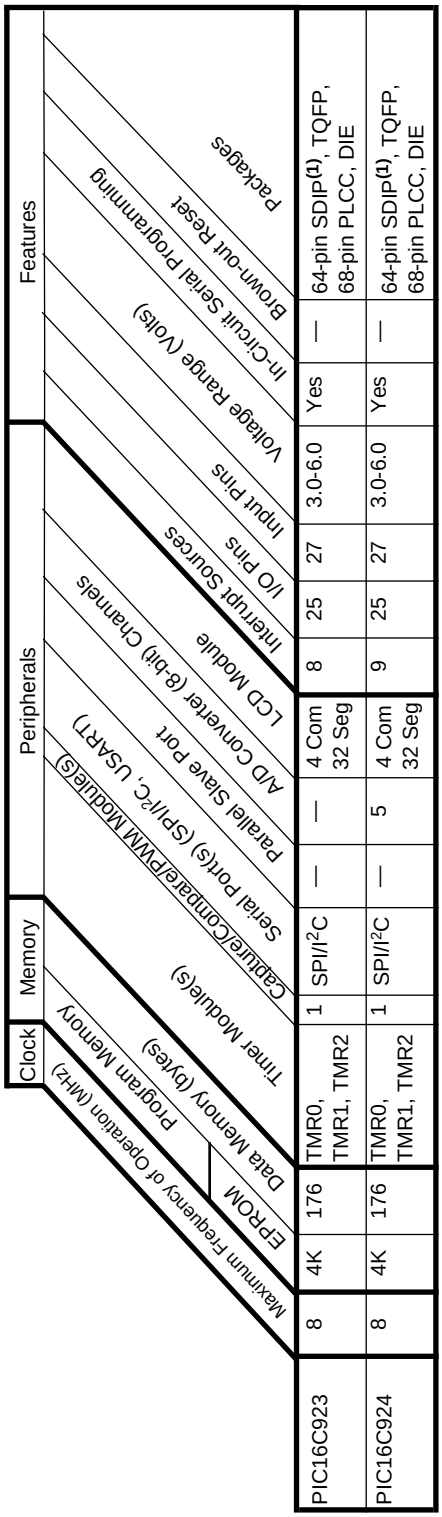
	Clock	Memory				Peripherals		Features	
	Maximum Frequency of Operation (MHz)	Program Memory	Flash	EEPROM	ROM	Data Memory (bytes)	Data EEPROM (bytes)	Timer Module(s)	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Packages
PIC16C84	10	—	1K	—	36	64	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16F84(1)	10	1K	—	—	68	64	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16CR84(1)	10	—	—	1K	68	64	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16F83(1)	10	512	—	—	36	64	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC
PIC16CR83(1)	10	—	—	512	36	64	64	TMR0	4	13	2.0-6.0	18-pin DIP, SOIC

All PIC16/17 family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect, and high I/O current capability.

All PIC16C8X family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local sales office for availability of these devices.

F.7 PIC16C9XX Family Of Devices



	Clock	Memory			Peripherals						Features			
	Maximum Frequency of Operation (MHz)	EPROM Program Memory (bytes)	Data Memory (bytes)	Timer Module(s)	Capture/Compare/PWM Module(s)	Serial Port(s) (SPI/I²C, USART)	Parallel Slave Port	A/D Converter (8-bit) Channels	LCD Module	Interrupt Sources	I/O Pins	Input Pins	Voltage Range (Volts)	In-Circuit Serial Programming	Brown-out Reset	Packages
PIC16C923	8	4K	176	TMR0, TMR1, TMR2	1	SPI/I²C	—	—	4 Com 32 Seg	8	25	27	3.0-6.0	Yes	—	64-pin SDIP(1), TQFP, 68-pin PLCC, DIE
PIC16C924	8	4K	176	TMR0, TMR1, TMR2	1	SPI/I²C	—	5	4 Com 32 Seg	9	25	27	3.0-6.0	Yes	—	64-pin SDIP(1), TQFP, 68-pin PLCC, DIE

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.
All PIC16CXX Family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local Microchip representative for availability of this package.

F.8 PIC17CXX Family of Devices

	Clock	Memory					Peripherals					Features			
	Maximum Frequency of Operation (MHz)	EPROM	ROM	RAM Data Memory (bytes)	Program Memory (Words)	Timer Module(s)	Captures	PWMs	Serial Port(s) (USART)	Hardware Multiply	External Interrupts	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Number of Instructions	Packages
PIC17C42	25	2K	—	232		TMR0,TMR1, TMR2,TMR3	2	2	Yes	—	Yes	11	33	4.5-5.5	55	40-pin DIP; 44-pin PLCC, MQFP
PIC17C42A	25	2K	—	232		TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-5.5	58	40-pin DIP; 44-pin PLCC, MQFP
PIC17CR42	25	—	2K	232		TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-5.5	58	40-pin DIP; 44-pin PLCC, MQFP
PIC17C43	25	4K	—	454		TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17CR43	25	—	4K	454		TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17C44	25	8K		454		TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.

F.9 Pin Compatibility

Devices that have the same package type and VDD, VSS and $\overline{MCLR}$ pin locations are said to be pin compatible. This allows these different devices to operate in the same socket. Compatible devices may only requires minor software modification to allow proper operation in the application socket (ex., PIC16C56 and PIC16C61 devices). Not all devices in the same package size are pin compatible; for example, the PIC16C62 is compatible with the PIC16C63, but not the PIC16C55.

Pin compatibility does not mean that the devices offer the same features. As an example, the PIC16C54 is pin compatible with the PIC16C71, but does not have an A/D converter, weak pull-ups on PORTB, or interrupts.

TABLE F-1: PIN COMPATIBLE DEVICES

Pin Compatible Devices	Package
PIC16C54, PIC16C54A, PIC16CR54, PIC16CR54A, PIC16CR54B, PIC16C56, PIC16CR56, PIC16C58A, PIC16CR58A, PIC16CR58B, PIC16C61, PIC16C620, PIC16C621, PIC16C622, PIC16C70, PIC16C71, PIC16C71A PIC16C83, PIC16CR83, PIC16C84, PIC16C84A, PIC16CR84	18 pin (20 pin)
PIC16C55, PIC16CR55, PIC16C57, PIC16CR57A, PIC16CR57B	28 pin
PIC16C62, PIC16CR62, PIC16C62A, PIC16C63, PIC16C72, PIC16C73, PIC16C73A	28 pin
PIC16C64, PIC16CR64, PIC16C64A, PIC16C65, PIC16C65A, PIC16C74, PIC16C74A	40 pin
PIC17C42, PIC17C43, PIC17C44	40 pin

NOTES:

ON-LINE SUPPORT

Microchip provides two methods of on-line support. These are the Microchip BBS and the Microchip World Wide Web (WWW) site.

Use Microchip's Bulletin Board Service (BBS) to get current information and help about Microchip products. Microchip provides the BBS communication channel for you to use in extending your technical staff with microcontroller and memory experts.

To provide you with the most responsive service possible, the Microchip systems team monitors the BBS, posts the latest component data and software tool updates, provides technical help and embedded systems insights, and discusses how Microchip products provide project solutions.

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

Connecting to the Microchip Internet Web Site

The Microchip web site is available by using your favorite Internet browser to attach to:

www.microchip.com

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

ftp.mchip.com/biz/mchip

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Datasheets, 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

Connecting to the Microchip BBS

Connect worldwide to the Microchip BBS using either the Internet or the CompuServe® communications network.

Internet:

You can telnet or ftp to the Microchip BBS at the address:

mchipbbs.microchip.com

CompuServe Communications Network:

When using the BBS via the Compuserve Network, in most cases, a local call is your only expense. The Microchip BBS connection does not use CompuServe membership services, therefore you do not need CompuServe membership to join Microchip's BBS. There is no charge for connecting to the Microchip BBS. The procedure to connect will vary slightly from country to country. Please check with your local CompuServe agent for details if you have a problem. CompuServe service allow multiple users various baud rates depending on the local point of access.

The following connect procedure applies in most locations.

1. Set your modem to 8-bit, No parity, and One stop (8N1). This is not the normal CompuServe setting which is 7E1.
2. Dial your local CompuServe access number.
3. Depress the <Enter> key and a garbage string will appear because CompuServe is expecting a 7E1 setting.
4. Type +, depress the <Enter> key and “Host Name:” will appear.
5. Type MCHIPBBS, depress the <Enter> key and you will be connected to the Microchip BBS.

In the United States, to find the CompuServe phone number closest to you, set your modem to 7E1 and dial (800) 848-4480 for 300-2400 baud or (800) 331-7166 for 9600-14400 baud connection. After the system responds with “Host Name:”, type NETWORK, depress the <Enter> key and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 723-1550 for your local CompuServe number.

Microchip regularly uses the Microchip BBS to distribute technical information, application notes, source code, errata sheets, bug reports, and interim patches for Microchip systems software products. For each SIG, a moderator monitors, scans, and approves or disapproves files submitted to the SIG. No executable files are accepted from the user community in general to limit the spread of computer viruses.

Systems Information and Upgrade Hot Line

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

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

1-602-786-7302 for the rest of the world.

960513

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (602) 786-7578.

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

To: Technical Publications Manager Total Pages Sent

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: **PIC16C84** Literature Number: **DS30445A**

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this data sheet easy to follow? If not, why?

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

5. What deletions from the data sheet could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?

8. How would you improve our software, systems, and silicon products?

INDEX

A

Absolute Maximum Ratings 67
ALU 7
Architectural Overview 7
Assembler 64

B

Block Diagram
- Interrupt Logic 45
- On-Chip Reset Circuit 38
- RA3:RA0 and RA5 Port Pins 19
- RA4 Pin 19
- RB3:RB0 Port Pins 21
- RB7:RB4 Port Pins 21
- TMR0/WDT Prescaler 28
- Watchdog Timer 47

Brown-out Protection Circuit 43

C

C Compiler (MP-C) 65
Carry 7
CLKIN 9
CLKOUT 9
Code Protection 35, 49
Compatibility, Upward 3
Computed GOTO 17
Configuration Bits 35

D

DC Characteristics 68, 69, 70, 71
Development Support 63
Development Tools 63
Digit Carry 7

E

Electrical Characteristics 67
External Power-on Reset Circuit 40

F

Family of Devices
- PIC14XXX 99
- PIC16C5X 100
- PIC16C62X 100
- PIC16C6X 102
- PIC16C7X 103
- PIC16C8X 3, 104
- PIC17CXX 106

FSR 18, 39
Fuzzy Logic Dev. System (*fuzzy*TECH®-MP) 63, 65

G

GIE 44

I

I/O Ports 19
I/O Programming Considerations 23
In-Circuit Serial Programming 35, 49
INDF 39
Instruction Set Summary 51
INT Interrupt 46
INTCON 16, 39, 44, 46
INTEDG 46
Interrupts
- Flag 44
- Interrupt on Change Feature 21
- Interrupts 35, 44

L

Loading of PC 17

M

$\overline{MCLR}$ 9, 38, 39
Memory Organization
- Data Memory 12
- Memory Organization 11
- Program Memory 11

MPASM Assembler 63, 64
MP-C C Compiler 65
MPSIM Software Simulator 63, 65

O

OPTION 15, 39, 46
OSC selection 35
OSC1 9
OSC2 9
Oscillator
- HS 36, 43
- LP 36, 43

Oscillator Configurations 36

P

Paging, Program Memory 17
PCL 17, 39
PCLATH 17, 39
$\overline{PD}$ 14, 38, 43
PICDEM-1 Low-Cost PIC16/17 Demo Board 63, 64
PICDEM-2 Low-Cost PIC16CXX Demo Board 63, 64
PICDEM-3 Low-Cost PIC16C9XXX Demo Board 64
PICMASTER™ RT In-Circuit Emulator 63
PICSTART™ Low-Cost Development System 63
Pin Compatible Devices 107
Pinout Descriptions 9
POR 40
- Oscillator Start-up Timer (OST) 35, 40
- Power-on Reset (POR) 35, 39, 40
- Power-up Timer (PWRT) 35, 40
- Time-out Sequence 43
- Time-out Sequence on Power-up 41
- $\overline{TO}$ 14, 38, 43

Port RB Interrupt 46
PORTA 9, 19, 39
PORTB 9, 21, 39
Power-down Mode (SLEEP) 48
Prescaler 27
PRO MATE™ Universal Programmer 63
Product Identification System 115

R

RBIF bit 21, 46
RC Oscillator 36, 37, 43
Read-Modify-Write 23
Register File 12
Reset 35, 38
Reset on Brown-Out 43

S

Saving W Register and STATUS in RAM 46
SLEEP 35, 38, 48
Software Simulator (MPSIM) 65
Special Features of the CPU 35
Special Function Registers 12
Stack 17

Overflows 17
Underflows 17
STATUS 7, 14, 39

T

Time-out 39
Timer0
Switching Prescaler Assignment 29
T0IF 46
Timer0 Module 25
TMR0 Interrupt 46
TMR0 with External Clock 27
Timing Diagrams
Time-out Sequence 41
Timing Diagrams and Specifications 73
TRISA 19
TRISB 21, 39

W

W 39
Wake-up from SLEEP 39, 48
Watchdog Timer (WDT) 35, 38, 39, 47
WDT 39
Period 47
Programming Considerations 47
Time-out 39

X

XT 36, 43

Z

Zero bit 7

LIST OF EXAMPLES

Example 3-1: Instruction Pipeline Flow 10
Example 4-1: Indirect Addressing 18
Example 4-2: How to Clear RAM Using Indirect Addressing 18
Example 5-1: Initializing PORTA 19
Example 5-2: Initializing PORTB 22
Example 5-3: Read-Modify-Write Instructions on an I/O Port 23
Example 6-1: Changing Prescaler (Timer0→WDT) 29
Example 6-2: Changing Prescaler (WDT→Timer0) 29
Example 7-1: Data EEPROM Read 32
Example 7-2: Data EEPROM Write 32
Example 7-3: Write Verify 33
Example 8-1: Saving STATUS and W Registers in RAM 46

LIST OF FIGURES

Figure 3-1: PIC16C84 Block Diagram 8
Figure 3-2: Clock/Instruction Cycle 10
Figure 4-1: Program Memory Map and Stack 11
Figure 4-2: Register File Map 12
Figure 4-3: STATUS Register (Address 03h, 83h) 14
Figure 4-4: OPTION Register (Address 81h) 15
Figure 4-5: INTCON Register (Address 0Bh, 8Bh) 16
Figure 4-6: Loading of PC in Different Situations 17
Figure 4-7: Direct/Indirect Addressing 18
Figure 5-1: Block Diagram of Pins RA3:RA0 19
Figure 5-2: Block Diagram of Pin RA4 19
Figure 5-3: Block Diagram of Pins RB7:RB4 21
Figure 5-4: Block Diagram of Pins RB3:RB0 21
Figure 5-5: Successive I/O Operation 23
Figure 6-1: TMR0 Block Diagram 25
Figure 6-2: TMR0 Timing: Internal Clock/ No Prescaler 25
Figure 6-3: TMR0 Timing: Internal Clock/ Prescale 1:2 26
Figure 6-4: TMR0 Interrupt Timing 26
Figure 6-5: Timer0 Timing With External Clock 27
Figure 6-6: Block Diagram of the TMR0/WDT Prescaler 28
Figure 7-1: EECON1 Register (Address 88h) 31
Figure 8-1: Configuration Word 35
Figure 8-2: Crystal/Ceramic Resonator Operation (HS, XT or LP OSC Configuration) 36
Figure 8-3: External Clock Input Operation (HS, XT or LP OSC Configuration) 36
Figure 8-4: External Parallel Resonant Crystal Oscillator Circuit 37
Figure 8-5: External Series Resonant Crystal Oscillator Circuit 37
Figure 8-6: RC Oscillator Mode 37
Figure 8-7: Simplified Block Diagram of On-Chip Reset Circuit 38
Figure 8-8: External Power-on Reset Circuit (For Slow VDD Power-up) 40
Figure 8-9: Time-out Sequence on Power-up ($\overline{MCLR}$ not Tied to VDD): Case 1 41
Figure 8-10: Time-out Sequence on Power-up ($\overline{MCLR}$ Not Tied To VDD): Case 2 41
Figure 8-11: Time-out Sequence on Power-up ($\overline{MCLR}$ Tied to VDD): Fast VDD Rise Time 42
Figure 8-12: Time-Out Sequence on Power-Up ($\overline{MCLR}$ Tied to VDD): Slow VDD Rise Time 42
Figure 8-13: Brown-out Protection Circuit 1 43
Figure 8-14: Brown-out Protection Circuit 2 43
Figure 8-15: Interrupt Logic 45
Figure 8-16: INT Pin Interrupt Timing 45
Figure 8-17: Watchdog Timer Block Diagram 47
Figure 8-18: Wake-up From Sleep Through Interrupt 48
Figure 8-19: Typical In-system Serial Programming Connection 49
Figure 9-1: General Format for Instructions 51
Figure 11-1: Parameter Measurement Information 72
Figure 11-2: Load Conditions 72
Figure 11-3: External Clock Timing 73
Figure 11-4: CLKOUT and I/O Timing 74
Figure 11-5: Reset, Watchdog Timer, Oscillator Start-up Timer and Power-up Timer Timing 75
Figure 11-6: Timer0 Clock Timings 76
Figure 12-1: Typical RC Oscillator Frequency vs. Temperature 77
Figure 12-2: Typical RC Oscillator Frequency vs. VDD (CEXT = 20 pF) 78
Figure 12-3: Typical RC Oscillator Frequency vs. VDD (CEXT = 100 pF) 79
Figure 12-4: Typical RC Oscillator Frequency vs. VDD (CEXT = 300 pF) 79
Figure 12-5: Typical IPD vs. VDD Watchdog Disabled (25˚C) 80
Figure 12-6: Typical IPD vs. VDD Watchdog Enabled (25˚C) 80
Figure 12-7: Maximum IPD vs. VDD Watchdog Disabled 81
Figure 12-8: Maximum IPD vs. VDD Watchdog Enabled* 81
Figure 12-9: VTH (Input Threshold Voltage) of I/O Pins vs. VDD 82
Figure 12-10: VTH (Input Threshold Voltage) of OSC1 Input (in XT, HS, and LP Modes) vs. VDD 82
Figure 12-11: VIH, VIL OF $\overline{MCLR}$, T0CKI and OSC1 (in RC Mode) vs. VDD 83
Figure 12-12: Typical IDD vs. Freq (Ext Clock, 25˚C) 84
Figure 12-13: Maximum IDD vs. Freq (Ext Clock, -40˚ to +85˚C) 84
Figure 12-14: WDT Time-out Period vs. VDD 85
Figure 12-15: Transconductance (gm) of HS Oscillator vs. VDD 85
Figure 12-16: Transconductance (gm) of LP Oscillator vs. VDD 86
Figure 12-17: Transconductance (gm) of XT Oscillator vs. VDD 86
Figure 12-18: IOH vs. VOH, VDD = 3V 87
Figure 12-19: IOH vs. VOH, VDD = 5V 87
Figure 12-20: IOL vs. VOL, VDD = 3V 88
Figure 12-21: IOL vs. VOL, VDD = 5V 88

LIST OF TABLES

Table 1-1: PIC16C8X Family of Devices 4
Table 3-1: PIC16C8X Pinout Description..... 9
Table 4-1: Register File Summary 13
Table 5-1: PORTA Functions..... 20
Table 5-2: Summary of Registers Associated With PORTA 20
Table 5-3: PORTB Functions..... 22
Table 5-4: Summary of Registers Associated With PORTB 22
Table 6-1: Registers Associated with Timer0 29
Table 7-1: Registers/Bits Associated with Data EEPROM..... 33
Table 8-1: PIC16C84 Capacitor Selection for Ceramic Resonators 36
Table 8-2: PIC16C84 Capacitor Selection for Crystal Oscillator..... 36
Table 8-3: Reset Condition for Program Counter and the STATUS Register 39
Table 8-4: Reset Conditions for All Registers 39
Table 8-5: Time-out in Various Situations..... 43
Table 8-6: STATUS bits and Their Significance 43
Table 8-7: Summary of Registers Associated With the Watchdog Timer 47
Table 9-1: OPCODE Field Descriptions..... 51
Table 9-2: Instruction Set Summary 52
Table 10-1: Development Tools from Microchip 66
Table 11-1: Cross Reference of Device Specs for Oscillator Configurations and Frequencies of Operation (Commercial Devices) 67
Table 11-2: Timing Parameter Symbology 72
Table 11-3: External Clock Timing Requirements 73
Table 11-4: CLKOUT and I/O Timing Requirements 74
Table 11-5: Reset, Watchdog Timer, Oscillator Start-up Timer and Power-up Timer Requirements 75
Table 11-6: Timer0 Clock Requirements 76
Table 12-1: RC Oscillator Frequencies * 77
Table 12-2: Input Capacitance * 89
Table F-1: Pin Compatible Devices..... 107

PIC16C84 PRODUCT IDENTIFICATION SYSTEM

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

PART NO.	-XX	X	/XX	XXX
Device	Frequency Range	Temperature Range	Package	Pattern

Device	PIC16C84[2], PIC16C84T[3] PIC16LC84[2], PIC16LC84T[3]	
Frequency Range	04 10	= 4 MHz = 10 MHz
Temperature Range	b[1] I	= 0°C to +70°C (Commercial) = -40°C to +85°C (Industrial)
Package	P SO	= PDIP = SOIC (Gull Wing, 300 mil body)
Pattern	3-digit Pattern Code for QTP, ROM (blank otherwise)	

Examples:

a) PIC16C84 -04/P 301 = Commercial temp., PDIP package, 4MHz, normal V_{DD} limitis, QTP pattern #301.

b) PIC16LC84 - 04I/SO = Industrial temp., SOIC package, 200kHz, Extended V_{DD} limits.

Note 1: b = blank
2: C = Standard V_{DD} range
LC = Extended V_{DD} range
3: T = in tape and reel - SOIC, SSOP packages only.

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 (see last page)
2. The Microchip Corporate Literature Center U.S. FAX: (602) 786-7277
3. The Microchip's Bulletin Board, via your local CompuServe number (CompuServe membership NOT required).

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

Development Tools

For the latest version information and upgrade kits for Microchip Development Tools, please call 1-800-755-2345 or 1-602-786-7302. The latest version of Development Tools software can be downloaded from either our Bulletin Board or Worldwide Web Site. (Information on how to connect to our BBS or WWW site can be found in the On-Line Support section of this data sheet.)

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: http://www.mchip.com/microchip

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
Suite 150
Two Prestige Place
Miamisburg, OH 45342
Tel: 513 291-1654 Fax: 513 291-9175

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

Toronto
Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905 405-6279 Fax: 905 405-6253

ASIA/PACIFIC

Hong Kong
Microchip Technology
RM 3801B, Tower Two
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea
Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore
Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan
Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 1628 851077 Fax: 44 1628 850259

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 39 689 9939 Fax: 39 39 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

6/14/96

MICROCHIP