



PIC16C52

Table of Contents

1.0	General Description	3
2.0	PIC16C52 Device Varieties	5
3.0	Architectural Overview	7
4.0	Memory Organization	11
5.0	I/O Ports.....	17
6.0	Timer0 Module and TMR0 Register.....	19
7.0	Special Features of the CPU	23
8.0	Instruction Set Summary	33
9.0	Development Support	45
10.0	Electrical Characteristics	51
11.0	DC and AC Characteristics.....	59
12.0	Packaging Information	67
	Appendix A: Compatibility	71
	Appendix B: What's New	71
	Appendix C: PIC16/17 Microcontrollers.....	73
	Index	81
	List of Examples	82
	List of Figures	83
	List of Tables	84
	Connecting to Microchip BBS	85
	Access to the Internet	85
	Reader Response	86
	PIC16C52 Product Identification System.....	87

To Our Valued Customers

We constantly strive to improve the quality of all our products and documentation. We have spent an exceptional amount 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.

To assist you in the use of this document, Appendix B contains a list of new information in this data sheet, while Appendix C contains information that has changed

1.0 GENERAL DESCRIPTION

The PIC16C52 from Microchip Technology is a low-cost, high performance, 8-bit, fully static, EPROM-based CMOS microcontroller. It employs a RISC architecture with only 33 single word/single cycle instructions. All instructions are single cycle except for program branches which take two cycles. The PIC16C52 delivers performance an order of magnitude higher than its competitors in the same price category. The 12-bit wide instructions are highly symmetrical resulting in 2:1 code compression over other 8-bit microcontrollers in its class. The easy to use and easy to remember instruction set reduces development time significantly.

The PIC16C52 is equipped with special features that reduce system cost and power requirements. The Power-On Reset (POR) and Device Reset Timer (DRT) eliminate the need for external reset circuitry. There are two oscillator configurations to choose from: the cost-saving RC oscillator and the standard XT crystal/resonator. Power-saving SLEEP mode and code protection features improve system cost, power and reliability.

This cost-effective, One Time Programmable (OTP) device is suitable for production in any volume. The customer can take full advantage of Microchip's price leadership in OTP microcontrollers while benefiting from the OTP's flexibility.

The PIC16C52 is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a 'C' compiler, fuzzy logic support tools, a low-cost development programmer, and a full featured programmer. All the tools are supported on IBM PC-AT[®] and compatible machines.

PIC16C52

TABLE 1-1: PIC16C5X FAMILY OF DEVICES

	Clock				Memory		Peripherals		Features	
	Maximum Frequency of Operation (MHz)	Program Memory (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	3.0-6.25	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	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP	
PIC16CR54 ⁽²⁾	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	
PIC16CR54B ⁽¹⁾	20	—	512	25	TMR0	12	2.5-6.25	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	
PIC16CR56 ⁽¹⁾	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	
PIC16CR57A ⁽²⁾	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	2.5-6.25	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	
PIC16CR58B ⁽¹⁾	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 (except PIC16C52), selectable code protect and high I/O current capability (except PIC16C52).

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

2: Not recommended for new designs.

2.0 PIC16C52 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 PIC16C52 Product Identification System at the back of this data sheet to specify the correct part number.

2.1 One-Time-Programmable (OTP) Devices

The availability of OTP devices is especially useful for customers expecting frequent code changes and updates.

The OTP devices, packaged in plastic packages, permit the user to program them once. In addition to the program memory, the configuration bits must be programmed.

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 are identical to the OTP devices but with all EPROM locations and configuration bit options already programmed by the factory. Certain code and prototype verification procedures apply before production shipments are available. Please contact your Microchip Technology sales office for more details.

2.3 Serialized Quick-Turnaround-Production (SQTP) 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.

PIC16C52

NOTES:

3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16C52 can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16C52 uses a Harvard architecture in which program and data are accessed on separate buses. This improves bandwidth over traditional von Neumann architecture where program and data are fetched on the same bus. Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word. Instruction opcodes are 12-bits wide making it possible to have all single word instructions. A 12-bit wide program memory access bus fetches a 12-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions. Consequently, all instructions (33) execute in a single cycle except for program branches.

The PIC16C52 addresses 384 x 12 program memory. All program memory is internal.

The PIC16C52 can directly or indirectly address its register files and data memory. All special function registers including the program counter are mapped in the data memory. The PIC16C52 has a highly 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 PIC16C52 simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16C52 device contains 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 W (working) register. The other operand is a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

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

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

A simplified block diagram is shown in Figure 3-1, with the corresponding device pins described in Table 3-1.

PIC16C52

FIGURE 3-1: PIC16C52 BLOCK DIAGRAM

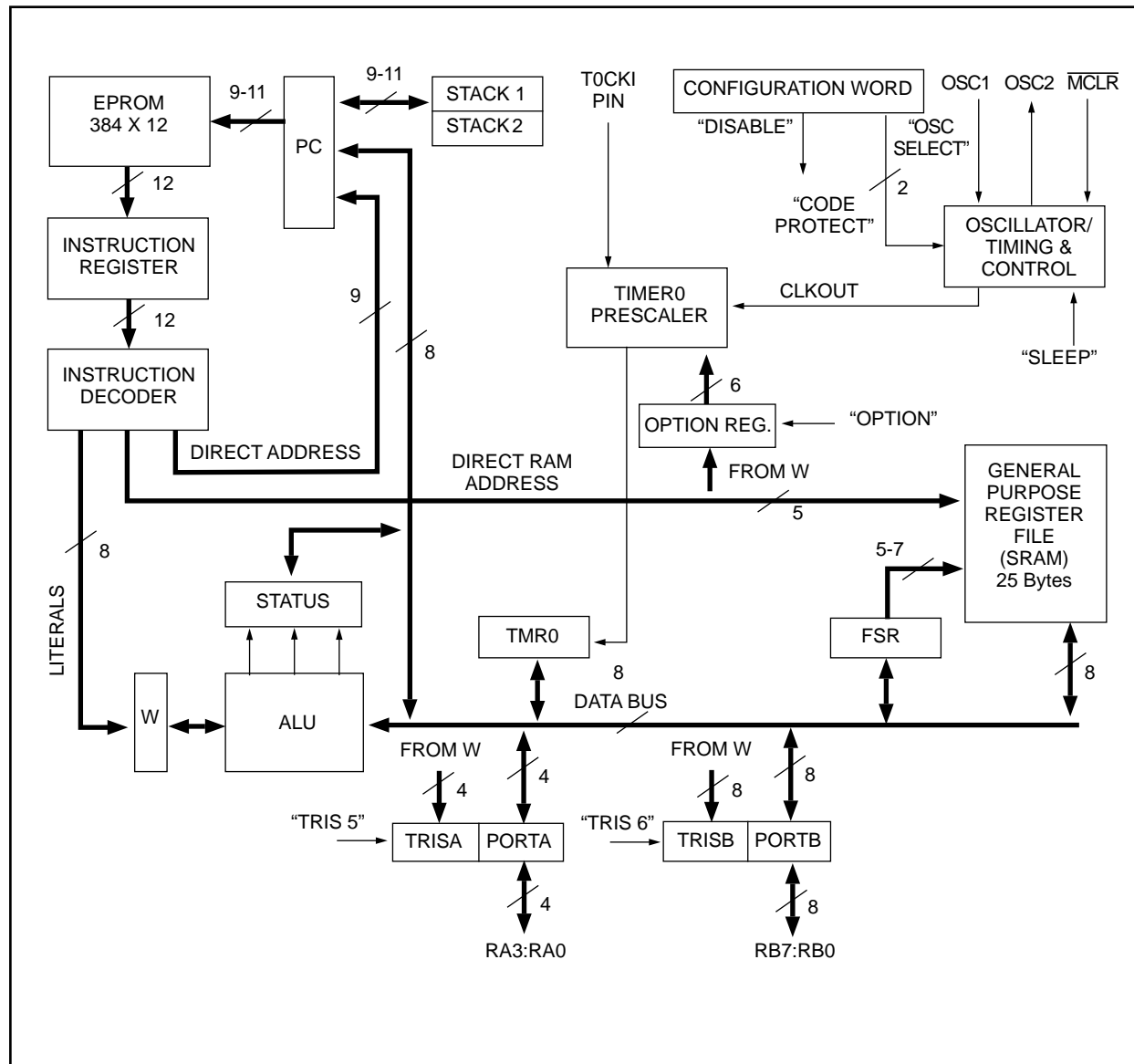


TABLE 3-1: PIC16C52 PINOUT DESCRIPTION

Name	PDIP, SOIC No.	I/O/P Type	Input Levels	Description
RA0	17	I/O	TTL	Bi-directional I/O port
RA1	18	I/O	TTL	
RA2	1	I/O	TTL	
RA3	2	I/O	TTL	
RB0	6	I/O	TTL	Bi-directional I/O port
RB1	7	I/O	TTL	
RB2	8	I/O	TTL	
RB3	9	I/O	TTL	
RB4	10	I/O	TTL	
RB5	11	I/O	TTL	
RB6	12	I/O	TTL	
RB7	13	I/O	TTL	
T0CKI	3	I	ST	Clock input to Timer0. Must be tied to Vss or VDD, if not in use, to reduce current consumption.
MCLR/VPP	4	I	ST	Master clear (reset) input/programming voltage input. This pin is an active low reset to the device. Voltage on MCLR/VPP must not exceed VDD to avoid unintended entering of programming mode.
OSC1/CLKIN	16	I	ST	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	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.
VDD	14	P	—	Positive supply for logic and I/O pins.
Vss	5	P	—	Ground reference 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

PIC16C52

3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN pin) is internally divided by four to generate four non-overlapping quadrature clocks namely Q1, Q2, Q3 and Q4. Internally, the program counter (PC) is incremented every Q1, and the instruction is fetched from program memory and latched into the instruction register in Q4. It is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2 and Example 3-1.

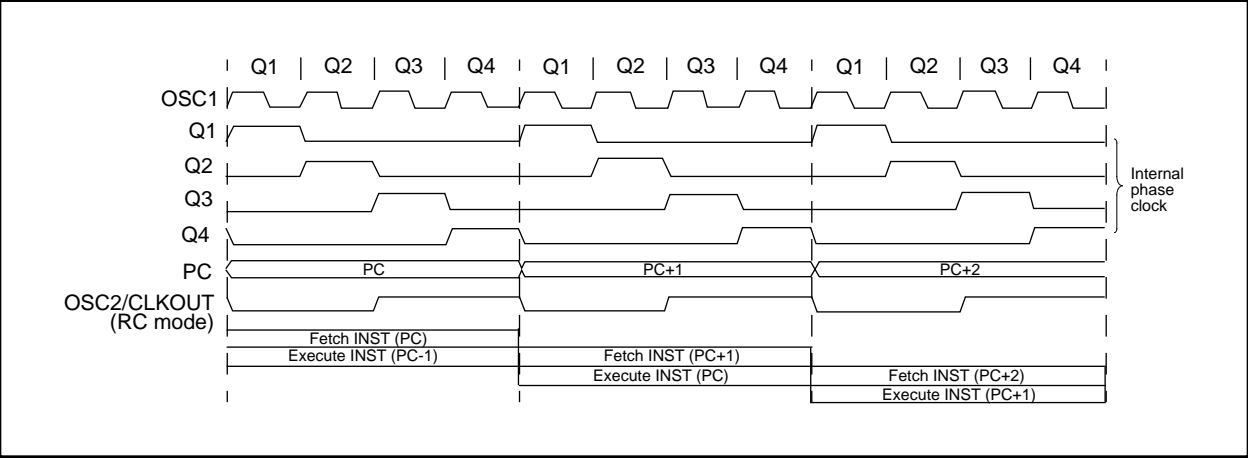
3.2 Instruction Flow/Pipelining

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

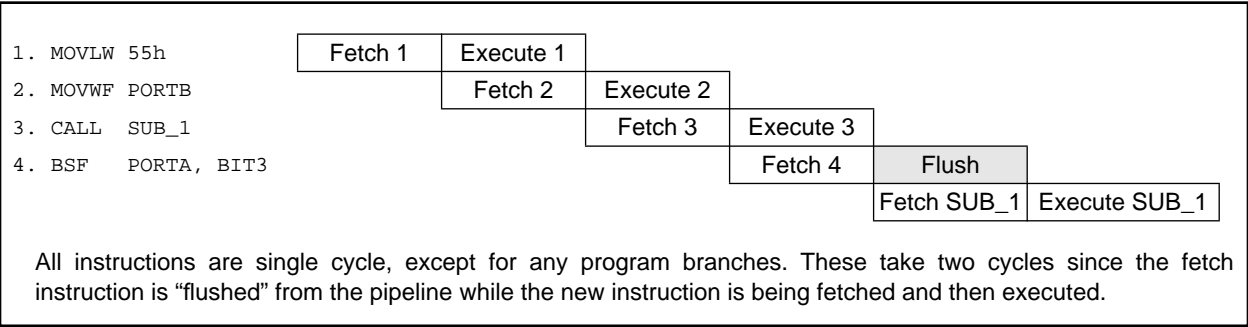
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW



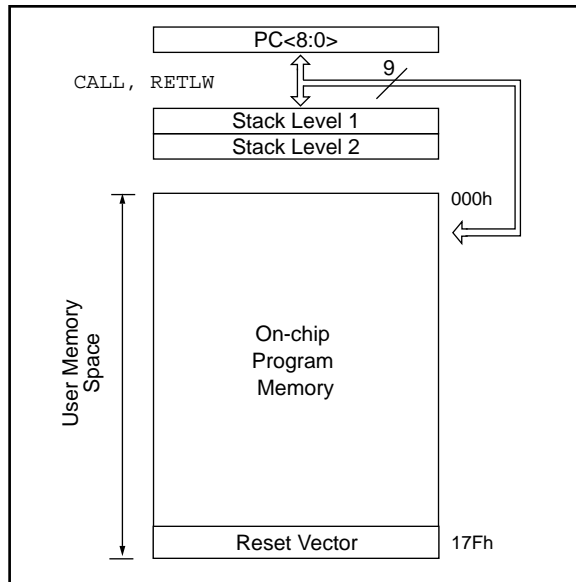
4.0 MEMORY ORGANIZATION

4.1 Program Memory Organization

The PIC16C52 has a 9-bit Program Counter (PC) capable of addressing a 384 x 12 program memory space (Figure 4-1).

The reset vector for the PIC16C52 is at 17Fh. A NOP at the reset vector location will cause a restart at location 000h.

FIGURE 4-1: PIC16C52 PROGRAM MEMORY MAP AND STACK



4.2 Data Memory Organization

Data memory is composed of registers, or bytes of RAM. Therefore, data memory for a device is specified by its register file. The register file is divided into two functional groups: special function registers and general purpose registers.

The special function registers include the TMR0 register, the Program Counter (PC), the Status Register, the I/O registers (ports), and the File Select Register (FSR). In addition, special purpose registers are used to control the I/O port configuration and prescaler options.

The general purpose registers are used for data and control information under command of the instructions.

For the PIC16C52, the register file is composed of seven special function registers and 25 general purpose registers (Figure 4-2).

4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is accessed either directly or indirectly through the file select register FSR (Section 4.7).

FIGURE 4-2: PIC16C52 REGISTER FILE MAP

File Address	
00h	INDF ⁽¹⁾
01h	TMR0
02h	PCL
03h	STATUS
04h	FSR
05h	PORTA
06h	PORTB
07h	General Purpose Registers
0Fh	
10h	
1Fh	

Note 1: Not a physical register. See Section 4.7

PIC16C52

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral functions to control the operation of the device (Table 4-1).

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

TABLE 4-1: SPECIAL FUNCTION REGISTER 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 MCLR Reset
N/A	TRIS	I/O control registers (TRISA and TRISB)								1111 1111	1111 1111
N/A	OPTION	Contains control bits to configure Timer0 and Timer0 prescaler								--11 1111	--11 1111
00h	INDF	Uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu
01h	TMR0	8-bit real-time clock/counter								xxxx xxxx	uuuu uuuu
02h ⁽¹⁾	PCL	Low order 8 bits of PC								1111 1111	1111 1111
03h	STATUS	PA2	PA1	PA0	T0	PD	Z	DC	C	0001 1xxx	000q quuu
04h	FSR	Indirect data memory address pointer								1xxx xxxx	1uuu uuuu
05h	PORTA	—	—	—	—	RA3	RA2	RA1	RA0	---- xxxx	---- uuuu
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu

Legend: Shaded boxes = unimplemented or unused, — = unimplemented, read as '0' (if applicable)
x = unknown, u = unchanged, q = see the tables in Section 7.6 for possible values.

Note 1: The upper byte of the Program Counter is not directly accessible. See Section 4.5 for an explanation of how to access these bits.

4.3 STATUS Register

This register contains the arithmetic status of the ALU, and the RESET status.

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

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

It is recommended, therefore, that only `BCF`, `BSF` and `MOVWF` instructions be used to alter the STATUS register because these instructions do not affect the Z, DC or C bits from the STATUS register. For other instructions which do affect STATUS bits, see Table 8-2, Instruction Set Summary.

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

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
PA2	PA1	PA0	\overline{TO}	\overline{PD}	Z	DC	C
bit7	6	5	4	3	2	1	bit0

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

bit 7-5: **PA2:PA0:** Page select bits - unused.
Use of the PA2:PA0 bits as a general purpose read/write bit is not recommended, since this may affect upward compatibility.

bit 4: **\overline{TO} :** Time-out bit
This bit always set on the PIC16C52.

bit 3: **\overline{PD} :** Power-down bit
1 = After power-up
0 = After `SLEEP` instruction

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

bit 1: **DC:** Digit carry/borrow bit (for `ADDWF` and `SUBWF` instructions)
ADDWF
1 = A carry from the 4th low order bit of the result occurred
0 = A carry from the 4th low order bit of the result did not occur
SUBWF
1 = A borrow from the 4th low order bit of the result did not occur
0 = A borrow from the 4th low order bit of the result occurred

bit 0: **C:** Carry/borrow bit (for `ADDWF`, `SUBWF` and `RRF`, `RLF` instructions)
ADDWF
1 = A carry occurred
0 = A carry did not occur
SUBWF
1 = A borrow did not occur
0 = A borrow occurred
RRF or RLF
Load bit with LSB or MSb

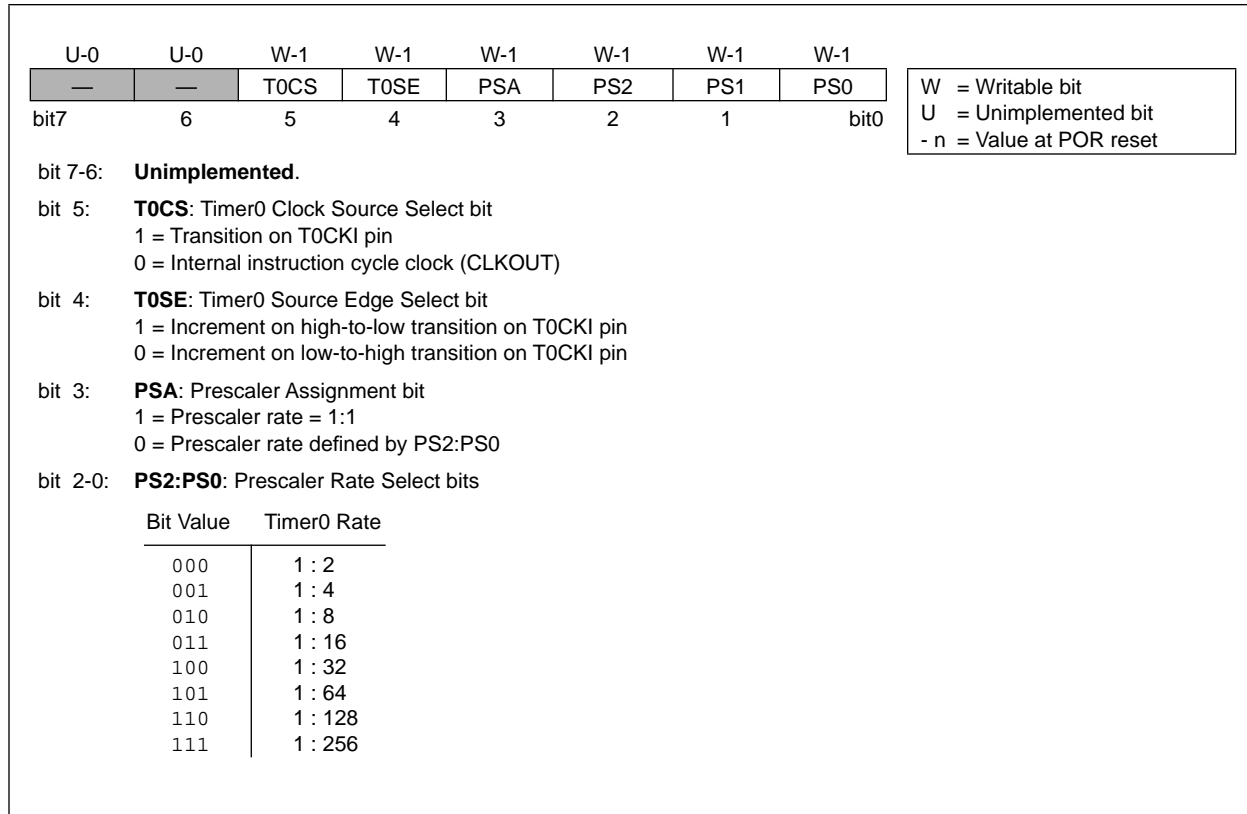
PIC16C52

4.4 OPTION Register

The OPTION register is a 6-bit wide, write-only register which contains various control bits to configure the Timer0 prescaler and Timer0.

By executing the OPTION instruction, the contents of the W register will be transferred to the OPTION register. A RESET sets the OPTION<5:0> bits.

FIGURE 4-4: OPTION REGISTER



4.5 Program Counter

As a program instruction is executed, the Program Counter (PC) will contain the address of the next program instruction to be executed. The PC value is increased by one every instruction cycle, unless an instruction changes the PC.

For a **GOTO** instruction, bits 8:0 of the PC are provided by the **GOTO** instruction word. The PC Latch (PCL) is mapped to PC<7:0> (Figure 4-5).

For a **CALL** instruction, or any instruction where the PCL is the destination, bits 7:0 of the PC are provided by the instruction word. However, PC<8> does not come from the instruction word, but is always cleared.

Instructions where the PCL is the destination, or Modify PCL instructions, include **MOVWF PC**, **ADDWF PC**, and **BSF PC, 5**.

For the **RETLW** instruction, the PC is loaded with the Top Of Stack (TOS) contents. All of the devices covered in this data sheet have only two stacks. Each stack has the same bit width as the device PC.

4.6 Stack

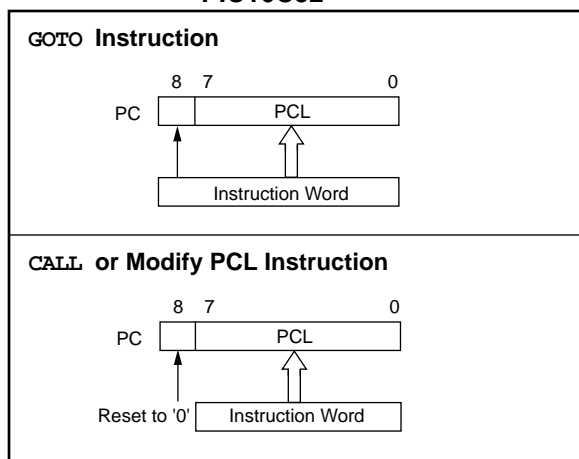
The PIC16C52 device has a 9-bit wide, two-level hardware push/pop stack (Figure 4-1).

A **CALL** instruction will *push* the current value of stack 1 into stack 2 and then push the current program counter value, incremented by one, into stack level 1. If more than two sequential **CALL**'s are executed, only the most recent two return addresses are stored.

A **RETLW** instruction will *pop* the contents of stack level 1 into the program counter and then copy stack level 2 contents into level 1. If more than two sequential **RETLW**'s are executed, the stack will be filled with the address previously stored in level 2.

Note: The W register will be loaded with the literal value specified in the instruction. This is particularly useful for the implementation of data look-up tables within the program memory.

**FIGURE 4-5: LOADING OF PC
BRANCH INSTRUCTIONS -
PIC16C52**



4.5.1 EFFECTS OF RESET

The Program Counter is set upon a **RESET**, which means that the PC addresses the last location in the last page (i.e., the reset vector).

PIC16C52

4.7 Indirect Data 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 INDF 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 10h-1Fh using indirect addressing is shown in Example 4-2.

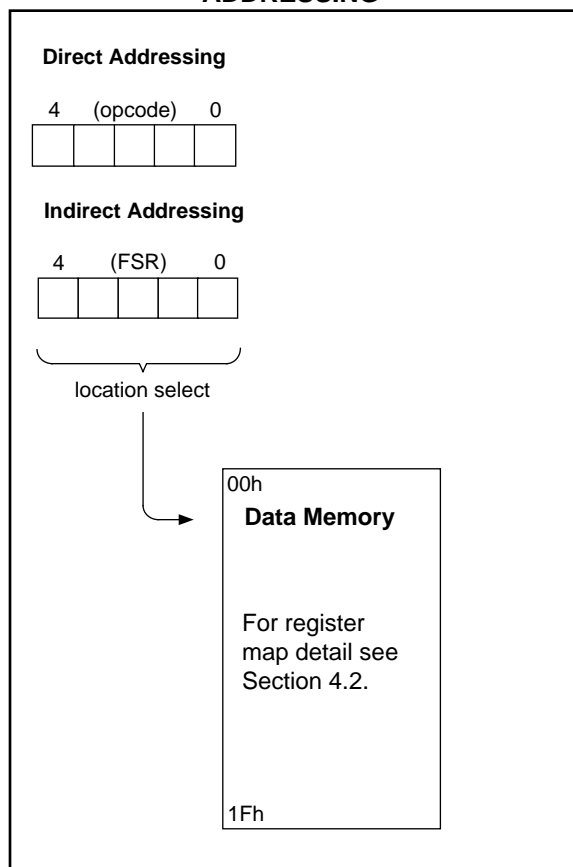
EXAMPLE 4-2: HOW TO CLEAR RAM USING INDIRECT ADDRESSING

```
        movlw 0x10 ;initialize pointer
        movwf FSR ; to RAM
NEXT    clrf  INDF ;clear INDF register
        incf  FSR,F ;inc pointer
        btfsc FSR,4 ;all done?
        goto  NEXT ;NO, clear next

CONTINUE
        :          ;YES, continue
```

The FSR is a 5-bit wide register. It is used in conjunction with the INDF register to indirectly address the data memory area.

FIGURE 4-6: DIRECT/INDIRECT ADDRESSING



5.0 I/O PORTS

As with any other register, the I/O registers can be written and read under program control. However, read instructions (e.g., `MOVF PORTB, W`) always read the I/O pins independent of the pin's input/output modes. On RESET, all I/O ports are defined as input (inputs are at hi-impedance) since the I/O control registers (TRISA, TRISB) are all set.

5.1 PORTA

PORTA is a 4-bit I/O register. Only the low order 4 bits are used (RA3:RA0). Bits 7-4 are unimplemented and read as '0's.

5.2 PORTB

PORTB is an 8-bit I/O register (PORTB<7:0>).

5.3 TRIS Registers

The output driver control registers are loaded with the contents of the W register by executing the `TRIS f` instruction. A '1' from a TRIS register bit puts the corresponding output driver in a hi-impedance mode. A '0' puts the contents of the output data latch on the selected pins, enabling the output buffer.

Note: A read of the ports reads the pins, not the output data latches. That is, if an output driver on a pin is enabled and driven high, but the external system is holding it low, a read of the port will indicate that the pin is low.

The TRIS registers are "write-only" and are set (output drivers disabled) upon RESET.

5.4 I/O Interfacing

The equivalent circuit for an I/O port pin is shown in Figure 5-1. All ports may be used for both input and output operations. For input operations these ports are non-latching. Any input must be present until read by an input instruction (e.g., `MOVF PORTB, W`). The outputs are latched and remain unchanged until the output latch is rewritten. To use a port pin as output, the corresponding direction control bit (in TRISA, TRISB) must be cleared (= 0). For use as an input, the corresponding TRIS bit must be set. Any I/O pin can be programmed individually as input or output.

FIGURE 5-1: EQUIVALENT CIRCUIT FOR A SINGLE I/O PIN

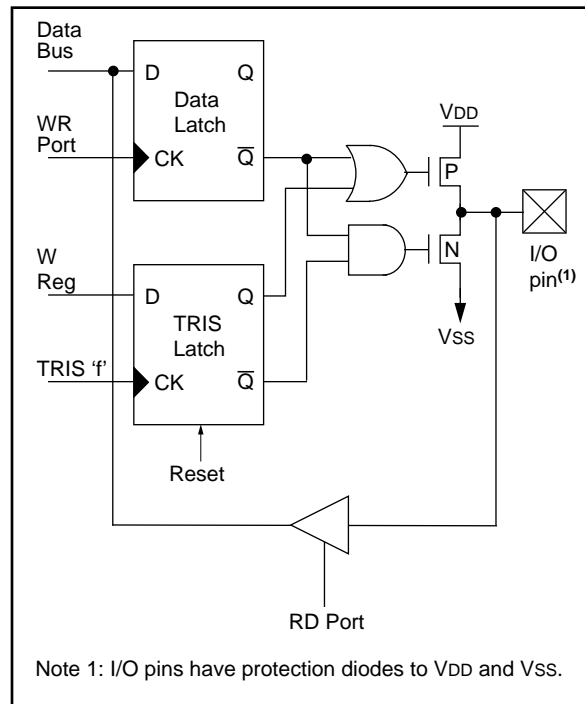


TABLE 5-1: SUMMARY OF PORT REGISTERS

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 MCLR Reset
N/A	TRIS	I/O control registers (TRISA and TRISB)								1111 1111	1111 1111
05h	PORTA	—	—	—	—	RA3	RA2	RA1	RA0	---- xxxx	---- uuuu
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu

Legend: Shaded boxes = unimplemented, read as '0',
 — = unimplemented, read as '0', x = unknown, u = unchanged

PIC16C52

5.5 I/O Programming Considerations

5.5.1 BI-DIRECTIONAL I/O PORTS

Some instructions operate internally as read followed by write operations. The BCF and BSF instructions, for example, read the entire port into the CPU, execute the bit operation and re-write the result. Caution must be used when these instructions are applied to a port where one or more pins are used as input/outputs. For example, a BSF operation on bit5 of PORTB will cause all eight bits of PORTB to be read into the CPU, bit5 to be set and the PORTB value to be written to the output latches. If another bit of PORTB is used as a bi-directional I/O pin (say 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 may now be unknown.

Example 5-1 shows the effect of two sequential read-modify-write instructions (e.g., BCF, BSF, etc.) on an I/O port.

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

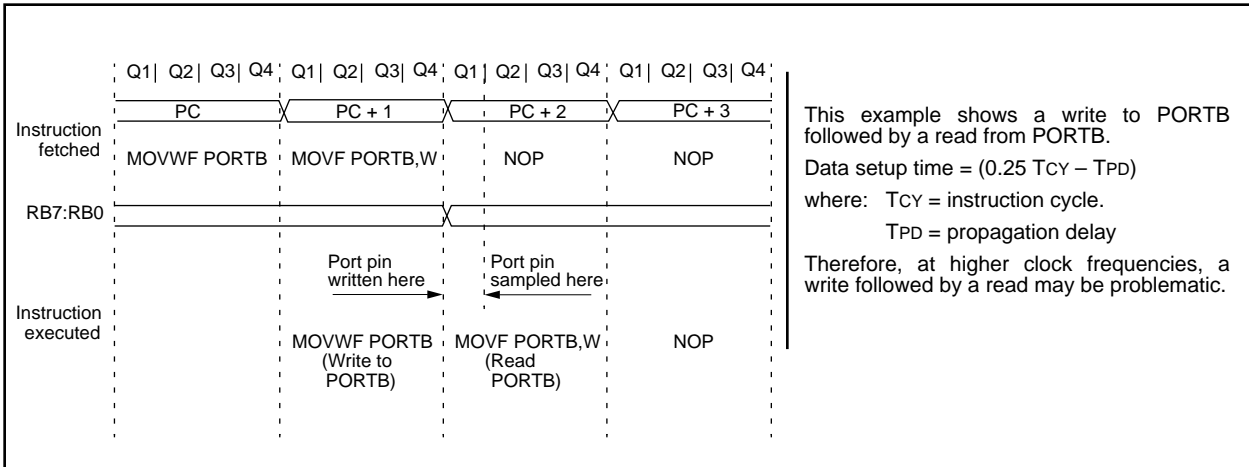
EXAMPLE 5-1: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

```
;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 pppp  11pp pppp
BCF  PORTB, 6  ;10pp pppp  11pp pppp
MOVLW 03Fh    ;
TRIS  PORTB    ;10pp pppp  10pp pppp
;
;Note that the user may have expected the pin
;values to be 00pp pppp. The 2nd BCF caused
;RB7 to be latched as the pin value (High).
```

5.5.2 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 5-2). 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 allow the pin voltage to stabilize (load dependent) before the next instruction, which causes that file to be read into the CPU, is executed. Otherwise, the previous state of that pin may be read into the CPU rather than the new state. When in doubt, it is better to separate these instructions with a NOP or another instruction not accessing this I/O port.

FIGURE 5-2: SUCCESSIVE I/O OPERATION



6.0 TIMER0 MODULE AND TMR0 REGISTER

The Timer0 module has the following features:

- 8-bit timer/counter register, TMR0
 - Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
 - Edge select for external clock

Figure 6-1 is a simplified block diagram of the Timer0 module, while Figure 6-2 shows the electrical structure of the Timer0 input.

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If TMR0 register is written, the increment is inhibited for the following two cycles (Figure 6-3 and Figure 6-4). 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, Timer0 will increment either on every rising or falling edge of pin T0CKI. The incrementing edge is determined by the source edge select bit T0SE (OPTION<4>). Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.1.

The prescaler assignment is controlled in software by the control bit PSA (OPTION<3>). When the PSA bit is set, the prescaler is not used (prescaler = 1:1). When the PSA bit is cleared, prescale values of 1:2, 1:4,..., 1:256 are selectable by the PS2:PS0 bits. Section 6.2 details the operation of the prescaler.

A summary of registers associated with the Timer0 module is found in Table 6-1.

FIGURE 6-1: TIMER0 BLOCK DIAGRAM

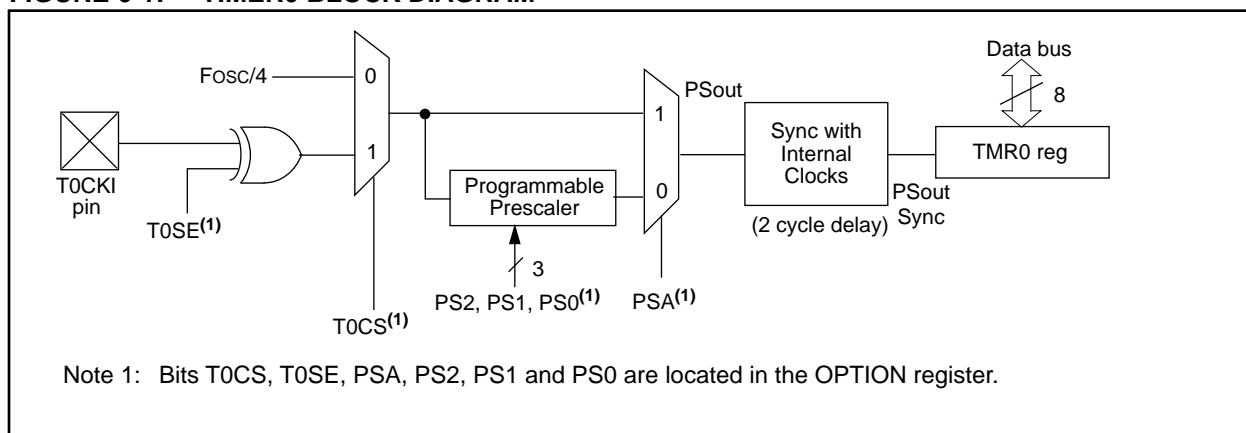
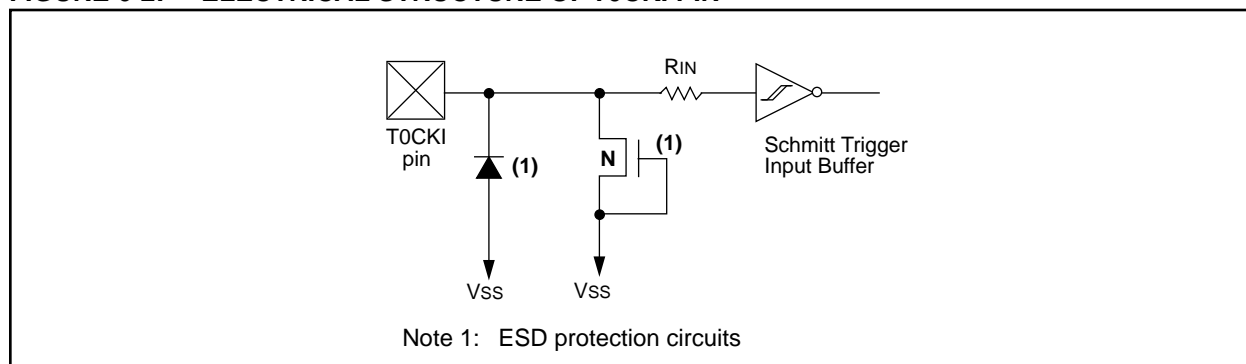


FIGURE 6-2: ELECTRICAL STRUCTURE OF T0CKI PIN



6.1 Using Timer0 with an External Clock

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

6.1.1 EXTERNAL CLOCK SYNCHRONIZATION

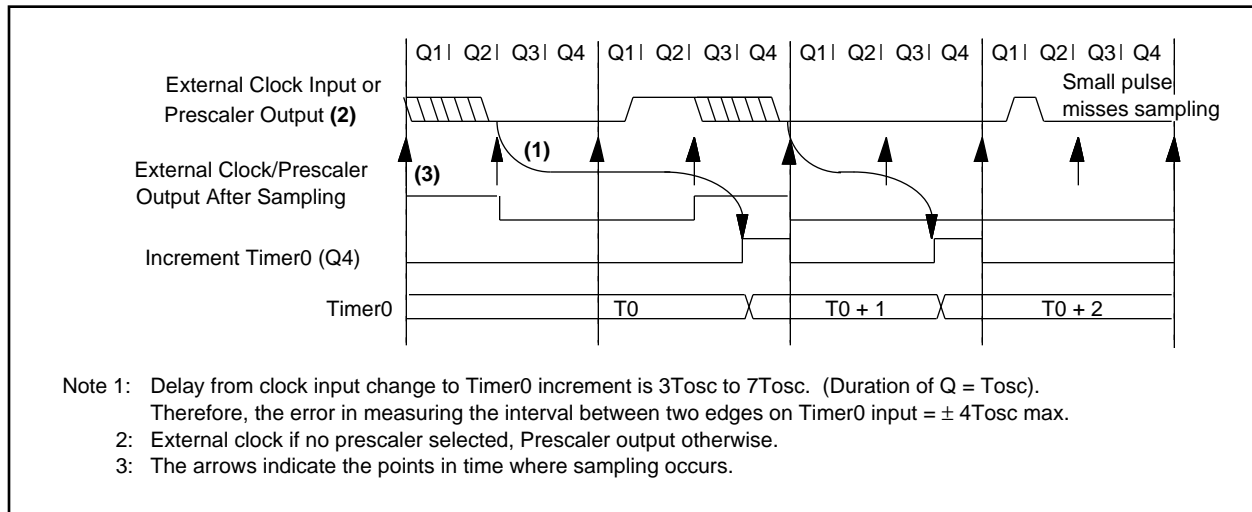
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 6-5). Therefore, it is necessary for T0CKI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

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

6.1.2 TIMER0 INCREMENT DELAY

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

FIGURE 6-5: TIMER0 TIMING WITH EXTERNAL CLOCK



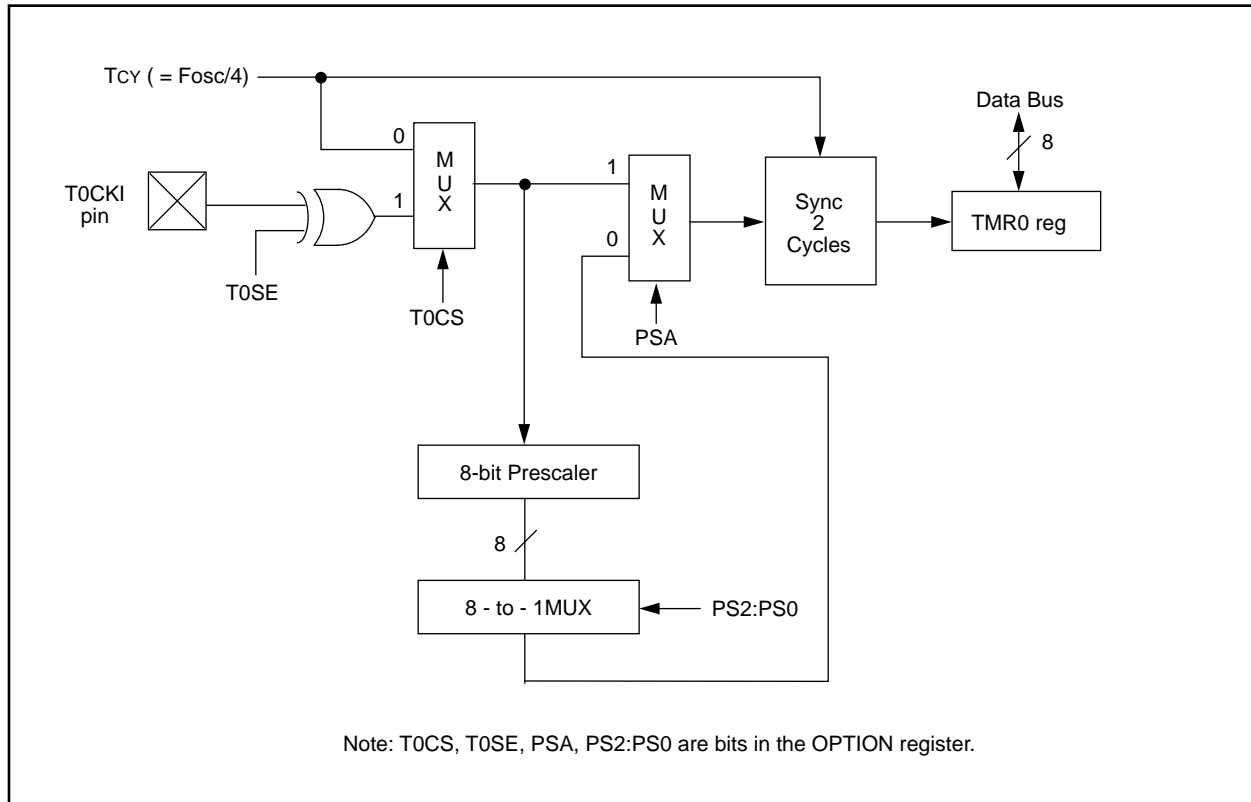
PIC16C52

6.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The PSA and PS2:PS0 bits (OPTION<3:0>) determine prescaler assignment and prescale ratio.

When assigned, all instructions writing to the TMR0 register (e.g., CLRF 1, MOVWF 1, BSF 1,x, etc.) will clear the prescaler. The prescaler is neither readable nor writable. On a RESET, the prescaler contains all '0's.

FIGURE 6-6: BLOCK DIAGRAM OF THE TIMER0 PRESCALER



7.0 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 PIC16C52 microcontroller 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:

- Oscillator selection
- Reset
- Power-On Reset (POR)
- Device Reset Timer (DRT)
- SLEEP
- Code protection
- ID locations

For the PIC16C52, there is an 18 ms delay provided by the Device Reset Timer (DRT), intended to keep the chip in reset until the crystal oscillator is stable. With this timer on-chip, most applications need no external reset circuitry.

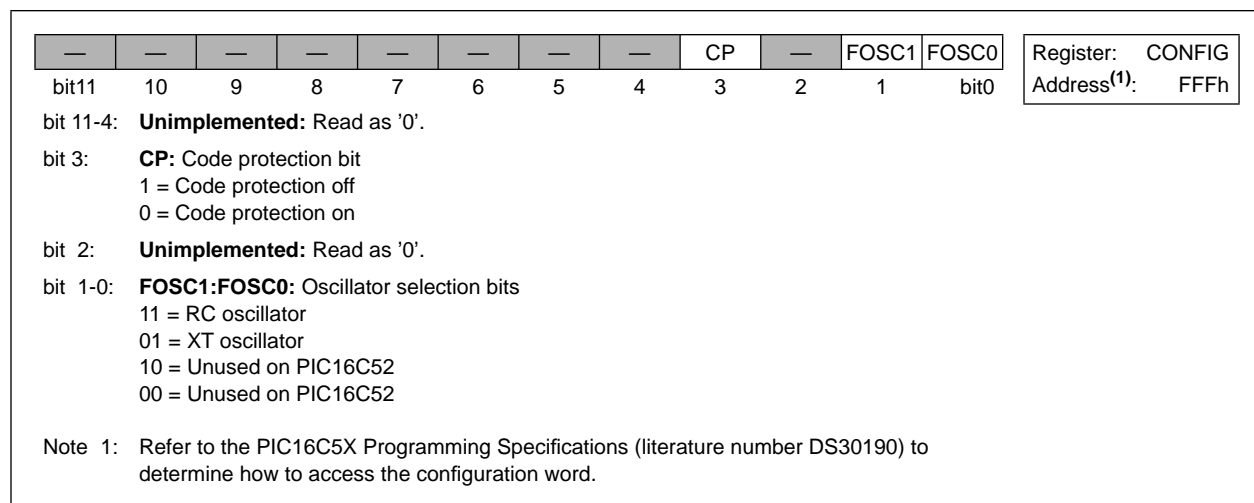
The SLEEP mode is designed to offer a very low current power-down mode. The user can wake up from SLEEP through external reset. Several oscillator options are also made available to allow the part to fit the application. The RC oscillator option saves system cost. A set of configuration bits are used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

7.1 Configuration Bits

The PIC16C52 configuration word consists of 12 bits, 4 of which are implemented. Configuration bits can be programmed to select various device configurations. Two bits are for the selection of the oscillator type and two are unused on this device (Figure 7-1).

OTP or QTP devices have the oscillator configuration programmed at the factory and these parts are tested accordingly (see "Product Identification System" on the inside back cover).

FIGURE 7-1: CONFIGURATION WORD FOR PIC16C52



PIC16C52

7.2 Oscillator Configurations

7.2.1 OSCILLATOR TYPES

The PIC16C52 can be operated in two different oscillator modes. The user can program two configuration bits (FOSC1:FOSC0) to select one of these modes:

- RC: Resistor/Capacitor
- XT: Crystal/Resonator

7.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT mode, a crystal or ceramic resonator is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 7-2). The PIC16C52 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 mode, the device can have an external clock source drive the OSC1/CLKIN pin (Figure 7-3).

FIGURE 7-2: CRYSTAL OPERATION OR CERAMIC RESONATOR (XT OSC CONFIGURATION)

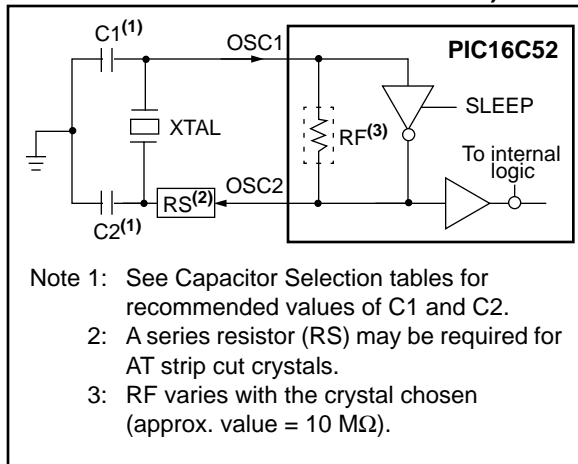


FIGURE 7-3: EXTERNAL CLOCK INPUT OPERATION (XT OSC CONFIGURATION)

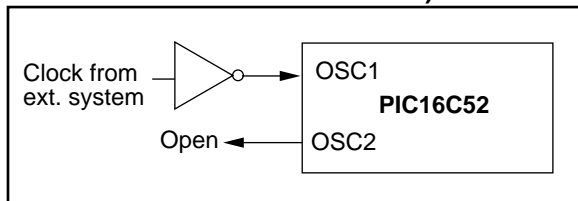


TABLE 7-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS - PIC16C52

Osc Type	Resonator Freq	Cap. Range C1	Cap. Range C2
XT	455 kHz	68-100 pF	68-100 pF
	2.0 MHz	15-33 pF	15-33 pF
	4.0 MHz	10-22 pF	10-22 pF

These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.

TABLE 7-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR - PIC16C52

Osc Type	Resonator Freq	Cap. Range C1	Cap. Range C2
XT	100 kHz	15-30 pF	200-300 pF
	200 kHz	15-30 pF	100-200 pF
	455 kHz	15-30 pF	15-100 pF
	1 MHz	15-30 pF	15-30 pF
	2 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF

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.

7.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

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

Figure 7-4 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180-degree phase shift that a parallel oscillator requires. The 4.7 k Ω resistor provides the negative feedback for stability. The 10 k Ω potentiometers bias the 74AS04 in the linear region. This circuit could be used for external oscillator designs.

FIGURE 7-4: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

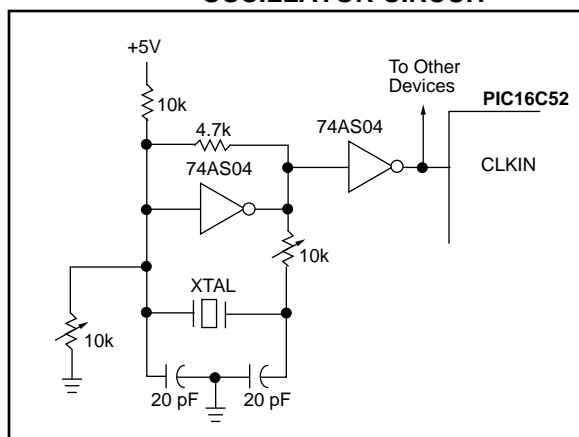
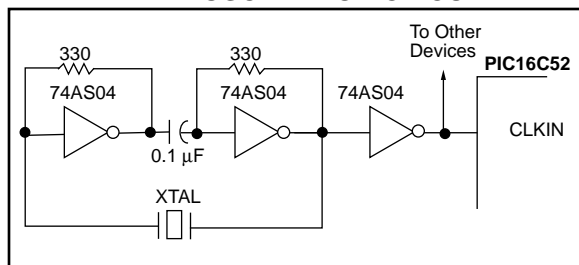


Figure 7-5 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330 Ω resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 7-5: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



7.2.4 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 (R_{ext}) and capacitor (C_{ext}) 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 will also affect the oscillation frequency, especially for low C_{ext} values. The user also needs to take into account variation due to tolerance of external R and C components used.

Figure 7-6 shows how the R/C combination is connected to the PIC16C5X. For R_{ext} values below 2.2 k Ω , the oscillator operation may become unstable, or stop completely. For very high R_{ext} values (e.g., 1 M Ω) the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend keeping R_{ext} between 3 k Ω and 100 k Ω .

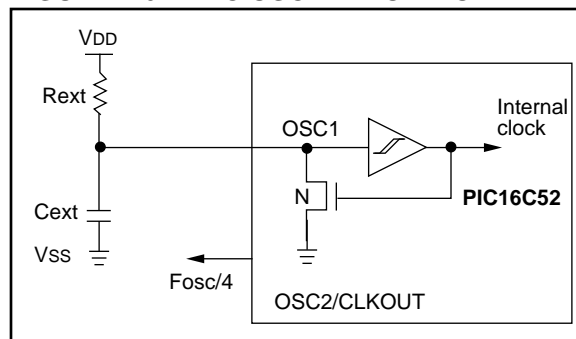
Although the oscillator will operate with no external capacitor ($C_{ext} = 0$ pF), we recommend using values above 20 pF for noise and stability reasons. With no or small external capacitance, the oscillation frequency can vary dramatically due to changes in external capacitances, such as PCB trace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

The Electrical Specifications sections show 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 will affect RC frequency more).

Also, see the Electrical Specifications sections for variation of oscillator frequency due to V_{DD} for given R_{ext}/C_{ext} values as well as frequency variation due to operating temperature for given R, C, and V_{DD} values.

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.

FIGURE 7-6: RC OSCILLATOR MODE



PIC16C52

7.3 Reset

The PIC16C52 device may be reset in one of the following ways:

- Power-On Reset (POR)
- $\overline{\text{MCLR}}$ reset (normal operation)
- $\overline{\text{MCLR}}$ wake-up reset (from SLEEP)

Table 7-3 shows these reset conditions for the PCL and STATUS registers.

Some registers are not affected in any reset condition. Their status is unknown on POR and unchanged in any other reset. Most other registers are reset to a “reset state” on Power-On Reset (POR) or $\overline{\text{MCLR}}$. A $\overline{\text{MCLR}}$ wake-up from SLEEP also results in a device reset, and not a continuation of operation before SLEEP.

The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits (STATUS <4:3>) are set or cleared depending on the different reset conditions (Section 7.6). These bits may be used to determine the nature of the reset.

Table 7-4 lists a full description of reset states of all registers. Figure 7-7 shows a simplified block diagram of the on-chip reset circuit.

TABLE 7-3: RESET CONDITIONS FOR SPECIAL REGISTERS

Condition	PCL Addr: 02h	STATUS Addr: 03h
Power-On Reset	1111 1111	0001 1xxx
$\overline{\text{MCLR}}$ reset (normal operation)	1111 1111	000u uuuu ⁽¹⁾
$\overline{\text{MCLR}}$ wake-up (from SLEEP)	1111 1111	0001 0uuu

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

Note 1: $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits retain their last value until one of the other reset conditions occur.

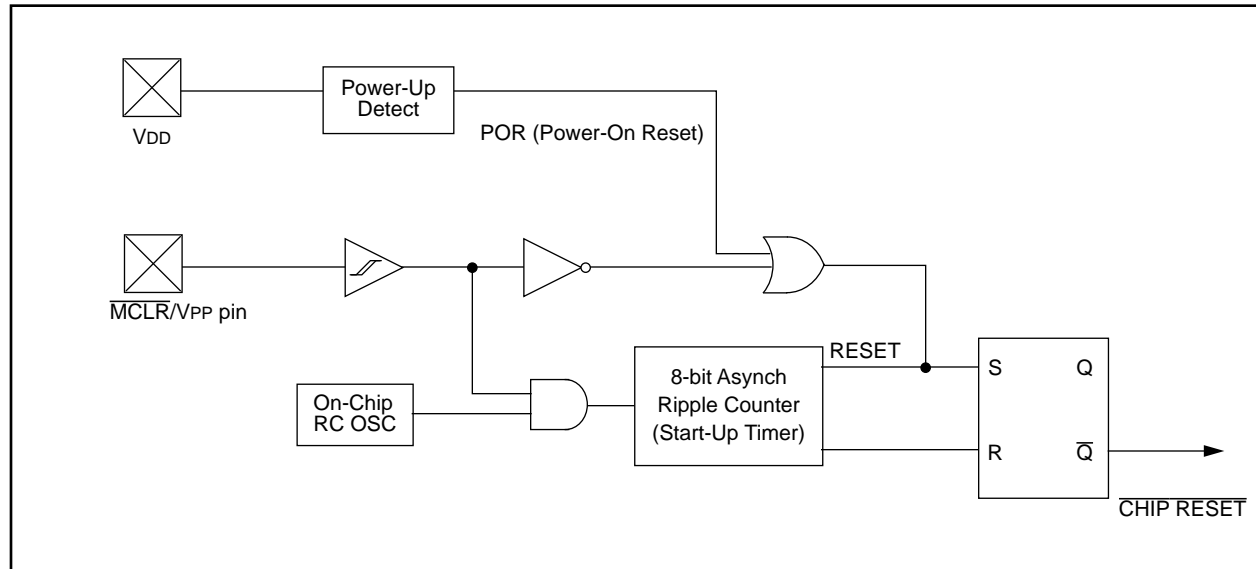
TABLE 7-4: RESET CONDITIONS FOR ALL REGISTERS

Register	Address	Power-On Reset	$\overline{\text{MCLR}}$ Reset
W	N/A	xxxx xxxx	uuuu uuuu
TRIS	N/A	1111 1111	1111 1111
OPTION	N/A	--11 1111	--11 1111
INDF	00h	xxxx xxxx	uuuu uuuu
TMR0	01h	xxxx xxxx	uuuu uuuu
PCL ⁽¹⁾	02h	1111 1111	1111 1111
STATUS ⁽¹⁾	03h	0001 1xxx	000q quuu
FSR	04h	1xxx xxxx	1uuu uuuu
PORTA	05h	---- xxxx	---- uuuu
PORTB	06h	xxxx xxxx	uuuu uuuu
General Purpose register files	08-7Fh	xxxx xxxx	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented, read as '0',
q = see tables in Section 7.6 for possible values.

Note 1: See Table 7-3 for reset value for specific conditions.

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



PIC16C52

7.4 Power-On Reset (POR)

The PIC16C5X family incorporates on-chip Power-On Reset (POR) circuitry which provides an internal chip reset for most power-up situations. To use this feature, the user merely ties the $\overline{\text{MCLR}}/\text{VPP}$ pin (Figure 7-8) to VDD . A simplified block diagram of the on-chip Power-On Reset circuit is shown in Figure 7-7.

The Power-On Reset circuit and the Device Reset Timer (Section 7.5) circuit are closely related. On power-up, the reset latch is set and the DRT is reset. The DRT timer begins counting once it detects $\overline{\text{MCLR}}$ to be high. After the time-out period, which is typically 18 ms, it will reset the reset latch and thus end the on-chip reset signal.

A power-up example where $\overline{\text{MCLR}}$ is not tied to VDD is shown in Figure 7-10. VDD is allowed to rise and stabilize before bringing $\overline{\text{MCLR}}$ high. The chip will actually come out of reset TDRT msec after $\overline{\text{MCLR}}$ goes high.

In Figure 7-11, the on-chip Power-On Reset feature is being used ($\overline{\text{MCLR}}$ and VDD are tied together). The VDD is stable before the start-up timer times out and there is no problem in getting a proper reset. However, Figure 7-12 depicts a problem situation where VDD rises too slowly. The time between when the DRT senses a high on the $\overline{\text{MCLR}}/\text{VPP}$ pin, and when the $\overline{\text{MCLR}}/\text{VPP}$ pin (and VDD) actually reach their full value, is too long. In this situation, when the start-up timer times out, VDD has not reached the $\text{VDD}(\text{min})$ value and the chip is, therefore, not guaranteed to function correctly. For such situations, we recommend that external RC circuits be used to achieve longer POR delay times (Figure 7-9).

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

For more information on PIC16C52 POR, see *Power-Up Considerations* - AN522 in the *Embedded Control Handbook*.

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

FIGURE 7-8: ELECTRICAL STRUCTURE OF $\overline{\text{MCLR}}/\text{VPP}$ PIN

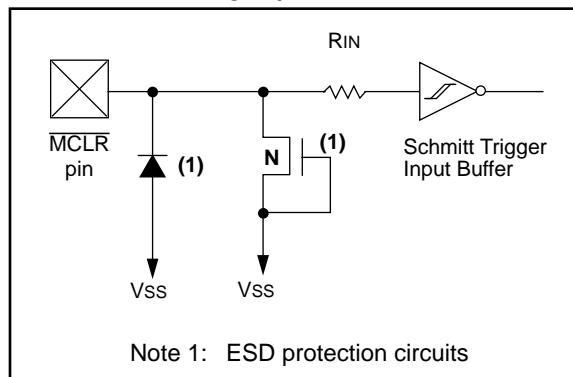


FIGURE 7-9: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)

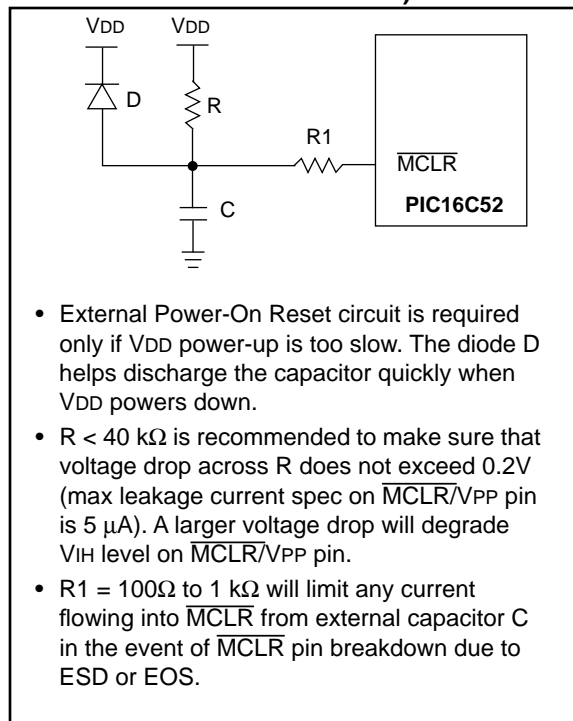


FIGURE 7-10: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD})

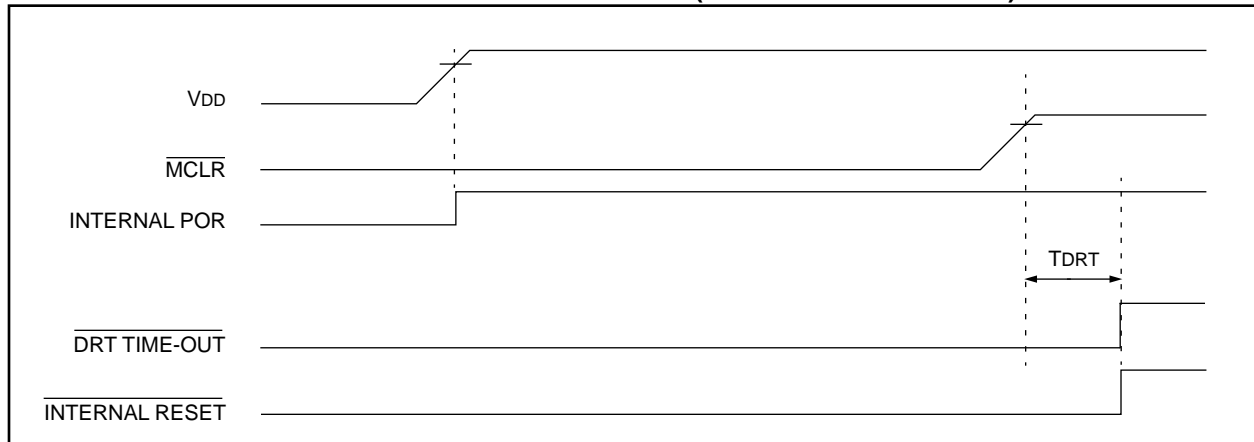


FIGURE 7-11: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO V_{DD}): FAST V_{DD} RISE TIME

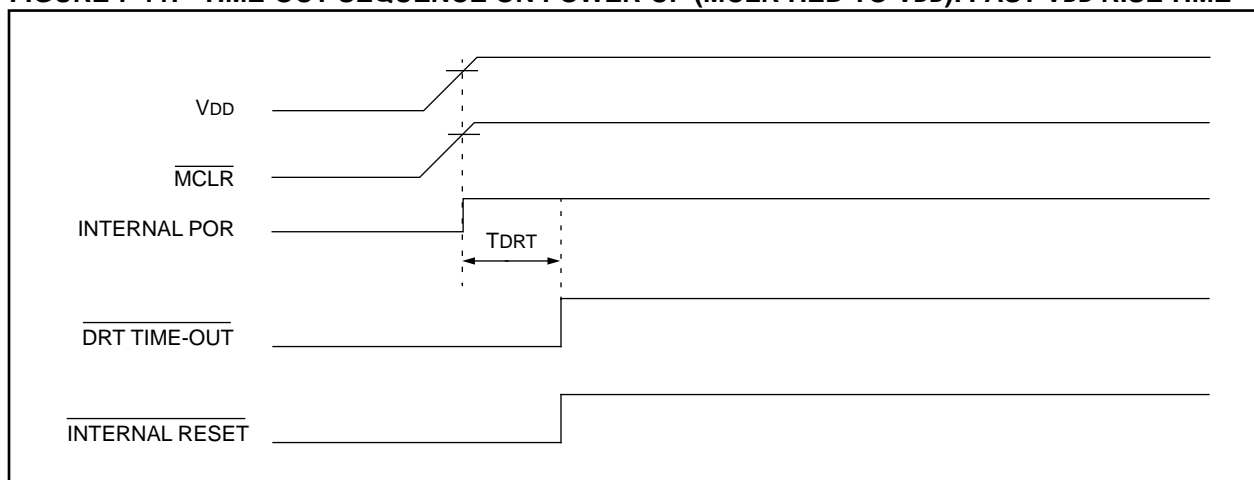
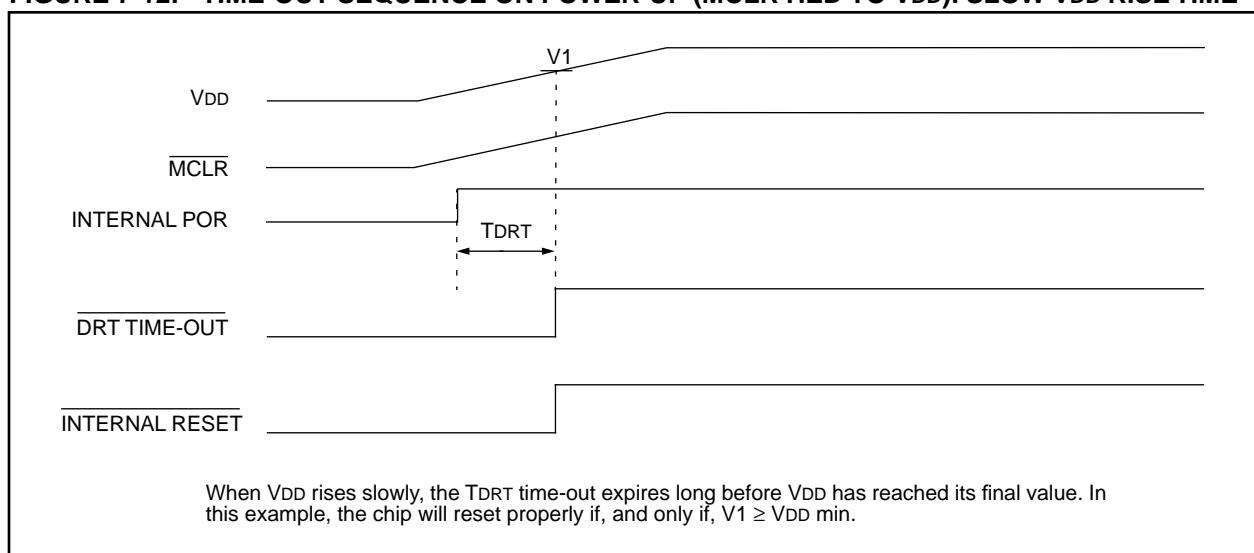


FIGURE 7-12: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO V_{DD}): SLOW V_{DD} RISE TIME



7.5 Device Reset Timer (DRT)

The Device Reset Timer (DRT) provides a fixed 18 ms nominal time-out on reset. The DRT operates on an internal RC oscillator. The processor is kept in RESET as long as the DRT is active. The DRT delay allows VDD to rise above VDD min., and for the oscillator to stabilize.

Oscillator circuits based on crystals or ceramic resonators require a certain time after power-up to establish a stable oscillation. The on-chip DRT keeps the device in a RESET for approximately 18 ms after the voltage on the MCLR/VPP pin has reached a logic high (VIHMC) level. Thus, external RC networks connected to the MCLR input are not required in most cases, allowing for savings in cost-sensitive and/or space restricted applications.

The Device Reset time delay will vary from chip to chip due to VDD, temperature, and process variation. See AC parameters for details.

7.6 Time-Out Sequence and Power Down Status Bits (\overline{TO} / \overline{PD})

The \overline{TO} and \overline{PD} bits in the STATUS register can be tested to determine if a RESET condition has been caused by a power-up condition or MCLR reset, or a MCLR wake-up reset.

TABLE 7-5: \overline{TO} / \overline{PD} STATUS AFTER RESET

\overline{TO}	\overline{PD}	RESET was caused by
1	1	Power-up (POR)
u	u	MCLR reset (normal operation) ⁽¹⁾
1	0	MCLR wake-up reset (from SLEEP)

Legend: u = unchanged

Note 1: The \overline{TO} and \overline{PD} bits maintain their status (u) until a reset occurs. A low-pulse on the MCLR input does not change the \overline{TO} and \overline{PD} status bits.

These STATUS bits are only affected by events listed in Table 7-6.

TABLE 7-6: EVENTS AFFECTING \overline{TO} / \overline{PD} STATUS BITS

Event	\overline{TO}	\overline{PD}	Remarks
Power-up	1	1	
SLEEP instruction	1	0	

Legend: u = unchanged

A SLEEP instruction will be executed, regardless of the status of the \overline{PD} bit. Table 7-5 reflects the status of \overline{TO} and \overline{PD} after the corresponding event.

Table 7-3 lists the reset conditions for the special function registers, while Table 7-4 lists the reset conditions for all the registers.

7.7 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 PIC16C52 devices when a brown-out occurs, external brown-out protection circuits may be built (Figure 7-13 and Figure 7-14).

FIGURE 7-13: BROWN-OUT PROTECTION CIRCUIT 1

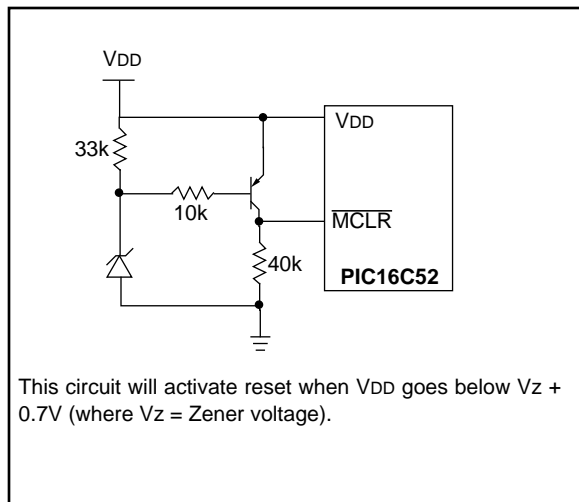
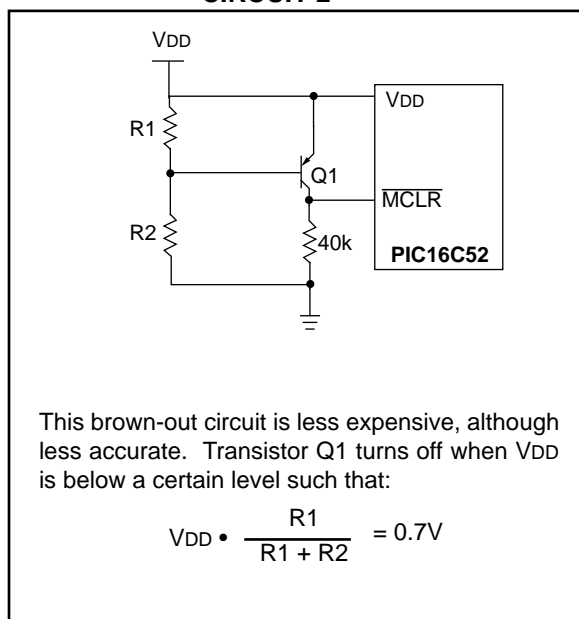


FIGURE 7-14: BROWN-OUT PROTECTION CIRCUIT 2



7.8 Power-Down Mode (SLEEP)

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

7.8.1 SLEEP

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

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

For lowest current consumption while powered down, the T0CKI input should be at VDD or VSS and the MCLR/VPP pin must be at a logic high level (VIHMC).

7.8.2 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through an external reset input on MCLR/VPP pin. The \overline{PD} bit, which is set on power-up, is cleared when SLEEP is invoked.

7.9 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.

7.10 ID Locations

Four memory locations are designated as ID locations where the user can store checksum or other code-identification numbers. These locations are not accessible during normal execution but are readable and writable during program/verify.

Use only the lower 4 bits of the ID locations and always program the upper 8 bits as '1's.

Note: Microchip will assign a unique pattern number for QTP and SQTP requests. This pattern number will be unique and traceable to the submitted code.

PIC16C52

NOTES:

8.0 INSTRUCTION SET SUMMARY

Each PIC16C5X instruction is a 12-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 PIC16C5X instruction set summary in Table 8-2 groups the instructions into byte-oriented, bit-oriented, and literal and control operations. Table 8-1 shows the opcode field descriptions.

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator is used to specify which one of the 32 file registers 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 in the instruction.

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

For **literal and control** operations, 'k' represents an 8 or 9-bit constant or literal value.

TABLE 8-1: 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
TO	Time-Out bit
PD	Power-Down bit
dest	Destination, either the W register or the specified register file location
[]	Options
()	Contents
→	Assigned to
< >	Register bit field
∈	In the set of
<i>italics</i>	User defined term (font is courier)

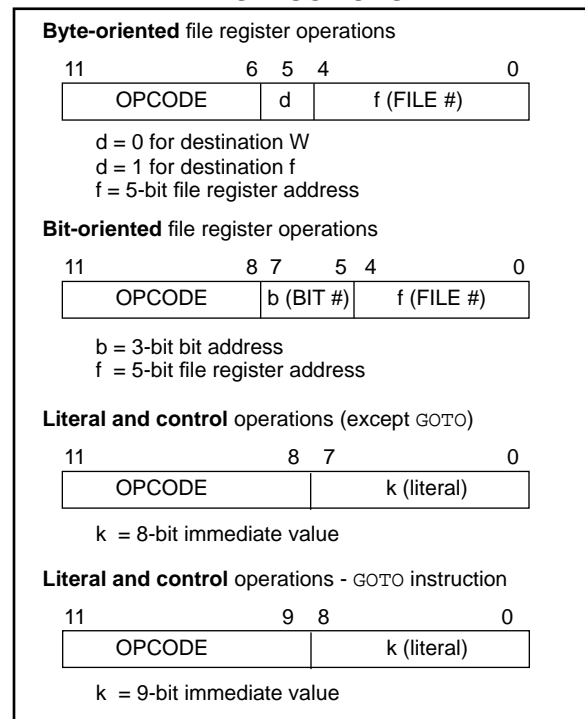
All instructions are executed within one single instruction cycle, unless a conditional test is true or the program counter is changed as a result of an instruction. In this case, the execution takes two instruction cycles. One instruction cycle consists of four oscillator periods. Thus, for an oscillator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

Figure 8-1 shows the three general formats that the instructions can have. All examples in the figure use the following format to represent a hexadecimal number:

0xhhh

where 'h' signifies a hexadecimal digit.

FIGURE 8-1: GENERAL FORMAT FOR INSTRUCTIONS



PIC16C52

TABLE 8-2: INSTRUCTION SET SUMMARY

Mnemonic, Operands		Description	Cycles	12-Bit Opcode			Status Affected	Notes
				MSb	LSb			
ADDWF	f, d	Add W and f	1	0001	11df	ffff	C, DC, Z	1, 2, 4
ANDWF	f, d	AND W with f	1	0001	01df	ffff	Z	2, 4
CLRF	f	Clear f	1	0000	011f	ffff	Z	4
CLRW	—	Clear W	1	0000	0100	0000	Z	
COMF	f, d	Complement f	1	0010	01df	ffff	Z	
DECF	f, d	Decrement f	1	0000	11df	ffff	Z	2, 4
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	0010	11df	ffff	None	2, 4
INCF	f, d	Increment f	1	0010	10df	ffff	Z	2, 4
INCFSZ	f, d	Increment f, Skip if 0	1(2)	0011	11df	ffff	None	2, 4
IORWF	f, d	Inclusive OR W with f	1	0001	00df	ffff	Z	2, 4
MOVF	f, d	Move f	1	0010	00df	ffff	Z	2, 4
MOVWF	f	Move W to f	1	0000	001f	ffff	None	1, 4
NOP	—	No Operation	1	0000	0000	0000	None	
RLF	f, d	Rotate left f through Carry	1	0011	01df	ffff	C	2, 4
RRF	f, d	Rotate right f through Carry	1	0011	00df	ffff	C	2, 4
SUBWF	f, d	Subtract W from f	1	0000	10df	ffff	C, DC, Z	1, 2, 4
SWAPF	f, d	Swap f	1	0011	10df	ffff	None	2, 4
XORWF	f, d	Exclusive OR W with f	1	0001	10df	ffff	Z	2, 4
BIT-ORIENTED FILE REGISTER OPERATIONS								
BCF	f, b	Bit Clear f	1	0100	bbbdf	ffff	None	2, 4
BSF	f, b	Bit Set f	1	0101	bbbdf	ffff	None	2, 4
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	0110	bbbdf	ffff	None	
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	0111	bbbdf	ffff	None	
LITERAL AND CONTROL OPERATIONS								
ANDLW	k	AND literal with W	1	1110	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	1001	kkkk	kkkk	None	1
CLRWDT	k	Clear Watchdog Timer	1	0000	0000	0100	\overline{TO} , \overline{PD}	5
GOTO	k	Unconditional branch	2	101k	kkkk	kkkk	None	
IORLW	k	Inclusive OR Literal with W	1	1101	kkkk	kkkk	Z	
MOVLW	k	Move Literal to W	1	1100	kkkk	kkkk	None	
OPTION	k	Load OPTION register	1	0000	0000	0010	None	
RETLW	k	Return, place Literal in W	2	1000	kkkk	kkkk	None	
SLEEP	—	Go into standby mode	1	0000	0000	0011	\overline{TO} , \overline{PD}	
TRIS	f	Load TRIS register	1	0000	0000	0fff	None	3
XORLW	k	Exclusive OR Literal to W	1	1111	kkkk	kkkk	Z	

Note 1: The 9th bit of the program counter will be forced to a '0' by any instruction that writes to the PC except for **GOTO**. (Section 4.5)

- When an I/O register is modified as a function of itself (e.g. **MOVF** `PORTB`, 1), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
- The instruction **TRIS** f, where f = 5, 6, or 7 causes the contents of the W register to be written to the tristate latches of `PORTA`, B or C, respectively. A '1' forces the pin to a hi-impedance state and disables the output buffers.
- If this instruction is executed on the `TMR0` register (and, where applicable, d = 1), the prescaler will be cleared (if assigned to `TMR0`).
- Do not use in PIC16C52 code.

ADDWF Add W and f

Syntax: [*label*] ADDWF f,d

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

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

Status Affected: C, DC, Z

Encoding:

0001	11df	ffff
------	------	------

Description: Add the contents of the W register and 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: $0 \leq f \leq 31$
 $d \in [0,1]$

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

Status Affected: Z

Encoding:

0001	01df	ffff
------	------	------

Description: The contents of the W register are AND'ed 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

ANDLW And literal with W

Syntax: [*label*] ANDLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{AND}. (k) \rightarrow (W)$

Status Affected: Z

Encoding:

1110	kkkk	kkkk
------	------	------

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

Words: 1

Cycles: 1

Example: ANDLW 0x5F

Before Instruction

W = 0xA3

After Instruction

W = 0x03

BCF Bit Clear f

Syntax: [*label*] BCF f,b

Operands: $0 \leq f \leq 31$
 $0 \leq b \leq 7$

Operation: $0 \rightarrow (f)$

Status Affected: None

Encoding:

0100	bbbf	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

PIC16C52

BSF Bit Set f

Syntax: [*label*] BSF f,b

Operands: $0 \leq f \leq 31$
 $0 \leq b \leq 7$

Operation: $1 \rightarrow (f)$

Status Affected: None

Encoding:

0101	bbbbf	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: $0 \leq f \leq 31$
 $0 \leq b \leq 7$

Operation: skip if $(f) = 0$

Status Affected: None

Encoding:

0110	bbbbf	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 an 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: $0 \leq f \leq 31$
 $0 \leq b < 7$

Operation: skip if $(f) = 1$

Status Affected: None

Encoding:

0111	bbbbf	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 an NOP is executed instead, making this a 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example: HERE BTFSS FLAG, 1
 FALSE GOTO PROCESS_CODE
 TRUE •
 •
 •

Before Instruction
 PC = address (HERE)

After Instruction
 If FLAG<1> = 0,
 PC = address (FALSE);
 if FLAG<1> = 1,
 PC = address (TRUE)

CALL	Subroutine Call			
Syntax:	[<i>label</i>] CALL k			
Operands:	$0 \leq k \leq 255$			
Operation:	(PC) + 1 → Top of Stack; k → PC<7:0>; (STATUS<6:5>) → PC<10:9>; 0 → PC<8>			
Status Affected:	None			
Encoding:	<table border="1"><tr><td>1001</td><td>kkkk</td><td>kkkk</td></tr></table>	1001	kkkk	kkkk
1001	kkkk	kkkk		
Description:	Subroutine call. First, return address (PC+1) is pushed onto the stack. The eight bit immediate address is loaded into PC bits <7:0>. The upper bits PC<10:9> are loaded from STATUS<6:5>, PC<8> is cleared. CALL is a two cycle instruction.			
Words:	1			
Cycles:	2			
Example:	HERE CALL THERE			
Before Instruction				
PC = address (HERE)				
After Instruction				
PC = address (THERE)				
TOS = address (HERE + 1)				

CLRF	Clear f			
Syntax:	[<i>label</i>] CLRF f			
Operands:	$0 \leq f \leq 31$			
Operation:	00h \rightarrow (f); 1 \rightarrow Z			
Status Affected:	Z			
Encoding:	<table border="1"><tr><td>0000</td><td>011f</td><td>ffff</td></tr></table>	0000	011f	ffff
0000	011f	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				

CLR W				
Syntax:	[<i>label</i>] CLRW			
Operands:	None			
Operation:	00h → (W); 1 → Z			
Status Affected:	Z			
Encoding:	<table><tr><td>0000</td><td>0100</td><td>0000</td></tr></table>	0000	0100	0000
0000	0100	0000		
Description:	The W register is cleared. Zero bit (Z) is set.			
Words:	1			
Cycles:	1			
Example:	CLR W			
Before Instruction				
W	= 0x5A			
After Instruction				
W	= 0x00			
Z	= 1			

CLRWDT		Clear Watchdog Timer			
Syntax:	[<i>label</i>] CLRWDT				
Operands:	None				
Operation:	1 → \overline{TO} ; 1 → \overline{PD}				
Status Affected:	\overline{TO} , \overline{PD}				
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0100</td></tr></table>		0000	0000	0100
0000	0000	0100			
Description:	Since WDT is not available on the PIC16C52, the CLRWDT instruction will execute as a NOP. Status bits \overline{TO} and \overline{PD} are set.				
Words:	1				
Cycles:	1				
Example:	CLRWDT				
After Instruction					
\overline{TO}	=	1			
\overline{PD}	=	1			
Do not use in PIC16C52 code.					

PIC16C52

COMF Complement f

Syntax: [*label*] COMF f,d

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

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

Status Affected: Z

Encoding:

0010	01df	ffff
------	------	------

Description: The contents of register 'f' are complemented. 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: COMF REG1, 0

Before Instruction

REG1 = 0x13

After Instruction

REG1 = 0x13

W = 0xEC

DECf Decrement f

Syntax: [*label*] DECf f,d

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

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

Status Affected: Z

Encoding:

0000	11df	ffff
------	------	------

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

Words: 1

Cycles: 1

Example: DECf CNT, 1

Before Instruction

CNT = 0x01

Z = 0

After Instruction

CNT = 0x00

Z = 1

DECFSZ Decrement f, Skip if 0

Syntax: [*label*] DECFSZ f,d

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

Operation: $(f) - 1 \rightarrow d$; skip if result = 0

Status Affected: None

Encoding:

0010	11df	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 and an 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 \neq 0,

PC = address (HERE+1)

GOTO Unconditional Branch

Syntax: [*label*] GOTO k

Operands: $0 \leq k \leq 511$

Operation: $k \rightarrow \text{PC}<8:0>;$
 $\text{STATUS}<6:5> \rightarrow \text{PC}<10:9>$

Status Affected: None

Encoding:

101k	kkkk	kkkk
------	------	------

Description: GOTO is an unconditional branch. The 9-bit immediate value is loaded into PC bits <8:0>. The upper bits of PC are loaded from STATUS<6:5>. GOTO is a two cycle instruction.

Words: 1

Cycles: 2

Example: GOTO THERE

After Instruction

PC = address (THERE)

INCF		Increment f				
Syntax:	[<i>label</i>] INCF f,d					
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$					
Operation:	$(f) + 1 \rightarrow (\text{dest})$					
Status Affected:	Z					
Encoding:	<table border="1"><tr><td>0010</td><td>10df</td><td>ffff</td></tr></table>			0010	10df	ffff
0010	10df	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:	[<i>label</i>] INCFSZ f,d					
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$					
Operation:	$(f) + 1 \rightarrow (\text{dest})$, skip if result = 0					
Status Affected:	None					
Encoding:	<table border="1"><tr><td>0011</td><td>11df</td><td>ffff</td></tr></table>			0011	11df	ffff
0011	11df	ffff				
Description:	<p>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'.</p> <p>If the result is 0, then the next instruction, which is already fetched, is discarded and an NOP is executed instead making it a two cycle instruction.</p>					
Words:	1					
Cycles:	1(2)					
Example:	HERE	INCFSZ	CNT, 1			

Before Instruction

PC = address (HERE)

After Instruction

CNT = CNT + 1;
if CNT = 0,
PC = address (CONTINUE);
if CNT \neq 0,
PC = address (HERE + 1)

IORLW	Inclusive OR literal with W			
Syntax:	[<i>label</i>] IORLW k			
Operands:	$0 \leq k \leq 255$			
Operation:	(W) .OR. (k) \rightarrow (W)			
Status Affected:	Z			
Encoding:	<table border="1"><tr><td>1101</td><td>kkkk</td><td>kkkk</td></tr></table>	1101	kkkk	kkkk
1101	kkkk	kkkk		
Description:	The contents of the W register are OR'ed with the eight bit literal 'k'. The result is placed in the W register.			
Words:	1			
Cycles:	1			
Example:	IORLW 0x35			

Before Instruction

W = 0x9A

After Instruction

W = 0xBF
Z = 0

IORWF	Inclusive OR W with f			
Syntax:	[<i>label</i>] IORWF f,d			
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$			
Operation:	(W).OR. (f) \rightarrow (dest)			
Status Affected:	Z			
Encoding:	<table border="1"><tr><td>0001</td><td>00df</td><td>ffff</td></tr></table>	0001	00df	ffff
0001	00df	ffff		
Description:	Inclusive OR the W register with register 'f'. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.			
Words:	1			
Cycles:	1			
Example:	IORWF RESULT, 0			

Before Instruction

RESULT = 0x13
W = 0x91

After Instruction

RESULT = 0x13
W = 0x93
Z = 0

PIC16C52

MOVF	Move f			
Syntax:	[<i>label</i>] MOVF f,d			
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$			
Operation:	$(f) \rightarrow (dest)$			
Status Affected:	Z			
Encoding:	<table border="1"><tr><td>0010</td><td>00df</td><td>ffff</td></tr></table>	0010	00df	ffff
0010	00df	ffff		
Description:	The contents of register 'f' is moved to destination 'd'. If 'd' is 0, destination is the W register. If 'd' is 1, the destination is file register 'f'. 'd' is 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			

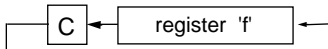
MOVLW	Move Literal to W			
Syntax:	[<i>label</i>] MOVLW k			
Operands:	$0 \leq k \leq 255$			
Operation:	$k \rightarrow (W)$			
Status Affected:	None			
Encoding:	<table border="1"><tr><td>1100</td><td>kkkk</td><td>kkkk</td></tr></table>	1100	kkkk	kkkk
1100	kkkk	kkkk		
Description:	The eight bit literal 'k' is loaded into the W register. The don't cares will assemble as 0s.			
Words:	1			
Cycles:	1			
Example:	MOVLW 0x5A			
After Instruction				
W = 0x5A				

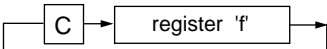
MOVWF	Move W to f			
Syntax:	[<i>label</i>] MOVWF f			
Operands:	$0 \leq f \leq 31$			
Operation:	$(W) \rightarrow (f)$			
Status Affected:	None			
Encoding:	<table border="1"><tr><td>0000</td><td>001f</td><td>ffff</td></tr></table>	0000	001f	ffff
0000	001f	ffff		
Description:	Move data from the W register to register 'f'.			
Words:	1			
Cycles:	1			
Example:	MOVWF TEMP_REG			
Before Instruction				
TEMP_REG	= 0xFF			
W	= 0x4F			
After Instruction				
TEMP_REG	= 0x4F			
W	= 0x4F			

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

OPTION		Load OPTION Register						
Syntax:	[<i>label</i>] OPTION							
Operands:	None							
Operation:	(W) → OPTION							
Status Affected:	None							
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0010</td></tr></table>					0000	0000	0010
0000	0000	0010						
Description:	The content of the W register is loaded into the OPTION register.							
Words:	1							
Cycles:	1							
Example	OPTION							
Before Instruction								
W		=	0x07					
After Instruction								
OPTION		=	0x07					

RETLW		Return with Literal in W				
Syntax:	[<i>label</i>] RETLW k					
Operands:	0 ≤ k ≤ 255					
Operation:	k → (W); TOS → PC					
Status Affected:	None					
Encoding:	<table border="1"><tr><td>1000</td><td>kkkk</td><td>kkkk</td></tr></table>			1000	kkkk	kkkk
1000	kkkk	kkkk				
Description:	The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two cycle instruction.					
Words:	1					
Cycles:	2					
Example:	<pre>CALL TABLE ;W contains ;table offset ;value. ;W now has table • ;value. • • TABLE ADDWF PC ;W = offset RETLW k1 ;Begin table RETLW k2 ; • • • RETLW kn ; End of table</pre>					
Before Instruction						
W = 0x07						
After Instruction						
W = value of k8						

RLF		Rotate Left f through Carry				
Syntax:	[<i>label</i>] RLF f,d					
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$					
Operation:	See description below					
Status Affected:	C					
Encoding:	<table border="1"><tr><td>0011</td><td>01df</td><td>ffff</td></tr></table>			0011	01df	ffff
0011	01df	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					

RRF		Rotate Right f through Carry			
Syntax:	[<i>label</i>] RRF f,d				
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$				
Operation:	See description below				
Status Affected:	C				
Encoding:	<table border="1"><tr><td>0011</td><td>00df</td><td>ffff</td></tr></table>		0011	00df	ffff
0011	00df	ffff			
Description:	<p>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'.</p> 				
Words:	1				
Cycles:	1				
Example:	RRF REG1,0				
Before Instruction					
REG1	=	1110 0110			
C	=	0			
After Instruction					
REG1	=	1110 0110			
W	=	0111 0011			
C	=	0			

PIC16C52

SLEEP Enter SLEEP Mode

Syntax:	[<i>label</i>] SLEEP			
Operands:	None			
Operation:	$1 \rightarrow \overline{TO}$; $0 \rightarrow \overline{PD}$			
Status Affected:	\overline{TO} , \overline{PD}			
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0011</td></tr></table>	0000	0000	0011
0000	0000	0011		
Description:	Time-out status bit (\overline{TO}) is set. The power down status bit (\overline{PD}) is cleared. The processor is put into SLEEP mode with the oscillator stopped. See section on SLEEP for more details.			
Words:	1			
Cycles:	1			
Example:	SLEEP			

SUBWF Subtract W from f

Syntax:	[label] SUBWF f,d			
Operands:	$0 \leq f \leq 31$ $d \in [0,1]$			
Operation:	$(f) - (W) \rightarrow (\text{dest})$			
Status Affected:	C, DC, Z			
Encoding:	<table border="1"><tr><td>0000</td><td>10df</td><td>ffff</td></tr></table>	0000	10df	ffff
0000	10df	ffff		
Description:	Subtract (2's complement method) the 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 Nibbles in f

Syntax: `[label] SWAPF f,d`

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

Operation: $(f<3:0>) \rightarrow (dest<7:4>);$
 $(f<7:4>) \rightarrow (dest<3:0>)$

Status Affected: None

Encoding:

0011	10df	ffff
------	------	------

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0 the result is placed in W register. If 'd' is 1 the result is placed in register 'f'.

Words: 1

Cycles: 1

Example `SWAPF REG1, 0`

Before Instruction
`REG1 = 0xA5`

After Instruction
`REG1 = 0xA5`
`W = 0x5A`

TRIS Load TRIS Register

Syntax: `[label] TRIS f`

Operands: $f = 5, 6 \text{ or } 7$

Operation: $(W) \rightarrow \text{TRIS register } f$

Status Affected: None

Encoding:

0000	0000	0fff
------	------	------

Description: TRIS register 'f' ($f = 5, 6, \text{ or } 7$) is loaded with the contents of the W register

Words: 1

Cycles: 1

Example `TRIS PORTA`

Before Instruction
`W = 0xA5`

After Instruction
`TRISA = 0xA5`

XORLW Exclusive OR literal with W

Syntax: `[label] XORLW k`

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{XOR. } k \rightarrow (W)$

Status Affected: Z

Encoding:

1111	kkkk	kkkk
------	------	------

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

Words: 1

Cycles: 1

Example: `XORLW 0xAF`

Before Instruction
`W = 0xB5`

After Instruction
`W = 0x1A`

XORWF Exclusive OR W with f

Syntax: `[label] XORWF f,d`

Operands: $0 \leq f \leq 31$
 $d \in [0,1]$

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

Status Affected: Z

Encoding:

0001	10df	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`

PIC16C52

NOTES:

9.0 DEVELOPMENT SUPPORT

9.1 Development Tools

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

- PICMASTER™ Real-Time In-Circuit Emulator
- PRO MATE™ Universal Programmer
- PICSTART™ Low-Cost Prototype Programmer
- PICDEM-1 Low-Cost Demonstration Board
- PICDEM-2 Low-Cost Demonstration Board
- MPASM Assembler
- MPSIM Software Simulator
- C Compiler (MP-C)
- Fuzzy logic development system (*fuzzyTECH*®-MP)

9.2 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 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. A PICMASTER System configuration is shown in Figure 9-1.

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 PIC16C5X, PIC16CXX and PIC17CXX 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 better) machine platform and Microsoft® Windows® 3.x environment was chosen to best make these features available to you, the end user.

The PICMASTER Universal Emulator System consists primarily of four major components:

- Host-Interface Card
- Emulator Control Pod
- Target-Specific Emulator Probe
- PC-Host Emulation Control Software

The Windows operating system allows the developer to take full advantage of the many powerful features and functions of the PICMASTER system.

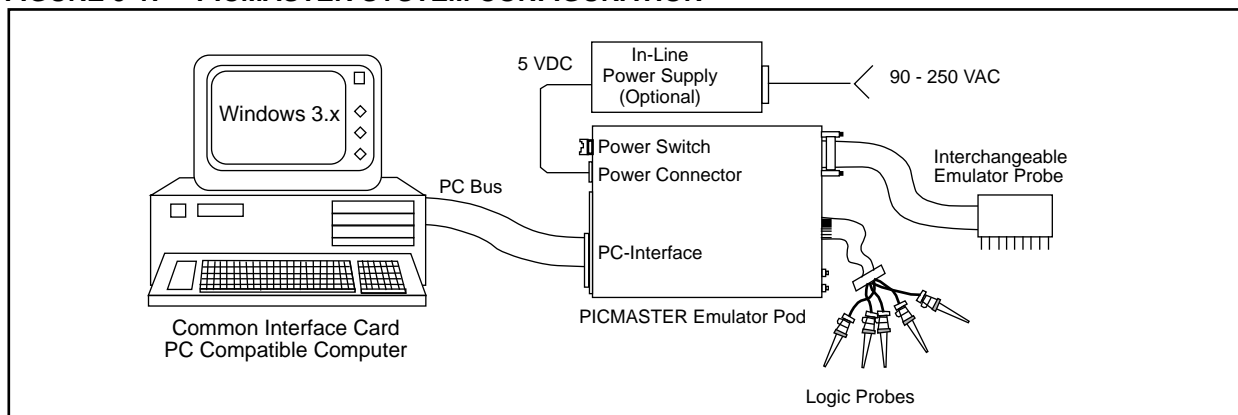
PICMASTER emulation can operate in one window, while a text editor is running in a second window.

PC-Host Emulation Control software takes full advantage of Dynamic Data Exchange (DDE), a feature of Windows. DDE allows data to be dynamically transferred between two or more Windows programs. With this feature, data collected with PICMASTER can be automatically transferred to a spreadsheet or database program for further analysis.

Under Windows, as many as four PICMASTER emulators can be run simultaneously from the same PC making development of multi-microcontroller systems possible (e.g., a system containing a PIC16CXX processor and a PIC17CXX processor).

The PICMASTER probes specifications are shown in Table 9-1.

FIGURE 9-1: PICMASTER SYSTEM CONFIGURATION



PIC16C52

TABLE 9-1: PICMASTER PROBE SPECIFICATION

Devices	PICMASTER PROBE	PROBE	
		Maximum Frequency	Operating Voltage
PIC16C52	PROBE-16D	4 MHz	4.5V - 5.5V
PIC16C54	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16C54A	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR54	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR54A	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16CR54B	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16C55	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR55	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16C56	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR56	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16C57	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR57A	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR57B	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16C58A	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR58A	PROBE-16D	20 MHz	4.5V - 5.5V
PIC16CR58B	PROBE-16D ⁽¹⁾	20 MHz	4.5V - 5.5V
PIC16C61	PROBE-16G	10 MHz	4.5V - 5.5V
PIC16C62	PROBE-16E	10 MHz	4.5V - 5.5V
PIC16C62A	PROBE-16E ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16CR62	PROBE-16E ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C63	PROBE-16F ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C64	PROBE-16E	10 MHz	4.5V - 5.5V
PIC16C64A	PROBE-16E ⁽¹⁾	10 MHz	4.5V - 5.5V

TABLE 9-1: PICMASTER PROBE SPECIFICATION (CON'T)

Devices	PICMASTER PROBE	PROBE	
		Maximum Frequency	Operating Voltage
PIC16CR64	PROBE-16E ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C65	PROBE-16F	10 MHz	4.5V - 5.5V
PIC16C65A	PROBE-16F ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C620	PROBE-16H	10 MHz	4.5V - 5.5V
PIC16C621	PROBE-16H	10 MHz	4.5V - 5.5V
PIC16C622	PROBE-16H	10 MHz	4.5V - 5.5V
PIC16C70	PROBE-16B ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C71	PROBE-16B	10 MHz	4.5V - 5.5V
PIC16C71A	PROBE-16B ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C72	PROBE-16F ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C73	PROBE-16F	10 MHz	4.5V - 5.5V
PIC16C73A	PROBE-16F ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C74	PROBE-16F	10 MHz	4.5V - 5.5V
PIC16C74A	PROBE-16F ⁽¹⁾	10 MHz	4.5V - 5.5V
PIC16C83	PROBE-16C	10 MHz	4.5V - 5.5V
PIC16C84	PROBE-16C	10 MHz	4.5V - 5.5V
PIC17C42	PROBE-17B	20 MHz	4.5V - 5.5V
PIC17C43	PROBE-17B	20 MHz	4.5V - 5.5V
PIC17C44	PROBE-17B	20 MHz	4.5V - 5.5V

Note 1: This PICMASTER probe can be used to functionally emulate the device listed in the previous column. Contact your Microchip sales office for details.

9.3 PRO MATE Universal Programmer

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

The PRO MATE has programmable VDD and VPP supplies which allows it to verify programmed memory at VDD min and VDD max for maximum reliability. It has an LCD display for 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 can read, verify or program PIC16C5X, PIC16CXX and PIC17CXX devices. It can also set configuration and code-protect bits in this mode.

In PC-hosted mode, the PRO MATE connects to the PC via one of the COM (RS-232) ports. PC based user-interface software makes using the programmer simple and efficient. The user interface is full-screen and menu-based. Full screen display and editing of data, easy selection of bit configuration and part type, easy selection of VDD min, VDD max and VPP levels, load and store to and from disk files (Intel® hex format) are some of the features of the software. Essential commands such as read, verify, program and blank check can be issued from the screen. Additionally, serial programming support is possible where each part is programmed with a different serial number, sequential or random.

The PRO MATE has a modular "programming socket module". Different socket modules are required for different processor types and/or package types.

PRO MATE supports all PIC16C5X, PIC16CXX and PIC17CXX processors.

9.4 PICSTART Low-Cost Development System

The PICSTART programmer is an easy to use, very low-cost prototype programmer. It connects to the PC via one of the COM (RS-232) ports. A PC-based user interface software makes using the programmer simple and efficient. The user interface is full-screen and menu-based. PICSTART is not recommended for production programming.

9.5 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 (PIC16C52 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 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.

9.6 PICDEM-2 Low-Cost PIC16CXX Demonstration Board

The PICDEM-2 is a simple demonstration board that supports the 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 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 I²C bus and separate headers for connection to an LCD module and a keypad.

9.7 MPLAB Integrated Development Environment Software

The MPLAB 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 (available soon)
- 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 (available soon) 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.

9.8 MPASM Assembler

The MPASM Cross 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.

- **Data Directives** are those that control the allocation of memory and provide a way to refer to data items symbolically (i.e., by meaningful names).
- **Control Directives** control the MPASM listing display. They allow the specification of titles and sub-titles, page ejects and other listing control. This eases the readability of the printed output file.
- **Conditional Directives** permit sections of conditionally assembled code. This is most useful where additional functionality may wished to be added depending on the product (less functionality for the low end product, then for the high end product). Also this is very helpful in the debugging of a program.
- **Macro Directives** control the execution and data allocation within macro body definitions. This makes very simple the re-use of functions in a program as well as between programs.

9.9 MPSIM Software Simulator

The MPSIM 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. MPSIM fully supports symbolic debugging using MP-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.

9.10 MP-C C Compiler

The MP-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 PICMASTER Universal Emulator memory display (PICMASTER emulator software versions 1.13 and later).

The MP-C Code Development System is supplied directly by Byte Craft Limited of Waterloo, Ontario, Canada. If you have any questions, please contact your regional Microchip FAE or Microchip technical support personnel at (602) 786-7627.

9.11 fuzzyTECH-MP Fuzzy Logic Development System

fuzzyTECH-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, *fuzzyTECH-MP*, edition for implementing more complex systems.

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

9.12 Development Systems

For convenience, the development tools are packaged into comprehensive systems as listed in Table 9-2.

TABLE 9-2: DEVELOPMENT SYSTEM PACKAGES

Item	Name	System Description
1.	PICMASTER System	PICMASTER In-Circuit Emulator, PRO MATE Programmer, Assembler, Software Simulator, Samples and your choice of Target Probe.
2.	PICSTART System	PICSTART Low-Cost Prototype Programmer, Assembler, Software Simulator and Samples.
3.	PRO MATE System	PRO MATE Universal Programmer, full featured stand-alone or PC-hosted programmer, Assembler, Simulator

PIC16C52

NOTES:

10.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings†

Ambient Temperature under bias	–55°C to +125°C
Storage Temperature	–65°C to +150°C
Voltage on VDD with respect to VSS	0 V to +7.5 V
Voltage on $\overline{\text{MCLR}}$ with respect to VSS ⁽²⁾	0 V to +14 V
Voltage on all other pins with respect to VSS	–0.6 V to (VDD + 0.6 V)
Total Power Dissipation ⁽¹⁾	800 mW
Max. Current out of VSS pin	150 mA
Max. Current into VDD pin	50 mA
Max. Current into an input pin (T0CKI only)	±500 µA
Input Clamp Current, I _{IK} (V _I < 0 or V _I > VDD)	±20 mA
Output Clamp Current, I _{OK} (V _O < 0 or V _O > VDD)	±20 mA
Max. Output Current sunk by any I/O pin	10 mA
Max. Output Current sourced by any I/O pin	10 mA
Max. Output Current sourced by a single I/O port (PORTA, B or C)	10 mA
Max. Output Current sunk by a single I/O port (PORTA, B or C)	10 mA

Note 1: Power Dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$

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

†NOTICE: Stresses above those listed under “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.

PIC16C52

10.1 DC Characteristics: PIC16C52-04 (Commercial) PIC16C52-I04 (Industrial)

DC Characteristics Power Supply Pins		Standard Operating Conditions (unless otherwise specified)				
		Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial) $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial)				
Characteristic	Sym	Min	Typ ⁽¹⁾	Max	Units	Conditions
Supply Voltage	VDD	3.0		6.25	V	FOSC = DC to 4 MHz
RAM Data Retention Voltage ⁽²⁾	VDR		1.5*		V	Device in SLEEP Mode
Supply Current ^(3,4)	IDD		1.8	3.3	mA	FOSC = 4 MHz, VDD = 5.5 V
Power Down Current ⁽⁵⁾	IPD					
Commercial			0.6	9*	μA	VDD = 3.0 V
Industrial			0.6	12*	μA	VDD = 3.0 V

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is based on characterization results at 25°C. This data is for design guidance only and is not tested.

- 2: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.
- 3: The supply current is mainly a function of the operating voltage and frequency. Other factors such as bus loading, oscillator type, bus rate, internal code execution pattern, and temperature also have an impact on the current consumption.
 - a) 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 VSS, T0CKI = VDD, MCLR = VDD.
 - b) For standby current measurements, the conditions are the same, except that the device is in SLEEP mode.
- 4: For RC option, does not include current through Rext. The current through the resistor can be estimated by the formula: $I_R = V_{DD}/2R_{ext}$ (mA) with Rext in kΩ.
- 5: 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 and VSS.

10.2 DC Characteristics: PIC16C52-04 (Commercial) PIC16C52-I04 (Industrial)

DC Characteristics All Pins Except Power Supply Pins		Standard Operating Conditions (unless otherwise specified) Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial) $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial) Operating Voltage V_{DD} range is described in Section 10.1.				
Characteristic	Sym	Min	Typ ⁽¹⁾	Max	Units	Conditions
Input Low Voltage I/O ports $\overline{\text{MCLR}}$ (Schmitt Trigger) T0CKI (Schmitt Trigger) OSC1 (Schmitt Trigger)	V_{IL}	V_{SS} V_{SS} V_{SS} V_{SS}		$0.2 V_{DD}$ $0.15 V_{DD}$ $0.15 V_{DD}$ $0.15 V_{DD}$ $0.3 V_{DD}$	V V V V V	Pin at hi-impedance RC ⁽⁴⁾ option only XT option
Input High Voltage I/O ports $\overline{\text{MCLR}}$ (Schmitt Trigger) T0CKI (Schmitt Trigger) OSC1 (Schmitt Trigger)	V_{IH}	$0.45 V_{DD}$ 2.0 $0.36 V_{DD}$ $0.85 V_{DD}$ $0.85 V_{DD}$ $0.85 V_{DD}$ $0.7 V_{DD}$		V_{DD} V_{DD} V_{DD} V_{DD} V_{DD} V_{DD} V_{DD}	V V V V V V V	For all V_{DD} ⁽⁵⁾ $4.0 \text{ V} < V_{DD} \leq 5.5 \text{ V}$ ⁽⁵⁾ $V_{DD} > 5.5 \text{ V}$ RC ⁽⁴⁾ option only XT option
Hysteresis of Schmitt Trigger inputs	V_{HYS}	$0.15 V_{DD}^*$			V	
Input Leakage Current^(2,3) I/O ports $\overline{\text{MCLR}}$ T0CKI OSC1	I_{IL}	-1 -5 -3 -3	0.5 0.5 0.5 0.5	+1 +5 +3 +3	μA μA μA μA	For $V_{DD} \leq 5.5 \text{ V}$ $V_{SS} \leq V_{PIN} \leq V_{DD}$, Pin at hi-impedance $V_{PIN} = V_{SS} + 0.25 \text{ V}$ $V_{PIN} = V_{DD}$ $V_{SS} \leq V_{PIN} \leq V_{DD}$ $V_{SS} \leq V_{PIN} \leq V_{DD}$, XT option
Output Low Voltage I/O ports OSC2/CLKOUT	V_{OL}			0.6 0.6	V V	$I_{OL} = 2.0 \text{ mA}$, $V_{DD} = 4.5 \text{ V}$ $I_{OL} = 1.6 \text{ mA}$, $V_{DD} = 4.5 \text{ V}$, RC option
Output High Voltage I/O ports ⁽³⁾ OSC2/CLKOUT	V_{OH}	$V_{DD} - 0.7$ $V_{DD} - 0.7$			V V	$I_{OH} = -2.0 \text{ mA}$, $V_{DD} = 4.5 \text{ V}$ $I_{OH} = -1.0 \text{ mA}$, $V_{DD} = 4.5 \text{ V}$, RC option

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is based on characterization results at 25°C . This data is for design guidance only and is not tested.

- 2: The leakage current on the $\overline{\text{MCLR}}$ / V_{PP} 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 voltage.
- 3: Negative current is defined as coming out of the pin.
- 4: For RC option, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C52 be driven with external clock in RC mode.
- 5: The user may use the better of the two specifications.

PIC16C52

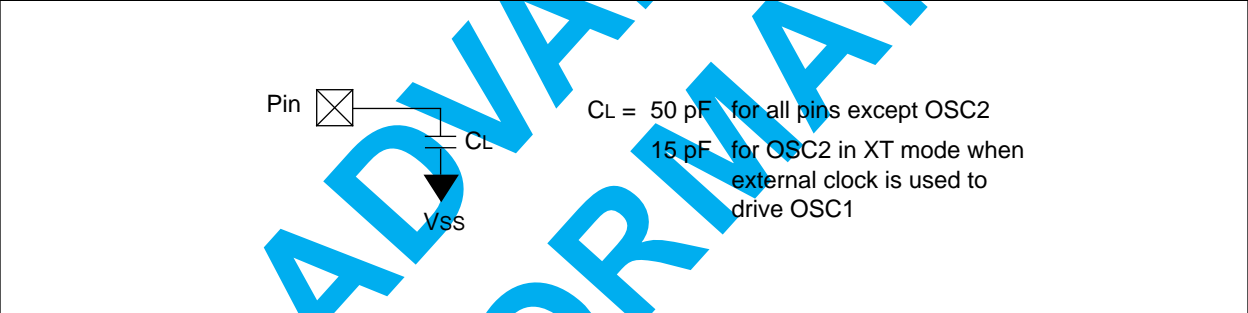
10.3 Timing Parameter Symbology and Load Conditions

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

- 1. TppS2ppS
- 2. TppS

T			
F	Frequency	T	Time
Lowercase subscripts (pp) and their meanings:			
pp			
2	to	mc	$\overline{\text{MCLR}}$
ck	CLKOUT	osc	oscillator
cy	cycle time	os	OSC1
drt	device reset timer	t0	T0CKI
io	I/O port		
Uppercase letters and their meanings:			
S		P	Period
F	Fall	R	Rise
H	High	V	Valid
I	Invalid (Hi-impedance)	Z	Hi-impedance
L	Low		

FIGURE 10-1: LOAD CONDITIONS - PIC16C52



10.4 Timing Diagrams and Specifications

FIGURE 10-2: EXTERNAL CLOCK TIMING - PIC16C52

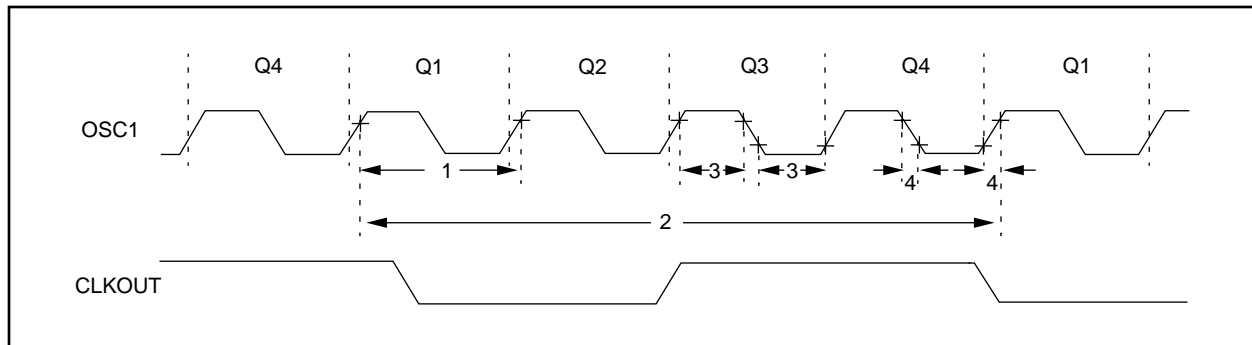


TABLE 10-1: EXTERNAL CLOCK TIMING REQUIREMENTS - PIC16C52

AC Characteristics Standard Operating Conditions (unless otherwise specified) Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial), $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial), Operating Voltage V_{DD} range is described in Section 10.1.							
Parameter No.	Sym	Characteristic	Min	Typ ⁽¹⁾	Max	Units	Conditions
	FOSC	External CLKIN Frequency ⁽²⁾	DC	—	4	MHz	RC osc mode
			DC	—	4	MHz	XT osc mode
		Oscillator Frequency ⁽²⁾	DC	—	4	MHz	RC osc mode
			0.1	—	4	MHz	XT osc mode
1	TOSC	External CLKIN Period ⁽²⁾	250	—	—	ns	RC osc mode
			250	—	—	ns	XT osc mode
		Oscillator Period ⁽²⁾	250	—	—	ns	RC osc mode
			250	—	10,000	ns	XT osc mode
2	TCY	Instruction Cycle Time ⁽³⁾	—	4/FOSC	—	—	
3	TosL, TosH	Clock in (OSC1) Low or High Time	50*	—	—	ns	XT oscillator
4	TosR, TosF	Clock in (OSC1) Rise or Fall Time	—	—	25*	ns	XT oscillator

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

2: 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.

When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

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

PIC16C52

FIGURE 10-3: CLKOUT AND I/O TIMING - PIC16C52

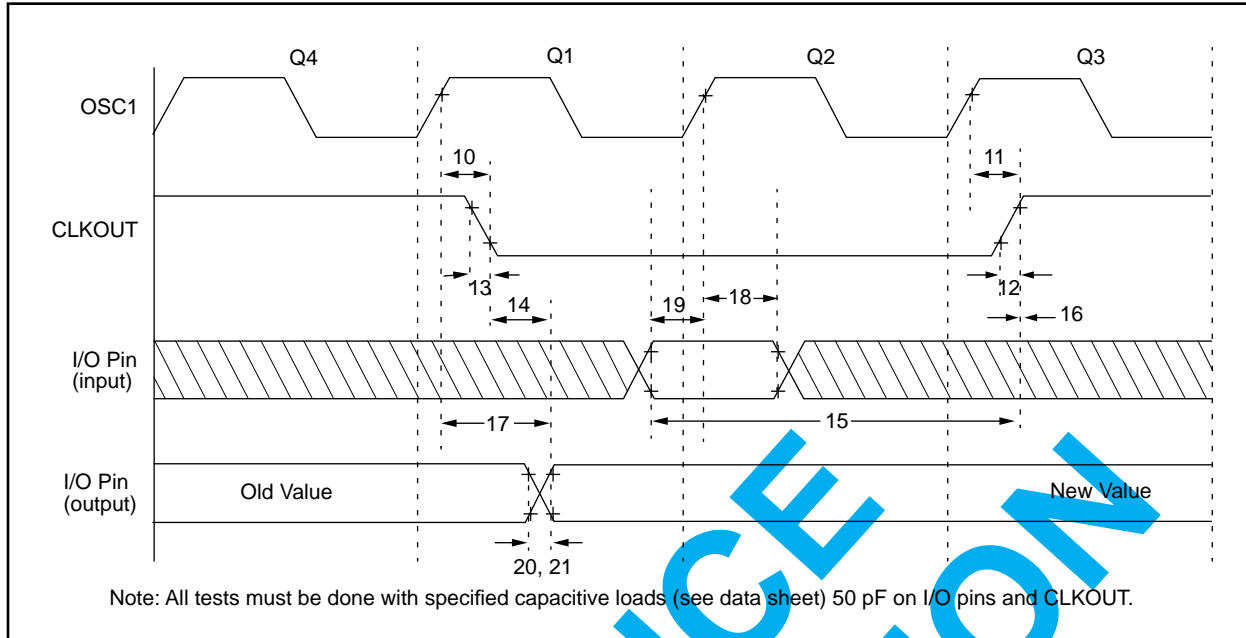


TABLE 10-2: CLKOUT AND I/O TIMING REQUIREMENTS - PIC16C52

AC Characteristics		Standard Operating Conditions (unless otherwise specified)				
		Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial), $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial),				
		Operating Voltage V_{DD} range is described in Section 10.1.				
Parameter No.	Sym	Characteristic	Min	Typ ⁽¹⁾	Max	Units
10	TosH2ckL	OSC1 \uparrow to CLKOUT \downarrow ⁽²⁾	—	15	30**	ns
11	TosH2ckH	OSC1 \uparrow to CLKOUT \uparrow ⁽²⁾	—	15	30**	ns
12	TckR	CLKOUT rise time ⁽²⁾	—	5	15**	ns
13	TckF	CLKOUT fall time ⁽²⁾	—	5	15**	ns
14	TckL2ioV	CLKOUT \downarrow to Port out valid ⁽²⁾	—	—	40**	ns
15	TioV2ckH	Port in valid before CLKOUT \uparrow ⁽²⁾	0.25 TCY+30*	—	—	ns
16	TckH2iol	Port in hold after CLKOUT \uparrow ⁽²⁾	0*	—	—	ns
17	TosH2ioV	OSC1 \uparrow (Q1 cycle) to Port out valid ⁽³⁾	—	—	100*	ns
18	TosH2iol	OSC1 \uparrow (Q2 cycle) to Port input invalid (I/O in hold time)	TBD	—	—	ns
19	TioV2osH	Port input valid to OSC1 \uparrow (I/O in setup time)	TBD	—	—	ns
20	TioR	Port output rise time ⁽³⁾	—	10	25**	ns
21	TioF	Port output fall time ⁽³⁾	—	10	25**	ns

* These parameters are characterized but not tested.

** These parameters are design targets and are not tested. No characterization data available at this time.

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

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

3: See Figure 10-1 for loading conditions.

FIGURE 10-4: RESET AND DEVICE RESET TIMER TIMING - PIC16C52

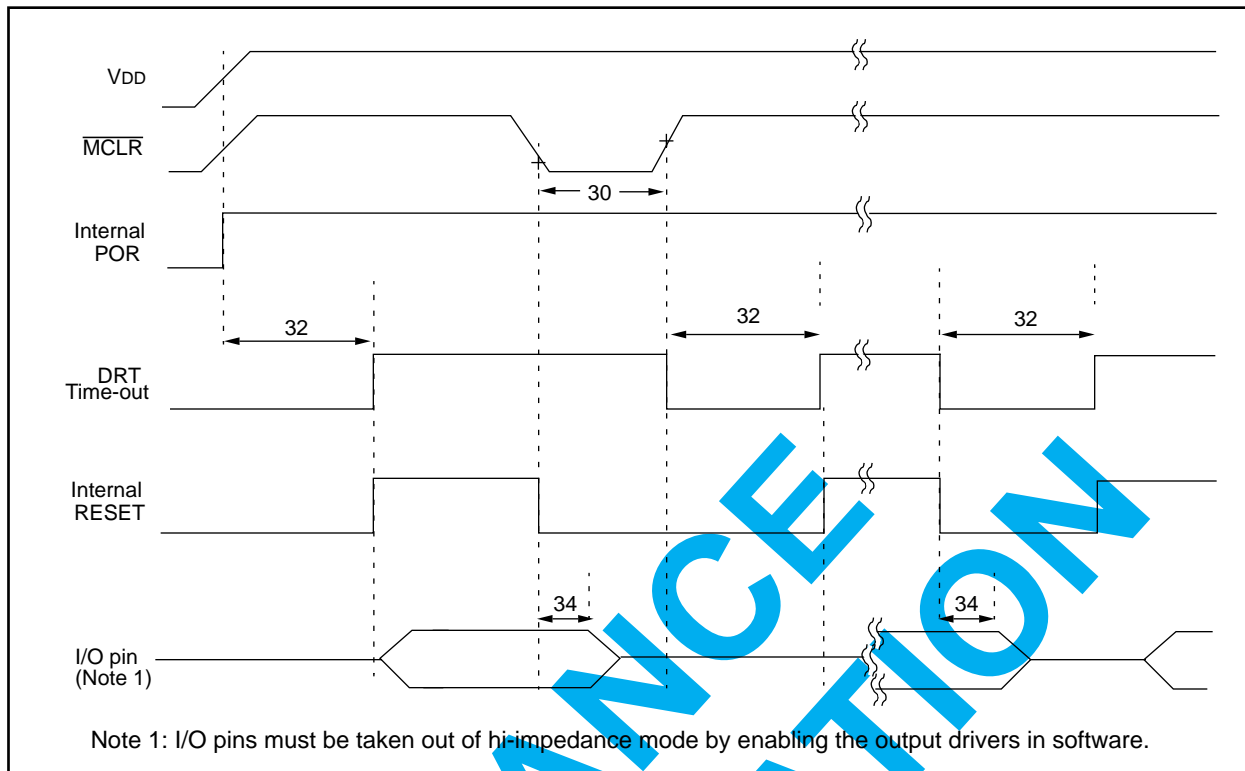


TABLE 10-3: RESET AND DEVICE RESET TIMER - PIC16C52

AC Characteristics Standard Operating Conditions (unless otherwise specified)							
Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial), $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial), Operating Voltage V_{DD} range is described in Section 10.1.							
Parameter No.	Sym	Characteristic	Min	Typ ⁽¹⁾	Max	Units	Conditions
30	TmclL	MCLR Pulse Width (low)	100*	—	—	ns	$V_{DD} = 5\text{ V}$
32	T _{DRT}	Device Reset Timer Period	9*	18*	30*	ms	$V_{DD} = 5\text{ V}$ (Commercial)
34	Tioz	I/O Hi-impedance from MCLR Low	—	—	100*	ns	

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

PIC16C52

FIGURE 10-5: TIMER0 CLOCK TIMINGS - PIC16C52

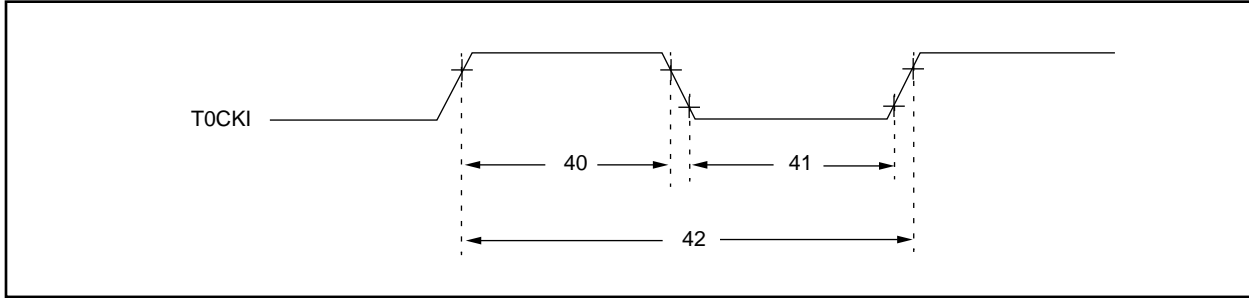


TABLE 10-4: TIMER0 CLOCK REQUIREMENTS - PIC16C52

AC Characteristics		Standard Operating Conditions (unless otherwise specified)					
		Operating Temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial), $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial), Operating Voltage V_{DD} range is described in Section 10.1.					
Parameter No.	Sym	Characteristic	Min	Typ ⁽¹⁾	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width - No Prescaler	$0.5 T_{CY} + 20^*$	—	—	ns	
		- With Prescaler	10^*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width - No Prescaler	$0.5 T_{CY} + 20^*$	—	—	ns	
		- With Prescaler	10^*	—	—	ns	
42	Tt0P	T0CKI Period	$20 \text{ or } \frac{T_{CY} + 40^*}{N}$	—	—	ns	Whichever is greater. N = Prescale Value (1, 2, 4,..., 256)

* These parameters are characterized but not tested.

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

ADVANCE INFORMATION

11.0 DC AND AC CHARACTERISTICS

The 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 are outside specified operating range (e.g., outside specified V_{DD} range). This is for information only and devices will 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 11-1: TYPICAL RC OSCILLATOR FREQUENCY vs. TEMPERATURE

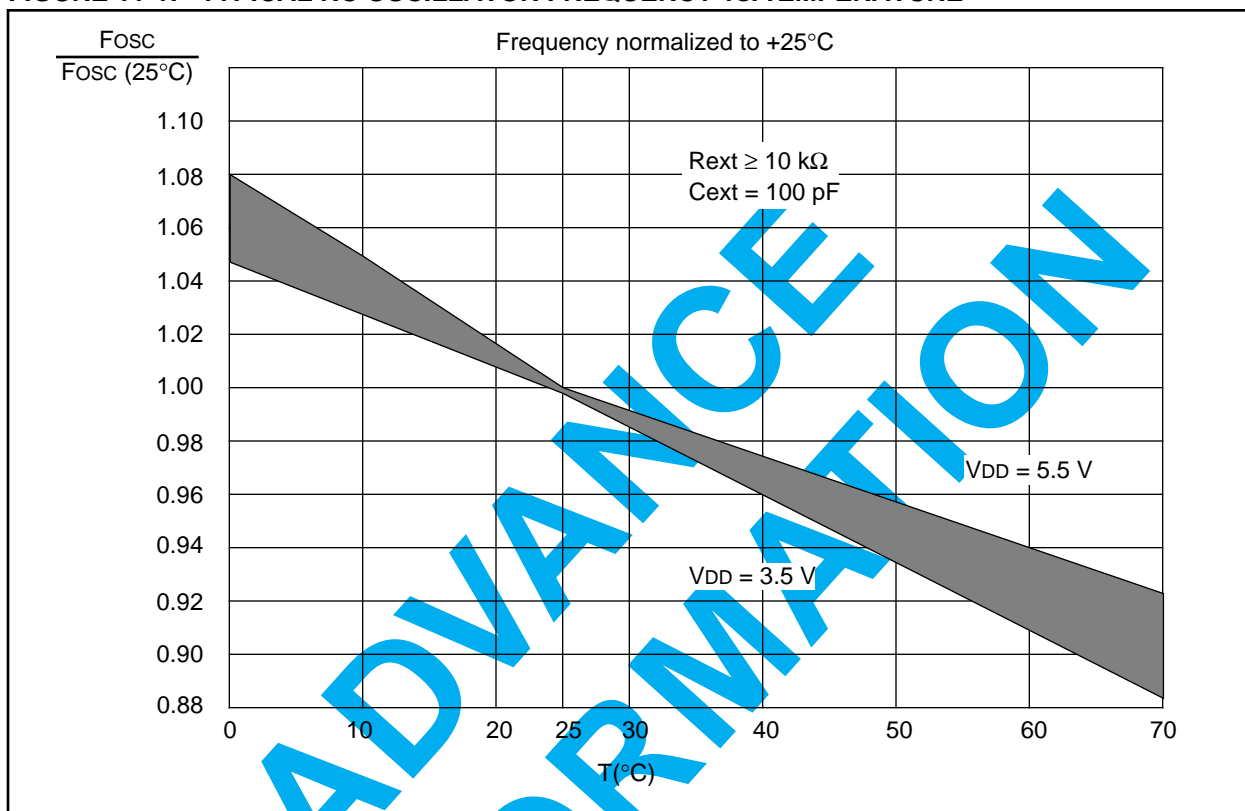


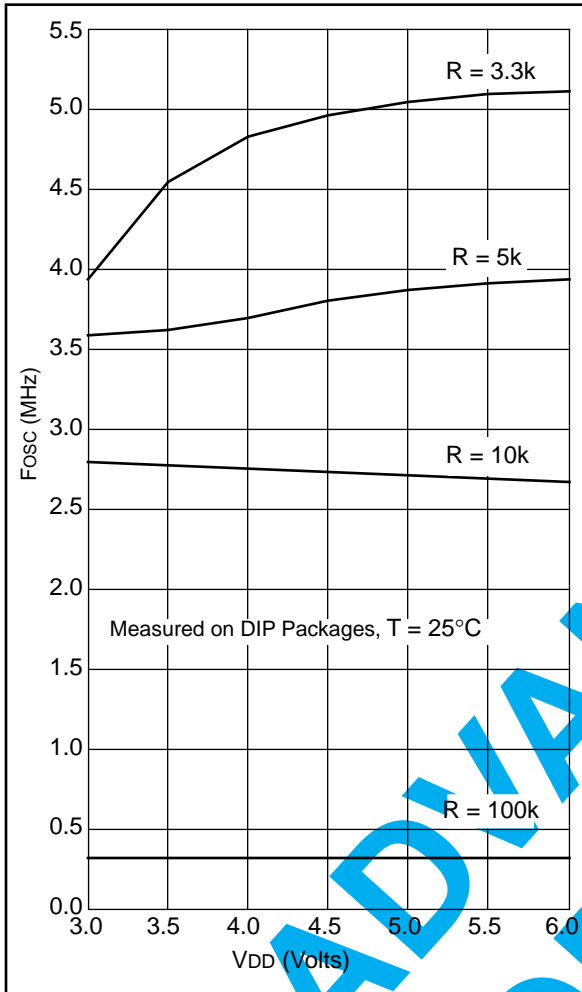
TABLE 11-1: RC OSCILLATOR FREQUENCIES

C_{ext}	R_{ext}	Average F_{osc} @ 5 V, 25°C	
20 pF	3.3 k	4.973 MHz	± 27%
	5 k	3.82 MHz	± 21%
	10 k	2.22 MHz	± 21%
	100 k	262.15 kHz	± 31%
100 pF	3.3 k	1.63 MHz	± 13%
	5 k	1.19 MHz	± 13%
	10 k	684.64 kHz	± 18%
	100 k	71.56 kHz	± 25%
300 pF	3.3 k	660 kHz	± 10%
	5.0 k	484.1 kHz	± 14%
	10 k	267.63 kHz	± 15%
	160 k	29.44 kHz	± 19%

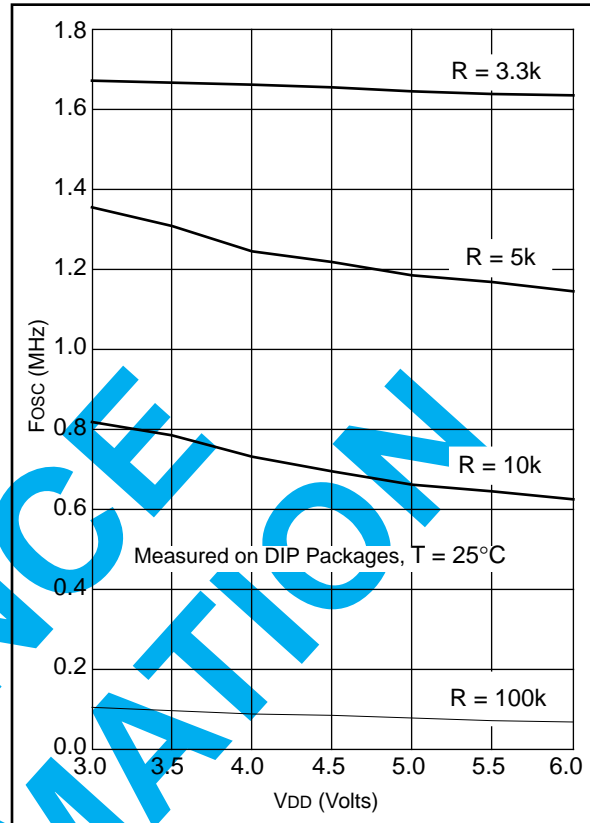
The frequencies are measured on DIP packages.

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 for $V_{DD} = 5\text{ V}$.

**FIGURE 11-2: TYPICAL RC OSCILLATOR
FREQUENCY vs. V_{DD} ,
 $C_{EXT} = 20\text{pF}$**



**FIGURE 11-3: TYPICAL RC OSCILLATOR
FREQUENCY vs. V_{DD} ,
 $C_{EXT} = 100\text{pF}$**



**FIGURE 11-4: TYPICAL RC OSCILLATOR
FREQUENCY vs. V_{DD} ,
 $C_{EXT} = 300\text{pF}$**

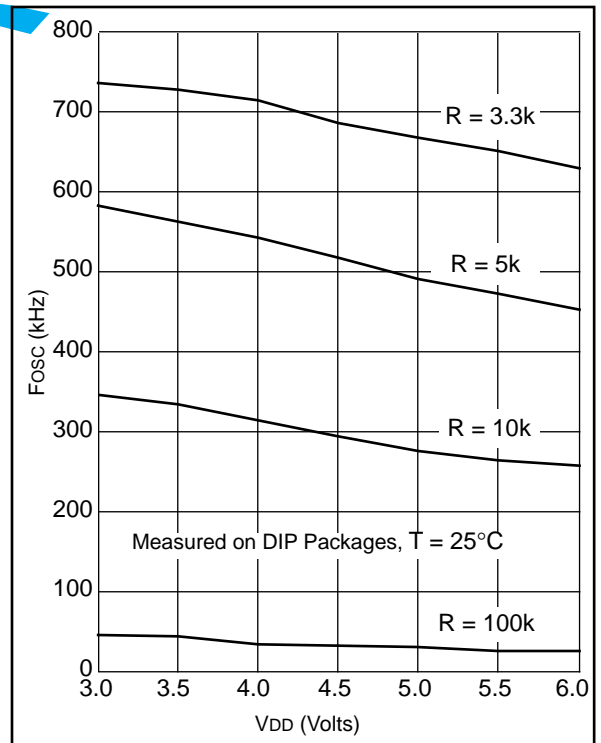


FIGURE 11-5: TYPICAL I_{PD} vs. V_{DD}

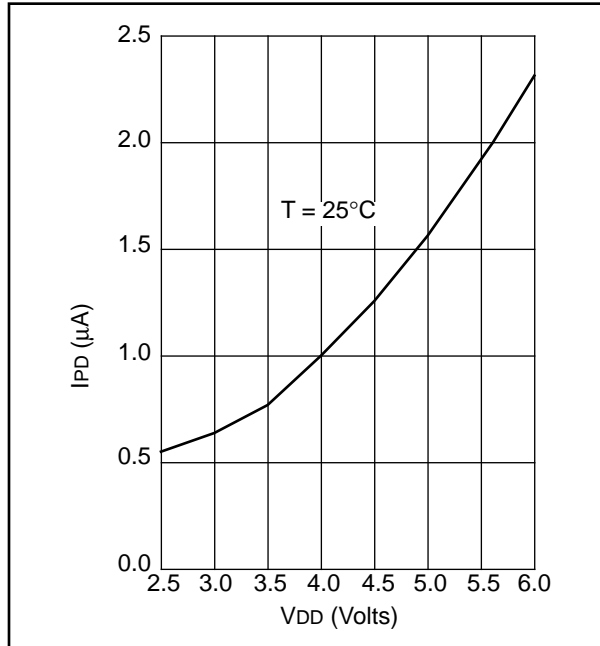
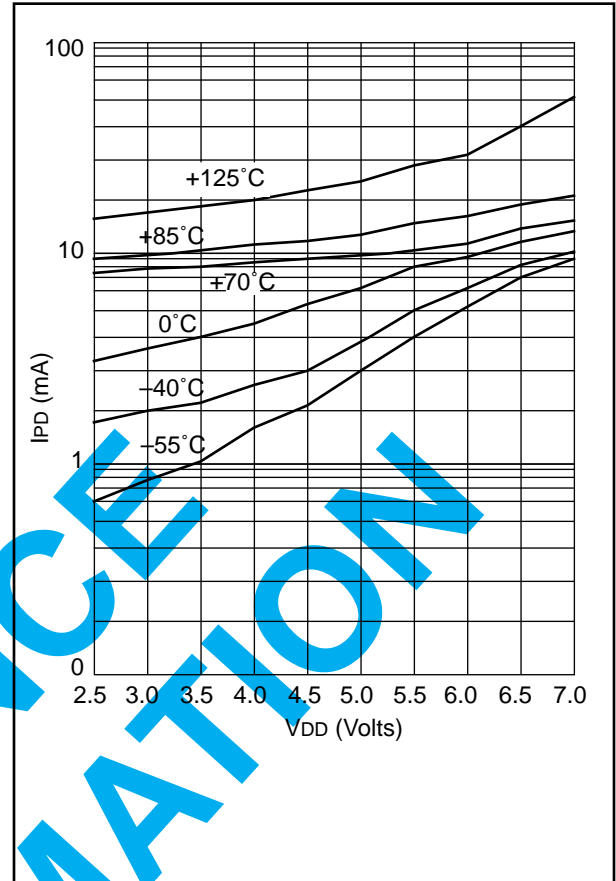


FIGURE 11-6: MAXIMUM I_{PD} vs. V_{DD}



ADVANCE
INFORMATION

PIC16C52

FIGURE 11-7: V_{TH} (INPUT THRESHOLD VOLTAGE) OF I/O PINS vs. V_{DD}

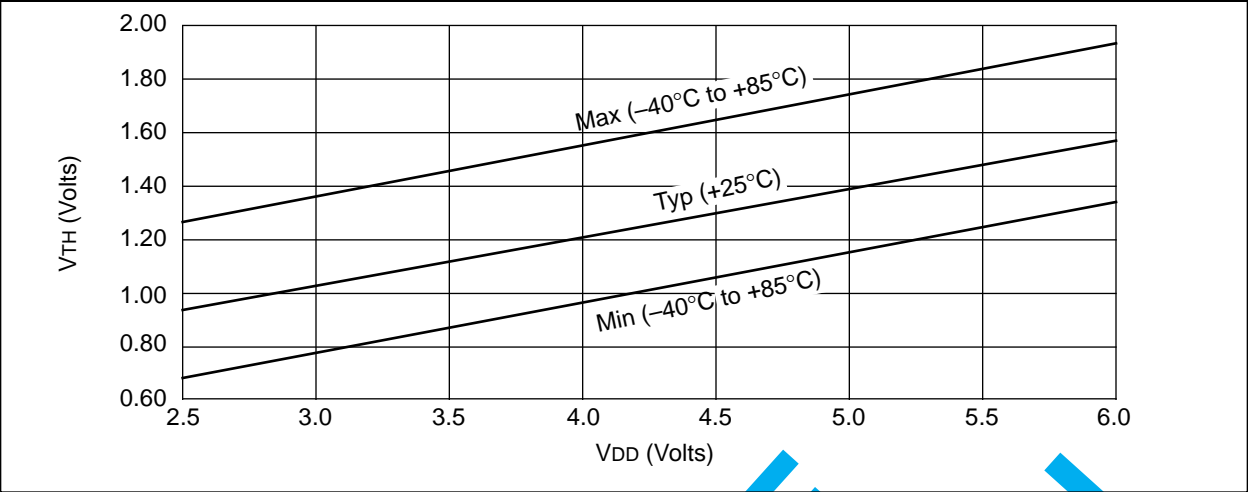


FIGURE 11-8: V_{IH} , V_{IL} OF \overline{MCLR} , $T0CKI$ AND $OSC1$ (IN RC MODE) vs. V_{DD}

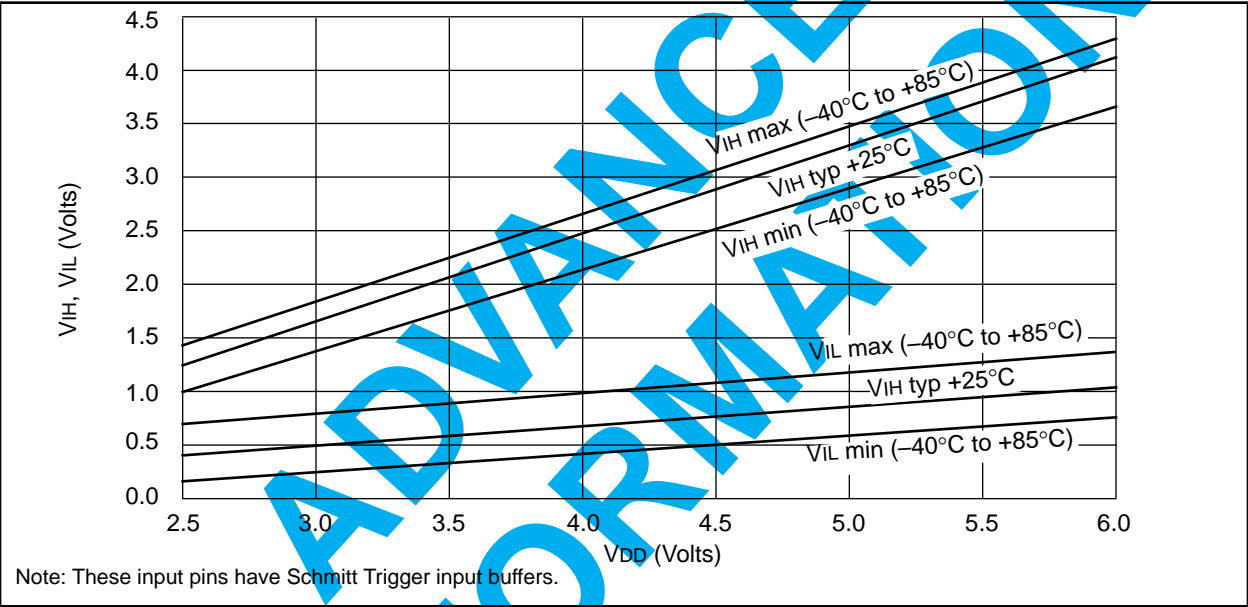


FIGURE 11-9: V_{TH} (INPUT THRESHOLD VOLTAGE) OF $OSC1$ INPUT (IN XT, HS, AND LP MODES) vs. V_{DD}

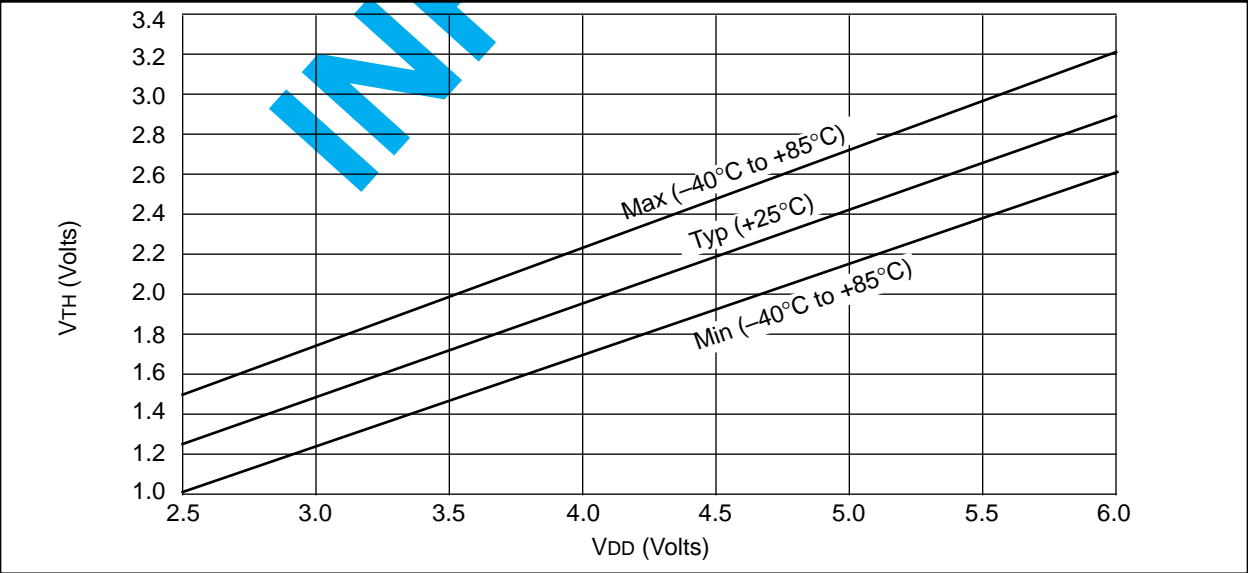


FIGURE 11-10: TYPICAL I_{DD} vs. FREQUENCY (EXTERNAL CLOCK, 25°C)

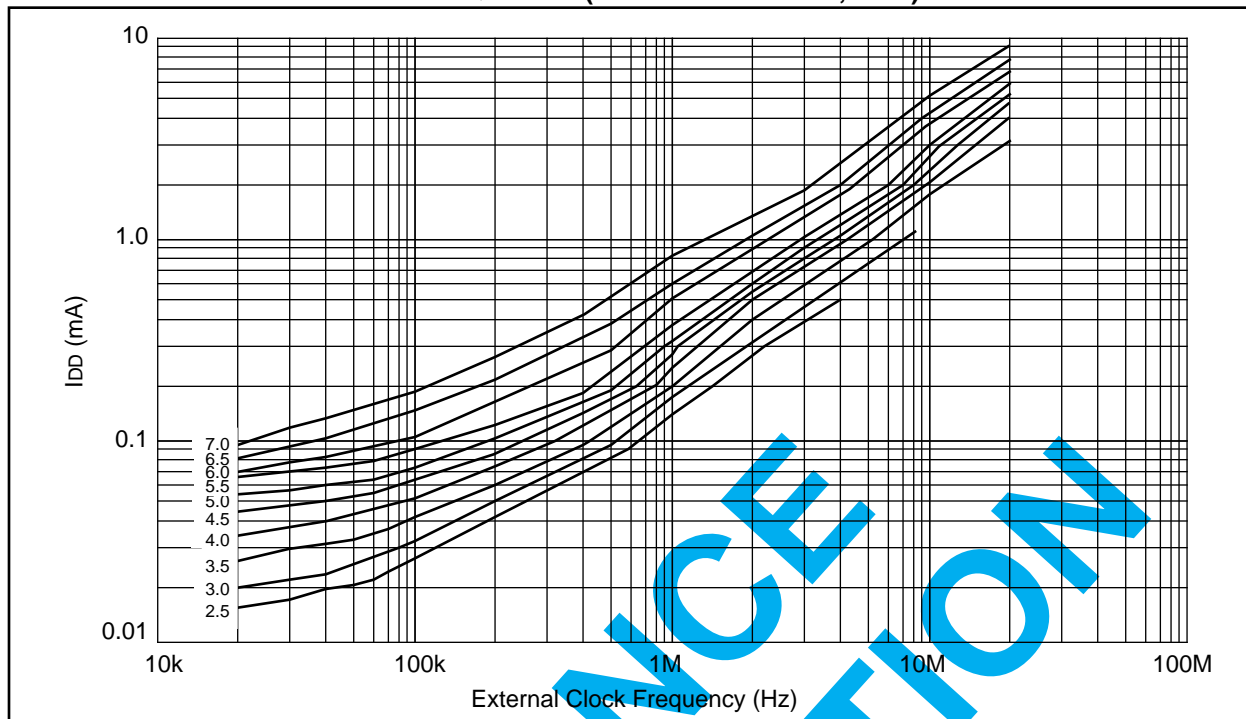
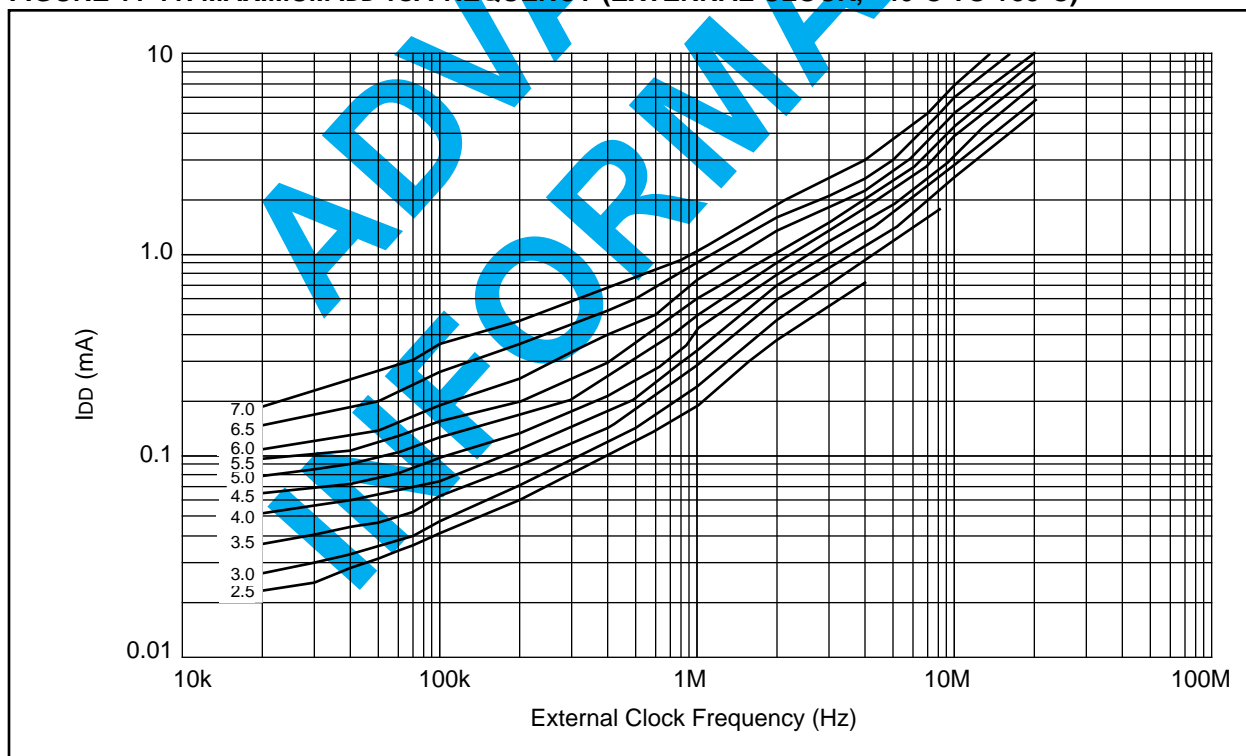


FIGURE 11-11: MAXIMUM I_{DD} vs. FREQUENCY (EXTERNAL CLOCK, -40°C TO +85°C)



PIC16C52

FIGURE 11-12: MAXIMUM I_{DD} vs. FREQUENCY (EXTERNAL CLOCK -55°C TO $+125^{\circ}\text{C}$)

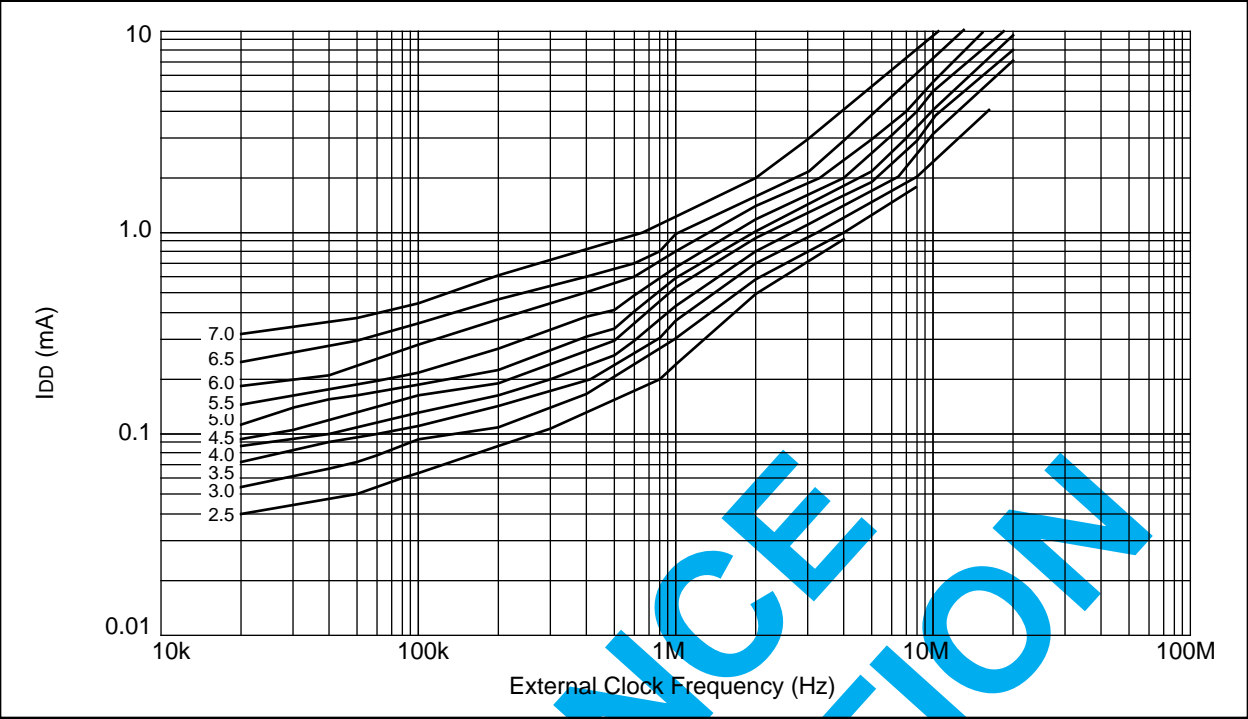


FIGURE 11-13: TRANSCONDUCTANCE (g_m) OF XT OSCILLATOR vs. V_{DD}

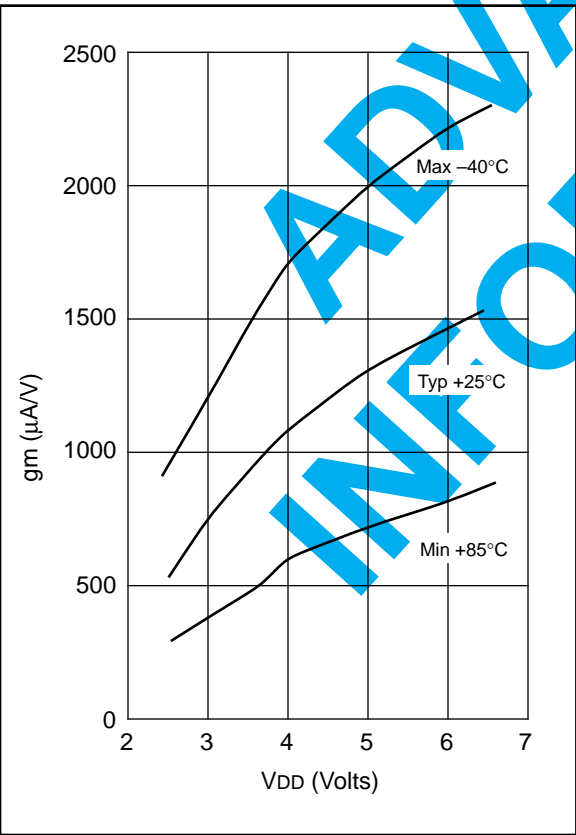


TABLE 11-2: INPUT CAPACITANCE FOR PIC16C52

P_{in}	Typical Capacitance (pF)	
	18L PDIP	18L SOIC
RA port	5.0	4.3
RB port	5.0	4.3
MCLR	17.0	17.0
OSC1	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 $\pm 25\%$ (three standard deviations) should be taken into account.

FIGURE 11-14: I_{OH} vs. V_{OH} , $V_{DD} = 3\text{ V}$

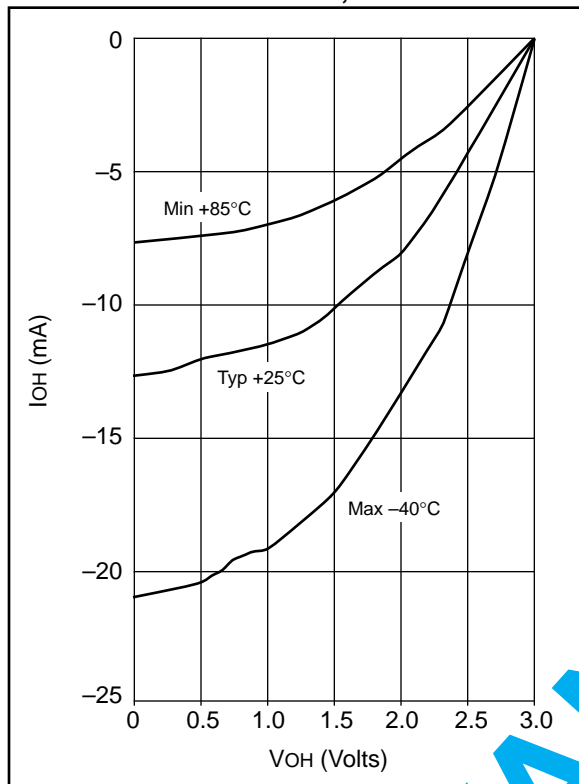


FIGURE 11-16: I_{OL} vs. V_{OL} , $V_{DD} = 3\text{ V}$

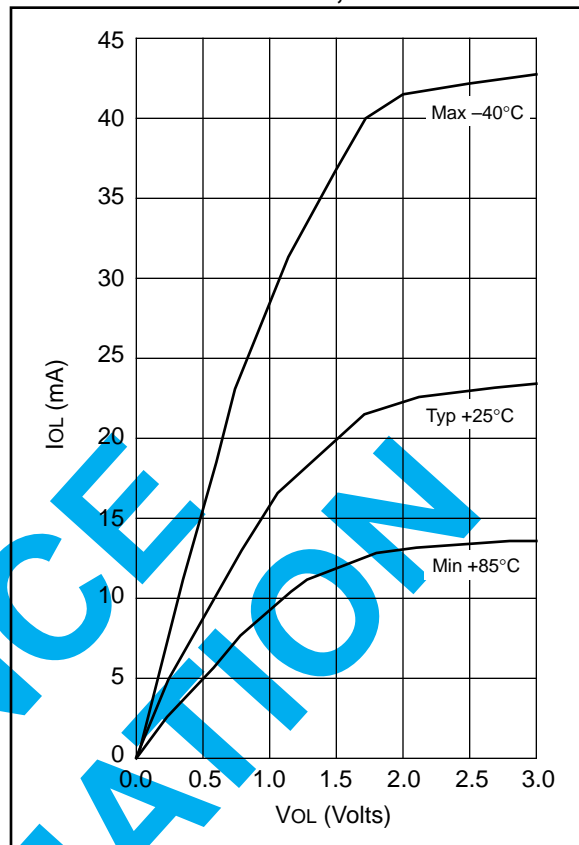


FIGURE 11-15: I_{OH} vs. V_{OH} , $V_{DD} = 5\text{ V}$

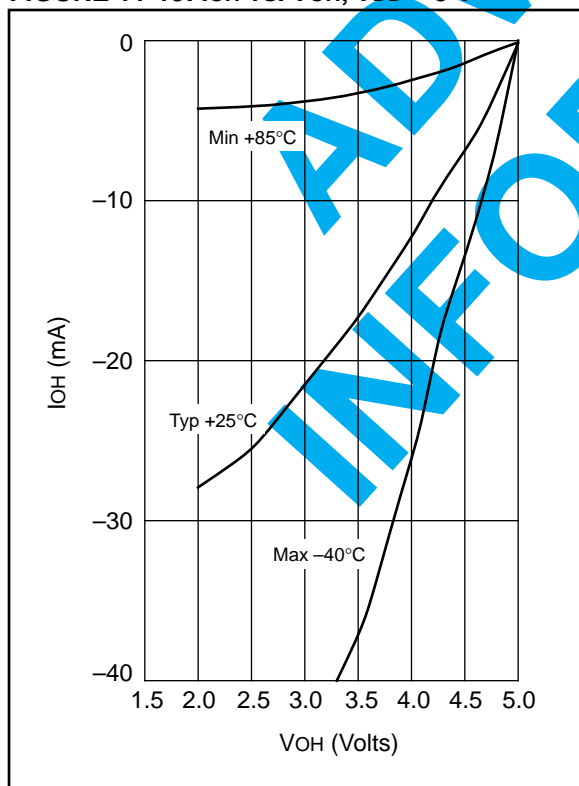
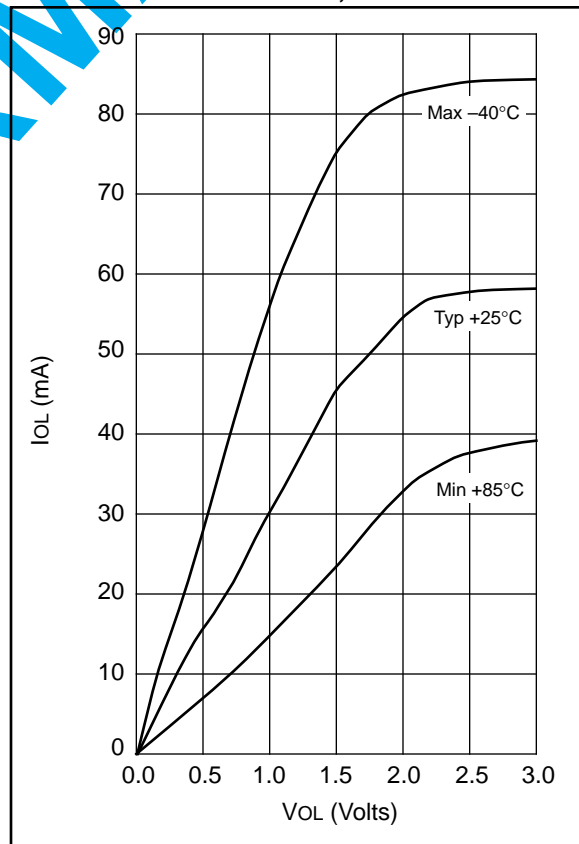


FIGURE 11-17: I_{OL} vs. V_{OL} , $V_{DD} = 5\text{ V}$



NOTES:

ADVANCE
INFORMATION

12.0 PACKAGING INFORMATION

12.1 Package Marking Information

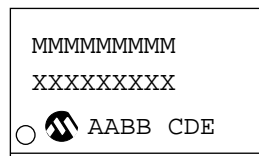
18-Lead PDIP



Example



18-Lead SOIC



Example

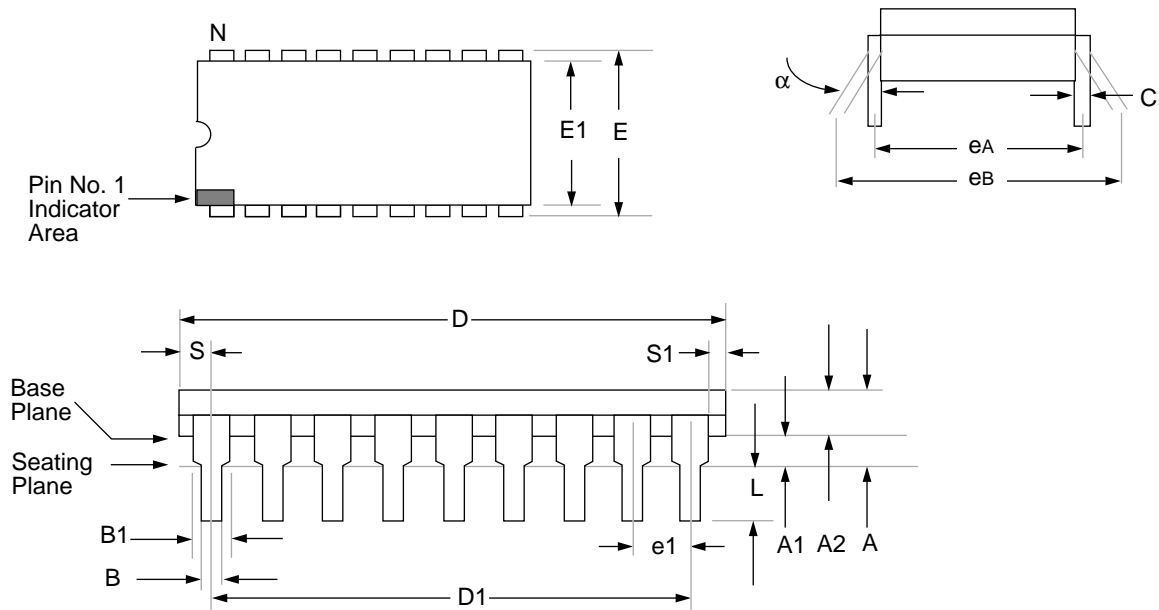


Legend:	MM...M	Microchip part number information
	XX...X	Customer specific information*
	AA	Year code (last two digits of calendar year)
	BB	Week code (week of January 1 is week '01')
	C	Facility code of the plant at which wafer is manufactured
		C = Chandler, Arizona, U.S.A.,
		S = Tempe, Arizona, U.S.A.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

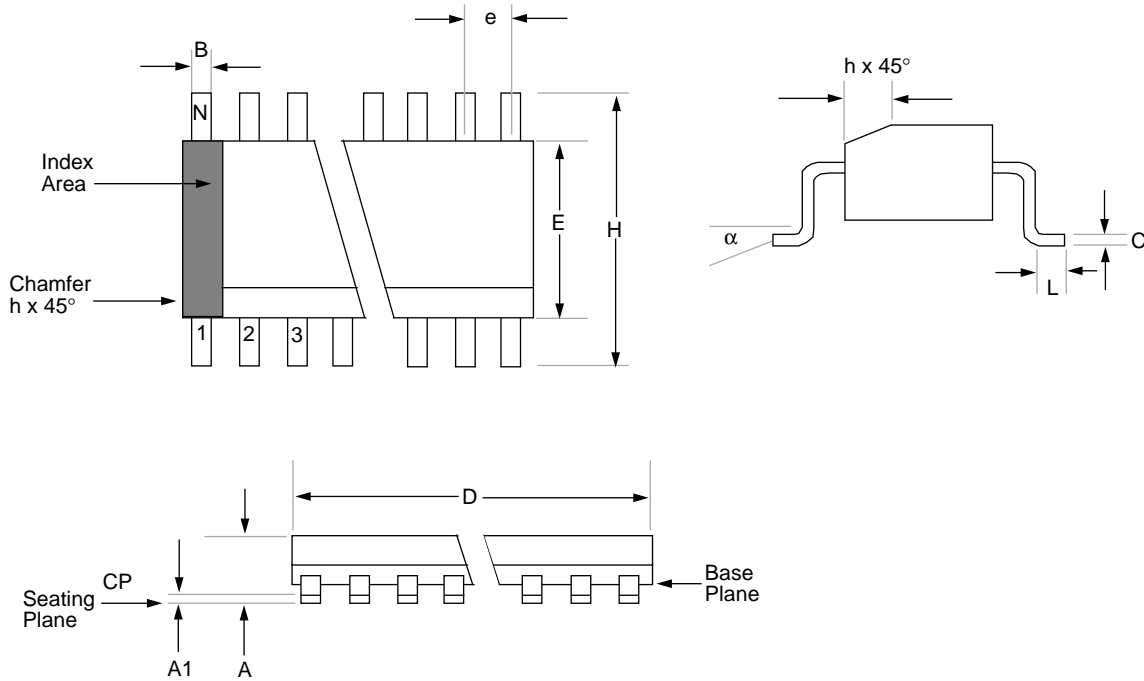
PIC16C52

12.2 18-Lead Plastic Dual In-Line (PDIP) - 300 mil



Package Group: Plastic Dual In-Line (PLA)						
Symbol	Millimeters			Inches		
	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	—	

12.3 18-Lead Plastic Surface Mount (SOIC) - 300 mil



Package Group: Plastic SOIC (SO)						
Symbol	Millimeters			Inches		
	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	

PIC16C52

NOTES:

APPENDIX A: COMPATIBILITY

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

1. Check any `CALL`, `GOTO` or instructions that modify the PC to determine if any program memory page select operations (PA2, PA1, PA0 bits) need to be made.
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 special function register page switching. Redefine data variables to reallocate them.
4. Verify all writes to STATUS, OPTION, and FSR registers since these have changed.
5. Change reset vector to proper value for processor used.
6. Remove any use of the `ADDLW` and `SUBLW` instructions.
7. Rewrite any code segments that use interrupts.

APPENDIX B: WHAT'S NEW

This is the first version of the PIC16C52 data sheet. It is based on the PIC16C5X data sheet.

PIC16C52

NOTES:

APPENDIX C: PIC16/17 MICROCONTROLLERS

TABLE C-1: PIC16C5X FAMILY OF DEVICES

	Clock		Memory		Peripherals		Features		
	Maximum Frequency of Operation (MHz)		Program Memory (words)		Timer Module(s)		I/O Pins		
			RAM Data Memory (bytes)						
PIC16C52	4	384	—	25	TMR0	12	3.0-6.25	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	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP
PIC16CR54 ⁽²⁾	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
PIC16CR54B ⁽¹⁾	20	—	512	25	TMR0	12	2.5-6.25	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
PIC16CR56 ⁽¹⁾	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
PIC16CR57A ⁽²⁾	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	2.5-6.25	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
PIC16CR58B ⁽¹⁾	20	—	2K	73	TMR0	12	2.5-6.25	33	18-pin DIP, SOIC; 20-pin SSOP

PIC16C52

TABLE C-2: PIC16C62X FAMILY OF DEVICES

Clock			Memory			Peripherals			Features																													
PIC16C620	20	512	Maximum Frequency of Operation (MHz)		80	TMR0	2	Yes	4	13	3.0-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP																								
			Program Memory																																			
			Data Memory (bytes)																																			
			EPROM																																			
PIC16C621	20	1K	80		80	TMR0	2	Yes	4	13	3.0-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP																								
			128																																			
PIC16C622	20	2K	128		128	TMR0	2	Yes	4	13	3.0-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP																								
			2K																																			
All PIC16C62X 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.																																						

TABLE C-3: 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 Ports (SPI/I ² C, USART)	Parallel Slave Port	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Brown-out Reset
PIC16C61	20	1K	—	36	—	TMR0	—	—	—	3	13	3.0-6.0	Yes	—
PIC16C62	20	2K	—	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	—	7	22	3.0-6.0	Yes	—
PIC16C62A ⁽¹⁾	20	2K	—	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	—	7	22	3.0-6.0	Yes	Yes
PIC16C62 ⁽¹⁾	20	—	2K	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	—	7	22	3.0-6.0	Yes	Yes
PIC16C63 ⁽¹⁾	20	4K	—	192	—	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	—	10	22	3.0-6.0	Yes	Yes
PIC16C64	20	2K	—	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	Yes	8	33	3.0-6.0	Yes	—
PIC16C64A ⁽¹⁾	20	2K	—	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	Yes	8	33	3.0-6.0	Yes	Yes
PIC16C64 ⁽¹⁾	20	—	2K	128	—	TMR0, TMR1, TMR2	1	SPI/I ² C	Yes	8	33	3.0-6.0	Yes	Yes
PIC16C65	20	4K	—	192	—	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	Yes	11	33	3.0-6.0	Yes	—
PIC16C65A ⁽¹⁾	20	4K	—	192	—	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	Yes	11	33	3.0-6.0	Yes	Yes

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 sales office for availability of these devices.

PIC16C52

TABLE C-4: PIC16C7X FAMILY OF DEVICES

	Clock		Memory		Peripherals					Features																
	Maximum Frequency of Operation (MHz)		Program Memory		Timer Modules(s)					Serial Ports (SPI/I ² C, USART)		Parallel Slave Port		AD Converter (8-bit) Channels		Interrupt Sources		I/O Pins		Voltage Range (Volts)		In-Circuit Serial Programming		Brown-out Reset		Packages
PIC16C70 ⁽¹⁾	20	512	36	TMR0	—	—	—	4	4	4	13	3.0-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP											
PIC16C71	20	1K	36	TMR0	—	—	—	4	4	4	13	3.0-6.0	Yes	—	18-pin DIP, SOIC											
PIC16C71A ⁽¹⁾	20	1K	68	TMR0	—	—	—	4	4	4	13	3.0-6.0	Yes	Yes	18-pin DIP, SOIC; 20-pin SSOP											
PIC16C72 ⁽¹⁾	20	2K	128	TMR0, TMR1, TMR2	1	SPI/I ² C	—	5	8	22	3.0-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP												
PIC16C73	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	—	5	11	22	3.0-6.0	Yes	—	28-pin SDIP, SOIC												
PIC16C73A ⁽¹⁾	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	—	5	11	22	3.0-6.0	Yes	Yes	28-pin SDIP, SOIC												
PIC16C74	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	Yes	8	12	33	3.0-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP												
PIC16C74A ⁽¹⁾	20	4K	192	TMR0, TMR1, TMR2	2	SPI/I ² C, USART	Yes	8	12	33	3.0-6.0	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 PIC16CXX 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.

TABLE C-5: 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	
ROM	EEPROM										
PIC16C83 ⁽¹⁾	10	512	—	36	64	TMR0	4	13	Yes	2.0-6.0	18-pin DIP, SOIC
PIC16CR83 ⁽¹⁾	10	—	512	36	64	TMR0	4	13	Yes	2.0-6.0	18-pin DIP, SOIC
PIC16C84	10	1K	—	36	64	TMR0	4	13	Yes	2.0-6.0	18-pin DIP, SOIC
PIC16C84A ⁽¹⁾	10	1K	—	68	64	TMR0	4	13	Yes	2.0-6.0	18-pin DIP, SOIC
PIC16CR84 ⁽¹⁾	10	—	1K	68	64	TMR0	4	13	Yes	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 PIC16CXX 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.

PIC16C52

TABLE C-6: PIC17CXX FAMILY OF DEVICES

	Clock			Memory			Peripherals				Features				
	Maximum Frequency of Operation (MHz)	Program Memory (bytes)	RAM Data Memory (bytes)	EPROM	Timer Modules(s)	Captures PPMs	Serial Port(s) (USART)	External Interrupts	I/O Pins	Voltage Range (Volts)	Single Instruction Multiply	In-Circuit Serial Programming	Number of Instructions	Packages	
PIC17C42	25	2K	232	TMR0, TMR1, TMR2, TMR3	2 2	Yes	Yes	11	33	4.5-5.5	Yes	Yes	55	40-pin DIP; 44-pin PLCC, MQFP	
PIC17C43	25	4K	454	TMR0, TMR1, TMR2, TMR3	2 2	Yes	Yes	11	33	2.5-6.0	Yes	Yes	58	40-pin DIP; 44-pin PLCC, TQFP	
PIC17C44	25	8K	454	TMR0, TMR1, TMR2, TMR3	2 2	Yes	Yes	11	33	2.5-6.0	Yes	Yes	58	40-pin DIP; 44-pin PLCC, TQFP	

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

C.1 Pin Compatibility

Devices that have the same package type and VDD, VSS and MCLR pin locations are said to be pin compatible. This allows these different devices to operate in the same socket. Compatible devices may only require 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 C-7: PIN COMPATIBLE DEVICES

Pin Compatible Devices	Package
PIC16C52, 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

PIC16C52

NOTES:

INDEX

A

Absolute Maximum Ratings	51
ALU	7
Architectural Overview	7
Assembler	48

B

Block Diagram	
On-Chip Reset Circuit	27
PIC16C52	8
Timer0	19
Timer0 Prescaler	22
Brown-Out Protection Circuit	31

C

C Compiler (MP-C)	45, 49
Carry	7
Clocking Scheme	10
Code Protection	23, 31
Configuration Bits	23
Configuration Word - PIC16C52	23

D

DC Characteristics	52, 53
Development Support	45
Development Systems	49
Development Tools	45
Device Varieties	5
Digit Carry	7
Dynamic Data Exchange (DDE)	45

E

Electrical Characteristics	51
External Power-On Reset Circuit	28

F

Family of Devices	4
PIC16C5X	73
PIC16C62X	74
PIC16C6X	75
PIC16C7X	76
PIC16C8X	77
PIC17CXX	78
FSR	16, 26
Fuzzy Logic Dev. System (<i>fuzzyTECH</i> ®-MP)	45, 49

I

I/O Interfacing	17
I/O Ports	17
I/O Programming Considerations	18
ID Locations	23, 31
INDF	16, 26
Indirect Data Addressing	16
Instruction Cycle	10
Instruction Flow/Pipelining	10
Instruction Set Summary	34
Integrated	48

L

Loading of PC	15
---------------------	----

M

MCLR	26
Memory Organization	11
Data Memory	11
Program Memory	11
MPASM Assembler	45, 48
MP-C C Compiler	49
MPSIM Software Simulator	45, 49

O

One-Time-Programmable (OTP) Devices	5
OPTION Register	14
OSC selection	23
Oscillator Configurations	24
Oscillator Types	
RC	24
XT	24

P

Packaging Information	67
18-Lead Plastic Dual In-Line (PDIP) - 300 mil	68
18-Lead Plastic Surface Mount (SOIC) - 300 mil	69
PCL	26
PIC16C52 DC and AC Characteristics	59
PICDEM-1 Low-Cost PIC16/17 Demo Board	45, 47
PICDEM-2 Low-Cost PIC16CXX Demo Board	45, 47
PICMASTER Probes	46
PICMASTER System Configuration	45
PICMASTER™ RT In-Circuit Emulator	45
PICSTART™ Low-Cost Development System	45, 47
Pin Compatible Devices	79
Pinout Description	9
POR	
Oscillator Start-Up Timer (OST)	23, 28, 30
PD	26, 30
Power-On Reset (POR)	23, 26, 28
TO	26, 30
PORTA	17, 26
PORTB	17, 26
Power-Down Mode	31
Prescaler	22
PRO MATE™ Universal Programmer	45, 47

Q

Quick-Turnaround-Production (QTP) Devices	5
---	---

R

RC Oscillator	25
Read Modify Write	18
Reset	23, 26
Reset on Brown-Out	31

S

Serialized Quick-Turnaround-Production (SQTP) Devices ..	5
SLEEP	23, 31
Software Simulator (MPSIM)	49
Special Features of the CPU	23
STATUS	7, 26
STATUS Word Register	13
Summary of Port Registers	17

PIC16C52

T

Timer0	19
Timer0 Module	19
Timer0 with External Clock	21
TMR0 Register	19
Timing Diagrams and Specifications	55
Timing Parameter Symbolology and Load Conditions	54
TRIS Registers	17

W

W	26
Wake-up from SLEEP	31

Z

Zero bit	7
----------------	---

LIST OF EXAMPLES

Example 3-1: Instruction Pipeline Flow	10
Example 4-1: Indirect Addressing.....	16
Example 4-2: How To Clear RAM Using Indirect Addressing.....	16
Example 5-1: Read-Modify-Write Instructions on an I/O Port	18

LIST OF FIGURES

Figure 3-1: PIC16C52 Block Diagram	8	Figure 11-8: V_{IH} , V_{IL} of \overline{MCLR} , $T0CKI$ and $OSC1$ (in RC Mode) vs. V_{DD}	62
Figure 3-2: Clock/Instruction Cycle	10	Figure 11-9: V_{TH} (Input Threshold Voltage) of $OSC1$ Input (in XT, HS, and LP modes) vs. V_{DD}	62
Figure 4-1: PIC16C52 Program Memory Map and Stack	11	Figure 11-10: Typical I_{DD} vs. Frequency (External Clock, 25°C)	63
Figure 4-2: PIC16C52 Register File Map	11	Figure 11-11: Maximum I_{DD} vs. Frequency (External Clock, -40°C to +85°C)	63
Figure 4-3: STATUS Register (Address:03h)	13	Figure 11-12: Maximum I_{DD} vs. Frequency (External Clock -55°C to +125°C)	64
Figure 4-4: OPTION Register	14	Figure 11-13: Transconductance (gm) of XT Oscillator vs. V_{DD}	64
Figure 4-5: Loading of PC Branch Instructions - PIC16C52	15	Figure 11-14: I_{OH} vs. V_{OH} , $V_{DD} = 3\text{ V}$	65
Figure 4-6: Direct/Indirect Addressing	16	Figure 11-15: I_{OH} vs. V_{OH} , $V_{DD} = 5\text{ V}$	65
Figure 5-1: Equivalent Circuit for a Single I/O Pin	17	Figure 11-16: I_{OL} vs. V_{OL} , $V_{DD} = 3\text{ V}$	65
Figure 5-2: Successive I/O Operation	18	Figure 11-17: I_{OL} vs. V_{OL} , $V_{DD} = 5\text{ V}$	65
Figure 6-1: Timer0 Block Diagram	19		
Figure 6-2: Electrical Structure of $T0CKI$ Pin	19		
Figure 6-3: Timer0 Timing: Internal Clock/No Prescale	20		
Figure 6-4: Timer0 Timing: Internal Clock/Prescale 1:2	20		
Figure 6-5: Timer0 Timing With External Clock	21		
Figure 6-6: Block Diagram of the Timer0 Prescaler	22		
Figure 7-1: Configuration Word for PIC16C52	23		
Figure 7-2: Crystal Operation or Ceramic Resonator (XT OSC Configuration)	24		
Figure 7-3: External Clock Input Operation (XT OSC Configuration)	24		
Figure 7-4: External Parallel Resonant Crystal Oscillator Circuit	25		
Figure 7-5: External Series Resonant Crystal Oscillator Circuit	25		
Figure 7-6: RC Oscillator Mode	25		
Figure 7-7: Simplified Block Diagram of On-Chip Reset Circuit	27		
Figure 7-8: Electrical Structure of \overline{MCLR}/V_{PP} Pin	28		
Figure 7-9: External Power-On Reset Circuit (For Slow V_{DD} Power-Up)	28		
Figure 7-10: Time-Out Sequence on Power-Up (\overline{MCLR} Not Tied to V_{DD})	29		
Figure 7-11: Time-Out Sequence on Power-Up (\overline{MCLR} Tied to V_{DD}): Fast V_{DD} Rise Time	29		
Figure 7-12: Time-Out Sequence on Power-Up (\overline{MCLR} Tied to V_{DD}): Slow V_{DD} Rise Time	29		
Figure 7-13: Brown-Out Protection Circuit 1	31		
Figure 7-14: Brown-Out Protection Circuit 2	31		
Figure 8-1: General Format for Instructions	33		
Figure 9-1: PICMASTER System Configuration	45		
Figure 10-1: Load Conditions - PIC16C52	54		
Figure 10-2: External Clock Timing - PIC16C52	55		
Figure 10-3: $CLKOUT$ and I/O Timing - PIC16C52	56		
Figure 10-4: Reset and Device Reset Timer Timing - PIC16C52	57		
Figure 10-5: Timer0 Clock Timings - PIC16C52	58		
Figure 11-1: Typical RC Oscillator Frequency vs. Temperature	59		
Figure 11-2: Typical RC Oscillator Frequency vs. V_{DD} , $C_{EXT} = 20\text{ pF}$	60		
Figure 11-3: Typical RC Oscillator Frequency vs. V_{DD} , $C_{EXT} = 100\text{ pF}$	60		
Figure 11-4: Typical RC Oscillator Frequency vs. V_{DD} , $C_{EXT} = 300\text{ pF}$	60		
Figure 11-5: Typical I_{PD} vs. V_{DD}	61		
Figure 11-6: Maximum I_{PD} vs. V_{DD}	61		
Figure 11-7: V_{TH} (Input Threshold Voltage) of I/O Pins vs. V_{DD}	62		

PIC16C52

LIST OF TABLES

Table 1-1:	PIC16C5X Family of Devices.....	4
Table 3-1:	PIC16C52 Pinout Description	9
Table 4-1:	Special Function Register Summary.....	12
Table 5-1:	Summary of Port Registers	17
Table 6-1:	Registers Associated With Timer0	20
Table 7-1:	Capacitor Selection For Ceramic Resonators - PIC16C52	24
Table 7-2:	Capacitor Selection For Crystal Oscillator - PIC16C52.....	24
Table 7-3:	Reset Conditions for Special Registers	26
Table 7-4:	Reset Conditions for All Registers	26
Table 7-5:	$\overline{TO}/\overline{PD}$ Status After Reset.....	30
Table 7-6:	Events Affecting $\overline{TO}/\overline{PD}$ Status Bits	30
Table 8-1:	OPCODE Field Descriptions	33
Table 8-2:	Instruction Set Summary	34
Table 9-1:	PICMASTER Probe Specification	46
Table 9-2:	Development System Packages	49
Table 10-1:	External Clock Timing Requirements - PIC16C52	55
Table 10-2:	CLKOUT and I/O Timing Requirements - PIC16C52	56
Table 10-3:	Reset and Device Reset Timer - PIC16C52 ..	57
Table 10-4:	Timer0 Clock Requirements - PIC16C52.....	58
Table 11-1:	RC Oscillator Frequencies	59
Table 11-2:	Input Capacitance for PIC16C52	64
Table C-1:	PIC16C5X Family of Devices.....	73
Table C-2:	PIC16C62X Family of Devices.....	74
Table C-3:	PIC16C6X Family of Devices.....	75
Table C-4:	PIC16C7X Family of Devices.....	76
Table C-5:	PIC16C8X Family of Devices.....	77
Table C-6:	PIC17CXX Family of Devices	78
Table C-7:	Pin Compatible Devices	79

CONNECTING TO MICROCHIP BBS

Connect worldwide to the Microchip BBS using the CompuServe® communications 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 BBS, except toll charge to CompuServe access number, where applicable. You do not need to be a CompuServe member to take advantage of this connection (you never actually log in to CompuServe).

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 allows multiple users at baud rates up to 14,400 bps.

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 **<ENTER>** and a garbage string will appear because CompuServe is expecting a 7E1 setting.
4. Type **+**, depress **<ENTER>** and Host Name: will appear.
5. Type **MCHIPBBS**, depress **< ENTER>** and you will be connected to the Microchip BBS.

In the United States, to find CompuServe's 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 **< ENTER>** and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 457-1550 for your local CompuServe number.

ACCESS TO THE INTERNET

Microchip's current WWW address is listed on the back page of this data sheet under Worldwide Sales & Service - Americas - Corporate Office.

Trademarks:

PICMASTER and PICSTART are trademarks of Microchip Technology Incorporated. PIC is a registered trademark of Microchip Technology Incorporated in the U.S.A.

PRO MATE, *fuzzyLAB*, the Microchip logo and name are trademarks of Microchip Technology Incorporated.

fuzzyTECH is a registered trademark of Inform Software Corporation.

I²C is a trademark of Philips Corporation.

IBM, IBM PC-AT are registered trademarks of International Business Machines Corporation.

Pentium is a trademark of Intel Corporation.

MS-DOS and Microsoft Windows are registered trademarks of Microsoft Corporation. Windows is a trademark of Microsoft Corporation.

CompuServe is a registered trademark of CompuServe Incorporated.

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

PIC16C52

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
RE: Reader Response

Total Pages Sent _____

From: Name _____
Company _____
Address _____
City / State / ZIP / Country _____
Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional): _____

Would you like a reply? ____ Y ____ N

Device: **PIC16C52**

Literature Number: **DS30254B**

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? _____

PIC16C52 PRODUCT IDENTIFICATION SYSTEM

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

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

<u>PART NO.</u>	<u>-XX</u>	<u>X</u>	<u>/XX</u>	<u>XXX</u>
Device	Oscillator Type	Temperature Range	Package	Pattern

Device	PIC16C52, PIC16C52T ⁽²⁾			
Frequency Range	04	= 4 MHz		
Temperature Range	b ⁽¹⁾	= 0°C to +70°C (Commercial)		
	I	= -40°C to +85°C (Industrial)		
Package	P	= PDIP		
	SO	= SOIC (Gull Wing, 300 mil body)		
Pattern	3-digit Pattern Code for QTP (blank otherwise)			

Examples:

a) PIC16C52 - 04/PXXX = "RC" oscillator, commercial temp., PDIP, QTP pattern.

b) PIC16C52 - 04I/SO = "XT" oscillator, industrial temp., SOIC (OTP device)

Note 1: b = blank
2: T = in tape and reel - SOIC packages only.

Sales and Support

Products supported by a preliminary Data Sheet may possibly 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 below)
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.

For latest version information and upgrade kits for Microchip Development Tools, please call 1-800-755-2345 or 1-602-786-7302.

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 455
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

ASIA/PACIFIC

Hong Kong

Microchip Technology
Unit No. 3002-3004, Tower 1
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 0 1628 851077 Fax: 44 0 1628 850259

France

Arizona Microchip Technology SARL
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: 39 039 689 9939 Fax: 39 039 689 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

12/04/95



MICROCHIP

All rights reserved. © 1995, Microchip Technology Incorporated, USA.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.