

## EPROM/ROM Memory Programming/Verify Specification

This document includes the programming specifications for the following devices:

- PIC16C52      • PIC16C55      • PIC16C57
- PIC16C54      • PIC16C55A      • PIC16C57C
- PIC16C54A      • PIC16C56      • PIC16CR57B
- PIC16C54B      • PIC16C56A      • PIC16CR57C
- PIC16C54C      • PIC16CR56A      • PIC16C58A
- PIC16CR54A      • PIC16C58B
- PIC16CR54B      • PIC16CR58A
- PIC16CR54C      • PIC16CR58B

## INTRODUCTION

### Overview

The PIC16C5X Series is a family of single-chip CMOS microcontrollers with on-chip EPROM for program storage. The programming specification also applies to ROM products for verification only.

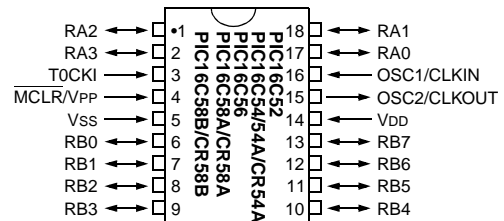
Due to the special architecture of these microcontrollers (12-bit wide instruction word) and the low pin counts (starting at 18 pins), the EPROM programming methodology is different from that of standard (byte-wide) EPROMs (e.g., 27C256).

The PIC16C5X Series can be programmed by applying the 12-bit wide data word to the 12 available I/O pins while the address is generated by the on-chip Program Counter. The MCLR/VPP pin provides the programming supply voltage (VPP). Programming/verify chip enable is controlled by the T0CKI pin while the OSC1 pin controls the Program Counter.

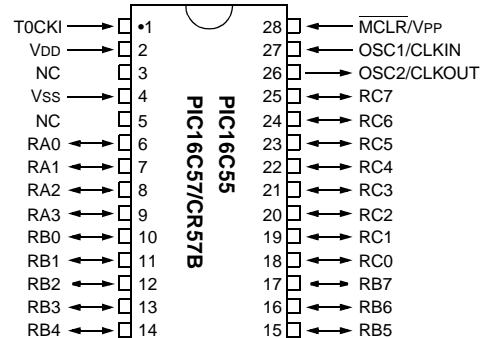
This document describes all the programming details of the PIC16C5X Series and the requirements for programming equipment to be used from programming prototypes in the engineering lab up to high volume programming on the factory floor.

### Pin Diagrams

#### PDIP, SOIC, Windowed Cerdip



#### PDIP, SOIC, Windowed Cerdip



### PIN DESCRIPTIONS (DURING PROGRAMMING): PIC16C52/54/54A/54B/55/56/57/58A/58B

Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
T0CKI	PROG/VER	I	Program pulse input/verify pulse input
RA0 - RA3	D0 - D3	I/O	Data input/output
RB0 - RB7	D4 - D11	I/O	Data input/output
OSC1	INCP	I	Increment Program Counter input
MCLR/VPP	VPP	P	Programming Power
VDD	VDD	P	Power Supply
Vss	Vss	P	Ground

Legend: I = Input, O = Output, P = Power

## 1.0 PROGRAM/VERIFY MODES

The PIC16C5X Series uses the internal Program Counter (PC) to generate the EPROM address. VPP is supplied through the  $\overline{\text{MCLR}}$  pin.

The T0CKI pin acts as chip enable, alternating between programming and verifying.

The OSC1 pin is used for incrementing the PC.

Data is applied to, or can be read on PORTA and PORTB (MSB on RB7, LSB on RA0).

The programming/verify mode is entered by raising the level on the  $\overline{\text{MCLR}}$  pin from  $V_{IL}$  to  $V_{HH}$  (= VPP) while the T0CKI pin is held at  $V_{IH}$  and the OSC1 pin is held at  $V_{IL}$ .

The Program Counter now has the value "0xFFFF", because  $\overline{\text{MCLR}}$  was at  $V_{IL}$  before. This condition selects the configuration word as the very first EPROM location to be accessed after entering the program/verify mode.

Since the  $\overline{\text{MCLR}}$  pin was initially at  $V_{IL}$ , the device is in the reset state (the I/O pins are in the reset state).

Incrementing the PC once (by pulsing the OSC1 pin) selects location "0x000" of the user program memory. Afterwards all other memory locations from 001h through end of memory can be addressed by incrementing the PC.

If the Program Counter has reached the last address of the user memory area (e.g. "0x1FF" for the PIC16C54), and is incremented again, the on-chip special EPROM area will be addressed. (See Figure 1-2 to determine where the special EPROM area is located for the various PIC16C5X devices).

### 1.1 Program/Verify Without PC Increment

After entering the program/verify mode, pulsing the T0CKI pin LOW programs the data present on PORTA and PORTB into the memory location selected by the Program Counter. The duration of the T0CKI LOW time determines the length of the programming pulse.

Pulsing the T0CKI pin LOW again without changing the signals on  $\overline{\text{MCLR}}$  and OSC1 puts the contents of the selected memory location out on PORTA and PORTB for verification of a successful programming cycle. This verification pulse on T0CKI can be much shorter than the programming pulse. If the programming was not successful, T0CKI can be pulsed LOW again to apply another programming pulse, followed again by a shorter T0CKI LOW pulse for another verification cycle.

This sequence can be repeated as many times as required until the programming is successful.

### 1.2 Verify with PC Increment

If a verification cycle shows that programming was successful, the Program Counter can be incremented by keeping the T0CKI input at a HIGH level while pulsing the OSC1 input HIGH. When both T0CKI and OSC1 are HIGH, the contents of the selected memory location is put out on Ports A and B (= Verify). The falling edge of OSC1 will increment the Program Counter.

A fast VERIFY- ONLY with automatic increment of the PC can be performed by entering the program/verify mode as described above and then clocking the OSC1 input. If OSC1 is HIGH, the selected memory location is output on Ports A and B, while the falling edge of OSC1 will increment the Program Counter. Thus, the first memory location to be verified after entering the program/verify mode, is the configuration word. The next location is 000h followed by 001h and so on. The program memory location "N" can be reached by generating "N + 1" falling edges on OSC1. When OSC1 is brought HIGH again, the contents of address "N" are output on Ports A and B as long as OSC1 stays HIGH.

### 1.3 Programming/Verifying Configuration Word

The configuration word is logically mapped at program memory location "0xFFFF". The PC points to the configuration word after  $\overline{\text{MCLR}}$  pin goes from LOW to  $V_{HH}$  (HIGH). The configuration word can be programmed or verified using the techniques described in Section 1.1 and Section 1.2.

If PC is incremented, the next location it will point to is "0x000" in user memory. Incrementing PC 4096 times will not allow the user to point to the configuration word. The only way to point to it again is to reset and re-enter program mode.

## 1.4 Programming Method

The programming technique is described in the following section. It is designed to guarantee good programming margins. It does, however, require a variable power supply for VCC.

### 1.4.1 PROGRAMMING METHOD DETAILS

Essentially, this technique includes the following steps:

1. Perform blank check at  $V_{DD} = V_{DDmin}$ . Report failure. The device may not be properly erased.
2. Program location with pulses and verify after each pulse at  $V_{DD} = V_{DDP}$ :  
where  $V_{DDP} = V_{DD}$  range required during programming (4.75V - 5.25V).
  - a) Programming condition:  
 $V_{PP} = 13.0V$  to 13.25V  
 $V_{DD} = V_{DDP} = 4.75V - 5.25V$   
 $V_{PP}$  must be  $\geq V_{DD} + 7.25V$  to keep "programming mode" active.
  - b) Verify condition:  
 $V_{DD} = V_{DDP}$   
 $V_{PP} \geq V_{DD} + 7.5V$  but not to exceed 13.25V  
 If location fails to program after "N" pulses, (suggested maximum program pulses of 8) then report error as a programming failure.

**Note:** Device must be verified at minimum and maximum specified operating voltages as specified in the data sheet.

3. Once location passes "Step 2", apply 11X over-programming, i.e., apply eleven times the number of pulses that were required to program the location. This will guarantee a solid programming margin. The overprogramming should be made "software programmable" for easy updates.
4. Program all locations.
5. Verify all locations (using speed verify mode) at  $V_{DD} = V_{DDmin}$
6. Verify all locations at  $V_{DD} = V_{DDmax}$   
 $V_{DDmin}$  is the minimum operating voltage spec. for the part.  $V_{DDmax}$  is the maximum operating voltage spec. for the part.

### 1.4.2 SYSTEM REQUIREMENTS

Clearly, to implement this technique, the most stringent requirements will be that of the power supplies:

**VPP:** VPP can be a fixed 13.0V to 13.25V supply. It must not exceed 14.0V to avoid damage to the pin and should be current limited to approximately 100mA.

**VDD:** 2.0V to 6.5V with 0.25V granularity. Since this method calls for verification at different VDD values, a programmable VDD power supply is needed.

**Current Requirement:** 100mA maximum

Microchip may release PIC16C5Xs in the future with different VDD ranges which make it necessary to have a programmable VDD.

It is important to verify an EPROM at the voltages specified in this method to remain consistent with Microchip's test screening. For example, a PIC16C5X specified for 4.75V - 5.25V should be tested for proper programming from 4.75V - 5.25V.

**Note:** Any programmer not meeting the programmable VDD requirement and the verify at  $V_{DDmax}$  and  $V_{DDmin}$  requirement may only be classified as "prototype" or "development" programmer but not a production programmer.

### 1.4.3 SOFTWARE REQUIREMENTS

Certain parameters should be programmable (and therefore easily modified) for easy upgrade.

- a) Pulse width
- b) Maximum number of pulses, current limit 8.
- c) Number of over-programming pulses: should be  $= (A \cdot N) + B$ , where N = number of pulses required in regular programming. In our current algorithm  $A = 11$ ,  $B = 0$ .

## 1.5 Programming Pulse Width

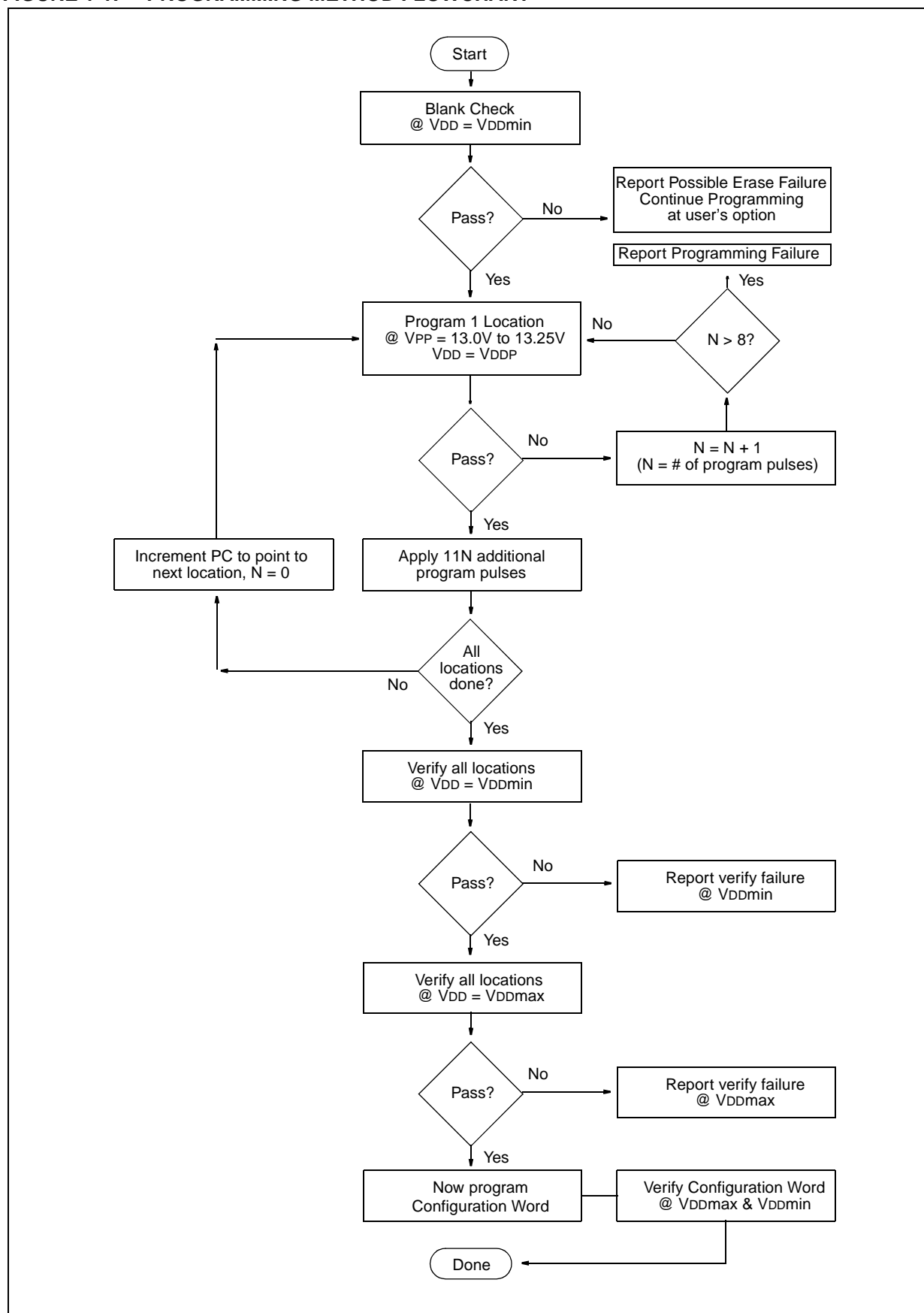
**Program Memory Cells:** When programming one word of EPROM, a programming pulse width (TPW) of 100  $\mu s$  is recommended.

The maximum number of programming attempts should be limited to 8 per word.

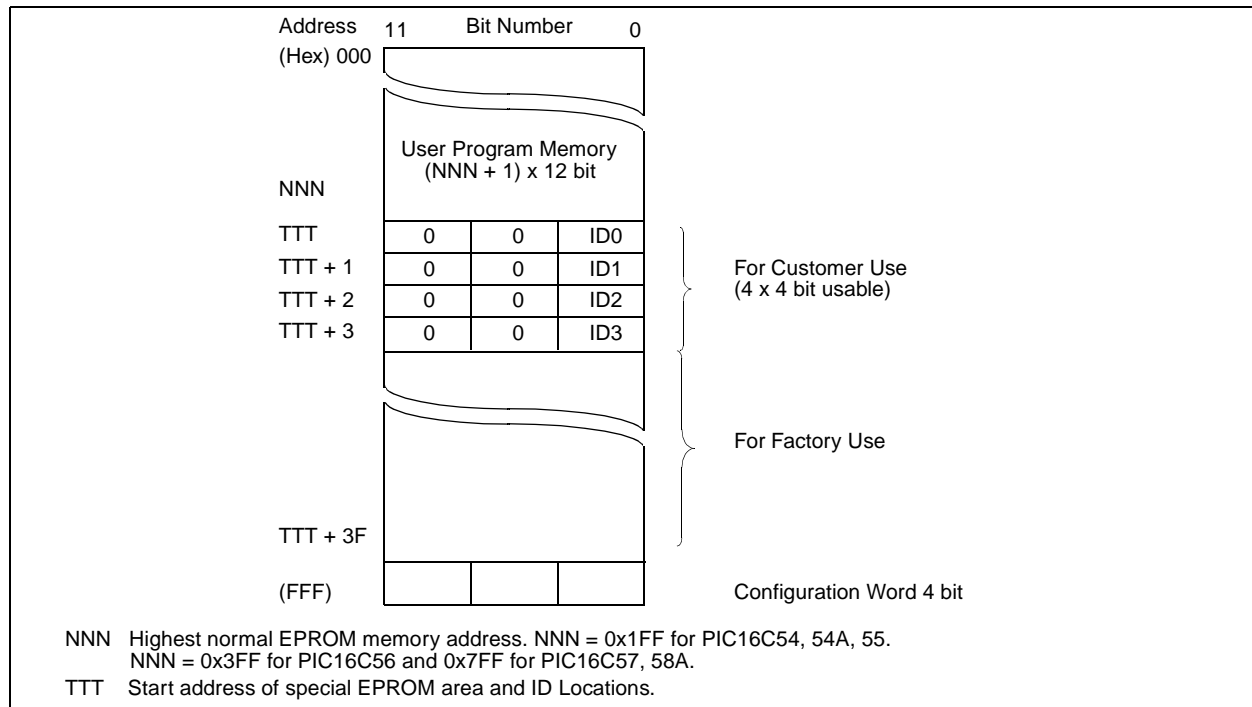
After the first successful verify, the same location should be over-programmed with 11X over-programming.

**Configuration Word:** The configuration word for oscillator selection, WDT (watchdog timer) disable and code protection, requires a programming pulse width (TPWF) of 10 ms. A series of 100  $\mu s$  pulses is preferred over a single 10 ms pulse.

FIGURE 1-1: PROGRAMMING METHOD FLOWCHART



**FIGURE 1-2: PIC16C5X SERIES PROGRAM MEMORY MAP IN PROGRAM/VERIFY MODE**



## 1.6 Special Memory Locations

The ID Locations area is only enabled if the device is in a test or programming/verify mode. Thus, in normal operation mode only the memory location 0x000 to 0xNNN will be accessed and the Program Counter will just roll over from address 0xNNN to 0x000 when incremented.

The configuration word can only be accessed immediately after MCLR going from V<sub>IL</sub> to V<sub>HH</sub>. The Program Counter will be set to all '1's upon MCLR = V<sub>IL</sub>. Thus, it has the value "0xFFFF" when accessing the configuration EPROM. Incrementing the Program Counter once by pulsing OSC1 causes the Program Counter to roll over to all '0's. Incrementing the Program Counter 4K times after reset (MCLR = V<sub>IL</sub>) does not allow access to the configuration EPROM.

### 1.6.1 CUSTOMER ID CODE LOCATIONS

Per definition, the first four words (address TTT to TTT + 3) are reserved for customer use. It is recommended that the customer use only the four lower order bits (bits 0 through 3) of each word and filling the eight higher order bits with '0's.

A user may want to store an identification code (ID) in the ID locations and still be able to read this code after the code protection bit was programmed. This is possible if the ID code is only four bits long per memory location, is located in the least significant nibble boundary of the 12-bit word, and the remaining eight bits are all '0's.

### EXAMPLE 1: CUSTOMER CODE 0xD1E2

The Customer ID code "0xD1E2" should be stored in the ID locations 200-203 like this:

```
200: 0000 0000 1101
201: 0000 0000 0001
202: 0000 0000 1110
203: 0000 0000 0010
```

Reading these four memory locations, even with the code protection bit programmed would still output on Port A the bit sequence "1101", "0001", "1110", "0010" which is "0xD1E2".

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

# PIC16C5XX

## 2.0 CONFIGURATION WORD

The configuration word is the very first memory location which is accessed after entering the program/verify mode of the PIC16C5X. It contains the two bits for the selection of the oscillator type, the watchdog timer enable bit, and the code protection bit. All other bits (4 through 11) are read as '1's.

One-Time-Programmable (OTP) devices may have the oscillator configuration bits "FOSC0" and "FOSC1" set by the factory and are tested accordingly. Therefore, it is essential that the inputs RA0 and RA1 are held at '1's when programming the "WDTE" and/or the "CP" bit of the configuration word. Otherwise, the factory tested and selected oscillator configuration could be overwritten and the functionality of the device is not guaranteed any more.

**FIGURE 2-1: CONFIGURATION WORD FOR PIC16CR54A/C54B/CR54B/C54C/CR54C/C55A/C56A/CR56A/C57C/CR57B/CR57C/C58B/CR58A/CR58B**

CP	CP	CP	CP	CP	CP	CP	CP	CP	WDTE	FOSC1	FOSC0	Register: CONFIG Address: FFFh
bit11	10	9	8	7	6	5	4	3	2	1	bit0	
bit 11-3: <b>CP:</b> Code protection bits 1 = Code protection off 0 = Code protection on												
bit 2: <b>WDTE:</b> Watchdog timer enable bit 1 = WDT enabled 0 = WDT disabled												
bit 1-0: <b>FOSC1:FOSC0:</b> Oscillator selection bits 11 = RC oscillator 10 = HS oscillator 01 = XT oscillator 00 = LP oscillator												

**FIGURE 2-2: CONFIGURATION WORD FOR PIC16C52/C54/C54A/C55/C56/C57/C58A**

—	—	—	—	—	—	—	—	CP	WDTE	FOSC1	FOSC0	Register: CONFIG Address: FFFh
bit11	10	9	8	7	6	5	4	3	2	1	bit0	
bit 11-4: <b>Unimplemented:</b> Read as '0'												
bit 3: <b>CP:</b> Code protection bit. 1 = Code protection off 0 = Code protection on												
bit 2: <b>WDTE:</b> Watchdog timer enable bit (not implemented on PIC16C52) 1 = WDT enabled 0 = WDT disabled												
bit 1-0: <b>FOSC1:FOSC0:</b> Oscillator selection bits <sup>(2)</sup> 11 = RC oscillator 10 = HS oscillator 01 = XT oscillator 00 = LP oscillator												
<b>Note:</b> PIC16C52 supports XT and RC oscillator only. PIC16LV54A supports XT, RC and LP oscillator only. PIC16LV58A supports XT, RC and LP oscillator only.												

## 3.0 CODE PROTECTION

The program code written into the EPROM can be protected by writing to the “CP” bit of the configuration word. All memory locations starting at 0x40 and above are protected against programming. It is still possible to program locations 0x00 through 0x3F, the ID locations, and the configuration word.

**Note:** Locations [0x000 : 0x03F] are not secure after code protection.

### 3.1 Programming Locations 0x000 to 0x03F after Code Protection

In a code protected part, these locations will program with the exception of the PIC16CRXX devices. They will read back scrambled data, with the exception of PIC16CR54A and PIC16CR58A. In any event, the programmer cannot verify the device once it is code protected.

In code protected parts, the contents of the program memory cannot be read out in a way that the program code can be reconstructed. A location when read out will read as: 0000 0000 xxxx where xxxx is the XOR of the three nibbles.

For example, if the memory location contains 0xC04 (movlw 4), after code protection the output will be 0x008.

In addition, all memory locations starting at 0x40 and above are protected against programming. It is still possible to program locations 0x000 through 0x03F and the configuration word. However, performing a verify with activated code protection logic puts a 4-bit wide “checksum” on PORTA while the 8-bits of PORTB are read as '0's. The checksum is computed as follows:

The four high order bits of an instruction word are “XOR’ed” with the four middle and the four low order bits, and the result is transferred to PORTA. All memory locations are affected.

To program location 0x000 to 0x03F in a code protected part, the programmer should program one nibble at a time and verify the result through the XOR’ed output. For example, to program a location with 0xA93, first program the location with 0xFF3, verify checksum to be 0x003; then program the location with 0xF93 and verify the XOR’ed output to be 0x00C and finally program the location with 0xA93 and verify the read-out to be 0x006.

### 3.2 Embedding Configuration Word and ID Information in the Hex File

To allow portability of code, a PIC16C5X programmer is required to read the configuration word and ID locations from the hex file when loading the hex file. If configuration word information was not present in the hex file then a simple warning message may be issued. Similarly, while saving a hex file, all configuration word and ID information must be included. Configuration word should have the address of 0xFFFF. ID locations are mapped at addresses described in Section 1.6.1 and Table 3-1. An option to not include this information may be provided.

Microchip Technology Inc. feels strongly that this feature is important for the benefit of the end customer.

**TABLE 3-1: CONFIGURATION WORD**  
**PIC16C52 (CP enable pattern: XXXXXXXX0XXX)**

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x17F]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

**PIC16C54 (CP enable pattern: XXXXXXXX0XXX)**

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

# PIC16C5XX

## PIC16C54A (CP enable pattern: XXXXXXXX0XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR54A (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled	Read Unscrambled
ID Words [0x800 : 0x803]	Read Unscrambled	Read Unscrambled
[0x040 : 0x7FF]	Read Disabled	Read Unscrambled
[0x000 : 0x03F]	Read Unscrambled	Read Unscrambled

## PIC16C54B (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read 0's, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR54B (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled	Read Unscrambled
ID Words [0x800 : 0x803]	Read Unscrambled	Read Unscrambled
[0x040 : 0x7FF]	Read 0's	Read Unscrambled
[0x000 : 0x03F]	Read Unscrambled	Read Unscrambled

## PIC16C54C (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read 0's Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR54C (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled	Read Unscrambled
ID Words [0x800 : 0x803]	Read Unscrambled	Read Unscrambled
[0x040 : 0x7FF]	Read 0's	Read Unscrambled
[0x000 : 0x03F]	Read Unscrambled	Read Unscrambled



## PIC16C55 (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C55A (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C56 (CP enable pattern: XXXXXXXX0XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x400 : 0x403]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x3FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C56A (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x400 : 0x403]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x3FF]	Read 0's	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR56A (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x400 : 0x403]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x3FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C57 (CP enable pattern: XXXXXXXX0XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C57C (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

# PIC16C5XX

## PIC16CR57B (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR57C (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16C58A (CP enable pattern: XXXXXXXX0XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR58A (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled	Read Unscrambled
ID Words [0x800 : 0x803]	Read Unscrambled	Read Unscrambled
[0x040 : 0x7FF]	Read Disabled	Read Unscrambled
[0x000 : 0x03F]	Read Unscrambled	Read Unscrambled

## PIC16C58B (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x800 : 0x803]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x7FF]	Read 0s, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Unscrambled, Write Enabled	Read Unscrambled, Write Enabled

## PIC16CR58B (CP enable pattern: 00000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFFF)	Read Unscrambled	Read Unscrambled
ID Words [0x800 : 0x803]	Read Unscrambled	Read Unscrambled
[0x040 : 0x7FF]	Read 0's	Read Unscrambled
[0x000 : 0x03F]	Read Unscrambled	Read Unscrambled

Legend: X = Don't care

## 3.3 Checksum

### 3.3.1 CHECKSUM CALCULATIONS

Checksum is calculated by reading the contents of the PIC16C5X memory locations and adding up the opcodes up to the maximum user addressable location, e.g., 0x1FF for the PIC16C54/55. Any carry bits exceeding 16-bits are neglected. Finally, the configuration word (appropriately masked) is added to the checksum. Checksum computation for each member of the PIC16C5X devices is shown in Table .

The checksum is calculated by summing the following:

- The contents of all program memory locations
- The configuration word, appropriately masked
- Masked ID locations (when applicable)

The least significant 16 bits of this sum is the checksum.

The following table describes how to calculate the checksum for each device. Note that the checksum calculation differs depending on the code protect setting. Since the program memory locations read out differently depending on the code protect setting, the table describes how to manipulate the actual program memory values to simulate the values that would be read from a protected device. When calculating a checksum by reading a device, the entire program memory can simply be read and summed. The configuration word and ID locations can always be read.

Note that some older devices have an additional value added in the checksum. This is to maintain compatibility with older device programmer checksums.

**TABLE 3-1: CHECKSUM COMPUTATION**

Device	Code Protect	Checksum*	Blank Value	0x723 at 0 and max address
PIC16C52	OFF ON	SUM[0x000:0x17F] + CFGW & 0x00B SUM_XOR4[0x000:0x17F] + CFGW & 0x00B	0xFE8B 0x1683	0xECD3 0x1671
PIC16C54	OFF ON	SUM[0x000:0x1FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x1FF] + CFGW & 0x00F	0x0DFF 0x1E07	0xFC47 0x1DF5
PIC16C54A	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM_XOR4[0x000:0x1FF] + CFGW & 0x00F	0x0DFF 0x1E07	0xFC47 0x1DF5
PIC16C54B	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x00:0x3F] + CFGW & 0xFFFF + SUM_1D	0x0DFF 0x0DC6	0xFC47 0xF332
PIC16C54C	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x00:0x3F] + CFGW & 0xFFFF + SUM_1D	0x0DFF 0x0DC6	0xFC47 0xF332
PIC16CR54A	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x00:0x3F] + CFGW & 0xFFFF + SUM_1D	0x0DFF 0x0DC6	0xFC47 0xF332
PIC16CR54B	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x00:0x3F] + CFGW & 0xFFFF + SUM_1D	0x0DFF 0x0DC6	0xFC47 0xF332
PIC16CR54C	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x00:0x3F] + CFGW & 0xFFFF + SUM_1D	0x0DFF 0x0DC6	0xFC47 0xF332
PIC16C55	OFF ON	SUM[0x000:0x1FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x1FF] + CFGW & 0x00F	0x0DFF 0x1E07	0xFC47 0x1DF5
PIC16C55A	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFFF SUM[0x000:0x3F] + CFGW & 0xFFFF + SUM_ID	0x0DFF 0x0DC6	0xFC47 0xF322
PIC16C56	OFF ON	SUM[0x000:0x3FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x3FF] + CFGW & 0x00F	0x0BFF 0x3C07	0xFA47 0x3BF5
PIC16C56A	OFF ON	SUM[0x000:0x3FF] + CFGW & 0xFFFF SUM_XOR4[0x000:0x3F] + CFGW & 0x00F + SUM_ID	0x0BFF 0x0BC6	0xFA47 0xF132

Legend: CFGW = Configuration Word

SUM[a:b] = Sum of locations a through b inclusive

SUM\_XOR4[a:b] = XOR of the four high order bits with the four middle and the four low of memory location order bits summed over the locations a through b inclusive. For example, location\_a = 0x123 and location\_b = 0x456, then SUM\_XOR [location\_a : location\_b] = 0x0007.

SUM\_ID = ID locations masked by 0xF then made into a 16-bit value with ID0 as the most significant nibble.

For example, ID0 = 0x1, ID1 = 0x2, ID3 = 0x3, ID4 = 0x4, then SUM\_ID = 0x1234.

\*Checksum = Sum of all individual expressions **modulo** [0xFFFF]

+ = Addition

& = Bitwise AND

# PIC16C5XX

**TABLE 3-1: CHECKSUM COMPUTATION (CONTINUED)**

Device	Code Protect	Checksum*	Blank Value	0x723 at 0 and max address
PIC16CR56A	OFF ON	SUM[0x000:0x1FF] + CFGW & 0xFFF SUM[0x000:0x3F] + CFGW & 0xFFF + SUM_ID	0x0BFF 0x0BC6	0xFA47 0xF132
PIC16C57	OFF ON	SUM[0x000:0x7FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x7FF] + CFGW & 0x00F	0x07FF 0x7807	0xF647 0x77F5
PIC16C57C	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x000:0x3F] + CFGW & 0xFFF + SUM_ID	0x07FF 0x07C6	0xF647 0xED32
PIC16CR57A	OFF ON	SUM[0x000:0x7FF] + CFGW & 0x00F + FF0 SUM_XOR4[0x00:0x7FF] + CFGW & 0xF	0x07FF 0x7807	0xF647 0x77F5
PIC16CR57B	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x00x:0x3F] + CFGW & 0xFFF + SUM_ID	0x07FF 0x07C6	0xF647 0xED32
PIC16CR57C	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x00x:0x3F] + CFGW & 0xFFF + SUM_ID	0x07FF 0x07C6	0xF647 0xED32
PIC16C58A	OFF ON	SUM[0x000:0x7FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x7FF] + CFGW & 0x00F	0x07FF 0x7807	0xF647 0x77F5
PIC16C58B	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x000:0x7FF] + CFGW & 0xFFF + SUM_ID	0x07FF 0x7C6	0xF647 0xED32
PIC16CR58A	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x000:0x3F] + CFGW & 0xFFF + SUM_ID	0x07FF 0x07C6	0xF647 0xED32
PIC16CR58B	OFF ON	SUM[0x000:0x7FF] + CFGW & 0xFFF SUM[0x000:0x3F] + CFGW & 0xFFF + SUM_ID	0x07FF 0x07C6	0xF647 0xED32

Legend: CFGW = Configuration Word

SUM[a:b] = Sum of locations a through b inclusive

SUM\_XOR4[a:b] = XOR of the four high order bits with the four middle and the four low of memory location order bits summed over the locations a through b inclusive. For example, location\_a = 0x123 and location\_b = 0x456, then SUM\_XOR [location\_a : location\_b] = 0x0007.

SUM\_ID = ID locations masked by 0xF then made into a 16-bit value with ID0 as the most significant nibble.

For example, ID0 = 0x1, ID1 = 0x2, ID3 = 0x3, ID4 = 0x4, then SUM\_ID = 0x1234.

\*Checksum = Sum of all individual expressions **modulo** [0xFFFF]

+ = Addition

& = Bitwise AND

## 4.0 PROGRAM/VERIFY MODE ELECTRICAL CHARACTERISTICS

### 4.1 DC Program Characteristics)

**TABLE 4-1: DC CHARACTERISTICS (TA = +10°C TO +40°C) (25°C IS RECOMMENDED)**

Parameter No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
PD1	VDDP	Supply voltage during programming	4.75	5.0	5.25	V	Note 1
PD2	IDDP	Supply Current (from VDD)			25.0	mA	VDD = 5.0V, Fosc1 = 5MHz
PD3	VDDV	Supply Voltage during verify	VDDmin		VDDmax		
PD5	VHH2	Voltage on MCLR during programming	12.5		13.5	V	
PD6	IHH	Supply current from programming voltage source			100	mA	
PD7	IHH2	Current into MCLR pin during programming (T0CKI=0)		10.0	25.0	mA	VHH = 13.5V, VDD = 6.0V
PD8	VIL	Input Low Voltage	VSS		0.15VDD	V	
PD9	VIH	Input High Voltage	0.85VDD	5.0	VDD	V	

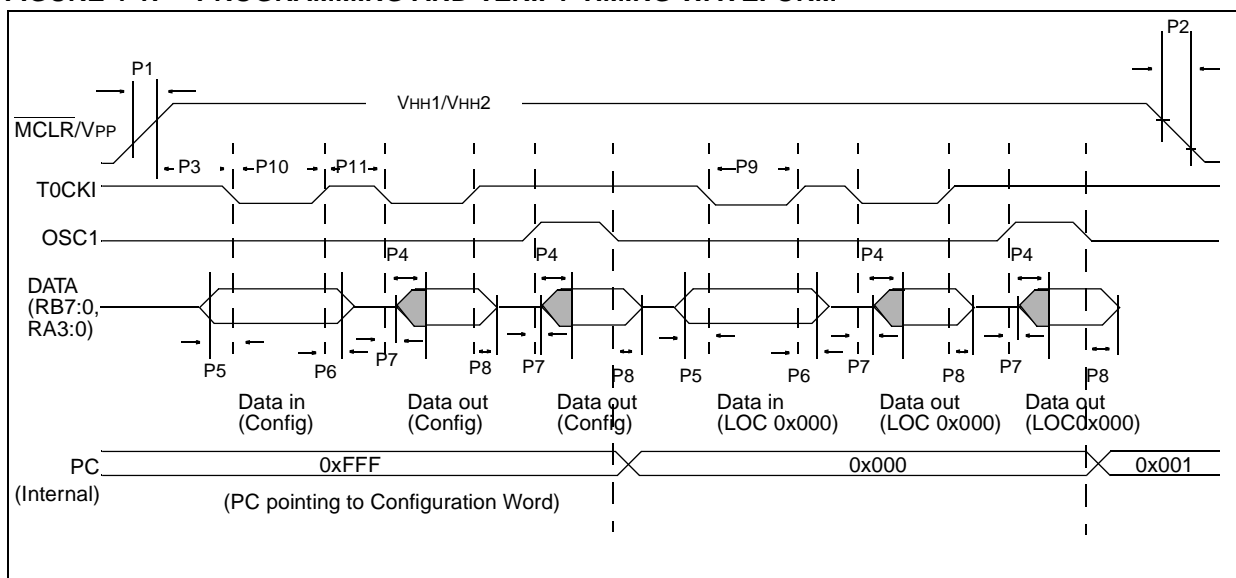
Note 1: Device must be verified at minimum and maximum operating voltages specified in the data sheet.

### 4.2 AC Program and Test Mode Characteristics

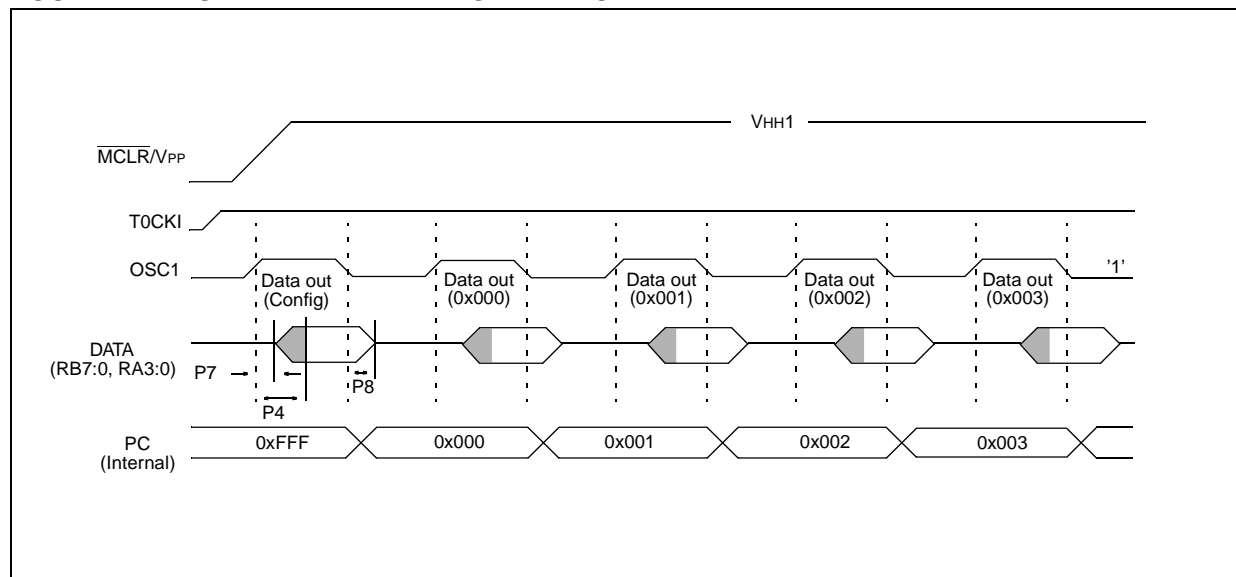
**TABLE 4-1: AC CHARACTERISTICS (TA = +10°C TO +40°C, VDD = 5.0V ± 5%) (25°C IS RECOMMENDED)**

Parameter No.	Symbol	Characteristics	Min	Typ	Max	Units	Conditions
P1	TR	MCLR Rise Time	0.15	1.0	8	μs	
P2	TF	MCLR Fall Time	0.5	2.0	8	μs	
P3	TPS	Program Mode Setup Time	1.0			μs	
P4	TACC	Data Access Time			250	ns	
P5	TDS	Data Setup Time	1.0			μs	
P6	TDH	Data Hold Time	1.0			μs	
P7	TOE	Output Enable Time	0		100	ns	
P8	TOZ	Output Disable Time	0		100	ns	
P9	TPW	Programming Pulse Width	10.0	100		μs	
P10	TPWF	Programming Pulse Width		10,000		μs	Configuration Word only
P11	TRC	Recovery Time	10.0			μs	
P12	FOSC	Frequency on OSC1	DC		5	MHz	For incrementing of the PC

**FIGURE 4-1: PROGRAMMING AND VERIFY TIMING WAVEFORM**



**FIGURE 4-2: SPEED VERIFY TIMING WAVEFORM**



**NOTES:**



## WORLDWIDE SALES AND SERVICE

### AMERICAS

#### Corporate Office

Microchip Technology Inc.  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-786-7200 Fax: 480-786-7277  
Technical Support: 480-786-7627  
Web Address: <http://www.microchip.com>

#### Atlanta

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

#### Boston

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

#### Chicago

Microchip Technology Inc.  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

#### Dallas

Microchip Technology Inc.  
4570 Westgrove Drive, Suite 160  
Addison, TX 75248  
Tel: 972-818-7423 Fax: 972-818-2924

#### Dayton

Microchip Technology Inc.  
Two Prestige Place, Suite 150  
Miamisburg, OH 45342  
Tel: 937-291-1654 Fax: 937-291-9175

#### Detroit

Microchip Technology Inc.  
Tri-Atria Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48334  
Tel: 248-538-2250 Fax: 248-538-2260

#### Los Angeles

Microchip Technology Inc.  
18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 949-263-1888 Fax: 949-263-1338

#### New York

Microchip Technology Inc.  
150 Motor Parkway, Suite 202  
Hauppauge, NY 11788  
Tel: 613-273-5305 Fax: 613-273-5335

#### San Jose

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

### AMERICAS (continued)

#### Toronto

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

### ASIA/PACIFIC

#### Hong Kong

Microchip Asia Pacific  
Unit 2101, Tower 2  
Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2-401-1200 Fax: 852-2-401-3431

#### Beijing

Microchip Technology, Beijing  
Unit 915, 6 Chaoyangmen Bei Dajie  
Dong Erhuan Road, Dongcheng District  
New China Hong Kong Manhattan Building  
Beijing 100027 PRC  
Tel: 86-10-85282100 Fax: 86-10-85282104

#### India

Microchip Technology Inc.  
India Liaison Office  
No. 6, Legacy, Convent Road  
Bangalore 560 025, India  
Tel: 91-80-229-0061 Fax: 91-80-229-0062

#### Japan

Microchip Technology Intl. Inc.  
Benex S-1 6F  
3-18-20, Shinyokohama  
Kohoku-Ku, Yokohama-shi  
Kanagawa 222-0033 Japan  
Tel: 81-45-471-6166 Fax: 81-45-471-6122

#### Korea

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

#### Shanghai

Microchip Technology  
RM 406 Shanghai Golden Bridge Bldg.  
2077 Yan'an Road West, Hong Qiao District  
Shanghai, PRC 200335  
Tel: 86-21-6275-5700 Fax: 86 21-6275-5060

### ASIA/PACIFIC (continued)

#### Singapore

Microchip Technology Singapore Pte Ltd.  
200 Middle Road  
#07-02 Prime Centre  
Singapore 188980  
Tel: 65-334-8870 Fax: 65-334-8850

#### Taiwan, R.O.C

Microchip Technology Taiwan  
10F-1C 207  
Tung Hua North Road  
Taipei, Taiwan, ROC  
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

### EUROPE

#### United Kingdom

Arizona Microchip Technology Ltd.  
505 Eskdale Road  
Winnersh Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44 118 921 5858 Fax: 44-118 921-5835

#### Denmark

Microchip Technology Denmark ApS  
Regus Business Centre  
Lautrup høj 1-3  
Ballerup DK-2750 Denmark  
Tel: 45 4420 9895 Fax: 45 4420 9910

#### France

Arizona Microchip Technology SARL  
Parc d'Activite du Moulin de Massy  
43 Rue du Saule Trapu  
Batiment A - 1er Etage  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

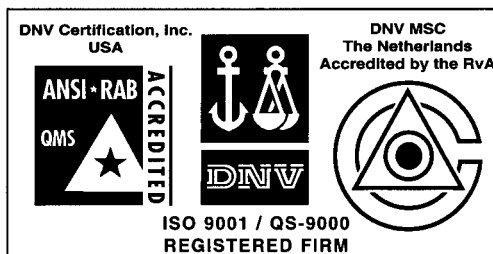
#### Germany

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

#### Italy

Arizona Microchip Technology SRL  
Centro Direzionale Colleoni  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-039-65791-1 Fax: 39-039-6899883

11/15/99



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELoc® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

All rights reserved. © 1999 Microchip Technology Incorporated. Printed in the USA. 11/99 Printed on recycled paper.

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