This document includes the programming specifications for the following devices:

- PIC16LF1566
- PIC16LF1567

1.0 OVERVIEW

The devices can be programmed using either the high-voltage In-Circuit Serial Programming™ (ICSP™) method or the low-voltage ICSP™ method.

1.1 Hardware Requirements

1.1.1 HIGH-VOLTAGE ICSP PROGRAMMING

In High-Voltage ICSP™ mode, these devices require two programmable power supplies: one for VDD and one for the MCLR/VPP pin.

1.1.2 LOW-VOLTAGE ICSP PROGRAMMING

In Low-Voltage ICSP™ mode, these devices can be programmed using a single VDD source in the operating range. The MCLR/VPP pin does not have to be brought to a different voltage, but can instead be left at the normal operating voltage.

1.1.2.1 Single-Supply ICSP Programming

The LVP bit in Configuration Word 2 enables single-supply (low-voltage) ICSP programming. The LVP bit defaults to a ‘1’ (enabled) from the factory. The LVP bit may only be programmed to ‘0’ by entering the High-Voltage ICSP mode, where the MCLR/VPP pin is raised to VIHH. Once the LVP bit is programmed to a ‘0’, only the High-Voltage ICSP mode is available and only the High-Voltage ICSP mode can be used to program the device.

Note 1: The High-Voltage ICSP mode is always available, regardless of the state of the LVP bit, by applying VIHH to the MCLR/VPP pin.

2: While in Low-Voltage ICSP mode, MCLR is always enabled, regardless of the MCLRE bit, and the port pin can no longer be used as a general purpose input.

1.2 Pin Utilization

Five pins are needed for ICSP™ programming. The pins are listed in Table 1-1.

**TABLE 1-1: PIN DESCRIPTIONS DURING PROGRAMMING**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>During Programming</th>
<th>Pin Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RB6</td>
<td>ICSPCLK</td>
<td>Clock Input – Schmitt Trigger Input</td>
</tr>
<tr>
<td>RB7</td>
<td>ICSPDAT</td>
<td>Data Input/Output – Schmitt Trigger Input</td>
</tr>
<tr>
<td>MCLR/VPP/RE3</td>
<td>Program/Verify mode</td>
<td>Program Mode Select/Programming Power Supply</td>
</tr>
<tr>
<td>VDD</td>
<td>VDD</td>
<td>Power Supply</td>
</tr>
<tr>
<td>VSS</td>
<td>VSS</td>
<td>Ground</td>
</tr>
</tbody>
</table>

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

Note 1: The programming high voltage is internally generated. To activate the Program/Verify mode, high voltage needs to be applied to MCLR input. Since the MCLR is used for a level source, MCLR does not draw any significant current.
2.0 DEVICE PINOUTS

The pin diagrams are shown in Figure 2-1 through Figure 2-4. The pins that are required for programming are listed in Table 1-1 and shown in bold lettering in the pin diagrams.

FIGURE 2-1: 28-PIN SPDIP, SOIC, SSOP DIAGRAM FOR PIC16LF1566

```
VPP/MCLR/RE3  28  RB7/ICSPDAT
RA0           27  RB6/ICSPCLK
RA1           26  RB5
RA2           25  RB4
RA3           24  RB3
RA4           23  RB2
RA5           22  RB1
Vss           21  RB0
RA7           20  Vdd
RA6           19  Vss
RC0           18  RC7
RC1           17  RC6
RC2           16  RC5
RC3           15  RC4
```

FIGURE 2-2: 28-PIN UQFN DIAGRAM FOR PIC16LF1566
FIGURE 2-3: 40-PIN PDIP DIAGRAM FOR PIC16LF1567

FIGURE 2-4: 40-PIN UQFN DIAGRAM FOR PIC16LF1567
FIGURE 2-5: 44-PIN TQFP DIAGRAM FOR PIC16LF1567
3.0 MEMORY MAP

The memory is broken into two sections: program memory and configuration memory. Only the size of the program memory changes between devices, the configuration memory remains the same.

FIGURE 3-1: PIC16LF1566/7 PROGRAM MEMORY MAPPING
3.1 User ID Location

A user may store identification information (user ID) in four designated locations. The user ID locations are mapped to 8000h-8003h. Each location is 14 bits in length. Code protection has no effect on these memory locations. Each location may be read with code protection enabled or disabled.

Note: MPLAB® IDE only displays the seventh Least Significant bits (LSb) of each user ID location, the upper bits are not read. It is recommended that only the seventh LSbs be used if MPLAB IDE is the primary tool used to read these addresses.

3.2 Device/Revision ID

The 14-bit device ID word is located at 8006h and the 14-bit revision ID is located at 8005h. These locations are read-only and cannot be erased or modified.

REGISTER 3-1: DEVICEID: DEVICE ID REGISTER (1)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DEV<13:8>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DEV<7:0>

Legend:

R = Readable bit
‘0’ = Bit is cleared ‘1’ = Bit is set x = Bit is unknown

bit 13-0  DEV<13:0>: Device ID bits

Refer to Table to determine what these bits will read on which device. A value of 3FFFh is invalid.

Note 1: This location cannot be written.

REGISTER 3-2: REVISIONID: REVISION ID REGISTER (1)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

REV<13:8>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

REV<7:0>

Legend:

R = Readable bit
‘0’ = Bit is cleared ‘1’ = Bit is set x = Bit is unknown

bit 13-0  REV<13:0>: Revision ID bits

These bits are used to identify the device revision.

Note 1: This location cannot be written.
3.3 Configuration Words

There are two Configuration Words, Configuration Word 1 (8007h) and Configuration Word 2 (8008h). The individual bits within these Configuration Words are used to enable or disable device functions such as the Brown-out Reset, code protection and Power-up Timer.

3.4 Calibration Words

The internal calibration values are factory calibrated and stored in Calibration Words 1 and 2 (8009h, 800Ah). The Calibration Words do not participate in erase operations. The device can be erased without affecting the Calibration Words.

<table>
<thead>
<tr>
<th>DEVICE</th>
<th>DEVICE ID</th>
<th>Revision ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16LF1566</td>
<td>3046h</td>
<td>2xxxh</td>
</tr>
<tr>
<td>PIC16LF1567</td>
<td>3047h</td>
<td>2xxxh</td>
</tr>
</tbody>
</table>
REGISTER 3-3: CONFIGURATION WORD 1

<table>
<thead>
<tr>
<th></th>
<th>U-1</th>
<th>U-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>U-1(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 13</td>
<td></td>
<td></td>
<td>CLKOUTEN</td>
<td>BOREN1</td>
<td>BOREN0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP</td>
<td>MCLRE</td>
<td>PWRTÉ</td>
<td>WDTE1</td>
<td>WDTE0</td>
<td>—</td>
<td>FOSC1</td>
<td>FOSC0</td>
</tr>
<tr>
<td>bit 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Legend:

- W = Writable bit
- '0' = Bit is cleared
- '1' = Bit is set
- x = Bit is unknown
- -n = Value at POR
- U = Unimplemented bit
- P = Programmable bit

bit 13-12: Unimplemented: Read as ‘1’

bit 11: CLKOUTEN: Clock Out Enable bit
- 1 = CLKOUT function is disabled. I/O or oscillator function on CLKOUT pin
- 0 = CLKOUT function is enabled on CLKOUT pin

bit 10-9: BOREN<1:0>: Brown-out Reset Enable bits(1)
- 11 = BOR enabled
- 10 = BOR enabled during operation and disabled in Sleep
- 01 = BOR controlled by SBOREN bit of the PCON register
- 00 = BOR disabled

bit 8(3): Unimplemented: Read as ‘1’

bit 7: CP: Code Protection bit(2)
- 1 = Program memory code protection is disabled
- 0 = Program memory code protection is enabled

bit 6: MCLRE: MCLR/VPP Pin Function Select bit
- If LVP bit = 1:
  - This bit is ignored.
- If LVP bit = 0:
  - 1 = MCLR/VPP pin function is MCLR; Weak pull-up enabled.
  - 0 = MCLR/VPP pin function is digital input; MCLR internally disabled; Weak pull-up under control of WPUA register.

bit 5: PWRTÉ: Power-up Timer Enable bit(1)
- 1 = PWRT disabled
- 0 = PWRT enabled

bit 4-3: WDTE<1:0>: Watchdog Timer Enable bit
- 11 = WDT enabled
- 10 = WDT enabled while running and disabled in Sleep
- 01 = WDT controlled by the SWDTEN bit in the WDTCN register
- 00 = WDT disabled

bit 2: Unimplemented: Read as ‘1’

bit 1-0: FOSC<1:0>: Oscillator Selection bits
- 11 = ECH: External Clock, High-Power mode: on CLKIN pin
- 10 = ECM: External Clock, Medium-Power mode: on CLKIN pin
- 01 = ECL: External Clock, Low-Power mode: on CLKIN pin
- 00 = INTOSC oscillator: I/O function on OSC1 pin

Note:
1: Enabling Brown-out Reset does not automatically enable Power-up Timer.
2: The entire program memory will be erased when the code protection is turned off.
3: This bit should be maintained as ‘1’ when programmed.
**REGISTER 3-4: CONFIGURATION WORD 2**

<table>
<thead>
<tr>
<th></th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>U-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEBUG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPBOR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BORV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STVREN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>bit 13</td>
<td>bit 12</td>
<td>bit 11</td>
<td>bit 10</td>
<td>bit 9</td>
<td>bit 8-2</td>
</tr>
<tr>
<td>U-1</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WRT1</td>
</tr>
<tr>
<td>U-1</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WRT0</td>
</tr>
</tbody>
</table>

**Legend:**
- **W** = Writable bit
- **’0’** = Bit is cleared
- **’1’** = Bit is set
- **x** = Bit is unknown
- **U** = Unimplemented bit
- **P** = Programmable Bit

**bit 13**
- **LVP**: Low-Voltage Programming Enable bit(1)
  - 1 = Low-voltage programming enabled
  - 0 = HV on MCLR/VPP must be used for programming

**bit 12**
- **DEBUG**: Debugger Mode bit
  - 1 = In-Circuit Debugger disabled, ICSPCLK and ICSPDAT pins are general purpose I/O pins
  - 0 = In-Circuit Debugger enabled, ICSPCLK and ICSPDAT pins are dedicated to the debugger

**bit 11**
- **LPBOR**: Low-Power BOR bit
  - 1 = Low-Power BOR is disabled
  - 0 = Low-Power BOR is enabled

**bit 10**
- **BORV**: Brown-out Reset Voltage Selection bit
  - 1 = Brown-out Reset Voltage (VBOR) set to 1.9V
  - 0 = Brown-out Reset Voltage (VBOR) set to 2.7V

**bit 9**
- **STVREN**: Stack Overflow/Underflow Reset Enable bit
  - 1 = Stack Overflow or Underflow will cause a Reset
  - 0 = Stack Overflow or Underflow will not cause a Reset

**bit 8-2**
- **Unimplemented**: Read as ‘1’

**bit 1-0**
- **WRT<1:0>**: Flash Memory Self-Write Protection bits
  - 8 kW Flash memory:
    - 11 = Write protection off
    - 10 = 0000h to 01FFh write-protected, 0200h to 1FFFh may be modified by PMCON control
    - 01 = 0000h to 0FFFh write-protected, 1000h to 1FFFh may be modified by PMCON control
    - 00 = 0000h to 1FFFh write-protected, no addresses may be modified by PMCON control

**Note 1:** The LVP bit cannot be programmed to ’0’ when Programming mode is entered via LVP.
4.0 PROGRAM/VERIFY MODE

In Program/Verify mode, the program memory and the configuration memory can be accessed and programmed in serial fashion. ICSPDAT and ICSPCLK are used for the data and the clock, respectively. All commands and data words are transmitted LSb first. Data changes on the rising edge of the ICSPCLK and latched on the falling edge. In Program/Verify mode both the ICSPDAT and ICSPCLK are Schmitt Trigger inputs. The sequence that enters the device into Program/Verify mode places all other logic into the Reset state. Upon entering Program/Verify mode, all I/Os are automatically configured as high-impedance inputs and the address is cleared.

4.1 High-Voltage Program/Verify Mode Entry and Exit

There are two different methods of entering Program/Verify mode via high-voltage:

• VPP – First entry mode
• VDD – First entry mode

4.1.1 VPP – FIRST ENTRY MODE

To enter Program/Verify mode via the VPP-first method the following sequence must be followed:

1. Hold ICSPCLK and ICSPDAT low. All other pins should be unpowered.
2. Raise the voltage on MCLR from 0V to VIHH.
3. Raise the voltage on VDD from 0V to the desired operating voltage.

The VPP-first entry prevents the device from executing code prior to entering Program/Verify mode. For example, the device will execute code when Configuration Word 1 has MCLR disabled (MCLRE = 0), the Power-up Timer is disabled (PWRTE = 0), the internal oscillator is selected (FOSC = 100), and ICSPCLK and ICSPDAT pins are driven by the user application. Since this may prevent entry, VPP-first entry mode is strongly recommended. See the timing diagram in Figure 8-2.

4.1.2 VDD – FIRST ENTRY MODE

To enter Program/Verify mode via the VDD-first method the following sequence must be followed:

1. Hold ICSPCLK and ICSPDAT low.
2. Raise the voltage on VDD from 0V to the desired operating voltage.
3. Raise the voltage on MCLR from VDD or below to VIHH.

The VDD-first method is useful when programming the device when VDD is already applied, for it is not necessary to disconnect VDD to enter Program/Verify mode. See the timing diagram in Figure 8-1.

4.1.3 PROGRAM/VERIFY MODE EXIT

To exit Program/Verify mode take MCLR to VDD or lower (VIL). See Figures 8-3 and 8-4.

4.2 Low-Voltage Programming (LVP) Mode

The Low-Voltage Programming mode allows devices to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Word 2 register is set to ‘1’, the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to ‘0’. This can only be done while in the High-Voltage Entry mode.

Entry into the Low-Voltage ICSP Program/Verify modes requires the following steps:

1. MCLR is brought to VIL.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

The key sequence is a specific 32-bit pattern, ‘0100 1101 0100 0011 0100 1000 0101 0000’ (more easily remembered as MCHP in ASCII). The device will enter Program/Verify mode only if the sequence is valid. The Least Significant bit of the Least Significant nibble must be shifted in first. Once the key sequence is complete, MCLR must be held at VIL for as long as Program/Verify mode is to be maintained.

For low-voltage programming timing, see Figure 8-8 and Figure 8-9. Exiting Program/Verify mode is done by no longer driving MCLR to VIL. See Figure 8-8 and Figure 8-9.

**Note:** To enter LVP mode, the LSB of the Least Significant nibble must be shifted in first. This differs from entering the key sequence on other parts.
4.3 Program/Verify Commands

The devices implement ten programming commands; each six bits in length. The commands are summarized in Table 4-1. Commands that have data associated with them are specified to have a minimum delay of TDLY between the command and the data. After this delay 16 clocks are required to either clock in or clock out the 14-bit data word. The first clock is for the Start bit and the last clock is for the Stop bit.

**TABLE 4-1: COMMAND MAPPING**

<table>
<thead>
<tr>
<th>Command</th>
<th>Mapping</th>
<th>Data/Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Configuration</td>
<td>x 0 0 0 0 0 0 0</td>
<td>00h, 0, data (14), 0</td>
</tr>
<tr>
<td>Load Data For Program Memory</td>
<td>x 0 0 0 0 1 0</td>
<td>02h, 0, data (14), 0</td>
</tr>
<tr>
<td>Read Data From Program Memory</td>
<td>x 0 0 1 0 0</td>
<td>04h, 0, data (14), 0</td>
</tr>
<tr>
<td>Increment Address</td>
<td>x 0 0 0 1 1 0</td>
<td>06h —</td>
</tr>
<tr>
<td>Reset Address</td>
<td>x 1 0 1 1 1 0</td>
<td>16h —</td>
</tr>
<tr>
<td>Begin Internally Timed Programming</td>
<td>x 0 1 0 0 0</td>
<td>08h —</td>
</tr>
<tr>
<td>Begin Externally Timed Programming</td>
<td>x 1 1 0 0 0</td>
<td>18h —</td>
</tr>
<tr>
<td>End Externally Timed Programming</td>
<td>x 0 1 0 1 0</td>
<td>0Ah —</td>
</tr>
<tr>
<td>Bulk Erase Program Memory</td>
<td>x 0 1 0 0 1</td>
<td>09h Internally Timed</td>
</tr>
<tr>
<td>Row Erase Program Memory</td>
<td>x 1 0 0 0 1</td>
<td>11h Internally Timed</td>
</tr>
</tbody>
</table>

Note: Externally timed writes are not supported for Configuration and Calibration bits. Any externally timed write to the Configuration or Calibration Word will have no effect on the targeted word.

The only way to get back to the program memory (address '0') is to exit Program/Verify mode or issue the Reset Address command after the configuration memory has been accessed by the Load Configuration command.

![FIGURE 4-1: LOAD CONFIGURATION](image)

4.3.2 LOAD DATA FOR PROGRAM MEMORY

The Load Data for Program Memory command is used to load one 14-bit word into the data latches. The word programs into program memory after the Begin Internally Timed Programming or Begin Externally Timed Programming command is issued (see Figure 4-2).
4.3.3 READ DATA FROM PROGRAM MEMORY

The Read Data from Program Memory command will transmit data bits out of the program memory map currently accessed, starting with the second rising edge of the clock input. The ICSPDAT pin will go into Output mode on the first falling clock edge, and it will revert to Input mode (high-impedance) after the 16th falling edge of the clock. If the program memory is code-protected (CP), the data will be read as zeros (see Figure 4-3).

4.3.4 INCREMENT ADDRESS

The address is incremented when this command is received. It is not possible to decrement the address. To reset this counter, the user must use the Reset Address command or exit Program/Verify mode and re-enter it. If the address is incremented from address 7FFFh, it will wrap-around to location 0000h. If the address is incremented from FFFPh, it will wrap-around to location 8000h.
4.3.5  RESET ADDRESS

The Reset Address command will reset the address to 0000h, regardless of the current value. The address is used in
program memory or the configuration memory.

4.3.6  BEGIN INTERNALLY TIMED PROGRAMMING

A Load Configuration or Load Data for Program Memory command must be given before every Begin Programming
command. Programming of the addressed memory will begin after this command is received. An internal timing
mechanism executes the write. The user must allow for the program cycle time, TPINT, for the programming to complete.

The End Externally Timed Programming command is not needed when the Begin Internally Timed Programming is used
to start the programming.

The program memory address that is being programmed is not erased prior to being programmed.
4.3.7 BEGIN EXTERNALLY TIMED PROGRAMMING

A Load Configuration or Load Data for Program Memory command must be given before every Begin Programming command. Programming of the addressed memory will begin after this command is received. To complete the programming the End Externally Timed Programming command must be sent in the specified time window defined by TPEXT (see Figure 4-7).

Externally timed writes are not supported for Configuration and Calibration bits. Any externally timed write to the Configuration or Calibration Word will have no effect on the targeted word.

**FIGURE 4-7: BEGIN EXTERNALLY TIMED PROGRAMMING**

4.3.8 END EXTERNALLY TIMED PROGRAMMING

This command is required after a Begin Externally Timed Programming command is given. This command must be sent within the time window specified by TPEXT after the Begin Externally Timed Programming command is sent.

After sending the End Externally Timed Programming command, an additional delay (TDIS) is required before sending the next command. This delay is longer than the delay ordinarily required between other commands (see Figure 4-8).

**FIGURE 4-8: END EXTERNALLY TIMED PROGRAMMING**

4.3.9 BULK ERASE PROGRAM MEMORY

The Bulk Erase Program Memory command performs two different functions dependent on the current state of the address.

Address 0000h-7FFFh:
- Program Memory is erased
- Configuration Words are erased

Address 8000h-8008h:
- Program Memory is erased
- Configuration Words are erased
- User ID Locations are erased

A Bulk Erase Program Memory command should not be issued when the address is greater than 8008h.

After receiving the Bulk Erase Program Memory command the erase will not complete until the time interval, TERAB, has expired.

**Note:** The code protection Configuration bit (CP) has no effect on the Bulk Erase Program Memory command.
4.3.10 ROW ERASE PROGRAM MEMORY

The Row Erase Program Memory command will erase an individual row. Refer to Table 4-2 for row sizes of specific devices and the PC bits used to address them. If the program memory is code-protected, the Row Erase Program Memory command will be ignored. When the address is 8000h-8008h, the Row Erase Program Memory command will only erase the user ID locations, regardless of the setting of the CP Configuration bit.

After receiving the Row Erase Program Memory command, the erase will not complete until the time interval, TERAR, has expired.

TABLE 4-2: PROGRAMMING ROW SIZE AND LATCHES

<table>
<thead>
<tr>
<th>Devices</th>
<th>PC</th>
<th>Row Size</th>
<th>Number of Latches</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16LF1566</td>
<td>&lt;15:5&gt;</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>PIC16LF1567</td>
<td>&lt;15:5&gt;</td>
<td>32</td>
<td>32</td>
</tr>
</tbody>
</table>

5.0 PROGRAMMING ALGORITHMS

The devices use internal latches to temporarily store the 14-bit words used for programming. Refer to Table 4-2 for specific latch information. The data latches allow the user to write the program words with a single Begin Externally Timed Programming or Begin Internally Timed Programming command. The Load Program Data or the Load Configuration command is used to load a single data latch. The data latch will hold the data until the Begin Externally Timed Programming or Begin Internally Timed Programming command is given.

The data latches are aligned with the LSbs of the address. The PC’s address at the time the Begin Externally Timed Programming or Begin Internally Timed Programming command is given will determine which location(s) in memory are written. Writes cannot cross the physical boundary. For example, with the PIC16LF1566/7, attempting to write from address 0002h-0009h will result in data being written to 0008h-000Fh.

If more than the maximum number of data latches are written without a Begin Externally Timed Programming or Begin Internally Timed Programming command, the data in the data latches will be overwritten. The following figures show the recommended flowcharts for programming.
FIGURE 5-1: DEVICE PROGRAM/VERIFY FLOWCHART

Start

Enter Programming Mode

Bulk Erase Device

Write Program Memory\(^1\)

Write User IDs

Verify Program Memory

Verify User IDs

Write Configuration Words\(^2\)

Verify Configuration Words

Exit Programming Mode

Done

Note 1: See Figure 5-2.
2: See Figure 5-5.
FIGURE 5-2: PROGRAM MEMORY FLOWCHART

Start

Bulk Erase Program Memory(1, 2)

Program Cycle(3)

Read Data from Program Memory

Data Correct?

No

Report Programming Failure

Yes

Increment Address Command

No

All Locations Done?

Yes

Done

Note 1: This step is optional if the device has already been erased or has not been previously programmed.

2: If the device is code-protected or must be completely erased, then Bulk Erase the device per Figure 5-6.

3: See Figure 5-3 or Figure 5-4.
FIGURE 5-3: ONE-WORD PROGRAM CYCLE

Program Cycle

Load Data for Program Memory

Begin Programming Command (Internally timed)

Wait TPINT

Begin Programming Command (Externally timed)(1)

Wait TPEXT

End Programming Command

Wait TDIS

Note 1: Externally timed writes are not supported for Configuration and Calibration bits.
FIGURE 5-4:  MULTIPLE-WORD PROGRAM CYCLE

Program Cycle

- Load Data for Program Memory
  - Increment Address Command
  - Load Data for Program Memory
  - Latch 1
  - Latch 2
  - Latch n
  - Increment Address Command
  - Load Data for Program Memory
- Begin Programming Command (Internally timed)
  - Wait TPINT
- Begin Programming Command (Externally timed)
  - Wait TPEXT
- End Programming Command
  - Wait TDIS
FIGURE 5-5: CONFIGURATION MEMORY PROGRAM FLOWCHART

Note 1: This step is optional if the device is erased or not previously programmed.

Note 2: See Figure 5-3.
6.0  CODE PROTECTION

Code protection is controlled using the CP bit in Configuration Word 1. When code protection is enabled, all program memory locations (0000h-7FFFh) read as ‘0’. Further programming is disabled for the program memory (0000h-7FFFh). The user ID locations and Configuration Words can be programmed and read out regardless of the code protection settings.

6.1  Program Memory

Code protection is enabled by programming the CP bit in Configuration Word 1 register to ‘0’.

The only way to disable code protection is to use the Bulk Erase Program Memory command.

7.0  HEX FILE USAGE

In the hex file there are two bytes per program word stored in the Intel® INHX32 hex format. Data is stored LSB first, MSB second. Because there are two bytes per word, the addresses in the hex file are 2x the address in program memory. (Example: Configuration Word 1 is stored at 8007h on the PIC16LF1566. In the hex file this will be referenced as 1000Eh-1000Fh).

7.1  Configuration Word

To allow portability of code, it is strongly recommended that the programmer is able to read the Configuration Words and user ID locations from the hex file. If the Configuration Words information was not present in the hex file, a simple warning message may be issued. Similarly, while saving a hex file, Configuration Words and user ID information should be included.

7.2  Device ID and Revision

If a device ID is present in the hex file at 1000Ch-1000Dh (8006h on the part), the programmer should verify the device ID (excluding the revision) against the value read from the part. On a mismatch condition the programmer should generate a warning message.

Note: This sequence does not erase the Calibration Words.
7.3 Checksum Computation

The checksum is calculated by two different methods dependent on the setting of the CP Configuration bit.

**TABLE 7-1: CONFIGURATION WORD MASK VALUES**

<table>
<thead>
<tr>
<th>Device</th>
<th>Config. Word 1 Mask</th>
<th>Config. Word 2 Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16LF1566</td>
<td>0EFBh</td>
<td>3E03h</td>
</tr>
<tr>
<td>PIC16LF1567</td>
<td>0EFBh</td>
<td>3E03h</td>
</tr>
</tbody>
</table>

### 7.3.1 PROGRAM CODE PROTECTION DISABLED

With the program code protection disabled, the checksum is computed by reading the contents of the program memory locations and adding up the program memory data starting at address 0000h, up to the maximum user addressable location. Any Carry bits exceeding 16 bits are ignored. Additionally, the relevant bits of the Configuration Words are added to the checksum. All unimplemented Configuration bits are masked to ‘0’.

**EXAMPLE 7-1: CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION DISABLED (CP = 1), PIC16LF1566/7, BLANK DEVICE**

- PIC16LF1566/7
- Sum of Memory addresses 0000h-1FFFh: E000h
- Configuration Word 1: 3FFFh
- Configuration Word 1 mask: 0EFBh
- Configuration Word 2: 3FFFh
- Configuration Word 2 mask: 3E03h
- Checksum = E000h + (3FFFh and 0EFBh) + (3FFFh and 3E03h)
- = E000h + 0EFBh + 3E03h
- = 2CFEh

**Note 1:** This value is obtained by taking the total number of program memory locations (0x000 to 0x1FFF which is 2000h) and multiplying it by the blank memory value of 0x3FFF to get the sum of 7FF E000h. Then, truncate to 16 bits, thus having a final value of E000h.

**2:** This value is obtained by making all bits of the Configuration Word 1 a ‘1’, then converting it to hex, thus having a value of 3FFFh.

**3:** This value is obtained by making all used bits of the Configuration Word 1 a ‘1’, then converting it to hex, thus having a value of 0EFBh.

**4:** This value is obtained by making all bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3FFFh.

**5:** This value is obtained by making all used bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3E03h.

**6:** This value is obtained by ANDing the Configuration Word value with the Configuration Word Mask Value and adding it to the sum of memory addresses: (3FFFh and 0EFBh) + (3FFFh and 3E03h) + E000h = 1 2CFEh. Then, truncate to 16 bits, thus having a final value of 2CFEh.
**EXAMPLE 7-2: CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION DISABLED (CP = 1), PIC16LF1566/7, 00AAh AT FIRST AND LAST ADDRESS**

<table>
<thead>
<tr>
<th>PIC16LF1566/7</th>
<th>Sum of Memory addresses 0000h-1FFFh</th>
<th>6156h(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Configuration Word 1</td>
<td>3FFFh(2)</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 1 mask</td>
<td>0EFBh(3)</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2</td>
<td>3FFFh(4)</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2 mask</td>
<td>3E03h(5)</td>
</tr>
<tr>
<td>Checksum</td>
<td>= 6156h + (3FFFh and 0EFBh) + (3FFFh and 3E03h)</td>
<td>AE54h</td>
</tr>
<tr>
<td></td>
<td>= 6156h + 0EFBh + 3E03h</td>
<td></td>
</tr>
<tr>
<td></td>
<td>= AE54h</td>
<td></td>
</tr>
</tbody>
</table>

**Note 1:** This value is obtained by taking the total number of program memory locations (0x000 to 0x1FFF which is 2000h) subtracting 2h which yields 1FFEh, then multiplying it by the blank memory value of 0x3FFF to get the sum of 7FF 6002h. Then, truncate to 16 bits the value of 6002h. Now add 0154h (00AAh + 00AAh) to 6002h to get the final value of 6156h.

2: This value is obtained by making all bits of the Configuration Word 1 a ‘1’, then converting it to hex, thus having a value of 3FFFh.

3: This value is obtained by making all used bits of the Configuration Word 1 a ‘1’, then converting it to hex, thus having a value of 0EFBh.

4: This value is obtained by making all bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3FFFh.

5: This value is obtained by making all used bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3E03h.

6: This value is obtained by ANDing the Configuration Word value with the Configuration Word Mask Value and adding it to the sum of memory addresses: (3FFFh and 0EFBh) + (3FFFh and 3E03h) + 6156h = AE54h. Then, truncate to 16 bits, thus having a final value of AE54h.
7.3.2 PROGRAM CODE PROTECTION ENABLED

With the program code protection enabled, the checksum is computed in the following manner: The Least Significant nibble of each user ID is used to create a 16-bit value. The masked value of user ID location 8000h is the Most Significant nibble. This sum of user IDs is summed with the Configuration Words (all unimplemented Configuration bits are masked to ’0’).

EXAMPLE 7-3: CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION ENABLED (CP = 0), PIC16LF1566/7, BLANK DEVICE

<table>
<thead>
<tr>
<th>PIC16LF1566/7</th>
<th>Configuration Word 1</th>
<th>3F7Fh(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Configuration Word 1 mask</td>
<td>0E7Bh(2)</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2</td>
<td>3FFFh(3)</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2 mask</td>
<td>3E03h(4)</td>
</tr>
<tr>
<td>User ID (8000h)</td>
<td></td>
<td>0006h(6)</td>
</tr>
<tr>
<td>User ID (8001h)</td>
<td></td>
<td>0007h(6)</td>
</tr>
<tr>
<td>User ID (8002h)</td>
<td></td>
<td>0001h(6)</td>
</tr>
<tr>
<td>User ID (8003h)</td>
<td></td>
<td>0002h(6)</td>
</tr>
</tbody>
</table>

Sum of User IDs = (0006h and 000Fh) << 12 + (0007h and 000Fh) << 8 + (0001h and 000Fh) << 4 + (0002h and 000Fh)(6) = 6000h + 0700h + 0010h + 0002h = 6712h

Checksum = (3F7Fh and 0E7Bh) + (3FFFh and 3E03h) + Sum of User IDs(7) = 0E7Bh + 3E03h + 6712h = B390h

Note 1: This value is obtained by making all bits of the Configuration Word 1 a ’1’, but the code-protect bit is ’0’ (thus, enabled), then converting it to hex, thus having a value of 3F7Fh.

2: This value is obtained by making all used bits of the Configuration Word 1 a ’1’, but the code-protect bit is ’0’ (thus, enabled), then converting it to hex, thus having a value of 0E7Bh.

3: This value is obtained by making all bits of the Configuration Word 2 a ’1’, then converting it to hex, thus having a value of 3FFFh.

4: This value is obtained by making all used bits of the Configuration Word 2 a ’1’, then converting it to hex, thus having a value of 3E03h.

5: These values are picked at random for this example; they could be any 16-bit value.

6: In order to calculate the sum of user IDs, take the 16-bit value of the first user ID location (0006h), AND the address to (000Fh), thus masking the MSB. This gives you the value 0006h, then shift left 12 bits, giving you 6000h. Do the same procedure for the 16-bit value of the second user ID location (0007h), except shift left eight bits. Also, do the same for the third user ID location (0001h), except shift left four bits. For the fourth user ID location do not shift. Finally, add up all four user ID values to get the final sum of user IDs of 6712h.

7: This value is obtained by ANDing the Configuration Word value with the Configuration Word Mask Value and adding it to the sum of user IDs: (3F7Fh AND 0E7Bh) + (3FFFh AND 3E03h) + 6712h = B390h.
**EXAMPLE 7-4: CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION ENABLED (CP = 0), PIC16LF1566/7, 00AAh AT FIRST AND LAST ADDRESS**

<table>
<thead>
<tr>
<th>PIC16LF1566/7</th>
<th>Configuration Word 1</th>
<th>3F7Fh(^{(1)})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Configuration Word 1 mask</td>
<td>0E7Bh(^{(2)})</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2</td>
<td>3FFFh(^{(3)})</td>
</tr>
<tr>
<td></td>
<td>Configuration Word 2 mask</td>
<td>3E03h(^{(4)})</td>
</tr>
<tr>
<td>User ID (8000h)</td>
<td>000Eh(^{(5)})</td>
<td></td>
</tr>
<tr>
<td>User ID (8001h)</td>
<td>0008h(^{(5)})</td>
<td></td>
</tr>
<tr>
<td>User ID (8002h)</td>
<td>0005h(^{(5)})</td>
<td></td>
</tr>
<tr>
<td>User ID (8003h)</td>
<td>0008h(^{(5)})</td>
<td></td>
</tr>
<tr>
<td>Sum of User IDs</td>
<td>((000Eh \text{ and } 000Fh) \ll 12 + (0008h \text{ and } 000Fh) \ll 8 + (0005h \text{ and } 000Fh) \ll 4 + (0008h \text{ and } 000Fh))(^{(6)})</td>
<td>E858h</td>
</tr>
<tr>
<td>Checksum</td>
<td>((3F7Fh \text{ and } 0E7Bh) + (3FFFh \text{ and } 3E03h) + \text{Sum of User IDs})(^{(7)})</td>
<td>34D6h</td>
</tr>
</tbody>
</table>

**Note 1:** This value is obtained by making all bits of the Configuration Word 1 a ‘1’, but the code-protect bit is ‘0’ (thus, enabled), then converting it to hex, thus having a value of 3F7Fh.

2: This value is obtained by making all used bits of the Configuration Word 1 a ‘1’, but the code-protect bit is ‘0’ (thus, enabled), then converting it to hex, thus having a value of 0E7Bh.

3: This value is obtained by making all bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3FFFh.

4: This value is obtained by making all used bits of the Configuration Word 2 a ‘1’, then converting it to hex, thus having a value of 3E03h.

5: These values are picked at random for this example; they could be any 16-bit value.

6: In order to calculate the sum of user IDs, take the 16-bit value of the first user ID location (000Eh), AND the address to (000Fh), thus masking the MSB. This gives you the value 000Eh, then shift left 12 bits, giving you E000h. Do the same procedure for the 16-bit value of the second user ID location (0008h), except shift left eight bits. Also, do the same for the third user ID location (0005h), except shift left four bits. For the fourth user ID location do not shift. Finally, add up all four user ID values to get the final sum of user IDs of E858h.

7: This value is obtained by ANDing the Configuration Word value with the Configuration Word Mask Value and adding it to the sum of user IDs: \((3F7Fh \text{ AND } 0E7Bh) + (3FFFh \text{ AND } 3E03h) + E858h = 34D6h\).
8.0 ELECTRICAL SPECIFICATIONS

Refer to the device-specific data sheet for absolute maximum ratings.

### AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE

<table>
<thead>
<tr>
<th>Ac/Dc Characteristics</th>
<th>Standard Operating Conditions</th>
<th>Conditions/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supply Voltages and Currents</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VDD Read/Write and Row Erase operations</td>
<td>VDD min. — VDD max.</td>
<td>V</td>
</tr>
<tr>
<td>Bulk Erase operations</td>
<td>2.7 — VDD max.</td>
<td>V</td>
</tr>
<tr>
<td>IDD</td>
<td>Current on VDD, Idle</td>
<td>— — 1.0 mA</td>
</tr>
<tr>
<td>IDDP</td>
<td>Current on VDD, Programming</td>
<td>— — 3.0 mA</td>
</tr>
<tr>
<td>IPP</td>
<td>Current on MCLR/VPP</td>
<td>— — 600 μA</td>
</tr>
<tr>
<td>VIHH</td>
<td>High voltage on MCLR/VPP for Program/Verify mode entry</td>
<td>8.0 — 9.0 V</td>
</tr>
<tr>
<td>TVHHR</td>
<td>MCLR rise time (VIL to VIHH) for Program/Verify mode entry</td>
<td>— — 1.0 μs</td>
</tr>
<tr>
<td>I/O pins</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VIL (ICSPCLK, ICSPDAT, MCLR/VPP) input low level</td>
<td>— — 0.2 VDD</td>
<td>V</td>
</tr>
<tr>
<td>VOH (ICSPCLK, ICSPDAT, MCLR/VPP) input high level</td>
<td>VDD-0.7 — — V</td>
<td></td>
</tr>
<tr>
<td>IOH = 3 mA, VDD = 3.3V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOH = 2 mA, VDD = 1.8V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VOL</td>
<td>ICSPDAT output low level</td>
<td>— — VSS+0.6</td>
</tr>
<tr>
<td>IOH = 6 mA, VDD = 3.3V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOH = 3 mA, VDD = 1.8V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming Mode Entry and Exit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TENTS</td>
<td>Programing mode entry setup time: ICSPCLK, ICSPDAT setup time before VDD or MCLR</td>
<td>100 — — ns</td>
</tr>
<tr>
<td>TENTH</td>
<td>Programing mode entry hold time: ICSPCLK, ICSPDAT hold time after VDD or MCLR</td>
<td>250 — — μs</td>
</tr>
<tr>
<td>Serial Program/Verify</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCKL</td>
<td>Clock Low Pulse Width</td>
<td>100 — — ns</td>
</tr>
<tr>
<td>TCKH</td>
<td>Clock High Pulse Width</td>
<td>100 — — ns</td>
</tr>
<tr>
<td>TDS</td>
<td>Data in setup time before clock</td>
<td>100 — — ns</td>
</tr>
<tr>
<td>TDH</td>
<td>Data in hold time after clock</td>
<td>100 — — ns</td>
</tr>
<tr>
<td>TCO</td>
<td>Clock to data out valid (during a Read Data command)</td>
<td>0 — 80 ns</td>
</tr>
<tr>
<td>TLZD</td>
<td>Clock to data low-impedance (during a Read Data command)</td>
<td>0 — 80 ns</td>
</tr>
<tr>
<td>THZD</td>
<td>Clock to data high-impedance (during a Read Data command)</td>
<td>0 — 80 ns</td>
</tr>
<tr>
<td>TDLY</td>
<td>Data input not driven to next clock input (delay required between command/data or command/command)</td>
<td>1.0 — — μs</td>
</tr>
<tr>
<td>TERAB</td>
<td>Bulk Erase cycle time</td>
<td>— — 5 ms</td>
</tr>
<tr>
<td>TERAR</td>
<td>Row Erase cycle time</td>
<td>— — 2.5 ms</td>
</tr>
<tr>
<td>TPINT</td>
<td>Internally timed programming operation time</td>
<td>— — 2.5 ms</td>
</tr>
<tr>
<td></td>
<td>Program memory Configuration Words</td>
<td></td>
</tr>
<tr>
<td>TPEXT</td>
<td>Externally timed programming pulse</td>
<td>1.0 — 2.1 ms</td>
</tr>
<tr>
<td>Note 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**
- TTL voltages are defined with respect to VSS.
- The setup time required for successful command/data or command/command access is determined by tip-to-tip propagation delays. Any setup time required by a command/data or command/command is included in the propagation delays.
- Internal timing parameters indicate uncertainty in the worst-case condition.
- All timing parameters are valid over operating conditions unless otherwise specified.
### AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE (CONTINUED)

<table>
<thead>
<tr>
<th>Sym.</th>
<th>Characteristics</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Units</th>
<th>Conditions/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDIS</td>
<td>Time delay from program to compare (HV discharge time)</td>
<td>300</td>
<td>—</td>
<td>—</td>
<td>μs</td>
<td></td>
</tr>
<tr>
<td>TEXIT</td>
<td>Time delay when exiting Program/Verify mode</td>
<td>1</td>
<td>—</td>
<td>—</td>
<td>μs</td>
<td></td>
</tr>
</tbody>
</table>

**Note 1:** Externally timed writes are not supported for Configuration and Calibration bits.
8.1 AC Timing Diagrams

FIGURE 8-1: PROGRAMMING MODE ENTRY – VDD FIRST

FIGURE 8-2: PROGRAMMING MODE ENTRY – VPP FIRST

FIGURE 8-3: PROGRAMMING MODE EXIT – VPP LAST

FIGURE 8-4: PROGRAMMING MODE EXIT – VDD LAST
FIGURE 8-5: CLOCK AND DATA TIMING

FIGURE 8-6: WRITE COMMAND-PAYLOAD TIMING

FIGURE 8-7: READ COMMAND-PAYLOAD TIMING
FIGURE 8-8: LVP ENTRY (POWERED)

FIGURE 8-9: LVP ENTRY (POWERING UP)

Note 1: Sequence matching can start with no edge on MCLR first.
APPENDIX A: REVISION HISTORY

Revision A (06/2015)

Initial release of this document.
Note the following details of the code protection feature on Microchip devices:

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

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, Keeloq, Keeloq logo, Kleer, LANcheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC32 logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KleerNet, KleerNet logo, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscent Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestiC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

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

© 2015, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-63277-478-1

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company’s quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, Keeloq® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001:2000 certified.
<table>
<thead>
<tr>
<th><strong>AMERICAS</strong></th>
<th><strong>ASIA/PACIFIC</strong></th>
<th><strong>ASIA/PACIFIC</strong></th>
<th><strong>EUROPE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Corporate Office</strong>&lt;br&gt;2355 West Chandler Blvd.&lt;br&gt;Chandler, AZ 85224-6199&lt;br&gt;Tel: 480-792-7200&lt;br&gt;Fax: 480-792-7277&lt;br&gt;Technical Support:&lt;br&gt;<a href="http://www.microchip.com/support">http://www.microchip.com/support</a>&lt;br&gt;Web Address:&lt;br&gt;www.microchip.com</td>
<td><strong>Asia Pacific Office</strong>&lt;br&gt;Suites 3707-14, 37th Floor&lt;br&gt;Tower 6, The Gateway&lt;br&gt;Harbour City, Kowloon&lt;br&gt;Hong Kong&lt;br&gt;Tel: 852-2943-5100&lt;br&gt;Fax: 852-2401-3431</td>
<td><strong>China - Xiamen</strong>&lt;br&gt;Tel: 86-592-2388138&lt;br&gt;Fax: 86-592-2388130</td>
<td><strong>Austria - Wels</strong>&lt;br&gt;Tel: 43-7242-2244-39&lt;br&gt;Fax: 43-7242-2244-393</td>
</tr>
<tr>
<td><strong>Atlanta</strong>&lt;br&gt;Duluth, GA&lt;br&gt;Tel: 678-957-9614&lt;br&gt;Fax: 678-957-1455</td>
<td><strong>China - Zhumai</strong>&lt;br&gt;Tel: 86-756-3210040&lt;br&gt;Fax: 86-756-3210049</td>
<td><strong>Denmark - Copenhagen</strong>&lt;br&gt;Tel: 45-4450-2828&lt;br&gt;Fax: 45-4485-2829</td>
<td><strong>France - Paris</strong>&lt;br&gt;Tel: 33-1-69-53-63-20&lt;br&gt;Fax: 33-1-69-30-90-79</td>
</tr>
<tr>
<td><strong>Austin, TX</strong>&lt;br&gt;Tel: 512-257-3370</td>
<td><strong>India - Bangalore</strong>&lt;br&gt;Tel: 91-80-3090-4444&lt;br&gt;Fax: 91-80-3090-4123</td>
<td><strong>Germany - Dusseldorf</strong>&lt;br&gt;Tel: 49-2129-3766400</td>
<td><strong>Germany - Munich</strong>&lt;br&gt;Tel: 49-89-627-144-0&lt;br&gt;Fax: 49-89-627-144-44</td>
</tr>
<tr>
<td><strong>Boston</strong>&lt;br&gt;Westborough, MA&lt;br&gt;Tel: 774-760-0087&lt;br&gt;Fax: 774-760-0088</td>
<td><strong>India - New Delhi</strong>&lt;br&gt;Tel: 91-11-4160-8631&lt;br&gt;Fax: 91-11-4160-8632</td>
<td><strong>Germany - Pforzheim</strong>&lt;br&gt;Tel: 49-7231-424750</td>
<td><strong>Italy - Milan</strong>&lt;br&gt;Tel: 39-0331-742611&lt;br&gt;Fax: 39-0331-466781</td>
</tr>
<tr>
<td><strong>Chicago</strong>&lt;br&gt;Itasca, IL&lt;br&gt;Tel: 630-285-0071&lt;br&gt;Fax: 630-285-0075</td>
<td><strong>India - Pune</strong>&lt;br&gt;Tel: 91-20-3019-1500</td>
<td><strong>Italy - Venice</strong>&lt;br&gt;Tel: 39-049-7625286</td>
<td><strong>Italy - Venice</strong>&lt;br&gt;Tel: 39-049-7625286</td>
</tr>
<tr>
<td><strong>Cleveland</strong>&lt;br&gt;Independence, OH&lt;br&gt;Tel: 216-447-0464&lt;br&gt;Fax: 216-447-0643</td>
<td><strong>Japan - Osaka</strong>&lt;br&gt;Tel: 81-6-6152-7160&lt;br&gt;Fax: 81-6-6152-9310</td>
<td><strong>Netherlands - Drunen</strong>&lt;br&gt;Tel: 31-416-690399&lt;br&gt;Fax: 31-416-690340</td>
<td><strong>Poland - Warsaw</strong>&lt;br&gt;Tel: 48-22-3325737</td>
</tr>
<tr>
<td><strong>Dallas</strong>&lt;br&gt;Addison, TX&lt;br&gt;Tel: 972-818-7423&lt;br&gt;Fax: 972-818-2924</td>
<td><strong>Japan - Tokyo</strong>&lt;br&gt;Tel: 81-3-6880-3770&lt;br&gt;Fax: 81-3-6880-3771</td>
<td><strong>Poland - Warsaw</strong>&lt;br&gt;Tel: 48-22-3325737</td>
<td><strong>Spain - Madrid</strong>&lt;br&gt;Tel: 34-91-708-08-90&lt;br&gt;Fax: 34-91-708-08-91</td>
</tr>
<tr>
<td><strong>Detroit</strong>&lt;br&gt;Novi, MI&lt;br&gt;Tel: 248-848-4000</td>
<td><strong>Korea - Daegu</strong>&lt;br&gt;Tel: 82-53-744-4301&lt;br&gt;Fax: 82-53-744-4302</td>
<td><strong>Sweden - Stockholm</strong>&lt;br&gt;Tel: 46-8-5090-4654</td>
<td><strong>UK - Wokingham</strong>&lt;br&gt;Tel: 44-118-921-5800&lt;br&gt;Fax: 44-118-921-5820</td>
</tr>
<tr>
<td><strong>Houston, TX</strong>&lt;br&gt;Tel: 281-894-5983</td>
<td><strong>Korea - Seoul</strong>&lt;br&gt;Tel: 82-2-554-7200&lt;br&gt;Fax: 82-2-558-5932 or 82-2-558-5934</td>
<td><strong>UK - Wokingham</strong>&lt;br&gt;Tel: 44-118-921-5800&lt;br&gt;Fax: 44-118-921-5820</td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>Indianapolis</strong>&lt;br&gt;Nobleville, IN&lt;br&gt;Tel: 317-773-8323&lt;br&gt;Fax: 317-773-5453</td>
<td><strong>Malaysia - Kuala Lumpur</strong>&lt;br&gt;Tel: 60-3-6201-9857&lt;br&gt;Fax: 60-3-6201-9859</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Los Angeles</strong>&lt;br&gt;Mission Viejo, CA&lt;br&gt;Tel: 949-462-9523&lt;br&gt;Fax: 949-462-9608</td>
<td><strong>Malaysia - Penang</strong>&lt;br&gt;Tel: 60-4-227-8870&lt;br&gt;Fax: 60-4-227-4068</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>New York, NY</strong>&lt;br&gt;Tel: 631-435-6000</td>
<td><strong>Philippines - Manila</strong>&lt;br&gt;Tel: 63-2-634-9065&lt;br&gt;Fax: 63-2-634-9069</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>San Jose, CA</strong>&lt;br&gt;Tel: 408-735-9110</td>
<td><strong>Singapore</strong>&lt;br&gt;Tel: 65-6334-8870&lt;br&gt;Fax: 65-6334-8850</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>Canada - Toronto</strong>&lt;br&gt;Tel: 905-673-0699&lt;br&gt;Fax: 905-673-6509</td>
<td><strong>Taiwan - Hsin Chu</strong>&lt;br&gt;Tel: 886-3-5778-366&lt;br&gt;Fax: 886-3-5770-955</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>Canada - Toronto</strong>&lt;br&gt;Tel: 905-673-0699&lt;br&gt;Fax: 905-673-6509</td>
<td><strong>Taiwan - Kaohsiung</strong>&lt;br&gt;Tel: 886-7-213-7828</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>Canada - Toronto</strong>&lt;br&gt;Tel: 905-673-0699&lt;br&gt;Fax: 905-673-6509</td>
<td><strong>Taiwan - Taipei</strong>&lt;br&gt;Tel: 886-2-2508-8600&lt;br&gt;Fax: 886-2-2508-0102</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
<tr>
<td><strong>Canada - Toronto</strong>&lt;br&gt;Tel: 905-673-0699&lt;br&gt;Fax: 905-673-6509</td>
<td><strong>Thailand - Bangkok</strong>&lt;br&gt;Tel: 66-2-694-1351&lt;br&gt;Fax: 66-2-694-1350</td>
<td><strong>Worldwide Sales and Service</strong></td>
<td><strong>Worldwide Sales and Service</strong></td>
</tr>
</tbody>
</table>