1.0 PROGRAMMING THE PIC16F87X

The PIC16F87X is programmed using a serial method. The Serial mode will allow the PIC16F87X to be programmed while in the user’s system. This allows for increased design flexibility. This programming specification applies to PIC16F87X devices in all packages.

1.1 Programming Algorithm Requirements

The programming algorithm used depends on the operating voltage (VDD) of the PIC16F87X device. Algorithm 1 is designed for a VDD range of 2.2V \(\leq VDD < 5.5V\). Algorithm 2 is for a range of 4.5V \(\leq VDD \leq 5.5V\). Either algorithm can be used with the two available programming entry methods. The first method follows the normal Microchip Programming mode entry of applying a VPP voltage of 13V ± .5V. The second method, called Low Voltage ICSPTM or LVP for short, applies VDD to MCLR and uses the I/O pin RB3 to enter Programming mode. When RB3 is driven to VDD from ground, the PIC16F87X device enters Programming mode.

1.2 Programming Mode

The Programming mode for the PIC16F87X allows programming of user program memory, data memory, special locations used for ID, and the configuration word.
**PIN DESCRIPTIONS (DURING PROGRAMMING): PIC16F87X**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Function</th>
<th>Pin Type</th>
<th>Pin Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RB3</td>
<td>PGM</td>
<td>I</td>
<td>Low voltage ICSP programming input if LVP configuration bit equals 1</td>
</tr>
<tr>
<td>RB6</td>
<td>CLOCK</td>
<td>I</td>
<td>Clock input</td>
</tr>
<tr>
<td>RB7</td>
<td>DATA</td>
<td>I/O</td>
<td>Data input/output</td>
</tr>
<tr>
<td>MCLR</td>
<td>VTEST MODE</td>
<td>P*</td>
<td>Program Mode Select</td>
</tr>
<tr>
<td>VDD</td>
<td>VDD</td>
<td>P</td>
<td>Power Supply</td>
</tr>
<tr>
<td>VSS</td>
<td>VSS</td>
<td>P</td>
<td>Ground</td>
</tr>
</tbody>
</table>

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

* In the PIC16F87X, the programming high voltage is internally generated. To activate the Programming mode, high voltage needs to be applied to the MCLR input. Since the MCLR is used for a level source, this means that MCLR does not draw any significant current.
2.0 PROGRAM MODE ENTRY

2.1 User Program Memory Map

The user memory space extends from 0x0000 to 0x1FFF (8K). In Programming mode, the program memory space extends from 0x0000 to 0x3FFF, with the first half (0x0000-0x1FFF) being user program memory and the second half (0x2000-0x3FFF) being configuration memory. The PC will increment from 0x0000 to 0x1FFF and wrap to 0x0000, 0x2000 to 0x3FFF and wrap around to 0x2000 (not to 0x0000). Once in configuration memory, the highest bit of the PC stays a ‘1’, thus always pointing to the configuration memory. The only way to point to user program memory is to reset the part and re-enter Program/Verify mode, as described in Section 2.4.

In the configuration memory space, 0x2000-0x200F are physically implemented. However, only locations 0x2000 through 0x2007 are available. Other locations are reserved. Locations beyond 0x200F will physically access user memory (see Figure 2-1).

2.2 Data EEPROM Memory

The EEPROM data memory space is a separate block of high endurance memory that the user accesses using a special sequence of instructions. The amount of data EEPROM memory depends on the device and is shown below in number of bytes.

<table>
<thead>
<tr>
<th>Device</th>
<th># of Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16F870</td>
<td>64</td>
</tr>
<tr>
<td>PIC16F871</td>
<td>64</td>
</tr>
<tr>
<td>PIC16F872</td>
<td>64</td>
</tr>
<tr>
<td>PIC16F873</td>
<td>128</td>
</tr>
<tr>
<td>PIC16F874</td>
<td>128</td>
</tr>
<tr>
<td>PIC16F876</td>
<td>256</td>
</tr>
<tr>
<td>PIC16F877</td>
<td>256</td>
</tr>
</tbody>
</table>

The contents of data EEPROM memory have the capability to be embedded into the HEX file.

The programmer should be able to read data EEPROM information from a HEX file and conversely (as an option), write data EEPROM contents to a HEX file, along with program memory information and configuration bit information.

The 256 data memory locations are logically mapped starting at address 0x2100. The format for data memory storage is one data byte per address location, LSB aligned.

2.3 ID Locations

A user may store identification information (ID) in four ID locations. The ID locations are mapped in [0x2000 : 0x2003]. It is recommended that the user use only the four Least Significant bits of each ID location. In some devices, the ID locations read out in an unscrambled fashion after code protection is enabled. For these devices, it is recommended that ID location is written as "11 1111 1000 bbbb" where 'bbbb' is ID information.

In other devices, the ID locations read out normally, even after code protection. To understand how the devices behave, refer to Table 5-1.

To understand the scrambling mechanism after code protection, refer to Section 4.0.
<table>
<thead>
<tr>
<th>Address</th>
<th>2K words</th>
<th>4K words</th>
<th>8K words</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID Location</td>
<td>Implemented</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>ID Location</td>
<td>Implemented</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>ID Location</td>
<td>Implemented</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>ID Location</td>
<td>Implemented</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Reserved</td>
<td>Implemented</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Device ID</td>
<td>Reserved</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
</tbody>
</table>

NOTE: Addresses 0x0 to 0x3FFF are reserved.
2.4 Program/Verify Mode

The Program/Verify mode is entered by holding pins RB6 and RB7 low, while raising MCLR pin from VIL to VIHH (high voltage). In this mode, the state of the RB3 pin does not affect programming. Low voltage ICSP Programming mode is entered by raising RB3 from VIL to VDD and then applying VDD to MCLR. Once in this mode, the user program memory and the configuration memory can be accessed and programmed in serial fashion. The mode of operation is serial, and the memory that is accessed is the user program memory. RB6 and RB7 are Schmitt Trigger Inputs in this mode.

The sequence that enters the device into the Programming/Verify mode places all other logic into the RESET state (the MCLR pin was initially at VIL). This means that all I/O are in the RESET state (high impedance inputs).

The normal sequence for programming is to use the load data command to set a value to be written at the selected address. Issue the begin programming command followed by read data command to verify, and then increment the address.

A device RESET will clear the PC and set the address to 0. The “increment address” command will increment the PC. The “load configuration” command will set the PC to 0x2000. The available commands are shown in Table 2-2.

2.4.1 LOW VOLTAGE ICSP PROGRAMMING MODE

Low voltage ICSP Programming mode allows a PIC16F87X device to be programmed using VDD only. However, when this mode is enabled by a configuration bit (LVP), the PIC16F87X device dedicates RB3 to control entry/exit into Programming mode.

When LVP bit is set to ‘1’, the low voltage ICSP programming entry is enabled. Since the LVP configuration bit allows low voltage ICSP programming entry in its erased state, an erased device will have the LVP bit enabled at the factory. While LVP is ‘1’, RB3 is dedicated to low voltage ICSP programming. Bring RB3 to VDD and then MCLR to VDD to enter programming mode. All other specifications for high voltage ICSP™ apply.

To disable low voltage ICSP mode, the LVP bit must be programmed to ‘0’. This must be done while entered with High Voltage Entry mode (LVP bit = 1). RB3 is now a general purpose I/O pin.

2.4.2 SERIAL PROGRAM/VERIFY OPERATION

The RB6 pin is used as a clock input pin, and the RB7 pin is used for entering command bits and data input/output during serial operation. To input a command, the clock pin (RB6) is cycled six times. Each command bit is latched on the falling edge of the clock, with the Least Significant bit (LSb) of the command being input first. The data on pin RB7 is required to have a minimum setup and hold time (see AC/DC specifications), with respect to the falling edge of the clock. Commands that have data associated with them (read and load) are specified to have a minimum delay of 1 µs between the command and the data. After this delay, the clock pin is cycled 16 times with the first cycle being a START bit and the last cycle being a STOP bit. Data is also input and output LSb first.

Therefore, during a read operation, the LSb will be transmitted onto pin RB7 on the rising edge of the second cycle, and during a load operation, the LSb will be latched on the falling edge of the second cycle. A minimum 1 µs delay is also specified between consecutive commands.

All commands are transmitted LSb first. Data words are also transmitted LSb first. The data is transmitted on the rising edge and latched on the falling edge of the clock. To allow for decoding of commands and reversal of data pin configuration, a time separation of at least 1 µs is required between a command and a data word (or another command).

The commands that are available are:

2.4.2.1 Load Configuration

After receiving this command, the program counter (PC) will be set to 0x2000. By then applying 16 cycles to the clock pin, the chip will load 14-bits in a “data word,” as described above, to be programmed into the configuration memory. A description of the memory mapping schemes of the program memory for normal operation and Configuration mode operation is shown in Figure 2-1. After the configuration memory is entered, the only way to get back to the user program memory is to exit the Program/Verify Test mode by taking MCLR low (VIL).

2.4.2.2 Load Data for Program Memory

After receiving this command, the chip will load in a 14-bit “data word” when 16 cycles are applied, as described previously. A timing diagram for the load data command is shown in Figure 6-1.
2.4.2.3 Load Data for Data Memory

After receiving this command, the chip will load in a 14-bit “data word” when 16 cycles are applied. However, the data memory is only 8-bits wide, and thus, only the first 8-bits of data after the START bit will be programmed into the data memory. It is still necessary to cycle the clock the full 16 cycles in order to allow the internal circuitry to reset properly. The data memory contains up to 256 bytes. If the device is code protected, the data is read as all zeros.

2.4.2.4 Read Data from Program Memory

After receiving this command, the chip will transmit data bits out of the program memory (user or configuration) currently accessed, starting with the second rising edge of the clock input. The RB7 pin will go into Output mode on the second rising clock edge, and it will revert back to Input mode (hi-impedance) after the 16th rising edge. A timing diagram of this command is shown in Figure 6-2.

2.4.2.5 Read Data from Data Memory

After receiving this command, the chip will transmit data bits out of the data memory starting with the second rising edge of the clock input. The RB7 pin will go into Output mode on the second rising edge, and it will revert back to Input mode (hi-impedance) after the 16th rising edge. As previously stated, the data memory is 8-bits wide, and therefore, only the first 8-bits that are output are actual data.

2.4.2.6 Increment Address

The PC is incremented when this command is received. A timing diagram of this command is shown in Figure 6-3.

2.4.2.7 Begin Erase/Program Cycle

A load command must be given before every begin programming command. Programming of the appropriate memory (test program memory, user program memory or data memory) will begin after this command is received and decoded. An internal timing mechanism executes an erase before write. The user must allow for both erase and programming cycle times for programming to complete. No “end programming” command is required.

2.4.2.8 Begin Programming

A load command must be given before every begin programming command. Programming of the appropriate memory (test program memory, user program memory or data memory) will begin after this command is received and decoded. An internal timing mechanism executes a write. The user must allow for program cycle time for programming to complete. No “end programming” command is required.

This command is similar to the ERASE/PROGRAM CYCLE command, except that a word erase is not done. It is recommended that a bulk erase be performed before starting a series of programming only cycles.

**TABLE 2-2: COMMAND MAPPING FOR PIC16F87X**

<table>
<thead>
<tr>
<th>Command</th>
<th>Mapping (MSB … LSB)</th>
<th>Data</th>
<th>Voltage Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Configuration</td>
<td>X X 0 0 0 0 0 0 0 0</td>
<td>0, data (14), 0</td>
<td>2.2V - 5.5V</td>
</tr>
<tr>
<td>Load Data for Program Memory</td>
<td>X X 0 0 0 1 0 0</td>
<td>0, data (14), 0</td>
<td>2.2V - 5.5V</td>
</tr>
<tr>
<td>Read Data from Program Memory</td>
<td>X X 0 1 0 0 0</td>
<td>0, data (14), 0</td>
<td>2.2V - 5.5V</td>
</tr>
<tr>
<td>Increment Address</td>
<td>X X 1 1 0 0 0</td>
<td>2.2V - 5.5V</td>
<td></td>
</tr>
<tr>
<td>Begin Erase Programming Cycle</td>
<td>0 0 1 0 0 0 0</td>
<td>2.2V - 5.5V</td>
<td></td>
</tr>
<tr>
<td>Begin Programming Only Cycle</td>
<td>0 1 1 0 0 0 0</td>
<td>4.5V - 5.5V</td>
<td></td>
</tr>
<tr>
<td>Load Data for Data Memory</td>
<td>X X 0 0 1 1 1</td>
<td>0, data (14), 0</td>
<td>2.2V - 5.5V</td>
</tr>
<tr>
<td>Read Data from Data Memory</td>
<td>X X 0 1 0 1 1</td>
<td>0, data (14), 0</td>
<td>2.2V - 5.5V</td>
</tr>
<tr>
<td>Bulk Erase Setup1</td>
<td>0 0 0 0 0 1 1 1</td>
<td>4.5V - 5.5V</td>
<td></td>
</tr>
<tr>
<td>Bulk Erase Setup2</td>
<td>0 0 0 0 0 1 1 1</td>
<td>4.5V - 5.5V</td>
<td></td>
</tr>
</tbody>
</table>

*Note:* The Begin Program operation must take place at 4.5 to 5.5 VDD range.
2.5 Erasing Program and Data Memory

Depending on the state of the code protection bits, program and data memory will be erased using different procedures. The first set of procedures is used when both program and data memories are not code protected. The second set of procedures must be used when either memory is code protected. A device programmer should determine the state of the code protection bits and then apply the proper procedure to erase the desired memory.

2.5.1 ERASING NON-CODE PROTECTED PROGRAM AND DATA MEMORY

When both program and data memories are not code protected, they must be individually erased using the following procedures. The only way that both memories are erased using a single procedure is if code protection is enabled for one of the memories. These procedures do not erase the configuration word or ID locations.

Procedure to bulk erase program memory:
1. Execute a Load Data for Program Memory command (000010) with a '1' in all locations (0x3FFF)
2. Execute a Bulk Erase Setup1 command (000001)
3. Execute a Bulk Erase Setup2 command (000111)
4. Execute a Begin Erase/Programming command (001000)
5. Wait 8 ms
6. Execute a Bulk Erase Setup1 command (000001)
7. Execute a Bulk Erase Setup2 command (000111)

Procedure to bulk erase data memory:
1. Execute a Load Data for Data Memory command (000011) with a '1' in all locations (0x3FFF)
2. Execute a Bulk Erase Setup1 command (000001)
3. Execute a Bulk Erase Setup2 command (000111)
4. Execute a Begin Erase/Programming command (001000)
5. Wait 8 ms
6. Execute a Bulk Erase Setup1 command (000001)
7. Execute a Bulk Erase Setup2 command (000111)

2.5.2 ERASING CODE PROTECTED MEMORY

For the PIC16F87X devices, once code protection is enabled, all protected program and data memory locations read all '0's and further programming is disabled. The ID locations and configuration word read out unscrambled and can be reprogrammed normally. The only procedure to erase a PIC16F87X device that is code protected is shown in the following procedure. This method erases program memory, data memory, configuration bits and ID locations. Since all data within the program and data memory will be erased when this procedure is executed, the security of the data or code is not compromised.

1. Execute a Load Configuration command (000000) with a '1' in all locations (0x3FFF)
2. Execute Increment Address command (000110) to set address to configuration word location (0x2007)
3. Execute a Bulk Erase Setup1 command (000001)
4. Execute a Bulk Erase Setup2 command (000111)
5. Execute a Begin Erase/Programming command (001000)
6. Wait 8 ms
7. Execute a Bulk Erase Setup1 command (000001)
8. Execute a Bulk Erase Setup2 command (000111)
FIGURE 2-1: FLOW CHART - PIC16F87X PROGRAM MEMORY (2.2V ≤ VDD < 5.5V)

START

Set VDD = VDDP

Load Data Command

Begin Erase/Programming Command

Wait tera + tprog

Increment Address Command

All Locations Done?

Verify all Locations

Report Verify Error

Data Correct?

DONE
FIGURE 2-2: FLOW CHART – PIC16F87X PROGRAM MEMORY (4.5V ≤ VDD ≤ 5.5V)

START

Bulk Erase Sequence

Set VDD = VDDP

Load Data Command

Begin Programming Only Command

Wait tprog

Increment Address Command

No All Locations Done?

Verify all Locations

Report Verify Error

Data Correct?

DONE

Report Verify Error

INC

Verify all Locations

Data Correct?

DONE
FIGURE 2-3: FLOW CHART – PIC16F87X CONFIGURATION MEMORY (2.2V ≤ VDD < 5.5V)

START

Load Configuration Data

Program ID Location?

Yes

Program Cycle

Read Data Command

No

Program Cycle

Report Programming Failure

Data Correct?

Yes

Increment Address Command

Address = 0x2004?

Yes

Increment Address Command

Yes

Address = 0x2004?

No

Increment Address Command

Report Program Configuration Word Error

Data Correct?

No

Program Cycle (Config. Word)

Read Data Command

Yes

DONE

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment Address Command

Increment AddressCommand
FIGURE 2-4: FLOW CHART - PIC16F87X CONFIGURATION MEMORY

* Assumes that a bulk erase was issued before programming configuration word. If not, use the program flow from Figure 2-4.
3.0 CONFIGURATION WORD

The PIC16F87X has several configuration bits. These bits can be set (reads ‘0’), or left unchanged (reads ‘1’), to select various device configurations.

3.1 Device ID Word

The device ID word for the PIC16F87X is located at 2006h.

<table>
<thead>
<tr>
<th>Device</th>
<th>Device ID Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16F870</td>
<td>00 1101 000 x xxxx</td>
</tr>
<tr>
<td>PIC16F871</td>
<td>00 1101 001 x xxxx</td>
</tr>
<tr>
<td>PIC16F872</td>
<td>00 1000 111 x xxxx</td>
</tr>
<tr>
<td>PIC16F873</td>
<td>00 1001 011 x xxxx</td>
</tr>
<tr>
<td>PIC16F874</td>
<td>00 1001 001 x xxxx</td>
</tr>
<tr>
<td>PIC16F876</td>
<td>00 1001 111 x xxxx</td>
</tr>
<tr>
<td>PIC16F877</td>
<td>00 1001 101 x xxxx</td>
</tr>
</tbody>
</table>
REGISTER 3-1: CONFIG: CONFIGURATION WORD FOR PIC16F873/874/876/877 (ADDRESS 2007h)

<table>
<thead>
<tr>
<th>bit 13-12</th>
<th>CP1:CP0: FLASH Program Memory Code Protection bits(^{(2)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 5-4</td>
<td>4 K Devices:</td>
</tr>
<tr>
<td></td>
<td>11 = Code protection off</td>
</tr>
<tr>
<td></td>
<td>10 = 0F00h to 0FFFh code protected</td>
</tr>
<tr>
<td></td>
<td>01 = 0800h to 0FFFh code protected</td>
</tr>
<tr>
<td></td>
<td>00 = 0000h to 0FFFh code protected</td>
</tr>
<tr>
<td>bit 11</td>
<td>Reserved: Set to ‘1’ for normal operation</td>
</tr>
<tr>
<td>bit 10</td>
<td>Unimplemented: Read as ‘1’</td>
</tr>
<tr>
<td>bit 9</td>
<td>WRT: FLASH Program Memory Write Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = Unprotected program memory may be written to by EECON control</td>
</tr>
<tr>
<td></td>
<td>0 = Unprotected program memory may not be written to by EECON control</td>
</tr>
<tr>
<td>bit 8</td>
<td>CPD: Data EE Memory Code Protection bit</td>
</tr>
<tr>
<td></td>
<td>1 = Code protection off</td>
</tr>
<tr>
<td></td>
<td>0 = Data EE memory code protected</td>
</tr>
<tr>
<td>bit 7</td>
<td>LVP: Low Voltage ICSP Programming Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = RB3/PGM pin has PGM function, low voltage programming enabled</td>
</tr>
<tr>
<td></td>
<td>0 = RB3 is digital I/O, HV on MCLR must be used for programming</td>
</tr>
<tr>
<td>bit 6</td>
<td>BODEN: Brown-out Reset Enable bit(^{(2)})</td>
</tr>
<tr>
<td></td>
<td>1 = BOR enabled</td>
</tr>
<tr>
<td></td>
<td>0 = BOR disabled</td>
</tr>
<tr>
<td>bit 3</td>
<td>PWRT: Power-up Timer Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = PWRT disabled</td>
</tr>
<tr>
<td></td>
<td>0 = PWRT enabled</td>
</tr>
<tr>
<td>bit 2</td>
<td>WDTE: Watchdog Timer Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = WDT enabled</td>
</tr>
<tr>
<td></td>
<td>0 = WDT disabled</td>
</tr>
<tr>
<td>bit 1-0</td>
<td>FOSC1:FOSC0: Oscillator Selection bits</td>
</tr>
<tr>
<td></td>
<td>11 = RC oscillator</td>
</tr>
<tr>
<td></td>
<td>10 = HS oscillator</td>
</tr>
<tr>
<td></td>
<td>01 = XT oscillator</td>
</tr>
<tr>
<td></td>
<td>00 = LP oscillator</td>
</tr>
</tbody>
</table>

**Note 1:** Enabling Brown-out Reset automatically enables Power-up Timer (PWRT), regardless of the value of bit PWRT. Ensure the Power-up Timer is enabled any time Brown-out Reset is enabled.

**Note 2:** All of the CP1:CP0 pairs have to be given the same value to enable the code protection scheme listed.

**Legend:**
- R = Readable bit
- P = Programmable bit
- U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed
- u = Unchanged from programmed state
### REGISTER 3-2: CONFIG: CONFIGURATION WORD FOR PIC16F870/871/872 (ADDRESS 2007h)

<table>
<thead>
<tr>
<th>CP1</th>
<th>CP0</th>
<th>RESV</th>
<th>WRT</th>
<th>CPD</th>
<th>LVP</th>
<th>BODEN</th>
<th>CP1</th>
<th>CP0</th>
<th>PWRTE</th>
<th>WDTE</th>
<th>F0SC1</th>
<th>F0SC0</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-0</td>
<td>U-0</td>
<td>U-0</td>
<td>U-0</td>
<td>U-0</td>
<td>U-0</td>
<td>U-0</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 13-12</th>
<th><strong>CP1:CP0</strong>: FLASH Program Memory Code Protection bits(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Code protection off</td>
</tr>
<tr>
<td>10</td>
<td>Not supported</td>
</tr>
<tr>
<td>01</td>
<td>Not supported</td>
</tr>
<tr>
<td>00</td>
<td>0000h to 07FFh code protected</td>
</tr>
</tbody>
</table>

| bit 11    | **Reserved**: Set to ‘1’ for normal operation             |
| bit 10    | **Unimplemented**: Read as ‘1’                            |
| bit 9     | **WRT**: FLASH Program Memory Write Enable bit            |
|           | 1 = Unprotected program memory may be written to by EECON control |
|           | 0 = Unprotected program memory may not be written to by EECON control |

| bit 8     | **CPD**: Data EE Memory Code Protection bit               |
|           | 1 = Code protection off                                   |
|           | 0 = Data EE memory code protected                          |

| bit 7     | **LVP**: Low Voltage ICSP Programming Enable bit          |
|           | 1 = RB3/PGM pin has PGM function, low voltage programming enabled |
|           | 0 = RB3 is digital I/O, HV on MCLR must be used for programming |

| bit 6     | **BODEN**: Brown-out Reset Enable bit(2)                  |
|           | 1 = BOR enabled                                           |
|           | 0 = BOR disabled                                          |

| bit 3     | **PWRTE**: Power-up Timer Enable bit                      |
|           | 1 = PWRT disabled                                         |
|           | 0 = PWRT enabled                                          |

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

| bit 1-0   | **F0SC1:F0SC0**: Oscillator Selection bits                |
|           | 11 = RC oscillator                                       |
|           | 10 = HS oscillator                                       |
|           | 01 = XT oscillator                                       |
|           | 00 = LP oscillator                                       |

**Note 1**: Enabling Brown-out Reset automatically enables Power-up Timer (PWRT), regardless of the value of bit PWRTE. Ensure the Power-up Timer is enabled any time Brown-out Reset is enabled.

**Note 2**: All of the CP1:CP0 pairs have to be given the same value to enable the code protection scheme listed.

**Legend:**
- **R** = Readable bit
- **P** = Programmable bit
- **U** = Unimplemented bit, read as ‘0’
- **n** = Value when device is unprogrammed
- **u** = Unchanged from programmed state
4.0 EMBEDDING THE CONFIGURATION WORD AND ID INFORMATION IN THE HEX FILE

To allow portability of code, the 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, configuration word and ID information must be included. An option to not include this information may be provided.

Specifically for the PIC16F87X, the EEPROM data memory should also be embedded in the HEX file (see Section 2.2).

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

Checksum is calculated by reading the contents of the PIC16F87X memory locations and adding up the opcodes, up to the maximum user addressable location, e.g., 0x1FF for the PIC16F87X. 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 PIC16F87X devices is shown in Table 5-1.

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 are 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 5-1: Checksum Computation

<table>
<thead>
<tr>
<th>Device</th>
<th>Code Protect</th>
<th>Checksum*</th>
<th>Blank Value</th>
<th>0x25E6 at 0 and max address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC16F870</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x33FF</td>
<td>0xFFCD</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x3FCE</td>
<td>0x0B9C</td>
</tr>
<tr>
<td>PIC16F871</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x33FF</td>
<td>0xFFCD</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x3FCE</td>
<td>0x0B9C</td>
</tr>
<tr>
<td>PIC16F872</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x33FF</td>
<td>0xFFCD</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x3FCE</td>
<td>0x0B9C</td>
</tr>
<tr>
<td>PIC16F873</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x2BFF</td>
<td>0xF7CD</td>
</tr>
<tr>
<td>0x0F00 : 0xFFF</td>
<td>SUM[0x0000:0x0EFF] + CFGW &amp; 0x3BFF</td>
<td>0x48EE</td>
<td>0xFAA3</td>
<td></td>
</tr>
<tr>
<td>0x0800 : 0xFFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x3FDE</td>
<td>0xF193</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x37CE</td>
<td>0x039C</td>
</tr>
<tr>
<td>PIC16F874</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x2BFF</td>
<td>0xF7CD</td>
</tr>
<tr>
<td>0x0F00 : 0xFFF</td>
<td>SUM[0x0000:0x0EFF] + CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x48EE</td>
<td>0xFAA3</td>
<td></td>
</tr>
<tr>
<td>0x0800 : 0xFFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x3FDE</td>
<td>0xF193</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x37CE</td>
<td>0x039C</td>
</tr>
<tr>
<td>PIC16F876</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x1BFF</td>
<td>0xE7CD</td>
</tr>
<tr>
<td>0x1000 : 0x1FFF</td>
<td>SUM[0x0000:0x01EFF] + CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x28EE</td>
<td>0xDA93</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x27DE</td>
<td>0xD93</td>
</tr>
<tr>
<td>PIC16F877</td>
<td>OFF</td>
<td>SUM[0x0000:0x07FF] + CFGW &amp; 0x3BFF</td>
<td>0x1BFF</td>
<td>0xE7CD</td>
</tr>
<tr>
<td>0x1000 : 0x1FFF</td>
<td>SUM[0x0000:0x01EFF] + CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x28EE</td>
<td>0xDA93</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>CFGW &amp; 0x3BFF + SUM_ID</td>
<td>0x27CE</td>
<td>0xF39C</td>
</tr>
</tbody>
</table>

Legend: CFGW = Configuration Word  
SUM[a:b] = [Sum of locations a to b inclusive]  
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 the individual expressions] MODULO [0xFFFF]  
 Addition  
& = Bitwise AND
## 6.0 PROGRAM/VERIFY MODE ELECTRICAL CHARACTERISTICS

### TABLE 6-1: TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE

<table>
<thead>
<tr>
<th>AC/DC CHARACTERISTICS</th>
<th>Standard Operating Conditions (unless otherwise stated)</th>
<th>Conditions/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Operating Temperature: 0°C ≤ TA ≤ +70°C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operating Voltage: 2.2V ≤ VDD ≤ 5.5V</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Sym</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Units</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Limited command set</td>
</tr>
<tr>
<td>VDD level for Algorithm 1</td>
<td>VDD</td>
<td>2.2</td>
<td>5.5</td>
<td>V</td>
<td></td>
<td>(See Table 2-2)</td>
</tr>
<tr>
<td>VDD level for Algorithm 2</td>
<td>VDD</td>
<td>4.5</td>
<td>5.5</td>
<td>V</td>
<td></td>
<td>All commands available</td>
</tr>
<tr>
<td>High voltage on MCLR for high voltage programming entry</td>
<td>VHH</td>
<td>VDD + 3.5</td>
<td>13.5</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voltage on MCLR for low voltage ICSP programming entry</td>
<td>ViH</td>
<td>2.2</td>
<td>5.5</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCLR rise time (Vss to Vih) for Test mode entry</td>
<td>tVHHR</td>
<td>1.0</td>
<td>μs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(RB6, RB7) input high level</td>
<td>ViH1</td>
<td>0.8 VDD</td>
<td>V</td>
<td></td>
<td>V</td>
<td>Schmitt Trigger input</td>
</tr>
<tr>
<td>(RB6, RB7) input low level</td>
<td>ViL1</td>
<td>0.2 VDD</td>
<td>V</td>
<td></td>
<td>V</td>
<td>Schmitt Trigger input</td>
</tr>
<tr>
<td>RB&lt;7:6&gt; setup time before MCLR ↑</td>
<td>tset0</td>
<td>100</td>
<td>ns</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB&lt;7:6&gt; hold time after MCLR ↑</td>
<td>thld0</td>
<td>5</td>
<td>μs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB3 setup time before MCLR ↑</td>
<td>tset2</td>
<td>100</td>
<td>ns</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial Program/Verify</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data in setup time before clock ↓</td>
<td>tset1</td>
<td>100</td>
<td>ns</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data in hold time after clock ↓</td>
<td>thld1</td>
<td>100</td>
<td>ns</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data input not driven to next clock input (delay required between command/data or command/command)</td>
<td>tdl1</td>
<td>1.0</td>
<td>μs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay between clock ↓ to clock ↑ of next command or data</td>
<td>tdl2</td>
<td>1.0</td>
<td>μs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock ↑ to data out valid (during read data)</td>
<td>tdl3</td>
<td>80</td>
<td>ns</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erase cycle time</td>
<td>tera</td>
<td>2</td>
<td>4</td>
<td>ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming cycle time</td>
<td>tprog</td>
<td>2</td>
<td>4</td>
<td>ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FIGURE 6-1: LOAD DATA COMMAND MCLR = VIHH (PROGRAM/VERIFY)

FIGURE 6-2: READ DATA COMMAND MCLR = VIHH (PROGRAM/VERIFY)

FIGURE 6-3: INCREMENT ADDRESS COMMAND MCLR = VIHH (PROGRAM/VERIFY)
FIGURE 6-4: LOAD DATA COMMAND MCLR = VDD (PROGRAM/VERIFY)

FIGURE 6-5: READ DATA COMMAND MCLR = VDD (PROGRAM/VERIFY)

FIGURE 6-6: INCREMENT ADDRESS COMMAND MCLR = VDD (PROGRAM/VERIFY)
Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products 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 PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be 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 product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, FilterLab, KEELOQ, microID, MPLAB, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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

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

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

Printed on recycled paper.
### AMERICAS

**Corporate Office**  
2335 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200 Fax: 480-792-7277  
Technical Support: 480-792-7627  
Web Address: http://www.microchip.com  

**Rocky Mountain**  
2335 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7966 Fax: 480-792-7456  

**Atlanta**  
500 Sugar Mill Road, Suite 200  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0070  

**Boston**  
2 Lan Drive, Suite 120  
Westford, MA 01886  
Tel: 978-692-3848 Fax: 978-692-3821  

**Chicago**  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075  

**Dallas**  
4570 Westgrove Drive, Suite 160  
Addison, TX 75001  
Tel: 972-818-7924 Fax: 972-818-2924  

**Detroit**  
Tri-Atrie Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48304  
Tel: 248-538-2250 Fax: 248-538-2260  

**Kokomo**  
2767 S. Albright Road  
Kokomo, Indiana 46902  
Tel: 765-864-8360 Fax: 765-864-8387  

**Los Angeles**  
18201 Von Karman, Suite 1090  
Irvine, CA 92660  
Tel: 949-263-1888 Fax: 949-263-1338  

**New York**  
150 Motor Parkway, Suite 202  
Hauppauge, NY 11788  
Tel: 631-273-5050 Fax: 631-273-5355  

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

**Toronto**  
6285 Northdale Drive, Suite 108  
Mississauga, Ontario L4V 1X5, Canada  
Tel: 905-673-0699 Fax: 905-673-6509  

### ASIA/PACIFIC

**Australia**  
Microchip Technology Australia Pty Ltd  
Suite 22, 41 Lawson Street  
Epping 2121, NSW  
Australia  
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755  

**China - Beijing**  
Microchip Technology Consulting (Shanghai) Co., Ltd.  
Beijing Liaison Office  
Unit 915  
Bei Hai Wen Tai Bldg. No. 8 TIDU Street  
Beijing, 100027, No. China  
Tel: 68-10-85282100 Fax: 68-10-85282104  

**China - Chengdu**  
Microchip Technology Consulting (Shanghai) Co., Ltd.  
Chengdu Liaison Office  
Rm. 2401, 24th Floor, Ming Xing Financial Tower  
No. 88 TIDU Street  
Chengdu 610016, China  
Tel: 86-28-6766200 Fax: 86-28-6766599  

**China - Fuzhou**  
Microchip Technology Consulting (Shanghai) Co., Ltd.  
Fuzhou Liaison Office  
No. 71 Wush Road  
Fuzhou 350001, China  
Tel: 86-591-7503506 Fax: 86-591-7503521  

**China - Shanghai**  
Microchip Technology Consulting (Shanghai) Co., Ltd.  
Shenzhen Liaison Office  
Unit 28F, World Trade Plaza  
Rm. 28F, World Trade Plaza  
Shanghai, 200051  
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060  

**China - Shenzhen**  
Microchip Technology Consulting (Shanghai) Co., Ltd.  
Shenzhen Liaison Office  
Room 701, Bldg. B  
Far East International Plaza  
No. 317 Xian Xia Road  
Shanghai, 200051  
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060  

### Japan

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

### Korea

**Korea**  
Microchip Technology Korea  
68-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea 135-882  
Tel: 82-2-554-7200 Fax: 82-2-558-5934  

### Singapore

**Singapore**  
Microchip Technology Singapore Pte Ltd.  
200 Middle Road  
#07-02 Prime Centre  
Singapore, 188980  
Tel: 65-6334-8870 Fax: 65-6334-8850  

### Taiwan

**Taiwan**  
Microchip Technology Taiwan  
11F-3, No. 207  
Tung Hua North Road  
Taipei, 105, Taiwan  
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139  

### EUROPE

**Danmark**  
Microchip Technology Nordic ApS  
Regus Business Centre  
Lastrup høj 1-3  
Ballapur DK-2750 Denmark  
Tel: 45 4420 9895 Fax: 45 4420 9910  

**France**  
Microchip Technology SARL  
Parc d’Activite du Moulin de Massy  
43 Rue du Saule Trapu  
Batiment A - ler Etage  
91900 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79  

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

**Italy**  
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  

**United Kingdom**  
Microchip Technology Ltd.  
505 Eskdale Road  
Winnersh Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44 118 921 5869 Fax: 44 118 921 5820  

03/01/02