MICROCHIP

MPLAB® ICD 2 In-Circuit Debugger User's Guide

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contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates.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, KEELOQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Accuron, dsPIC, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICkit, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerTool, rfPIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn 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.

Printed on recycled paper.

Table of Contents

Customer Support

Chapter 1. How MPLAB ICD 2 Works

Chapter 2. Getting Started

Chapter 3. MPLAB ICD 2 Tutorial

Chapter 4. USB Installation Details

Chapter 5. Troubleshooting

Chapter 6. FAQs

Customer Support

INTRODUCTION

This chapter details how to obtain support for the product(s) mentioned in this book.

HIGHLIGHTS

Topics covered in this guide:

- The Microchip Web Site
- Customer Change Notification Service
- Contact Information

THE MICROCHIP WEB SITE

On-line support is provided on the Microchip web site at:

http://www.microchip.com

The web site is designed to make files and information easily available to customers. To access the site, the user must have Internet access and a web browser, such as Netscape® Navigator or Microsoft® Explorer®. Files are also available for FTP download from an FTP site.

The file transfer site is available using an FTP program/client connected to:

ftp://ftp.microchip.com

Users may download files for the latest development tools, data sheets, application notes, user's guides, articles and sample programs. A list of Microchip sales offices and distributors is also available.

Technical Support

- Frequently Asked Questions (FAQ)
- Online Discussion Groups – Conferences for products, development systems, technical information and more
- Microchip Consultant Program Member Listing
- Links to other web sites related to Microchip products

Developer's Toolbox

- Design Tips
- Device Errata

Additional Information

- Latest Microchip Press Releases
- Listing of Seminars and Events
- Job Postings

CUSTOMER CHANGE NOTIFICATION SERVICE

The customer notification service helps customers stay current on Microchip products. After subscribing, an e-mail notification will be sent whenever a change, update, revision or errata is released related to the specific product family or development tool. Access the Microchip web site (www.microchip.com) and click on "Customer Change Notification", then follow the instructions to register.

The Development Systems product categories are listed below:

Compilers – Current information on Microchip C compilers and other language tools. Including: MPLAB® C17, MPLAB C18 and MPLAB C30 C Compilers; MPASM and MPLAB ASM30 assemblers; MPLINK™ and MPLAB LINK30 linkers; and MPLIB™ and MPLAB LIB30 librarians.

Emulators – Current information on Microchip in-circuit emulators. Including: MPLAB ICE 2000 and MPLAB ICE 4000.

In-circuit Debuggers – Current information on Microchip in-circuit debuggers. Including: MPLAB ICD and MPLAB ICD 2.

MPLAB IDE – Current information on Microchip MPLAB IDE, the Windows® Integrated Development Environment for development systems tools. This list is focused on the MPLAB IDE, MPLAB SIM simulator, MPLAB IDE Project Manager and general editing and debugging features.

Programmers – Current information on Microchip PICmicro® MCU device programmers. Including: PRO MATE® II device programmer and PICSTART® Plus development programmer.

CONTACT INFORMATION

Microchip product support is available through:

- Distributors
- Local Sales Offices – a list of sales office locations is available at the end of this book.
- Field Application Engineers (FAE)
- Corporate Applications Engineers (CAEs) may be contacted at (480) 792-7627.
- Hot Line

The **Systems Information and Upgrade Line** provides system users with a list of the current versions of all Microchip development systems software products. This line also provides information on how to receive available upgrade kits.

The Hot Line Numbers are:

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

1-480-792-7302 for the rest of the world.

NOTES:

Chapter 1. How MPLAB ICD 2 Works

1.1 INTRODUCTION

This section provides a simplified description of how MPLAB ICD 2 works. It is intended to provide enough information so a target board can be designed that is compatible with MPLAB ICD 2 for both programming and debugging operations. The basic theory of programming and in-circuit debugging is described so that problems, if encountered, are quickly resolved.

1.2 HIGHLIGHTS

This section provides an overview of MPLAB ICD 2, followed by a description of MPLAB ICD 2 programming and debugging modes. The following topics are covered:

- ICD vs. ICE
- Modular Interface Connections
- Debug Mode
- Requirements for Debug Mode
- Debug Reserved Resources
- Programmer Mode

1.3 ICD vs. ICE

The in-circuit debugger (ICD) is a cost-efficient alternative to an in-circuit emulator (ICE). It can do many things that were previously done only with more expensive hardware, but the cost benefits come with a trade-off of some of the conveniences of an in-circuit emulator. If users are willing to design their application to be ICD compatible, they can enjoy the benefits of a low-cost hardware debugger.

As opposed to an ICE, some of the requirements of the in-circuit debugger are:

- The in-circuit debugger requires exclusive use of some hardware and software resources of the target.
- The target PICmicro MCU must have a functioning clock and be running.
- The ICD can debug only when all the links in the system are fully functional.

An emulator provides memory and a clock, and can run code – even without being connected to the target application board. During the development and debugging cycle, an ICE provides the most power to get the system fully functional, whereas an ICD may not be able to debug at all if the application does not run. On the other hand, an in-circuit debug connector can be placed on the application board and connected to an ICD even after the system is in production, allowing easy testing, debugging and reprogramming of the application. Even though an ICD has some drawbacks in comparison to an ICE, in this situation it has some distinct advantages:

- A connection to the application after the production cycle does not require extraction of the microcontroller in order to insert an ICE probe.
- The ICD can re-program the firmware in the target application without any other connections or equipment.

Note: An ICE uses custom hardware to emulate the target microcontroller. An ICD uses hardware on the target microcontroller to do some of the functions of an ICE. An ICD also employs software running on the target to do ICE-like functions and, as a result, relies upon the target microcontroller for some memory space, CPU control, stack storage and I/O pins for communication.

1.4 MODULAR INTERFACE CONNECTIONS

MPLAB ICD 2 is connected to the target PICmicro MCU with the modular interface cable, a six conductor cable. The pin numbering for the MPLAB ICD 2 connector is shown from the bottom of the target PC board in Figure 1-1.

Note: The ICD cable has mirror imaged connections on each end, and connections on the MPLAB ICD 2 module are the opposite of connections shown here.

FIGURE 1-1: PIN NUMBERING FOR MODULAR CONNECTOR

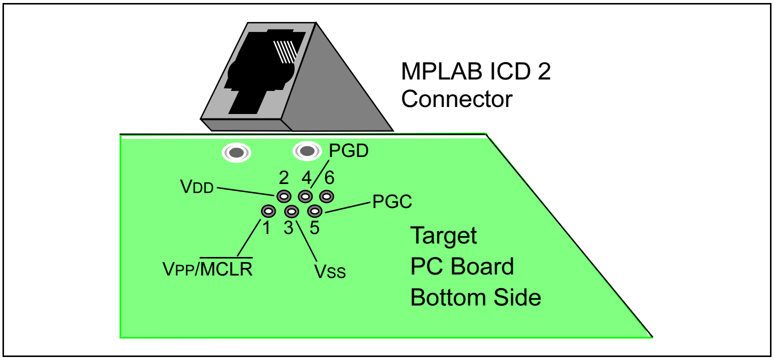
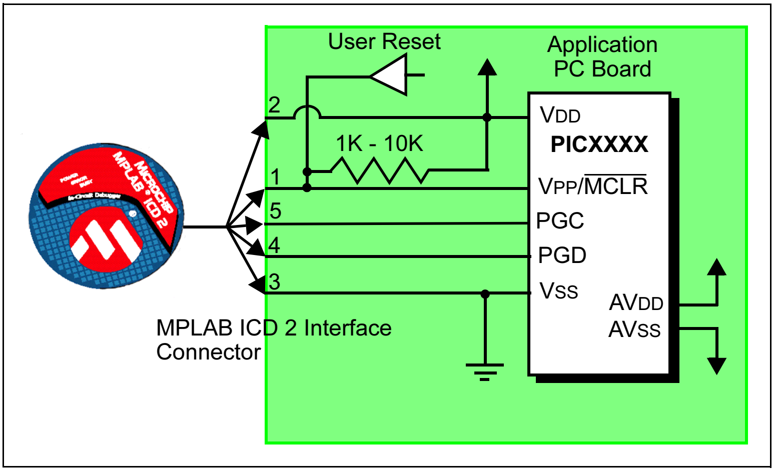


Figure 1-2 shows the interconnections of the MPLAB ICD 2 to the modular connector on the target board. There are six pins on the ICD connector, but only five are used. The diagram also shows the wiring from the connector to the PICmicro MCU device on the target PC board. A pull-up resistor (usually around 10k Ohm) is recommended to be connected from the VPP/$\overline{\text{MCLR}}$ line to VDD so that the line may be strobed low to reset the PICmicro MCU.

FIGURE 1-2: MPLAB ICD 2 CONNECTIONS TO TARGET BOARD



Although pin 2 (VDD) can supply a limited amount of power to the target application under certain conditions, for the purposes of these descriptions, pins 2 and 3 (VSS) are omitted. They are shown on the diagram for completeness, but in the following descriptions only three lines are active and relevant to core MPLAB ICD 2 operation: VPP/$\overline{\text{MCLR}}$, PGC and PGD.

Note: In the following discussions, VDD is ignored. But be aware that the target VDD is also used to power the output drivers in MPLAB ICD 2. This allows level translation for target low voltage operation. If MPLAB ICD 2 does not have voltage on its VDD line (pin 2 of the ICD connector), either from power being supplied to the target by MPLAB ICD 2 or from a separate target power supply, it will not operate.

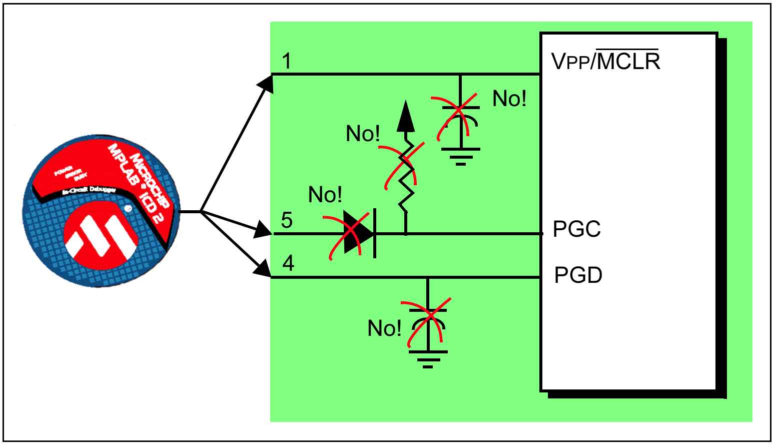
Not all PICmicro MCUs have the AVDD and AVSS lines, but if they are present on the target PICmicro MCU, all must be connected in order for MPLAB ICD 2 to operate.

The interconnection is very simple, any problems experienced are often caused by other connections or components on these critical lines that interfere with the operation of MPLAB ICD 2, as discussed in Section 1.4.1.

1.4.1 Circuits That Will Prevent MPLAB ICD 2 From Functioning

Figure 1-3 shows the active MPLAB ICD 2 lines with some components that will prevent MPLAB ICD 2 from functioning:

FIGURE 1-3: IMPROPER CIRCUIT COMPONENTS



Specifically, these guidelines must be followed:

- No pull-ups on PGC/PGD – they will divide the voltage levels, since these lines have 4.7KΩ pull-down resistors in MPLAB ICD 2.
- No capacitors on PGC/PGD – they will prevent fast transitions on data and clock lines during programming and debug communications.
- No capacitors on $\overline{\text{MCLR}}$ – they will prevent fast transitions of VPP. A simple pull-up resistor is generally sufficient.
- No diodes on PGC/PGD – they will prevent bidirectional communication between MPLAB ICD 2 and the target PICmicro MCU.

1.5 DEBUG MODE

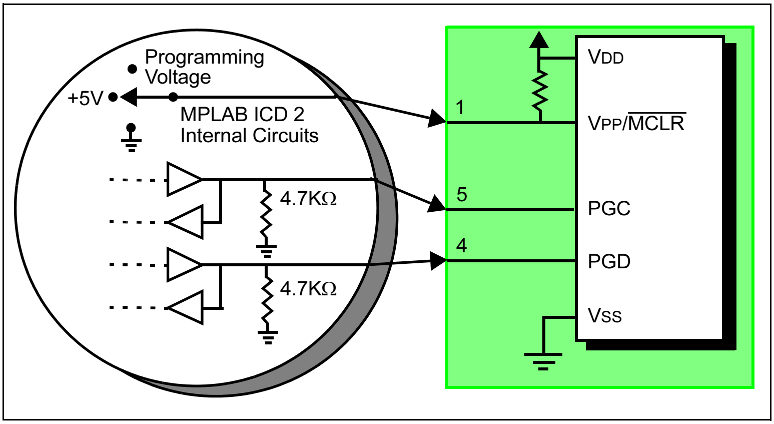
There are two steps to using MPLAB ICD 2 as a debugger. The first requires that an application be programmed into the target PICmicro MCU. The second uses the internal in-circuit debug hardware of the target FLASH PICmicro MCU to run and test the application program. These two steps are directly related to the MPLAB IDE operations:

1. Programming the code into the target.
2. Using the debugger to set breakpoints and run.

If the target PICmicro MCU cannot be programmed correctly, MPLAB ICD 2 will not be able to debug.

Figure 1-4 shows the basic interconnections required for programming. Note that this is the same as Figure 1-2, but for the sake of clarity, the VDD and VSS lines from MPLAB ICD 2 are not shown.

FIGURE 1-4: PROPER CONNECTIONS FOR PROGRAMMING



A simplified diagram of some of the internal interface circuitry of the MPLAB ICD 2 is shown. For programming, no clock is needed on the target PICmicro MCU, but power must be supplied. When programming, MPLAB ICD 2 puts programming levels on VPP, sends clock pulses on PGC and serial data via PGD. To verify that the part has been programmed correctly, clocks are sent to PGC and data is read back from PGD. This conforms to the ICSP™ protocol of the PICmicro MCU under development.

1.6 REQUIREMENTS FOR DEBUG MODE

To debug (set breakpoints, see registers, etc.) with the MPLAB ICD 2 there are critical elements that must be working correctly:

- MPLAB ICD 2 must be connected to a PC. It must be powered by an external power supply or the PC via the USB cable, and it must be communicating with MPLAB IDE software via the RS-232 or USB cable. See the on-line help for details.
- The MPLAB ICD 2 must be connected as shown to the VPP, PGC and PGD pins of the target PICmicro MCU with the modular interface cable (or equivalent). VSS and VDD are also required to be connected between the MPLAB ICD 2 and target PICmicro MCU.
- The target PICmicro MCU must have power and a functional, running oscillator. If the target PICmicro MCU does not run – for whatever reason – MPLAB ICD 2 cannot debug.
- The target PICmicro MCU must have its configuration words programmed correctly:
  - The oscillator configuration bits should correspond to RC, XT, RC, etc., depending upon the target design.
  - The target PICmicro MCU must not have the Watchdog Timer enabled.
  - The target must not have code protection enabled.
  - The target must not have table read protection enabled.

1.6.1 Sequence of Operations Leading to Debug Mode

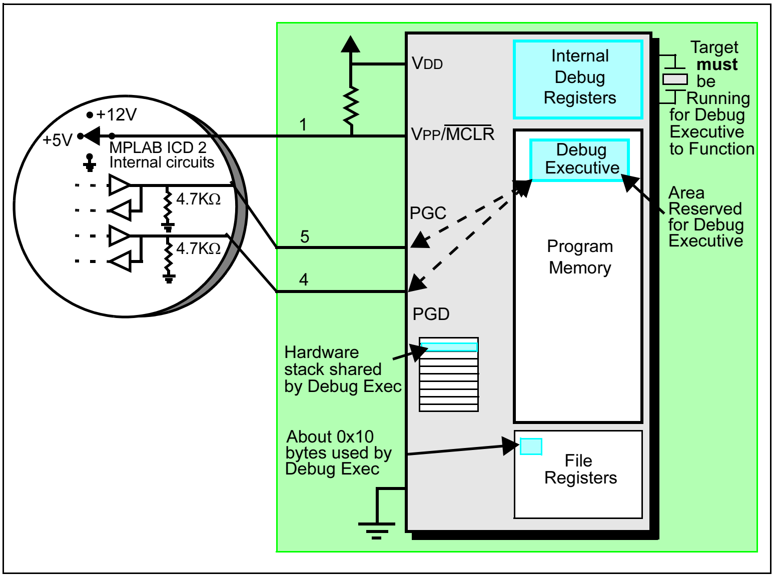
If the above conditions are met, then these actions can be performed when MPLAB ICD 2 is set as the current debugger (*Debugger>Select Tool*):

- When *Debugger>Program* is selected, the application code is programmed into the PICmicro MCU's memory via the ICSP protocol as described above.
- A small "debug executive" program is loaded into the high area of program memory of the target PICmicro MCU. Since the debug executive must reside in program memory, the application program must not use this reserved space. The debug executive typically needs about 0x120 words of program memory.
- Special "in-circuit debug" registers in the target PICmicro MCU are enabled. These allow the debug executive to be activated by the MPLAB ICD 2.
- The target PICmicro MCU is held in reset by keeping the VPP/$\overline{\text{MCLR}}$ line low.

1.6.2 Debug Mode Details

Figure 1-5 illustrates the MPLAB ICD 2 ready for debugging.

FIGURE 1-5: MPLAB ICD 2 READY FOR DEBUGGING


+12V
+5V
MPLAB ICD 2
Internal circuits
4.7KΩ
4.7KΩ
1
5
4
VDD
VPP/MCLR
PGC
PGD
Internal Debug Registers
Debug Executive
Program Memory
File Registers
Target **must** be Running for Debug Executive to Function
Area Reserved for Debug Executive
Hardware stack shared by Debug Exec
About 0x10 bytes used by Debug Exec

Typically, in order to find out if an application program will run correctly, a breakpoint is set early in the program code. When a breakpoint is set from the user interface of the MPLAB IDE, the address of the breakpoint is stored into the special internal debug registers of the target PICmicro MCU. Commands on PGC and PGD communicate directly to these registers to set the breakpoint address.

Next, the *Debugger>Run* function or the Run icon (forward arrow) is usually pressed from MPLAB IDE. MPLAB ICD 2 will raise the VPP/MCLR line to allow the target to run, the target will start from address zero and execute until the program counter reaches the breakpoint address previously stored in the internal debug registers.

After the instruction at the breakpoint address is executed, the in-circuit debug mechanism of the target PICmicro MCU “fires” and transfers the PICmicro MCU’s program counter to the debug executive (much like an interrupt) and the user’s application is effectively halted. MPLAB ICD 2 communicates with the debug executive via PGC and PGD, gets the breakpoint status information and sends it back to the MPLAB IDE. The MPLAB IDE then sends a series of queries to MPLAB ICD 2 to get information about the target PICmicro MCU, such as file register contents and the state of the CPU. These queries are ultimately performed by the debug executive.

The debug executive runs just like an application in program memory. It uses some locations on the hardware stack (usually just one or two) and, typically, about fourteen file registers for its temporary variables. If the PICmicro MCU does not run – for whatever reason; no oscillator, faulty power supply, shorts on the target board, etc. – then the debug executive cannot communicate back to the MPLAB ICD 2 and MPLAB will issue an error message.

Another way to get a breakpoint is to press the MPLAB IDE’s “Halt” button (the “pause” symbol to the right of the Run arrow). This toggles the PGC and PGD lines in such a way that the in-circuit debug mechanism of the target PICmicro MCU switches the program counter from the user’s code in program memory to the debug executive. Again, the target application program is effectively halted, and MPLAB IDE uses MPLAB ICD 2 communications with the debug executive to interrogate the state of the target PICmicro MCU.

1.7 DEBUG RESERVED RESOURCES

When developing with MPLAB ICD 2, the user must be aware of the device resources reserved for debugging. Check the MPLAB IDE on-line help or the README file of the current software to find out details of the processor being used. In general, in-circuit debugging uses the following on-chip resources:

- $\overline{\text{MCLR}}$/V_{PP} is shared for programming.
- Low-voltage programming (LVP) is disabled.
- PGC and PGD are reserved for programming and in-circuit debugging.
- One or two stack locations are used by the MPLAB ICD 2 debugger.
- Several general purpose file registers are reserved.
- Upper words of program memory area are reserved for the debug executive code.
- The PIC18FXXXX MCU shadow stack is not available while debugging. The application can use the shadow stack, but cannot debug a return from a fast interrupt or fast CALL routine because the ICD will overwrite the shadow stack when it gets a breakpoint.

1.8 PROGRAMMER MODE

When using the *Programmer>Program* selection to program a device, the in-circuit debug registers should be disabled in the MPLAB IDE so the MPLAB ICD 2 will program only the target application code and the configuration bits (and EEPROM data, if available and selected) into the target PICmicro MCU. The debug executive will not be loaded. In this mode the MPLAB ICD 2 can only toggle the $\overline{\text{MCLR}}$ line to reset and start the target. A breakpoint cannot be set, and register contents cannot be seen or altered.

The MPLAB ICD 2 programs the target using ICSP. No clock is required while programming, and all modes of the processor can be programmed, including code protect, Watchdog Timer enabled and table read protect.

Note: A header board is required to debug smaller pin count parts with the MPLAB ICD 2. These parts can be programmed without the header by connecting the V_{PP}, PGC and PGD lines as described previously.

NOTES:

Chapter 2. Getting Started

2.1 INTRODUCTION

This chapter will describe the use of MPLAB IDE software with MPLAB ICD 2 and will explain the power options and recommended power-up sequences. Chapter 3 is a step-by-step tutorial using MPLAB ICD 2. Before continuing on to the tutorial, be sure that the software and hardware is configured according to Sections 2.3 through 2.5 of this chapter.

2.2 HIGHLIGHTS

Topics covered in this chapter:

- MPLAB ICD 2 System Components
- Installing and Configuring MPLAB IDE for MPLAB ICD 2
- Applying Power
- Overview: Setting Up the Environment
- Overview: MPLAB ICD 2 Development Process

2.3 MPLAB ICD 2 SYSTEM COMPONENTS

In addition to the MPLAB ICD 2 module, the following components are required:

- MPLAB IDE software (version 6.20 or later) – Installed on the PC to control MPLAB ICD 2.
- RS-232 or USB cable – To connect the MPLAB ICD 2 module to a COM or USB port on the PC.

Note:	Do not connect the USB cable until installation of the MPLAB IDE software and instructions for configuring the USB driver have been completed.

- Modular interface cable – To connect the MPLAB ICD 2 module to a demo board or the user's application.
- Demo board or target application – To connect the PICmicro MCU with on-board debug capabilities to the modular interface (and the MPLAB ICD 2). Although the serial or USB communications from the MPLAB IDE to the MPLAB ICD 2 can be set up without a target connection, MPLAB ICD 2 will not function as a debugger without being connected to a target.
- Power adapter(s) – To power MPLAB ICD 2 and the target application.

2.4 INSTALLING AND CONFIGURING MPLAB IDE FOR MPLAB ICD 2

Note:	When using the USB cable, do not connect it before installing MPLAB IDE software.

To install the MPLAB IDE software, first acquire the latest MPLAB IDE installation executable (`MP6xxxx.exe`, where `6xxxx` represents the version of MPLAB IDE) from either the Microchip web site (www.microchip.com) or the MPLAB IDE CD-ROM (DS51123).

2.4.1 Setting Up Communications

Follow the instructions supplied with MPLAB IDE and install the USB driver from the `DriversXX` folder under MPLAB IDE. See Chapter 4, "USB Installation Details."

If using the RS-232 cable, connect it to the MPLAB ICD 2 and the PC.

> **Note:** Do not connect both USB and RS-232 cables to MPLAB ICD 2.

2.4.2 Starting MPLAB IDE

After installing the MPLAB IDE software, invoke it by using any of these methods:

- Select *Start>Programs>Microchip MPLAB IDE>MPLAB IDE*.
- Double click the MPLAB IDE desktop icon.
- Execute the file `mplab.exe` in the `\dlls` subdirectory of the MPLAB IDE installation directory.

> **Note:** For more information on using the MPLAB IDE software, see
>
> -"MPLAB v6.xx Quick Start," DS51281
>
> -The on-line help for MPLAB IDE
>
> -The file `Readme for MPLAB IDE.txt` included in the MPLAB IDE installation directory.
>
> -Also see the file `Readme for MPLAB ICD 2.txt` for the most current information about MPLAB ICD 2.

After starting MPLAB IDE, it will need to be configured for MPLAB ICD 2 use:

1. Select a PICmicro device supported by MPLAB ICD 2.
2. Set MPLAB ICD 2 as the current debugger.
3. Configure the MPLAB ICD 2 RS-232 or USB interface.

2.4.3 Select the Device

Use the Device Selection dialog (*Configure>Select Device*) to select the device to be debugged with the MPLAB ICD 2. Devices supported by MPLAB ICD 2 will have a "green light" icon next to MPLAB ICD 2 under Microchip Tool Support. Devices not supported will have a "red light" icon. A "yellow light" icon indicates that a part may be selected, but it is available with possible limitations and may not be suitable for production work, but is released for preliminary use with MPLAB ICD 2.

2.4.4 Set MPLAB ICD 2 as the Debug Tool

Select *Debugger>Select Tool>MPLAB ICD 2* to choose MPLAB ICD 2 as the debug tool. The Debugger menu and MPLAB IDE toolbar will change to display debug options once the tool is selected. Also, the Output window will open and messages concerning ICD status and communications will be displayed on the MPLAB ICD 2 tab.

> **Note:** MPLAB ICD 2 may be selected as either a debug tool (*Debugger* Menu) or a programmer (*Programmer* menu). Do not select both.

2.4.5 Configure the Interface

Go to the *Debugger>Settings* Communications tab to select the USB interface or the correct COM port for RS-232 communications.

2.5 APPLYING POWER

There are a number of configurations for powering MPLAB ICD 2 and the target. The following are configuration essentials:

- When using the USB connection, MPLAB ICD 2 can be powered from the PC but a power supply must be connected to the target.
- When using the RS-232 connection to the PC, MPLAB ICD 2 must have a power supply connected to it.
- When MPLAB ICD 2 has its own power supply, it can provide a limited amount of current, up to 200 mA, at five volts to a small target board.
- MPLAB ICD 2 cannot be powered from the target board.
- Power should be applied to MPLAB ICD 2 before applying power to the target.

2.5.1 Power Sequence when MPLAB ICD 2 is Providing Power to the Target

Use this sequence to power the target board from MPLAB ICD 2. This configuration allows up to 200 mA of current at 5 volts only.

1. Apply power to MPLAB ICD 2. DO NOT power the target.
2. Start MPLAB IDE 6.xx.
3. Under the Debugger menu of MPLAB IDE, select **Connect**.
4. After establishing communications with the MPLAB ICD 2, select *Debugger>Settings*.
5. In the Settings dialog, click the **Power** tab and ensure that the check box for “Power target circuit from MPLAB ICD 2” is checked. Click **OK**.

> **Note:** If any of the self tests on the **Status** tab of the **Settings** dialog do not pass, it will not be possible to erase and program chips. This will require further troubleshooting. See Chapters 5 and 6 or the on-line help for more information.

2.5.2 Power Sequence when Target has a Separate Power Supply

Use this sequence to power the target board from its own power supply. For this configuration, the target power can be 2-5 volts at higher currents than can be provided by the MPLAB ICD 2 (>200 mA). Check the PICmicro MCU data sheet to verify the operational voltage range for the device being used.

1. Apply power to MPLAB ICD 2. DO NOT power the target.
2. Start MPLAB IDE 6.xx.
3. Under the Debugger menu of MPLAB IDE, select **Connect**.
4. After establishing communications with the MPLAB ICD 2, select *Debugger>Settings*.
5. In the Settings dialog, click the **Power** tab and ensure that the check box for "Power target circuit from MPLAB ICD 2" is NOT checked. Click **OK**.
6. Power the target system and then select *Debugger>Connect*.

> **Note:** If any of the self tests on the **Status** tab of the **Settings** dialog do not pass, it will not be possible to erase and program chips. This will require further troubleshooting. See Chapters 5 and 6 or on-line help for more information.

2.6 OVERVIEW: SETTING UP THE ENVIRONMENT

The rest of this chapter provides an overview of MPLAB ICD 2 usage, generally describing the functions and menus. This section will provide a quick summary on MPLAB ICD 2 operation.

2.6.1 Setting Debugging and Programming Options

Various dialogs allow the user to set debugging and programming options:

- Configuration Bits dialog (*Configure>Configuration Bits*) – Select the configuration bits on the PICmicro processor. For complete details about these options, see the Special Features – Configuration Bits section of the data sheet for the device being programmed.

> **Note:** Configuration bits can be specified in the source code instead of setting them in this dialog. When doing this, every time the project is rebuilt, the configuration bits will get reset to the values specified in the source code.

- Program tab, ICD Settings dialog (*Debugger>Settings* or *Programmer>Settings*) – Set up programming options (Select Memories, Program and External Memory Ranges, ID, and Program Options, Erase All). If programming the ID bits is desired, set the value to be programmed in the *Configure>ID Memory* dialog.
- Power tab, ICD Settings dialog (*Debugger>Settings* or *Programmer>Settings*) – Check MPLAB ICD 2 and target voltage values and select “Power the target from the MPLAB ICD 2” if a power supply is connected to MPLAB ICD 2 (5V, up to 200mA).

- Status tab, ICD Settings dialog (*Debugger>Settings* or *Programmer>Settings*) – In addition to auto connect on startup, output messages may be logged to a file in this dialog. These messages may assist users or, if necessary, Microchip technical support in finding errors.

2.6.2 Creating a Project

The easiest way to create a new project is to select *Project>Project Wizard*. With the help of the Project Wizard, a new project and the language tools for building that project can be created. The wizard will guide users through the process of adding source files, libraries, linker scripts, etc. to the various "nodes" on the project window. The tutorial in Chapter 3 gives detailed steps for using this wizard.

2.6.3 Building a Project

After the project is created, choose *Project>Build All* to build the application. This will create object code for the application that can be programmed into the target with MPLAB ICD 2.

2.7 OVERVIEW: MPLAB ICD 2 DEVELOPMENT PROCESS

The development process with MPLAB ICD 2 consists of four steps:

1. Program the application code into the target application, using MPLAB ICD 2 as the current debug tool.
2. Debug the application.
3. Modify the source code, rebuild the project and repeat until the application performs as designed.
4. Select MPLAB ICD 2 as the current programmer and program the part for use in the target application.

2.7.1 Programming the Target Processor for Debugging

To program the application project's code into the MPLAB ICD 2 for debugging, follow these steps:

1. Select *Debugger>Settings* and then click the Program tab to set the Start and End address of Program Memory for the application.
2. Select *Configure>Configuration Bits* and set the configuration bit settings appropriate to the target.
3. Select *Debugger>Program*.

2.7.2 Debugging The Application

To execute the code in real-time:

1. Open the source file (double click on the file name in the Project Window or use *File>Open*) or program memory window (*View>Program Memory*) for viewing.
2. Select *Debugger>Run* (or click the toolbar Run button).

The processor will run until a breakpoint is reached or until the processor is halted by selecting *Debugger>Halt* (or clicking the toolbar Halt button).

Select *Debugger>Breakpoints* to specify the location for the breakpoint. Or, right-click on the line of source code or program memory where the break is desired and select *Set Break Point*. MPLAB ICD 2 allows only one breakpoint at a time.

After the processor is halted, the code can be executed in step mode. To step through a single statement of code, select *Debugger>Step Into* (or click the toolbar Step Into button). The processor halts after each step.

> **Note:** When single-stepping, interrupts are disabled, so an interrupt routine will never be entered. A breakpoint can be set inside an interrupt routine and then run to it to single step through the interrupt code.

2.7.3 Modifying Target Application Code and Rebuilding the HEX File

To modify the code and rebuild the HEX file:

1. Open the source file (double click on the file name in the Project Window or use *File>Open*).
2. Make the necessary changes to debug the code.
3. Rebuild the HEX file using *Project>Build All*.
4. Select *Debugger>Program* to program the device with the updated HEX file.

2.7.4 Finishing the Application: Using MPLAB ICD 2 as a Programmer

Once the user has finished debugging the code and has the application running as desired, the device can be programmed without the debugger enabled. The device resources reserved for ICD operation then will be free for other use.

> **Note:** MPLAB ICD 2 may be selected as either a debug tool or a programmer. Do not select both.

First, set the Debugger to "None" from the *Debugger>Select Tool* menu.

Then, select *Programmer>Select Programmer>MPLAB ICD 2* to choose MPLAB ICD 2 as a programmer. The Programmer menu and MPLAB IDE toolbar will change to display programmer options once the tool is selected. Also, the Output window will open and messages concerning ICD status and communications will be displayed on the MPLAB ICD 2 tab.

The project should rebuild now so that all debugging modes will be disabled and the configuration bits as defined in the source code will be programmed into the target.

Programming options (e.g., memory areas and ranges) can be set up on the Program tab of the *Programmer>Settings* dialog.

Configuration bits will be set as defined in the source files. They can be manually selected by using the *Configuration> Configuration Bits* dialog.

If desired, set the ID bits by selecting *Configure>ID Memory*.

NOTES:

Chapter 3. MPLAB ICD 2 Tutorial

3.1 INTRODUCTION

This tutorial walks through the process of developing a simple project using the sample program, `TUT452.asm`, found in a folder in the MPLAB IDE installation directory. This program is an implementation of the PIC18F452 analog-to-digital (A/D) converter using the PICDEM 2 Plus Demo Board (DM163022). The program configures the A/D module to convert input from A/D channel 0 (connected to the potentiometer on the demo board) and display the results on the four PORTB LEDs (RB3:RB0).

3.2 HIGHLIGHTS

Topics covered in this chapter:

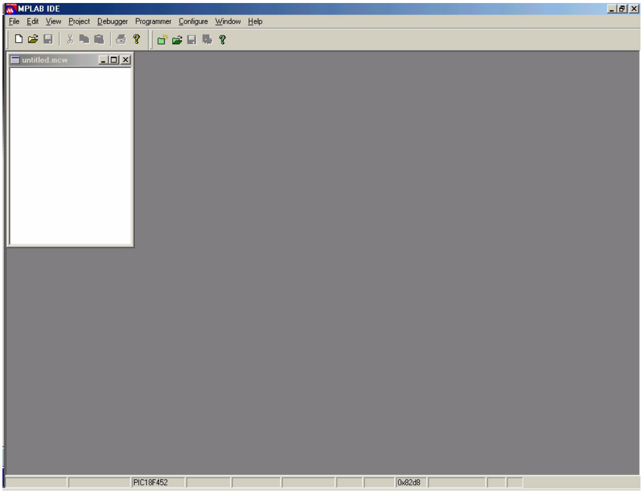
- Setting Up the Environment
- Running the Project Wizard
- Viewing the Project
- Creating a HEX File
- Setting Debug Options
- Setting Up the Demo Board
- Loading Program Code For Debugging
- Running TUT452
- Debugging TUT452
- Programming the Application
- TUT452 Main Routine and Source Code

3.3 SETTING UP THE ENVIRONMENT

Before beginning this tutorial, follow the steps in Chapter 2 "Getting Started with MPLAB IDE", Section 2.3 through 2.5, to set up the hardware and MPLAB IDE software. Some of the initial settings covered in this tutorial may already be in place as a result of the set up in Chapter 2.

Once launched, the MPLAB IDE desktop should appear (Figure 3-1).

FIGURE 3-1: MPLAB IDE V6.XX


MPLAB IDE
File Edit View Project Debugger Programmer Configure Window Help
untitled.mcw
PIC18F452
0x82d8

3.3.1 Selecting the Device and Development Mode

To select the device for this tutorial:

1. Select *Configure>Select Device*.
2. In the Device Selection dialog, choose PIC18F452 from the Device list box. The “light” icon next to MPLAB ICD 2 in the Microchip Tool Support section should be green.
3. Click **OK**.

To select MPLAB ICD 2 as a debugger:

1. Select *Debugger>Select Tool>MPLAB ICD 2*. The Debugger menu will then show the other debug options available. Also, the Output window will open to display connection information.

> **Note:** If MPLAB IDE attempts to connect on start up to the MPLAB ICD 2 but fails because a USB port or a COM port other than COM1 is being used, allow the connection to fail and proceed to Step 2 for communications port setup.

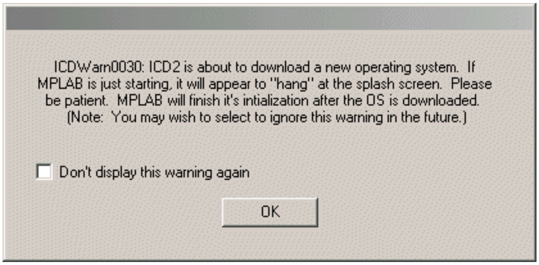
2. Select *Debugger>Settings*, Communications tab. Select USB or the COM port/baud rate to be used with the PC.
3. If it is desired that the MPLAB ICD 2 connects automatically, click on the Status tab and select “Automatically connect at startup”. Check that the Message Level is set to Minimal.
4. Click **OK**.
5. Select *Debugger>Connect* to connect to the MPLAB ICD 2.

3.3.2 Updating MPLAB ICD 2 Firmware (Operating System)

Depending on the version of MPLAB IDE or the device selected, a message may appear indicating that the firmware needs to be updated. MPLAB IDE will automatically install new firmware.

The update dialog may look like this:

FIGURE 3-2: UPDATING MPLAB ICD 2 FIRMWARE DIALOG



ICDWarn0030: ICD2 is about to download a new operating system. If MPLAB is just starting, it will appear to "hang" at the splash screen. Please be patient. MPLAB will finish it's intialization after the OS is downloaded. (Note: You may wish to select to ignore this warning in the future.)

Also, since different MPLAB ICD 2 firmware is used for different families of PICmicro MCUs, this dialog may appear when switching to a different PICmicro MCU.

The firmware can be manually changed by following these steps:

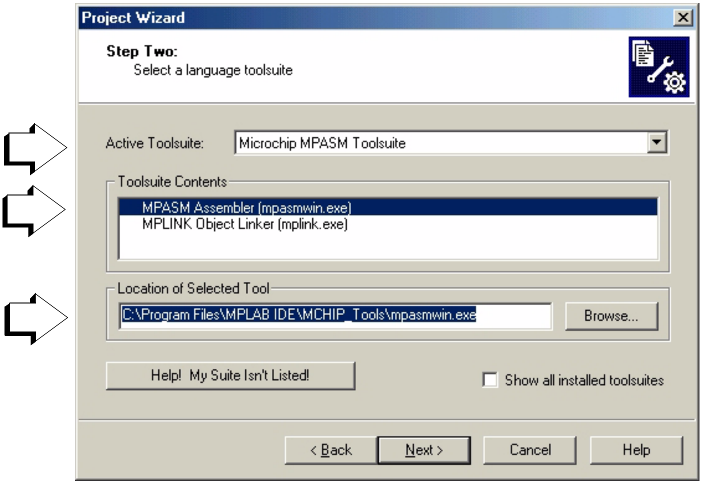
1. Select *Debugger>Download ICD2 Operating System*. The Select ICD 2 Firmware File dialog will open.
2. Select the firmware file to download from the list or browse for it. The file will be of the form `icdxxxxxx.hex`, where `xxxxxx` is the version number. See the README file or `MPLAB ICD 2.txt` for information on the latest versions supplied with MPLAB IDE. Versions may vary according to the PICmicro device selected.
3. Click **Open**. MPLAB IDE will download the new operating system to the MPLAB ICD 2.

3.4 RUNNING THE PROJECT WIZARD

The MPASM assembler will be used in this project.

1. To set up the first project, select *Project>Project Wizard*.
2. Proceed to the second dialog of the wizard. The PIC18F452 should be selected.
3. Proceed to the next dialog of the wizard to set up MPASM as the language tool. In the "Active Toolsuite" pull-down, select "Microchip MPASM Toolsuite." Make sure that MPASM and MPLINK are set as the proper executables in the `MPLAB IDE\MCHIP_Tools` folder. MPASM should be pointing to `mpasmwin.exe`, MPLINK should be pointing to `mplink.exe`.

FIGURE 3-3: WIZARD TOOLSUITE DIALOG



Project Wizard

Step Two:
Select a language toolsuite

Active Toolsuite: Microchip MPASM Toolsuite

Toolsuite Contents
MPASM Assembler (mpasmwin.exe)
MPLINK Object Linker (mplink.exe)

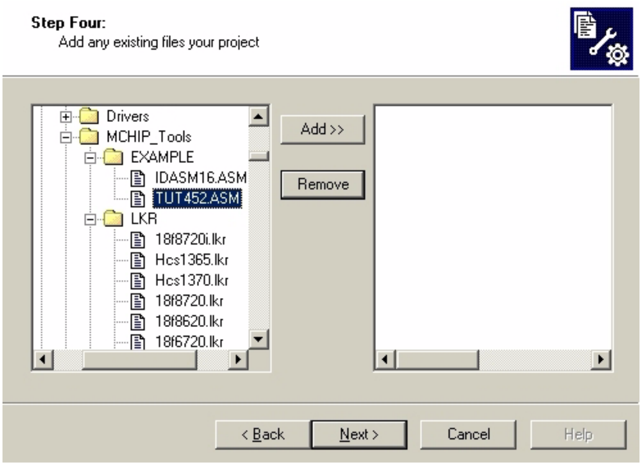
Location of Selected Tool
C:\Program Files\MPLAB IDE\MCHIP_Tools\mpasmwin.exe

□ Show all installed toolsuites

Next project files can be added to the wizard.

Files can also be added later if something is missed. For this example, select **Browse** to go to the MPLAB IDE installation folder (the default folder is `C:\Program Files\MPLAB IDE` on the hard disk), then go to the `MCHIP_Tools\Example` folder to select `TUT452.ASM`.

FIGURE 3-4: WIZARD ADD FILES

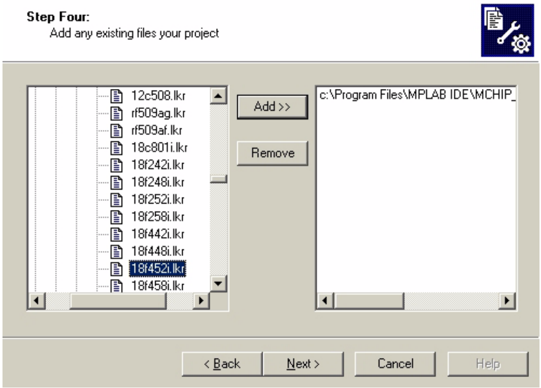


Step Four:
Add any existing files your project

Click on `TUT452.ASM` to highlight it, then Click on **ADD>>** to add it to the right pane.

The second file needed for this project is the linker script. Click on the `\LKR` folder to expand it, and then scroll down to select the file `18F452i.lkr`. Make sure that the file name has an “`i`.”

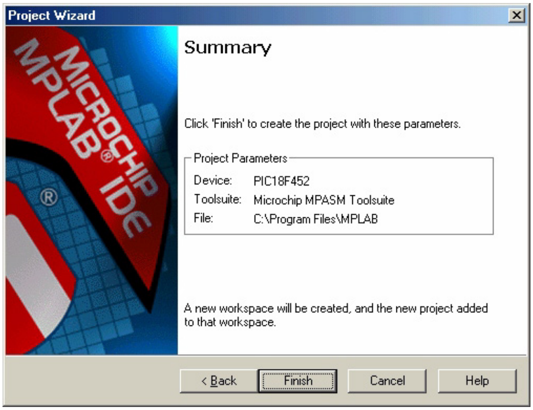
FIGURE 3-5: WIZARD ADD FILES – LINKER SCRIPT



Step Four:
Add any existing files your project

Press **ADD>>** to move the linker script to the list on the right. When done, click **Next**. When finished with the wizard, a Summary screen similar to Figure 3-6 will be seen.

FIGURE 3-6: WIZARD EXIT SCREEN



Project Wizard

Summary

Click 'Finish' to create the project with these parameters.

Project Parameters

Device: PIC18F452
Toolsuite: Microchip MPASM Toolsuite
File: C:\Program Files\MPLAB

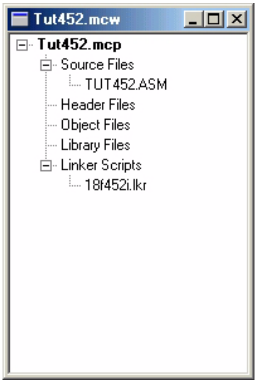
A new workspace will be created, and the new project added to that workspace.

3.5 VIEWING THE PROJECT

After exiting the wizard, the MPLAB IDE desktop will again be visible. Close all other windows on the desktop to see the Project window.

The project window should now look like Figure 3-7.

FIGURE 3-7: PROJECT WINDOW



Tut452.mcw

```
Tut452.mcp
  Source Files
    TUT452.ASM
  Header Files
  Object Files
  Library Files
  Linker Scripts
    18f452i.lkr
```

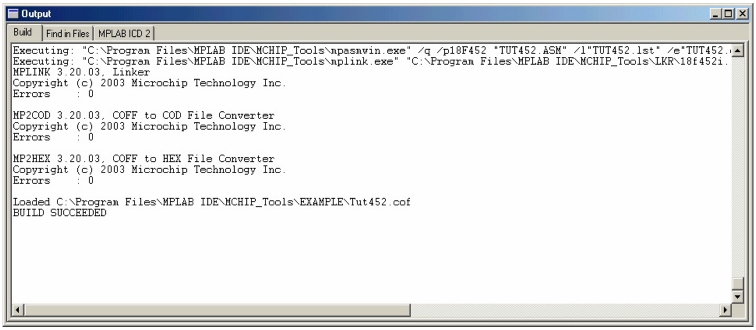
In case of error, files can be manually added to the project window. Put the cursor on the "Source Files" line or the "Linker Scripts" line to add files. Delete a file by highlighting it and clicking the right mouse button to bring up a menu to delete the file.

3.6 CREATING A HEX FILE

To create a HEX file for debugging, select *Project>Build All* or right-click on the project name in the project window and select "Build All" from the popup menu. MPASM assembler always makes a `.hex` file with the same name as the source `.asm` file.

The Output Window should look like Figure 3-8.

FIGURE 3-8: OUTPUT WINDOW



```
Executing: "C:\Program Files\MPLAB IDE\MCHIP_Tools\mpasmwin.exe" /q /p18F452 "TUT452.ASM" /l"TUT452.lst" /e"TUT452.
Executing: "C:\Program Files\MPLAB IDE\MCHIP_Tools\mplink.exe" "C:\Program Files\MPLAB IDE\MCHIP_Tools\LKR\18f452i.
MPLINK 3.20.03, Linker
Copyright (c) 2003 Microchip Technology Inc.
Errors    : 0

MP2COD 3.20.03, COFF to COD File Converter
Copyright (c) 2003 Microchip Technology Inc.
Errors    : 0

MP2HEX 3.20.03, COFF to HEX File Converter
Copyright (c) 2003 Microchip Technology Inc.
Errors    : 0

Loaded C:\Program Files\MPLAB IDE\MCHIP_Tools\EXAMPLE\Tut452.cof
BUILD SUCCEEDED
```

3.7 SETTING DEBUG OPTIONS

3.7.1 Configuration Bits

To set configuration bits to be programmed into the device, select *Configure>Configuration Bits*. By clicking on the text in the "Settings" column, these can be changed. In this dialog, the following configuration bits should be set for this tutorial:

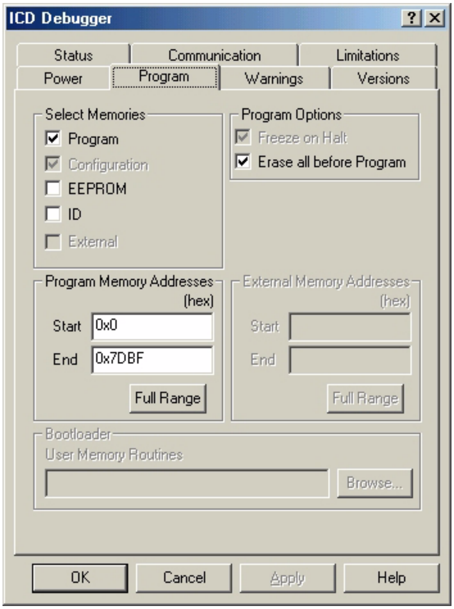
- Oscillator - EC-OS2 as RA6 (this is for PICDEM 2 Plus, if using another target, change to match)
- OSC Switch Enable - Disabled
- Power Up Timer - Enabled
- Brown Out Detect - Disabled
- Watchdog Timer - Disabled
- CCP2 Mux - RC1

- Low Voltage Programming - Disabled
- Background Debug - Enabled (This enables the in-circuit debug features. Normally this will be automatically enabled when MPLAB ICD 2 is selected as the current debugger)
- Stack Overflow Reset - Disabled
- All other configuration bits should be disabled.

3.7.2 Programming Options

To set program options, select *Debugger>Settings* and click on the **Program** tab.

FIGURE 3-9: MPLAB ICD 2 SETTINGS DIALOG, PROGRAM TAB



Select Memories

☑ Program
☑ Configuration
□ EEPROM
□ ID
□ External

Program Options

☑ Freeze on Halt
☑ Erase all before Program

Program Memory Addresses (hex)

Start 0x0
End 0x7DBF
Full Range

External Memory Addresses (hex)

Start
End
Full Range

Bootloader

User Memory Routines

Browse...

- For the PIC18FXX2 devices, all memory will be erased each time the chip is programmed. Therefore, “Erase all before Program” will have no effect.
- The Select Memories section should have “Program” checked, and “EEPROM” and “ID” unchecked. When using MPLAB ICD 2 as a debugger, configuration bits will always be programmed and the Configuration box will be checked and grayed out.
- The Program Memory Addresses (Start Address and End Address) sets the range of program memory that will be read, programmed or verified. Click **Full Range** to set the address range to the maximum program memory available based on the device selected. Note that the end of program memory is adjusted to leave room for the MPLAB ICD 2 debug executive. Programs cannot extend beyond this upper limit while using MPLAB ICD 2 as a debugger.

3.8 SETTING UP THE DEMO BOARD

Before beginning to debug, make sure the PICDEM 2 Plus Demo Board is set up as follows:

- EC OSC option has been selected using jumper J7
- LEDs have been enabled using jumper J6
- Power is supplied

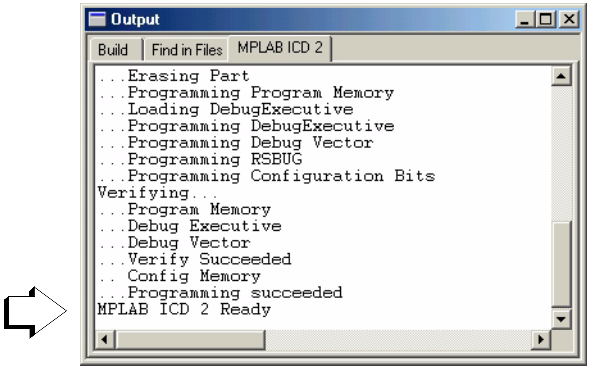
3.9 LOADING PROGRAM CODE FOR DEBUGGING

Select *Debugger>Program* to program `TUT452.hex` into the PIC18F452 in the PICDEM 2 Plus demo board.

Note: The debug executive code is automatically programmed in upper program memory for MPLAB ICD 2 debug functions. Debug code must be programmed into the target PICmicro MCU to use the in-circuit debugging capabilities of the MPLAB ICD 2.

Programming may take a couple of minutes. During programming, the MPLAB ICD 2 tab of the Output dialog shows the current phase of operation. When programming is complete, the dialog should look similar to Figure 3-10.

FIGURE 3-10: CURRENT PHASE OF OPERATION DISPLAY



```
...Erasing Part
...Programming Program Memory
...Loading DebugExecutive
...Programming DebugExecutive
...Programming Debug Vector
...Programming RSBUG
...Programming Configuration Bits
Verifying...
...Program Memory
...Debug Executive
...Debug Vector
...Verify Succeeded
.. Config Memory
...Programming succeeded
MPLAB ICD 2 Ready
```

3.10 RUNNING TUT452

MPLAB ICD 2 executes in real time or in step mode.

- Real time execution occurs when the PIC18F452 in the PICDEM 2 Plus demo board is put in the MPLAB IDE's Run mode.
- Step mode execution can be accessed after the processor is halted.

These toolbar buttons can be used for quick access to commonly used debug operations.

Debugger Menu	Toolbar Buttons
Run	▶
Halt	⏸
Step Into	{↵}
Reset	↻

Begin in real time mode:

1. Open the `TUT452.asm` file (double click on the file name in the Project Window or use *File>Open*).
2. Select *Debugger>Run* (or click the Run toolbar button).
3. On the demo board, turn the arrow on the potentiometer (RA0). Observe the LEDs. If the program were working correctly, the user would see a binary representation of the voltage value across the potentiometer. However, there is a bug in the TUT452 program. Section 3.11 will detail debugging the code and correcting it.
4. Select *Debugger>Halt* (or click the Halt toolbar button) to stop the program execution.
5. Select *Debugger>Reset* to reset the program.

3.11 DEBUGGING TUT452

Any of the following can prevent the TUT452 program from working.

- The A/D converter value is not being properly written to PORTB (LEDs).
- The A/D converter is not on or has not been set to convert.
- A typing error in the source code is causing the program to function improperly.

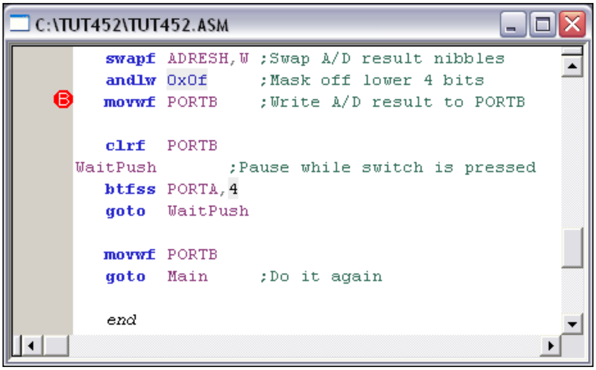
To explore the first possibility, set a breakpoint at the line of the file that writes the value of A/D result to PORTB.

1. Highlight or place the cursor on the following line of code from `TUT452.asm`:

   `movwfPORTB;Write A/D result to PORTB`

2. Right-click to display a shortcut menu.
3. Select *Set Break Point* from the shortcut menu. This line is now marked as a breakpoint (B in red stop sign) as shown in Figure 3-11.

FIGURE 3-11: SET BREAKPOINT



```
        swapf  ADRESH,W  ;Swap A/D result nibbles
        andlw  0x0f      ;Mask off lower 4 bits
        movwf  PORTB     ;Write A/D result to PORTB

        clrf   PORTB
WaitPush            ;Pause while switch is pressed
        btfss  PORTA,4
        goto   WaitPush

        movwf  PORTB
        goto   Main      ;Do it again

        end
```

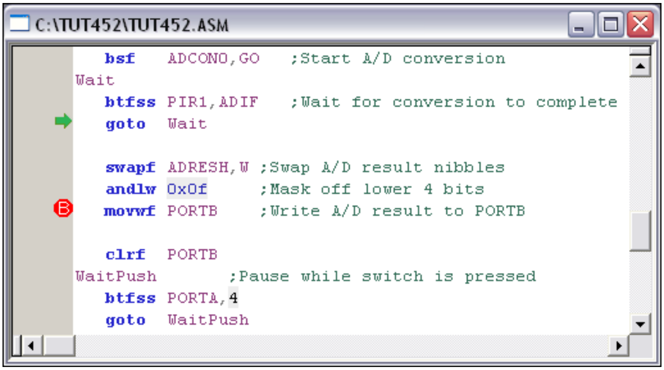
4. Select *Debugger>Run* (or click the Run toolbar button) to run the program once again in real time mode.

   A breakpoint stops a program's execution when the program executes the line marked as a breakpoint. However, the sample program is not halting.

5. Select *Debugger>Halt* (or click the Halt button on the toolbar) to halt the program now.

   In the source code window, the sample program will have halted on one of the two lines in the Wait routine as shown in Figure 3-12 (at the arrow).

FIGURE 3-12: PROGRAM HALTED



C:\TUT452\TUT452.ASM

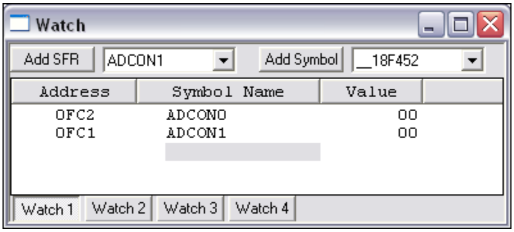
```
    bsf    ADCON0,GO    ;Start A/D conversion
Wait
    btfss  PIR1,ADIF    ;Wait for conversion to complete
    goto   Wait

    swapf  ADRESH,W  ;Swap A/D result nibbles
    andlw  0x0f      ;Mask off lower 4 bits
    movwf  PORTB     ;Write A/D result to PORTB

    clrf   PORTB
WaitPush           ;Pause while switch is pressed
    btfss  PORTA,4
    goto   WaitPush
```

Based on the halt location and the fact that the program never reaches the breakpoint, it can be concluded that the problem is in the A/D conversion. The A/D flag for conversion complete is not being set. A/D conversion initialization and setup occurs at the beginning of the program.

6. To reset the program, select *Debugger>Reset*. The first instruction should be indicated with a green arrow.

7. Open a new Watch window to watch the A/D register value change as the program executes. Select *View>Watch*. The Watch dialog opens with the Watch_1 tab selected. Select "ADCON0" from the list next to Add SFR, and then click the button. ADCON0 is added to the Watch window. Repeat for ADCON1. The selected symbols should now be visible in the Watch window as shown in Figure 3-13.

FIGURE 3-13: WATCH WINDOW



Address	Symbol Name	Value
0FC2	ADCON0	00
0FC1	ADCON1	00

8. In the `TUT452.asm` source code, set a breakpoint at the first instruction after Start. Highlight or place the cursor in the following line of code from `TUT452.asm`:

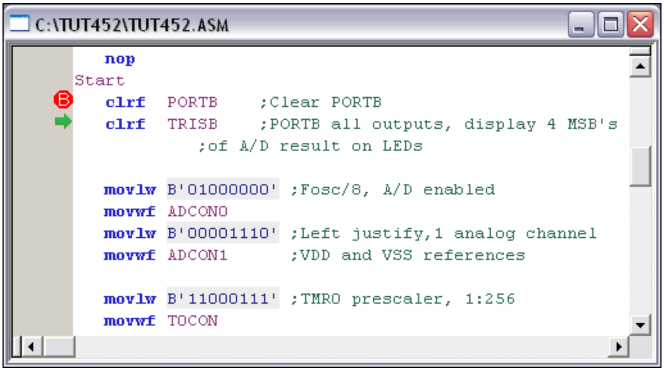
```
clrfPORTB;Clear PORTB
```

Right-click to display a shortcut menu. Select *Set Break Point* from the shortcut menu. This line will now be marked as a breakpoint.

9. Select *Debugger>Run* (or click the Run toolbar button) to run the program in real time mode.

This time the program will stop after it executes the breakpoint line of code and the instruction after the breakpoint will be indicated as shown in Figure 3-14.

FIGURE 3-14: PROGRAM HALTED AFTER BREAK



```
      nop
Start
      clrf   PORTB     ;Clear PORTB
      clrf   TRISB     ;PORTB all outputs, display 4 MSB's
                 ;of A/D result on LEDs

      movlw  B'01000000' ;Fosc/8, A/D enabled
      movwf  ADCON0
      movlw  B'00001110' ;Left justify,1 analog channel
      movwf  ADCON1      ;VDD and VSS references

      movlw  B'11000111' ;TMR0 prescaler, 1:256
      movwf  T0CON
```

10. Single step five times, select *Debugger>Step* (or click the Step toolbar button) to single step through the program. The following line of code should be indicated when finished:

    ```
    movlwB'11000111'  ;TMR0 prescaler, 1:256
    ```

11. Examine the values of the registers ADCON0 and ADCON1 in the Watch window. Notice that ADCON0 has a value of 40 HEX. This corresponds to the binary value designated in the program, but is this value correct?

    A review of the PIC18F452 Data Sheet (DS39564) section on A/D, indicates that the last bit should be a one, not a zero, to turn the A/D module on.

    To fix this bug, change:

    ```
    movlwB'01000000';Fosc/8, A/D enabled
    ```

    to:

    ```
    movlwB'01000001';Fosc/8, A/D enabled
    ```

12. Select *File>Save* to save the changes.

13. Select Project>Build All to rebuild the project. A message will indicate that the program has been rebuilt. The MPLAB ICD 2 must be reprogrammed for the changes to take effect.
14. Select Debugger>Program to reprogram the MPLAB ICD 2 with the changes. When the MPLAB ICD 2 dialog indicates '...Programming succeeded', the program is ready to run again.
15. Right-click on the line of code that previously had the breakpoint (now indicated by a red stop sign outline). Select Remove Break Point.
16. Select Debugger>Run (or click the Run toolbar button) to run the program in real time mode. Turn the potentiometer (RA0) to change the value displayed on the LEDs.

The source code in this tutorial contained only one bug. However, real code may have more. Using the MPLAB ICD 2 and MPLAB IDE debugging functions, users can successfully find and fix the problems in their code.

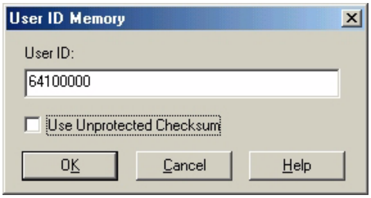
3.12 PROGRAMMING THE APPLICATION

When the program is successfully debugged and running, usually the next step is to program the PICmicro MCU for stand alone operation in the finished design. When doing this, the resources reserved by the ICD are released for use by the application.

To program the application follow these steps:

1. Disable MPLAB ICD 2 as a debug tool by selecting Debugger>Select Tool>none.
2. Select MPLAB ICD 2 as the programmer in Programmer>Select Tool menu.
3. *Optional:* Set up the ID in Configure>ID Memory (Figure 3-15).

FIGURE 3-15: USER ID MEMORY DIALOG



User ID Memory

User ID: 64100000

□ Use Unprotected Checksum

4. Set up the parameters for programming on the *Programmer>Settings* Program tab.
5. Select *Programmer>Program*.

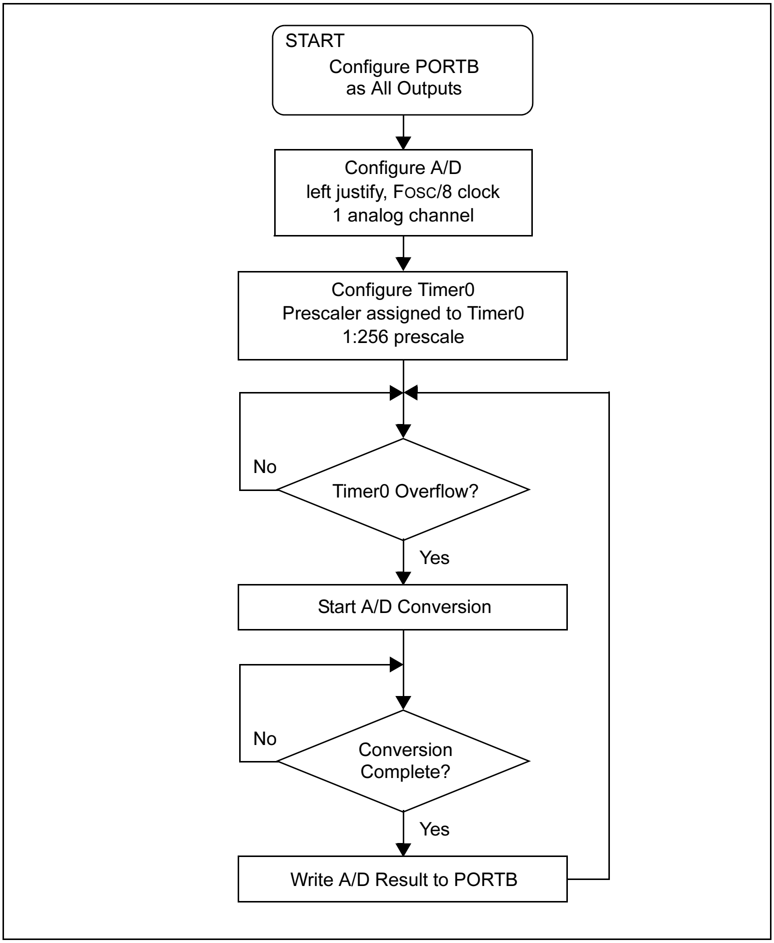
Now MPLAB ICD 2 can reset and run the target (MPLAB ICD 2 can also be disconnected from the target, after selecting the reset button, the application will run).

3.13 TUT452 MAIN ROUTINE AND SOURCE CODE

The main routine of `TUT452.asm` (Figure 3-15) begins by configuring PORTB, the A/D module and Timer0. It then waits for a Timer0 overflow to start the A/D conversion of the value from the potentiometer. When the conversion is complete, the value is displayed on the LEDs, and the program loops back to wait for another Timer0 overflow to start another A/D conversion.

For more information on A/D module operation and a list of related application notes, refer to the *PICmicro 18C MCU Family Reference Manual* (DS39500).

FIGURE 3-16: TUT452 MAIN ROUTINE FLOW


START
Configure PORTB as All Outputs
Configure A/D left justify, FOSC/8 clock 1 analog channel
Configure Timer0 Prescaler assigned to Timer0 1:256 prescale
No
Timer0 Overflow?
Yes
Start A/D Conversion
No
Conversion Complete?
Yes
Write A/D Result to PORTB

```
;****************************************
;*              TUT452.ASM             *
;****************************************

    list p=18f452

    include "p18f452.inc"

Reset_Vector code 0x0   ; Put a GOTO at reset address
         goto  Start

         code  0x0002A      ; Start app beyond vector area
Start
    clrf   PORTB        ;Clear PORTB
    clrf   TRISB        ;PORTB all outputs,
                        ;display 4 MSB's of A/D result on LEDs
    movlw  B'01000000'  ;Fosc/8,A/D enabled
    movwf  ADCON0
    movlw  B'00001110'  ;Left justify,1 analog channel
    movwf  ADCON1       ;VDD and VSS ref's
    movlw  B'11000111'  ;TMR0 prescaler,1:256
    movwf  T0CON

Main
    btfss  INTCON,TMR0IF ;Wait for Timer0 to timeout
    goto   Main
    bcf    INTCON,TMR0IF

    bsf    ADCON0,GO     ;Start A/D
                         ;conversion
Wait
    btfss  PIR1,ADIF     ;Wait for conversion to complete
    goto   Wait

    swapf  ADRESH,W      ;Swap A/D result nibbles
    andlw  0x0f          ;Mask off lower 4 bits
    movwf  PORTB         ;Write A/D result to PORTB

    clrf   PORTB

WaitPush                 ;Pause while switch is pressed
    btfss  PORTA,4
    goto   WaitPush

    movwf  PORTB
    goto   Main        ;Do it again

    end
```

NOTES:

Chapter 4. USB Installation Details

4.1 INTRODUCTION

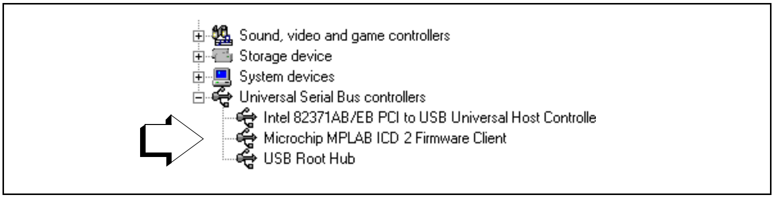
This chapter covers the installation details for the USB driver for the various Windows operating systems. These systems are covered:

- Windows 98/ME
- Windows 2000
- Windows XP

USB problems will be experienced if an incorrect driver is installed. Windows will often try to install a default USB driver. This driver will not function with MPLAB ICD 2. If problems are encountered, look at the hardware device manager to verify that the Microchip USB driver is listed under USB devices. Sometimes an unknown USB device will be listed under “other devices.” This device should be removed if found, then continue following these steps for the appropriate operating system of the PC.

Figure 4-1 shows the correct configuration:

FIGURE 4-1: DEVICE MANAGER



Sound, video and game controllers
Storage device
System devices
Universal Serial Bus controllers
Intel 82371AB/EB PCI to USB Universal Host Controlle
Microchip MPLAB ICD 2 Firmware Client
USB Root Hub

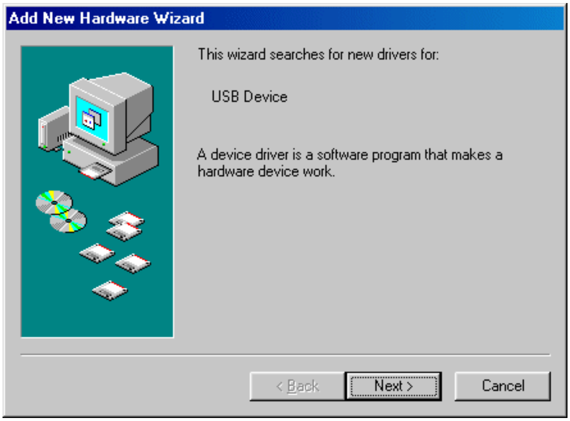
4.2 WINDOWS 98 AND WINDOWS ME INSTALLATION

4.2.1 Step 1

The first time a new USB device is plugged into a Windows 98/ME system, a dialog box will appear indicating that the system has found a new hardware device. There may also be a dialog box that informs the user that a device data base is being built or updated.

After these dialogs appear, the "Add New Hardware Wizard" dialog box is displayed. Click **Next** to continue the installation.

FIGURE 4-2: WINDOWS 98/ME NEW HARDWARE WIZARD - 1



This wizard searches for new drivers for:

USB Device

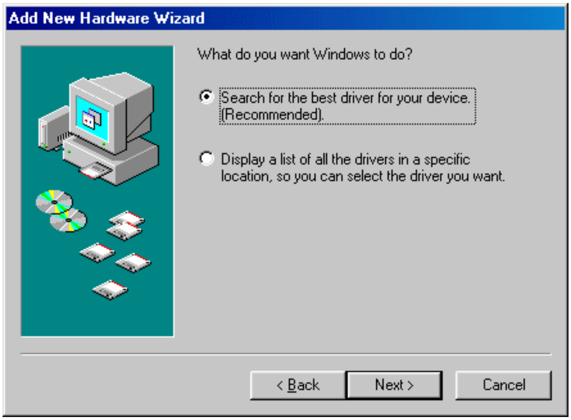
A device driver is a software program that makes a hardware device work.

4.2.2 Step 2

Make sure that "Search for the best driver..." is selected and click **Next**.

> **Note:** In the event that subsequent steps fail to find or recognize the INF or device driver files, repeat Step 2, selecting "Display a list of all the drivers...". The next screen should ask for a device type. At that point, click on the **Have Disk** button. The next dialog box will allow selection of the directory path in which the `.INF` file is to be found. Once this directory is selected, click **OK** and continue with the install.

FIGURE 4-3: WINDOWS 98/ME NEW HARDWARE WIZARD - 2



Add New Hardware Wizard

What do you want Windows to do?

● Search for the best driver for your device. (Recommended).

○ Display a list of all the drivers in a specific location, so you can select the driver you want.

4.2.3 Step 3

This next dialog box allows the user to specify where needed files for the new device are to be found. Make sure all the check boxes are *unchecked* except for the one that says "Specify a location". Select "Specify a location" and then enter the path to the device driver files (either type the path in or use the **Browse** button to search). Under a normal installation, this path should be:

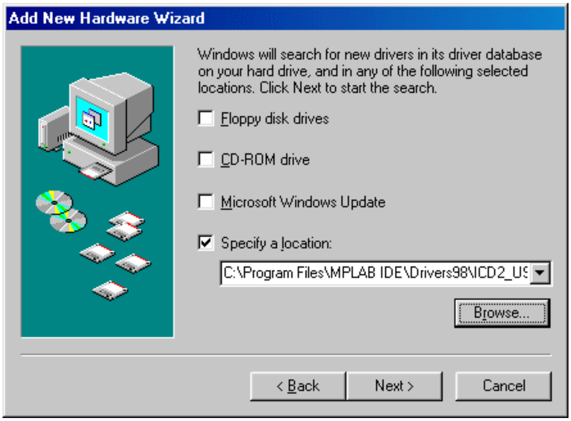
```
C:\Program Files\MPLAB IDE\Drivers98\ICD2_USB
```

This path must contain the following files:

- `ICD2W98.INF`
- `ICD2W98.SYS`
- `ICD2W98L.SYS`

Once the path is specified, click **Next**.

FIGURE 4-4: WINDOWS 98/ME NEW HARDWARE WIZARD - 3



Add New Hardware Wizard

Windows will search for new drivers in its driver database on your hard drive, and in any of the following selected locations. Click Next to start the search.

□ Floppy disk drives

□ CD-ROM drive

□ Microsoft Windows Update

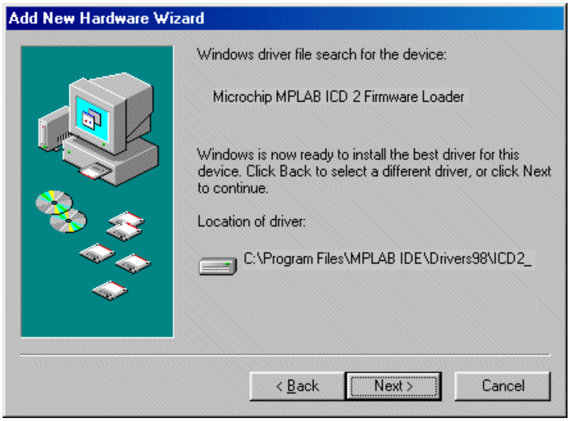
☑ Specify a location:

C:\Program Files\MPLAB IDE\Drivers98\ICD2_US

4.2.4 Step 4

This next dialog box should indicate that the “Microchip MPLAB ICD 2 Firmware Loader” has been found and show the path that was previously selected. Click **Next**.

FIGURE 4-5: WINDOWS 98/ME NEW HARDWARE WIZARD - 4



Windows driver file search for the device:

Microchip MPLAB ICD 2 Firmware Loader

Windows is now ready to install the best driver for this device. Click Back to select a different driver, or click Next to continue.

Location of driver:

C:\Program Files\MPLAB IDE\Drivers98\ICD2_

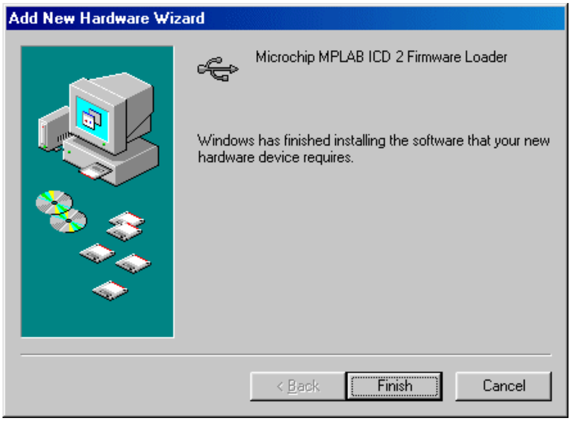
4.2.5 Step 5

The last dialog box will indicate that the “Microchip MPLAB ICD 2 Firmware Loader” has been installed. To complete the process, click **Finish**.

At this point the MPLAB ICD 2 will “re-enumerate”. Another set of dialog boxes indicating this condition may be seen. The system should automatically load the MPLAB ICD 2 USB client driver needed to communicate with the device.

In the event that the new dialog boxes ask for another path, specify the same path that was given in Step 3.

FIGURE 4-6: WINDOWS 98/ME NEW HARDWARE WIZARD - 5



Add New Hardware Wizard

Microchip MPLAB ICD 2 Firmware Loader

Windows has finished installing the software that your new hardware device requires.

4.3 WINDOWS 2000 INSTALLATION

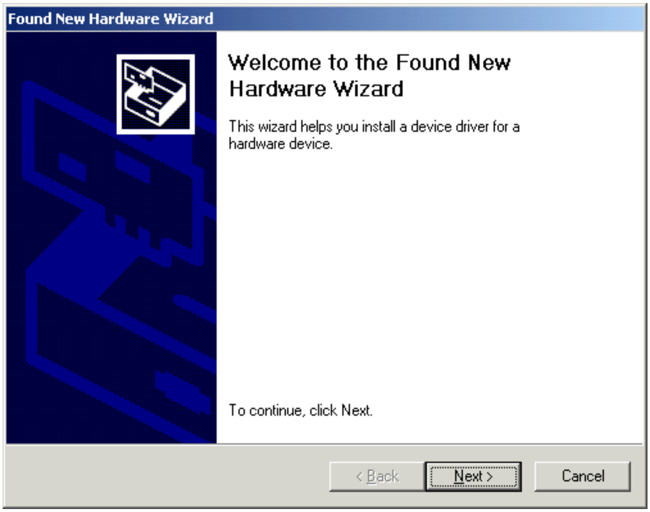
When the MPLAB ICD 2 is plugged into the USB for the first time on a Windows 2000 system, the following steps will have to be taken in order to properly install the device drivers.

4.3.1 Step 1

The first time a new USB device is plugged into a Windows 2000 system, a dialog box will appear indicating that the system has found a new hardware device. There may also be a dialog box that informs the user that a device data base is being built or updated.

After these dialogs appear, the “Found New Hardware Wizard” dialog box is displayed. Click **Next** to continue the installation.

FIGURE 4-7: WINDOWS 2000 NEW HARDWARE WIZARD - 1



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

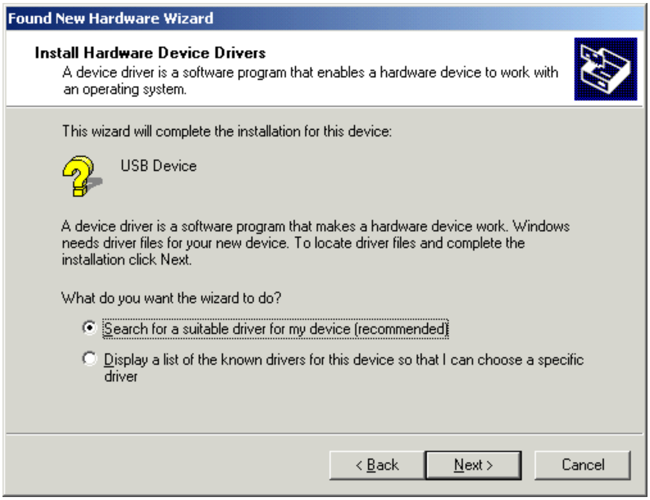
This wizard helps you install a device driver for a hardware device.

To continue, click Next.

4.3.2 Step 2

Make sure that "Search for a suitable driver..." is selected and click **Next**.

FIGURE 4-8: WINDOWS 2000 NEW HARDWARE WIZARD - 2



Found New Hardware Wizard

Install Hardware Device Drivers

A device driver is a software program that enables a hardware device to work with an operating system.

This wizard will complete the installation for this device:

USB Device

A device driver is a software program that makes a hardware device work. Windows needs driver files for your new device. To locate driver files and complete the installation click Next.

What do you want the wizard to do?

● Search for a suitable driver for my device (recommended)

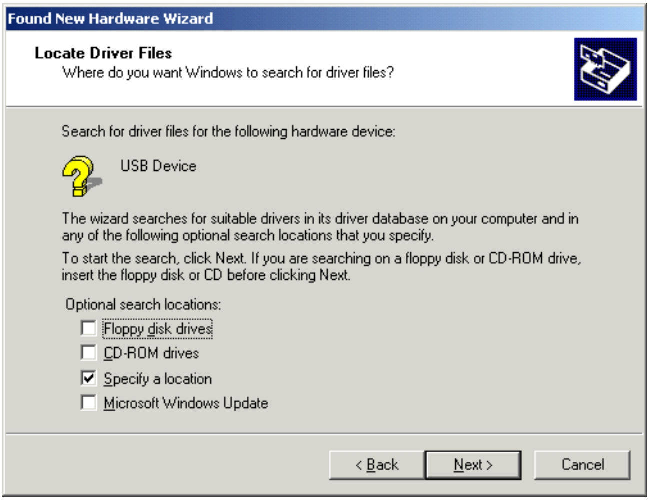
○ Display a list of the known drivers for this device so that I can choose a specific driver

Note: In the event that subsequent steps fail to find or recognize the INF or device driver files, repeat Step 2, selecting ”Display a list of the known drivers...”. The next screen should ask for a device type. At that point, click on the **Have Disk** button. The next dialog box will allow selection of the directory path in which the INF file is to be found. Once this directory is selected, click **OK** and continue with the install.

4.3.3 Step 3

This next dialog box allows the user to specify where needed files for the new device are to be found. Make sure all the check boxes are *unchecked* except for the one that says “Specify a location”. Select “Specify a location” and then click **Next**.

FIGURE 4-9: WINDOWS 2000 NEW HARDWARE WIZARD - 3



Found New Hardware Wizard

Locate Driver Files

Where do you want Windows to search for driver files?

Search for driver files for the following hardware device:

USB Device

The wizard searches for suitable drivers in its driver database on your computer and in any of the following optional search locations that you specify.

To start the search, click Next. If you are searching on a floppy disk or CD-ROM drive, insert the floppy disk or CD before clicking Next.

Optional search locations:

- □ Floppy disk drives
- □ CD-ROM drives
- ☑ Specify a location
- □ Microsoft Windows Update

4.3.4 Step 4

Enter the path to the device driver files in the edit field labeled "Copy manufacturer's files from:" (either type the path in or use the "Browse" button to search and fill it in). Under a normal installation, this path should be:

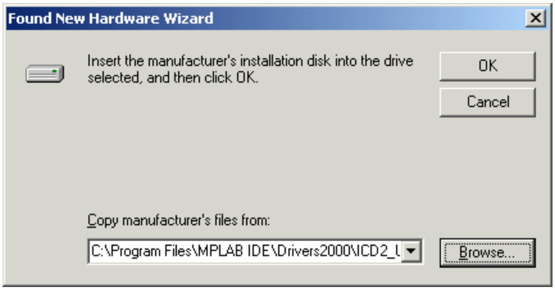
```
C:\Program Files\MPLAB IDE\Drivers2000\ICD2_USB
```

This path must contain the following files:

- `ICD2W2K.INF`
- `ICD2W2K.SYS`
- `ICD2W2KL.SYS`

Once the path is specified, click **OK**.

FIGURE 4-10: WINDOWS 2000 NEW HARDWARE WIZARD - 4



Found New Hardware Wizard

Insert the manufacturer's installation disk into the drive selected, and then click OK.

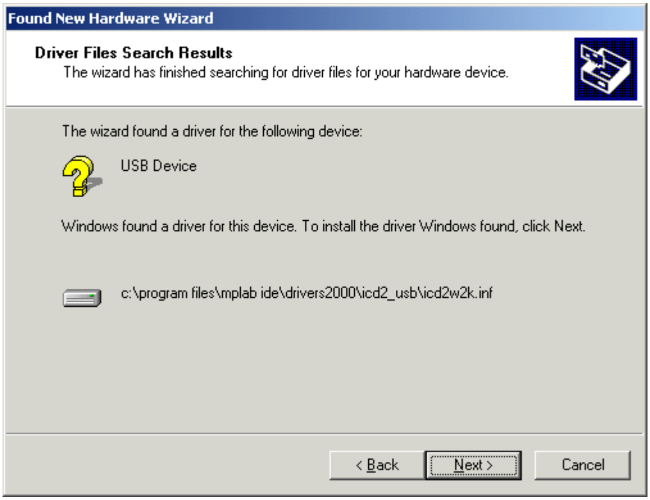
Copy manufacturer's files from:

C:\Program Files\MPLAB IDE\Drivers2000\ICD2_U

4.3.5 Step 5

This next dialog box should indicate that the Wizard found a device driver in the location specified in the previous step. Click **Next** to continue.

FIGURE 4-11: WINDOWS 2000 NEW HARDWARE WIZARD - 5



Found New Hardware Wizard

Driver Files Search Results

The wizard has finished searching for driver files for your hardware device.

The wizard found a driver for the following device:

USB Device

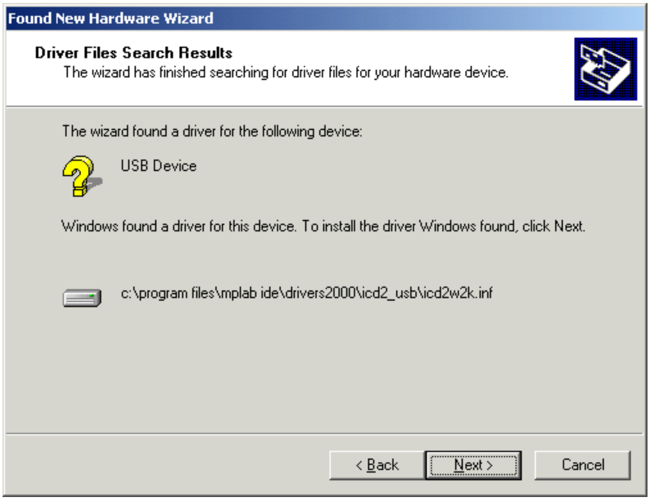
Windows found a driver for this device. To install the driver Windows found, click Next.

c:\program files\mplab ide\drivers2000\icd2_usb\icd2w2k.inf

4.3.6 Step 6

This next dialog box should indicate that the “Microchip MPLAB ICD 2 Firmware Loader” has been found and show the path that was previously selected. Click **Next**.

FIGURE 4-12: WINDOWS 2000 NEW HARDWARE WIZARD - 6



Found New Hardware Wizard

Driver Files Search Results

The wizard has finished searching for driver files for your hardware device.

The wizard found a driver for the following device:

USB Device

Windows found a driver for this device. To install the driver Windows found, click Next.

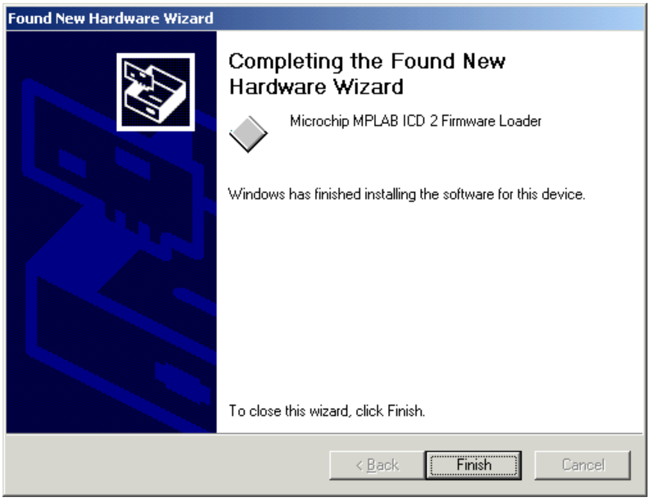
c:\program files\mplab ide\drivers2000\icd2_usb\icd2w2k.inf

The last dialog box will indicate that the “Microchip MPLAB ICD 2 Firmware Loader” has been installed. To complete the process, click **Finish**.

At this point the MPLAB ICD 2 will “re-enumerate”. Another set of dialog boxes may be seen indicating this condition. The system should automatically load the MPLAB ICD 2 USB client driver needed to communicate with the device.

In the event that the new dialog boxes ask for another path, specify the same path that was given in Step 4.

FIGURE 4-13: WINDOWS 2000 NEW HARDWARE WIZARD - 7



Found New Hardware Wizard

Completing the Found New Hardware Wizard

Microchip MPLAB ICD 2 Firmware Loader

Windows has finished installing the software for this device.

To close this wizard, click Finish.

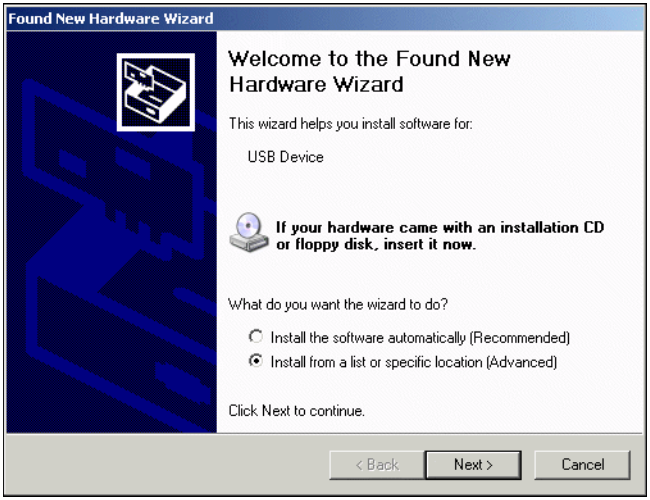
4.4 WINDOWS XP INSTALLATION

4.4.1 Step 1

The first time a new USB device is plugged into a Windows XP system, a dialog box will appear indicating that the system has found a new hardware device. There may also be a dialog box that informs the user that a device data base is being built or updated.

After these dialogs appear, the Found New Hardware Wizard dialog box is displayed. Select "Install from a list or specific location (Advanced)" and click **Next** to continue the installation.

FIGURE 4-14: WINDOWS XP NEW HARDWARE WIZARD - 1



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

This wizard helps you install software for:

USB Device

If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

○ Install the software automatically (Recommended)

● Install from a list or specific location (Advanced)

Click Next to continue.

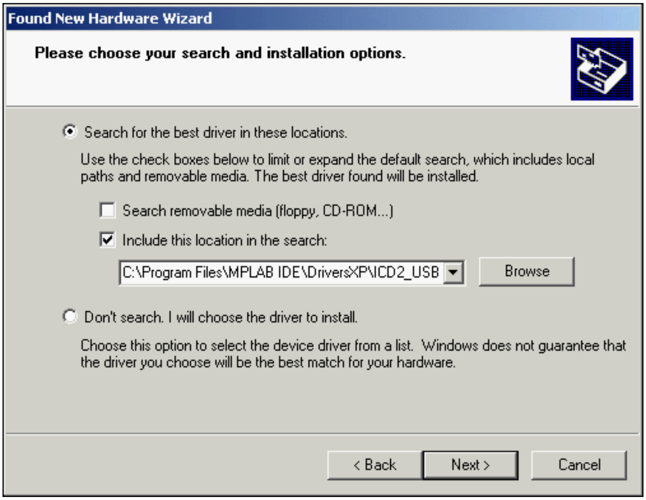
4.4.2 Step 2

Make sure that “Search for the best driver...” is selected. Unselect “Search removable media” and check “Include this location in the search”. Browse to the MPLAB IDE installation directory, and select the subdirectory DriversXP\ICD2_USB. Click **Next**.

During the install, a dialog will pop up stating, “The software you are installing for this hardware...has not passed Windows Logo testing...” Click **Continue Anyway**.

> **Note:** In the event that subsequent steps fail to find or recognize the INF or device driver files, repeat Step 2, selecting “Don't search. I will choose...” and then click **Next**. The next screen should ask for a hardware type. Make sure “Show All Devices” is selected and then click **Next**. In the “Select the device driver...” window, click on the **Have Disk** button. The next dialog box will allow selection of the directory path in which the INF file is to be found. Browse to the MPLAB IDE installation directory, and select the subdirectory DriversXP\ICD2_USB. Once this directory is selected, click **OK** and continue with the install.

FIGURE 4-15: WINDOWS XP NEW HARDWARE WIZARD - 2



Found New Hardware Wizard

Please choose your search and installation options.

● Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

□ Search removable media (floppy, CD-ROM...)

☑ Include this location in the search:

C:\Program Files\MPLAB IDE\DriversXP\ICD2_USB

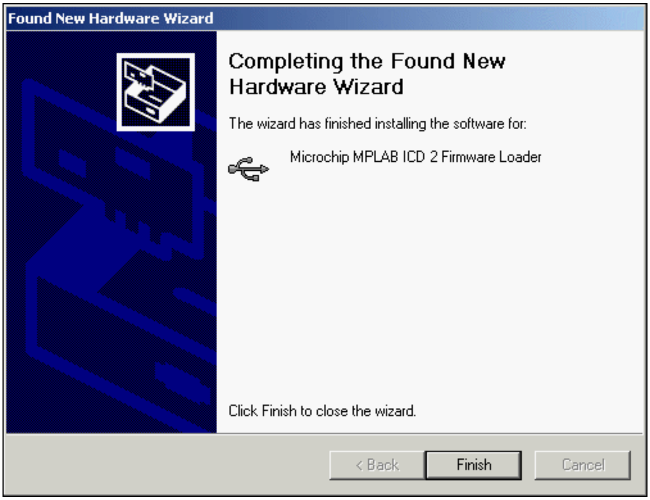
○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

4.4.3 Step 3

The "Completing the Found New Hardware Wizard" will appear once the Microchip MPLAB ICD 2 Firmware Loader has been installed. Click **Finish** to end this installation and begin the install of the Firmware Client.

FIGURE 4-16: WINDOWS XP NEW HARDWARE WIZARD - 3



Found New Hardware Wizard

Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

Microchip MPLAB ICD 2 Firmware Loader

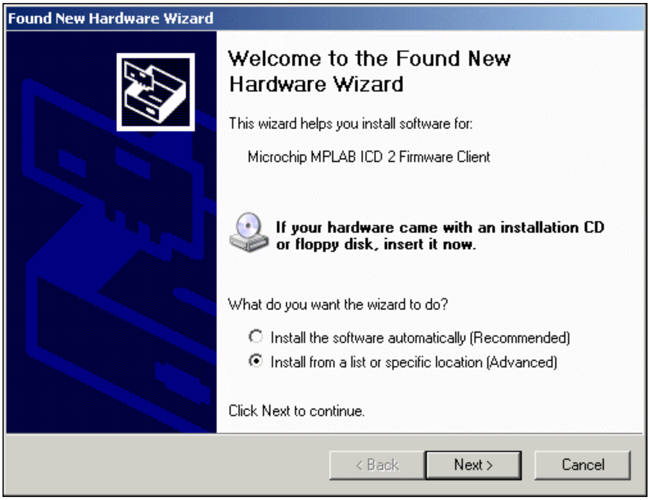
Click Finish to close the wizard.

4.4.4 Step 4

A dialog box will appear indicating that the system completed adding a hardware device, immediately followed by the statement that it has found a new hardware device.

After this dialog appears, the Found New Hardware Wizard dialog box is displayed again, this time to install the Microchip MPLAB ICD 2 Firmware Client. Select "Install from a list or specific location (Advanced)" and click **Next** to continue the installation.

FIGURE 4-17: WINDOWS XP NEW HARDWARE WIZARD - 4



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

This wizard helps you install software for:

Microchip MPLAB ICD 2 Firmware Client

If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

○ Install the software automatically (Recommended)

● Install from a list or specific location (Advanced)

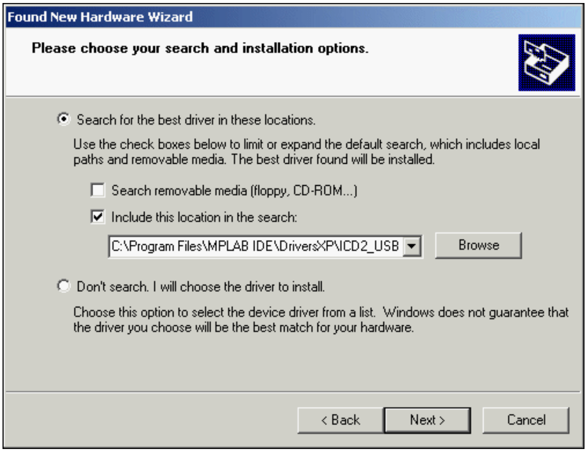
Click Next to continue.

4.4.5 Step 5

Make sure that “Search for the best driver...” is selected. Unselect “Search removable media” and check “Include this location in the search”. Browse to the MPLAB IDE installation directory, and select the subdirectory DriversXP\ICD2_USB. Click **Next**.

During the install, a dialog will pop up stating, “The software you are installing for this hardware...has not passed Windows Logo testing...” Click **Continue Anyway**.

FIGURE 4-18: WINDOWS XP NEW HARDWARE WIZARD - 5



Found New Hardware Wizard

Please choose your search and installation options.

● Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

□ Search removable media (floppy, CD-ROM...)

☑ Include this location in the search:

C:\Program Files\MPLAB IDE\DriversXP\ICD2_USB

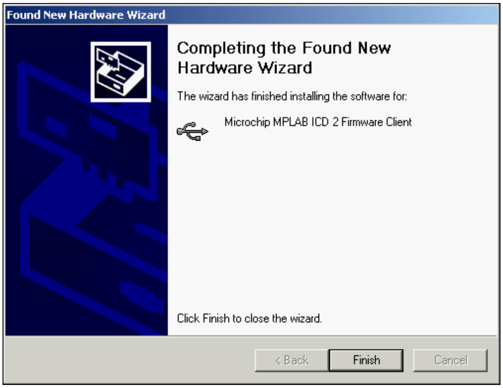
○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

4.4.6 Step 6

The "Completing the Found New Hardware Wizard" will appear once the Microchip MPLAB ICD 2 Firmware Client has been installed. Click **Finish** to end the USB installation.

FIGURE 4-19: WINDOWS XP NEW HARDWARE WIZARD - 6



Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

Microchip MPLAB ICD 2 Firmware Client

Click Finish to close the wizard.

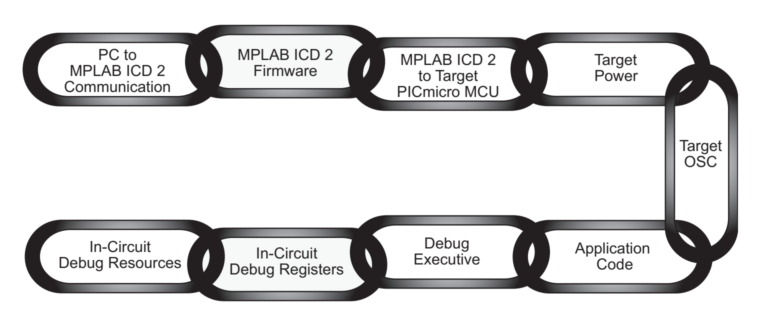
Chapter 5. Troubleshooting

5.1 WHAT TO DO IN CASE OF TROUBLE

Since many things in MPLAB IDE, MPLAB ICD 2 and the PICmicro MCU target application circuit can affect the operation of MPLAB ICD 2, it is important to understand the operation of each link in the chain of ICD 2 functionality.

Figure 5-1 is a diagram showing the various links. Each link must be functional for MPLAB ICD 2 to program and debug. This section discusses these links to help find and fix problems if they occur.

FIGURE 5-1: CHAIN OF MPLAB ICD 2 FUNCTIONALITY



5.1.1 Link: PC to ICD 2 Communications

This link is the foundation for all subsequent links. If it doesn’t work, users will experience communications errors with MPLAB IDE. Make sure that the MPLAB USB drivers are installed according to the instructions. Look at the Windows hardware manager dialog under the USB section to see that the MPLAB ICD 2 driver is properly installed on the PC. If using RS-232, make sure that the FIFO buffers are turned off and flow control is set for hardware in the COMM driver dialog. If the MPLAB ICD 2 USB driver is not visible in the USB section of the Hardware Manager, go to the MPLAB IDE installation directory and open the `\DriversXXX` folder (`XXX` corresponds to the operating system, i.e., 98, NT, XP, etc.). Open the.HTM files with a web browser to view details on driver installation for the appropriate operating system. See Chapter 4, USB Installation Details for more information.

5.1.2 Link: ICD 2 Firmware

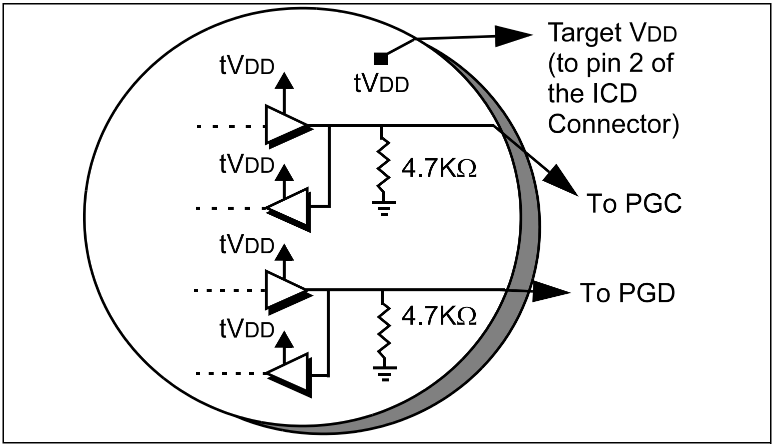
Be sure that the latest version of firmware is loaded in the MPLAB ICD 2. Updated firmware can be downloaded with the MPLAB IDE software. The latest production software should always be downloaded from the Microchip web site.

The version of firmware will determine which target PICmicros are supported. See the README file for the latest information on the firmware and on MPLAB ICD 2.

5.1.3 Link: ICD 2 to Target PICmicro

Verify that all lines are connected and that no other signals or components are interfering with ICD signals on VPP, PGC and PGD. If MPLAB ICD 2's VDD and VSS lines are not connected to the target power and ground, it will not work. MPLAB ICD 2 can provide VDD to the target (<200mA, 5 volts only) and to these output buffers when providing power to the target circuit (selected by the *Debugger>Settings* dialog Power tab, "Power target circuit from MPLAB ICD 2"). Otherwise, the target's VDD will be used. VDD is sensed by MPLAB ICD 2 to check that voltage levels are correct. Powering from the target VDD allows level translation for target low voltage operation. A simplified circuit for the internal buffer circuits of MPLAB ICD 2 is shown in Figure 5-2.

FIGURE 5-2: MPLAB ICD 2 INTERNAL BUFFER CIRCUITS



If MPLAB ICD 2 does not have voltage on its VDD line (pin 2 of the ICD connector) it will not operate. Using a scope, communications on PGC and PGD with a full target VDD to VSS peak-to-peak waveform should be seen. VPP should show +12, +5, and 0V levels depending upon the operation being performed. Signals should be seen on those lines only after performing an MPLAB ICD 2 function from the MPLAB user interface. Otherwise, all these lines should be idle and noise free.

5.1.4 Link: Target Power

MPLAB ICD 2 can work within a range of about 2V to 5.5V on the target PICmicro MCU's VDD. The application must be powered by its own power supply. Remember, that the PGC and PGD I/O drivers in MPLAB ICD 2 are powered from the target VDD (see Figure 5-2).

5.1.5 Link: Target Oscillator

When programming a device, it will program without a running oscillator, but the debug function will not work unless the target oscillator is functioning. Unlike an in-circuit emulator, while debugging, MPLAB ICD 2 requires that the target PICmicro MCU be running.

Testing may be necessary to make sure that the target system is executing instructions properly. For instance, a small program could be written to flash an LED and then programmed into the target PICmicro MCU with MPLAB ICD 2 debug disabled. Disconnect the ICD and reset the target. If the LED does not flash, check out the circuitry to find out why it's not operating. If MPLAB ICD 2 is connected after programming the target with debug disabled, MPLAB can still toggle the reset. If it runs standalone but not with ICD connected (after selecting *Debugger>Run* in the MPLAB IDE), then the interconnections may be suspicious. Use a scope to look at PGC, PGD and VPP. Make sure that the oscillator starts up quickly. If it takes too long to get the target oscillator started, then MPLAB ICD 2 can time out and give errors.

Make sure that the correct oscillator mode is selected (look at *Configure>Configurations Bits*). The oscillator mode will depend upon the oscillator selected. Try HS mode for a crystal, and RC mode for an external resistor/capacitor oscillator. These settings may need to be changed for the final design, but they are some of the easiest modes to start with. Oscillator operation can be checked on OSC2 after configuration bits are set at programming time and power is applied in debug or normal mode.

5.1.6 Link: Application Code

Verify that MPLAB ICD 2 is programming properly by either running an LED blink test similar to that described in Link: Target Oscillator or by programming code into the target device. MPLAB ICD 2 will execute a verify function after programming to confirm that the data in the target PICmicro MCU matches MPLAB's program memory. *Programmer>Verify* can also be selected manually to compare the target PICmicro's memory with the MPLAB IDE image. If the target PICmicro MCU has AVDD and AVSS, ensure that they are connected properly. These lines should be connected to power (VDD) and ground (VSS) respectively. Consult the device data sheet for details. MPLAB ICD 2 will not operate if any of these are available and not connected. Make sure that Low Voltage Programming is disabled (*Config>Configuration Bits...*).

5.1.7 Link: Debug Executive

When using the *Debugger>Program* function from MPLAB IDE, the debug executive will be downloaded and the in-circuit debug registers in the target PICmicro MCU will be enabled. Programming should always be done from the Debug menu not the Programmer menu when debugging. When programming from the Programmer menu (after turning MPLAB ICD 2 off as a debugger), the debug executive will not be downloaded and the in-circuit debug registers will be disabled.

By looking at the *Config>Configuration Bits...* menu selection, it can be determined whether MPLAB is going to download the debug executive. If the bits labeled “Background Debug” are enabled, then the debug executive will be downloaded when the device is programmed.

5.1.8 Link: In Circuit Debug Registers

Make sure that debug is enabled on the MPLAB ICD 2 dialog in MPLAB before programming a device. By looking at the *Config> Configuration Bits...* dialog, it can be verified that the in-circuit debug registers are enabled. The line labeled “Background Debug” should say “Enabled.” Programming should be done from the *Debug* menu, not the *Programmer* Menu.

5.1.9 Link: In-Circuit Debug Resources

Look at the specific registers used for a device in the on-line help. The file registers or program memory areas reserved by MPLAB ICD 2 can not be used. If using the `xxxxxi.lkr` linker script (name ends in “`i`”), then, unless the script has been modified, these resources will be marked reserved and will not be available to the application. If the linker is not being used, use `CBLOCK` or `EQU`s for variable storage in the code, make sure that registers required by MPLAB ICD 2 are not used. The in-circuit debugger will not work if “code protect” or “table read protect” is enabled, if the Watchdog Timer is running, or if the oscillator is not set to the correct mode by the configuration bits. If using fast interrupts or the `CALL FAST` instruction, MPLAB ICD 2 uses the shadow stack and users will not be able to exit properly from the fast interrupt routine or `CALL FAST` function.

Chapter 6. FAQs

6.1 FREQUENTLY ASKED QUESTIONS

These issues are covered in this chapter on frequently asked questions (FAQ):

6.1.1 Why does my system fail to program or verify?

6.1.2 Why do I need the ICD header adapters?

6.1.3 I can't connect to MPLAB ICD 2. What do I do now?

6.1.4 ICD 2 responds with “Target not in debug mode error.” What does this mean?

6.1.5 Can MPLAB ICD 2 operate with a target device running at low voltage?

6.1.6 Does MPLAB ICD 2 support Low Voltage Programming (LVP)?

6.1.7 Why do I have problems when configuring the PLL Oscillator? MPLAB ICD 2 gets “hung up.”

6.1.8 When I try to install, why are the drivers not found, even though I can see them in the driver folders, and I point the driver wizard to the correct folder?

6.1.9 Can I use code protection with MPLAB ICD 2?

6.1.10 How does MPLAB ICD 2 deal with Calibration Data?

6.1.11 Why am I getting erratic values from my EEDATA area?

6.1.12 Why is “Erase All Before Programming” grayed out?

6.1.13 Can my program read and write from Port B or GPIO without interfering with MPLAB ICD 2?

6.1.14 Why are the timers behaving erratically when single-stepping?

6.1.15 Why am I getting warnings and errors when using the PIC12F629/675 or PIC16F630/676?

6.1.16 What would make the power and busy LED blink on and off?

6.1.17 What does the MPLAB ICD 2 "Self Test" do?

6.1.18 Why are the W, STATUS and BSR registers getting changed when using high priority interrupts with the `RETFIE` instruction?

6.1.19 When setting a breakpoint at the first location in my program, why does it stop at address 0001 rather than 0000?

6.1.20 Why is my calibration memory displayed as the erase value?

6.1.21 When single-stepping through the code, my timer times out, but why does my timer interrupt routine not execute?

6.1.1 Q: *Why does my system fail to program or verify?*

A: Check out the PGC, PGD and VPP connections and voltages as described in Chapter 5, Troubleshooting. Make sure the target PICmicro MCU has power applied. If the target PICmicro MCU has AVSS and AVPP pins, verify that these are all connected properly.

6.1.2 Q: *Why do I need the ICD header adapters*?

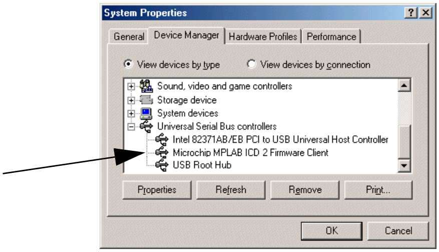
A: The low pin-count parts that are supported by MPLAB ICD 2 could not be used very effectively if in-circuit debug pins are reserved – imagine losing three out of six I/O lines on an eight-pin part! For this reason, special bond-out PICmicros have been manufactured that can emulate these low pin-count parts with an adapter, and which will allow the use of all pins in the target application. The bond-out PICmicro MCU has the in-circuit communications pins to interface to MPLAB ICD 2.

The advantage of this is that the low-pin parts can be used in development with MPLAB ICD 2. The disadvantage is that these parts cannot simply have an ICD 2 connector on the target application for in-circuit debugging. These bond-out PICmicros are similar to emulator chips. They can support more than one device. Jumpers are available on the header boards to configure the chip to match the device under development. These low pin count devices can be programmed with MPLAB ICD 2 using the Universal Programmer Adapter or by putting an ICD 2 connector on the target application to connect VPP, PGC and PGD on these parts.

6.1.3 Q: *I can't connect to MPLAB ICD 2. What do I do now?*

A: Is the MPLAB ICD 2 power light on? The LED should shine brightly. If it is dim, only USB may be connected and the user may need to connect the power supply. The RS-232 connection from the PC cannot supply power to MPLAB ICD 2, but a USB connection can. Note that some USB hubs cannot supply power. Check the on-line help for current troubleshooting hints. Are the USB drivers installed properly? The MPLAB ICD 2 USB driver should be visible in the Windows Device Manager dialog (see Figure 6-1). Some USB hubs cannot supply power to the attached USB devices. A power supply will need to be connected to MPLAB ICD 2 with these hubs.

FIGURE 6-1: SYSTEM PROPERTIES DIALOG



● View devices by type ○ View devices by connection

- Sound, video and game controllers
- Storage device
- System devices
- Universal Serial Bus controllers
  - Intel 82371AB/EB PCI to USB Universal Host Controller
  - Microchip MPLAB ICD 2 Firmware Client
  - USB Root Hub

6.1.4 Q: *ICD 2 responds with "Target not in debug mode error." What does this mean?*

A: Usually this means that MPLAB ICD 2 cannot communicate with the debug executive. The debug executive can only be downloaded by programming the target PICmicro MCU with the user's application from the Debugger>Program menu selection. There are other reasons that the debug executive might not be able to communicate, such as target clock or power supply problems. Check the configuration bits to see that the "Background Debug" is enabled. Look at *Config>Configuration Bits...* to make sure that the Watchdog TImer is disabled, code protection is turned off, and that the oscillator setting is correct.

6.1.5 Q: *Can MPLAB ICD 2 operate with a target device running at low voltage?*

A: Yes. As long as the target PICmicro MCU supports low voltage operation, it can run down to about 2 volts VDD. There are level converters in the input/output buffers of the MPLAB ICD 2. These are powered from the VDD of the target device. Also, MPLAB ICD 2 will sense the operating voltage of the target and correctly adjust its functionality to deal with this operation, i.e., using the correct FLASH erase algorithms. VDD needs to be selected "From Target" in the MPLAB ICD 2 "Settings" dialog, and have a power supply on the target for low voltage operation.

6.1.6 Q: *Does MPLAB ICD 2 support Low Voltage Programming (LVP)?*

A: No. But this does not mean that it will not work correctly when running at low voltage VDD on the target. It just means that the programming voltages applied to VPP will always be +12V.

6.1.7 Q: Why do *I have problems when configuring the PLL Oscillator? MPLAB ICD 2 gets "hung up."*

A: This is a requirement of the actual PICmicro. After programming the configuration bits for the PLL Oscillator, power needs to be removed then re-applied to the target. If this is not done, the target PICmicro MCU will not have a clock. With no clock, debug mode will not function. Also, if power is not removed and reconnected when switching to PLL mode, the device may run, but without the PLL multiplier.

6.1.8 Q: *When I try to install, why are the drivers not found, even though I can see them in the driver folders, and I point the driver wizard to the correct folder?*

A: This problem can be solved by exiting from the driver install wizard, and going to the Control Panel "Add New Hardware" selection. After the system searches for new hardware, choose, "No, the device isn't in the list." Then choose "No, I want to select the hardware from a list." Then choose "Universal Serial Bus Controller," and when the "Have Disk..." button appears, go to the driver folder and select the proper driver.

6.1.9 Q: *Can I use code protection with MPLAB ICD 2?*

A: No. Code protection, especially table read protection on any area in program memory will prevent MPLAB ICD 2 from functioning. Do not use any code protection or table read protection configuration settings when debugging with MPLAB ICD 2. Code protection can be enabled when programming a part for testing without MPLAB ICD 2.

6.1.10 Q: *How does MPLAB ICD 2 deal with Calibration Data?*

A: It is automatically handled. Any values that are in program memory that are used by the PICmicro MCU for calibration data will be read and preserved by MPLAB ICD 2 when erasing, programming and debugging. No action is required to protect this data.

6.1.11 Q: Why am *I getting erratic values from my EEDATA area?*

A: MPLAB ICD 2 can read the EEDATA areas directly, without going through the EECON register's required TABLRD sequence of instructions. The buffers that MPLAB uses can sometimes interfere with the user's code. Avoid going back and forth from program reads of data to MPLAB ICD 2 reads of the EEDATA area when single stepping code.

6.1.12 Q: *Why is "Erase All Before Programming" grayed out?*

A: In some of the newer FLASH parts, the programming algorithm requires that non-contiguous areas of program memory are programmed in banks. For these parts, all memory must be erased before programming.

6.1.13 Q: *Can my program read and write from Port B or GPIO without interfering with MPLAB ICD 2?*

A: Yes. When the in-circuit debug facilities are enabled, PGC and PGD are always used by MPLAB ICD 2, and user code that reads or writes from PORT B will not interfere. Note that the values read from PGC and PGD are not necessarily valid, and writes to these two pins will be ignored. In addition, if interrupts are enabled for changes on Port B, signals on PGC and PGD will not cause an interrupt.

6.1.14 Q: *Why are the timers behaving erratically when single-stepping?*

A: This is one of the drawbacks of using the in-circuit debugger. Since code is actually running in the debug executive, timers can continue to run during the operation of the debug executive, even when the user's application program is "halted."

6.1.15 Q: *Why am I getting warnings and errors when using the PIC12F629/675 or PIC16F630/676?*

A: The GP1/RA1 pins on these parts cannot be pulled high while using MPLAB ICD 2. Refer to the MPLAB ICD 2 Header document for more information on using ICD 2 for these parts (DS51292).

6.1.16 Q: *What would make the power and busy LED blink on and off?*

A: This may indicate that the target MPLAB ICD 2 connector is wired backwards (opposite from the wiring diagram). The blinking indicates that the MPLAB ICD 2 is shutting down due to high current. The following tests indicate that the target is connected backwards:

- Look for blinking Power and Busy LEDs (power may go out altogether).
- Execute a 'Self Test' and look for a min. error on "$\overline{\text{MCLR}}$=V_{PP}" (all other tests will probably pass).
- Use the default address range for the target controller and execute a 'Program' cycle. A voltage level on pin 1 (V_{PP}) of 7-8 volts is too low.

Note that if the target was wired backwards, protection circuitry in the MPLAB ICD 2 will prevent damage to the module. Normal operation should be seen when the target is wired correctly.

6.1.17 Q: *What does the MPLAB ICD 2 “Self Test” do?*

A: The ‘Self Test’ button is helpful in determining problems with the MPLAB ICD 2 module or target connection.

Target VDD

Tests the VDD provided from the MPLAB ICD 2 (5 volts only) if ‘Power from ICD2’ is selected. Tests the VDD provided from the target (2 to 6 volts) if ‘Power from target’ is selected.

Pass/Fail codes:

00 = Pass	VDD is within specified limits
01 = Min error	VDD is below specified limits
80 = Max error	VDD is above specified limits

An error can indicate Incorrect ‘Power’ setting in the Advanced Dialog Box or VDD from the target is above/below specification.

Module VPP

Tests the programming voltage (VPP) provided from the MPLAB ICD 2 to the target VPP/$\overline{\text{MCLR}}$ pin during a programming cycle.

Pass/Fail codes:

00 = Pass	VPP is within specified limits
01 = Min error	VPP is below specified limits
80 = Max error	VPP is above specified limits

An error can indicate the target VPP/$\overline{\text{MCLR}}$ pin is not wired correctly.

$\overline{\text{MCLR}}$=Gnd

Tests the ability of the MPLAB ICD 2 to provide a ground level to the target VPP/$\overline{\text{MCLR}}$ pin for target reset.

Pass/Fail codes:

00 = Pass	Ground can be provided to target VPP/$\overline{\text{MCLR}}$ pin
80 = Max error	Ground level too high for target VPP/$\overline{\text{MCLR}}$ pin

An error can indicate the target VPP/$\overline{\text{MCLR}}$ pin is not wired correctly.

$\overline{\text{MCLR}}$=VDD

Tests the ability of the MPLAB ICD 2 to provide VDD to the target VPP/$\overline{\text{MCLR}}$ pin during normal operation (such as ‘Run’).

Pass/Fail codes:

00 = Pass	VDD can be provided to target VPP/$\overline{\text{MCLR}}$ pin
01 = Min error	VDD too low for target VPP/$\overline{\text{MCLR}}$ pin
80 = Max error	VDD too high for target VPP/$\overline{\text{MCLR}}$ pin

An error can indicate Incorrect ‘Power’ setting in the Advanced Dialog Box or VDD from the target is above/below specification.

$\overline{\text{MCLR}}$=VPP

Test the ability of the MPLAB ICD 2 to provide VPP to the target VPP/$\overline{\text{MCLR}}$ pin during programming operations.

Pass/Fail codes:

00 = Pass	VPP can be provided to target VPP/$\overline{\text{MCLR}}$ pin
01 = Min error	VPP too low for target VPP/$\overline{\text{MCLR}}$ pin
80 = Max error	VPP too high for target VPP/$\overline{\text{MCLR}}$ pin

An error can indicate the target VPP/$\overline{\text{MCLR}}$ pin is not wired correctly.

6.1.18 Q: Why are *the W, STATUS and BSR registers getting changed when using high priority interrupts with the* `RETFIE` *instruction?*

A: The shadow registers, which are used for high priority interrupts and CALL FAST are used by MPLAB ICD 2. These are reserved resources for MPLAB ICD 2 operation. A problem will be encountered if a breakpoint is set inside a `CALL FAST` subroutine or a inside the service routine for a high priority interrupt that uses the shadow registers with the `RETURN FAST` or `RETFIE` instruction.

6.1.19 Q: *When setting a breakpoint at the first location in my program, why does it stop at address 0001 rather than 0000?*

A: MPLAB ICD 2 stops at the instruction *after* the breakpoint. This means that a breakpoint at address 0000 will be executed, then the program counter will be pointing at address 0001 when it gets a breakpoint. If a user needs to stop at the first instruction in their code, they must put a NOP at address 0000.

6.1.20 Q: *Why is my calibration memory displayed as the erase value?*

A: MPLAB IDE is displaying the default memory value. A device read must be performed with MPLAB ICD 2 to display the actual value on the device.

6.1.21 Q: *When single-stepping through the code, my timer times out, but why does my timer interrupt routine not execute?*

A: When single-stepping, the in-circuit debugger will not allow the PICmicro MCU to respond to interrupts. If it did, and users had external interrupts, then single-stepping would almost always end up in interrupt routines. To debug an interrupt, set a breakpoint *inside* the interrupt service routine and run to get a breakpoint after the interrupt.

Index

NOTES:

NOTES:

NOTES:

Worldwide Sales and Service

AMERICAS

Corporate Office

Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
480-792-7627
Web Address:
www.microchip.com

Atlanta

Tel: 770-640-0034
Fax: 770-640-0307

Boston

Tel: 978-692-3848
Fax: 978-692-3821

Chicago

Tel: 630-285-0071
Fax: 630-285-0075

Dallas

Tel: 972-818-7423
Fax: 972-818-2924

Detroit

Tel: 248-538-2250
Fax: 248-538-2260

Kokomo

Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles

Tel: 949-263-1888
Fax: 949-263-1338

Phoenix

Tel: 480-792-7966
Fax: 480-792-4338

San Jose

Tel: 408-436-7950
Fax: 408-436-7955

Toronto

Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia

Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing

Tel: 86-10-85282100
Fax: 86-10-85282104

China - Chengdu

Tel: 86-28-86766200
Fax: 86-28-86766599

China - Fuzhou

Tel: 86-591-7503506
Fax: 86-591-7503521

China - Shanghai

Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen

Tel: 86-755-82901380
Fax: 86-755-82966626

China - Hong Kong SAR

Tel: 852-2401-1200
Fax: 852-2401-3431

India

Tel: 91-80-2290061
Fax: 91-80-2290062

Japan

Tel: 81-45-471-6166
Fax: 81-45-471-6122

Korea

Tel: 82-2-554-7200
Fax: 82-2-558-5934

Singapore

Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan

Tel: 886-2-2717-7175
Fax: 886-2-2545-0139

EUROPE

Austria

Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark

Tel: 45-4420-9895
Fax: 45-4420-9910

France

Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany

Tel: 49-89-627-144-100
Fax: 49-89-627-144-44

Italy

Tel: 39-0331-742611
Fax: 39-0331-466781

United Kingdom

Tel: 44-118-921-5869
Fax: 44-118-921-5820

02/12/03