MPLAB™

IDE, SIMULATOR, EDITOR USER'S GUIDE

MPLAB™
IDE, SIMULATOR, EDITOR
USER'S GUIDE

Development Systems User's Guide

Table of Contents

General Information

Part 1 – IDE and Simulator

Chapter 1. MPLAB Preview

Chapter 2. MPLAB Installation

Chapter 3. Getting Started with MPLAB – A Tutorial

Part 2 – Editor

Part 3 – Appendices

Appendix E. 16-Bit Core Device Simulator Issues

Glossary

General Information

MPLAB™ User's Guide

General Information

Introduction

This first chapter contains general information that will be useful to know before running MPLAB.

Highlights

The information you will garner from this chapter:

- About This Guide
- Warranty Registration
- Recommended Reading
- The Microchip Internet Website
- Customer Support

About This Guide

Document Layout

This document describes how to use MPLAB. The manual layout is as follows:

Part 1 – IDE and Simulator

- **Chapter 1: MPLAB Preview** – An overview of what MPLAB is and how it works.
- **Chapter 2: MPLAB Installation** – How to install MPLAB on your computer.
- **Chapter 3: Getting Started with MPLAB – A Tutorial** – How to begin using MPLAB.
- **Chapter 4: MPLAB Projects Tutorial** – A tutorial on how to use MPLAB projects.
- **Chapter 5: MPLAB Basic Functions** – A discussion of the basic MPLAB debugging functions, including MPLAB-SIM, and of the interrelationship of these functions.
- **Chapter 6: MPLAB Menu and Toolbar Options** – A description of the options available via the MPLAB menus and toolbars.

Part 2 – Editor

- **Chapter 1: MPLAB Editor Preview** – Gives the user a quick look at what the MPLAB Editor is and how it benefits the user in firmware development.
- **Chapter 2: Using the MPLAB Editor** – Describes the procedures for using the MPLAB Editor functions.
- **Chapter 3: MPLAB Editor Menu Commands** – Gives detailed information only on MPLAB Editor Menu Options. The items are organized in the same order as seen in the pull down menus.
- **Chapter 4: MPLAB Editor Default Key Commands** – Give a brief description of each default key command. The entries are organized by function.
- **Chapter 5: MPLAB Editor Error Messages** – Gives information suggesting possible actions to take when receiving an error message. This chapter contains warning and informative messages.
- **Chapter 6: MPLAB Text Editor Command Line Options** – Describes command line options available when starting MPLAB.

Part 3 – Appendices

- **Appendix A: MPLAB Key Mapping Functions** – Lists the available MPLAB key mapping functions.
- **Appendix B: Customizing MPLAB After Installation** – Discusses the user configurable customizations available by modifying MPLAB configuration file (MPLAB.ini).
- **Appendix C: MPLAB-SIM PIC16C5X Simulator Issues** – Discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using the PIC16C5X family of microcontrollers.
- **Appendix D: MPLAB-SIM PIC16CXX Simulator Issues** – Discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using the PIC16CXX and PIC14000 families of microcontrollers.
- **Appendix E: MPLAB-SIM PIC17CXX Simulator Issues** – Discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using the PIC17CXX family of microcontrollers.
- **Glossary** – A glossary of terms used in this guide.
- **Index** – Cross-reference listing of terms, features and sections of this document.
- **Worldwide Sales and Service** – A listing of Microchip sales and service locations and telephone numbers worldwide.

Conventions Used in this Guide

This manual uses the following documentation conventions:

Table: Documentation Conventions

Description	Represents	Examples
Angle Brackets: < >	Delimiters for special keys.	<TAB>, <ESC>
Pipe Character: |	Choice of mutually exclusive arguments; an OR selection	
Lower case characters	Type of data	hex
Italic characters	A variable argument; it can be either a type of data (in lower case characters) or a specific example (in uppercase characters).	
`Courier Font`	User entered code or sample code	`#define ENIGMA`
Underlined, Italics Text with Right Arrow	Defines a menu selection from the menu bar.	*File > Save*
0xnnn	0xnnn represents a hexadecimal number where n is a hexadecimal digit	0xFFFF, 0x007A
In-text Bold Characters	Designates a button.	**OK, Cancel**

Updates

All documentation becomes dated, and this user's guide is no exception. Since MPLAB and other Microchip tools are constantly evolving to met customer needs, some MPLAB dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site to obtain the latest documentation available.

Warranty Registration

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in your Warranty Registration Card entitles you to receive new product updates. Interim software releases are available at the Microchip web site.

Recommended Reading

This user's guide describes how to use MPLAB. The user may also find the data sheets for specific microcontroller devices informative in developing in developing firmware.

README.LAB
For the latest information on using MPLAB, read the README.LAB file (an ASCII text file) on the MPLAB CD-ROM. README.LAB contains update information that may not be included in the MPLAB User's Guide.

README.XXX
For the latest information on using other tools, refer to an information file about the product that is more current than the printed manual. Check the MPLAB directory for other README files. (In the case of MPASM, for instance, the file is called README.ASM.)

Microchip Technology Library CD-ROM (DS00161)
This CD-ROM contains comprehensive application notes, data sheets, and technical briefs for all of Microchip products. To obtain this CD-ROM, contact the nearest Microchip Sales and Service location (see back page).

Embedded Control Handbook Vol.1 & 2 (DS00092 & DS00167)
These handbooks contain a wealth of information about microcontroller applications. To obtain these documents, contact the nearest Microchip Sales and Service location (see back page).

The application notes described in these manuals are also obtainable from Microchip Sales and Service locations or from the Microchip website (http://www.microchip.com).

Microsoft Windows Manuals
This manual assumes that users are familiar with Microsoft Windows operating system. Many excellent references exist for this software program, and should be consulted for general operation of Windows.

The Microchip Internet Website

Microchip provides on-line support on the Microchip World Wide Web (WWW) site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® Communicator or Microsoft® Internet Explorer®. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Website

The Microchip website is available by using your favorite Internet browser to attach to:

www.microchip.com

The file transfer site is available by using an FTP service to connect to:

ftp://ftp.microchip.com

The website and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles, and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Corporate Applications Engineer (CAE)
- Hot line

Customers should call their distributor, representative, or field application engineer (FAE) for support. Local sales offices are also available to help customers. See the back cover for a listing of sales offices and locations.

Corporate applications engineers (CAEs) may be contacted at (602) 786-7627.

In addition, there is a Systems Information and Upgrade Line. This line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive any currently available upgrade kits.

The Hot Line Numbers are:

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

1-602-786-7302 for the rest of the world.

Part 1 – IDE and Simulator

MPLAB™ User's Guide

Chapter 1. MPLAB Preview

1.1 Introduction

This chapter will give an overview of MPLAB.

1.2 Highlights

In this chapter, we discuss:

- What is MPLAB
- How MPLAB Helps You
- MPLAB – An Integrated Development Environment (IDE)
- MPLAB Development Tools

1.3 What is MPLAB

MPLAB is a Windows®-based Integrated Development Environment (IDE) for the Microchip Technology Incorporated PICmicro microcontroller families. MPLAB allows you to write, debug, and optimize PICmicro applications for firmware product designs. MPLAB includes a text editor, simulator, and project manager. MPLAB also supports the MPLAB-ICE and PICMASTER® emulators, PICSTART® Plus and PRO MATE® II programmers, and other Microchip or third party development system tools.

1.4 How MPLAB Helps You

The organization of MPLAB tools by function helps make pull-down menus and customizable quick keys easy to find and use. MPLAB tools allow you to:

- Assemble, compile and link source code
- Debug the executable logic by watching program flow with the simulator, or in real time with the MPLAB-ICE emulator
- Make timing measurements
- View variables in watch windows
- Program firmware with PICSTART Plus or PRO MATE II
- Find quick answers to questions from the MPLAB on-line Help

and much more.

1.5 MPLAB – An Integrated Development Environment (IDE)

MPLAB is an easy-to-learn and use Integrated Development Environment (IDE). The IDE provides firmware development engineers the flexibility to develop and debug firmware for Microchip's PICmicro microcontroller families. The MPLAB IDE runs under Microsoft Windows 3.1x, Windows 95 or later.

MPLAB provides functions that allows you to:

- Create and Edit Source Files
- Group Files into Projects
- Debug Source Code
- Debug Executable Logic Using the Simulator or Emulator(s)

The MPLAB IDE allows you to create and edit source code by providing you with a full-featured text editor.

Further, you can easily debug source code with the aid of a Build Results window that displays the errors found by the compiler, assembler, and linker when generating executable files.

A Project Manager allows you to group source files, precompiled object files, libraries, and linker script files into a project format.

The MPLAB IDE also provides feature-rich simulator and emulator environments to debug the logic of executables. Some of the features are:

- A variety of windows allowing you to view the contents of all data and program memory locations
- Source Code, Program Memory, and Absolute Listing windows allowing you to view the source code and its assembly-level equivalent separately and together (Absolute Listing)
- The ability to step through execution, or apply Break, Trace, Standard, or Complex Trigger points.

1.6 MPLAB Development Tools

The MPLAB IDE integrates several tools to provide a complete development environment.

- **MPLAB Project Manager**

  Use the Project Manager to create a project and work with the specific files related to the project. When using a project, source code is rebuilt and downloaded to the simulator or emulator with a single mouse click.

- **MPLAB Editor**

  Use the MPLAB Editor to create and edit text files such as source files, code, and linker script files.

- **MPLAB-SIM Simulator**

  The software simulator models the instruction execution and I/O of the PICmicro Microcontrollers (MCUs).

- **MPLAB-ICE Emulator**

  The MPLAB-ICE emulator uses hardware to emulate PICmicros in real time, either with or without a target system.

- **MPASM Universal Assembler/MPLINK Relocatable Linker/ MPLIB Librarian**

  The MPASM assembler allows source code to be assembled without leaving MPLAB. MPLINK creates the final application by linking relocatable modules from MPASM and MPLAB-C17. MPLIB manages custom libraries for maximum code reuse.

- **MPLAB-C17 C Compiler**

  The MPLAB-C17 C Compiler provides an ANSI-based high level source code solution. Complex projects can use a combination of C and assembly source files to obtain the maximum benefits of speed and maintainability.

- **PRO MATE® II and PICSTART® Plus Programmers**

  Develop code with the simulator or an emulator, assemble or compile it, and then use one of these tools to program devices. This can all be accomplished with MPLAB. Although PRO MATE II does not require MPLAB to operate, programming is easier using MPLAB.

- **PICMASTER and PICMASTER-CE Emulators**

  These emulators use hardware to emulate PICmicros in real time, either with or without a target system. MPLAB-ICE is the newest emulator from Microchip.

- **Third Party Tools**

  Many other companies have development tools for Microchip products that work with MPLAB. Consult the *Microchip Third Party Guide* (DS00104).

NOTES:

Chapter 2. MPLAB Installation

2.1 Introduction

This chapter describes the procedure for installing MPLAB.

2.2 Highlights

The items discussed in this chapter include:

- Host Computer System Requirements
- Obtaining the Program Files
- Installing MPLAB
- Uninstalling MPLAB

2.3 Host Computer System Requirements

The following minimum configuration is required to run MPLAB:

- PC compatible 486 or better class system (Pentium recommended)
- Microsoft Windows 3.1x or Windows 95/98
- VGA display (Super VGA recommended)
- 8 MB memory (32 MB recommended)
- 20 MB of hard disk space
- Mouse or other pointing device

2.4 Obtaining the Program Files

The MPLAB application is shipped with every Microchip Development System. Also, MPLAB may be obtained by contacting any Microchip sales office and requesting the Technical Library CD-ROM or by downloading the files from the Microchip website (www.microchip.com).

Depending on the version, the files are named differently. Version 4.00 of MPLAB, for example, would have these files:

MP40000.EXE

MP40000.W02

MP40000.W03

MP40000.W04

MP40000.W05

MP40000.W06

2.5 Installing MPLAB

The executable file MPvvvvv.EXE installs the Microchip MPLAB Integrated Development Environment, where vvvvv is the version number of MPLAB.

To install MPLAB, follow these steps:

1. Enter Microsoft Windows
2. If you are installing from the MPLAB CD-ROM, place the CD-ROM into the drive now.
3. Execute the install program:

   Windows 3.1: From the File Manager, or from the *Program Manager>Run* option, run **X:\MPvvvvv.exe**, where **X** is the drive designation of the install files.

   Windows 95/98: Click the Start button and select Run. Enter **X:\MPvvvvv.exe**, where **X** is the drive designation of the install files. Then click **OK**.

4. When the .EXE file is run, it will start installing MPLAB onto your system. You will be prompted for the elements of MPLAB you'd like to install on your system.

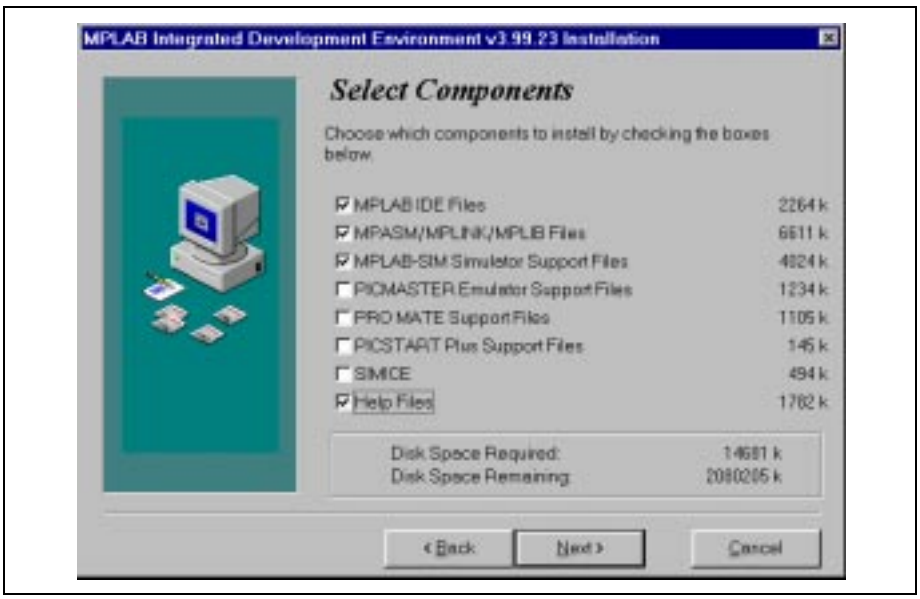


Figure 2.1: Select Components Dialog

Unless you have purchased a device programmer or emulator, you should install just the software tools:

- MPLAB IDE files
- MPASM/MPLINK/MPLIB files
- MPLAB-SIM Simulator Support Files
- Help Files

You can re-install MPLAB later to add additional components.

5. The next menu allows you to select which Microchip language components you want. Usually you should select them all (the default).

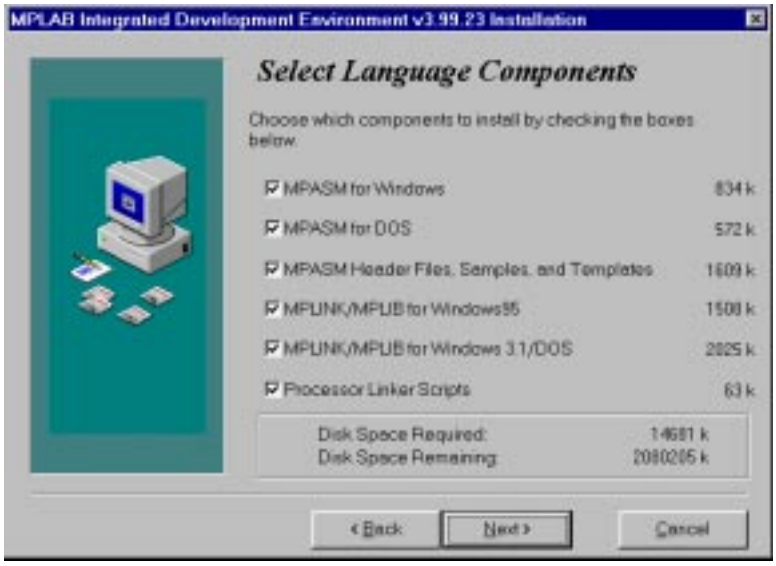


Figure 2.2: Select Language Components Dialog

6. The install program will now commence to install MPLAB on your computer. After the installation, execute MPLAB.EXE or click on the MPLAB icon to start up the system. You will see MPLAB's desktop as in Figure 2.3.

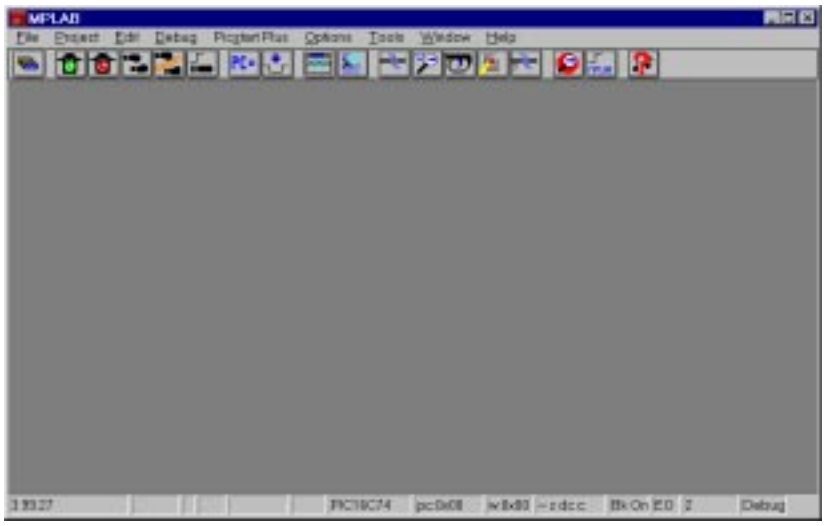


Figure 2.3: MPLAB Desktop

2.6 Uninstalling MPLAB

The MPLAB install program keeps track of the files that it installs in a file called INSTALL.LOG. The log file (INSTALL.LOG) has a record of each file copied during the installation process.

If you wish to remove MPLAB from your system, the uninstall routine uses the INSTALL.LOG file to determine which files to remove from the MPLAB, Windows, and System directories.

To uninstall, run `unwise.exe` in the MPLAB directory.

Chapter 3. Getting Started with MPLAB – A Tutorial

3.1 Introduction

This tutorial is intended to be a quick introduction to the MPLAB user interface. It should take about 1 to 2 hours to get through the Tutorial.

This is not intended to discuss all of the details of MPLAB, only to provide a beginning understanding so you can use MPLAB right away.

3.2 Highlights

In this tutorial, you will learn about:

- Setting up the Development Mode
- Creating a Simple New Project
- Creating a Simple New Source File
- Entering Source Code
- Assembling the Source File
- Running Your Program
- Opening Other Windows for Debugging
- Creating a Watch Window
- Saving the Watch Window
- Setting a Break Point

In addition, there is an overview of other features to be discussed in later chapters.

With the operation of MPLAB provided by this tutorial, you should be able to:

- Become familiar with the MPLAB Desktop
- Create a new assembly source code file and enter it into a new project for the PIC16F84
- Identify and correct simple errors
- Run the built in simulator
- Set break points
- Create watch windows
- Become familiar with the various debugging windows

3.3 Setting up the Development Mode

The previous chapter discussed how you would install MPLAB. Now you will begin setting up the application.

The basic MPLAB desktop looks like most Windows applications (Figure 3.1) with the menu across the top line, a tool bar, and it also includes a status bar at the bottom. See that the status bar includes information about how the system is currently configured. We'll cover some of these features in more detail later. For now, let's see how to set the development mode.

The development mode sets which tool, if any, will execute code. For this tutorial we will use MPLAB-SIM, the software simulator. Later you may switch to one of the emulator operations if you have an emulator. Operation will be similar. "Editor Only" mode does not allow code execution, and is mainly useful if you have not installed the simulator, do not have an emulator, and are just creating code to program a PICmicro.

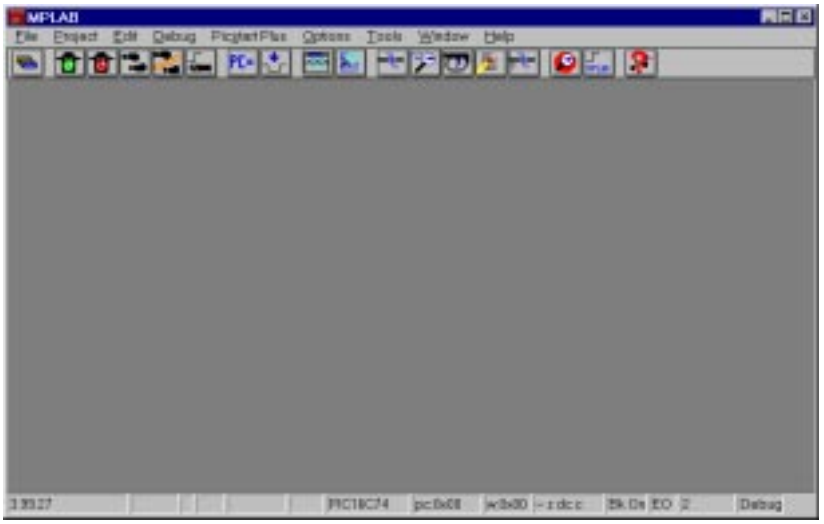


Figure 3.1: MPLAB Desktop

Selecting the *Options>Development Mode* menu item, you will find a dialog box that looks something like Figure 3.2.

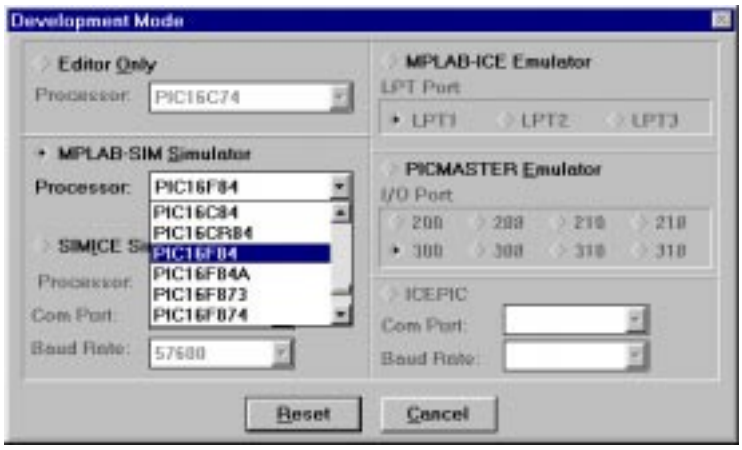


Figure 3.2: Development Mode Dialog

MPLAB is a constantly evolving product, so there may be subtle differences between what you see and the picture here. Select the radio button next to MPLAB-SIM Simulator and choose the PIC16F84 from among the pull down list of available processors supported by the simulator. Click PIC16F84, and then press the **Reset** button. The simulator will initialize and you should see "PIC16F84" and "Sim" in the status bar on the bottom of the MPLAB desktop. You are now in simulator mode for the PIC16F84 device.

3.4 Creating a Simple New Project

The simulator runs from the same file, called a "hex file," that can be programmed into the PICmicro. For the simulator to run you must first create a source code file and successfully assemble the source code.

The assembler produces, among other things, a hex file. This file has the file extension `.hex`. In this tutorial the file would be named `tutor84.hex`. Later this file can be loaded directly into a device programmer without using the assembler or an MPLAB project. This file can also be loaded by most other third party programmers.

Select *File>New* from the menu and you will be confronted with a dialog that looks like Figure 3.3.

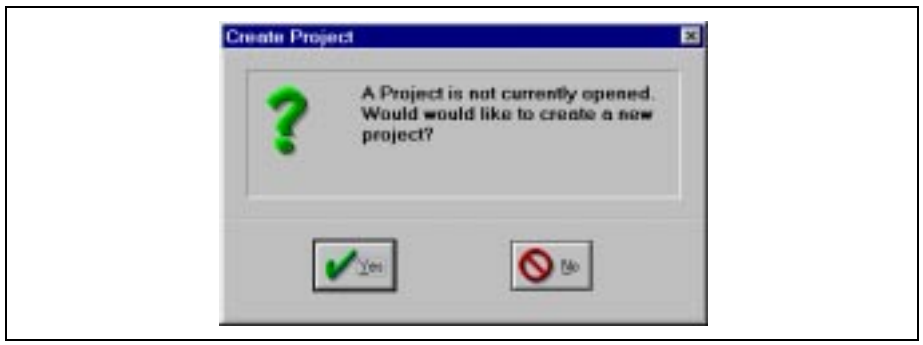


Figure 3.3: Create Project Dialog

Click on the **Yes** button, and a standard Windows browsing dialog will appear (Figure 3.4). Decide where you want to create your project and remember where you put it. You'll need this information later. This tutorial uses a directory in c:\temp\tutorial and creates the project file named `tutor84.pjt`.

"PJT" is a standard suffix for MPLAB project files. The prefix of the project file name, in this case `tutor84`, will become a default prefix for many of the files that MPLAB will use or create for this tutorial.

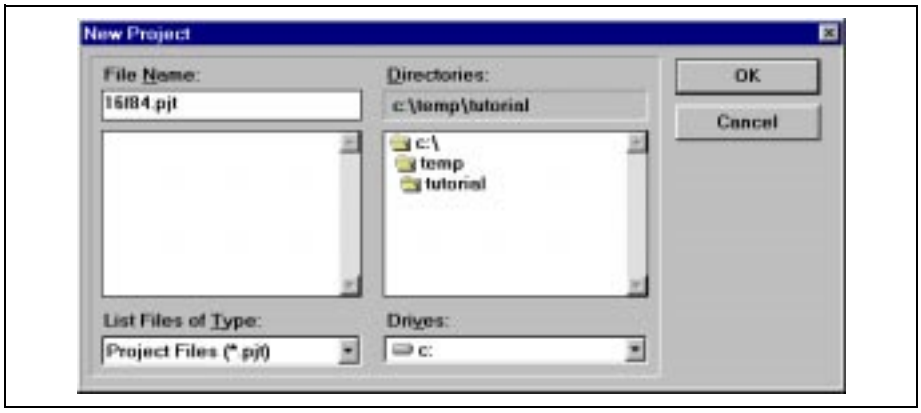


Figure 3.4: New Project Dialog

Using the mouse to click on **OK** will bring you to the Edit Project Dialog (Figure 3.5).

The simulator, programmers and emulator systems that work with MPLAB use a hex file created by assembling, compiling and/or linking source code. Several different tools can create hex files, and these tools are part of each project. Projects give you flexibility to describe how the application will be built and which software tools will be used to create the `.hex` file. We will not get into these details in this tutorial, but as you need these features you will be using the Node Properties to set these. See Chapter 4 for more information on more complex projects.

Figure 3.5: Edit Project Dialog – Node Properties Enabled

Notice that the target file name of the Edit Project dialog has been filled in for you. It already knows about the development mode that we set previously and it is assuming that we will be using the Microchip language tool suite. In the Project Files window, you will find `tutor84.`[hex]. Highlighting this name will cause the Node Properties button to become usable. Before we do anything else, we must tell MPLAB how to create the hex file. Do this by clicking the **Node Properties** button. The Node Properties dialog will appear (Figure 3.6).

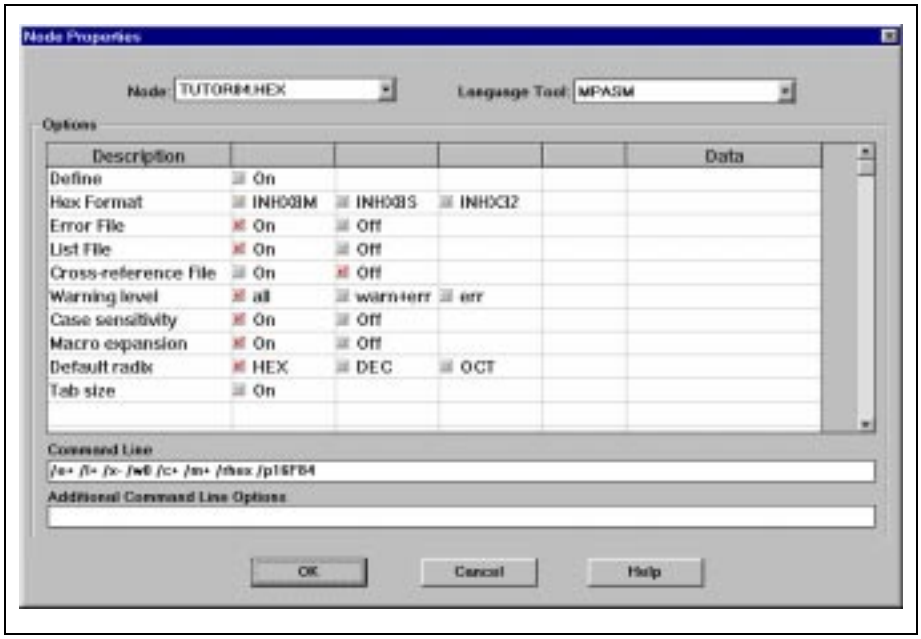


Figure 3.6: Node Properties Dialog

This dialog contains all of the default settings for a language tool, in this case MPASM, as can be seen in the upper right of the dialog. In the simplest form, a project contains a hex file created from one assembly source file. This is the default as the Node Properties dialog appears.

You can see that there are a lot of rows and columns on this dialog. Each row usually corresponds to a "switch," those things that are often set on the command line when the tool is invoked. In fact, the setting of these switches is reflected in the Command Line window near the bottom. This is the actual command line that will be issued to MPASM when it is invoked from MPLAB.

For now you can use the default settings for all other entries, but as you become more familiar with building an application, you will probably find that you'll want to change some of these.

Clicking the **OK** button will apply these defaults, bring you back to the Edit Project dialog and enable the **Add Node** button (Figure 3.7).

Figure 3.7: Edit Project Dialog – Add Node Enabled

Press the **Add Node** button. You will be confronted with the standard windows browse dialog (Figure 3.8), with the working directory being the same as that for the project. Enter the file name `tutor84.asm`, and press **OK**.

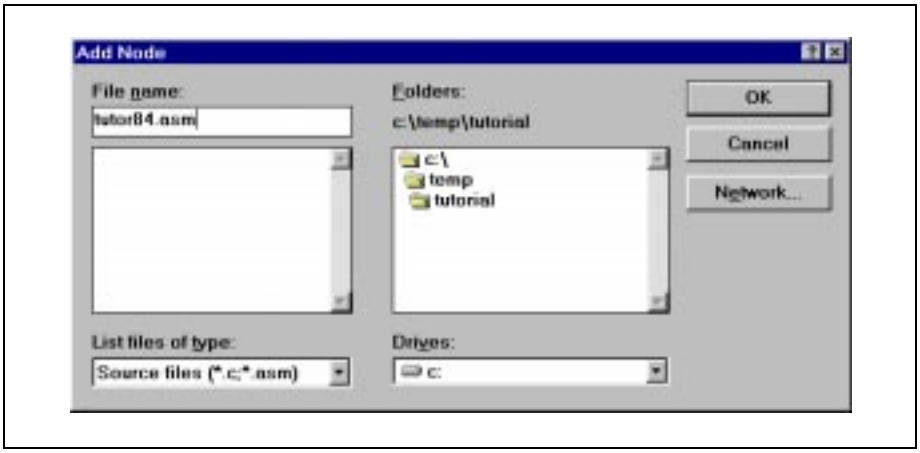


Figure 3.8: Add Node Dialog

You will return to the Edit Project dialog and should see `tutor84.asm` indented and below the hex file, indicating that it is a contributing node (Figure 3.9).

Figure 3.9: Edit Project Dialog – Node Added

Pressing **OK** takes you back to the MPLAB desktop with an open and as yet unnamed source code file.

3.5 Creating a Simple New Source File

Click once in the blank space of the empty file window that has been created for you. It is probably titled “Untitled.” This gives the window “focus.” Use the *File>Save As...* menu option and save the empty file as `tutor84.asm`. When the standard browse dialog opens, you will find that it is located in the current working directory of the project. Enter the file name and press **OK**.

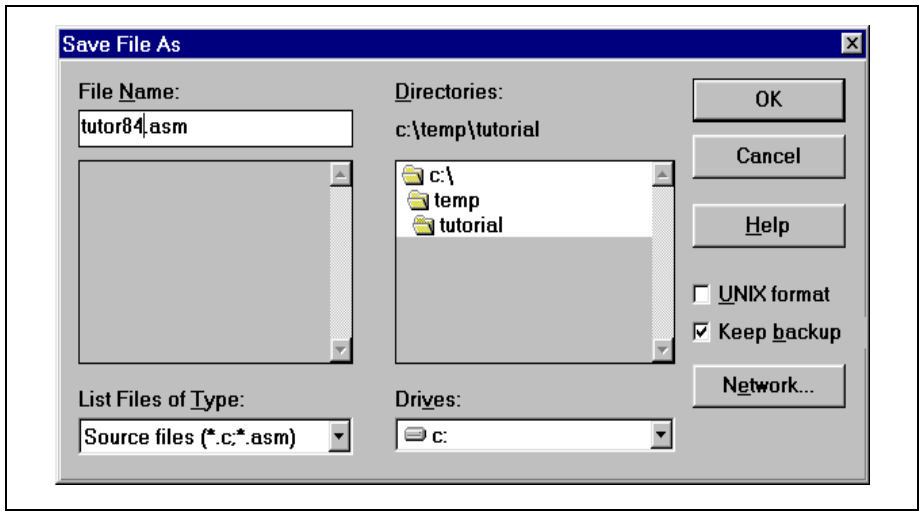


Figure 3.10: Save Source File

You will now be presented with the MPLAB desktop and the empty file window, but the name of the file window will reflect its new name.

The name of the source file and the name of the project (`tutor84` in this tutorial) must be the same in this kind of project. There are other projects that use the linker and allow the output file name to be different from the input file (and there is a separate tutorial for projects using the linker). In this tutorial, for the single source file type of project, MPASM will always create an output hex file with the same name as the source file, and this cannot be changed. If you change the name of the source file, you must also change the name of the project to match.

Note: MPASM creates its output hex file with the same name as the source file. For this reason, when you use MPASM for a single file project, the project name, hex file and source file MUST have the same name.

3.6 Entering Source Code

Use the mouse to locate the cursor at the beginning of the `tutor84.asm` empty file window, and enter the following text, exactly as written on each line. You don't have to enter the comments, the text following the semicolons.

```
    list    p=16f84
    include <p16F84.inc>
c1  equ     0x0c   ; Set temp variable counter c1 at address 0x0c

    org     0x00   ; Set program memory base at reset vector 0x00
reset
    goto    start  ; Go to start of the main program

    org     0x04   ; Set program memory base to beginning of user code
start
    movlw   0x09   ; Initialize counter to arbitrary value greater than zero
    movwf   c1     ; Store value in temp variable a defined above
loop
    incfsz  c1,F   ; Increment counter, place results in file register
    goto    loop   ; Loop until counter overflows

    goto    bug    ; When counter overflows, got to start to re-initialize
    end
```

This code is a very simple program that increments a counter and resets to a predetermined value when the counter rolls over to zero.

All labels start in the first column, and the last line has an `end` directive. Refer to the *MPASM with MPLINK and MPLIB User's Guide* for more information about directives. The PICmicro data sheets contain full information about instructions along with examples of their use.

Save the file by using the *File>Save* menu function.

3.7 Assembling the Source File

Assembling the file can be accomplished in several ways. Described here is one method. Use the *Project>Build All* menu item. This will execute the MPASM assembler in the background using the defaults saved with the project as noted before. Once the assembly process is complete, the Build Results window will appear (Figure 3.11).

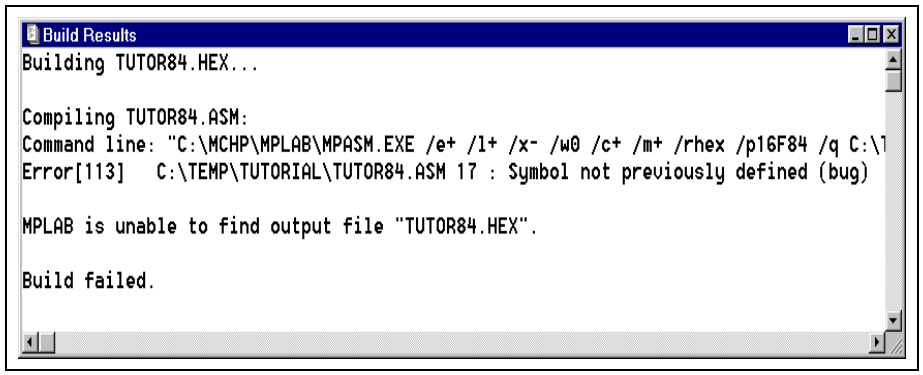


Figure 3.11: Build Results Window – Build Failed

You have intentionally entered at least one error if you entered the code as written above. The last `goto` in the program references a non-existent label called `bug`. Since this label has not been defined before, the assembler reports an error. You may have other errors as well.

Using the mouse, double click on the error message. This will bring the cursor to the offending line in the source code. Change `bug` to `start`. Use the Build Results window to help find the errors, and repair any other bugs in your source code. Reassemble by executing the *Project>Build All* menu function. This process may take a couple of iterations.

Note: Whenever you rebuild a project all of your source files will be saved to disk.

When you've fixed all problems in the source code, the Build Results window will display "Build completed successfully" (Figure 3.12). You now have a complete project that can be executed using the simulator.

Figure 3.12: Build Results Window – Build Successful

3.8 Running Your Program

Use the *Debug>Run>Reset* to initialize the system. The program counter will be reset to zero, which is the reset vector on the PIC16F84. The source code line at this address may be highlighted with a dark bar. Also, you may notice that PC is set to 0x00 in the status bar at the bottom of MPLAB.

Use the *Debug>Run>Step* menu item. This causes the program counter to advance to the next instruction location. The dark bar will follow the source code and the program counter displayed in the status bar should advance to "pc:0x04."

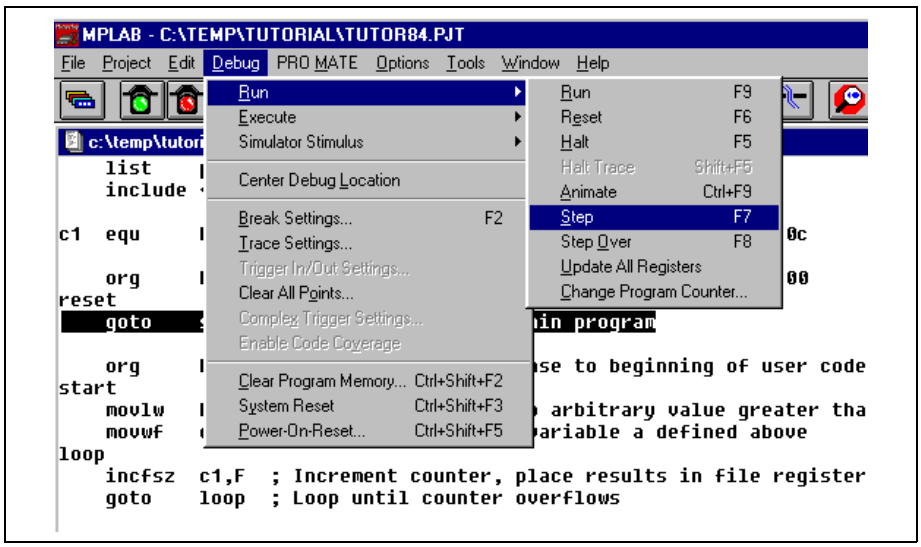


Figure 3.13: Debug>Run>Step Menu Item

You may notice as you execute the *Debug>Run>Step* menu item that there is text on the right side of the menu item that says <F7>. This stands for "function key seven" on your keyboard. Many MPLAB functions are assigned to "hot keys." These keys have the same affect as executing the menu item itself. Press <F7> a few times and watch the program counter and dark bar advance through the program.

Execute the *Debug>Run>Run* menu item or press <F9> to start the program running from the current location counter. The status bar will change colors indicating the program is executing instructions. None of the other fields on the status bar will be updated until the program is halted.

Stop the program by executing the *Debug>Run>Halt* menu item or by pressing <F5>. The status bar will change back to its original color, and the current program counter and other status information will be updated.

Another way to execute functions is to use the tool bar at the top of the screen. If you place the cursor over the items in the tool bar, you can see the name of the function in the status bar at the bottom. The left button is a standard **Change Tool bar** button that allows you to scroll through the

available toolbars. These can be customized. On the debug toolbar, the green light is equivalent to <F9> (Run) and the red light is the same as <F5> (Halt).

3.9 Opening Other Windows for Debugging

There are many ways to look at your program and its execution using MPLAB. For example, this program is intended to increment a temporary counter, but how do you know for sure that is happening? One way is to open and inspect the file register window. Do this by executing the *Window>File Registers* menu item. A small window with all of the file registers, or RAM, of the PIC16F84 will appear.

Press <F7> (Execute Single-Step) a few times and watch the values update in the file register window. We put the counter variable at address location 0x0C. As the temporary counter is incremented, this is reflected in the file register window. File registers change colors when their value changes so that they can easily be noticed on inspection. However, in very complex programs, many values may change, making it difficult to focus on one or two variables. This problem can be solved by using a watch window.

3.10 Creating a Watch Window

Execute the *Window>New Watch Window* menu item. The Add Watch Symbol dialog will appear (Figure 3.14).

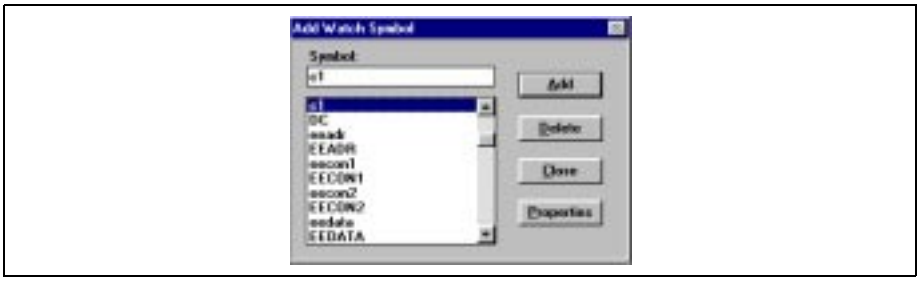


Figure 3.14: Add Watch Symbol Dialog

Typing ‘c1’ in the symbol name box will cause the list to scroll to the desired symbol. Highlight the symbol and press the **Add** button, and then press the **Close** button. You will be left with the watch window on your MPLAB desktop (Figure 3.15) displaying the current value of the temporary counter value ‘c1.

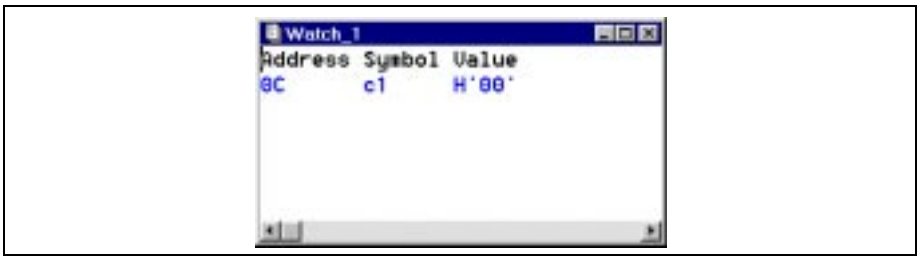


Figure 3.15: Watch Window

Press <F7> to single step the program a few times and notice that as the counter value is incremented, the display is updated in the watch window. If you've left the file register window open, it will update as well.

3.11 Saving the Watch Window

You can save the watch window and its settings by executing the *Save Watch* item under the system menu of the watch window. The system menu button is located in the upper left-hand corner of the watch window. Clicking this button once will cause the menu underneath to cascade down. Select *Save Watch* and you will be presented with the standard browsing dialog located in the current working directory of the project. Choose any arbitrary name and press **OK**.

The window's open or closed status and location on the screen is saved with the project so the next time you open your project, your watch windows will be restored as well.

You can also edit the watch windows after you've created them. Using the system menu, select *Add Watch* to bring up a dialog to add more items. The <Ins> key will do the same. Highlighting an item and pressing the <Del> key will delete that watch from the window. You can select *Edit Watch* from the system menu to change how that item is displayed (in hex, binary, as a 16-bit variable rather than 8-bit, etc.).

3.12 Setting a Break Point

Press <F5> (*Debug>Run>Halt*) to make sure that the simulator processor is halted. Click in the source code window on the line immediately after the `start` label that says `movlw 0x09`. Press the right mouse button and a menu will appear (Figure 3.16).

Figure 3.16: Right Mouse Button Menu

Click on the *Break Point(s)* menu item and the menu will disappear and the line where the cursor was located will change colors, indicating that a break point has been set at that location.

Press <F6>, or execute the *Debug>Run>Reset* menu item, to reset the system. Then run the system by pressing <F9>. The program will run and then halt at the instruction just after the break point. 'c1', as displayed in either the watch window or the file register window (if one is still open), will

reflect the reset status of zero, stepping once will execute the load and ‘c1’ will reflect a value of 0x09. Press <F9> a few times and notice the status bar change color while running and then change again when the processor halts

3.13 Summary

This tutorial has shown you how to:

- set up a new project
- create and enter a source file into a project
- assemble code
- run your code using the simulator
- set break points and single step your code
- watch variables in your code

Once you are comfortable with the topics introduced here, you should look at the next section for more information on MPLAB.

Some hints and tips:

Break Points – You can set break points in the *Windows>Program Memory* window, in the source file window (in this case `tutor84.asm`), or in the *Windows>Absolute Listing* window.

Source Files – Use the *Window>Project Window* to bring up a list of your source files. You can double click on the file name here to bring up that file in the editor.

MPASM Errors – If MPASM gives you an error, double click on that error in the error window to go to the error in the source code. If you’ve got multiple errors, always choose the first error. Often one error will cause subsequent errors and fixing the first one may fix them all.

Configuration Bits and Processor Mode – Configuration bits in the source file will not set the mode of the processor for the simulator (or emulators). Use *Options>Processor Setup>Hardware* for those settings. Even though you can set these bits in your MPASM or MPLAB-C17 source file, MPLAB does not automatically change modes.

For instance, the Watch Dog Timer Enable configuration bit can be set so that when you program a device, the Watch Dog Timer will be turned on. In MPLAB you will also need to go to the *Options>Processor Setup>Hardware* dialog to enable the WDT for the simulator or emulator. This allows you to debug with it on or off without changing your source code.

Options – Go to *Options>Environment Setup* to do the following:

- Map European Keys to MPLAB functions and special ASCII characters
- Change the screen font or font size
- Position the tool bar on the side or bottom of the screen
- Modify the tool bar
- Change the number of characters displayed for labels

Map Files – Go to *Project>Edit Project* dialog and change MPASM's Node Properties to produce a MAP file named `tutor84.map`. After you've built the project, look at `tutor84.map` to see build information.

Grayed Out Menus – If you find menu's "grayed out," check to make sure that you haven't somehow entered the Editor Only mode. If you're sure everything is set up correctly, try exiting MPLAB and restarting the program.

Chapter 4. MPLAB Project Tutorial

4.1 Introduction

This chapter discusses in detail how to use projects in MPLAB. If you went through the previous tutorial, you may want to skip this chapter for now and return to it when you are ready to learn more about projects.

The project manager beginning with MPLAB v3.40 has been extended to support multiple files. Previously established projects from MPLAB v3.31 and earlier will be converted automatically by newer versions of MPLAB when they are opened. Converted projects cannot be reopened from previous versions of MPLAB.

4.2 Highlights

In this tutorial you will learn these functions of MPLAB Projects:

- Overview of MPLAB Projects
- Making a Project with One MPASM Source File
- Compiling a Single MPASM Source File Without Creating a Project
- Making a Project with Multiple MPASM Source Files with MPLINK
- Making a Project with MPLAB-C17
- Making a Project with Hi-Tech PIC C

To perform these tasks, you will use the following features of MPLAB:

- Install Language Tool
- New Project
- Add Nodes to a Project
- Set Project Node Properties
- Make/Build Project
- Project Window

4.3 Overview of MPLAB Projects

A Project in MPLAB is the group of files needed to build an application along with their associations to various build tools. A project is made up of a project node and one or more source nodes. The source nodes are typically MPASM source files, MPLAB-C17 source files, precompiled libraries and object files, and linker scripts. Usually the project is placed in the same directory as the main source files.

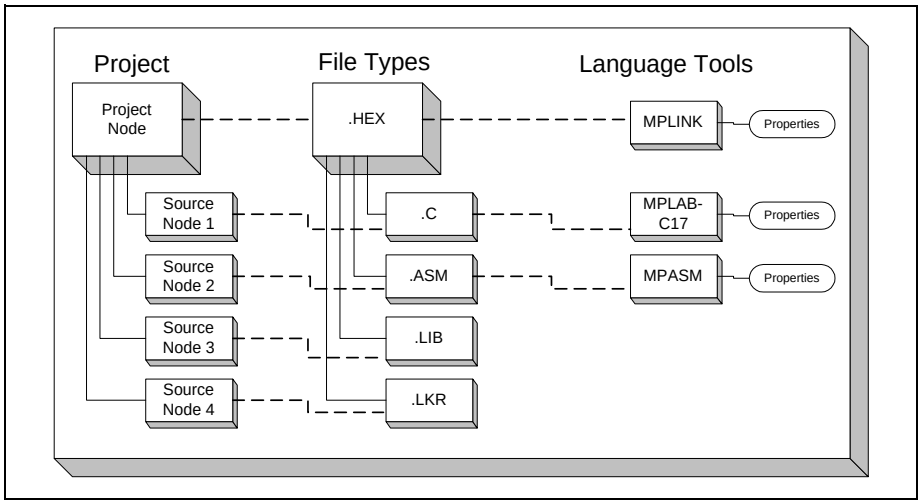


Figure 4.1: Project Relationships

Associated with each node in a project is a language tool. When you create a project, you must assign a language tool unless such an assignment would be redundant. For instance, if you are using MPASM with a single source file (first tutorial below), you won't have to assign a language tool to the source file node, since the project node is already set up for MPASM. Similarly, precompiled libraries, precompiled object files, and linker scripts for MPLAB-C17 or MPASM will not need to be assigned a tool since these will always be built with MPLINK.

There are multiple options for each language tool that are usually set via command line switches when invoked from DOS. These options show up as Node Properties for each editable node. Such switches control the generation of `.MAP` files, choose the proper `.HEX` format, and enable/disable warning messages, among other things. They can be set differently for the various source nodes on a project, although, typically, they will be the same. The Node Properties dialog corresponds to the command line switches for the various tools and, when first viewed, this dialog shows the default values. Refer to the MPASM and MPLAB-C17 User's Guides for information on these command line switches.

Built into MPLAB's Project Manager is a Make facility. This tool looks at the date/time stamp for the source files that go into an application and figures out which components have changed and need to be recompiled or assembled when the project is relinked.

4.4 Making a Project with One MPASM Source File

To make a project that has only one MPASM source file, or that uses the previous method of projects (MPLAB v3.31 or earlier), wherein a single source file would `#include` other files, follow these steps.

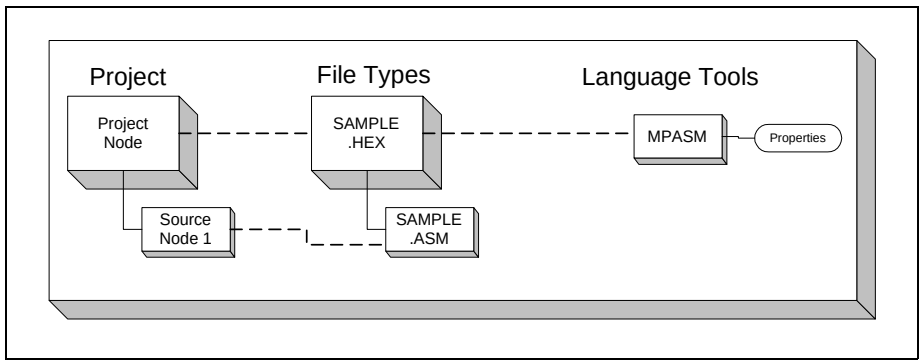


Figure 4.2: Project Relationships For One MPASM Source File Tutorial

4.4.1 Set Development Mode

Select the proper development mode for the application. For this tutorial set *Options>Development Mode* to MPLAB-SIM simulator and select the PIC16F84 PICmicro.

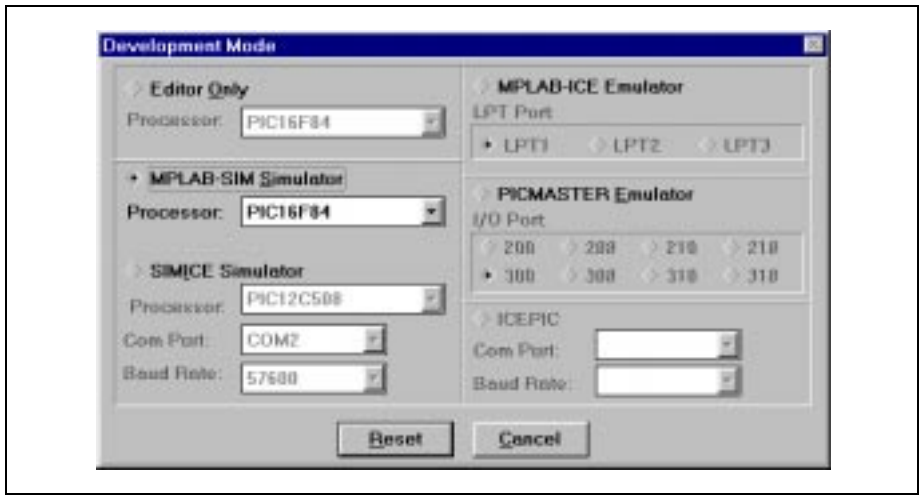


Figure 4.3: Development Mode

4.4.2 New Project

Select *Project>New Project*, select a directory for the new project, then type in its name. Use the MPLAB installation directory and name it SAMPLE.PJT for this tutorial.

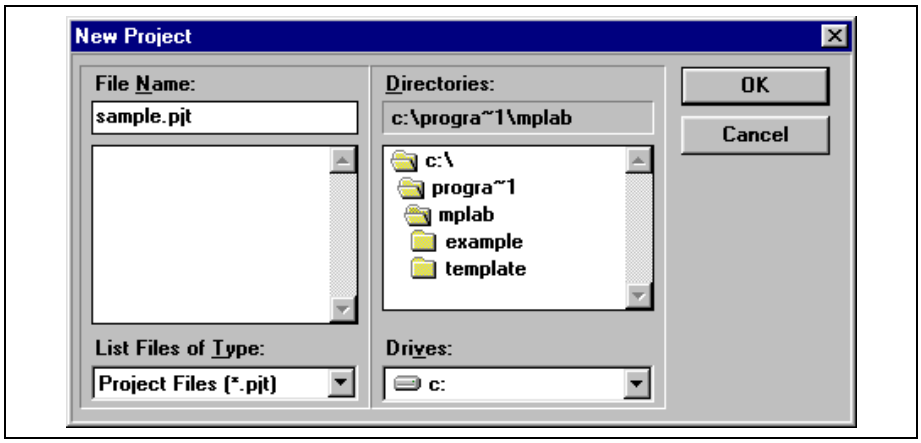


Figure 4.4: New Project – sample.pjt

4.4.3 Project Dialog

After clicking on **OK**, you will see the Edit Project Dialog:

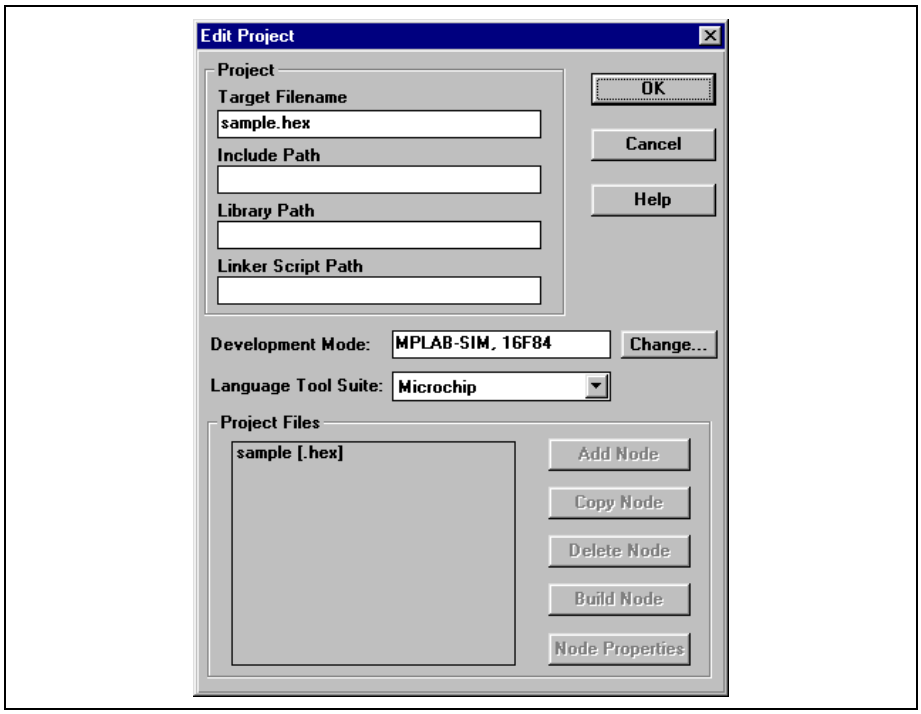


Figure 4.5: Edit Project Dialog

4.4.4 Set Node Properties

Click on the file name, SAMPLE.HEX, in the Project Files window, then select the **Node Properties** button.

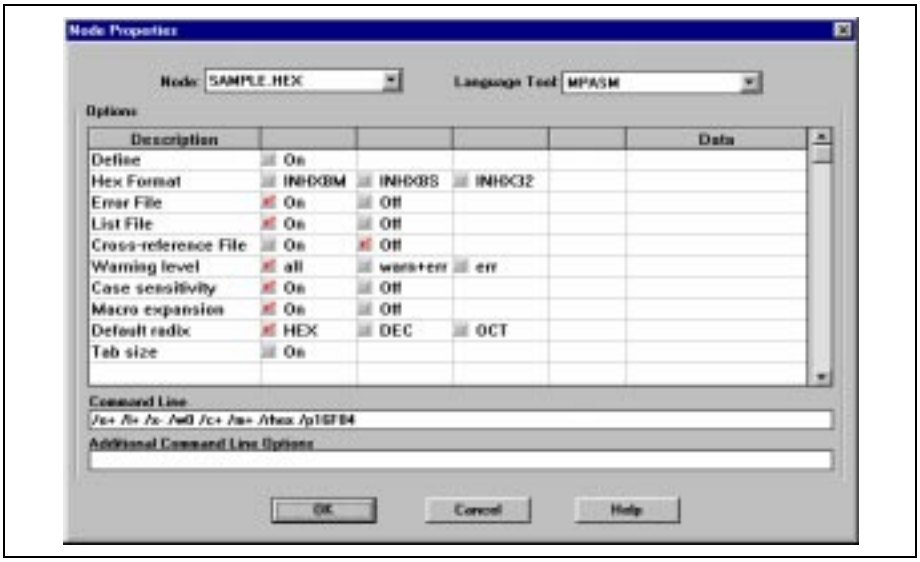


Figure 4.6: Node Properties Dialog

The Node Properties dialog shows the command line switches for the tool, in this case MPASM. When you first open this dialog, the checked boxes represent the default values for the tool. For this tutorial, these do not need to be changed. Refer to the *MPASM with MPLINK and MPLIB User's Guide* for more information on these command line switches.

Select **OK** to return to the Edit Project Dialog box.

4.4.5 Add Node

Select **Add Node** from the Edit Project Dialog. Use SAMPLE.ASM for this tutorial. This is the browse window that pops up from choosing **Add Node**.

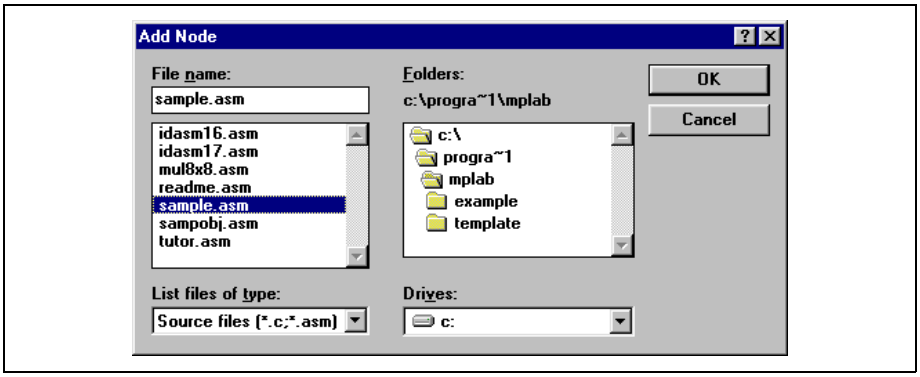


Figure 4.7: Add Node

MPASM always makes a .HEX file with the same name as the source .ASM file. The Project Manager will create a SAMPLE.HEX file when the project is built.

The Edit Project dialog should look like this:

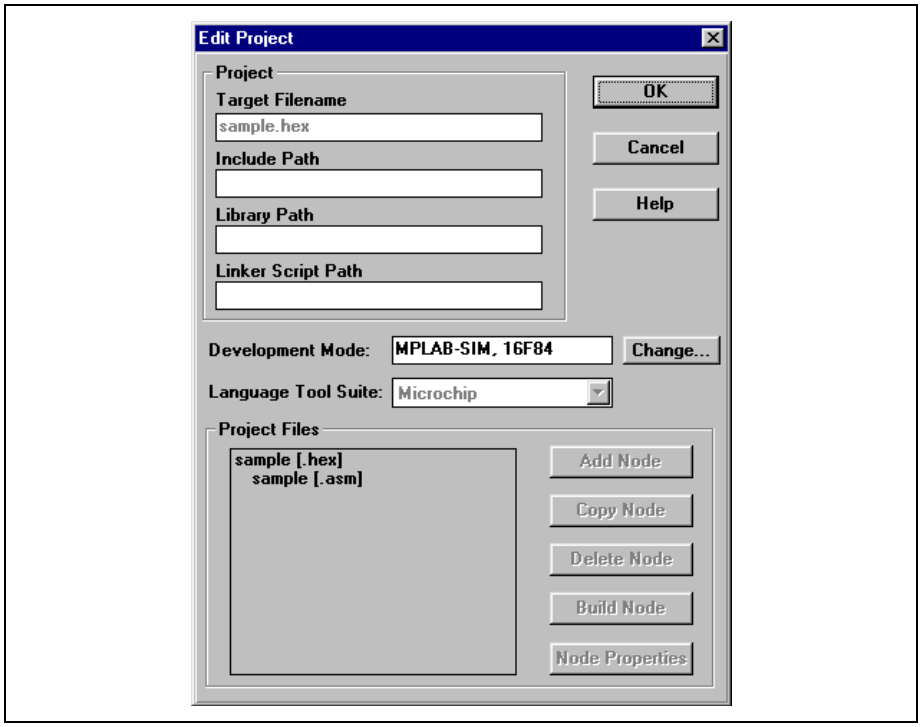


Figure 4.8: Edit Project Dialog with Node

In this simple example, no entries were made in the Path boxes. As your application becomes more complex, you may need to enter the directories of your Include Files in the appropriate box.

Press **OK** on the Edit Project Dialog.

4.4.6 Make Project

Select *Project>Make Project* from the menu to compile the application using MPASM. A Build Results window is created that shows the command line sent to the assembler. It should look like this:

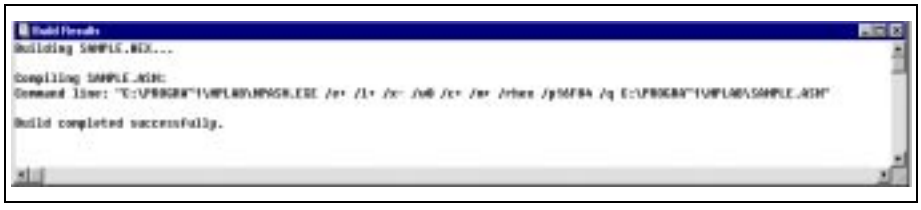


Figure 4.9: Build Results

4.4.7 Troubleshooting

If this did not work check these items:

Select *Project>Install Language Tool...* and check that MPASM is pointed to the MPASM.EXE in the MPLAB installation directory. Alternatively, MPASM can point to `MPASMWIN.EXE`, but the “Windowed” option should be selected.

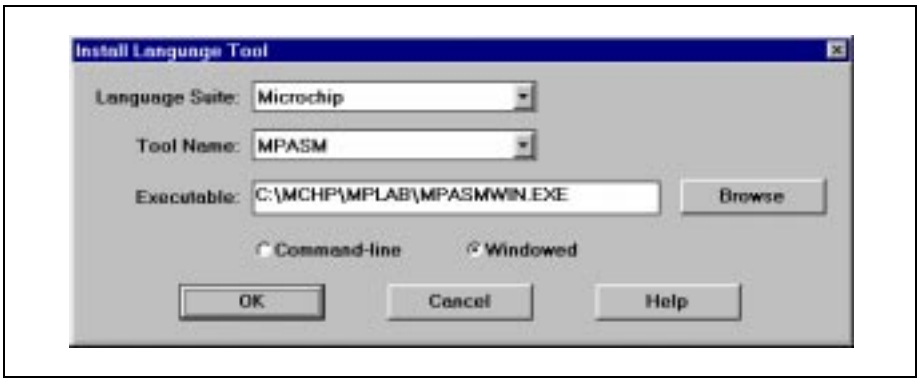


Figure 4.10: Install Language Tool Dialog

If you get a message from DOS saying that you have run out of environment space, use Windows Explorer to select the `MPASMWIN.EXE` file in the MPLAB installation directory, and click on the right mouse button to bring up the Properties dialog:

Figure 4.11: Properties Dialog

Increase the size of the Initial Environment. Usually a setting of 2048 will suffice, but if you have a lot of applications that set variables and add to your path statement in your AUTOEXEC.BAT file, you may need to make it larger.

4.4.8 Project Window

Open the *Window>Project* window to see that the target name is set properly to match the Node source name. They will have different file extensions, .ASM and .HEX, but both are named SAMPLE for this tutorial.

The Project window should look like this:

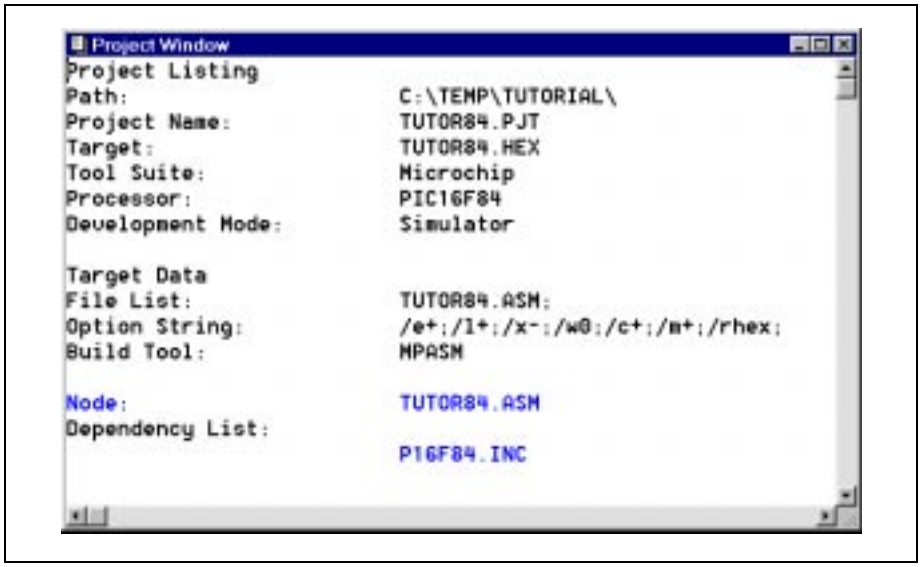


Figure 4.12: Project Window

4.4.9 Summary

Here is a quick list of the steps to set up a new project as described above:

- Create new project with *Project>NewProject*
- Set project Node Properties to MPASM and select the desired build options
- Add Source file node

4.5 Compiling a Single MPASM Source File Without Creating a Project

It is possible to compile a single file without opening up a project. The disadvantage of this method is that although no initial project setup is needed, it requires that you specify options every time you compile the file. This example will use the same assembly-language file used in the last example.

You must first close any open projects. To do this, select *Project>Close Project*.

4.5.1 Set Development Mode

Select the proper development mode for the application. For this tutorial, set *Options>Development Mode* to MPLAB-SIM simulator and select the PIC16F84 PICmicro.

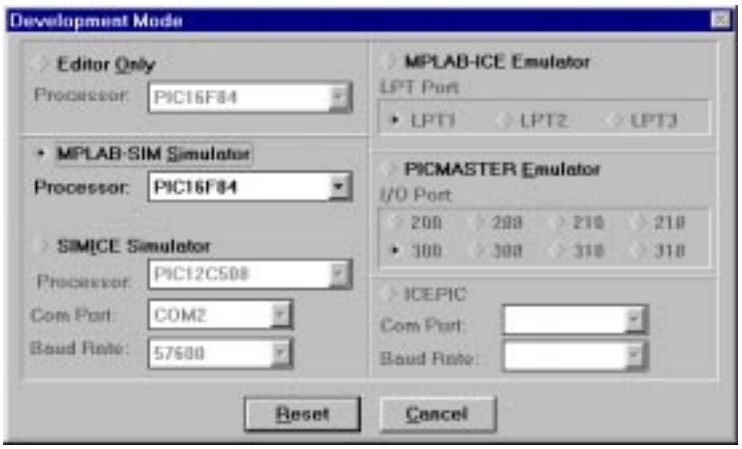


Figure 4.13: Development Mode

4.5.2 Open Source File

Open the source file that you wish to assemble. For this tutorial, use SAMPLE.ASM from the MPLAB installation directory.

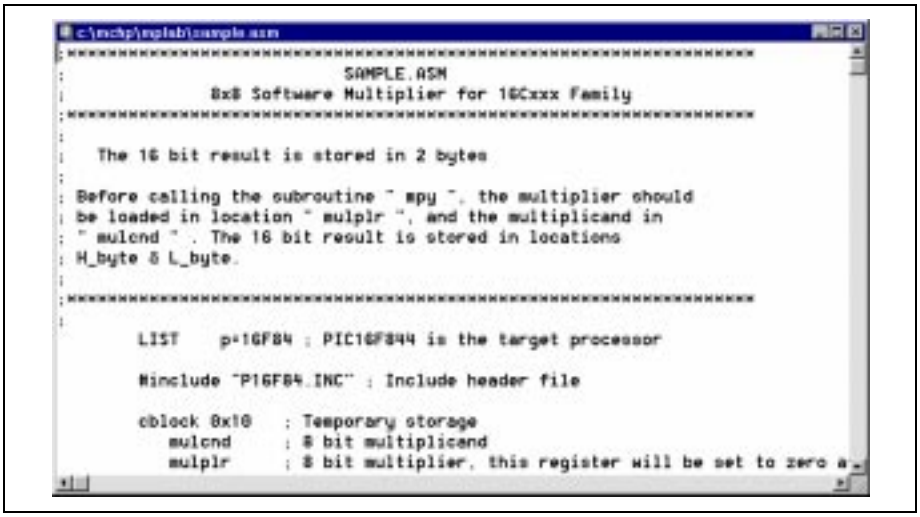


Figure 4.14: Source File

4.5.3 Compile Source File

Select *Project>Build Node* from the menu to compile SAMPLE.ASM using MPASM. MPLAB brings up an Invoke Build Tool Dialog that looks like this:

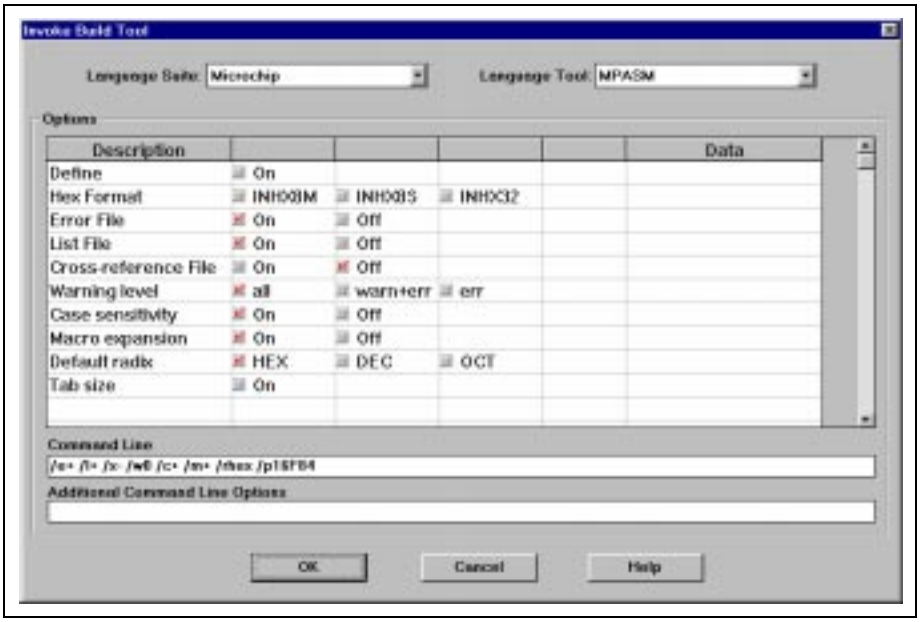


Figure 4.15: Build Tool Dialog

Verify that MPASM is selected, and set the tool options to match those shown above. Press **OK** in the Invoke Build Tool Dialog to start the build process. A Build Results window is generated that shows the command line sent to the assembler and the build output. It should look like this:

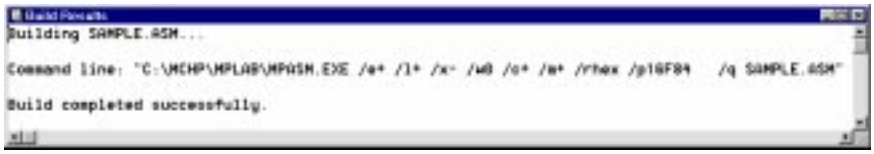


Figure 4.16: Build Results

4.5.4 Troubleshooting

If this did not work check these items:

Select *Project>Install Language Tool...*. Check that MPASM is pointing to `MPASMWIN.EXE` in the MPLAB installation directory and the “Windowed” option is selected. Alternatively, MPASM can point to `MPASMWIN.EXE`, and the “Command Line” option should be selected.

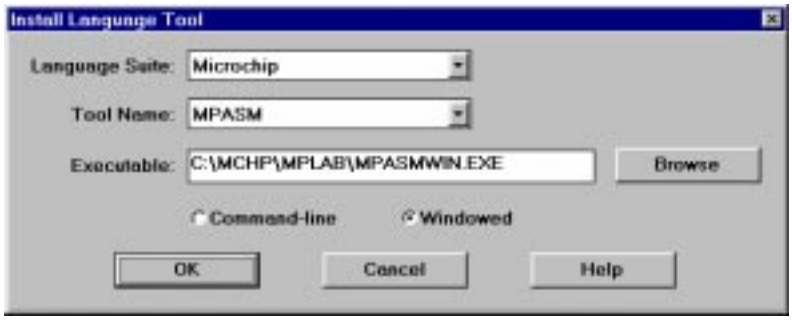


Figure 4.17: Install Language Tool

If you get a message from DOS saying that you have run out of environment space, use Windows Explorer to select the `MPASMWIN.EXE` file in the MPLAB installation directory, and click on the right mouse button to bring up the Properties dialog:

Figure 4.18: MPASM.EXE Properties

Increase the size of the Initial Environment. Usually a setting of 2048 will suffice, but if you have a lot of applications that set variables and add to your path statement in your AUTOEXEC.BAT file, you may need to make it larger.

4.5.5 Summary

Here is a quick list of the steps to set up a new project as described above:

- If a project is open, close it using *Project>Close Project*
- Open the source file you wish to compile
- Select *Project>Build Node*
- Select the desired language suite, build tool, and build options in the Invoke Build Tool Dialog

4.6 Making a Project with Multiple MPASM Source Files with MPLINK

To use MPLINK to link two or more MPASM object files, follow these steps. If you followed through the previous section, select *Project>Close Project*.

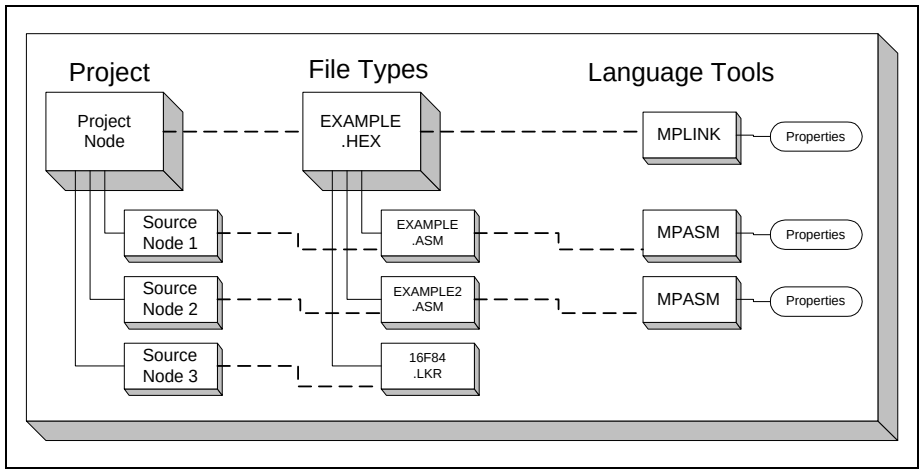


Figure 4.19: Project Relationships For Multiple MPASM Source File Tutorial

4.6.1 Set Development Mode

Set *Options>Development Mode* to MPLAB-SIM simulator and select the PIC16F84 PICmicro for this example.

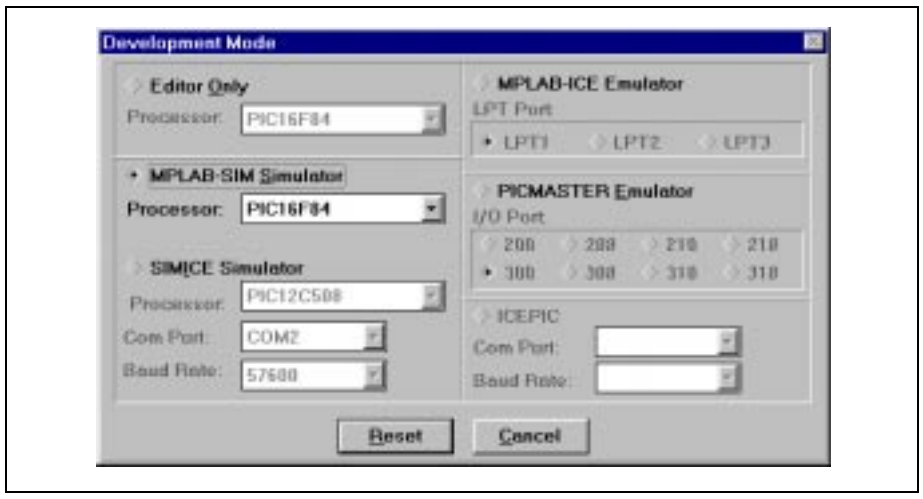


Figure 4.20: Development Mode

4.6.2 New Project

Select *Project>New Project*, browse to select a directory for a new project, then type in its name. Use the \MPLAB\EXAMPLE directory for this tutorial and name it EXAMPLE.PJT.

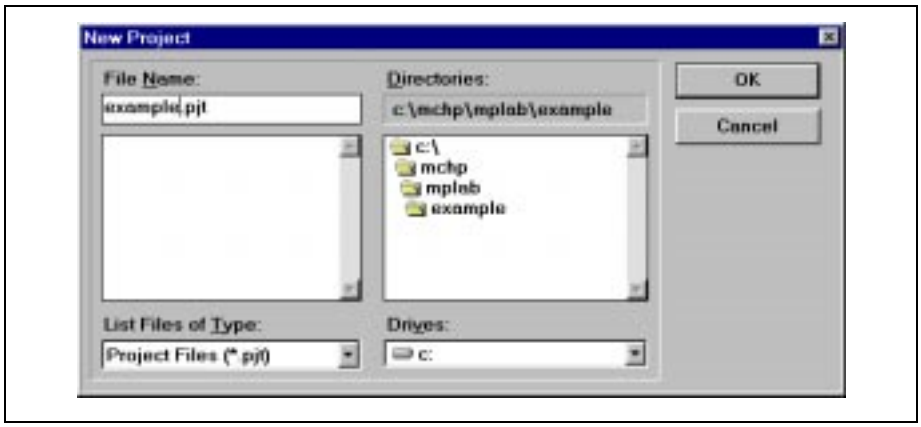


Figure 4.21: New Project – example.pjt

4.6.3 Set Node Properties

Select the name of the project in the Project Files dialog of the Edit Project Dialog and press **Node Properties** to bring up this dialog. Set the language tool to MPLINK.

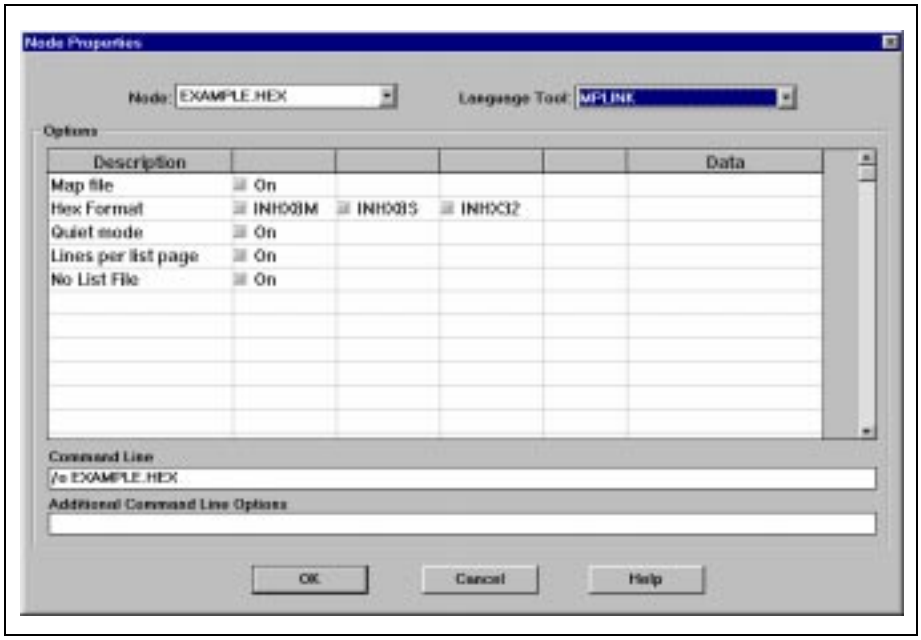


Figure 4.22: Set Node Properties

The Node Properties dialog shows the command line switches for the tool, in this case MPLINK. When you first open this dialog, the checked boxes represent the default values for the tool. For this tutorial, these do not need to be changed. Refer to the *MPASM with MPLINK and MPLIB User's Guide* for more information on these command line switches.

Select **OK** to return to the Edit Project Dialog box.

4.6.4 Add First Source File Node

Select **Add Node** from the Edit Project Dialog. Use `EXAMPLE.ASM` in the `\MPLAB\EXAMPLE` directory for this tutorial.

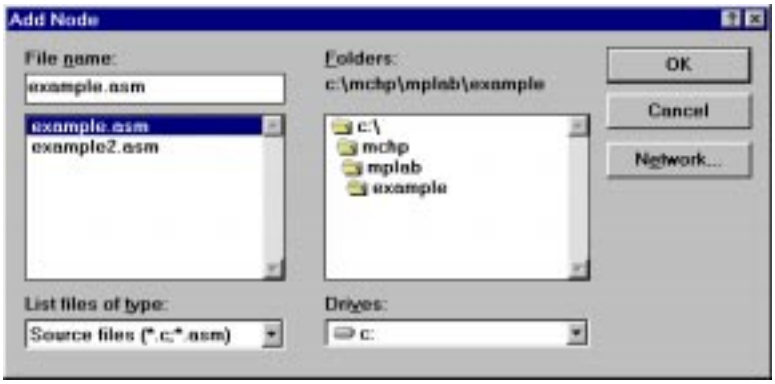


Figure 4.23: Add Node

You may select more than one file at a time from this dialog using the standard Windows methods of selecting one file and then selecting another while holding down the control key (selects only those two files) or the shift key (selects both files and all those in between).

Select `example.asm` from the list of project files in the Edit Project dialog, and press **Node Properties**.

Verify that the language tool is set to MPASM.

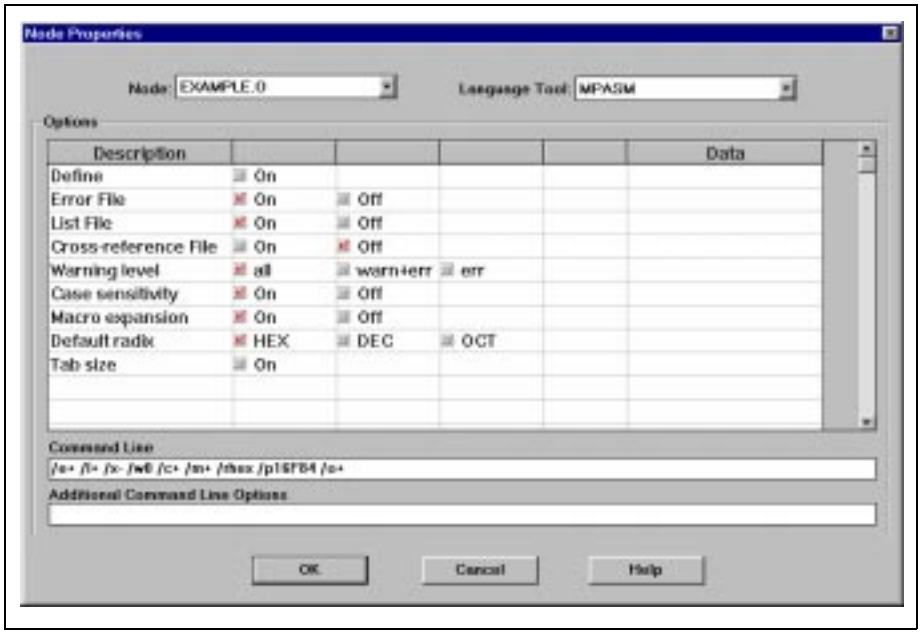


Figure 4.24: Node Properties – Example.o

The Node Properties dialog shows the command line switches for the tool, in this case MPASM. Refer to the *MPASM with MPLINK and MPLIB User's Guide* for more information on these command line switches.

Select **OK** to return to the Edit Project Dialog box.

4.6.5 Adding Additional Source Files

Follow the previous two steps to add the rest of the source files to the project. For this tutorial, select `EXAMPLE2.ASM` from the `\MPLAB\EXAMPLE` directory. You can also use **Copy Node** to enter subsequent files with the same Node Options as the first source file. Make sure the Node Options are set properly on each file.

To use the Copy Node feature**,** select one of the source node files listed in the Project Files box. Then select **Copy Node**. In the Add Node dialog box, select one or more source files. Once the files are selected chose the **OK** button. This step will set up the node properties for this selected files the same as the referenced node used. This is especially useful for adding multiple source files with identical node properties.

4.6.6 Select Linker Script

Select a linker script using the **Add Node** button and the method described above. A linker script is a file that MPLINK uses to define the memory architecture of each PICmicro. Standard linker scripts come with MPLINK and are in the MPLAB installation directory. For this tutorial select `PIC16F84.LKR` from the `\MPLAB\EXAMPLE` directory. Node options can not be set for a linker script.

Figure 4.25: Edit Project Window – Linker script

Press **OK** on the Edit Project Dialog.

In this simple example, no entries were made in the three “Path” boxes. As your application becomes more complex, you may need to enter the directories of your Include Files, Library Files and Linker Script Files in the appropriate box if they are not in the same directory as the project.

4.6.7 Make Project

Select *Project>Make Project* from the menu to compile the application using MPASM and MPLINK. A Build Results window is created that shows the command lines sent to each tool. It should look like this:

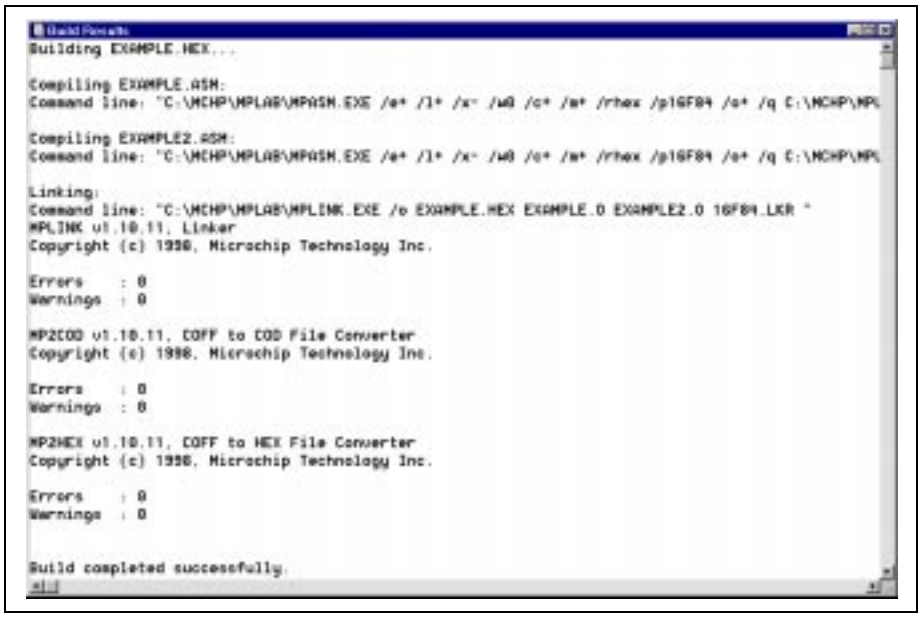


Figure 4.26: Build Results

4.6.8 Troubleshooting

If this did not work, check these items:

If MPLAB reports the message, “Time-out,” select **OK** to continue. Depending on the speed of your PC and the size of your project, you may wish to configure the length of time MPLAB will wait before reporting a timeout message. This value is set by selecting *Options>Environment Setup...* and adjusting the timeout settings in that dialog box.

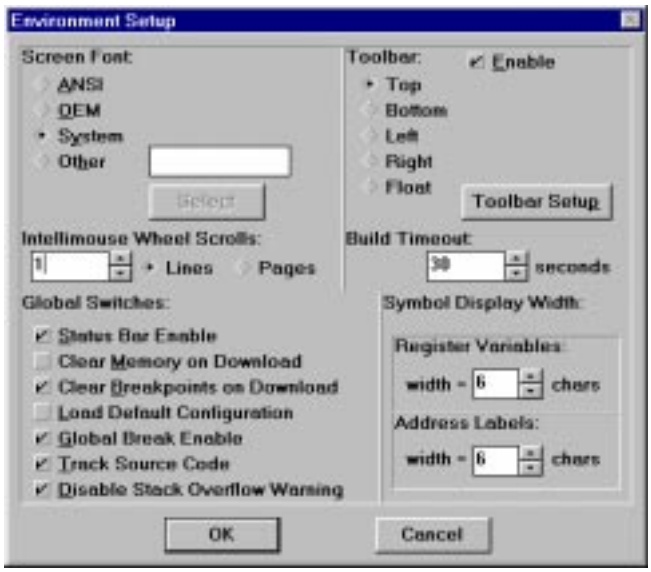


Figure 4.27: Environment Setup

Select *Project>Install Language Tool...*. Check that MPASM is pointing to `MPASMWIN.EXE` in the MPLAB installation directory and the “Windowed” option is selected. Alternatively, MPASM can point to `MPASMWIN.EXE`, and the “Command Line” option should be selected.

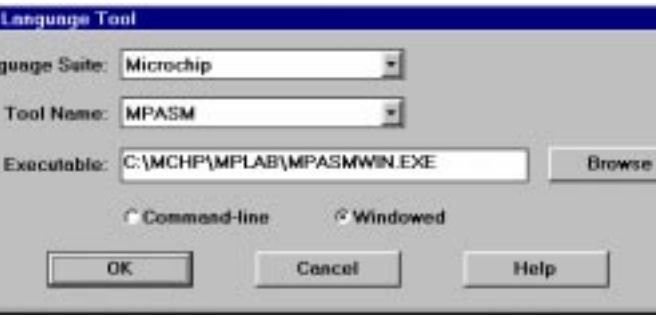


Figure 4.28: Install Language Tool – MPASM

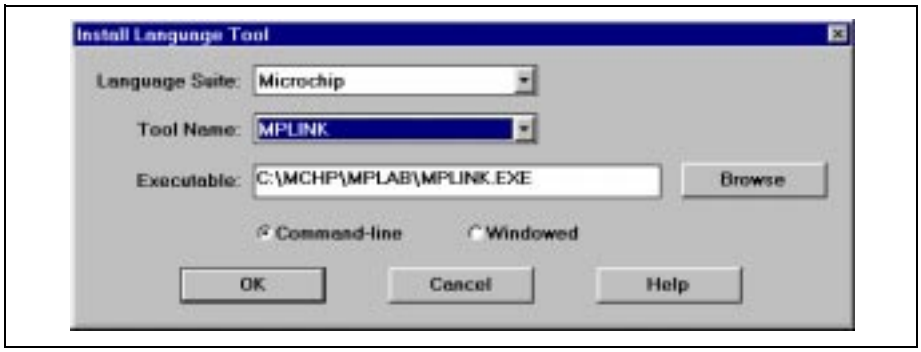


Figure 4.29: Install Language Tool – MPLINK

If you get a message from DOS saying that you have run out of environment space, use Windows Explorer to select the MPASMWIN.EXE file in the MPLAB installation directory, and click on the right mouse button to bring up the Properties dialog:

Figure 4.30: MPASM Properties

Increase the size of the Initial Environment. Usually a setting of 2048 will suffice, but if you have a lot of applications that set variables and add to your path statement in your AUTOEXEC.BAT file, you may need to make it larger.

4.6.9 Project Window

Open the *Window>Project* window. It should look like this:

Figure 4.31: Project Window

4.6.10 Summary

Here is a quick list of the steps to set up a new project as described above:

- Create new project with *Project>NewProject*.
- Set project Node Properties to MPLINK.
- Add Source file nodes, and set node properties as needed.
- Add Linker Script file node.

4.7 Making a Project with MPLAB-C17

This tutorial will show you how to use MPLAB-C17 with projects in MPLAB to build applications. If you have followed along the previous tutorials, select *Project>Close Project*.

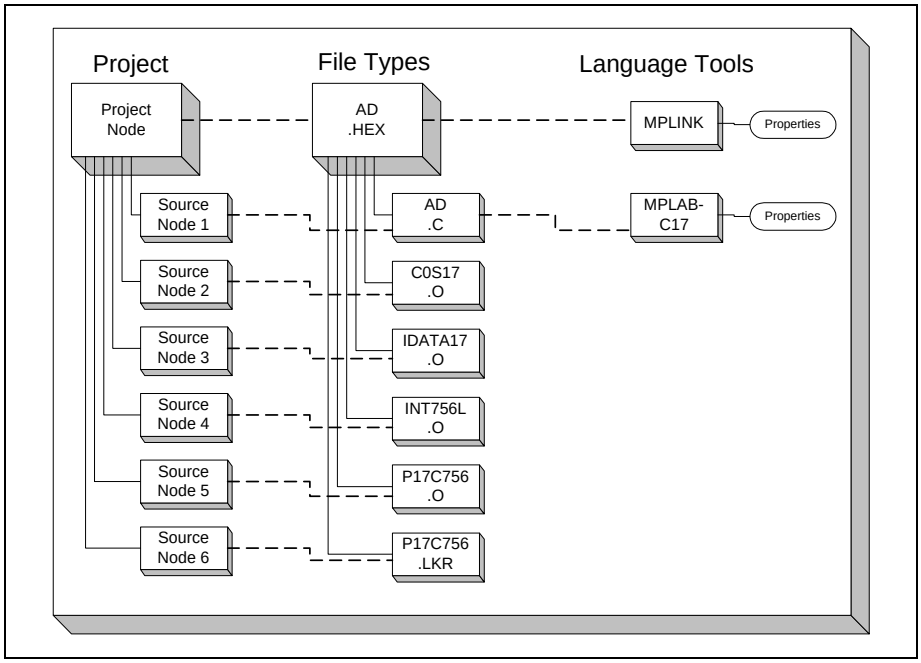


Figure 4.32: Project Relationships for MPLAB-C17 Tutorial

4.7.1 Set Development Mode

Set *Options>Development Mode* to MPLAB-SIM simulator and select the 17C756 PICmicro for this example.

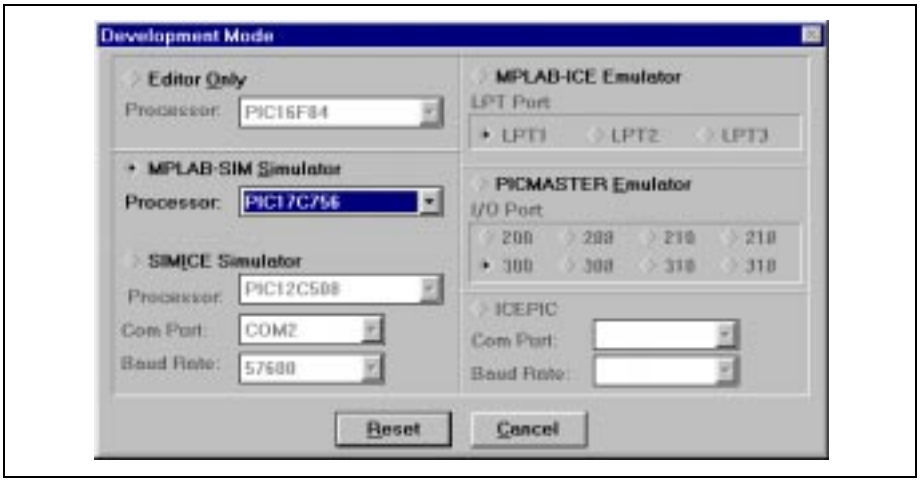


Figure 4.33: Development Mode – PIC17C756

4.7.2 Install MPLAB-C17 Language Tool

Make certain that MPLAB-C17 is installed correctly in MPLAB. The *Project>Install Language Tool* dialog should look like this (your executable path may be different):

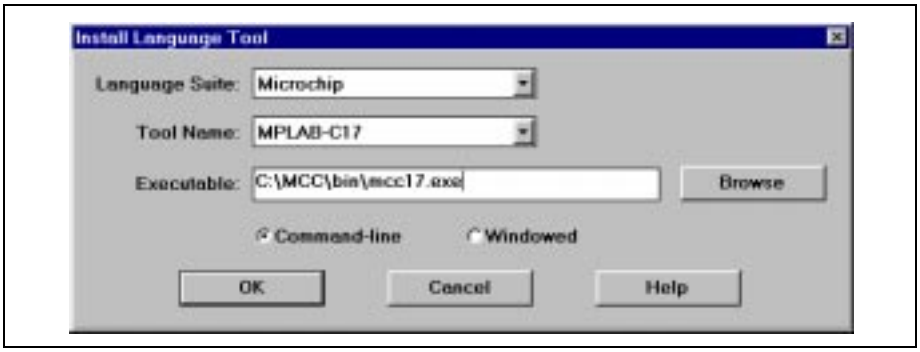


Figure 4.34: Install Language Tool – MPLAB-C17

If the executable is not shown in the window, use the Browse button to point to `MCC17.EXE` on your system.

4.7.3 New Project

Select *Project>New Project* and select a directory for a new project, then type in its name. Name it `AD.PJT` in the `\MCC\EXAMPLES\AD` directory.

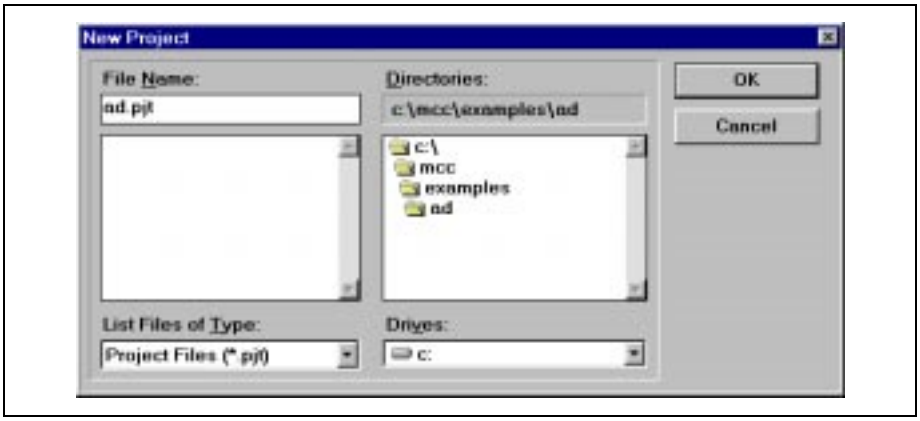


Figure 4.35: New Project – ad.pjt

After setting the project name, press **OK** and the Edit Project dialog will be shown.

Figure 4.36: Edit Project – ad.pjt

4.7.4 Set Node Properties

Select the name of the project in the Project Files dialog of the Edit Project Dialog and press **Node Properties**. Set the language tool to MPLINK.

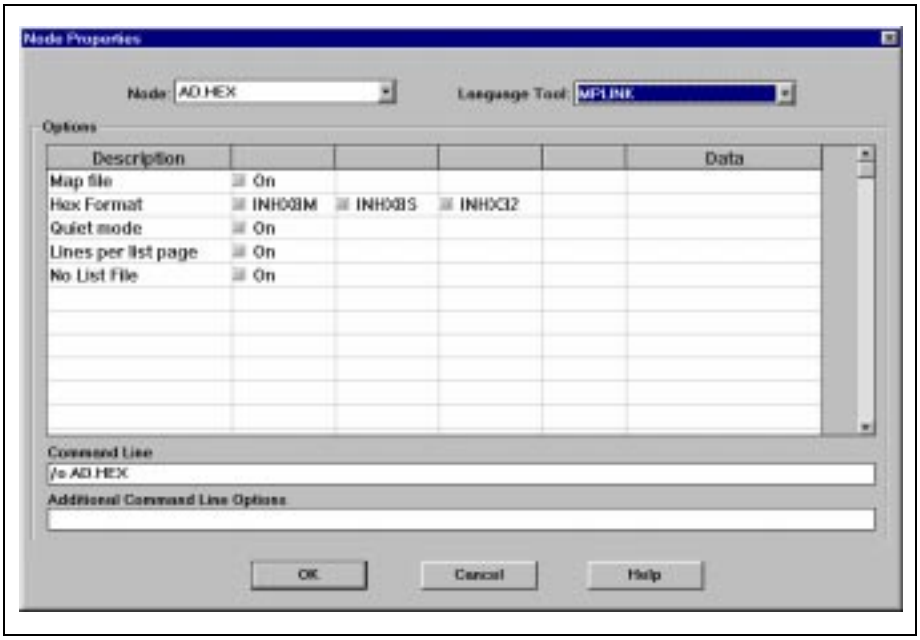


Figure 4.37: Node Properties – ad.hex

The Node Properties dialog shows the command line switches for the tool, in this case MPLINK. When you first open this dialog, the checked boxes represent the default values for the tool. For this tutorial, these do not need to be changed. Refer to the *MPASM with MPLINK and MPLIB User's Guide* for more information on these command line switches.

4.7.5 Add First Source File

To determine which nodes to set up from this tutorial, look at AD.BAT. This is the batch file that can be used to compile this example from a DOS command line and is in the \MCC\EXAMPLES\AD directory. Use this data to add all required nodes. Here is a listing of the batch file:

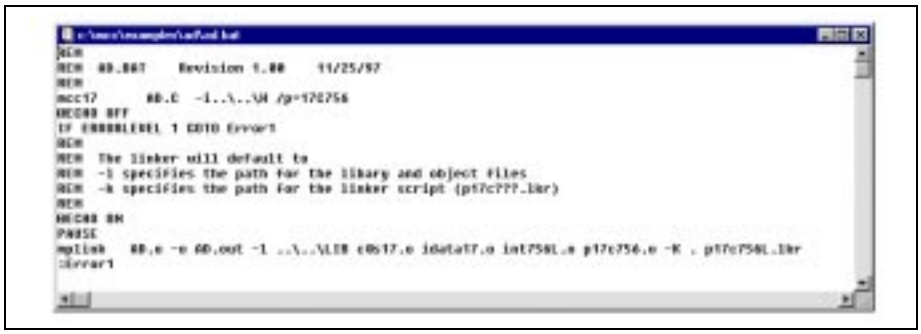


Figure 4.38: Source Listing – ad.bat

The nodes required are AD.C – the main source file which must be compiled, the linker script P17C756L.LKR, and the following object files:

C0S17.O – Start Up Code

IDATA17.O – Code to Initialize Data

INT756L.O – Interrupt Service Routines

P17C756.O – PIC17C756 Register Definitions

You can return to setting up the project from the *Project>Edit Project* menu selection.

Select **Add Node** from the Edit Project Dialog. Add the source file, AD.C from the \MCC\EXAMPLES\AD directory.

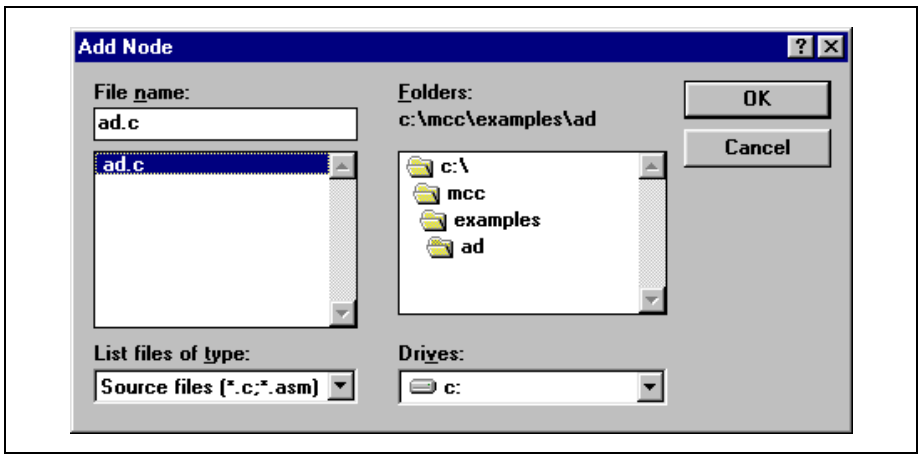


Figure 4.39: Add Node – ad.c

Select the file name in the Edit Project dialog and press Node Properties. Verify that the language tool is set to MPLAB-C17.

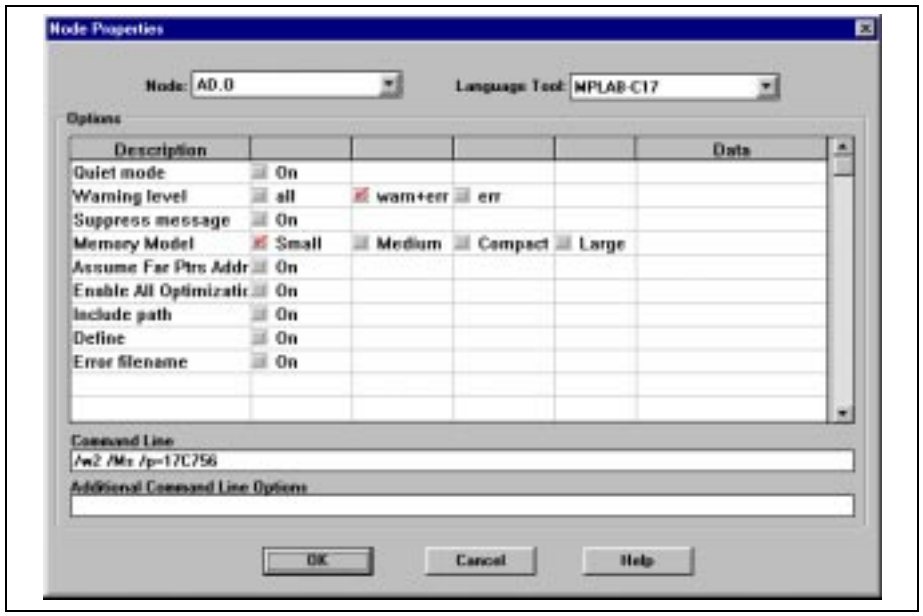


Figure 4.40: Node Properties – ad.o

The “Object filename” is set to AD.O automatically. Nothing else needs to be changed in this dialog.

The Node Properties dialog shows the command line switches for the tool, in this case MPLAB-C17. The checked boxes represent the default values for the tool. For this tutorial, none of the setting need to be changed from their default values. Refer to the *MPLAB-C17 User’s Guide* for more information on these command line switches.

4.7.6 Adding Precompiled Object Files

Use the **Add Node** button from the Edit Project dialog to add the precompiled object files from the MPLAB-C17 library in `\MCC\LIB`. Add all of the required object files: `C0S17.O`, `IDATA17.O`, `INT756L.O`, and `P17C756.O`. To select more than one file at a time, hold down the **Ctrl** key on your keyboard while selecting the files with your mouse. Options can not be set on precompiled object files.

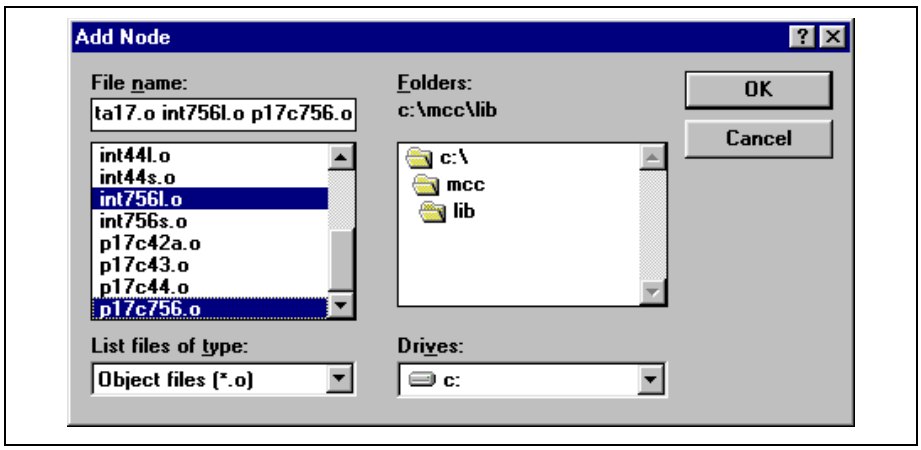


Figure 4.41: Add Node – object files

In the event you need to add a `xxx.lib` (library) file, follow the same procedure as shown above.

4.7.7 Select Linker Script

Select a linker script and add it as a node. Use the linker script `P17C756L.LKR` in the `\MCC\EXAMPLES\AD` directory. Options can not be set on a linker script.

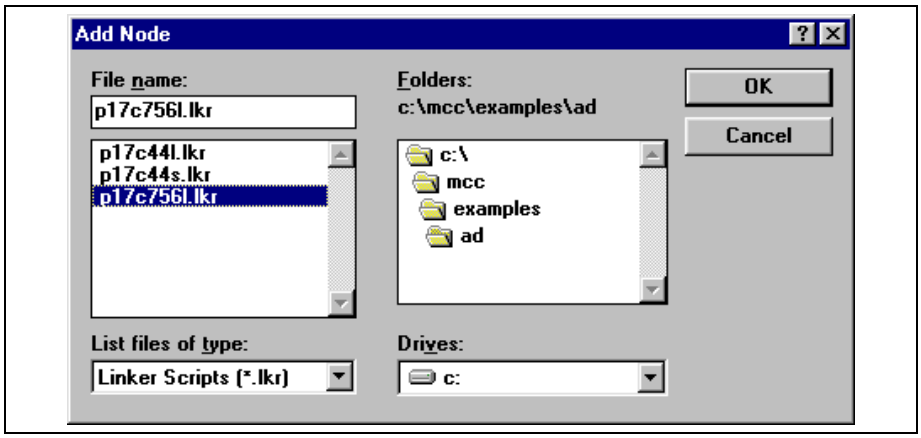


Figure 4.42: Add Node – p17c756.lkr

The Edit Project window should now look like this:

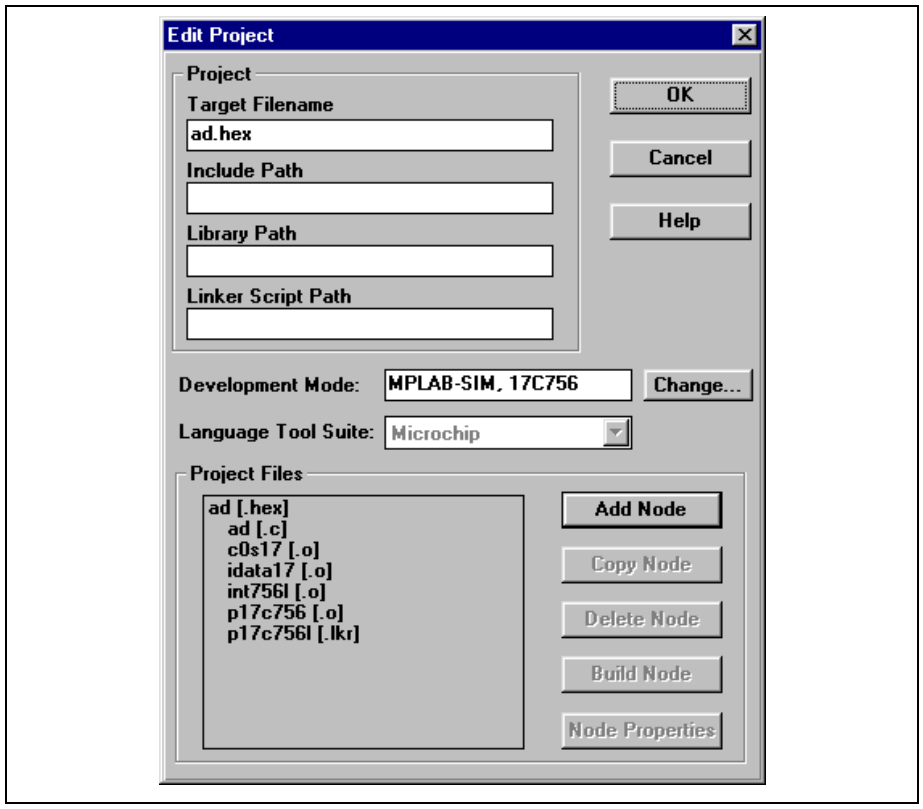


Figure 4.43: Edit Project – ad.hex

Press **OK** on the Edit Project Dialog.

4.7.8 Make Project

Select *Project>Make Project* from the menu to compile the application using MPLAB-C17 and MPLINK. A Build Results window is created that shows the command lines sent to each tool. It should look like this:

Figure 4.44: Build Results – ad.hex

4.7.9 Troubleshooting

If this did not work, check these items:

Select *Project>Install Language Tool...* and check that MPLAB-C17 and MPLINK are pointed to the `MCC17.EXE` and `MPLINK.EXE` executables.

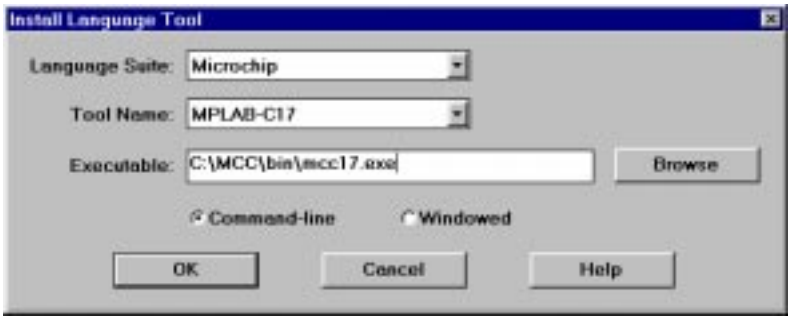


Figure 4.45: Install Language Tool – MPLAB-C17

4.7.10 Project Window

Open the *Window>Project* window. It should look like this:

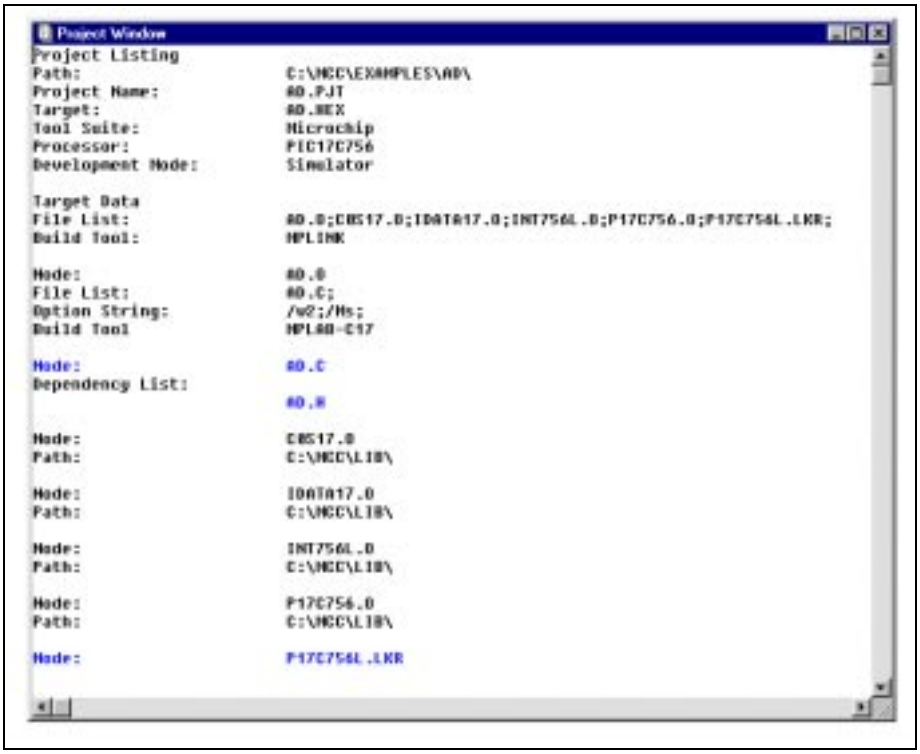


Figure 4.46: Project Window – ad.pjt

4.7.11 Summary

Here is a quick list of the steps to set up a new project as described above:

- Create new project with *Project>NewProject*
- Set project Node Properties to MPLINK
- Add Source files, setting language tool to MPLAB-C17 or MPASM
- Add Precompiled Nodes (`.O` files and `.LIB` files)
- Add Linker Script node

4.8 Making a Project with Hi-Tech PIC C

This tutorial will show you how to use Hi-Tech's PIC C compiler with projects in MPLAB to build applications. If you have followed along the previous tutorials, select *Project>Close Project*.

4.8.1 Set Development Mode

Set *Options>Development Mode* to MPLAB-SIM simulator and select the 16C77 PICmicro for this example.

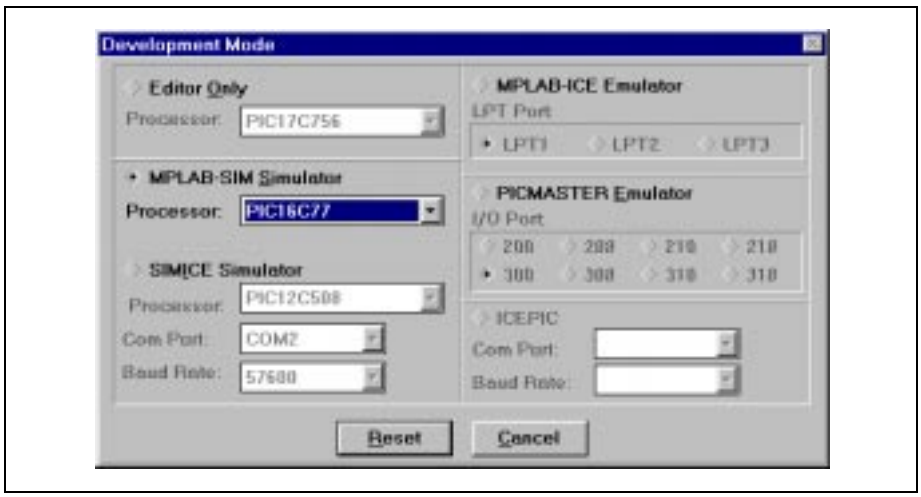


Figure 4.47: Development Mode – PIC16C77

4.8.2 Install PIC C Language Tools

Make certain that PIC C is installed correctly in MPLAB. The *Project>Install Language Tool* dialog should be set to something similar to this for the HI-TECH tools (your executable path may be different):

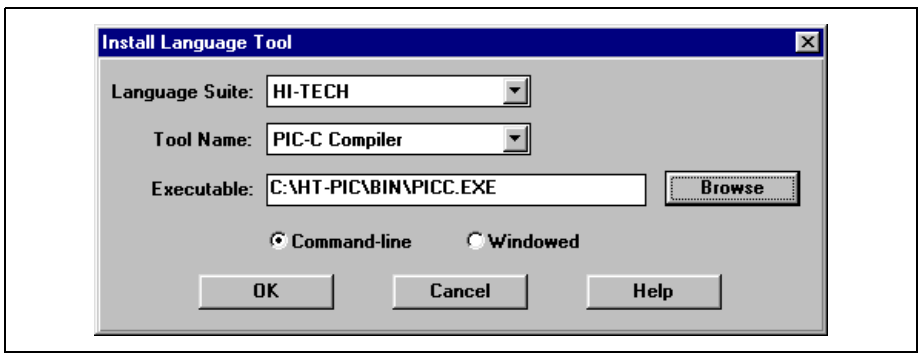


Figure 4.48: Install Language Tool – PIC C Compiler

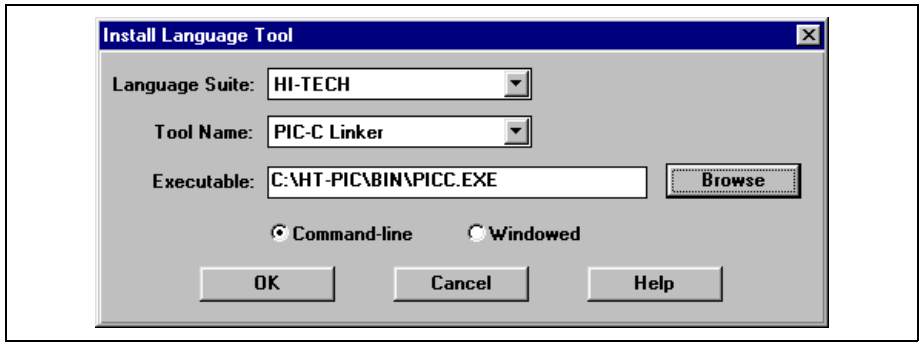


Figure 4.49: Install Language Tool – PIC C Linker

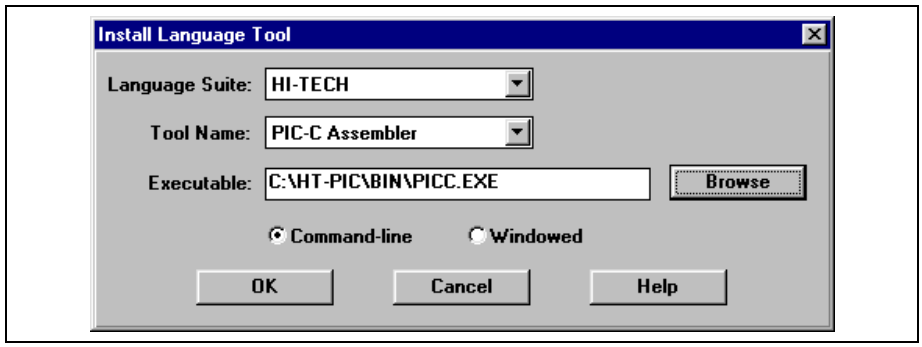


Figure 4.50: Install Language Tool – PIC C Assembler

Use the **Browse** button to point to `PICC.EXE` on your system for the compiler, linker, and assembler.

4.8.3 New Project

Select *Project>New Project* and select a directory for a new project, then type in its name. Name it `SAMPLE.PJT` in the `\HT-PIC\SAMPLES` directory.

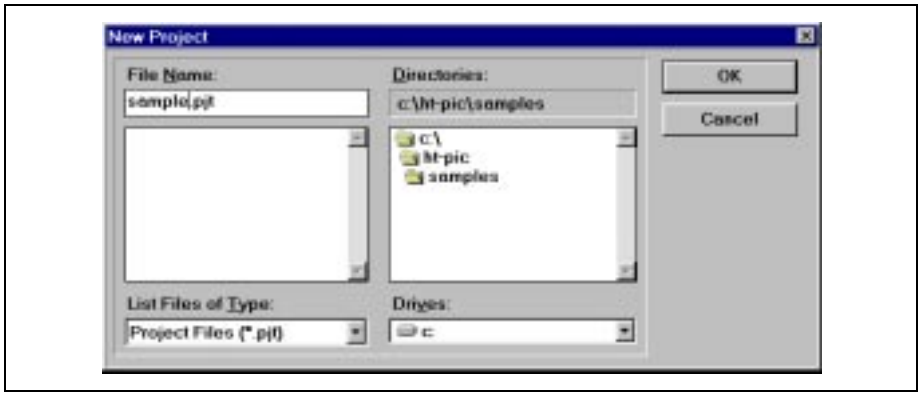


Figure 4.51: New Project – sample.pjt

After setting the project name, press **OK** and the Edit Project dialog will be shown.

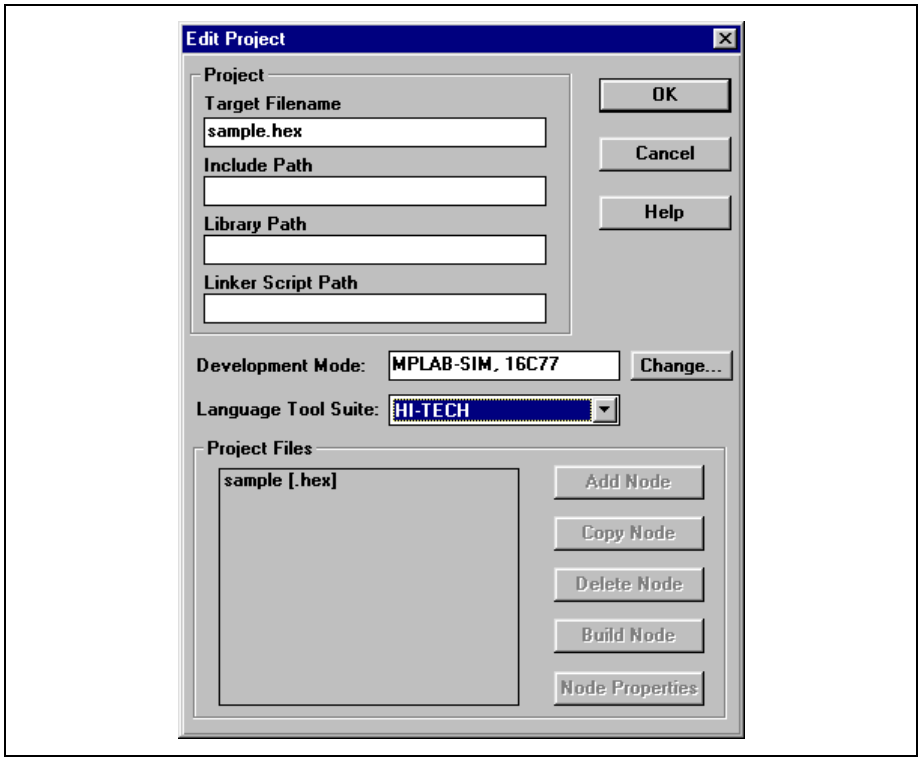


Figure 4.52: Edit Project – sample.hex

Make sure to set the Language Tool Suite to HI-TECH.

4.8.4 Set Node Properties

Select the name of the project in the Project Files dialog of the Edit Project Dialog and press **Node Properties**. Set the language tool to “PIC-C Linker” and check the Generate Debug Info check box. Type in “Microchip” in the Data column as shown below:

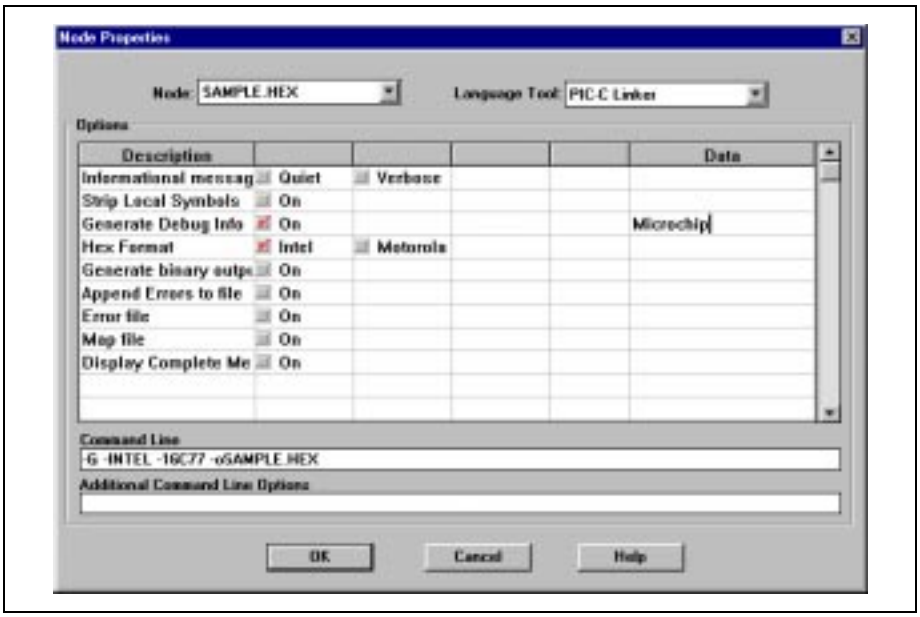


Figure 4.53: Node Properties – sample.hex

The Node Properties dialog shows the command line switches for the tool, in this case PIC C. When you first open this dialog, the checked boxes represent the default values for the tool. For this tutorial, only the debug info setting needs to be changed. Refer to the Hi-Tech documentation for more information on these command line switches.

Select **OK** in the Node Properties Dialog to return to the Edit Project Dialog.

4.8.5 Add First Source File

Select **Add Node** from the Edit Project Dialog. Add the source file, SAMPLE.C from the \HT-PIC\SAMPLES directory.

When the file name is shown and selected in the Add Node dialog, press **Node Properties**.

Set up this dialog this way:

- Set the Language Tool to PIC C Compiler
- Check the Generate Debug Info box
- Enter "Microchip" in the Generate Debug Info data column.

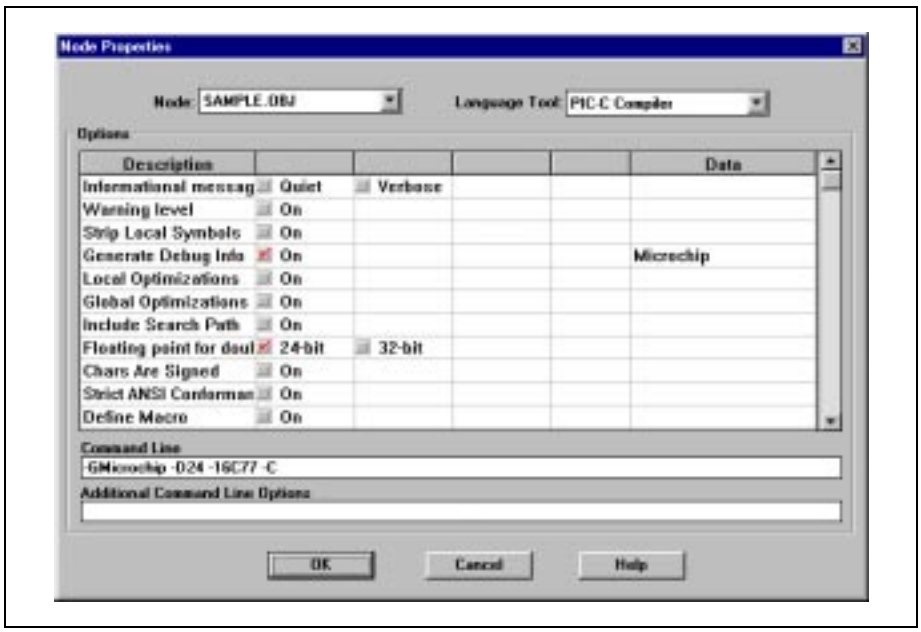


Figure 4.54: Node Properties – sample.obj

The Object filename is set to SAMPLE.OBJ automatically.

The Node Properties dialog shows the command line switches for the tool, in this case PIC C. The checked boxes represent the default values for the tool. For this tutorial, only the debug info setting needs to be changed. Refer to the Hi-Tech PIC C documentation for more information on these command line switches.

Press **OK**, select the SAMPLE.C node and use the **Copy Node** button to add ADC.C, DELAY.C, and LCD.C with the same Node Properties as SAMPLE.C. When you are finished, the project should look like this:

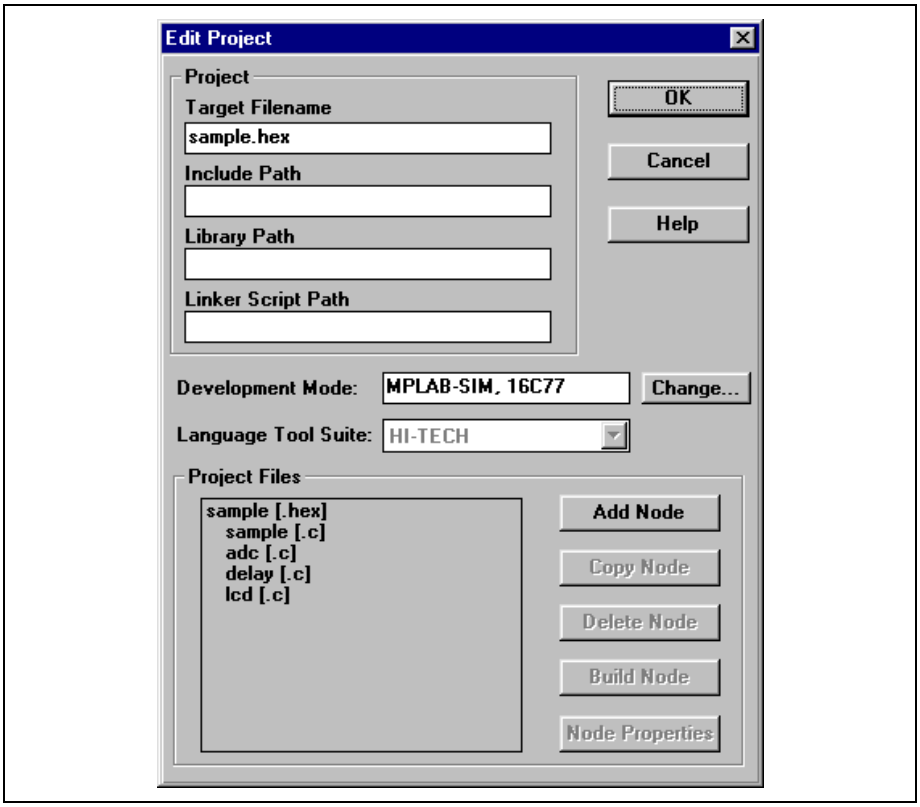


Figure 4.55: Edit Project – sample.hex

4.8.6 Make Project

Select *Project>Make Project* from the menu to compile the application using the Hi-Tech compiler and linker. A Build Results window is created that shows the command lines sent to each tool. It should look like this:

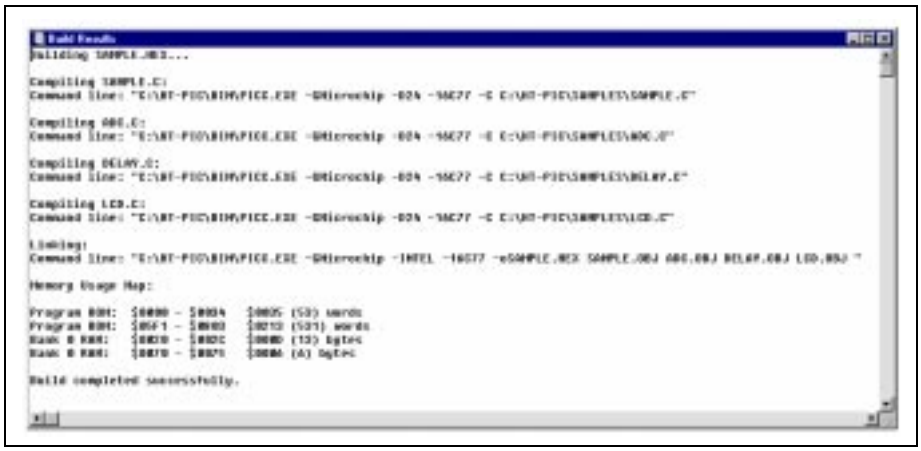


Figure 4.56: Build Results – sample.hex

4.8.7 Troubleshooting

If this did not work, check these items:

Select *Project>Install Language Tool...* and check that PIC C Compiler and PIC C Linker are both pointing to the `PICC.EXE` executable.

4.8.8 Project Window

Open the *Window>Project* window. It should look like this:

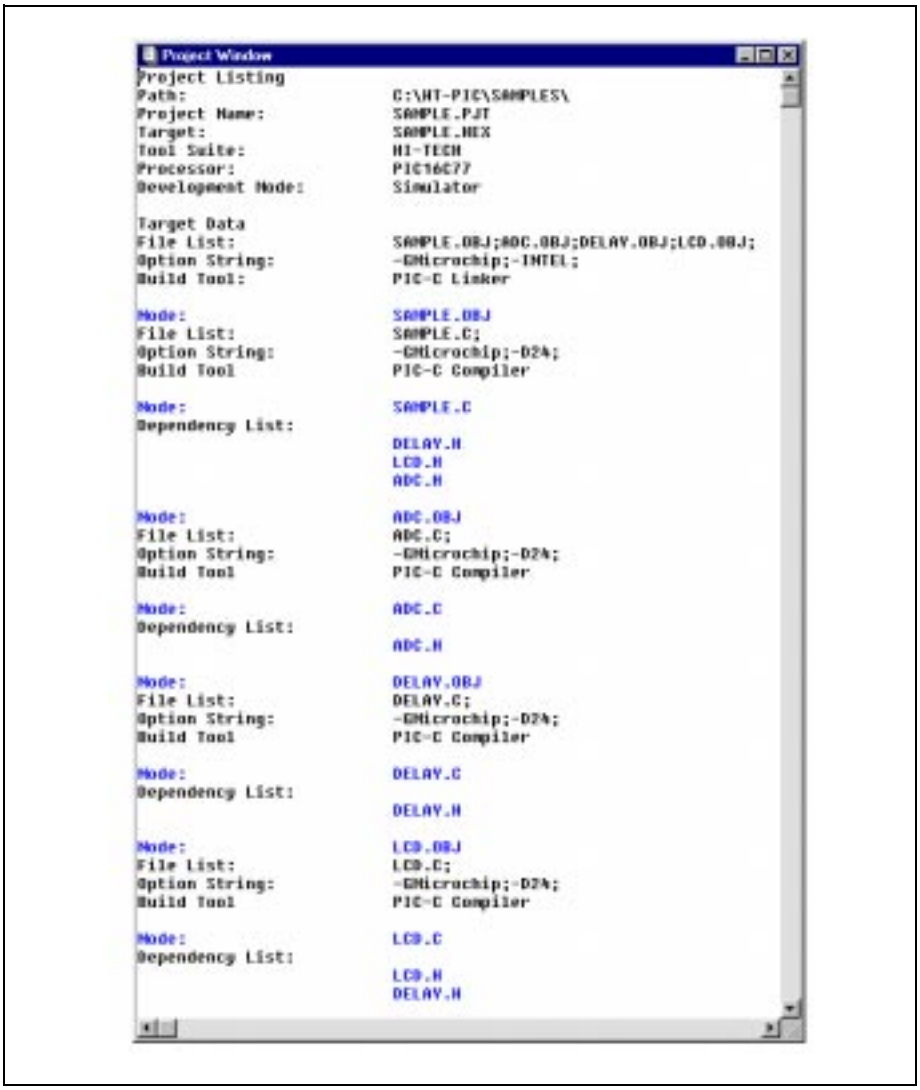


Figure 4.57: Project Window – sample.pjt

4.8.9 Summary

Here is a quick list of the steps to set up a new project as described above:

- Set up Language Tools for PIC C Compiler, Linker, and Assembler
- You may need to set the Include File Directory to \HT-PIC\H (or where PIC C include files are installed on your system)
- Create new project with *Project>NewProject*
- Turn on Generate Debug Info for project node
- Set project Node Properties to PIC C Linker
- Add Source files, setting language tool to PIC C Compiler or Assembler
- Turn on Generate Debug Info for each source node
- Set Generate Debug Info Data to "Microchip" for each source node

Chapter 5. MPLAB Basic Functions

5.1 Introduction

This chapter discusses the basic MPLAB debugging functions. You can be in either the MPLAB-SIM simulator or the MPLAB-ICE emulator mode to access debugging functions.

5.2 Highlights

This chapter covers the following information:

- MPLAB Debugging Functions
- Real-Time Program Execution
- MPLAB-SIM Simulator Environment
- Simulator Considerations
- Break, Trace, and Trigger Points
- Conditional Break Dialog
- Special Windows
- Stimulus Functions
- File Extensions Used by MPLAB

5.3 MPLAB Debugging Functions

After setting up and compiling projects in MPLAB, you'll want to see how your code runs. If you have a device programmer, you can program a microcontroller device and plug the programmed device in your actual application to verify that the application runs as expected. Usually, an application will not run correctly the first time, and you'll have to debug the code. You can use MPLAB-SIM to simulate your code or you can use the MPLAB-ICE emulator to run your firmware in the actual application while you debug.

Either way, you will use break and trace points as you run your code. Look at register values in the Register window or Special Function Register window to see the processor's state as you run and single-step your code.

The MPLAB-ICE emulator runs code at the actual execution speed (real-time) on your target hardware, stopping only at specified break points. MPLAB-SIM simulates the execution of any PICmicro and simulates I/O conditions at speeds that are dependent upon the speed of your PC.

The following debug functions work the same with the simulator or the emulator. The main functions are:

- Emulation Memory (Program Memory Window)
- Break, Trace Points
- Single-Stepping
- Register Monitoring (Special Function Register or File Register Windows)

All of these functions use information from an MPLAB project. Line labels in source code, symbolic locations in memory, and function names from code can be used to set break and trace points and to examine and modify registers.

5.4 Real-Time Program Execution

In this document the term "real-time" is usually applicable only to the MPLAB-ICE emulator.

5.4.1 Execution in MPLAB-SIM Simulator Mode

When the system is said to be running in real-time in the simulator mode, instructions are executing as quickly as possible by the software. This is usually slower than the actual PICmicro microcontroller would run at its rated clock speed.

The speed at which the simulator runs is dependent upon the speed of your computer and how many other tasks you have running in the background. The software simulator must update all of the simulated registers and RAM, as well as monitor I/O, set and clear flags, check for break and trace points in software, and simulate the PICmicro instruction with instructions being executed on your computer's CPU.

Note: Often loops will be used in your code to generate timing delays. When using the simulator, you might wish to decrease these time delays or conditionally remove those sections of your code with "IFDEF" statements to increase simulation speed.

In general when this manual says "real-time" and you are in the simulator mode, this means that the software simulation is executing simulated PICmicro code as fast as your PC can simulate the instructions.

5.4.2 Animate Mode

Animate Mode is a method of automatically single-stepping the processor. The simulator actually executes single steps while in Run mode, but it only updates the values of the registers when it is halted. To view the changing registers in the Special Function Register window or the Watch windows, use Animate mode. Animate mode runs slower than the Run function, but allows you to view changing register values.

5.5 MPLAB-SIM Simulator Environment

MPLAB-SIM is a discrete-event simulator for the PICmicro microcontroller families and is integrated into the MPLAB IDE. The MPLAB-SIM simulator tool is designed to:

- Model operation of Microchip Technology's PICmicro microcontrollers, e.g., PIC12CXX, PIC14000, PIC16C5X, PIC16CXX, and PIC17CXX.
- Assist users in debugging software that uses Microchip microcontroller devices.

A discrete-event simulator, as opposed to an in-circuit emulator (like the MPLAB-ICE emulator) is designed to debug software. MPLAB-SIM allows you to modify object code and immediately reexecute, inject external stimuli to the simulated processor, and trace the execution of the object code. A simulator differs from an in-circuit emulator in three important areas:

- I/O timing
- Execution speed
- Cost

5.5.1 I/O Timing

External timing in MPLAB-SIM is processed only once during each instruction cycle. Transient signals, such as a spikes on $\overline{\text{MCLR}}$ smaller than an instruction cycle, will not be simulated but may be caught by an in-circuit emulator.

Note:	Stimulus is injected into MPLAB-SIM prior to the next instruction cycle.

5.5.2 Execution Speed

The execution speed of a discrete-event software simulator is orders of magnitude less than a hardware oriented solution. Users may view slower execution speed as a handicap or as a tool. MPLAB-SIM attempts to provide the fastest possible simulation cycle, and depending upon the mode of operation, can operate on the order of milliseconds per instruction.

5.5.3 Cost

Microchip Technology has developed the MPLAB-SIM simulator to be the most cost-effective tool for debugging application firmware. MPLAB-SIM does not require any external hardware to your PC, and operates in most respects exactly the same as the MPLAB-ICE emulator. Unless you need to debug your application in real-time on your actual hardware, MPLAB-SIM can usually be used to find and correct most coding errors.

5.5.4 Debugging Tool

MPLAB-SIM is particularly suitable for optimizing algorithms. Unlike an emulator, the simulator makes many internal registers visible and can provide software tools that are difficult or expensive to implement in a hardware in-circuit emulator. For the most part, MPLAB-SIM can be used to fully debug your system unless you run into situations where an in-circuit emulator is required.

5.6 Simulator Considerations

MPLAB-SIM executes on instruction cycle boundaries, and resolutions shorter than one instruction cycle (Tcy) can not be simulated. MPLAB-SIM is a discrete-event simulator where all stimuli are evaluated, and all responses are generated, at instruction boundaries, or Tcy = 4 Tosc, where Tosc is the input clock period. Therefore, there are some physical events that can not be accurately simulated. These fall into the following categories:

- Purely asynchronous events
- Events that have periods shorter than one instruction cycle

Note: In summary, the net result of instruction boundary simulation is that all events get synchronized at instruction boundaries, and events smaller than one instruction cycle are not recognized.

The following list itemizes the functions and peripherals among the entire PICmicro family of microcontrollers that are affected by simulation on instruction cycle boundaries:

- Clock pulse inputs smaller than one cycle can not be simulated even though timer prescalers are capable of accepting clock pulse inputs smaller than one cycle.
- PWM output pulse resolution less than one cycle is not supported.
- Compares greater than 8-bits are not supported.
- In unsynchronized counter mode, clock inputs smaller than one cycle can not be used.
- The oscillator waveform on RC0/RC1 pins can not be shown.
- MPLAB-SIM does not simulate serial I/O.

5.7 Break, Trace, and Trigger Points

The debug functions affect execution of program instructions based upon the following elements:

- Break Points
- Trace Points
- Pass Counter Addresses

MPLAB limits the number of named address ranges to a maximum of 16 in each dialog.

Trace points and break points function totally independent of each other, and you can set them at any program memory location.

The following figures show the dialog boxes for assigning names to address ranges. Access the Break Point Settings dialog through the *Debug>Break Settings...* menu item, and the Trace Settings dialog through the *Debug>Trace Settings...* menu item.

Figure 5.1: Break Point Settings Dialog

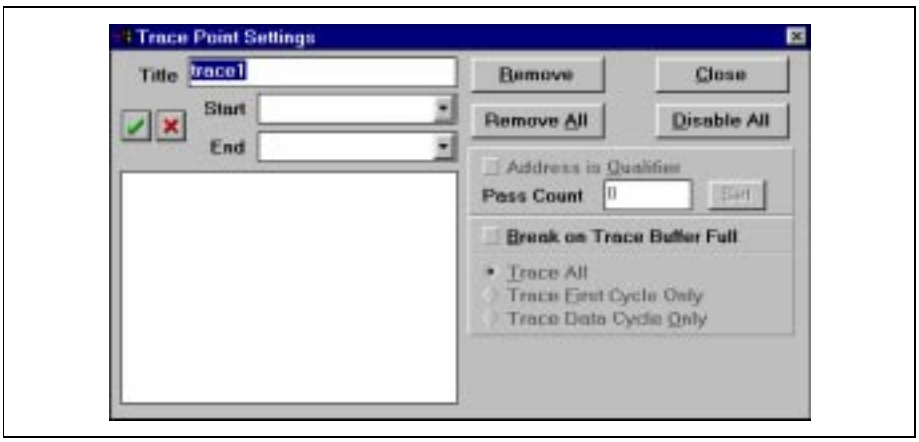


Figure 5.2: Trace Point Settings Dialog

5.7.1 Real-Time Break Points

A break point is a condition in which the processor executes code and halts after a certain condition is met.

MPLAB provides the following ways to set a break point:

- Break on Address Match
- Break on Trace Buffer Full
- Break on Pass Count Reached
- Break on Stack Overflow
- Break on Watch Dog Timer Time Out
- User Halt

The Program Memory Window shown in Figure 5.3 shows the following information:

B Break Points

T Trace Points

Q Pass Counter Addresses

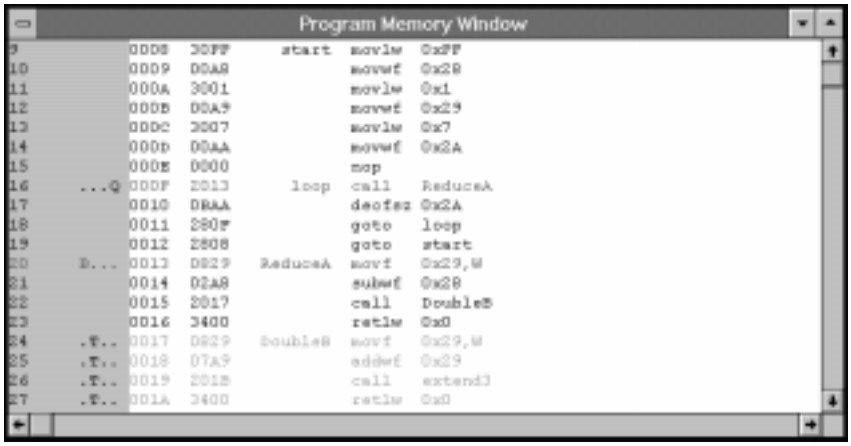


Figure 5.3: Program Memory Window

5.7.1.1 Break on Address Match

Break on Address Match allows you to halt the processor when the processor program counter equals a certain value. The processor breaks after the valid instruction is executed. For example, if a break point is set at address 5Ah, then the processor breaks after executing the instruction at address 5Ah.

5.7.1.2 Break on Trace Buffer Full

MPLAB can be set to halt the processor after capturing 8K selected cycles (when the trace buffer is full).

5.7.1.3 Break on Pass Counter Equal to Predefined Value

MPLAB has a Pass Counter switch that you can assign to either trace logic or break logic. The pass counter can be used to break or trace after the processor executes an address a predefined number of times.

For example, if the Pass Counter is assigned to break logic, then when the pass counter decrements to zero, the pass counter acts as a break point and halts the processor.

5.7.1.4 Break on Stack Overflow

Break on Stack Overflow causes MPLAB to execute a break when the stack overflows.

5.7.1.5 Break on Watchdog Timer

If enabled, MPLAB executes a break when a Watchdog Timer time-out generates a device reset.

5.7.1.6 User Halt

MPLAB IDE provides three ways to stop at a break point any time the processor is running:

- Click *Debug > Run > Halt*
- Press **F5**
- Click the Halt Icon (red stop light)

5.7.2 Real-Time Trace Points

A trace is a function that logs program execution. The MPLAB-SIM simulator has an 8K real-time trace buffer that logs addresses and opcodes as they execute. This circular trace buffer continues logging data after the buffer is full, losing the oldest data (unless you have selected Break on Trace Buffer Full in the Break Point Settings dialog).

5.7.2.1 Circular Trace Buffer

MPLAB continuously captures selected bus cycles into the trace buffer.

The status information captured into the trace buffer is grouped as follows:

- 16 Bits of Address
- 16 Bits of Opcode/Data
- Time Stamp and Changed Registers

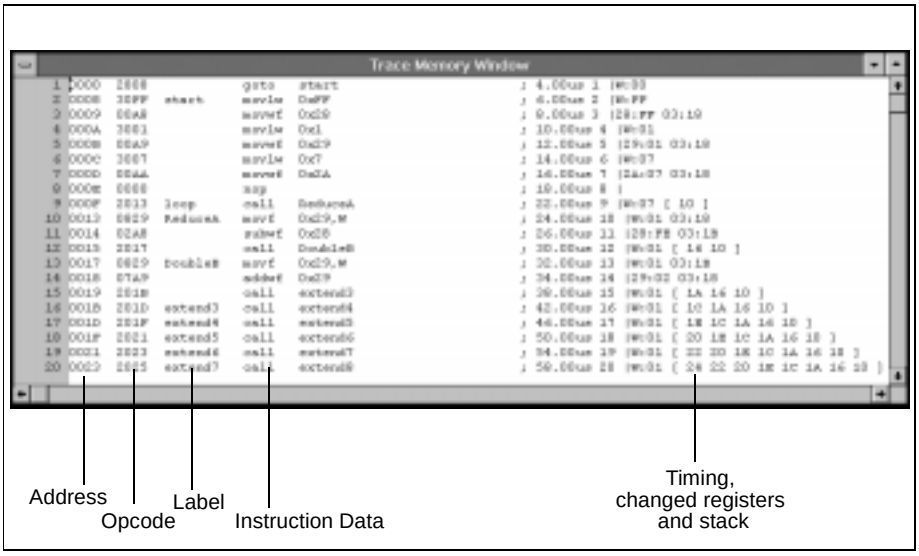


Figure 5.4: Trace Memory Window

5.7.2.2 Halting Trace from the Tool Bar 

Halt Trace allows you to take a snapshot of the trace buffer and look at the captured trace without halting the processor. In the Tool Bar, click **Halt Trace** to display a snapshot of the trace buffer without halting the processor. Once the trace buffer is halted, click **Halt Trace** again to take another trace snapshot.

5.7.2.3 MPLAB-SIM Simulator Trace Display

The trace window can be used to collect executed instructions from the MPLAB-SIM simulator. In addition to the data shown by the MPLAB-ICE emulator, the simulator will show a time stamp on each line and will echo changes to registers. The time stamp uses the same data as the MPLAB Stop Watch. You can reset the time stamp by resetting the Stop Watch.

Note: Saving the trace display to a file in the simulator mode will produce even bigger files than saved MPLAB-ICE emulator trace files.

5.7.3 Assigning a Pass Count to Break or Trace Points

MPLAB has a 16-bit Pass Counter that decrements by one on any address match in program memory.

When the processor is in a Halt state, you can modify the count value in the Break Point Settings or Trace Point Settings dialog box. To set up the Pass Counter, first set the desired address ranges and then load the counter with a desired count value (up to 16 bits). When the counter decrements to zero, the emulator will halt.

5.7.3.1 Pass Counter Assigned to Break

If the Pass Counter is assigned to Break, the processor halts upon encountering a break point (either internal or external conditions) or when the Pass Counter reaches zero.

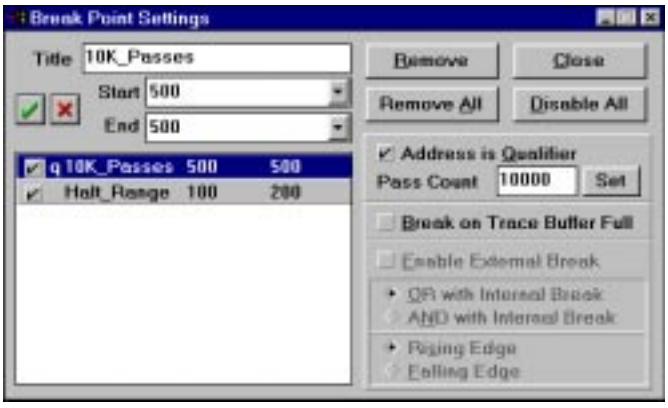


Figure 5.5: Break Point Settings Dialog – Pass Counter

Example 5.1: This example shows break points and pass counter addresses used in the same code (Figure 5.5). Keep in mind that break points and pass counter addresses are independent of each other.

1. Set up a named break point range from address 100 to 200.
   - Type Halt_Range in Title box
   - Type 100 in the Start box and 200 in the Stop box
   - Click the check button to enter the break point
2. Set a Pass Counter Address at 500.
   - Type 10K_Passes in Title box
   - Type 500 in the Start box and in the Stop box
   - Click the check button to enter the break point
3. Load the Pass Counter with a value of 10000.
   - Select (click on) the 10K_Passes break point
   - Click on the check box of the now-ungrayed Address is Qualifier
   - Type 10000 in the Pass Count box and click on **Set**

   The processor halts if it executes any instructions within the address range 100 to 200 or after executing 10000 instructions at address 500.

5.7.3.2 Pass Counter Assigned to Trace

If the Pass Counter is assigned to trace, then the real-time trace buffer does not capture data until the Pass Counter decrements down to zero. When the pass counter decrements to zero, the trace buffer starts capturing data on valid cycles.

5.7.3.3 Using Pass Counter to Count Events

The Pass Counter decrements each time an event occurs. You can use this feature to count how many times an event happens.

5.8 Conditional Break Dialog

When a conditional break is set, MPLAB halts when the value of a specified internal register reaches a preset value or condition.

Access the Conditional Break dialog through the *Debug>Execute>Conditional Break...* menu item.

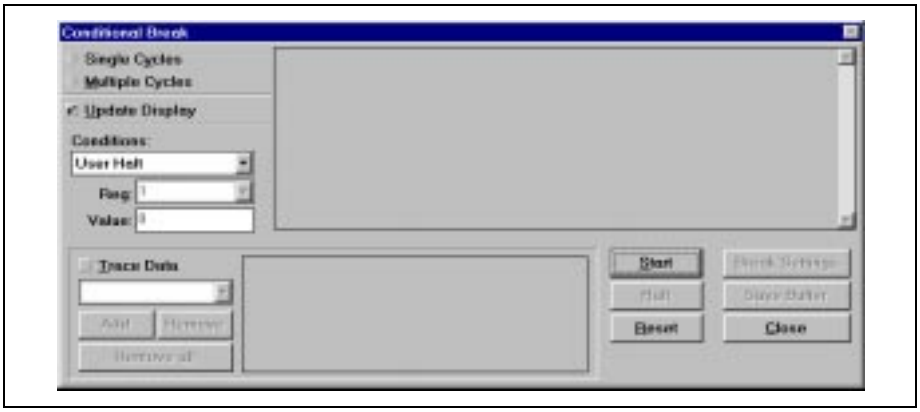


Figure 5.6: Conditional Break Dialog

5.8.1 Conditions

MPLAB will stop at a break point in the Conditional Break dialog based on one of the following conditions:

- User Halt – MPLAB executes until you press the **Halt** button on the Conditional Break display.
- Number of Cycles – MPLAB halts after the target processor executes the specified number of cycles.
- Logic Condition satisfied.

5.8.2 Trace Data

Trace Data allows you to track the value of the registers in the Conditional Break dialog.

5.8.3 Single Cycle

In the Single Cycle mode, MPLAB single steps the processor until the condition is met.

5.8.4 Multiple Cycles

In Multiple Cycle mode:

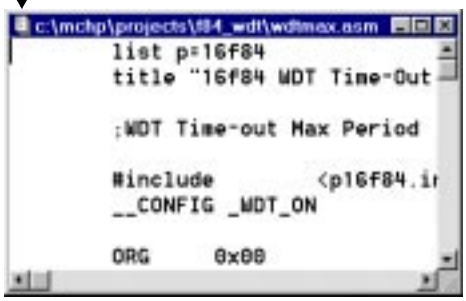
- Conditional Break executes instructions in real-time (in the emulator), halts at user selected break points, checks the specified condition and continues executing instructions in real-time. The emulator or simulator only stop when meeting the specified condition.
- Break points and register conditions are only checked at the break points you specify in the Break Setting dialog.

5.9 Special Windows

MPLAB provides windows for viewing various information.

Note: Use the system window control to change how data is displayed in the window.

system window control



5.9.1 Program Memory

Program memory can be viewed by selecting *Window>Program Memory*.

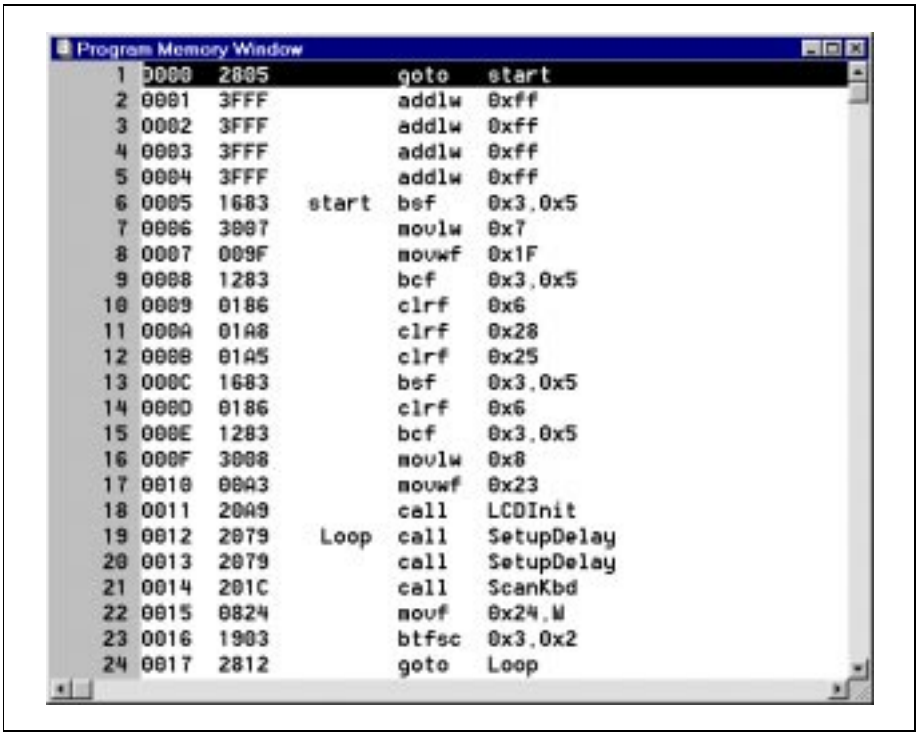


Figure 5.7: Program Memory Window – Machine Code Display

The program memory can be displayed three ways. The desired format is chosen through the system menu.

- **Hex Code Display** – This displays the program memory as hex data. This option is most useful when using a device programmer (Figure 5.8).
- **Machine Code Display** – This displays the disassembled hex code with no symbolic information (Figure 5.7).
- **Disassembly Display** – This displays the disassembled hex code with symbols.

When this window is in Machine Code Display mode or Disassembly Display mode, the instruction at the current program counter address will be highlighted. Other features of MPLAB can alter the display of the program memory window.

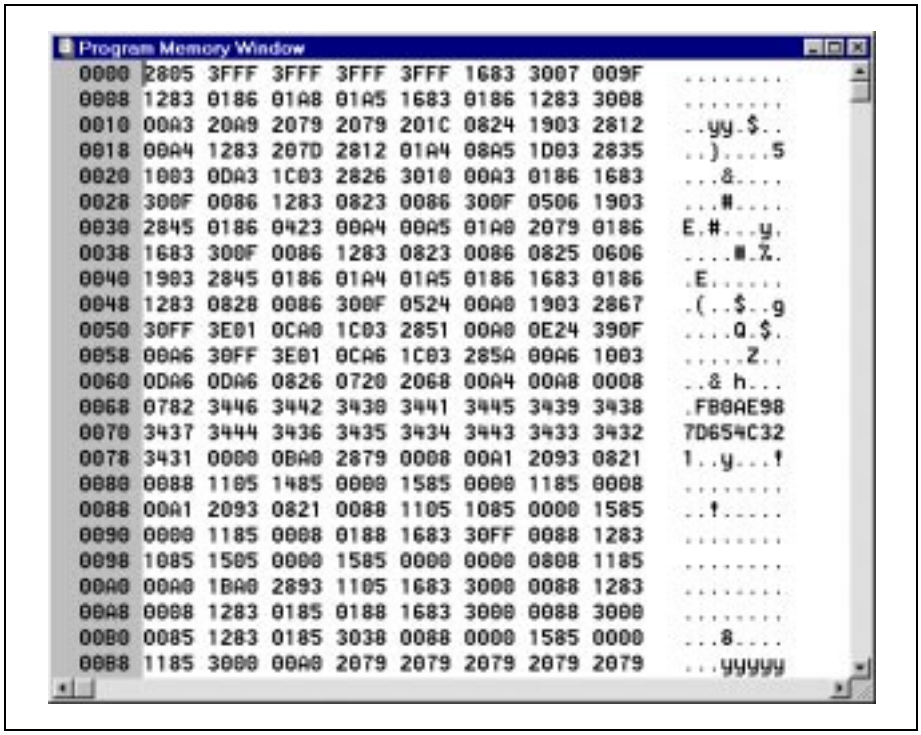


Figure 5.8: Program Memory Window – Hex Code Display

5.9.2 Trace Memory

The trace memory window takes a “snapshot” of your program’s execution. For emulators that have a trace buffer, this shows how your program runs at full speed.

Some applications, such as motor control systems, can not be halted. Some bugs may only appear when the application is running, i.e., they don’t occur when single-stepping through the code The trace buffer gives you another tool for testing such applications. Check your emulator user’s guide for more information on the information collected in its hardware trace buffer.

In the simulator, the trace buffer is useful for collecting a long record of program execution so you can go back and analyze it later. The simulator will show a little different information from the emulator’s trace.

To use the simulator’s trace buffer, first you must select code to trace. If you click and drag across the program memory window you can select instructions to trace, then press the right mouse button. This will bring up a menu where you can select “Trace Point(s).”

Now reset and run the code, then halt it after it runs for a few seconds. Select *Window>Trace* to see the collected trace:

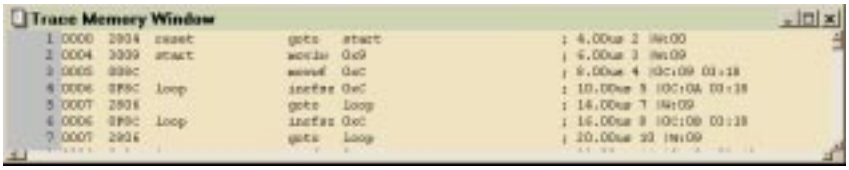


Figure 5.9: Trace Memory Window

The simulator puts a time stamp on each line and also shows any registers that were changed along with their values.

5.9.3 EEPROM Memory

If the emulated device contains EEPROM, the EEPROM contents can be viewed by selecting *Window>EEPROM Memory*.

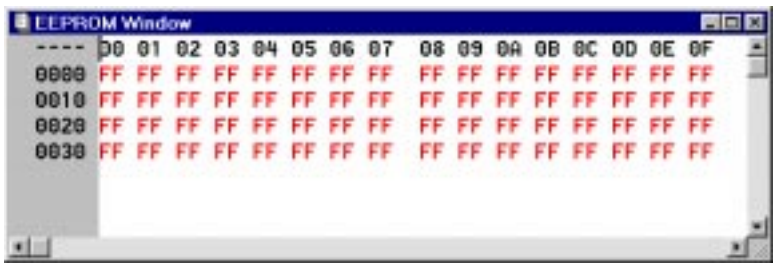


Figure 5.10: EEPROM Window

EEPROM memory cannot be modified through this window. Use the Modify dialog, accessed by selecting *Window>Modify...*.

5.9.4 Calibration Memory

If the emulated device contains calibration memory, the calibration memory can be viewed by selecting *Window>Calibration Memory*. The appearance of this window will be different, depending on the emulated device.

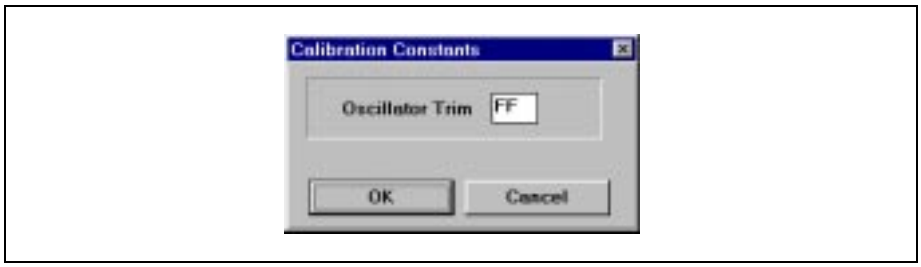


Figure 5.11: Calibration Constants Dialog – PIC12C509

5.9.5 Absolute Listing

The Absolute Listing Window shows the list file generated by the assembler or compiler. The Absolute Listing displays source code with the generated object code.

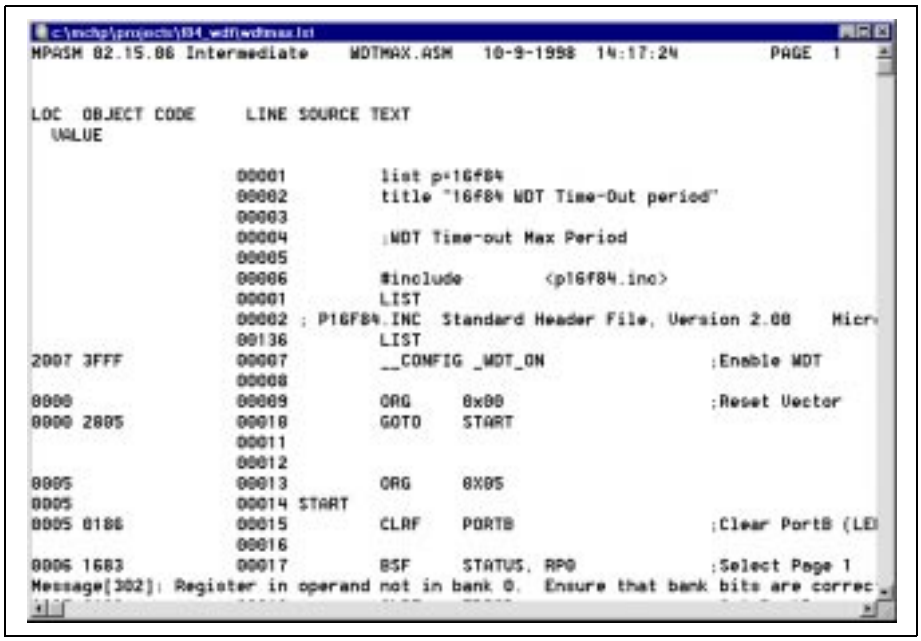


Figure 5.12: Absolute Listing Window

5.9.6 Stack

The contents of the stack can be viewed by selecting *Window>Stack*.

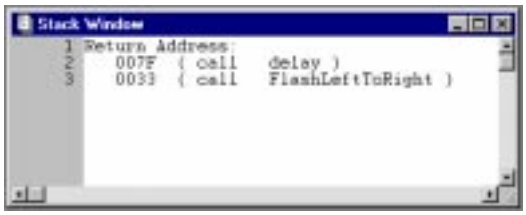


Figure 5.13: Stack Window

Note: If Stack Break Enable is set, MPLAB will display stack overflow and underflow warnings when they occur.

The contents of the stack may be displayed with (shown) or without line numbers. The desired format is chosen through the system menu.

Note: The system menu is accessed by clicking the top left corner of the program memory window.

5.9.7 File Registers

File registers can be viewed by selecting *Window>File Registers*. This window displays all of the file registers of the emulated device.

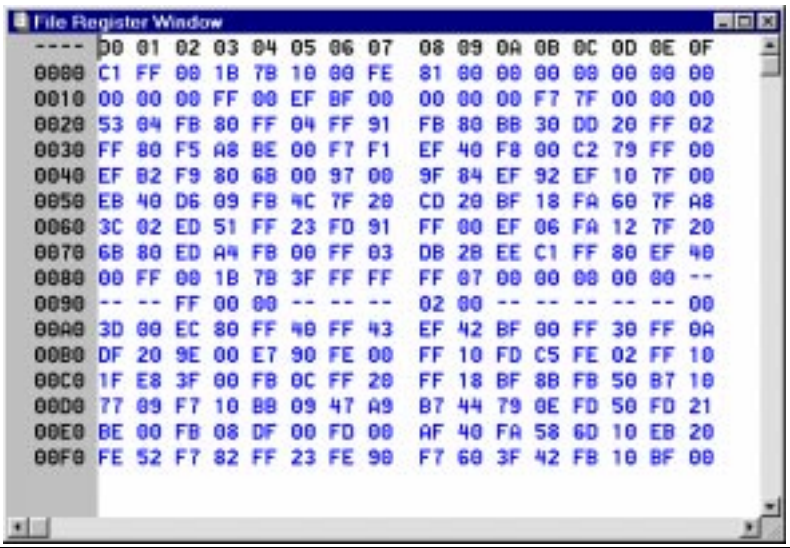


Figure 5.14: File Register Window – Hex Display

File registers can be displayed three ways. The desired format is chosen through the system menu.

- **Hex Display** – This displays the file registers as hex data (Figure 5.14).
- **Symbolic Display** – This displays each file register symbolically with corresponding data in hex, decimal, binary and character formats (Figure 5.15).
- **ASCII Display** – This displays the file registers as ASCII data.

Note: The system menu is accessed by clicking the top left corner of the program memory window.

File register contents can also be modified through this window. To change a range of registers, click the left mouse button and drag over the values you want to change. To change one register, simply place the cursor over the register you want to change. Then click the right mouse button to display a pop-up menu. Select **Fill Register** to display the Modify dialog (Figure 5.24) with the address range already entered.

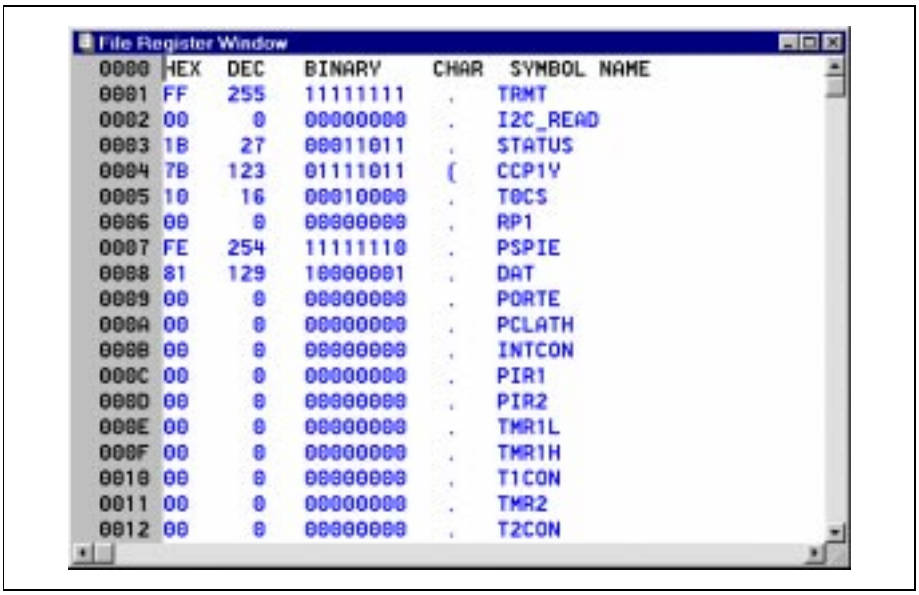


Figure 5.15: File Register Window – Symbolic Display

5.9.8 Special Function Registers (SFRs)

The special function registers can be displayed by selecting *Window>Special Function Registers*. The format provided by this window is more useful for viewing the SFRs than the normal file register window, since each SFR name is included and several number formats are presented.

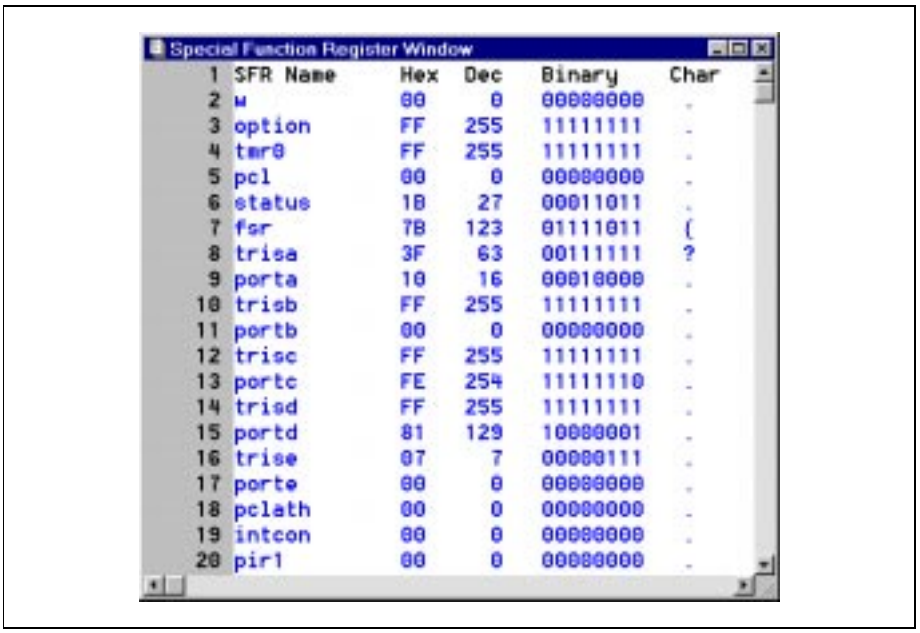


Figure 5.16: SFR Window

SFRs may be displayed with (shown) or without line numbers. The desired format is chosen through the system menu.

> **Note:** The system menu is accessed by clicking the top left corner of the program memory window.

To modify an SFR, double click on an SFR name to display the Modify dialog (Figure 5.24) with the address already entered.

5.9.9 Show Symbol List

The Show Symbol List Window displays the symbols from your source code. These symbols are from the *.COD file in your project.

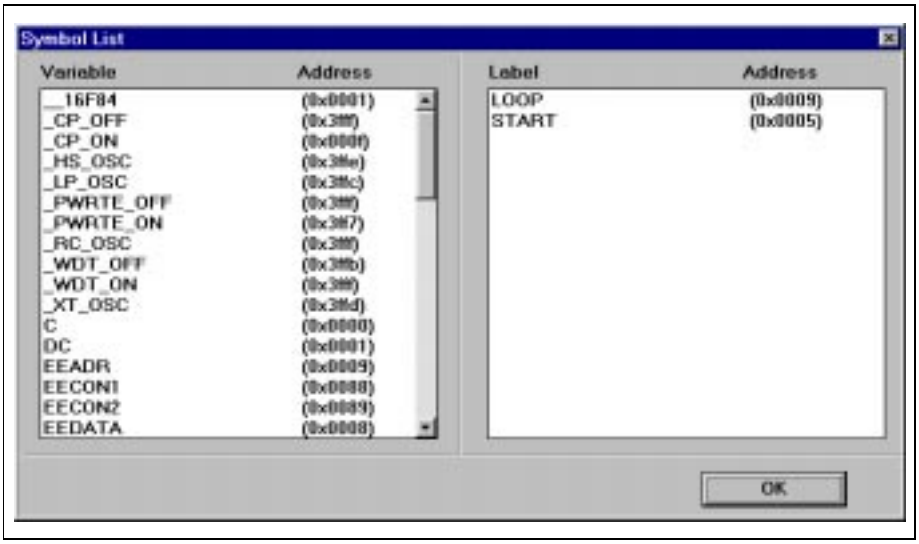


Figure 5.17: Symbol List

5.9.10 Stopwatch and Clock Frequency

The stopwatch allows you to measure code execution time. It is not always accurate while single stepping. The stopwatch calculates time based upon the clock frequency of the PICmicro.

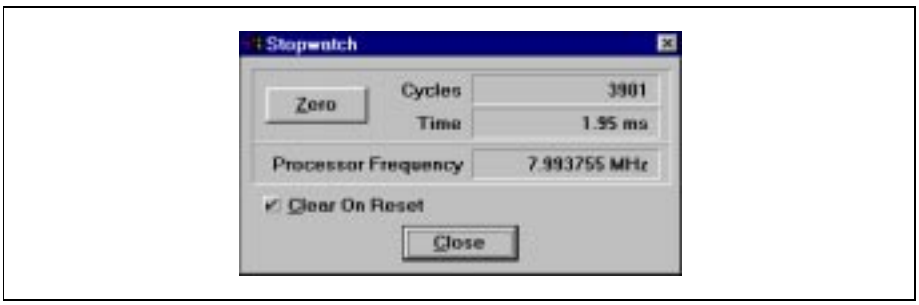


Figure 5.18: Stopwatch

Set the clock frequency in *Options>Processor Setup>Clock Frequency...* dialog.

To program the clock, display the Processor Clock dialog by selecting *Options>Processor Setup>Clock Frequency*. Refer to the specific device's data sheet to determine the supported frequency range.

Figure 5.19: Processor Clock Dialog – On-Board Clock

Select the Desired Frequency magnitude (MHz, kHz, or Hz), enter the Desired Frequency, and click **Set Clock**. The “Actual Frequency” is for use with MPLAB-ICE only.

Click **Close** to exit the dialog.

5.9.11 Watch Windows

MPLAB allows the contents of file registers to be monitored through a watch window. To open a watch window, select *Window>New Watch Window*.

Both the Add Watch Symbol dialog (Figure 5.20) and the Watch_1 window (Figure 5.21) will open. Select symbols to add by clicking on them in the Add Watch Symbol list box. Click **Add** to add them to the Watch_1 window. You can also delete symbols by clicking on them and then clicking **Delete**. To see and change the properties of a symbol, click on **Properties** to open the Properties dialog (Figure 5.23).

Click **Close** when you have finished adding symbols.

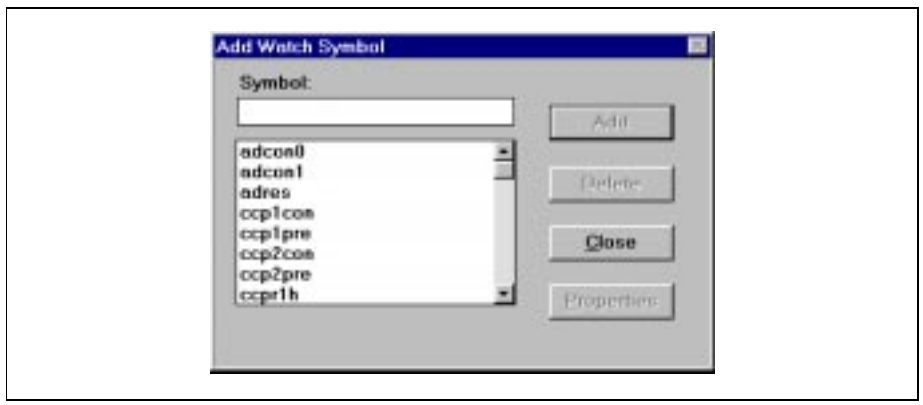


Figure 5.20: Add Watch Symbol Dialog

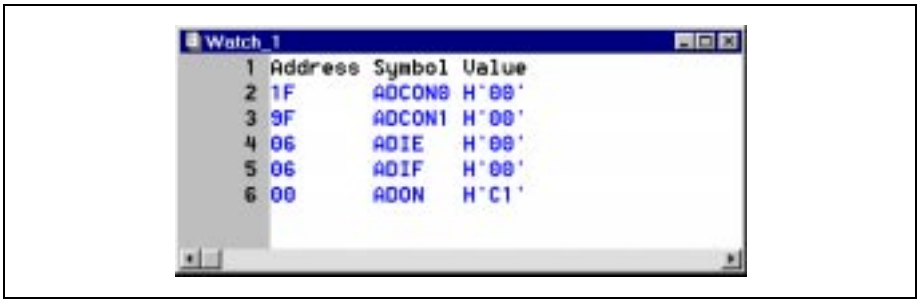


Figure 5.21: Watch Window

The contents of the watch window may be displayed with (shown) or without line numbers. The desired format is chosen through the system menu.

The system menu also is used to edit the information in the watch window.

- **Add Watch** – Used to add a symbol. Works the same as pressing the **<Ins>** key. Causes the Add Watch Symbol dialog to appear (Figure 5.20).
- **Delete Watch** – Used to delete a symbol. Works the same as pressing the **<Del>** key. You can also delete symbols through the Add Watch Symbol dialog (Figure 5.20) using the delete button.
- **Edit Watch** – Used to specify the display format of symbols. Causes the Edit Watch dialog to appear (Figure 5.22).

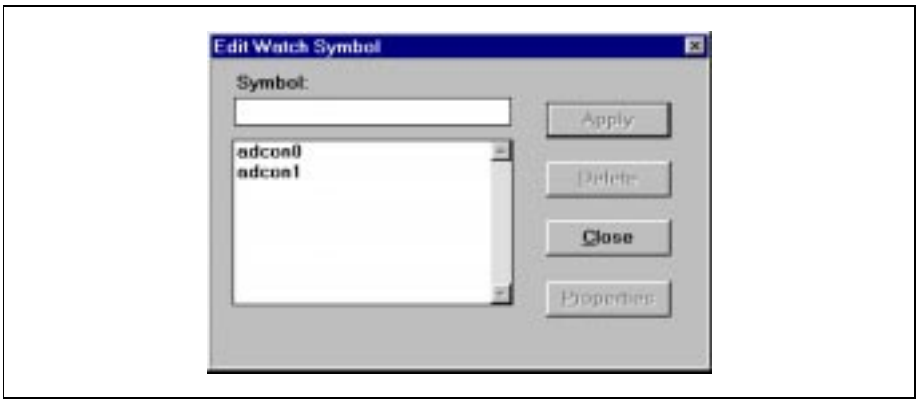


Figure 5.22: Edit Watch Symbol Dialog

- **Save Watch** – Used to save the set up of a window.

The Properties dialog box allows you to select the format in which the symbols will be displayed. You may display the Properties dialog box by selecting the Properties button from the Add Watch Symbol and Edit Watch Symbol dialog box.

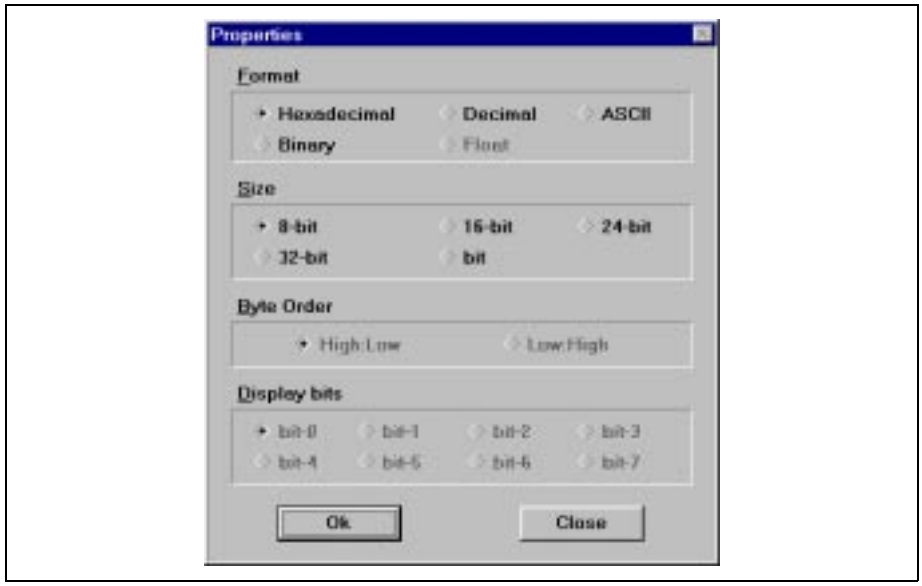


Figure 5.23: Properties Dialog

- **Format** – Determines what type of number to display. Float is available only for 24- and 32-bit Size selections.
- **Size** – Determines how many bytes will be included in the display of the number.
- **Byte Order** – Determines the display order of each byte, available only for 16-bit numbers.
- **Display Bits** – Determines which bit to display when Size-bit is selected.

5.9.12 Modify

The Modify dialog lets you read and write a memory location or a range of memory locations. Modify can work on the following memory areas:

- Data
- Stack
- Program
- EEPROM (If available)

The Modify dialog may be accessed by selecting *Window>Modify...*.

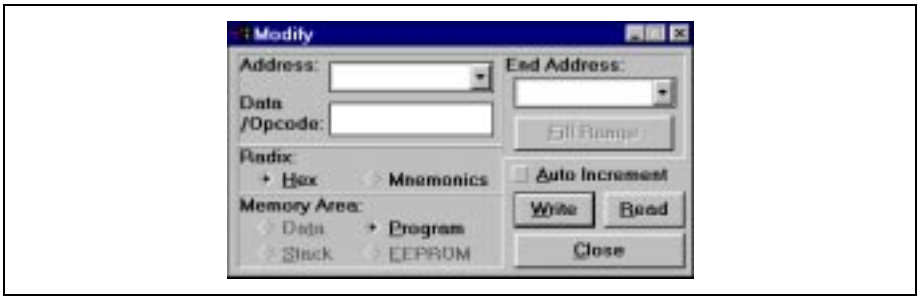


Figure 5.24: Modify Dialog

5.10 Stimulus Functions

The stimulus generates signals for the simulator. You can set pins high or low, and inject values directly into registers. There are four stimulus modes:

- Asynchronous stimulus – An interactive dialog to control signals on input pins
- Stimulus Pin File – The contents of a text file describes signals to input pins
- Stimulus Register File – The contents of a text file are used to set 8-bit values directly into a register
- Clock Stimulus – A regular, programmable, periodic source of stimulus pulses

5.10.1 Asynchronous Stimulus Dialog

This stimulus feature provides a dialog button to simulate +5 and 0 volts being applied to input pins. As your program executes with the simulator, you can press buttons on this dialog to change levels on pins.

As an example, we'll set up a signal that will toggle the level on a pin on I/O portb of the PIC16F84.

Select *Debug>Simulator Stimulus>Asynchronous Stimulus...*. This dialog will be displayed:

Figure 5.25: Asynchronous Stimulus Dialog

Now put the cursor over the button labelled "Stim1 (P)" and click the right mouse button. A dialog will pop up. Scroll down and click on *Toggle*.

Figure 5.26: Toggle Option

Again put the cursor over the button now labelled "Stim1 (T)" (the "P" was replaced by a "T", meaning "Toggle"), press the right mouse button and select *Assign Pin...*.

Another dialog will pop up with a list of pins on the PIC16F84.

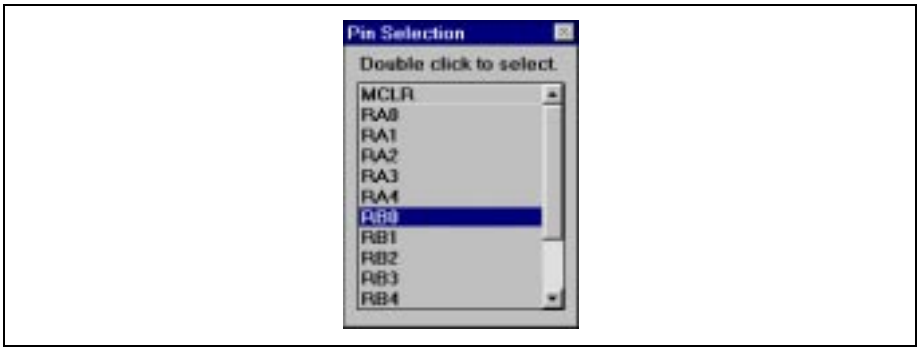


Figure 5.27: Pin Selection

Put the cursor over "RB0" and double click. The Asynchronous Stimulus dialog should now look like Figure 5.28. Note that the button now shows "RB0 (T)."

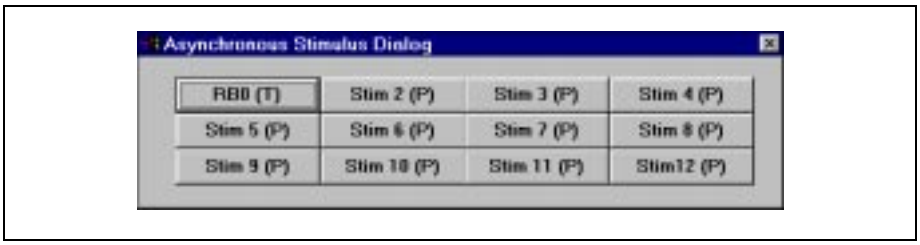


Figure 5.28: Asynchronous Stimulus Dialog – RB0(T)

Choose *Debug>Run>Animate* to get the processor running in a "fast single step" mode. The status bar will alternate run to stop very fast.

Press the "RB0 (T)" button on the Asynchronous Stimulus dialog. You should see the value of portb in the Special Function Register window change as you repeatedly click the button to simulate a high signal then a low signal applied to portb pin 0.

5.10.2 Pin Stimulus Files

A Pin Stimulus file consists of columns of input ones and zeroes that will be applied to pins when the “Cycle” value in the Stopwatch matches the CYCLE column.

Use *File>New File* and type in the following text. You do not have to type in the text after the “;” and “!” comment delimiters.

```
CYCLE   RB1    RB0
20      0       0
41      1       0   ; apply high to port b bit 1
52      0       1   ; apply high to port b bit 0, set bit 1 low
55      1       1
60      0       0
65      1       0   ; toggle bit 1, then...
76      0       1   ! ...toggle bit 0.
```

Use *File>Save As...* to save as tutor84.sti.

Note: For backward compatibility with earlier versions of MPSIM, the first line must always start with the word CYCLE or STEP. This column specifies the CYCLE (as determined by MPLAB’s Stopwatch Window) where the values in the other columns will be applied.

After the word CYCLE in the first line of the file are the pin names for the PICmicro pins that will receive the high and low stimulus values. In this example pins RB1 and RB0, two inputs on Port B, will receive stimulus inputs.

In this file, the second column contains values that will be applied to RB1 (PortB bit 1) and the third column has values for RB0 (PortB bit 0). These names must match the Microchip PICmicro pin names for the processor being simulated. You can see a list of all supported pins by looking at the pin assignment pull down for the Asynchronous Stimulus (right click on a stimulus button).

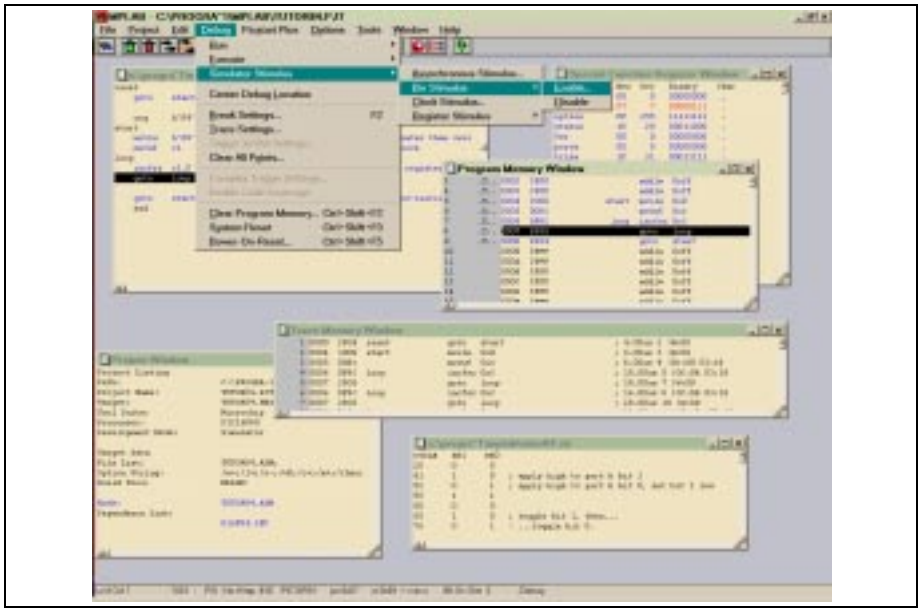


Figure 5.29: Enable Pin Stimulus

Comments can be put on a line using the ";" or "!" character preceded and followed by at least one space.

The Stopwatch Window will also show the elapsed time at each instruction, as determined from the CYCLE value and the clock frequency. If the Stopwatch is reset to 0, the pin stimulus file will also be effectively reset.

Open the Stopwatch window by selecting *Window>Stopwatch*. Also select *Window>Special Function Registers*. We'll be interested in watching PortB. Alternatively we could have just added PortB to a watch window.

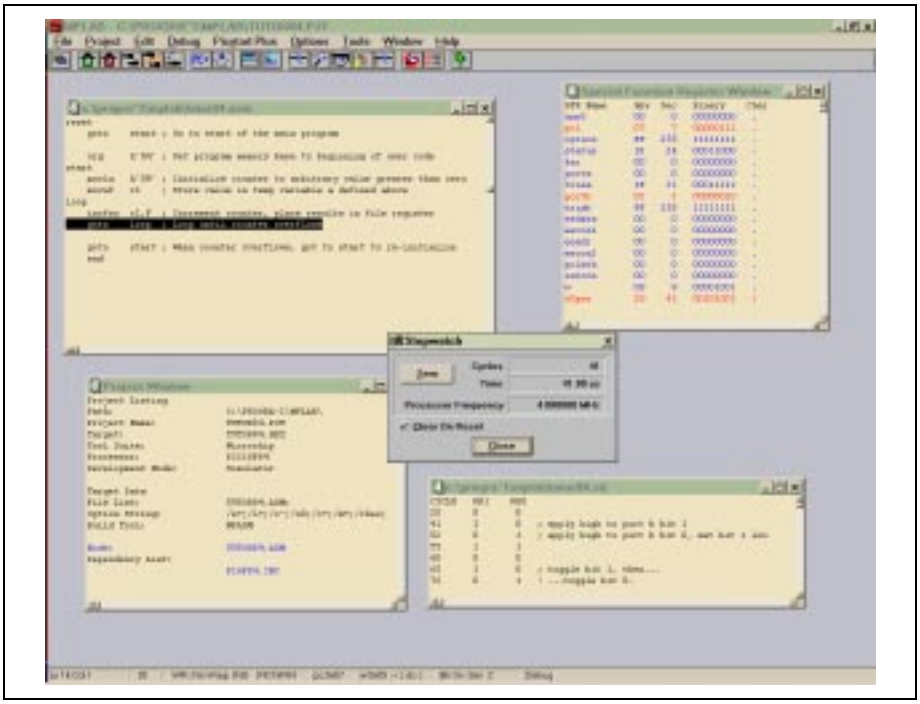


Figure 5.30: Stopwatch Window

Reset and single step until you execute 41 cycles. Here you'll see that PortB change its value as set in the second line of the stimulus file.

5.10.3 Register Stimulus Files

A Register Stimulus file consists of a single column of values that will be sent to a register when the program memory address reaches the location set in the Register Stimulus Dialog. This is useful for simulating an A/D operation.

Open a new file with *File>New File* and type in the following list of numbers:

```
10
2E
38
41
50
7A
99
A0
FD
```

Save it with *File>Save As...* and name it `tutor84.reg`. This file will be used to sequentially inject these values into a register.

Select *Simulator Stimulus> Register Stimulus>Enable...*, then set `loop` to be the place in the program when values are injected, and for demonstration purposes let s inject them into the file register at address 0x0d. After you set `loop` and `0d` in the appropriate boxes, press **Browse** to bring up the file dialog, and then select tutor84.reg as the register stimulus file:

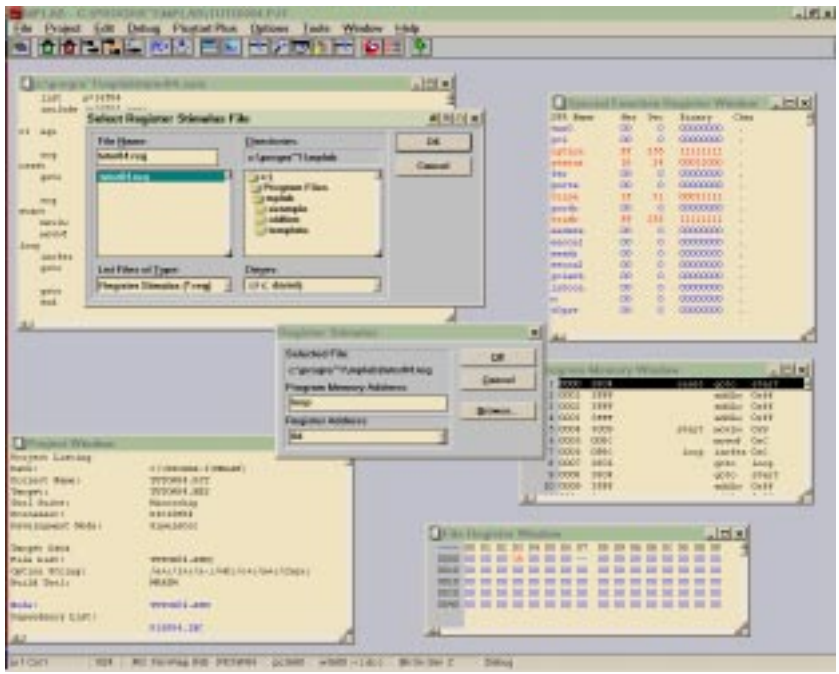


Figure 5.31: Register Stimulus

Open up the *Window>File Registers* window to see the effect of this stimulus.

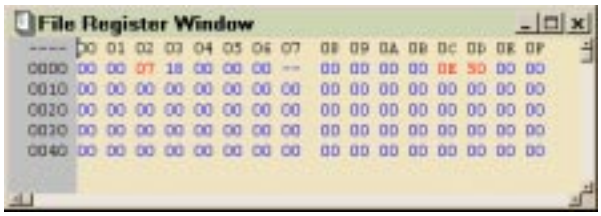


Figure 5.32: File Register Window

Reset and then single step the processor. Every time you get to `loop` the value in the file register at address `0x0D` will change. The list of values in `tutor84.reg` will be sequentially injected into the selected file register (`0x0D` as shown having a value of `0x50` above). Values of `0x10`, `0x2E`, etc. will be injected into the register selected in the *Debug>Simulator Stimulus>Register Stimulus* dialog every time `loop` is executed.

After the last value is injected (`0xFD` in `tutor84.reg`), the first value will be used again (`0x10`). The list will cycle as long as MPLAB-SIM executes.

5.10.4 Clock Stimulus

The clock stimulus generates a regular waveform on a pin with a duty cycle that's specified in terms of the processor clock cycles.

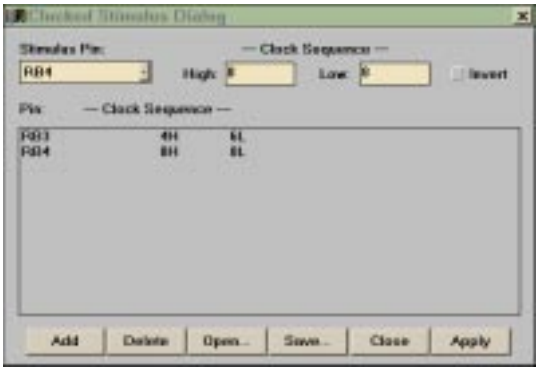


Figure 5.33: Clock Stimulus

If you bring up the *Debug>Simulator Stimulus>Clock Stimulus...* dialog, you can enter various stimulus clocks as shown above. When you step or run using the above settings, RB3 will be high for 4 clock cycles, then go low for 6 clock cycles. RB4 will go high for 8 clock cycles then low for 8 clock cycles. Both will repeat until you exit MPLAB or delete them using this dialog.

5.11 File Extensions Used by MPLAB

The default extensions of files used by the MPLAB are listed below:

*.ASM	Assembly language source file
*.C	C source file
*.CFG	Configuration/setup files
*.COD	Contains symbolic information and object code
*.DAT	Simulator data file
*.ERR	Error file generated by assembler/compiler
*.H	C include file
*.HEX	PICmicro machine code in hex format
*.HLP	Help file
*.INC	Assembly language include file
*.INI	MPLAB and language tool configuration file
*.KEY	MPLAB key mappings
*.LKR	MPLINK linker script
*.LST	Absolute listing file generated by assembler/compiler
*.MTC	Language tool configuration file
*.PJT	Contains most of the information related to a project
*.REG	Stimulus register file
*.STI	Stimulus pin file
*.TB	Conditional break trace file
*.TBR	Toolbar file
*.TPL	Template file
*.TRC	Trace save files
*.TXT	Trace save files (MPLAB-2000, CSV format)
*.WAT	Watch window file

Chapter 6. MPLAB Toolbar and Menu Options

6.1 Introduction

This chapter gives detailed information on using the MPLAB desktop toolbars and menu options. The chapter organization follows the entries on the pull-down menus.

6.2 Highlights

This chapter will discuss the following:

- MPLAB Desktop
- MPLAB Menus and Functions
- File Menu
- Project Menu
- Edit Menu
- Debug Menu
- Programmer Menu
- Options Menu
- Tools Menu
- Window Menu
- Help Menu

6.3 MPLAB Desktop

The MPLAB desktop is a resizable window that operates independently of the rest of the menu items. You can maximize the desktop to provide a full screen view by clicking the left mouse button on the up-arrow in the upper right hand corner of the desktop.

The Figure 6.1 shows a maximized desktop. The up arrow replaces the up-down arrow combination. Click on the up-down arrow combination to reduce the window and size it manually.

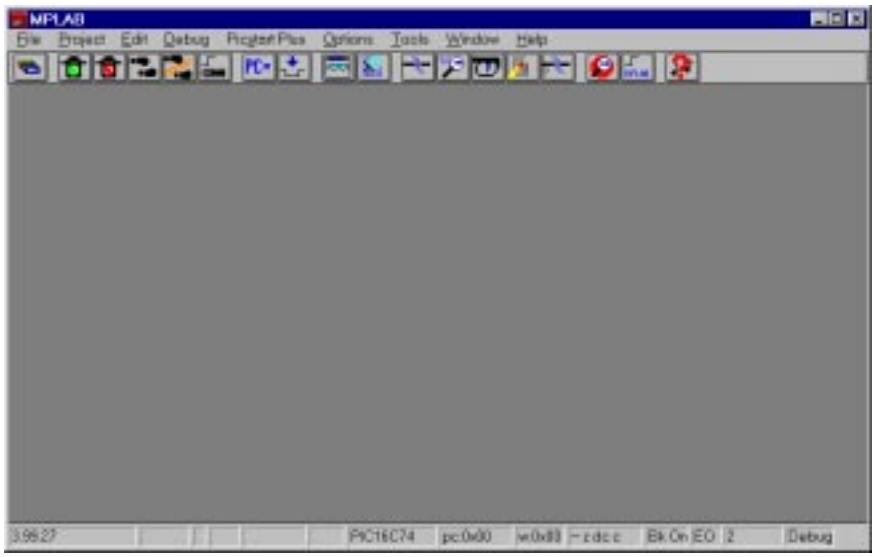


Figure 6.1: MPLAB Desktop

MPLAB dialog boxes behave as normal Windows applications and allow you to access standard Windows functions (as well as MPLAB-specific functions) through the MS Windows system button in the upper left hand corner. Other standard Windows features include window size buttons, icon buttons, vertical and horizontal scroll bars, and elevator buttons.

All MPLAB functions are accessible through the menu bar located across the top of the desktop. MPLAB menus that pull down from the menu bar allow you to access the emulator functions. Underlined characters on the pull down menus are key accelerators. The key accelerators enable the named function when the menu is pulled down. For example, if the File menu is pulled down, pressing **O** enables Open.

When not using MPLAB, you can iconize the window by clicking the minimize button.

6.3.1 Toolbars

For your convenience, MPLAB contains four toolbars to provide you with shortcuts for performing routine tasks. The four toolbars are:

- Edit Toolbar
- Debug Toolbar
- Project Toolbar
- User Defined Toolbar

The toolbar icons can be reconfigured by the user to pertain to their specific needs.

Figure 6.2: MPLAB Debug Toolbar

6.3.2 Status Bar

The table below describes the information presented in the status bar (Figure 6.3).

Figure 6.3: Status Bar

Table: 6.4 MPLAB Status Bar

Title	Typical Entry	Description	Result from Double Clicking
Line No., Column–Windows Open MPLAB Version Number–No Open Windows	Ln 1 Col 1 2.00.00	Displays current line number and column in file. Current Version Number	Opens Goto Line Dialog No Action
Lines in File	72	Displays number of lines in current text file.	No Action
File Modified	#	Displays # Symbol if File Has Been Changed Since Opening	No Action
Write/Read Only	WR	Displays Write/Read Only Status. WR = Editable File RO = Read Only File	Toggles between write and read only.
Text Wrap	No Wrap	Displays current wrap mode and wrap column if text wrap is on. Example 1: NoWrap Example 2: WR 72 Useful for text files. Use *Options > Current Editor Modes* to change wrap column.	Toggles between wrap and no wrap. No Wrap Wrap at Column 72

Title	Typical Entry	Description	Result from Double Clicking
Insert/Strikeover	INS	Toggles typing mode between insert and strikeover. INS = Insert Characters OVR = Type over characters	Toggles between INS and OVR.
Current Processor	PIC16C61	Displays the currently selected processor.	No Action
Current Program Counter	pc:0x5f	Displays the current program counter.	Opens Change Program Counter dialog.
Current w Register Value	W:0x00	Displays current w register value.	No Action
Status Bits	ov Z dc c	Upper Case = Set (1) Lower Case = Reset (0)	No Action
Global Break Enable	Bk On	Displays current status of Global Break Enable.	Toggles Global Break Enable On and Off.
Current Development Mode	Sim	Displays Current Development Mode. Examples: EO = Editor Only Sim = Simulator – MPLAB-SIM Si = Simulator – SIMICE ICE = Emulator – MPLAB-ICE Em = Emulator – PICMASTER	Displays Development Mode Dialog.
Processor Frequency	4MHz	Displays current processor frequency	Opens processor clock dialog
Current Tool Bar	Edit	Displays current tool bar	No Action

6.4 MPLAB Menus and Functions

FILE
New
Open...
View...
Save
Save As...
Save All
Close
Close All
Import
  Download to Memory...
  Download to Target Memory...
  Copy from Target Memory
Export
  Save Trace Buffer...
  Save Hex File...
Print...
Print Setup...
Exit
(*Most Recently Used Files*)

PROJECT
New Project...
Open Project...
Close Project
Save Project
Edit Project
Make Project
Build All
Build Node
Install Language Tool...
(*Most Recently Used Projects*)

EDIT
Undo
Cut
Copy
Paste
Select All
Select Word
Delete Line
Delete EOL
Goto Line...
Find...
Replace...
Repeat Find
Repeat Replace
Match Brace
Template
  Attach File
  Detach File
  Create File
  Save File
  Insert
  Edit
  New
  Store
  Store As
  Delete
  Insert Mark
  Find Mark
Text
  Transpose
  Upper Case
  Lower Case
  Indent
  Un-Indent

DEBUG
Run
  Run
  Reset
  Halt
  Halt Trace
  Animate
  Step
  Step Over
  Update All Registers
  Change Program Counter...
Execute
  Execute an Opcode...
  Conditional Break...
Simulator Stimulus
  Asynchronous Stimulus...
  Pin Stimulus
    Enable...
    Disable
  Clock Stimulus...
  Register Stimulus
    Enable...
    Disable
Center Debug Location
Break Settings...
Trace Settings...
Clear All Points...
Clear Program Memory
System Reset
Power-On-Reset
(*Programmer Menu*)

OPTIONS
Development Mode...
Window Setup
  Save Setup...
  Load Setup...
  Default Configuration
Default Editor Modes...
Current Editor Modes...
Reset Editor Modes
Key Mappings...
Environment Setup...
Colors...
Processor Setup
  Hardware...
  Clock Frequency...
Programmer Options
  Select Programmer
  Communications Port Setup

TOOLS
DOS Command to Window...
Repeat DOS Command to Window
(*Verify Emulator...*)
(*Configure Emulator...*)

WINDOW
Program Memory
Trace Memory
(EEPROM Memory)
(Calibration Data)
Absolute Listing
Stack
File Registers
Special Function Registers
Show Symbol List
Stopwatch...
Project
New Watch Window
Load Watch Window...
Modify...
Tile Horizontal
Tile Vertical
Cascade
Iconize All
Arrange Icons
(*Open Windows*)

HELP
Release Notes
(*Tool Release Notes*)
MPLAB Help
Editor Help
MPASM Help
(*Tool Help*)
About

6.5 File Menu

Several of the options under the File menu are standard to Windows. They are:

- New
- Open...
- View...
- Save
- Save As...
- Save All
- Close
- Close All

Please refer to Windows documentation for more detail on these options.

Import and Export have special meanings for MPLAB and are discussed in greater detail below.

Print, Print Setup and Exit are similar to the Windows standard, but differ slightly and are discussed later in this section.

6.5.1 Import

The *File > Import* functions allow you to move data from a PC file to the emulator and into target memory or into the simulator memory. This function also allows you to transfer data from the target into emulator memory for debugging.

6.5.1.1 Download to Memory

Select *File > Import > Download to Memory* to display the dialog box for selecting a file to download. The file you select is downloaded to the emulator memory or simulator memory. The file must be a valid hex file.

6.5.1.1.1. PIC12CXX, PIC14000, PIC16C5X, or PIC16CXX Devices

If the current target processor belongs to the PIC12CXX, PIC14000, PIC16C5X or PIC16CXX device family, then the file should be in the Intel® 8-bit HEX file format (INHX8M). If the object file is successfully downloaded, then the symbols are loaded automatically from the corresponding *.COD file if it exists. The default extension for the object code is *.HEX.

6.5.1.1.2. PIC17CXX Devices

If the current target processor belongs to the PIC17CXX device family, then the file should be in the Intel extended HEX file format (INHX32). If the object file is successfully downloaded, then the symbols are loaded automatically from the file ∗.COD. The default extension for the object code file is.HEX.

If you do not have a *.COD file, you may want to turn off source tracking in the *Options > Environment Setup* dialog.

Note: Use INHX32 if your application addresses memory beyond 32K.

6.5.2 Export

6.5.2.1 Save Trace Buffer

Select *File > Export > Save Trace Buffer* to display the Save Trace Buffer dialog box. The simulator trace buffer is saved to the selected file.

Note: If you have a full trace buffer with maximum length labels, saving the trace for the MPLAB-ICE emulator could require over 7 MB (typically only 2 MB). Saving the complete trace buffer as it appears in the Trace Window (address, data, disassembled code and external logic probe lines) can exceed 1 M for the MPLAB-SIM simulator.

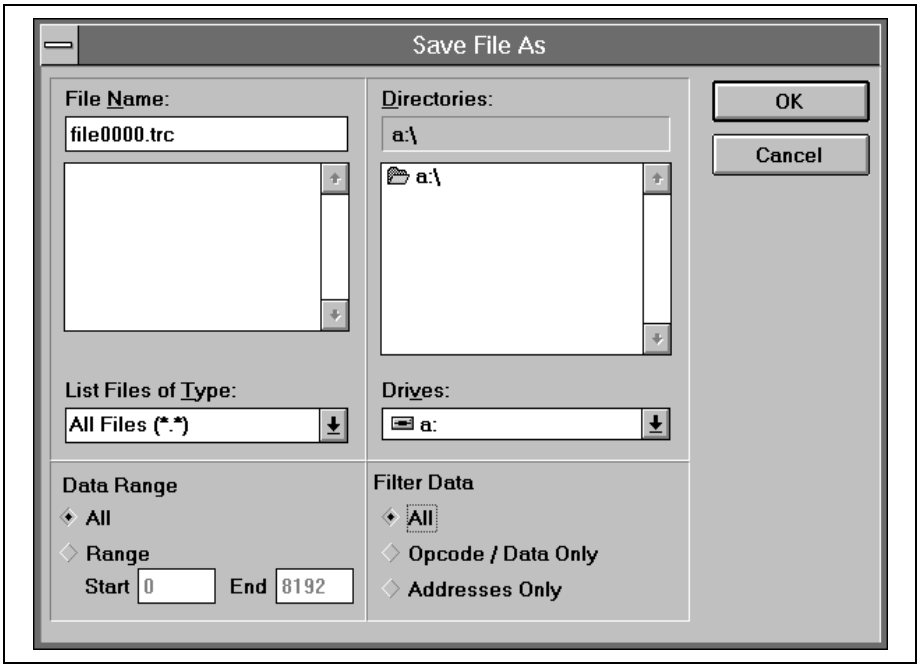


Figure 6.5: Save Trace Buffer Dialog

Range	Select the range (0 to 8191) of the trace buffer that you want to save. Enter the desired value for the Start line number and for the End line number.
Filter Data: All	Writes the complete trace buffer to the selected file.
Filter Data: Opcode (Data Only)	Saves Opcode / Data only. (For PIC17CXX external read/write cycles) **Available only with the MPLAB-ICE emulator**
Filter Data: Address Only	Saves Address only.

6.5.2.2 Save Hex File

Select *File > Export > Save Hex File* to display the Save Hex File dialog box.

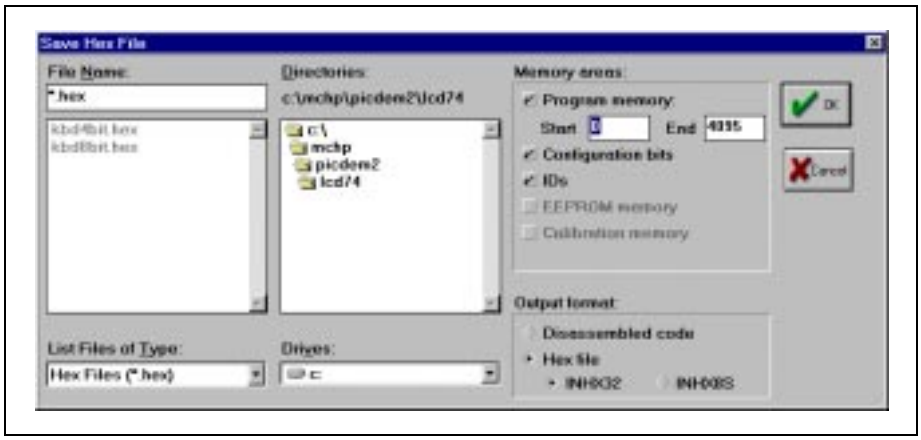


Figure 6.6: Save Hex File Dialog

Memory areas	Determine which memory areas you want to save in the hex file.
Program Memory	Select to save program memory. Also state the Start and End address of program memory to save (Entire range is the default).
Configuration Bits	Select to save configuration bit settings.
IDs	Select to save ID information.
EEPROM memory	Select to save EEPROM memory, if applicable.
Calibration memory	Select to save the calibration memory if applicable.
Output format	Determine the format of the output file.
Disassembled code	Select to save in disassembled code format.
Hex Code	Select either hex code format INHX32 or INHX8X format.

6.5.3 Print (Ctrl+P)

Click *File > Print* to print some or all of the current file on your currently selected printer.

The Print command starts a dialog to specify the details of how the file is to be printed and on which printer.

By default, MPLAB uses the same printer that you specified the last time you printed a file. The default details of the printing, such as line folding and page headers, are taken from the window modes set on the current window.

The dialog box shows you the name of the printer that MPLAB is currently using—if you haven't specified otherwise, this will be your system default printer.

The file is printed using the same tab width that is being used in the window, and with line numbers if you have line numbering turned on. You can configure both these settings with the *Options > Current Editor Modes* menu command.

6.5.3.1 Print Current File Options

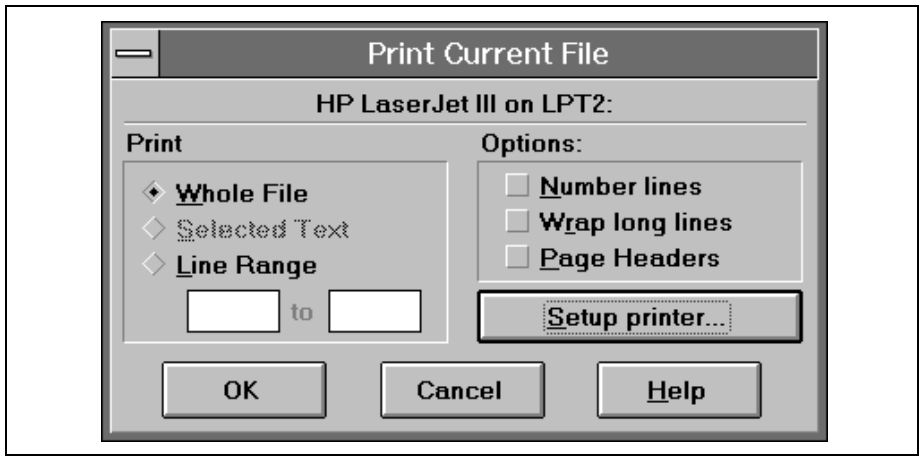


Figure 6.7: Print Current File Dialog

Setup Printer	Click the Setup Printer button to change printers, printer font, or page margins.
Whole File	Click Whole File (the default) to print entire file.
Selected Text	Click Selected Text to print only the text that is highlighted. This option will not be available if you don't have any text highlighted.
Line Range	Click Line Range and fill in the start and end line numbers to print a range of lines. You can use the words “start” and “end” to represent the first and last lines of the file.
Number Lines	Select Number Lines to print lines with numbering.
Wrap Long Lines	Select Wrap Long Lines to fold lines too long to fit the page rather than being truncated.
Page Headers	Select Page Headers to start each page with a header giving the file name and other information.

6.5.4 Print Setup

Click *File > Print Setup* to set up details of the printer that MPLAB will use, run the printer's setup dialog, and select the font. MPLAB records the values you set with the Print Setup option for the selected printer. Thus, you can have different settings for different printers. These values become the defaults.

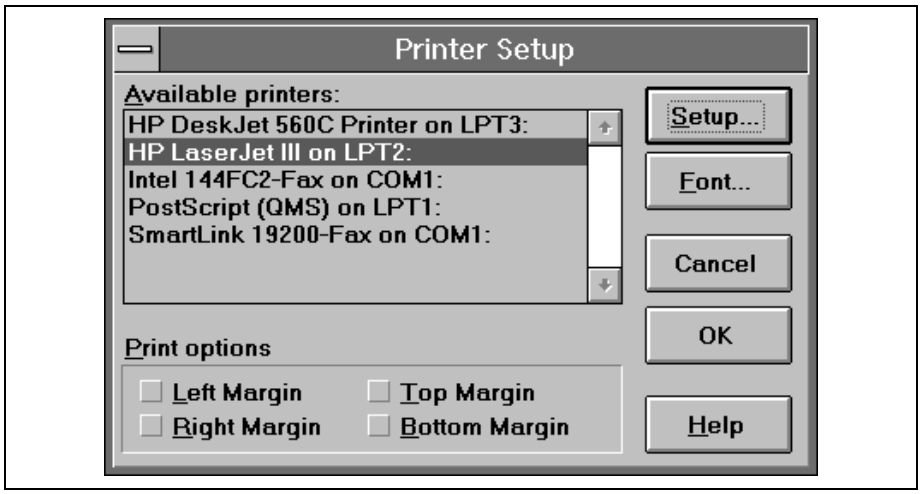


Figure 6.8: Printer Setup Dialog

File > Print Setup lets you specify:

- Which printer to use
- What page margins are to be applied when you print on this device
- What font to use with this printer

File > Print Setup also allows you to run the printer's own setup dialog to set device-specific information.

Available Printers	If you want to use a different printer than the one that is highlighted in the Available Printers list, scroll the list and click the left button on the printer name.
Print Options	To change the margins used on each page, check or uncheck the boxes in the Print Options area.
Setup	To run the highlighted printer's own setup dialog, click **Setup**.
Font	To change the printer font that MPLAB will use for the highlighted printer, click **Font**.

Because MPLAB is a text editor rather than a word processor, you're restricted to choosing fixed pitch fonts, where the characters are all the same width.

If you run the printer's own setup dialog after selecting a font, you may find that the font is no longer available; some printers offer different fonts in different operating modes.

6.5.5 Exit (Alt+F4)

Click *File > Exit* to terminate your MPLAB session.

This command will end your MPLAB session. If any of the files you're working with have changed and you haven't saved the changes to disk, MPLAB will prompt you for each one in turn. You can choose to save the changes, discard them, or cancel the operation. You will also be prompted to save the current project.

6.6 Project Menu

For detailed information on Project menu items, refer to Chapter 4.

6.7 Edit Menu

For detailed information on Edit menu items, refer to Part 2.

6.8 Debug Menu

The Debug menu contains all the options you would use when debugging your code.

6.8.1 Run

The Run menu options allow you to control the execution of your firmware in the target processor.

6.8.1.1 Run (F9)

Debug > Run > Run takes the processor out of the halt state and puts the processor into execution until a break point is encountered or until Halt is pressed.

Execution starts at the current program counter (as displayed in the status bar). The current program counter location is also highlighted in the Program Memory window. While the processor is running, the Step and Run buttons are disabled.

6.8.1.2 Reset (F6)

Debug > Run > Reset issues a reset sequence to the target processor. This issues an $\overline{\text{MCLR}}$ to reset the Program Counter to the reset vector. If the processor is running it will continue running from the reset vector address.

6.8.1.3 Halt (F5)

Debug > Run > Halt forces the processor into the halt state. When you click **Halt**, the processor is forced into a Halt state (Program Counter is stopped) and the processor status information is updated.

6.8.1.4 Halt Trace (Shift+F5)

Debug > Run > Halt Trace halts the trace buffer from capturing data but allows the processor to continue running. Refer to your emulator documentation for more information on Halt Trace.

6.8.1.5 Animate

Debug > Run > Animate causes the simulator to actually execute single steps while in run mode, updating the values of the registers as it runs.

Use Animate mode to view the changing registers in the Special Function Register window or in the Watch windows. Animate mode runs slower than the Run function, but allows you to view changing register values.

6.8.1.6 Step (F7)

Debug > Run > Step single steps the processor. This command executes one processor instruction (single or multiple cycle instructions) and then puts the processor back into halt state. After execution of one instruction, all the windows are updated with the current state of the processor. While the processor runs in real time, MPLAB ignores the Step button.

6.8.1.7 Step Over (F8)

Select *Debug > Run > Step Over* to execute the instruction at the current program counter location. At a call instruction, Step Over executes the called subroutine, and halts at the address following the call.

6.8.1.8 Update All Registers

Debug>Run>Update All Registers updates all registers for the current instruction.

6.8.1.9 Change Program Counter

Debug > Run > Change Program Counter allows you to change the current program counter.

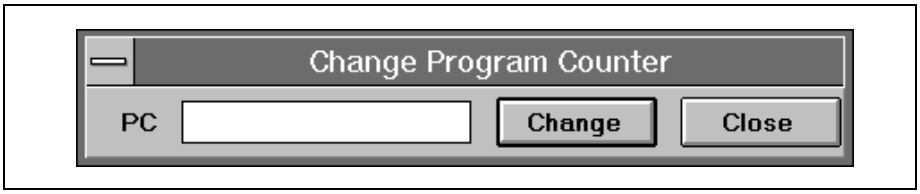


Figure 6.9: Change Program Counter Dialog

PC	Enter desired Program Counter address.
Change	Click **Change** to change to the new program counter address. The processor must be halted for the change to take effect.
Close	Exits from the Change Program Counter dialog box.

6.8.2 Execute

The Execute menu options allow you to control the polled execution of your firmware in the target processor.

6.8.2.1 Execute an Opcode

Select *Debug > Execute > Execute an Opcode* to execute a single instruction or a series of instructions without modifying the object code or program memory. After executing the instruction, you may resume executing from the current program memory location.

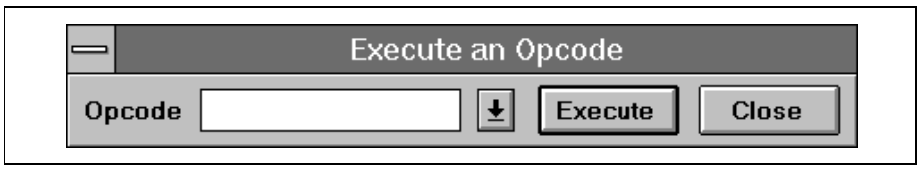


Figure 6.10: Execute an Opcode

Opcode	Enter the instruction as an opcode (in hex digits) or enter a symbolic instruction (such as ADDWF 0x19).Click the Opcode list to display the last eight commands. After executing a command, MPLAB highlights the command so you can type in a new instruction. (MPLAB tracks the instructions you enter so you don't get two copies of the same instruction in the opcode list.)
Execute	Click **Execute** to execute an instruction without modifying the current location of the program counter.

6.8.2.2 Conditional Break

Select *Debug > Execute > Conditional Break* to display a dialog box that performs an automated single stepping of the processor. Execution starts upon pressing the **Start** button and continues executing until meeting the condition presented in the dialog or until you click **Halt**.

For additional information on conditional breaks, see Section 5.8.

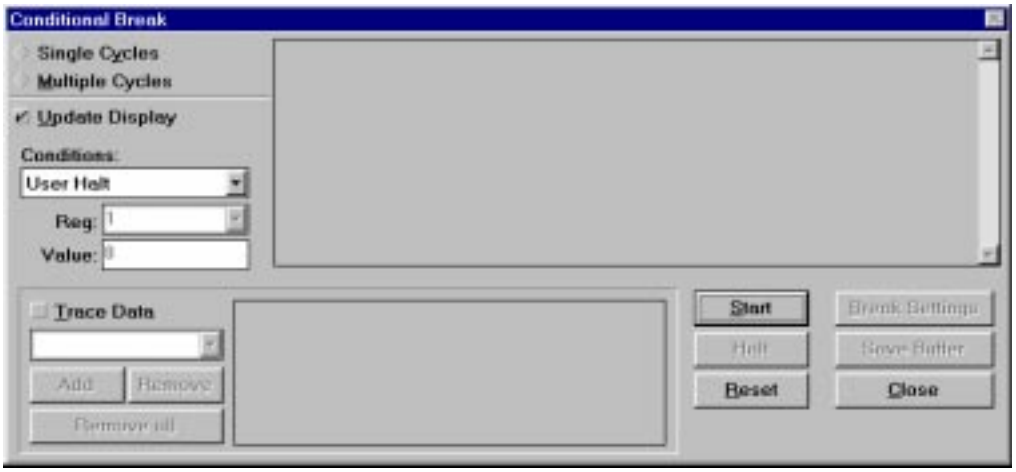


Figure 6.11: Conditional Break Dialog

Single Cycle	Checks condition at every instruction. The Single Cycle option samples conditions after every instruction, allowing you to catch a particular condition.
Multiple Cycles	Checks condition only at user-defined break points. The Multiple Cycles option runs at real time except for the halt at break points. This option allows interrupts to be serviced.
Update Display	Executes the conditional break but does not update the disassembled code in the window. MPLAB stores the last 1000 lines.

Conditions — The condition that you set up is tested on any register location and an 8-bit constant value that you enter. You can test for the following conditions:

1. User Halt When processor is running, click **Halt** to stop the processor.
2. Number of Cycles Enter the number of cycles in the **Value** box.
3. Register Value Conditions

RAM Addr Data Value Entered	==	Equals	Value
RAM Addr Data Value Entered	<>	Not Equals	Value
RAM Addr Data Value Entered	>	Greater Than	Value
RAM Addr Data Value Entered	<	Less Than	Value
RAM Addr Data Value Entered	>=	Greater or Equal	Value
RAM Addr Data Value Entered	<=	Less or Equal	Value

If the tested condition is true, execution stops before executing the next instruction. The next instruction in the Program Memory window will be highlighted.

Caution: All register values are treated as 8-bit unsigned values. Therefore, the condition <0 will never be true.

Reg — Register Condition Enter a RAM address location where you want to test against the data value at that location. The location that you enter must be a file register location.

Value — Enter an 8-bit value in the Value box that you want to test against.

Trace Data — Trace Data samples specific registers at each time the processor is halted and displays the register data in the list.

Add, Remove, Remove All — Edits the list of data variables sampled at each break point.

Start — Starts execution and continues to execute single steps until the condition is met or until you press the **Halt** button.

Halt — Halts execution of the Conditional Break.

Reset — Resets the processor.

Break Settings — Opens the Break Point Settings dialog.

Save Buffer — Opens the Save File dialog to save information from the list box in a *.TB file.

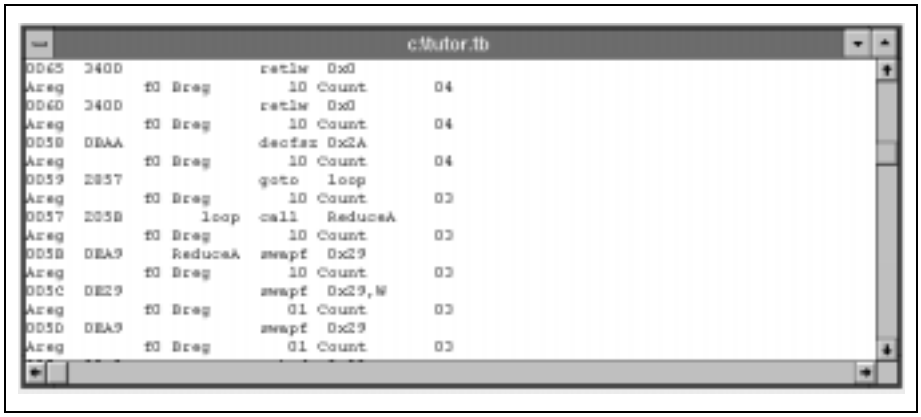


Figure 6.12: *.TB File

Close — Exits out of the Conditional Break dialog.

6.8.3 Simulator Stimulus

MPLAB-SIM simulator functions allow you to set up regular clock stimulus signals and allow the simulator to respond to events from files on your PC. The files can be written with the MPLAB Editor or any other suitable text editor or word processor and should be saved in the same directory as the current project.

Simulator functions are:

- Asynchronous Stimulus
- Pin Stimulus
- Clock Stimulus
- Register Stimulus

For a detailed discussion of simulator stimulus, see Section 5.10.

6.8.4 Center Debug Location

Select *Debug > Center Debug Location* to move the current program counter to the middle of the debugging window.

This function works on the Source Code Window, the Program Memory Window, and the Absolute Listing Window.

6.8.5 Break Settings

Select *Debug > Break Settings* to display the Break Point Settings dialog box for defining up to 16 named break point ranges.

After entering a break point title, start address, and end address, click ✔ or press **Enter** to accept the break point range definition.

For more functional information on break points, see Section 5.7.

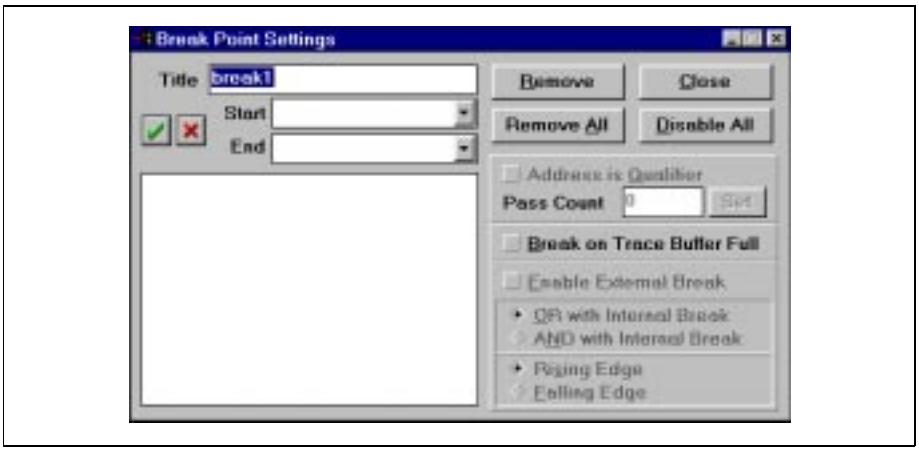


Figure 6.13: Break Settings Dialog

6.8.5.1 Saving Break Point Settings

Break points are saved as part of the project.

6.8.5.2 Break Point Range Definition

Title — Enter a unique title (up to 32 characters) for each break point range. MPLAB accepts the underscore character but does not allow spaces.

MPLAB automatically enters a default, unique title if you choose to not enter a title. The break point range requires a title.

Start, End — Enter a Start and End address in hex or as a label for the break point range. The address range is restricted to the valid address range of the target processor. You can just enter the start address and MPLAB will fill in the same end address for a break point on a single location (rather than a range).

You can enter the Start and End values as addresses or labels. If you use labels, MPLAB allows you to modify the labels by using offsets: "MAIN+2," EXECTIMR-10." when you use labels and recompile a project (and the label moves due to the compilation) MPLAB assigns the break points to the new address range.

You can use an existing break point range item as the starting point for entering a new break point range. Click on a desired item in the list box. Type in a new title. Then click ✔ or press **Enter** to accept the defined break point range.

6.8.5.3 Break Point Settings

✔	Click the check box to accept the selected range and add the range to the list of ranges. Entering a range enables break points in that range in program memory. Disabled break point ranges will not clear break points included in other ranges.
X	Click **X** to not add the selected range to the list and to clear the Title, Start, and End fields.
Break Point List	The list box allows you to enter up to 16 break point ranges. On selecting a range, the break point Settings dialog box displays Start, End, and Title to allow you to edit the start and end address. The list box contains the following elements:

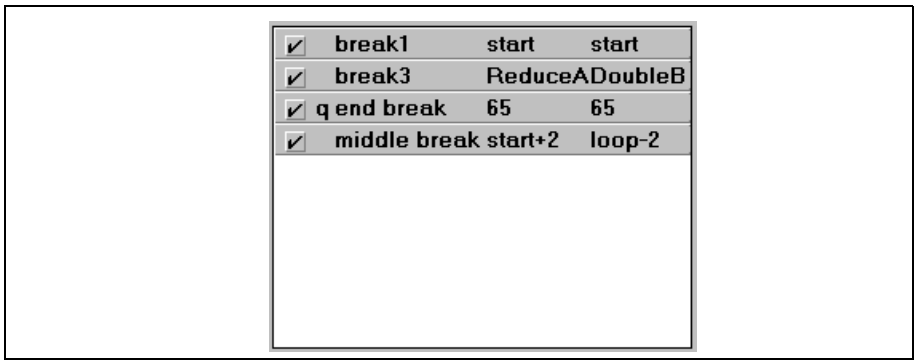


Figure 6.14: Break Point List

Enable	This switch enables the break point range in program memory.
q	Qualifier point designator. When a range item displays the letter q, the address range is a Pass Count address.
Title, Start, End	Click an item in the list box to display the title, start, and end data. You can then edit the start and end address or change the title to enter a new break point range.

Remove	Removes the selected break point range. If an item is not selected, Remove does nothing.
Remove All	Removes all break point ranges from the list.
Close	Closes the Break Point Settings dialog box.
Disable All	Disables all break point ranges.

6.8.5.4 Pass Counter

Address is Qualifier	You can assign a Pass Count qualifier address to either break logic or trace logic. You must have an element in the list selected. With an element selected, the dialog will enable the Address is Qualifier check box. When you check Address is Qualifier, you are assigning the pass counter to all addresses in the selected range. Usually it is useful to set only a single address for the pass counter. After selecting Address is Qualifier, MPLAB enables the Pass Count Edit Box and the Set button. With the Pass Count Edit Box enabled, you can set the pass counter to a desired value (up to 65,534).
Pass Count	Type in a pass count value. The pass count value defines the number of times the program can pass a qualifier address before halting the processor. Each time the program encounters an address that has been set as a qualifier, it decrements the pass count. When the pass count reaches 0, it halts the processor.
Set	Click **Set** to enter a pass count value. The displayed Pass Count value does not download the value to the emulator's pass counter until you click on Set.

6.8.5.5 Break on Trace Buffer Full

Break on Trace Buffer Full	Enables break on trace buffer full. Check this box (✔) to halt the processor when the trace buffer is full. The trace buffer is full when it captures 8K instructions/cycles.

6.8.6 Trace Settings

Select *Debug > Trace Settings* to display the Trace Point Settings dialog box for defining up to 16 named trace point ranges.

For more functional information on trace points, see Section 5.7.

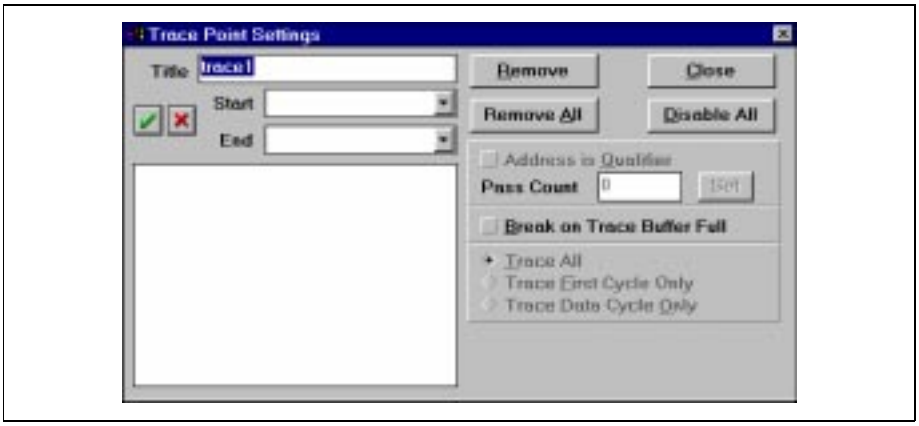


Figure 6.15: Trace Settings

Note: 1. The pass counter qualifier address can be assigned either to the Trace logic or break logic.

2. The pass counter qualifier address can be set either in this menu or in the Settings dialog.

Trace All — Lets you trace each enabled trace address location. This option displays the trace buffer exactly the way the processor fetches and executes instructions at each clock cycle.

To save captured data cycles into a file for plotting or analysis, use *File > Export > Save Trace Buffer*.

A simulator trace memory buffer looks like Figure 6.16. For information on emulator display, see the documentation for that emulator.

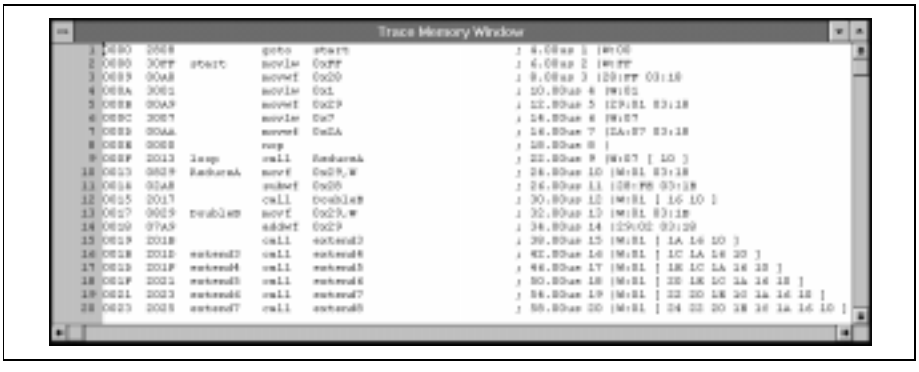


Figure 6.16: Trace Memory Window

6.8.7 Clear All Points

Select *Debug > Clear All Points* to clear all break and trace points.

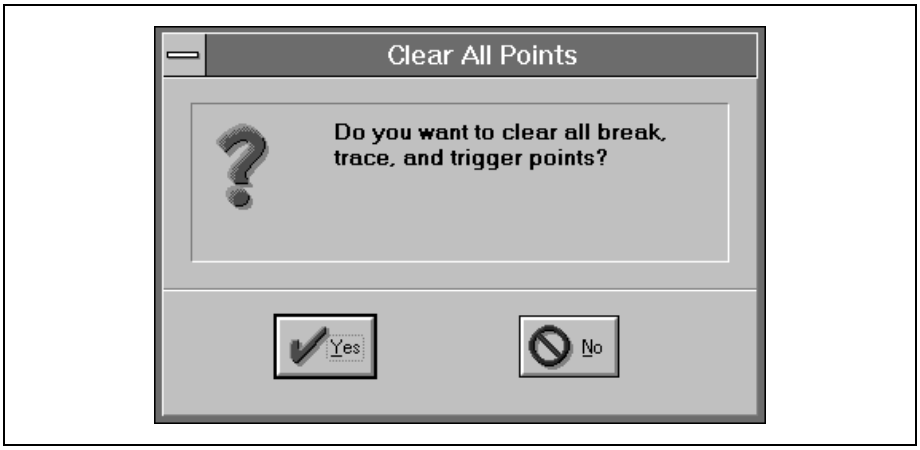


Figure 6.17: Clear All Points Message Box

6.8.8 Clear Program Memory (Ctrl+Shift+F2)

Select *Debug > Clear Program Memory* to clear program memory to all 0xFF's. This function sets all program memory bits to ones.

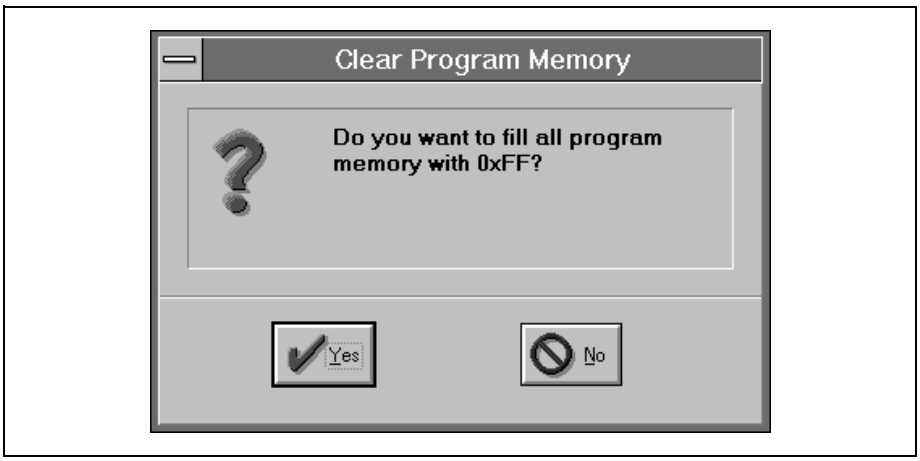


Figure 6.18: Clear Program Memory Message Box

6.8.9 System Reset (Ctrl+Shift+F3)

Select *Debug > System Reset* to reset the entire emulator system including the MPLAB-ICE emulator hardware (if connected), software and the target processor. System Reset performs the same initialization that is performed when MPLAB is first entered.

Note: To perform a processor reset ($\overline{\text{MCLR}}$), select *Debug > Run > Reset*.

Note: Always power down the emulator pod when changing probes, and then run a system reset. If you do not run a system reset after changing probes, MPLAB will not be properly configured for the new probe.

6.8.10 Power-On-Reset (Ctrl+Shift+F5)

Select *Debug > Power-On-Reset* to display the Power-On-Reset dialog box for selecting a POR option.

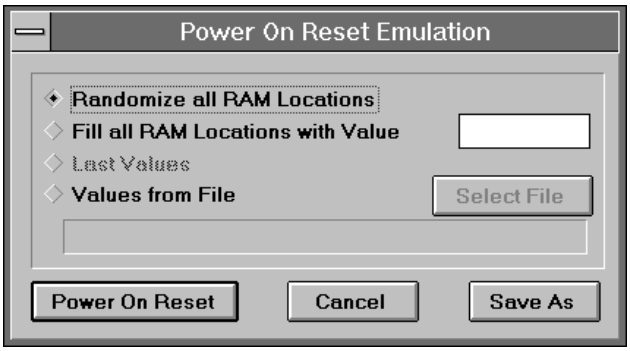


Figure 6.19: Power-On-Reset Dialog

Power-On-Reset allows you to emulate the power-on-reset function of the target processor.

Often, uninitialized registers can lead to a program malfunction that is hard to track down. POR will simulate the action of randomizing registers when the application first starts up. If the application misbehaves sometimes, the Power-On-Reset function may help you isolate the problem.

Note: Power-On-Reset is implemented on a device by tying the $\overline{\text{MCLR}}$ pin to V_{DD}. The POR signal resets registers as described in the Microchip Databook for the specific device.

The Power-On-Reset dialog can be used for the following functions:

- Randomize registers that have an unknown value at POR.
- Fill registers with a value, or clear registers.
- Set registers to the POR Condition shown in the Microchip Databook for the respective part. The Fill with Value POR function will not affect the registers that have specifically defined reset values.
- Save current POR values to a file.
- Load POR values from a file.

Note: Program memory and break points are undisturbed when using the POR dialog.

Randomize	Select **Randomize** to enter random values into registers that have an unknown value at POR.
Fill with Value	Type a fill value in the Enter Value box that you want to enter into registers at POR.
Last Values	Select **Last Values** to enter the last randomized or filled values into the device at POR.
Power-On-Reset	Click **Power-On-Reset** to reset and set selected register values.
Cancel	Click **Cancel** to close the Power-On-Reset dialog box without performing a POR.
Save As	Opens a dialog box to enter the name of a file (*.POR) to save Power-On-Reset settings.
Values from File	To load data values from a file, click **Values from File** and then click **Select** to open a dialog box to enter the name of an *.POR file containing values to load at power-on-reset.

6.9 Programmer Menu

To select a programmer, open the Select Programmer dialog from the *Options > Programmer Options > Select Programmer* menu item. Once selected, MPLAB must restart before the programmer options are available. Then a menu specific to that programmer will appear on the menu bar.

For more information on the operation of individual programmers, refer to the documentation for that programmer.

PICSTART Plus Menu

Enable/Disable Programmer	Enable or Disable the programmer.
Program/Verify	Program the device or verify that the device was programmed properly.
Read Device	Read the program and configuration bits of the device.
Blank Check All	Check that all memory and configuration bits are blank.
Blank Check OTP	Check that OTP program memory is blank.
Display Error Log	Display the error log on the screen if any errors occur.
Erase Program Memory	For electrically erasable devices, erase the program memory.
Erase Configuration Bits	For electrically erasable devices, erase the configuration bits.
Reset Programmer	Reset the programmer.

PRO MATE Menu

Enable/Disable Programmer	Enable or Disable the programmer.
Program/Verify	Program the device or verify that the device was programmed properly.
Read Device	Read the program and configuration bits of the device.
Blank Check All	Check that all memory and configuration bits are blank.
Blank Check OTP	Check that OTP program memory is blank.
Display Error Log	Display the error log on the screen if any errors occur.
Erase Program Memory	For electrically erasable devices, erase the program memory.
Erase Configuration Bits	For electrically erasable devices, erase the configuration bits.
Reset Voltages	Reset the programmer.

Transfer to PRO MATE	Transfer device and firmware information to PRO MATE.
Transfer from PRO MATE	Transfer device and firmware information from PRO MATE.
Generate SQTP File	Generate an SQTP file for programming devices.
Load SQTP File	Load an SQTP file from a device.
Download Firmware	Download the latest PRO MATE firmware.
Establish Communications	Establish or reestablish communications with the programmer.

6.10 Options Menu

6.10.1 Development Mode

Select *Options > Development Mode* to change the current Development Mode setting.

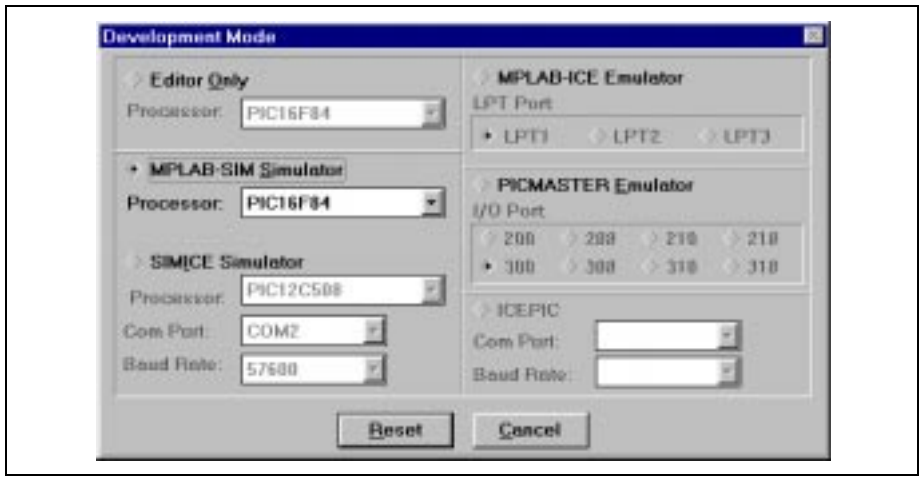


Figure 6.20: Development Mode Dialog

Editor Only	Selects the Editor Only mode and displays EO on the Status Bar. In Editor Only mode, all emulator and simulator functions are disabled. You can only edit, compile, and perform project management operations.
MPLAB-SIM Simulator	Selects the MPLAB-SIM simulator mode. Processor: Selects the processor that you are simulating.
SIMICE Simulator	Selects the SIMICE Simulator (if connected). Processor: Selects the processor that you are simulating. Com Port: Selects the communications port to which SIMICE is connected. Baud Rate: Selects the baud rate for communicating with SIMICE.

MPLAB-ICE Emulator	Selects the MPLAB-ICE Emulator mode (if connected) and displays Ice on the Status Bar. LPT Port: Selects the parallel port to which MPLAB-ICE is connected.

WARNING 	**Do not** select the MPLAB-ICE Emulator mode if any other device (i.e., printer, scanner, zip drive) is installed on the parallel port or **permanent damage to that device** may result. See the *MPLAB-ICE User's Guide* (DS51159) for more details.

PICMASTER Emulator	Selects the PICMASTER Emulator mode (if connected) and displays Em on the Status Bar. I/O Port: Selects the I/O address of the PICMASTER interface board in your PC. Your I/O address setting will correspond to your current dip switch settings on your PC Interface Board.
ICEPIC	Selects the ICEPIC Emulator (if connected). ComPort: Selects the communications port to which ICEPIC is connected. Baud Rate: Selects the baud rate for communicating with ICEPIC.
Reset	Click **Reset** to reinitialize the emulator hardware (if connected). If the message "Unable to find Emulator System..." displays, then the selected port may not be available. This option is useful mainly after changing the processor probe.

Note: This reset function is a manual reset that you execute and is the same as the *Debug > System Reset* option. This reset function is different from Power-On-Reset.

Cancel	Select **Cancel** to cancel your selection and exit this display.

6.10.2 Window Setup

6.10.2.1 Save Setup

Select *Options > Window Setup > Save Setup* to save the current configuration to a file. The default extension of the configuration file is *.CFG.

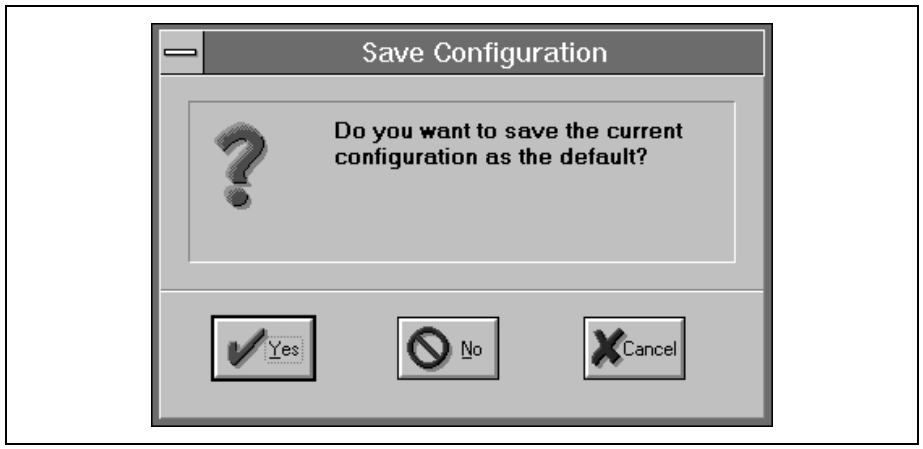


Figure 6.21: Save Setup as Default Message Box

Save Setup only saves the currently open windows—not the break, trace, and trigger points.

Yes — If you answer **Yes** to the question, "Do you want to save the current configuration as the default?" MPLAB will save the current configuration as the default user configuration that MPLAB loads during start-up.

No — If you answer **No**, MPLAB displays the Save Configuration dialog box. At this point, enter the name and path where you want to save the setup.

Cancel — Select **Cancel** to exit Save Setup without saving.

6.10.2.2 Load Setup

Select *Options > Window Setup > Load Setup* to load a configuration setup from a file previously stored by the *Options > Window Setup > Save Setup* command. Select the file that you want to read from the dialog box and press **Enter**. The default extension of the setup file is *.CFG.

6.10.2.3 Default Configuration

To load the default configuration select one of the following:

Options > Window Setup > Default Configuration, or

Options > Environment Setup and click on Load Default Configuration.

6.10.3 Key Mappings

Select *Options > Key Mappings* to display the MPLAB key mappings. These key mappings enable you to perform the most common operations quickly. You may use the existing key mappings or modify the key mappings to meet your specific needs for your current project.

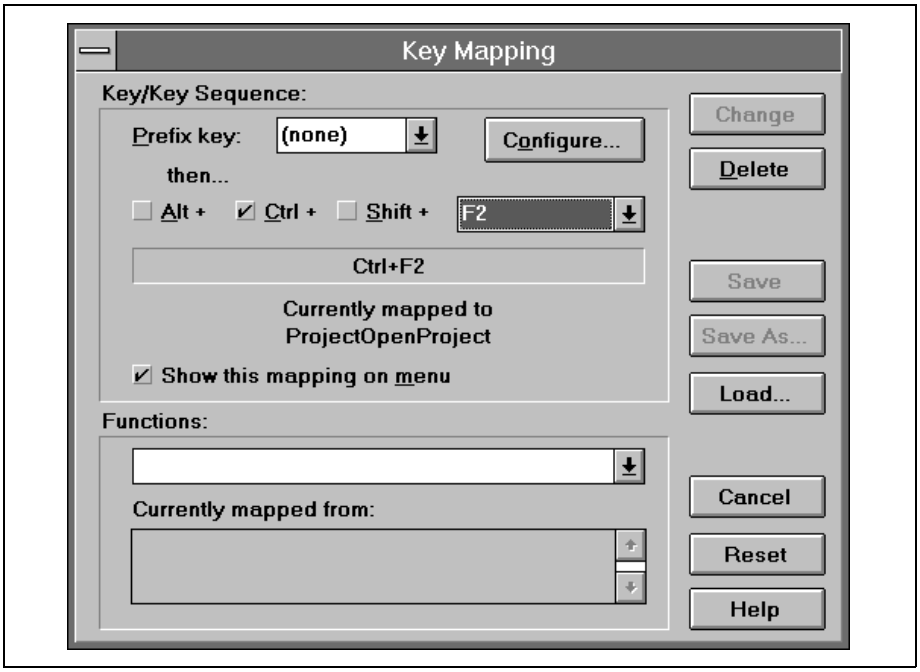


Figure 6.22: Key Mapping Dialog

By default, no prefix keys are enabled. This means that only single keys are available for mapping.

MPLAB uses the binary initialization file, MPLAB.KEY, in the MPLAB directory to record the key mapping values that will be carried from one session to another.

Functions — The Functions pull-down menu displays the MPLAB key mappings. (Refer to the MPLAB Key Mapping Functions appendix for details.)

6.10.4 Environment Setup (Ctrl+F7)

Options > Environment Setup opens a dialog box for changing the screen font, toolbar, global switches, and the symbol display width.

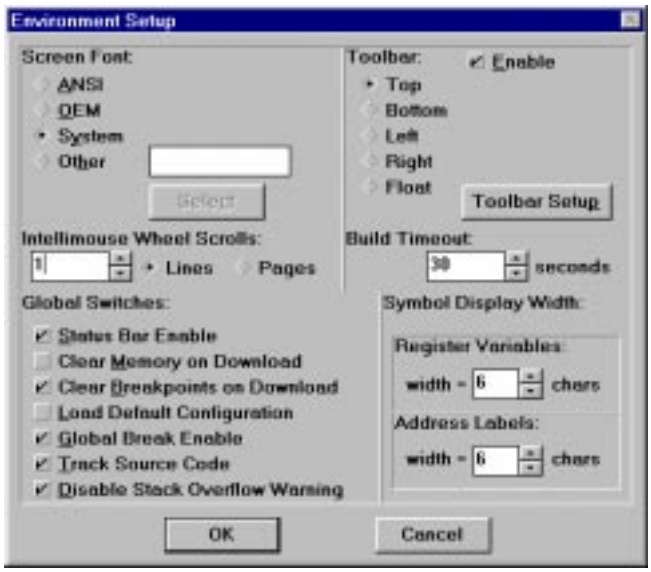


Figure 6.23: Environment Setup Dialog

6.10.4.1 Screen Font

The screen font option allows you to select a fixed point font for MPLAB screen displays.

6.10.4.2 Toolbar

The Toolbar options allow you to select a location for the toolbar on the screen. Available locations are: Top, Bottom, Left, Right, and Float.

Enable — Select **Enable** to display the toolbar.

6.10.4.3 Toolbar Setup

The toolbar setup dialog box allows you to redefine the operations that the toolbar icons perform.

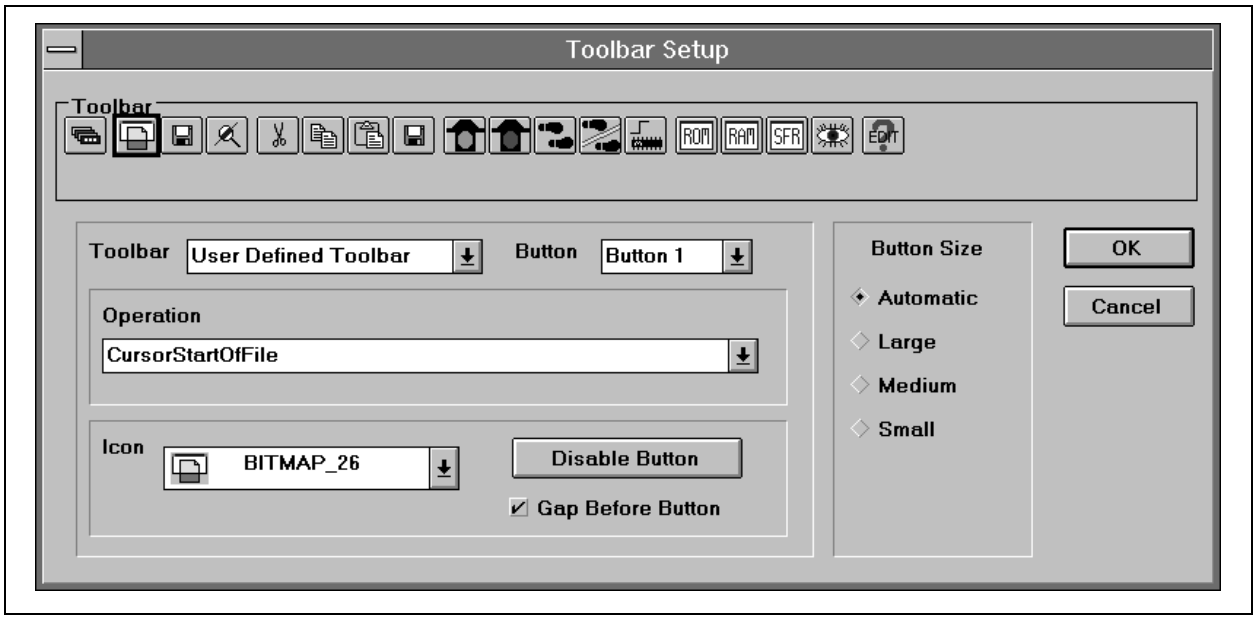


Figure 6.24: Toolbar Settings Dialog

Toolbar	Selects which toolbar to edit. There are four possible selections: Edit, Debug, Project, and User Toolbar.
Button	Selects an Icon button location. The toolbar has 16 available locations.
Operation	Selects the operation that MPLAB will perform for the selected button location. Refer to the MPLAB Key Mapping Functions appendix for a list of available operations.
Icon	Selects an icon to display in the selected button location.
Disable Button	Disables the icon button at the selected location on the toolbar.
Gap Before Button	Inserts a small gap before the icon button.
Button Size	Changes the icon button size. Button size options are: Automatic, Large, Medium, and Small.
Ok	Defines the toolbar as currently shown in the Setup dialog box.
Cancel	Returns the toolbar to its previous state.

6.10.4.4 Global Switches

The Global Switches allow you to turn these user selections on or off.

Status Bar Enable	Turns the Status Bar on and off.
Clear Memory on Download	When selected, this global switch clears memory before MPLAB downloads to the emulator. This function sets all program memory bits to one.
Clear Break Points on Download	When selected, removes all break points, trace points, trigger points, and pass count addresses on download to the emulator.
Load Default Configuration	When selected, MPLAB loads the default user configuration at start-up. To change the default window setup, open the windows you want to load at start-up, click *Options > Window Setup > Save Setup* and click **Yes**. The MPLAB window setup may contain any available MPLAB window. MPLAB.CFG is the default user configuration file.
Global Break Enable	When selected, enables all break points. If Global Break Enable is not selected, then all break points are disabled. Global Break Enable is also available from the Status Bar.
Track Source Code	When selected, MPLAB updates the current line in the source code when single stepping. You may wish to turn this feature off if you have a *.HEX file, but no *.COD file.
Disable Stack Overflow Warning	Disables the warning dialog that the Status register has overflowed.

6.10.4.5 Intellimouse® Wheel Scrolls

If you have an Intellimouse, you can set the wheel scroll rate here.

6.10.4.6 Build Timeout

Set the maximum time allowed for a build. If a build takes longer than the alotted time, a timeout will occur and no new hex file will result.

If Build Timeout is set to “off”, then no timeout will occur.

6.10.4.7 Symbol Display Width

The symbol display width options allow you to specify how many character spaces MPLAB allocates when displaying symbolic information.

Register Variables	Allows you to select a width of 6 characters wide to 32 characters.
Address Labels	Allows you to select a width of 6 characters wide to 32 characters.

6.10.5 Colors

Select *Options > Colors* to change the colors assigned to displayed data. To change the color, select the text that you want to alter by clicking on it. Next, select the new color.

6.10.6 Processor Setup

The Processor Setup commands allow you to configure the simulator.

6.10.6.1 Hardware

Select *Options > Processor Setup> Hardware* to display a dialog box for entering additional hardware setup data for the processor you are simulating.

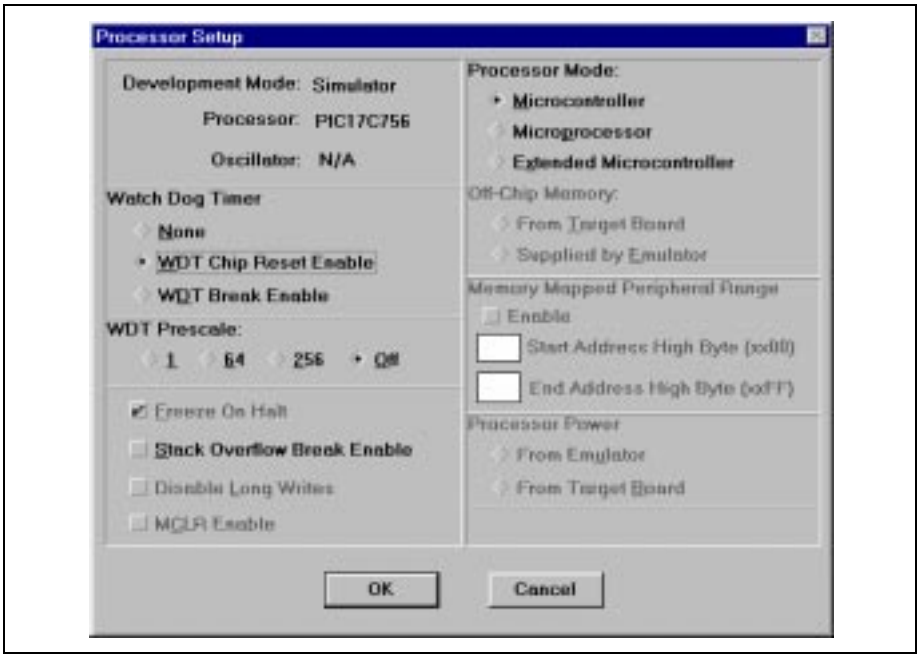


Figure 6.25: Processor Setup Dialog

Development Mode	Displays current development mode. To change the development mode, select *Options>Development Mode*.
Processor	Displays the selected processor to be simulated or emulated. To change the processor type, select *Options>Development Mode*.

Oscillator	Choose the type of oscillator for an emulation system. Selections are: RC: resistor-capacitor LP: low power XT: crystal HF: high frequency
Watch Dog Timer	Enable/Disable the Watch Dog Timer. None: Disables the Watch Dog Timer. WDT Chip Reset Enable: Resets the processor when the Watch Dog Timer times out. WDT Break Enable: Halts the processor when the Watch Dog Timer times out.
WDT Prescale	Set the WDT prescaler value.
Freeze on Halt	The simulator does not support this function.
Stack Overflow Break Enable	Select Stack Break Enable to halt the processor if a stack overflow or underflow occurs.
Disable Long Writes	Disable/Enable long writes.
MCLR Enable	Enable/Disable Master Clear.
Processor Mode	For PIC17CXXX devices, select the processor mode. Refer to individual data sheets for more information on each mode. Microcontroller: Accesses internal program memory only. Microprocessor: Accesses external program memory only. Extended Microcontroller: Accesses internal and external program memory.
Off-Chip Memory	For PIC17CXXX devices, select off-chip memory From Target Board or Supplied By Emulator.
Memory Mapped Peripheral Range	For PIC17CXXX devices, Enable/Disable the memory mapped peripheral range, and specify the starting and ending high byte address.
Processor Power	For an emulator system, state whether the processor gets its power From Emulator or From Target System.
Message Box	An additional box for mode-specific information. Ex: For MPLAB-ICE, low voltage mode status is indicated.
OK	Accepts entries made.
Cancel	Closes this dialog box without making any changes.

6.11 Tools Menu

6.11.1 DOS Command to Window (F11)

Tools > DOS Command to Window allows you to run a DOS program such as a compiler, or an internal command such as DIR, and capture the output into an edit window.

The command starts a dialog that prompts you for the command line to be executed. This can be any DOS command, including built-in ones like DIR.

You can specify the working directory that the command is to run in. This affects only the DOS command. MPLAB will still use its previous working directory.

When you start the dialog, MPLAB sets the command string and the proposed working directory name to the values you set the last time you used it.

You may execute only one DOS command at a time. Until the command completes, MPLAB will reject attempts to start another.

Note: You cannot start a Windows application with this dialog.

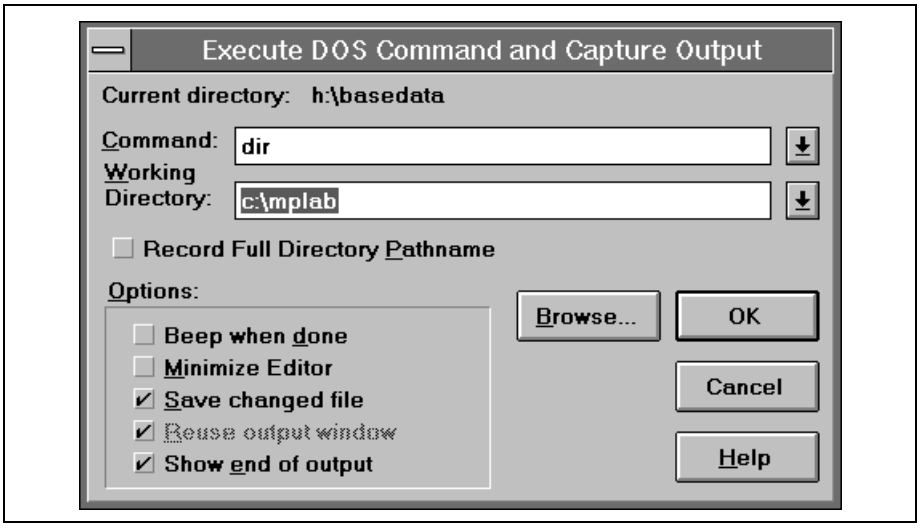


Figure 6.26: DOS Command to Windows

1. Set the various options you want to apply:

   Beep When Done causes the Editor to sound the standard system beep when the DOS program finishes.

   Minimize Editor causes the Editor to make itself into an icon before running the DOS program

   Save Changed Files causes the Editor to see if any of the files you're editing have changed and will give you the opportunity to save the changes before running the DOS program. If you agree to save the changes, all the files you've altered will be written to disk.

Reuse Output Window causes the Editor to place the DOS program's output into the window used the last time you used this dialog. If not, a new output window will be created.

Show End Of Output causes the Editor to automatically scroll the window showing the output to show you the end of it rather than the start.

2. Click **Ok** button to run the DOS program.

   The command line, working directory path, and options are recorded and become the default the next time you use this dialog.

6.11.2 Repeat DOS Command to Window (Ctrl+F11)

Tools > Repeat DOS Command to Window exactly repeats the last DOS command you ran with Execute DOS Command To Window.

This command shows you the command output in a window when it completes. If you have not previously run a command, the Editor will act as if you selected Execute DOS Command To Window and will show you the dialog.

Repeat DOS Command To Window repeats the last command you issued with this dialog.

6.11.3 Verify Emulator

If you are using an emulator system with MPLAB, you can verify that it is operating properly with this command. Dialog boxes will vary depending on the emulator that you are using.

6.11.4 Configure Emulator

If you are using an emulator system with MPLAB, you can configure the emulator with this commend. Dialog boxes will vary depending on the emulator that you are using.

6.12 Window Menu

All Window options are available in simulator mode or emulator mode.

In Editor Only mode, Absolute Listing and Show Symbol List are available. In addition, the window positioning options and the Open Windows selections are also available.

Available windows are:

- Program Memory
- Trace Memory
- EEPROM Memory
- Calibration Data
- Absolute Listing
- Stack
- File Registers
- Special Function Registers
- Show Symbol List
- Stopwatch
- Project
- Watch Window
- Modify

More information on these windows can be found in Section 5.9.

Other items that operate as standard Windows items are:

- Tile Horizontal
- Tile Vertical
- Cascade
- Iconize All
- Arrange Icons
- (*Open Windows*)

For more information on these items, please consult Windows documentation.

6.12.1 Program Memory

Select *Window > Program Memory* to display program memory. The program memory window can display locations in the range of program memory for the currently selected processor. You can leave the Program Memory window open at all times and move and resize the window.

The Program Memory window is only available in Emulator and Simulator mode.

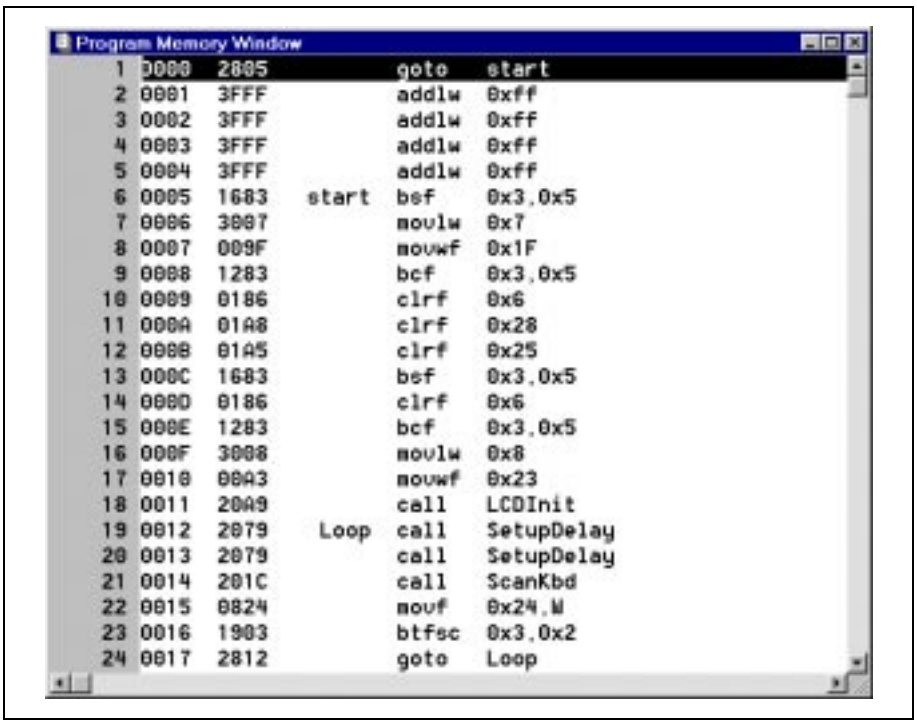


Figure 6.27: Program Memory Window

6.12.1.1 System Button Options

Click the system button in the upper left corner of the Program Memory screen to display the following options:

Toggle Line Numbers	Toggles field for displaying line numbers and qualifier points.
Hex Code Display	Displays program memory information as hex data.
Machine Code Display	Displays disassembled hex code with no symbolic information.
Disassembly Display	Displays disassembled hex code with symbols.

6.12.1.2 Program Memory Field Descriptions

MPLAB displays data in the Program Memory window that it reads directly from emulation memory. Program Memory fields contain the following information:

Field One	Address in hex.
Field Two	Opcode (or data) in hex.
Field Three	Program Label in symbolic format. You can increase the display width of labels from *Options > Environment Setup > Address Label*.
Field Four	Machine code, disassembled code, or source code.
Highlight Bar	Current location of the program counter.

Program Memory locations display break, trace and trigger out status at each memory location as follows:

6.12.1.2.1. Selecting Points

Symbol	Point Type	Menu Selection (RMB = Right Mouse Button)
B	Break Points	*RMB > Break Point(s)*
T	Trace Points	*RMB > Trace Point(s)*
O	Trigger Output	*RMB > Trigger Point(s)* (Available only with the MPLAB-ICE emulator)
Q	Pass Count Address	Set from Break or Trace Point Settings Dialog

MPLAB uses a combination of color and notations in the Program Memory window to show break/trace/trigger points. At a particular address, if no points have been set, the text will display normally. If a point is set, the color of the text will change, the width of the line number window will increase to show the active points. Unset points are displayed as periods.

6.12.1.3 Creating a Temporary Real-Time Break Point

To set up a temporary real-time break point from the Program Memory Window, double click the left mouse button anywhere on a valid address line. The processor runs in real time until one of the following occurs:

- The line containing the temporary break point is executed,
- A break point is encountered, or
- You click on **Halt**.

6.12.2 Trace Memory

Select *Window > Trace Memory* to display the contents of the trace buffer. This window can be left open at all times, moved around and resized.

Field One	Address in hex.
Field Two	Opcode (or data) in hex.
Field Three	Program Label in symbolic format. You can increase the display width of labels from *Options > Environment Setup > Address Label*.
Field Four	Machine code, disassembled code, or source code.
Field Five	Emulator: Status on External Logic Probe Lines. The status fields are displayed on the extreme right hand side of the window. Simulator: Time stamp and changed register information.

To save the contents of the trace buffer to a file, execute *File > Save Trace Buffer*.

6.12.3 EEPROM Memory

Select *Window > EEPROM Memory* to display the EEPROM data memory window for a microcontroller device that has EEPROM data memory. The PIC16C84 is an example of a device that supports EEPROM memory.

The EEPROM window can be left open at all times, moved around and resized. This window is for information only and you cannot change values from this window.

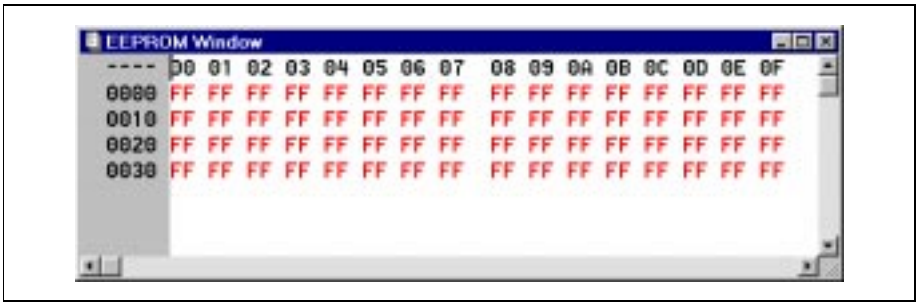


Figure 6.28: EEPROM Memory Window

The EEPROM window displays the following data/opcode hex information of the particular processor being emulated. When an EEPROM register value changes or the processor is halted, the data in the EEPROM window is updated.

6.12.3.0.1. System Button Options

Click the system button in the upper left corner of the EEPROM Memory screen to display the following options:

Toggle Line Numbers	Toggles field for displaying line numbers.
Hex Display	Displays program memory information as Hex data.
ASCII Display	Displays the ASCII character at each memory location.

6.12.4 Calibration Data

The Calibration Data dialog is for use with the PIC12CXX or PIC14000 device families. It displays the floating point data in the Emulator Probe for alteration by the user.

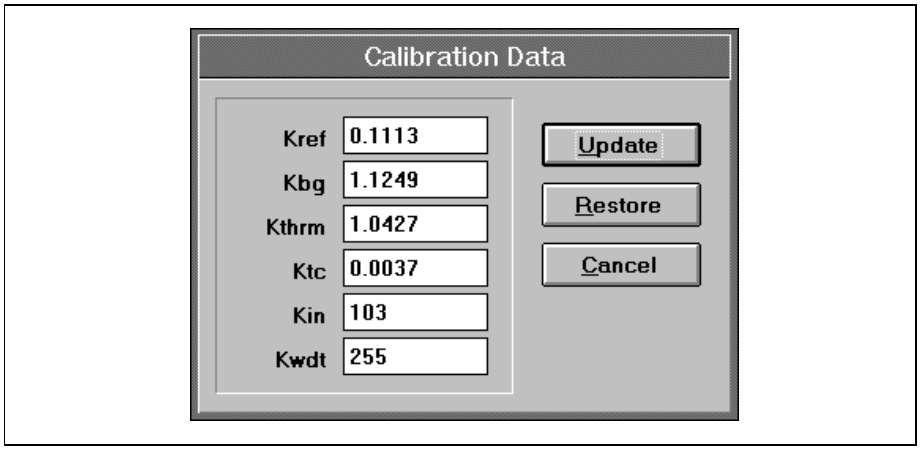


Figure 6.29: Calibration Data Dialog

Update The Update button takes the information from calibration data dialog, converts it from the IEEE 754 format to the Microchip version of IEEE 754, and stores it in the memory area for use in the embedded code. Only the first four numbers KREF, KBG, KTHRM, and KTC are in floating point format. The last two, KIN and KWDT, are 8-bit, unsigned numbers with values from 0 to 255.

Restore The Restore button takes the original calibration data uploaded from the probe during system reset and resets the values in the memory area for use in the embedded code.

Cancel The Cancel button closes the dialog and does not modify the values in the memory area.

6.12.5 Absolute Listing

Select *Window > Absolute Listing* to display the *.LST file and to single step through the *.LST file generated by MPASM or a compatible C compiler.

Use this function to display both C code and the assembly code that corresponds to the C code. The Absolute Listing gives you a better idea how the C compiler implemented your code. If you are not using C, you will get a similar listing file generated by MPASM.

6.12.6 Stack

Select *Window > Stack* to open a window displaying the contents of the stack. The number of available levels depends on the processor type being emulated. The Stack Window can be left open at all times, moved around and resized.

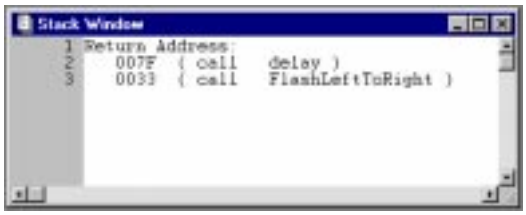


Figure 6.30: Stack Window

6.12.6.1 Hardware Stack Levels

6.12.6.1.1. 12-Bit Core Hardware Stack – 2-Levels Deep

Devices with a 12-bit core, such as PIC12CXXX and PIC16C5X, have an 8-level deep hardware stack.

6.12.6.1.2. 14-Bit Core Hardware Stack – 8-Levels Deep

Devices with a 14-bit core, such as PIC14000 and PIC16CXX, have an 8-level deep hardware stack.

Note: If you set Stack Break Enable on the Hardware configuration display, (*Options > Processor Setup > Hardware*) and push or pop the stack beyond its limit, for the PIC16CXX, MPLAB will display an underflow or overflow message.

6.12.6.1.3. 16-Bit Core Hardware Stack – 16-Levels Deep

Devices with a 16-bit core, such as PIC17CXXX, have an 8-level deep hardware stack.

Note: If you set Stack Break Enable on the Hardware configuration display, (*Options > Processor Setup > Hardware*) and push the stack beyond its limit, for the PIC17C4X, MPLAB will display an underflow or overflow message.

6.12.6.2 Simulator Stack—12-Bit Core Devices

The MPLAB-SIM simulator presents an accurate simulation of the hardware stack on the PIC12CXX and PIC16C5X devices and additionally provides warning messages if an underflow or overflow condition occurs. When a CALL instruction is encountered, or when an interrupt has occurred, the value of the PC+ 1 is pushed to the stack, and the stack is popped when a RETLW instruction is executed. If more than two values are pushed to the stack before it is popped, the value will be pushed to the stack, but a warning message will be issued, indicating a stack overflow condition. An error message will also be generated if the user attempts to pop an empty stack. Popping an empty stack will cause the last value popped to be put in the PC.

6.12.6.3 Simulator Stack—14-Bit Core Devices

The MPLAB-SIM simulator presents an accurate simulation of the hardware stack on the PIC14000 and PIC16CXX devices, and additionally provides warning messages if an underflow or overflow condition occurs. When a CALL instruction is encountered, or when an interrupt has occurred, the value of the PC+ 1 is pushed to the stack, and the stack is popped when a RETLW, RETURN, or RETFIE instruction is executed. If more than eight values are pushed to the stack before it is popped, the value will be pushed to the stack, but a warning message will be issued, indicating a stack overflow condition. An error message will also be generated if the user attempts to pop an empty stack. Popping an empty stack will cause the stack pointer to point to the top of a full stack, and will not generate an error message if another pop is initiated.

6.12.6.4 Simulator Stack—16-Bit Core Devices

The MPLAB-SIM simulator presents an accurate simulation of the hardware stack on the PIC17CXX, and additionally provides warning messages if an underflow or overflow condition occurs. When a CALL or LCALL instruction is encountered or when an interrupt has occurred, the value of the PC+ 1 is pushed to the stack. The stack is popped when a RETLW, RETURN, or RETFIE instruction is executed. If more than sixteen values are pushed to the stack before it is popped, the value will be pushed to the stack, a warning message will be issued indicating a stack overflow condition, and the STAKAVL bit will be cleared until a reset condition occurs.

6.12.7 File Registers

Select *Window > File Registers* to display a window of all the File Registers of the particular processor being emulated. When a file register value changes, or the processor is interrogated, the data in the File Register window is updated. The File Register window can be left opened at all times, moved around and resized.

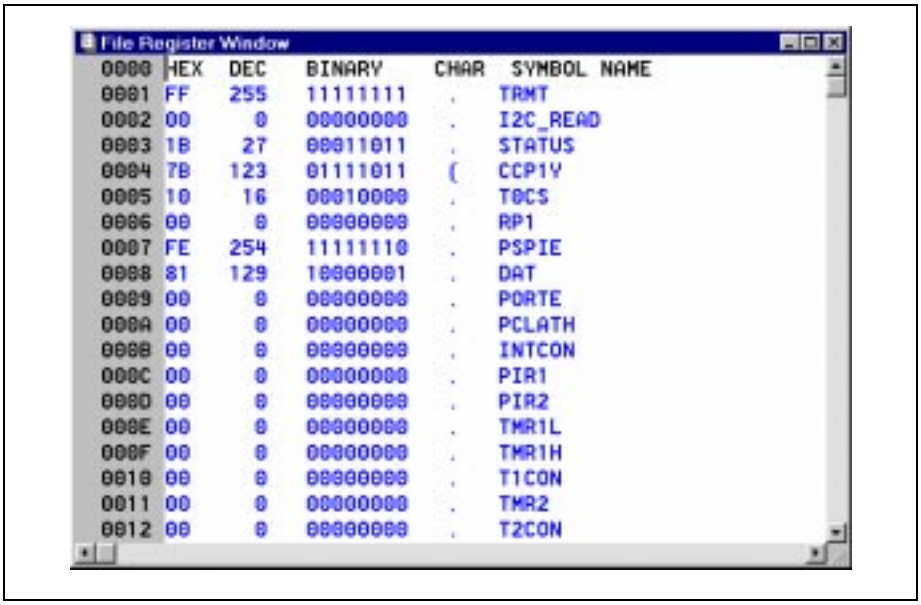


Figure 6.31: File Registers Window

From the File Register Window, you can open the Modify dialog box to fill a range of registers with a constant value. Either select a desired range by clicking the left mouse button and dragging over the values that you want to change, or simply place the cursor on the register you want to change and click the right mouse button to display the Fill Register popup menu. Select the Fill Register menu option to display the Modify dialog box with the address range already entered.

6.12.7.1 System Button Options

Click the system button in the upper left corner of the File Register Memory screen to display the following options:

Toggle Line Numbers	Toggles field for displaying line numbers.
Hex Display	Displays file registers information as Hex data.
Symbolic Display	Displays data at each memory location in the following formats: Hex, Decimal, Binary, ASCII Character, Symbol, and Name.
ASCII Display	Displays the ASCII character at each memory location.

6.12.8 Special Function Registers

Select *Window > Special Function Registers* to display the contents of the Special Function Registers (SFR) for the processor being emulated. Whenever a break occurs, the contents of the Special Function Registers are updated.

This window can be left opened at all times, moved around, and resized. A sample of this window is shown below.

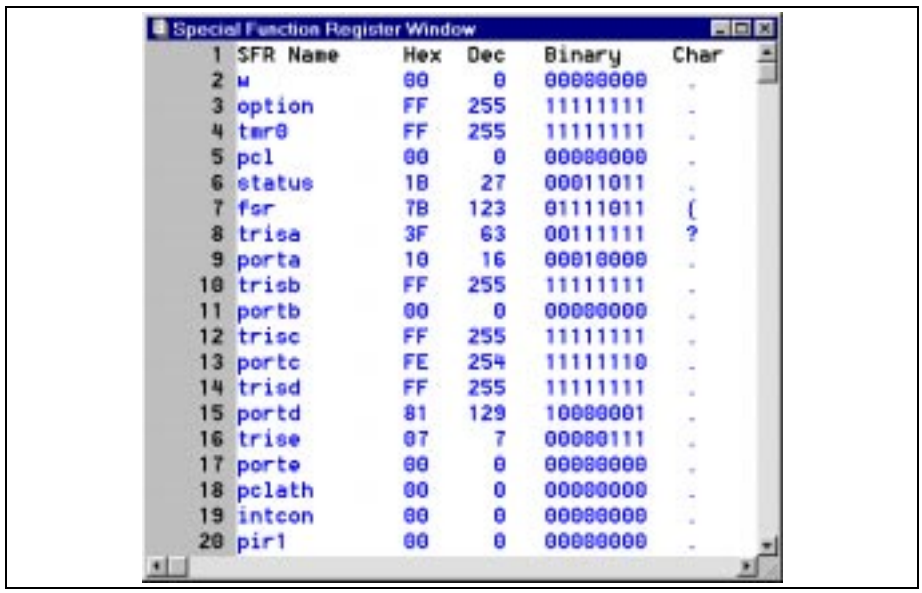


Figure 6.32: Special Function Registers

Field One	Name of Special Function Register.
Field Two	Data as a hexadecimal value.
Field Three	Data as a decimal value.
Field Four	Data as a binary number.
Field Five	Data as ASCII characters.

To modify the contents of a particular Special Function Register:

1. Double click on a register in this window to invoke the Modify dialog box with the symbol/address and data fields already filled in, or
2. Use the execute *Window > Modify* menu.

Note: The SFR names and addresses are different for every device.

6.12.9 Show Symbol List (Ctrl+F8)

Click *Window > Show Symbol List* to display all symbols known to MPLAB. Symbols include constants and labels. Show Symbol List is an information only dialog box. To display Show Symbol List, a project **MUST BE OPEN AND BUILT.**

The symbols displayed in this dialog box represent the symbols imported from your source code after compiling or assembling.

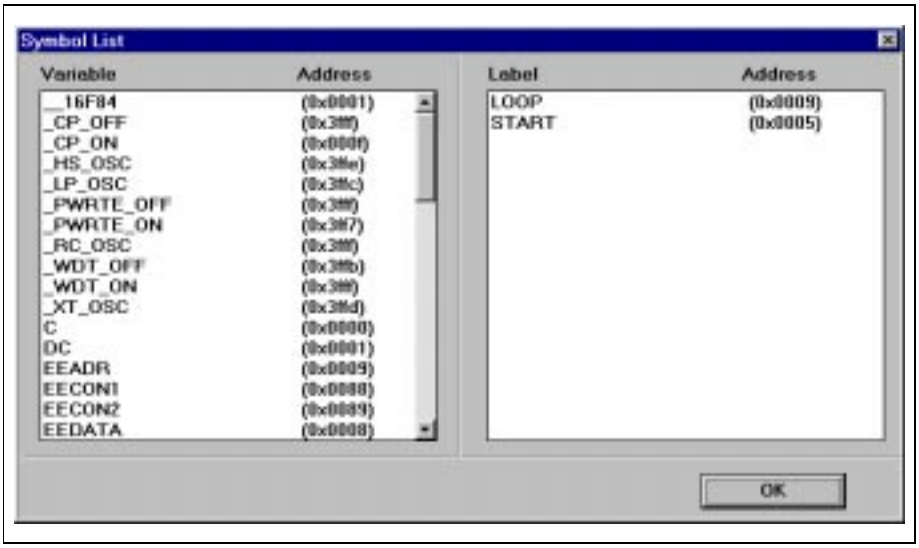


Figure 6.33: Show Symbol List

Variable, Address	Displays variables from the File Register memory and the address of each variable.
Label, Address	Displays labels from program memory and the address of each label.
Constants	Constants defined in the source code can be used in executing opcode and as operands for instructions in Modify.

6.12.10 Stopwatch

Select *Window > Stopwatch* to display the current value of the Cycle counter. The system Stopwatch counts the number of clock cycles that the processor executes. The counting occurs with real-time execution and with polled execution. The timer triggers on every cycle of an instruction.

Figure 6.34: Stopwatch Dialog Box

Cycles	Displays the number of cycles that the processor executes.
Time	Displays the stopwatch time in seconds. Computed from the number of cycles executed and from the processor frequency.
Zero	Click **Zero** to reset the cycle counter to zero. You can reset the timer at any time when the processor is halted.
Example:	You can use this timer for precise timing measurements. If you need to measure the exact time a subroutine takes to execute, then simply reset the timer before entering the subroutine and put a break point at the end of the subroutine. The timer displays the total number of cycles executed in the subroutine and also displays the execution time.
Processor Frequency	Displays the selected Processor Frequency. To change the frequency, you must go to the *Options>Processor Setup>Clock Frequency* dialog.

6.12.11 Project Window

The Project Window is only available when a project is open. It displays the list of files currently in the project. If the project has been compiled, the project window displays a list of all included files in the project. Otherwise, the Project Window it only displays the main project file. Double click on any file displayed in the Project Window to open that file for editing.

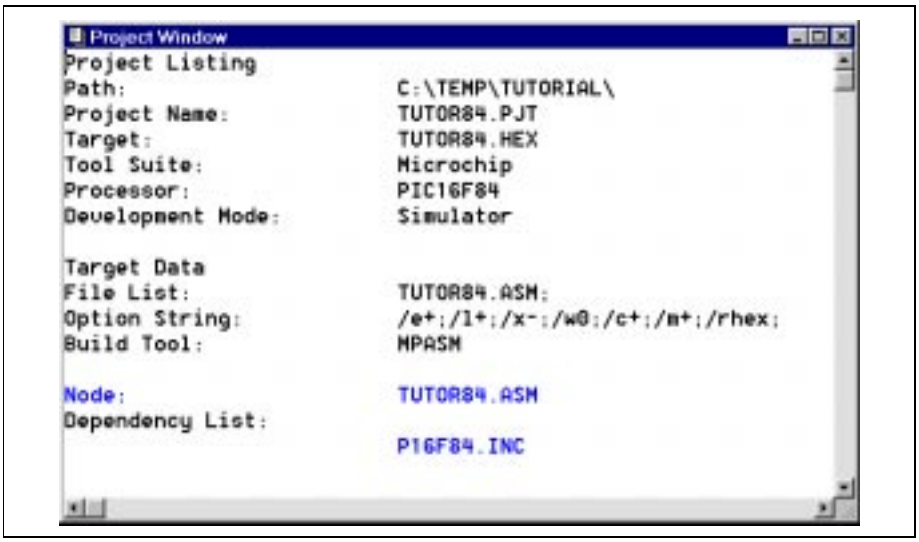


Figure 6.35: Project Window

6.12.12 New Watch Window

MPLAB allows the contents of file registers to be monitored through a watch window. To open a watch window, select *Window>New Watch Window*.

For more information on Watch Window dialogs, see Section 5.9.11.

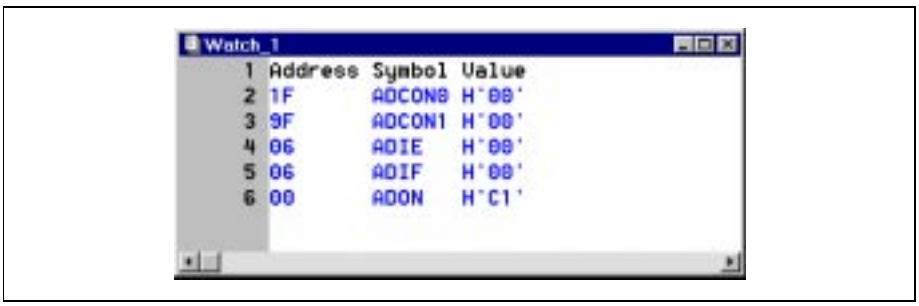


Figure 6.36: Watch Window

6.12.13 Load Watch Window

Select *Window > Load Watch Window* to load a watch window that you previously saved to disk. Select a watch window file to load and click **Ok**, or double click the desired file.

6.12.14 Modify

Select *Window > Modify* to display and/or modify the contents of Data Memory, Program Memory, the Stack, or EEPROM memory.

Modify allows you to Read/Write to a specific address, Read/Write while incrementing to the next address, or fill an address block. MPLAB allows you to leave the Modify window open at all times and move it around.

Figure 6.37: Modify Dialog Box

MPLAB provides four ways to open the Modify dialog box:

- Select *Window > Modify*.
- Double click on a item in the Special Functions Register window
- Double click an item in a Watch Window.
- Select an address or range in the File Register Window and click the right mouse button to display the Fill Register button. Click the **Fill Register** button to display the Modify dialog box.

Address	Enter the Address at which data is to be read or modified. You can enter a numerical address or a symbol. (Label)
Data/Opcode	Click **Read** to display data value/ Opcode at a selected address and memory area. Click **Write** to write data value/ Opcode to the selected address and memory area.
Radix	Hex, Decimal, or Mnemonics
Memory Area	Select the Memory Area that you want to modify: Data Memory: RAM Memory Program Memory: ROM Memory in the emulator Stack: Stack Memory on the Device EEPROM: EE Data Memory
End Adr	The ending address for Fill Range.

Fill Range	Fills the range defined by the two addresses with the value entered in Data/Opcode.
Auto Increment	Select Auto Increment to increment to the next address after each Read/Write.

Note: Auto increment increments to the next address, displays the next address, and reads the contents at the address. If you are using Auto Increment to read a range, enter the address of the memory area minus one because the first read will increment the address.

Write	Enter new data in the Data/Opcode field, and click Write to modify the data at the specified address. (You can enter data in symbolic format.) When data is modified, all the appropriate windows are updated with the new information.
Read	Click **Read** to read the data at a specified address.
Close	Click **Close** to exit from Modify.

Caution: Use care when modifying the stack or File register 2 (PCL). The effect of modifying these registers is not seen until the processor is taken out of halt.

6.12.15 Tile Horizontal

The *Window > Tile Horizontal* command sizes open windows in a horizontal format making each window as wide as possible to allow you to see as much of each line in as many windows as possible. The command arranges all open windows in a tile pattern, placing the windows above one another. Excess windows are tiled in a horizontal pattern in the lower part of the screen.

Windows containing the output from commands run by the *Tools > DOS Command To Window* command are arranged preferentially at the top of the screen.

6.12.16 Tile Vertical

Window > Tile Vertical command sizes open windows vertically in columns to allow you to see as many lines as possible in each window.

The Tile Vertical command arranges all open windows in a tile pattern, placing them side by side so that each window is as deep as possible.

6.12.17 Cascade

The *Window > Cascade* command arranges all open windows in a cascade pattern.

6.12.18 Iconize All

Window > Iconize All makes all windows into icons.

6.12.19 Arrange Icons

Window > Arrange Icons arranges all iconized windows so that their icons are visible in rows at the bottom of the desktop. Open windows are not affected by this command.

6.12.20 Open Windows

Window > Open Windows lists the open windows at the end of the Window commands.

Whenever you open a window, MPLAB records the name in the list, ordering it so that the windows you have used most recently always appear at the top.

More Windows

When the list contains more files than can be displayed on the Window menu, an extra menu command More Windows is automatically added. The command starts a dialog that shows you the entire contents of the list and lets you pick the window you want to open.

To open a window from the list, either double-click the window name in the list with the left mouse button, or select the name and press the **Open** button.

The More Windows dialog box also provides additional options. The settings of the additional options are remembered and become the default for the next time you use this dialog.

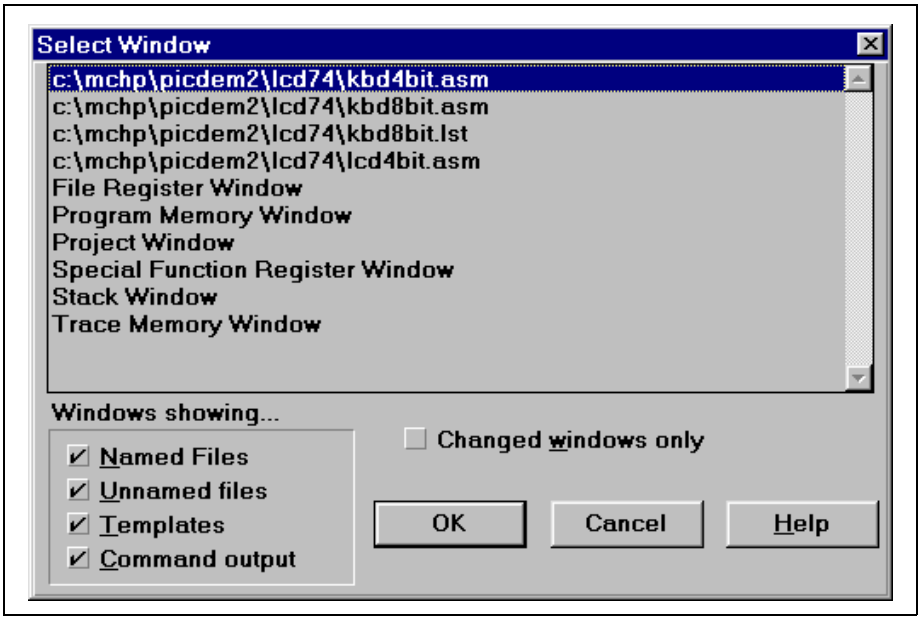


Figure 6.38: More Windows Dialog

Changed Windows Only	Displays only windows that have been changed since opening.
Named Files	Lists all windows having an associated file name.
Unnamed Files	Lists all windows that do not have an associated file name.
Templates	Lists all template windows.
Command Output	Lists all windows containing command output obtained by executing *Tools > DOS Command to Window*.

6.13 Help Menu

6.13.1 Release Notes (Shift+F1)

Help > Release Notes opens and displays the recent change history of MPLAB software. The Release Notes are contained in the file, README.1ST.

6.13.2 Tool Release Notes

If you are using additional tools, there may be an item on the Help menu on release notes for those tools.

6.13.3 MPLAB Help

Help > MPLAB Help contains help on using MPLAB.

6.13.4 Editor Help

Help > Editor Help contains help on using the MPLAB Editor.

6.13.5 MPASM Help

Help > MPASM Help opens the entire MPASM User's Guide. Click on the highlighted items to get more information on a particular item.

MPASM Help also contains a Quick Reference Guide with the following information:

- Compile Directives
- PIC16C5X Instruction Set
- PIC16CXX Instruction Set
- PIC17C4X Instruction Set

6.13.6 Tool Help

If you are using additional tools, there may be an item on the Help menu on help for those tools.

6.13.7 About

Help > About displays:

- MPLAB IDE Version
- Microchip's Address
- Processor Version
- Disassembler Version
- Information on other registered applications

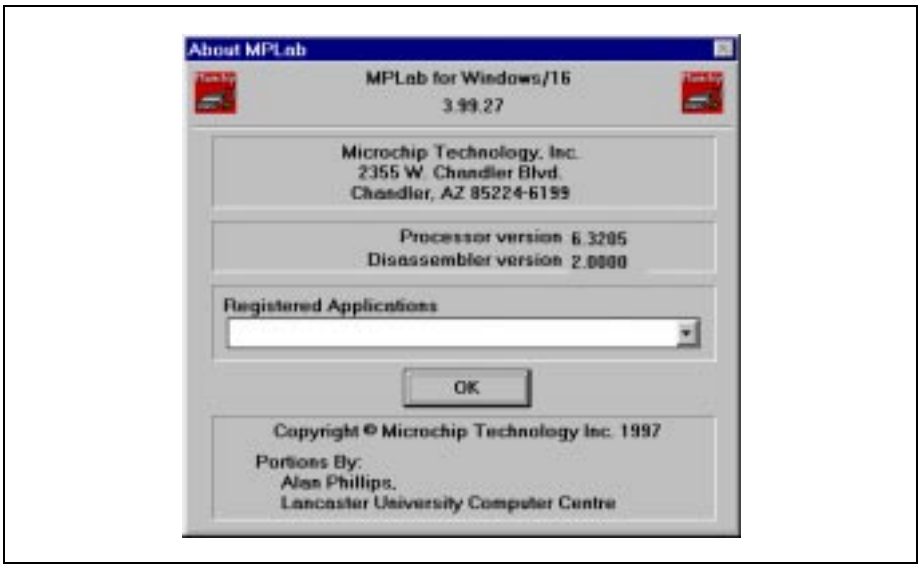


Figure 6.39: About Help Dialog

Part 2 – Editor

Chapter 1. MPLAB Editor Preview

1.1 Introduction

This chapter defines what MPLAB Editor is and how it helps you, as well as itemizing the features and functions of the editor.

1.2 Highlights

The following topics are addressed in this chapter:

- What is MPLAB Editor
- How MPLAB Editor Helps You
- MPLAB Editor Features
- MPLAB Editor Functions

1.3 What is MPLAB Editor

The MPLAB Editor is an integrated part of the MPLAB Integrated Development Environment (IDE). The editor is always available when MPLAB is running. It is not a separate executable file but a set of features in MPLAB.

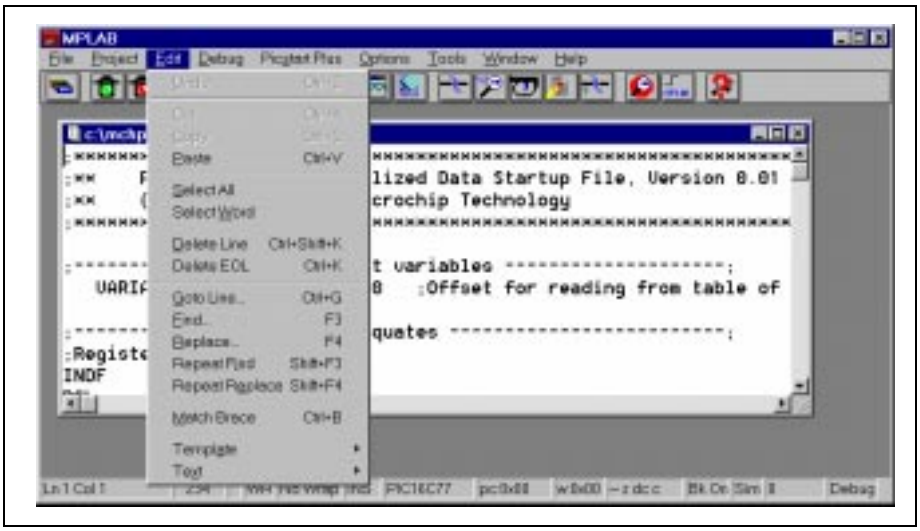


Figure 1.1: Using the MPLAB Editor

Part 2 – Editor

1.4 How MPLAB Editor Helps You

The MPLAB IDE and Editor are designed to allow PICmicro microcontroller developers an easy and quick method to develop and debug firmware for Microchip Technology Incorporated's PICmicro microcontroller product families.

1.5 MPLAB Editor Features

1.5.1 File Size

The MPLAB Editor is limited only by the total amount of available memory on your system. There are no limits on the number of editable files or on the number of open edit windows. The MPLAB Editor also has no limit on the size of file that it can open and has no limit on the number of lines that a file may contain.

1.5.2 Windows MDI Conventions

MPLAB Editor adheres to the Windows MDI conventions:

- Invokes most commands and facilities from menus
- Moves around with a mouse or with standard keyboard shortcuts
- Cut and Paste capabilities from the clipboard

1.5.3 Reconfigure Keyboard

You can reconfigure the keys to meet your requirements. Commands may be invoked by two-character key sequences like **Esc G** and **Ctrl+K Ctrl+B**. You may also use key sequences such as **Alt+F** and **Alt+S**. You can map almost all keyboard keys, in any combination.

1.5.4 Build Files Easily

MPLAB Editor allows you to:

- Define sets of templates – standard lines of text – that you can insert into the current file with just a few mouse clicks
- Group the templates you work with into distinct files
- Load templates for automatic usage

1.6 MPLAB Editor Functions

The MPLAB Editor provides functions that allow you to perform the following operations:

1.6.1 File Commands

- Create A New File
- Open an Existing File
- Change a File Name
- Save A File To Disk
- Save Changes to a File or Template
- Backup an Existing File
- Close a File

1.6.2 Template Operations

- Create a Template File
- Attach a Template File
- Automatically Attach Template Files
- Create a Template
- Insert a Template
- Edit a Template
- Delete a Template
- Save a Changed Template File
- Detach a Template File

1.6.3 Text Handling

- Type Text
- Select Text
- Delete, Cut, Copy, and Paste Text
- Find Text
- Replace Text
- Text Processing
  - Automatic Text Wrapping
  - Indenting and Unindenting Text
  - Changing Case
  - Handling Braces
  - Undoing Edit Actions

1.6.4 C Language Awareness

- Position the "#" Character in Column One
- Position Closing Brace "}" in Same Column as Opening Brace

Chapter 2. Using the MPLAB Editor

2.1 Introduction

This chapter contains overviews and descriptive information that explains the features of MPLAB Editor, and gives step-by-step instructions for using the MPLAB Editor.

2.2 Highlights

This chapter discusses the following topics:

- **File Operations**
- **Window Modes**
- **Templates**
- **Text Handling**
- **C Language Awareness**

2.3 File Operations

This section describes how to create, open, change, insert, save, backup, and close files with the MPLAB Editor.

2.3.1 Create A New File

The *File > New Source* command opens a new, empty window in which you can type. The window does not initially have a file name. In order to save what it contains use *File > Save As*.

When MPLAB Editor creates the window, it applies the set of modes that are defined for "new files."

2.3.2 Open Existing Files From MPLAB Editor

To edit one or more existing files, use the *File > Open Source* command. It opens a standard dialog to select the files to edit. If a selected file is already open, MPLAB Editor activates the window that is currently showing the file.

2.3.3 Save A File To Disk

To save a file to disk you have a choice of two commands.

- Use *File > Save* to save the contents of the current window to disk. The MPLAB Editor replaces the file with the contents of the current window.
- Use *File > Save As* to supply the name of the disk file to write to. The MPLAB Editor confirms overwrites of existing files.

2.3.4 Close A File

To close a file use the *File > Close* command. MPLAB Editor prompts to either write the file to disk or to abandon the operation.

Use the *File > Close All* command to close all the open files.

2.4 Window Modes

MPLAB Editor associates a set of Window Modes with every edit window. The modes can be set with the *Options > Default Editor Modes* commands.

The possible window modes are:

- **Auto Indenting** If this mode is on, MPLAB Editor automatically indents new lines to match the indentation of the previous line.
- **Language** This mode is a string which defines the language information MPLAB Editor should use for the window. Currently, you can set this only to C.
- **Overwrite** If this mode is on, MPLAB Editor replaces the characters under the cursor as you type.
- **Page Headers** If this mode is on, MPLAB Editor begins each printed page with a title showing the name of the file, the date and time, and the page number.
- **Soft Tabs** If this is mode on, MPLAB Editor inserts a suitable number of spaces to bring the cursor to the next tab stop, as defined by the Tab Size mode. Otherwise, MPLAB Editor inserts a single tab character into the file when you press the TAB key.
- **Show Line Numbers** If this mode is on, MPLAB Editor displays the number of each line in the window.
- **Strip Trailing Spaces** If this mode is on, MPLAB Editor removes any trailing space or tab characters from a line when you press **Enter**.
- **Tab Size** This mode is a numeric value that defines the width of a tab stop on the screen.
- **Wrap Column** This mode is a numeric value that defines the column for automatic text wrapping.
- **Wrap Enabled** If this mode is on, MPLAB Editor automatically wraps the current line at the Wrap Column selection.
- **Wrap Long Lines** If this mode is on, MPLAB Editor wraps lines that do not fit on the page when printing. Otherwise, MPLAB truncates long lines.

2.5 Templates

Templates are pre-built sections of text that you can include in your files. They can help you avoid repetitive typing of the same information.

For example:

If documented C procedures begin by preceding each procedure with an explanatory section like this:

```
/************************************************
   start_editor
   ------------
   This procedure starts the system in edit mode
************************************************/
```

Rather than type the lines of asterisks every time, create a template containing the text:

```
/************************************************
       <???>
   -------------
************************************************/
```

The “<???>” characters are a template mark that shows where to type in extra items. The template marks can be searched for automatically, as will be explained shortly.

Templates are held in libraries called template files. MPLAB Editor can have multiple template files ready for use.

> **Note:** Template files are not ordinary text files and cannot be edited directly.

You can attach a template file manually at any time, or you can have MPLAB Editor attach a set of templates automatically each time you invoke it.

2.5.1 Create a Template File

A template file must be created before storing a template. Use the *Editor > Template > Create File* command to have MPLAB Editor create the file. Templates are auto-attached when created.

2.5.2 Attach a Template File

A template file must be attached before accessing any templates in the template file. When you attach a template file, MPLAB Editor loads it into memory and constructs an index of the templates it contains.

Use the *Editor > Template > Attach File* command to attach a template file.

2.5.3 Automatically Attach Template Files

MPLAB Editor can attach a template file automatically, rather than manually, with the *Editor > Template > Attach File* command. MPLAB Editor scans the current directory for a valid template file called AUTO.TPL and attaches it.

2.5.4 Create a Template

Use the *Editor > Template > New* command to create a template. MPLAB Editor opens an edit window whose caption displays that it is a template.

Edit the window normally, using the *Editor > Template > Insert Mark* command to insert template marks.

The template must be stored in a template file to use it. However, MPLAB Editor does not save the template to disk until it receives *Editor > Template Save File* command.

2.5.5 Insert a Template

MPLAB Editor can insert a template from any attached template file at the position of the cursor.

The *Editor > Template > Insert* command gives a list of all the attached template files. Select one of the template files to see a list of the templates it contains.

2.5.6 Using Templates

Once inserted, use the *Editor > Template > Find Mark* command to search for template marks. Enter the desired text at the mark.

2.5.7 Edit a Template

To change a template in a template file, attach the template file and use the *Template > Edit* command.

When you select a template, MPLAB Editor places it in an edit window. The window is the same as for a normal text file, but the caption shows that it is actually a template.

2.5.8 Delete a Template

To delete a template from a template file, attach the template file and use the *Editor > Template > Delete* command.

2.5.9 Save a Changed Template File

When you edit, create, or delete a template, MPLAB Editor changes a copy of the template in memory. MPLAB Editor does not change the template file on disk until you use the *Editor > Template > Save File* command

2.5.10 Detach a Template File

Use the *Editor > Template > Detach File* command to remove a template file from memory.

2.6 Text Handling

2.6.1 Type Text

MPLAB Editor inserts text in either insert or strikeover mode. MPLAB Editor shows the mode as either “INS” or “OVR” on the status bar.

2.6.2 Select Text

MPLAB Editor allows the user to select an area of text and operate on it. The selected area appears highlighted. Select text by using either a mouse, or with the keyboard in the standard Windows fashion.

2.6.3 Find Text

To search for text in a window, place the cursor at the start point of the search, and select *Editor > Find*.

To repeat a search without using the Find Dialog, select *Editor > Repeat Find*.

2.6.4 Find Special Characters

In the *Editor > Find* dialog you can search for special characters by using the escape sequence for the character. You can use “\n” for carriage returns, “\t” for tabs, and “\\” for backslashes.

2.6.5 Replace Text

To search for text in a window and replace it, place the cursor at the start point, and use the *Editor > Replace* command.

2.6.6 Replace Special Characters

In the *Editor > Replace* dialog you can replace special characters by using the escape sequence for the character. You can use “\n” for carriage returns, “\t” for tabs, and “\\” for backslashes.

2.6.7 Text Processing

Although MPLAB Editor is intended to be used as a program text editor, it has several features that make it useful in general text editing.

2.6.7.1 Automatic Text Wrapping

When typing ordinary text, it may be convenient to have the program fit the text into the available line width. This would typically not be the case when editing a program source code file.

To change the text wrapping mode, double-click the left mouse button in the wrap area of the status bar. This area shows the text "No Wrap" when wrapping is not active. The double click action turns wrapping on. The status bar shows "Wr 72" when wrapping is enabled and set at column 72.

The points at which MPLAB Editor wraps a line vary with the language type defined for the window.

2.6.7.1.1. Language type "(none)" or "C"

MPLAB Editor breaks the line at the closest white space character or hyphen to the defined wrap column.

2.6.7.1.2. Language type "TeXt"

MPLAB Editor breaks the line at the closest white space character to the defined wrap column.

> **Note:** MPLAB Editor wraps the line being typed only when the cursor is at the end of the line. If you move the cursor to somewhere within the line and enter text, MPLAB Editor does not wrap the line even if it extends past the wrap column.

2.6.8 Indenting And Unindenting Text

When editing program source, indenting and unindenting source is very common. MPLAB Editor provides a facility to change the indentation level of either a single line, or of a larger block of text.

To indent a single line, place the cursor anywhere within it and use the *Editor > Text Indent* command. MPLAB Editor moves the entire text of the line to the right by one tab stop.

To un-indent a single line, place the cursor within it and use the *Editor > Text Un-Indent* command. MPLAB Editor moves the text left by one tab stop. MPLAB Editor does not alter a line that does not start with either a tab or a space.

2.6.8.1 Changing Case

Use the *Editor > Text > Uppercase* or the *Editor > Text > Lowercase* commands to change the case of a selected block of text.

2.6.8.2 Handling Braces

MPLAB Editor allows the user to manipulate brace characters such as brackets and parentheses, which often delimit sections of text or program sources.

The definition of a brace character varies, depending on the language type set in a window's window modes. For C, braces are defined as the characters that have syntactic meaning – opening braces are { [or (, and closing braces are }] or).

For the language type "none," brace characters are defined as those commonly used in text or many other languages – opening braces are { [(or <, and closing braces are }]) or >.

To locate matching braces, place the cursor on one of the braces and use *Editor > Match Brace*. The cursor moves to the matching brace, respecting the level of nesting in the code.

2.6.8.3 Undoing Edit Actions

MPLAB Editor records edit actions and can reverse them with *Editor > Undo*. The undo depth is configurable by the user. For additional details, see the "*Customizing MPLAB after Installation*" appendix.

2.7 C Language Awareness

When editing files that have language type set to "C", MPLAB Editor provides these facilities:

- MPLAB Editor always moves a "#" character typed in an otherwise empty line to column 1.
- MPLAB Editor moves a closing "}" brace typed in an otherwise empty line to the same column as the matching preceding opening brace "{" if the opening brace is the only character in its line.

For example:

```
//
*****************************************
// EXAMPLE.C
//
*****************************************
#include <PIC16C84.H>
void delay(void);
void main(void)
{
    unsigned int i,j;
    TRISB = 0xff;
    PORTB = 0;
    i = 0x1;
    while(1)
    {
        PORTB = i;
        if (i == 0x80)
            i = 0x1;
        else
            i <<= 1;
        TRISB = 0;
        delay();
        TRISB = 0xff;
        delay();
    }
}
void delay(void)
{
    int x, y;
    x = 0x3f;
    y = 0xff;
    while(x--)
    {
        while(y--)
            NOP();
    }
}
```

Chapter 3. MPLAB Editor Menu Commands

3.1 Introduction

The MPLAB Editor commands are supported by the following menu options:

File Menu	Commands for opening, viewing, saving, and closing files.
Editor Menu	Text-manipulating commands.
Options Menu	Commands for setting editor modes.

3.2 Highlights

This chapter discusses the following topics:

- **File Menu**
- **Editor Menu**
- **Options Menu**

3.3 File Menu

Only the File Menu options printed in black are described in this section.

File

New Source	<ctrl+N>	Import
Open Source	<ctrl+O>	Download to Memory
View		Download to Target Memory
Save	<ctrl+S>	Copy from Target Memory
Save As		Load Simulator Stimulus
Save All		Export
Close		Save Trace Buffer
Close All	<shift+F9>	Save Hex File
		Print
		Print Setup
		Exit
		Most Recently Used Files

3.3.1 File Menu Commands

3.3.1.1 New Source – Ctrl+N

This menu item creates a new, empty edit window. The window has no file name associated with it. The tab size and other settings are the values you set up for "new files" with the *Options > Default Editor Modes* command.

3.3.1.2 Open Source – Ctrl+O

Opens one or more existing files for editing. *File > Open Source* opens a dialog that allows you to open one or more existing files.

Steps

1. Use the Drive and Directory list boxes to select the disk drive and the directory.
2. Select the files you want to open in the File Name list box.

   To add single files from the list, hold down the **Ctrl** key and click on the desired files.

   To add a list of files, either hold the **Shift** key and click on the first and last file in the desired range, or click on the first file and drag down to the last file.

   - or -

   Type the name of the files to open in the File Name field.
3. If you want to open the files in read only mode, check the **Read Only** box. This affects all the files you open in this operation.
4. Click the **OK** box to open the files.

The "List Files of Type" list at lower left allows you to restrict the files shown in the list to those matching specific filename patterns. for example, "*.asm" will list all files with the suffix ".asm".

Whether you close the dialog with the **OK** or the **Cancel** buttons, MPLAB Editor makes its current working directory the one in the dialog.

If a file is selected that is already being edited, MPLAB Editor activates the window showing the file.

3.3.1.3 View

Opens one or more existing files in read only mode. You can examine their contents, but not alter them. The *File > View* action is exactly as if you used the *File > Open Source* command and checked the Read Only box.

3.3.1.4 Save – Ctrl+S

Saves the contents of a file being edited to disk. If the current file is unnamed, MPLAB Editor prompts for a new filename. If a file of the same name exists, MPLAB Editor makes that the backup copy and saves the current file.

3.3.1.5 Save As

Saves the contents of a file to disk, allowing you to specify the file name.

Steps

1. Use the Drive and Directory list boxes to select the disk drive and the directory where you want to save the file.
2. Either type the name of the file into in the File Name edit control, or select the name of an existing file you want to overwrite from the list box.
3. Click the **OK** box to save the data to the file. If you specify the name of a file that already exists, MPLAB Editor confirms the overwrite.

As previously noted, the List Files of Type list at the lower left allows you to restrict the files shown in the list to those matching specific filename patterns.

3.3.1.6 Save All

Saves all altered files, stores all altered templates into template files, and saves all altered template files in a single operation.

Clicking **Cancel** in any of the dialogs that occur in this process cancels the entire operation.

3.3.1.7 Close

Closes the file being shown in the current window. If you have changed the file and haven't saved the changes to disk, MPLAB Editor prompts you to save the changes, discard them, or cancel the close.

3.3.1.8 Close All – Shift+F9

Closes all the files that you're working on. If you have changed any of the files and haven't saved the changes to disk, MPLAB Editor prompts you to save the changes, discard them, or cancel the entire operation.

3.3.1.9 Most-Recently-Used Files

The MPLAB Editor adds the Most-Recently-Used Files list to the end of the File Menu. Whenever you open a file, MPLAB Editor records the file name in the list, ordering it so that the files that have been most recently used appear at the top. Any file in the list can be reopened simply by clicking on the menu item. This is a user configuarable option. See "Customizing MPLAB After Installation" in the accompanying *MPLAB MPLAB-ICE User's Guide*.

3.3.1.10 More Files

This menu item is the list of files that you have opened recently. You can configure the number of files that MPLAB Editor records. When the Most-Recently-Used Files list contains more files than can be displayed on the File Menu, MPLAB Editor adds an extra menu command, **More Files**.

The More Files dialog shows you the entire file list, and lets you pick the file you want to open. To open a file from the list, either double-click its name in the list with the left mouse button, or select the name and press the **Open** button.

3.4 Editor Menu

The Editor Menu contains options that allow you to change text. If you have no files open, the menu options display in grey characters.

Editor

Undo	Template
Cut	Attach File
Copy	Detach File
Paste	Create File
Select All	Save File
Select Word	Insert
Delete Line	Edit
Delete EOL	New
Goto Line	Store
Find	Store As
Replace	Delete
Repeat Find	Insert Mark
Repeat Replace	Find Mark
Match Brace	Text
	Transpose
	Upper Case
	Lower Case
	Indent
	Un-Indent

3.4.1 Editor Options

3.4.1.1 Undo – Ctrl+Z

Undo the last edit action. When there is no edit action to undo, the menu command shows Can't Undo, and you cannot select the command.

The undo depth is configurable by the user. See "Customizing MPLAB After Installation" in the accompanying *MPLAB MPLAB-ICE User's Guide*.

3.4.1.2 Cut – Ctrl+X

Deletes the highlighted text in the current window, placing it on the clipboard. After this operation, you can paste the deleted text into another MPLAB Editor window, or into another Window's application.

3.4.1.3 Copy – Ctrl+C

Copies the highlighted text in the current window onto the clipboard. After this operation, you can paste the copied text into another MPLAB Editor window, or into another Window's application.

3.4.1.4 Paste – Ctrl+V

Pastes the contents of the clipboard into the current window at the position of the cursor. You can only perform this operation if the clipboard contains data in text format. MPLAB Editor does not support pasting of bitmaps or other clipboard formats.

3.4.1.5 Select All

Highlights the entire contents of the current window.

3.4.1.6 Select Word – Left Mouse Button Double-Click

Highlights the word containing the cursor.

3.4.1.7 Delete Line – Ctrl+Shift+K

Deletes the entire line containing the cursor and moves the cursor to the start of the next line.

3.4.1.8 Delete EOL – Ctrl+K

Deletes the text from the position of the cursor to the end of the line. If the cursor is at the start of a line that is completely empty, the next line is moved up to close the gap.

3.4.1.9 Goto Line – Ctrl+G

Moves the cursor to the start of a specific line.

This menu command allows you to specify either an absolute or a relative line number.

3.4.1.10 Find – F3

Searches the current window for a text string.

This command lets you specify search parameters. If there is highlighted text in the current window, the Find operation uses it as the search string.

3.4.1.11 Replace – F4

Searches the current window for a text string, and optionally replaces occurrences with another string.

This command lets you specify search parameters. If there is highlighted text in the current window, the Replace operation uses it as the search string.

3.4.1.12 Repeat Find – Shift+F3

Repeats the last search operation, without prompting for details.

3.4.1.13 Repeat Replace – Shift+F4

Repeats the last replace operation, without prompting for details.

3.4.1.14 Match Brace – Ctrl+B

This command locates braces that match each other. Position the cursor on either an opening or a closing brace.

3.4.2 Template Options

This menu item contains options that relate to handling templates.

3.4.2.1 Attach Template File

This menu item attaches a template file, loading it into memory so the templates within it can be used. The command starts a dialog that allows you to browse the files on your disk for template files, *.TPL.

3.4.2.2 Detach Template File

This command allows you to specify which attached template files you wish to detach. Once a template file is detached, you cannot use the templates it contains.

You cannot detach a template file if you're currently editing any of the templates it contains. You must save any changes you want to keep with the *Editor > Template > Store* or *Editor > Template > Store As* commands, and then close the windows.

3.4.2.3 Create Template File

This command creates a file of type *.TPL. Template files contain binary information; this is the only method by which you can create them.

3.4.2.4 Save Template File

Saves a template file to disk after you have edited its contents.

3.4.2.5 Insert Template

Inserts a template from an attached template file into a window. The template is inserted into the current window at the current cursor position.

3.4.2.6 Edit Template

Edits a template contained in an attached template file.

MPLAB Editor creates a new window with the text of the template in it. After editing the template use the *Editor > Template > Store* command to update the template.

3.4.2.7 New Template

Creates a new, empty edit window for the new template.

3.4.2.8 Store

Stores a template in the template file. This command overwrites the previous version of the template in the template file.

3.4.2.9 Store Template As

Stores a template in a template file, allowing you to specify both the template file and the template.

If you select the name of an existing template, this command overwrites the previous template's contents. MPLAB Editor confirms before executing this command.

The template file on disk is not altered by this command. To save your template changes permanently, use *Editor > Template > Save File*.

3.4.2.10 Delete Template

Deletes a template from an attached template file.

The template file on disk is not altered by this command. To save your template changes permanently, use *Editor > Template > Save File*.

3.4.2.11 Insert Template Mark

Inserts a template mark into the current template.

A template mark is a sequence of characters ("**<???>**") that marks a position in a template to put variable details.

3.4.2.12 Find Template Mark

Moves the cursor to the next template mark and highlights it.

3.4.3 Text Options

This menu shows a list of text-related commands.

3.4.3.1 Transpose – Ctrl+T

Transposes the character to the right of the cursor and the character to the left of the cursor. This command has no effect if the cursor is positioned at the start or end of a line.

3.4.3.2 Uppercase

Changes all lowercase characters in the currently highlighted text to uppercase characters.

3.4.3.3 Lowercase

Changes all uppercase characters in the currently highlighted text to lowercase characters

3.4.3.4 Indent

Moves the one or more lines of text to the right by one tab stop.

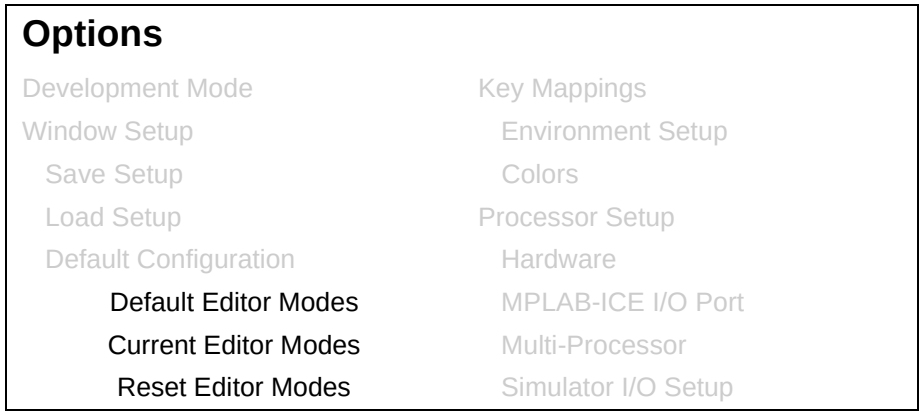
MPLAB Editor indents all highlighted lines. If no lines are highlighted, MPLAB Editor indents only the line that the cursor is in.

3.4.3.5 Unindent

Moves one or more lines of text to the left by one tab stop. Lines that do not start with white space are not affected.

3.5 Options Menu

This item on the menu bar contains various system configuration options. The options printed in black are described in this section.



3.5.1 Default Editor Modes

This command sets the modes that MPLAB Editor uses for existing and new files.

Steps

1. Determine file type.

   To set the default editor modes for files created with the *File > New Source* command, click the **New Files** button.

   To set the default editor modes for a given type of files click the **File Type** button, and either select the file type from the adjoining list, or type a new one (with a leading ".") into the edit control. This mode is applied when you use the *File > Open Source*, *File > View*, and *File > Save As* commands,

   To set default modes for file types not in the list set above, click the **Other Types** button.
2. Check the mode boxes in the dialog to specify display, formatting, printing, and file save modes.
3. Click the **Apply** button. You can set modes for other file types by returning to Step 1.
4. To record the mode settings permanently, click the **Save** button.

The modes you can set in the dialog are:

Display/Input Modes	
These settings control how text is displayed in the window and any special actions MPLAB Editor takes as you type.	
Auto Indent	Indents new lines to the same level as the preceding line.
Strip Trailing Spaces	Removes white space from the end of a line whenever you press **Enter**.
Show Line Numbers	Displays line numbers in the window.
Overwrite	Overwrites the characters under the cursor with the characters you type. Otherwise MPLAB Editor inserts characters at the cursor.
Language	Select from this list to associate a language with a window.

Screen Formatting	
These settings affect tabs and text wrapping.	
Tab Size	Defines the width of a tab character.
Soft Tabs	Inserts spaces instead of tabs when Soft Tabs is checked.
Wrap Enabled	Automatically breaks the text at the column specified in Wrap Column.

Printing	
These settings affect how MPLAB Editor prints the text in the current window. These settings become the defaults for the *File > Print* command, but can be overridden at the time of printing.	
Page Headers	Prints each page headed with the file name, the page number and the date.
Wrap Long Lines	Wraps lines that are too wide to fit the page to the next line during printing.

File Modes	
These settings are the file modes that apply to file.	
Read Only	With this mode set, you cannot change the file in any window showing it.
Backup When Saving	Makes a backup copy of any existing file of the same name when you write the file to disk.
Strip Ctrl+Z On Load	Automatically removes any **Ctrl+Z** character at the end of the file when it is loaded.
Add Ctrl+Z On Save	Automatically adds a **Ctrl+Z** character to the end of the file on disk when it saves the file.
No EOLN after last line	Writes an end-of-line terminator (**CR-LF** or **LF**) after the last character of the last line when it saves the file.

3.5.2 Current Editor Modes

This command allows you to configure the modes that apply to the current window. For a full description of the sections below please see the Default Editor Modes section.

3.5.2.1 Display/Input Modes

These settings control how text is displayed in the window and any special actions MPLAB Editor takes as you type.

3.5.2.2 Screen Formatting

These settings affect tabs and text wrapping.

3.5.2.3 Printing

These settings affect how MPLAB Editor prints the text in the current window. These settings become the defaults for the *File > Print* command, but can be overridden at the time of printing.

3.5.2.4 Current File

These settings are file modes that apply to the current file.

3.5.3 Reset Editor Modes

Resets the modes that apply to the current window to the default values.

NOTES:

Chapter 4. MPLAB Editor Default Key Commands

4.1 Introduction

This appendix describes the default key commands specific to the MPLAB Editor and lists the equivalent menu command (if any).

The key commands perform the most common operations quickly. These key commands can be modified to suit individual preferences. By default, no prefix keys are enabled, so that only single keys are available for mapping.

For a table that lists the keys with the names of the functions used in the key mapping dialog, see Appendix A: MPLAB Key Mapping Functions.

4.2 Highlights

The categories for the default key commands are:

- **Function Keys**
- **Movement Keys**
- **Control Keys**
- **Formatting and Editing Keys**

4.3 Function Keys

F1	Opens the MPLAB Editor Help File at the Contents topic.
F2	Executes the *Debug > Break Settings* command to open the Break Settings dialog.
F3	Executes the *Edit > Find* command to find strings.
F4	Executes the *Edit > Replace* command to replace strings.
F5	Executes the *Debug > Halt* command to halt debugging.
F6	Executes the *Debug > Reset* command to issue a reset to the emulated or simulated processor.
F7	Executes the *Debug > Step* command to execute a single opcode from program memory.
F8	Executes the *Debug > Step Over* command to step over a call instruction in program memory.
F9	Executes the *Debug > Run* command to issue a run to the emulated/simulated processor.
F11	Executes the *Execute > DOS Command To Window* command to run a DOS command and capture its output in a window.
Shift+F3	Executes the *Edit > Repeat Last Find* command to repeat the last find operation.

Shift+F4	Executes the *Edit > Repeat Last Replace* command to repeat the last replace operation.
Shift+F5	Executes the *Debug > Halt Trace* command to halt the execution trace for the simulated processor.
Shift+F7	Executes the *Debug > Step Trace* command to step the execution trace for the simulated processor.
Shift+F9	Executes the *File > Close All* command to close all files and windows.
Ctrl+F2	Executes the *Project > Open Project* command to open the Open Project Dialog.
Ctrl+F3	Executes the *Project > Edit Project* command to open the Edit Project Dialog.
Ctrl+F4	Executes the *Project > Make Project* command to initiate a "make" for the current project.
Ctrl+F5	Executes the *Project > Build All* command to build all the source files for the current project.
Ctrl+F6	Executes the *Debug > Development Mode* command to open the Development Mode dialog.
Ctrl+F7	Executes the *Options > Environment Setup* to open the Environment Setup dialog.
Ctrl+F8	Executes the *Windows > Show Symbol List* command to open the Symbol List dialog.
Alt+F4	Executes the *File > Exit* command to end your MPLAB Editor session.
Shift+Ctrl+F2	Executes the *Debug > Clear Program Memory* command to clear all of program memory to an "erased" state.
Shift+Ctrl+F3	Executes the *Debug > System Reset* command to reset the entire emulator system.
Shift+Ctrl+F5	Executes the *Debug > POR Reset Emulation* command to open the POR Reset dialog.

4.4 Movement Keys

For these keys, adding Shift to the combination causes a selection to be extended.

Note that where the **Alt** key is combined with either an arrow key, or one of **Home**, **End**, **PgDn**, **PgUp**, Ins or **Del**, you must use the keys in the extended key areas, and not those in the numeric keypad.

Up	Moves the cursor up by one line
Shift+Up	Moves the cursor up by one line, extending the highlighting
Down	Moves the cursor down by one line
Shift+Down	Moves the cursor down by one line, extending the selection
Left	Moves the cursor left by one character
Shift+Left	Moves the cursor left by one character, extending the selection
Ctrl+Left	Moves the cursor left by one word
Ctrl+Shift+Left	Moves the cursor left by one word, extending the selection
Right	Moves the cursor right by one character
Shift+Right	Moves the cursor right by one character, extending the selection
Ctrl+Right	Moves the cursor right by one word
Ctrl+Shift+Right	Moves the cursor right by one word, extending the selection
PgDn	Moves the cursor down by one page
Shift+PgDn	Moves the cursor down by one page, extending the selection
Ctrl+PgDn	Moves the cursor to the start of the last line in the window
Ctrl+Shift+PgDn	Moves the cursor to the start of the last line in the window, extending the selection
PgUp	Moves the cursor up by one page
Shift+PgUp	Moves the cursor up by one page, extending the selection
Ctrl+PgUp	Moves the cursor to the start of the first line of the window
Ctrl+Shift+PgUp	Moves the cursor to the start of the first line of the window, extending the selection
Home	Moves the cursor to the start of the line
Shift+Home	Moves the cursor to the start of the line, extending the selection

Ctrl+Home	Moves the cursor to the start of the file
Ctrl+Shift+Home	Moves the cursor to the start of the file, extending the selection
Alt+Home	Moves the cursor to the first non-white-space character in the current line
Alt+Shift+Home	Moves the cursor to the first non-white-space character in the current line, extending the selection
End	Moves the cursor to the end of the line
Shift+End	Moves the cursor to the end of the line, extending the selection
Ctrl+End	Moves the cursor to the end of the file
Ctrl+Shift+End	Moves the cursor to the end of the file, extending the selection

4.5 Control Keys

Ctrl+B	Executes the *Edit > Text Match Brace* command to move the cursor to the brace character matching the one currently under it
Ctrl+Shift+B	Executes the *Edit > Text Match Brace Select* command to move the cursor to the brace character matching the one currently under it and highlight all the text between and including the brace characters
Ctrl+C	Executes the *Edit > Copy* command to copy selected text to the clipboard
Ctrl+G	Executes the *Edit > Goto Line* command to move the cursor to a specific line
Ctrl+H	Deletes the character to the left of the cursor
Ctrl+I	Inserts a TAB character, or the required number of spaces to bring the cursor to the next TAB stop, depending on whether the current window's Window Mode is set for hard or soft tabs.
Ctrl+K	Executes the *Edit > Delete To End Of Line* command to delete everything from the cursor position to the end of the line
Ctrl+Shift+K	Executes the *Edit > Delete Line* command to delete the entire line that the cursor is in
Ctrl+N	Executes the *File > New* command to create a new, empty edit window
Ctrl+O	Executes the *File > Open* command to open an existing file
Ctrl+Shift+O	Splits the current line at the position of the cursor, leaving the cursor unmoved

Ctrl+P	Executes the *File > Print* command to print the file showing in the current window
Ctrl+Q	Executes the *Edit > Text Insert ASCII Code* command to insert a character specified by its ASCII code number
Ctrl+S	Executes the *File > Save* command to save the current file to disk
Ctrl+T	Executes the *Edit > Text Transpose Characters* command to transpose the character under the cursor with the one to its left
Ctrl+V	Executes the *Edit > Paste* command to paste data from the clipboard into the current window
Ctrl+W	Executes the *Window > Select* command to let you choose between many open windows
Ctrl+Shift+W	Executes the *Edit > Text Widen Brace Select* command to highlight all the text between and including the closest pair of brace characters that encompasses either the currently-highlighted text, or the character under the cursor if no text is selected
Ctrl+X	Executes the *Edit > Cut* command to cut selected text to the clipboard
Ctrl+Z	Executes the *Edit > Undo* command to undo the last edit action

4.6 Formatting and Editing Keys

Enter	Inserts a new line
BackSpace	Deletes the character to the left of the cursor
Del	Deletes the character to the right of the cursor
Shift+Del	Executes the *Edit > Cut* command to cut selected text to the clipboard
Ins	Toggles the current window between Insert and Overwrite modes
Shift+Ins	Executes the *Edit > Paste* command to paste data from the clipboard into the current window
Ctrl+Ins	Executes the *Edit > Copy* command to copy selected text to the clipboard
Keypad 5	Cancels the current selection, removing the highlight
Tab	Inserts a tab character, or the required number of spaces to bring the cursor to the next tab stop, depending on whether the current window's Window Mode is set for hard or soft tabs.

NOTES:

Chapter 5. MPLAB Editor Error Messages

5.1 Introduction

This appendix lists the most common error messages you might receive while using the MPLAB Editor.

5.2 Error Messages

5.2.1 Cannot Detach Template File

A template file cannot be detached if one or more of the templates it contains is being edited.

Steps

1. For each of the windows containing such templates, use the *Editor > Template > Store* or *Editor > Template > Store As* commands to save changes you want to keep. Then close the window with the *File > Close* command.
2. Preserve the changes to the template file itself with the *Editor > Template > Save File* command.

The template file can then be detached.

5.2.2 Create A Further Backup Level

MPLAB Editor has detected that the file you are about to overwrite in this operation is already in a directory that contains backups of overwritten files. You may not wish to make a further backup of the file.

If you click the **Yes** button, MPLAB Editor continues as normal and makes a backup copy of the file before saving the new data to it. This involves creating a new sub-directory (called by default "$MPLABBK") to contain it; depending on how deep your directory structure is at this point, the full path name of the directory, or of the backed up file within it, may exceed the limits imposed by the operating system.

If you click the **No** button, MPLAB Editor does not make a backup copy of the existing file, but simply overwrites it with the new data. The current contents of the disk file are irrevocably lost.

If you click the **Cancel** button, MPLAB Editor abandons the save operation and does not change the disk file.

5.2.3 Default Printer Not Configured

Whenever you exit from MPLAB Editor, the name of the printer you last used is recorded in the MPLAB Initialization file, MPLAB.INI. This device is used as the default printer for the MPLAB Editor the next time you run MPLAB.

While starting this session, MPLAB Editor has detected that the printer/port combination you used last time is no longer configured in your system. MPLAB Editor now uses the Windows default printer unless you specify otherwise. You can check that this is what you want, and correct it if not, by using the *File > Print Setup* command.

5.2.4 Invalid ASCII Code

You have entered an invalid ASCII code number in the dialog. Make sure that your entry is a decimal number, in the range 1 to 255.

5.2.5 Invalid Line Number

You have entered an invalid line number in the **Line To Go To** edit control. Check that what you have typed is either a numeric string, or is the word **end.**

If you have typed a number preceded by either **+** or **-** to specify a line number relative to the line that the cursor is now in, verify that the resulting line number is not less than 1, or greater than the number of the last line in the file.

5.2.6 Save Changes To File?

You have changed this file since the last time you saved it to disk. Before MPLAB Editor closes it, you have a chance to update the disk copy.

Yes Click **Yes** to save the contents of the window to disk. If the file hasn't yet got a name, you'll see a dialog asking you to specify the file to save to.

No Click **No** to irretrievably discard the changes you've made since the last save.

Cancel Click **Cancel** to abandon the operation completely and return to editing the window.

5.2.7 Save Changes To Template?

You have changed this template since the last time you stored it in the in-memory copy of a template file. Before MPLAB Editor closes it, you have a chance to update the in-memory template file.

Yes Click **Yes** to save the contents of the window to an in-memory copy of the template file. If the template has not yet been named, MPLAB Editor will prompt you to specify a name and template file location.

No Click **No** to discard the changes you've made since the last save.

Cancel Click **Cancel** to abandon the operation completely and return to editing the window.

NOTES:

Chapter 6. MPLAB Text Editor Command Line Options

6.1 Introduction

This appendix describes command line options that you can use when starting MPLAB.

6.2 Command Line Options

When you start MPLAB, you can give several options on the command line to control its actions. These options must follow the name of the MPLAB Text Editor executable itself, and precede any file names.

6.2.1 Options for Opening Files

MPLAB Starts MPLAB. The example below opens the files data.txt and results.txt in the current directory.

```
Example: MPLAB data.txt results.txt
```

/v The /v option opens files for editing in read-only mode.

```
Example: MPLAB /v data.txt results.txt
```

6.2.2 Configuration Options

/k The configuration options affect how MPLAB configures itself at start-up.

The /k option starts a new instance of MPLAB and loads the key mappings contained in the file mapping.key in the current directory.

By default, MPLAB will look for a file called MPLAB.KEY in the Windows directory at start-up, and will load the key mappings contained in MPLAB.KEY.

```
Example: MPLAB /k mapping.key
```

The /k option specifies a key map file to be loaded on start-up instead of the MPLAB.KEY file in the Windows directory.

Pathname can specify any valid key map file; TEnter the single character "-" (minus) to disable MPLAB from loading a key mapping, but use built-in default key map values.

Using this option will disable the default MPLAB.key file. The Save button in the key mapping dialog will not be available when you change the mapping. To save to this file, use the Save As button.

To load mappings from a different file on start-up, or to force MPLAB to use the built-in defaults described in this help file, you can use the "/k" command line option to name any key map file. For example, starting MPLAB with a command line:

```
MPLAB/k c:\kmaps\uemap.key
```

would load the key mappings in the file c:\key\maps\uemap.key. If you have a key map file MPLAB.KEY, but want to use the built-in key mappings described in this help file instead, use the "/k" option and give the filename as the single character "-" (minus).

Note: When you use the "/k" option, the dialog started by the *Options > Key Mappings* command will not save changed mappings to the file you specify, or to the standard key mapping file, by default. When saving, you will need to give an explicit file name.

Part 3 – Appendices

Appendix A. MPLAB Key Mapping Functions

A.1 Introduction

This appendix lists the available MPLAB key mapping functions.

A.2 MPLAB Key Mapping Functions

Key Map Function	Definition	Default Key Assignment
CursorBottomOfWindow	"Move Cursor to Bottom of Window"	Ctrl+PgDn
CursorBottomOfWindowSelect	"Move Cursor to Bottom of Window Selecting"	Ctrl+Shift+PgDn
CursorDown	"Move Cursor Down"	Down
CursorDownSelect	"Move Cursor Down Selecting"	Shift+Down
CursorEndOfFile	"Goto End of File"	Ctrl+End
CursorEndOfFileSelect	"Goto End of File Selecting"	Ctrl+Shift+End
CursorEndOfLine	"Move Cursor to End of Line"	End
CursorEndOfLineSelect	"Move Cursor to End of Line Selecting"	Shift+End
CursorLeft	"Move Cursor Left"	Left
CursorLeftSelect	"Move Cursor Left Selecting"	Shift+Left
CursorLeftWord	"Move Cursor Left by Word"	Ctrl+Left
CursorLeftWordSelect	"Move Cursor Left by Word Selecting"	Ctrl+Shift+Left
CursorPageDown	"Page Down"	PgDn
CursorPageDownSelect	"Page Down Selecting"	Shift+PgDn
CursorPageUp	"Page Up"	PgUp
CursorPageUpSelect	"Page Up Selecting"	Shift+PgUp
CursorRight	"Cursor Right"	Right
CursorRightSelect	"Cursor Right Selecting"	Shift+Right
CursorRightWord	"Cursor Right by Word"	Ctrl+Right
CursorRightWordSelect	"Cursor Right by Word Selecting"	Ctrl+Shift+Right
CursorStartOfFile	"Cursor Start of File"	Ctrl+Home
CursorStartOfFileSelect	"Cursor Start of File Selecting"	Ctrl+Shift+Home
CursorStartOfLine	"Cursor Start of Line"	Home
CursorStartOfLineSelect	"Cursor Start of Line Selecting"	Shift+Home
CursorStartOfText	"Cursor Start of Text"Alt Home	Alt+Home
CursorStartOfTextSelect	"Cursor Start of Text Selecting"	Alt+Shift+Home
CursorTopOfWindow	"Cursor Top of Window"	Ctrl+PgUp

Key Map Function	Definition	Default Key Assignment
CursorTopOfWindowSelect	"Cursor Top of Window Selecting"	Ctrl+Shift+PgUp
CursorUp	"Cursor Up"	Up
CursorUpSelect	"Cursor Up Selecting"	Shift+Up
DebugAnimate	"Animate"	
DebugBreak	"Set Break Settings"	F2
DebugCenterDebug	"Center Debug Location"	
DebugChangePC	"Change Program Counter"	
DebugClearAll	"Clear All Qualifiers"	
DebugClearMemory	"Clear Program Memory"	Ctrl+Shift+F2
DebugConditionalBreak	"Conditional Break"	
DebugExecuteOpcode	"Execute an Opcode"	
DebugHalt	"Halt the Processor"	F5
DebugHaltTrace	"Halt the Trace"	Shift+F5
DebugPerformanceAnalysis	"Performance Analysis"	
DebugPORReset	"Power-On-Reset"	Ctrl+Shift+F5
DebugStep	"Step"	F7
DebugReset	"Reset Processor"	F6
DebugRun	"Run"	F9
DebugStepOver	"Step Over"	F8
DebugStepTrace	"Step The Trace Window"	Shift+F7
DebugAsyncStim	"Asynchronous Stimulus	
DebugClockStim	"Clock Stimulus"	
DebugPinStim	"Pin Stimulus"	
DebugRegStim	"Register Stimulus"	
DebugSystemReset	"System Reset"	Ctrl+Shift+F3
DebugTrace	"Trace Settings"	
DebugTrigger	"Trigger Settings"	
DebugUpdateRegisters	"Update Registers"	
EditCancelSelection		keypad5
EditClearUndo	"Forgets details of all stored undo actions"	
EditCopy	"Copies highlighted text to the clipboard"	Ctrl+C Ctrl+Ins
EditCut	"Cuts highlighted text to the clipboard"	Ctrl+X Shift+Del

Appendix A. MPLAB Key Mapping Functions

Key Map Function	Definition	Default Key Assignment
EditDeleteBackwards	“Delete Character Backwards”	Backspace Ctrl+H
EditDeleteForwards	“Delete Forwards”	Del
EditDeleteLine	“Deletes the entire line containing the cursor”	Ctrl+Shift+K
EditDeleteSelection	“Delete Selection”	
EditDeleteToEndOfLine	“Deletes from the cursor to end of line”	Ctrl+K
EditFind	“Searches for a text string”	F3
EditGotoLine	“Moves the cursor to a specific line”	Ctrl+G
EditInsertHardTab	“Insert Hard Tab”	
EditInsertSoftTab	“Insert Soft Tab”	
EditInsertTab	“Insert Tab”	Tab Ctrl+I
EditMarkUnchanged	“Mark File As Unchanged”	
EditNewLine	“Insert New Line”	Enter Ctrl+M
EditPaste	“Pastes the clipboard at the cursor position”	Ctrl+V Shift+Ins
EditRepeatLastFind	“Repeats the last search action exactly”	Shift+F3
EditRepeatLastReplace	“Repeats the last replace action exactly”	Shift+F4
EditReplace	“Replaces a text string”	F4
EditSelectAll	“Highlights all the text in the current window”	
EditSelectLine	“Select Line”	
EditSelectWord	“Highlights the word containing the cursor”	
EditShowCursor	“Scrolls the window to bring the cursor into view”	
EditShowNextLine	“Show Next Line”	
EditShowNextPage	“Show Next Page”	
EditShowPreviousLine	“Show Previous Line”	
EditShowPreviousPage	“Show Previous Page”	
EditSplitLine	“Split Line”	Ctrl+Shift+O
EditTextIndent	“Moves text right by one tab stop”	
EditTextInsertASCIICode	“Inserts an arbitrary character code”	Ctrl+Q
EditTextLowercaseSelection	“Converts the highlighted text to lower case”	
EditTextMatchBrace	“Moves to a matching brace character”	Ctrl+B
EditTextMatchBraceSelect	“Moves to a matching brace character and highlights”	Shift+Ctrl+B

Key Map Function	Definition	Default Key Assignment
EditTextTransposeCharacters	"Swaps the characters to left and right of the cursor"	Ctrl+T
EditTextUnIndent	"Moves text left by one tab stop"	
EditTextUppercaseSelection	"Converts the highlighted text to upper case"	
EditTextWidenBraceSelect	"Highlights the next largest braced area of text"	Shift+Ctrl+W
EditUndo	"Undoes the last edit action"	Ctrl+Z
ExecDosCommand	"Runs a DOS command and captures output"	F11
ExecRepeatDosCommand	"Repeats the last DOS command-with-capture"	Ctrl+F11
FileAbandon	"Abandon File"	
FileClose	"Closes the file in the current window"	
FileCloseAll	"Closes all open files"	Shift+F9
FileExit	"Ends your MPLAB session"	Alt+F4
FileImportDownloadToMemory	"Download A Hex file to the Engine"	
FileImportDownloadToTarget	"Download A Hex file to the Target"	
FileImportReadTarget	"Copy Engine Memory to Target"	
FileInsert	"Inserts a file at the position of the cursor"	
FileName	"Changes the file name for the current window"	
FileNew	"Creates a new, empty edit window"	Ctrl+N
FileOpen	"Opens an existing file"	Ctrl+O
FilePrint	"Prints the current file"	Ctrl+P
FilePrintSetup	"Changes details of the current printer"	
FileSave	"Saves the current file to disk"	Ctrl+S
FileSaveAll	"Saves all open files to disk"	
FileSaveAs	"Saves the current file to disk"	
FileSaveHex	"Save Hex File"	
FileSaveTrace	"Save Trace File"	
FileSimulatorStimulus	"Load Simulator Stimulus File"	
FileView	"Opens an existing file in read-only mode"	
FileWrite	"Writes the current file to disk"	
HelpAbout	"Gives information about this MPLAB version"	
HelpBugs	"MPLAB Bug List"	
HelpCommands	"Gives help on MPLAB commands"	
HelpContents	"Enters the MPLAB help file at the Contents screen"	F1
HelpEditor	"Editor Help"	

Appendix A. MPLAB Key Mapping Functions

Key Map Function	Definition	Default Key Assignment
HelpMpasm	“MPASM Help”	
HelpMpc	“MPLAB-C Help”	
HelpOnHelp	“Gives help on using the help system”	
HelpPICmicro	“PICmicro Users Guide”	
HelpReleaseNotes	“Release Notes”	Shift+F1
OptionsColors	“Edit Color Options”	
OptionsCommunicationsPort	“Set Communications Address”	
OptionsCurrent	“Sets modes for the current window and file”	
OptionsDefault	“Sets default modes for files”	
OptionsDevelopmentMode	“Select Development Mode”	Ctrl+F6
OptionsEnvironmentSetup	“Setup Environment”	Ctrl+F7
OptionsHardware	“Select Hardware Options”	
OptionsKeyMapping	“Changes the mapping of keys to commands”	
OptionsLoadSetup	“Load Setup File”	
OptionsMultiProcessor	“Setup Multi-Processor”	
OptionsPreferences	“Sets MPLAB configuration options”	
OptionsResetModes	“Sets the current file/window modes to default values”	
OptionsSaveSetup	“Save Setup File”	
OptionsScreenFontANSI	“Sets the screen font to the standard ANSI fixed-pitch font”	
OptionsScreenFontOEM	“Sets the screen font to the standard OEM font”	
OptionsScreenFontOther	“Selects the screen font from all available fixed-pitch fonts”	
OptionsScreenFontSystem	“Sets the screen font to the standard system font”	
OptionsSimulatorIOSetup	“Setup Simulator I/O”	
OptionsToggleInsertMode	“Toggle Insert Mode”	Ins
OptionsToggleLineNumbers	“Toggle Line Numbers”	
OptionsToggleStatusBar	“Hides or shows the status bar”	
OptionsToolbarBottom	“Moves the tool bar to the bottom of the window”	
OptionsToolbarFloat	“Makes the tool bar a floating window”	
OptionsToolbarHide	“Makes the tool bar invisible”	
OptionsToolbarLeft	“Moves the tool bar to the left of the window”	
OptionsToolbarRight	“Moves the tool bar to the right of the window”	
OptionsToolbarShow	“Makes the tool bar visible”	

Key Map Function	Definition	Default Key Assignment
OptionsToolbarTop	"Moves the tool bar to the top of the window"	
ProjectBuildAll	"Build Full Project"	Ctrl+F10
ProjectCloseProject	"Close Project"	
ProjectCompileSingle	"Compile Single File"	Alt+F10
ProjectEditProject	"Edit the Project Definition"	Ctrl+F3
ProjectMakeProject	"Make the Current Project"	F10
ProjectMakeSetup	"Setup the Project Make"	
ProjectNewProject	"Create a New Project"	
ProjectOpenProject	"Open a Project"	Ctrl+F2
ProjectSaveProject	"Save the Current Project"	
SwapToolbar	"Swap Toolbar"	
SysSetMenuMode	"Set Menu Mode"	
TemplateDelete	"Deletes a template from a template file"	
TemplateEdit	"Edits a template from a template file"	
TemplateFileAttach	"Loads a template file for use"	
TemplateFileCreate	"Creates an empty template file"	
TemplateFileDetach	"Releases an attached template file"	
TemplateFileSave	"Saves an altered template file to disk"	
TemplateFindMark	"Searches for a template marker in the current window"	
TemplateInsert	"Inserts a template at the position of the cursor"	
TemplateInsertMark	"Inserts a template marker at the position of the cursor"	
TemplateNew	"Creates a new window for a template"	
TemplateStore	"Saves a template into a template file"	
TemplateStoreAs	"Saves a template into a template file"	
ToolsEmulatorConfiguration	"Setup Emulator Configuration"	
ToolsProgramHeader	"Program Emulator Header"	
ToolsProgramPod	"Program Emulator Pod"	
ToolsVerifyEmulator	"Verify Emulator Components"	
WindowAbsoluteListing	"Absolute Listing"	
WindowArrangeIcons	"Arranges all iconic windows neatly"	
WindowCascade	"Arranges windows in a cascade pattern"	
WindowClose	"Closes the current window"	Ctrl+F4
WindowDuplicate	"Makes a duplicate of the current window"	

Appendix A. MPLAB Key Mapping Functions

Key Map Function	Definition	Default Key Assignment
WindowEeprom	“EEPROM Memory Window”	
WindowFileRegisters	“File Register Memory”	
WindowIconize	“Iconize Window”	
WindowIconizeAll	“Makes all windows into icons”	
WindowLoadWatch	“Load a Watch Window”	
WindowMaximize	“Maximize Window”	
WindowModify	“Modify Window”	
WindowNewWatch	“Create New Watch Window”	
WindowNext	“Activates the next non-iconic window”	
WindowProgramMemory	“Program Memory Window”	
WindowRestore	“Restore Window”	
WindowSelect	“Chooses a window to activate from a list”	Ctrl+W
WindowSpecialFunctionRegisters	“Special Functions Register Window”	
WindowStack	“Stack Window”	
WindowStopwatch	“Stopwatch Window”	
WindowSymbolList	“Symbol List Window”	Ctrl+F8
WindowTileHorizontal	“Tiles windows to maximize height”	
WindowTileVertical	“Tiles windows to maximize width”	
WindowTrace	“Trace Window”	
WindowWiden	“Maximizes the width of the current window”	

NOTES:

Appendix B. Customizing MPLAB After Installation

B.1 Introduction

MPLAB uses the initialization file, MPLAB.INI, to record the values that carry over from one session to another. You can also enter various options in MPLAB.INI to customize how MPLAB operates.

MPLAB.INI is held in your Windows directory, and can be edited with MPLAB or any other editor. However, MPLAB will change many parts of the file when it exits. You should not edit sections of MPLAB.INI with the MPLAB Editor.

B.2 Initialization File Format

The information contained in MPLAB.INI is in the standard format used by all Windows applications. MPLAB.INI is divided into sections with section titles enclosed by square brackets. The editable section of MPLAB.INI is in the options section.

All lines following a section name are treated as a group. Each section contains lines of the general form:

key string=argument,argument,....

The key string is text (which may not include spaces) that MPLAB uses to locate a particular setting. The comma-separated arguments that follow the "=" sign provide the data relevant to the key.

Note: Both the section (options) and the key string are case sensitive. If you place a key string in the list, make sure to use all lower case letters, otherwise MPLAB will not find the change.

```
[options]
mru-list-size=5
mru-files-shown=5
open-maximized=0
allow-save-always=0
backup-mode=5
```

Figure D.1 MPLAB.INI Options Section

B.3 Options Section

This section contains various items that allow you to customize MPLAB to suit your own preferences. MPLAB never alters the contents of this section itself.

You can safely use MPLAB to alter the contents of this section. Any changes you make will not take effect until you terminate MPLAB and restart.

Key String	Possible Values	Default
allow-save-always	0 - *File > Save* enabled only after a file is changed. 1 - *File > Save* always enabled.	0
auto-file-action	0 - Perform no action when MPLAB opened. 1 - Perform a *File > New* when MPLAB opened. 2 - Perform a *File > Open* when MPLAB opened.	0
backup-directory	Eight character name using valid DOS characters (see backup-mode).	$MPLABBK
backup-mode	0 - Backups are in same directory as source, but with extension "*.$$$". 1 - Backups are in subdirectory of source file using name from backup-directory option 5 - Create no backups.	0
deselect-on-copy	0 - Leave selection highlighted after copy. 1 - Remove highlight after copy.	0
dragdrop-flip	0 - Drag-and-drop performs a move operation by default and copy if Ctrl key is pressed. 1 - Drag-and-drop copies by default, and moves with Ctrl key.	0
max-undo-actions	Number of undo actions, 8 to 128	32
max-vertical-tile	0 - Leave room for one row of icons at bottom of screen 1 - Cover entire screen	0
minimize-on-empty	0 - Do nothing when all child windows are closed 1 - Minimize MPLAB when all child windows are closed	0
mru-files-shown	Number of files shown on File menu, 0 to 8.	5
mru-list-size	Number of files MPLAB will record for File menu, 0 to 64	5
open-maximized	0 - Open window in a default state. 1 - Open window maximized always. 2 - Open window maximized if current window is maximized. 11 - Open window maximized if MPLAB is maximized. 12 - Open window maximized if MPLAB and current window are both maximized.	0
save-clears-undo	0 - Do not clear undo buffer on save. 1 - Clear undo buffer on save	1

save-find-strings	0 - Do not record find strings in MPLAB.INI file. 1 - Record last eight string used in Find (or Replace) dialog in MPLAB.INI file.	0
select-search-match	0 - Do not highlight matching string when found. 1 - Highlight matching string when found. Note: You can override this value in the Find Dialog.	1
sound-beep	0 - Do not generate beeps for error messages. 1 - Generate beeps for error messages. Note: MPLAB always generates beeps for conditions where there is no other error indication, ie. attempting to insert something into a read-only window.	1
start-maximized	0 - Start MPLAB in a default state. 1 - Start MPLAB in a maximized state. Note: If this options does not exist in MPLAB.INI, MPLAB starts in the state it was at the end of the last session.	0
track-vertical-thumb-tack	0 - Update text when vertical scroll bar is released. 1 - Scroll text to follow vertical scroll bar.	1
use-dragdrop	0 - drag-and-drop not available. 1 - drag-and-drop available.	1

NOTES:

Appendix C. 12-Bit Core Device Simulator Issues

C.1 Introduction

This appendix discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using 12-Bit Core devices.

C.2 12-Bit Core Devices

At the time this document was generated, the following list of devices were available in the PIC16C5X family:

- PIC12508/509
- PIC12CE518/519
- PIC16C52/54/55/56/57/58
- PIC16C505

The above list implies all device variants, i.e., ROM versions (PIC16CR5X), and device revisions (PIC16C5XA).

C.3 I/O Pins

When modifying pins either manually or via the stimulus file, use the following pin names only. These are the only ones that the MPLAB-SIM simulator recognizes as valid I/O pins. Because the pinout is device-specific, some pins (for example RC0 on a PIC16C54) will not be available on all parts in this family.

- $\overline{\text{MCLR}}$
- T0CKI
- RA0-RA3
- RB0-RB7
- RC0-RC7

These pin names can be used in the Modify window (*Window > Modify*) and in stimulus files.

C.4 CPU Mode

C.4.1 Reset and Sleep Conditions

All reset conditions are supported by the MPLAB-SIM simulator.

An $\overline{\text{MCLR}}$ reset during normal operation or during SLEEP can easily be simulated by driving the $\overline{\text{MCLR}}$ pin low (and then high) via the stimulus file or by using *Debug > Run > Reset*.

A WDT time-out reset is simulated when WDT is enabled and proper prescaler is set (by initializing OPTION register appropriately) and WDT actually overflows. WDT time-out period (with prescale = 1) is approximated at 18 ms (to closest instruction cycle multiple).

The Time-out ($\overline{\text{TO}}$) and Power-down ($\overline{\text{PD}}$) bits in the Status register reflect appropriate reset condition. This feature is useful for simulating various power-up and time out forks in the user code.

C.4.2 Watch Dog Timer

The Watchdog timer is fully simulated in the MPLAB-SIM simulator. Because it is fuse-selectable on the device, it must be enabled by *Options > Processor Setup > Hardware* in the MPLAB-SIM simulator. The period of the WDT is determined by the prescaler settings in the OPTION register. The basic period (with prescaler = 1) is approximated at 18 ms (to closest instruction cycle multiple).

C.5 Peripherals

Along with providing core support, the TIMER0 timer/counter module is fully supported. In both internal and external clock modes. The prescaler is made readable and writable as 'T0PRE' symbol.

It is important to remember that because the MPLAB-SIM simulator executes on instruction cycle boundaries, resolutions below 1 T_{CY} cannot be simulated.

Appendix D. 14-Bit Core Device Simulator Issues

D.1 Introduction

This appendix discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using 14-Bit Core devices.

D.2 14-Bit Core Devices

At the time this document was generated, the following list of devices were available in the 14-Bit Core family:

- PIC12C671/672/673/674
- PIC14000
- PIC16C554/558
- PIC16C61/62/63/64/65/66/67
- PIC16C620/621/622
- PIC16CE623/624/625
- PIC16C642/662
- PIC16C710/711/715
- PIC16C71/72/73/74/76/77
- PIC16C773/774
- PIC16F83/84
- PIC16F877/876/873/874
- PIC16C923/924

The above list implies all device variants, i.e., ROM versions (PIC16CRXX), and device revisions (PIC16CXXA).

D.3 I/O Pins

The 14-Bit Core devices have I/O pins multiplexed with other peripherals (and therefore referred by more than one name). When modifying pins either manually or via the stimulus file, use the following pin names only. These pin names are the only ones that the MPLAB-SIM simulator recognizes as valid I/O pins. (Pins are only available as described in the data sheet of the specific device.)

- $\overline{\text{MCLR}}$
- RA0-RA5
- RB0-RB7
- RC0-RC7
- RD0-RD7
- RE0-RE7

These pin names can be used in the Modify window (*Window > Modify*) and in stimulus files.

D.4 Interrupts

The MPLAB-SIM simulator supports all interrupts of 14-Bit Core devices. (Peripherals are only available as described in the data sheet of the particular device.)

- Timer0 overflow
- Timer1 overflow
- Timer2
- CCP1
- CCP2
- SSP (in SPI mode ONLY)
- Change on Port RB <7:4>
- External interrupt from RB0/INT pin
- Parallel Slave Port
- Comparators
- A/D complete
- EEPROM write complete

D.5 CPU Model

D.5.1 Reset Conditions

All reset conditions are supported by the MPLAB-SIM simulator.

An $\overline{\text{MCLR}}$ reset during normal operation or during SLEEP can easily be simulated by driving the $\overline{\text{MCLR}}$ pin low (and then high) via the stimulus file or by using MPLAB *Debug > Run > Reset*.

The Time-out ($\overline{\text{TO}}$) and Power-down ($\overline{\text{PD}}$) bits in the Status register reflect appropriate reset condition. This feature is useful for simulating various power-up and time out forks in the user code.

D.5.2 Sleep

The MPLAB-SIM simulator simulates the SLEEP instruction, and will appear “asleep” until a wake-up from sleep condition occurs. For example, if the Watchdog timer has been enabled, it will wake the processor up from sleep when it times out (depending upon the prescaler setting in the OPTION register).

Another example of a wake-up-from-sleep condition, would be Timer1 wake-up from sleep. In this case, when the processor is asleep, Timer1 would continue to increment until it overflows, and if the interrupt is enabled, will wake the processor on overflow and branch to the interrupt vector.

D.5.3 Watch Dog Timer

The Watchdog timer is fully simulated in the MPLAB-SIM simulator. Because it is fuse-selectable on the device, it must be enabled with *Options > Processor Setup > Hardware*. The period of the WDT is determined by the prescaler settings in the OPTION register. The basic period (with prescaler = 1) is approximated at 18 ms (to closest instruction cycle multiple).

D.6 Special Registers

To aid in debugging this device, certain items that are normally not observable have been declared as "special" registers. Prescalers and postscalers cannot be declared in your code as "registers", so there are special labels that show up in the Special Function Registers window.

The following are special symbols that are available for the processors in the 14-bit core family. (Consult the data sheet for the particular device you are using for information on which symbols are implemented.)

- T0PRE - Prescaler for timer0
- T1PRE - Prescaler for timer1
- T2PRE - Prescaler for timer2
- T2POS - Postscaler for timer2
- CCP1PRE - Prescaler for CCP1
- SPIPRE - Prescaler for SPI
- SSPSR - SSP Shift register

D.7 Peripherals

D.7.1 Peripherals supported

Along with providing core support, the following peripheral modules (in addition to general-purpose I/O) are supported. (Consult the data sheet for the particular device you are using for information on which symbols are implemented.)

- Timer0
- Timer1
- Timer2
- CCP1
- CCP2
- Parallel Slave Port
- SSP (in SPI Mode only)
- Comparators
- A/D (Limited)

D.7.2 TIMER0

Timer0 (and the interrupt it can generate on overflow) is fully supported by the MPLAB-SIM simulator, and will increment by the internal or external clock. Clock input must have a minimum high time of 1 TCY and a minimum low time of 1 TCY due to stimulus requirements. The prescaler for Timer0 is made accessible as T0PRE.

D.7.3 TIMER1

Timer1 in its various modes is supported by the MPLAB-SIM simulator, except when running in counter mode by an external crystal. The MPLAB-SIM simulator supports timer1 interrupts generated on overflow, and interrupts generated by wake-up from sleep. The prescaler for Timer1 is viewable as T1PRE in the Special Function Registers window. The external oscillator on RC0/RC1 is not simulated, but a clock stimulus can be assigned to those pins.

D.7.4 TIMER2

Timer2 and the interrupt that can be generated on overflow are fully supported by the MPLAB-SIM simulator, and both the prescaler and postscaler for Timer2 are viewable as T2PRE and T2POS.

D.7.5 CCP1 and CCP2

CAPTURE

The MPLAB-SIM simulator fully supports capture and the interrupt generated. The prescaler for the CCP module is viewable CCP1PRE.

COMPARE

Compare mode, its interrupt, and the special event trigger (resetting Timer1 by CCP1) are supported in this version of the MPLAB-SIM simulator.

PWM

PWM output (resolution greater than 1 TCY only) are not supported in this version of the MPLAB-SIM simulator.

D.7.6 SSP

The Synchronous Serial Port is supported in SPI mode only. the shift register (SSPSR) can be added to the view screen, observed and modified. The MPLAB-SIM simulator currently does not support the I^2C™ mode.

D.7.7 A/D Converter

All the registers, timing function and interrupt generation are implemented. The simulator, however, does not load any meaningful value into A/D result register (ADRES) at the end of a conversion.

D.7.8 EEPROM Data Memory

The EEPROM data memory (for PIC16F8X devices) is fully simulated. The registers and the read/write cycles are fully implemented. The write cycle time is approximated to 10 ms (to nearest instruction cycle multiple).

The simulator simulates the functions of WRERR and WREN control bits in the EECON1 register.

Appendix E. 16-Bit Core Device Simulator Issues

E.1 Introduction

This appendix discusses I/O pins, interrupts, registers, peripherals, modes, and conditions for using 16-Bit Core devices.

E.2 16-Bit Core Devices

At the time this document was generated, the following list of devices were available in the 16-Bit Core family:

- PIC17C42/43/44
- PIC17C752/756
- PIC17C762/766

The above list implies all device variants, i.e., ROM versions (PIC17CRXX), and device revisions (PIC17CXXA).

E.3 I/O Pins

The 16-Bit Core devices have I/O pins multiplexed with other peripherals (and therefore referred by more than one name). When modifying pins either manually or via the stimulus file, use the following pin names only. These are the only ones that the MPLAB-SIM simulator recognizes as valid I\O pins:

- $\overline{\text{MCLR}}$
- RA0-RA5
- RB0-RB7
- RC0-RC7
- RD0-RD7
- RE0-RE2

These pin names can be used in the Modify window (*Window > Modify*) and in stimulus files.

E.4 Interrupts

MPLAB-SIM supports all interrupts on 16-Bit Core devices:

- External interrupt on INT pin
- TMR0 overflow interrupt
- External interrupt on RA0 pin
- Port B input change interrupt
- Timer/Counter1 interrupt
- Timer/Counter2 interrupt
- Timer/Counter3 interrupt
- Capture1 interrupt
- Capture2 Interrupt

E.5 CPU Model

E.5.1 Reset Conditions

All reset conditions are supported by the MPLAB-SIM simulator.

An $\overline{\text{MCLR}}$ reset during normal operation or during SLEEP can easily be simulated by driving the $\overline{\text{MCLR}}$ pin low (and then high) via the stimulus file, by clicking on the Reset button on the toolbar or selecting *Debug>Run>Reset*.

A WDT time-out reset is simulated when the WDT is enabled (see *Options>ProcessorSetup* dialog), the proper prescaler is set, and the WDT actually overflows. WDT time-out period is approximated at 12 ms (to closest instruction cycle multiple) but can be changed by using the dialog.

The Time out ($\overline{\text{TO}}$) and Power-Down ($\overline{\text{PD}}$) bits in the ALUSTA register reflect appropriate reset condition. This feature is useful for simulating various power-up and time-out forks in the user code.

E.5.2 Sleep

The MPLAB-SIM simulator simulates the SLEEP instruction and will appear "asleep" until a wake-up from sleep condition occurs. For example, if the Watchdog timer has been enabled, it will wake the processor up from sleep when it times out. Another example of a wake-up-from-sleep condition, would be an input change on Port B. If the interrupt is enabled and the GLINTD bit is set, the processor will wake-up and will resume executing from the instruction following the SLEEP command. If the GLINTD = 0, the normal interrupt response will take place.

E.5.3 Watch Dog Timer

The Watchdog Timer is fully simulated in the MPLAB-SIM simulator. Because it is fuse-selectable and fuse-configurable on the device, it must be enabled and configured by the *Options>Processor Setup* dialog in the MPLAB-SIM simulator. The basic period of the WDT (with prescaler = 1) is approximated at 12ms (to closest instruction cycle multiple).

E.6 Special Registers

To aid in debugging this device, certain items that are normally not observable have been declared as “special” registers. Prescalers cannot be declared in user code as “registers”, so the following special symbols are available in the Special Function Registers window:

- T0PRE (Prescaler for Timer 0)
- WDTPRE (Prescaler for WDT)

E.7 Peripherals

E.7.1 Peripherals Supported

Along with providing core support, the following peripheral modules (in addition to general-purpose I/O) are supported:

- Timer 0 in both internal and external clock modes
- Timer1 and Timer2 (and their respective period registers)
- Timer3
- Two Capture Modules
- Two PWM Modules

E.7.2 TIMER0

Timer0 (and the interrupt it can generate on overflow) is fully supported by the MPLAB-SIM simulator, and will increment by the internal or external clock. Delay from external clock edge to timer increment has also been simulated, as well as the interrupt latency period. Clock input must have a minimum high time of 1Tcy and a minimum low time of 1Tcy due to the stimulus file requirements. The prescaler for Timer0 is made accessible as T0PRE. It can be watched and modified.

E.7.3 TIMER1 and TIMER2

Timer1 and Timer2 in its various modes is fully supported by the MPLAB-SIM simulator. Delays from clock edge to increment (when configured to increment from rising or falling edge of external clock) is simulated as well as the interrupt latency periods. Clock input must have a minimum high time of 1Tcy and a minimum low time of 1Tcy due to the stimulus file requirements.

E.7.4 TIMER3 and Capture

The MPLAB-SIM simulator fully supports Timer3 and the Capture module in all of its modes. Delays from clock edge to increment (when configured in external mode), delay for capture and interrupt latency periods are fully supported. Clock input must have a minimum high time of 1Tcy and a minimum low time of 1Tcy due to the stimulus file requirements.

E.7.5 PWM

Both PWM outputs are supported (resolution greater than 1Tcy only) are supported in this version of the MPLAB-SIM simulator.

E.8 Memory Modes

The following memory modes are supported by the MPLAB-SIM simulator:

- Microcontroller Mode
- Extended Microcontroller Mode
- Microprocessor Mode

The default is Microcontroller mode. If you would like to use any of the other modes, you must use the *Options > Processor Setup > Hardware* dialog.

Glossary

Introduction

To provide a common frame of reference, this Glossary defines the terms that follow.

Highlights

This glossary contains definitions for the following systems:

- MPLAB IDE

Terms

Application

A set of software and hardware developed by the user, usually designed to be a product controlled by a PICmicro microcontroller.

Assembler Source Code

A text file that is processed by an assembler to produce a one-to-one correspondence between assembler instructions and PICmicro machine code.

Assemble

To translate a user's "ASM" source text code into machine code.

Asynchronous Stimulus

Data generated to simulate external inputs to the simulator.

Break point – Software

An address where execution of the firmware will halt.

Break point – Hardware

An event whose execution will cause a halt.

Build

A function that recompiles all the source files for an application.

C Source Code

A program written in the high level language called "C," and which will be converted into PICmicro machine code.

Compile

To translate a user's "C" source text code into machine code.

Configuration Bits

Unique bits programmed to set modes of operation. A configuration bit may or may not be preprogrammed. For MPLAB-ICE, these bits are set in the *Options>Processor Setup* dialog.

Data RAM

General purpose file registers from RAM on the PICmicro device being emulated. The File Register window displays data RAM.

Download

Download is the process of sending data from the host PC to the emulator or to the target board.

EEPROM

Electrically Erasable Programmable Read Only Memory.

Emulation

The process of executing software loaded into memory on the emulator probe as if the firmware resided on the microcontroller device under development.

Emulation Memory

Program memory contained within the emulator.

Emulator

Hardware that performs emulation.

Emulator System

The Microchip Emulator System includes the MPLAB-ICE Pod, processor module, device adapter, cables and the MPLAB Software.

Event

A description of a bus cycle which may include address, data, pass count, external input, cycle type (fetch, R/W) and time stamp.

Export

Send data out of MPLAB in a standardized format.

Extended Microcontroller Mode (PIC17CXX Devices Only)

In extended microcontroller mode, on-chip program memory as well as external memory is available. Execution automatically switches to external if the program memory address is greater than the internal memory space of the PIC17CXX device. Inaccessible memory in extended microcontroller mode includes configuration bits, test memory, and boot memory.

External RAM (PIC17CXX Devices Only)

Off-chip Read/Write memory.

File Registers

On-chip general purpose and special function registers.

FNOP

Forced No Operation. A forced NOP cycle is the second cycle of a two-cycle instruction. Since the PICmicro architecture is pipelined, it prefetches the next instruction in the physical address space while it is executing the current instruction. However, if the current instruction changes the program counter, this prefetched instruction is explicitly ignored, causing a forced NOP cycle.

Halt

The command that stops the emulator. Executing Halt is the same as stopping at a break point. The program counter stops, and the user can inspect and change register values, and single step through code.

Hex Code

A file of executable instructions assembled or compiled from source code into standard hex format code. Hex code can be directly converted to object code.

High Level Language

A language for writing programs that is of a higher level of abstraction from the processor than assembler code. High level languages (such as C) employ a compiler to translate statements into machine instructions that the target processor can execute.

ICE

In-Circuit Emulator. MPLAB-ICE is Microchip's in-circuit emulator that works with MPLAB.

IDE

Integrated Development Environment. An application that has multiple functions for firmware development. The MPLAB IDE integrates a compiler, an assembler, a project manager, an editor, a debugger, a simulator, and an assortment of other tools within one Windows application. A user developing an application should be able to write code, compile, debug and test an application without leaving the MPLAB desktop.

Import

Bring data into the MPLAB Integrated Development Environment (IDE) from an outside source

Logic Probes

Up to fourteen logic probes connected to the emulator. The logic probes provide external trace inputs, trigger output signal, +5V and a common ground.

Make Project

A command that rebuilds an application, re-compiling only those source files that have changed since the last complete compilation.

Microcontroller Mode

One of the possible program memory configurations of the PIC17CXX family of microcontrollers. In microcontroller mode, only internal execution is allowed. Thus, only the on-chip program memory is available in microcontroller mode. Accessible memory includes: program memory, configuration bits, test memory, and boot memory (FE00h to FFFFh).

Microprocessor Mode

One of the possible program memory configurations of the PIC17CXX family of microcontrollers. In microprocessor mode, the on-chip program memory is not used. The entire 64K program memory is mapped externally. Inaccessible memory in microprocessor mode includes configuration bits, test memory, and boot memory.

MPLAB-ICE

Microchip's in-circuit emulator that works with MPLAB.

MPLAB-SIM

Microchip's simulator that works with MPLAB.

MPLAB Software

The name of the main executable program that supports the IDE with an Editor, Project Manager, and Emulator/Simulator Debugger. The MPLAB Software resides on the PC host. The executable file name is MPLAB.EXE. MPLAB.EXE calls many other files.

MRU

Most Recently Used. Refers to files and windows available to be selected from MPLAB main pull down menus.

Non Real-Time

Refers to the processor executing single step instructions, or MPLAB being run in simulator mode.

NOP

No operation.

Object code

The machine code that is produced from the source code after it is processed by an assembler or compiler. This code will be the memory-resident code that will run on the PICmicro in the user's application. Relocatable code is code produced by MPASM or MPLAB-C17 that can be run through MPLINK.

Off-Chip Memory

Off-chip memory refers to the memory selection option for the PIC17CXX device where memory may reside on the target board, or where all program memory may be supplied by the Emulator. *Options > Processor Setup > Hardware* provides the Off-Chip Memory selection dialog box.

Pass Counter

A counter that decrements each time an event (such as the execution of an instruction at a particular address) occurs. When the pass count value reaches zero, the event is satisfied. You can assign the Pass Counter to any sequential event in the complex trigger dialog.

PC

Any IBM® or compatible Personal Computer. MPLAB needs a 486X or better machine.

PC Host

The computer running Windows 3.1x or Windows 95/98.

PICmicro

PICmicro refers to the PIC12CXX, PIC14000, PIC16C5X, PIC16CXX, and PIC17CXX Microchip microcontroller families.

PICMASTER

The hardware unit that provides tools for emulating and debugging firmware applications. This unit contains emulation memory, break point logic, counters, timers, and a trace analyzer among some of its tools. MPLAB-ICE is the newest emulator from Microchip.

Pod

The external emulator box that contains emulation memory, trace memory, event and cycle timers, and trace/break point logic. Occasionally used as an abbreviated name for the MPLAB-ICE Emulator.

Power-on-Reset Emulation

A software randomization process that writes random values in data RAM areas to simulate uninitialized values in RAM upon initial power application.

Program Counter

A register that specifies the current execution address.

Program Memory

Memory in the emulator or simulator containing the downloaded target application firmware.

Project

A set of source files and instructions to build the object code for an application.

Prototype System

A term referring to a user's target application, or target board.

PWM Signals

Pulse Width Modulation Signals. Certain PICmicro devices have a PWM peripheral.

Qualifier

An address or an address range used by the Pass Counter.

Radix

The number base, hex, or decimal, used in selecting an address and for entering data in the *Window > Modify* command.

Real-time

When released from the halt state, the processor runs in real-time mode and behaves exactly as the normal chip would behave. In real-time mode, the real-time trace buffer is enabled and constantly captures all selected cycles, and all break logic is enabled. In the emulator the processor executes in real-time until a valid break point causes a halt, or until the user halts the emulator.

In the simulator real-time simply means execution of the microcontroller instructions as fast as they can be simulated by the host CPU.

Run

The command that releases the emulator from halt, allowing it to run the application code and change or respond to I/O in real time.

SFR

Special Function Registers of a PICmicro.

Simulator

A software program that models the operation of the PICmicro microprocessor.

Simulator Stimulus

Data generated to exercise the response of simulation to external signals. Often the data is put into the form of a list of actions in a text file.

Single Step

This command steps though code, one instruction at a time. After each instruction, MPLAB updates register windows, watch variables, and status displays so you can analyze and debug instruction execution.

You can also single step C compiler source code, but instead of executing single instructions, MPLAB will execute all assembly level instructions generated by the line of the high level C statement.

Source

Source code, usually a text file of assembly instructions or C code.

Special Function Registers

Registers that control I/O processor functions, I/O status, timers, or other modes or peripherals.

Stack

"Push-Down" list of calling routines. Each time a PICmicro microcontroller executes a call or responds to an interrupt, the software pushes the return address to the stack. A return command pops the address from the stack.

Static RAM or SRAM

Static Random Access Memory. Program memory you can Read/Write on the target board that does not need refreshing frequently.

Step-Into

This command is the same as Single Step. Step-Into (as opposed to Step-Over) follows a CALL instruction into a subroutine.

Step-Over

Step-Over allows you to debug code without stepping into subroutines. When stepping over a CALL instruction, the next break point will be set at the instruction after the CALL. If for some reason, the subroutine gets into an endless loop or does not return properly, the next break point will never be reached.

The Step-Over command is similar to Single Step except for its handling of CALL instructions.

Stopwatch

A counter for measuring execution cycles.

Symbol

An label usually produced by an assembler or compiler that refers to machine locations by function names, variable locations, constant declarations, source line-number, or other reference to user source code.

System Button

The System Button is located in the upper left corner of Windows and some dialogs. This button usually has “Minimize,” “Maximize,” and “Close.” In some MPLAB windows, additional modes or functions can be found under the System Button.

Target

Refers to user hardware.

Target Application

Firmware residing on the target board.

Target Board

The circuitry and programmable device that makes up the target application.

Target Processor

The microcontroller device on the target application board that is being emulated.

Template

Lines of text that you build for inserting into your files at a later time. The MPLAB Editor stores templates in template files.

Tool Bar

A row or column of icons that you can click on to execute MPLAB functions.

Trace

An emulator or simulator function that logs program execution. The emulator logs program execution into its trace buffer which is uploaded to MPLAB's trace window.

Trace Memory

Trace memory contained within the Emulator. Trace Memory is sometimes called the Trace Buffer.

Trigger Output

Trigger output refers to an emulator output signal that can be generated at any address or address range, and is independent of the trace and break point settings. Any number of trigger output points can be set.

Upload

The Upload function transfers data from the emulator to the host PC or from the target board to the emulator.

Watch Dog Timer

A timer on a PICmicro microcontroller that resets the processor after a selectable length of time.

Watch Variable

A variable that you may monitor during a debugging session. Watch windows contain a list of watch variables that are updated at each break point.

Index

Index

Index

NOTES:

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602-786-7200 Fax: 602-786-7277
Technical Support: 602 786-7627
Web: http://www.microchip.com

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

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

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

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 972-991-7177 Fax: 972-991-8588

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

Detroit
Microchip Technology Inc.
42705 Grand River, Suite 201
Novi, MI 48375-1727
Tel: 248-374-1888 Fax: 248-374-2874

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 714-263-1888 Fax: 714-263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 516-273-5305 Fax: 516-273-5335

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

AMERICAS (continued)

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

ASIA/PACIFIC

Hong Kong
Microchip Asia Pacific
RM 3801B, Tower Two
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

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

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

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

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

ASIA/PACIFIC (continued)

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

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

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-1189-21-5858 Fax: 44-1189-21-5835

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

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

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-39-6899939 Fax: 39-39-6899883

9/8/98

Microchip received ISO 9001 Quality System certification for its worldwide headquarters, design, and wafer fabrication facilities in January, 1997. Our field-programmable PICmicro® 8-bit MCUs, Serial EEPROMs, related specialty memory products and development systems conform to the stringent quality standards of the International Standard Organization (ISO).

Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602.786.7200 Fax: 602.899.9210

 DS51025