



MPLAB[®] IDE v6.xx QUICK START

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable".
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

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

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contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates.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

Trademarks

The Microchip name and logo, the Microchip logo, KEELoQ, MPLAB, PIC, PICmicro, PICSTART and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


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

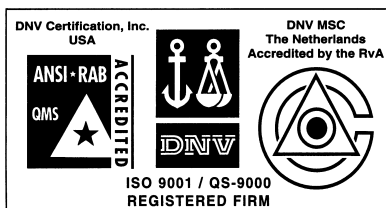
dsPIC, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and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

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

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

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

 Printed on recycled paper.



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELoQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

Quick Start

INTRODUCTION

This document will show how to install the new MPLAB® IDE version 6.1x and set up this software for debugging and programming an application into a PICmicro® MCU. An overview of debugging capabilities will be discussed using an example application as a guide. In addition, some related areas of the MPLAB IDE system are presented to help you get your application finished *fast*.

HIGHLIGHTS

The first section of this document guides you through the installation of MPLAB IDE on your PC. The second section is a simple step-by-step tutorial that creates a project and gets you familiar with the debug capabilities of MPLAB. Once you understand how to use MPLAB to build and test your application, the last section will be of interest as you employ other tools and want to customize MPLAB for your own debugging environment.

- Getting Started with MPLAB IDE
- Debugging a Simple Project
 - Creating a Project
 - Running the Simulator
 - Debugging Your Application
- What Do I Do Next?
 - Programming a Device
 - Advanced Simulator Options
 - Using MPLAB ICD 2
 - MPLAB IDE on-line help
 - Workspace and Project Debug Settings

GETTING STARTED WITH MPLAB IDE

MPLAB IDE is a Windows® OS based Integrated Development Environment for the Microchip Technology Incorporated PICmicro microcontroller (MCU) families and the dsPIC™ Digital Signal Controllers. The MPLAB IDE provides the ability to:

- Create source code using the built-in editor.
- Assemble, compile and link source code using various language tools. An assembler, linker and librarian come with MPLAB IDE. Supported C compilers are available from Microchip. Third party compilers may be supported also. Check the release notes or readme files for details.
- Debug the executable logic by watching program flow with the built-in simulator, or in real time with the MPLAB ICE 2000 emulator or MPLAB ICD 2 in-circuit debugger. Third party emulators may be supported also. Check the release notes or readme files for details.
- Make timing measurements with the simulator or emulator.
- View variables in Watch windows.

MPLAB® IDE v6.xx Quick Start

- Program firmware into devices with PICSTART® Plus or PRO MATE® II device programmers. Third party programmers may be supported also. Check the release notes or readme files for details.
- Find quick answers to questions from the MPLAB IDE on-line help.

System Requirements

The following minimum configuration is required to run MPLAB IDE:

- PC-compatible Pentium® class system
- Microsoft Windows 98, Windows 2000 SP2, Windows NT® SP6, Windows ME, Windows XP
- 16 MB memory (32 MB recommended)
- 45 MB of hard disk space
- Internet Explorer 5.0 or greater for installation and on-line help

Install/Uninstall MPLAB IDE

To install MPLAB IDE on your system:

- For some Windows OS's, you will need administrative access in order to install software on your PC.
- If you are installing from a CD-ROM, place the CD-ROM into the drive. Follow the on-screen menu to install MPLAB IDE. If no on-screen menu appears, use Explorer to find and execute the CD-ROM menu by double-clicking on the executable file `menu.exe`.
- If you have downloaded MPLAB IDE from the Microchip web site, double-click on the downloaded executable file to begin installation.

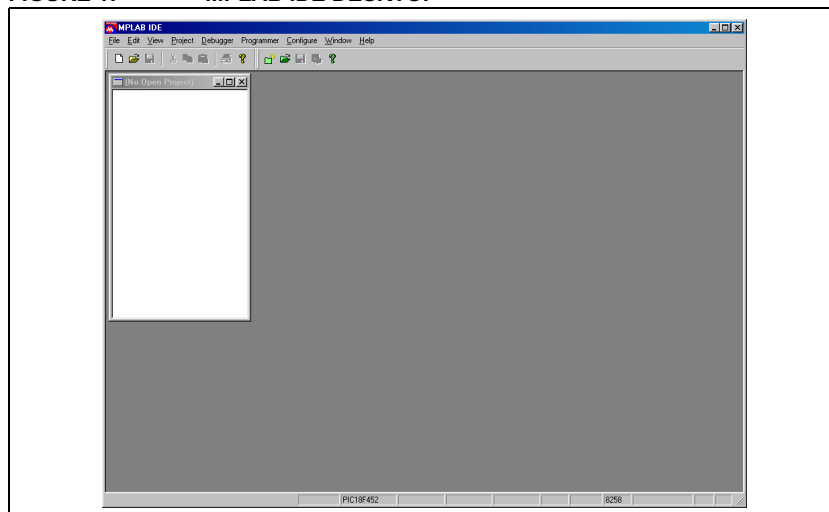
To uninstall MPLAB IDE:

- Select UNWISE32 from the *Start>Programs>Microchip MPLAB IDE* menu, or
- Execute the file `unwise32.exe` in the MPLAB IDE installation directory.

Running MPLAB IDE

To start the IDE, select *Start>Programs>Microchip MPLAB IDE>MPLAB IDE*. A splash screen will display first, followed by the MPLAB IDE desktop.

FIGURE 1: MPLAB IDE DESKTOP



DEBUGGING A SIMPLE PROJECT

Create a Project

The best way to develop your application is to use projects. In this example, you will put all the files needed to build your application in a project. Then, you will set the project up to use a particular set of language tools.

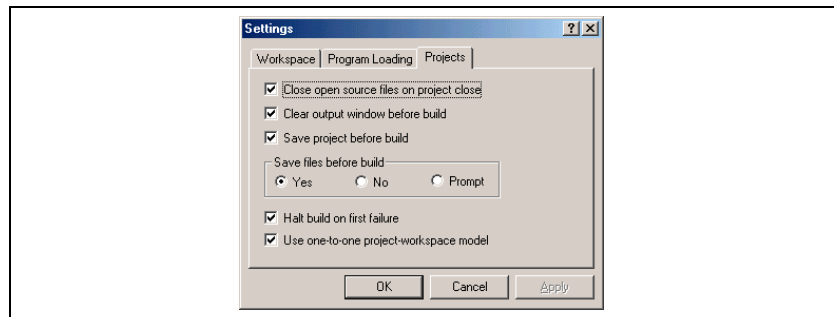
The output of a project is a HEX file, that is loaded into memory for debugging. There are other files generated by a project that keep track of the symbols defined in your program and show you where your code has been put in memory.

CONFIGURE THE PROJECT

Before getting started, it is important that you have the workspace and project set up correctly. In this example, you will use a one-to-one project-workspace model so you will not have to be concerned with workspace management. For future reference, you might want to refer to the last section in this document concerning workspaces and projects.

Go to the *Configure>Settings* menu and choose the “Projects” tab. Make sure that all items are checked. Then click **OK**.

FIGURE 2: SETTINGS: PROJECTS TAB

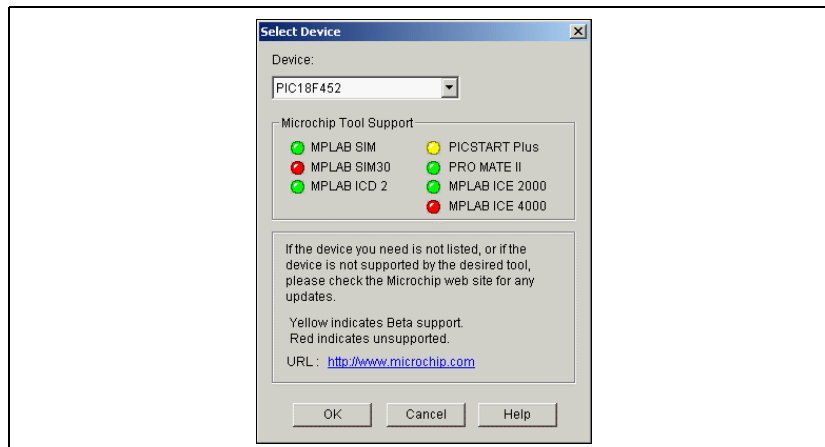


SELECTING THE DEVICE

You must now tell MPLAB IDE the device that you want to use. For this tutorial, we will use the PIC18F452.

1. Select *Configure>Select Device*.

FIGURE 3: SELECT DEVICE DIALOG



2. Select **PIC18F452** from the Device pull-down list. A green light indicates that the tool supports the selected device. A red light indicates that the tool does not support the selected device. A yellow light indicates that the tool provides early adopter or Beta support for the selected device.

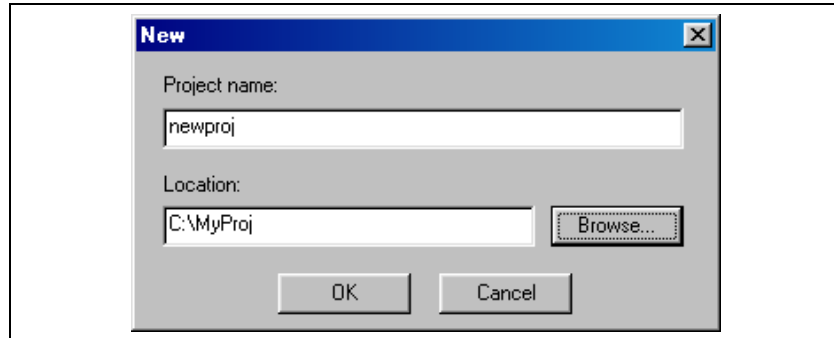
Click **OK**.

CREATE A NEW PROJECT

Follow the steps below to create a new project.

1. Using Windows Explorer, create a directory in which you will create your project (e.g., C:\MyProj).
2. In MPLAB IDE, select *Project>New*. The New dialog will open (Figure 4).
3. Enter a name for your new project (e.g., newproj).
4. Use the **Browse** button to select the path you created in step 1, or type in the path.
5. When the Project name and Location are correct, click **OK**.

FIGURE 4: NEW DIALOG

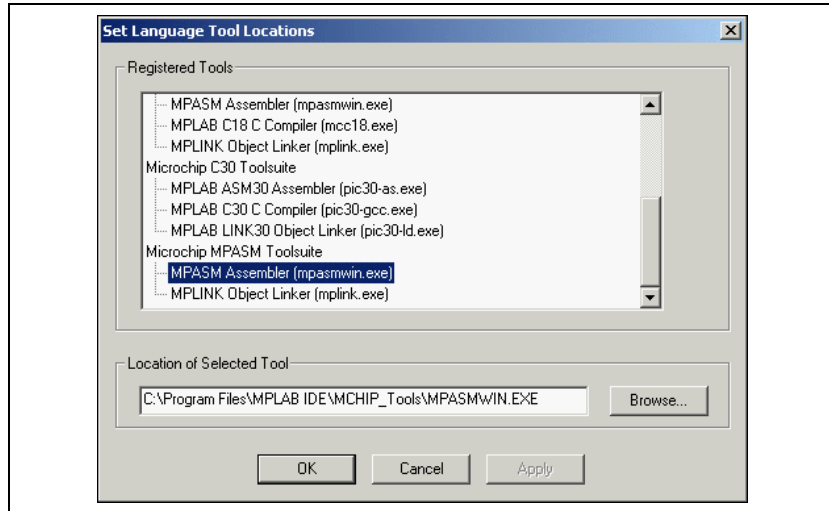


Note: The MPLAB IDE supports long file names. However, not all language tools do. Before using long file names as paths or source file names, or using spaces, be sure to check the constraints of your language tool.

SET LANGUAGE TOOL LOCATION

Select *Project > Set Language Tool Locations* to confirm the location of the Microchip Toolsuite. Click on "MPASM Assembler (mpasmwin.exe)". The full path to the MPASM Assembler executable should appear in the "Location of Selected Tool" text box as shown. If it is incorrect or empty, click **Browse** to locate mpasmwin.exe.

FIGURE 5: SET LANGUAGE TOOL LOCATION DIALOG



For this tutorial, only the MPASM Assembler is required. For projects that require other language tools, be sure to confirm the location of each tool required by the project.

Click **OK** after setting the necessary tools.

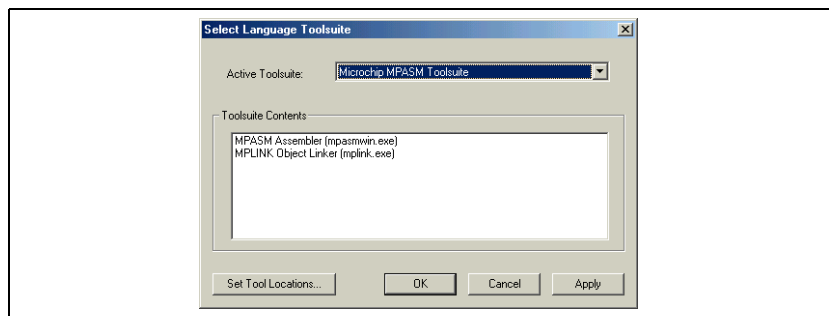
SELECTING THE TOOLSUITE

Before you start editing your code, you will set the language toolsuite that you will use for your project. This allows the MPLAB IDE to tailor its operation more accurately with regard to context-sensitive editing and file extensions.

For this tutorial, select the Microchip Toolsuite:

1. Select *Project > Set Language Toolsuite*.
2. For "Active Toolsuite", select "Microchip MPASM Toolsuite". The PICmicro language tools will appear under "Toolsuite Contents".
3. Click **OK**.

FIGURE 6: SELECT LANGUAGE TOOLSUITE DIALOG



CREATING SOURCE CODE

You can now begin developing code for your application using the MPLAB IDE editor.

Select *File>New*. You should now have a blank edit window open in the workspace. Enter the example code listed (or copy and paste from this document).

```
        title "PIC18F452 counting program"
        list p=18f452,f=inhx32
        #include <p18f452.inc>

COUNT    equ    0x00
DVAR      equ    0x01
DVAR2     equ    0x02

        org    00h            ;reset vector
        goto   Start

Start      org    1Ch

        clrf   WREG           ;clear W register
        movwf  PORTC          ;clear PORTC
        movwf  TRISC          ;config PORTC as outputs

Init      clrf   COUNT        ;clr count

IncCount  incf   COUNT,F       ;increment count
          movf   COUNT,W       ;
          movwf  PORTC         ;display on port c

          call  Delay          ;wait
          goto  IncCount       ;loop

Delay     movlw  0xFF          ;set delay loop
          movwf  DVAR2         ;

D0        movwf  DVAR          ;reset inner loop

D1        decfsz DVAR,F
          goto  D1

          decfsz DVAR2,F
          goto  D0
          return
        end
```

Once you have completed entering the code, select *File>Save* and save the file in the project directory as `cnt452.asm`.

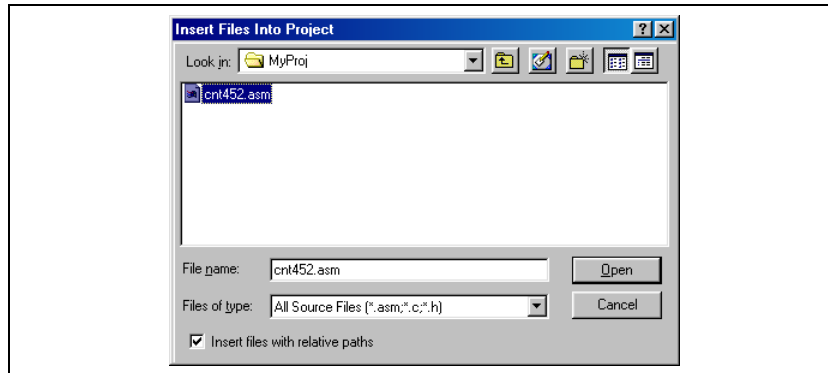
After you save the code, the code is shown with identifying colors for readability. This context sensitive colorations is customizable. For more information about the editor, see *Help>MPLAB Editor Help*.

ADDING THE SOURCE CODE FILE TO THE PROJECT

Now that you have created the source code file, it needs to be added to the project.

1. Select *Project>Add Files to Project*.

FIGURE 7: SELECT INPUT FILE DIALOG

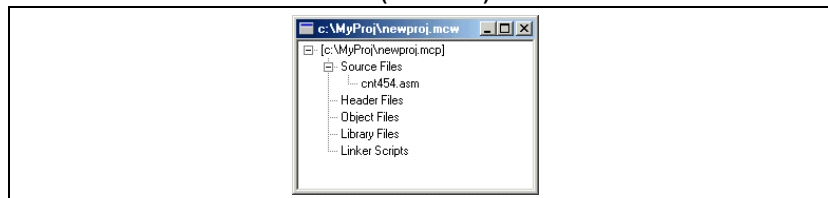


2. Select the file you just created and saved (cnt452.asm).
3. Click **Open**.

Note: Be sure to follow the naming convention of your language tools. For example, assembler source files for the MPASM Assembler should have the extension .asm. Not all language tools support long file names.

The newly inserted file should appear in the project pane under "Source Files" (Figure 8). If it appears under "Unclassifiable", confirm that you have selected the correct language toolsuite by selecting *Project>Set Language Toolsuite*. "Microchip Toolsuite" should appear as the Active Toolsuite. If not, select it and click **OK**. The file name should now appear under "Source Files."

FIGURE 8: PROJECT PANE (WINDOW)



Save the project by selecting *Project>Save*.

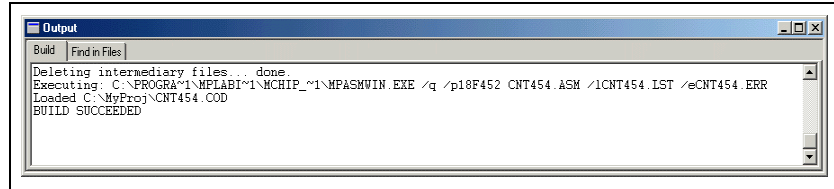
TIP: You can also add files and save projects by using the right mouse button in the project pane. Experiment by clicking the right mouse button while the cursor is over the project path and name. Then click the right mouse button while the cursor is over "Source Files". The menu options are different.

BUILDING THE PROJECT

Now that you have finished entering your source code, it is time to build the project. This will compile the source code using the selected language toolsuite.

Select **Project>Build All** to build your project. Your file should assemble successfully.

FIGURE 9: OUTPUT WINDOW



If your file does not assemble successfully, please check the following items and then build the project again:

- Check the spelling and format of the code you entered in the editor window. If the assembler reported errors in the Output window, double click on the error in the Output window. This will indicate the corresponding line in your source code with a yellow arrow in the gutter of the source code window.
- Check that the correct assembler (MPASM assembler) for PICmicro devices is being used. Select **Project>Set Language Tool Locations**. Click "MPASM Assembler (mpasmwin.exe)" and review its location in the display. If the location is correct, click **Cancel**. If it is not, change it and then click **OK**.

Upon a successful build, the debug file (*.cod or *.cof) generated by the language tool will be loaded. This file allows you to debug using your source code and view your variables symbolically in Watch windows.

PROJECT BENEFITS

For this one-file example, a project does not seem necessary. However, the real power of projects comes when there are many files to be compiled/assembled and linked to form the final executable application. Projects keep track of all of this for you.

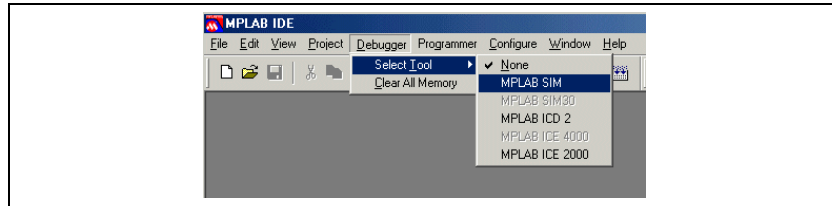
In addition, the linking process using the Microchip toolsuite can generate more comprehensive debugging information than the assembler alone. Only the linker, for example, can generate information regarding local variables.

Running the Simulator

Now that your project is built, you will want to check that it is functioning the way you intended. To do this, you will need to select a debug tool. For this tutorial, we will use the simulator.

The device you have chosen for this tutorial, PIC18F452, is supported by the MPLAB SIM simulator. Select the MPLAB SIM simulator as the debugger by selecting *Debugger>Select Tool>MPLAB SIM*.

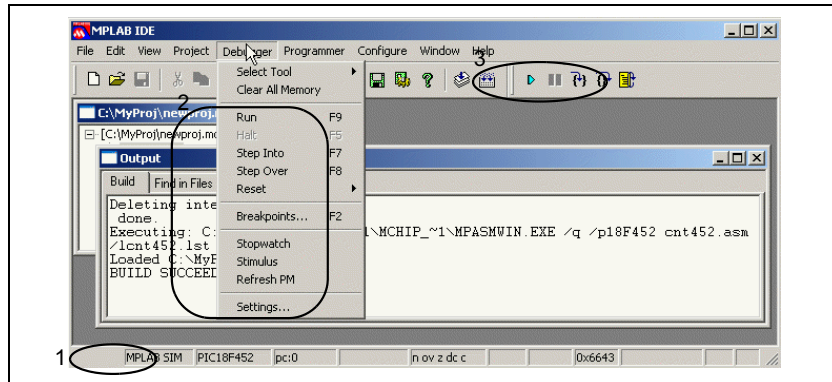
FIGURE 10: SELECT SIMULATOR AS THE DEBUGGER



After you select MPLAB SIM you should see the following changes:

1. The status bar on the bottom of the MPLAB IDE window should change to "MPLAB SIM".
2. Additional menu items should now appear in the Debugger menu.
3. Additional toolbar icons should appear in the Debug Tool Bar:

FIGURE 11: MPLAB IDE DESKTOP WITH MPLAB SIM AS DEBUGGER



TIP: Position the mouse cursor over a toolbar button to see a brief description of the button's function.

The simulator is a software program that runs on your PC to simulate the instructions of the PICmicro MCU. It does not run in "real time," since it is dependent upon the speed of your PC, the complexity of the code, overhead from the operating system and how many other tasks are running. It does, however, keep track of how much time it would take to execute your code if it were operating in real time in your application.

Other debuggers include MPLAB ICE 2000 and MPLAB ICD 2. These are external hardware tools to run code on your application PC board. As opposed to the simulator, these tools allow the PICmicro MCU to run at full speed.

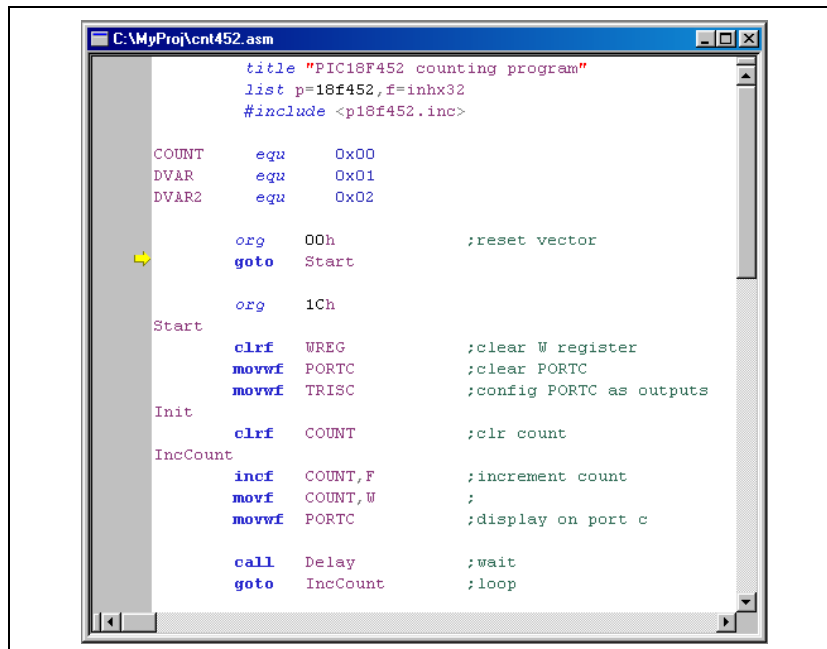
Debug Your Application

You are now ready to run and debug your application.

RUNNING YOUR CODE

First, select *Debugger>Reset*. There should be a yellow arrow in the gutter of your source code window, indicating the first source code line that will be executed.

FIGURE 12: SOURCE CODE WINDOW - AFTER RESET



Select *Debugger>Run* to run your application. You will see "Running..." appear on the status bar.

To halt program execution, select *Debugger>Halt*. The line of code where the application halted will be indicated by the yellow arrow.

You may also single step through the application program by selecting *Debugger>Step Into*. This will execute the currently indicated line of code and move the arrow to the next line of code that will be executed.

TIP: You may click on the appropriate icon on the toolbar or use the hotkey shown next to the menu item instead of selecting the menu item. This is usually the best approach for repeated stepping.

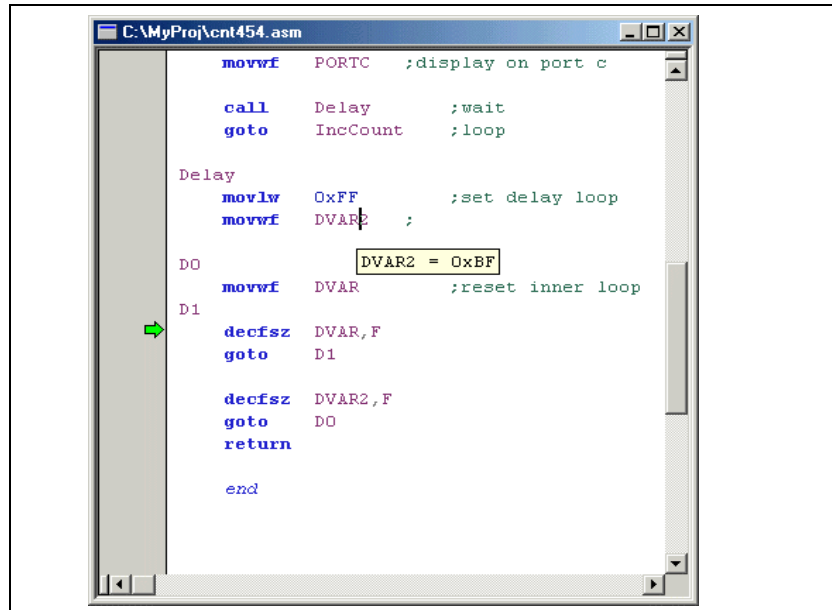
MPLAB® IDE v6.xx Quick Start

VIEWING VARIABLES

You can see the values of variables at any time by putting the mouse cursor over their names anywhere in the source file. A small output window will pop up to show the current value.

Note: The popup variable value feature can display local variables only when the program has been compiled and linked to generate such information. The Microchip toolsuite generates the necessary information. Others may not.

FIGURE 13: MOUSE OVER VARIABLE

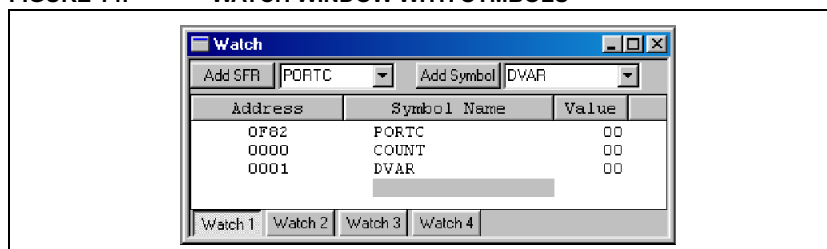


WATCH WINDOWS

Often you want to watch certain key variables all the time. Rather than floating the mouse cursor over the name each time you want to see the value, you can open a watch window. The watch window will remain on the screen and show the current variable values. Watch windows may be found under the View menu.

1. Select **View>Watch** to open a new Watch window.
2. Select `PORTC` from the SFR selection box at the top of the window. Click **Add SFR** to add it to the Watch window list. If you want to quickly advance through the list, start typing `PORTC` after selecting the pull down icon.
3. Select `COUNT` from the symbol selection box at the top of the window. Click **Add Symbol** to add it to the Watch window list.
4. Select `DVAR` from the symbol selection box at the top of the window. Click **Add Symbol** to add it to the Watch window list.

FIGURE 14: WATCH WINDOW WITH SYMBOLS

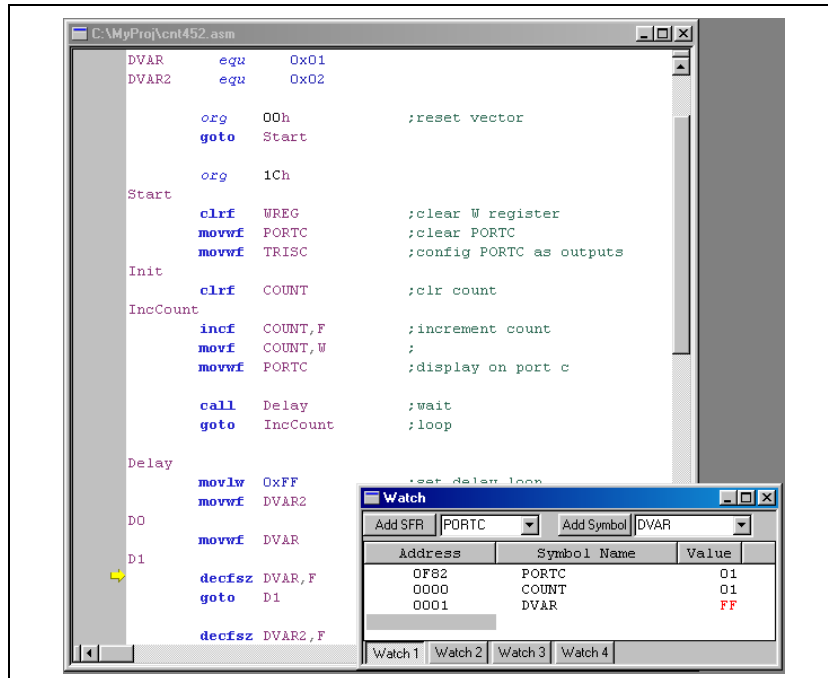


You should now have three symbols in the Watch window. The file register address of the symbols is listed first, followed by the symbol name, and finally the value of the symbol. Now watch the symbol values change as you step through the program.

1. Select **Debugger>Reset** to reset your application.
2. Select **Debugger>Step Into** (or click the equivalent toolbar icon) until you have stepped to the following program line:


```
incf  COUNT,F          ;increment count
```
3. Step one more time to see the value of `COUNT` in the Watch window change from 0 to 1.
4. Step twice more to see the value of `PORTC` in the Watch window change from 0 to 1.
5. Step four more times to see the value of `DVAR` in the Watch window change to `FF`. Note that the values in the Watch window are red if they were changed by the previous debug operation, and are black if they were not changed by the previous debug operation

FIGURE 15: STEPPING THROUGH CODE



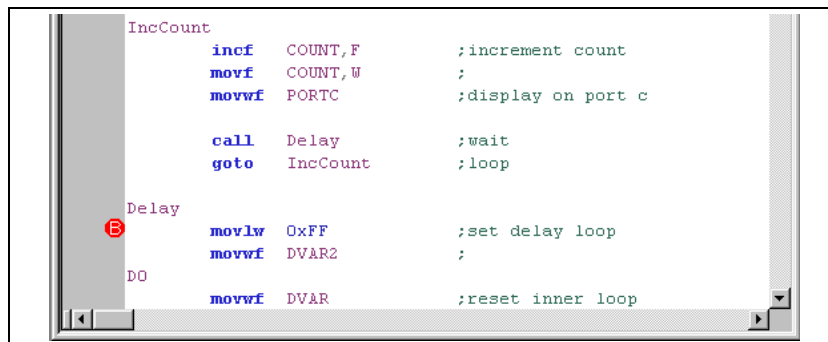
SETTING BREAKPOINTS

Sometimes you want your code to run to a specific location and then halt. This is accomplished by using breakpoints. Setting a simple breakpoint is discussed here. Setting a breakpoint using the Breakpoint dialog is discussed in the on-line help.

1. Select *Debugger>Reset* to reset your application.
2. Find the following line of code and use the right mouse button to click on it:

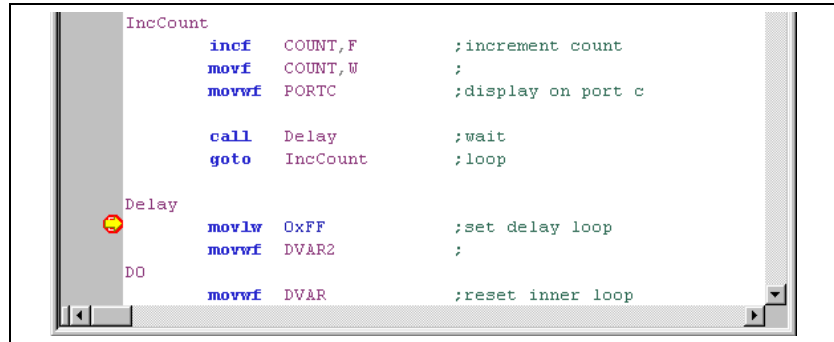
```
movlw 0xFF          ;set delay loop
```
3. From the pop-up menu that appears, select Set Break Point. A stop sign should appear in the gutter next to the line.

FIGURE 16: SOURCE CODE WINDOW - SET BREAKPOINT



4. Select *Debugger>Run* to run your application. It should run briefly and then halt on the line at which you set the breakpoint.

FIGURE 17: SOURCE CODE WINDOW - BREAKPOINT HALT

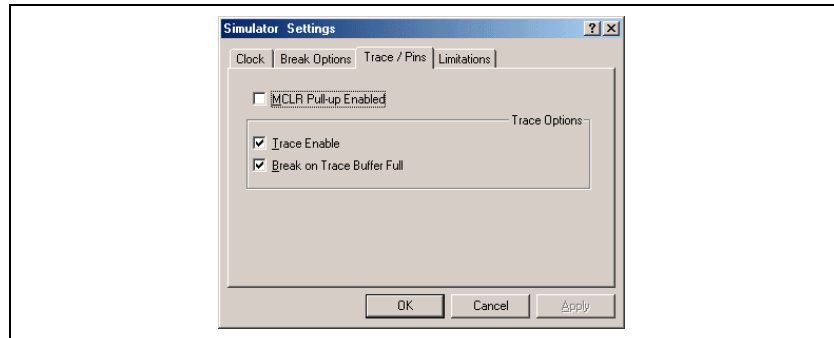


Note: You can place your cursor on any instruction line and double click to halt. This is a “run to here” action, and is convenient when you are halted at a breakpoint and want to run and then immediately break at an instruction to be executed later in the code. If that instruction is never executed, however, the application will continue to run until you select *Debugger>Halt*.

TRACING CODE

The simulator trace can be used to record the execution of your program. Rather than single step through lines of code, you can capture the code in action. Enable the simulator trace by *Debugger>Settings* and choose the “Trace/Pins” tab.

FIGURE 18: SIMULATOR TRACE ENABLE



There are two check boxes to control how the simulator trace collects data. When only the top box is checked, the simulator collects data when the simulator is in Run mode. It collects data until you halt at a breakpoint or manually stop the simulator. It will show the last 8192 cycles collected. This mode is useful if you want to see the record of instructions leading up to a breakpoint.

If the second button is also checked, then the trace memory will collect 8192 cycles of data then stop collecting and halt your application at a breakpoint. This mode is useful for seeing the record of instructions after you press run.

Select *View>Simulator Trace*. The simulator trace shows you more information than just the sequence of executed instructions. The trace display shows a time stamp at every cycle, plus the data that is was read or written into file registers will be captured and displayed.

FIGURE 19: SIMULATOR TRACE DISPLAY

Addr	Op	Label	Instruction	SA	SD	DA	DD	Time
0000	EFOE		GOTO 0x00001c	----	--	----	--	0.000002000
001C	6AE8	Start	CLRF 0xfe8, 0	----	--	0F88	00	0.000003000
001E	6E82		MOVWF 0xf82, 0	----	--	0F82	00	0.000004000
0020	6E94		MOVWF 0xf94, 0	----	--	0F94	FF	0.000005000
0022	6A00	Init	CLRF 0, 0	----	--	0000	00	0.000006000
0024	2A00	IncCount	INCF 0, 0x1, 0	0000	00	0000	00	0.000007000
0026	5000		MOVF 0, 0, 0	0000	01	0F88	01	0.000008000
0028	6E82		MOVWF 0xf82, 0	----	--	0F82	00	0.000009000
002A	EC19		CALL 0x000032, 0	----	--	----	--	0.000010000
0032	0EFF	Delay	MOVLW 0xff	----	--	0F88	FF	0.000012000
0034	6E02		MOVWF 0x2, 0	----	--	0002	00	0.000013000
0036	6E01	D0	MOVWF 0x1, 0	----	--	0001	00	0.000014000
0038	2E01	D1	DECFSZ 0x1, 0x1, 0	0001	FF	0001	FF	0.000015000
003A	EF1C		GOTO 0x000038	----	--	----	--	0.000017000
0038	2E01	D1	DECFSZ 0x1, 0x1, 0	0001	FE	0001	FE	0.000018000

The display is made up of columns. On the left is the Program Counter address (“Addr”) and the machine code value of the instruction (“Op”). The “Label” column shows any symbolic label from your source code. Next, the disassembled instruction is shown. The four columns to the right of the “Instruction” column show data values being read and written to file registers:

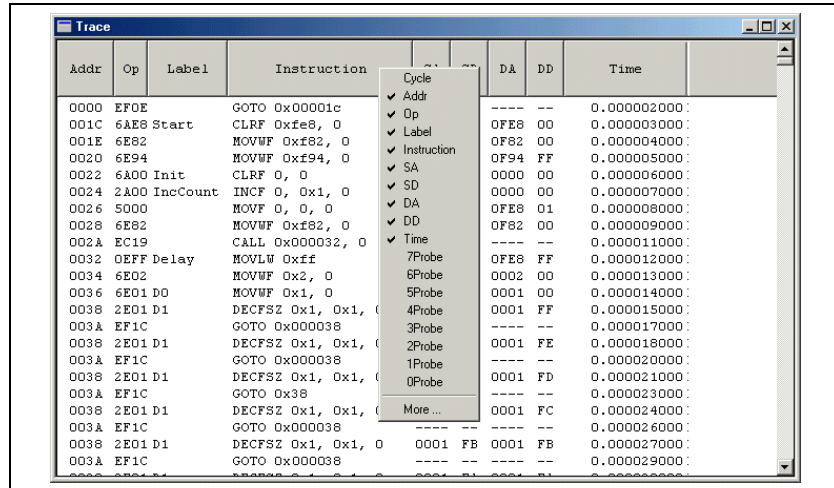
- SA - is Source Address, the register address of *read* operations
- SD - is Source Data, the data read from the register
- DA - is Destination Address, the register address of *write* operations
- DD - is Destination Data, the data written to the register

If there are dashes in the row for these values, it means that the operation did not access any file registers for this instruction.

On the far right is the time stamp. This can be used to measure the execution time of routines. The time is calculated based upon the clock frequency entered in the *Debugger>Settings*, “Clock” tab.

If you put the cursor over the top row of the trace display where the column headings are listed and press the right mouse button, a configuration dialog will pop up.

FIGURE 20: SIMULATOR TRACE CONFIGURE



The checked items will appear in the trace window. You can uncheck columns to reduce clutter if you are not interested in the data in those columns. The bottom of the pop up dialog has some entries labeled “7Probe,” “6Probe,” etc. These are for the MPLAB ICE 2000 emulator trace and are not relevant to the simulator.

WHAT DO I DO NEXT?

For on-line tutorials, look under the Help menu of MPLAB IDE. Check out the various documents that come with other MPLAB IDE components, such as the *MPASM™ User's Guide with MPLINK™ and MPLIB™*. Much of the documentation for MPLAB IDE and its components is on line, part of help system of MPLAB IDE. These following sections will point out some of the features that were not covered in the project tutorial, but which will be some of the areas you'll want to learn more about.

Programming a Device

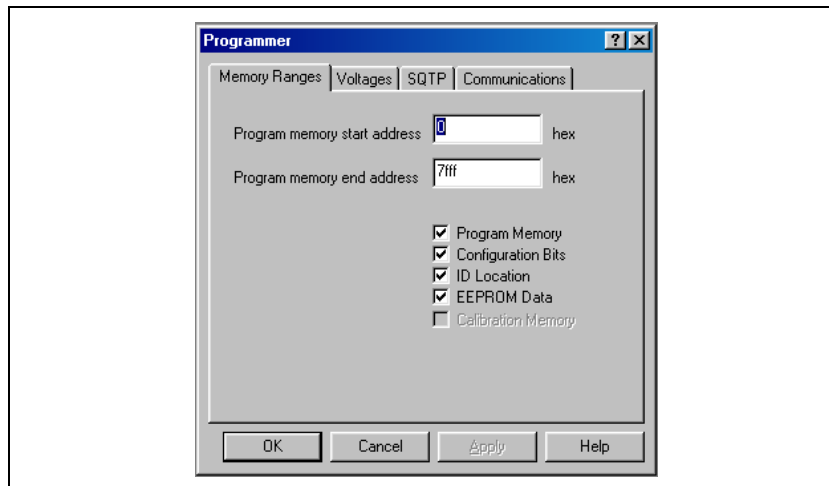
Once the application code is running the way you want it, you will be ready to program an actual device. If you have a PIC18F452 device and one of the following programmers, you can program the example code into the device:

- MPLAB ICD 2
- PICSTART Plus Development Programmer (requires PICSTART Plus firmware v3.00 or higher, and software v3.10 or higher, which will be available with MPLAB IDE v6.10)
- PRO MATE II Device Programmer

To select and set up the programmer, do the following:

1. Select *Programmer>Select Programmer*, and select the desired programmer. The Programmer menu items will change appropriately for the selected tool, and toolbar items will be added.
2. Establish communications with the programmer. For the PICSTART Plus or PRO MATE II, select *Programmer>Enable Programmer*. For the MPLAB ICD 2, select *Programmer>Connect*.
3. Use the *Programmer>Settings* dialog to select the proper communications method for your programmer. For this example, use the default memory ranges.

FIGURE 21: PRO MATE II SETTINGS DIALOG



- Specify the configuration bits. For this example, the default configuration bit settings are fine. For your own application, you can specify the configuration bits setting in your source code (recommended), or set them manually by using the Configuration Bits window, invoked by selecting *Configure>Configuration Bits*.

TIP: If you set configuration bits in your source code, they will affect the debugger operation. For example, if your source code specifies the oscillator configuration, then the debugger will use that oscillator configuration.

- Click *Programmer>Program* to program the information currently loaded in the MPLAB IDE into the device. The operation progress is indicated in the status bar. Results will be displayed in the Output window under the PRO MATE tab. E.g.,

```
PRO MATE Error Log File
Programming
31-May-2002. 13:06:19
Device Type: PIC18F452
```

Programming/Verification Successful!

When the device is programmed, it is also automatically verified. To perform an extra verification that the device programmed correctly, click *Programmer>Verify*.

Advanced Simulator Options

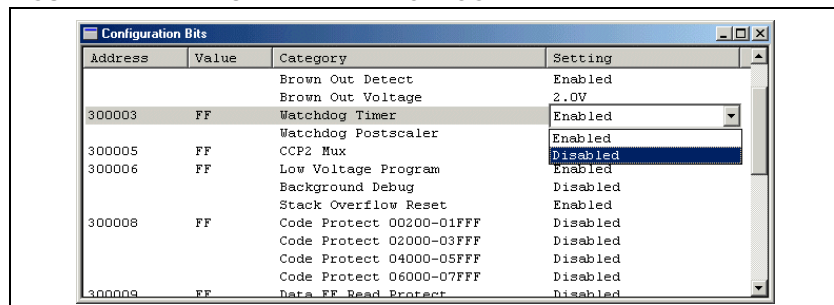
There are other characteristics of the simulator that can be configured from MPLAB IDE dialogs.

CONFIGURATION BITS SETTINGS

Normally, the default condition of the configuration bits has the Watchdog Timer (WDT) enabled. This will cause the simulator to reset when the internal WDT times out.

- Select *Configure>Configuration Bits* to bring up this dialog.
- If you click on the "Disabled" setting for the Watch Dog Timer, a selection box will appear. Scroll down to select "Disabled" to prevent the WDT from causing your program to reset.

FIGURE 22: DISABLE THE WATCHDOG TIMER

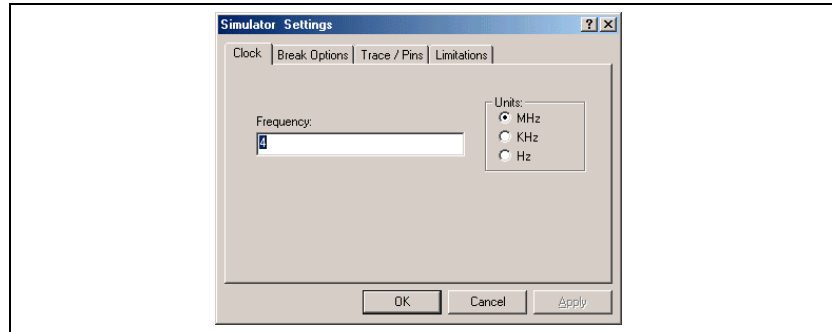


DEBUGGER SETTINGS FOR THE SIMULATOR

Select *Debugger>Settings* to bring up the dialog to configure the debugger, which, in this case, is the MPLAB SIM simulator.

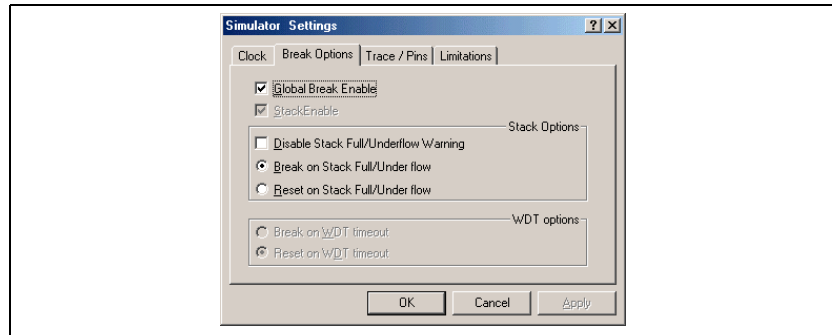
The “Clock” tab sets the frequency of the simulator’s clock. This is important because the time stamp in the simulator trace as well as times in the Stopwatch dialog are calculated based upon this setting. This allows you to make accurate time measurements based upon the actual speed of your intended application.

FIGURE 23: DEBUGGER SETTINGS: CLOCK



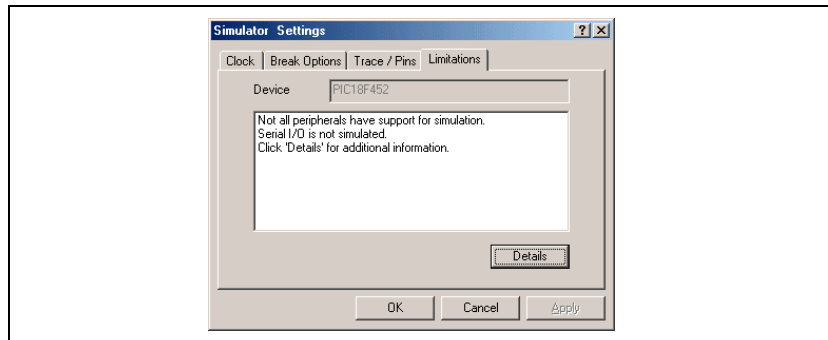
The “Break Options” tab contains additional breakpoint features. If “Global Break Enable” is unchecked, then breakpoints will not operate. This is useful if you have many breakpoints inserted and wish to disable them all without clearing them. You can go back to this dialog and re-enable when you wish to activate breakpoints again.

FIGURE 24: DEBUGGER SETTINGS: BREAK OPTIONS



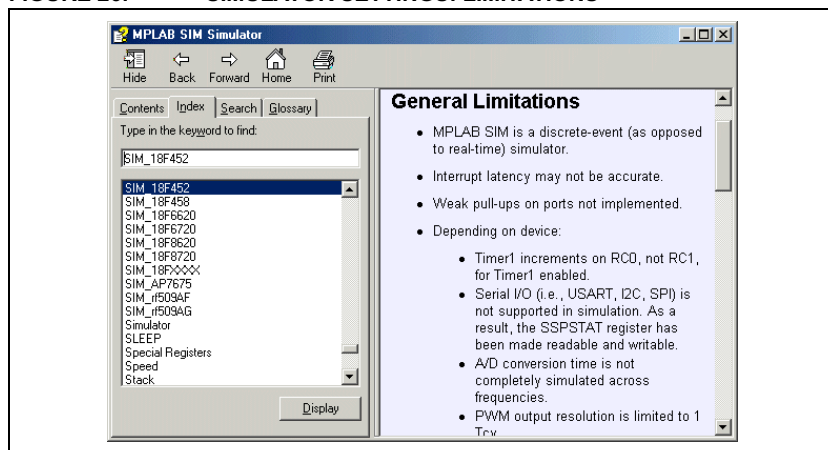
The “Limitations” tab details any limitations the simulator has compared to the actual device being simulated. General limitations are shown in display on the tab.

FIGURE 25: SIMULATOR SETTINGS: LIMITATIONS



Press the **Details** button to show specific limitations of the device being simulated. From this display you can also access help on general limitations related to the simulator.

FIGURE 26: SIMULATOR SETTINGS: LIMITATIONS



Using MPLAB ICD 2

This section is intended to be a quick overview and a look at some of the dialogs that have changed from v6.00 to v6.10. For more on using MPLAB ICD 2, refer to the *MPLAB ICD 2 Quick Start Guide* (DS51268).

After you have MPLAB ICD 2 hooked up to your PC and your target is powered as recommended in the MPLAB ICD 2 documentation, you may select the ICD as either a debugger or a programmer.

To select MPLAB ICD 2 as a debugger, use *Debugger>Select Tool>MPLAB ICD 2*.

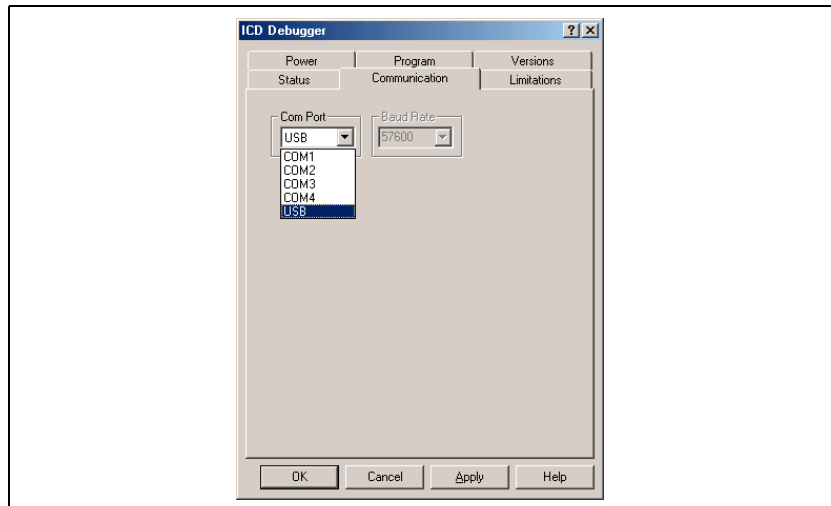
To select MPLAB ICD 2 as a programmer, use *Programmer>Select Programmer>MPLAB ICD 2*.

Note: When MPLAB ICD 2 is used as a programmer, it is recommended that it be disabled as a debugger, and vice versa.

MPLAB ICD 2 COMMUNICATIONS SETUP

The dialog to set up the communications is on the *Debugger>Settings* “Communication” tab.

FIGURE 27: MPLAB ICD COMMUNICATIONS SETTINGS



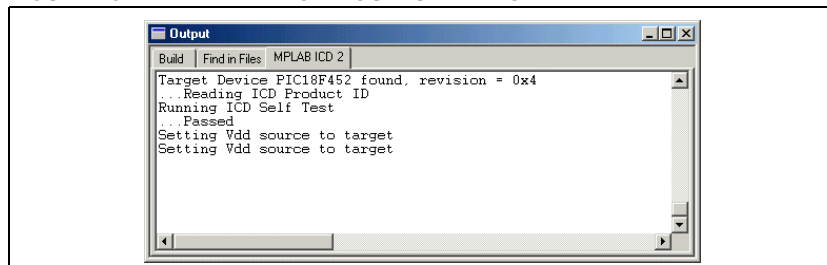
MPLAB ICD 2 OPERATING SYSTEM UPDATE

When first using MPLAB ICD 2, and when updating to a new version of MPLAB IDE, check to see if there is a new version of firmware. Incorrect versions may cause communications problems so make sure that the current versions are installed. If in doubt, look at the README document for MPLAB ICD 2.

To install the latest firmware select either *Debugger>Download ICD 2 Operating System* or *Programmer>Download ICD 2 Operating System*.

You should see successful initialization in the Output window. Otherwise, check your connections and try again. See MPLAB ICD 2 on-line help for troubleshooting information.

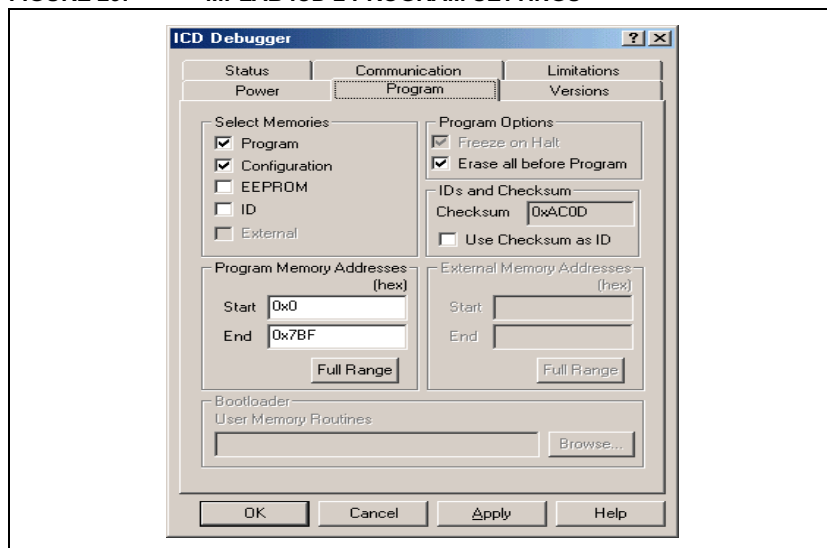
FIGURE 28: MPLAB ICD 2 OUTPUT WINDOW



MPLAB ICD 2 PROGRAMMING OPTIONS

Control of programming is from the Settings dialog, Program tab.

FIGURE 29: MPLAB ICD 2 PROGRAM SETTINGS



When you have selected MPLAB ICD 2 as a debugger, the "Configuration Bits" will always be selected in the "Program" tab. This is because some of these bits need to be written to enable the in-circuit debug features of the target device.

When you have selected MPLAB ICD 2 as a programmer, you have full control over the configuration bits.

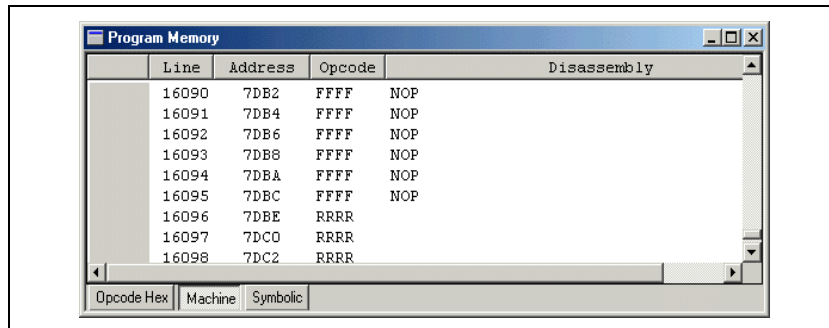
MPLAB® IDE v6.xx Quick Start

The “End” program address can be set lower than the full range of the device to speed up downloading, as shown in the dialog. Press the **Full Range** button to ensure that the entire program gets loaded into program memory.

MPLAB ICD 2 RESERVED MEMORY AREAS

Some of the memory window displays in MPLAB IDE will look different when using MPLAB ICD 2. Areas that are reserved will show with “RR” or “RRRR.” This is an indication that the device has memory at these locations, but it is not available while using MPLAB ICD 2 as a debugger.

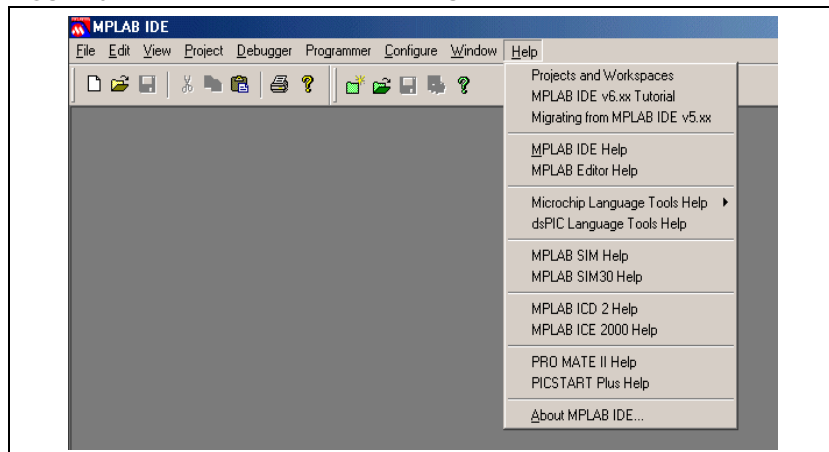
FIGURE 30: MPLAB ICD 2 RESERVED PROGRAM MEMORY AREAS



MPLAB IDE On-line Help

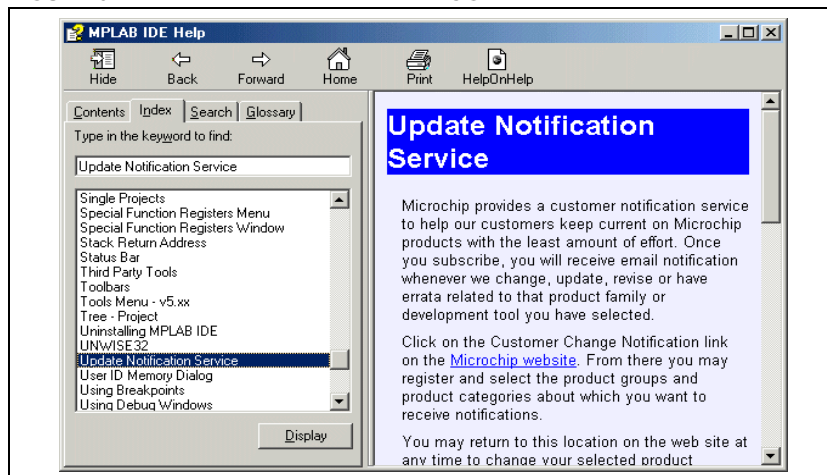
The MPLAB IDE comes with extensive on-line help, which is constantly being expanded. If you have questions while using MPLAB IDE, be sure to check the on-line Help for answers. Most importantly, the on-line help lists the support limitations that exist for a particular tool in its support of a particular device. Always try to review this section before working with a new device/tool combination.

FIGURE 31: MPLAB IDE HELP MENU



MPLAB IDE Help covers all aspects of MPLAB IDE and all the Microchip Tools. It also directs you to other types of assistance, such as the Microchip Update Notification system.

FIGURE 32: MPLAB IDE HELP DIALOG



Workspace and Project Debug Settings

It is important that you understand the difference between the MPLAB workspace and MPLAB projects.

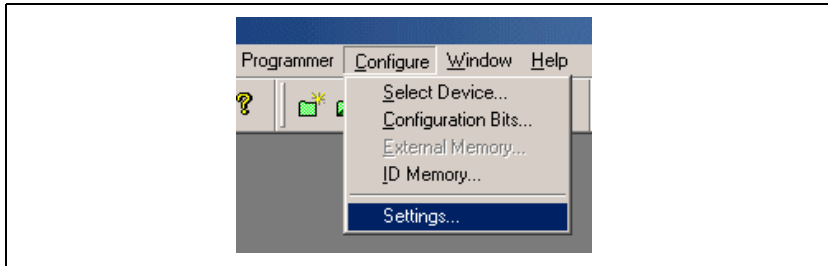
The MPLAB workspace is the desktop area in the MPLAB IDE application window. The workspace remembers which windows are open, which PICmicro device is selected, which debugger and programmer is being used, and how the hardware tools are connected to your PC. Generally, the workspace needs to be set up before you start making a project.

Projects are opened in the MPLAB workspace and they contain your source files, how to build them and which tools are used to build them. Projects are portable and can be moved to different directories or to a different computer, but the workspace is unique to your PC.

It is possible to have multiple projects in your workspace, but you can only have one workspace open in MPLAB IDE at a time.

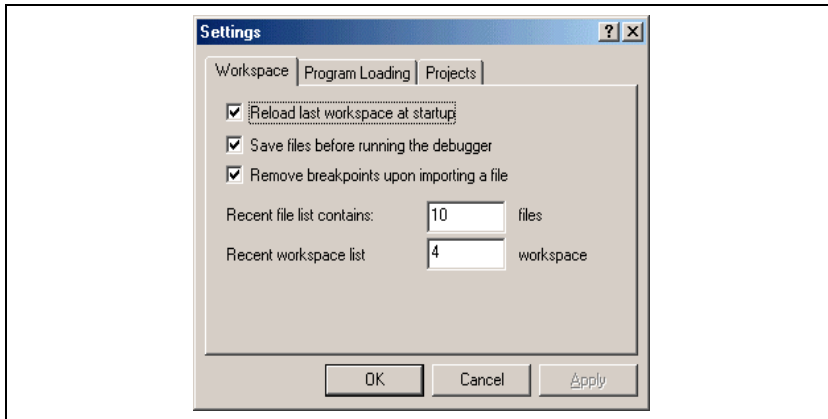
The *Configure>Settings* menu allows you to fine tune your debugging desktop workspace on MPLAB IDE. You do not have to change anything from the default settings for the Quick Start in this document, but you should be aware of these settings in case you want to change them later.

FIGURE 33: SETTINGS MENU SELECTION



The first thing you will see is the left tab on this multiple tabbed dialog, named "Workspace."

FIGURE 34: SETTINGS: WORKSPACE TAB



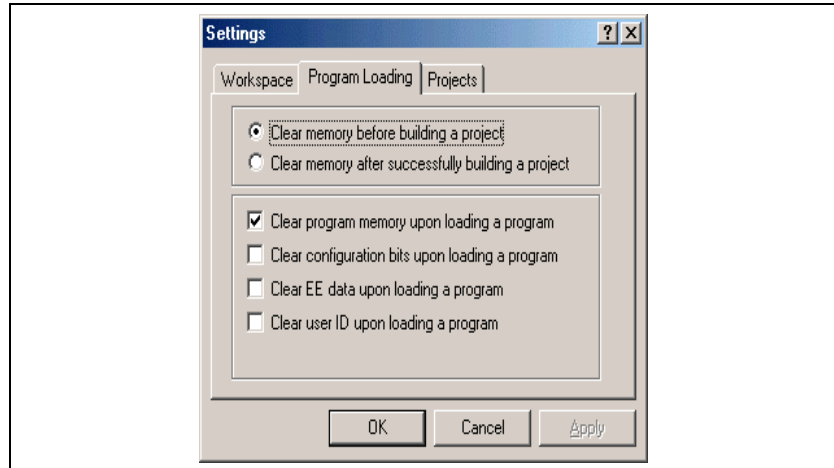
The "Workspace" tab on the *Configure>Settings* dialog is used to do these things:

- Start from your last workspace when entering MPLAB IDE. This is useful if you want to continue working on your project where you last left off.
- Save all your text files before starting emulation or simulation. This ensures that your work will be saved and any changes recompiled into your application before you start debugging.
- Remove breakpoints when you import a HEX file. Typically this is the desired action, but if, for some reason, you are making small changes to code manually and then reloading the HEX file, you might not want to clear all your breakpoints.

Note: The main reason for importing a HEX file is to program a device with previously compiled code. Every project produces a HEX file when it is built.

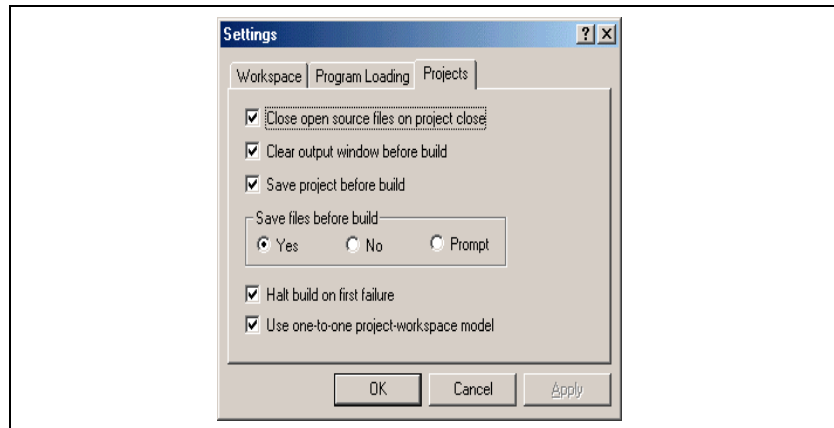
The “Program Loading” tab allows you to choose between clearing various areas of memory when a new program is loaded.

FIGURE 35: SETTINGS: PROGRAM LOADING TAB



The “Projects” tab on the *Configure>Settings* dialog has additional controls to customize actions when projects are built.

FIGURE 36: SETTINGS: PROJECTS TAB



This tab determines actions related to projects. These are the default settings, and it is recommended to keep these settings. Unchecking some boxes may result in loss of edited material if you are not very careful. The bottom selection "Use one-to-one project-workplace model" is related to how MPLAB IDE treats projects. When this is checked, the workspace allows only one project and, for all practical purposes, the workspace is the same as the project.

Note: If this box is unchecked, then the workspace can contain more than one project. This is useful if you are building an application "a block at a time," where different areas of code are compiled into separate memory blocks. You might have a project, for instance, that has a bootloader and the first version of your application. The bootloader is independent of the application, and will be used in the future to download an updated version of your application. Only one project is active at a time. To activate a project in a multiple project workspace, put the cursor on the project in the Project window and use the right mouse button to activate the current project.

NOTES:



WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office

2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: <http://www.microchip.com>

Rocky Mountain

2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-4338

Atlanta

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

Boston

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

Chicago

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

Dallas

4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Detroit

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

Kokomo

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

Los Angeles

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

San Jose

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

Toronto

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

ASIA/PACIFIC

Australia

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

China - Beijing

Microchip Technology Consulting (Shanghai)
Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu

Microchip Technology Consulting (Shanghai)
Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200 Fax: 86-28-86766599

China - Fuzhou

Microchip Technology Consulting (Shanghai)
Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai

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

China - Shenzhen

Microchip Technology Consulting (Shanghai)
Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-82350361 Fax: 86-755-82366086

China - Hong Kong SAR

Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India

Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaughnessy Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan

Microchip Technology Japan K.K.
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 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore

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

Taiwan

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

EUROPE

Austria

Microchip Technology Austria GmbH
Durlisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark

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

France

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

Germany

Microchip Technology GmbH
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy

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

United Kingdom

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

10/18/02