Freescale Semiconductor, Inc...

MCUez

Easy development software from the company that knows MCU hardware best

MCUez Installation and Configuration User's Manual
MCUEZINS/D
Rev. 2

MOTOROLA

MCUEZINS/D
Rev. 2

MCUez Installation and Configuration

User's Manual

Important Notice to Users

While every effort has been made to ensure the accuracy of all information in this document, Motorola assumes no liability to any party for any loss or damage caused by errors or omissions or by statements of any kind in this document, its updates, supplements, or special editions, whether such errors are omissions or statements resulting from negligence, accident, or any other cause. Motorola further assumes no liability arising out of the application or use of any information, product, or system described herein; nor any liability for incidental or consequential damages arising from the use of this document. Motorola disclaims all warranties regarding the information contained herein, whether expressed, implied, or statutory, *including implied warranties of merchantability or fitness for a particular purpose*. Motorola makes no representation that the interconnection of products in the manner described herein will not infringe on existing or future patent rights, nor do the descriptions contained herein imply the granting or license to make, use or sell equipment constructed in accordance with this description.

The computer program contains material copyrighted by Motorola, Inc., first published in 1997, and may be used only under a license such as the License For Computer Programs (Article 14) contained in Motorola's Terms and Conditions of Sale, Rev. 1/79.

Trademarks

This document includes these trademarks:

MCUez is a trademark of Motorola, Inc.

Windows, Windows 95, Windows NT, and Microsoft Word are registered trademarks of Microsoft Corporation in the U.S. and other countries.

WinEdit is a trademark of Wilson WindowWare.

List of Sections

List of Sections

Table of Contents

Table of Contents

Section 4. MCUez Environment

List of Figures

List of Figures

Section 1. General Information

1.1 Contents

1.2 Introduction

Motorola's MCUez™ is an integrated development environment which includes these tools:

MCUez Shell — A 32-bit application that serves as a configuration tool to manage MCUez projects. The **MCUez Shell** uses a graphical interface to create new projects or modify the environment for existing projects.

MCUez assembler — A 32-bit application that translates an assembly file into native code for the target processor. The assembler supports absolute and relocatable assembly code and generates a Motorola S file for absolute assembly. The MCUez assembler uses a graphical interface to select the assembly file, set up options, and provide on-line help. The input syntax for the assembler conforms to the Motorola assembly language input standard. The output object code format conforms to the *ELF/DWARF 2.0* object code format.

MCUez linker Merges all objects of an application into one absolute (*.abs*) file. The absolute file contains absolute (unrelocatable) code that can be loaded to a target using the MCUez debugger. The linker also generates a Motorola S file that can be burned into an EPROM.

MCUez debugger Used for emulation or cross-debugging of an embedded application for a specific hardware design. The MCUez debugger incorporates a powerful graphical interface and command line that enable the user to debug assembly files, correlate them with the data, manipulate register content, and read and manipulate memory content.

Additional features include:

- Unlimited memory, data, etc., windows selection for debug
- Drag and drop from one window to another for a different view
- Displays procedure chain and function parameters
- Displays function local variables and parameters in varying formats
- Performs stepping at source and assembly levels

1.3 Document Conventions

This section describes styles and terms used throughout this manual.

1.3.1 Typographic Styles in This Manual

This list identifies typographic styles used in this manual to define notational conventions:

- **Bold face** type is used for literal strings that must be used exactly as shown in the example and for the names of menus, windows, dialog boxes, icons, and buttons.

- `Courier` type face is used for all C code program listings, command lines, and directories.
- *Italics* are used where the string is a place holder that may be substituted for a string of the user's own design.
- Variable user inputs are in *`Courier`* italics.
- Filenames are in italics with all lower case letters, for example, *proj. ext.*

1.3.2 Extended Backus–Naur Form

Extended Backus–Naur Form (EBNF) is used frequently in this user's manual to describe file formats and syntax rules. See Appendix C in the *MCUez HC12 Debugger User's Manual*, Motorola document order number MCUEZDBG12, for more information.

1.3.3 Executable Files

Generic Name	Application Name
ahcxx.exe	MCUez assembler
elflink.exe	MCUez linker
mcushell.exe	MCUez shell tool
mcuez.exe	MCUez debugger

NOTE: *xx represents the MCU Family (HC05, HC08, HC12), depending on which MCU target is installed.*

1.4 System Requirements

MCUez applications require Windows 95® or Windows NT®.

General Information

Section 2. MCUez Installation

2.1 Contents

2.2 CD Installation

1. Insert the CD into the CD drive.
2. Select **Run** from the Windows 95/Windows NT **Start** menu.
3. Enter CD drive letter and setup program; for example: `D:\SETUP`.
4. Follow the instructions displayed on the monitor.

During installation, the system prompts the user to select one MCU target. Select one target per installation.

NOTE: *MCUez software and manuals can be downloaded from Motorola's website at http://www.mcu.motsps.com/mcuez.*

2.3 MCUez Directory

After installation, these subdirectories are created in the directory where the MCUez is installed:

`MCUez`	Root directory for default installation
`MCUez\Demo\HCXX`	Subdirectory for specific MCU Family. (XX defines the MCU release level HC05, HC08, or HC12.) The `DEMO` subdirectories contain sample programs that can be used to explore most of the features of the MCUez system. Each debugger or emulator creates its own subdirectory within this directory with programs configured for the selected target.
`MCUez\Docs`	Contains the user manuals in .pdf format
`MCUez\PROG`	Subdirectory that contains the MCUez programs (shell, assembler, linker, and debugger)
`MCUez\PROG\FPP`	Contains MCU-dependent FLASH parameter files that handle internal FLASH modules
`MCUez\PROG\MEM`	The `MEM` subdirectory contains memory files used to allocate memory maps for the HC05, HC08, and HC12.
`MCUez\PROG\REG`	The `REG` subdirectory contains register files for various devices that are emulated through the MCUez debugger command line and used to view input/output (I/O) devices and memory register information.

Section 3. MCUez Shell

3.1 Contents

3.2 Introduction

The **MCUez Shell** delivered with the MCUez package configures the project environment and provides a toolbar to launch related applications. **Figure 3-1** lists the different configuration files handled by the shell.

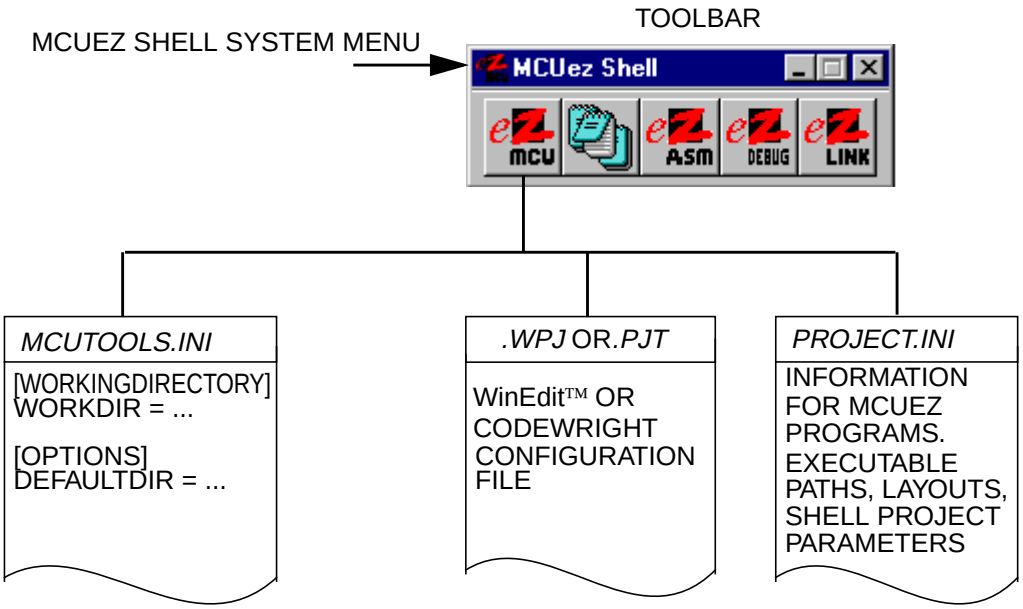

Figure 3-1. MCUez Shell

3.3 Executing the Shell

Double-click on the **MCUez Shell** icon in the MCUez program group or select **Start | Programs | MCUez ... | MCUez Shell** from the Windows® task bar.

The toolbar contains icons for related applications delivered with the MCUez package. It also contains icons for tools specified during project configuration, which is described in **3.6 Defining or Redefining a Project**.

3.4 Project Selection

Click on the **MCUez Shell** system menu to display the list of available projects. **Figure 3-2** illustrates that projects for all MCU targets were loaded during installation. User-defined projects also will be listed in this menu.

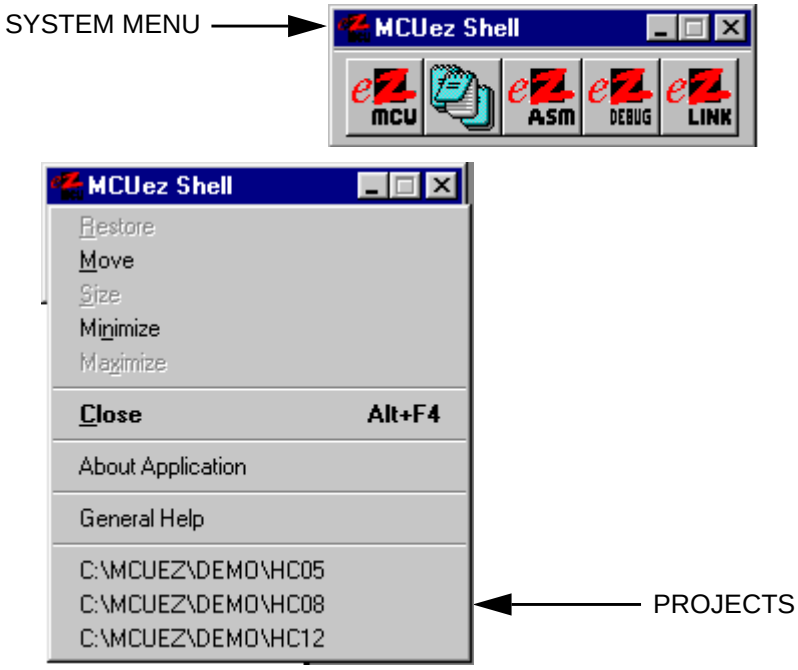

Figure 3-2. List of Projects

The user should select a project from the list associated with the MCU for the development environment. The current project directory will change to the directory containing the selected project.

The *project.ini* and *default.env* files residing in the selected project directory are maintained by the project **Configuration** dialog box and used to reconfigure the environment.

3.5 Project Configuration

Click on the first icon in the toolbar to create a new project, open an existing project, or change environment settings for a project.

The **Configuration** dialog will appear as shown in **Figure 3-3**.

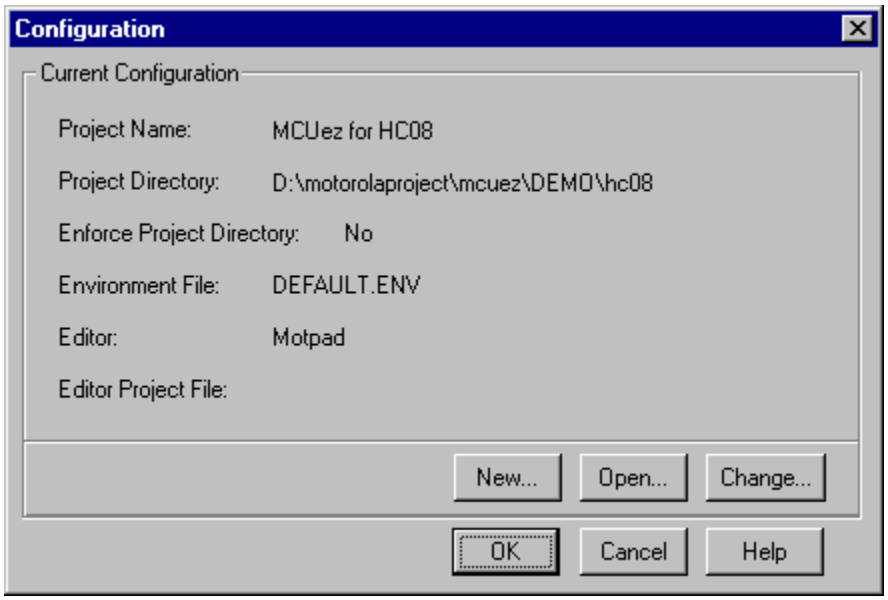

Figure 3-3. Current Configuration Window

The project configuration displays this information:

- **Project Name** — User-defined project name, usually denotes target
- **Project Directory** — User-defined working directory where source files are located
- **Enforce Project Directory** — Indicates whether the current project directory will be the working directory for all projects or not
- **Environment File** — Contains environment variables defined by using the **Configuration** dialog boxes
- **Editor** — User-assigned project editor
- **Editor Project File** — Editor configuration file, for instance, *project.wpj*

3.5.1 New Project

To start a new project, follow these steps:

1. Click **New...** in the **Configuration** dialog box to define a new project directory.
2. Create a new project by selecting the **Empty** radio button.
3. Select **From Current Project** to use configuration settings from the previously opened project for the new project.
4. Enter the directory path or browse to the directory, as shown in **Figure 3-4**.
5. Click **OK** and the **New Configuration** dialog box will appear.
6. Set project parameters, starting with the **Project Name** as shown in **Figure 3-5**.

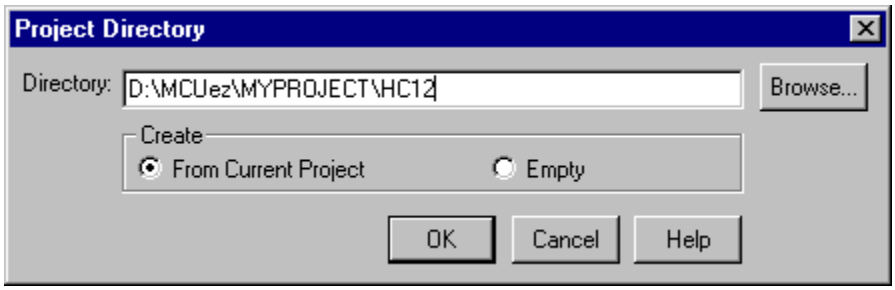

Figure 3-4. Defining a New Project

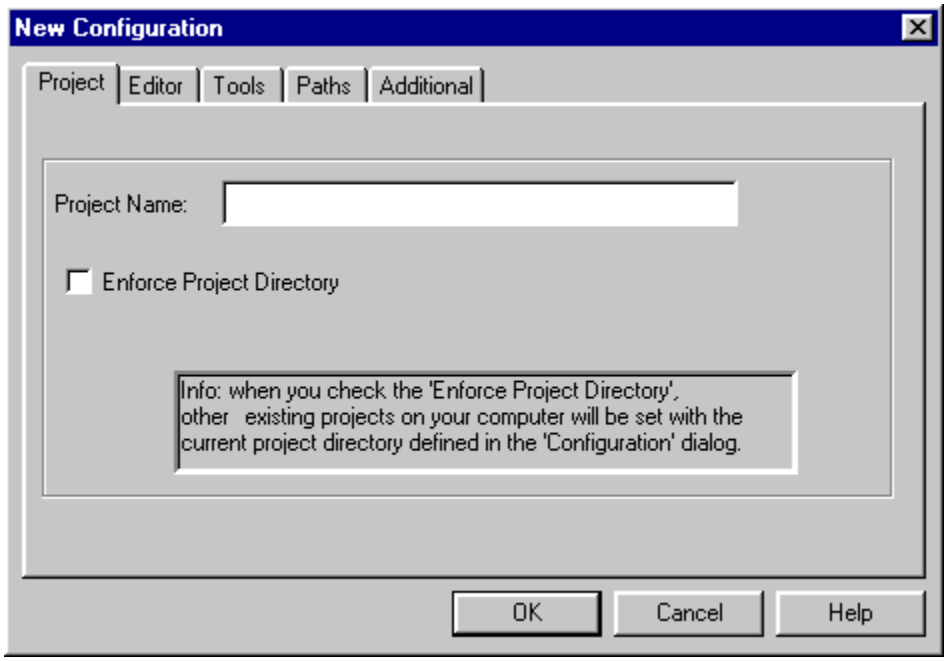

Figure 3-5. New Configuration Dialog Box

Refer to **3.6 Defining or Redefining a Project** for information about configuration settings (**Editor**, **Tools**, etc.).

3.5.2 Opening a Project

Select **Open...** in the **Configuration** dialog box. Select a project directory to open another project. This is equivalent to selecting a project from the list in the **MCUez Shell** system menu.

Ensure that a *default.env* file exists in this directory. For new projects, the *default.env* and *project.ini* files are automatically created. If a *default.env* file is not found, **Environment File: doesn't exis**t is displayed in the main **Configuration** dialog box.

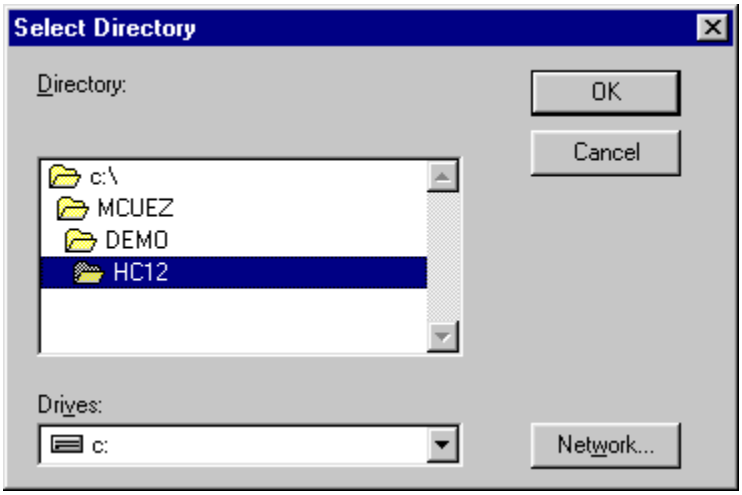

Figure 3-6. Selecting a Project Directory

After selecting an existing project directory that contains a *default.env* file, settings stored in the configuration files (*project.ini* and *default.env*) are displayed in the various tabs (**Project**, **Editor**, **Tools**, **Path**, and **Additional**) of the **Configuration** dialog box.

3.5.3 Changing Project Environment

Select **Change...** in the **Configuration** dialog box to display the **Current Configuration** dialog box and change parameters for the current project. This dialog box is the same as the one shown in **Figure 3-6**, except the title will be **Current Configuration**. Procedures for configuring the project environment are explained in **3.6 Defining or Redefining a Project**.

3.6 Defining or Redefining a Project

Each project can have its own parameters. Define all tools and parameters associated with a project in the **New Configuration** or **Current Configuration** dialog box. From the main **Configuration** dialog box, click **New...** to create a new configuration or **Change...** to edit the current project.

3.6.1 Project Tab

Refer to **Figure 3-5** for an example of the project tab in the **Configuration** dialog box. Define the name of the project in the **Project Name** edit box. This name will be displayed in the main **Configuration** dialog box.

Check **Enforce Project Directory** to define the project directory as the default directory. Leave this option unchecked to define the project directory as the temporary working directory.

Enforcing the project directory also enables MCUez tools to be started from the desktop or Windows Explorer.

3.6.2 Editor Tab

Set the project editor to be used with the application source files in the **Editor** tab as shown in **Figure 3-7**.

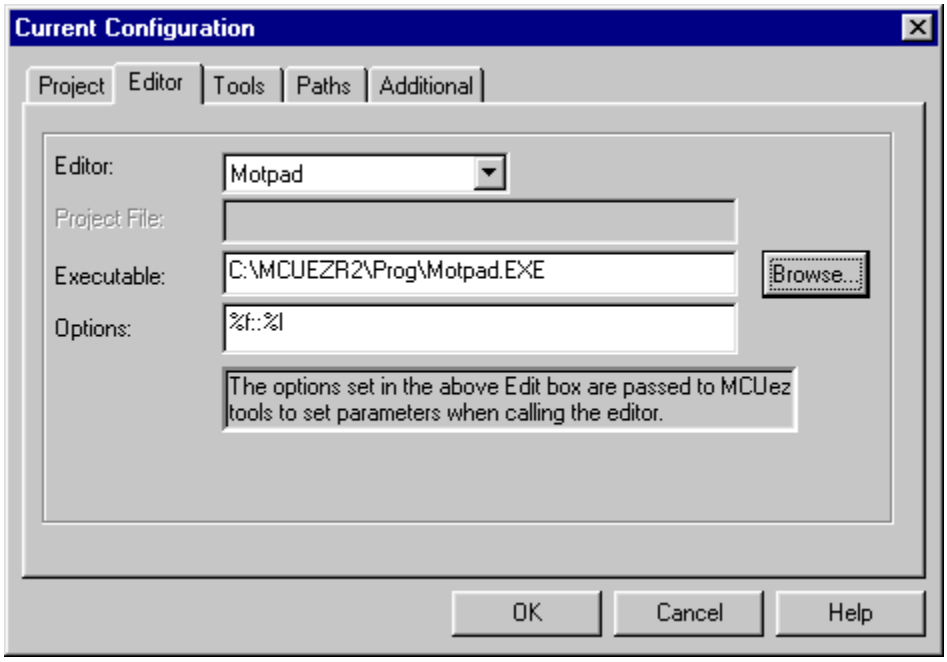

Figure 3-7. Setting the Editor

Editor

In the **Editor** combo box, select an editor such as: Motpad, WinEdit, Codewright, Notepad, or Wordpad. Select Others, to specify another editor.

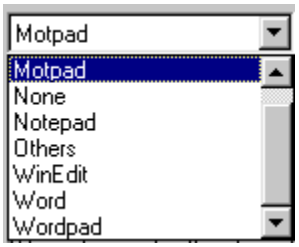

Figure 3-8. Editor Combo Box

Project File

Enter the name of the associated editor project file (if applicable) that contains the editor settings. The **MCUez Shell** automatically generates the appropriate project files for WinEdit and Codewright.

NOTE: *If WinEdit is selected, a .wpj file must be associated with the editor. If Codewright is selected, a .pjt file must be associated with the editor.*

Executable

In the **Executable** field, enter the path to the editor executable file. An icon associated with the editor will appear in the **MCUez Shell** toolbar. If the editor path is not valid, the icon shown to the left appears in the toolbar.

Options

Define editor options in the **Options** edit box. These options are saved in the editor's project file and appended to the editor command line when the editor is executed. Consult the appropriate editor manual for available options.

For most editors, the %f option identifies the source file containing an error. The %l option identifies the line number where an error occurred.

When an error message in the assembler or linker is double clicked, the %f and %l options enable the editor to open the source file and highlight the line containing the error.

3.6.3 Tools Tab

In this tab, define applications that can be launched from the **MCUez Shell** toolbar. Application paths are set in the **Executable** edit box as shown in **Figure 3-9**.

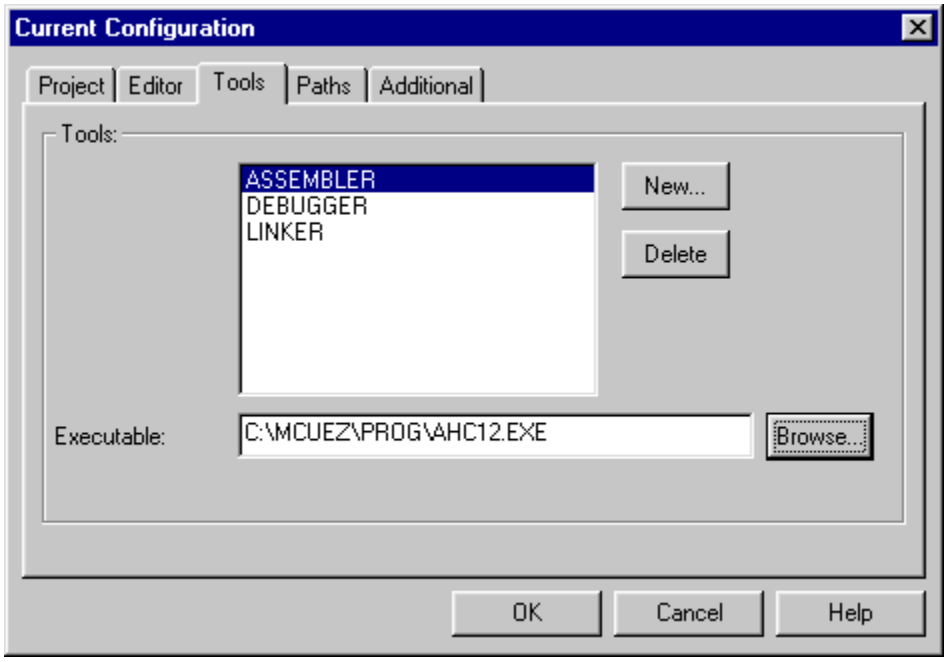

Figure 3-9. Defining Application Tools

Select **New...** to open the **New Tool** dialog box.

Enter the program name in the edit box. Some names are reserved for MCUez applications or other tools. Reserved names are ASSEMBLER, LINKER, DEBUGGER, COMPILER, BURNER, MAKER, LIBRARIAN, and DECODER. The program name is listed in the **Tools** tab and used for the help tip in the toolbar, as show in **Figure 3-10**.

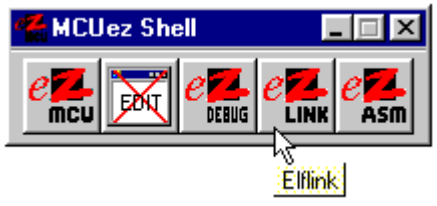

Figure 3-10. Toolbar Help Tip

Browse to or enter the path to the program in the **Executable** edit box. The program icon is loaded automatically from the executable file and displayed on the toolbar. If the shell cannot find the application defined in the executable path, the icon shown to the left is set in the toolbar.

If the shell finds the application, but the executable does not contain an icon, the default icon shown to the left is set in the toolbar.

3.6.4 Paths Tab

Working paths for defined tools are set in the **Paths** tab, as shown in **Figure 3-11**.

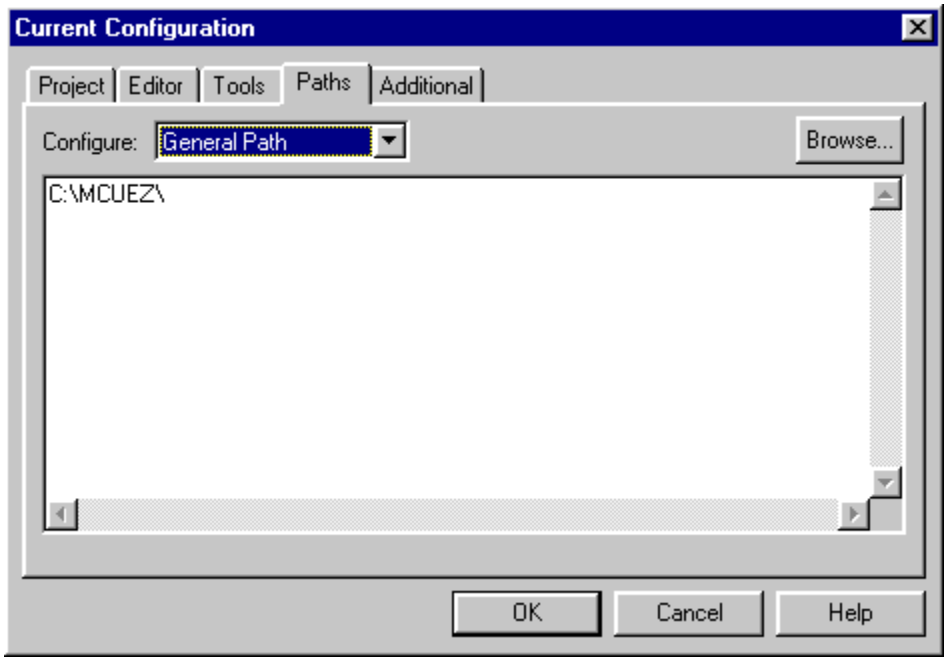

Figure 3-11. Setting Paths for Tools

Click on the **Configure** combo box to view the list of paths, as in **Figure 3-12**.

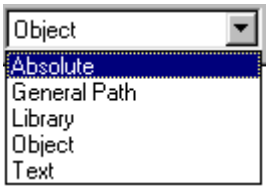

Figure 3-12. List of Paths

During installation, MCUez automatically sets up these paths. If the project requires specific paths, select an option from the combo box and enter the path or paths in the edit box. Click the browse button to help find the path. Paths defined here are valid only for MCUez tools or other user-assigned tools.

3.6.5 Additional Tab

Use the **Additional** tab to set additional parameters for tools. These parameters are saved in the *default.env* file. **Figure 3-13** is an example of setting additional parameters.

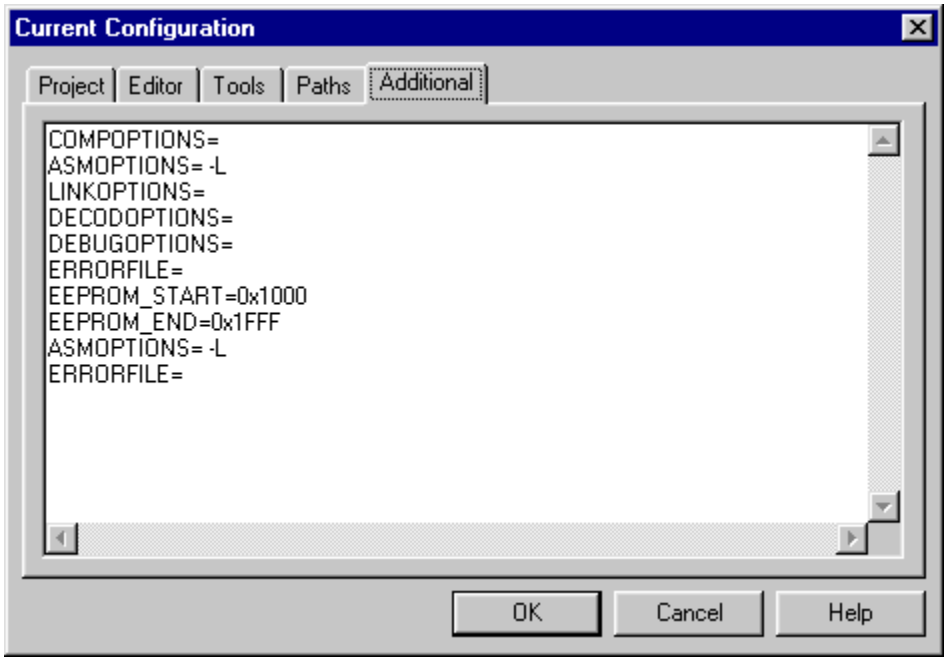

Figure 3-13. Setting Additional Tool Parameters

NOTE: *Lines stored in a default file (default.env, project.wpj, or .pjt file) are limited to 1024 characters.*

Section 4. MCUez Environment

4.1 Contents

4.2 Introduction

The **MCUez Shel**l is used to define the MCUez project environment. Parameters for the environment are stored in the *default.env* and *project.ini* files. This section describes how to edit the *default.env* file parameters and manually configure various editors, such as WinEdit and Codewright.

NOTE: *The Make command is used only when the C compiler has been installed. If the C compiler has not been installed, the assembler will be invoked and the Make command will be inactive.*

4.3 Setting Path Parameters

Various parameters of the MCUez environment are set by environment variables. No blanks are allowed in an environment variable definition. The syntax is:

```
KeyName=ParameterDefinition
```

Example:

```
GENPATH=D:\PROJECTS\TESTS
```

Methods of defining parameters:

- Use system environment variables supported by the operating system.
- Put definitions in a file called *default.env* in the default directory.
- Put definitions in the file specified in the system environment variable named `ENVIRONMENT`.

The default directory can be set with the `DEFAULTDIR` system environment variable. When searching for an environment variable, all MCUez programs search the system environment, then the *default.env* file, and finally the global environment file given by `ENVIRONMENT`. If no definition is found, a default value is assigned.

4.3.1 Paths

Most environment variables contain path lists that indicate where to look for files. A path list is a list of directory names separated by semicolons or a directory name preceeded by an asterick and follows this syntax:

```
DirSpec;DirSpec
*DirectoryName
```

Example:

```
GENPATH=C:\PROJECTS\TESTS;D:\PROJECTS\TESTS
GENPATH=*C:\PROJECTS\TESTS
```

If a directory name is preceded by an asterisk (*), the MCUez program recursively searches the directory tree for a file, not just the given directory. Directories are searched in the order they appear in the path list.

NOTE: *When using WinEdit or Codewright, if the system environment variable `DEFAULTDIR` is set, ensure that this variable is assigned the project directory specified in the WinEdit or Codewright project configuration. If the specified directories differ, files may be placed in the wrong directory. This could affect proper execution of the MCUez application.*

4.4 Configuring WinEdit

Follow these steps to configure WinEdit. The **MCUez Shell** automatically creates the editor project file required by WinEdit.

1. Click the **ezMCU** icon on the toolbar to start the **MCUez Shell**.
2. Click the **Change...** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
3. Select the **Editor** tab and select WinEdit from the **Editor** drop down box.
4. In the **Project File** edit box, enter a name for the WinEdit project file. WinEdit project files have a *.wpj* extension. Do not specify a path in front of the filename.
5. In the **Executable** edit box, enter the WinEdit executable file, including path. Use the **Browse** button to search for the executable file, if necessary.

6. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After selecting WinEdit, the shell automatically enters the appropriate options.

NOTE: *WinEdit version 3.1 or lower cannot be started with a line number (%l) as a parameter. For older versions of WinEdit, options should be changed to* ***%f*** *only.*

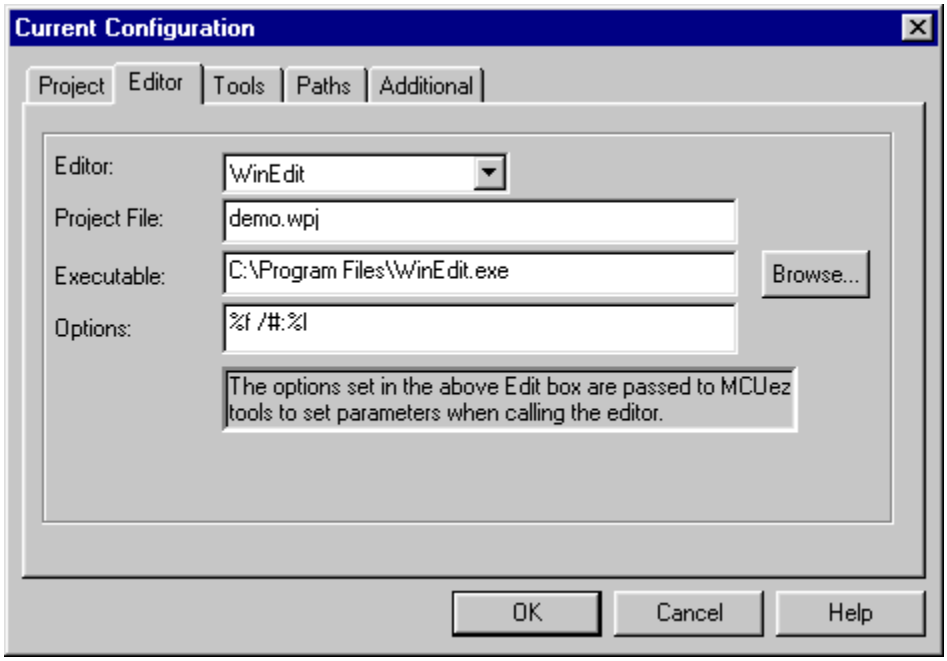

Figure 4-1. Editor Configuration for WinEdit

7. Select the **Tools** tab.
8. Some tools specified here are automatically associated with WinEdit commands (menu entries and buttons).

   `ASSEMBLER` is automatically associated with the WinEdit `Compile` command.

   `MAKER` is associated with the WinEdit `Make` command.

   `LINKER` is associated with the WinEdit `Rebuild` command.

   `DEBUGGER` is associated with the WinEdit `Debug` command.

   `DECODER` is associated with the WinEdit `Execute` command.

9. Click **OK** to close the **Current Configuration** dialog.
10. The specified editor project file is created in the project directory and the associated icon is displayed in the MCUez toolbar.

4.4.1 Configuring *WINEDIT.INI*

WinEdit looks for a file named *EDOUT* to retrieve information about errors or warnings detected during assembly, linking, or debugging. To ensure that the appropriate file is used, initialize the OUTPUT variable in the WinEdit section of the *winedit.ini* file:

```
[WinEdit]
...
OUTPUT=.\EDOUT
```

4.4.2 Storing and Restoring Configuration Files

After starting WinEdit, select **Configure...** from the **Project** menu to open the **Project Management** dialog box.

- Select **Save...** to save the current configuration.
- Select **Open...** to load a previously saved configuration.

NOTE: *Each time the working directory is changed, WinEdit must be closed and restarted.*

4.4.3 WinEdit Menu Selection

Select **Project | Compile** to execute the tool specified in the `Compile` command line. The **MCUez Shell** automatically links the MCUez assembler program to this entry.

Select **Project | Make** to execute the tool specified in the `Make` command line. The **MCUez Shell** automatically links the maker program to this entry.

Select **Project | Rebuild** to execute the tool specified in the `Rebuild` command line. The **MCUez Shell** automatically links the MCUez linker program to this entry.

Select **Project | Debug** to start the tool specified in the `Debug` command line. The shell automatically links the MCUez debugger program to this entry.

Select **Project | Execute** to start the tool specified in the `Execute` command line. The shell automatically links the program associated with the **Decoder** tool to this entry.

4.4.4 WinEdit Toolbar

Figure 4-2 shows the association between the command lines in the **Project Configuration** dialog and buttons in the toolbar.

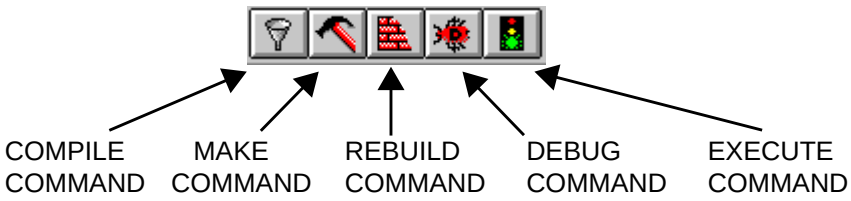

Figure 4-2. WinEdit Toolbar

4.5 Configuring Codewright

Follow these steps to configure Codewright. The **MCUez Shell** automatically creates the project file required by Codewright.

1. Click the **ezMCU** icon on the toolbar to start the **MCUez Shell**.
2. Click the **Change...** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
3. Select the **Editor** tab and select **Codewright** from the **Editor** drop down box.
4. In the **Project File** edit box, enter a name for the Codewright project file. Usually, Codewright project files have a *.pjt* extension. Do not specify a path in front of the filename.
5. In the **Executable** edit box, enter the Codewright executable file, including path. Use the **Browse** button to search for the executable file, if necessary.

6. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After selecting Codewright, the shell automatically enters the appropriate options. **Figure 4-3** is a typical setup for Codewright.

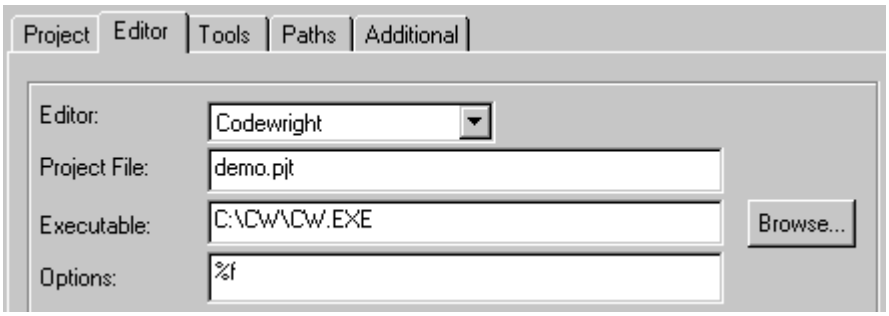

Figure 4-3. Editor Configuration for Codewright

7. Select the **Tools** tab.
8. Some tools specified here are automatically associated with Codewright commands, such as menu entries and buttons.

   `ASSEMBLER` is automatically associated with the Codewright `Compile` and `Debug Compile` commands.

   `MAKER` is associated with the Codewright `Make` command.

   `LINKER` is associated with the Codewright `Rebuild` command.

   `DEBUGGER` is associated with the Codewright `Debug` command.

   `DECODER` is associated with the Codewright `Execute` command.
9. Click **OK** to close the **Current Configuration** dialog.
10. The specified editor project file is created in the project directory and the associated icon is displayed in the MCUez toolbar.

4.5.1 Switching Configurations

Select **Project | Open** to load a previously saved project configuration file.

4.5.2 Codewright Menu

Select **Utility | Compil**e to execute the tool specified in the `Compile` command line. The **MCUez Shell** automatically links the MCUez assembler program to this entry.

Select **Utility | Compile(Debug)** to start the tool specified in the `Debug Compile` command line. The shell automatically links the assembler program to this entry.

Select **Utility | Make** to start the tool specified in the `Make` command line. The shell automatically links the maker program to this entry.

Select **Utility | Rebuild** to start the tool specified in the `Rebuild` command line. The shell automatically links the MCUez linker program to this entry.

Select **Utility | Debug** to start the tool specified in the `Debug` command line. The shell automatically links the MCUez debugger program to this entry.

Select **Utility | Execute** to start the tool specified in the `Execute` command line. The shell automatically links the decoder program to this entry.

4.5.3 Codewright Toolbar

Figure 4-4 illustrates the toolbar functions.

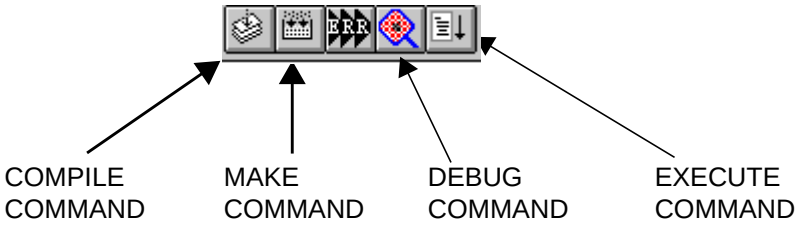

Figure 4-4. Codewright Toolbar

4.6 Configuring Motpad

Motpad is the default editor. However, if another editor has been defined as the default, redefine Motpad as follows:

1. Click the **ezMCU** icon on the toolbar to start the **MCUez Shell**.
2. Click the **Change...** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
3. Select the **Editor** tab and select Motpad from the **Editor** drop down box.
4. In the **Executable** edit box, enter the Motpad executable file, including path. Use the **Browse** button to search for the executable file, if necessary.
5. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After Motpad is selected, the shell automatically enters the appropriate options. **Figure 4-5** shows a typical Motpad configuration.

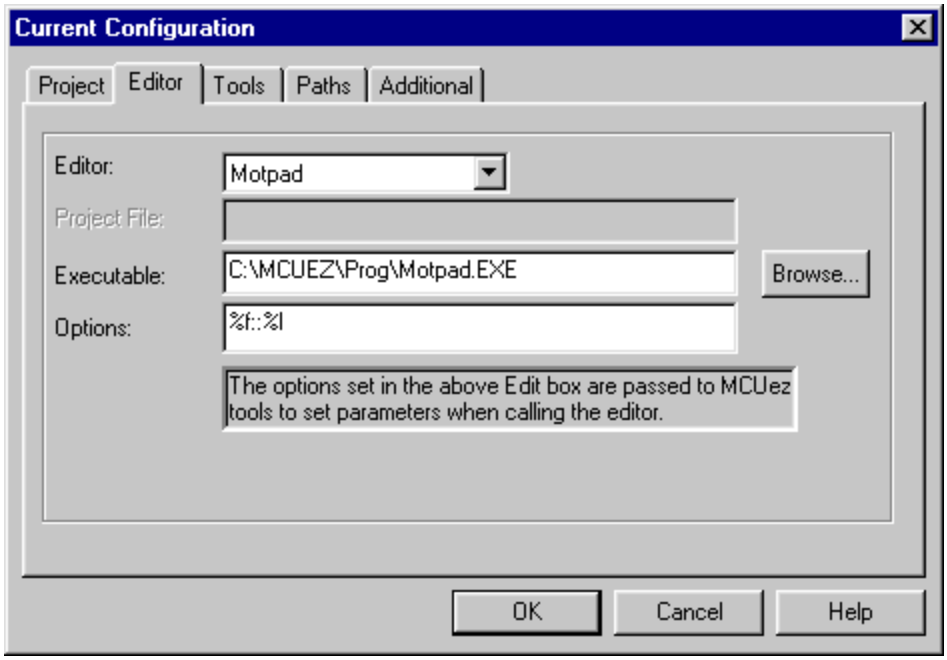

Figure 4-5. Editor Configuration for Motpad

6. Click **OK** to close the dialog.
7. Motpad is set as the default project editor and the associated icon is displayed in the MCUez toolbar.

4.7 Configuring Notepad

To define Notepad as the project editor:

1. Start the **MCUez Shell**.
2. Click the **ezMCU** icon in the toolbar to open the **Configuration** dialog box.
3. Select the ***Change...*** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
4. Select the **Editor** tab and select Notepad from the **Editor** drop down box.
5. In the **Executable** edit box, enter the Notepad executable file, including path. Use the **Browse** button to search for the executable file, if necessary.
6. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After selecting Notepad, the shell automatically enters the appropriate options. **Figure 4-6** shows a typical Notepad configuration.

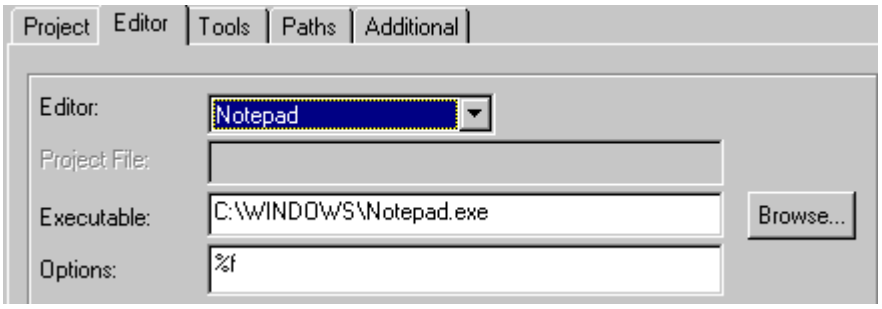

Figure 4-6. Editor Configuration for Notepad

7. Click **OK** to close the dialog.
8. Notepad is set as the default project editor and the associated icon is displayed in the MCUez toolbar.

4.8 Configuring Wordpad

The MCUez assembler and linker cannot process input files containing Wordpad format information. Save all source files as ASCII text files.

To define Wordpad as the project editor:

1. Start the **MCUez Shell**.
2. Click on the **ezMCU** icon in the toolbar to open the **Configuration** dialog box.
3. Select the **Change...** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
4. Select the **Editor** tab and select Wordpad from the **Editor** drop down box.
5. In the **Executable** edit box, enter the Wordpad executable file, including path. Use the **Browse** button to search for the executable file, if necessary.
6. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After selecting Wordpad, the shell automatically enters the appropriate options. **Figure 4-7** shows a typical Wordpad configuration.

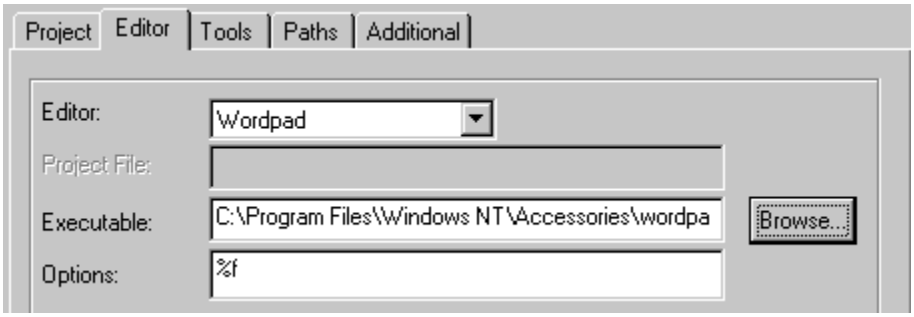

Figure 4-7. Editor Configuration for Wordpad

7. Click **OK** to close the dialog.
8. Wordpad is set as the default project editor and the associated icon is displayed in the MCUez toolbar.

4.9 Configuring Word

The MCUez assembler and linker cannot process input files containing Word format information. Save all source files as ASCII text files.

To define Microsoft Word as the project editor:

1. Start the **MCUez Shell**.
2. Click on the **ezMCU** icon in the toolbar to open the **Configuration** dialog box.
3. Select the **Change...** button in the **Configuration** dialog box to open the **Current Configuration** dialog box.
4. Select the **Editor** tab and select **Word** from the **Editor** drop down box.
5. In the **Executable** edit box, enter the **Word** executable file, including path. Use the **Browse** button to search for the executable file, if necessary.
6. The **Options** edit box contains options used for error feedback in the MCUez assembler and linker. After selecting **Word**, the shell automatically enters the appropriate options. **Figure 4-8** shows a typical Word configuration.

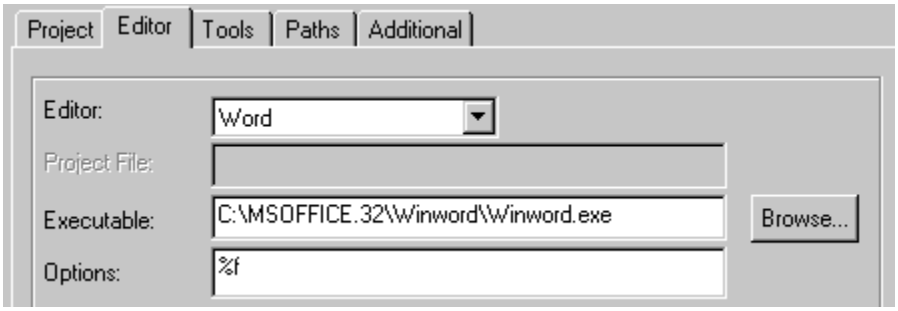

Figure 4-8. Editor Configuration for Word

7. Click **OK** to close the dialog.
8. **Word** is set as the default project editor and the associated icon is displayed in the MCUez toolbar.

Need to know more? That's ez, too.

Technical support for MCUez development tools is available through your regional Motorola office or by contacting:

Motorola, Inc.
6501 William Cannon Drive West
MD:OE17
Austin, Texas 78735
Phone (800) 521-6274
Fax (602) 437-1858
CRC@CRC.email.sps.mot.com

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217, 1–800–441–2447 or 1-303-675-2140.
Customer Focus Center: 1–800–521–6274
JAPAN: Motorola Japan Ltd.; SPD, Strategic Planning Office, 141, 4–32–1, Nishi–Gotanda, Shinagawa–ku, Tokyo, Japan, 03–5487–8488
ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd., Silicon Harbour Centre, 2 Dal King Street, Tai Po Industrial Estate, Tai Po, New Territories, Hong Kong, 852–26668334
Mfax™, Motorola Fax Back System: RMFAX0@email.sps.mot.com; http://sps.motorola.com/mfax/; TOUCHTONE, 1-602–244–8609; US & Canada ONLY, 1–800–774–1848
HOME PAGE: http://motorola.com/sps/

Mfax is a trademark of Motorola, Inc.

MOTOROLA
Semiconductor Products Sector