

# AxIDE 4

Owners Manual


Version 0.1


<b>Installing the PC Software .....</b>	<b>3</b>
<b>Installing the BDM drivers .....</b>	<b>3</b>
<b>Starting AxIDE .....</b>	<b>4</b>
<b>Creating a new project .....</b>	<b>4</b>
<b>The Edit Window.....</b>	<b>4</b>
<b>Choosing a Memory Map .....</b>	<b>5</b>
<b>Building the project.....</b>	<b>5</b>
<b>Loading the program into memory .....</b>	<b>5</b>
<b>Using the Terminal .....</b>	<b>6</b>
<b>Board Reset .....</b>	<b>7</b>
<b>Executing and Halting the program.....</b>	<b>7</b>
<b>Restarting the program.....</b>	<b>7</b>
<b>Single step execution.....</b>	<b>7</b>
<b>Setting Breakpoints.....</b>	<b>7</b>
<b>Debug Windows.....</b>	<b>8</b>
<b>Source Window .....</b>	<b>8</b>
<b>Disassembly Window .....</b>	<b>8</b>
<b>Register Windows.....</b>	<b>9</b>
<b>Function Window .....</b>	<b>9</b>
<b>Watch Window.....</b>	<b>10</b>
<b>Variables Window .....</b>	<b>10</b>
<b>Data Window.....</b>	<b>10</b>
<b>Debug Tester .....</b>	<b>11</b>
<b>Project Options.....</b>	<b>12</b>
Files.....	12
Compiler .....	12
Output .....	12
Debug.....	12
<b>Troubleshooting .....</b>	<b>13</b>

## Installing the PC Software

You should install this software BEFORE connecting the hardware to your PC.

AxIDE 4 requires a PC running Windows. To install the software insert the support CD. If the setup program doesn't start automatically run SETUP.EXE from the CD.

AxIDE 4 works with the open source GNU GCC compiler for HC11/12. Click  HC12 Compiler and follow the onscreen instructions to install it if it's not already on your PC or network. You can also download the latest version and other files at the projects main site here: <http://m68hc11.serveftp.org>

After installing the compiler click  AxIDE 4 to install the AxIDE software. Follow the onscreen instructions.

**NOTE:** some command line tools you may want to use may not work well using long directory paths (for example "c:\verylongpathname\commandline tools\"). For the most trouble free operation you should use short path names (without spaces) when installing. See Troubleshooting at the end of this document for more information on directory paths.

## Installing the BDM drivers

AxIDE 4 can be used with either a stand alone AxBDM pod or an Axiom development board with AxBDM onboard. In order to use either you must install the BDM drivers on your PC.

1. If using a development board with AxBDM onboard:
  - a. **NOTE:** All the jumpers should be set to their default positions to enable the BDM. See the board documentation for details.
  - b. If the board requires external power supply (see boards documentation) connect that to a power outlet now.
2. Use a USB cable to connect the USB BDM connector on the board or pod to your PC.
3. Windows should say "New Device Found"
4. If asked to search Windows Update for software select NO then Next
5. Choose "Install from a specific location"
6. Browse to the drive or folder containing the AxIDE 4 software (the directory you chose during installation)
7. When you choose Next, windows should locate and install the correct driver

## Starting AxIDE

AxIDE 4 uses project files to store your project settings such as source files, processor type, compiler options, etc.

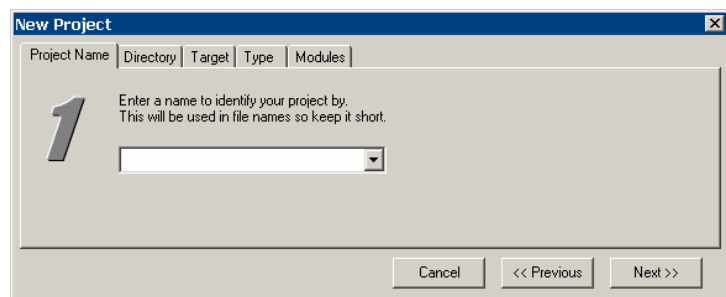
On startup a dialog box will open to let you select a previously opened project or create a new one. If you don't want to start with a project select Cancel. This dialog can be disabled by the checkbox at the bottom. You can see it again from the File menu by selecting Startup Wizard.



## Creating a new project

You can create a new project at startup by selecting Create a new Project and selecting OK. Or at any time while the program is running, from the **Project menu** select **New Project**.

1. Type a short name for your project (preferably without spaces) then click Next >>
2. Click [Browse] and choose a directory for your project or type the full path name, for example:  
C:\projects\test  
then click Next >>



3. Select the target board you're creating the project for then click Next >>
4. Select the type of project you're creating. If not sure just choose Simple. Click Next >>
5. You can select any example module code you want to include in this project. This is just source code that is added to create a sample project to get you started quickly. Different modules are available based on the project type you selected.

For example, checking SCI COM will include a simple driver and example code to send and receive serial data using the SCI serial port. You don't have to choose any modules here if you don't want to. In this case a simple main loop with initialization code only will be supplied.

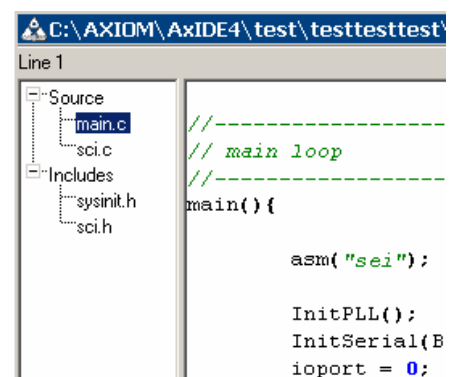
6. You can go back and change any selection you made previously by clicking the Tabs at the top of the dialog. When everything is set like you want, hit the Create button on the last screen. This will create your new project.

## The Edit Window

The Edit Window is useful for viewing and editing your project files. The left pane lists all the Source and Include files in your project. Double clicking on any file opens it in the right window.

The right window lets you edit the file. If you select another file the current file will be saved and the new file opened for editing.

This window also lets you view and edit Output files created by the Build process. More about this in the Build section.



## Choosing a Memory Map

Most development boards offer more than one type of memory such as Internal RAM, External RAM or Flash. A program must be compiled for a particular memory type in order for it to be located at the correct address when it's loaded.

You can modify the memory device you want the program to load and execute into at any time by selecting the memory drop down box from the main toolbar.

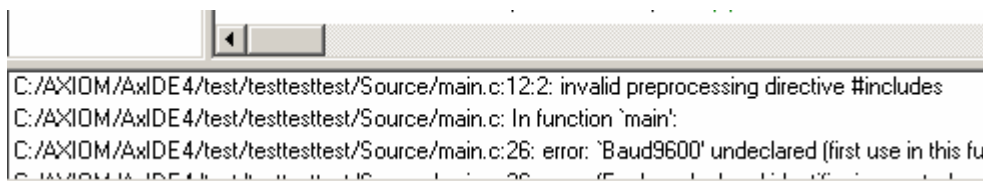


**NOTE:** After changing memory type you must **Build** the project again to modify the output files.

## Building the project

When you're ready to compile source code select the **Build** button on the toolbar. This will compile and link all the files in your project based on your **Project Settings**.

If there are any errors found an error message will be displayed. You will then be taken back to the **Edit Window** with the errors listed in the bottom pane.



Double clicking on the error message will open the source file indicated with the cursor near the point where the error was found. Edit the source to fix the error then click the Build button again.

After the project compiles without errors a new group of files will appear in the left pane of the Edit Window under Output. These are some of the output files created by the build process. You can click on them to view the output, however unless you know exactly what you're doing you should not modify them by hand.

## Loading the program into memory

After successfully Building your project you can load it into your board memory and execute it.

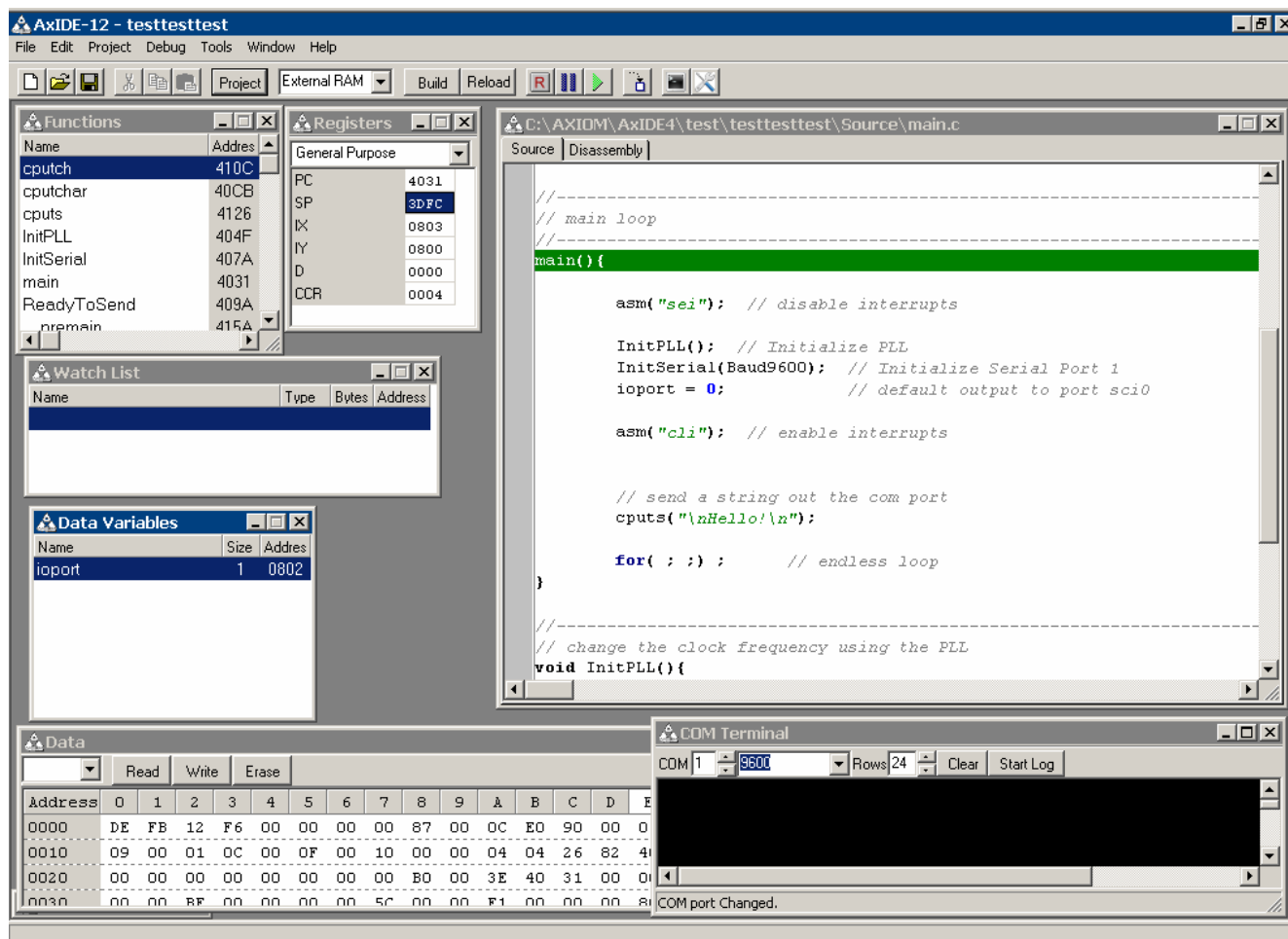
**NOTE:** In order to do this your development board **must be connected** to your PC either by an external AxBDM or thru the USB BDM connector on the board if available. See **Installing and Setup** for more information.

Click the **Debug** button on the main toolbar to load the project onto the board and prepare for debugging. At this point many things will happen quickly:

1. The compiled output files will be used to load the application binary onto the board's memory
2. Symbolic source information will be read and used to load the debug windows such as Functions, Registers, Source and Disassembly.
3. Based on option settings in the **Project Options, Debug** section) your program will:
  - a. Execute from its Start Address (usually `_start`)
  - b. Halt at the beginning of the `main()` function.


**NOTE:** The first time you debug a project you should resize and rearrange all the windows to your liking. AxIDE will remember this position and all future sessions will restore the screen to your last saved position.

Here is an example of a typical debugging session screen:



Notice the green bar in the Source or Disassembly window. This indicates the location of the current program counter, where the program is halted waiting to execute again.

## Using the Terminal

The screenshot above also shows the COM Terminal window open. Click the  button on the main toolbar to open it or choose **Serial Terminal** from the **Tools** menu.


This is useful when debugging programs that use the boards SCI serial port. To use it you must connect a DB9 serial cable between your PC COM port and the development board.

Whenever this terminal window is selected, any keys you press on your keyboard will be sent to the development board thru the serial cable connection. Likewise any SCI serial data your program outputs will be displayed in this terminal.


This provides a common, easy to implement terminal user interface for your programs.

## Board Reset


Your development board should have a Reset button that performs a hardware reset of the processor.

You can do the same thing from AxIDE by clicking the reset  button on the main toolbar or choosing Reset Target from the Debug menu.

## Executing and Halting the program

After loading the program into memory you can execute it by clicking the run  button on the main toolbar or choose **Run** from the **Debug** menu.

The program will execute from the current Program Counter location indicated by the green bar in the Source or Disassembly window. A status window will open indicating the execution status. You can not test functionality of your program.

You can halt execution at any time by pressing the STOP button in the status window or by pressing the  button on the toolbar.


## Restarting the program

Whenever you load your program into memory you may notice that the **Debug** button on the toolbar changes to a **Reload** button.



Selecting **Reload** will load the program into your boards memory again and reset the Program Counter to the beginning. Use this to restart your application from the beginning.

## Single step execution

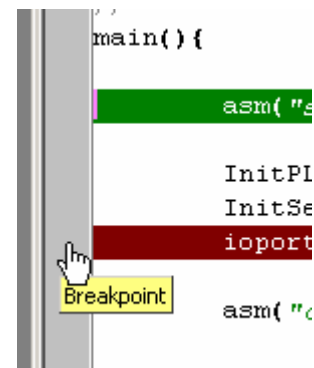
While debugging it can be useful to slow things down and execute just one instruction at a time. This is called single stepping. You can do this by pressing the single-step  button on the main toolbar. This will execute just one instruction in your program then halt. All the debug windows will be refreshed and updated with any changes caused by executing that instruction. If the program counter changed the green bar will be moved to the next instruction.

## Setting Breakpoints

You can have your program to halt when it reaches a certain place by setting a **Breakpoint**. Do this by **Double-Clicking** in the **Gutter** area on the left side of the source window.

This will place a breakpoint on the line you double-clicked next to. Turn the breakpoint off by double-clicking there again.

You can set more than one breakpoint, limited by the processor on your target board. During program execution, whenever the breakpoint line is reached the program will halt and the Source window will be displayed with the cursor on that line. You can then execute or single-step again when you're ready to continue.

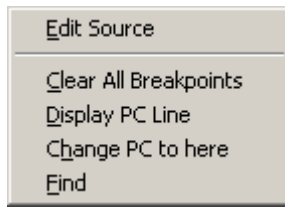


# Debug Windows

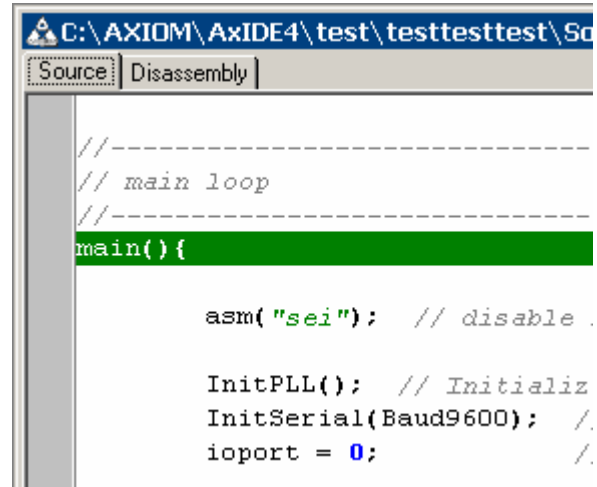
## Source Window

While debugging, the Source Window is displayed by clicking the Source Tab at the top of the window.

This window will only show one file at a time. While debugging this will be the project Source file that contains the current Program Counter (where the program is currently halted waiting to execute) indicated by the green bar.



Right-Clicking in this window will display a pop-up menu with several useful functions.



**Edit Source** – displays the current source file in the **Edit Window**.

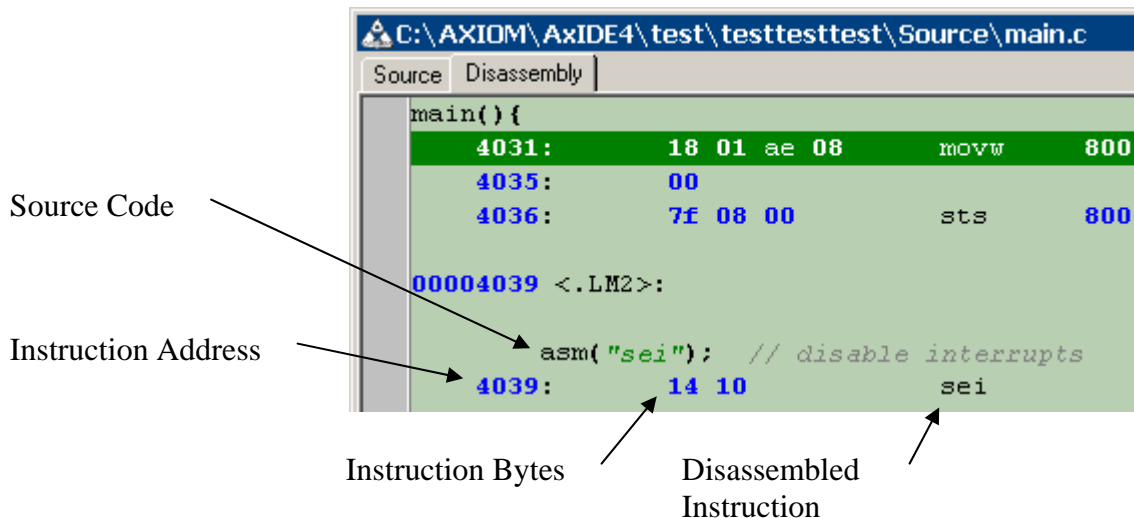
**Clear All Breakpoints** – removes any breakpoints you’ve set.

**Display PC Line** – opens the source file containing the current Program Counter and move the cursor to that line. Useful when you’ve browsed away and want to quickly come back to the line halted on.

**Change PC to here** – writes the Program Counter to the address of the line you right clicked on. After this any subsequent program execution or single stepping will begin from there.

## Disassembly Window

A single line of source code may contain more than one instruction to the processor. Each instruction consists of one or more bytes. When debugging it can be useful to look underneath the source code and see the instructions that are being executed. Use the Disassembly window for this.





Click the Disassembly tab at the top of the source window to display it. As you can see, this window contains much more information than the Source window.

Underneath each source line is the Program Counter address of the instruction, the low level processor code bytes in hexadecimal and they processor instruction itself.

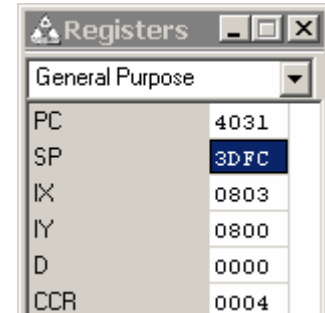
**NOTE:** The information in this window is generated by the linker and other tools at compile time (not real time) so it is accurate only for the most recent Build and memory Load.

The Disassembly window functions identically to the Source window. You can set breakpoints, single-step, etc from this window as well. You can switch back and forth between Source and Disassembly view at any time by clicking on their tabs. The window will synchronize their location.

## Register Windows

Register Windows let you view and modify the hardware registers for the target processor you're debugging.

Registers are grouped by function, for example General Purpose, Timers, SCI port, etc. By default the processors general purpose registers (Program Counter, Stack Pointer, etc.) are displayed first. You can change the group displayed by selecting the group name from the drop down control at the top of the window.



Registers	
General Purpose	
PC	4031
SP	3DFC
IX	0803
IY	0800
D	0000
CCR	0004

The name of the register is on the left and the current contents of that register is on the right. To change a register value, double-click on the current value and type in a new one. When you hit ENTER the new value will be written to the register. The register will then be read back and it's new value will be displayed. If the change was successful it will match what you entered.

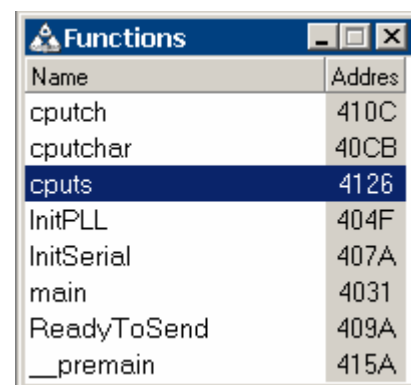
Some bit positions in registers cannot be changed. Consult your processors documentation for more information about these registers.

You can have more than one register windows open at the same time, each displaying different register groups and you don't have to be debugging a project to open them. Just select **Register Window** from the **Debug** menu to open a new one any time you're connected to the board.

## Function Window

The Function Window displays subroutine function names from your source code. Double-clicking on a function in this window will open its source file in the Source Window with the cursor located at the functions source code.

Right-click in the window and choose "Sort by Address" or "Sort by Name" to change the order of the listed functions.



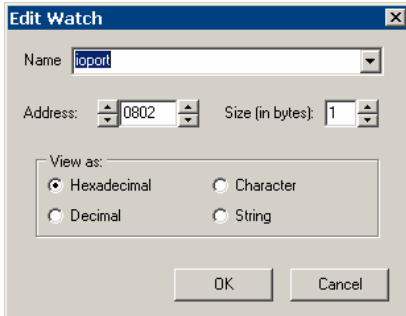
Functions	
Name	Address
cputch	410C
cputchar	40CB
cputs	4126
InitPLL	404F
InitSerial	407A
main	4031
ReadyToSend	409A
__premain	415A

## Watch Window

The Watch Window lets you set keep and eye on, or “watch”, specific variables or memory addresses while debugging.



Right-click in the window to add, edit or remove a watch. A dialog will display letting enter what you want to watch and how you want to display it.



To watch a global variable, use the drop down control at the top to select from the list of global variables you’ve defined.

The Address field will change to the variables address.

To change the number of bytes to watch, which is usually the size of the variable, modify the Size box.

To change how the variable is displayed in the watch window, select from the View as options: Hexadecimal, Character, Decimal or String.

When finished, click OK. Your watch will be added to the watch window.

From the Right-Click pop-up window you can also:

- Remove the selected watch
- Refresh all the watches by reading their contents from the board
- View the watch in the **Data Window**

## Variables Window

The Variables Window lists your programs Global Variables. Double-Clicking on the variable will quickly add that variable to the **Watch Window**.

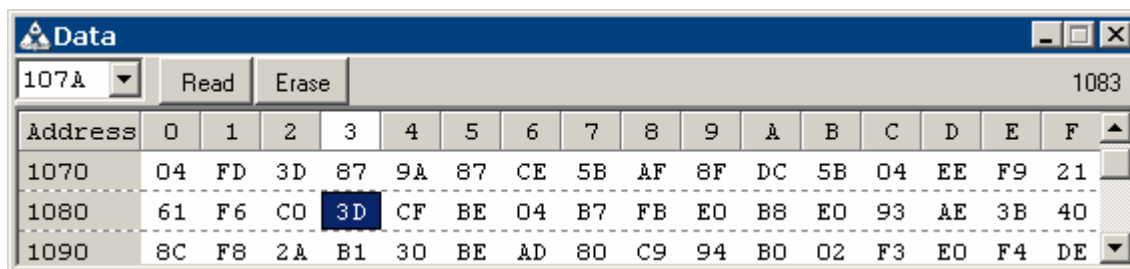


From the Right-Click pop-up window you can also:

- Sort the list by Address or Name
- View the variable in the **Data Window**

## Data Window

The Data Window lets you view and modify memory bytes directly by their address.



Enter a hexadecimal address in the upper left box and press ENTER to change the block of memory displayed in this window. Its drop down control keeps a history of address you enter letting you quickly return to previous locations.

- The **Read** button refreshes the display by reading the boards memory.
- The **Erase** button clears the memory block.


To change a byte of memory left click on it, type in a new value then hit ENTER. The memory location will be read back and it's new value will be displayed.

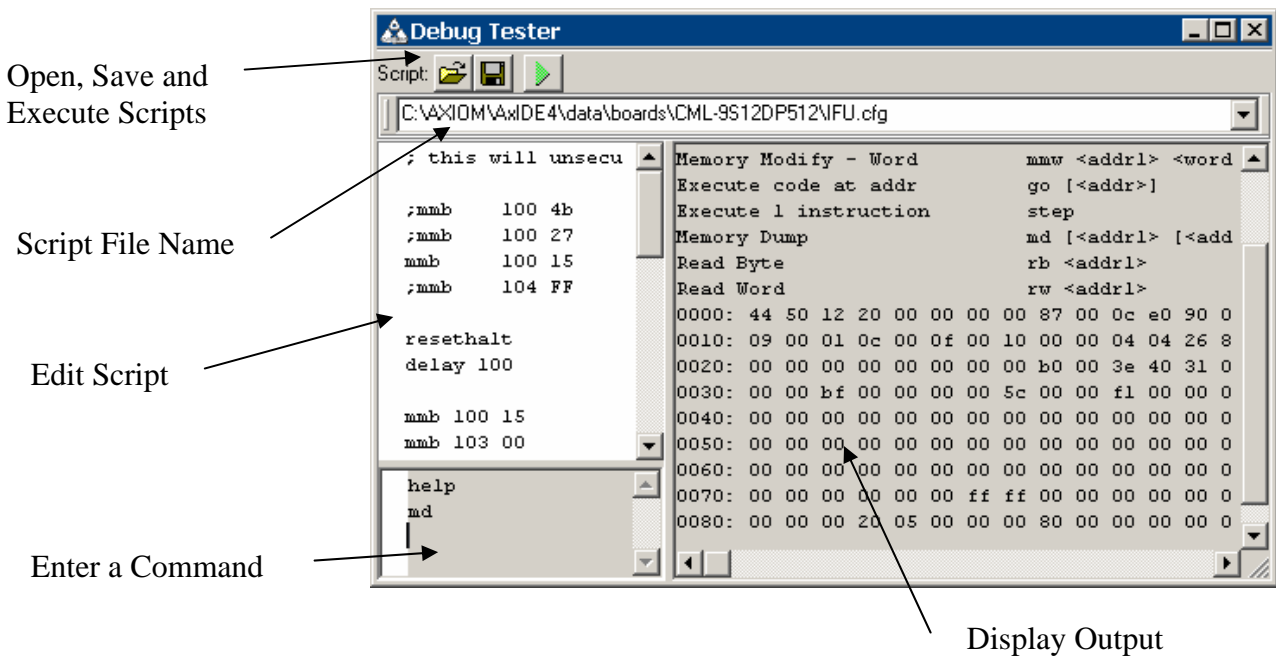
Some memory types cannot be written. In this case the memory will not change or will change to a different value that what you enter. Other things that can effect memory write ability are:

- Board jumper settings (see the board manual)
- Memory configuration (see Choosing a Memory Map)
- Hardware registers set by the Register Window or the executing program


## Debug Tester

The Debug Tester window is a powerful low-level interface to the target. You can type commands and run scripts to view and modify memory and execute code directly on the target thru the BDM interface.

Click the  Tester button on the main toolbar to launch the debug tester window.



You can enter single commands in the lower left command pane. Enter “MD 1000” to view a block of memory at address 1000 hex for example. Enter “help” to see a list of available commands.

You can also create script text files to execute several commands in sequence. Enter the path name for the script file you want to create in the toolbar. Edit the script in the script edit pane. To test the script click the  run button. Each command will execute and the output will be displayed on the right. While the script is executing, pressing the run button again will halt execution.

## Project Options

Project options can be changed by clicking the **Project** button on the main toolbar. Any changes made will only affect the current project. Click the TABS at the top of the options window to see all available options.

### Files

- Project file** Complete path to the current project file. Type a new file name or click the down arrow to see the most recent project files. Click the **Browse** button to open a previously saved project file.
- MCU** Project target processor. This may not be changeable.
- Board** Development board the project is for
- Source Files** Project source code files. Click the Add or Remove buttons to add or remove files from your project. **NOTE:** “include” files should NOT be added or removed here, they are included automatically from the source files.

### Compiler

- GCC Tools Prefix** The path to the target compiler binary files plus the first part of the GCC binary executable names. This will be the same for all GCC compiler tools for a given target. For the HC12 compiler installed to the default directory this should be “c:\cygwin\usr\bin\m6811-elf-”.
- Compiler Options** Options that will effect the output generated by the compiler. See the GCC compiler documentation for more information on what each option means.
- Custom Compiler Options** The GCC compiler has hundreds of command line options. Any options not available above can be entered manually here.
- Custom Linker Options** Additional options are available to the GCC Linker. Most of time you will want to modify the Linker Script instead, but you can also enter options manually here. See the GCC Linker documentation for more information on available options.

### Output

- Memory Map** Here you can customize Memory Maps used by your project. Select a memory map then click Edit to bring it up in the edit window. You should only do this if you’re familiar with modifying the Memory section of GCC linker scripts. See the Linker documentation for more information.

### Debug

- Start Address** The address where the project application will begin execution when debugging. Usually this should be set to “\_start” to have it automatically set to the beginning of the program. You can enter any other symbol or an address directly, for example “0xC000”.
- On program reset..** Check this box to have the debugger stop at a specific location after the debug session is started. The application will execute from the specified Start Address until it reaches this location. The application then halts to await your command to continue.

## Troubleshooting

If you change the GNU Compiler's default install directory during install you'll need to change your AxIDE Project Settings. Select the "Compiler" tab and change the "GCC Tools Prefix" value according to the directory you installed the compiler to (not AxIDE).

If you left the default directory during install, GCC Tools Prefix will be correctly set to:

```
c:\usr\bin\m6811-elf-
```

But if, for example, the GCC Compiler was installed to "z:\cygwin\usr" then you must set the GCC Tools Prefix to:

```
z:\cygwin\usr\bin\m6811-elf-
```

If set incorrectly, the [Build] operation will not find your GCC tools path.

© Axiom Manufacturing  
2813 Industrial Ln.  
Garland, TX 75041

972-926-9303  
[www.axman.com](http://www.axman.com)