Witztronics

Quality Electronic Designs

WTUSBDML Users Manual

Rev 1.4

2/16/2009

1 Introduction to the WTUSBDML

The WTUSBDML is a programming and debugging tool designed to work with the Freescale 9S08, Flexis ColdFire v1.x and 9S12 family of microcontrollers. It allows programming and debugging of a target device via the available BKGD pin. This is made possible by the Background Debug Controller (BDC) and In-Circuit Emulator (ICE) Debug (DBG) modules built into Freescale's 9S08, CFV1 and 9S12 family of devices.

The WTUSBDML is compatible with Freescale's CodeWarrior for HC(S)08 microcontrollers v6.2 (for S08 devices) and CodeWarrior HC(S)12 v4.7 (for S12 devcies) and newer. It was developed to support existing and future derivatives of the S08, S12 and CFV1 family of devices. As new devices are released, every effort will be made to support them. A list of currently supported device families can be found in the appendix.

Features

- Low-cost development tool
- Supports targets with operating voltages from 1.8 to 5.0V
- Supports target bus speeds from 1.0 to 20MHz
- Can supply optional 3.3V or 5.0V (80mA max) power to the MCU target board
- Designed for the USB interface

1.1 Background

The WTUSBDML was developed from the Turbo BDM Light (TBDML) project available from the Freescale 8-bit user forum: http://forums.freescale.com/freescale/board?board.id=TBDML. Because it is open source, the source code can be used and/or modified free of charge. For more technical information on the TBDML protocols, see the WTUSBDML Technical Manual.

2 About the S08, S12 and CFV1 BDC/ICE Debug Module

Freescale's microcontrollers contain a single-wire background debug interface, supporting in-circuit programming of on-chip nonvolatile memory and sophisticated non-intrusive debug capabilities. The BKGD pin on S08, S12 and CFV1 devices provides this single-wire background debug interface to the on-chip BDC and ICE Debug modules. While the interface is single wire, a 6-pin connector (BDM port) is used as the connection to the target. Figure 1 shows the target device 6-pin BDM connection.

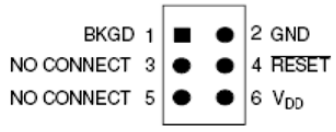

Figure 1. target device 6-pin BDM connection

3 Setting up the WTUSBDML module

The WTUSBDML can supply 3.3V or 5.0V to the target board at a maximum current of 80mA. If the target uses a voltages other than 3.3V or 5.0V, or requires a current greater than 80mA, it must be self powered. In this case, the target voltage is used in the voltage translation circuitry on the WTUSBDML This setting is made by moving the jumper on JP1 to the proper location.
An outline drawing of the WTUSBDML is shown in Figure 2.

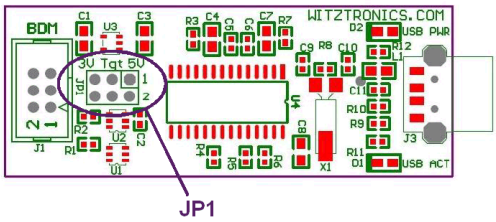

Figure 2.

Table 1 lists the possible target voltage settings using JP1.

Target Voltage	JP1
5.0V	1-2
Provided by Target	3-4
3.3V	5-6

Table 1.

4 Typical WTUSBDML usage

The WTUSBDML typically plugs into a USB port on a host computer running Freescale's CodeWarrior for HC(S)08 microcontrollers v6.2 (for S08 devices) or CodeWarrior HC(S)12 v4.7 (for S12 devcies) and newer. The BDM cable from the WTUSBDML is then connected to a BDM program/debug header on the target PCB.

A typical connection is shown in Figure 3.

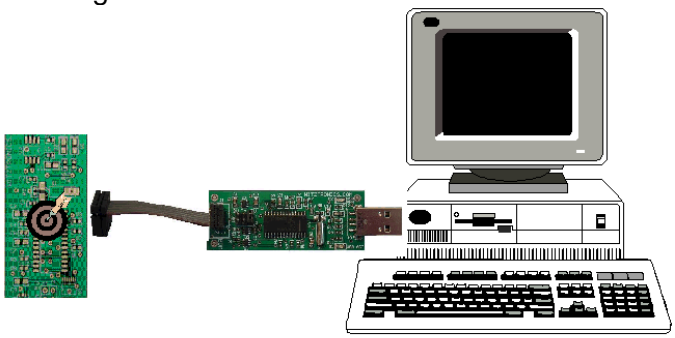

Figure 3. Typical WTUSBDML connection

5 Installing the WTUSBDML BDM DLL and USB Drivers

The following procedure specifies the installing of the WTUSBDML USB hardware drivers under the Windows operating system. This procedure assumes the development PC is running a default installation of Freescale's CodeWarrior for HC(S)08 microcontrollers v6.2 (for S08 devices) or CodeWarrior HC(S)12 v4.7 (for S12 devcies) and newer.

With the WTUSBDML properly configured, the WTUSBDML can be connected to the PC USB port. When the WTUSBDML is connected to the PC for the first time, the Windows operating system recognizes it as a new USB device. This initial connection starts the Windows driver installation procedure. Figure 4 shows the Windows New Hardware Wizard dialog box that opens.

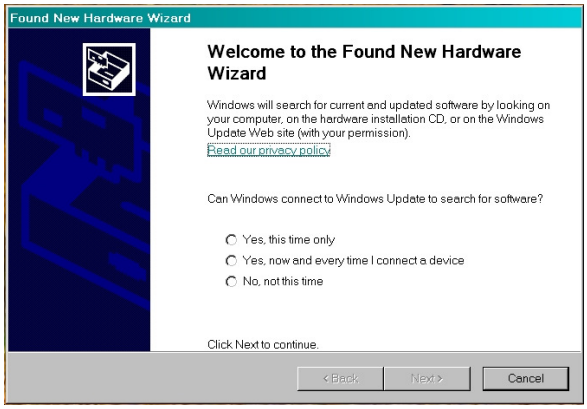

Figure 4. Found New Hardware Wizard dialog box

Select *No, not this time*, then click *Next*.

The Windows New Hardware Wizard installation dialog box will now open as shown in Figure 5. Select the option to *Install the software automatically,* then click *Next*.

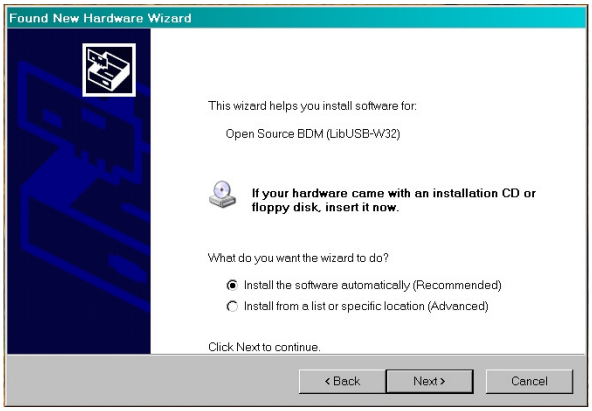

Figure 5. Found New Hardware Wizard installation dialog box

This will initiate the installation of the USB driver and DLL files used by the WTUSBDML, as shown in Figure 6.

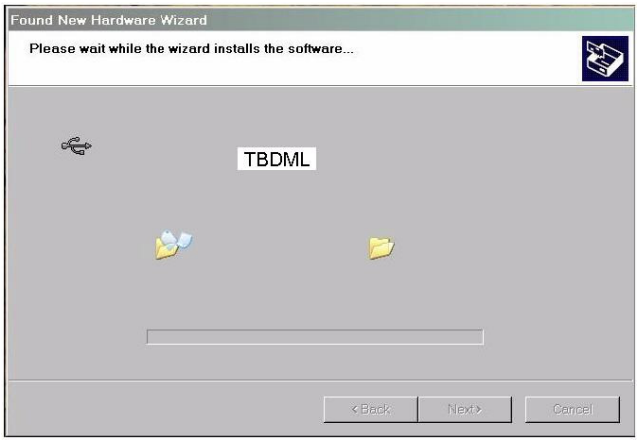

Figure 6. Driver Installation in Progress

Once the installation procedure is completed, the WTUSBDML will be ready to use, as shown in Figure 7.

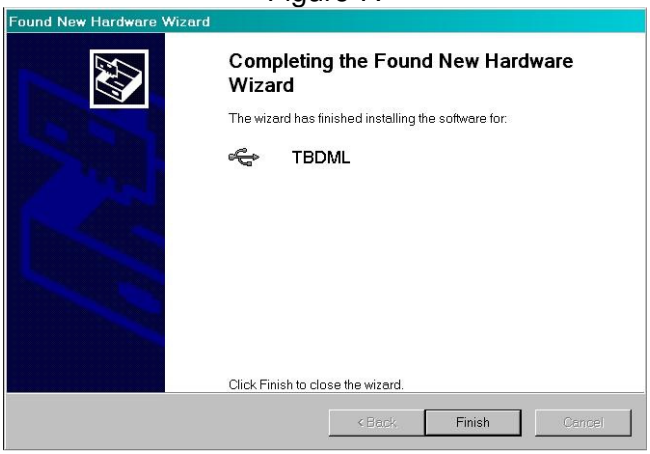

Figure 7. Finished Installation of the WTUSBDML Windows USB Driver

Select the *Finish* button at this point. Because of the *plug and play* nature of USB, a reboot of Windows should not be required.

5.1 Copy the tbdml.dll and opensourcebdm.dll files

Copy the tbdml.dll file from the included CD to the CodeWarrior HC(S)12 v4.7 "gdi" directory (typical location is: C:\Program Files\Freescale\CodeWarrior for HCS12 V4.7\Prog\gdi) for S12 devices and the opensourcebdm.dll file to the HC(S)08 microcontrollers v6.2 "gdi" directory (typical location is: C:\Program Files\Freescale\CodeWarrior for Microcontrollers V6.2\prog\gdi) for the S08 devices. If a file with the same name already exists, rename it to something like "tbdml_old.dll or opensourcebdm_old.dll, in case they are ever needed.

5.2 Selecting the WTUSBDML as the target debugger / programmer

When a project is created, it can be based on a target such as "TBDML", "HCS08 Open Source BDM" or "Cfv1 Open Source BDM" for a S12, S08 or Cfv1 devices respectively. CodeWarrior now provides a TBDML, HCS08 Open Source BDM or Cfv1 Open Source BDM option in this dialog box as appropriate. Figure 8 illustrates the *Connection Option* dialog box.

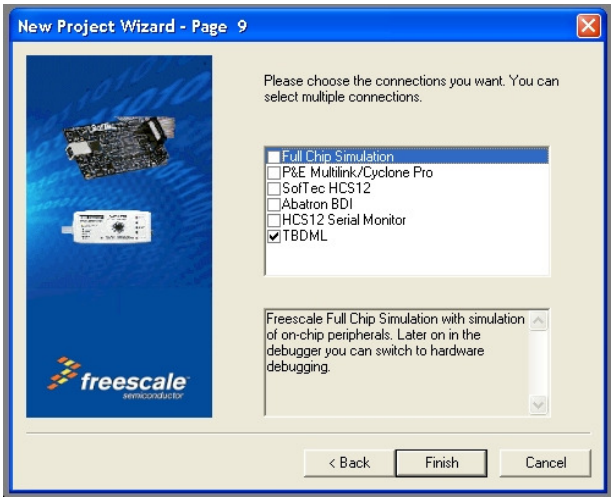

Figure 8. New Project Debugger Interface Connections dialog box

Alternatively, if the project was built with any other connection using the *New Project Debugger Interface Connections* dialog box, it can be easily changed in the CodeWarrior IDE project manager box with the pull down target list, illustrated in Figure 9.

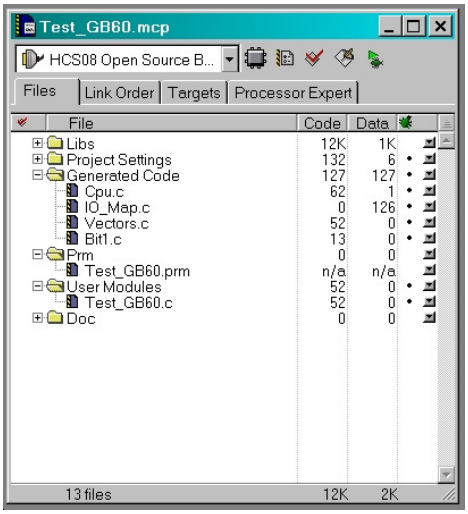

Figure 9. Project Manager

5.3 Opening the Hiwave Debugger

Once the project is created, the project code can be downloaded into the target MCU flash Memory. Figure 10 shows how pressing the debug icon in the CodeWarrior IDE's program manger window will initiate the programming of the target MCU flash memory. This icon executes a command to open the Hiwave debugger program.

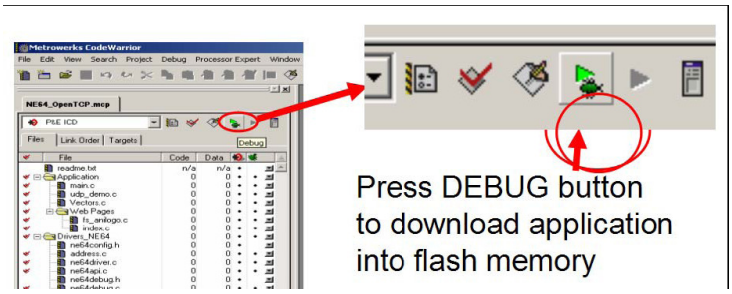

Figure 10. Code Warrior IDE Debug Icon

5.4 Options dialog box

When starting the debugger you will be presented with the dialog at right. After closing this dialogue debugging will proceed as usual. Some options may be disabled depending upon the hardware capabilities of the BDM interface.

The software controlled Vdd options are not supported with the current WTUSBDML hardware. See step 3 for Vdd options.

- **Connection Control menu for S08 and CFV1 devices**
  - Automatically re-connect - This option causes the BDM to continuously update the interface speed to prevent loss of communication in the case with some devices. This option has no effect if the SYNC feature is not available on the target.

- Force BDM Clock Source - Some HCS08 or Coldfire V1 targets provide an alternative BDM clock selection which may be at a lower speed that the default. This can have advantages when connecting to high speed targets. There may be other requirements before the alternative clock will be used.
- Use RESET signal - Many of the HCS08 or Coldfire V1 microcontrollers do not have a dedicated reset signal. This is not a problem as it is possible to reset the target using the BKGD mode commands through the BDM interface. However, the RESET signal on the BDM interface will often still be connected to the Reset signal of the processor. This option allows the BDM to monitor and control the RESET signal when appropriate.

These options are shown in Figure 11.

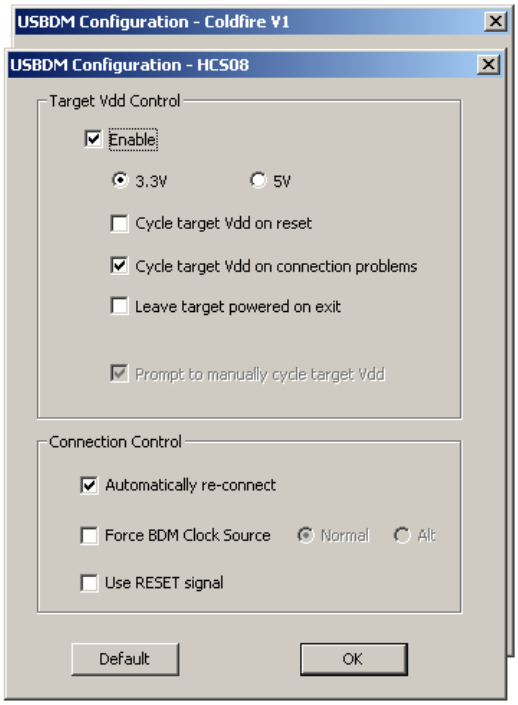

Figure 11 – S08 and CFV1 Connection options

- **Connection Control menu for S12 devices**

  - Automatically re-connect - This option causes the BDM to continuously update the interface speed to prevent loss of communication in this case. This option has no effect if the SYNC feature is not available on the target.
  - Guess speed if no SYNC - Early HCS12s do not support the SYNC feature which allows the BDM interface speed to be determined. Selecting this option will cause the BDM to attempt to find the communication speed by trial and error. This can take quite a while and may be unreliable. Alternatively, set the target speed in the debugger TBDML menu.
  - Force BDM Clock Source - Some HC12 targets provide an alternative BDM clock selection which may be at a lower speed that the default. This can have advantages when connecting to high speed targets. There may be other requirements before the alternative clock will be used. Note: it appears that HC12 use the Alt source by default so the Normal option may be of more interest.

  These options are shown in Figure 12.

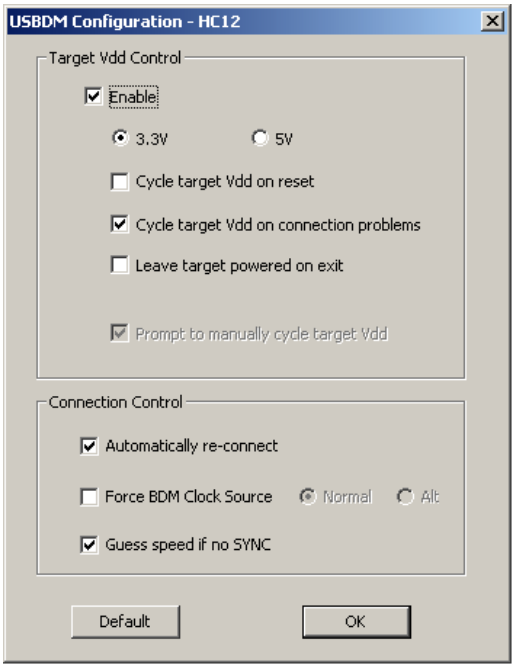

Figure 12 – HCS12 Connection options

6 Hiwave Debugger Options with the WTUSBDML

The following options are found in the WTUSBDML menu:

- Show status dialog box
- Reset to normal command option
- Select derivative option
- Detection and indication of target frequency changes
- Auto derivative selection

Debug and program operations used for the WTUSBDML, with the Hiwave debugger, are similar to other higher cost debug interfaces. Figure 13 shows the Hiwave program opened and configured for the WTUSBDML debugger interface, shown by the Hiwave menu bar with menu entry call Open Source BDM.

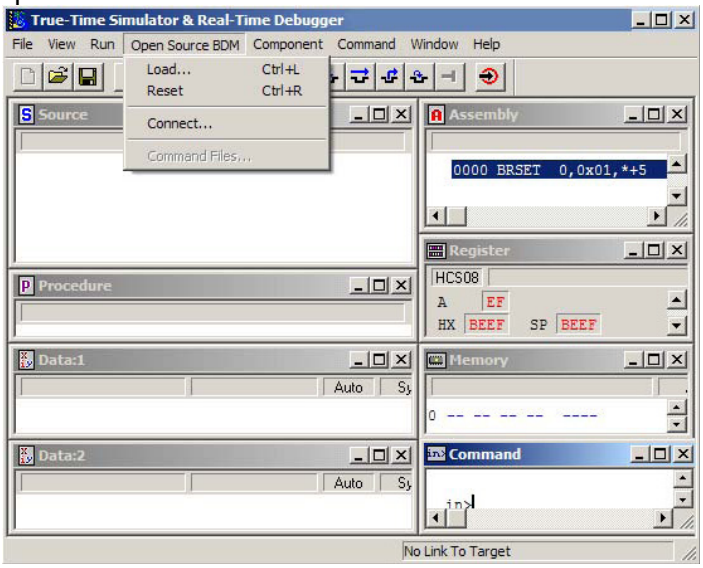

Figure 13. Hiwave program opened and configured

7 Flash Programming with the Hiwave Debugger

This sections provides detailed step-by-step instructions for flash programming with the Hiwave debugger. From the Open Source BDM menu, both the "Flash" and "Load" command can be used to flash program the target.

7.1 Using the Load Command

- Select Open Source BDM > Load... from the menu
- When the load command is executed, the "Load Executable File" dialog box opens as shown in Figure 14.
- Navigate to the file that will be used to program the part and select it
- Before pressing the "Open Button," the "Automatically erase and program into FLASH and EEPROM" checkbox must be checked

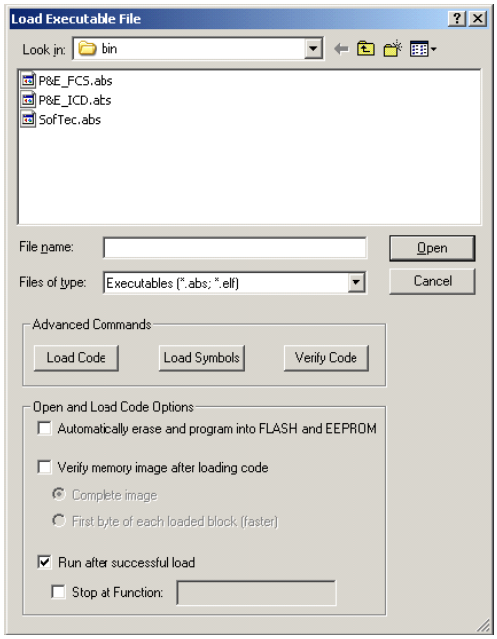

Figure 14. load executable file dialog box

To make the "Automatically erase and program into FLASH and EEPROM" option the default setting for a project, you can configure the debugger by following the steps below:

- Select File > Configuration from the menu
- The "Preference" dialog box opens as shown in Figure 15
- Select the "Load" Tab
- Check the "Automatically erase and program into FLASH and EEPROM" checkbox
- Close "Preference" dialog box by pressing the "OK" button
- Select File > Save Configuration from the menu
- Next time, auto erase and flash functions will be performed by default

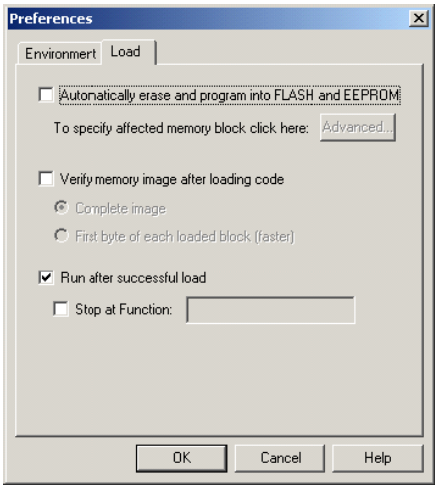

Figure 15. The preferences Dialog Box

7.2 Using the Flash Command

- Select Open Source BDM > Flash... from the menu
- The "Non Volatile Memory Control" dialog box opens as shown in Figure 16.

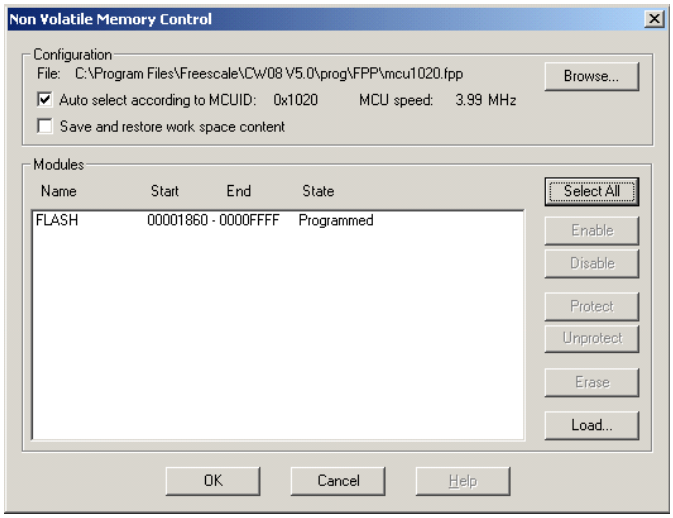

Figure 16. Non Volatile Memory Control Dialog Box

- Press the "Select All" button
- Press the "Erase" button
- Press the "Load..." button
- When the load button is pressed, the "Load Executable File" dialog box opens as shown in Figure 17.

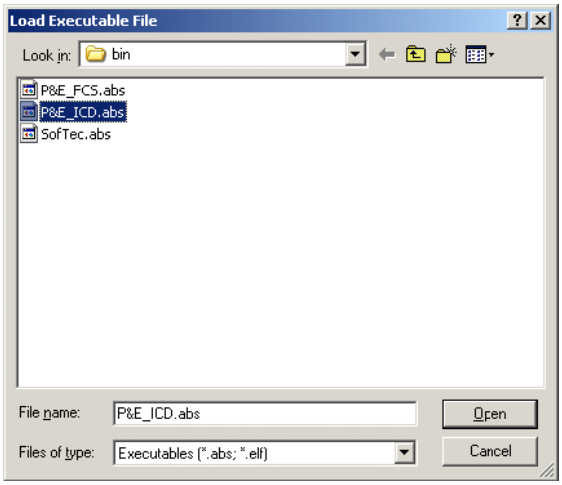

Figure 17. Load Executable Dialog Box

- The user must navigate to the file that will be used to program the part and select it
- Pressing the "Open" button programs the part
- To close the "Non Volatile Memory Control" dialog box, press the "Unselect All"
- Then press the "OK" button"

8 Typical WTUSBDML operation

The WTUSBDML only needs to be configured once. With the “Automatically erase and program into FLASH and EEPROM" option selected as the default mode, launching the Hiwave debugger will automatically perform the following:

- Connect to the WTUSBDML programmer/debugger
- Load the target debug information
- Load the target programming information
- Program the target device
- Run the target device

9 Appendix

9.1 Supported Devices

9.1.1 HCS08 supported devices:

- MC9S08AW derivatives
- MC9S08EN derivatives
- MC9S08GB/GT derivatives
- MC9S08QA/QD/QE/QG/QT derivatives
- MC9S08RC/RD/RE/RG derivatives
- MC9S08JR derivatives
- MC9S08EL/SL derivatives
- MC9S08SH/SG derivatives
- MC9S08DE/DN/DV/DZ derivatives
- MC9S08LC derivatives
- MC1321 derivatives
- MPXY8300

9.1.2 ColdFire Flexis supported devices:

- MC9S08AC derivatives
- MCF51AC derivatives
- MC9S08QE derivatives
- MCF51QE derivatives
- MC9S08JM derivatives

9.1.3 MCF51JM derivatives:

- HC912 devices
- HC9S12 devices
- HC9S12X devices
- HC9S12XE devices