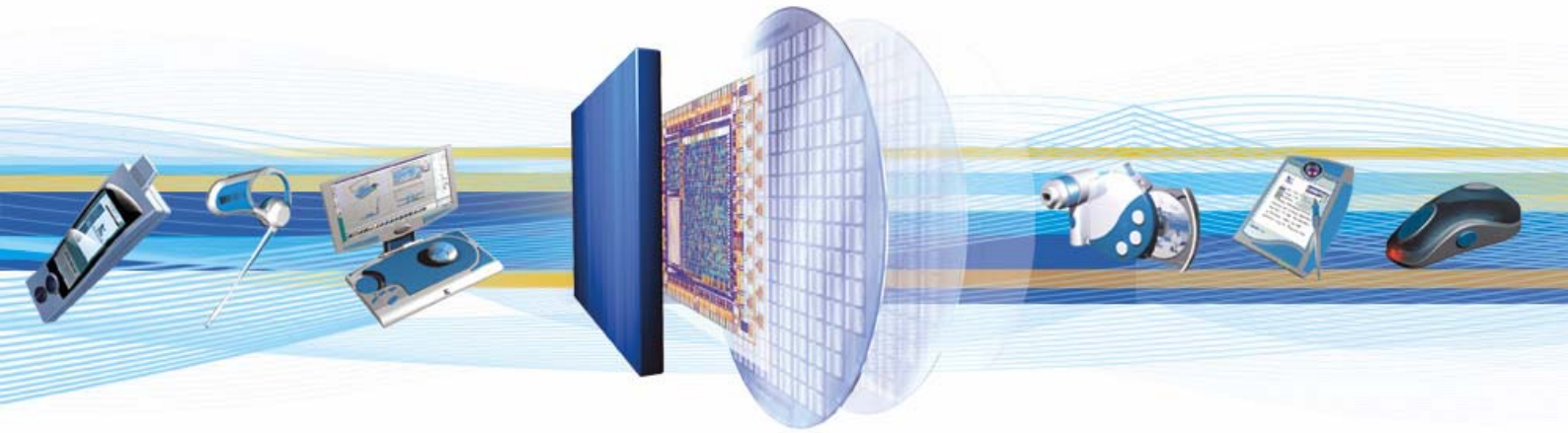
BlueLab xIDE

user guide

May 2005



CSR

Churchill House
Cambridge Business Park
Cowley Road
Cambridge CB4 0WZ
United Kingdom

Registered in England 3665875

Tel: +44 (0)1223 692000
Fax: +44 (0)1223 692001
www.csr.com

Contents

1 Introduction

This document provides a brief introduction to the Integrated Development Environment **xIDE** supplied with the BlueLab™ SDK (Software Development Kit).

The document is intended to provide developers with the information required to begin using xIDE to develop applications for BlueCore chips.

Note: Since xIDE provides a familiar environment with the tools and utilities required to write, build, run and debug code it is not intended to detail all these features. This document will concentrate on BlueCore-specific aspects of developing applications using xIDE.

1.1 General

xIDE allows software engineers to independently develop applications to run on CSR BlueCore chips.

It supports the development and debugging of both Virtual Machine (VM) applications written in ANSI C for all BlueCore variants and Digital Signal Processor (DSP) code written in assembler code for BlueCore Multimedia chips.

Code is written in the text editor and when complete, built and compiled along with the BlueCore firmware supplied as part of BlueLab.

When compiled the resultant machine code can be downloaded to, and run on, a BlueCore hardware development platform such as Casira or the multimedia development board available separately from CSR.

The code can then be debugged on-chip using the facilities in xIDE.

Applications can be developed from the reference application code provided, using the example code and library functions supplied, to adapt and add functionality.

The application source code provided implements various Bluetooth Profiles. These Profiles can be used as part of the user's own applications.

Using reference applications as a starting point for development greatly reduces the effort required to produce working Bluetooth applications that correctly implement the required Bluetooth Profile(s).

2 Installation

This chapter provides guidance on the installation of xIDE as supplied on the BlueLab CD-ROM.

2.1 Prerequisites

BlueLab xIDE should be installed on a PC with a Line Printer Terminal (LPT) port running Windows 2000 or Windows XP.

CSR recommend that 100Mbytes of free disk space is available.

> **Note:** A typical BlueLab installation requires 50Mbytes and each application built will need approximately 10Mbytes of additional space.

A minimum of Windows Power User privileges is required to install the software correctly.

> **Note:** If you are unsure of your current level of privileges, please contact your system administrator.

New BlueLab installations can coexist with previous releases provided they are installed in different directories. The use of a convention such as C:\BlueLabversion (eg `C:\Bluelab3_x`) to create the program folder will avoid conflict between versions.

> **Note:** Spaces in folder names of the directory path are not supported ie you should not try to install the software in a directory which itself has spaces in its name or is contained within a folder that has spaces in its name eg XIDE cannot be successfully installed in the `Program Files` directory.

2.2 Installation procedure

CSR recommend that any applications running on the PC are closed before installing the BlueLab software.

1. Insert the Bluelab CD-ROM into the computer's CD drive.
2. Right-click on the Windows Start menu and explore the CD-ROM.

   The CD-ROM contains PDFs of the Bluelab documentation and the BlueLab executables.
3. Double-click on `BlueLab-3.x.exe` file to launch the Setup wizard, which will guide you through the rest of the installation process.
4. Follow the on-screen instructions, clicking **Next** to continue.

   For a first time installation, CSR recommend that the default settings are accepted.
5. When the VM and DSP libraries have been built, the PC the Setup Wizard will display the final setup screen.
6. Click **Finish** to complete the installation.

   If the default option to install the Serial Peripheral Interface (SPI) device driver was accepted, the PC must be restarted to complete the installation.

2.2.1 MP3 Support

The Digital Signal Processor (DSP) libraries required to build the MP3 decoder must be obtained and installed separately (please contact the support channel for details).

They contain MP3 technology which incorporates intellectual property owned by Thomson and/or Fraunhofer Gesellschaft.

Supply of this product does not convey a license under the relevant intellectual property of Thomson and/or Fraunhofer Gesellschaft nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit http://www.mp3licensing.com.

2.2.2 Testing the completed BlueLab Installation

Before you begin:

Connect a suitable hardware development platform (eg Casira module) with an appropriate BlueCore chip to your PC using a Serial Peripheral Interface (SPI) cable.

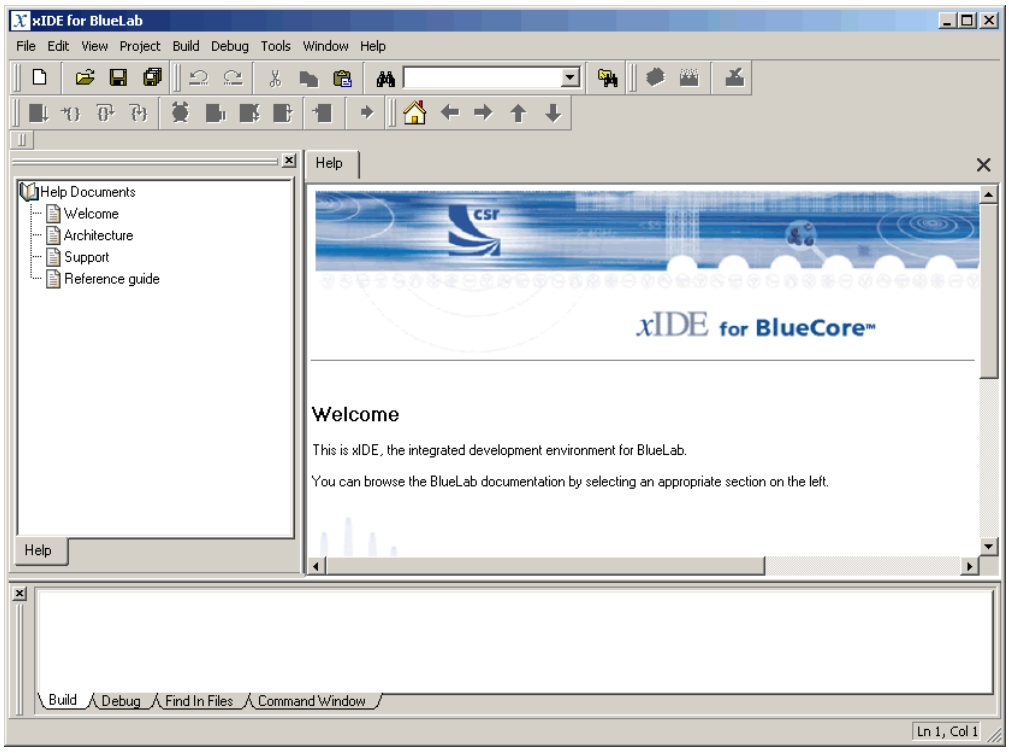
Note: The documentation accompanying the Hardware development platform gives advice on connecting the unit to your PC.

IMPORTANT: CSR strongly recommend that a 'golden' image is created to preserve the original chip settings. The required backup procedure is detailed in Chapter 3.

Testing the installation

Launch xIDE by double-clicking on the shortcut icon on your desktop, the icon in the Quick Launch bar or from the Windows' Start Menu.

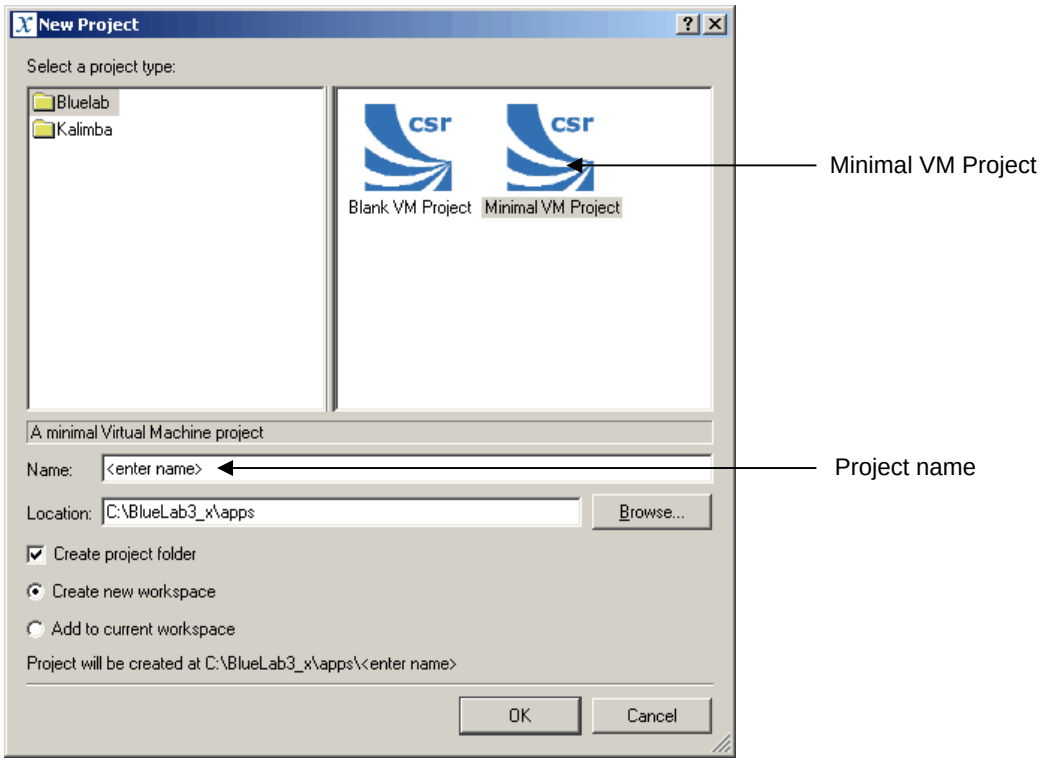
The xIDE application window opens:



To confirm the installation was successful and the application is working correctly run the simple 'Hello world' program supplied, to do this:

1. Select **New** from the **Project** menu.

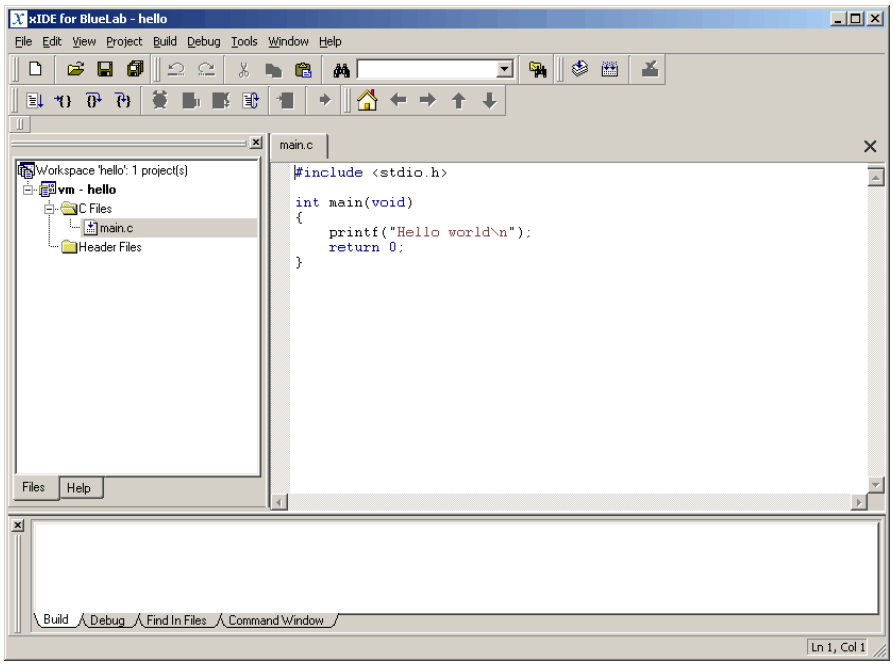
The New Project dialog displays:



2. Select the **Minimal VM Project** and give the project a name eg hello.
3. Click **OK**.

The project is loaded into xIDE.

4. Click on the **C Files** folder and select `main.c` to display the code in the Text Editor workspace:

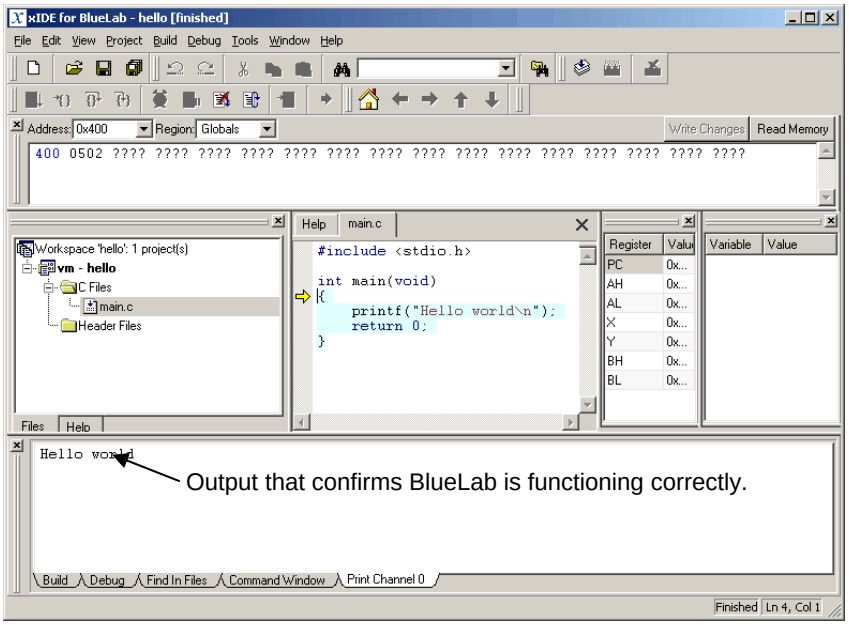


5. Select **Build** from the **Build** menu (or press the **F7** key).

6. Select **Run** from the **Debug** menu (or press the **F5** key).

   The program is downloaded to the BlueCore chip, when this process is complete.

7. Click on the **Print Channel 0** tab, to view the output:



The output ‘Hello world’ confirms that the BlueLab software has been installed and is working correctly.

3 Backup and Restore procedure

You can backup and restore an image of the firmware on the BlueCore chip at any time.

CSR strongly recommend that a backup is performed to save the original firmware configuration prior to downloading any development code.

The backup and restore procedures are carried out using the Blue Flash application supplied as part of the BlueLab toolset.

To run the Blue Flash application:

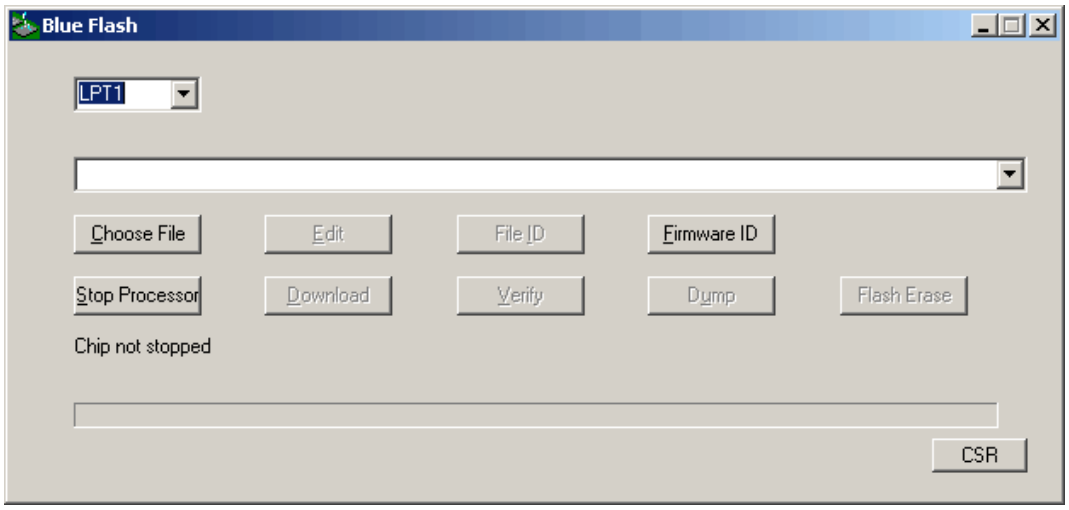
1. Locate `BlueFlash.exe` using Windows explorer.

   This file is located in the Bluelab installation directory.

   eg `C:\BlueLab\tools\bin\BlueFlash.exe`

2. Double click on `BlueFlash.exe` to run the application.

   The Blue Flash application window appears:



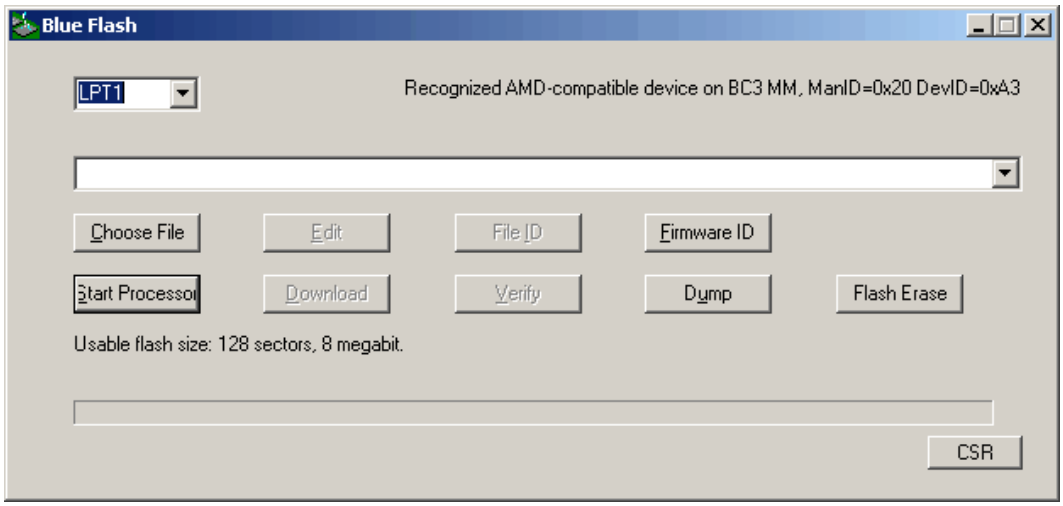
Backup and Restore procedures are described in detail in sections 3.1 and 3.2.

3.1 Backup procedure

To backup an image of the current firmware and configuration settings using Blue Flash:

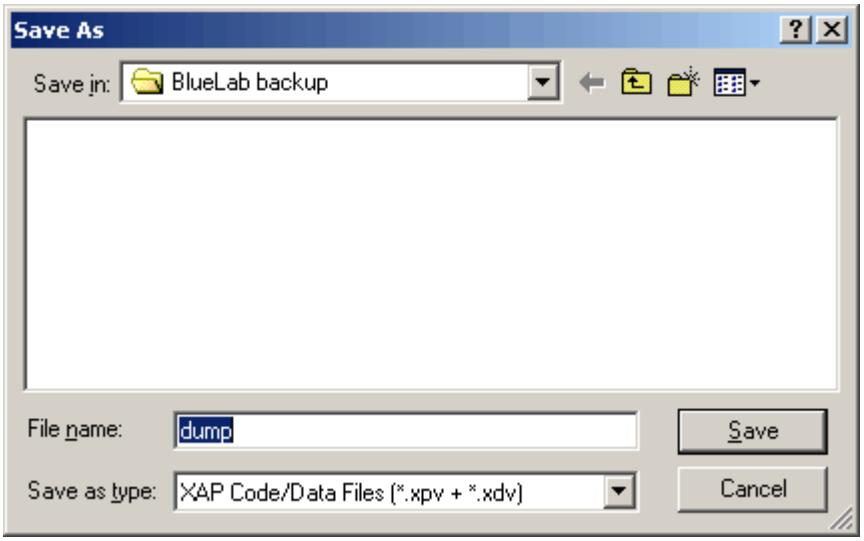
1. Click **Stop Processor**.

   The Processor is stopped and the **Dump** option becomes available:



2. Click **Dump**.

   A Save As dialog appears:



3. Browse to a location in which the backup is to be saved and enter a file name for the backup.
4. Click **Save**.

   Blue Flash will create and save two files (a `.xpv` and a `.xdv` file) to the selected location.

   Note: This may take a few minutes, a progress bar displays the progress.

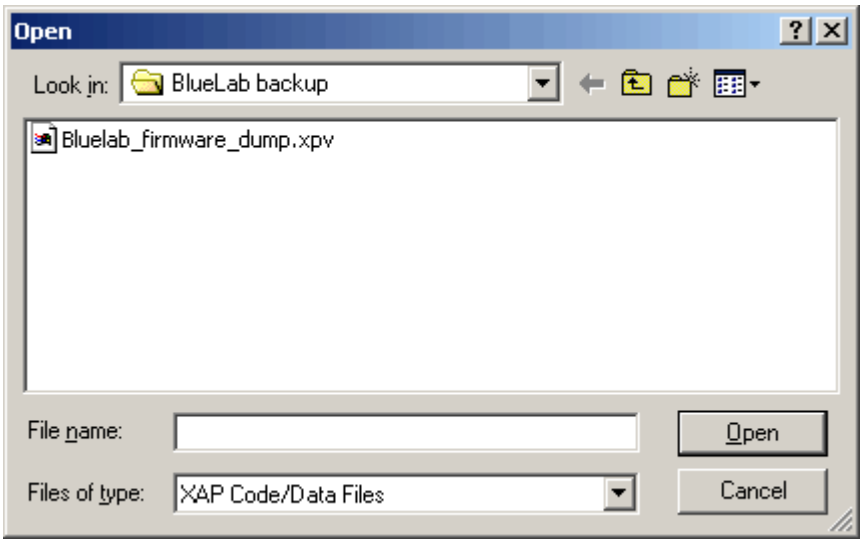
5. Click **Start Processor** to restart the chip.

3.2 Restore procedure

To restore a backup image using Blue Flash:

1. Click **Stop Processor**.
2. Click **Choose File**.

   An Open dialog appears:

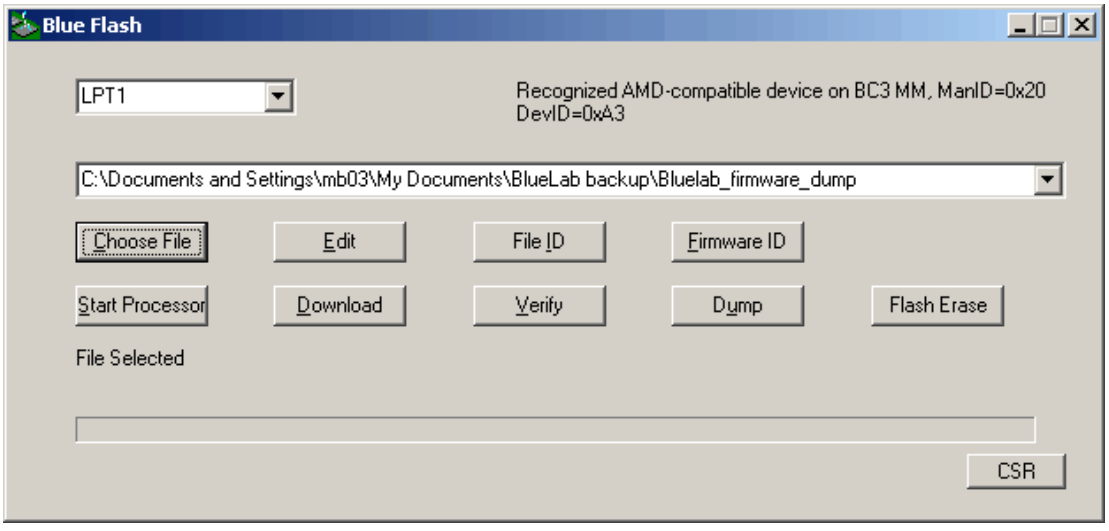


3. Browse to the backup file location and select the required file.

   Note: The dialog will only display the `.xpv` file(s) saved at the selected location. Blue Flash will automatically include the `.xdv` file when downloading the image to the chip.

4. Click **Open**.

   The path to the file is added to the Blue Flash dialog and the Download option becomes available:



5. Click **Download**.

   Blue Flash downloads the backup image to the BlueCore chip. This may take a few minutes.

6. When the download has completed, click **Start Processor** to restart the chip.

4 Working with xIDE

In most circumstances, it is envisioned that developers will make use of the reference applications provided as the basis for developing their own applications.

The reference applications provide basic functionality and conform to the relevant Bluetooth Profile(s) being implemented.

Adopting this approach greatly reduces the effort required to develop a final product application and allows software engineers to concentrate on developing the additional functionality and Man Machine Interface features required for their particular product.

This chapter describes the procedure for loading a reference application as a project in xIDE and running the code on a hardware development platform. Specific details will vary slightly depending on the application and hardware platform being used, further information is provided in the relevant product documentation.

Note: Guidance on the use of Reference applications and examples is provided in readme files within the subfolders in the apps folder: `C:\BlueLab\apps\...` where `C:\BlueLab` is the install directory.

4.1 Building a supplied application project in xIDE

To open a project workspace for a supplied application:

1. Select **Open Workspace** in the xIDE **Project** menu.

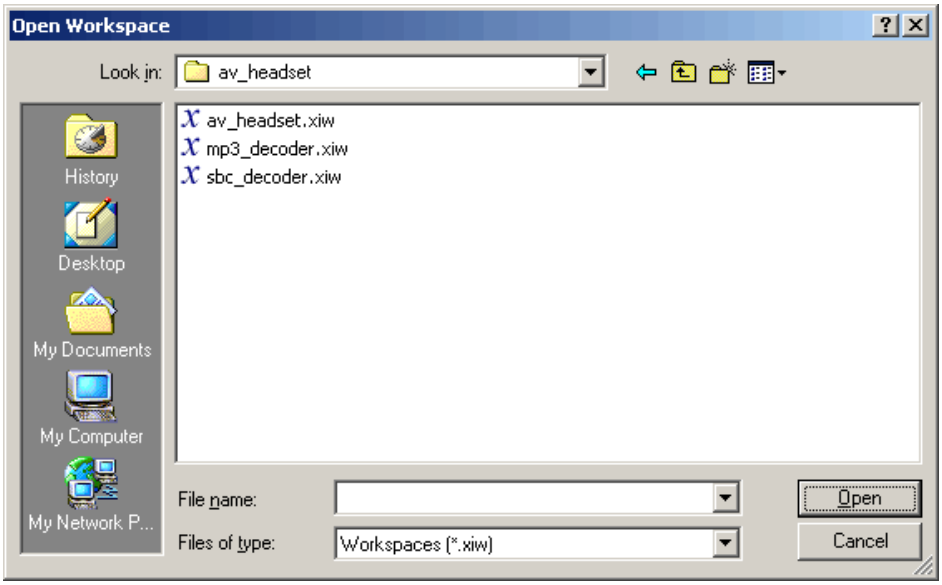
   An Open workspace dialog appears.

2. Browse to the apps folder that can be found in the Bluelab program directory's apps folder.

   eg `C:\BlueLab\apps`

3. Open the required application folder.

   Depending on the application chosen, 1or more `.xiw` project files will be displayed, see the example below:



Note: Where two or more files are displayed, the additional files are Assembler code that the application requires to be run on the on-chip DSP (Digital Signal Processor) of BlueCore Multimedia chips.

If DSP elements are required as part of the application these projects should be built in xIDE before attempting to build and finally run the application code.

4.1.1 xIDE Build and Run procedure

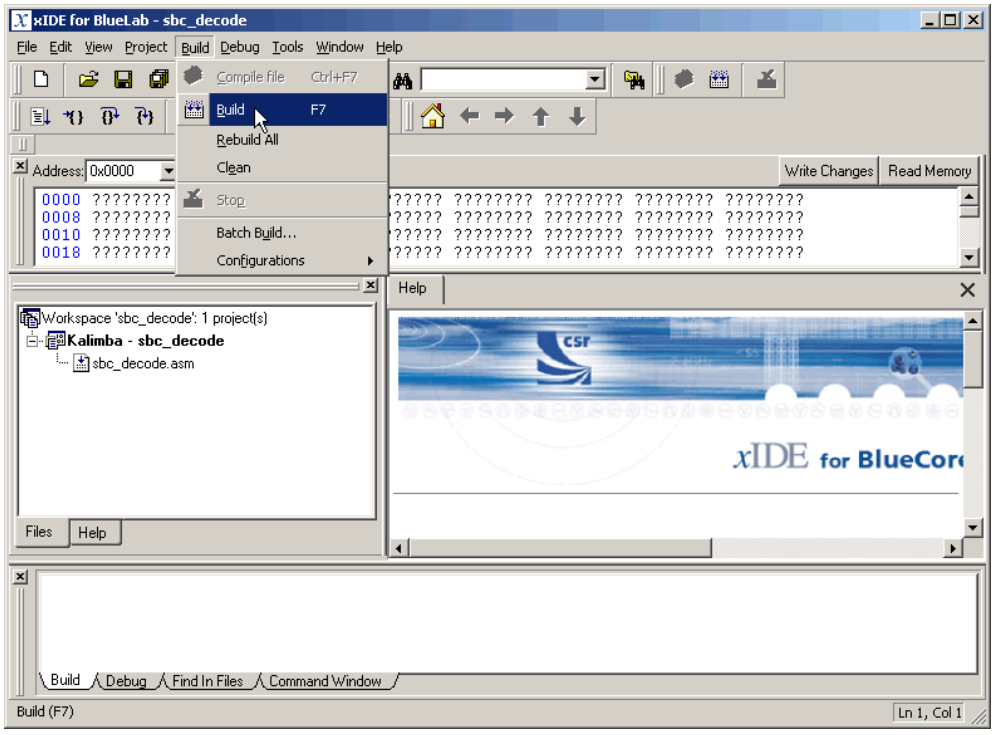
The following procedure should be followed to build all the .xiw files making up the application before downloading machine code to the BlueCore chip using the Run facility.

Build procedure

1. Select the required `.xiw` file in the Open Workspace dialog (**Project /Open Workspace**)
2. Click **Open**.

   The file will be loaded into xIDE.
3. Select **Build** from the xIDE **Build** menu or press the **F7** key:

   Note: If the development hardware has been used for previous development work it is recommended that the factory settings are restored on the chip prior to downloading the new application code. The procedure for this is described in question 4 of the FAQs, Chapter 5.



   xIDE will complete the build process for the loaded file.
4. If any other DSP `.xiw` files that support the application code (eg for mp3 support) are present, build these before continuing.
5. Build the Virtual Machine application code in a similar manner.

Run procedure

To complete the compilation of the source code and to download the machine code produced to the BlueCore chip:

With the main application file loaded in xIDE.

1. Select **Run** from the **Debug** menu or press the **F5** key.

The application should now be running on the BlueCore chip, please refer to the relevant application documentation for further details.

Note: If DSP code required by the application was not build before building and running the application code ,an error message will be displayed stating that the Debugger cannot be started and the code will fail to download. If this occurs build the necessary DSP code and rerun the application code.

4.2 Developing customised applications

When the application has been downloaded and is working correctly, developers can begin to customise the source code and add features to meet the specific requirements of the final product.

Note: If a backup image has not already been saved it may be advisable to create a backup image of the BlueCore chip before making further changes.

In order to work efficiently when developing an application it is important to become familiar with the library structure and functions provided within BlueLab. These are detailed in the library support documentation.

The example application code supplied is also a useful resource and can be found in `C:\BlueLab\apps`.

4.2.1 Amending project properties

When calling library functions the host Libraries must be listed in the Project Properties.

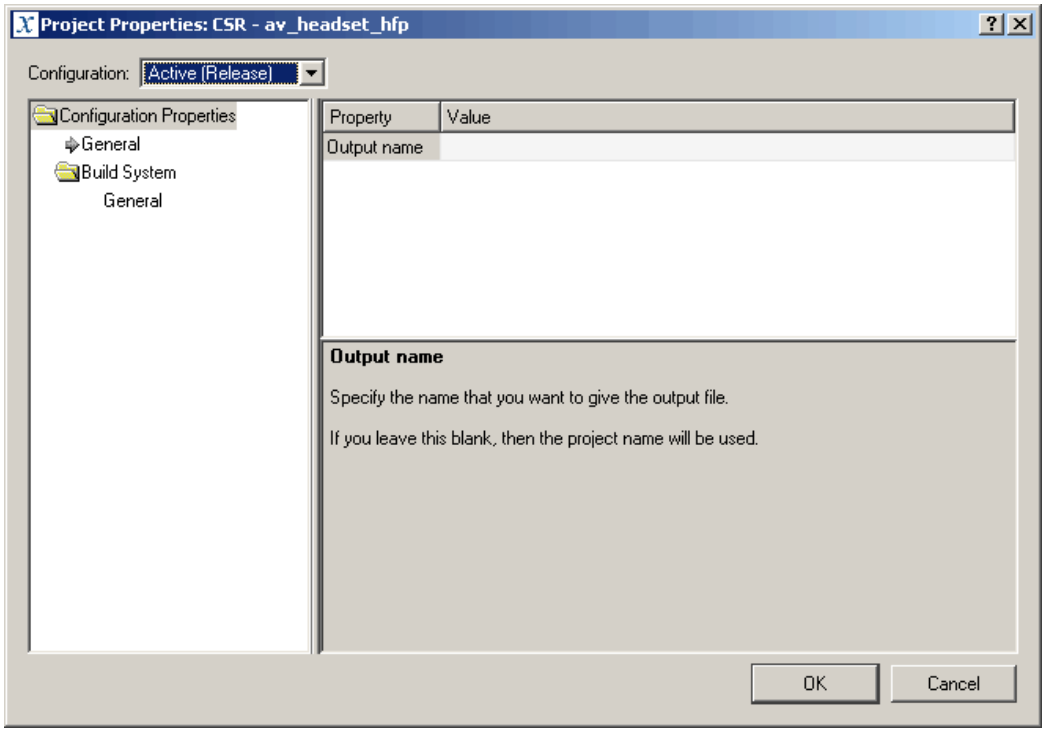
Note: Libraries used by the reference application code along with the other project properties for the particular application are specified when the `.xiw` file is loaded to xIDE.

When function calls to other libraries are added to the application code the appropriate libraries must be added to the list of libraries specified in the Project Properties.

To amend the project properties:

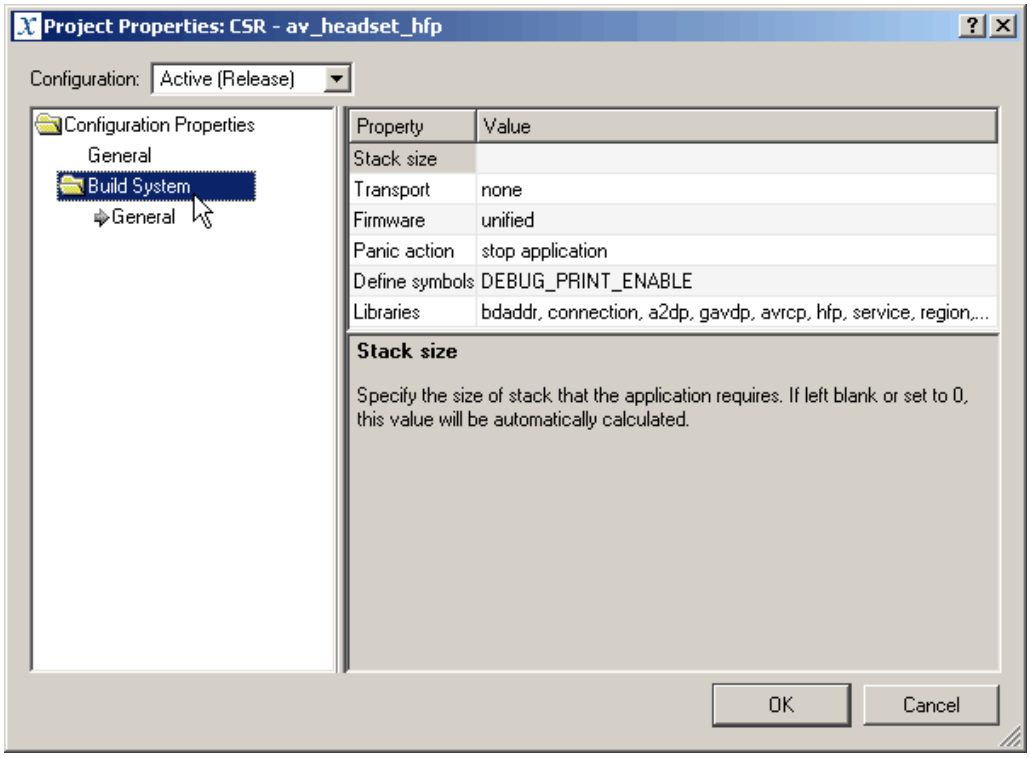
1. Select **Properties** from the **Project** menu.

   The Project Properties dialog appears:



2. Click on the **Build System** folder.

   The Build system properties are displayed:



3. Click on a row to activate the **Value** field for the **Property** you wish to amend.

   The text below the list of properties provides tips relevant to the selected Property.

4. When the required properties have been amended, click **OK** to set the properties for the project.

4.3 Debugging in xIDE

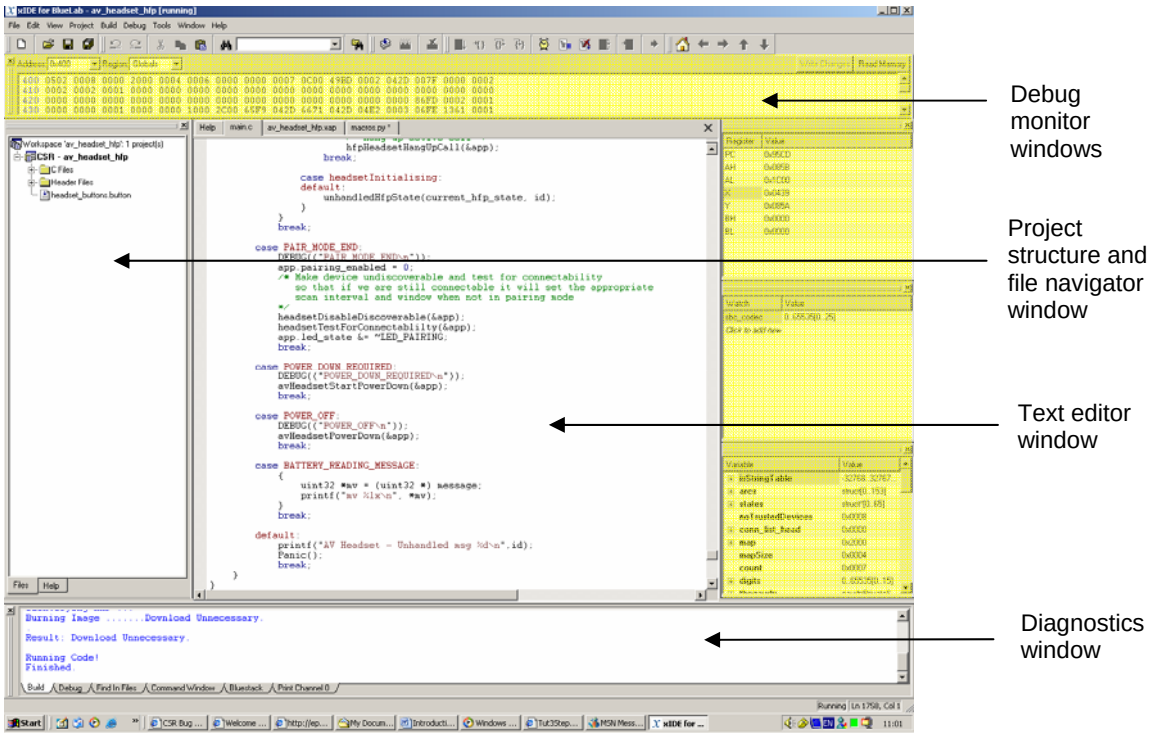
xIDE supports the debugging of the program code running on the Virtual Machine of the BlueCore chip and the Assembler code running on the on-chip DSP (Digital Signal Processor) of BlueCore Multimedia chips.

The application is run on-chip, thus ensuring the debug environment matches the final execution environment of the product as closely as is possible.

xIDE provides a familiar debugging toolset that includes facilities required to efficiently debug programs running on a BlueCore chip.

Whilst many of the facilities provided in xIDE are typical debugging tools, a few are more specific to an integrated implementation such as BlueCore, see section 4.3.1.

4.3.1 Brief Overview of Debug facilities



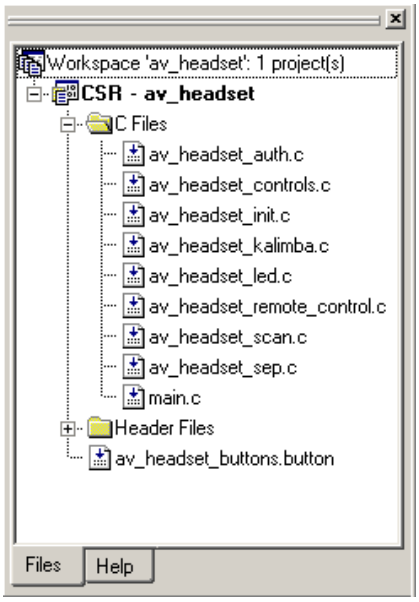
The xIDE debug view consists of four basic work areas:

- File navigation window
- Text editor window
- Debug monitor windows (highlighted in yellow)
- Diagnostics windows

A brief description of each is given below.

File Navigation window

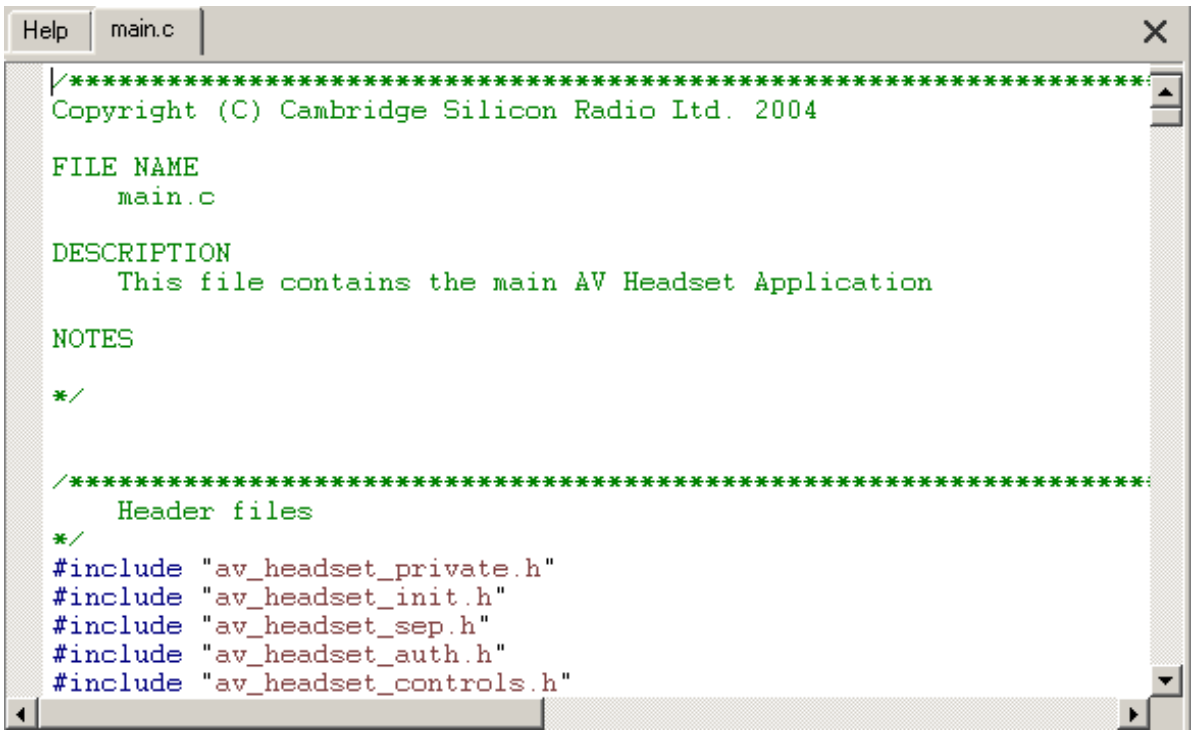
This area displays an explorer like view of the project workspace currently loaded in xIDE:



The file structure can be navigated and files opened in the text editor by double clicking on a file.

A right-click shortcut menu can be opened for items listed in the File Navigation menu, the menu options will depend on the item selected.

Text Editor window



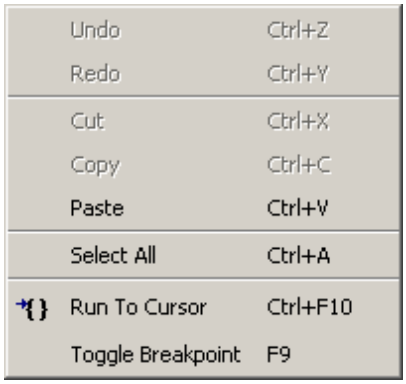
This area displays open project files and allows:

- Text editing
- Break points to be set

Tabs allow navigation between multiple files opened in the text editor.

Note: An * displayed after the file name on a file tab (eg main.c*) indicates that the file has been amended locally and has not been saved.

A right click in the text editor window displays a shortcut menu:



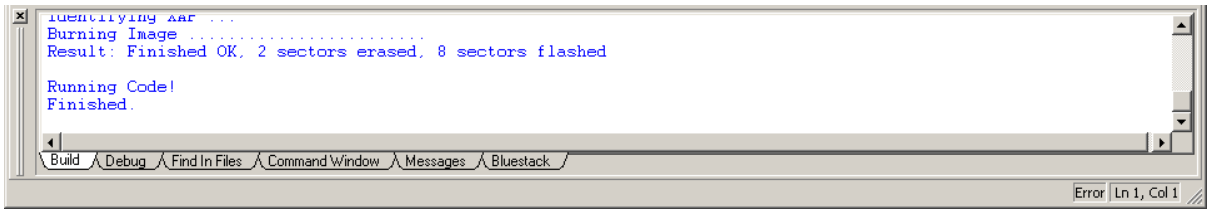
Debug monitor windows

Developers can select to view various windows that monitor the BlueCore chip state:

- **Memory:** displays current values of selected memory addresses
- **Register:** displays current values of registers
- **Variables:** displays current values of program variables
- **Watch:** allows the user to view the current value of specific program variables

The views available can be toggled on and off from the menu list found in **View /Debug Windows**.

Diagnostics windows



This area displays various tabbed windows that display useful information when building and debugging code:

- **Build**

  The Build tab displays progress information detailing the build process and status.

- **Debug**

  Displays progress information showing the debug process and status.

- **Find in Files**

  Displays the results of the **Find in** files facility accessed from the **Edit** menu or toolbar.

- **Command Window**

  This window can be used to evoke python scripts to extend xIDE.

  Note: Most developers need not concern themselves with this facility.

- **Messages**

  Displays a time stamped list of messages passed between tasks in the VM.

- **Bluestack**

  This tab displays the messages to and from the stack relating to the lower level radio interface.

  This can be particularly useful in confirming that the code is resulting in the expected event messages being sent and received by the radio.

- **Print Channel 0**

  This tab displays any print output generated by the program running on the chip.

A right-click shortcut menu can be opened in each of the diagnostic windows, the menu options will depend on the active tab.

Note: The Messages, Bluestack and Print Channel tabs only appear when xIDE has information to display.

5 Frequently Asked Questions (FAQs)

1. *Can I customise the xIDE environment?*

   Yes. The **Options** menu item in the **Tools** menu allows you to select various options affecting the appearance and behaviour of xIDE.

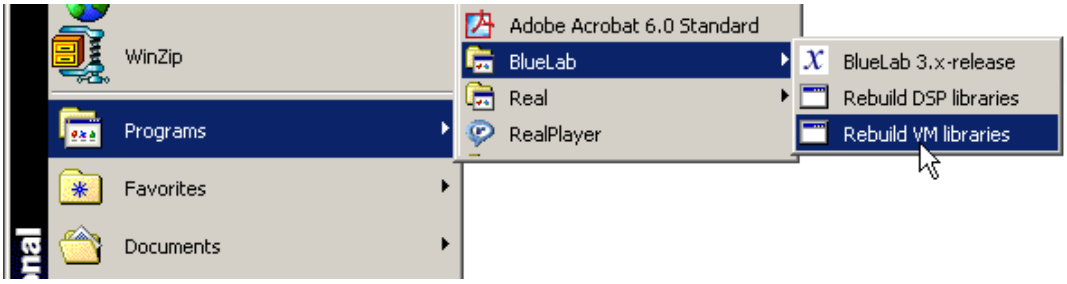
   The **View** menu offers a number of layout options allowing you to toggle the display to show your preferred debug windows and menu items.

   Windows can be reorganised by dragging and dropping within the main xIDE window or on the desktop to create separate displays.

2. *Can I rebuild the VM and DSP libraries?*

   Yes, you can rebuild the VM and DSP libraries from the Windows Start menu.

   Go to **Start/Programs/BlueLab** and select the Libraries you wish to rebuild:



   Note: you may need to build the libraries if the default option to build libraries was unchecked during the install process.

   Or

   If code has been added to or amended by the developer. However, CSR do not recommend making alterations to the supplied libraries.

3. *My application fails to initialise the Connection Library, why?*

   It may be that `PSKEY_ON_CHIP_HCI_CLIENT` is incorrectly set. This key enables the upper Bluetooth layers on the device and should be set to 1.

   Use the PSTool application described in question 4 to check that the `PSKEY_ON_CHIP_HCI_CLIENT` is set to 0001.

   Note: the friendly name for this PSKEY is **HCI traffic routed internally**.

   xIDE will normally set the value of this key automatically based on whether the application makes use of Bluestack

4. *How can I restore the chip's factory settings?*

   If you created a backup of the chip in its original state, you can use the procedure described in Chapter 3 to restore the chip.

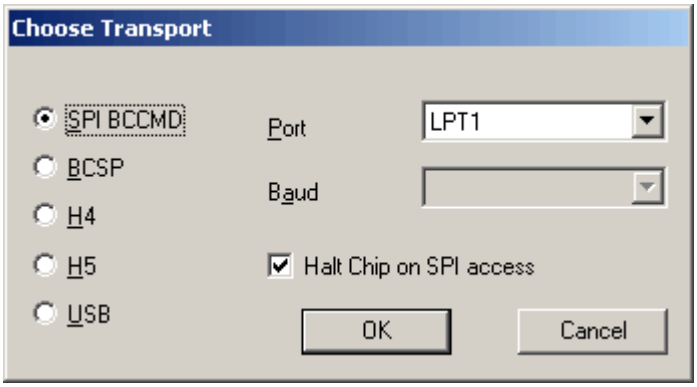
   Alternatively, the factory settings can be restored using the PSTool application supplied with BlueLab, the procedure is described in detail below:

   To restore the factory default settings using PSTool:

   4.1. Open the PSTool application.

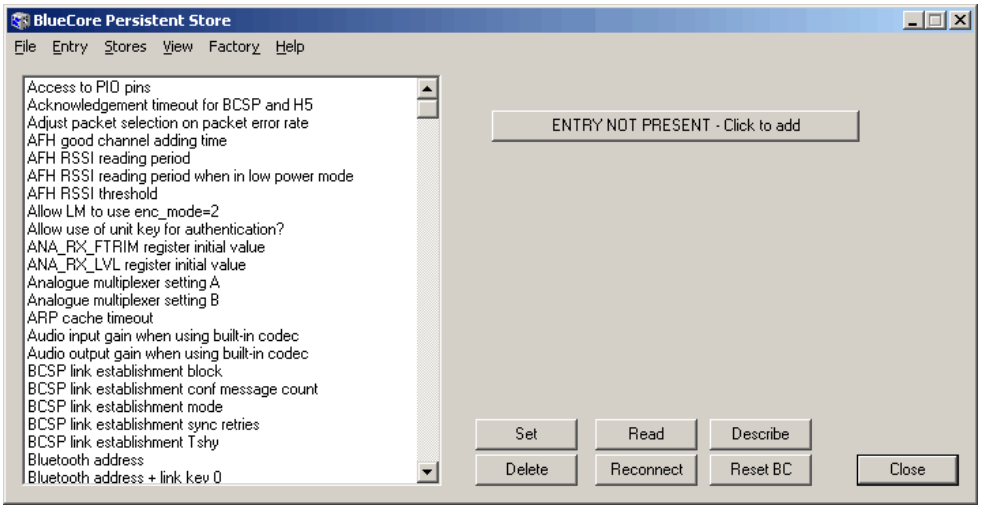
   Note: The PSTool.exe can be found in the BlueLab install directory.
   `eg C:\BlueLab\tools\bin\PSTool.exe`

   4.2. The Choose transport dialog is displayed:



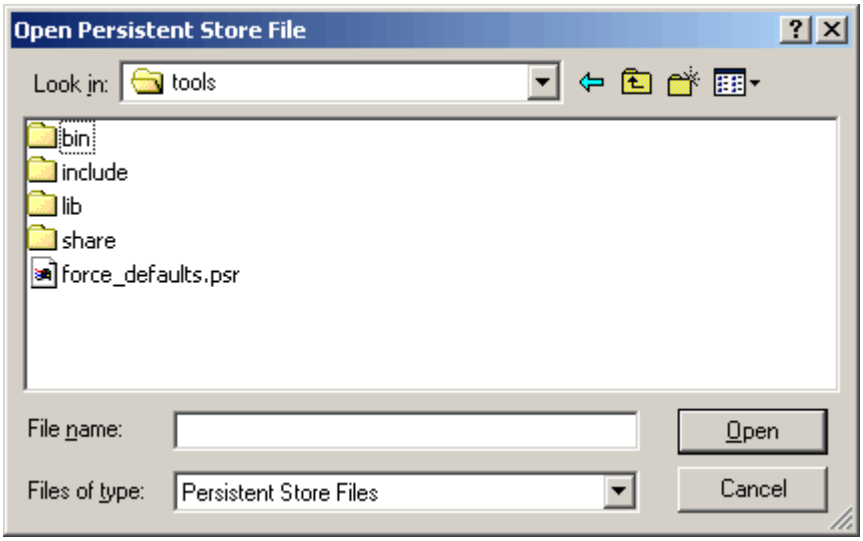
   4.3. Select **SPI BCCMD** and **LPT1** as the transport settings and click **OK**.

   4.4. The PSTool application dialog opens:



4.5. Select **Merge** in the **File** menu:

An 'Open Persistent Store File' chooser dialog appears:



4.6. Browse to the location of the `force_defaults.psr` file.

The file is located in the BlueLab install directory
eg `C:\Bluelab\tools\force_defaults.psr`.

4.7. Select the file and click **Open**.

The file will be run and the PSkeys, which if incorrectly set may result in unexpected chip behaviour, will be restored to their factory defaults.

Note: This may take a few minutes.

5. *I have modified the host transport using PSTool but every time I run my application in xIDE the host transport is modified, why?*

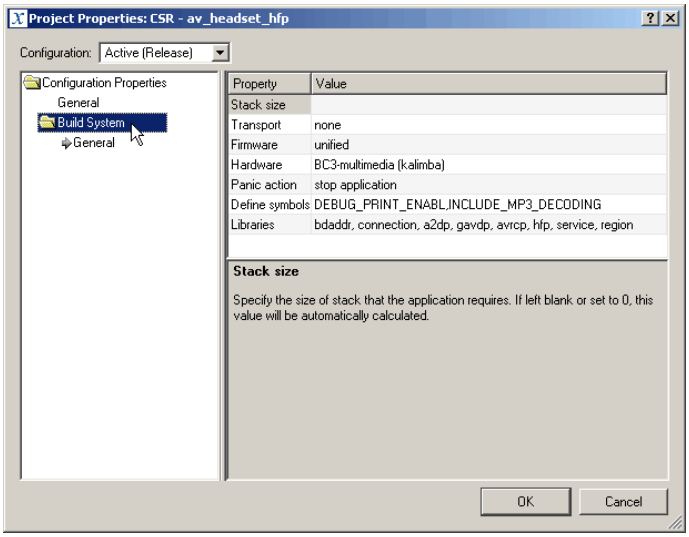
   xIDE sets the host transport to the type specified in the Project Properties settings each time the application is executed.

   To set the required transport method in the Project Properties dialog:

   5.1. Select **Properties** in the **Project** menu.

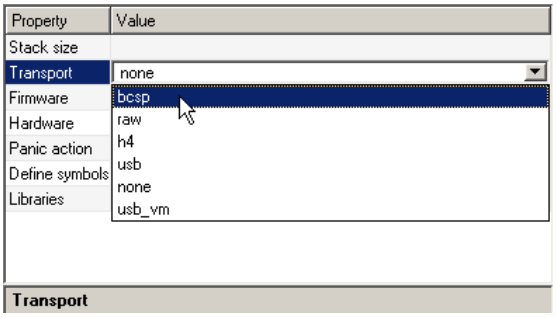
   The Project Properties dialog opens.

   5.2. Click on the **Build System** folder:



   5.3. Click on **Transport** in the list of properties, to activate the Value field.

   5.4. Select the required **Transport** from the drop down list:



   5.5. Click **OK** and then close the Project Properties dialog.

6. *I have debug print messages in my application. When I run my application within xIDE the application runs and I can view the debug messages. However, when I run the application on-chip, it fails to execute, why?*

   For the application to run with debug messages, the messages generated must be picked up either in xIDE or using the vmSpy application, otherwise the application will be stopped to avoid overflowing buffers.

   Debug print messages must be disabled or removed before the application can be run exclusively on-chip.

   If you have enabled debug printing using the print library and defining `DEBUG_PRINT_ENABLED` in the **Define symbols** field of the project properties remove it from the project properties and rebuild the application to disable the debug messages.

   Note: Use of VMSpy requires that the host transport is set to bscp, h4 or usb (as described in question 5) and that a suitable cable is used to provide the connection.

6 Technical Support

Further information on all CSR products can be found on the technical support website (http://www.csrsupport.com).

Developers are also recommended to view the public newsgroups hosted by CSR on the Internet news (NTTP) server news.csr.com. The newsgroups are a convenient forum for the Bluetooth community to exchange knowledge and are a valuable source of information.

Set up instructions and guidelines for the use of newsgroups can be found by following the links on the CSR support website.

Terms and Definitions

BCSP	BlueCore Serial Protocol, a proprietary transport protocol
BlueCore™	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth®	Set of technologies providing audio and data transfer over short-range radio connections
Bluetooth SIG	Bluetooth Special Interest Group
CSR	Cambridge Silicon Radio
BCSP	BlueCore Serial Protocol, a proprietary transport protocol
DSP	Digital Signal Processor: a microprocessor dedicated to real-time signal processing.
H4	UART-based HCI transport, described in section H4 of v1.0b of the Bluetooth Specification
LPT1	First Parallel Printer Port
MMI	Man Machine Interface
raw	raw transport protocol where the VM has full control of the UART port
SDK	Software Development Kit
SDI	Software Development Interface
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus protocol
usb_vm	Universal Serial Bus protocol where the VM has full control of the USB port
VM	Virtual Machine; environment in the BlueCore firmware for running application-specific code produced with BlueLab

Document History

Revision	Date	Reason for Change
a	28 MAY 05	Original publication of this document. (CSR reference: blab-ug-002Pa)

BlueLab™

xIDE User Guide

blab-ug-002Pa

May 2005