CML Microcircuits
COMMUNICATION SEMICONDUCTORS

Getting Started With The EC0003 Application

Publication: CL/EC0003START/UM/3
February 2016

User Manual for PE0003 C Environment
Quick Set-up

Contents

1 Introduction

This document forms a quick set-up reference for using the EC0003 application with the PE0003 hardware to develop, test and debug C routines for CML devices. It is necessary to install LPCXpresso and download the EC0003 with the set of CML libraries.

A practical example is given using step-by-step instructions to develop a project using the EC0003 GUI interface and LPCXpresso. It is not intended to give a detailed explanation of the development process or the LPCXpresso toolchain, but to illustrate how the tools can be used to create a project in a friendly and hassle-free manner. This guide also introduces tips to guide in you through the learning process. A more detailed explanation is given in the LPCXpresso User Guide, listed at the end of the document, see section 13

The EC0003 application bundle, with the CML libraries and LPCXpresso should be downloaded before continuing.

It is possible to use other tools instead of LPCXpresso but because the NXP processor in the PE0003 is the target, it is practical to use the libraries and tools provided for this chip (LPC43XX).

1.1 History

Version	Changes	Date
1	First issue of document	July 2015
2	Section 4.6: “Add directory path” dialog updated to show correct path. Added supporting text regarding libraries configuration. Section 4.6: “Properties for…” dialog updated	February 2016
3	Section 8 – “LPC-Link2 Configuration” added after finding issues with the LPCXpresso version 7.6.2 onwards.	February 2016

1.2 Glossary

CML CML Microsystems plc group of companies

EC0003 CML GUI providing a C interface and a programming path via the USB interface.

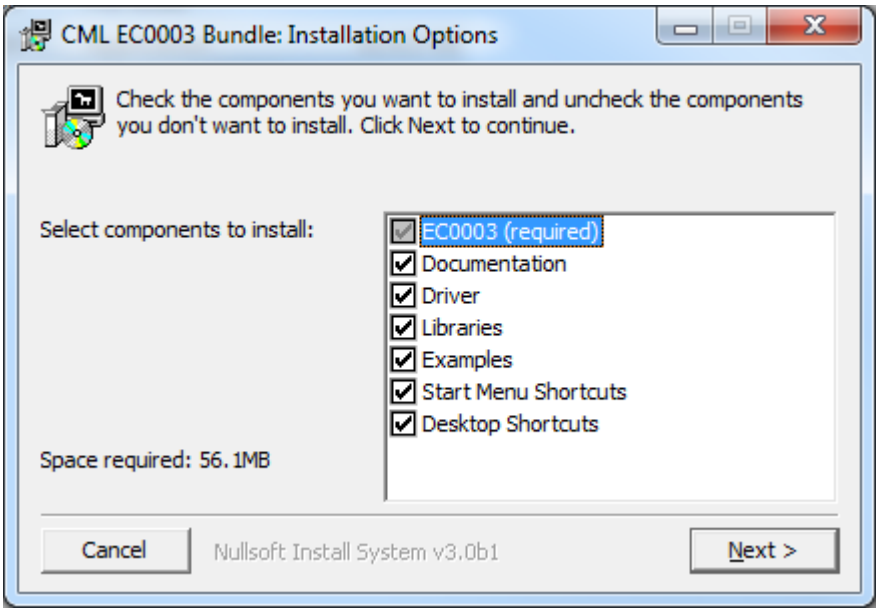
PE0003 Evaluation Kit Interface Card.

2 EC0003 Installation

Download and install the LPCXpresso tools. Download the PE0003 drivers from the CML website and install them. See the PE0003 User Manual for instructions.

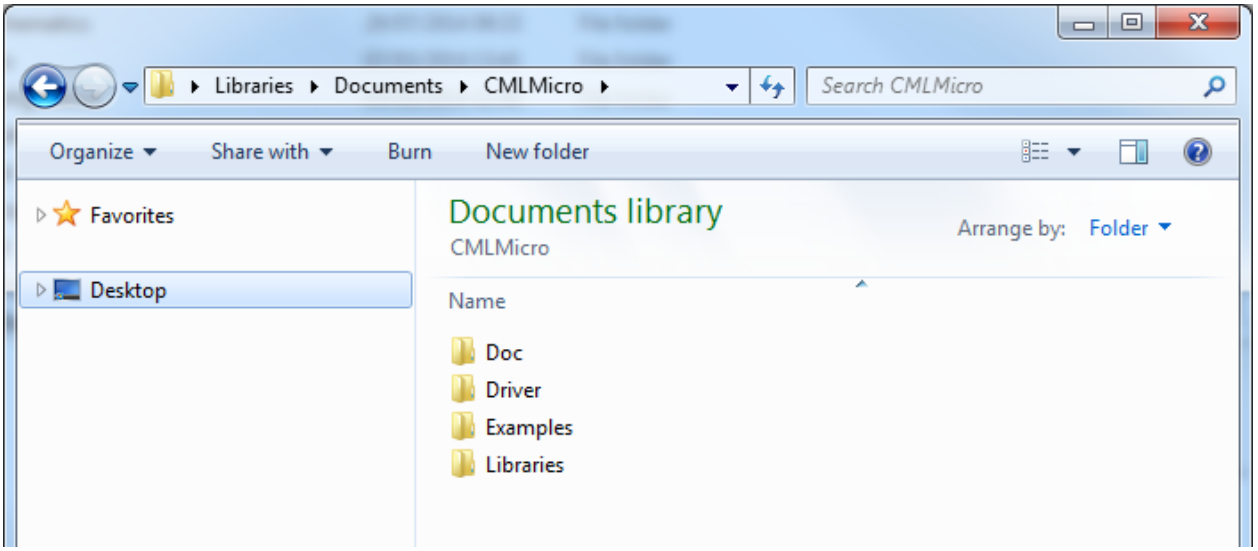
The EC0003 installation includes the ‘C’ libraries. Run ‘EC0003Installer.exe’

The installation has the option to create shortcuts in the start menu and the desktop. De-select them if they are not required.



The libraries, documentation, examples and PE0003 driver FILES are installed in a folder called CMLMicro under ‘Documents’. It is suggested that this folder is used initially. It can be changed later as required.

C:\Users\xxxx\Documents\CMLMicro

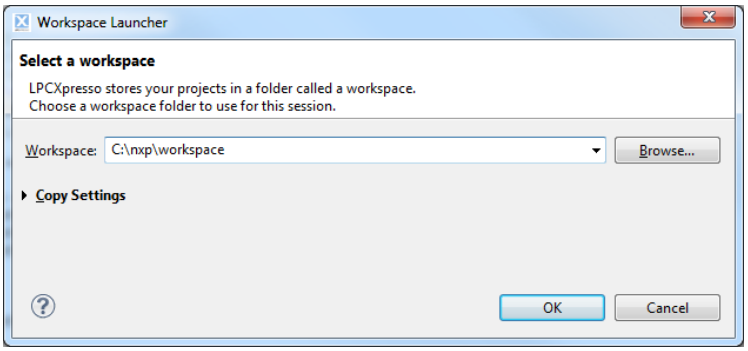


3 Creating a Base Project

The Base Project that will be created here is a template that can be used to create other projects quickly.

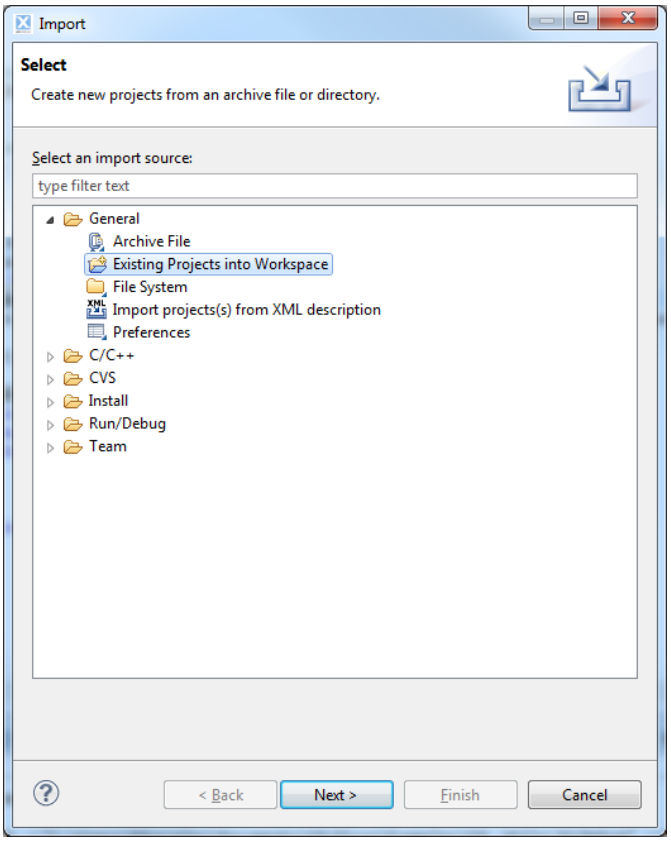
Start LPCXpresso.

Select the workspace (where your project will be held). If LPCXpresso does not ask for the workspace at startup, go to File → Switch workspace → Other



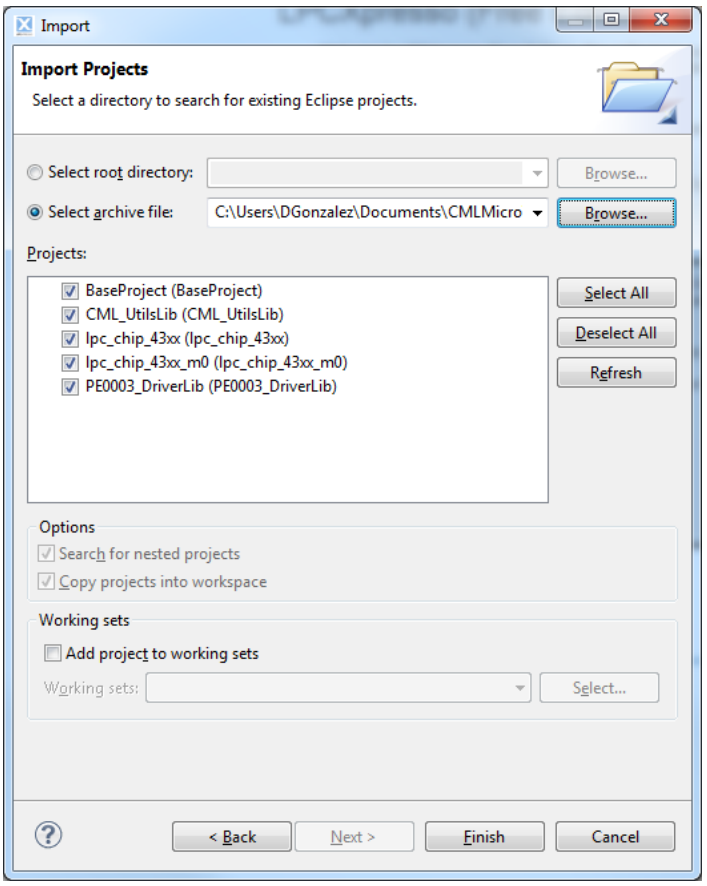
Import the CML libraries “CML_C_BundlevXX.zip” from Documents\CMLMicro\Libraries.

File → Import. Select General → Existing Projects into Workspace



Next

Select archive file, browse “C_BundlevXX.zip”.



Click Finish

In the Project Explorer is the BaseProject folder. This folder contains the file BaseProject.c which itself contains the main() function.

Select Project → Build All. This generates the .bin file.

Start EC0003.

Connect the PE0003 to a power supply and to the PC. Refer to the PE0003 User Manual for more information – see section 13.

Power up the PE0003.

Browse “BaseProject.bin” in “$YourWorkspace\BaseProject\Debug” and select the “.bin” file.

The program should display a message on the screen “Hello from PE0003”.

The BaseProject has been configured to use the CML libraries and does not require any other configuration.

4 Creating a LPCXpresso Project

The wizard will be used to create a project because this avoids the detailed configuration of libraries, paths, chip, memories, global variables and other options, required to configure a full project. This is a quick set-up procedure that can be used for most CML target projects.

4.1 STEP 1 – Open LPCXpresso and select workspace

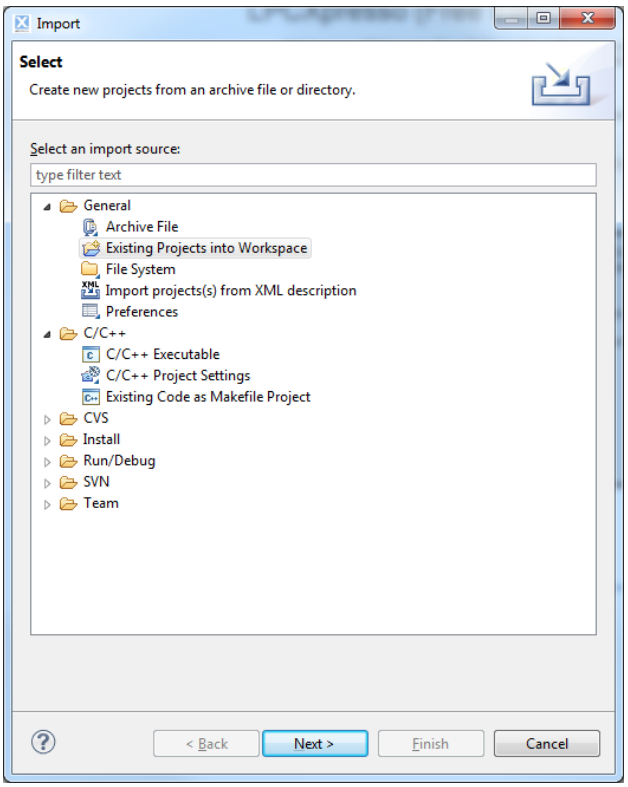
If LPCXpresso does not prompt you for a workspace, select the workspace where you want to save your projects.

File → Switch Workspace → Other.

4.2 STEP 2 – Import Libraries to your workspace

File → Import

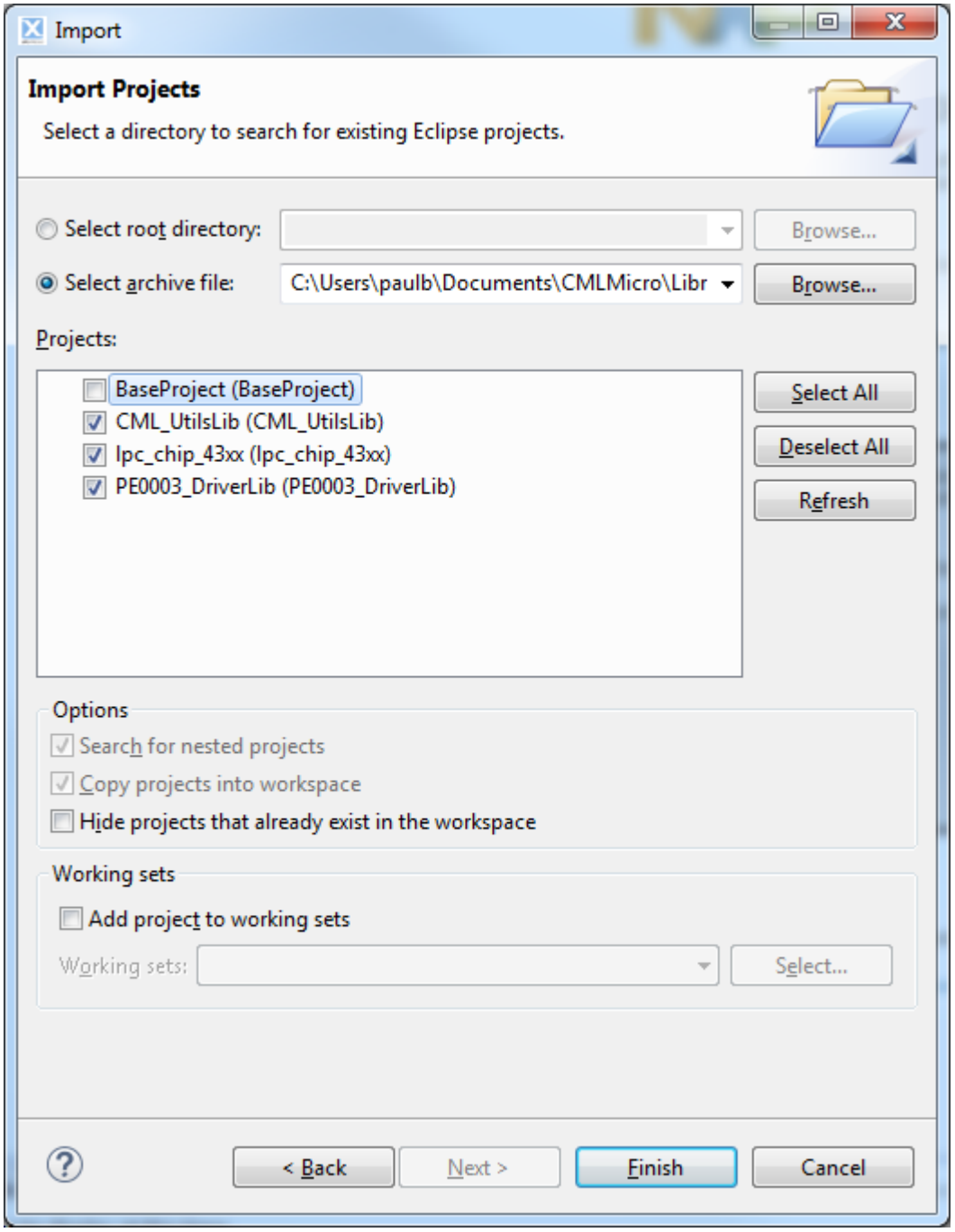
Expand General and select Existing Projects into Workspace.



Click Next.

Select Archive File and select the “CML_C_Bundlev1_0.zip” from “Documents\CMLMicro\Libraries”. Deselect all then select the libraries or projects you require. For this simple project select the libraries:

- Lpc_chip_43xx - Chip libraries
- PE0003_DriverLib – PE0003 peripheral libraries
- CML_UtilsLib – Libraries for the interface with EC0003

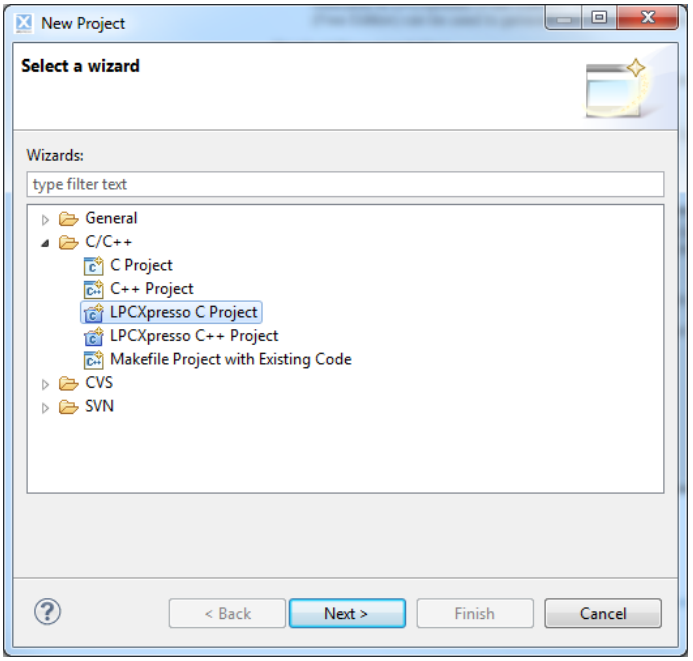


Click Finish

4.3 STEP 3 – Create new project

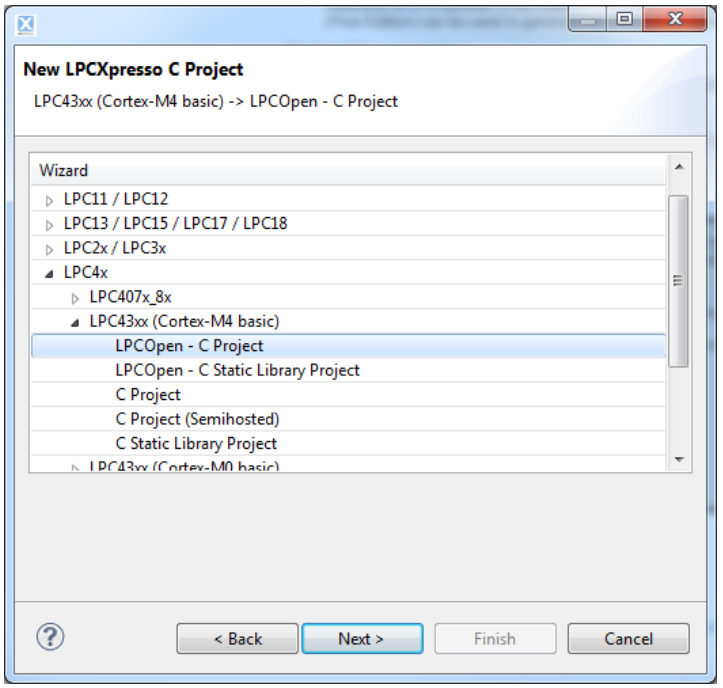
File → New → Project

Expand C/C++ and select “LPCXpresso C Project”



Click Next

Select the chip model and type of project. LPC4x → LPC43xx(Cortex-M4 basic) → LPCOpen – C Project



Click Next.

Set the project name.

Click Next

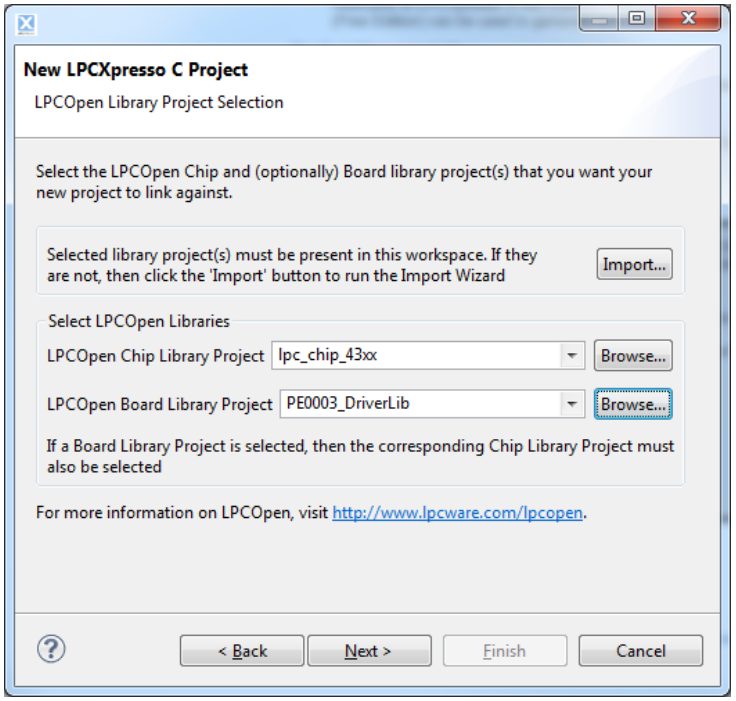
Select the chip “LPC4330”

Click Next

Select the required libraries:

In Select LPCOpen Libraries, click browse for LPCOpen Chip Library Project and select “lpc_chip_43xx”

In LPCOpen Board Library Project select “PE0003_DriverLib”.



Click Next.

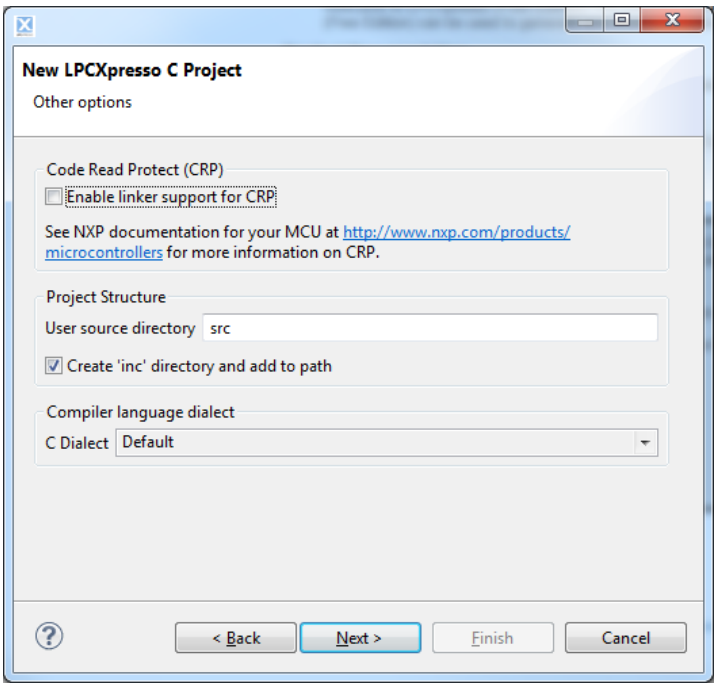
‘CMSIS DSP Library Project Selection‘ no changes required’

Click Next.

‘Part specific options ‘no changes required’

Click Next.

‘Other options ‘De-select ‘Code Read Protect (CRP)’ and select ‘Create ‘inc’ directory…’



Click Next

‘Memory Configuration Editor‘ no changes required

Click Next

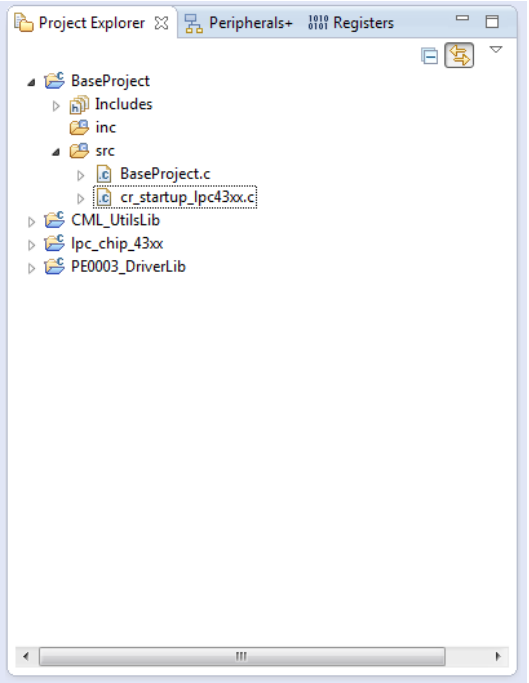
‘Printf options‘ no changes required

Click Finish.

4.4 STEP 4 – Clean project folder

Remove the unwanted files generated by the wizard:

“crp.c “ – this provides code read protection and is not required
“sysinit.c” – initialises the PE0003 environment. This file is already included in PE0003_Driver.lib and could be copied into your project.



4.5 STEP 5 – Code base for your project

The wizard creates a top level source file, BaseProject.c. The contents should be replaced with the following:

```
#include "chip.h"
#include "pe0003.h"
#include "pe0003_config.h"
#include "ftdi.h"
#include "cbus.h"
#include "gpio.h"
#include "timer.h"
#include "lpc_types.h"

#include "monitor.h"
#include "messages.h"
#include "mon_general.h"

#include <stdio.h>

//To use the GUI printfs
#define printf        mprintf
#define scanf       mscanf



int main(void)
{

        // Initialise board
        Pe0003_BoardInit();

        //FTDI usb
        Pe0003_UsbFtdiInit();


        // Initialise CBUS
        Pe0003_CbusInit(CBUS1);
        Pe0003_CbusInit(CBUS2);

        //Init Timer
        Pe0003_TimerInit();

        //GPIO
        Pe0003_GpioDedicatedIOInit();
        Pe0003_GpioGenIOInit();

        //Init GUI
        Cml_GuiSystemInit();


      mprintf("Hello from PE0003");

        while(1)
        {
            //Add code
        }
        return 1 ;
}
```

Notes:

- Chip.h – contains some configuration details for the chip and the basic data types
- Pe0003_config.h – contains the macros for the configuration

Inside Main() are these functions:

- Pe0003_BoardInit – configures all the pins and clocks for the PE0003 peripherals
- Pe0003_XXXXInit – initialises the PE0003 host's peripherals with a default configuration
- Cml_GuiSystemInit – initialises and sets the link for the message-based communication with the EC0003

4.6 STEP 6 – Configure properties

The wizard set-up will have done most of the job but not all, so there a few more configuration steps required:

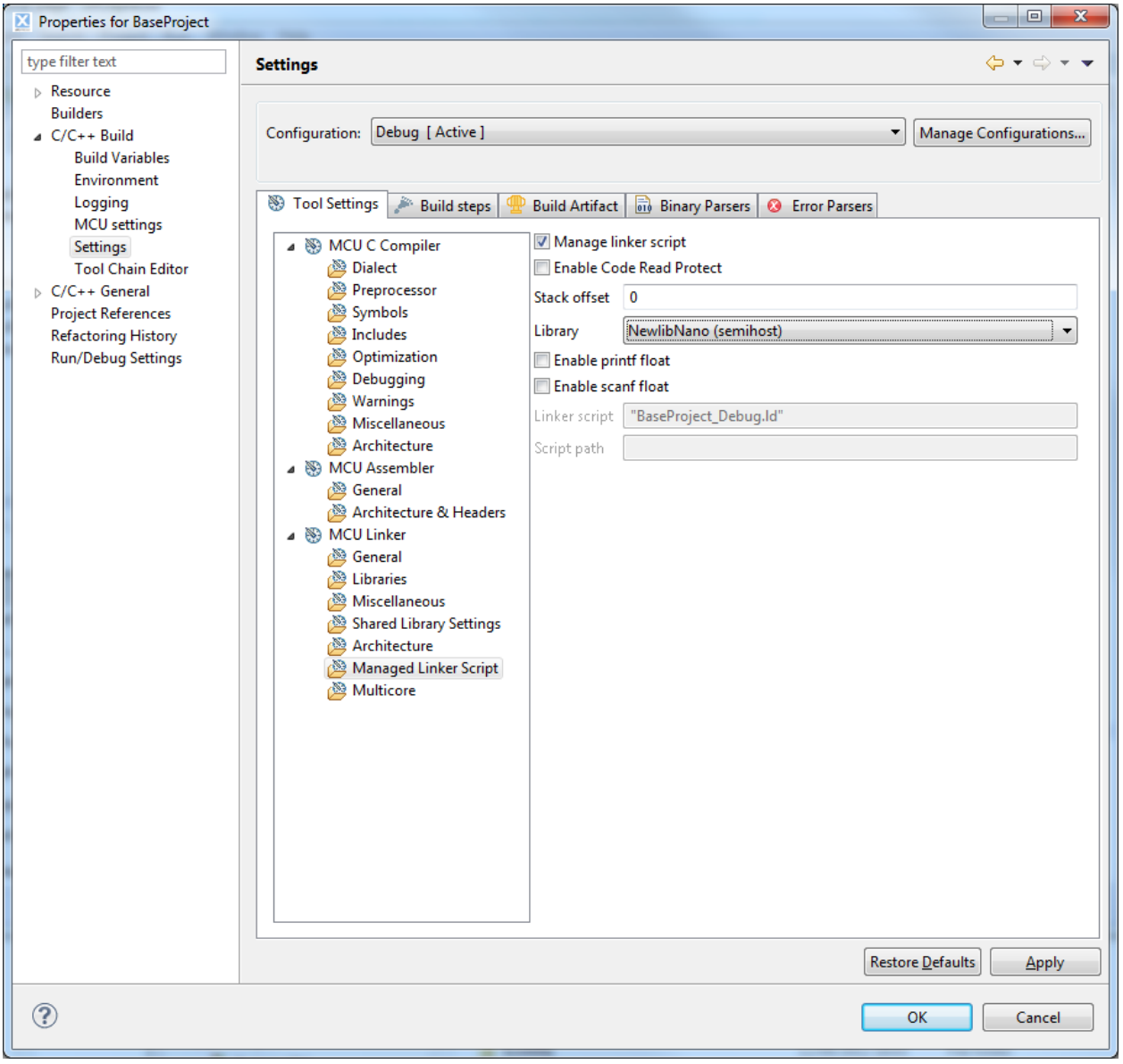
First, configure for ‘semihosting’ mode:

Select your project in the “Project Explorer” then right mouse click it and select “Properties”.

Select C/C++ Build → Settings. Inside Settings select MCULinker → Managed Linker Script.

In Library select ‘NewlibNano (semihost)’. Semihost mode allows the use of stdio.h, stdarg.h and other system libraries, so it is possible to use printf and scanf for example.

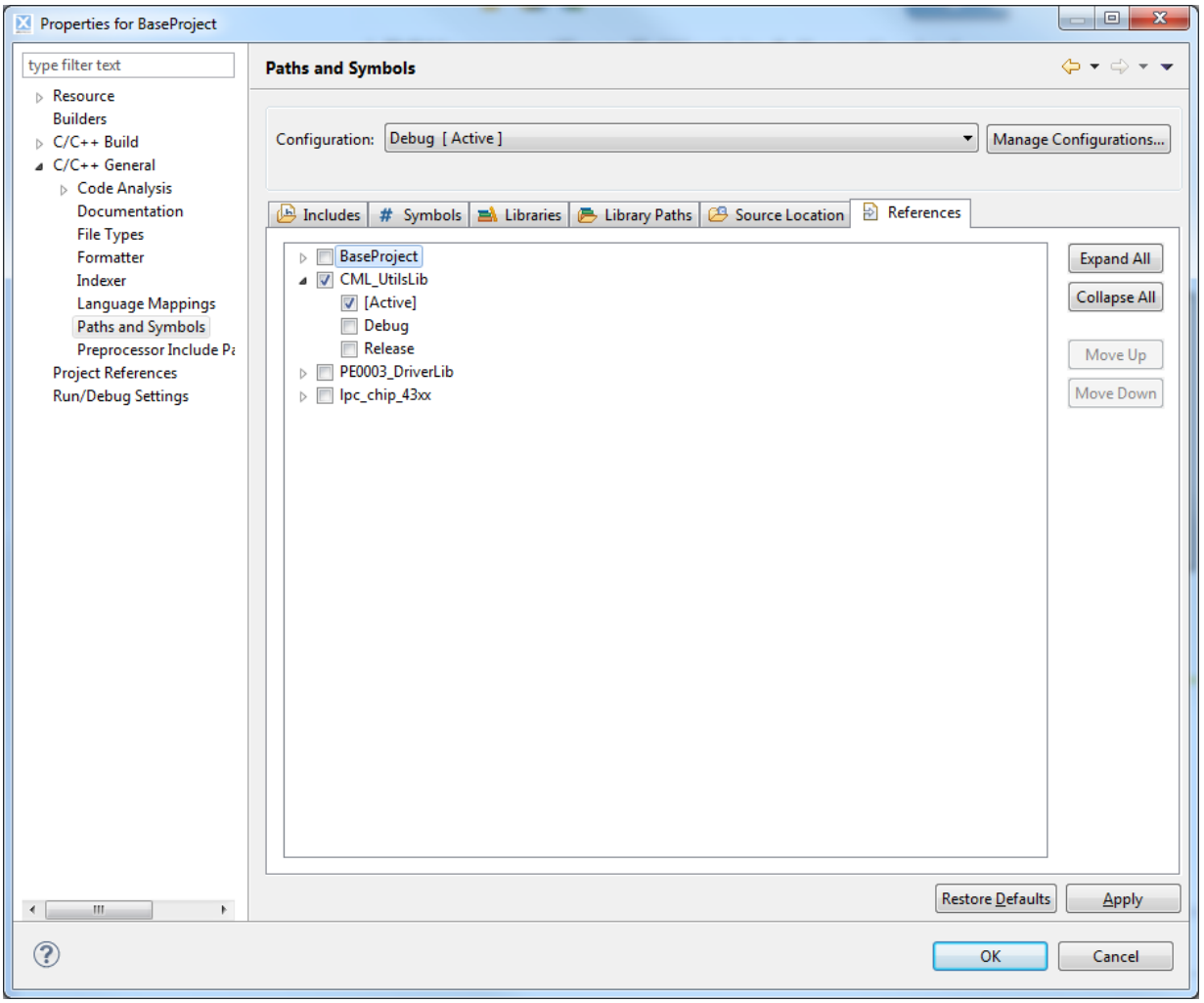
Note: For communicating with EC0003 using printf the library mon_general.h has been added. It has an implementation of printf called mprintf and scanf called mscanf. In the example code there is the macro ;
“`#define printf mprintf`”
This converts printf into mprintf, used by EC0003. They work in the same way as the generic printf and scanf.



Second, add the remaining libraries.

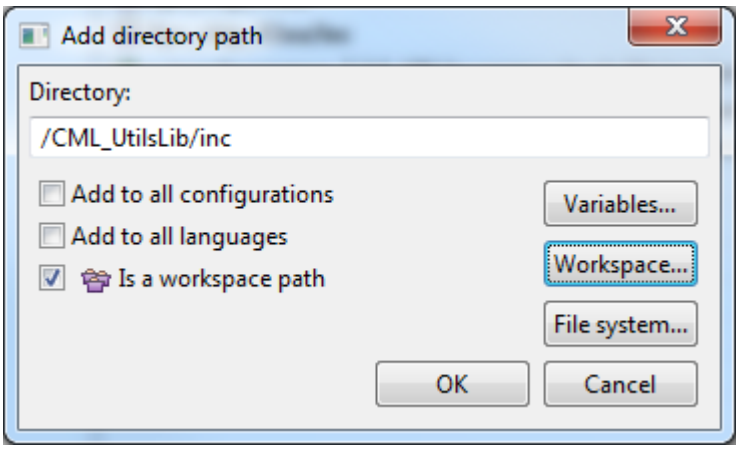
In properties select C/C++ General →Paths and Symbols.

Select the References tab and then select ‘CML_UtilsLib’



In “Includes” tab, select GNU C then click “Add”.

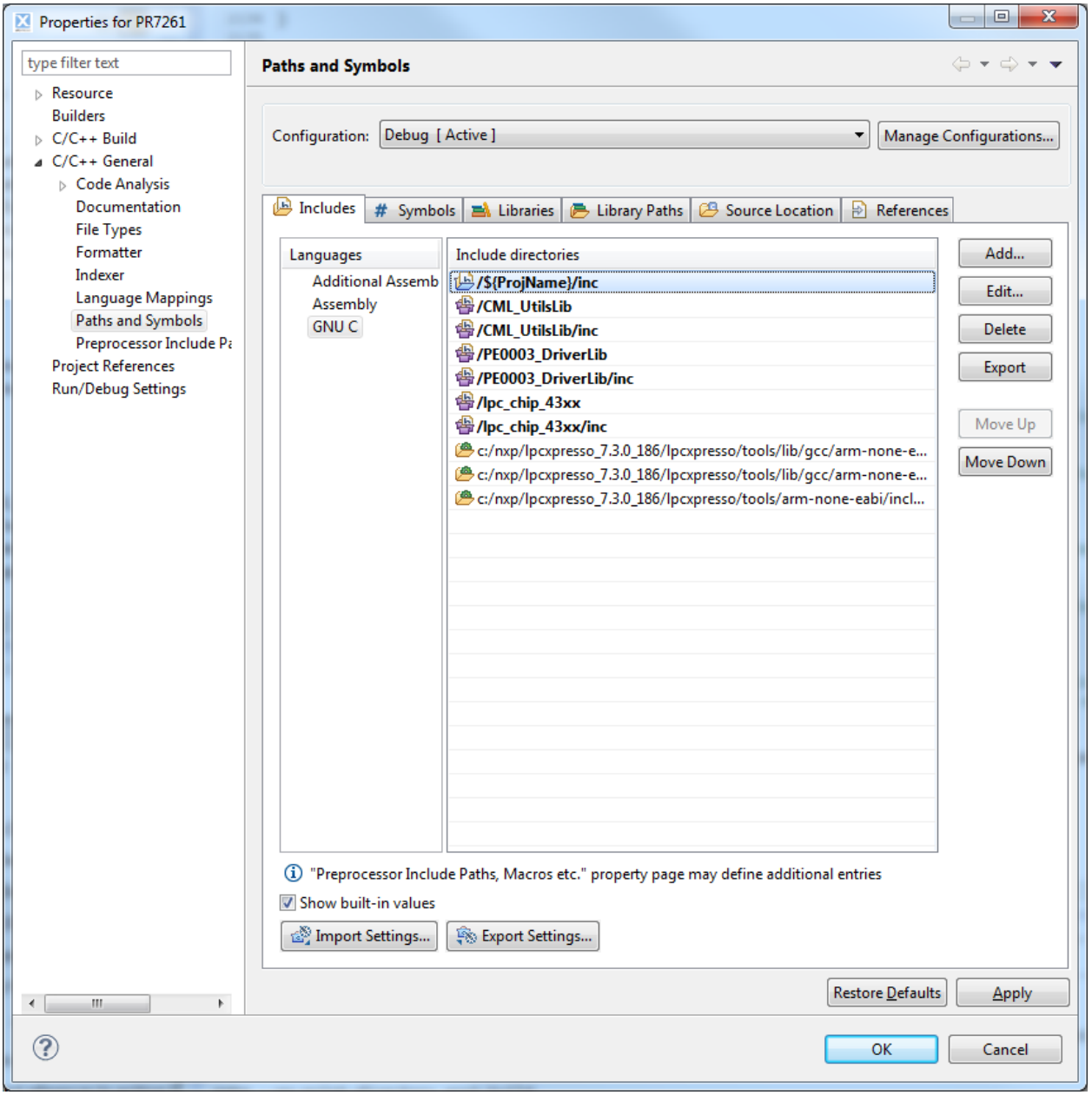
Click ‘Workspace’ then select the library inc folder “/CML_UtilsLib/inc”



Click OK

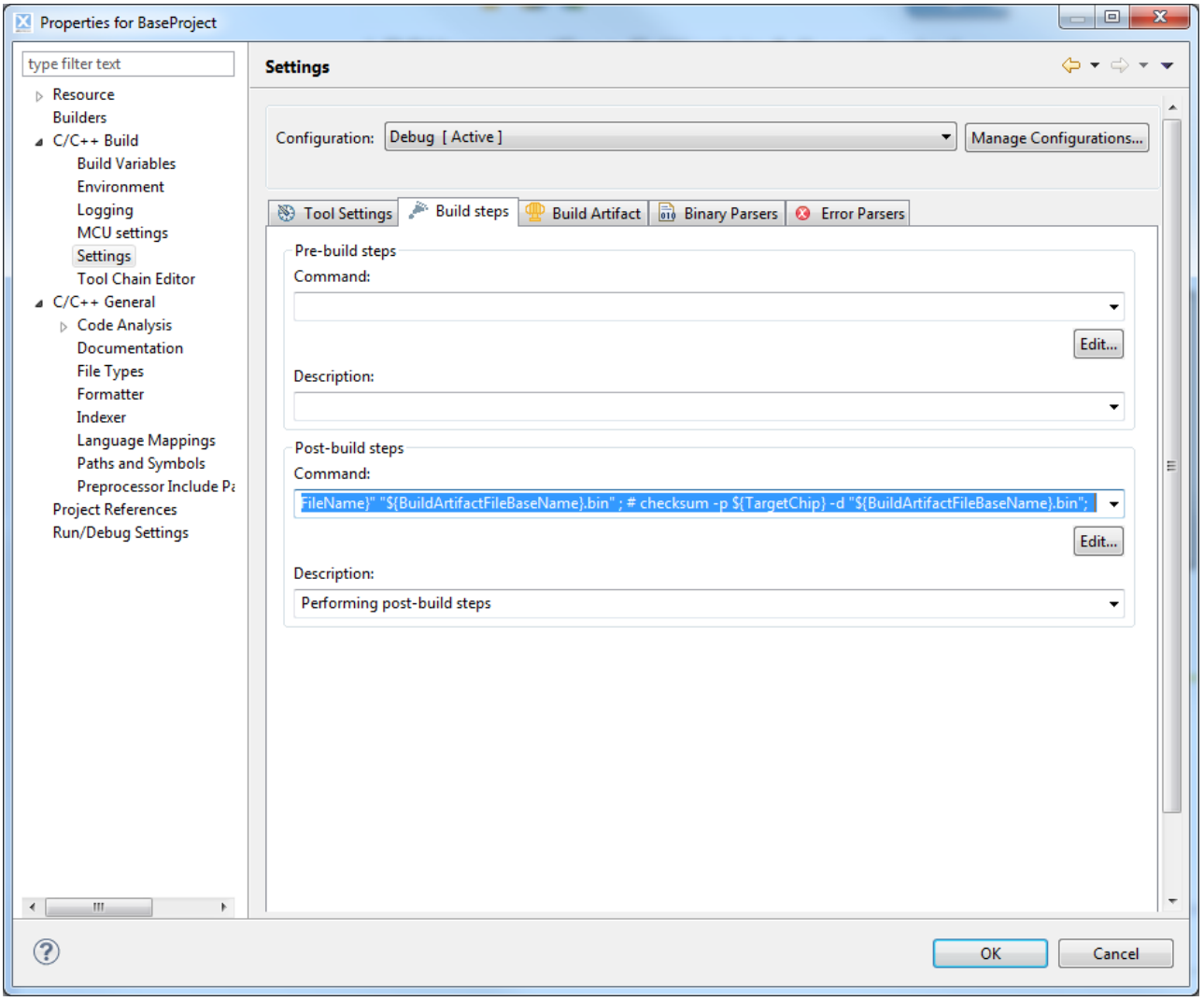
Add the rest of the libraries in the same way if required; lpc_chip_43xx/inc and PE0003_DriverLib/inc.

Rearrange the order of the libraries, placing the most dependent libraries at the top and the less dependent at the bottom. The image below has the libraries correctly rearranged. If the order is incorrect the linker may report an ‘undefined reference’ error.


Properties for PR7261
type filter text
Resource
Builders
C/C++ Build
C/C++ General
Code Analysis
Documentation
File Types
Formatter
Indexer
Language Mappings
Paths and Symbols
Preprocessor Include Pa
Project References
Run/Debug Settings
Paths and Symbols
Configuration: Debug [Active]
Manage Configurations...
Includes
Symbols
Libraries
Library Paths
Source Location
References
Languages
Additional Assemb
Assembly
GNU C
Include directories
/${ProjName}/inc
/CML_UtilsLib
/CML_UtilsLib/inc
/PE0003_DriverLib
/PE0003_DriverLib/inc
/lpc_chip_43xx
/lpc_chip_43xx/inc
c:/nxp/lpcxpresso_7.3.0_186/lpcxpresso/tools/lib/gcc/arm-none-e...
c:/nxp/lpcxpresso_7.3.0_186/lpcxpresso/tools/lib/gcc/arm-none-e...
c:/nxp/lpcxpresso_7.3.0_186/lpcxpresso/tools/arm-none-eabi/incl...
Add...
Edit...
Delete
Export
Move Up
Move Down
"Preprocessor Include Paths, Macros etc." property page may define additional entries
Show built-in values
Import Settings...
Export Settings...
Restore Defaults
Apply
OK
Cancel

Finally, LPCXpresso is configured for use with a debugger. By default, LPCXpresso will not generate the .bin file used by EC0003 to program the PE0003.

To generate the .bin file, in properties select C/C++ Build → Settings. Select Tab “Build steps” remove the comment symbols ‘#’ in “Command:” text box, using the ‘Edit…’ control.



Click OK.

5 Coding

Add project-specific libraries and code for the application.

5.1 Build your Project

Project→ Build all.

6 Running the Project

There are three options for running the project:

- Using the generated .bin to be loaded via the EC0003 GUI. This option does not require a debugger as the programme download is through the USB port to the PE0003.
- Using a debugger. NXP provides the LPC-LINK2, a low-cost debugger that is supported in LPCXpresso . The EC0003 interface can be used with the debugger, preloading the code with LPCXpresso.
- Using the debugger and LPCXpresso without the EC0003 GUI.

6.1 Option 1 - EC0003

Build the ".bin" file in LPCXpresso.

Connect the PE0003 to the PC using a USB cable and power up the PE0003.

Run EC0003.exe.

In EC0003, select Browse and navigate to the .bin file in the project folder. This will normally be found in the Debug or Release folder.

Note: See the examples in section 9

6.2 Option 2 - Debugger + EC0003

Build the project in LPCXpresso.

6.2.1 STEP 1 – Create the launcher in LPCXpresso for loading the code into the chip.

In the Project Explorer, right-click the project. Select Launch Configurations → Create New…

Two new files appear in your Project folder called "ProjectName Debug.launch" and "ProjectName Release.launch". These files must be re-configured for use with the PE0003. Right-click Debug.launch and select Launch Configurations → Edit current.

Go to Debugger tab (For an LPC-LINLK2 Configuration):

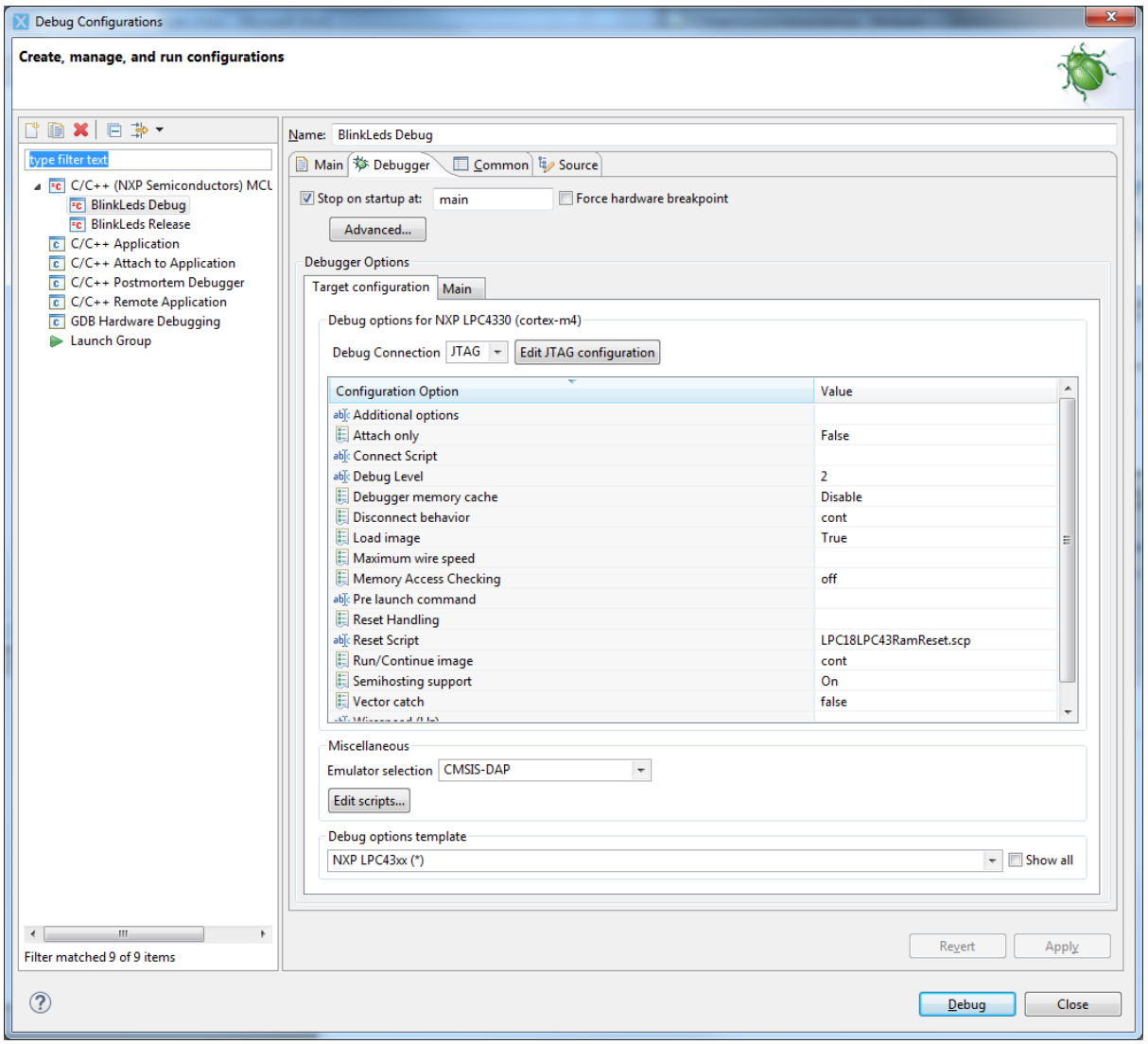
- In "Emulator selection" select Redlink server.
- In Debug options template select NXP LPC43xx (SWD) (it is possible to use JTAG too).
- In Debug options for NXP "Reset script" field. Browse Scripts and select "LPC18LPC43RamReset.scp". The PE0003 uses the LPC4330 and, because it is a flashless device, the code runs from RAM.

Click Apply.

Connect the debugger then click Debug to load and run the program.

Once running, the main() function is highlighted and the debug commands from the menu or shortcuts become active. It is possible to restart with the 'Debug' button in the toolbars. The configuration is remembered.

Note. Before restarting a debug session, check that the previous debug session has completely stopped otherwise LPCXpresso will stop with an error.



6.2.2 STEP2 – Connect to PE0003

The EC0003 is a Graphical User Interface to ease the interaction with the PE0003. This allows user control of the program, data to be displayed or saved to file and a host of other features of the PE0003.

Run EC0003.exe and toggle the Debug button. Once the code is loaded with the LPCXpresso click the connection button to set the link then debug or run normally.

6.3 Option 3 – LPCXpresso debugger

This is similar to section 6.2.2 but does not use the EC0003 GUI application. Remove function call Cml_GuiSystemInit() from the project code. With this option it is not possible to use of any of the PE0003 features that require the EC0003 GUI such as downloading files or function images and dialog display. The functionality of other libraries is available and the ‘semihost’ mode allows printfs to display information in the LPCXpresso console.

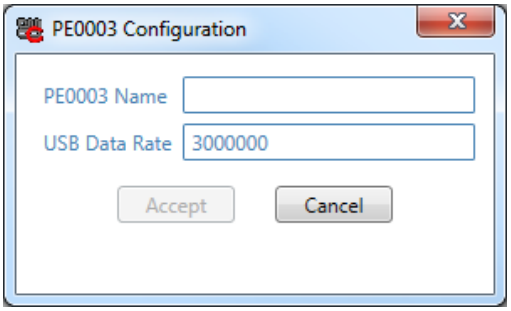
7 EC0003

This section describes the use of the EC0003 GUI application. There are two options for use:

- With a proprietary debugger and EC0003 acting as a display interface.
- Using standalone with the .bin file, generated by LPCXpresso or other compiler.

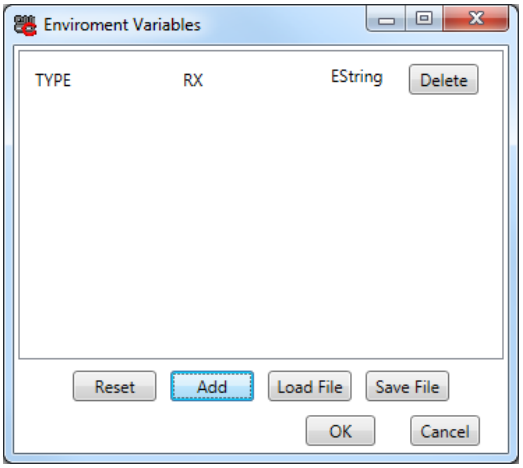
7.1 EC0003 Basic Configuration

In the menu Tools → PE0003 Configuration.



This control allows you to set the name of the board displayed at the top of the GUI or change the USB data rate. It is not recommended to change the USB data rate because this also requires reconfiguration of the firmware library.

From the menu, Tools → Environment variables

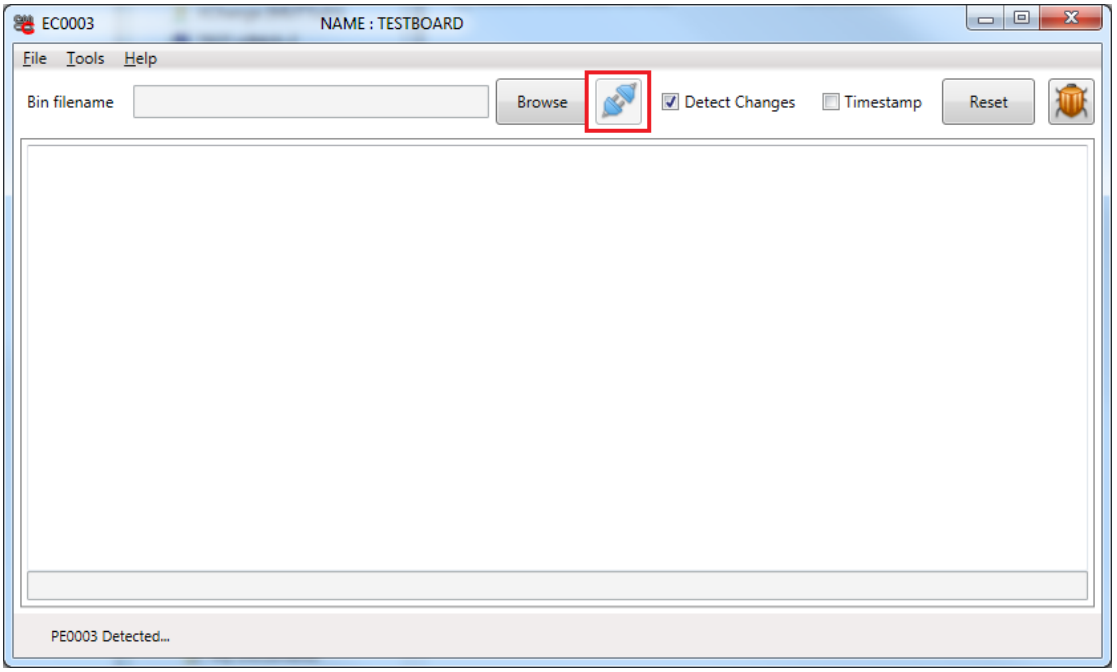


Allows the user to set some environmental variables that can be read from the source code. The GUI remembers the variable for the board name and reloads them, every time the GUI is executed.

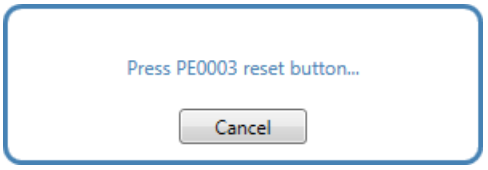
7.2 Using EC0003 Standalone

To use EC0003 standalone it is necessary to load the compiled project as a .bin file.

To allow a new .bin file to be selected, the EC0003 GUI must be disconnected from the PE0003:

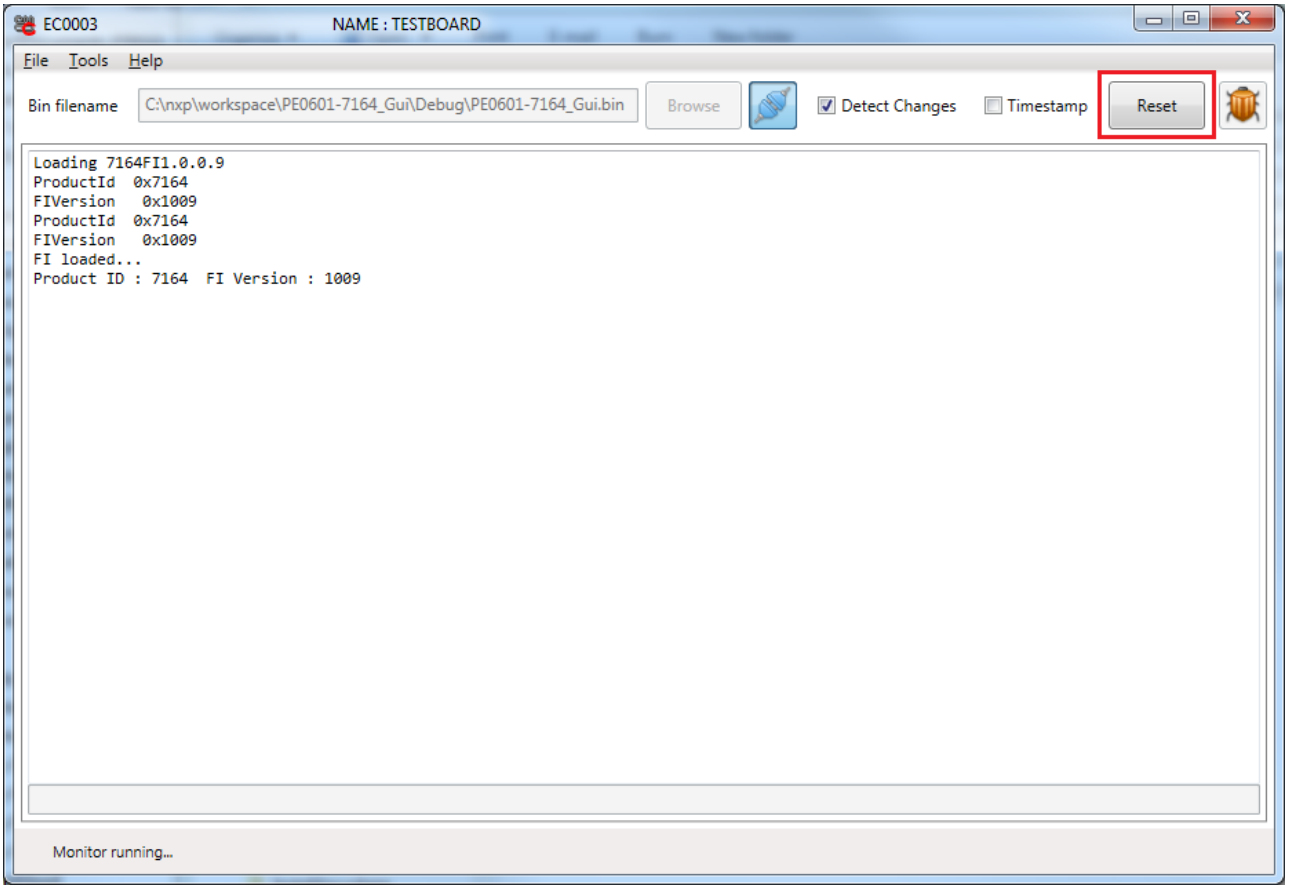


Click Browse and select the .bin file for loading to the PE0003. If the PE0003 was previously loaded, the GUI prompts the user to reset the PE0003 by pressing the reset button on the PE0003.

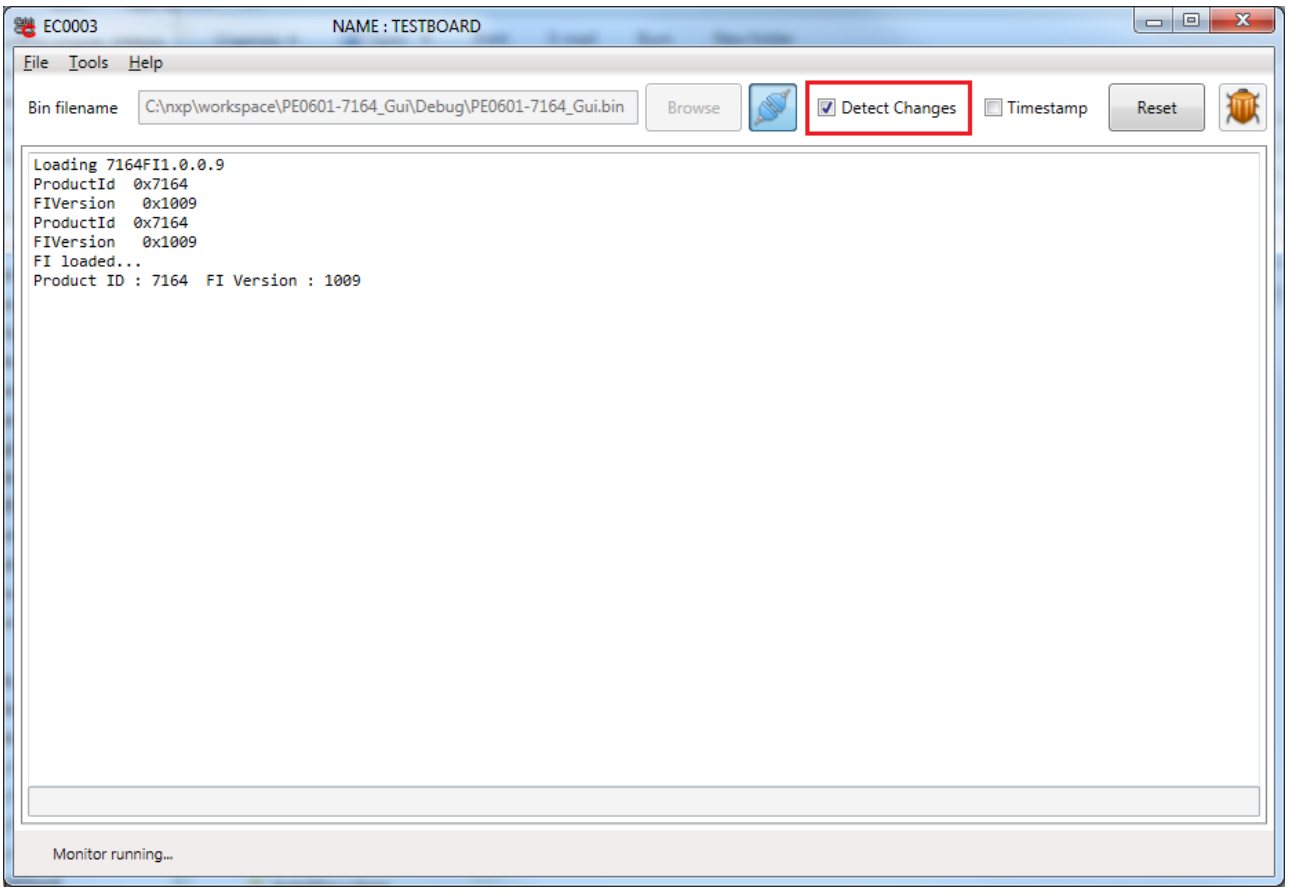


The dialog will clear when the PE0003 reset button is pressed.

To restart the project click the Reset control in the EC0003 GUI, see below. In this example an FI load library function has been called in the project code, resulting in the console messages shown.



If the Detect Changes checkbox is checked, the GUI monitors the selected .bin file for changes, normally as a result of recompiling. If a change is detected the updated .bin will be loaded onto the PE0003 automatically.


EC0003
NAME : TESTBOARD
File
Tools
Help
Bin filename
C:\nxp\workspace\PE0601-7164_Gui\Debug\PE0601-7164_Gui.bin
Browse
Detect Changes
Timestamp
Reset
Loading 7164FI1.0.0.9
ProductId 0x7164
FIVersion 0x1009
ProductId 0x7164
FIVersion 0x1009
FI loaded...
Product ID : 7164 FI Version : 1009
Monitor running...

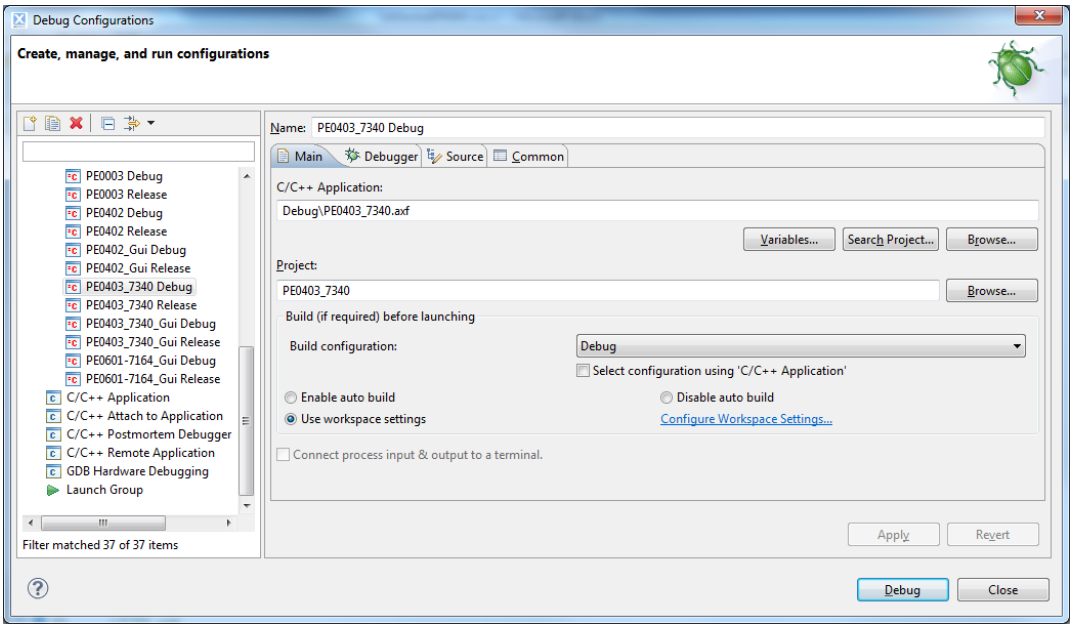
7.3 Using EC0003 With a Debugger

Debugging can be done using both LPCXpresso and the EC0003 GUI.

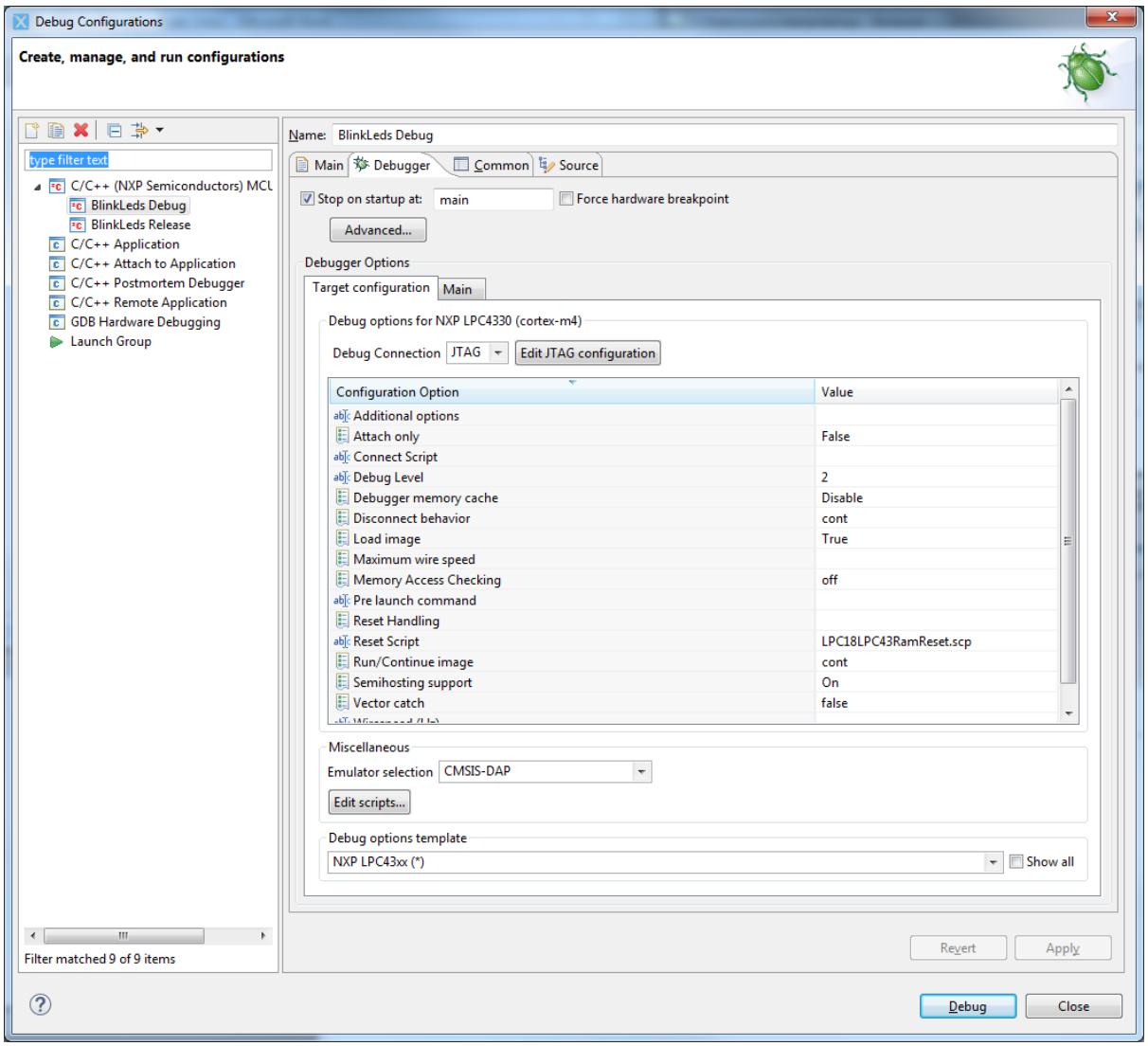
The firmware is loaded into the PE0003 using one of three modes:

MODE 1

- Create a launch file. Locate the project in the “Project Explorer”. Right click the project and select “Launch Configurations” → Create New…
- Or edit the launch configuration by right clicking the project and selecting “Launch Configurations”→ Edit current …



- Select the Debugger tab and configure the Emulator selection to “CMSIS-DAP”. Set the Debug Connection to (SWD) or (JTAG) and in Target configuration, change Reset script to LPC18LPC43RamReset.scp.



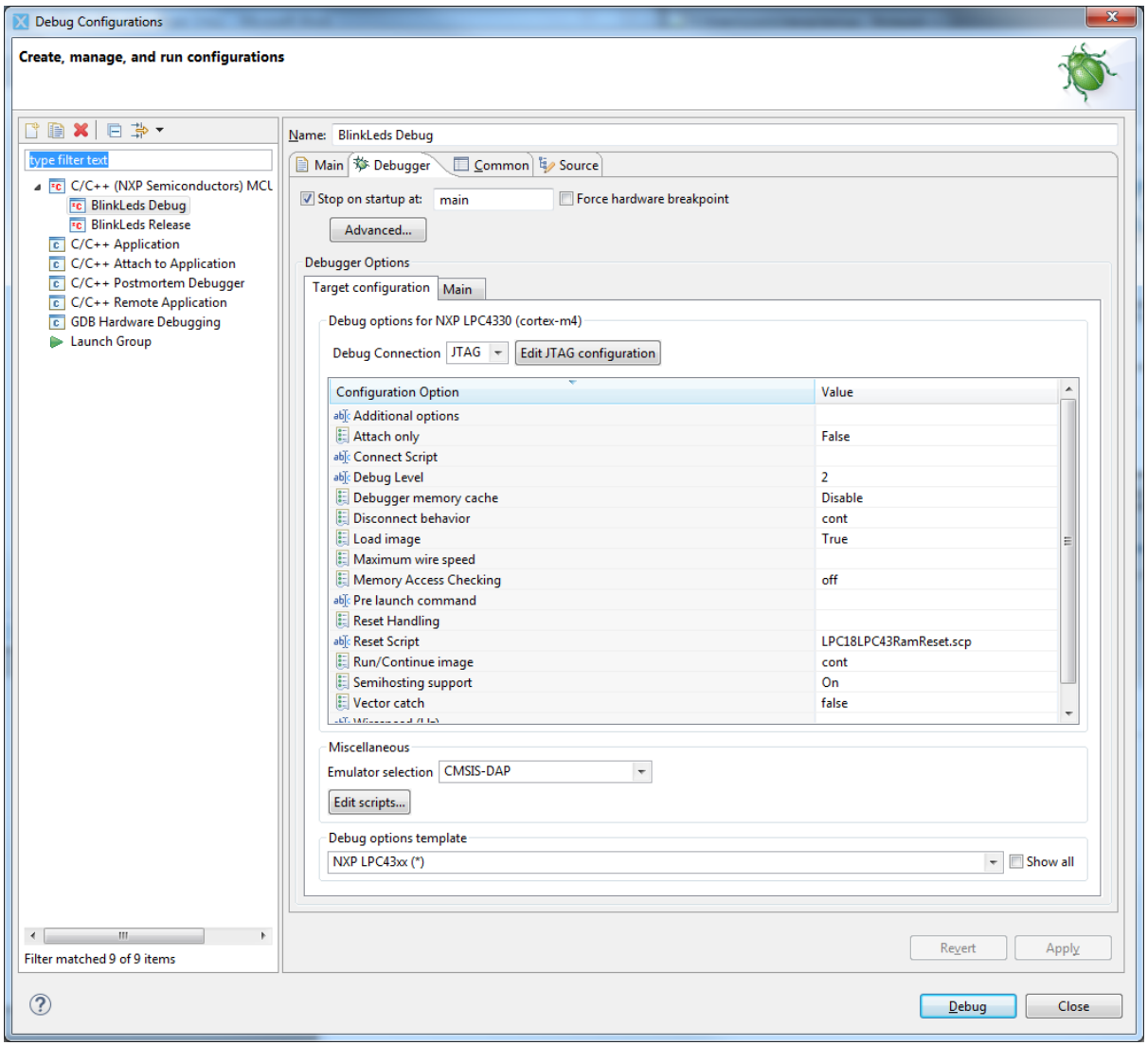
- Click debug.

The configuration is saved so that future debug sessions can be launched by clicking the debug button or using the menus.

MODE 2

This is basically the same as the previous configuration but access is through the Run menu.

- Run → Debug Configurations..
- If there is no launch file for the project, right-click C/C++ (NXP Semiconductor) or one of the other launch files. Select New.
- Select Debugger tab and configure Emulator selection to “CMSIS-DAP”. Set the Debug Connection to (SWD) or (JTAG) and inside the box Reset script changed it to LPC18LPC43RamReset.scp.



- Click debug.

The configuration is saved so that future debug sessions can be launched by clicking the debug button or using the menus.

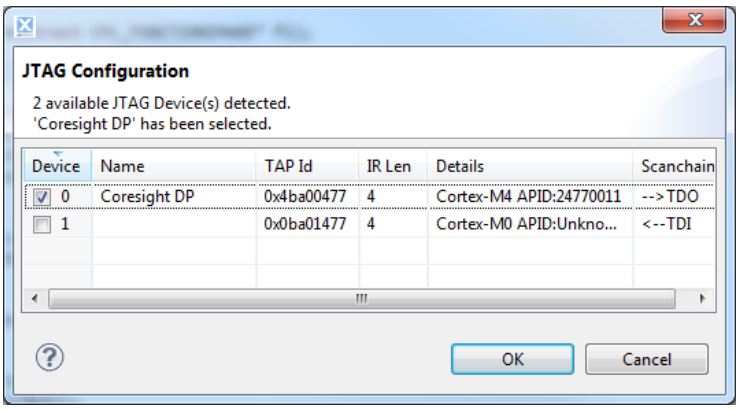
MODE 3

This mode can only be used after the Debug session has been configured. The shortcut button in the ribbon is tied to the last configuration used by the debugger.

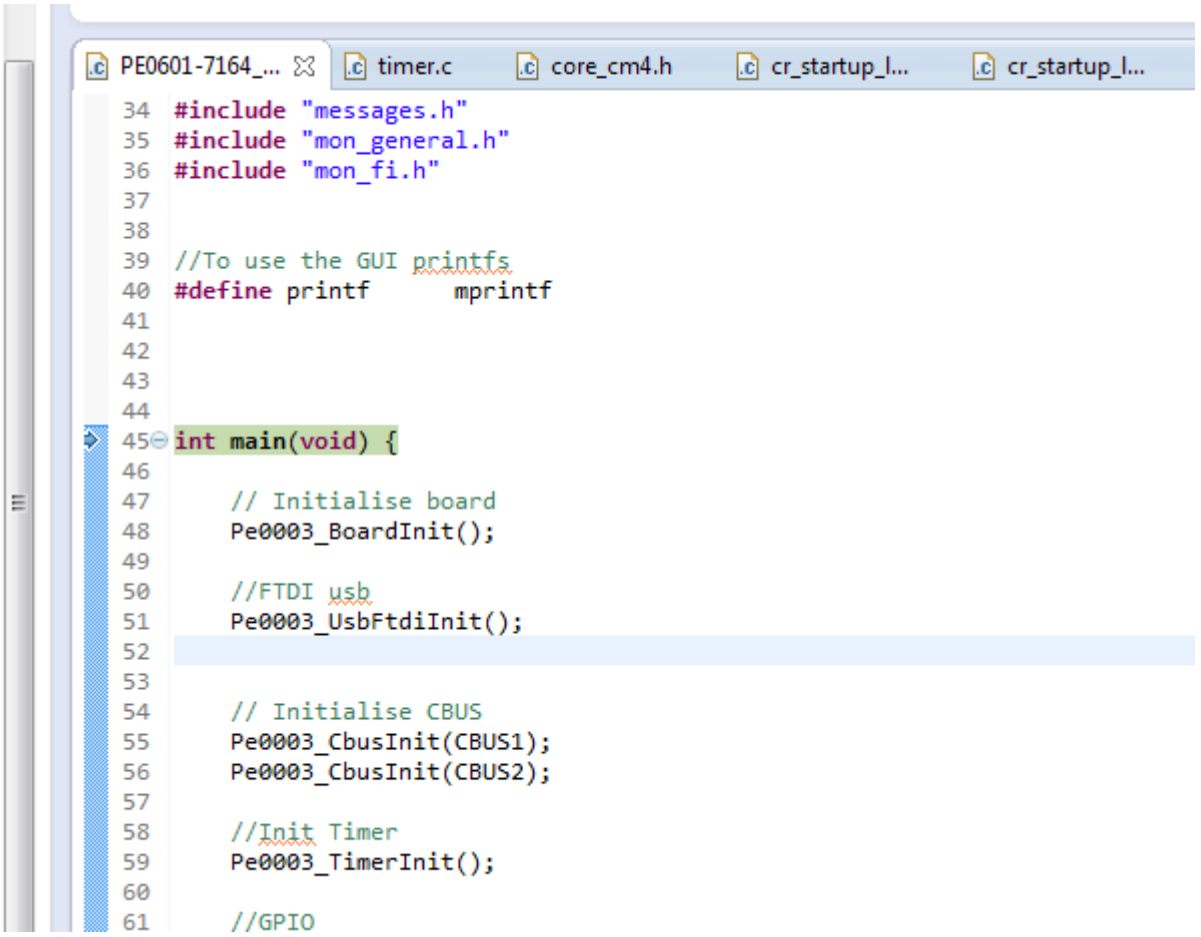
- Click the debug button.

In the figure above, the controls for debugging are displayed: Step into, step over, step through, pause, resume, stop, restart.

Note : If a JTAG Configuration dialogue is displayed when configuring the Debugger then select Device 0 Cortec-M4

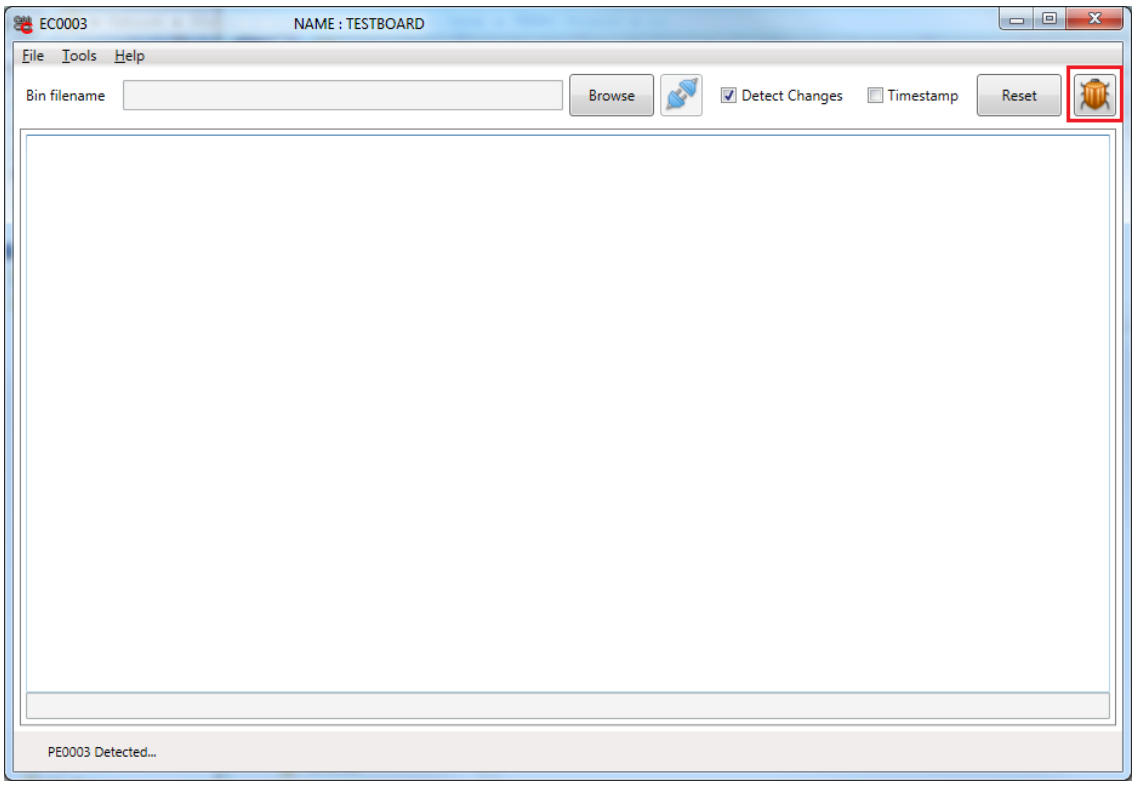


The following image is an example of an application that has been started in debug mode.



EC0003

For using the debugger with the EC0003, the button Debugger button must be switched enabled. This informs the GUI that the firmware has been preloaded.

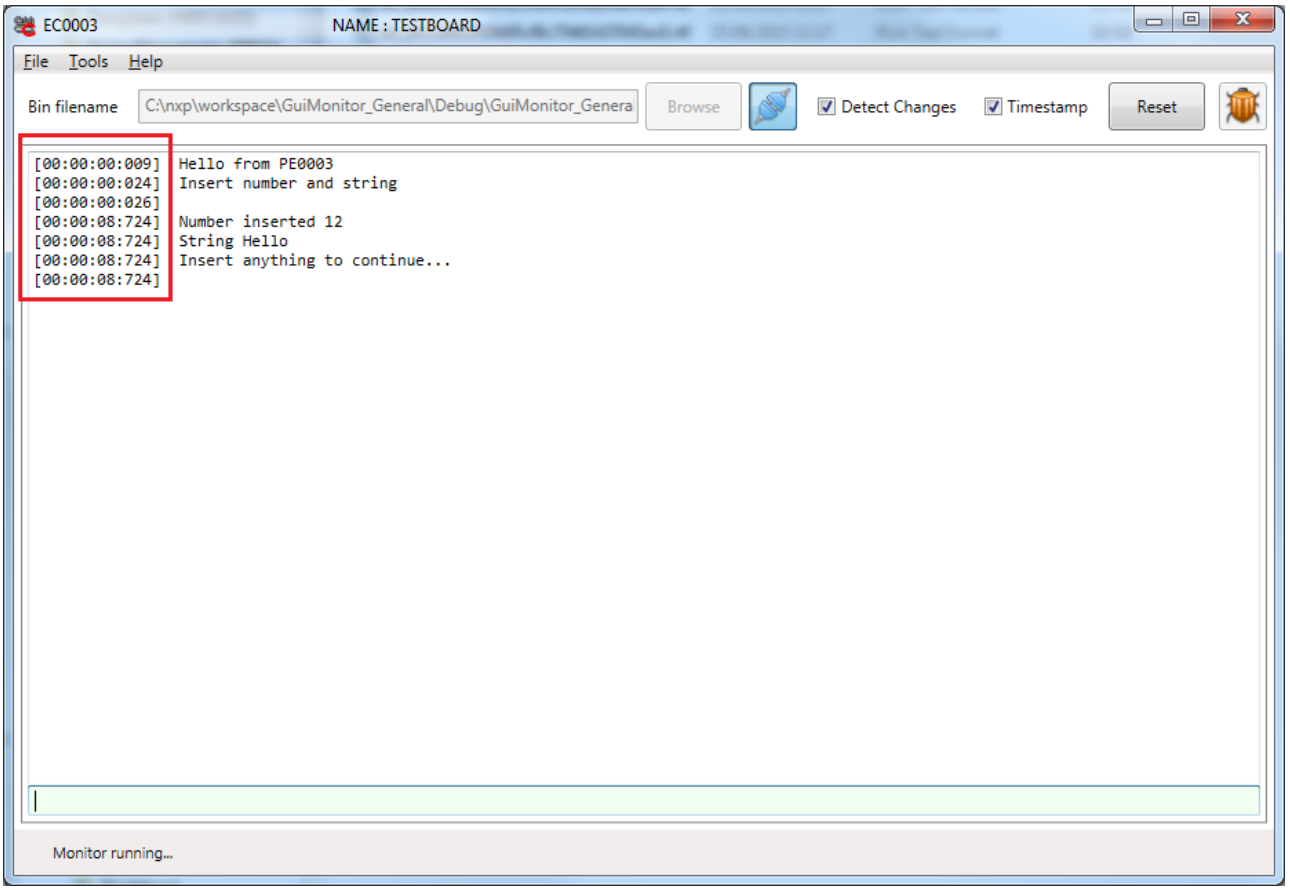


To enable or disable the connection to the PE0003, click the connect button.

To restart the firmware in the board, click the reset button in the toolbar.

7.4 EC0003 Console

If the timestamp checkbox is ticked a timestamp is displayed alongside each new line written to the console.



The File menu gives option for saving the console data to a file or for printing.

7.5 EC0003 Features

The features implemented by the libraries in “CML_UtilsLib” follow: These files should not be modified by users.

mon_general.h – handles basic operations in the EC0003 console such as printf, scanf, clear console and reset.

mon_file.h – used to work with data files.

mon_fi.h – handling FIs using the interface GUI. For loading from SD card or memory directly without GUI or human interaction use fi_tools.h.

mon_enviroment.h – used for reading environmental variables from the GUI.

mon_dialog.h – used to display dialogs and capture input from dialogs.

Monitor.h and messages.h – provides the message exchange system and messages used to communicate with the PE0003. The most important function call to use for any project that needs interaction with EC0003 is:

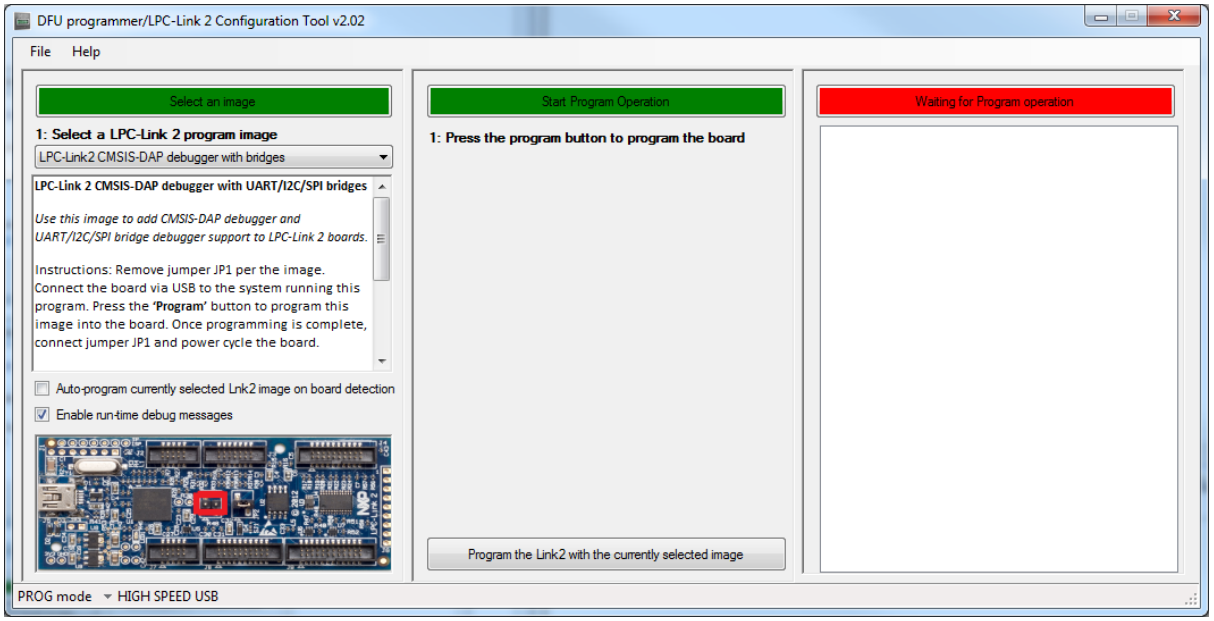
```
//Init GUI
      Cml_GuiSystemInit();
```

Note: Include this function call to enable communication between EC0003 and PE0003 otherwise the EC0003 GUI loads the firmware without interacting with the PE0003.

8 LPC-Link2 Configuration

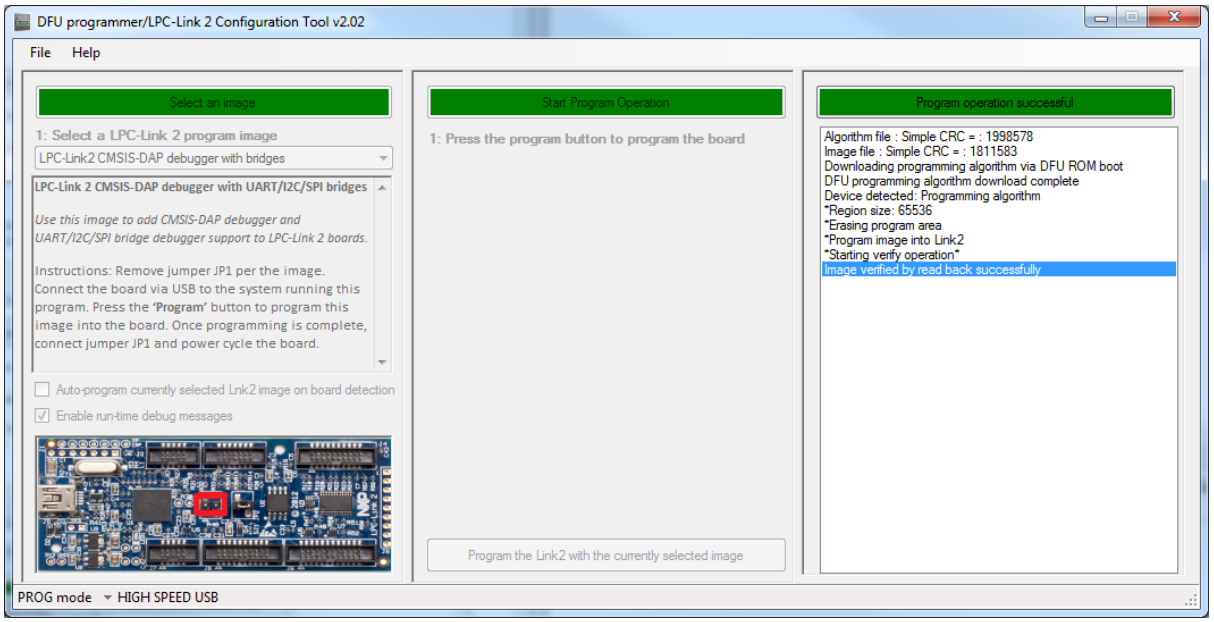
Since the release of LPCXpresso 7.9.2 (build 493) the default firmware was changed to CMSIS-DAP, from Redlink. Due to this a new image has to be downloaded to the LPC-Link 2 board; the "LPC-Link 2 Configuration Tool" provided by NXP can be used for this purpose.

Run the LPC-Link2 Configuration Tool. The LPC-Link2 board should be automatically recognised by the program.

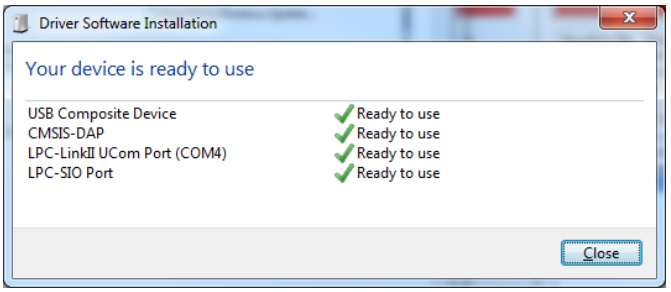


Select the image to program as CMSIS-DAP and follow the instructions provided in the small window (basically program removing the jumper JP1 and connect it once the firmware has successfully loaded). Click the button to Program.

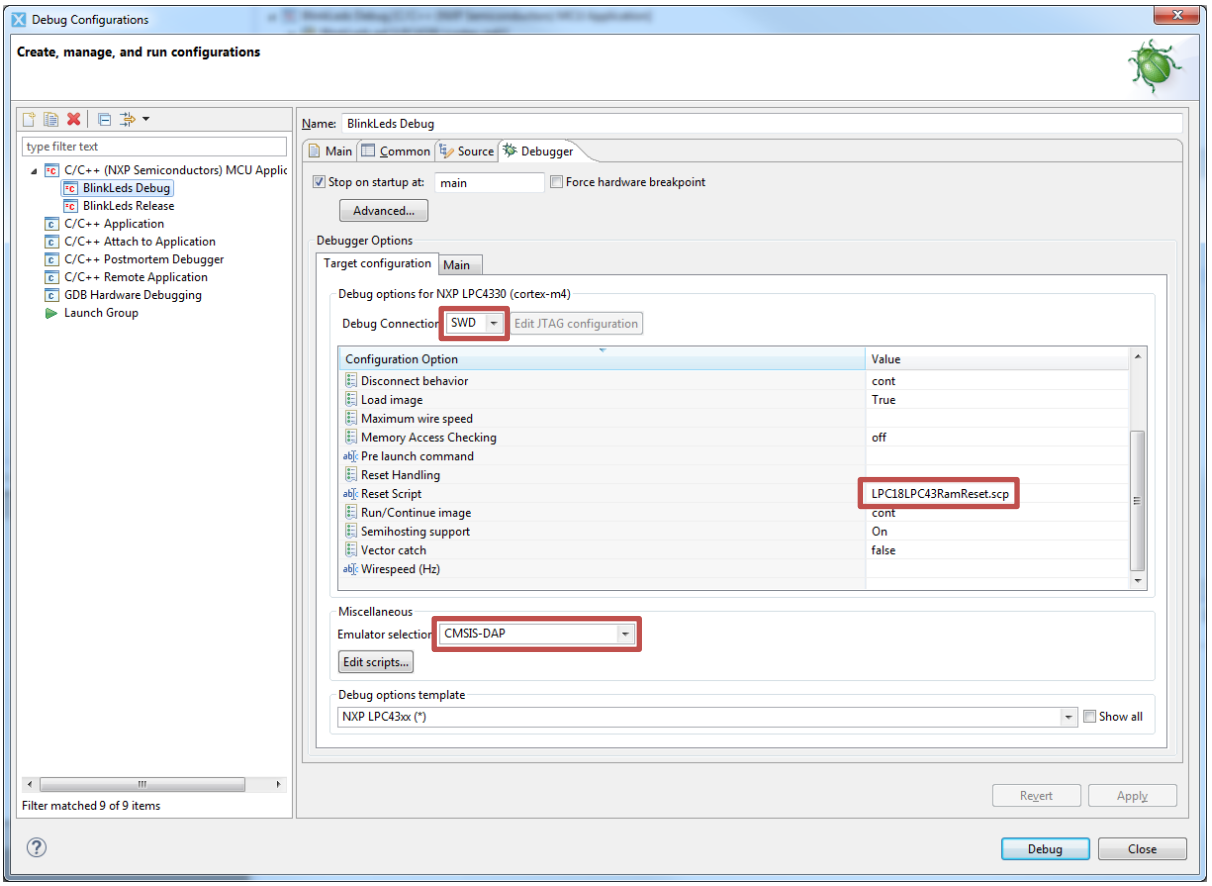
A dialog window similar to that shown below should be displayed if the operation is successful.



Disconnect and reconnect the USB. Windows should automatically start to install the drivers required.

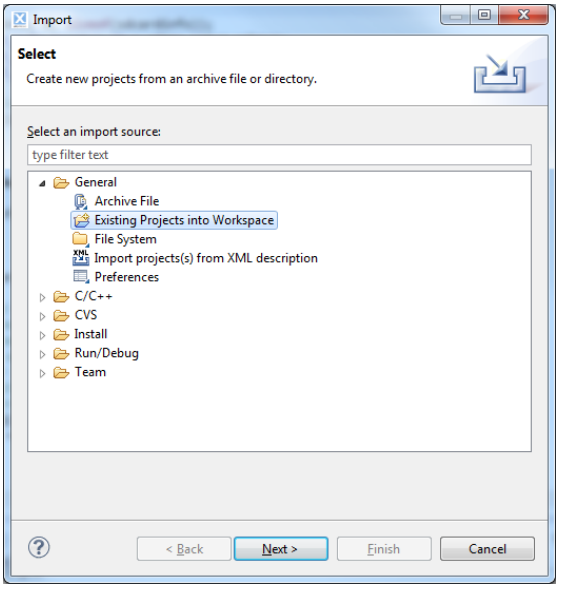


Once installed in your LPCXpresso debug configuration remember to update the options surrounded by red boxes in the following image.



9 Examples

Some example projects are provided with the EC0003 bundle. To work with the examples, run LPCXpresso and create a new workspace for the examples.



Use File → Import to open the Import dialog and under General select “Existing Projects into Workspace”.

Import all the projects from “Documents/CMLMicro/Examples”.

Build All projects.

To return to the original project, switch the workspace to your working space.

To switch workspaces in the LPCXpresso. Go to File → Switch Workspace → Other…

List of examples:

Project	Description
BlinkLeds	Blink the PE0003 LEDs D1-D4
CBusOperation	Write and Read a C-BUS register, using a CMX7x4X or CMX7x3x device
GPIOOperation	Toggle the GPIO lines using a counter incremented every 100µs.
GuiMonitor_Dialogs	Display sample dialogs using the EC0003 Monitor interface.
GuiMonitor_Environment	Configuring and using environment variables set in EC0003.
GuiMonitor_Files	PC File handling.
GuiMonitor_General	Example of mscanf and mprintf to send and receive data to/from the EC0003 GUI. Shows how to set the connection with the EC0003.
GuiMonitor_General2	Another example of scanf and printf but with a non-blocking scanf. A loop runs continuously waiting for a “0” input to exit the loop and continue execution.
InterruptionIRQN	Use of the IRQN1 and IRQN2 interrupts.
PE0402	Loads an FI directly from the chip memory into a PE0402.
PE0402_Gui	Loads an FI from the EC0003 GUI using a dialog box targeting the PE0402
PE0403_7340	Loads an FI directly from the chip memory into a PE0403_7340.
PE0403_7340_Gui	Load an FI from the EC0003 GUI using a dialog box targeting the PE0403_7340
PE0601_7164_Gui	Loads an FI from the PC without a dialog.

10 Known Issues List

There are some unresolved known issues:

- The Reset button may fail to trigger a reset during intensive USB operations. Toggling the Connect button will resolve the problem.
- There can be a delay of around 15 seconds when starting the EC0003 application. This occurs if the PC is connected to a network but does not have internet access. The problem is caused by the .NET framework and does not occur if the PC has internet access or is completely disconnected from the network.
- Dialogue boxes may not be displayed in the expected position if a multiscreen environment is used and the screens have different resolutions.
- From LPCXpresso version 7.6.2 onwards, a problem was found using the NewLib library for dynamic memory allocation. A hardware failure was caused by the use of the memory allocation after some iterations.
- From LPCXpresso version 7.9.2 onwards, the configuration used for LPC Link 2 must be changed to CMSIS-DAP. This requires an update of the firmware for the old LPC-Link2 debugger boards. Use the LPC-Link2 Configuration Tool provided by NXP to set the right firmware.

11 Summary

The PE0003 provides a set of libraries for developing and interacting with CML development kits.
The EC0003 provides an interface system to allow the user to interact with the application.
The Debug system using LPC-LINK2 provides a reliable method to debug user code.

More documentation about the libraries and how to use the tools can be found in section 13.

There are several examples on library use in the CML Bundle

12 Acknowledgments

The icons are kindly provided by Fatcow: http://www.fatcow.com/free-icons.

13 Related Documents

Document	Source	Date
LPCXpresso V7 User Guide	www.lpcware.com	June 2014
PE0003 User Manual	www.cmlmicro.com	August 2014
EC0003 C Driver Libraries CML_UtilsLib	www.cmlmicro.com	July 2015
EC0003 C Driver Libraries PE0003_DriverLib	www.cmlmicro.com	July 2015

CML Microcircuits (UK) Ltd COMMUNICATION SEMICONDUCTORS	CML Microcircuits (USA) Inc. COMMUNICATION SEMICONDUCTORS	CML Microcircuits (Singapore) Pte Ltd COMMUNICATION SEMICONDUCTORS
Tel: +44 (0)1621 875500 **Fax:** +44 (0)1621 875600 **Sales:** sales@cmlmicro.com **Tech Support:** techsupport@cmlmicro.com	**Tel:** +1 336 744 5050 800 638 5577 **Fax:** +1 336 744 5054 **Sales:** us.sales@cmlmicro.com **Tech Support:** us.techsupport@cmlmicro.com	**Tel:** +65 62 888129 **Fax:** +65 62 888230 **Sales:** sg.sales@cmlmicro.com **Tech Support:** sg.techsupport@cmlmicro.com
- www.cmlmicro.com -		