
NXP Semiconductor

Getting started with LPCOpen: Running the demo applications

By: Technical Information Center

About this document

This document will explain the steps to use LPCXpresso with the LPCOpen projects for your preferred device and platform.

The steps described in the document were done using the LPC54102 MCU like the one in the LPCXpresso Board for the LPC54100 family of MCUs, but the same principles are applicable to any LPC MCU.

Software versions

The steps described in this document are valid for the following versions of the software tools:

- LPCXpresso v8.1.4
- LPCOpen v3.xx

Contents

1. Overview and concepts.....	3
1.1 LPCOpen.....	3
1.1.1 Core driver library	3
1.1.2 Middleware	3
1.1.3 Examples	3
1.1.4 Using LPCOpen with an RTOS	3
2. Running the demo applications	4
2.1 Downloading a LPCOpen package.....	4
2.2 Importing the LPCOpen examples	6
2.3 Building and debugging blinky project.....	11
Appendix A - References.....	16

1. Overview and concepts

1.1 LPCOpen

LPCOpen is an extensive collection of free software libraries (drivers and middleware) and example programs that enable developers to create multifunctional products based on LPC microcontrollers. Access to LPCOpen is free to all LPC developers.

1.1.1 Core driver library

The core driver library contains common chip-specific drivers. It is divided into two layers: a chip driver layer containing drivers optimized for a specific device or family, and a board layer containing board-specific functions and low-level setup code.

1.1.2 Middleware

LPCOpen includes access to key middleware elements:

- SEGGER emWin graphics object library
- SWIM graphics library
- LWIP open-source networking stack - source code and examples
- USB libraries: USB device library for all LPC devices and LPCUSBLib open-source USB host stack - both use the USB ROM APIs or a Flash-based library

1.1.3 Examples

LPCOpen includes an extensive set of examples designed to illustrate how to use core driver library functions and middleware. Examples demonstrate use of:

- Peripherals such as I2C, UART, SPI, and GPIO
- USB host and device
- Ethernet use with an IP stack (LWIP)
- emWIN and SWIM graphics libraries

1.1.4 Using LPCOpen with an RTOS

LPCOpen libraries are RTOS agnostic and can be used with a simple control loop. Examples are also included in each software download package for use with FreeRTOS.

2. Running the demo applications

2.1 Downloading a LPCOpen package

- Got to the link <http://www.nxp.com/lpcopen> and select the corresponding family of microcontrollers, for this example we chose the **LPC54100** series:

LPCOpen ports for LPC Cortex-M series microcontrollers

- LPC800 Series
- LPC1100 Series
- LPC1300 Series
- LPC1500 Series
- LPC1700 Series
- LPC1800 Series
- LPC4000 Series
- LPC4300 Series
- **LPC54100 Series**
- LPC54110 Series

- On the next page you will find the latest available LPCOpen software package downloads along with older versions of the packages. Look for your board/device and click on the software download link, for this example we chose the package for the **LPCXpresso IDE** but there is also a package for the IAR and Keil IDEs:

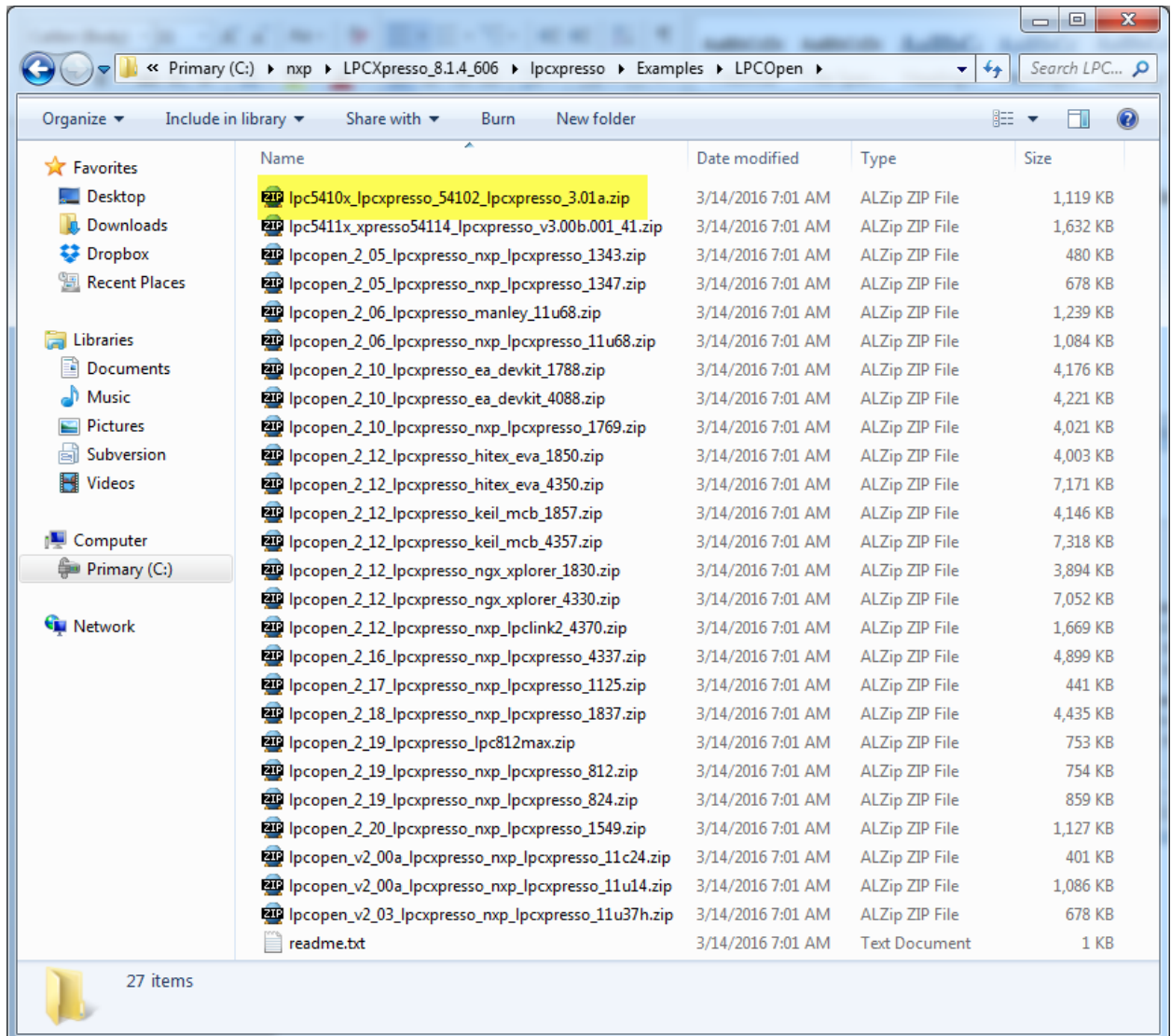
Latest available LPCOpen v3.xx Legacy LPCOpen 2.xx Older versions of LPCOpen 2.xx

Latest available LPCOpen v3.xx software package downloads

Supported Board(s) /Devices(s)	Software Download link	Toolchain ¹	Documentation download link ²	Debugger(s) ³	Related downloads	History
LPCXpresso LPC54102 board	v3.01a.000 Release Date 08/05/2015	LPCXpresso v7.9.0	Windows help file (chm)	Redlink / CMSIS-DAP	Link2 Windows driver	History
	v3.01.000 Release Date 08/04/2015	IAR EWARM 7.40.3 Keil MDK- ARM v5.15	PDF Document API migration (ROM to LPCOpen)	CMSIS-DAP		

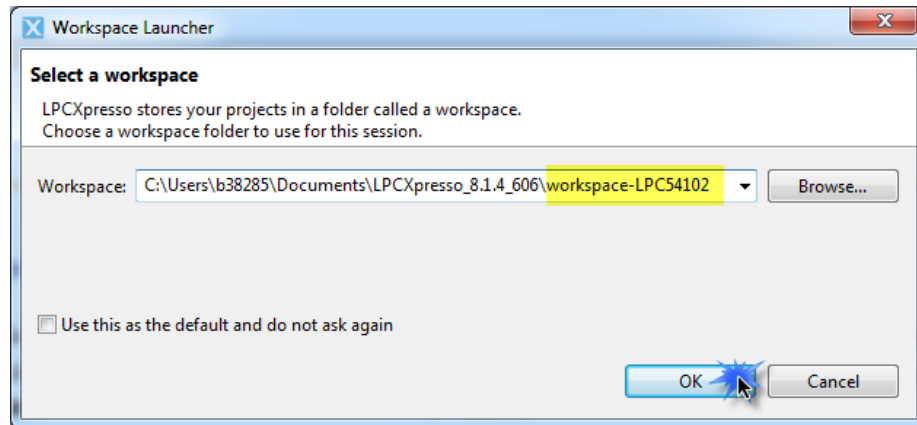
- The LPCOpen packages can also be found on the LPCXpresso installation folder, just make sure they are the latest package available:

C:\nxp\LPCXpresso_8.1.4_606\lpcxpresso\Examples\LPCOpen

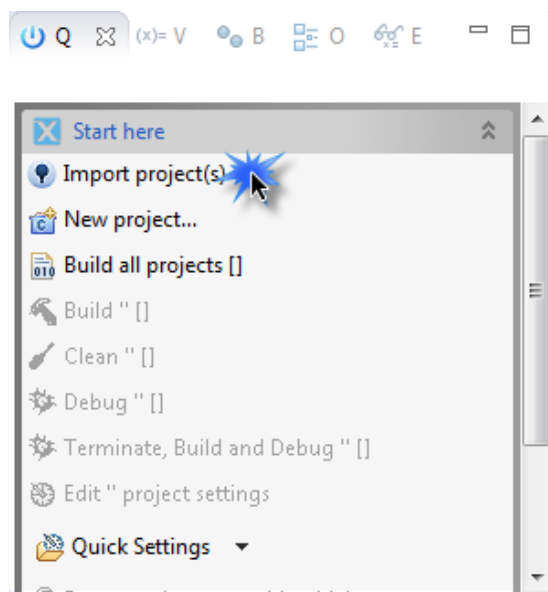


2.2 Importing the LPCOpen examples

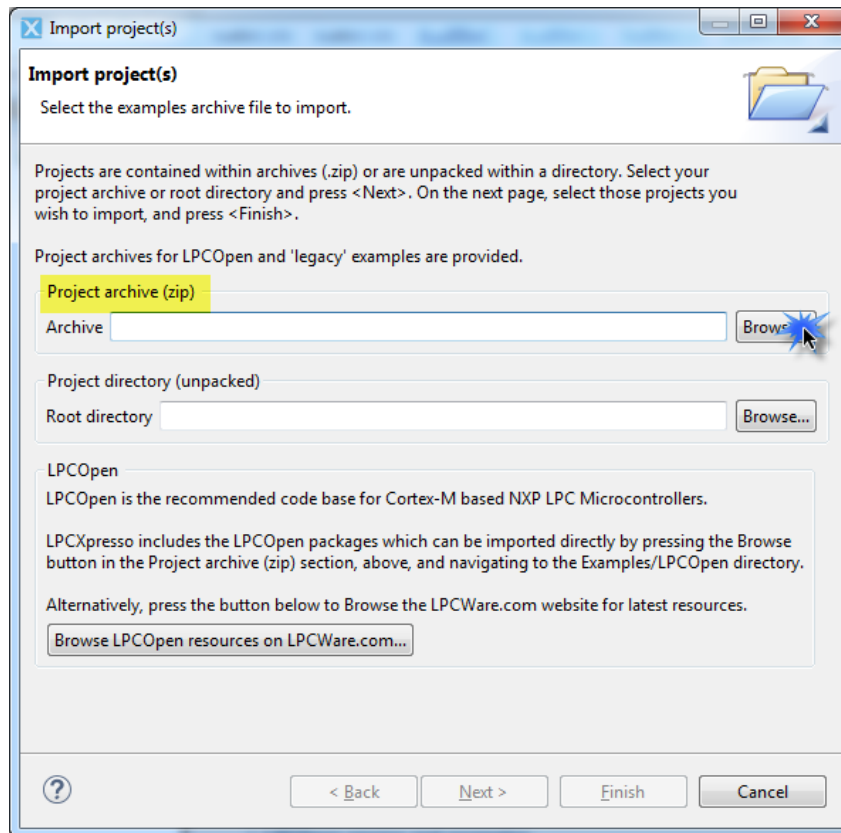
- Open **LPCXpresso**, create a new workspace and click on **OK**:



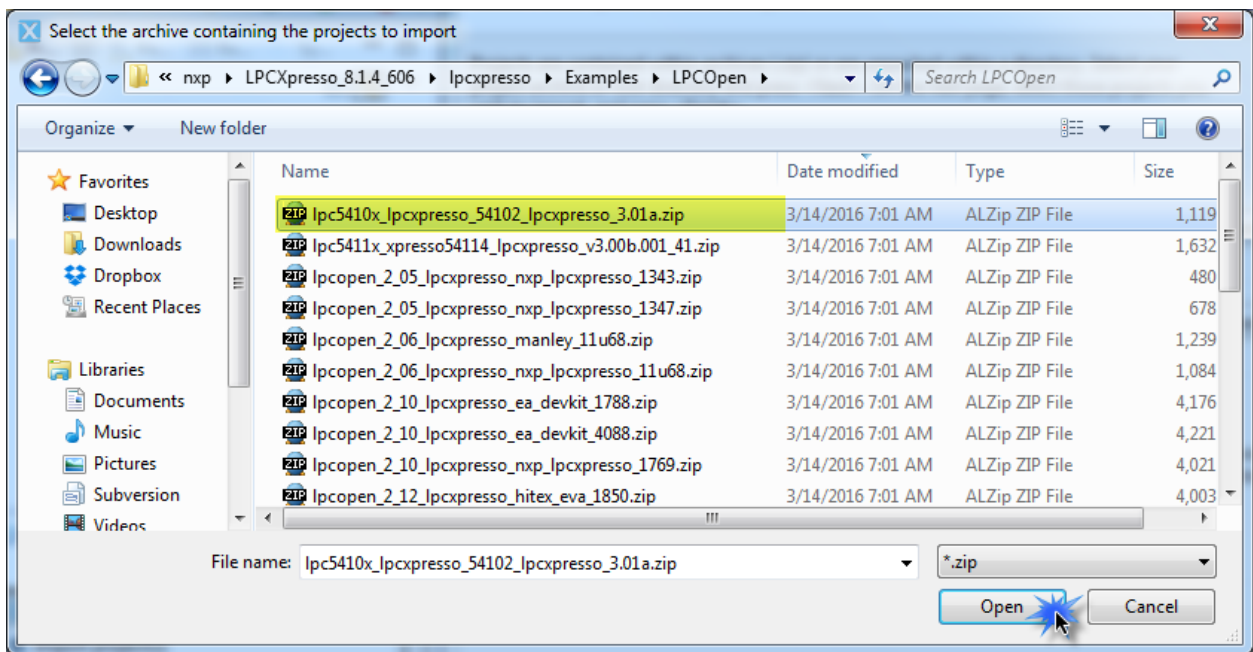
- On the **Quick Start** panel click on **Import project(s)**:



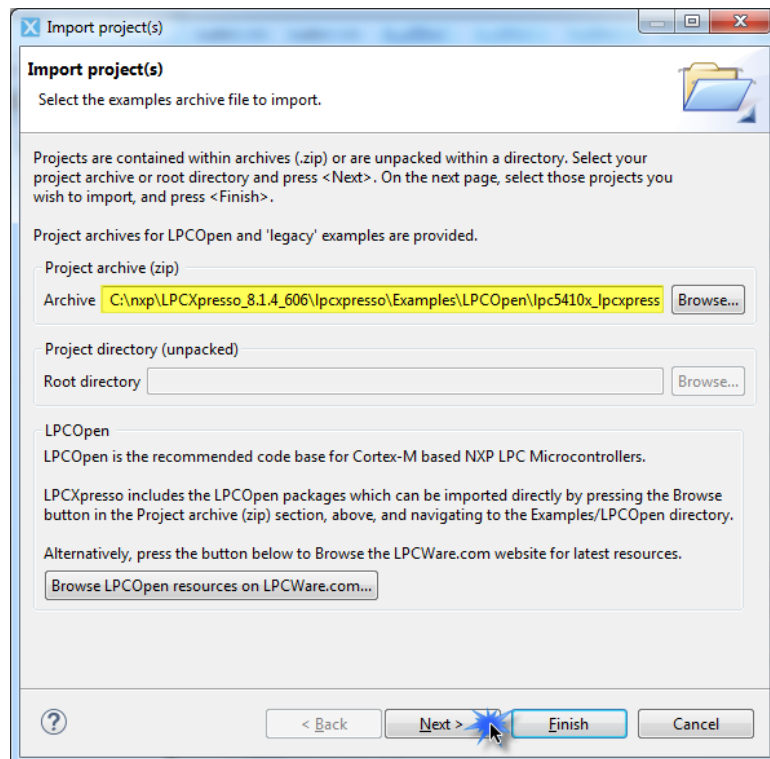
- The **Import project(s)** wizard will open. There are two ways to import projects, import projects contained in archives or unpacked projects, for this example chose the **Project archive (zip)** option and click on **Browse...:**

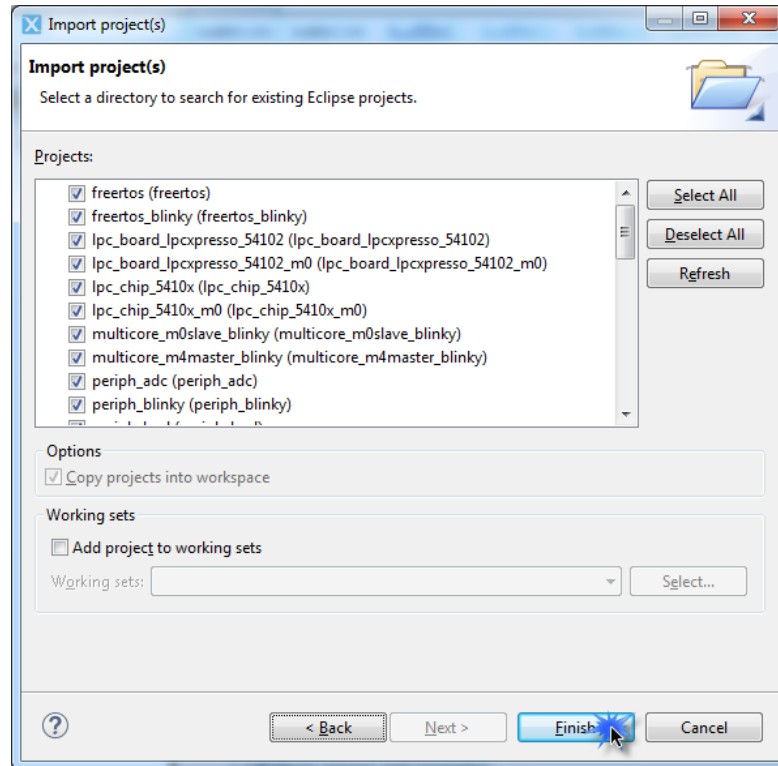


- Search for the downloaded *.zip file or locate the package included on the LPCpresso installation, select it and click on **Open**:

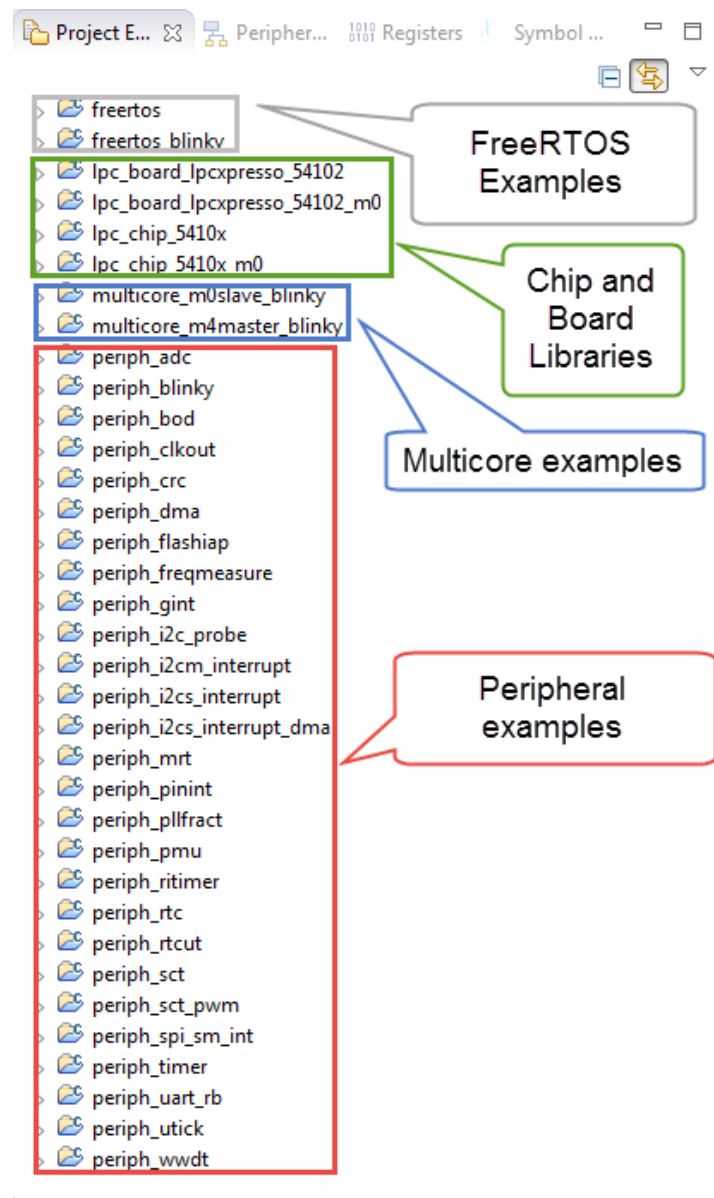


- Click on **Next** and then **Finish**:



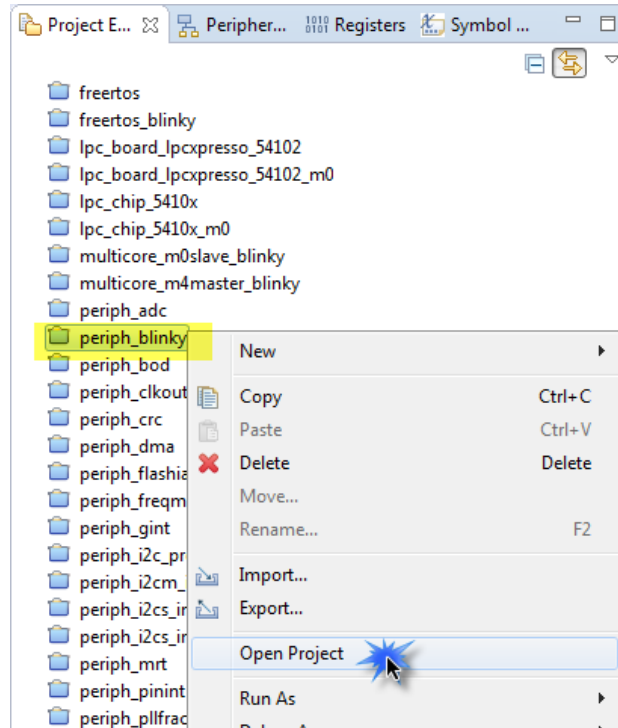


- The example projects along with the chip and board libraries are now shown on the **Project Explorer** window:

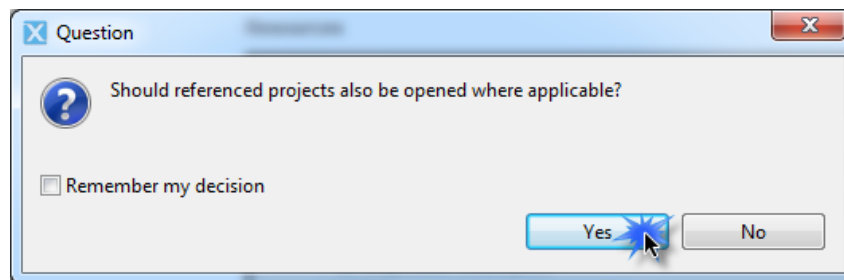


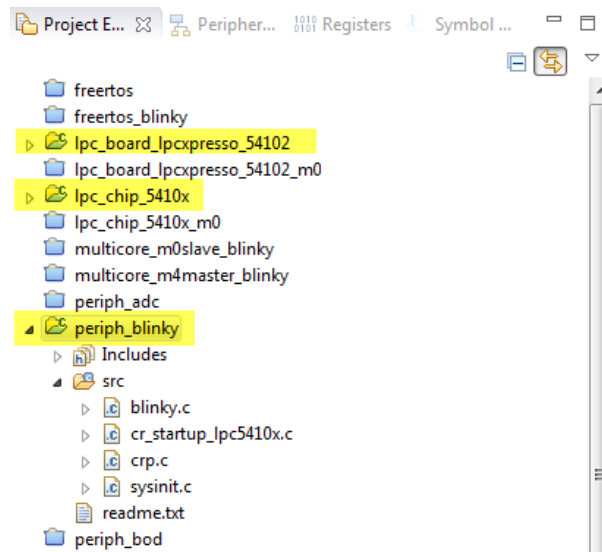
2.3 Building and debugging blinky project

- After importing the projects to the workspace these can be closed to avoid having all of them opened in the workspace. Then we can open only one of them, for this example we will use the **periph_blinky** project. Right click on the project and click on **Open**:

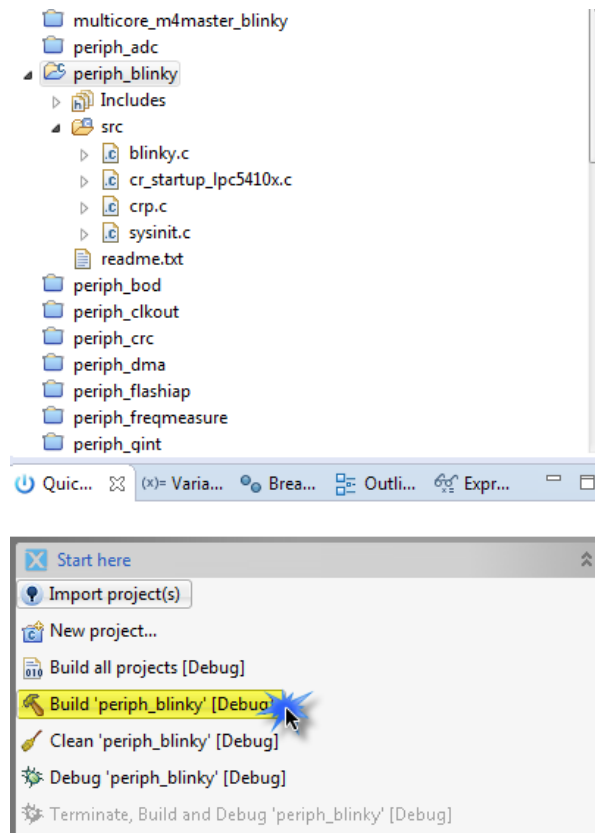


- A pop-up window will appear asking if the referenced projects should also be opened, click on **Yes**, you should see the **periph_blinky** project opened along with the M4 chip and board libraries:

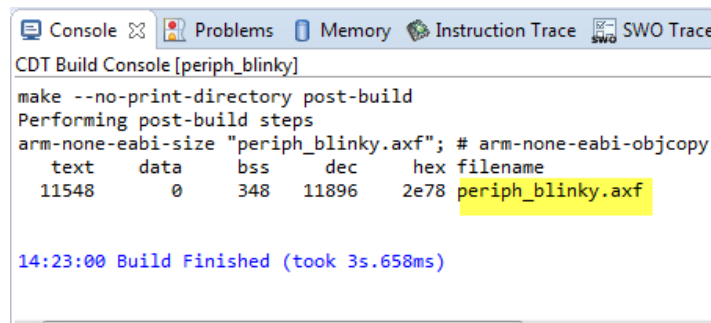




- The next step is to build the project, select the project, go to the **Quick start panel** and click on **Build 'periph_blinky' [Debug]**:



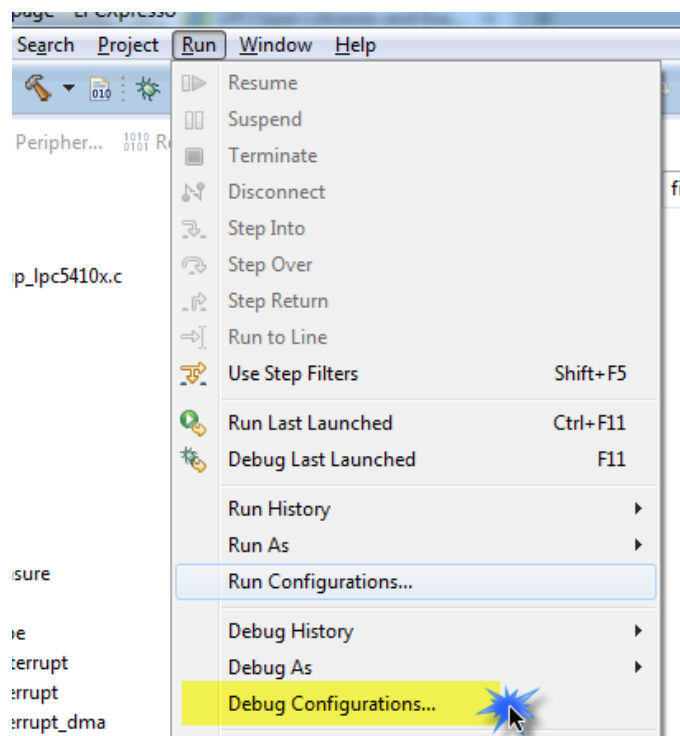
- The corresponding chip and board libraries will be built and the file **periph_blinky.axf** generated:



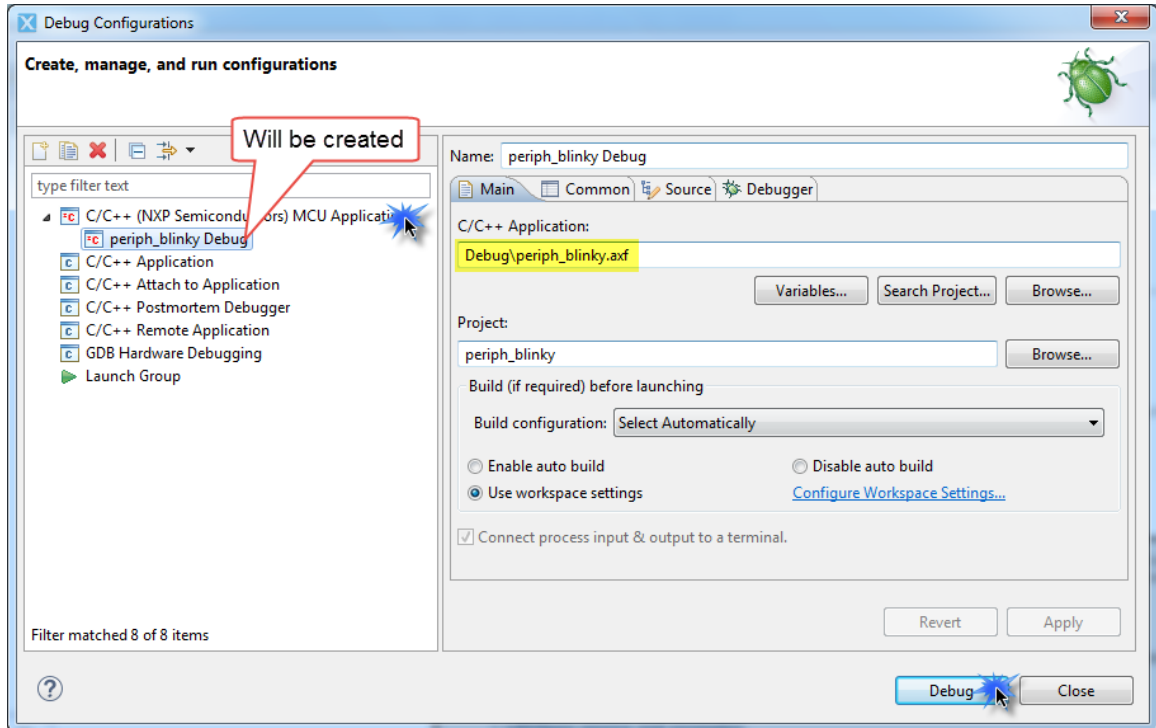
```
CDT Build Console [periph_blinky]
make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "periph_blinky.axf"; # arm-none-eabi-objcopy
text  data  bss  dec  hex  filename
11548  0     348  11896  2e78  periph_blinky.axf

14:23:00 Build Finished (took 3s.658ms)
```

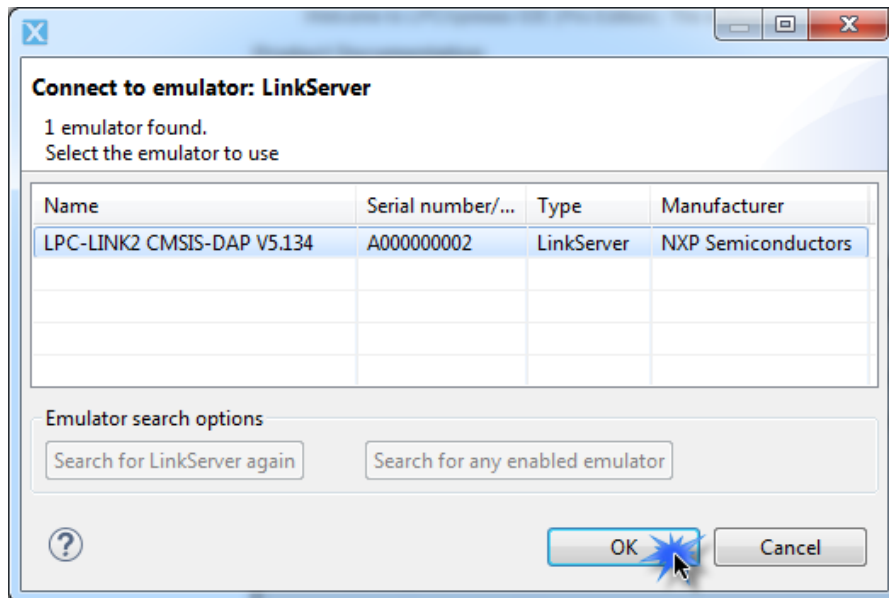
- Now select the project and go to menu **Run > Debug Configurations**:



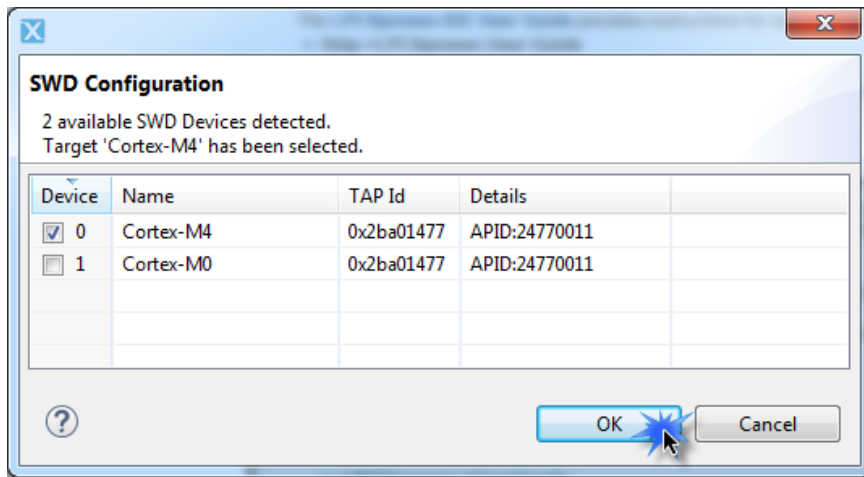
- The **Debug Configurations** window will open, double click on the “**C/C++ (NXP Semiconductors) MCU Applications**” option to create a new connection and program the MCU through the On-board Link2 debug probe and click on **Debug**:



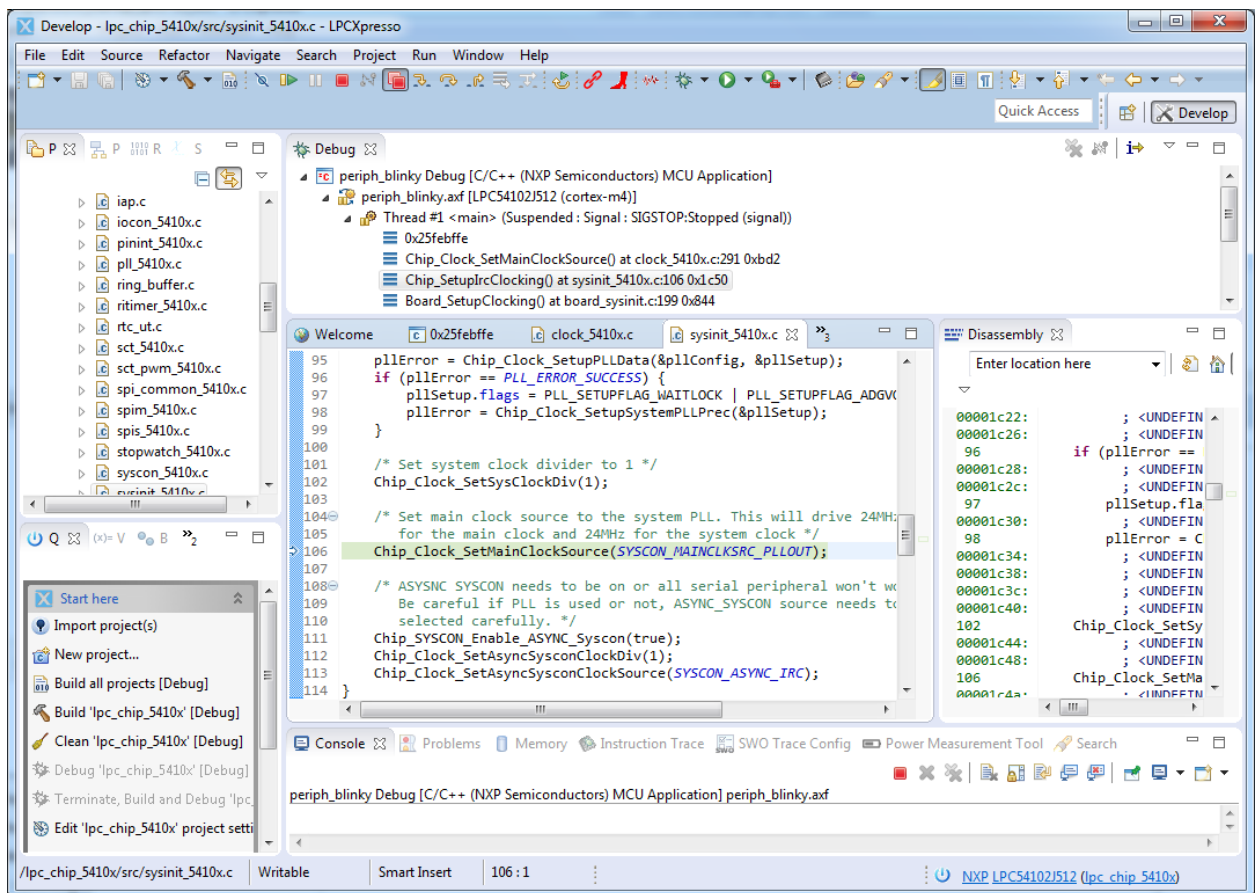
- A new window will show up and it will display the connected emulators, select the **LPC-LINK2 CMSIS-DAP** emulator and click on **OK**:



- After this a new message will show up, indicating that 2 SWD devices were found, select the **Cortex-M4** device which is the one used by the example:



- You should now be able to **debug** the program and **step** through the code:



Appendix A - References

- LPCOpen webpage:
<http://www.nxp.com/lpcopen>
- LPCXpresso webpage:
<http://www.nxp.com/lpcxpresso>
- LPCXpresso Board for the LPC54100 family of MCUs:
<http://www.nxp.com/products/software-and-tools/hardware-development-tools/lpcxpresso-boards/lpcxpresso-board-for-the-lpc54100-family-of-mcus:OM13077>