NXP Semiconductor

Getting started with the LPCOpen Ethernet Examples (lwip_tcpecho and webserver)

By: Technical Information Center

About this document

This document will explain how to build and run the LPCOpen Ethernet example projects, it also explains the needed board and PC connections and configurations.

The steps described in the document were done using the LPC1769 MCU like the one in the LPCXpresso board for LPC1769 with CMSIS DAP probe, 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 v2.xx

Boards

- LPCXpresso board for LPC1769 with CMSIS DAP probe
- EA LPCXpresso BaseBoard
- LPC-Link2

Contents

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: USBD 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 LPCOpen with an RTOS

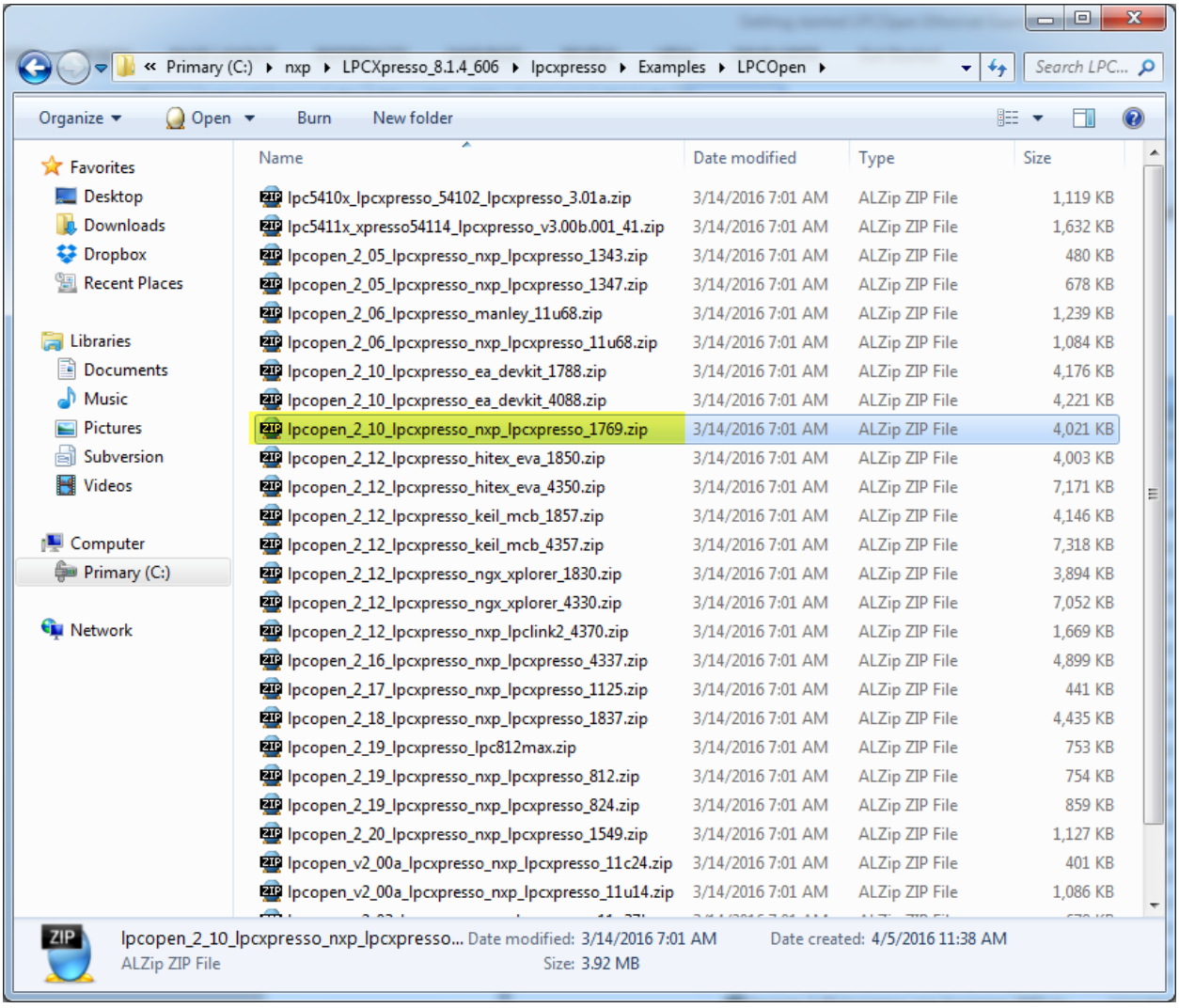
LPCOpen libraries are RTOS agnostic and can be used with a simple control loop. Examples are also included in each software download package for use with FreeRTOS.

2. Running the lwip_tcpecho and webserver demo applications

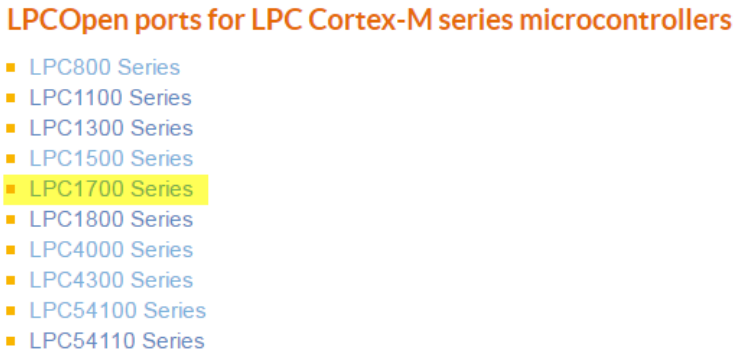
2.1 Downloading a LPCOpen package

- The LPCOpen packages can be found on the LPCXpresso installation folder, just make sure the package to be used is the latest one available:

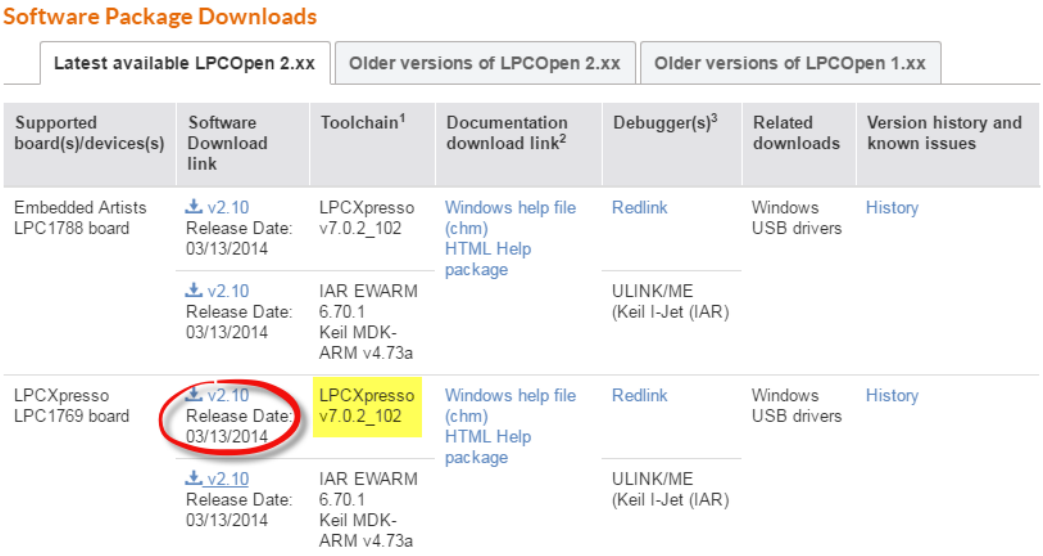
  C:\nxp\LPCXpresso_8.1.4_606\lpcxpresso\Examples\LPCOpen



- In case it is not the latest package available go to the link http://www.nxp.com/lpcopen and select the corresponding family of microcontrollers, for this example we chose the **LPC1700** 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:

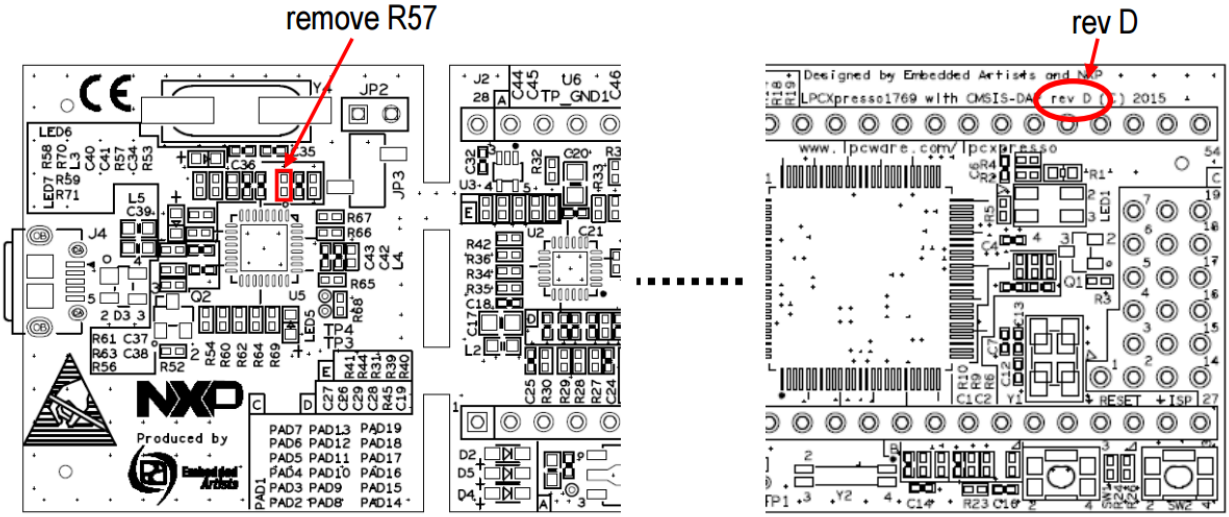


2.3 Setting up the hardware

Based on the readme information on the examples, the LPCXpresso LPC1769 board needs to be connected with base board in order to use the RS232/UART and Ethernet ports. These are the connections and configurations needed for the boards to work correctly.

2.3.1 LPCXpresso board for LPC1769 with CMSIS DAP probe

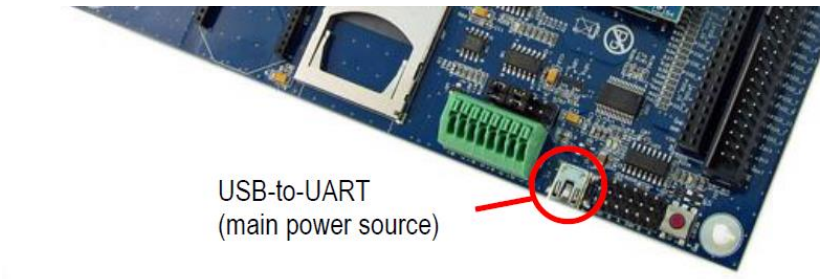
- There is a known issue with the LPCXpresso1769/CD board rev D (rev D1 boards are not affected). Debugging works when the LPCXpresso1769/CD board is standalone but stops working when mounted on a base board, it also causes that a "CRP DISABLD" USB drive appears on the PC instead of the debug interface. This happens because the debug MCU (LPC11U35) will start-up in ISP mode if an external reset signal is low during power-up. This is a typical situation if there is a reset generator on the base board. So the workaround is to remove R57 on the LPCXpresso1769/CD board. This will disconnect the ISP enable signal on the LPC11U35 from the external reset signal.



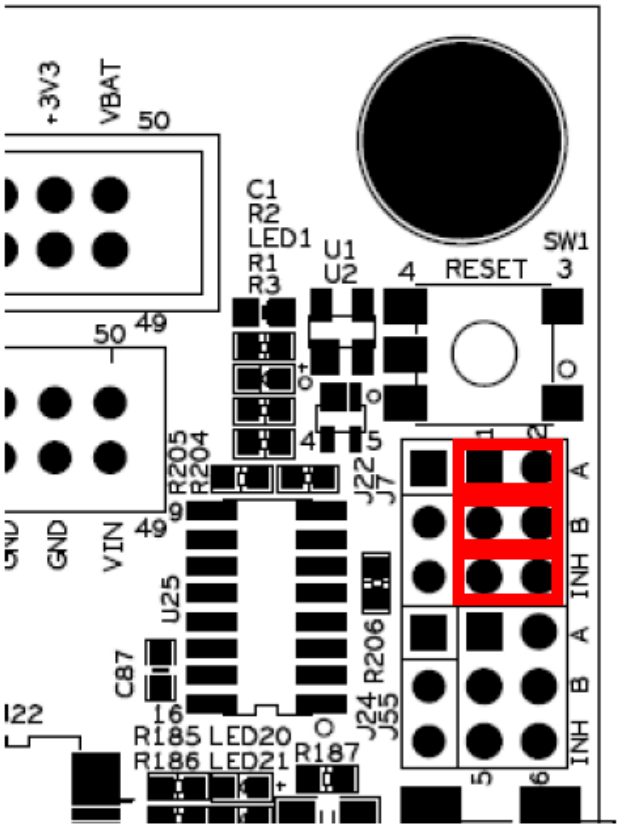
- Since the LPC1769 board will be powered by the baseboard the LPC-Link2 board will be used to program and debug the demo applications. In order to use an external debugger to program the LPC1769 MCU, the JP3 connector needs to be shorted to disable the CMSIS-DAP interface and allow an external debugger to be connected.

2.3.2 EA LPCXpresso BaseBoard

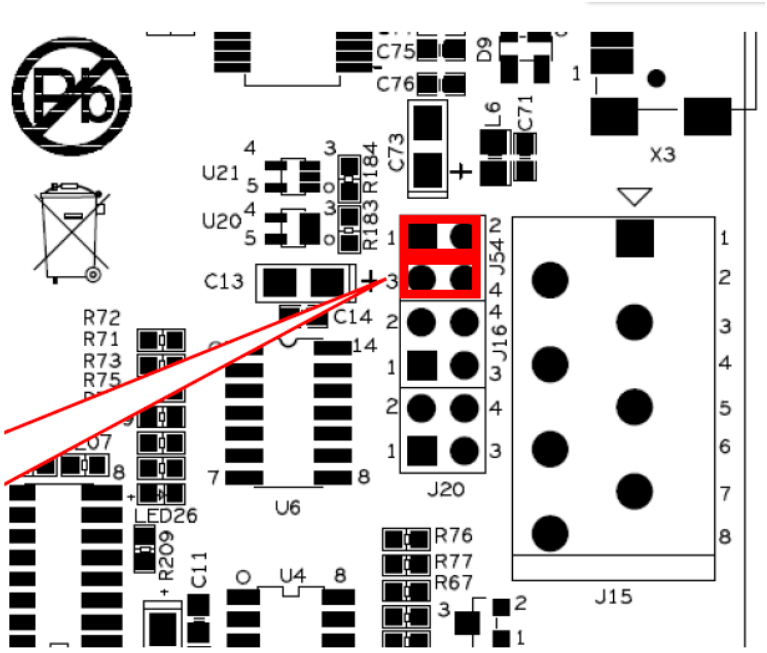
- The main power source will be used to power the system and will also be used as USB-to-UART interface:



- The LPC1769 UART can be connected to one of three different peripherals in the EA LPCXpresso BaseBoard, for this example we will use the USB-to-UART Bridge (U22) which is the default mode, all the three jumpers in J7 need to be inserted:

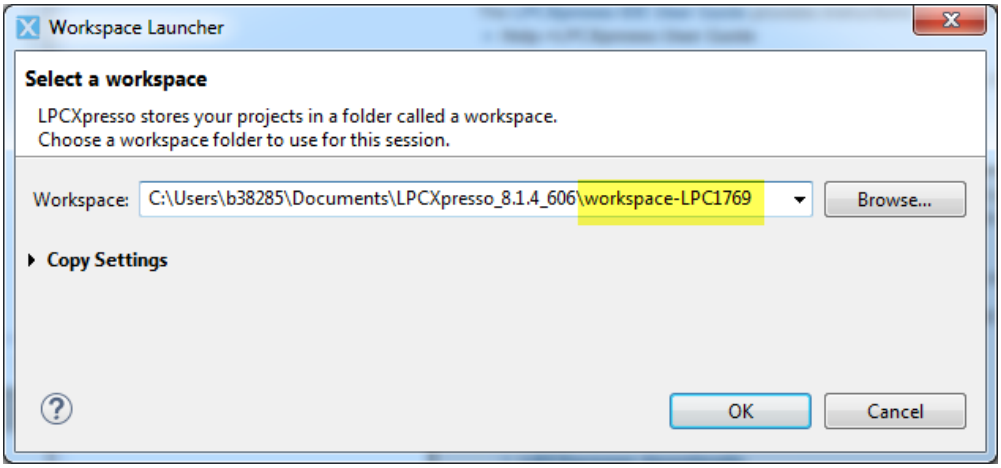


- Remove the J54 jumpers, if these are not removed these cause the board to be always on reset. This happens when using a terminal application, such as Tera Term, that is connected to the COM port associated with the board and at the same time have the J54 jumpers inserted, the terminal application might be the cause of this problem. If the terminal application is pulling DTR low the board will be reset and if it pulls RTS low it will control the bootload enable functionality.

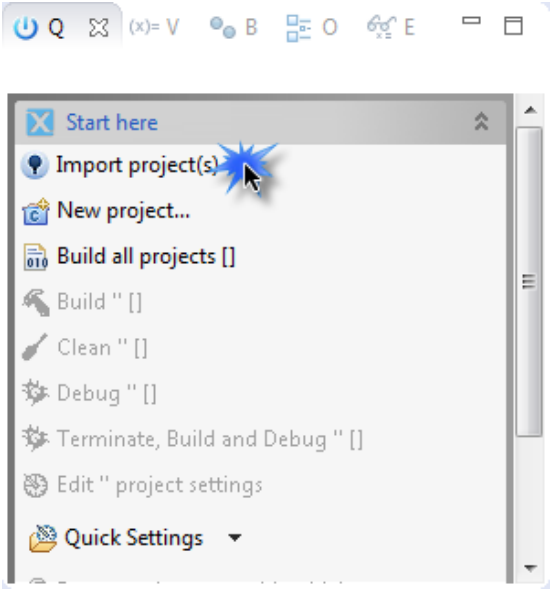


2.2 Importing the LPCOpen examples

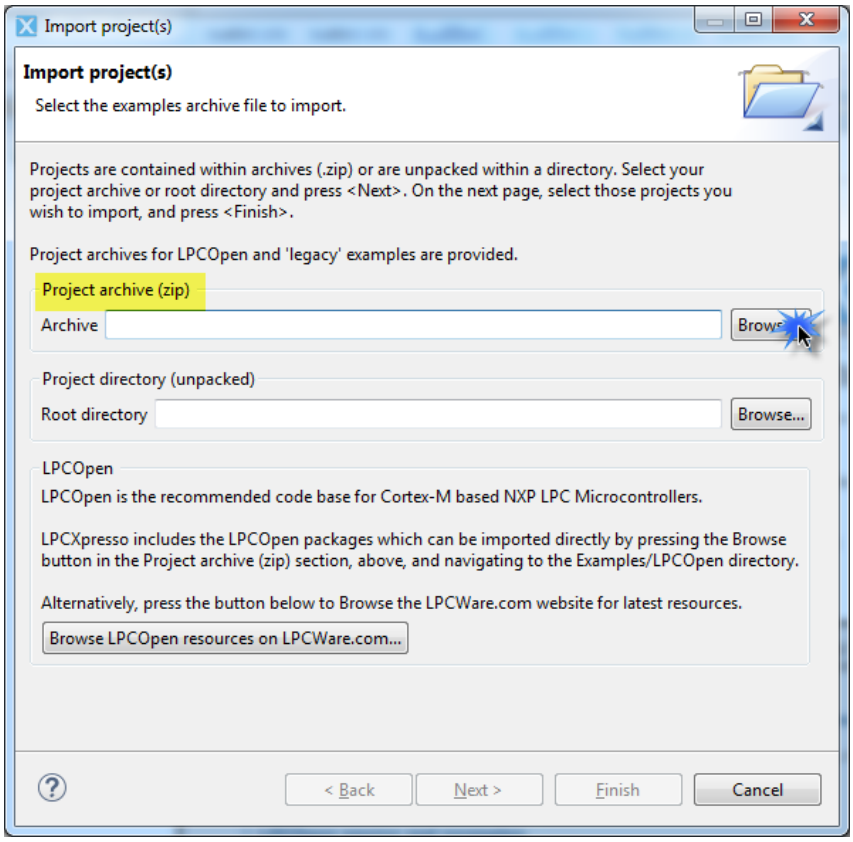
- Open **LPCXpresso**, create a new workspace and click on **OK**:



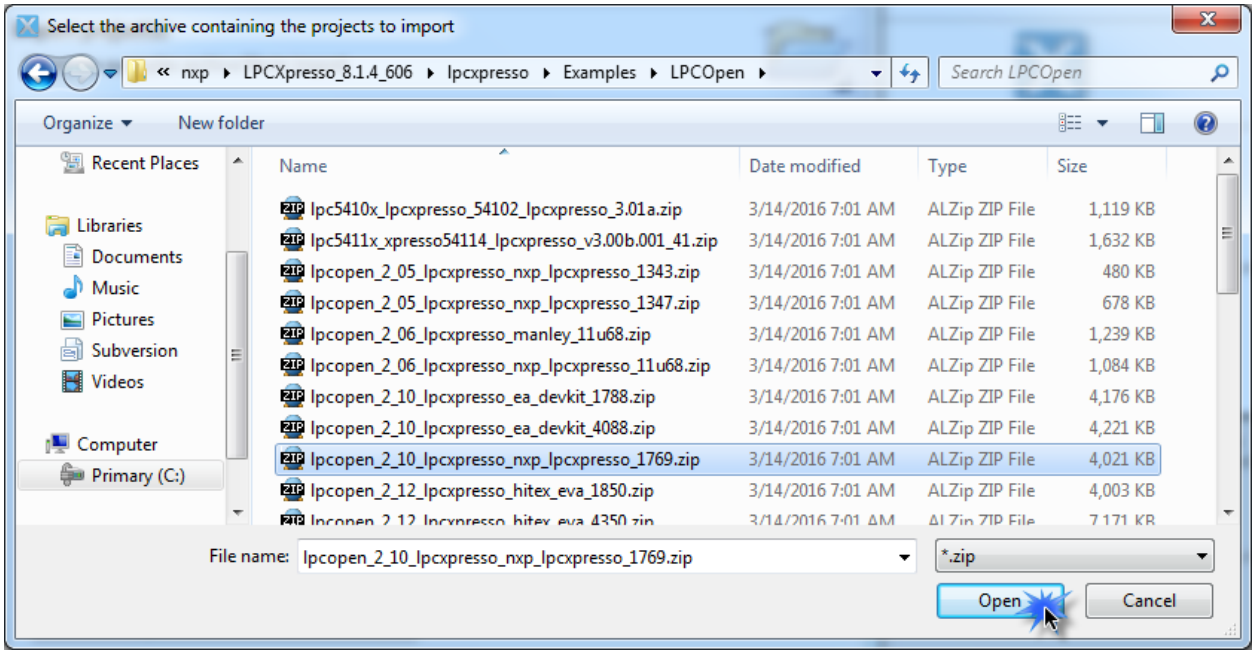
- On the **Quick Start panel** click on **Import project(s)**:



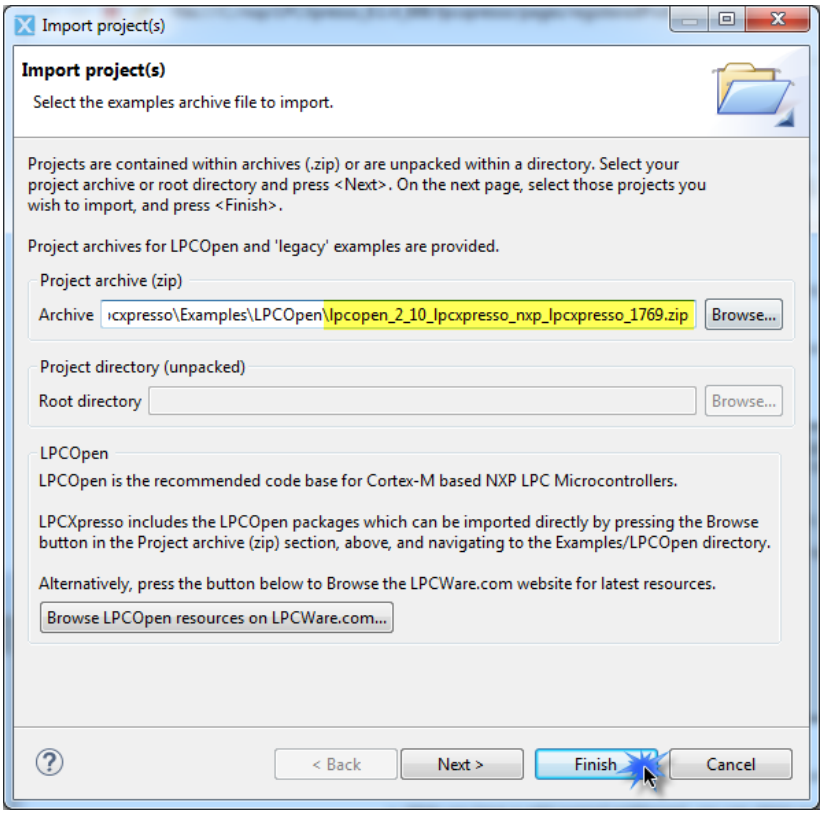
- The **Import project(s)** wizard will open. There are two ways to import projects, import projects contained in archives or unpacked projects, for this example chose the **Project archive (zip)** option and click on **Browse…**:



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

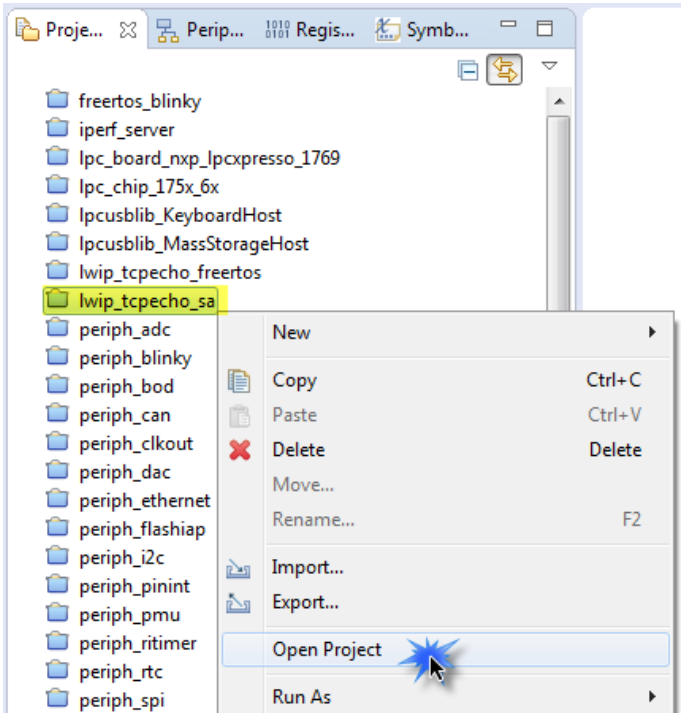


- Click on **Finish**:

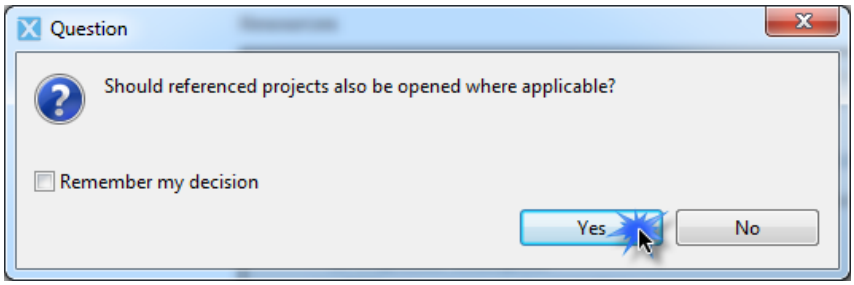


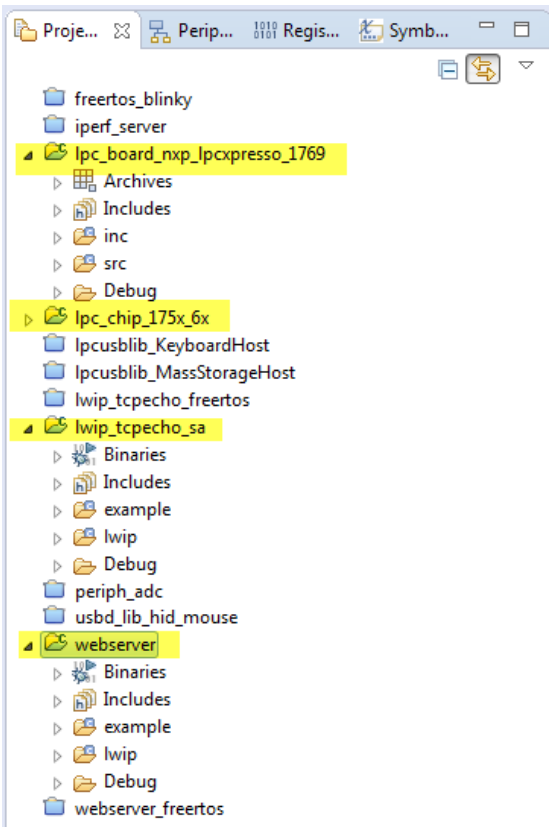
2.4 Building the demo applications

- 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 the **lwip_tcpecho** and **webserver** demos. Right click on the projects and click on **Open**:

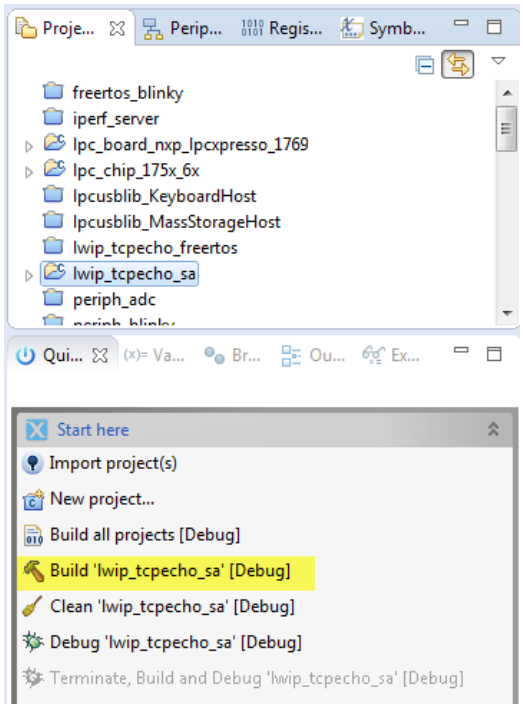


- A pop-up window will appear asking if the referenced projects should also be opened, click on **Yes**, you should see the projects opened along with the M4 chip and board libraries:

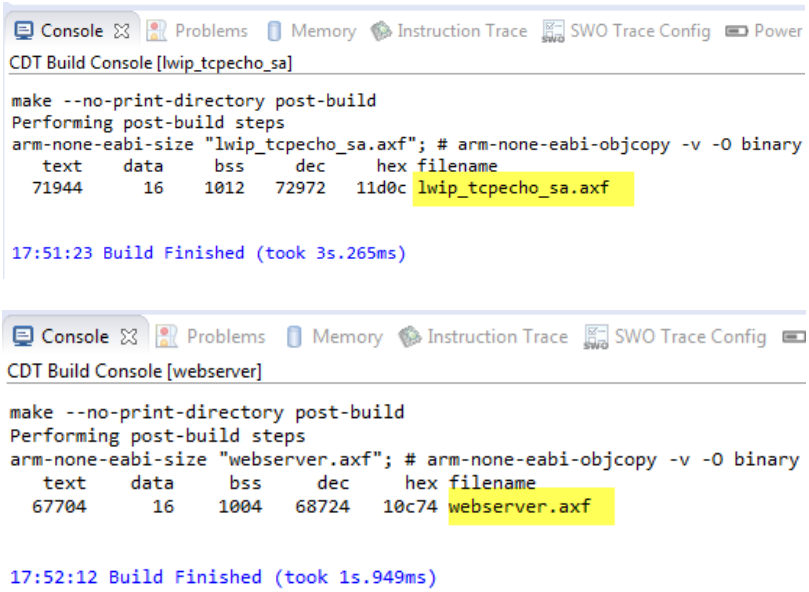




- The next step is to build the projects, select the project, go to the Quick start panel and click on Build 'lwip_tcpecho_sa' [Debug] and Build 'webserver' [Debug] respectively:



- The corresponding chip and board libraries will be built and the files lwip_tcpecho_sa.afx and webserver.afx will be generated:

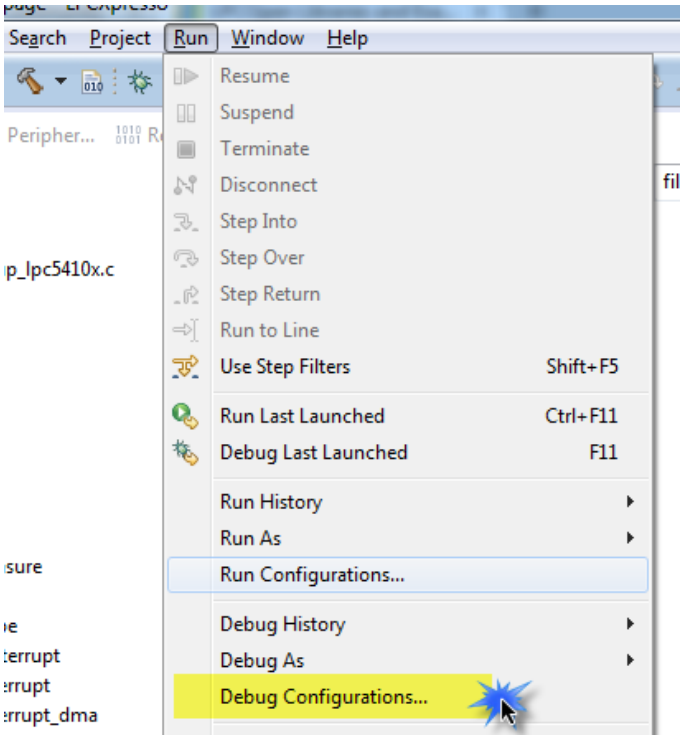


2.5 Running the demo applications

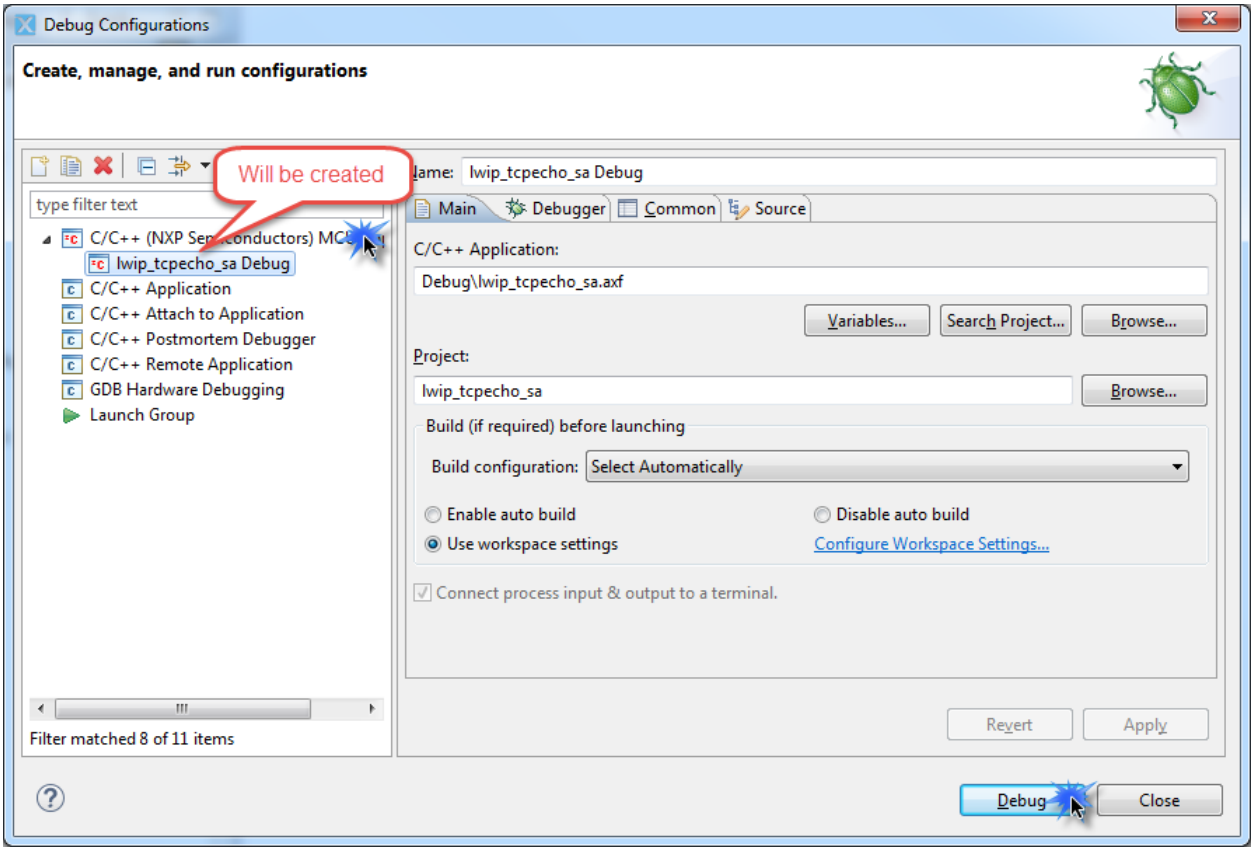
Both of the demos require the board and laptop to be connected to a router since the board will acquire an IP address via DHCP.

2.5.1 lwip_tcpecho_sa demo

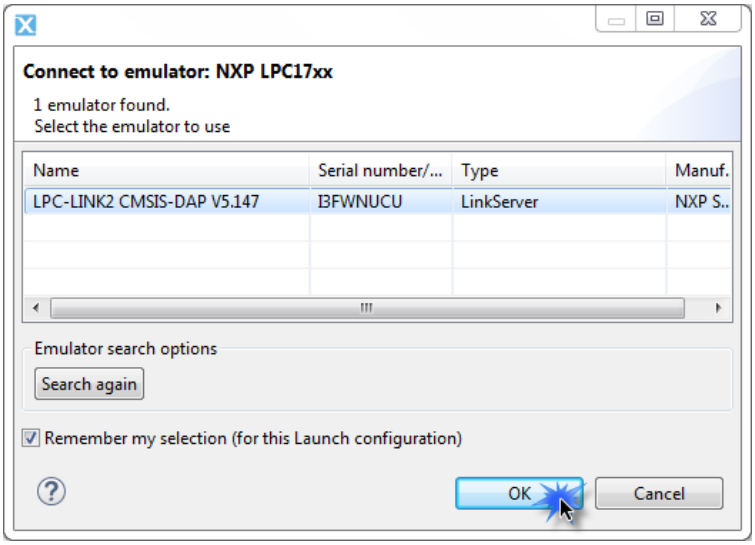
- Select the project and go to menu **Run** > **Debug Configurations**:



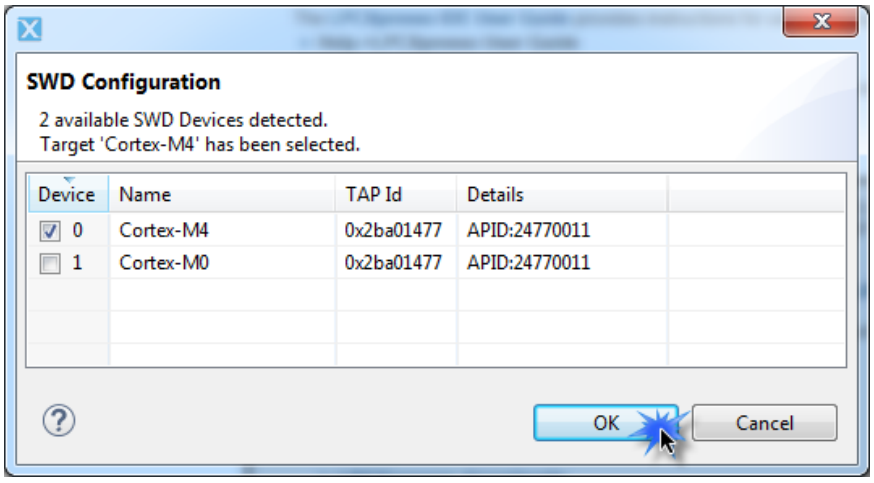
- The **Debug Configurations** window will open, double click on the "**C/C++ (NXP Semiconductors) MCU Applications**" option to create a new connection and program the MCU trough the LPC-Link2 debug probe and click on **Debug**:



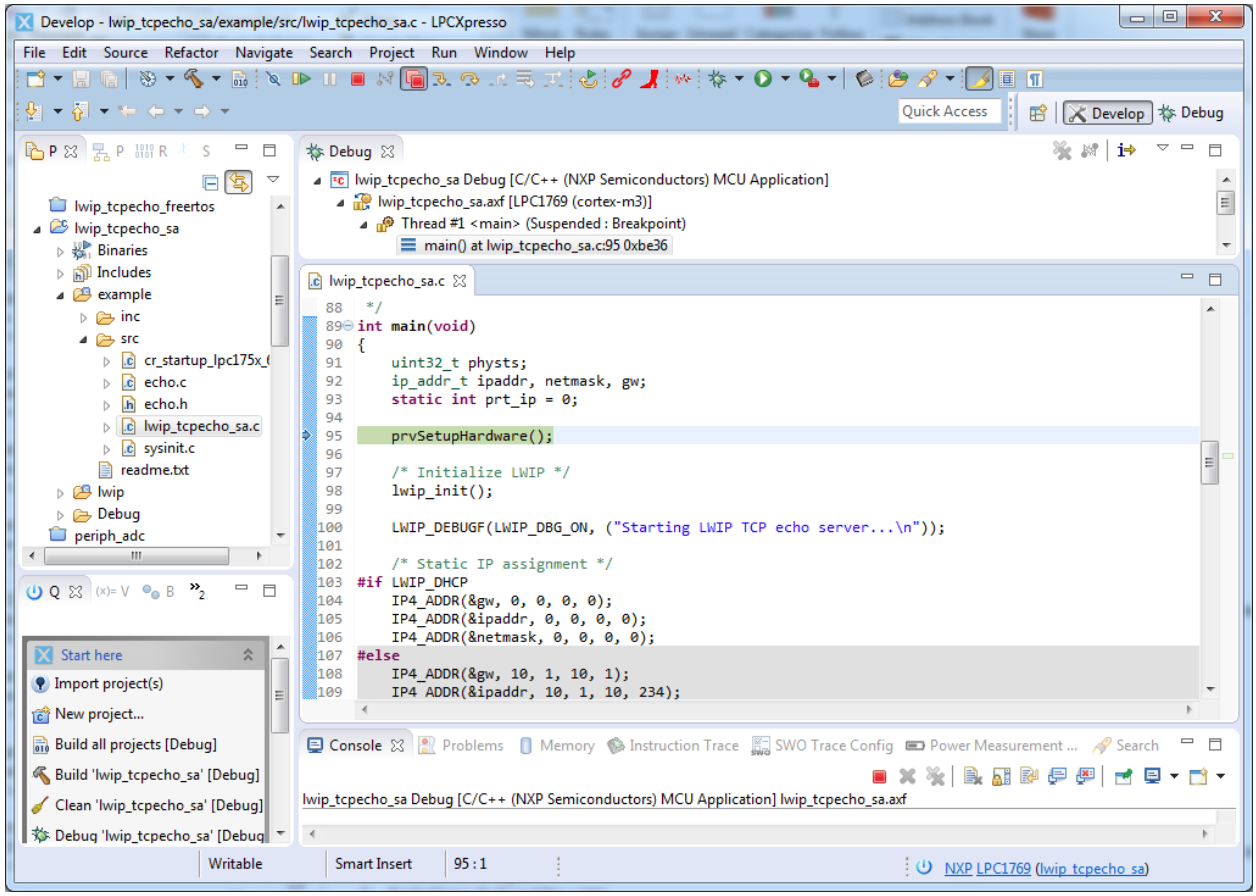
- A new window will show up and it will display the connected emulators, select the **LPC-LINK2 CMSIS-DAP** emulator and click on OK:



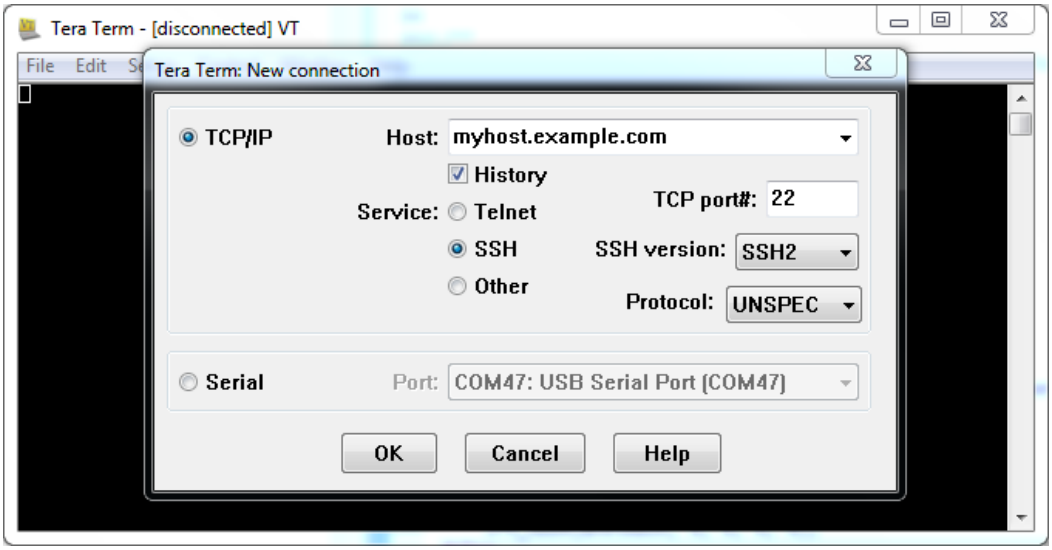
- After this a new message will show up, indicating that 2 SWD devices were found, select the **Cortex-M4** device which is the one used by the example:



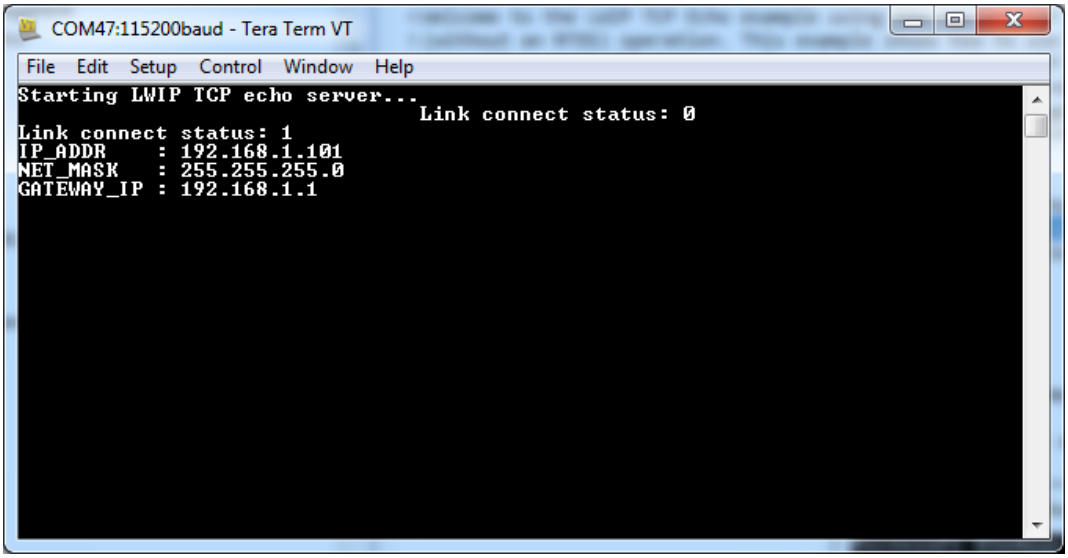
- You should now be able to **debug** the program and **step** through the code:



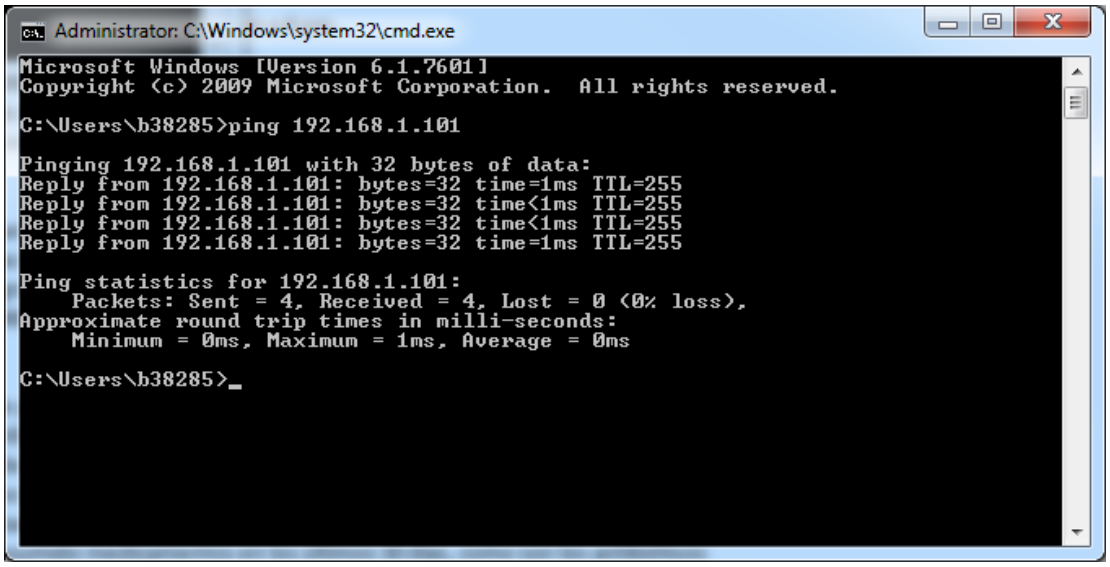
- Now open an hyperterminal, in this care Tera Term, and connect with the USB serial port and reset the board:



- You should see the following information printed on the hyperterminal:

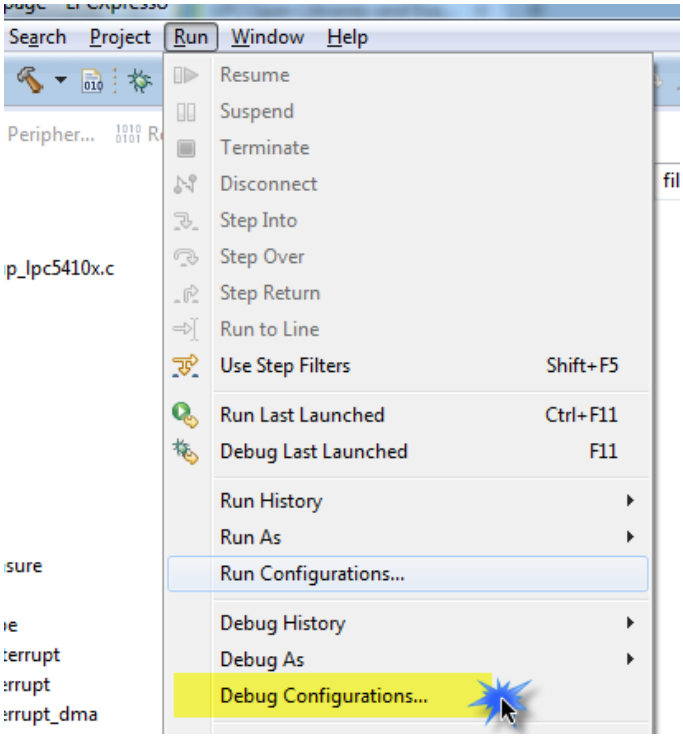


- Finally after the board has acquired an IP address via DHCP, you can you can ping the board at it's IP address:

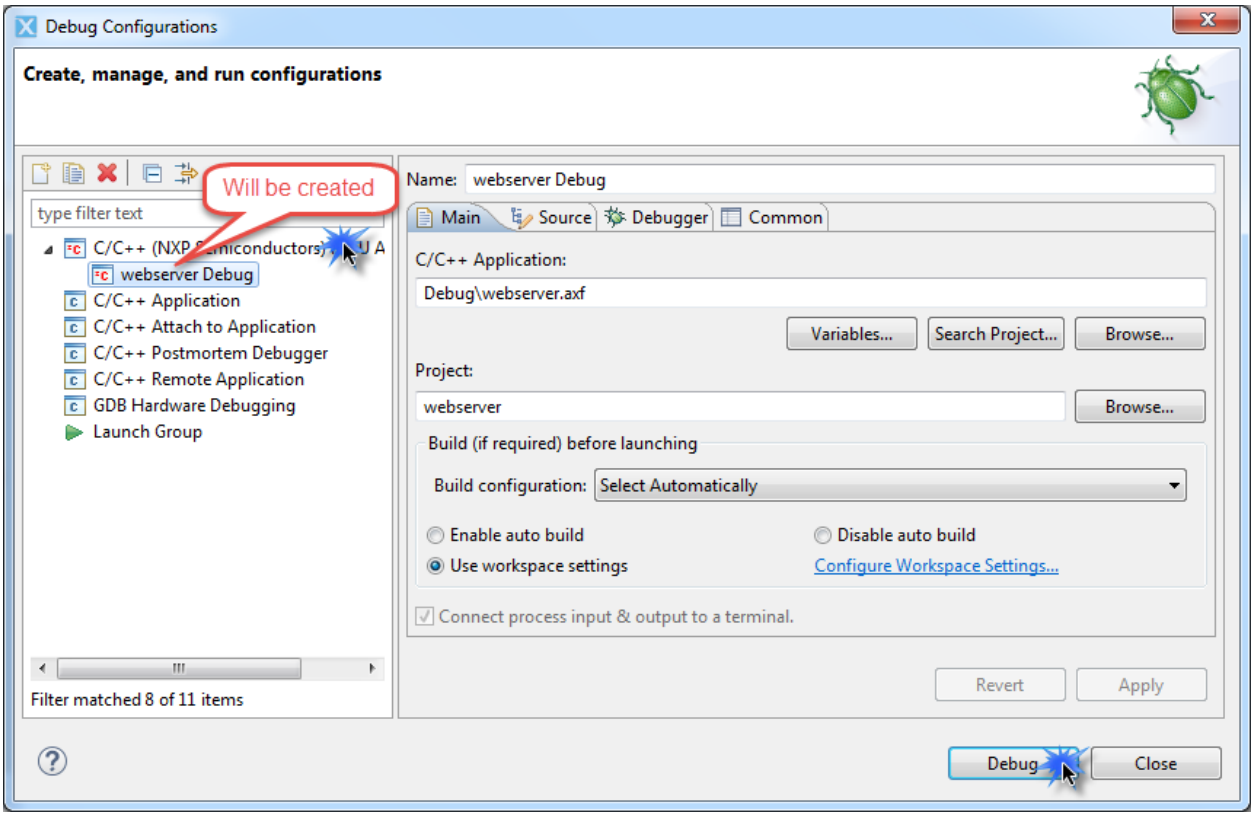


2.5.2 webserver demo

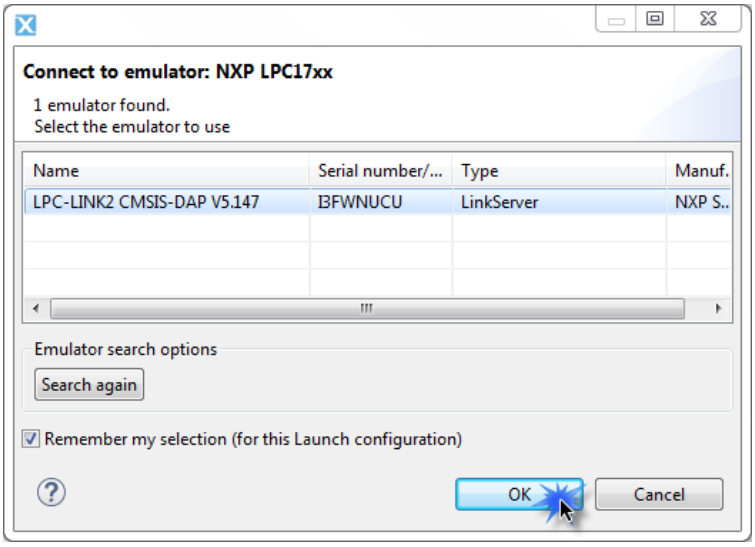
- Select the project and go to menu **Run** > **Debug Configurations**:



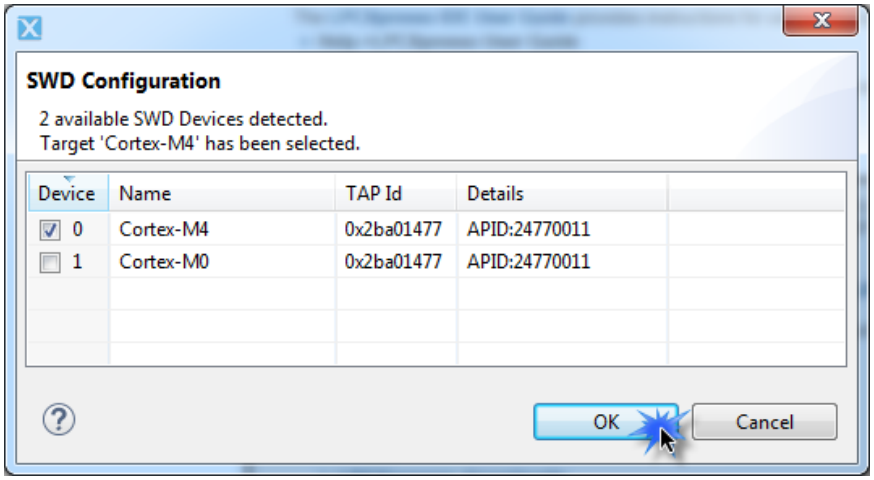
- The **Debug Configurations** window will open, double click on the "**C/C++ (NXP Semiconductors) MCU Applications**" option to create a new connection and program the MCU trough the LPC-Link2 debug probe and click on **Debug**:



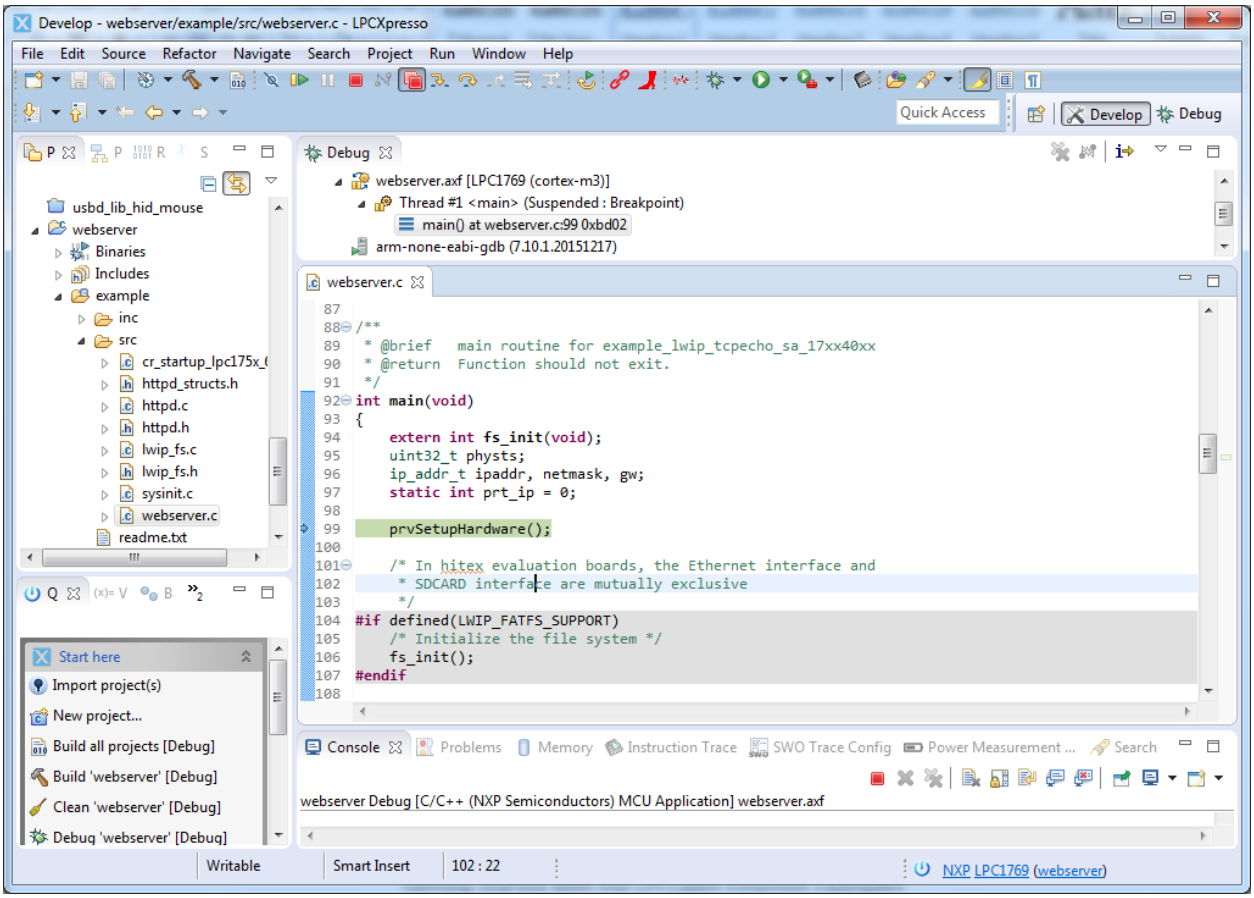
- A new window will show up and it will display the connected emulators, select the **LPC-LINK2 CMSIS-DAP** emulator and click on OK:



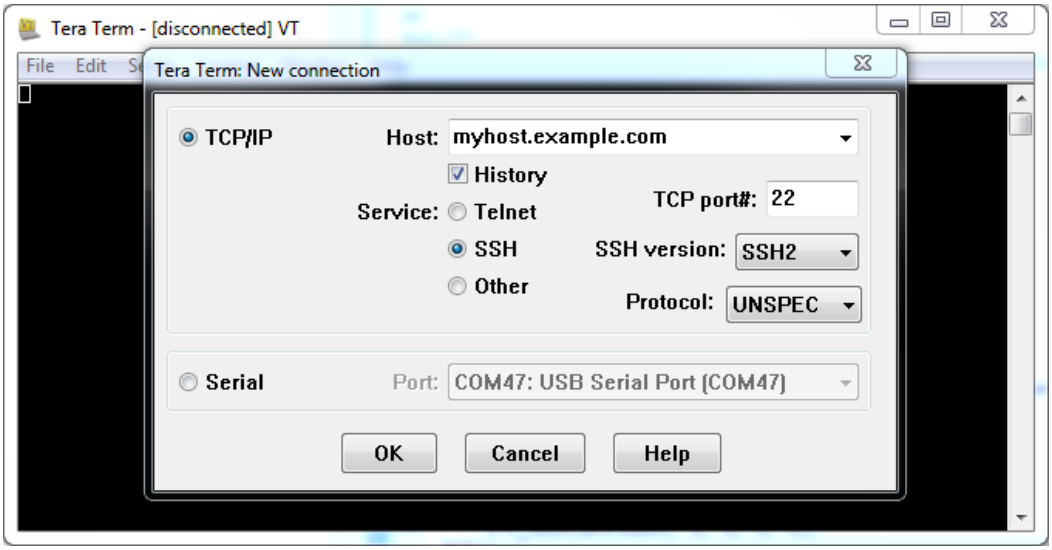
- After this a new message will show up, indicating that 2 SWD devices were found, select the **Cortex-M4** device which is the one used by the example:



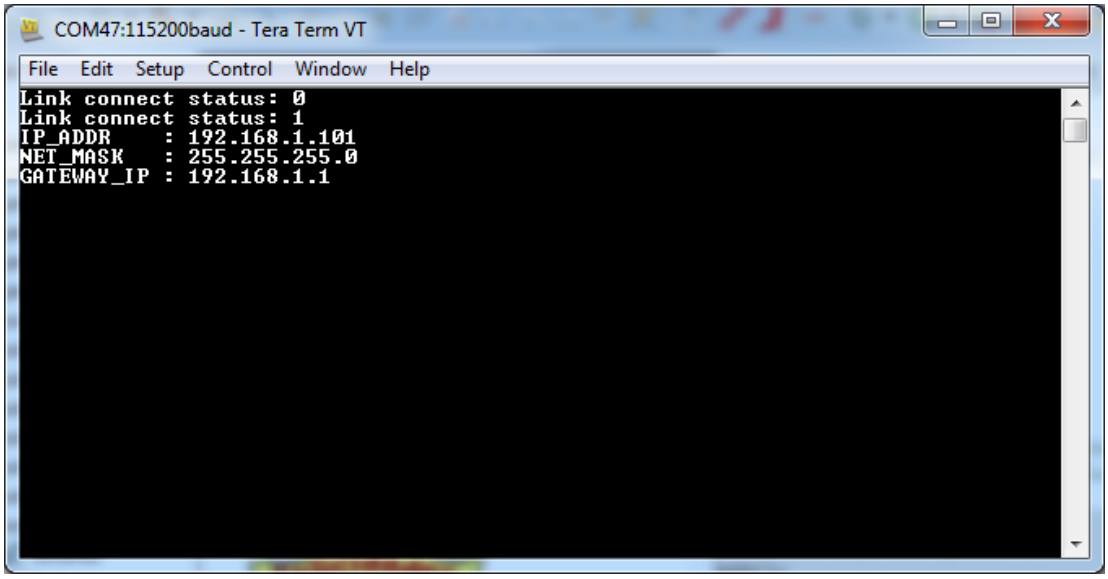
- You should now be able to **debug** the program and **step** through the code:



- Now open an hyperterminal, in this care Tera Term, and connect with the USB serial port and reset the board:



- You should see the following information printed on the hyperterminal:



- Finally after the board has acquired an IP address via DHCP, you can now access the HTTP Server by using a web browser from the host PC:

Appendix A - References

- LPCOpen webpage:
  http://www.nxp.com/lpcopen

- LPCXpresso webpage:
  http://www.nxp.com/lpcxpresso

- LPCXpresso board for LPC1769 with CMSIS DAP probe:
  http://www.nxp.com/products/software-and-tools/software-development-tools/software-tools/lpc-microcontroller-utilities/lpcxpresso-board-for-lpc1769-with-cmsis-dap-probe:OM13085

- EA LPCXpresso BaseBoard:
  http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc-cortex-m0-plus-m0/lpc800-low-cost-cortex-m0-plus/ea-lpcxpresso-baseboard:OM11083

- LPC-Link2:
  http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc-cortex-m0-plus-m0/lpc800-low-cost-cortex-m0-plus/ea-lpcxpresso-baseboard:OM11083

- Product Note LPCXpresso1769/CD rev D:
  https://www.embeddedartists.com/sites/default/files/support/xpr/LPCXpresso1769_CD_revD_ProductNote.pdf