JN517x LPCXpresso Installation and User Guide

JN-UG-3109
Revision 1.2
5 October 2016

Contents

 © NXP Laboratories UK 2016

Preface

This manual describes the use of the NXP LPCXpresso Integrated Development Environment (IDE) to develop applications for the NXP JN517x family of wireless microcontrollers.

Note: LPCXpresso is fully detailed in its own documentation set, available from the NXP web site (see Support Resources on page 6). This manual (JN-UG-3109) provides the additional information needed to develop JN517x applications within LPCXpresso.

Organisation

This manual consists of three chapters and four appendices, as follows:

- Chapter 1 introduces LPCXpresso and provides installation instructions for the plug-ins that are required for developing JN517x applications
- Chapter 2 describes how to develop a JN517x application in LPCXpresso starting from an NXP Application Note
- Chapter 3 describes how to debug a running JN517x application via LPCXpresso
- The Appendices contain information and procedures that may be useful in using LPCXpresso

Conventions

Files, folders, functions and parameter types are represented in **bold** type.

Function parameters are represented in *italics* type.

Code fragments are represented in the `Courier New` typeface.

This is a **Tip**. It indicates useful or practical information.

This is a **Note**. It highlights important additional information.

© NXP Laboratories UK 2016

*This is a **Caution**. It warns of situations that may result in equipment malfunction or damage.*

Acronyms and Abbreviations

IDE	Integrated Development Environment
SDK	Software Developer's Kit

Related Documents

JN517x Data Sheet (JN-DS-JN517x)

LPCXpresso IDE User Guide

LPCXpresso IDE Installation and Licensing Guide

LPCXpresso IDE SWO Trace Guide

Support Resources

To access online support resources such as SDKs, Application Notes and User Guides, visit the Wireless Connectivity area of the NXP web site:

www.nxp.com/products/interface-and-connectivity/wireless-connectivity

All NXP resources referred to in this manual can be found at the above address, unless otherwise stated.

LPCXpresso and its documentation are available from:

www.nxp.com/lpcxpresso

For information on obtaining and installing the appropriate version of LPCXpresso to develop applications for JN517x chips, refer to Section 1.1

Trademarks

All trademarks are the property of their respective owners.

1. Introduction and Installation

The NXP LPCXpresso Integrated Development Environment (IDE) is employed as a platform for the development of wireless network applications to be run on NXP's JN517x family of wireless microcontrollers. LPCXpresso is an Eclipse-based IDE that provides editing, compiling, debug and Flash programming functionality. For information on how to obtain and install LPCXpresso, refer to Section 1.1.

1.1 LPCXpresso for JN517x

LPCXpresso can be obtained from the following NXP web page:

www.nxp.com/lpcxpresso

In order to develop JN517x applications without limitation, we recommend that you purchase the Pro edition of LPCXpresso. You will also need to install plug-ins for JN517x (see Section 1.2).

Caution: *The recommended version of LPCXpresso to be used with an SDK can be found in the SDK Release Notes. This is the version with which the libraries within the SDK were compiled and verified. Other versions of LPCXpresso may not be compatible with the contents of the SDK and cannot be guaranteed to work or be supported with the JN51xx devices.*

To obtain LPCXpresso and install it on your development machine:

1. If you do not already have a web account with NXP, navigate to **www.nxp.com** and create an account.
2. Sign in to your NXP web account.
3. Navigate to the page **www.nxp.com/lpcxpresso**.
4. Select the **Downloads** tab and then click the **Download** button.
5. Check whether the displayed version is the recommended version indicated in the SDK Release Notes:
   - If it is the recommended version, download it.
   - If it is not the recommended version, click **Previous** and then select the recommended version and download it.
6. Launch the LPCXpresso installer and follow the on-screen instructions. Full installation details are provided in the *LPCXpresso IDE Installation and Licensing Guide*, available on the **Documentation** tab of the above web page.

The development of JN517x applications in LPCXpresso is supported on Microsoft Windows 7 and 8 as well as Linux (but not Mac OS X).

© NXP Laboratories UK 2016

Operational details of LPCXpresso are provided in the *LPCXpresso IDE User Guide*, available from the **Documentation** tab of the web page **www.nxp.com/lpcxpresso**. This manual (JN-UG-3109) provides the additional information needed to develop JN517x applications within LPCXpresso.

To develop applications for the JN517x wireless microcontrollers within LPCXpresso, you must install JN517x-specific plug-ins, as described in Section 1.2.

1.2 JN517x Plug-ins for LPCXpresso

Plug-ins are available to allow the development of wireless network applications for the JN517x devices within LPCXpresso. A set of plug-ins is supplied as a JN517x plug-in archive, which basically corresponds to the Software Developer's Kit (SDK) for a particular wireless network protocol together with some additional development tools. This section describes the contents (Section 1.2.1) and installation (Section 1.2.2) of the plug-in archives.

1.2.1 Plug-in Archive Contents

A JN517x plug-in archive is available for each supported wireless networking protocol and includes the following individual plug-ins:

- LPCXpresso support for JN517x
- JN517x Flash programmer
- Serial terminal view
- Software Developer's Kit (SDK) for a wireless network protocol, including stack software and Application Programming Interface (API) libraries
- Application Note wizards for the SDK, including source files for example applications (for some SDKs, the Application Notes may be supplied separately)

The first three plug-ins listed above are the same for all protocols. Only the last two (SDK) plug-ins differ between protocols. Therefore, a plug-in archive corresponds to the SDK for a particular protocol.

The JN517x plug-in archives for LPCXpresso have part numbers of the form JN-SW-42xx and are available via the NXP web site (see "Support Resources" on page 6). For more detailed information about the software components in a particular JN517x plug-in archive, refer to the Release Notes included in the archive.

1.2.2 Plug-in Installation

This section describes how to prepare LPCXpresso for the development of JN517x wireless network applications. This involves installing JN517x-specific plug-ins.

Before proceeding with the installation, note the following:

- You must obtain the appropriate version of LPCXpresso from the NXP web site and install it on your development machine, as described in Section 1.1
- You will need the JN517x plug-in archive (SDK) for your chosen wireless network protocol, which has a part number of the form JN-SW-42xx and can be obtained via the NXP web site (see “Support Resources” on page 6)
- JN517x plug-ins for more than one wireless network protocol can be installed on the same development machine but shared components are only installed once (after the initial installation, subsequent installations will only incorporate the protocol-specific plug-ins, unless newer versions of the shared plug-ins are available)
- When using LPCXpresso on Linux, to ensure that the JN517x plug-ins install their support files correctly in the LPCXpresso application directory, the plug-ins must be installed while running LPCXpresso as root using a command such as:

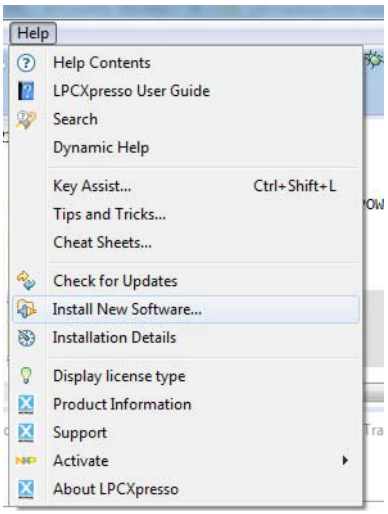
  ```
  sudo /usr/local/<lpcxpresso_install_dir>/lpcxpresso/lpcxpresso
  ```

- For detailed information about the JN517x plug-in archive to be installed, refer to the Release Notes included in the archive

 © NXP Laboratories UK 2016

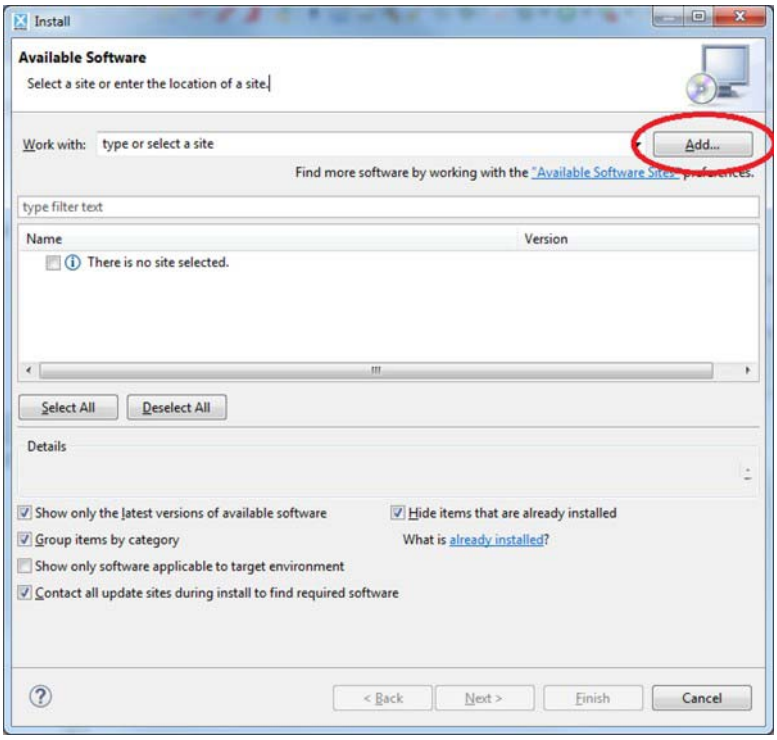
Once you have installed LPCXpresso and obtained the required JN517x plug-in archive, install the plug-ins as follows:

1. Start LPCXpresso on your development machine (if using Linux, run LPCXpresso as root, as described in the note on page 9).
2. In the **Help** menu of LPCXpresso, select **Install New Software**.



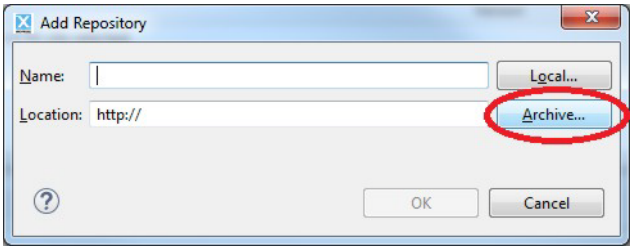
The **Available Software** dialogue box will appear.

3. In the **Available Software** dialogue box, click the **Add** button.



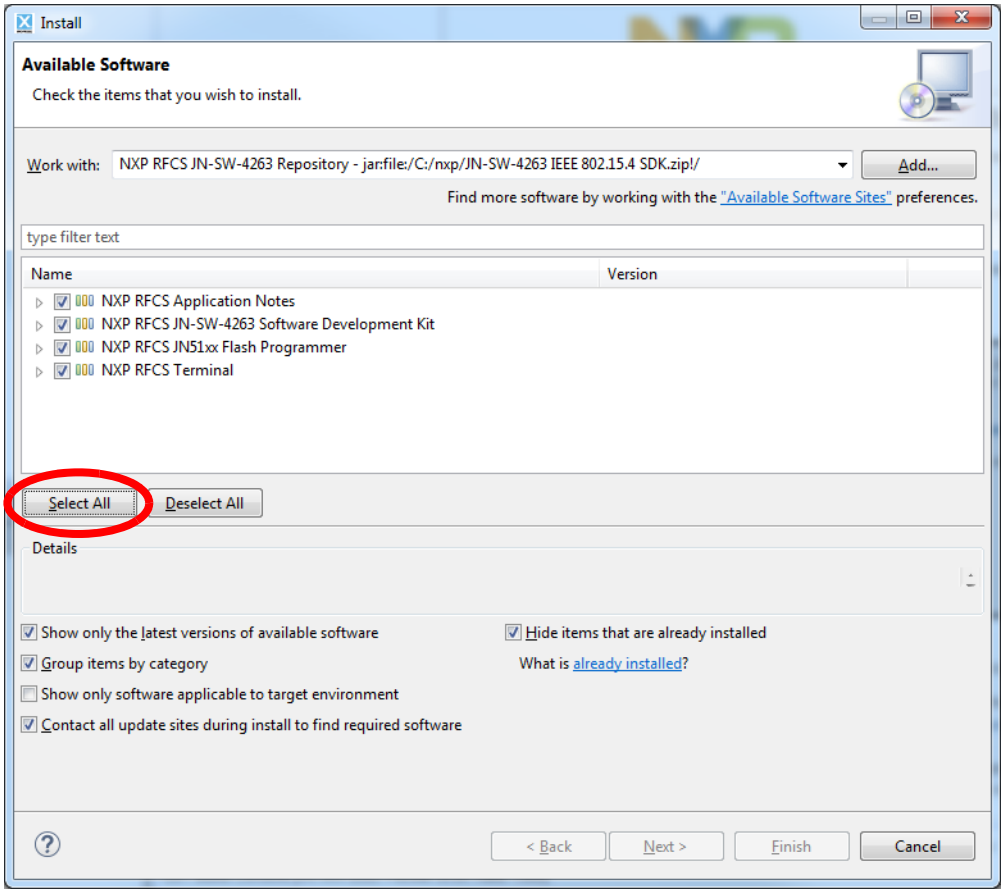
The **Add Repository** dialogue box will appear.

4. In the **Add Repository** dialogue box, click the **Archive** button, browse for the relevant plug-in archive (e.g. **JN-SW-4263 - IEEE802.15.4 SDK.zip**) and click **Open**, then click **OK**.



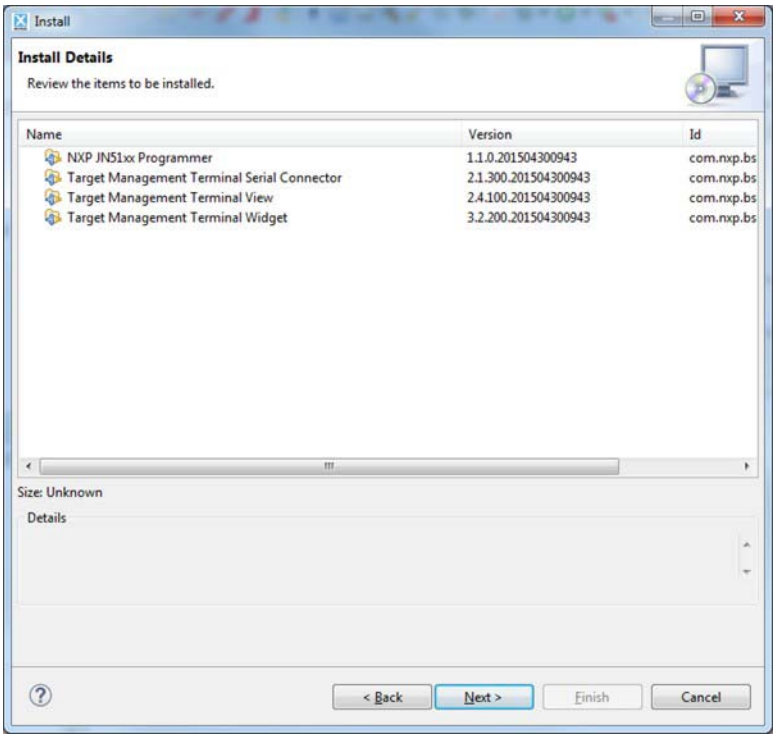
The contents of the plug-in archive are now displayed in the **Available Software** dialogue box.

5. In the **Available Software** dialogue box, click the **Select All** button to select all the plug-ins for installation, and then click **Next**.



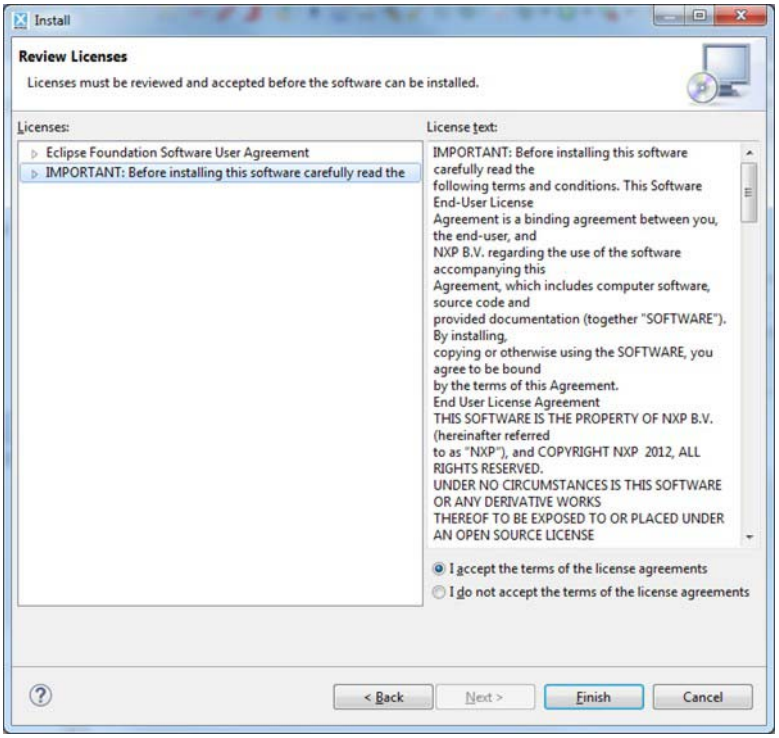
The **Install Details** dialogue box now appears.

6. In the **Install Details** dialogue box, click **Next**.



The **Review Licenses** dialogue box now appears.

7. In the **Review Licenses** dialogue box, read the software license agreements and select “I accept the terms of the license agreements” to indicate your acceptance, then click **Finish** to install the plug-ins.



Note that LPCXpresso will be restarted to complete the installation.

Note 1: Following the above initial installation, if another plug-in archive for a different protocol is installed on the development machine, only the protocol-specific plug-ins will be shown in the **Install Details** dialogue box and installed on the machine (common plug-ins that are already present on the machine will not be re-installed unless the subsequent installation contains newer versions).

Note 2: To use the installed serial Flash programmer within LPCXpresso to program a device via an FTDI USB-to-serial chip, the device driver for the FTDI chip is required on your PC. If this device driver is not already present, you can install it at any time before Flash programming is required. For FTDI device driver installation instructions, refer to Appendix D. For Linux, a very specific installation procedure is required.

 © NXP Laboratories UK 2016

1.3 Directory Structure

During plug-in installation, the following directory structure is created and populated.

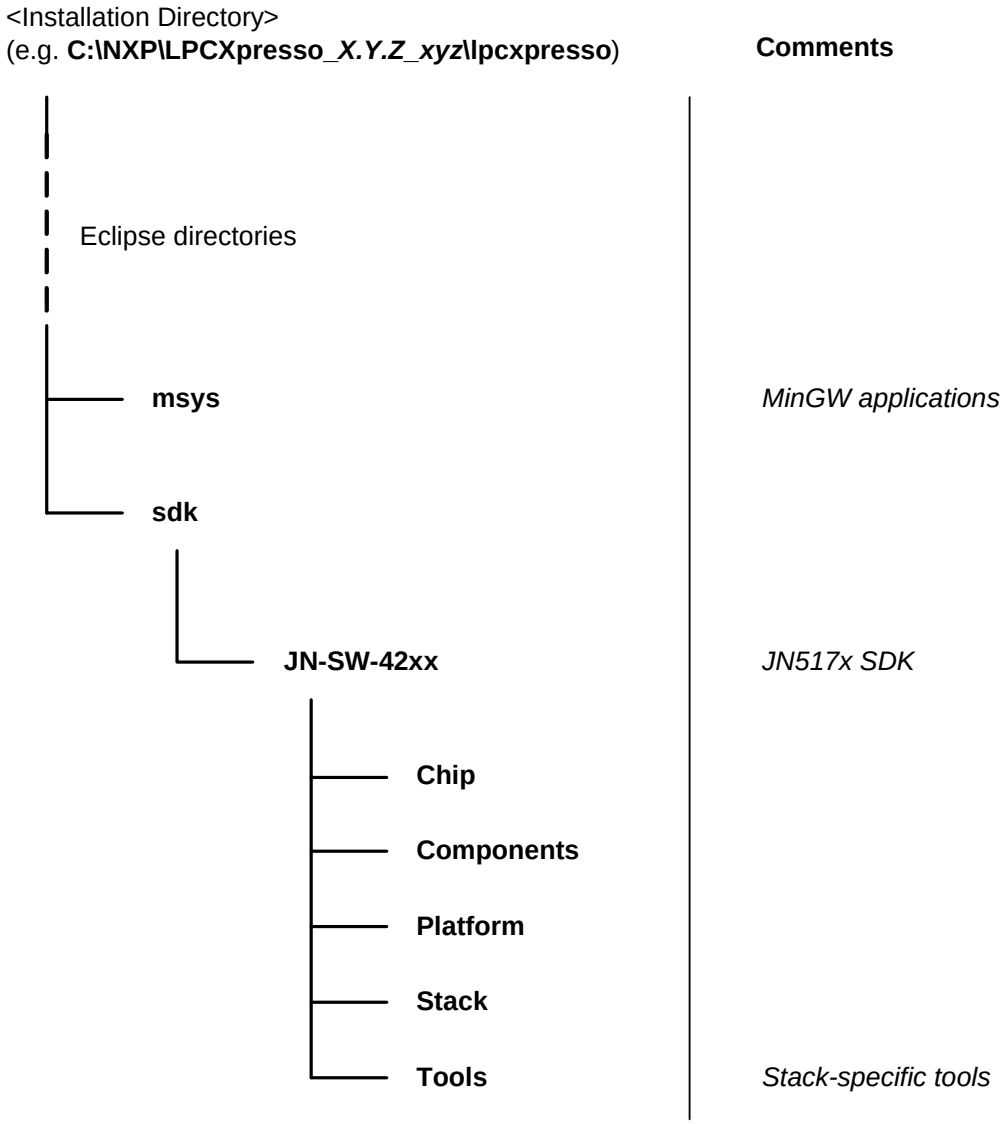


Figure 1: Directory Structure

Note: The above directory structure does not include the user's application workspace. By default, the workspace is within the user's home directory, but the user is free to choose whichever location they like - this choice is made the first time that LPCXpresso is run on the development machine (see Section 1.5.2).

1.4 Basic Workflow

This section outlines the main workflow stages of application development in LPCXpresso that are detailed in the remaining chapters of this manual. These stages are as follows:

Stage 1: Create/Import a project

In LPCXpresso, an application under development is termed a project. Therefore, a project must first be created. For the JN517x devices, NXP provides Application Notes containing template and example applications that can be used as a starting point for custom application development. Relevant templates and examples may be included in the SDK plug-in archive and additional Application Notes are available via the NXP web site (see "Support Resources" on page 6). A new project should be created from a template or example application rather than created from scratch. Creating a new project from a template is described in Section 2.1. You can then write/edit your application code within the source code editor of LPCXpresso.

Stage 2: Build, load and run the application

Once you have written your application code, you can build the application binary using the compiler within LPCXpresso. You can then use the integrated JN517x Flash programmer to load the binary file into the internal Flash memory of a JN517x wireless microcontroller. The processes of building an application and loading it into JN517x Flash memory are described in Section 2.2 and Section 2.3 respectively.

Tip: LPCXpresso provides a 'Run Configuration' feature which allows one or more applications to be built, loaded into the target device(s) and run as the result of a single click of a button. Setting up and using a Run Configuration are described in Section 2.4.

Stage 3: Debug the application

Application development normally involves a debug stage. This can be conducted from LPCXpresso, which interfaces to the application running on a JN517x device via a Single Wire Debug (SWD) interface. The process of setting up a debug session is described in Chapter 3.

© NXP Laboratories UK 2016

1.5 Basic LPCXpresso Features

LPCXpresso is based on the Eclipse IDE and inherits many Eclipse features. This section outlines some of these features.

Note: For more detailed information about the features and operation of LPCXpresso, refer to the *LPCXpresso IDE User Guide*.

1.5.1 Starting LPCXpresso

LPCXpresso can be started either via the Windows **Start** menu or by double-clicking on the desktop shortcut (if present). On starting LPCXpresso for the first time, you will be asked to specify a 'workspace' directory for your application development - see Section 1.5.2.

1.5.2 Workspace

The 'workspace' directory is a space on your computer where your project(s) are kept. On starting LPCXpresso for the first time, you will be asked to specify a workspace directory in the **Workspace Launcher** dialogue box. By default, the workspace is within your home directory, but you can choose whichever location you like.

- You can use a single workspace for all your projects. In this case, if you tick the "Use this as the default and do not ask again" checkbox in the **Workspace Launcher** dialogue box, during future start-ups this dialogue box will be by-passed and you will be taken directly to your workspace.
- You can have separate workspaces for separate projects. In this case, leave the **Workspace Launcher** dialogue box to appear on every start-up (so do not tick the above checkbox) and specify a new workspace when needed (the previously used workspace will be specified by default).

1.5.3 Perspectives

The main graphical interface of LPCXpresso can take one of a number of 'perspectives', where each perspective is a window containing a number of panes and has a particular use (e.g. debug). The following perspectives are available

- **C/C++ Perspective:** For general code development and editing
- **Debug Perspective:** For application debugging
- **Develop Perspective:** Simplified view for both code development and debugging - this is the default perspective in which LPCXpresso starts

The layout of perspectives and the functional roles of the panes are detailed in the *LPCXpresso IDE User Guide*.

 © NXP Laboratories UK 2016

2. Developing an Application

This chapter provides guidance on how to use LPCXpresso to create and develop a JN517x application project. The chapter describes:

- How to create a new project for the application - see Section 2.1
- How to build the application (for release or debug purposes) - see Section 2.2
- How to load a built application into a JN517x device - see Section 2.3
- How to set up and execute a Run Configuration - see Section 2.4

2.1 Starting a Project

In LPCXpresso, an application under development is termed a project. Therefore, a project must first be created.

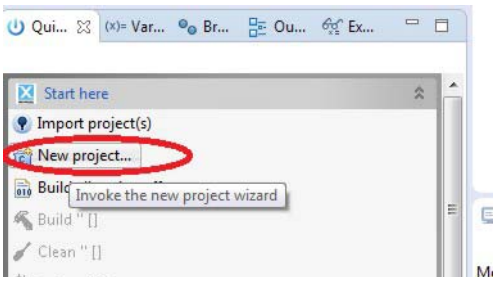
For the JN517x devices, NXP provide Application Notes containing template and example applications that can be used as a starting point for custom application development. Relevant templates and examples may be included in the SDK plug-in archive and additional Application Notes are available via the NXP web site (see "Support Resources" on page 6). A new project should be created from a template or example application rather than created from scratch.

This chapter therefore describes how to start a new project from one of the supplied Application Notes. This project creation process is automated by means of a wizard.

Note: The templates and examples supplied in a JN517x plug-in archive are stored in the **Wizards** and **Examples** directories within the LPCXpresso directory structure. However, the relevant files are copied to the user workspace when a new project is created from an Application Note. These files include the PDF documentation for the Application Note.

To create a new project from an installed Application Note:

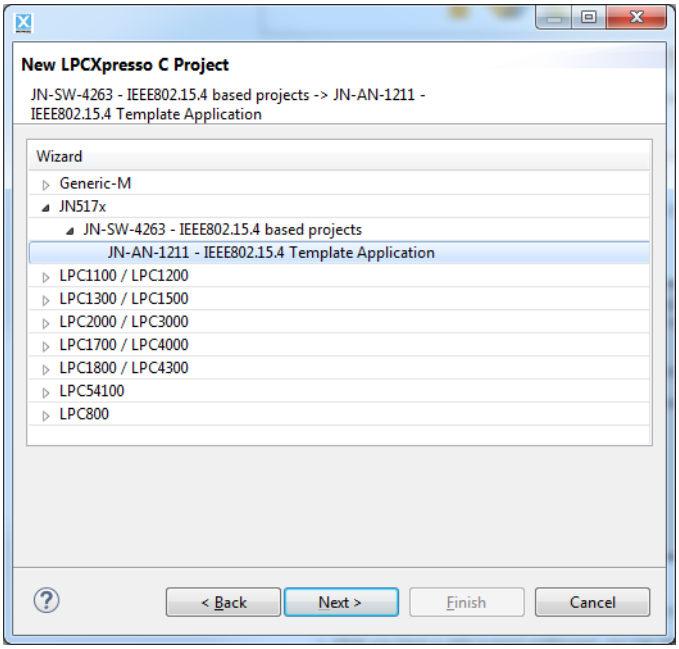
1. In the LPCXpresso **Quickstart Panel**, select **New project** (alternatively, follow the menu path **File > New > Project > C/C++ > LPCXpresso C Project**).



© NXP Laboratories UK 2016

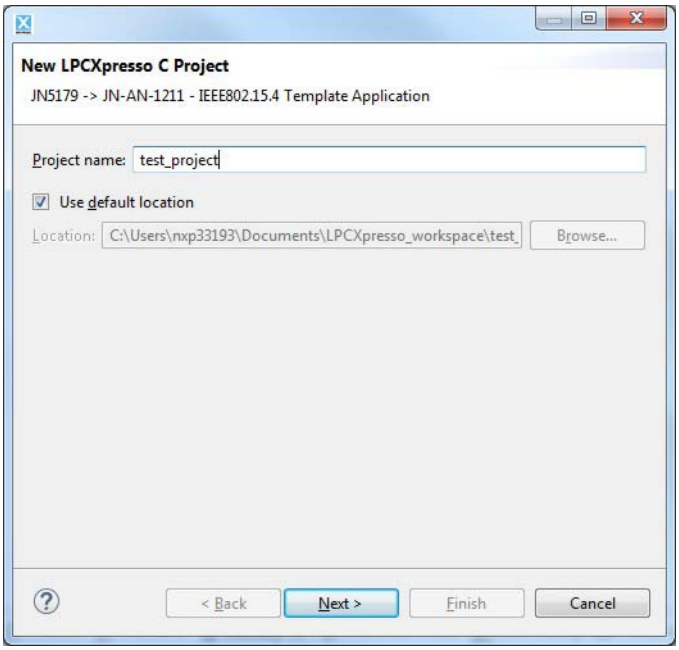
The **New LPCXpresso C Project** dialogue box now appears.

2. In the **New LPCXpresso C Project** dialogue box, select the JN517x family, then the relevant device, then the desired Application Note, then click **Next**.

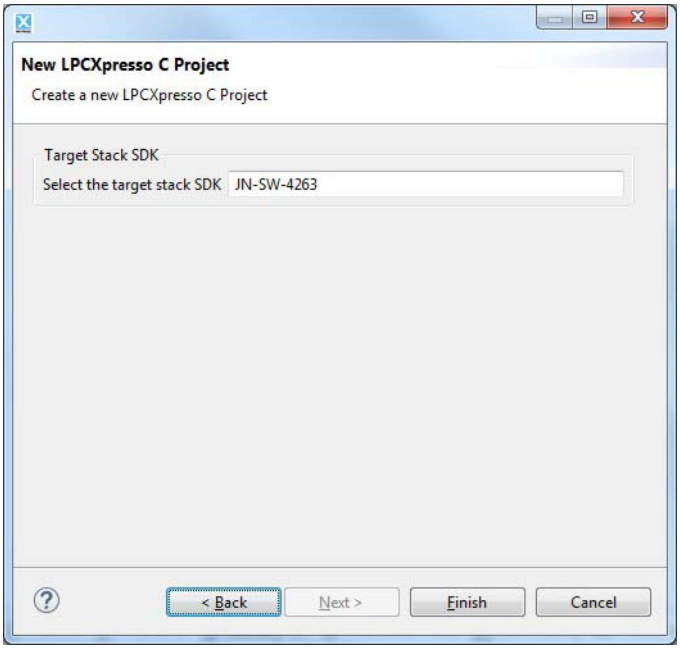


The project creation wizard for the selected Application Note will now run.

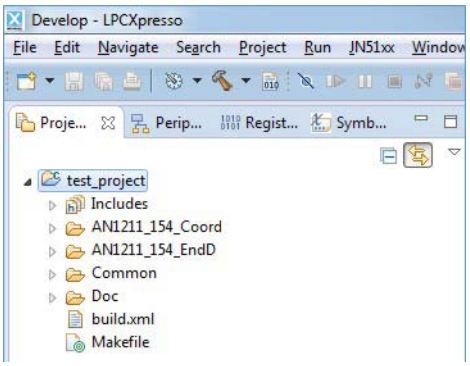
3. Follow the instructions provided in the project creation wizard. Give a new name to the project and then click **Next**.



4. Select the SDK (JN-SW-42xx) on which the project will be based and then click **Finish**.



The wizard will now create the project in the user workspace and set it up as selected.



The wizard creates two build targets, "Debug" and "Release", that will be available via the **Build** button ('hammer' 🔨 icon):

- The "Debug" target can be used to compile a debug binary file for SWD debugging (see Chapter 3)
- The "Release" target can be used to compile the final production binary file

Note: The project source files can now be developed in the Editor pane within the LPCXpresso Develop Perspective or C/C++ Perspective - for more information, refer to the *LPCXpresso IDE User Guide*.

2.2 Building a Project

LPCXpresso allows a project to be built to produce a binary executable file for either of the following two build targets:

- **Release target:** The application is built optimised for storage size and execution speed (as required for real-world deployment)
- **Debug target:** The application is built to incorporate information required for debugging and, as such, is not optimised for size or speed

Caution: *Do not attempt to debug the Release build of an application, as the debugger will not operate with a Release build.*

The **Build** button ('hammer' icon) on the toolbar is used to initiate a build.

To build your project:

Step 1 Select the required build type, Release or Debug.

Step 2 Start the build by either of the following methods:

- Click the the **Build** button on the toolbar and select the project to be built - the application will then build automatically.
- In the **Projects Explorer** pane, right-click on the relevant project and, from the pop-up menu, select **Build Project** - the application will then build automatically.

Step 3 Check the progress of the build in the bottom pane of the window. Standard output from the build can be seen on the **Console** tab. Any errors/warnings created by the build process will be displayed on the **Problems** tab.

The project binaries will be created as **.bin** files in the **Build** folder.

Note: Following one or more builds with the Debug configuration, the files created during the debug process can be deleted via the menu path **Project>Clean**. You should do this before rebuilding with the Release configuration to produce the final output.

2.3 Loading an Application into a JN517x Device

A built application can be loaded into the internal Flash memory of a JN517x device directly from LPCXpresso using the integrated serial Flash programming tool for JN517x. This tool is useful during application development and testing.

Note 1: The serial Flash programmer within LPCXpresso can be used to program JN517x internal Flash memory only, and not external Flash memory. To program an external Flash device attached to the JN517x microcontroller, the JN51xx Production Flash Programmer should be used (see Note 2 below).

Note 2: A CLI Flash programming tool is available for production situations. This software has the part number JN-SW-4107 and is described in the *JN51xx Production Flash Programmer User Guide (JN-UG-3099)*, which are both available from the NXP web site.

The integrated serial Flash programming tool also offers the facility to erase or program the JN517x EEPROM and to program a customer IEEE/MAC address into the target device. These items can only be programmed at the same time as programming Flash memory.

Before loading a built application, ensure that you have the following:

- A target device containing a JN517x microcontroller
- A suitable cable for connecting a USB port of your PC and to the target device (a cable is not needed when programming a JN517x USB dongle)
- If the target device is to be programmed via an FTDI USB-to-serial chip, you will need the device driver for this chip on your PC (guidance on FTDI device driver installation is given in Note 2 at the end of Section 1.2.2)
- The built application **.bin** file to be loaded (this file is located in the **Build** directory for the project)
- If applicable, the file containing data to be loaded into EEPROM and/or the customer IEEE/MAC address to be programmed

© NXP Laboratories UK 2016

To load a binary application into the Flash memory of the target JN517x device:

Step 1 Connect the target JN517x device (which may be mounted on a board or module) to a USB port of your PC (UART0 on the JN517x device is used for this connection). At this point, you may be prompted to install the driver for the cable (if used).

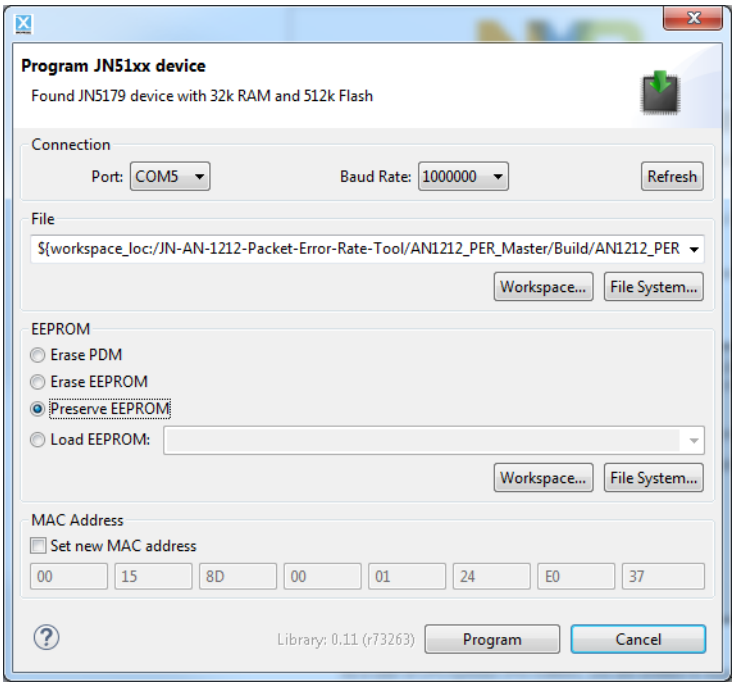
Step 2 Determine which serial communications port on your PC has been allocated to the USB connection - to identify the relevant port, refer to Appendix A.

Step 3 If not already running, launch LPCXpresso.

Step 4 Start the serial Flash programming tool within LPCXpresso by any one of the following methods:

- Follow the menu path **JN51xx>Program Device**
- Click on the **Program Device** button (icon) in the toolbar
- Use the keyboard shortcut **Ctrl+Shift+P**

The **Program JN51xx device** dialogue box will now appear (shown below).



Step 5 In the **Program JN51xx device** dialogue box (shown above), configure the serial Flash programming tool as follows:

- **Connection** - Fill in the **Port** and **Baud Rate** fields as follows:
  - **Port** - Select the serial communications port used for the connection, as determined in Step 2
  - **Baud Rate** - Select the required data rate, in bps (normally, the default value of 1000000 should be used)
- **File** - Navigate to the binary file to be loaded (if this file is in your workspace, use the **Workspace** button, otherwise use the **File System** button)

- **EEPROM** - Select the required operation on JN517x EEPROM:
  - **Erase PDM** to delete all persistent data records from EEPROM
  - **Erase EEPROM** to delete all the contents of EEPROM
  - **Preserve EEPROM** to keep all the contents of EEPROM
  - **Load EEPROM** to load/replace data in EEPROM (in this case, navigate to the data file to be loaded - if this file is in your workspace, use the **Workspace** button, otherwise use the **File System** button)
- **MAC Address** - The IEEE/MAC address of the target device will be displayed. If you wish to over-ride the factory-set MAC address with a customer MAC address, tick the **Set new MAC address** check-box and enter the new MAC address into the eight boxes (note that this is only one-time programmable).

Step 6 Click the **Program** button. The dialogue box will now disappear and loading will start. The progress of loading will be displayed in a progress box.

Step 7 Once the loading has successfully completed, disconnect the serial cable and, if required, reset the target device to run the application.

Caution: *If set, a customer MAC address will over-ride but not over-write the factory-set MAC address. Both MAC addresses will still be held in the device but the customer MAC address will be used. Once written, the customer MAC address cannot be changed or erased.*

Note 1: Optionally, a **Terminal** tab can be associated with a connected device, in order to allow output from an application running on the device to be displayed. This option can be configured as described in Appendix B.1.

Note 2: LPCXpresso also allows you to read the contents of Flash memory and/or EEPROM of a JN517x device and output this data to the file system of your PC. This type of memory access is described in Appendix C. The tool can also be used to later write this data back to EEPROM, to return to a known state.

2.4 Creating and Using a Run Configuration

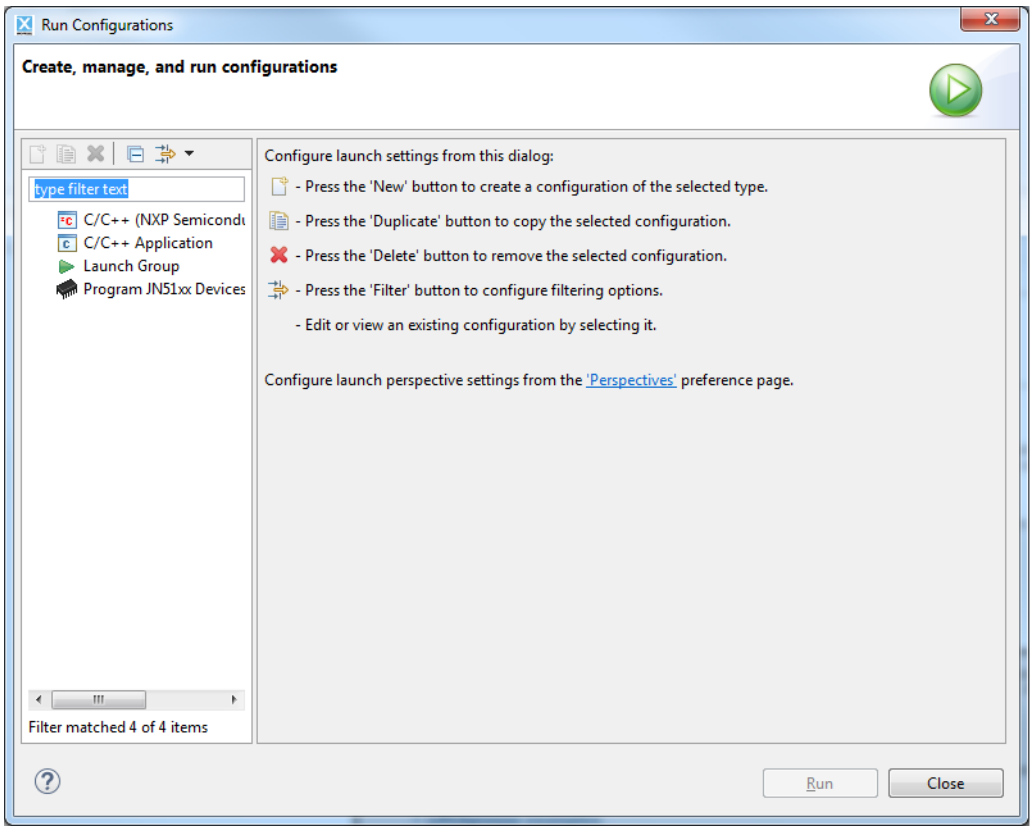
LPCXpresso allows you to create a 'Run Configuration' for one or more applications. A Run Configuration allows an application to be built, loaded into the target device and run as the result of a single click of a button in LPCXpresso. If the Run Configuration includes multiple applications and devices, all the applications will be built and loaded into their respective devices in parallel.

Note that in order to create and use a Run Configuration:

- The target devices must be connected to the PC and the serial communications ports to which the devices are connected must be determined in advance, as described in Appendix A.
- If the target device is to be programmed via an FTDI USB-to-serial chip, you will need the device driver for this chip on your PC (guidance on FTDI device driver installation is given in Note 2 at the end of Section 1.2.2).

To set up and execute a Run Configuration:

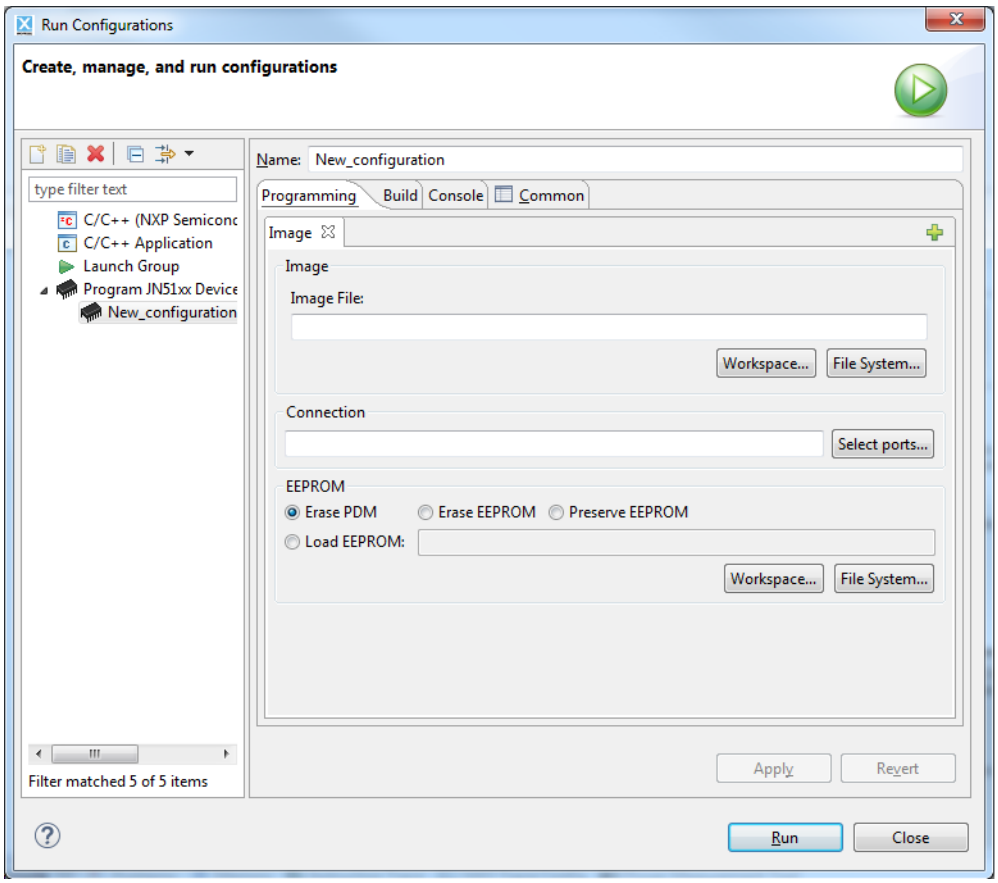
Step 1 Within LPCXpresso, follow the menu path **Run>Run Configurations**. This will launch the **Run Configurations** dialogue box (shown below).



Step 2 In the **Run Configurations** dialogue box (shown above), create a new Run Configuration as follows:

a) In the left pane of the dialogue box, click on **Program JN51xx Devices**.

b) Click on the **New** button (icon) at the top of the left pane of the dialogue box. The right side of the dialogue box will now be populated with fields and tabs (as shown below).



Step 3 In the **Name** field, replace ‘New_configuration’ with the name of your Run Configuration.

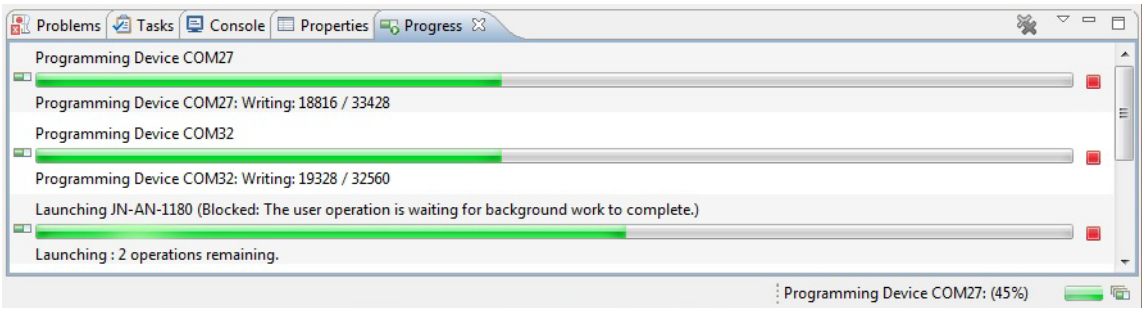
Step 4 On the **Programming** tab, fill in the required information (for one application and target device) in the fields on the **Image** sub-tab, as follows:

- **Image** - Navigate to the binary file to be loaded (if this file is in your workspace, use the **Workspace** button, otherwise use the **File System** button)
- **Connection** - Select the serial communications port(s) used for the connection to the target device(s). Multiple ports can be selected using the **Select ports** button and appear in the field as a comma-separated list
- **EEPROM** - Select the required operation on JN517x EEPROM:
  - **Erase PDM** to delete all persistent data records from EEPROM
  - **Erase EEPROM** to delete all the contents of EEPROM
  - **Preserve EEPROM** to keep all the contents of EEPROM
  - **Load EEPROM** to load/replace data in EEPROM (in this case, navigate to the data file to be loaded - if this file is in your workspace, use the **Workspace** button, otherwise use the **File System** button)

Then click the **Apply** button.

Step 5 If you wish to include another application and/or device in the Run Configuration, click the symbol on the right to add another **Image** sub-tab and repeat Step 4.

Step 6 Finally, to execute the Run Configuration, click the **Run** button, otherwise click the **Close** button. The dialogue box will disappear. If run, the progress of execution for all the applications/devices will be displayed in the **Progress** tab in the bottom pane of the workbench (see example below). The **Progress** tab is not visible by default but can be displayed by following the menu path **Window>Show View>Other** and then in the **Show View** dialogue box selecting **General>Progress**.



Note 1: A Run Configuration can be deleted by first highlighting it in the tree view in the left pane and then either clicking the **Delete** button (icon), or by right-clicking and selecting Delete from the pop-up menu.

Note 2: Optionally, a Run Configuration can associate a 'Terminal' tab with a connected device, in order to allow output from an application running on the device to be displayed. This option can be configured as described in Appendix B.2

3. Debugging an Application

This chapter describes how to use the debugging features of LPCXpresso. This allows an application which has been built with the Debug option enabled (see Section 2.2) to be run on a connected JN517x device and debugged through interactions via the Debug Perspective of LPCXpresso.

Application debugging on the JN517x device can be conducted via SWD (Serial Wire Debug). SWD uses the same physical interface connector to the target device as JTAG, but JTAG uses 5 pins while SWD uses only 2 pins. SWD debugging is described in Section 3.1.

When using SWD, the Cortex-M3 Serial Wire Viewer (SWV) trace may also be used. This feature is described in Section 3.2.

3.1 SWD Debugging

This section describes the debugging of JN517x applications via a Serial Wire Debug (SWD) 2-pin interface. This debug method requires the development PC to be connected to the target JN517x device via an NXP LPC-Link 2 debug probe (see **www.lpcware.com/lpcxpresso**) - this connection is described and illustrated in Section 3.1.2.

3.1.1 Preparing for Debug

To prepare an application for debug, follow the procedure below:

1. Build the application for a Debug target, as described in Section 2.2. The filename of the resulting binary application will end in **_hwdebug**. This application will instruct the JN517x bootloader to enable the SWD pins on the device.
2. Load the binary file into the JN517x device from LPCXpresso using the integrated JN517x Flash programmer tool, as described in Section 2.3. This step requires a direct serial connection from the PC to the board on which the JN517x device is located.

Note: This step of using the JN517x Flash programmer and a direct serial connection to load the debug binary file is only required once to enable the SWD pins. Subsequent debug binary files will be automatically loaded via the LPC-Link 2 debug probe.

3. Now remove the direct serial connection set up in Step 2 and connect the PC to the target board via the LPC-Link 2 debug probe, as described below in Section 3.1.2.

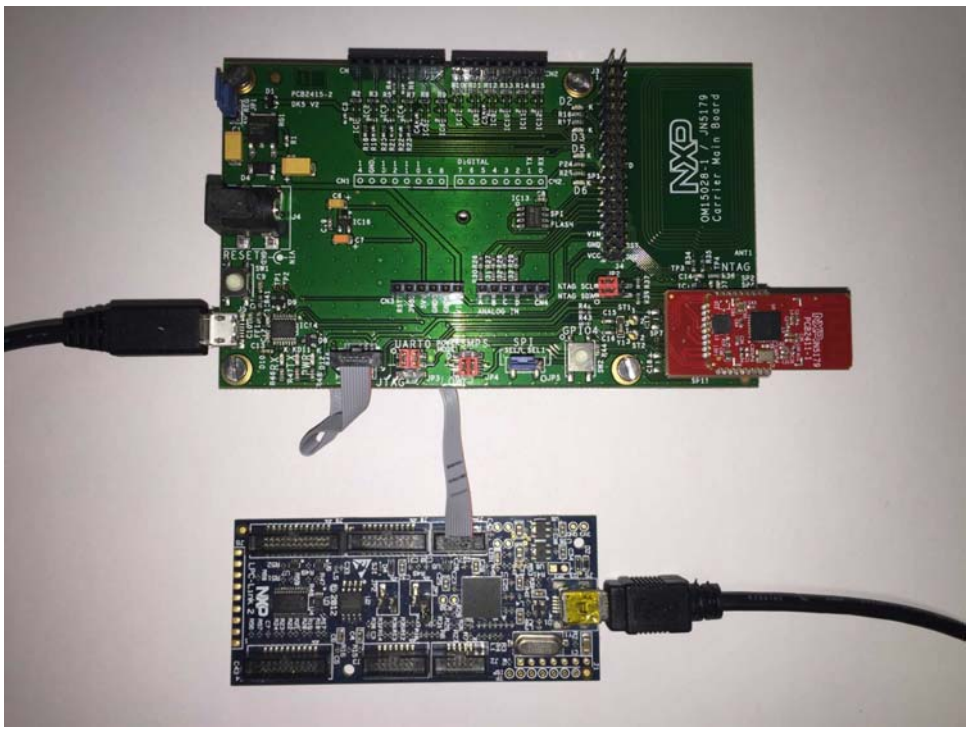
 © NXP Laboratories UK 2016

3.1.2 Connecting PC to JN517x Device for Debugging

Once the initial debug binary file has been loaded into the target JN517x device, as described in Section 3.1.2, connect the PC to the target device via the LPC-Link 2 debug probe as follows:

1. Ensure that there is no power to the board that contains the target JN517x device and enable the board for SWD debugging (as described in the board documentation).
2. Connect the development PC to the LPC-Link 2 debug probe via a USB cable. When you make this connection, you may be prompted to install the device driver for the debug probe (or it may be installed automatically from the Internet).
3. Connect the LPC-Link 2 debug probe to the target board's SWD header (thus, the PC is now connected to the board via a USB cable and the debug probe).
4. If the target board is self-powered (e.g. from batteries), reconnect this power source. Otherwise, reinstate the USB connection (removed in Section 3.1.1) to the PC in order to supply power to the device. If the USB connection is reinstated, it is only used to power the device during debug and cannot be used for serial output from the application.

The photograph below shows the final connections.



3.1.3 Launching the Debugger

Once the PC has been connected to the target JN517x device via the LPC-Link 2 debug probe, the debugger in LPCXpresso can be started.

Note 1: On launching the debugger, the application image to be debugged will be automatically loaded into the JN517x device via the SWD interface and run.

Note 2: By default, the LPCXpresso Debug configuration will set an initial breakpoint on the conventional C program entry-point function called **main()**. However, for the JN517x device, this function is equivalent to **AppColdStart()**. Therefore, execution in Debug mode on a JN517x device will first stop at **AppColdStart()**.

First Debug of Application

When debugging a particular application for the first time within LPCXpresso, the following procedure must be used in order to generate a Debug configuration as part of the debug session:

Step 1 Make sure your project is open in the Develop Perspective of LPCXpresso.

Step 2 In the **Quickstart Panel**, select the **Debug** option for your project. LPCXpresso will now create a Debug configuration for the project and build the application.

Step 3 The Debug application image will be automatically loaded into the JN517x device and run. Initially, application execution will be paused at the **AppColdStart()** function and will wait for user interaction. You can now debug the application.

Subsequent Debugs of Applicaton

After the first debug session for an application, described above, subsequent debug sessions should be conducted as follows:

Step 1 Make sure your project is open in the Develop or C/C++ Perspective of LPCXpresso.

Step 2 Click the down-arrow to the right of **Debug** button (‘bug’ icon) on the toolbar in order to access the drop-down menu. The applications with Debug configurations will be listed at the top of this menu - select the required application from this list. The application will now be built.

Note: If you simply click on the **Debug** button (and not the arrow), the last application that was debugged will be automatically selected.

 © NXP Laboratories UK 2016

Step 3 If you started in the C/C++ Perspective, LPCXpresso will now try to switch to the Debug Perspective but will request permission by displaying the **Confirm Perspective Switch** dialogue box. Accept this switch by clicking the **Yes** button (you may also wish to tick the **Remember my decision** checkbox so that this dialogue box is not displayed when the debugger is launched in the future). LPCXpresso will now display the Debug Perspective.

Step 4 The Debug application image will be automatically loaded into the JN517x device and run. Initially, application execution will be paused at the **AppColdStart()** function and will wait for user interaction. You can now debug the application via the Develop or Debug Perspective.

Important: When a debug session is no longer required, it must be ended using the **Terminate** button (icon) on the toolbar. You can then revert back to the C/C++ Perspective - for example, by following the menu path **Window>Open Perspective>C/C++**. You should load a Release build of the application into the JN517x device in order to deactivate the SWD hardware.

3.2 Serial Wire Viewer (SWV) Trace

The JN517x family of wireless microcontrollers support the standard Cortex-M3 Serial Wire Viewer (SWV) trace, which is available when debugging via an SWD interface. This tracing mechanism does not provide full instruction tracing, but provides coverage analysis and interrupt statistics. Use of the SWV trace mechanism requires an SWD connection from the PC to the target device, as described in Section 3.1.2.

The SDK includes a JN51xx Debug library, which is associated with the IEEE 802.15.4 layers of the software stack and is described in the *JN51xx Core Utilities User Guide (JN-UG-3116)*. When this library is initialised on the JN517x device via the **DBG_vJtagInit()** function, it is configured to send output from the **DBG_vPrintf()** function to the host debug console (PC). When an SWD (or JTAG) connection is used, the output is transferred from the target JN517x device to the host via the ARM semi-hosting mechanism. This mechanism affects the timing of the program under debug because the processor core must stop and transfer control to the debug hardware in order to output the data.

Note: SWV trace requires an additional pin to be enabled on the JN517x device. The DEBUG_MODE variable, which is configured in the application makefile or through the LPCXpresso build configuration environment, must be set to SWD_TRACE to enable the additional pin.

For information on performing an SWV trace from LPCXpresso, refer to the *LPCXpresso IDE SWO Trace Guide*.

Appendices

A. Identifying the PC Communications Port Used

When connecting a USB port of your PC to a serial device, you need to find out which serial communications port your PC has allocated to the connection, as described below (for Windows 7 and 8).

Step 1 In the Windows **Start** menu, open the **Control Panel** by following the menu path:

Start>Control Panel (Windows 7) or

Start>All Apps>Control Panel (Windows 8)

Step 2 From the **Control Panel**, open the **Device Manager** by following the path:

System>Device Manager (Windows 7) or

System & Security>Device Manager (Windows 8)

Step 3 Within the **Device Manager** screen:

a) Look for the **Ports** folder in the list of devices and unfold it.

b) Identify the port which is connected to the serial device (it will be labelled 'USB Serial Port') and make a note of it (e.g. COM1).

B. Creating and Using Serial Terminals

LPCXpresso can provide a serial terminal emulator which allows the display of output from an application running on a connected device, as well as input to the application typed into the PC keyboard. This terminal emulator appears as a 'Terminal' tab in the bottom pane of the workbench. It is possible to have multiple Terminal tabs associated with multiple devices. Each Terminal tab is associated with the serial communications port to which a device is connected - the relevant port number is displayed at the top of the Terminal tab 'screen'.

The sub-sections below describe how to create/enable a Terminal tab to receive the output from an application running on a connected device.

Note 1: The following procedures assume that the devices have been connected to the PC and the serial communications ports to which the devices are connected have been determined as described in Appendix A.

Note 2: A Terminal tab will be automatically disconnected from the associated serial port when the integrated Flash programming tool is used. The connection will be automatically re-established when the Flash programming has completed.

© NXP Laboratories UK 2016

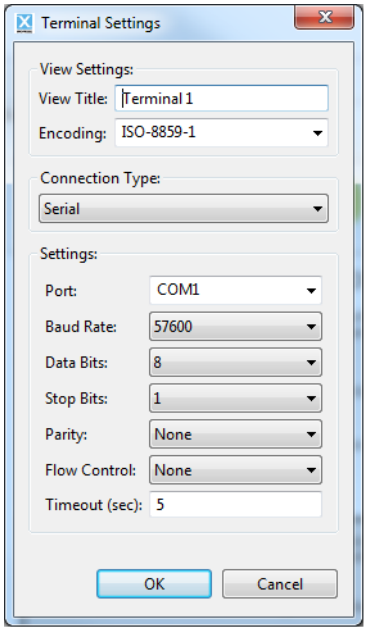
B.1 Creating/Enabling a Terminal Tab

A Terminal tab is provided by default in the bottom pane of the LPCXpresso window - this first such tab is numbered **Terminal 1**. To use this tab, it must be associated with a serial port/device. Further Terminal tabs can also be created for other serial connections.

To enable or create a Terminal tab and associate it with a particular serial port/device:

Step 1 To enable the default Terminal tab in LPCXpresso, click on it or follow the menu path **Window>Show View>Terminal**.

Step 2 To associate this Terminal tab with a device, click the **Settings** button (icon) on the toolbar in the top-right of the pane. This displays the **Terminal Settings** dialogue box (shown below).



Step 3 In the **Terminal Settings** dialogue box (shown above), provide the following information:

a) In the **Port** field, from the drop-down list, select the serial communications port used for the connection to the device to be associated with the Terminal tab.

b) In the **Baud Rate** field, from the drop-down list, select the baud rate for the connection between the device and Terminal tab.

If you wish, you may also change the name of the tab in the **View Title** field.

Click **OK** to save the settings.

Step 4 Connect the terminal to the associated serial communications port by clicking the **Connect** button (icon) in the top-right of the pane (also see Note 2 on page 31).

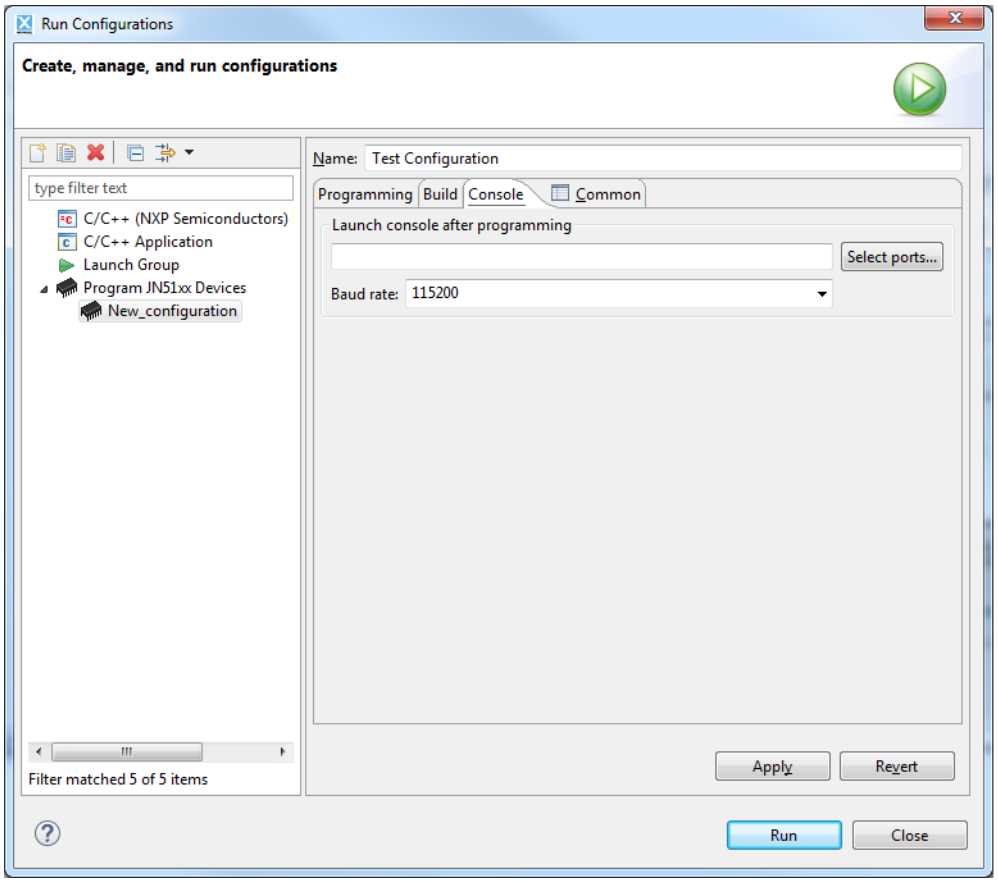
Step 5 If required, you can add another Terminal tab by clicking the **Add Terminal** button (icon) on the toolbar in the top-right of the pane and then configuring this serial terminal as described from Step 2.

B.2 Including a Terminal Tab in a Run Configuration

A Terminal tab can be assigned to a device in a Run Configuration (see Section 2.4) as follows (this procedure allows Terminal tabs to be assigned to multiple devices):

Step 1 Within LPCXpresso, follow the menu path **Run>Run Configurations** to launch the **Run Configurations** dialogue box.

Step 2 Under **Program NXP Devices** in the left panel, click on the Run Configuration to be modified.



Step 3 On the **Console** tab on the right (see above), provide the following information:

a) From the first drop-down list, select the serial communications port(s) used for the connection(s) to the target device(s) to be associated with a Terminal tab.

b) From the second drop-down list, select the baud rate for the connection(s) with the Terminal tab(s).

The configured Terminal tab(s) will automatically connect to the specified serial port(s) (also see Note 2 on page 31).

Step 4 To save the configuration, click the **Apply** button.

Step 5 To execute the Run Configuration, click the **Run** button, otherwise click the **Close** button.

© NXP Laboratories UK 2016

C. Reading the Contents of Flash Memory and EEPROM

LPCXpresso allows you to read the contents of the Flash memory and/or EEPROM of a JN517x device and dump the output to the file system on your PC.

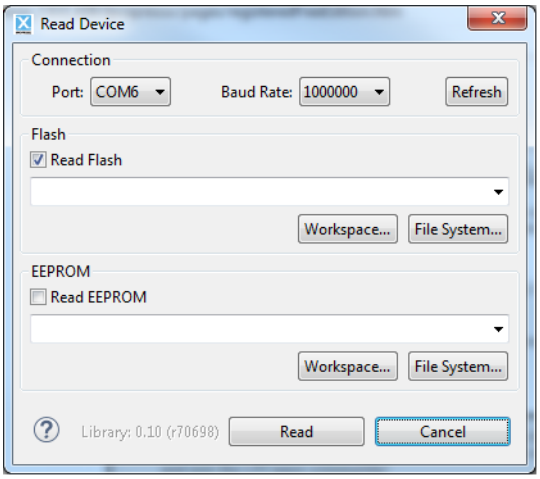
To read the contents of the Flash memory and/or EEPROM of a connected JN517x device:

Step 1 Ensure that the JN517x device to be accessed (which may be mounted on a board or module) is connected to a USB port of your PC.

Step 2 Determine which serial communications port on your PC has been allocated to the USB connection - to identify the relevant port, refer to Appendix A.

Step 3 If not already running, launch LPCXpresso.

Step 4 In LPCXpresso, follow the menu path **Devices>Read Device**. The following dialogue box will now appear.



Step 5 In the dialogue box (shown above), configure the memory access as follows:

- **Connection** - Fill in the **Port** and **Baud Rate** fields as follows:
  - **Port** - Select the serial communications port used for the connection, as determined in Step 2
  - **Baud Rate** - Select the required data rate, in bps (normally, the default value of 1000000 should be used)
- **Flash** - To read from Flash memory, complete this section as follows:
  - Tick the **Read Flash** checkbox.
  - Navigate to the directory in which you wish the output file to be created (if this is in your workspace, use the **Workspace** button, otherwise use the **File System** button) and add the required filename to the end of the path

- **EEPROM** - To read from EEPROM, complete this section as follows:
  - Tick the **Read EEPROM** checkbox.
  - Navigate to the directory in which you wish the output file to be created (if this is in your workspace, use the **Workspace** button, otherwise use the **File System** button) and enter the required filename

Click the **Read** button. The dialogue box will now disappear and the contents of the specified memory will be read and output to the specified file(s).

D. Installing the FTDI Device Driver for USB Connections

The first time that you make a USB connection between your PC and a kit component which features the FTDI FT232 chip, you may be prompted to install the device driver for the chip on your PC. You will need to obtain the driver from the Internet, but the method required to obtain the driver differs between Windows and Linux, as described below.

Note: To perform the installation, a device or cable containing an FTDI chip must be connected to a USB port of your PC.

Windows

You must obtain the driver for your operating system from the VCP drivers page of the FTDI web site:

www.ftdichip.com/Drivers/VCP.htm

Download the required driver to your desktop and double-click on its icon to install.

Linux

The FTDI driver must be downloaded and compiled manually. For example, on Ubuntu 14.04 the driver can be downloaded, compiled and installed by running the following commands:

```
sudo apt-get install build-essential cmake swig libusb-1.0-0-dev
wget http://www.intra2net.com/en/developer/libftdi/download/libftdi1-1.2.tar.bz2
tar xvf libftdi1-1.2.tar.bz2
cd libftdi1-1.2

mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX="/usr/" ../
make
sudo make install
```

 © NXP Laboratories UK 2016

Revision History

Version	Date	Comments
1.0	18-Feb-2016	First public release
1.1	11-July-2016	Updated information on how to obtain LPCXpresso
1.2	5-Oct-2016	• Added LPCXpresso download and installation information, including note about the LPCXpresso version • Updated references to the LCPXpresso documentation set

© NXP Laboratories UK 2016

Important Notice

Limited warranty and liability - Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

Right to make changes - NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use - NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications - Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control - This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

NXP Semiconductors

For online support resources and contact details of your local NXP office or distributor, refer to:

www.nxp.com

 © NXP Laboratories UK 2016