Software Developer's Kit Installation and User Guide

JN-UG-3064
Revision 4.4
6 February 2015

Contents

About this Manual

This manual provides guidance on installing and using the Software Developer's Kits (SDKs) for the NXP JN51xx range of microcontrollers, targeted at wireless network applications. The Eclipse Integrated Development Environment (IDE) is provided as a component of the SDKs and part of this manual is devoted to using Eclipse in the development of applications for JN51xx devices.

Note 1: The SDKs described in this manual cover the NXP JN516x family of microcontrollers - for example, the JN5168 device. Device variants may also be referred to - for example, JN5168-001.

Note 2: The JN516x SDKs for some protocols have moved to the 'BeyondStudio for NXP' toolchain. The installation of these SDKs and the toolchain is described in the *BeyondStudio for NXP Installation and User Guide (JN-UG-3098)*.

Organisation

The manual is divided into three parts:

- Part I: Introduction and Installation comprises three chapters:
  - Chapter 1 introduces the JN51xx SDKs, including their contents and the wireless network protocols that they support.
  - Chapter 2 describes how to install the JN51xx SDK Toolchain (JN-SW-4041), which must be installed first.
  - Chapter 3 describes how to install the JN516x SDK Libraries for ZigBee Smart Energy and ZigBee RF4CE (the SDKs for all other protocols have moved to the 'BeyondStudio for NXP' toolchain - see Note 2 above).
- Part II: Eclipse Integrated Development Environment comprises three chapters:
  - Chapter 4 introduces the Eclipse platform and describes how to install NXP external components into Eclipse.
  - Chapter 5 describes how to create a project in Eclipse and build an application to be run on the JN51xx device.
  - Chapter 6 describes how to download a built application to the Flash memory of a JN51xx-based module or board.
- Part III: Appendices comprises three appendices providing useful procedures that may be required during installation or use of the SDK.

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.

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

Acronyms and Abbreviations

API	Application Programming Interface
CLI	Command Line Interface
GUI	Graphical User Interface
HA	Home Automation
IDE	Integrated Development Environment
IP	Internet Protocol
ISR	Interrupt Service Routine
JenNet	Jennic Network layer protocol
JenOS	Jennic Operating System
LPRF	Low-Power Radio Frequency
MAC	Media Access Control
SDK	Software Developer's Kit
SE	Smart Energy
ZLL	ZigBee Light Link

ZPS ZigBee PRO Stack

Related Documents

JN-UG-3007 JN51xx Flash Programmer User Guide

JN-UG-3048 ZigBee PRO Stack User Guide

JN-UG-3059 ZigBee Smart Energy User Guide

JN-UG-3074 ZigBee RF4CE Stack User Guide

Support Resources

To access JN516x support resources such as SDKs, Application Notes and User Guides, visit the Wireless Connectivity TechZone:

www.nxp.com/techzones/wireless-connectivity

Trademarks

Part I:
Introduction and Installation

1. Overview

The Software Developer's Kits (SDKs) described in this manual are designed to aid software development for the NXP JN51xx microcontrollers, targeted at wireless network applications based on protocols such as ZigBee PRO (including Smart Energy) and ZigBee RF4CE. This chapter introduces the SDKs and the software development options that they provide, before the SDK installation instructions are given in Chapter 2 and Chapter 3.

Note: The JN516x SDKs for some protocols have moved to the 'BeyondStudio for NXP' toolchain. The installation of these SDKs and the toolchain is described in the *BeyondStudio for NXP Installation and User Guide (JN-UG-3098)*.

***Caution**: Before installing an SDK described in this manual, you must remove any previous JN51xx SDK version other than the JN5139 SDK.*

A list of the available JN51xx SDK installers is provided in Section 1.1.

The SDKs support application development within the Eclipse Integrated Development Environment (IDE), which is supplied as part of each SDK. Guidance on the use of Eclipse is provided in Part II: Eclipse Integrated Development Environment of this manual.

1.1 SDK Installers

A number of SDK installers are available to download free-of-charge (see "Support Resources" on page 7). The required installers depend on the wireless network protocol for which you wish to develop applications.

The following installer is required for all SDK installations described in this manual and must be used first:

- **JN51xx SDK Toolchain (JN-SW-4041):** This installs the NXP software tools that you will use to prepare wireless network applications for JN51xx devices. These utilities include development, compiler and Flash programming tools. For more details and installation instructions, refer to Chapter 2.

One of the following SDK Libraries installers should then be used, according to the required protocol:

- **JN516x ZigBee Smart Energy SDK (JN-SW-4064):** This installs the NXP software for the development of ZigBee PRO wireless network applications based on the Smart Energy (SE) profile for JN516x devices. The installer includes stack software, profile software, libraries and configuration editors. For more details and installation instructions, refer to Chapter 3.
- **JN516x ZigBee RF4CE (JN-SW-4060):** This installs the NXP software for the development of ZigBee RF4CE wireless network applications for JN516x devices. The installer includes stack software and libraries. For more details and installation instructions, refer to Chapter 3.

To complete the set-up of your application development environment, you may also need to install plug-ins which provide configuration editors and other external tools in the Eclipse IDE. This installation is described in Chapter 4.

Note: The JN516x SDKs for some protocols (e.g. JenNet-IP) have moved to the 'BeyondStudio for NXP' toolchain. The installation of these SDKs and the toolchain is described in the *BeyondStudio for NXP Installation and User Guide (JN-UG-3098)*.

1.2 Installation Pre-requisites

This section details the pre-requisites for your wireless network application development.

Before installing the desired SDK, make sure you have the following:

- A machine with the following specification:
  - Windows 7, Vista or XP operating system
  - At least 240 MB of hard disk space available
- Administrator rights on the machine
- The following SDK installers:
  - **JN-SW-4041-SDK-Toolchain-v*X*.*Y*.exe**
  - **JN-SW-4064-ZigBee-SmartEnergy1.*x*-v*XYZ*.exe** or **JN-SW-4060-ZigBee-RF4CE-v*XYZ*.exe**

Note 1: The SDK installers for the JN516x devices can be downloaded from the NXP Wireless Connectivity TechZone - see "Support Resources" on page 7.

Note 2: If the JN5139 SDK (installers JN-SW-4030 and JN-SW-4031) is already installed then you can choose to uninstall it or not. However, **any other** previous JN51xx SDK installation must be removed before installing the new SDK.

2. JN51xx SDK Toolchain Installation

This chapter describes how to install the JN51xx SDK Toolchain (JN-SW-4041), which must be installed before any other SDK component.

The SDK Toolchain installer is provided as the following file:

JN-SW-4041-SDK-Toolchain-v*X*.*Y*.exe

This installer is available from the NXP Wireless Connectivity TechZone (see "Support Resources" on page 7). It is supplied in a ZIP file which also contains the installer's Release Notes.

Note 1: If the JN5139 SDK (installers JN-SW-4030 and JN-SW-4031) is already installed then you can choose to uninstall it or not. However, **any other** previous JN51xx SDK installation must be removed before installing the new SDK.

Note 2: If you are currently using another version of the SDK, you are recommended to back up your **SDK/ Application** folder before installing the new SDK.

Note 3: The installed SDK Toolchain will be shared by all relevant JN51xx SDK installations on your machine (if you intend to have multiple SDK installations).

Caution: *Do not install the SDK Libraries or any other SDK components until you have installed the SDK Toolchain, as described in this chapter.*

2.1 Toolchain Contents

The software components that can be installed from the JN51xx SDK Toolchain are listed in the table below.

Component	Description
Cygwin	This is the Cygwin Command Line Interface (CLI) which emulates Linux. The SDK contains an NXP edition of Cygwin with reduced functionality. You can use this as a standalone development environment, if you wish, but it is also needed for the JN51xx compiler tools and for Eclipse. You must install this component, unless you already have a full Cygwin installation on your machine. Also refer to the important Cygwin information below this table.
Eclipse	This is the graphical Integrated Development Environment (IDE) used to develop applications for the JN51xx device. For more information on Eclipse, refer to Part II: Eclipse Integrated Development Environment of this manual.
Flash Programmer	This is the JN51xx Flash Programmer that you will need to download your built applications to the Flash memory used by the JN51xx device. You should use v1.8.6 or later of this application. If your SDK Toolchain installer does not contain a suitable version, you should use the standalone version, available separately (in JN-SW-4007). The Flash programmer (SDK version) is available from within the Eclipse environment.
JN51xx Compiler Tools	These tools include the JN51xx compiler and linker, which are always needed. The tools will be installed into the **Tools** directory within the **Jennic** installation folder and can be called from the Cygwin command line or from within Eclipse.

Table 1: Toolchain Components

It is important to note the following in relation to the installation of Cygwin:

- If you already have a **full** Cygwin environment on your machine, there is no need to install the NXP Cygwin environment from the SDK and you are advised **not** to do so, as the new Cygwin installation will overwrite the registry settings of the previous installation.
- If you intend to keep an earlier JN5139 SDK Toolchain installation (JN-SW-4031) that includes the Jennic/NXP edition of Cygwin, you should still install Cygwin as part of the new SDK installation as this SDK provides an updated version of the NXP Cygwin environment.

In both of the above cases, you should ensure that your path settings refer to the correct Cygwin and SDK installations, and that the paths are in the appropriate order.

Note: In addition to the above components, the SDK Toolchain contains the FTDI device driver for use with the JN51xx evaluation kits. If required, install this driver on your PC as described in Appendix B.

2.2 Toolchain Installation Procedure

To install the SDK Toolchain on your machine:

Step 1 Remove any previous JN51xx SDK installation (other than a JN5139 SDK installation from JN-SW-4030 and JN-SW-4031, which need not be removed) from your machine using the **Uninstall** option from the Windows **Start** menu or via **Add or Remove Programs** in **Control Panel**.

Step 2 Open Windows Explorer and check whether there is an existing **C:\Jennic** directory (or equivalent, if the SDK was previously installed somewhere other than the default location). If there was an existing **Application** folder, or if there were extra plug-ins installed, these may still be present in the **C:\Jennic** directory. If so, delete any unwanted remnants from the **C:\Jennic** directory. Also, back up the **Application** folder if you want to re-use the application files in the new set-up.

> **Note:** The default installation directory is **C:\Jennic** but for a new installation you can specify another drive/path (e.g. **D:\Appdata\NXP**). This Toolchain installation will be shared by all JN51xx SDK installations on your machine (if you will have multiple SDK installations).

Step 3 Launch the installer **JN-SW-4041-SDK-Toolchain-v*X.Y*.exe** on your machine. The Jennic Toolchain Setup wizard will start.

Step 4 Follow the on-screen instructions of the set-up wizard until you reach the **Choose Components** screen:

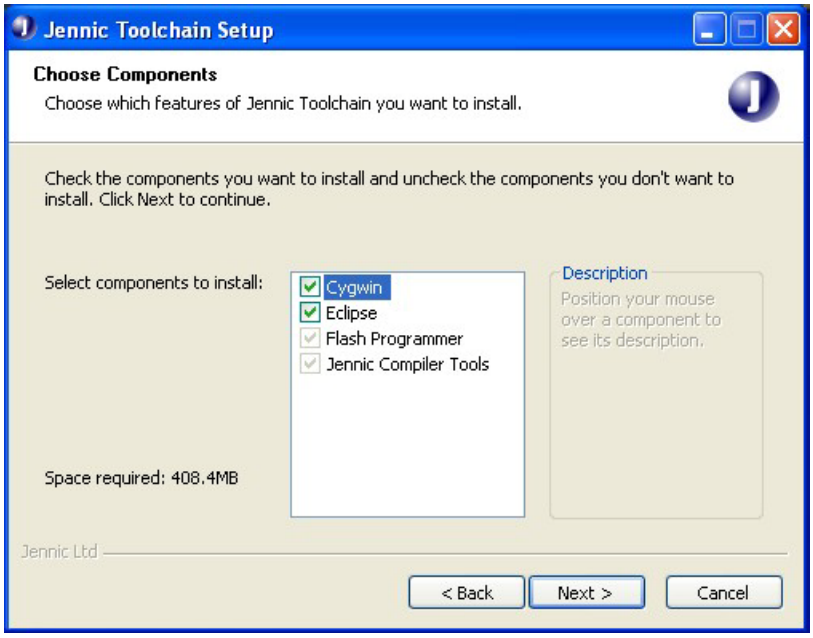

Figure 1: Choose Components Screen

By default, all the components are selected. De-select any component(s) that you do not wish to install. In particular, you should:

- de-select Cygwin if you already have a **full** Cygwin installation on your machine (see Section 2.1), otherwise leave it selected.
- de-select Eclipse if you already have the Eclipse IDE installed (although you can have more than one installation of Eclipse, if you wish).

A Cygwin installation is required on your machine, even if you wish to develop your applications using Eclipse. Refer to Section 2.1 for further information on the components.

Click **Next** to continue.

Step 5 In the next screen, choose the location where you want to install the tools:

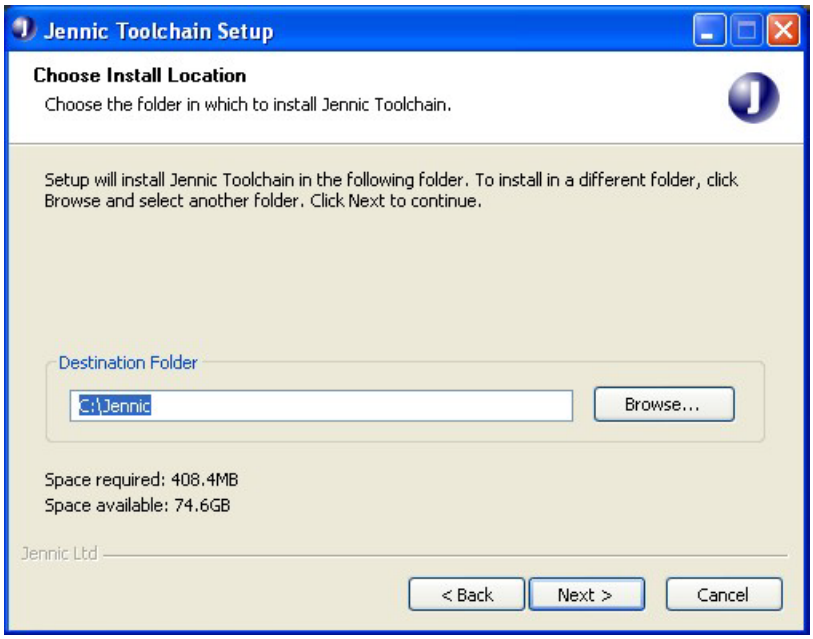

Figure 2: Choose Install Location Screen

The set-up wizard will automatically insert the installation directory. By default, this is **C:\Jennic**. If required, you can specify another drive/path (e.g. **D:\Appdata\NXP**).

Click **Next** to continue.

Step 6 In the next screen, specify the folder in which you want the installed tools to appear in the Windows **Start** menu. By default, this is set to **Jennic**.

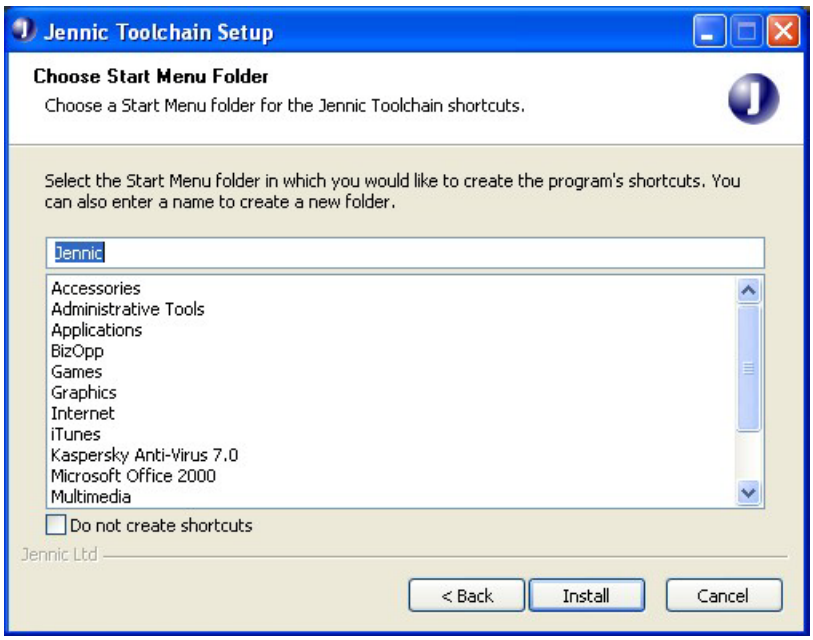

Figure 3: Choose Start Menu Folder Screen

Click **Install**.

Step 7 Wait for the installation to complete (this may take several minutes) and then click **Next** followed by **Finish**.

Step 8 Re-start your computer when prompted to do so.

Step 9 Continue to Chapter 3, Chapter 4 or Chapter 5 in order to install the SDK Libraries.

3. JN516x SDK Libraries Installation

This chapter describes how to install the SDK Libraries for developing wireless network applications for the JN516x-001 devices (also see Section 1.1). This SDK component must be installed after the SDK Toolchain (see Chapter 2).

A separate SDK Libraries installer is available for each of the wireless network protocols which are supported on the JN516x devices (and which use the installed SDK Toolchain). This chapter covers all the following installers:

- ZigBee PRO Smart Energy SDK (JN-SW-4064):

  JN-SW-4064-ZigBee-SmartEnergy1.*x*-v*XYZ*.exe

- ZigBee RF4CE SDK (JN-SW-4060):

  JN-SW-4060-ZigBee-RF4CE-v*XYZ*.exe

The above installers are available from the NXP Wireless Connectivity TechZone (see "Support Resources" on page 7). Each is supplied in a ZIP file which also contains the installer's Release Notes. The contents of the installers are listed in Section 3.1.

Note: The JN516x SDKs for some protocols have moved to the 'BeyondStudio for NXP' toolchain. The installation of these SDKs and the toolchain is described in the *BeyondStudio for NXP Installation and User Guide (JN-UG-3098)*.

Caution: *Do not install the relevant SDK Libraries until you have installed the SDK Toolchain, as described in Chapter 2.*

3.1 Contents of JN516x SDK Libraries

The software components that are installed from the JN516x SDK Libraries are listed in the sub-sections below for each of the supported protocols:

3.1.1 ZigBee Smart Energy SDK (JN-SW-4064)

Components	Comments
ZigBee PRO stack software and APIs	Needed for ZigBee PRO applications that run on the JN516x devices of a wireless network
ZigBee Smart Energy (SE) profile and APIs	Needed for ZigBee PRO SE applications
ZigBee Cluster Library (ZCL) and APIs	Needed for ZigBee PRO SE applications
JenOS (Jennic Operating System) and APIs	Needed for managing operating system resources
JN516x Integrated Peripherals and Board APIs	Needed for hardware control (JN516x and evaluation kit boards)
802.15.4 Stack API	Needed for developing applications directly on top of IEEE 802.15.4 stack layers
Eclipse plug-ins	ZPS and JenOS Configuration Editors

Table 2: Contents of ZigBee Smart Energy SDK

3.1.2 ZigBee RF4CE SDK (JN-SW-4060)

Components	Comments
ZigBee RF4CE stack software and APIs	Needed for ZigBee RF4CE applications that run on the JN516x devices of a wireless network
ZigBee Remote Control (ZRC) profile API	Can be used by consumer electronics remote control applications
ZigBee Input Device (ZID) profile API	Can be used to facilitate the use of Human Interface Devices
802.15.4 Stack API	Needed for developing applications directly on top of IEEE 802.15.4 stack layers
JN516x Integrated Peripherals and Board APIs	Needed for hardware control (JN516x and evaluation kit boards)

Table 3: Contents of ZigBee RF4CE SDK

3.2 Installation Procedure

The following procedure describes how to install the JN516x SDK Libraries for any of the supported protocols (the screenshots show JenNet-IP as an example, although this protocol has been moved to a new SDK for the 'BeyondStudio for NXP' toolchain).

Step 1 Ensure that you have installed the SDK Toolchain, as described in Chapter 2.

Step 2 Launch the relevant SDK Libraries installer on your machine - one of:

- **JN-SW-4064-ZigBee-SmartEnergy1.*x*-v*XYZ*.exe**
- **JN-SW-4060-ZigBee-RF4CE-v*XYZ*.exe**

The SDK Setup wizard will start.

Step 3 Follow the on-screen instructions of the set-up wizard. When you reach the **Choose Components** screen, you will not be able to select individual components, since the wizard always installs all components.

Click **Next** to continue.

Step 4 In the next screen, choose the location where you want to install the libraries:

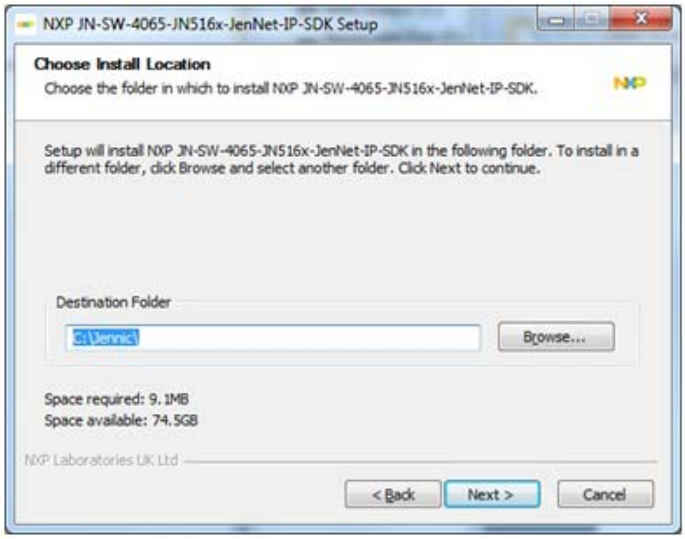

Figure 4: Choose Install Location Screen

The set-up wizard will automatically insert the installation directory. By default, this is **C:\Jennic**. If required, you can specify another drive/path (e.g. **D:\Appdata\NXP**).

Note: If installing more than one set of 'SDK Libraries', you should specify a different installation path for each SDK in order to avoid conflicts - for example, **D:\Appdata\NXP4060** and **D:\Appdata\NXP4064**.

Click **Next** to continue.

Step 5 In the next screen, specify the folder in which you want the libraries to appear in the Windows **Start** menu. By default, this set to **Jennic**.

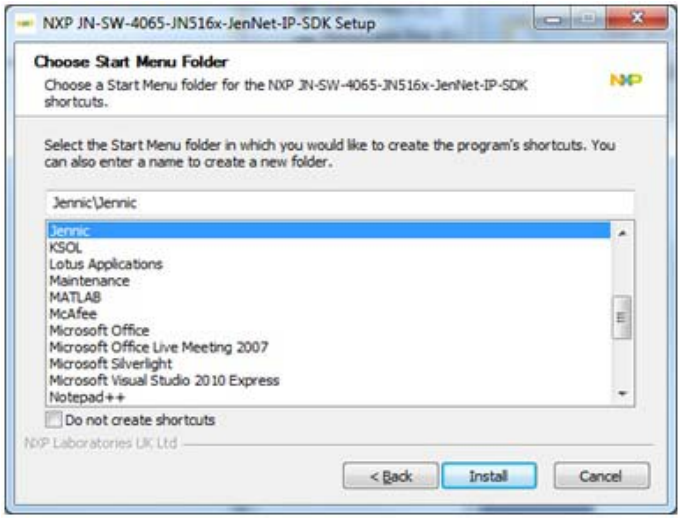

Figure 5: Choose Start Menu Folder Screen

Click **Install**.

Step 6 Wait for the installation to complete and then click **Finish**.

Step 7 Continue to Chapter 4 in order to install the required external components into Eclipse.

Part II:
Eclipse Integrated Development Environment

4. Getting Started in Eclipse

The Eclipse Integrated Development Environment (IDE) is installed as a component of the SDK Toolchain (JN-SW-4041) and is intended as the main development platform for designing wireless network applications for the NXP JN516x microcontrollers.

It is important to work through this chapter to fully prepare your Eclipse installation before embarking on your JN51xx application development within Eclipse.

4.1 Introduction to Eclipse

Eclipse is an open-source development platform, originally developed by IBM and now supported by the Eclipse Foundation (www.eclipse.org). The platform provides a fully-featured integrated environment for developing and building software applications, and is rapidly becoming the accepted standard IDE for use within the embedded software community.

The chapters of Part II: Eclipse Integrated Development Environment of this manual describe how to:

- Create an Eclipse project for your application (from an NXP template)
- Edit your application code using the Eclipse editor
- Build your application, to produce a binary file
- Download your binary file to the device that is to run the application

Note: While this manual provides guidance on using Eclipse in developing JN51xx applications, full user documentation for Eclipse is available on the Eclipse web site (**www.eclipse.org**).

NXP supply external tools and plug-ins for Eclipse (which are installed as described in Section 4.2). These add-ons have been developed and tested with the Ganymede version of Eclipse, which is the version supplied in the SDK Toolchain.

***Caution:** NXP do not guarantee that the supplied external tools and plug-ins will work properly with any version of Eclipse other than Ganymede.*

4.2 Installing External Components into Eclipse

Once you have installed the SDK, as described in Part I: Introduction and Installation, you will need to install various NXP external tools and plug-ins for the Eclipse IDE by following the procedures in this section.

The Eclipse external tools and plug-ins are outlined below.

External Tools

The external tools are provided in the SDK Toolchain and are as follows:

- Flash programmer CLI tool
- Flash programmer GUI tool
- Jennic/NXP Bash Shell

Plug-ins (ZigBee PRO only)

The plug-ins are configuration editors that are required for developing ZigBee PRO applications. They are provided as part of the JN516x SDK Libraries for the ZigBee profiles.

They are:

- **ZPS Configuration Editor:** This editor provides a convenient way to set ZigBee network parameters, such as the properties of the Co-ordinator, Routers and End Devices (for example, by setting elements of the device descriptors). For more information on this editor, refer to the *ZigBee PRO Stack User Guide (JN-UG-3048)*.
- **JenOS Configuration Editor:** This editor provides a graphical interface for configuring the way an application uses JenOS resources, such as timers, mutexes and ISRs. For more information on this editor, refer to the *JenOS User Guide (JN-UG-3075)*.

Note: Building an application requires the configuration tool command line utilities, which are provided in the SDK Libraries installer and were installed as part of the procedure that you followed in Chapter 3 or Chapter 4.

4.2.1 Installing the External Tools

To install the external tools into Eclipse, follow the procedure below:

Step 1 Start Eclipse, either from the Windows **Start** menu or by double-clicking on the **eclipse.exe** executable file in your 'Eclipse' directory (e.g. **C:\Jennic\Tools\eclipse**).

You will be presented with a workspace selection dialogue box (see the figure below). The workspace is the directory where all your projects will be stored. It can be anywhere, but it is advised that you re-direct it to the standard development directory to keep it consistent with the SDK, e.g. **C:\Jennic\Application**. Also, tick the box **Use this as the default and do not ask again**.

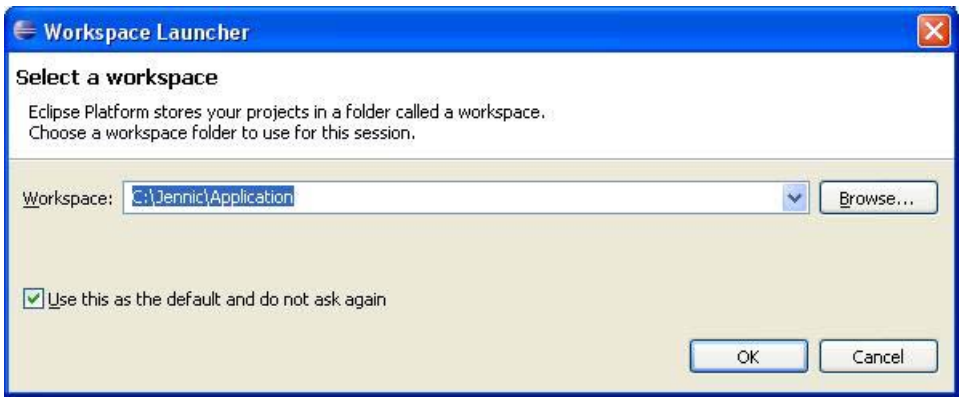

Figure 6: Workspace Launcher

Step 2 Click **OK**. If starting Eclipse for the first time, the initial start-up screen will appear.

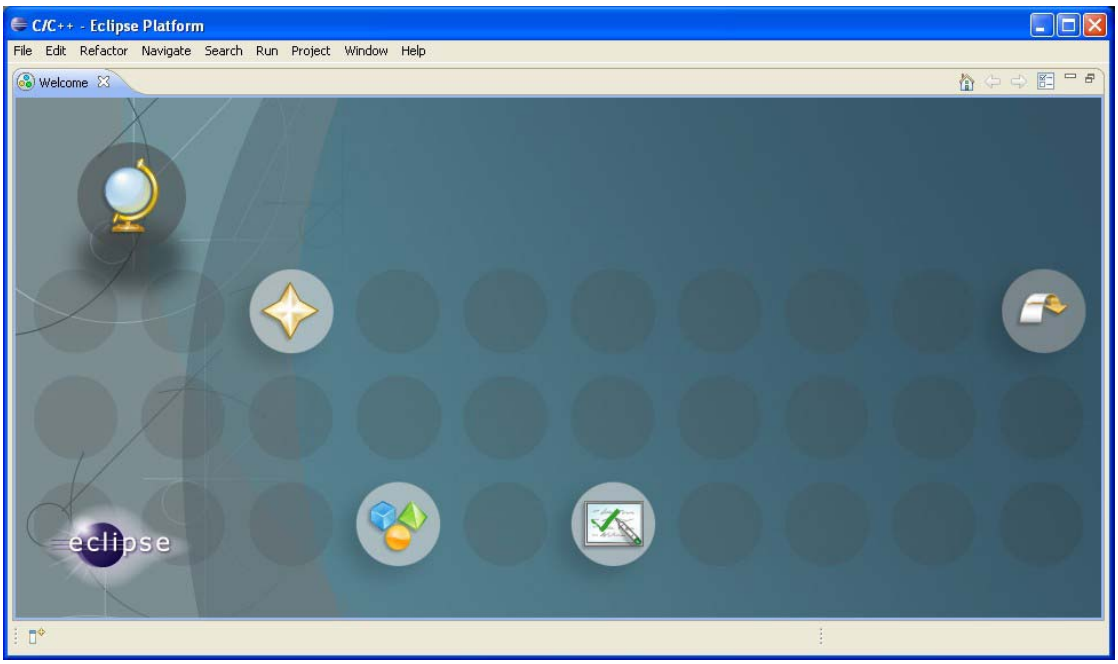

Figure 7: Eclipse Initial Start-up Screen

Step 3 Close this down by simply clicking the X on the Welcome tab. The display changes to the Eclipse main screen.

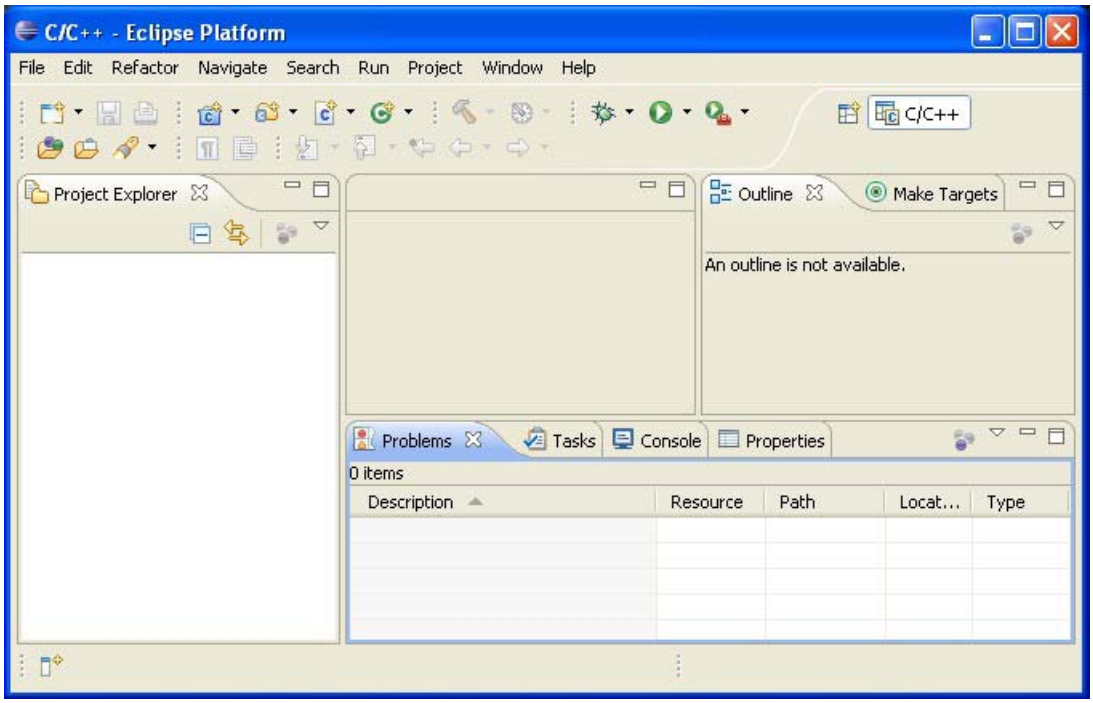

Figure 8: Eclipse Main Screen

Step 4 From the main menu, select **File > Import**. This opens the **Import** dialogue box.

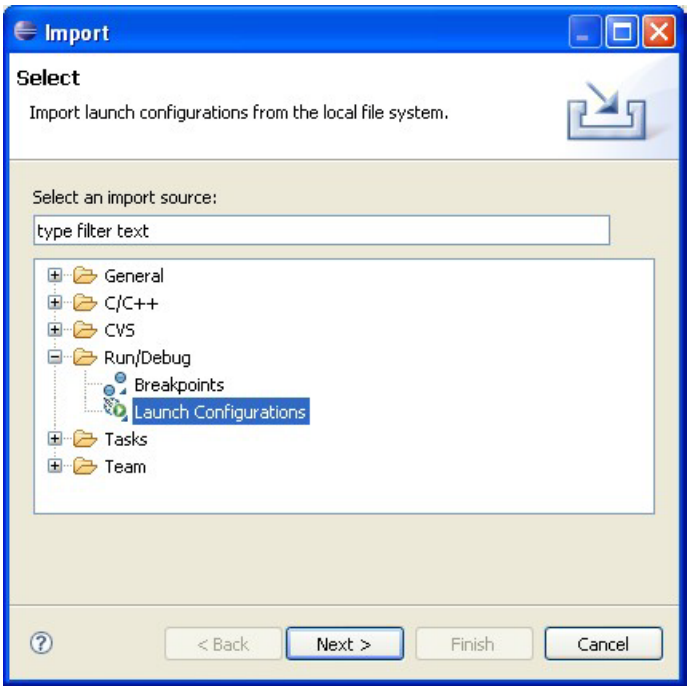

Figure 9: Import Dialogue Box

Step 5 Expand the **Run/Debug** folder and select **Launch Configurations**.

Step 6 Click **Next**. This opens the **Import Launch Configurations** dialogue box.

Step 7 Click **Browse**, browse to **C:/Jennic/Tools/eclipse_config**, click to select the folder and then click **OK**.

eclipse_config then appears in the left pane of the **Import Launch Configurations** dialogue box.

Step 8 Tick the check-box next to **eclipse_config**. All of the available **.launch** files appear in the right pane of the dialogue box - see Figure 10.

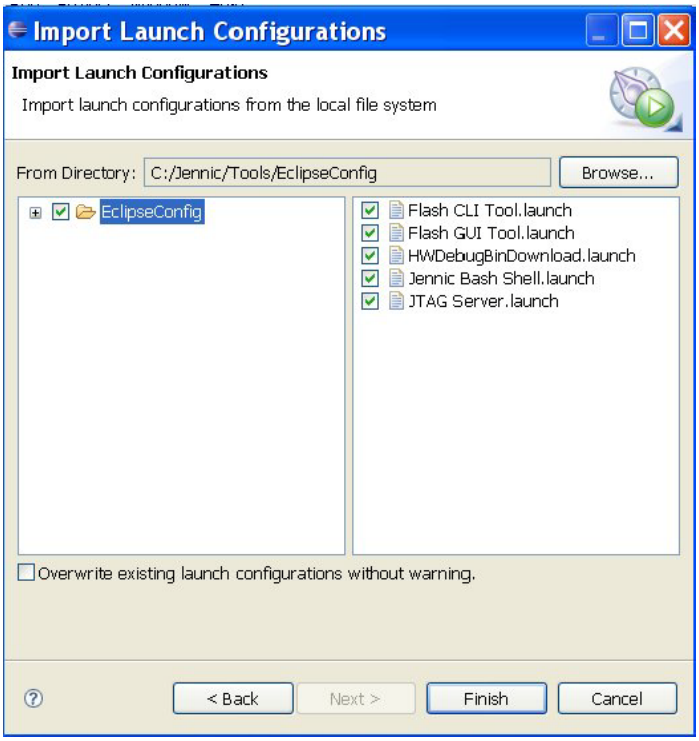

Figure 10: Import Launch Configurations

Step 9 Check that all the required **.launch** files are selected, namely:

- **Flash CLI Tool.launch**
- **Flash GUI Tool.launch**
- **HWDebugBinDownload.launch**
- **Jennic Bash Shell.launch**
- **JTAG Server.launch**

Step 10 Click **Finish**. The external tools are now automatically installed.

When the installation has finished, you should find five tools available in the **Run > External Tools** menu. They can also be accessed from the drop-down arrow next to the tools symbol  on the toolbar.

4.2.2 Installing the Configuration Editors (ZigBee PRO only)

If developing ZigBee PRO applications, you will need the plug-ins for the ZPS and JenOS Configuration Editors. To install these plug-ins, follow the procedure below:

Step 1 Start Eclipse (if not already started).

Step 2 In the Eclipse main menu, select **Help > Software Updates**, then select the **Available Software** tab.

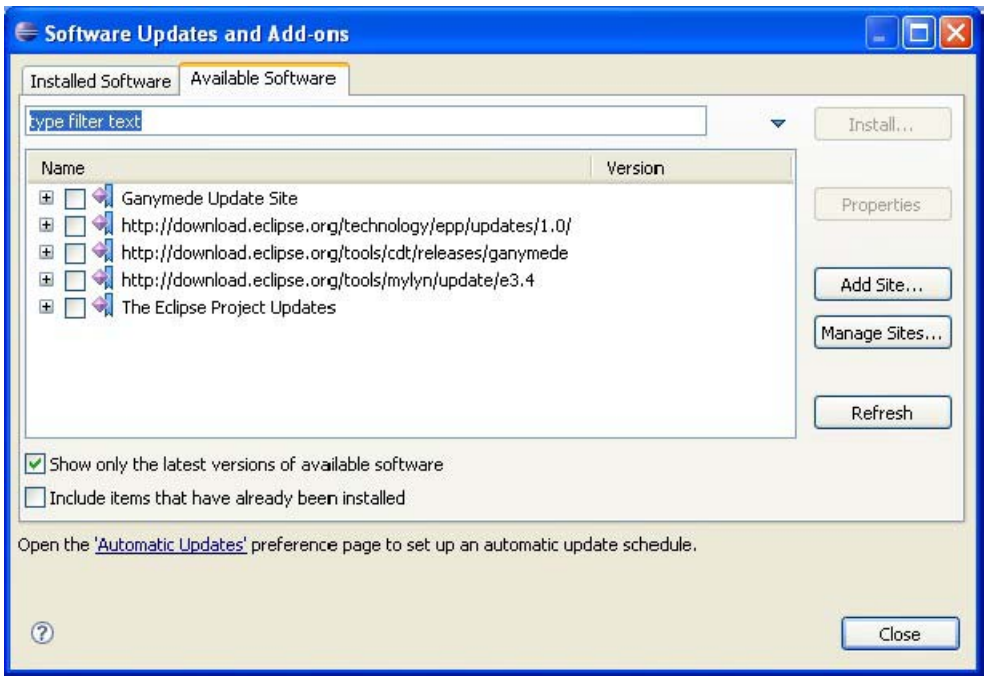

Figure 11: Available Software Tab

Step 3 Select **Add Site**.

Step 4 In the **Add Site** pop-up window, enter the local path to the plug-ins sub-folder of the **elipse** folder (e.g. under **C:\Jennic\Tools\Eclipse_plugins**)

Step 5 Click **OK**. The list of plug-in sites updates automatically.

Step 6 Expand the location of the relevant plug-ins and then expand **Jennic ZBPro SDK**. Under **Jennic ZBPro SDK**, ensure that the check-boxes next to **Jennic RTOS Configuration Editor** and **Jennic ZBPro Configuration Editor** are ticked, then click **Install**.

Step 7 Wait for **Calculating requirements and dependencies** to complete - this may take a few minutes.

> **Note**: Eclipse often begins its own update process automatically when user installations are being performed. You may see evidence of this in the bottom left-hand corner of the Eclipse screen, as it reports on the files that it has downloaded. This is the reason why Step 7 takes several minutes.

Step 8 When the **Install** window appears, click **Next**.

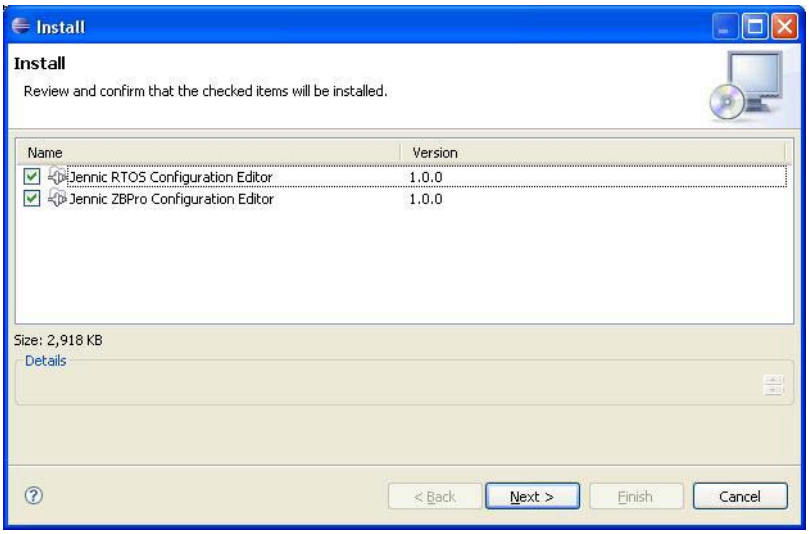

Figure 12: Software Updates Install Screen

Step 9 Read the terms of the license agreement (see the figure below) and click the button to accept.

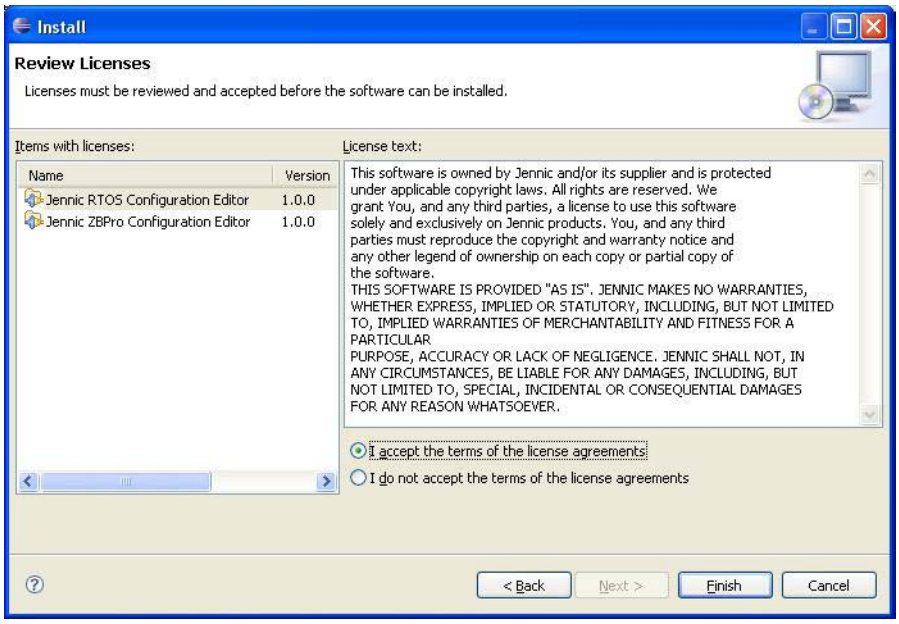

Figure 13: Review Licenses Screen

Step 10 Click **Finish** so that Eclipse installs the plug-ins that you selected. If desired, click the button for **Run in Background**.

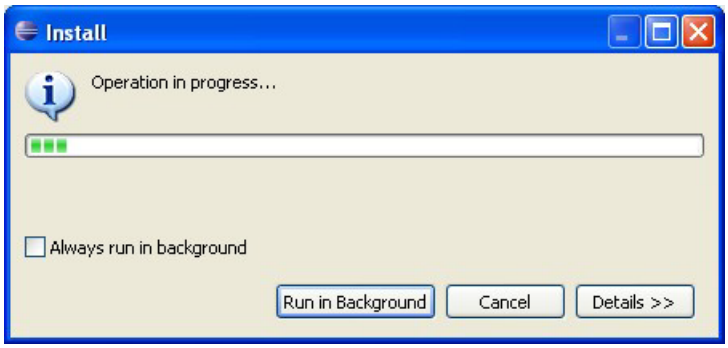

Figure 14: Installing Software Updates

Step 11 Once the plug-ins have been installed, a screen will appear which requests the acceptance of certificates for the two plug-ins. To accept them, tick the relevant boxes and click **OK**.

Step 12 Eclipse now needs to re-start to incorporate the new plug-ins. Only Eclipse itself will reboot, not the entire machine. Click **Yes** to allow the re-start.

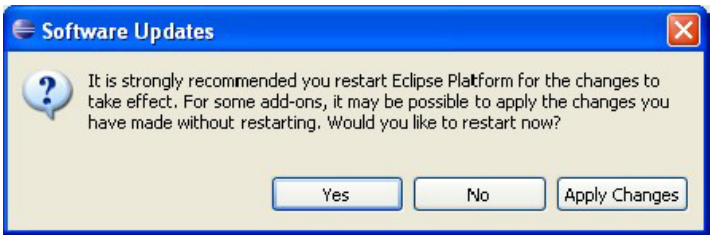

Figure 15: Software Updates Re-start Window

Step 13 You are now returned to the Eclipse main screen.

For information on how to use Eclipse, refer to Chapter 5.

For information on how to use the configuration editors, refer to the:

- *ZigBee PRO Stack User Guide (JN-UG-3048)* for the ZPS Configuration Editor
- *JenOS User Guide (JN-UG-3075)* for the JenOS Configuration Editor

5. Creating and Building Eclipse Projects

This chapter describes how to use Eclipse to create and then build your own applications to run on the JN51xx device. It is assumed that:

- You have installed Eclipse as part of the SDK Toolchain (JN-SW-4041), as described in Chapter 2
- You have installed the required SDK Libraries, as described in Chapter 3
- You have installed the NXP external tools and configuration editor plug-ins for Eclipse, as described in Section 4.2

5.1 Eclipse Projects and Templates

In Eclipse, an application under development is termed a project. The creation of an Eclipse project described in this manual involves importing an NXP project template to use as a starting point. For ZigBee Smart Energy and ZigBee RF4CE, you will need to use the example applications provided in the Application Notes JN-AN-1135 and JN-AN-1158 respectively (for access details, refer to "Support Resources" on page 7).

An Eclipse project folder is installed in the workspace directory that you specified when you first ran Eclipse - this should have been when you installed the NXP external tools into Eclipse, as described in Section 4.2.1.

Projects with their folders and files are displayed in a tree view in the **Project Explorer** panel on the left of the Eclipse main window. Project files can be displayed and manipulated in the Eclipse edit panel by selecting the appropriate tab.

The rest of this chapter describes:

- How to create an Eclipse project from an NXP template (see Section 5.2)
- How to work with files in an Eclipse project (see Section 5.3)
- How to build the application stored in a project (see Section 5.4)

5.2 Creating/Importing a Project (from an NXP Template)

This section describes how to create an Eclipse project for a wireless network application by importing a project template provided by NXP.

Step 1 Download the required application template (see Section 5.1). Open the **.zip** file and extract it to your workspace directory, e.g. **C:\Jennic\Application**.

If using WinZip, ensure that the **Use folder names** tickbox is ticked.

Step 2 Start Eclipse, either by double-clicking on the desktop shortcut (if set up) or from the Windows **Start** menu. This should take you to your workspace that you created when you first ran Eclipse to install the NXP external tools - see Section 4.2.1.

Step 3 From the Eclipse main menu, select **File > Import**. This opens the **Import** dialogue box.

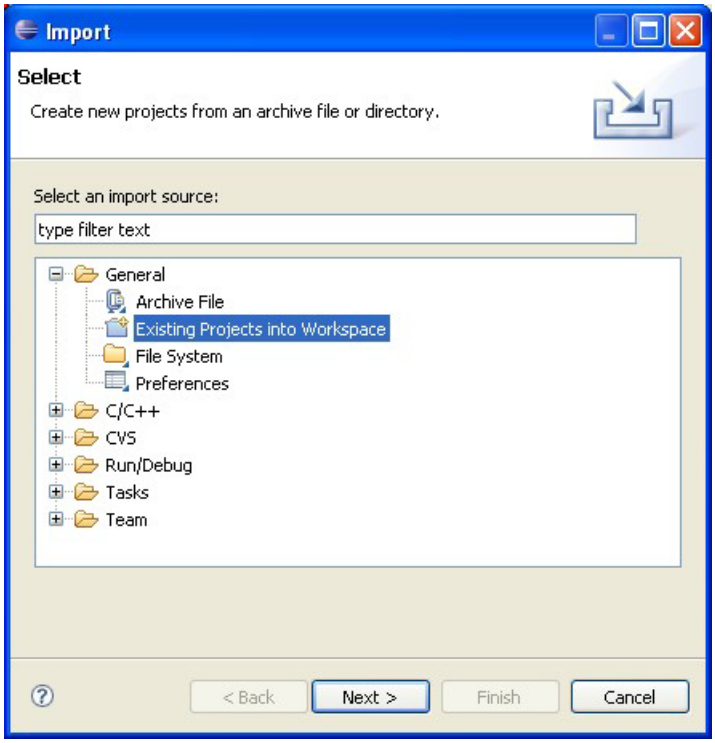

Figure 16: Import Screen

Step 4 Expand **General** and select **Existing Projects into Workspace**.

Step 5 Click **Next** and then navigate down to select your **Application** root folder in the **Browse For Folder** dialogue box.

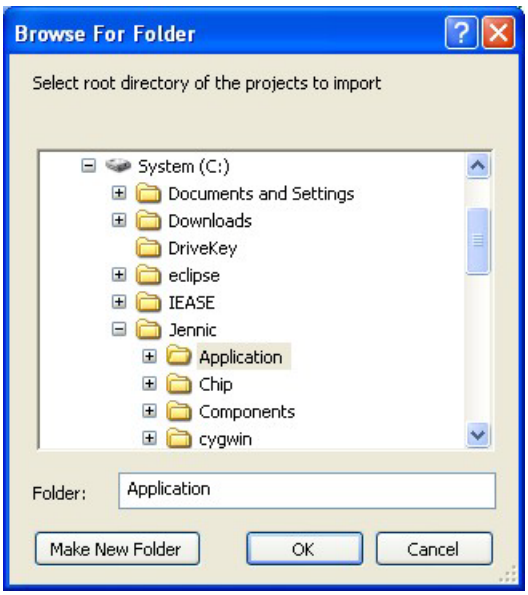

Figure 17: Browse for Application Folder

Step 6 Click **OK**.

Step 7 In the **Import** dialogue box, tick the project you want to import (untick any other projects), then confirm by clicking **Finish**. As an example, the screenshot below shows the ZigBee PRO application template.

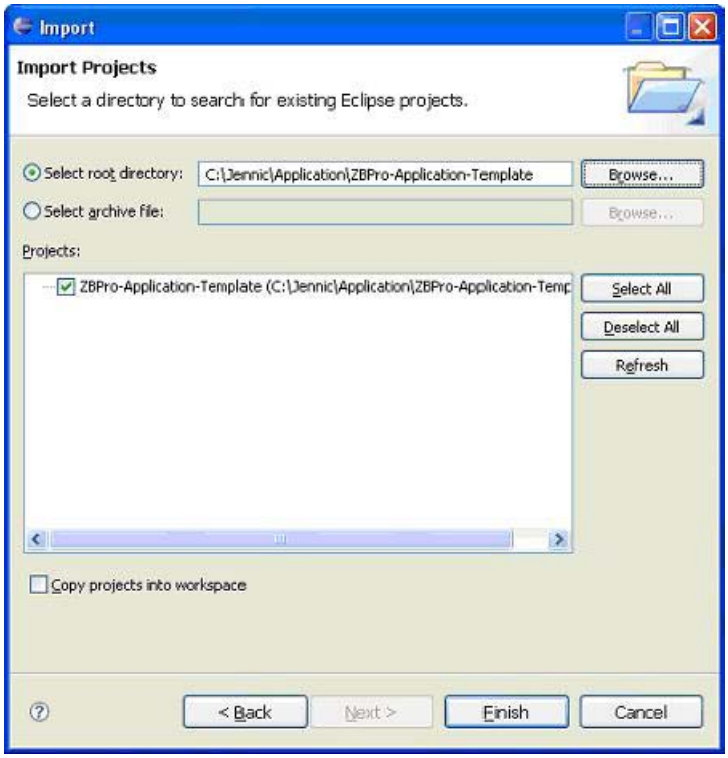

Figure 18: Import Project

Step 8 Wait a moment while the project is imported into your workspace. The project should appear in the left **Project Explorer** panel.

Your project should now include the following folders:

- **Includes** folder, containing the required library folders
- **Coordinator** folder, containing:
  - a **Build** directory which contains the makefile for the Co-ordinator
  - a **Source** directory which contains the source files for the Co-ordinator
- **Router** folder, containing:
  - a **Build** directory which contains the makefile for a Router
  - a **Source** directory which contains the source files for a Router
- **SleepingEndDevice** folder, containing:
  - a **Build** directory which contains the makefile for a Sleeping End Device
  - a **Source** directory which contains the source files for a Sleeping End Device
- **Common** folder, containing a **Source** directory which contains source files that common to the Co-ordinator, Router and End Device.
- **Doc** folder, containing the Application Note document.

Step 9 At this point, you should adapt the Eclipse project according to the needs of your application, including changing the project name:

- To re-name a project or source file, right-click on the project or file in the **Projects Explorer** view and, from the pop-up menu, select **Rename** and enter the new name. Then edit the Application Source section of the associated makefile to reflect the new name.
- To change the name of the application binary file that will result from a build, edit the associated makefile and change the Target definition as illustrated below:

  ```
  TARGET = myTargetName
  ```

- To add new source files (if any), follow the procedure in Appendix A. Then edit the associated makefile and add the new source file to the Application Source section as illustrated below:

  ```
  APPSRC += myNewSource.c
  ```


5.3 Working on Your Project

Once you have created your project, you can work on your application using Eclipse as the editor. The makefile as well as the C-code application file and any header files can be edited using Eclipse.

To edit your code, follow the procedure below:

Step 1 If the required project is not already open (if it has been closed since it was created), expand the project by clicking on the **+** symbol next to the project in the **Project Explorer** panel. This displays the project tree - see Figure 19. Similarly, click on the **+** symbol next to the **Source** folder.

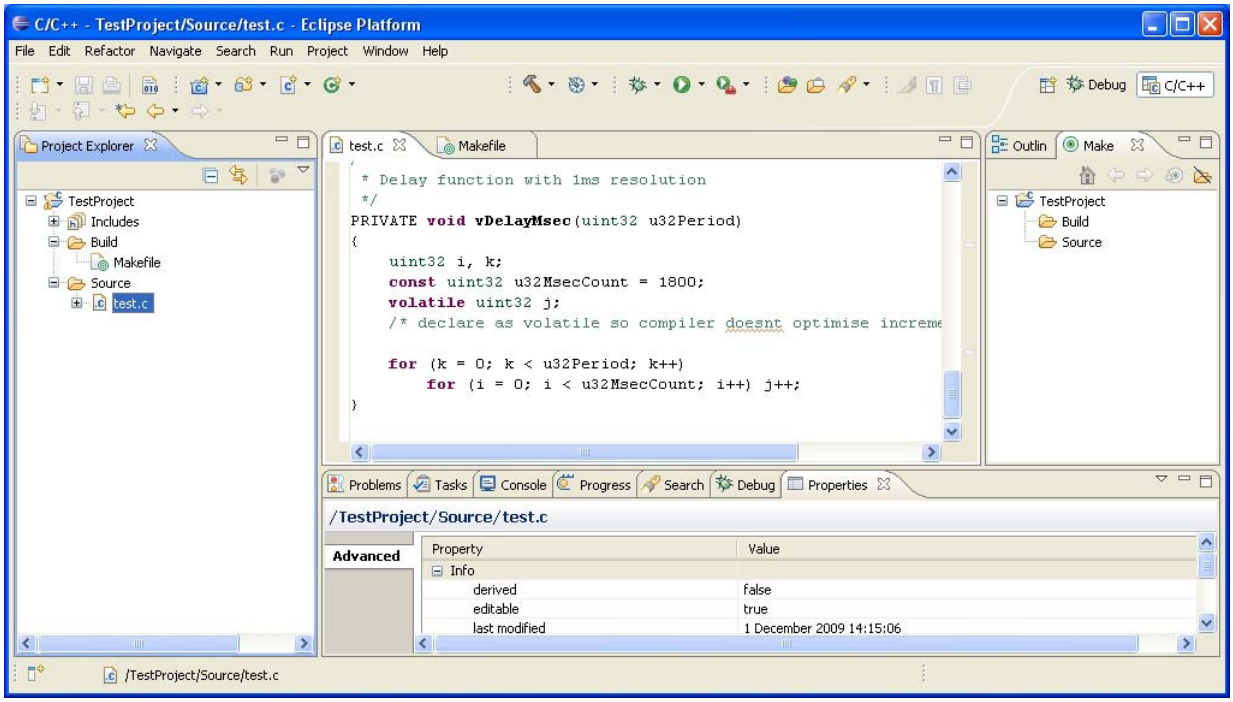

Figure 19: Editing Your C Project

Step 2 Double-click on the file to be edited in the **Project Explorer** panel. This displays a tab in the centre edit panel - see Figure 19.

Step 3 If required, rename the **.c** source file by right-clicking on it in the **Project Explorer** panel and selecting **Rename** from the pop-up menu.

The **Rename Resource** screen appears. Enter the new filename and then click **OK**.

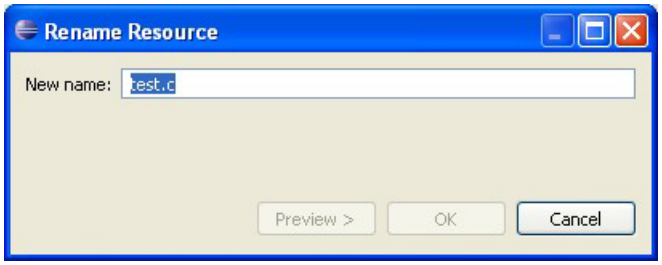

Figure 20: Rename Resource

Step 4 You can now edit the code in the main panel. If you prefer to use a different editor, right-click on the file to be edited in the **Project Explorer** panel and select **Open With** from the pop-up menu. This gives a choice of editors.

Step 5 When you have finished editing the **.c** source file, ensure that you save your changes (for example, by following the menu path **File > Save**) and then close the file (for example, by following the menu path **File > Close**).

You are recommended to save your changes regularly while editing.

Step 6 Once you have finished working on the project, save the project changes (for example, by following the menu path **File > Save All**) and close the project (for example, by following the menu path **File > Close All**).

Note: Make sure that you update the project makefile to contain the new filename specified above.

5.4 Building Your Project

Building your project can be performed simply within the Eclipse environment, as follows.

Step 1 Ensure that your makefile is present and complete (see Section 5.2), and that your editing is complete (see Section 5.3).

Step 2 Build your application by either of the following methods:

- Click the 'hammer' icon  on the toolbar - the application will then build automatically.
- In the **Projects Explorer** or **C/C++ Projects** view on the left, right-click on the relevant project and, from the pop-up menu, select **Build Project** - the application will then build automatically.

Step 3 Any errors/warnings created by the make process will be displayed in the **Problems** tab at the bottom of the screen. Standard output can be seen under the **Console** tab.

Step 4 The project binaries will be created as **.bin** files in the **Build** folder.

6. Downloading an Application Binary

Once you have built your project (as described in Section 5.4), you must download the binary output to the Flash memory used by the JN51xx device that is to run the code. This chapter describes how to perform the download.

You must use a Flash programmer to download your application's binary file to the Flash memory of the target device. Eclipse does not have a built-in Flash programmer, but the JN51xx Flash Programmer (supplied in the SDK Toolchain) can be run from Eclipse via the **External Tools** menu. As well as a GUI version of the Flash programmer, there is a command line version (CLI) which can be programmed for each target chip and build type (Debug or Release).

Note: The SDK Toolchain may not contain the latest version of the JN51xx Flash Programmer, but the latest version is available separately as a standalone utility (JN-SW-4007). When developing for JN516x platforms, you should always use the latest version. However, the standalone version cannot be launched from Eclipse.

For more information about the JN51xx Flash Programmer, refer to the procedure for downloading binary code in the *JN51xx Flash Programmer User Guide (JN-UG-3007)*.

6.1 Pre-requisites

Ensure that you have the following:

- A target device containing a JN51xx microcontroller.
- A serial cable and dongle allowing connection between your PC and the target device
- The **.bin** file to be downloaded – following a build, this file is placed in the **Build** directory for the project

In order to access the Flash programmer from within Eclipse, the **External Tools** menu must have been set up as described in Section 4.2.1.

6.2 Download Procedure

To download your **.bin** file to a device:

Step 1 Connect a USB port of your PC to the target device using an NXP-supplied 'USB A to Mini B' cable or 'USB-to-serial' cable. In the latter case, make sure you connect the black wire of the cable to Pin 1 of the serial connector on the target device. If prompted to install a device driver, refer to Appendix B.

Step 2 In Eclipse, follow the menu path **Run > External Tools > External Tools Configurations**.

This displays the **External Tools Configurations** dialogue box, containing a list of tools - see Figure 21.

> **Note:** You will need to set the PC communications port that has been assigned to the connection to the target device – to identify this port, refer to Appendix C.

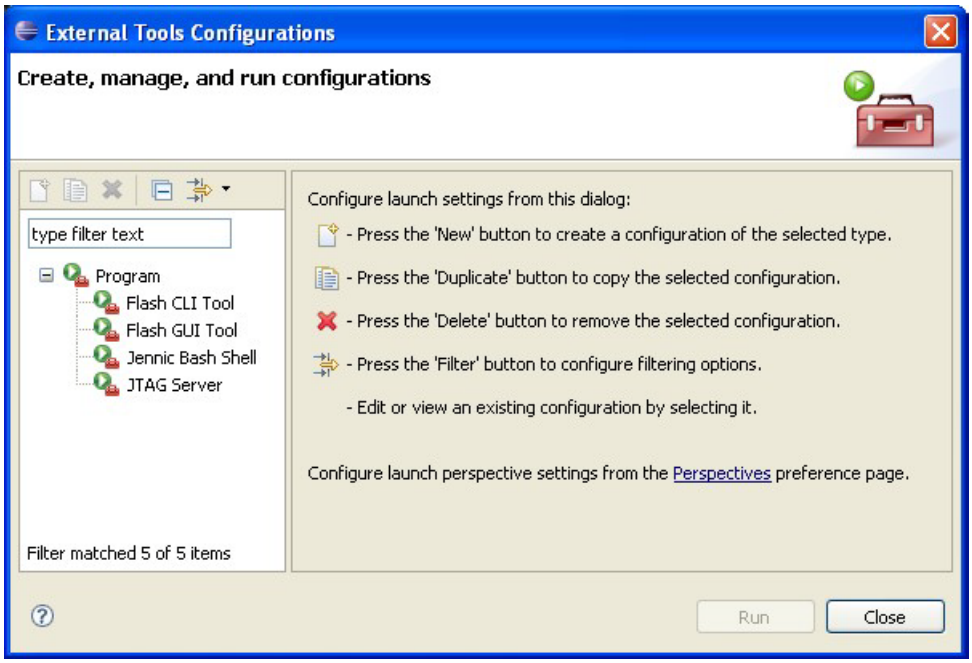

Figure 21: External Tools Configurations

Step 3 Choose the Flash programmer that you want to use - either the Flash CLI tool or the Flash GUI tool. Click to highlight it. The window changes, as illustrated in Figure 22 for the Flash GUI.

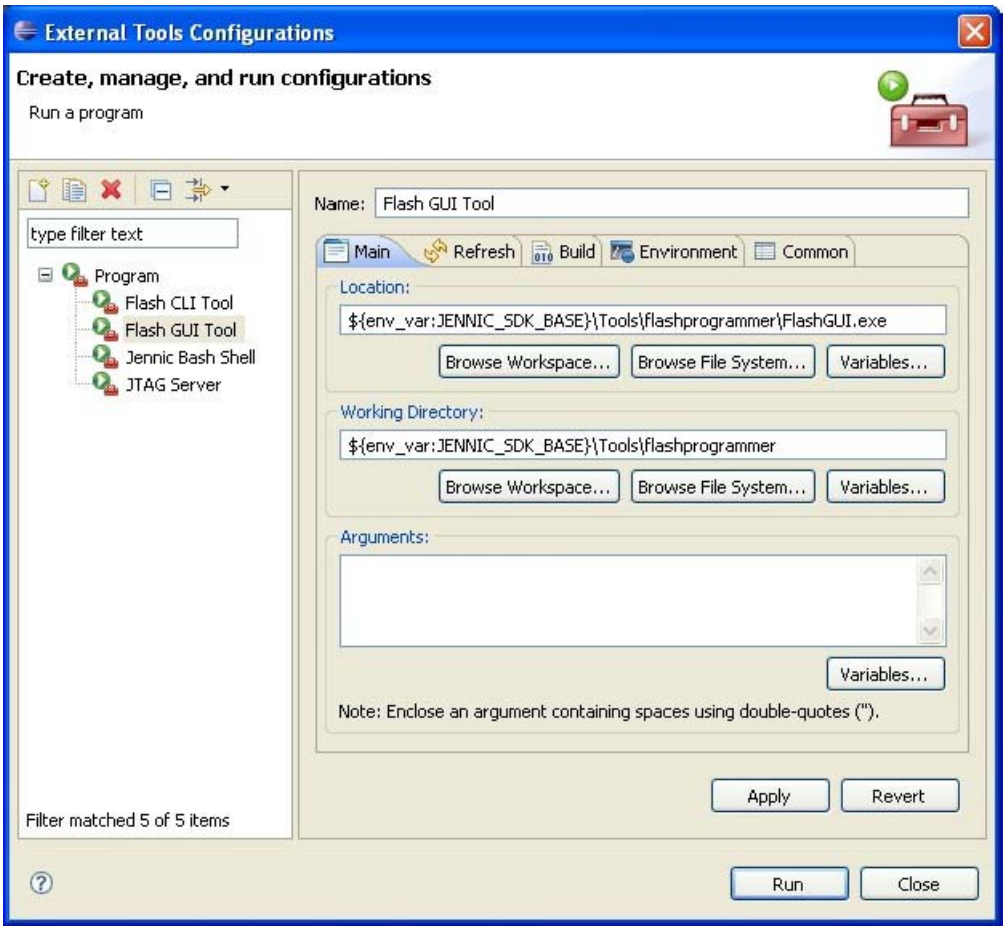

Figure 22: Flash GUI Configuration

Step 4 Click **Run** to run the tool.

> **Note:** After the tool has been run for the first time, it will appear as an option in the **Run > External Tools** menu. It is only necessary to access **Run > External Tools > External Tools Configuration** again if you need to change the parameters.

- If you selected the Flash GUI tool then the Flash Programmer GUI window is displayed. For further instructions, refer to the *JN51xx Flash Programmer User Guide (JN-UG-3007)* - you will need to continue from Step 3 of the download procedure for the Flash Programmer GUI.
- If you selected the Flash CLI tool then you will first be prompted to specify the communications port and binary file for the download - identifying the relevant port is described in Appendix C. For further instructions, refer to the *JN51xx Flash Programmer User Guide (JN-UG-3007)* - you will need to continue from Step 6 of the download procedure for the Flash Programmer CLI.

Step 5 Once the download has finished, disconnect the device from the PC and power-cycle the device.

Part III: Appendices

A. Creating an Eclipse Project Source File

The procedure below describes how to add a new C source file to an Eclipse project.

Step 1 In your project in Eclipse, expand the project name folder so that the required **Source** folder (in which the new source file will go) is visible and click on it to highlight it.

Step 2 To add a file to the **Source** folder, from the main menu select **File > New > Source File**. The **New Source File** dialogue box appears. As an example, the screenshot below shows a new source file called **test.c**.

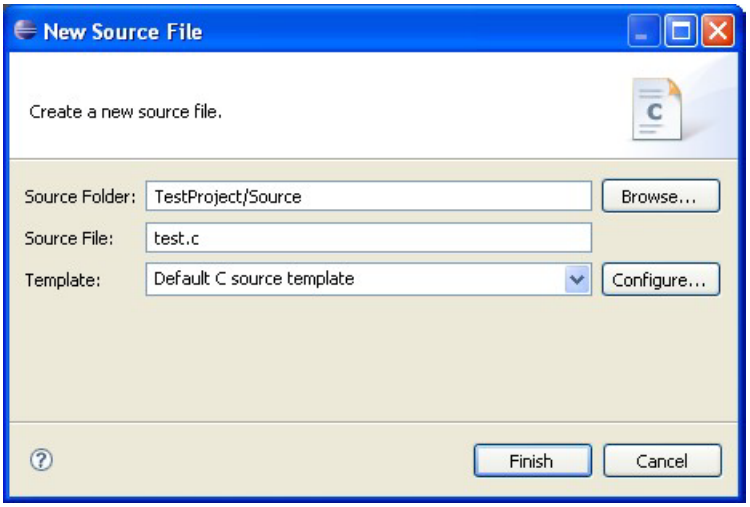

Figure 23: New Source File

Step 3 Enter the parameters as follows:

- **Source Folder**: This field should be automatically completed.
- **Source File**: Enter the name of the source file you want to create, e.g. **test.c**.
- **Template**: Select **Default C source template** from the drop-down menu.

Step 4 Click **Finish**. The new source file appears in the **Project Explorer** panel.

Step 5 The content of your new source file can be viewed and edited by clicking on the tab (e.g. **test.c**) in the centre panel.

Step 6 Edit the source file, as required.

B. Installing the FTDI Driver

The first time that you make a USB connection from your PC to or via a component which features the FTDI FT232 chip, you may be prompted to install the device driver for this chip on your PC. This driver is provided in the SDK Toolchain (JN-SW-4041). The installation of this driver is described below (although you may not need this procedure if Windows automatically finds the required driver on the Internet).

1. When you make the USB connection, check whether **Found new hardware wizard for TTL232r-3v3** is displayed on your PC.

   If this appears, you must install the driver by following the rest of this procedure. Otherwise, the driver is already installed.

2. Fill in the screen **Install from a specific location**, as follows:

   a) Select the radio button **Search for the best driver in these locations**.

   b) Tick the checkbox **Include this location in the search**.

   c) Using the **Browse** button, navigate to the directory **FTDI_drivers** in the installed SDK on your PC. For example, if the SDK has been installed on drive C of your PC, the required path will be:

   C:\Jennic\Tools\Drivers\FTDI_drivers

   d) Click **OK**.

   The wizard will automatically fill in the details in the drop-down search box.

3. In the **Found new hardware wizard** screen, click **Next**.

4. Wait for the wizard as it searches for and installs the new driver. On completion, it will display the message "Completing the Found new hardware wizard". Click **Finish** to complete.

   In some cases, you may need to repeat the procedure from Step 2, depending on your hardware configuration.

   Finally, the **Found new hardware** bubble will indicate that the hardware is installed and ready for use.

Note: Alternatively, you can obtain the relevant driver for your operating system from the FTDI web page **www.ftdichip.com/FTDrivers.htm**. Go to the VCP drivers, download the required driver to your desktop and double-click on its icon to install.

C. Identifying the PC Communications Port Used

When connecting your PC to a board, you need to find out which serial communications port your PC has allocated to the connection, as described below.

Step 1 In the Windows Start menu, follow the menu path:

Start > Control Panel > System

This displays the **System Properties** screen.

Step 2 In the **System Properties** screen:

a) Select the **Hardware** tab.

b) Click the **Device Manager** button

This displays the **Device Manager** screen.

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

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

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

Revision History

Version	Date	Comments
1.0	7-July-2009	First release
1.1	12-Mar-2010	Title of manual and SDK changed Jenie/JenNet added to protocol options Hardware debug .bin download added to Eclipse launch configurations Updates made concerning Cygwin installation advice and other minor issues
1.2	14-May-2010	References to Jenie/JenNet patch removed
1.3	17-Jun-2010	Minor modifications/corrections made
2.0	22-Nov-2010	Incorporated information from former *Eclipse IDE User Guide (JN-UG-3063)*
3.0	6-July-2012	Added JN-SW-4051 installer for JenNet-IP SDK
4.0	19-Dec-2012	Added installers for JN516x SDKs: JN-SW-4060, JN-SW-4062, JN-SW-4064, JN-SW-4065. Other minor updates also made
4.1	10-June-2013	Added JN516x SDK installer JN-SW-4067 for ZigBee Home Automation and made minor modifications/corrections
4.2	13-Feb-2014	Added JN516x SDK installer JN-SW-4063 for IEEE802.15.4 and made minor modifications/corrections
4.3	24-Sept-2014	Removed JN514x SDKs, JN516x IEEE802.15.4 SDK (JN-SW-4063) and JN516x JenNet-IP SDK (JN-SW-4065) *
4.4	6-Feb-2015	Removed JN516x ZigBee Light Link SDK (JN-SW-4062) and JN516x ZigBee Home Automation SDK (JN-SW-4067) *

* The removed JN516x SDKs have been migrated to new SDKs for use with the 'Beyond Studio for NXP' toolchain

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 the contact details of your local NXP office or distributor, refer to:

www.nxp.com

For online support resources, visit the Wireless Connectivity TechZone:

www.nxp.com/techzones/wireless-connectivity