QSG180: Zigbee EmberZNet Quick-Start Guide for SDK v7.0 and Higher

This quick-start guide provides basic information on configuring, building, and installing applications for the EFR32MG family of SoCs using the Zigbee EmberZNet Software Development Kit (SDK) v7.0 and higher with Simplicity Studio 5.3 and higher.

For similar information regarding earlier SDK versions, see QSG106: Zigbee EmberZNet Quick-Start Guide for SDK v6.10 and Earlier.

This guide is designed for developers who are new to Zigbee EmberZNet and the Silicon Labs development hardware. It provides instructions to get started using the example applications provided with the Zigbee EmberZNet SDK.

KEY FEATURES

- Product overview
- Setting up the development environment
- Installing software
- Creating an example application network
- Using the Network Analyzer

---

1   Product Overview

Before following the procedures in this guide you must have

- Purchased your EFR32MG Mesh Networking Kit (see https://www.silabs.com/wireless/zigbee).
- Downloaded the required software components, as described below. A card included in your development hardware kit contains a link to a Getting Started page, which will direct you to links for the Silicon Labs software products.

1.1   Note on Product History

Zigbee EmberZNet SDK v8 is our latest Zigbee stack introduced along with our newest embedded software platform, the Simplicity Software Development Kit (SDK). It is designed specifically for our Series 2 and Series 3 devices. The Simplicity SDK is a significant advancement in IoT development, providing our customers with a cohesive development environment for wireless innovation. It enables wireless developers to utilize advanced features and capabilities with the most recent Silicon Labs IoT devices. See https://github.com/SiliconLabs/simplicity_sdk for more information.

This document is based on Zigbee EmberZNet SDK v7, which contains a number of changes compared to SDK v6.x. Many of these changes are due to an underlying framework redesign that results in an improved developer experience within Simplicity Studio 5 (SSv5). Projects are now built on a component architecture instead of AppBuilder. Simplicity Studio 5 includes project configuration tools that provide an enhanced level of software component discoverability, configurability, and dependency management. These include a Project Configurator, a Component Editor, and a Zigbee Cluster Configurator. See *AN1301: Transitioning from Zigbee EmberZNet SDK 6.x to SDK 7.x* for more information about the differences between SDK 6.x and SDK 7.x.

SDKs beginning with version 6.8 are only compatible with Simplicity Studio 5 (SSv5). Among many other improvements, SSv5 introduced the *Simplicity Studio 5 User's Guide*, available online at https://docs.silabs.com/ and through SSv5's help menu. Standard information, such as how to download SSv5 and the Zigbee EmberZNet SDK, and descriptions of SSv5 features and functions, are provided in that guide, and are not repeated here.

1.2   Software

See the stack release notes for version restrictions and compatibility constraints for the stack and other software. Your Zigbee EmberZNet stack may be delivered as part of one of two SDKs:

- Zigbee EmberZNet v8 and above is included in the Simplicity SDK (SiSDK).
- Zigbee EmberZNet v7 is included in the Gecko SDK (GSDK).

This guide applies to both SDKs but will primarily reference the Gecko SDK. However, all steps are equally relevant for the Simplicity SDK.

To develop Zigbee EmberZNet applications, you will need the following.

- Simplicity Studio 5 (SSv5): Simplicity Studio is the core development environment designed to support the Silicon Labs IoT portfolio of system-on-chips (SoCs) and modules. It provides access to target device-specific web and SDK resources; software and hardware configuration tools; an integrated development environment (IDE); and advanced, value-add tools for network analysis and code-correlated energy profiling. When you install Simplicity Studio it will walk you through installing Gecko SDK, the suite of Silicon Labs SDKs, including the Zigbee EmberZNet SDK. Alternatively, Gecko SDK, including Zigbee EmberZNet, may be installed manually by downloading or cloning the latest from GitHub. See https://github.com/SiliconLabs/gecko_sdk for more information.
- The Zigbee EmberZNet stack, an advanced implementation of a Zigbee stack, installed as part of the Gecko SDK. The stack API is documented in an online API reference at https://docs.silabs.com/. The stack is delivered as a collection of libraries that you can link to your applications. A description of each library is provided in the development environment. The release notes contain details on the folders installed along with their contents.
- Simplicity Commander, installed along with Simplicity Studio. A GUI with limited functionality can be accessed through Simplicity Studio's Tools menu. Most functions are accessible through a CLI invoked by opening a command prompt in the Simplicity Commander directory (\SiliconLabs\SimplicityStudio\developer\adapter_packs\commander). See *UG162: Simplicity Commander Reference Guide* for more information.
- Compiler toolchain (see the SDK release notes for compatible versions):
  - GCC (The GNU Compiler Collection) is provided with Simplicity Studio. GCC is used in this document. However, you must use IAR to compile projects for a part with less than 512 kB, such as the EFR32xG1

    Note: Application images created with GCC are larger than those created with IAR. If you use GCC to compile the example applications in this SDK, you must use a part with at least 512 kB of flash.
 - IAR Embedded Workbench for ARM (IAR-EWARM).

Note: Download the compatible version from the Silicon Labs Support Portal, as described in section 1.6 Using IAR as a Compiler. Refer to the "QuickStart Installation Information" section of the IAR installer for additional information about the installation process and how to configure your license. Once IAR-EWARM is installed, the next time Simplicity Studio starts it will automatically detect and configure the IDE to use IAR-EWARM.

While Simplicity Studio and Simplicity Commander can be run on a Mac OS or Linux machine, these instructions assume you are working with a Microsoft Windows-based PC. If you are using a non-Windows system, IAR-EWARM must be run via WINE or some other form of emulator or virtual machine.

1.3   Support

You can access the Silicon Labs support portal at https://www.silabs.com/support through Simplicity Studio. Use the support portal to contact Customer Support for any questions you might have during the development process. Access is through the Welcome view under Learn and Support. Note that you can return to the Welcome view at any time through the Welcome button on the toolbar.
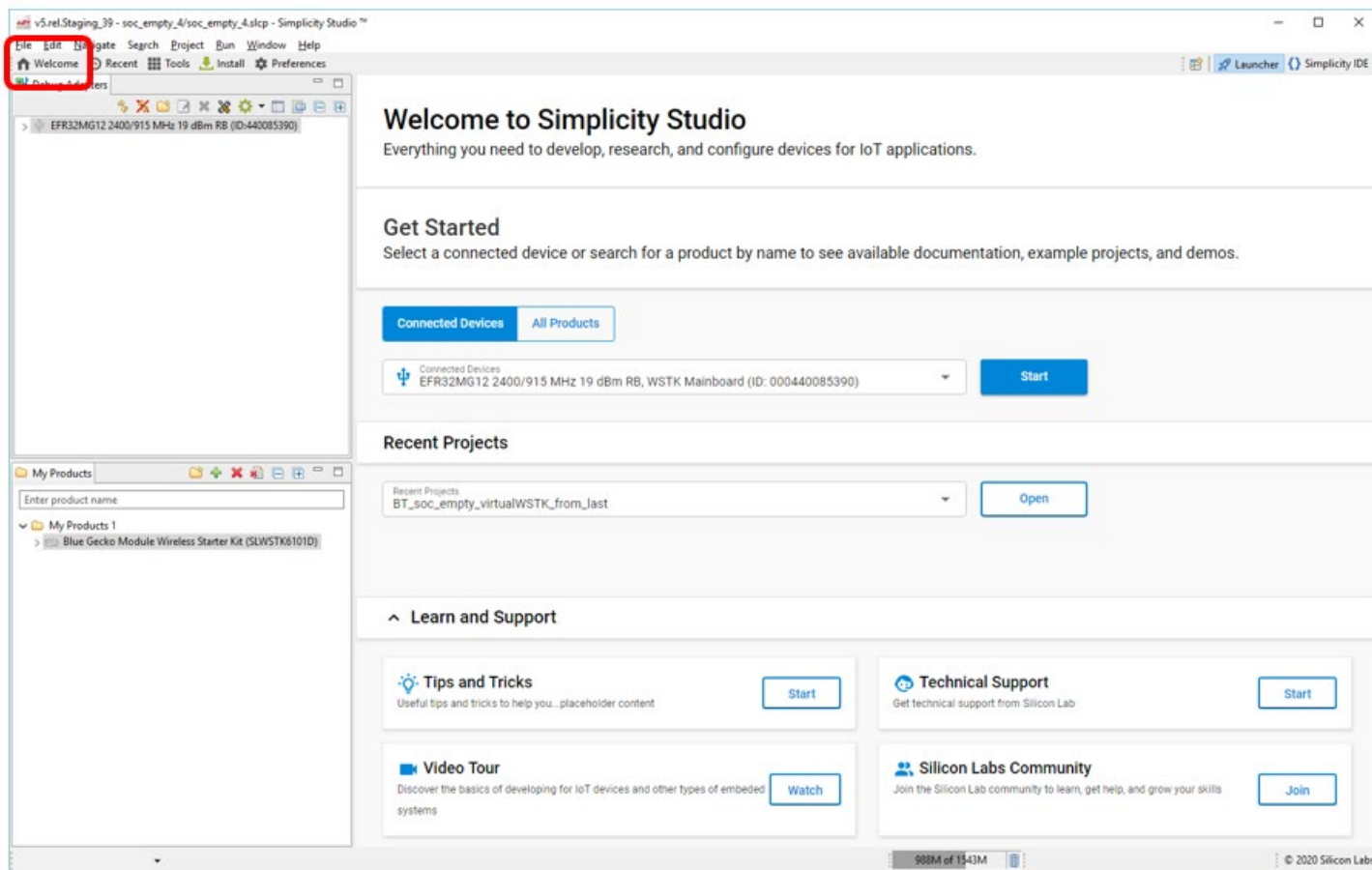
1.4   Documentation

Documentation is accessed through Simplicity Studio. Simplicity Studio filters documentation based on the connected or selected part. Hardware-specific documentation can be accessed through links on the part OVERVIEW tab.
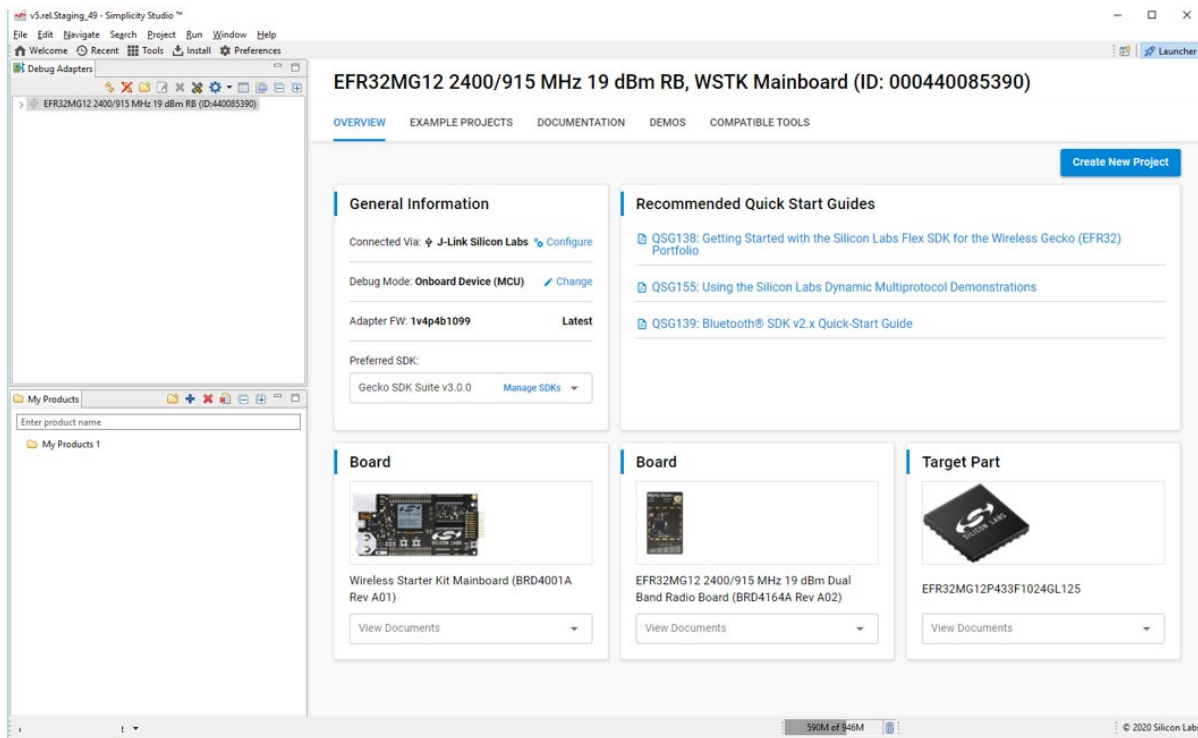
SDK documentation and other references are available through the DOCUMENTATION tab. Filter with the Zigbee Technology Type checkbox to see documentation most closely related to the Zigbee EmberZNet SDK.
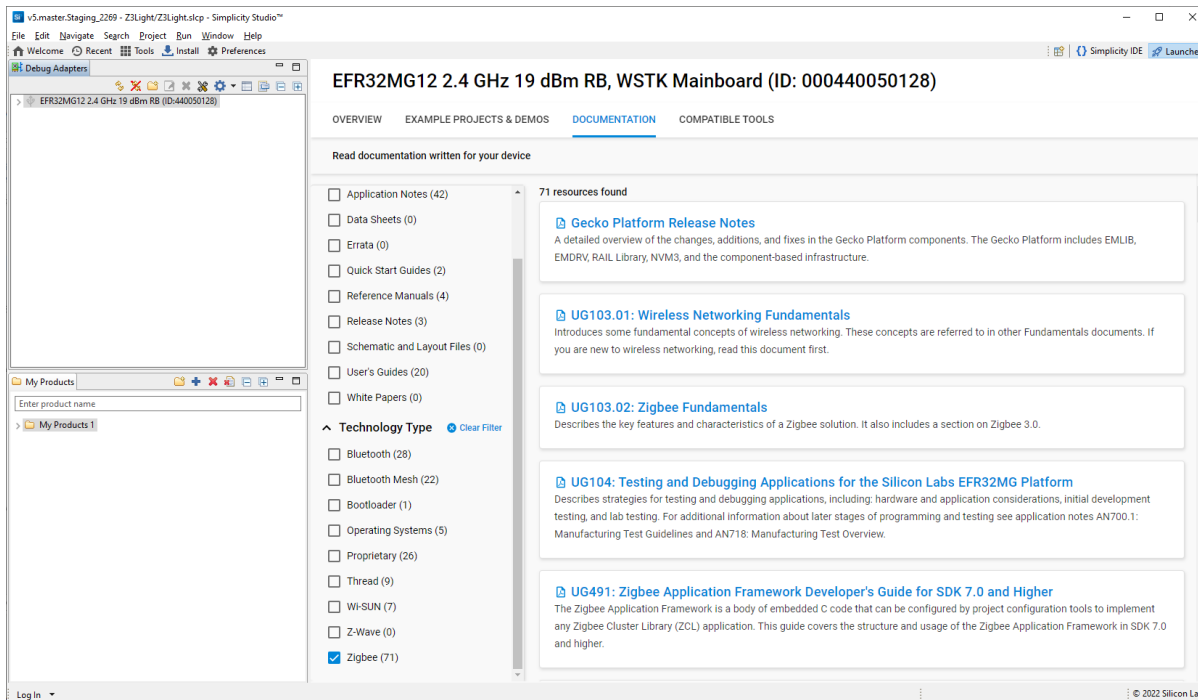
1.5   Gecko Platform

The Gecko Platform is a set of drivers and other lower-layer features that interact directly with Silicon Labs chips and modules. Gecko Platform components include EMLIB, EMDRV, RAIL Library, NVM3, and mbed TLS. Application developers using Zigbee EmberZNet components, the PIN tool, or APIs may not need to interact directly with the Gecko Platform, as the code does that for you. For more information about Gecko Platform, see the Gecko Platform release notes (on the DOCUMENTATION tab, filter by Resource Type: Release Notes).

1.6   Using IAR as a Compiler

If you plan to use IAR as your compiler (required for dynamic multiprotocol and Micrium OS examples), check the compatible IAR version in the SDK release notes. To install IAR-EWARM:

1.   Go to the Customer Support portal as described in section 1.3 Support.

2.   If you are not already signed in, sign in.

3.   Click the Software Releases tab. In the View list select **Latest EmberZNet Software**. Click **Go**. In the results is a link to the appropriate IAR-EWARM version.

4.   Download the IAR package. This is a large package - download time depends on connection speed but can take 1 hour or more.

5.   Install IAR.

6.   In the IAR License Wizard, click **Register with IAR Systems to get an evaluation license**.

7.   Complete the registration and IAR will provide an evaluation license.

2 Set Up Your Development Environment

2.1 Create a Support Account

Before you install Simplicity Studio, you need to create an account on the support portal. Be sure to record your account username and password as you will use it to log in to Simplicity Studio.

Properties in your Salesforce account determine what update notifications you will receive. To review or change your subscriptions, log in to the portal, click HOME to go to the portal home page and then click the Manage Notifications tile. Make sure that Software/Security Advisory Notices & Product Change Notices (PCNs) is checked, and that you are subscribed at minimum for your platform and protocol. Click **Save** to save any changes.

2.2 Connect the Mainboard

Connect your mainboard, with radio board mounted, to your PC using a USB cable.

Note: For best performance in Simplicity Studio, be sure that the power switch is in the Advanced Energy Monitoring or "AEM" position as shown in the following figure.
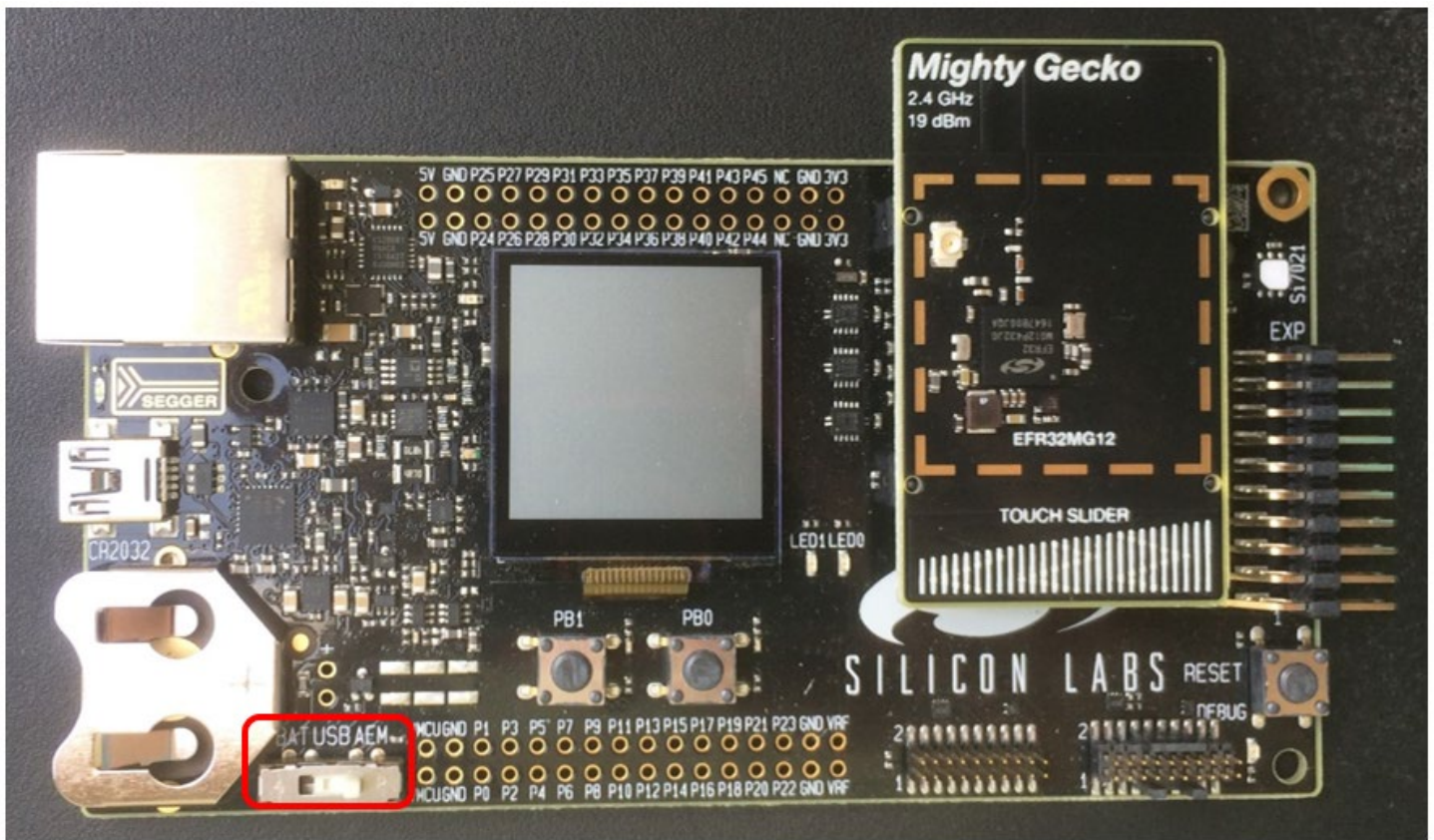
Figure 1. EFR32MG12 on a WSTK Mainboard

2.3 Install Simplicity Studio 5 (SSv5) and the Gecko SDK

General information on using SSv5 and the Gecko SDK installation can be found in the online Simplicity Studio 5 User's Guide.

3 About Demos and Examples

Because starting application development from scratch is difficult, the Zigbee EmberZNet SDK comes with a number of built-in demos and software examples covering the most frequent use cases.

Demos are pre-built application images that you can run immediately. Software examples can be modified before building the application image. The software examples with the same names as the demos provide the demo functionality.

Note: The demos and examples you see are determined by the part selected. If you are using a custom solution with more than one part, be sure to click on the part you are working with to see only those items applicable to that part.

Silicon Labs recommends that you start your own development with a use case-based example and modify it according to your needs. If none of the use case-based examples meet your needs, you can start with the **Zigbee – Minimal** example. The examples provide default configurations needed by the stack and a basic application structure that you can build upon.

3.1 Demos

Demos are prebuilt application examples that can be directly downloaded to your device. These demos are compatible with selected devices, as noted in the demo description. You will not see the demos in Simplicity Studio unless you have one of these devices selected. If you have more than one device connected, make sure that one of these devices is selected in the Debug Adapters view. To download and run a demo on your device, select your device and click the Example Projects & Demos tab in the Launcher perspective. Turn off the Example Projects switch to see only demos, and then click **RUN** next to the demo to load. The solution examples switch is provided for future use.
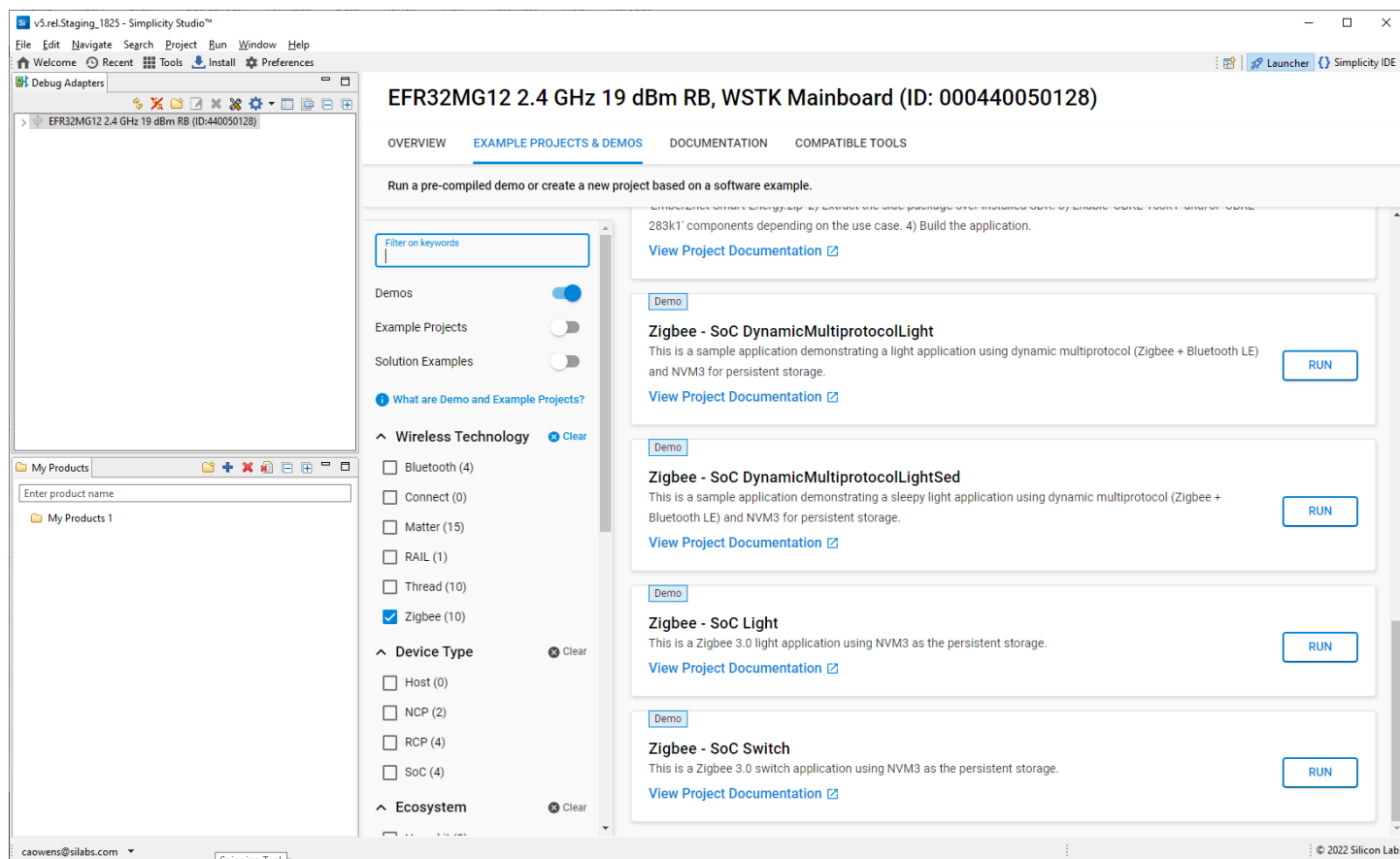
The Zigbee EmberZNet SDK demos are:

Simple Zigbee 3.0 Network

Zigbee - Z3Light: Zigbee 3.0 Light application.
Zigbee - Z3Switch: Zigbee 3.0 Switch application.

Compatible parts, brd4161a, brd4162a, brd4180a

For the Light and Switch demos, press Button0 on the Switch device to initiate the network. The light device should bind to the network automatically. Once the switch has finished finding and binding, you can use Button0 as an On/Off toggle.

This may not be successful, because some timeouts are included in the button functionality. If so, use the Command Line Interface to set up the demo network.

Once the Z3Light app starts running, it makes many attempts to join a network, at the end of which it sets up its own distributed network. Once the Info command indicates that it is in a network (that is, has a PAN ID and Node ID), use the CLI command:

```
plugin network-creator-security open-network
```

You should get the response:

```
NWK Creator Security: Open network: 0x00
pJoin for 254 sec: 0x00
NWK Creator Security: Open network: 0x00
```

Now the device is ready for joining. In the Z3Switch App, instead of using the button, first make sure it has not joined any networks by issuing:

```
network leave
```

Then you can enter the command:

```
plugin network-steering start 0
```

This will kick off the joining process.

Dynamic Multiprotocol

Zigbee BLE - DynamicMultiprotocolLight and **Zigbee BLE - DynamicMultiprotocolLightSed:** Companion applications demonstrating Zigbee/Bluetooth LE dynamic multiprotocol functionality.

Compatible parts: brd4161a, brd4162a

Multiprotocol Radio Co-Processor

Multiprotocol (OpenThread+Zigbee) - RCP (SPI) and **Multiprotocol (OpenThread+Zigbee) - RCP (UART)**: These applications support running OpenThread and Zigbee stacks simultaneously on a host processor. The host applications connect to the CPC daemon, which in turn connects to the EFR via a SPI or UART link, respectively.

Compatible parts: brd4151a, brd4161a, brd4166a, brd4168a, brd4180a, brd4304a

Multiprotocol (OpenThread+Zigbee+BLE) - RCP (SPI) and **Multiprotocol (OpenThread+Zigbee+BLE) - RCP (UART)**: These applications support running OpenThread, Zigbee, and Bluetooth stacks simultaneously on a host processor. The host applications connect to the CPC daemon, which in turn connects to the EFR via a SPI or UART link, respectively.

Compatible parts: brd4161a, brd4166a, brd4168a, brd4180a, brd4304a

NCP Applications

Zigbee - NCP UartHw: Network coprocessor (NCP) application that supports communication with a host application over a UART interface with hardware flow control.

Compatible parts: brd4158a, brd4161a, brd4162a, brd4163a, brd4164a, brd4169b, brd4170a, brd4180a

Zigbee - NCP ncp-spi: Network coprocessor (NCP) application that supports communication with a host application over an SPI interface.

Compatible parts: brd4158a, brd4161a, brd4162a, brd4163a, brd4164a, brd4169b, brd4170a, brd4180a

3.2 Software Examples

Note: Examples provided for the EFR32xG12 and newer parts include Silicon Labs Gecko Bootloader examples. Examples are provided for all compatible Simplicity Studio SDKs. When configuring security for a Gecko Bootloader example, you must use Simplicity Commander, not the Simplicity Studio IDE interface. For more information on using the Gecko Bootloader see *UG489: Silicon Labs Gecko Bootloader User's Guide for GSDK 4.0 and Higher.*

Some Zigbee EmberZNet software examples are specifically for SoC applications. If you are not familiar with the differences between System-on-Chip (SoC) and Network Coprocessor (NCP)/Radio Coprocessor (RCP) application models, see *UG103.03: Application Development Fundamentals: Design Choices.* For more information on Green Power Devices see *UG392: Using Silicon Labs Green Power with Zigbee EmberZNet.*

In the Debug Adapters view, select the device. The examples list is filtered by the selected device.

Click the EXAMPLE PROJECTS & DEMOS tab in the Launcher perspective. Under Technology Type, select Zigbee to see only Zigbee examples.
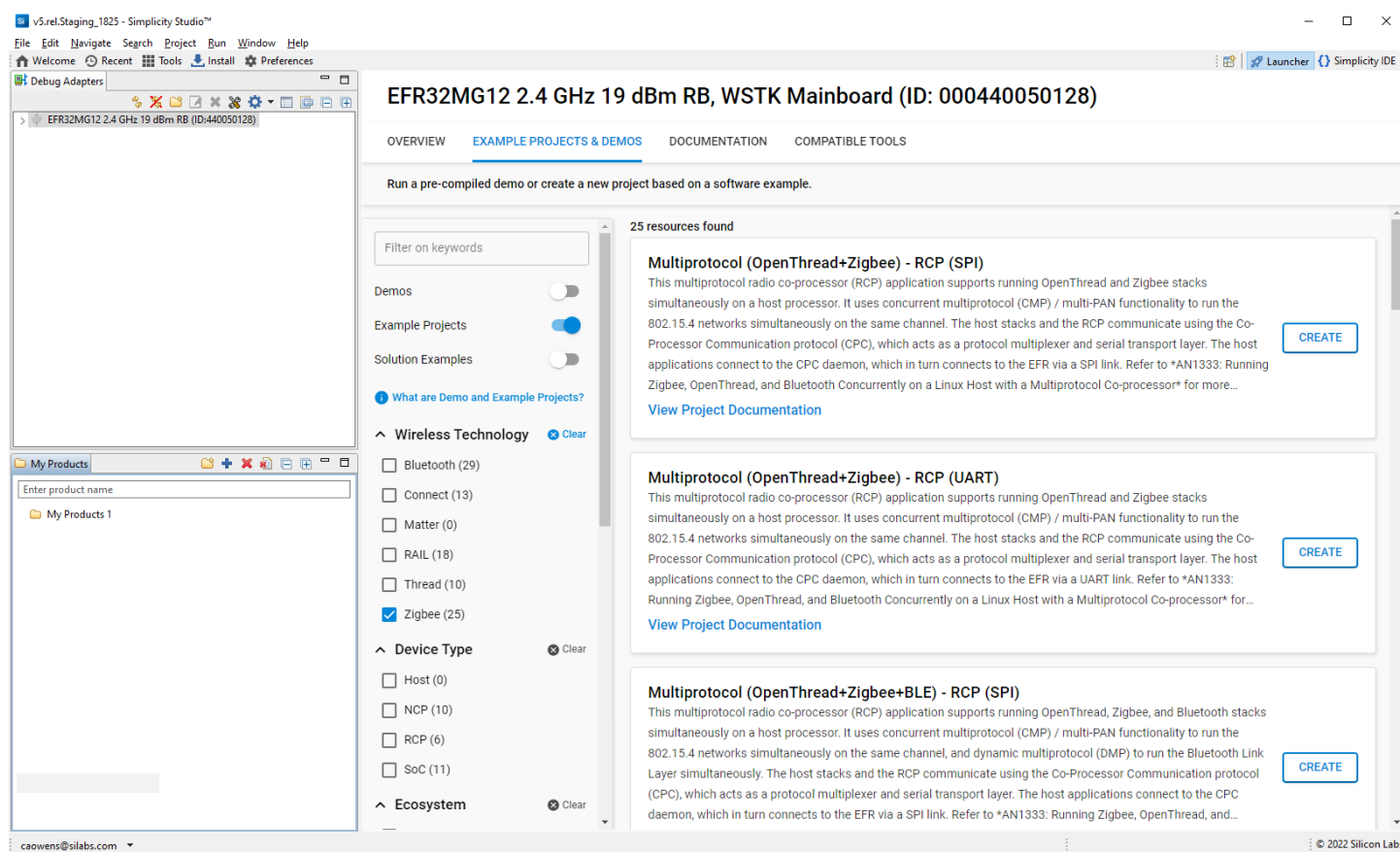
3.2.1 Zigbee 3.0 Network

The Z3Light and Z3Switch applications were developed for specific development kit hardware. Use on other devices may require some GPIO remapping using the PIN tool. See the online Simplicity Studio v5 User's Guide for more information about using the PIN tool.

Zigbee - Z3Light: Zigbee light application. Acting as a router it can form a distributed network. Acting as a touchlink target it can touchlink with the **Zigbee - Z3Switch**, which is acting as a touchlink initiator.

Zigbee - Z3Switch: Zigbee switch application, acting as an end device, can join the network.

Press Button0 on the Switch device to initiate the network. The light device should bind to the network automatically. Once the switch has finished finding and binding, you can use Button0 as an On/Off toggle.

This may not be successful, because some timeouts are included in the button functionality. If so, use the Command Line Interface to set up the demo network.

Once the zigbee_z3_light app starts running, it makes many attempts to join a network, at the end of which it sets up its own distributed network. Once the Info command indicates that it is in a network (that is, has a PAN ID and Node ID), use the CLI command:

```
plugin network-creator-security open-network
```

You should get the response:

```
NWK Creator Security: Open network: 0x00
pJoin for 254 sec: 0x00
NWK Creator Security: Open network: 0x00
```

Now the device is ready for joining. In the zigbee_z3_switch App, instead of using the button, first make sure it has not joined any networks by issuing:

```
network leave
```

Then you can enter the command:

```
plug network-steering start 0
```

This will kick off the joining process.

3.2.2   Dynamic Multiprotocol

Zigbee BLE - DynamicMultiprotocolLight and **Zigbee BLE - DynamicMultiprotocolLightMinimal:** Sample application with Zigbee 3.0 (Coordinator/Router) + Bluetooth LE. The Minimal application does not use buttons, LEDs, or the LCD.

Zigbee BLE - DynamicMultiprotocolLightSed and **Zigbee BLE - DynamicMultiprotocolLightSedMinimal:** Sample application with Zigbee 3.0 (Sleepy End Device) + Bluetooth LE. The Minimal application does not use buttons, LEDs, or the LCD.

See *AN1322: Dynamic Multiprotocol Development with Bluetooth and Zigbee in SDK 7.0 and Higher* for more information on using these examples.

3.2.3   Green Power

Zigbee - GPD Sensor: Demonstrates a Green Power occupancy sensor device with attributes pre-configured in the component Green Power Device Application Support.

Zigbee - GPD Switch: Demonstrates a Green Power switch device with attributes pre-configured in the component Green Power Device Application Support.

Zigbee GP - Z3LightCombo: Z3 Light GP Combo application demonstrates functionality of a Green Power combo device with proxy and sink instances.

3.2.4   NCP

Zigbee - NCP MpNcpSpi: The Multi-Pan NCP SPI application supports forming two personal area networks on same channel on single radio.

Zigbee - NCP MpNcpUartHw: The Multi-PAN NCP UART application supports forming two personal area networks on same channel on single radio.

Zigbee - NCP UartHw: Network coprocessor application that communicates with a host application over a UART interface with hardware flow control.

Zigbee - NCP ncp-spi: Network coprocessor application that communicates with a host application over a SPI interface.

Zigbee - NCP UartHwGpMultiRail: Network coprocessor application extends the ncp-uart-hw application with a multi-RAIL library, and a multi-RAIL demo component enabled.

Zigbee - NCP Xncp Led: Network processor application that communicates with a UNIX HOST using custom EZSP commands.

3.2.5 RCP

For more information about these examples, see *AN1333: Running Zigbee, OpenThread, and Bluetooth Concurrently on a Linux Host with a Multiprotocol Co-processor*

Multiprotocol (OpenThread+Zigbee) - RCP (SPI) and **Multiprotocol (OpenThread+Zigbee) - RCP (UART)**: These applications support running OpenThread and Zigbee stacks simultaneously on a host processor. The host applications connect to the CPC daemon, which in turn connects to the EFR via a SPI or UART link, respectively.

Multiprotocol (OpenThread+Zigbee+BLE) - RCP (SPI) and **Multiprotocol (OpenThread+Zigbee+BLE) - RCP (UART)**: These applications support running OpenThread, Zigbee, and Bluetooth stacks simultaneously on a host processor. The host applications connect to the CPC daemon, which in turn connects to the EFR via a SPI or UART link, respectively.

Zigbee - NCP + OpenThread - RCP zigbee_ncp-ot_rcp-spi and **Zigbee – NCP + OpenThread – RCP zigbee_ncp-ot_rcp-uart**: These concurrent multiprotocol (CMP) application run the Zigbee NCP simultaneously with the OpenThread RCP. The host applications connect to the CPC daemon, which in turn connects to the EFR via a SPI or UART link, respectively.

3.2.6 Minimal Configuration

Zigbee - Minimal: This is a Zigbee minimal network-layer application suitable as a starting point for new application development.

3.2.7 Testing

Zigbee - StandardizedRfTesting: This is a pre-standardization implementation of Zigbee's RF testing standard. It utilizes the TIS (Total Isotropic Sensitivity)/ TRP (Total Radiated Power) testing interfaces and is optional for Zigbee certifications.

4   Working with Examples

In these instructions you will work with two example applications, Zigbee - Z3Light Application and Zigbee - Z3Switch Application. These steps illustrate both how to compile and load example applications and use the Simplicity Studio console interface to work with them, but also how to configure them using three configuration tools:

- Simplicity Studio's Project Configurator allows you to add and remove functions from the applications in the form of components
- The Component Editor allows you to change the parameters of components installed in the application
- The Zigbee Cluster Configurator allows you to modify Zigbee clusters, attributes, and commands.

The following example procedures are provided. Each procedure builds on the next, so it is recommended to work through them in order.

- Create and Load a Zigbee - Z3Light Application: Provides a detailed walk-through of project creation.
- Create and Load a Zigbee - Z3Switch Application: Provides some shortcuts to project creation.
- Create a Distributed Network and Issue Commands: Shows how to use the Console interface, create a network, and turn a light on and off.
- Change to a Centralized Network: Describes how to use the Project Configurator and Component Editor to change an application.
- About the Zigbee Cluster Configurator: Introduces additional configuration options provided through the Zigbee Cluster Configurator.
- About Bootloaders provides instructions on uploading a bootloader image. This should not be necessary unless you erase the device.

Section 5 Using the Network Analyzer describes how to use Network Analyzer to observe traffic across the network.

These procedures are illustrated for a WSTK with an EFR32MG. **Note:** Your SDK version may be later than the version shown in the procedure illustrations.

4.1 Create and Load a Zigbee - Z3Light Application

SSv5 offers a variety of ways to begin a project using an example application. The online Simplicity Studio 5 User's Guide, available both through https://docs.silabs.com/ and the SSv5 help menu, describes them all. This procedure uses the **File > New > Silicon Labs Project Wizard** method, because it takes you through all three of the Project Creation Dialogs.

1. Open SSv5's File menu and select New > Silicon Labs Project Wizard. The Target, SDK, and Toolchain Selection dialog opens. If you want to change the toolchain from the default GCC to IAR, do so here. Click **NEXT**.

   Note: If you have both IAR and GCC installed, GCC is the default.
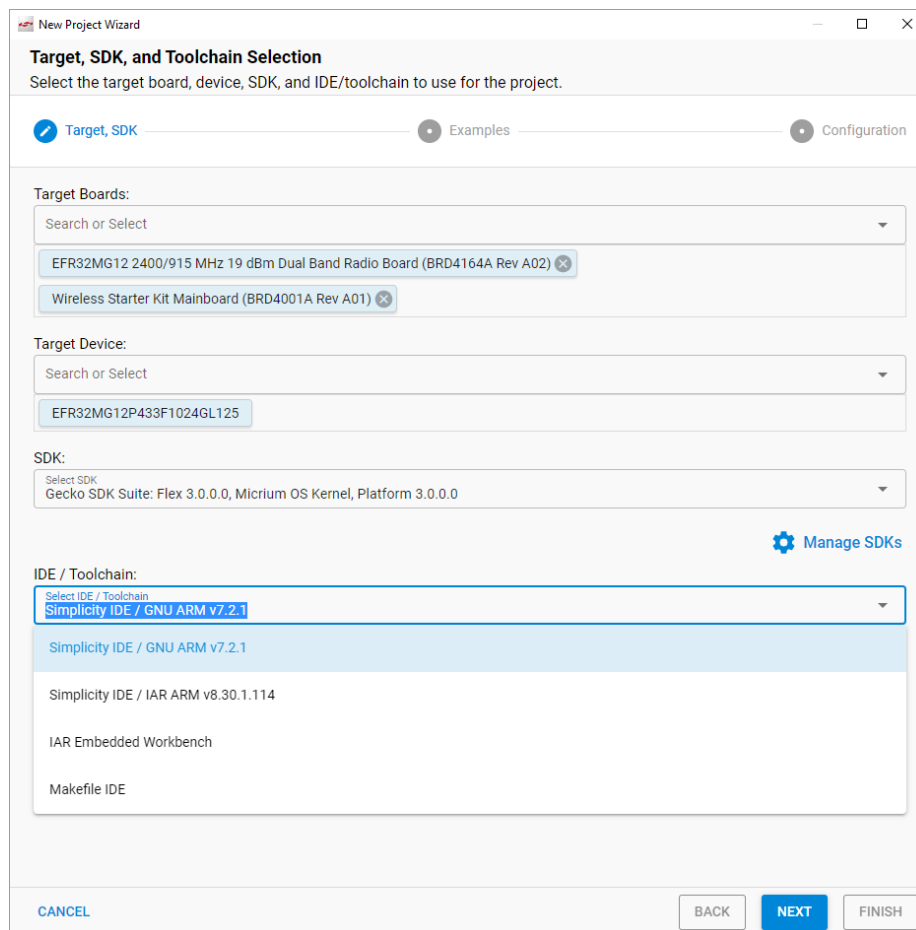
2. The Example Project Selection dialog opens. Use the Technology Type and Keyword filters to search for a specific example, in this case **Zigbee - Z3Light.** Select it and click **NEXT**.
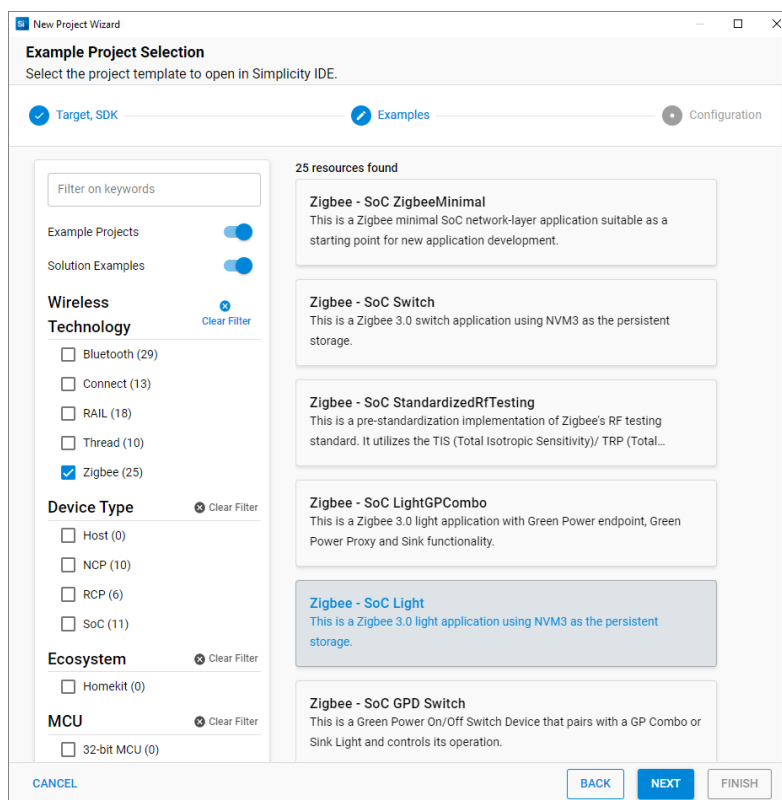
3. The Project Configuration dialog opens. Here you can rename your project, change the default project file location, and determine if you will link to or copy project files. Note that if you change any linked resource, it is changed for any other project that references it. Click **FINISH.**
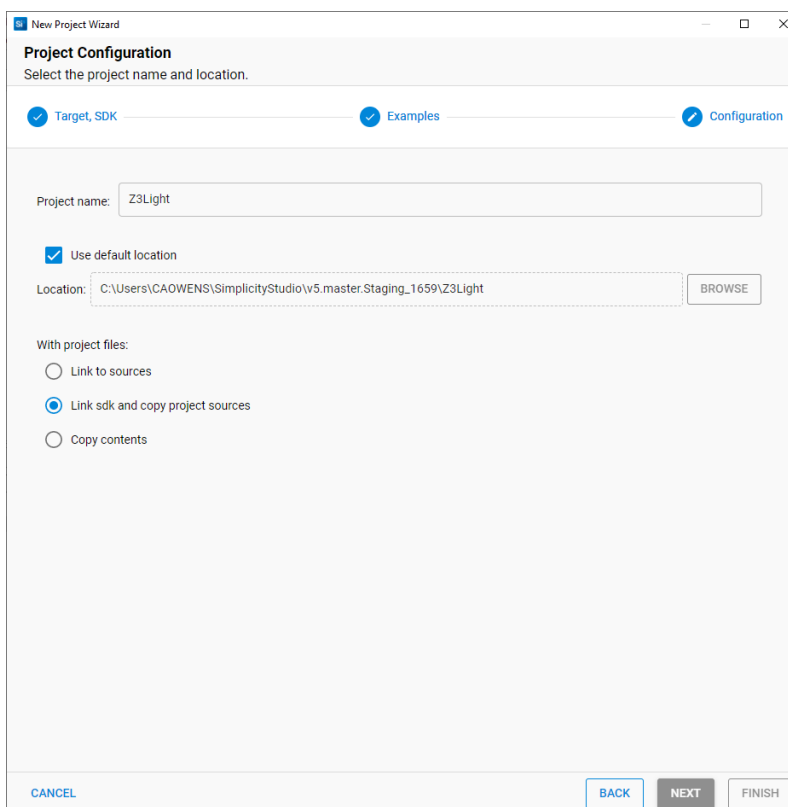
The Simplicity IDE perspective opens with the new project in Project Configurator view. The readme tab contains information about the project. See the online Simplicity Studio 5 User's Guide for details about the functionality available through the Simplicity IDE perspective and Project Configurator.

Note: You now have a Simplicity IDE button next to the Launcher button in the upper right.
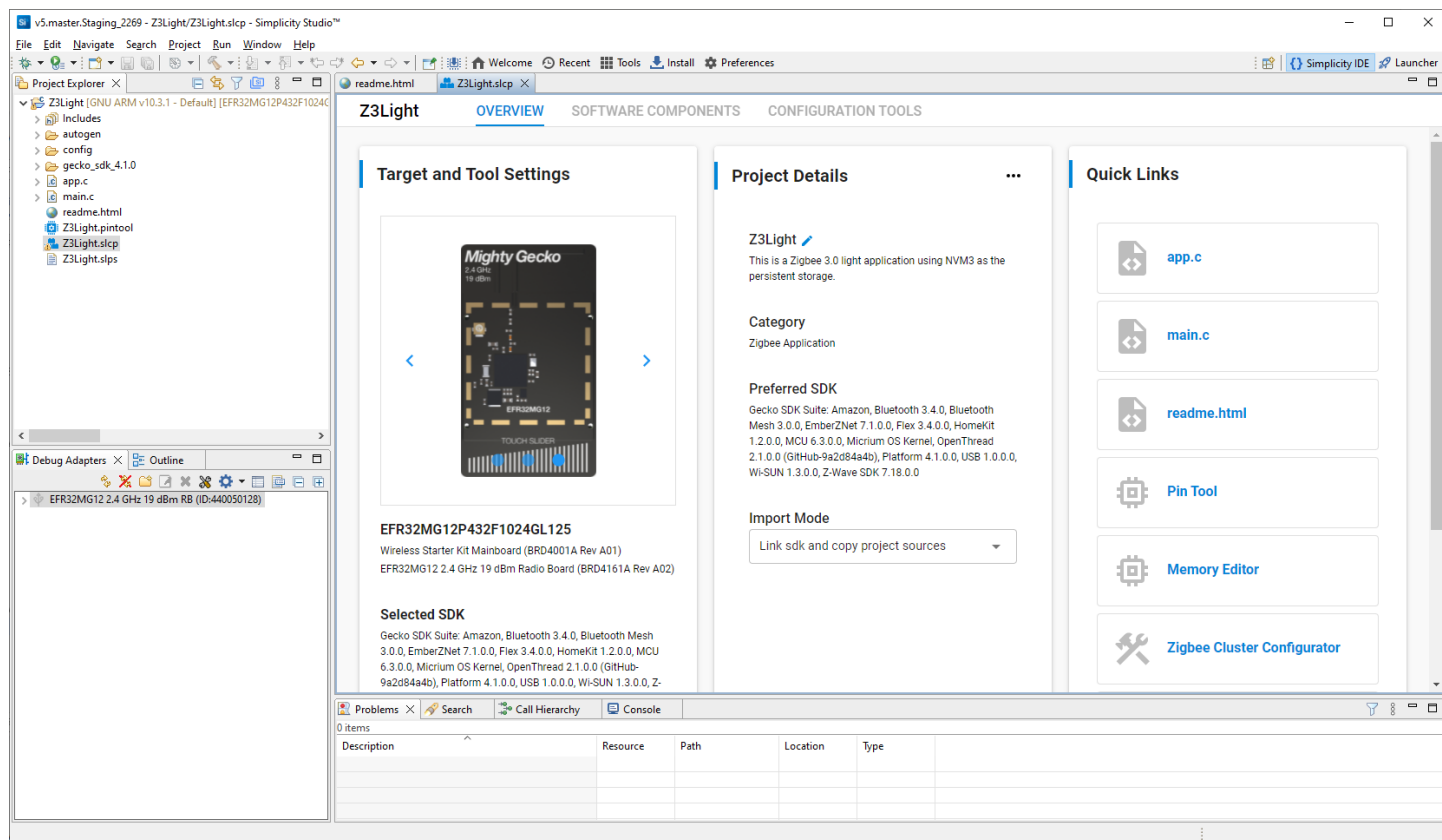
Simplicity Studio automatically generates the project files needed to build the application image. If you are accustomed to using Simplicity Studio 6.x, note that this means there is no **Generate** button on the Project Configurator interface. The Force Generation button on the Project Details card is not a replacement, but rather to be used under conditions in which auto-generation fails to work.

Files are created in the autogen folder, and are automatically updated when you make changes to the project configuration.
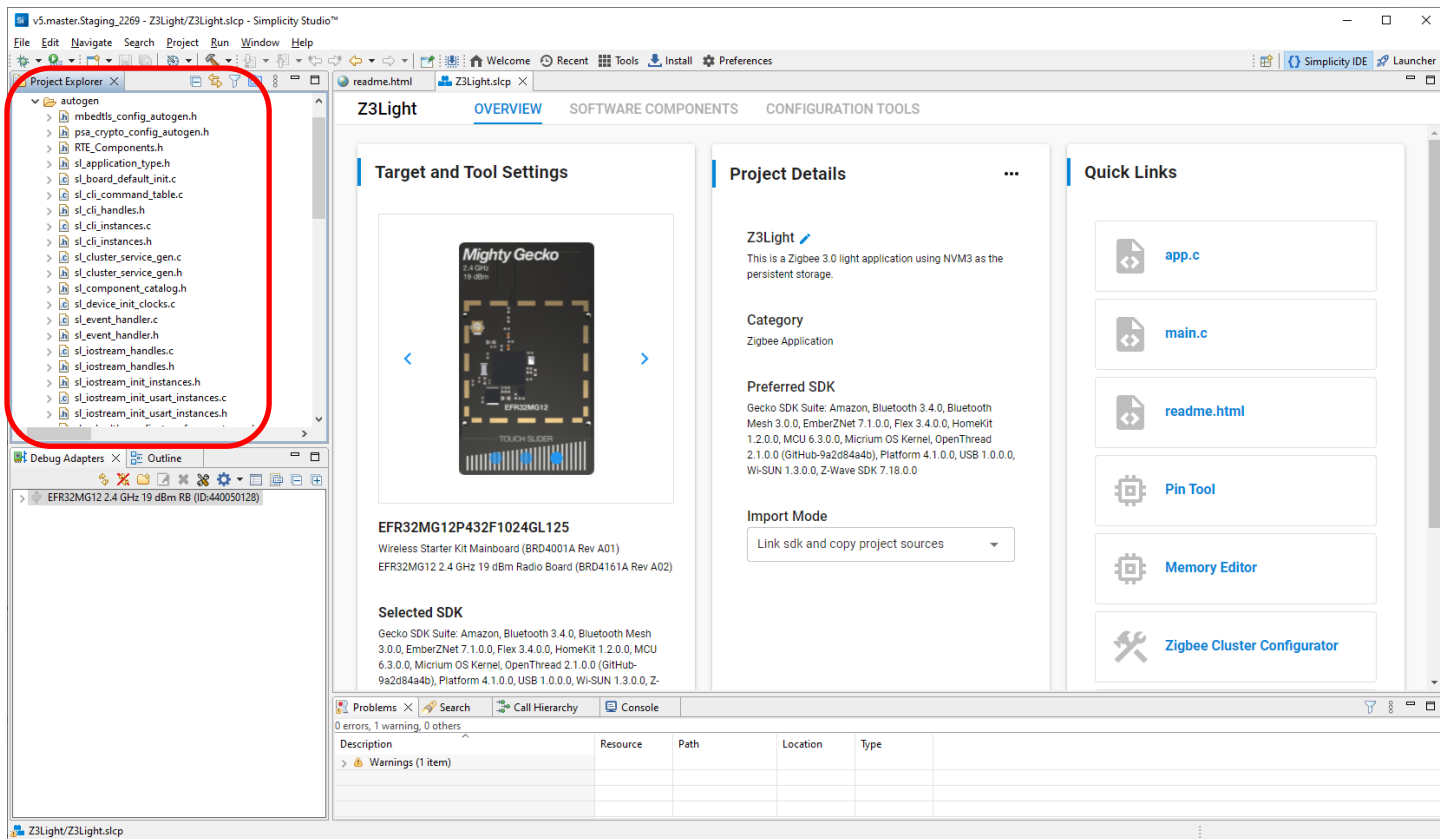
4. Example applications are pre-configured to support the example functionality. To see the configuration, click the SOFTWARE COMPONENTS tab and search for the component of interest, for example 'device configuration'. Installed components are indicated by a check on the left. Configurable components are indicated by a gear icon on the right. Select the component to see more information about it.

Note: All EFR32 parts have a unique RSSI offset. In addition, board, antenna and enclosure design can also impact RSSI. When creating a new project, install the **RAIL Utility, RSSI** component. This feature includes the default RSSI Offset Silicon Labs has measured for each part. This offset can be modified, if necessary, after RF testing of your complete product.
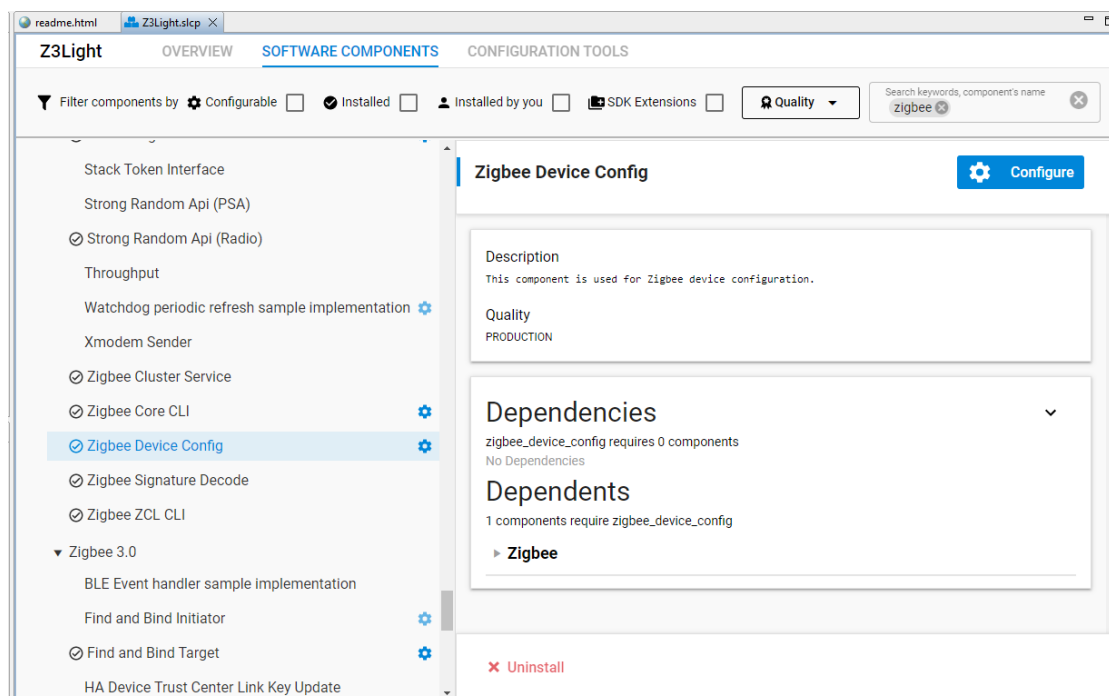
5. Click **Configure** to open the Component Editor on a new tab. The configurable parameters are shown. In this case, it indicates that the Z3Light application is configured as a router. Close the Component Editor.
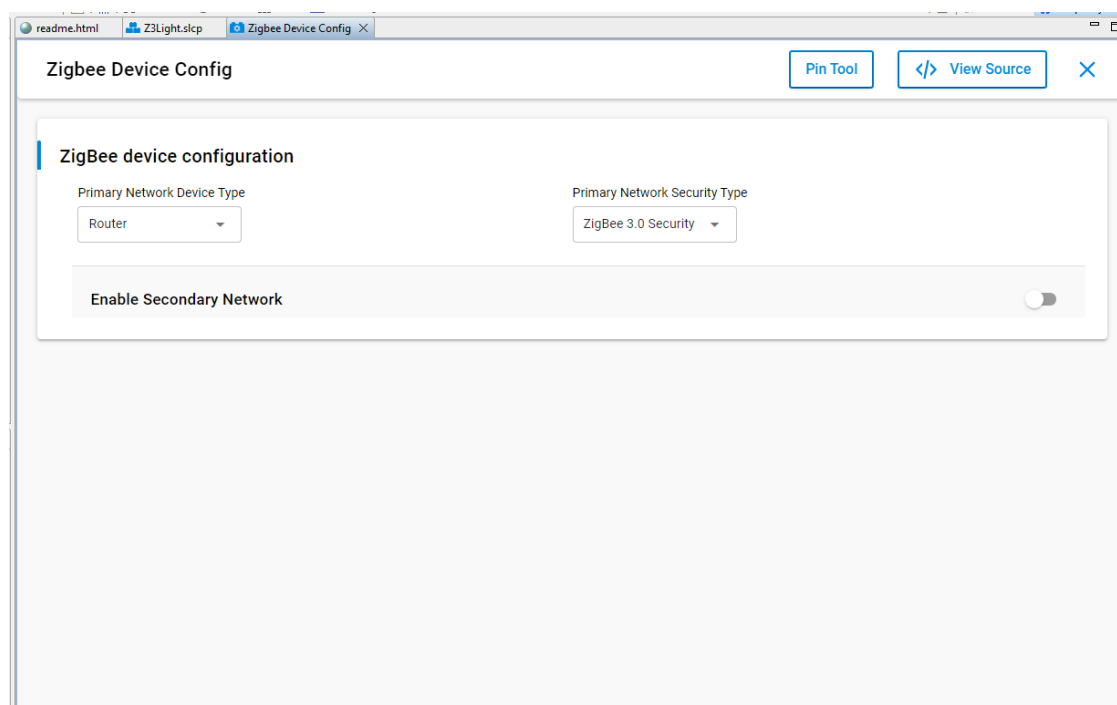
6. Once you are ready to build the application, click the **Build** (hammer) control in the top tool bar. During the build, progress is reported both in a window, which can be run in the background, and also in the lower right. The process may take over a minute.

    Note: If the Build control isn't active, click the project directory or a file in it.
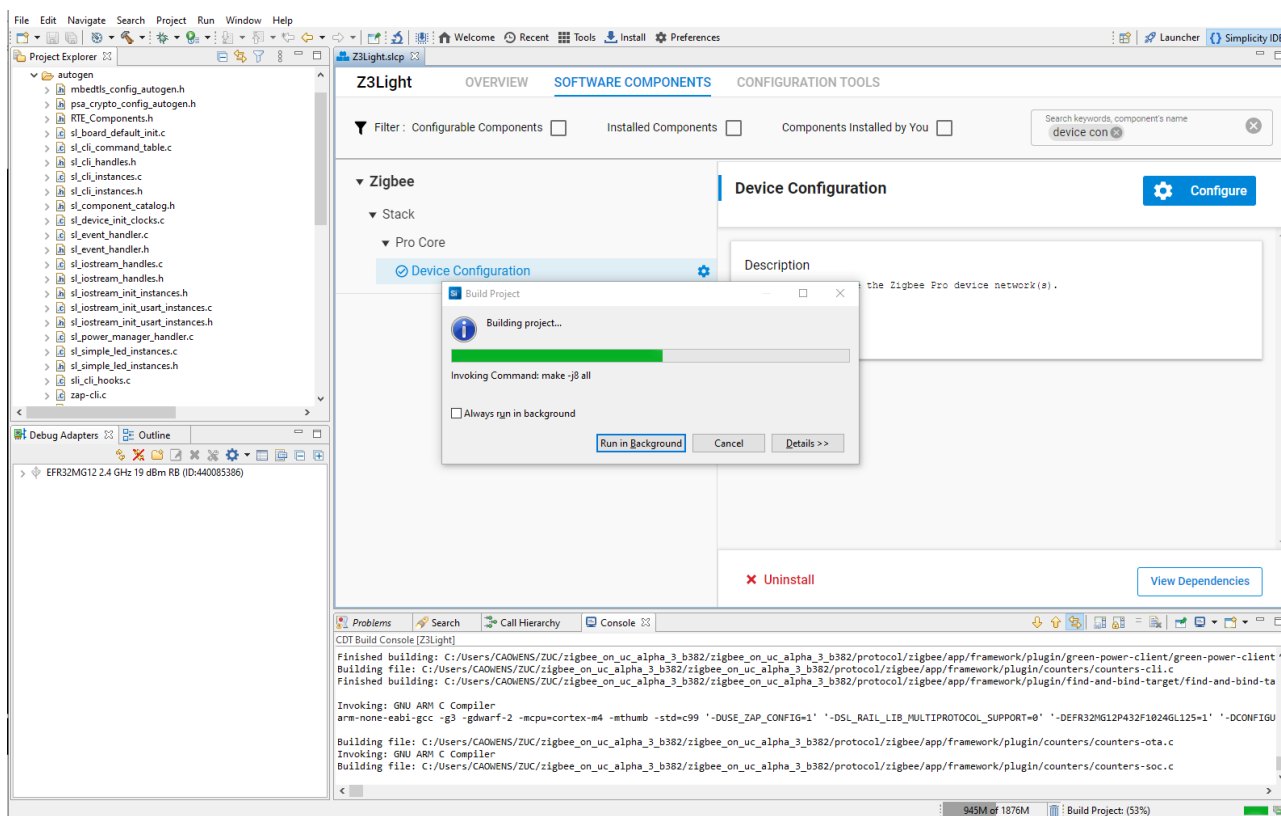
Build completion is reported in the Build Console. The build should complete without errors. If any errors occur, they are highlighted in red in the console. Contact technical support for assistance.
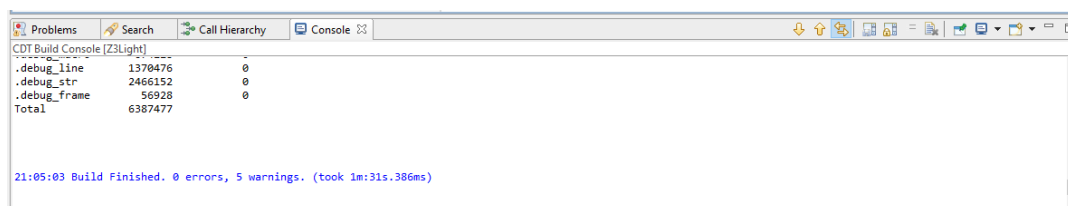
7. In the Project Explorer view, the binaries are shown in a new folder, Binaries.
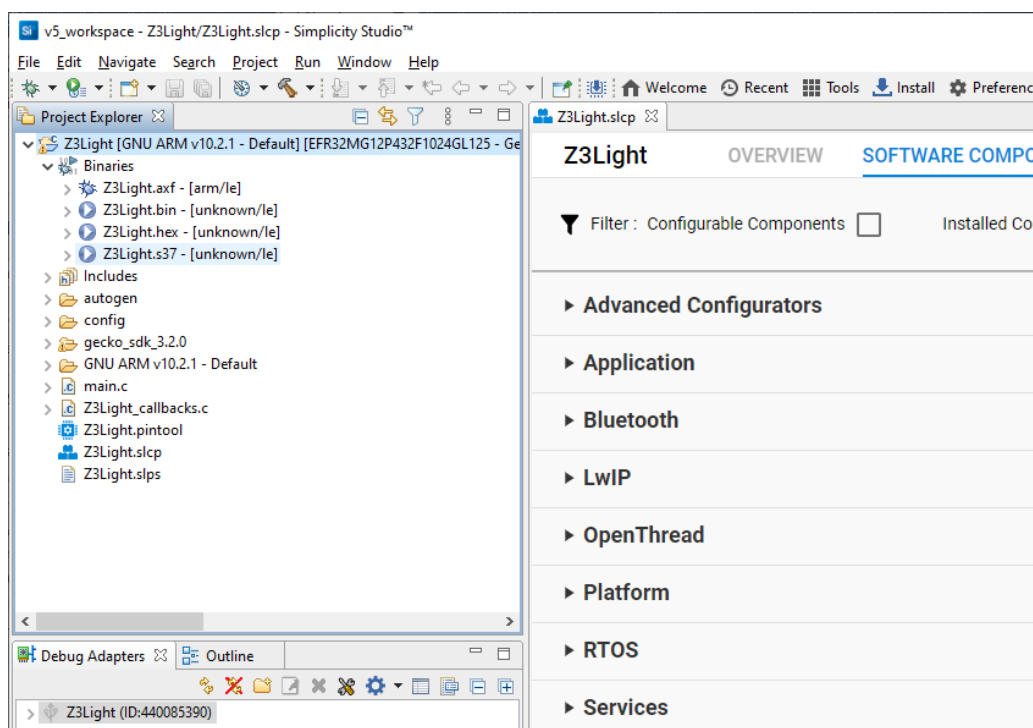
8. Right-click the <project>.s37 file, and select **Flash to Device**. If you have more than one device connected, you are prompted to select the target. Then the following dialog is displayed.
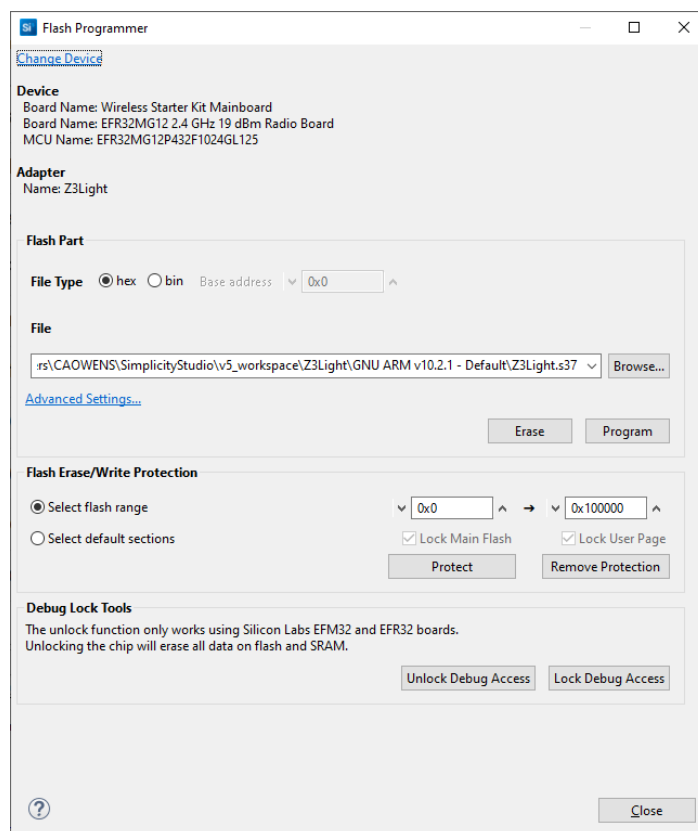
9. Click **Program**. The binary is flashed to the target device.

10. In the Debug Adapters view, right-click the Z3Light device and select **Launch Console**. In the console window, you will see four tabs: Serial 0 (the Virtual UART interface), Serial 1 (the physical UART interface), Admin (where you can configure the debug adapter such as the WSTK for EFR32), and Debug (where you see raw binary data over the debug interface). Click the Serial 1 tab. Type 'info' and press enter.
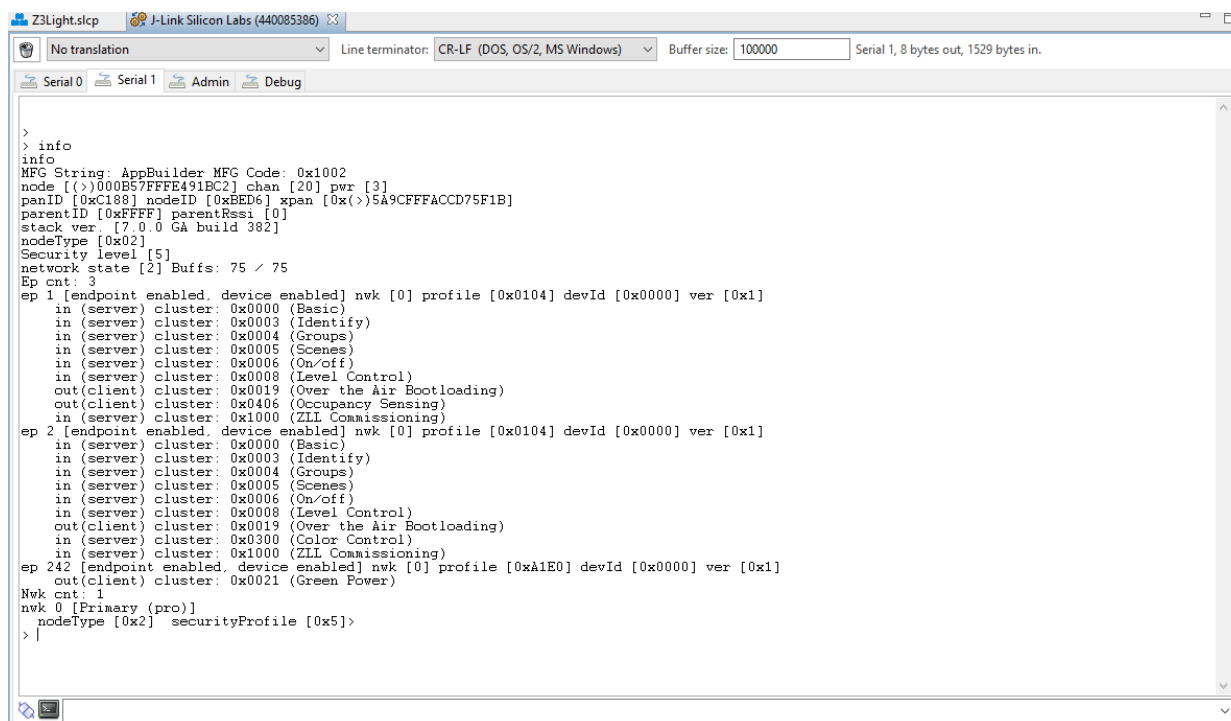
The application has created a distributed network with three endpoints, the last being the Green Power endpoint.

Note: During development you may make changes, work with the console, and then make other changes. While you can load a fresh application image with the device connected, you may have to repair the connection. Alternatively, disconnect from the console by right-clicking the device and selecting **Disconnect**.

If you have multiple devices connected, the device context menu also contains a Rename selection, which you can use to better identify the device roles.

4.2 Create and Load a Zigbee - Z3Switch Application

The **Zigbee - Z3Switch** example procedure follows a slightly different path than that illustrated for zigbee_z3_light. Use this shorter path if you do not need to modify devices or compilers.

1. Click **Launcher** in the upper right to return to the Launcher perspective.
2. In the Debug Adapters view, select the Switch device.
3. Go to the **EXAMPLE PROJECTS & DEMOS** tab and click **CREATE** next to the Zigbee - Z3Switch application.
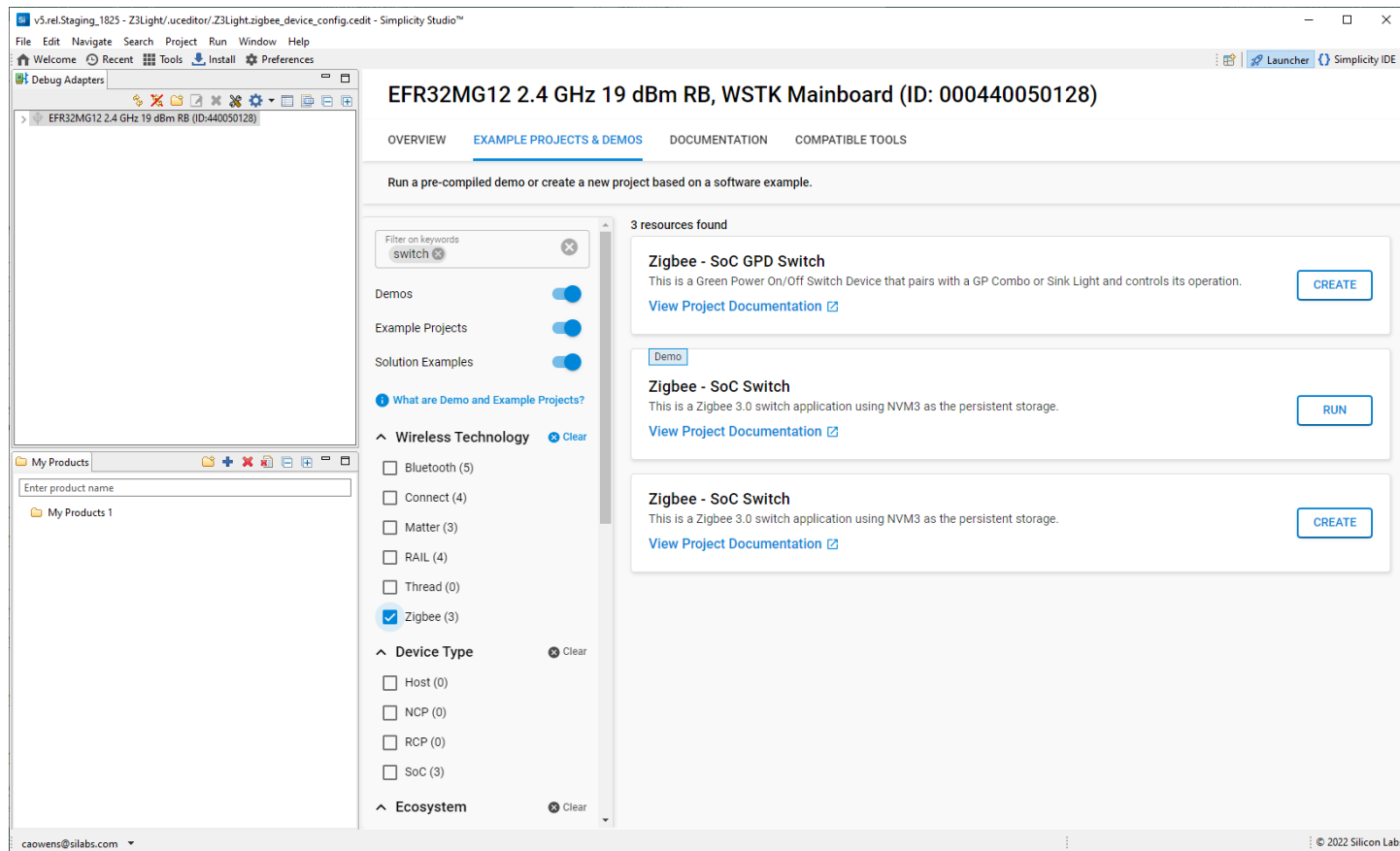
4. This opens the third of the three project creation dialogs illustrated in the zigbee_z3_light process. Change any values you wish to change, and click **FINISH**.
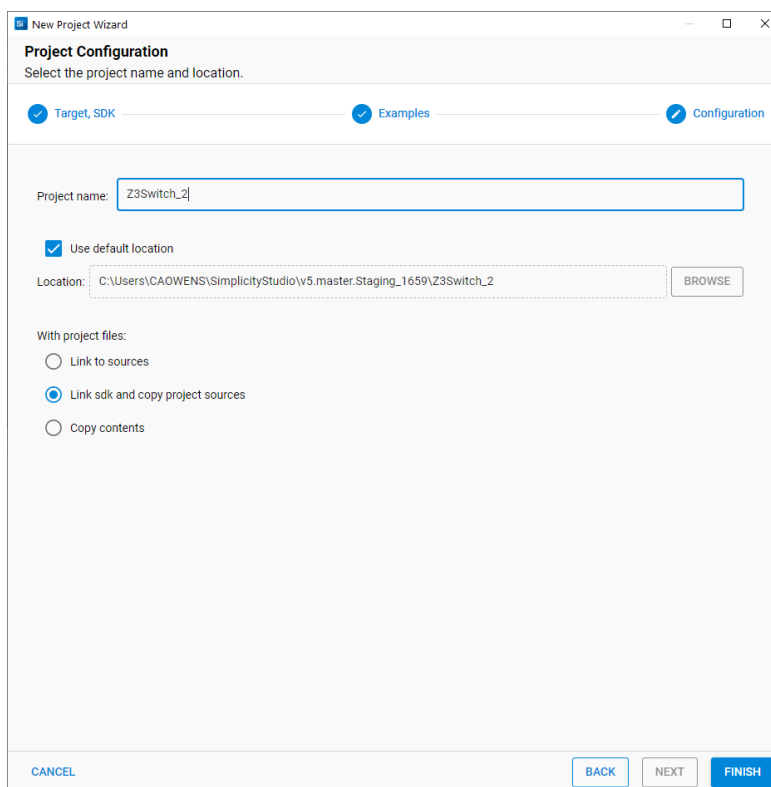
The project opens. All files are autogenerated. The Z3Switch application is configured as an end device.

5. Click the **Build** (hammer) control to build the application.
6. When the build is complete, right-click the <project>.s37 file and select **Flash to Device**. Click **Program**.

7. When upload is complete, right-click the Switch device and select **Launch Console**. Select the Serial 1 tab, and enter 'info'. The output should resemble the following. The switch has not yet joined a network.
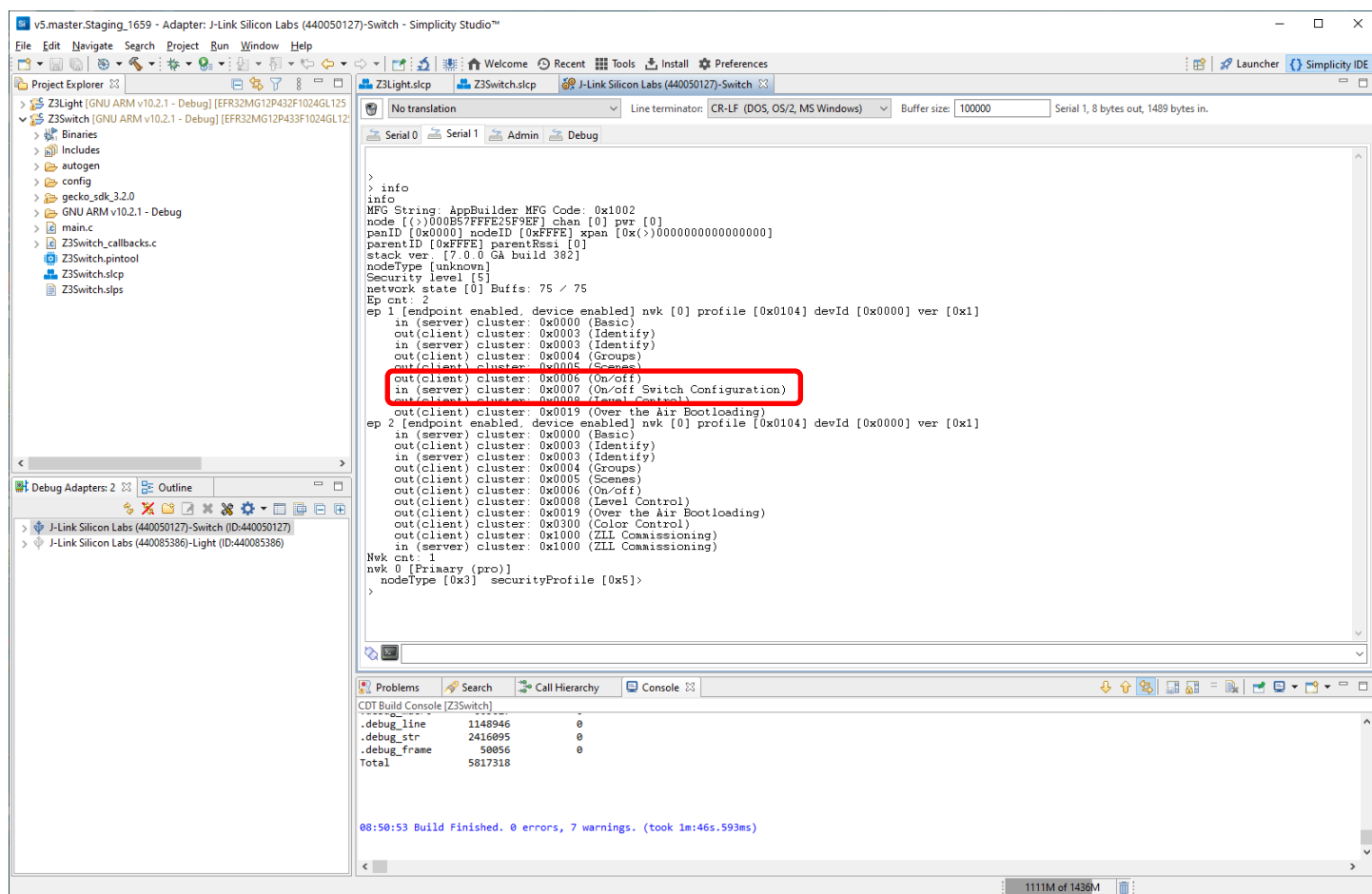
4.3 Create a Distributed Network and Issue Commands

Once you have downloaded both the light and switch applications to different radio boards, you can create a network. Make sure that the switch device is physically close to the light device.

To see the process from beginning to end, close the distributed network created automatically on the zigbee_z3_light device. In the zigbee_z3_light console, enter:

```
> network leave
```

Then form the network:

```
> plugin network-creator start 0
```

Open the network for joining:

```
> plugin network-creator-security open-network
```

You now have 55 seconds to join the switch to the open network. On the zigbee_z3_switch console, enter:

```
> plugin network-steering start 0
```

Enter 'info' on both devices and you will see that both the light and switch are on the same PAN ID.
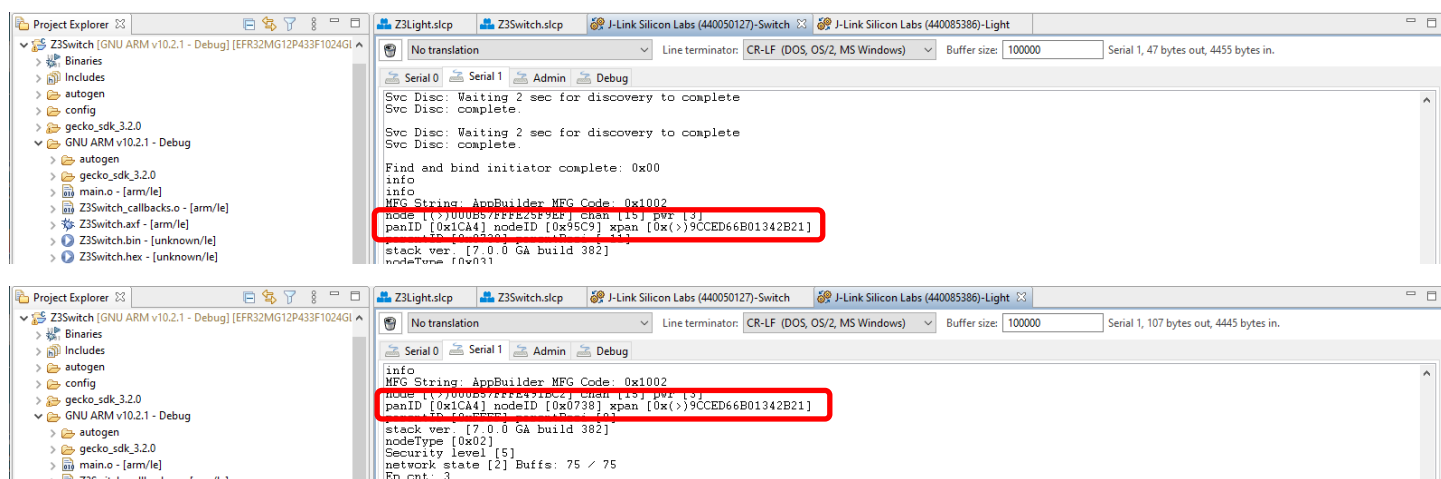
To send a command from the switch to the light, first build the toggle command and load it into a TX buffer:

```
> zcl on-off toggle
```

Then send the command to the light, using the device's Node ID (also known as the short ID), which is 0x0738. To toggle the device's LED on and off, on the zigbee_z3_switch device enter:

```
> send 0x0738 1 1
```

The light console should show:

```
T00000000:RX len 3, ep 01, clus 0x0006 (On/off) FC 01 seq 01 cmd 02 payload[]
On/Off set value: 01 02
Toggle on/off from 00 to 01
```

On the WSTK board, LED1 should turn on.

4.4   Change to a Centralized Network

In a distributed network, any router can issue network security keys. In a centralized network, a coordinator has the functionality of a Trust Center. Each node that joins is authenticated by the Trust Center before it can join the network. The coordinator is always node ID 0x0000. To change Z3Light to a Coordinator:

1. Go to the zigbee_z3_light.slcp tab.
2. On the SOFTWARE COMPONENTS tab, search for 'device config'.
3. Select the Zigbee Device Config component and click **Configure**.

4. Change the Primary Network Device Type to 'Coordinator or Router'. The change is saved automatically.
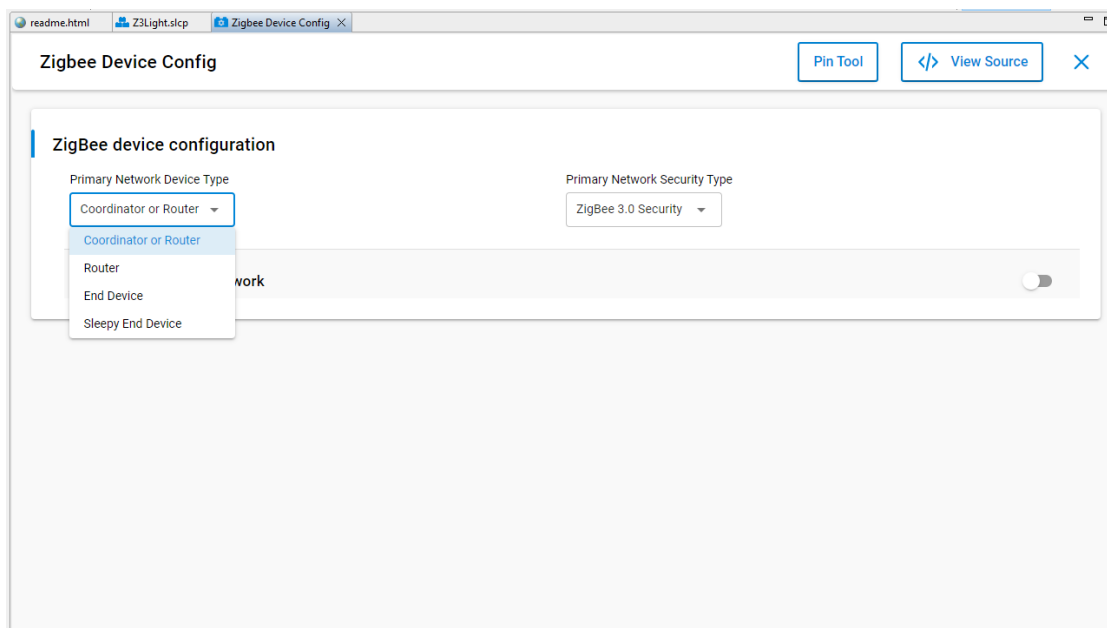
5. All coordinators must also have Concentrator functionality. Close the Component Editor and, back on the Software Components tab, search for 'concentrator'.

6. Select the Concentrator Support component and click **Install**. Changes are saved automatically.
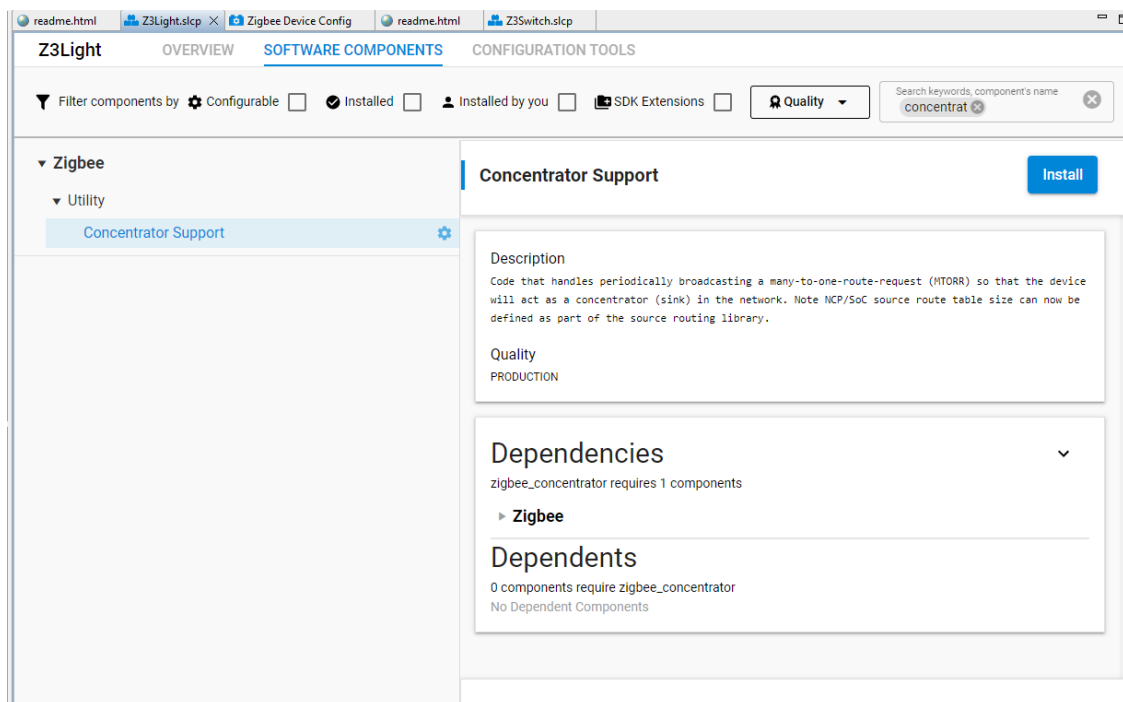
7. In the Project Explorer view, select zigbee_z3_light and click the **Build** (hammer) icon.

8. When build is complete, right-click the zigbee_z3_light device and click **Disconnect**.

9. When the build is complete, right-click the <project>.s37 file and select **Flash to Device**. Click **Program**.

10. Launch the console for the zigbee_z3_light device.

Next, take down the distributed network and start the centralized network.

```
> network leave
```

Then form the network:

```
> plugin network-creator form 1 0xfeed 20 11
```

Enter 'info'. The PAN ID is that specified, and the Node ID is 0x000
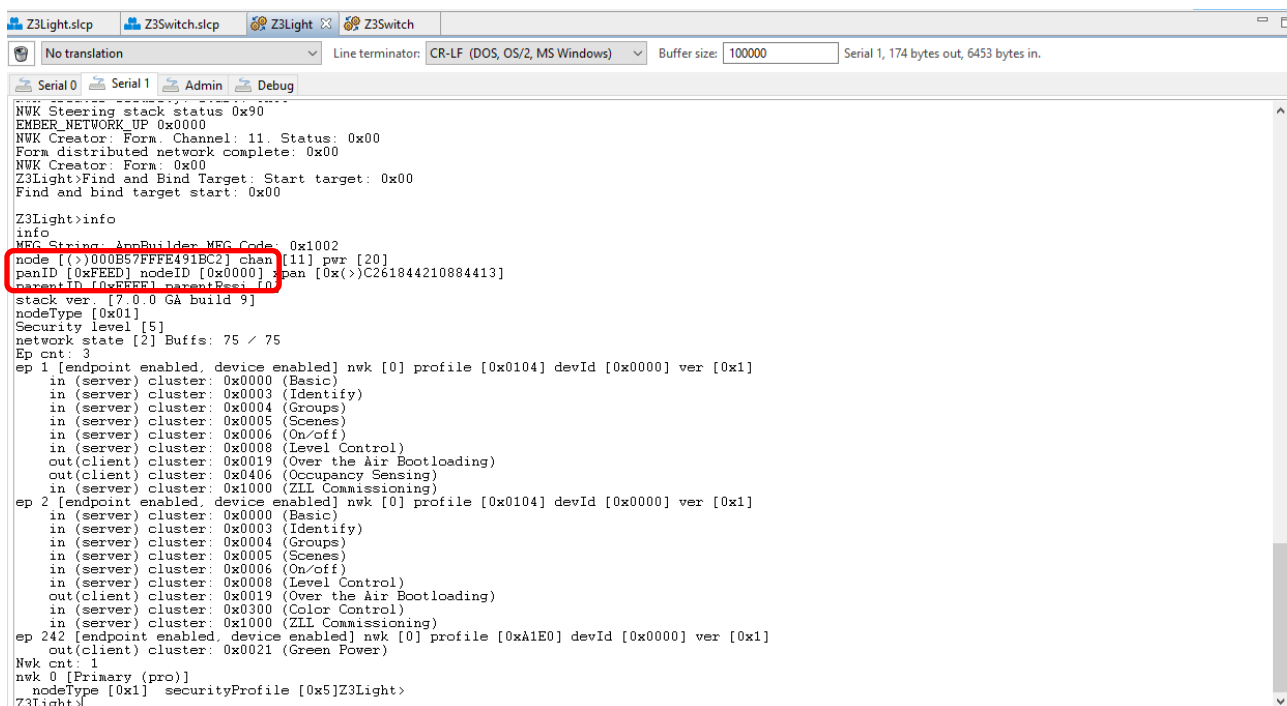
Open the network for joining:

```
> plugin network-creator-security open-network
```

You now have 55 seconds to join the switch to the open network. On the switch device:

```
> network leave
> plugin network-steering start 0
```

Enter 'info'. The PAN ID is that specified, and the parent ID is 0x0000.
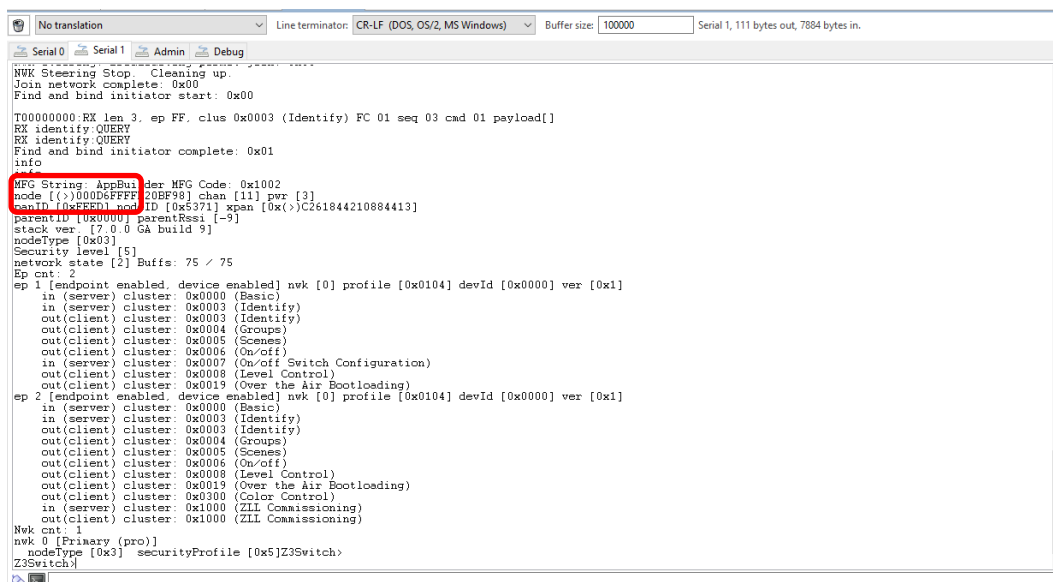
To send a command from the switch to the light, first build the toggle command and load it into a TX buffer:

```
> zcl on-off toggle
```

Then send the command to the light, using the device's Node ID 0x0000. To toggle the device's LED on and off, on the zigbee_z3_switch device enter:

```
> send 0 1 1
```

The light console should reflect the command and, on the WSTK board, LED1 should toggle from its previous state.

4.5   About the Zigbee Cluster Configurator

Additional customizations can be made through the Zigbee Cluster Configurator. Using this tool is described in detail in *AN1325: Zigbee Cluster Configurator User's Guide*.

4.6   About Bootloaders

All Silicon Labs examples require that a bootloader be installed. A bootloader is a program stored in reserved flash memory that can initialize a device, update firmware images, and possibly perform some integrity checks. Silicon Labs networking devices use bootloaders that perform firmware updates in two different modes: standalone (also called standalone bootloaders) and application (also called application bootloaders). An application bootloader performs a firmware image update by reprogramming the flash with an update image stored in internal or external memory. For more information about bootloaders see *UG103.6: Application Development Fundamentals: Bootloading*.

In March of 2017, Silicon Labs introduced the Gecko Bootloader, a code library configurable through Simplicity Studio's IDE to generate bootloaders that can be used with a variety of Silicon Labs protocol stacks. The Gecko Bootloader is used with all EFR32xG parts.

The Gecko Bootloader works with a specialized firmware update image format, ending in the extension .gbl (the GBL file). To create a GBL file from an .s37 or binary, follow the instructions in [UG162: Simplicity Commander Reference Guide](#), section 6.7.1, GBL File Creation. The exact format of the GBL file depends on the hardware you selected.

Note: When working with the Gecko Bootloader, you must use Simplicity Commander to enable some configuration option such as security features. See *UG489: Silicon Labs Gecko Bootloader User's Guide for GSDK 3.3 and Higher*.

By default, a new device is factory-programmed with a bootloader. If you have a new device, haven't cleared the bootloader region for your part or have a supported bootloader image already flashed on your device, you do not need to flash a bootloader. Once you have installed a bootloader image, it remains installed until you erase the device.

If you need to load a bootloader, select an example, such as **SPI Flash Storage Bootloader (single image)**, and build it and flash it as described above. For more information about the Gecko Bootloader, see *UG489: Silicon Labs Gecko Bootloader User's Guide for GSDK 4.0 and Higher*.

# 5   Using the Network Analyzer

Now that your network is set up, you can evaluate the data being transmitted using the Network Analyzer tool. Network Analyzer helps you debug network connectivity by displaying radio packets and certain debug interface events in a format that is easy to visualize and analyze. See the online Simplicity Studio User's Guide for more information.

1.   Click the Launcher button in the upper right and select Network Analyzer from the Tools menu. The Network Analyzer opens with your console window(s) still displaying data.
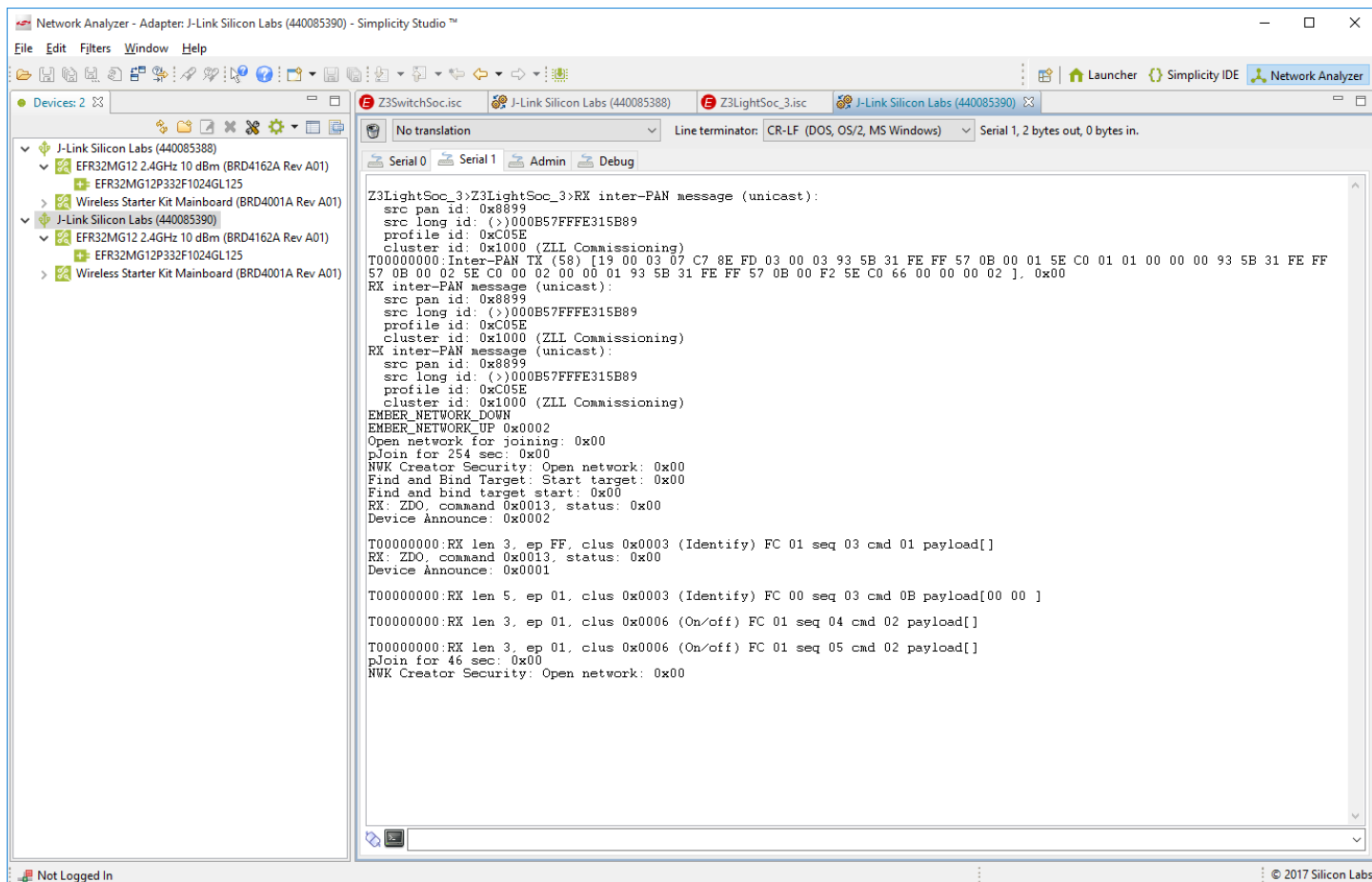
2. Make sure that Network Analyzer is set to decode the correct protocol. Select **Window > Preferences > Network Analyzer > Decoding > Stack Versions**, and verify it is set correctly. If you need to change it, click the correct stack, click **Apply**, and then **OK**.

Note: If you are working with a Zigbee+Bluetooth Dynamic Multiprotocol application, **Auto-detecting decoder stack** must be selected.
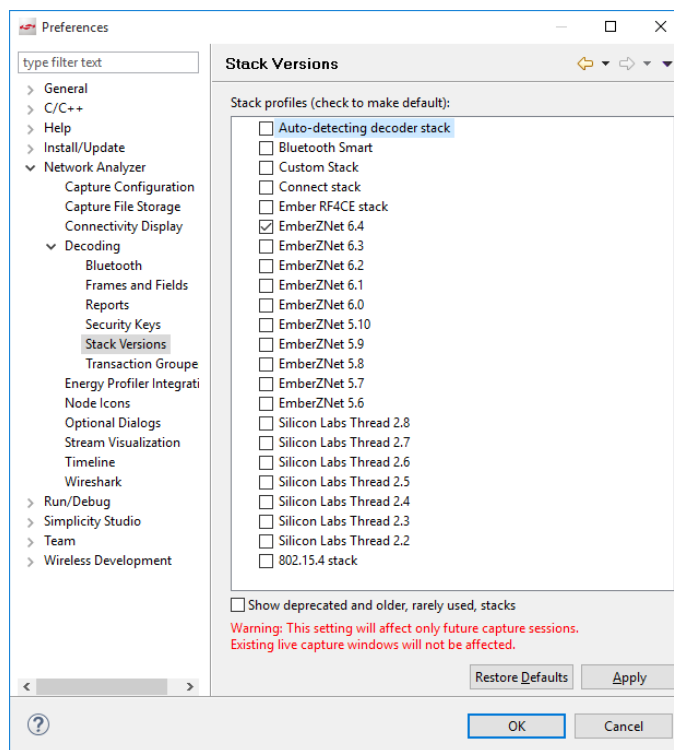
3. To make sure that packets decode correctly, manually enter the NWK key. In either the Switch or Light console window, type the following, being sure to include the 's' in 'keys':

```
Keys print
```

4. In the information returned, find the network key and copy it:

```
NWK Key: EF DE 0C 69 5B 72 6E C4 41 27 C6 E6 F1 36 26 26
```

5. In Window > Preferences, open Network Analyzer > Decoding > Security Keys, click **New**, name the new entry, and paste the copied key into it. Click **Apply**. Click **OK** to leave.
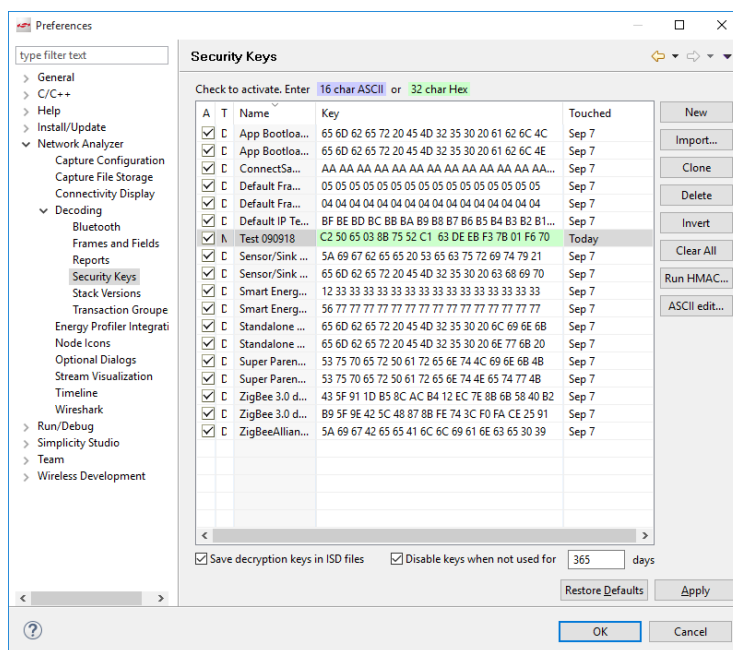
6. Right-click on the light or the switch device, and select **Start Capture**. Do the same for the other device.

7. If you are in an environment with a number of wireless devices, you may have a very noisy Network Analyzer environment, as reflected both in the event traffic and in the map. To show additional information in the map, click on the map.

   On the toolbar, click the PAN ID button, circled in the following image.
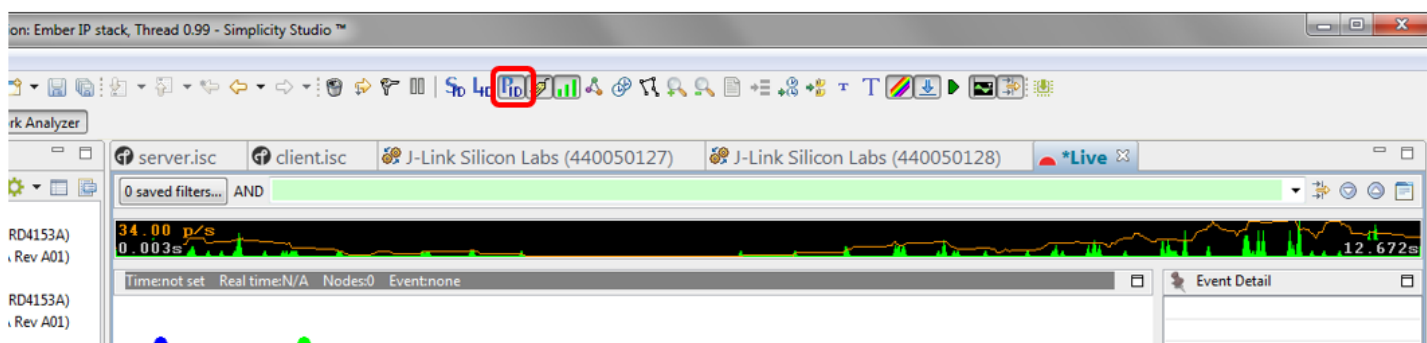
8. Right-click on the representation of your Switch device (the dot that has the same ISA3 adapter name or WSTK name or J-Link serial number as the device) and select **Show only this PAN**.

To filter transactions:

1. Press button 0 on the switch to get a complete transaction (in blue).
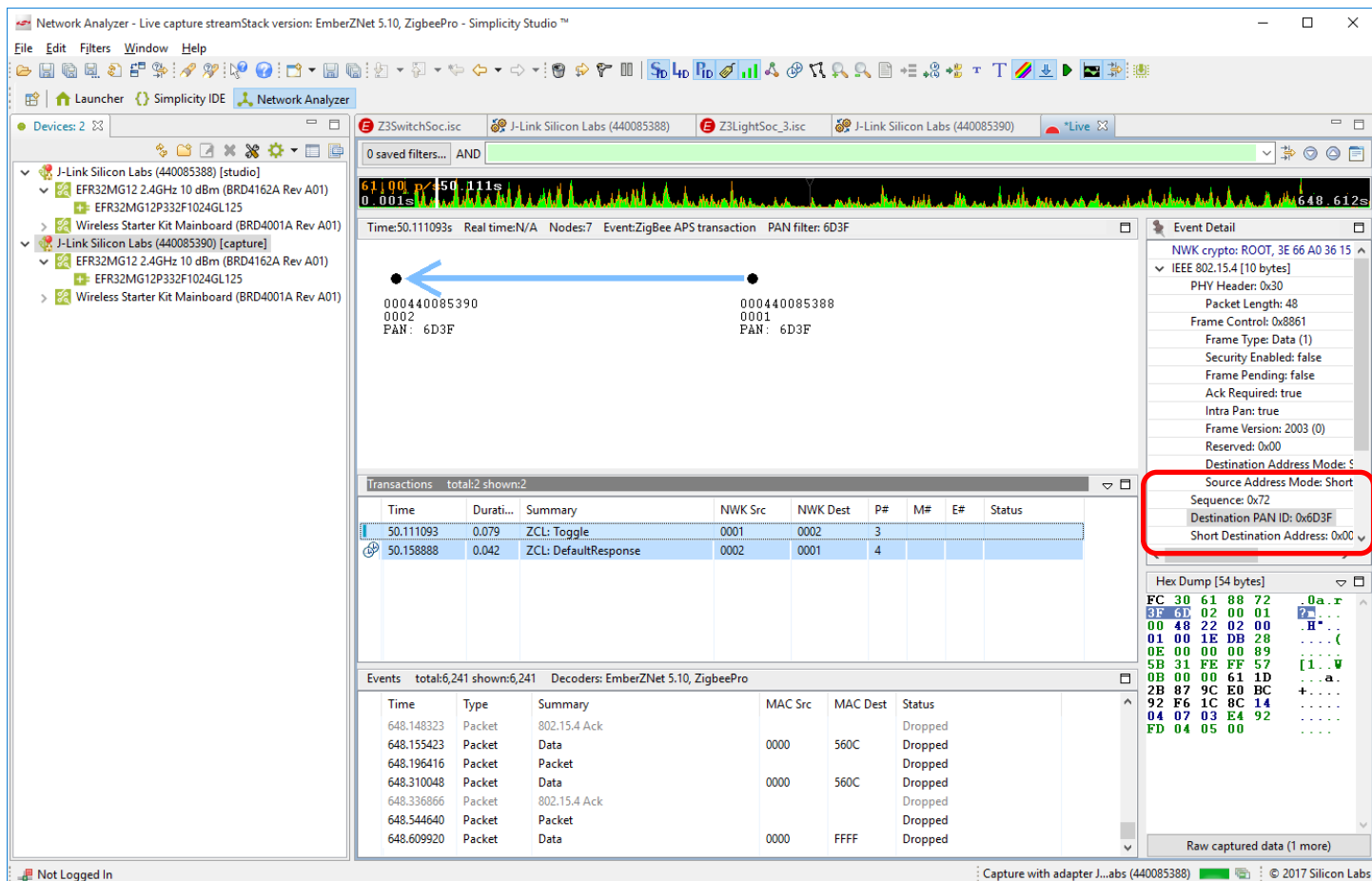
2. Click on one of the blue transactions.
3. In the Event Detail, expand IEEE802.15.4 and scroll down until you see the Destination PAN ID
4. Right-click on it, and select **Add to filter**.
5. Apply the filter by clicking the icon next to the green filter expression field, circled in the following image.
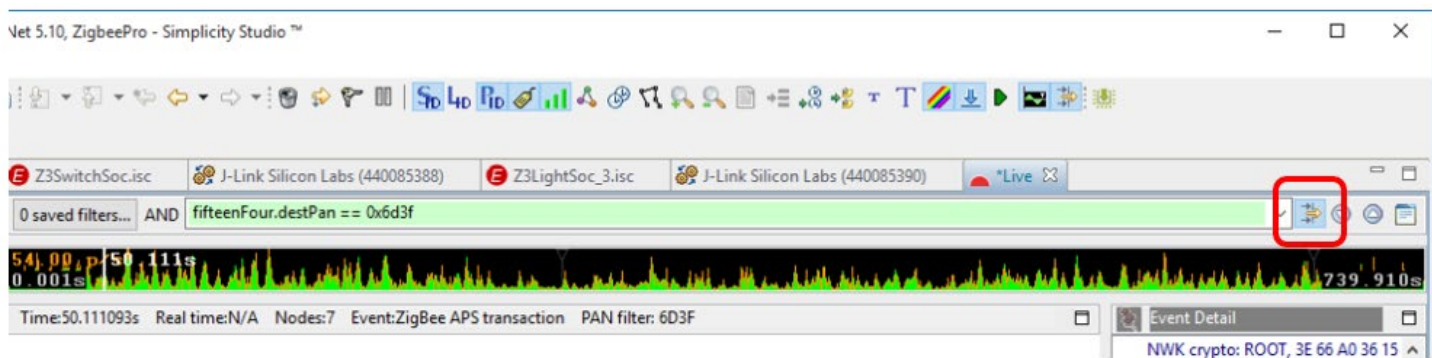
Now when you press button 0 you can clearly see each event associated with the transaction.
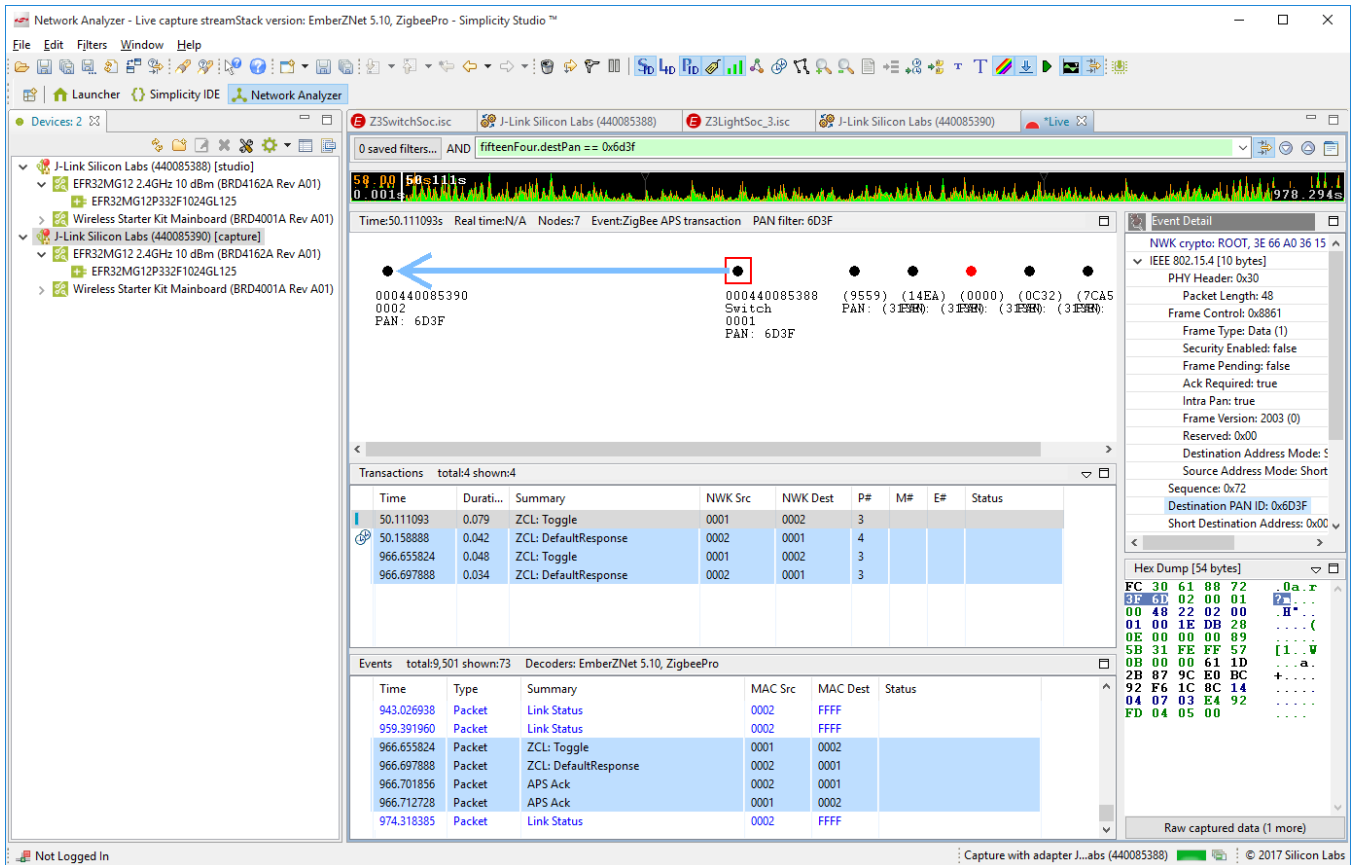
When analyzing more complex networks, you can drag and reposition the items shown in the map. By right-clicking on a device, you can also show connectivity and add labels. Labelling is useful not only in map, but also in the log. To label the full log, click **From beginning**.
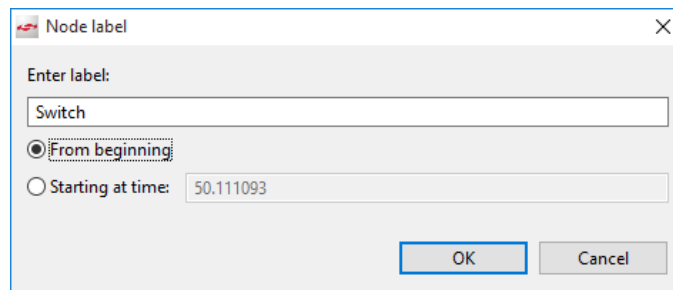
6   Next Steps

6.1   Example Applications

Explore configuring the example application to meet your needs. Much of the software configuration can be done through components and the Zigbee Cluster Configurator.

6.2   PIN Tool

Simplicity Studio offers a PIN tool that allows you to easily configure new peripherals or change the properties of existing ones. See the online Simplicity Studio v5 User's Guide: Pin Tool for more information.

6.3   Energy Profiler

You may also want to explore using the Energy Profiler tool. Energy Profiler provides energy debugging capability by displaying graphical real-time energy consumption information. This can be particularly useful for developing a low-power application. See the online Simplicity Studio 5 User's Guide: Energy Profiler for more information.

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

SILICON LABS

www.silabs.com