QSG181: Silicon Labs Wi-SUN SDK Quick-Start Guide

This document describes how to get started with Wi-SUN development using the Silicon Labs Wi-SUN software development kit (SDK) and Simplicity Studio® 5 with a compatible wireless starter kit.

NOTE: This version is compatible with Silicon Labs Wi-SUN SDK version 1.5.x and lower. For later SDK releases, see https://docs.silabs.com/wisun/latest/wisun-getting-started-overview/

KEY POINTS

- About the Silicon Labs Wi-SUN SDK
- About Example Applications and Demos
- Getting started with development
- Next steps
- Development tools

1 Introduction

Wireless Smart Ubiquitous Network (Wi-SUN) is the leading IPv6 sub-GHz mesh technology for smart city and smart utility applications. Wi-SUN brings Smart Ubiquitous Networks to service providers, utilities, municipalities/local government, and other enterprises, by enabling interoperable, multi-service, and secure wireless mesh networks. Wi-SUN can be used for large-scale outdoor IoT wireless communication networks in a wide range of applications covering both line-powered and battery-powered nodes.

Silicon Labs has enhanced Wi-SUN to work with Silicon Labs hardware. The Wi-SUN stack library is available as a software development kit (SDK) installed as part of the Gecko SDK (GSDK), the suite of Silicon Labs SDKs. To quickly get started with GSDK, start by installing Simplicity Studio 5 (SSv5), which will set up your development environment and walk you through GSDK installation. Simplicity Studio 5 includes everything needed for IoT product development with Silicon Labs devices, including a resource and project launcher, software configuration tools, full IDE with GNU toolchain, and analysis tools. Installation instructions are provided in the Simplicity Studio 5 online User's Guide.

Alternatively, Gecko SDK may be installed manually by downloading or cloning the latest from GitHub. See https://github.com/SiliconLabs/gecko_sdk for more information.

The Wi-SUN SDK includes a number of fully tested examples in source code. It supports a broad range of hardware, and includes documentation and tools.

This guide describes how to get started developing Wi-SUN applications using the Silicon Labs Wi-SUN SDK and SSv5.

1.1 About the Silicon Labs Wi-SUN SDK

The Silicon Labs Wi-SUN SDK is composed of the Wi-SUN FAN stack and example applications as well as the addition of metadata to allow for the seamless integration into Simplicity Studio 5.

The Silicon Labs Wi-SUN SDK is based on the Gecko Platform component-based design, where each component provides a specific function. Components are made up of a collection of source files and properties. The component-based design enables customization by adding, configuring, and removing components. The application developer can use SSv5's Project Configurator and Component Editor to easily assemble the desired features by including those components that match the required functionality and by configuring the various properties associated with those components.

The Silicon Labs Wi-SUN SDK contains the Wi-SUN stack in a library format.

For details on the Wi-SUN stack version included within the Silicon Labs Wi-SUN SDK, refer to the Wi-SUN SDK release notes.

1.1.1 Simplicity Studio 5 (SSv5)

SSv5 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) featuring industry-standard code editors, compilers and debuggers; and advanced, value-add tools for network analysis and code-correlated energy profiling.

SSv5 is designed to simplify developer workflow. It intelligently recognizes all Silicon Labs evaluation and development kit parts and, based on the selected development target, presents appropriate software development kits (SDKs) and other development resources.

The Silicon Labs Wi-SUN SDK is downloaded through SSv5. The GNU Compiler Collection (GCC) is provided with SSv5. Other important development tools provided with SSv5 are introduced in section 5 Development Tools.

1.1.2 Gecko Bootloader

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. By default, a new device is factory-programmed with a bootloader, which remains installed until you erase the device. The Gecko Bootloader is 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. For more information about bootloaders see *UG103.6: Bootloader Fundamentals*.

1.1.3 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 MbedTLS. For more information about Gecko Platform, see release notes that can be found in SSv5's Documentation tab, as well as online API documentation in https://docs.silabs.com/.

1.2 Prerequisites

Before following the procedures in this guide you must have:

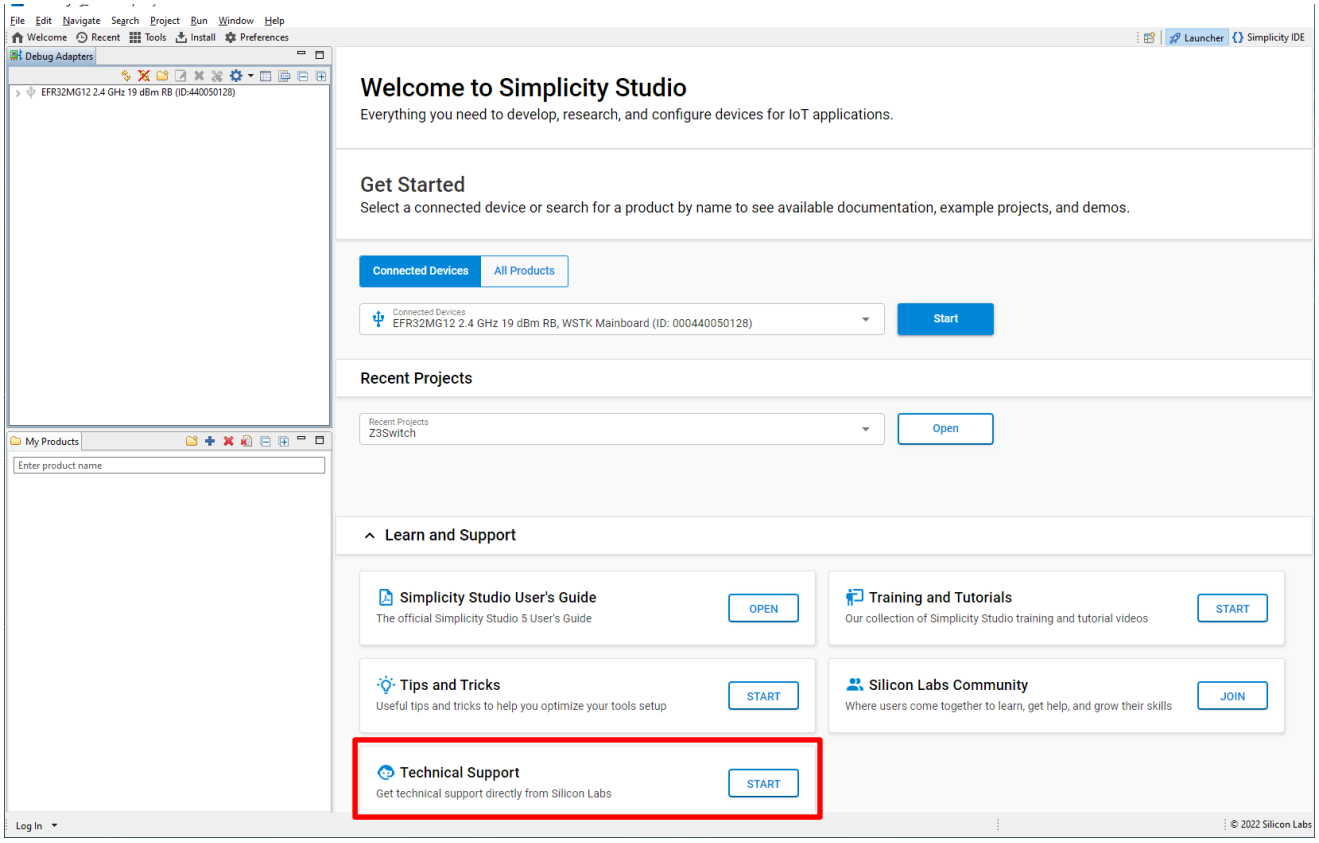
- Purchased one of the Wireless Gecko (EFR32) Portfolio Wireless Kits with compatible radio boards, listed on https://www.silabs.com/wireless/wi-sun.
- Downloaded SSv5 and the Gecko SDK and be generally familiar with the SSv5 Launcher perspective. SSv5 installation and getting started instructions along with a set of detailed references can be found in the online *https://docs.silabs.com/simplicity-studio-5-users-guide/latest/ss-5-users-guide-overview/*.
- Obtained a compatible compiler (See the Wi-SUN SDK's release notes for the compatible versions):
  - Simplicity Studio comes with a free GCC C-compiler.
  - IAR Embedded Workbench for ARM (IAR-EWARM) can also be used as the compiler for Silicon Labs Wi-SUN projects. Once IAR-EWARM is installed, the next time Simplicity Studio starts it will automatically detect and configure the IDE to use IAR-EWARM.

To get a 30-day evaluation license for IAR-EWARM:

- Go to the Silicon Labs support portal at https://www.silabs.com/support.
- Scroll down to the bottom of the page, and click **Contact Support**
- If you are not already signed in, sign in.
- Click the Software Releases tab. In the View list select **Development Tools**. Click **Go**. In the results is a link to the IAR-EWARM version named in the release notes.
- Download the IAR package (takes approximately 1 hour).
- Install IAR.
- In the IAR License Wizard, click **Register with IAR Systems to get an evaluation license**.
- Complete the registration and IAR will provide a 30-day evaluation license.
- Once IAR-EWARM is installed, the next time Simplicity Studio starts it will automatically detect and configure the IDE to use IAR-EWARM.

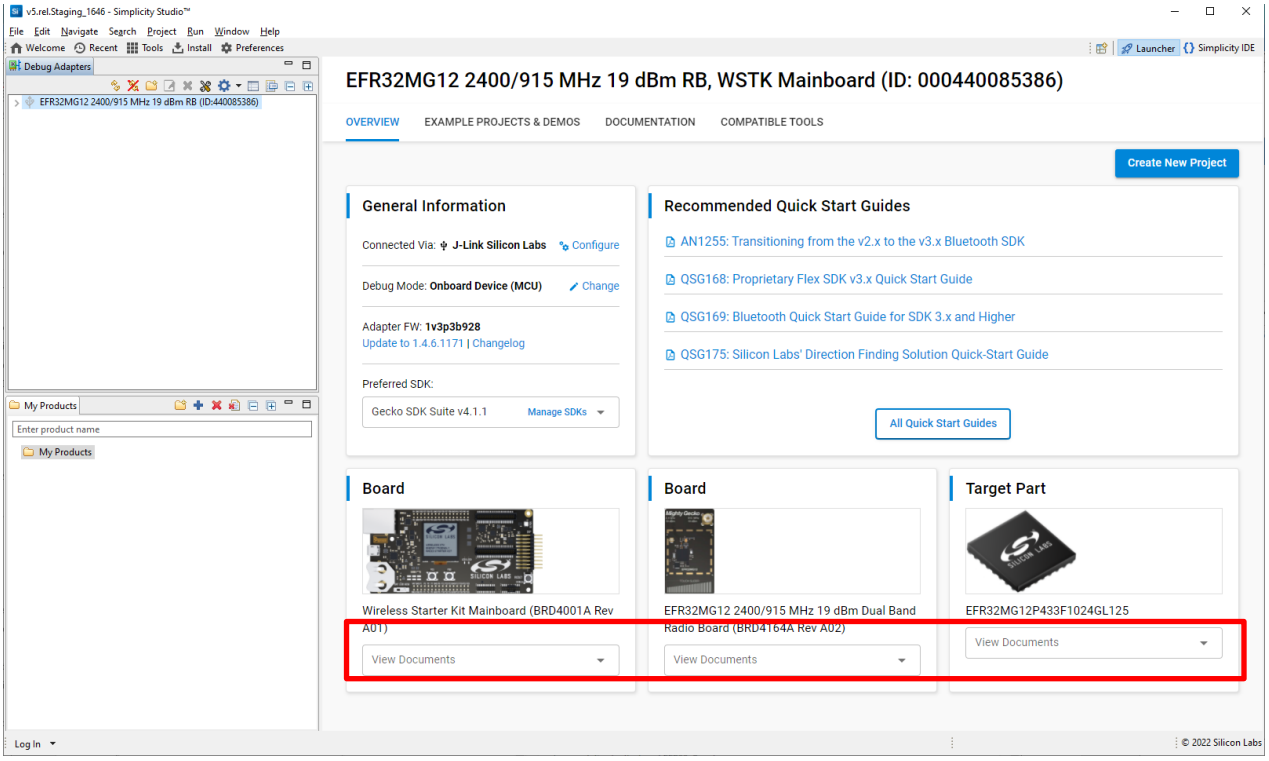
1.3 Support

Access the Silicon Labs support portal at https://www.silabs.com/support through SSv5's Welcome view under Learn and Support. Use the support portal to contact Customer Support for any questions you might have during the development process.

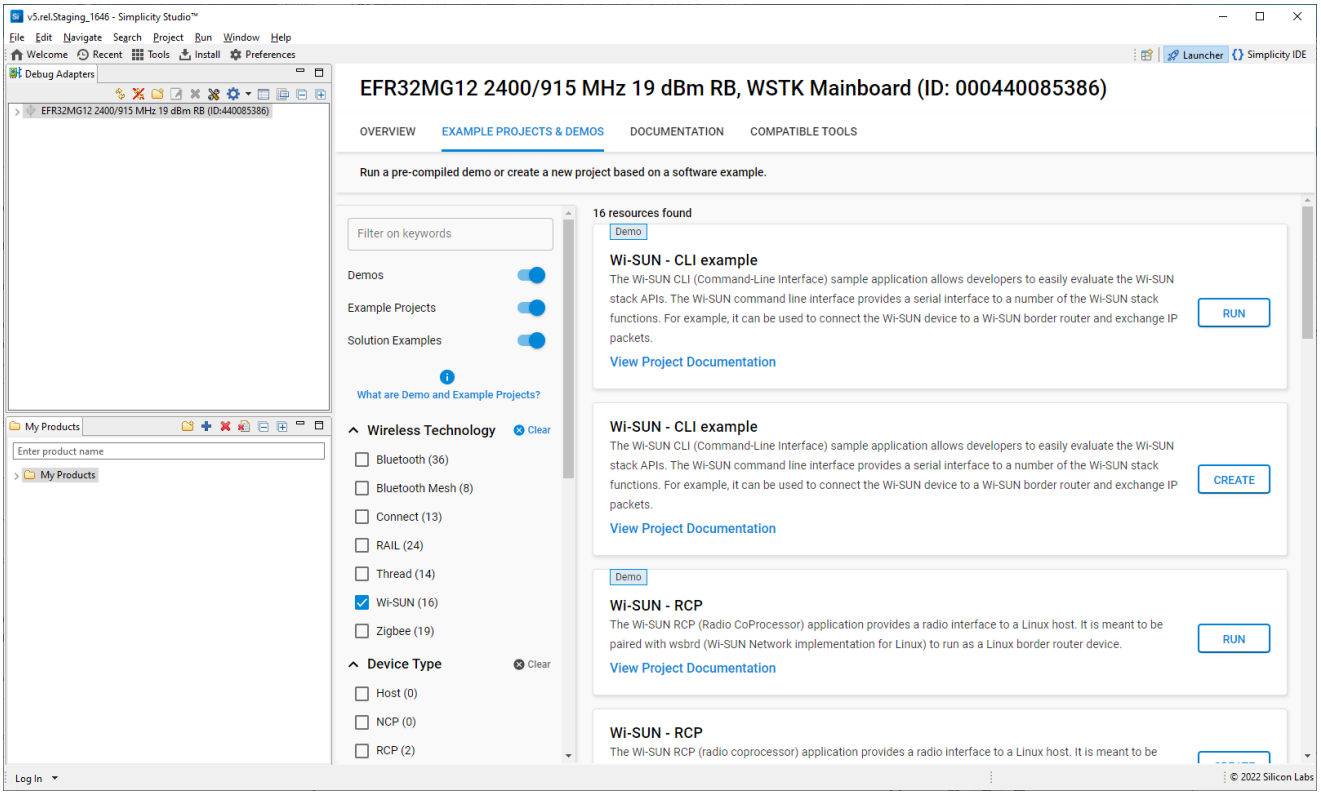


1.4 Documentation

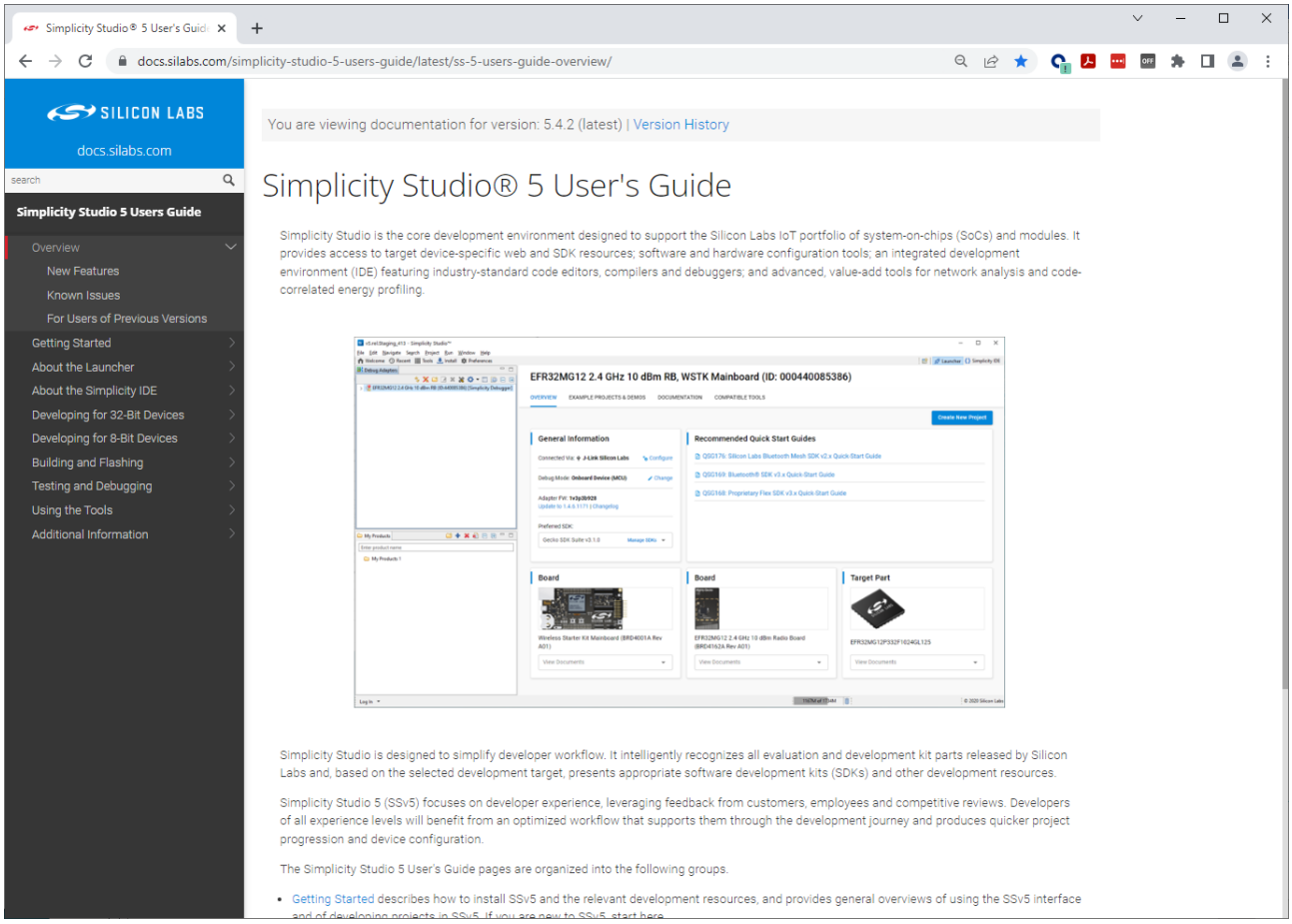
Relevant documentation is available through SSv5. It is filtered based on the device selected in either the Debug Adapters view or the My Product view. Hardware-specific documentation for the device can be accessed through links on the OVERVIEW tab.



SDK documentation and other references are available through the DOCUMENTATION tab. Filter with the Wi-SUN Technology Type checkbox to see documentation most closely related to the Wi-SUN SDK. To see documents specific to Wi-SUN, select 'Wi-SUN' under **Wireless Technology**.



SSv5 and its tools are documented in the online [Simplicity Studio 5 User's Guide](#).



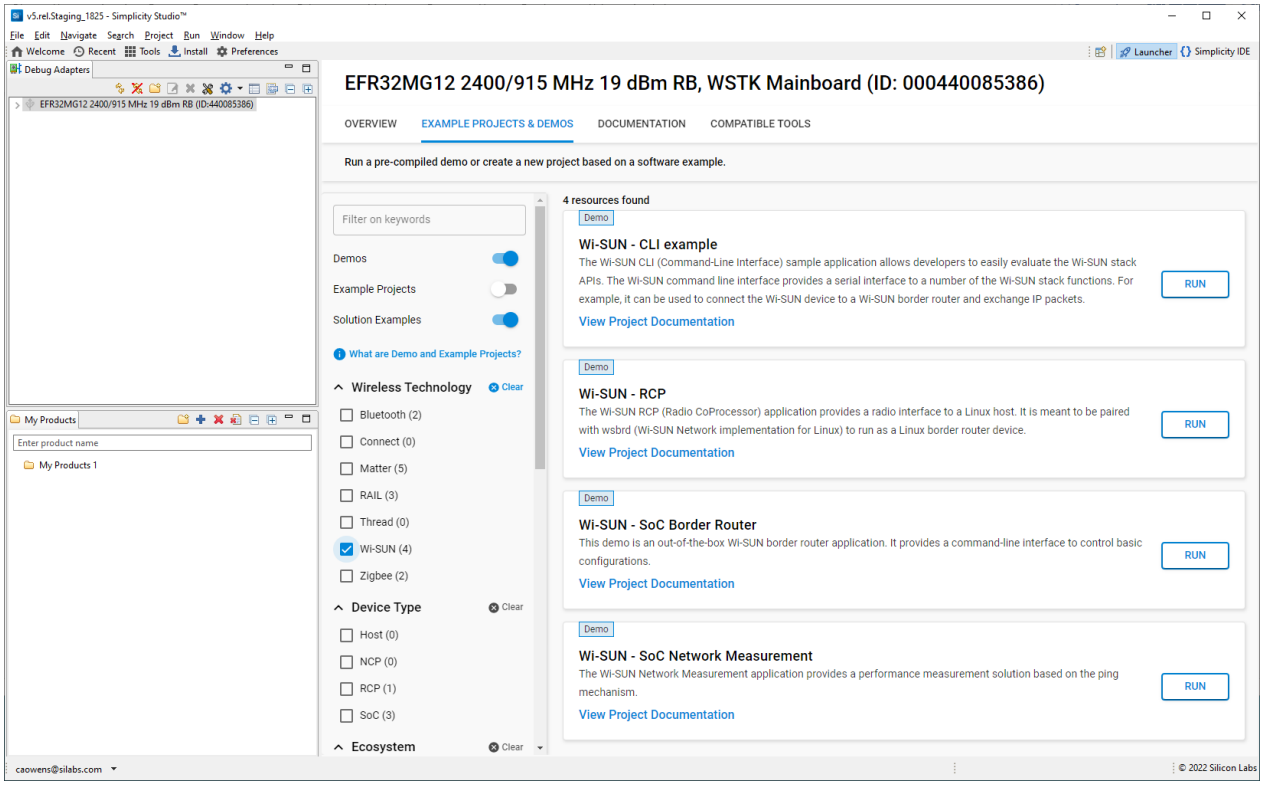
2 About Example Applications and Demos

Because starting application development from scratch is difficult, the Silicon Labs Wi-SUN SDK comes with a number of built-in example applications and demos covering the most frequent use cases designed to illustrate common application functions. Silicon Labs strongly recommends starting development from one of the example applications.

Like everything in SSv5, the examples and the demos shown on the EXAMPLE PROJECTS & DEMOS tab are filtered based on the part you have connected or selected.

2.1 Demos

Demos are prebuilt firmware images that are ready to download to a compatible device. The quickest way to find if a demo is available for your part is by adding the part or board information in the My Products view and then navigating to the EXAMPLE PROJECTS & DEMOS tab in the Launcher Perspective. Disable the Example Projects filter. The Solution Examples filter is provided for future use.



Precompiled demo application images provided with the Wi-SUN SDK are compatible with the following boards:

- brd4163a
- brd4164a
- brd4170a
- brd4172a
- brd4173a
- brd4253a
- brd4254a

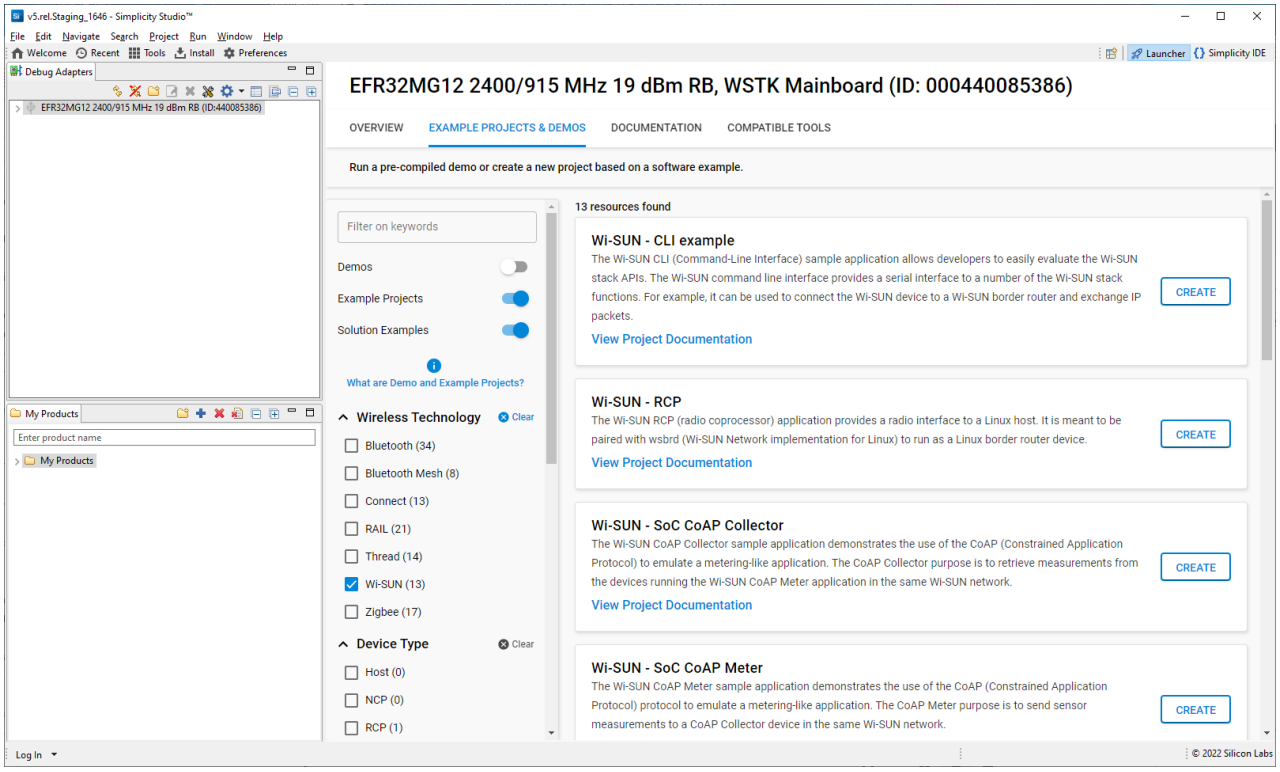
2.2 Software Examples

When you work with examples, the process is:

- Select an example
- Change the example configuration (if needed)
- Build the example

- Load the example to the target device

Since typically you will finish by flashing a compiled application image to a device, connect a device to your computer and select it in the Debug Adapters view. In the EXAMPLE PROJECTS & DEMOS tab on the Launcher perspective, enter 'wi-sun' as a keyword. A number of other filters are provided. To see the examples only, turn off **Demo**s.



Each example has its own documentation accessible by clicking **View Project Documentation** below the example description. The HTML document covers the steps to set up the example and run the associated demonstration.

The example applications provided with the Silicon Labs Wi-SUN SDK are as follows.

Wi-SUN – CLI example: Acts as a Wi-SUN router node in a network, and provides an interface device functionality.

Wi-SUN – SoC CoAP Collector: Collects data from other devices configured as meters using CoAP.

Wi-SUN – SoC CoAP Meter: Provides basic meter functionality to communicate with a collector using CoAP.

Wi-SUN – SoC Empty: Provides a basic framework to begin adding custom functionality.

Wi-SUN – SoC Collector: Collects data from other devices configured as meters.

Wi-SUN – SoC Network Measurement: Provides a tool to measure the Wi-SUN solution performance

Wi-SUN – SoC Meter: Provides basic meter functionality to communicate with a collector.

Wi-SUN – SoC Ping: Provides simple connectivity testing.

Wi-SUN – SoC TCP Client: Works with the TCP server example using the TCP protocol.

Wi-SUN – SoC TCP Server: Works with the TCP client example using the TCP protocol.

Wi-SUN – SoC UDP Client: Works with the UDP server example using the UDP protocol.

Wi-SUN – SoC UDP Server: Works with the UDP client example using the UDP protocol.

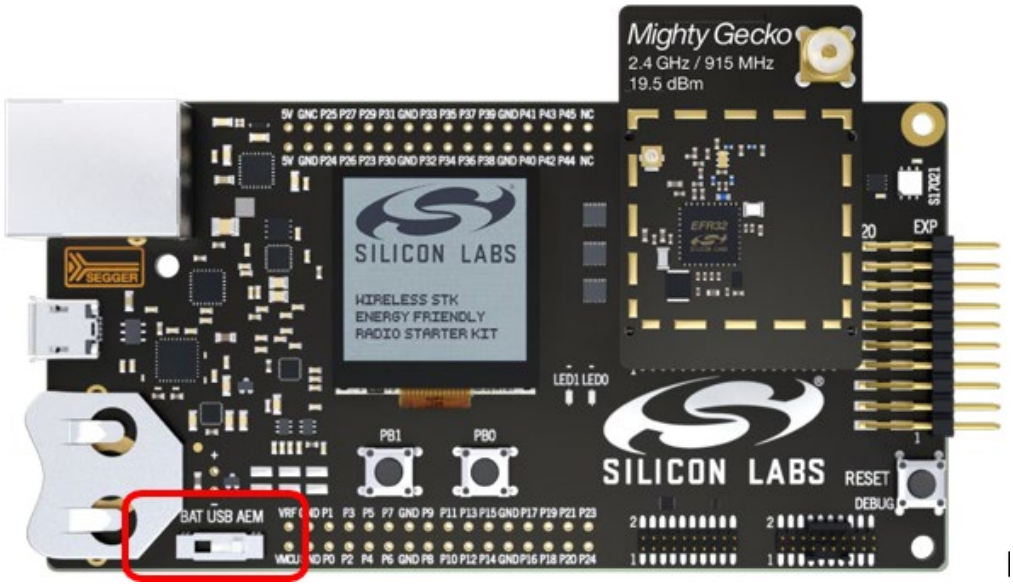
Wi-SUN – RCP: Radio co-processor border router implementation that pairs with a Linux host running the Wi-SUN stack upper layers.

3 Getting Started with Development

This section assumes that you have downloaded SSv5 and the Silicon Labs Wi-SUN SDK, and are familiar with the features of the SSv5 Launcher perspective.

You should have your mainboard connected.

Note: For best performance in Simplicity Studio 5, be sure that the power switch on your mainboard is in the Advanced Energy Monitoring or “AEM” position, as shown in the following figure.



In these instructions you will compile and load a simple **Wi-SUN Ping** application on a node and a **Wi-SUN Border Router** demo image on another node. Section 3.6 Creating a Network describes how to use the examples to create a network. Section 5.3 Network Analyzer describes how to use Network Analyzer to observe traffic across the network.

When working with an example application in Simplicity Studio, you will be executing the steps in the following order:

1. Create a project based on an example.
2. Configure the project.
3. Build the application image and flash it to your device.

These steps are described in detail in the following sections. These procedures are illustrated for a mainboard with an EFR32MG12. Note: Your SDK version may be later than the version shown in the figures.

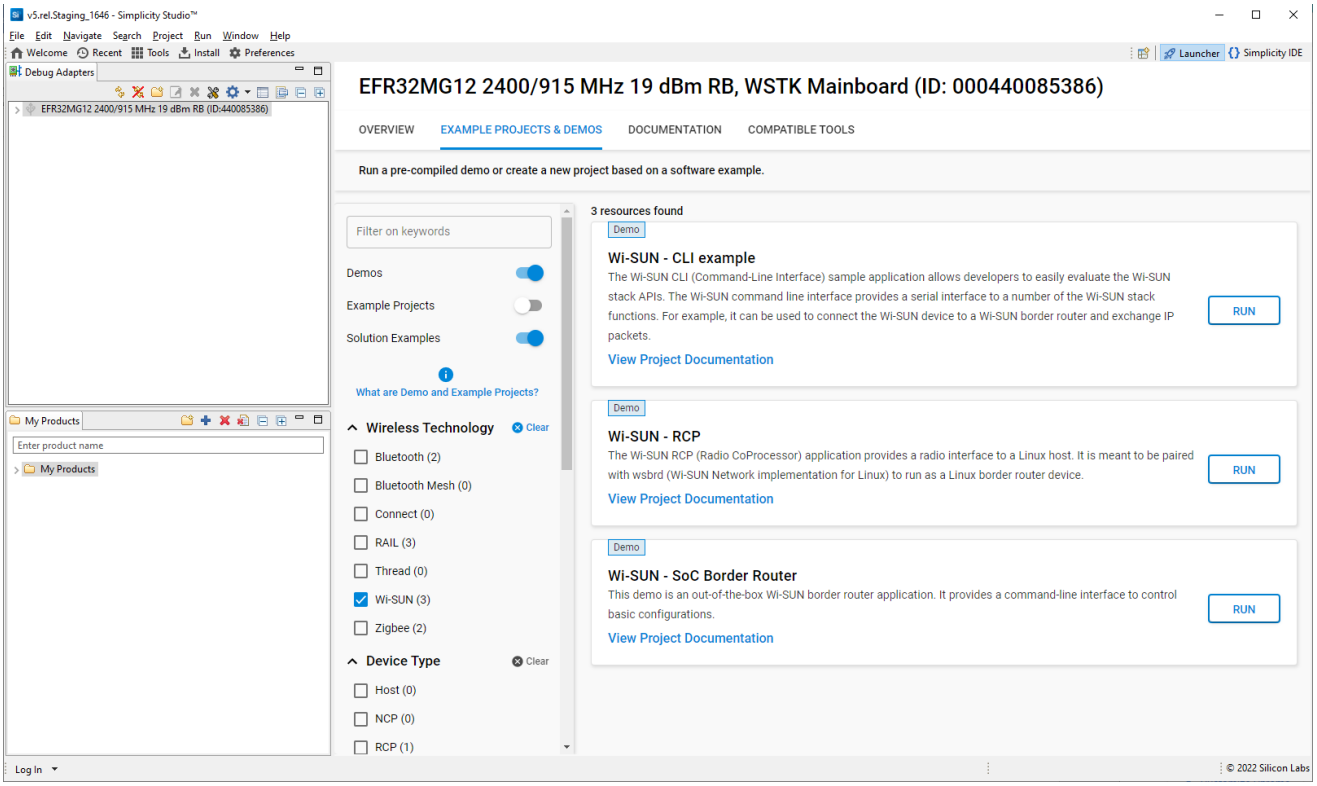
3.1 Flashing the Wi-SUN Border Router

Every Wi-SUN example requires a Wi-SUN Border Router to create and manage a Wi-SUN Network for the Wi-SUN devices to join. It allows an easy and quick medium to evaluate the Silicon Labs Wi-SUN stack solution without deploying an expensive and cumbersome production-grade Wi-SUN Border Router. A CLI (Command-Line Interface) is exposed to facilitate the configuration.

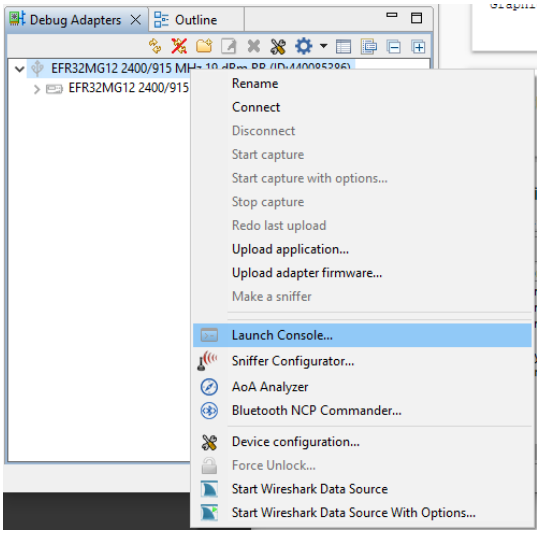
Note: Make sure your mainboard has the latest “Adapter FW” to avoid any issue when using the example CLI. To do so:

- In the Launcher perspective, click the radio board listed in the Debug Adapters view.
- In the OVERVIEW tab, verify the Adapter FW version in the General Information card.
- If Simplicity Studio 5 proposes to update the firmware, do so.

The Wi-SUN Border Router demonstration is delivered only in a binary format. The implementation does not scale for a production-grade Border Router maintaining several thousand Wi-SUN nodes. For a production-grade border router solution, refer to AN1332: Silicon Labs Wi-SUN Network Setup and Configuration to get started with Silicon Labs Linux-based Wi-SUN border router solution.



1. In the Debug Adapters view, select the device that will be the Wi-SUN border router. Note: To rename the device so that you know which device is the Border Router, right-click it and select **Rename** on the context menu.
2. Navigate to the EXAMPLE PROJECTS & DEMOS tab and turn off the Example Projects filter. Click **RUN** next to **Wi-SUN – SoC Border Router** demo.
3. Start the Wi-SUN Border Router with the CLI interface. In the Debug Adapters view, right-click the Border Router device and click **Launch Console**, as shown. Alternatively, click Tools in the Simplicity IDE menu and select Device Console.



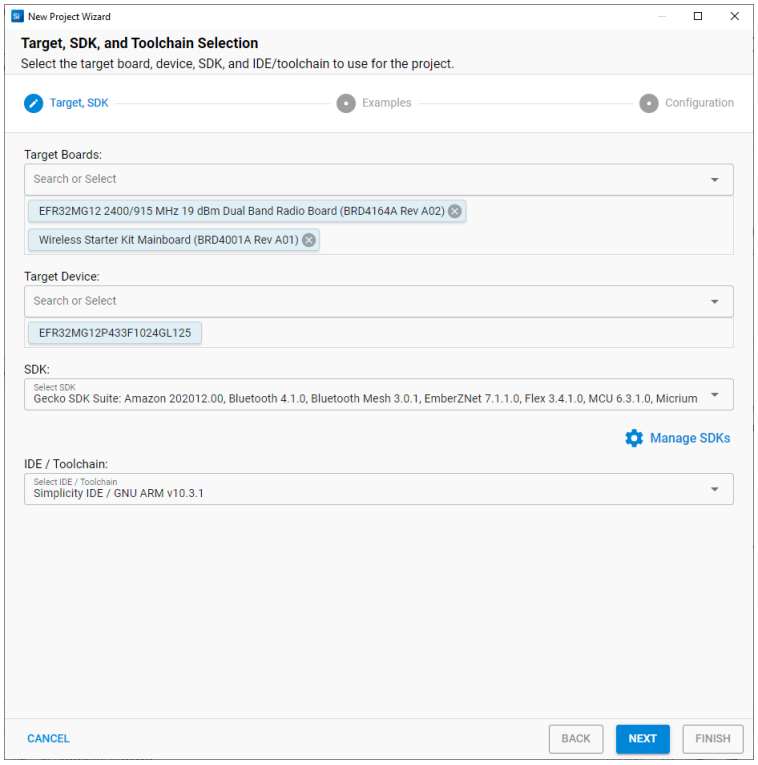
4. To get a prompt on the Console, go to the Serial 1 tab and press Enter. To start the Wi-SUN Border Router, enter:

```
> wisun start
Border router started
```

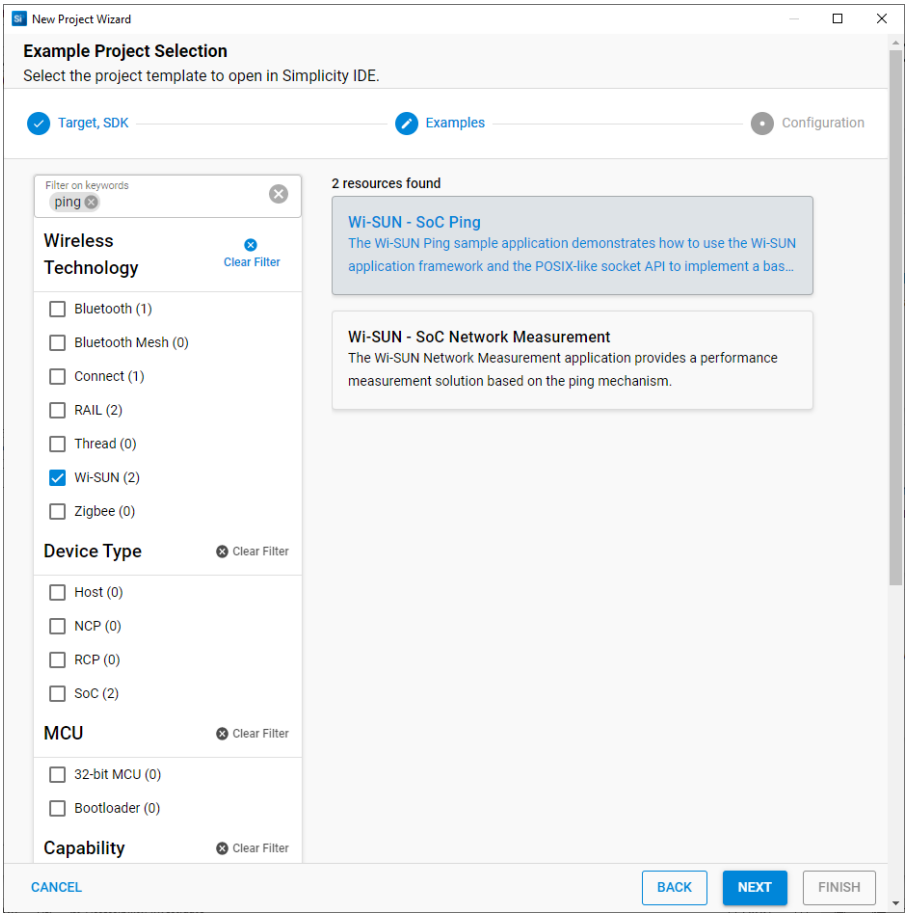
3.2 Creating a Project Based on an Example

Simplicity Studio 5 (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 guide uses the **File > New > Silicon Labs Project Wizard** method because it takes you through all three of the Project Creation Dialogs. Details on each creation dialog option may be found in the *Simplicity Studio 5 User's Guide*.

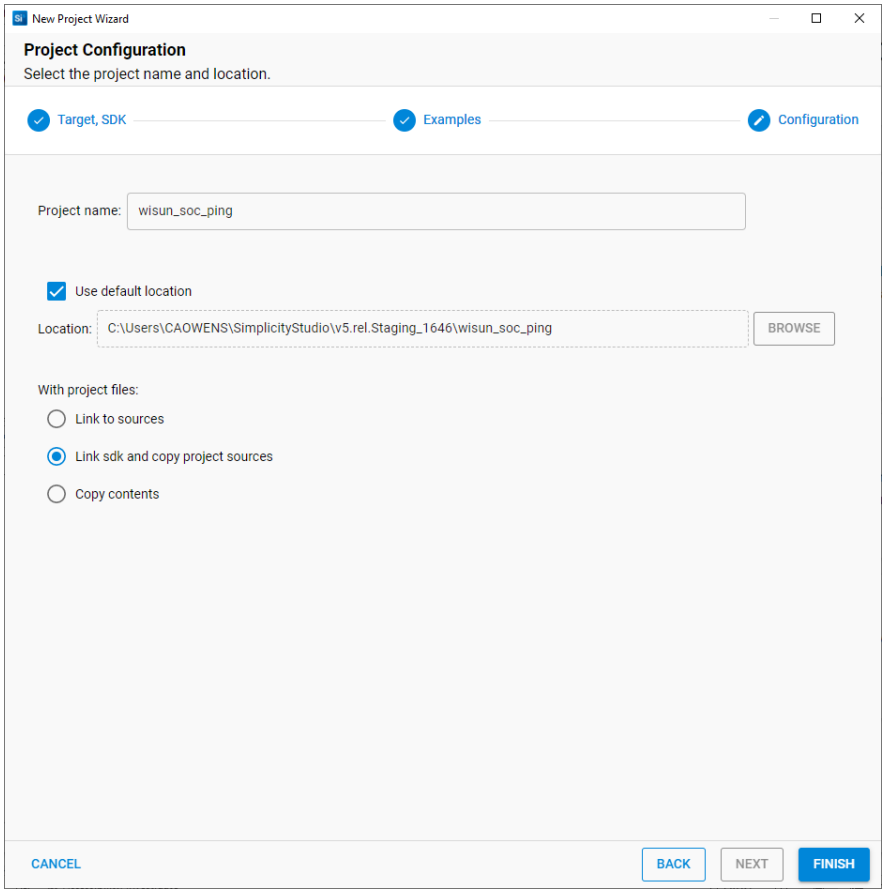
1. In the Debug Adapters view, select the target part for the application node. This should be a different part than the one used for the border router in the previous section.
2. Open SSv5's File menu and select **New > Silicon Labs Project Wizard**. The Target, SDK, and Toolchain Selection dialog opens. Do not change the default **Simplicity IDE / GNU ARM v<version>** toolchain supported by Wi-SUN. Click **NEXT**.



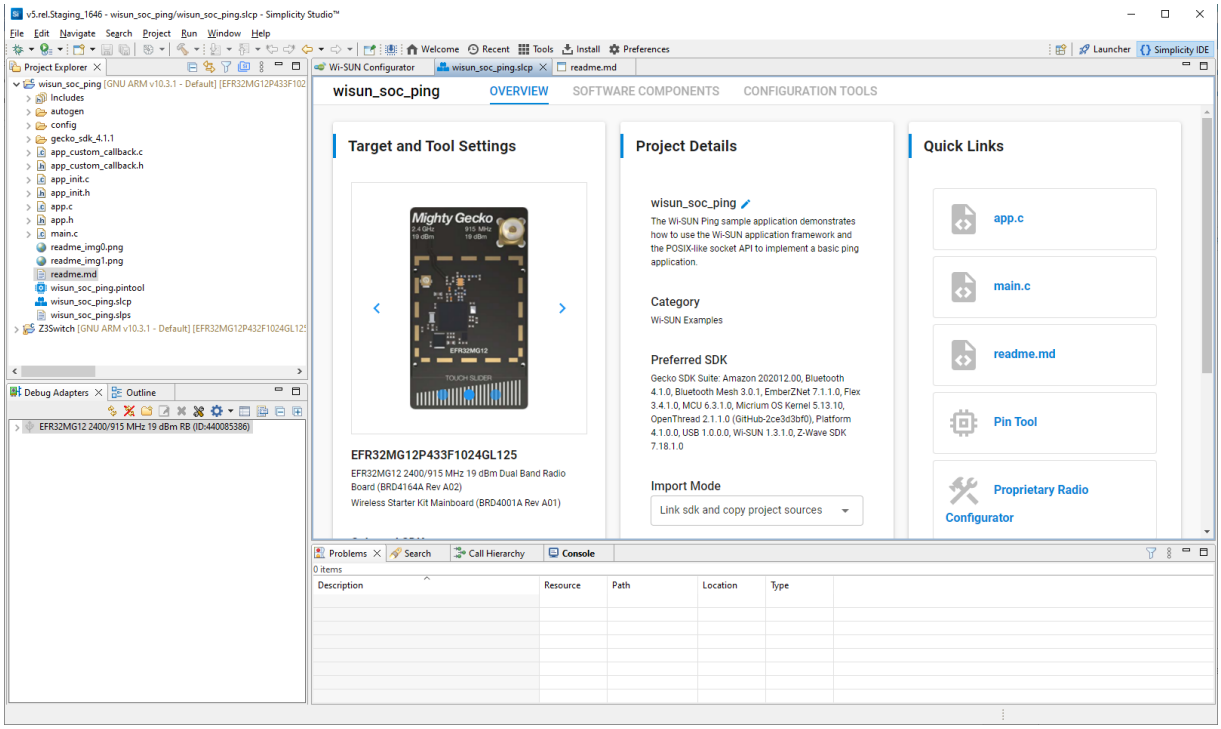
3. The Example Project Selection dialog opens. Use the 'Wi-SUN' Technology Type and Keyword filters to search for a specific example, in this case **Wi-SUN – SoC Ping**. Select it and click **NEXT**.



4. 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.**

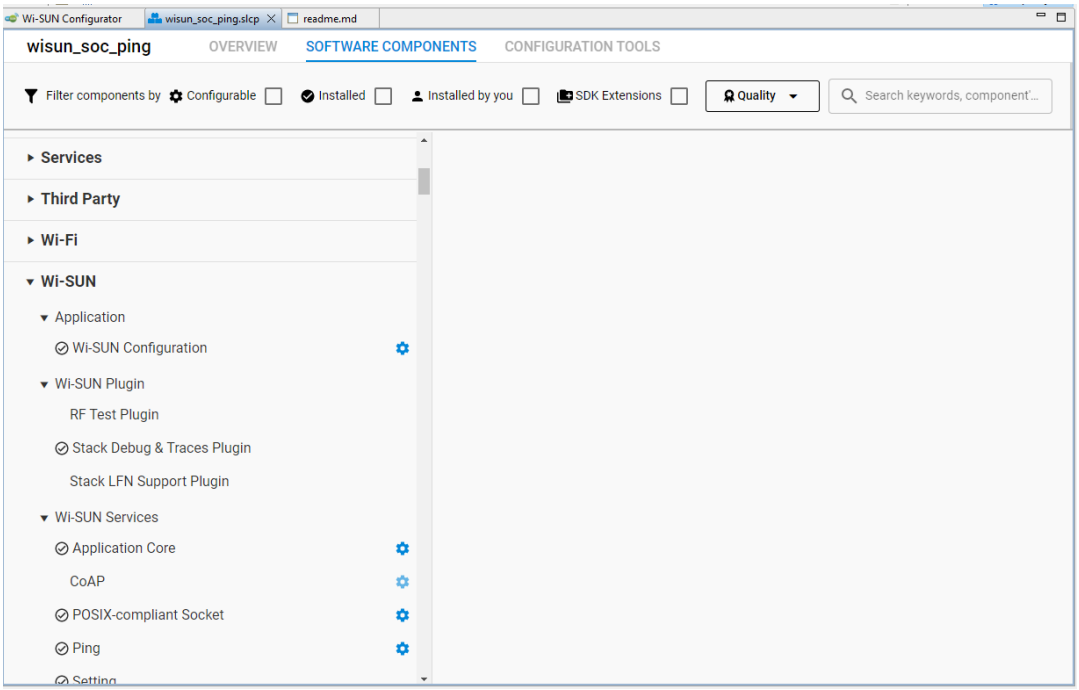


5. The Simplicity IDE Perspective opens with the project documentation (readme.md). Click the <project>.slcp tab to see the Project Configurator OVERVIEW tab. See the online Simplicity Studio 5 User's Guide for details about the functionality available through the Simplicity IDE perspective and the Project Configurator.



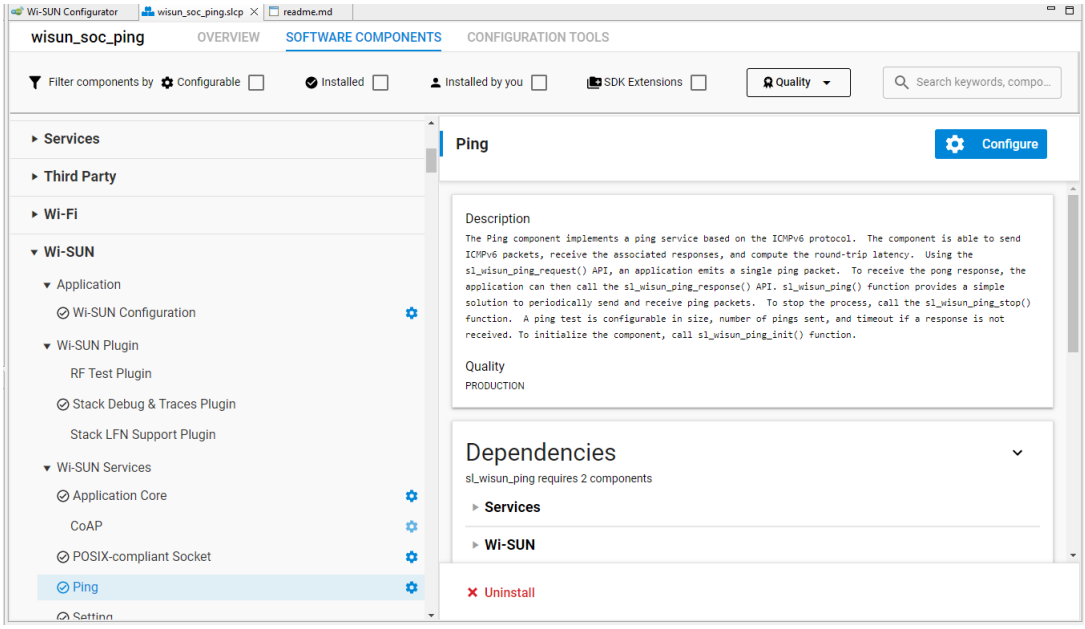
3.3 Configuring the Project

Silicon Labs Wi-SUN applications are built on a Gecko Platform component structure. Click the SOFTWARE COMPONENTS tab to see a complete list of Component categories.



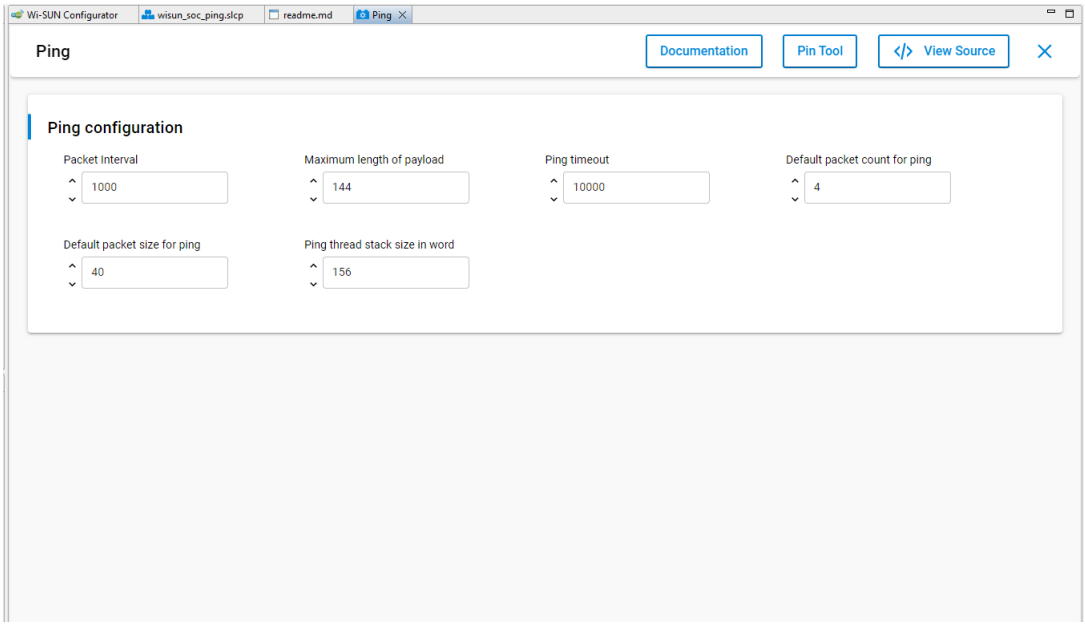
The project is configured by installing and uninstalling components and configuring installed components. Installed components are shown with a circled checkmark on their left. Click **Installed Components** to see a filtered list of components installed by the example application.

Configurable components have a gear symbol on their right. Select a component to see information about it.

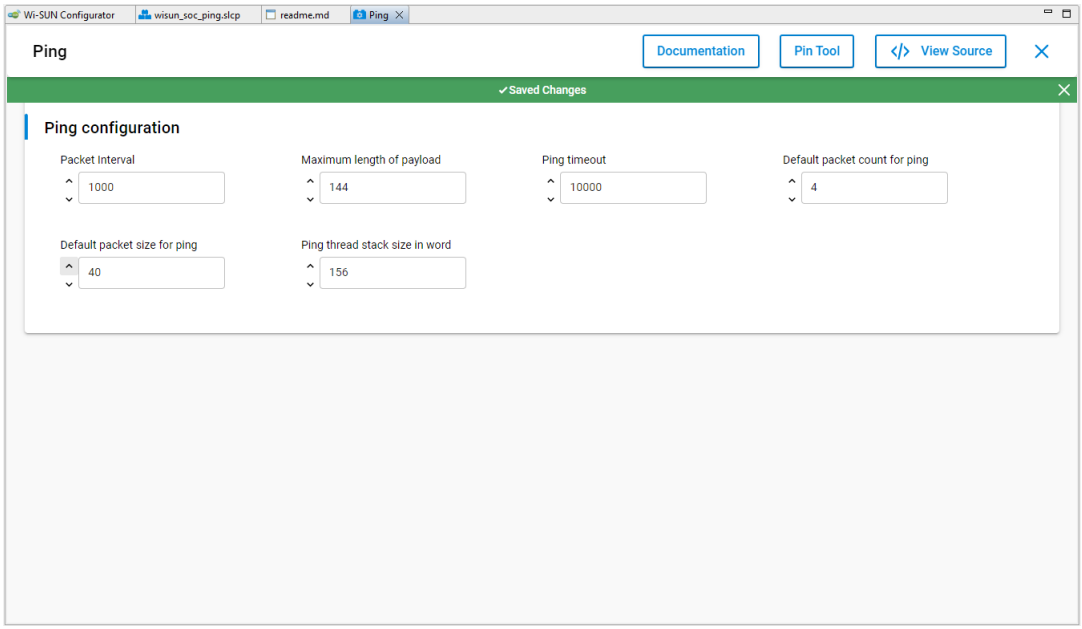


If the component is configurable, click **CONFIGURE** to open the Component Editor in a new tab.

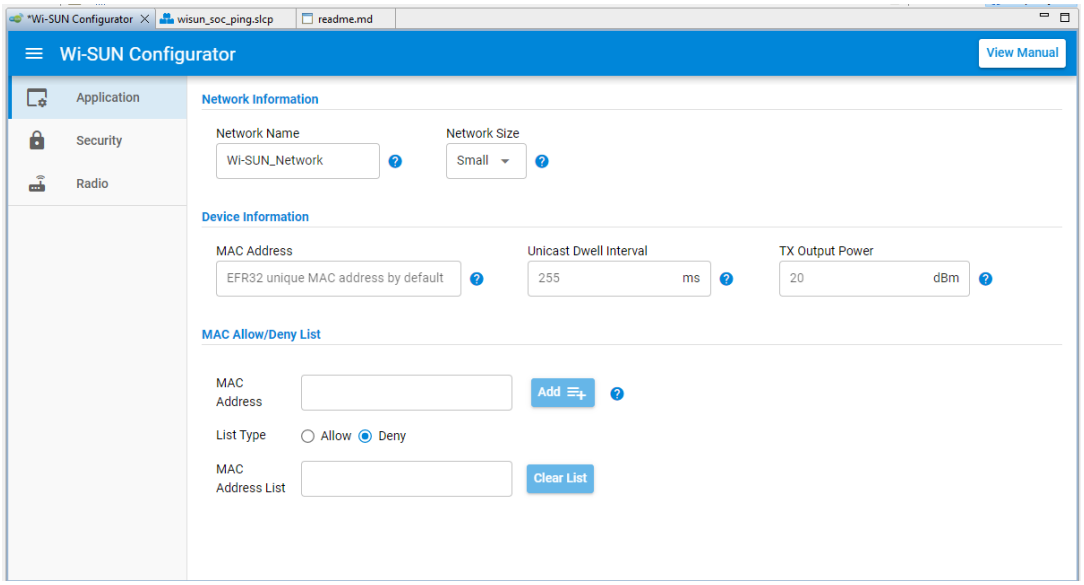
For example, in the Ping component you can configure various parameters that change the way the wisun-ping application behaves.



Any changes you make are autosaved, and project files are autogenerated.



In addition to the Project Configurator tab (<project>.slcp), a Wi-SUN Configurator tab is also available. For more information about using the Wi-SUN Configurator and how to change the default Wi-SUN PHY, see *UG495: Silicon Labs Wi-SUN Developer's Guide*.



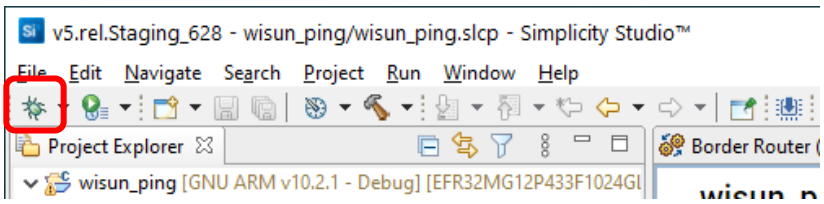
If you make changes, the Wi-SUN Configurator tab has an asterisk to the left. Save changes (CTRL-S) when you are finished updating Wi-SUN Configurator settings. If you build the project without saving changes, the changes are saved automatically.

3.4 Building the Project

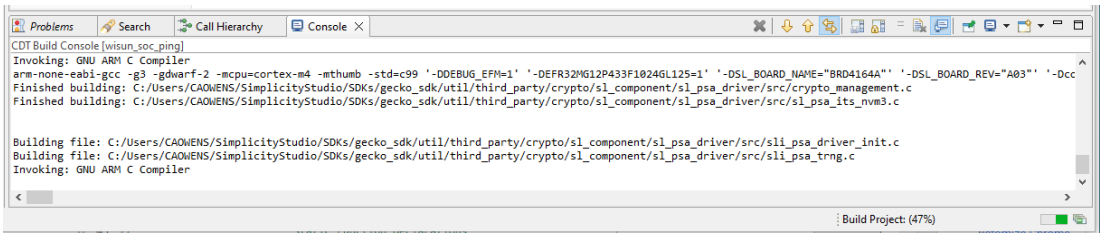
You can either compile and flash the application automatically, or manually compile it and then flash it.

3.4.1 Automatically Compile and Flash

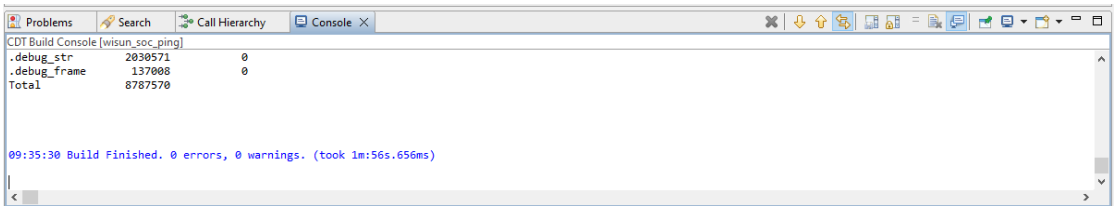
1. You can automatically compile and flash the application to your connected development hardware in the Simplicity IDE, and open a Debug interface. Click the **Debug** control.



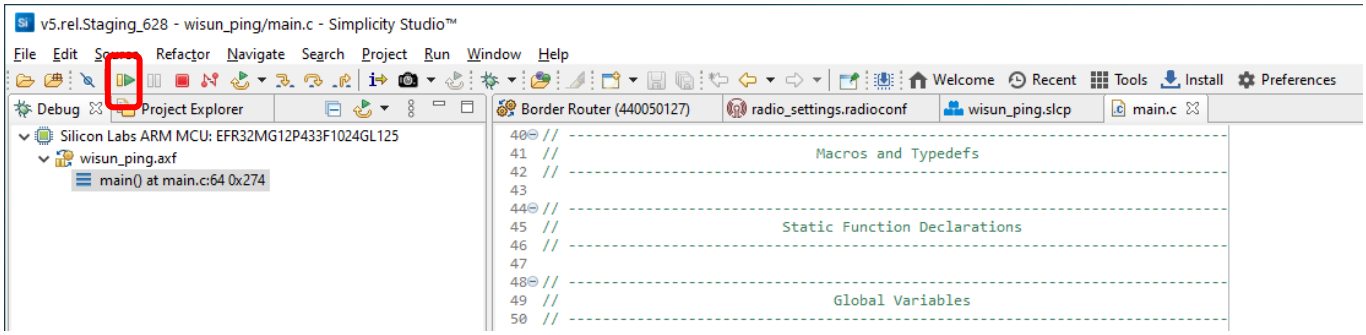
2. Progress is displayed in the console and a progress bar in the lower right.



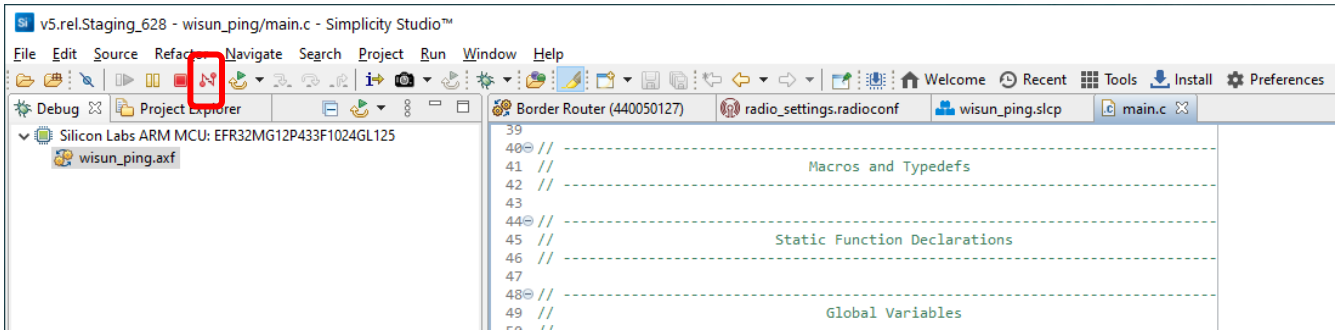
The project should build without error.



3. When building and flashing are complete a Debug perspective is displayed. Click the **Resume** control to start the application running on the WSTK.

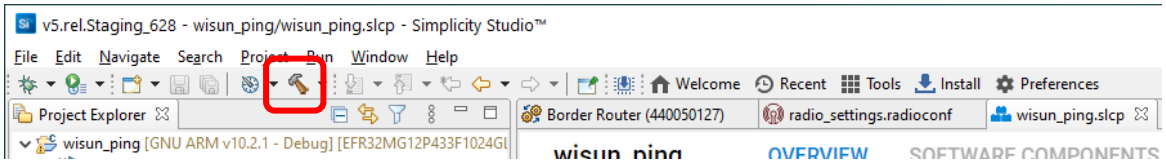


Next to the Resume control are **Suspend**, **Terminate**, **Disconnect**, and **stepping** controls. Click **Disconnect** when you are ready to exit Debug mode.



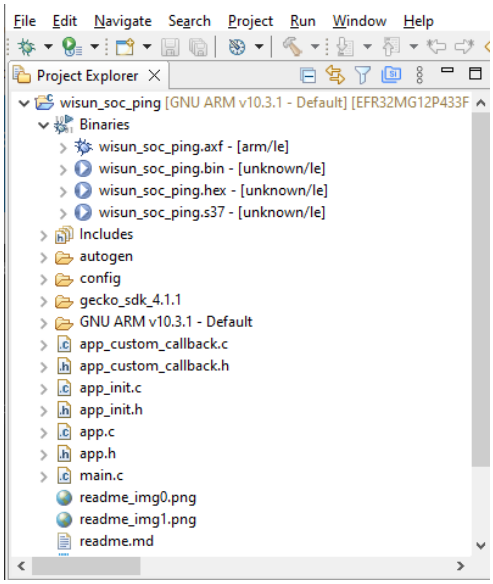
3.4.2 Manually Compile and Flash

1. After you generate your project files, instead of clicking Debug, click the **Build** control (hammer icon) in the top tool bar.

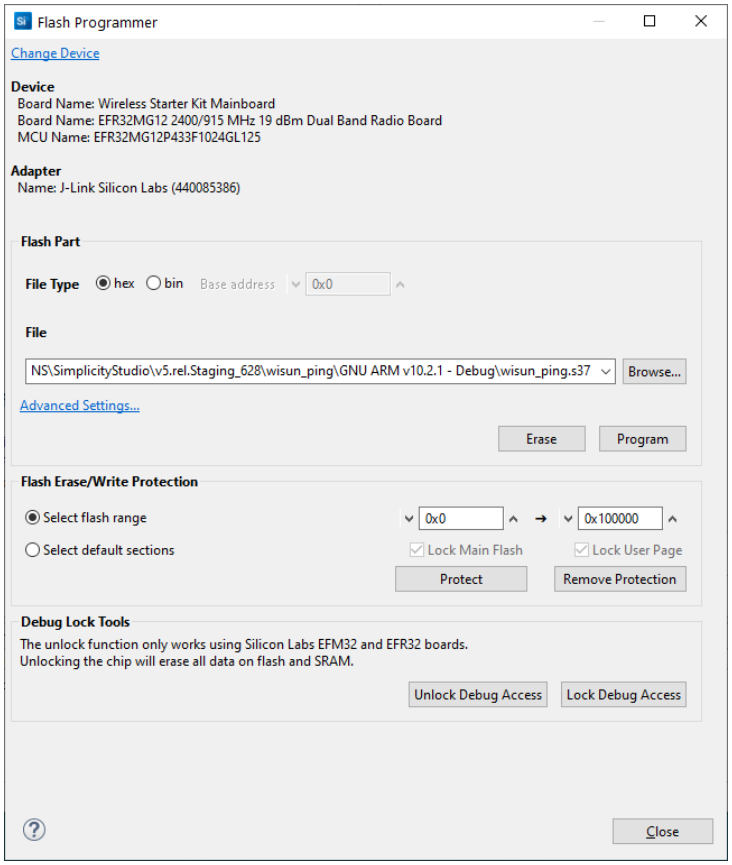


2. You can load the binary image through Project Explorer view.

   Locate the <project>.bin, .hex, or .s37 file in the Binaries subdirectory.



Right-click the file and select **Flash to Device...**. If you have more than one device connected, select a device to program. The Flash Programmer opens with the file path populated. Click **PROGRAM** to flash the image to the device.



3.5 Flashing a Bootloader

All Silicon Labs examples require that a bootloader be installed. 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, skip this step and continue with the next section.

With Silicon Labs Wi-SUN, the bootloader serves to start the application code within the image you created and flashed in the previous procedure. Once you have installed a bootloader image, it remains installed until you erase the device.

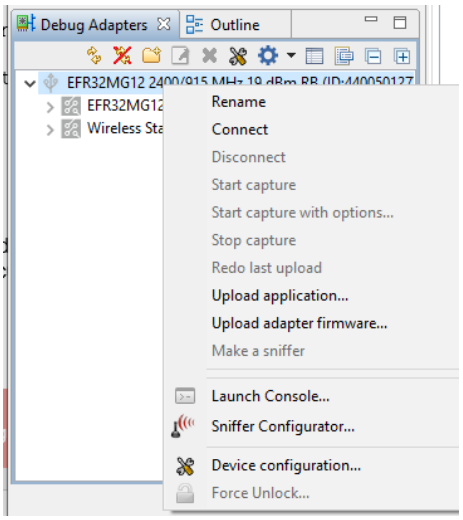
To flash a bootloader, first select one of the bootloader examples, such as **SPI Flash Storage Bootloader (single image)**, and build and flash it as described above. For more information see UG266: Silicon Labs Gecko Bootloader User's Guide for GSDK 3.2 and Lower or UG489: Silicon Labs Gecko Bootloader User's Guide for GSDK 4.0 and Higher.

If you are working with the Gecko Bootloader, bootloader images must be formatted as GBL files. 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.

3.6 Creating a Network

Depending on the example application, you may be able to interact with it through your development environment's Console interface using a CLI (command-line interface). The console interface allows you to form a network and send data using the border router device created in section 3.1 Flashing the Wi-SUN Border Router and the application node.

To launch the Console interface, in the Simplicity IDE perspective right-click the application node in the Debug Adapters View. Select **Launch Console**. Alternatively, click Tools in the Simplicity IDE menu and select Device Console.



To get a prompt on the Console, go to the Serial 1 tab and press Enter.

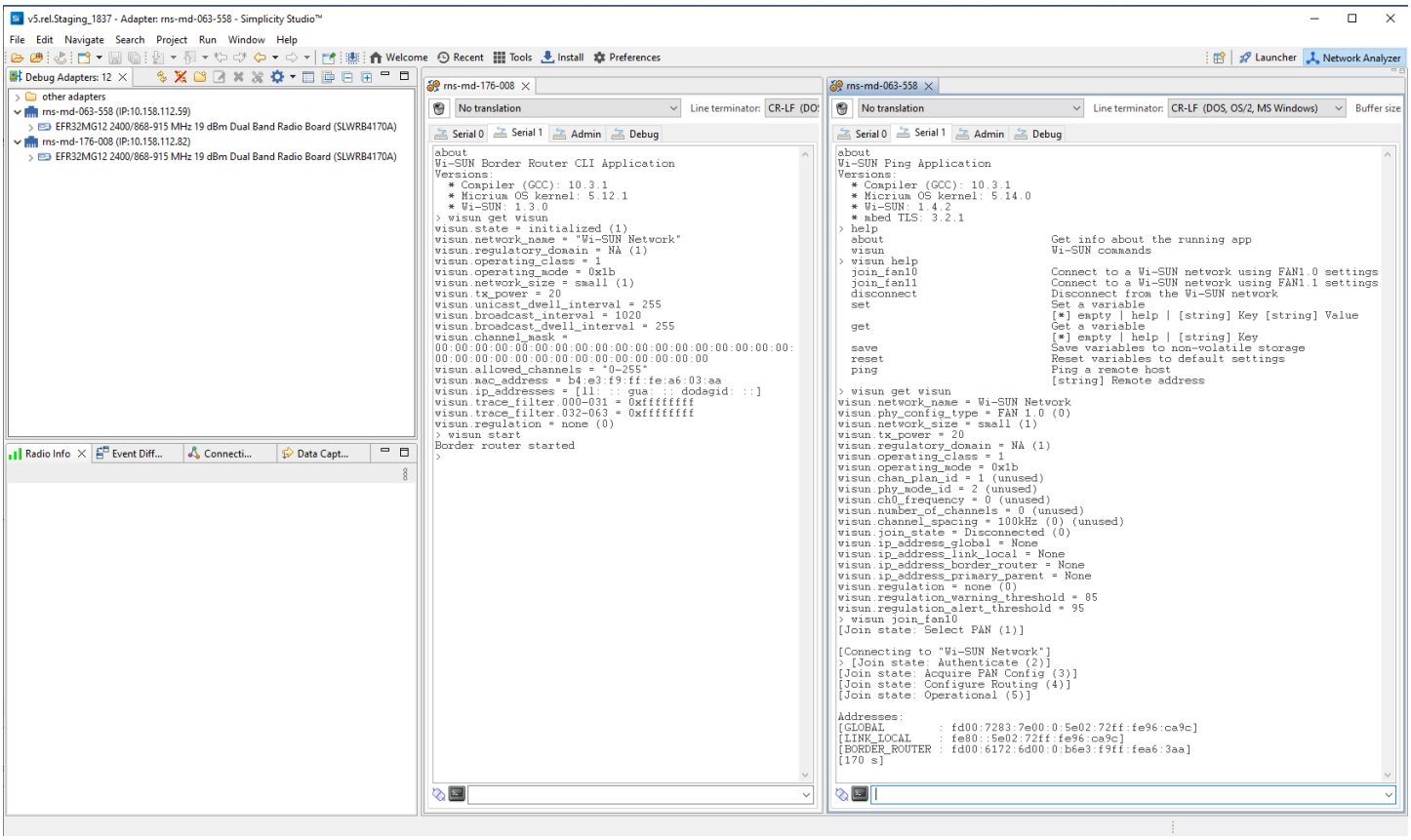
Connect the Wi-SUN Ping WSTK

The Wi-SUN Ping application automatically starts by connecting to the Border Router. If the connection is successful, the application should output the traces below in the console.

```
[Connecting to "Wi-SUN Network"]
> [Join state: Authenticate (2)]
[Join state: Acquire PAN Config (3)]
[Join state: Configure Routing (4)]
[Join state: Operational (5)]

Addresses:
[GLOBAL        : fd00:7283:7e00:0:5e02:72ff:fe96:ca9c]
[LINK_LOCAL    : fe80::5e02:72ff:fe96:ca9c]
[BORDER_ROUTER : fd00:6172:6d00:0:b6e3:f9ff:fea6:3aa]
[170 s]
```

The following is an illustration of connecting the Wi-SUN SoC Border Router application to the Wi-SUN SoC Ping Example Application. This setup allows showing the Border Router and the device consoles side by side, where Wi-SUN settings can be compared. If these do not match, the connection will fail. When using a Linux Border Router, use ‘`wsbrd_cli status`’ to check the Border Router settings.



The two Wi-SUN devices (Border Router and Wi-SUN SoC Ping) are now part of the same Wi-SUN network.

Ping the Wi-SUN Border Router

To check the commands exposed in the Wi-SUN Ping application, enter:

```
wisun help
```

To retrieve the Border Router IPv6 address, enter:

```
wisun get wisun.ip_address_border_router
```

The Wi-SUN Ping application has a specific command: `wisun ping [IPv6 address]`. Use the command to ping the Border Router.

```
wisun ping [Border Router Global IPv6 address]
```

If the ping command is successful, the pong message size and latency are output on the console.

```
> wisun ping fd00:6172:6d00:0:20d:6fff:fe20:bd95
PING fd00:6172:6d00:0:20d:6fff:fe20:bd95: 40 data bytes
> [40 bytes from fd00:6172:6d00:0:20d:6fff:fe20:bd95: icmp_seq=1 time=196.231 ms]
```

In this case, the ping took 196 milliseconds to come back to the Wi-SUN device. The ping command can be used to communicate with other Wi-SUN devices in the same Wi-SUN network.

Disconnect and Reconnect the Ping WSTK

You need to disconnect the WSTK and reconnect it to apply new network settings, in case you modified them on the Border Router (check these on the Linux Border Router using ‘`wsbrd_cli status`’).

Note: It may be worth going through this in situations where the Wi-SUN device does not connect to the Border Router.

To disconnect the WSTK, enter:

```
wisun disconnect
```

Check the Wi-SUN settings using:

```
wisun get wisun
```

Set the new Wi-SUN settings using:

```
wisun set wisun.<parameter> <value>
```

Use the online help or refer to the readme.md file (at the root of you project) to check what parameters are required for your FAN configuration, using:

```
wisun set wisun help
```

If you want to preserve your settings following a power cycle or reset, use:

```
wisun save
```

Depending on the FAN configuration, use one of the commands below to connect with your new settings:

```
wisun join_fan10
wisun join_fan11

wisun join_explicit (only for the Wi-SUN SoC CLI application)
```

4 Next Steps

Some next steps:

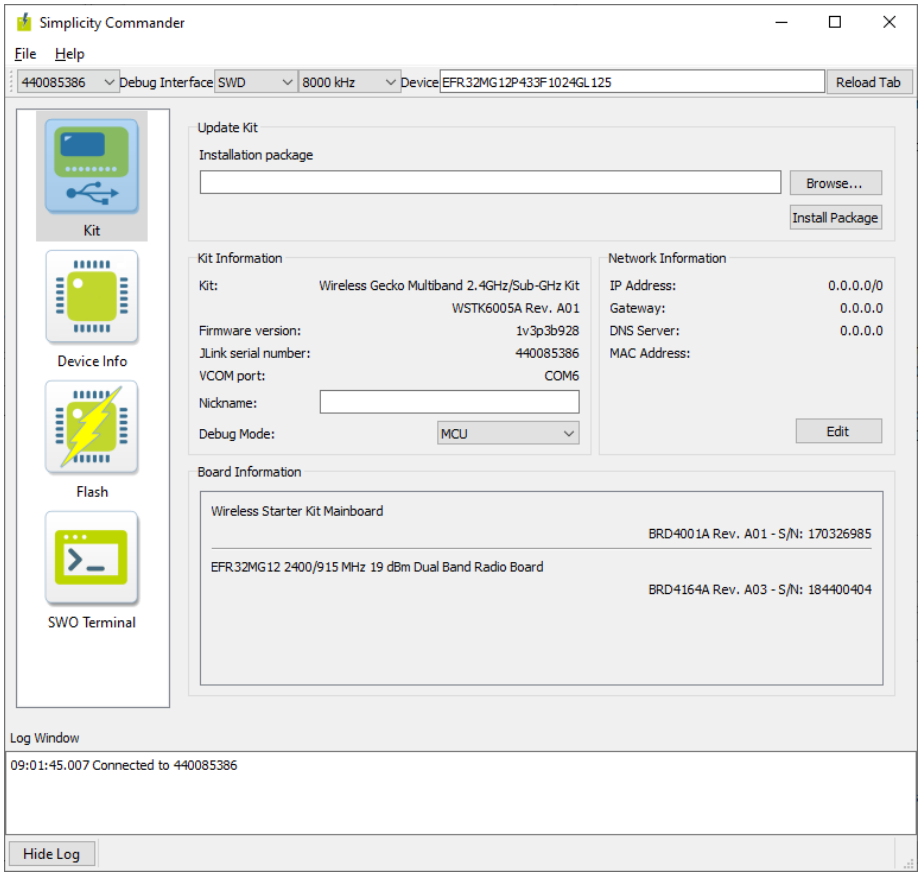
- Explore the functions available through the Wi-SUN CLI example. In the Serial 1 console connected to a device running the example, enter `wisun help` to see a list of commands. Enter `wisun get wisun` to see a list of Wi-SUN network parameters.
- Compile and flash different example applications and explore the functionality they provide.
- Explore configuring one of the example applications to meet your needs.
- Explore the documentation.

Document number	Title
Getting Started	
QSG181	Silicon Labs Wi-SUN Quick-Start Guide
	Silicon Labs Wi-SUN Release Notes
	Gecko Platform Release Notes
Wi-SUN References	
UG495	Silicon Labs Wi-SUN Developer's Guide
	API reference
UG162	Simplicity Commander Reference Guide
Bootloading	
UG103.06	Bootloader Fundamentals
UG266/UG489	Silicon Labs Gecko Bootloader User's Guide for GSDK 3.2 and Lower/4.0 and Higher
Memory Management	
UG103.07	Non-Volatile Data Storage Fundamentals
AN1135	Using Third Generation Non-Volatile Memory (NVM3) Data Storage
Performance	
AN1330	Silicon Labs Wi-SUN Mesh Network Performance

5 Development Tools

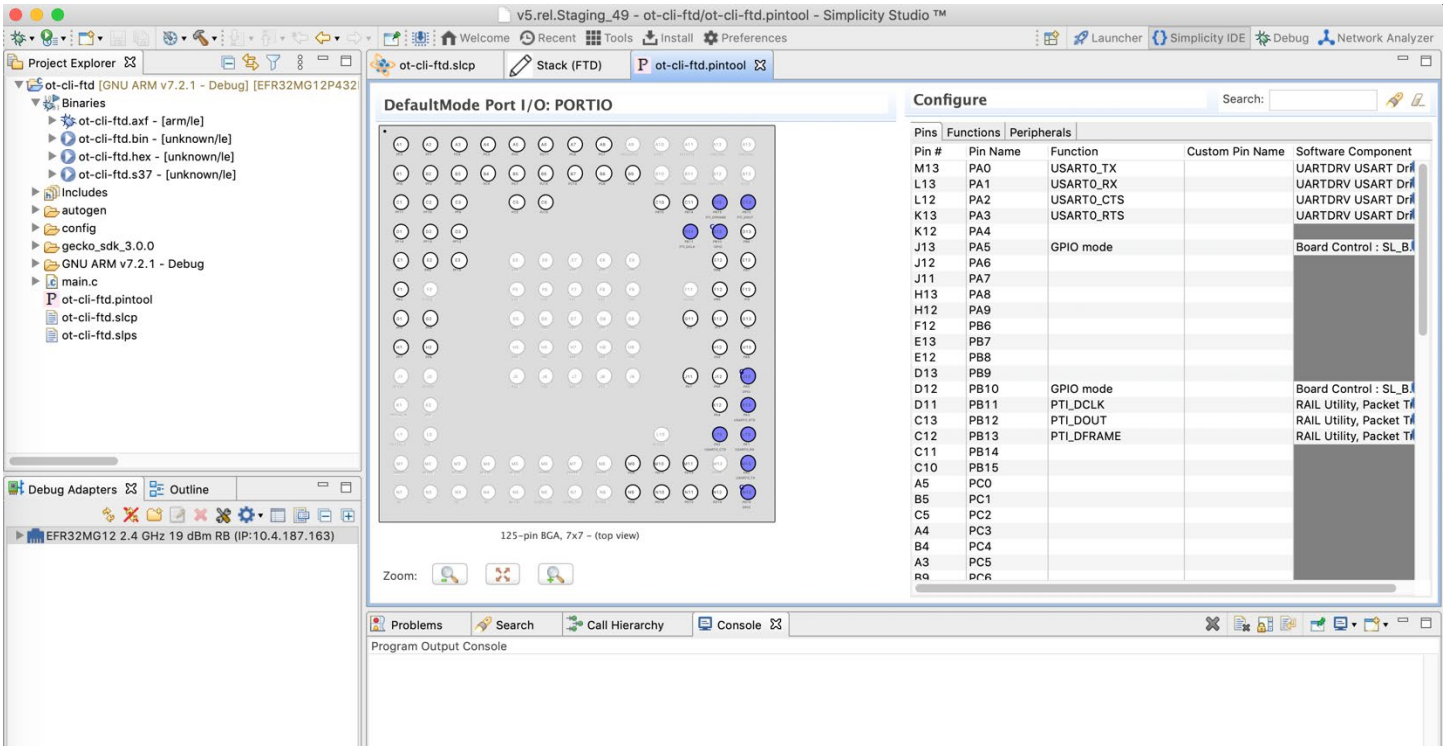
5.1 Simplicity Commander

Simplicity Commander is a simple flashing tool, available through the Simplicity Studio Tools control, which can be used to flash firmware images, erase flash, lock and unlock debug access, and write-protect flash pages via the J-Link interface. Both GUI and CLI (Command Line Interface) are available. See UG162: Simplicity Commander Reference Guide for more information.



5.2 Pin Tool

Simplicity Studio offers a Pin tool that allows you to easily configure new peripherals or change the properties of existing ones. If the project is not open, in the Project Explorer view double-click the <project>.slcp file. In the CONFIGURATION TOOLS tab, click **Open** on the Pin Tool card or double-click the file <project>.pintool in the Project Explorer view. See the Simplicity Studio v5 User's Guide for more information.



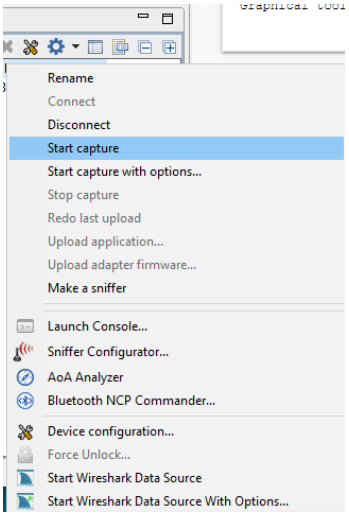
Double-click a Software Component to open the Component Editor and configure that function. Like the Wi-SUN Configurator, Pin Tool does not autosave. If you make changes, the Pin Tool tab has an asterisk to the left. Save changes (CTRL-S) when you are finished updating pin configurations. If you build the project without saving changes, the changes are saved automatically.

5.3 Network Analyzer

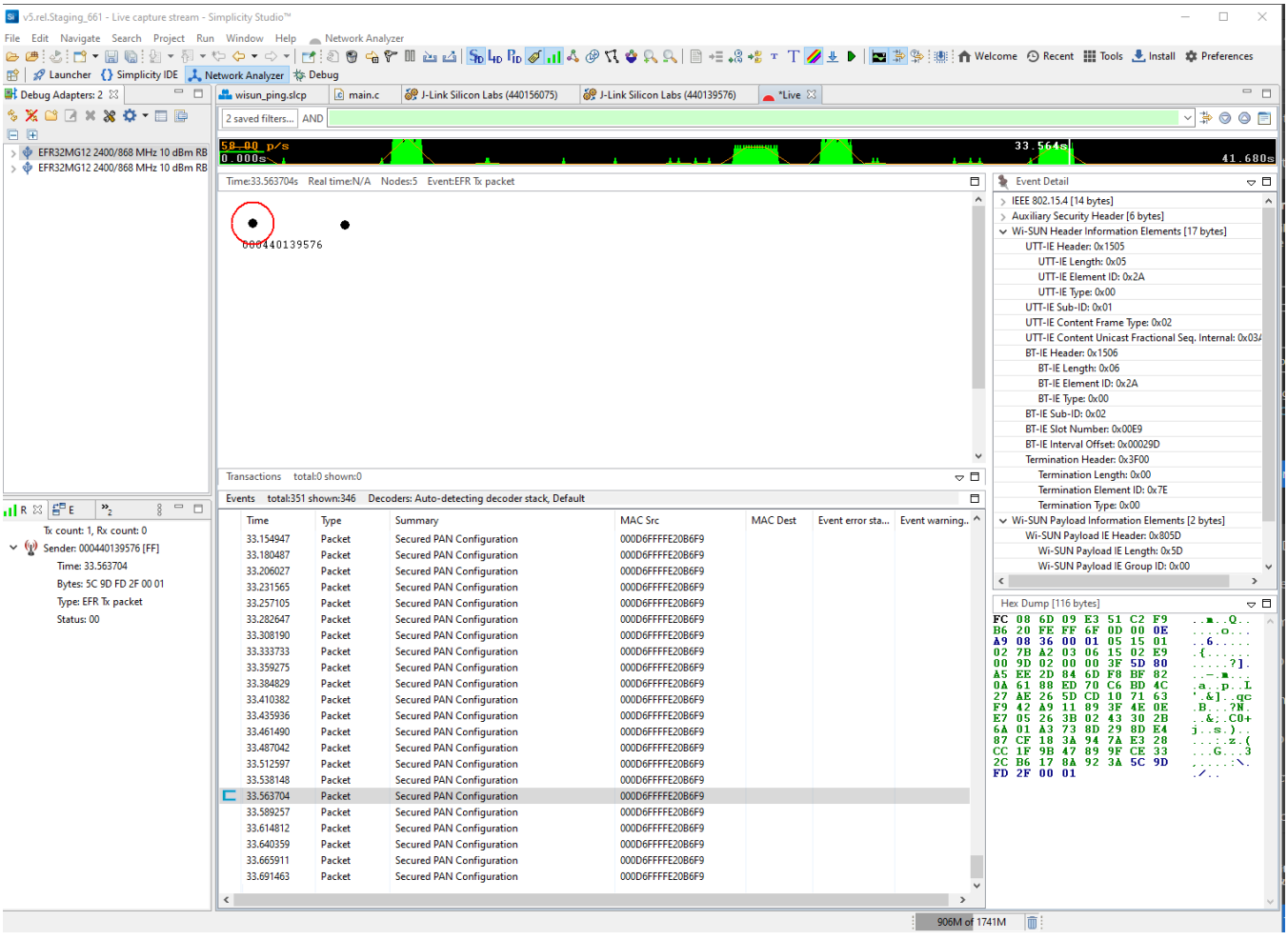
Silicon Labs Network Analyzer is a packet capture and debugging tool that can be used to debug connectivity between wireless nodes running Wi-SUN stack on EFR32 platform. It significantly accelerates the network and application development process with graphical views of network traffic, activity, and duration.

The Packet Trace application captures the packets directly from the Packet Trace Interface (PTI) available on the Wireless Gecko SoCs and modules. It therefore provides a more accurate capture of the packets compared to air-based capture.

To capture Wi-SUN packets using the Silicon Labs Network Analyzer, in the Debug Adapters view, first make sure you are connected and then right-click the device that is running on a Wi-SUN network and select **Start capture**.



You should now be able to see the Wi-SUN traffic as shown below. Click the packets to see more details about its contents in the Event Detail view (on the right).



In the current state, the Network Analyzer does not decrypt the Wi-SUN Upper Layer Application Data. To decrypt the packet and analyze the complete payload including the data, refer to UG495: Silicon Labs Wi-SUN Developer's Guide.

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 LABS

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

www.silabs.com