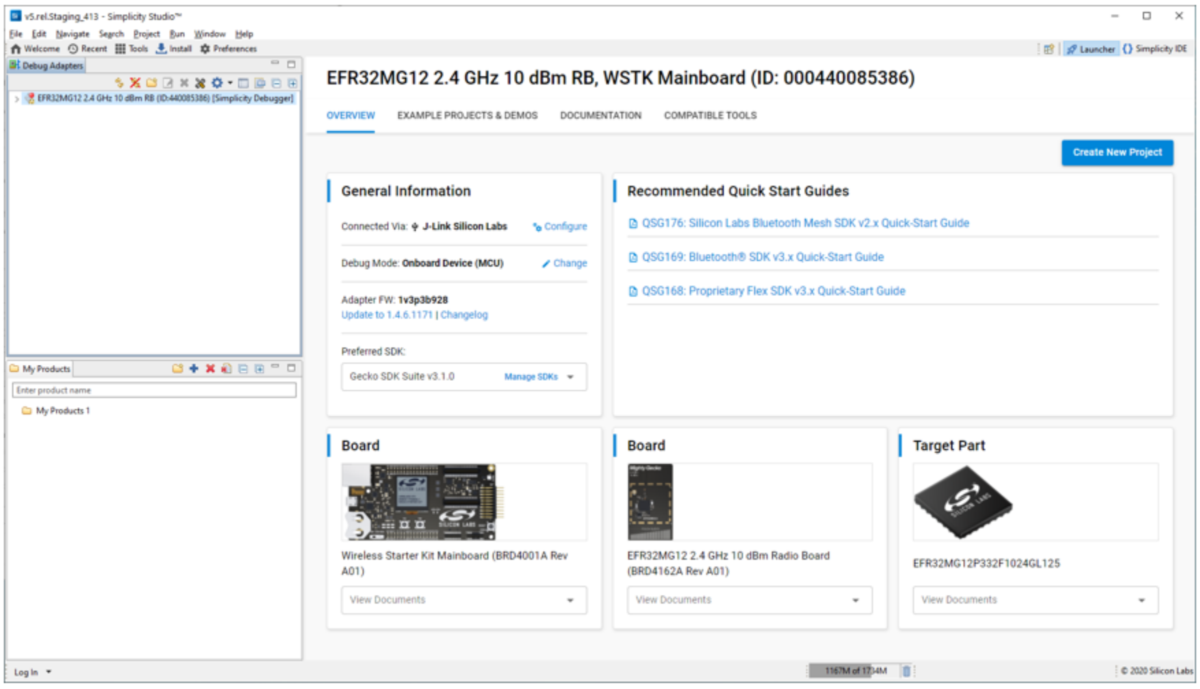
Simplicity Studio 5 Users Guide

Overview

Simplicity Studio® 5 User's Guide

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



EFR32MG12 2.4 GHz 10 dBm RB (ID:440085386) [Simplicity Debugger]

EFR32MG12 2.4 GHz 10 dBm RB, WSTK Mainboard (ID: 000440085386)

General Information

Connected Via: J-Link Silicon Labs

Debug Mode: Onboard Device (MCU)

Adapter FW: 1v3p3b928
Update to 1.4.6.1171 | Changelog

Preferred SDK:
Gecko SDK Suite v3.1.0

Recommended Quick Start Guides

- QSG176: Silicon Labs Bluetooth Mesh SDK v2.x Quick-Start Guide
- QSG169: Bluetooth® SDK v3.x Quick-Start Guide
- QSG168: Proprietary Flex SDK v3.x Quick-Start Guide

Board: Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Board: EFR32MG12 2.4 GHz 10 dBm Radio Board (BRD4162A Rev A01)

Target Part: EFR32MG12P332F1024GL125

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

Simplicity Studio 5 (SSv5) focuses on developer experience, leveraging feedback from customers, employees and competitive reviews. Developers of all experience levels will benefit from an optimized workflow that supports them through the development journey and produces quicker project progression and device configuration.

The *Simplicity Studio 5 User's Guide* pages are organized into the following groups.

- Getting Started describes how to install SSv5 and the relevant development resources, and provides general overviews of using the SSv5 interface and of developing projects in SSv5. If you are new to SSv5, start here.
- About the Launcher is a reference guide to the features and functions available when you first open SSv5. This is a general reference, although some items may not be applicable to all devices.
- About the Simplicity IDE is a reference guide to the features and functions in the Simplicity integrated development environment. This is a general reference, although some items may not be applicable to all devices.
- Developing for 32-Bit Devices provides instructions and reference material for 32-bit device development in the two development environments: Project Configurator and AppBuilder.

- Developing for 8-Bit Devices provides instructions and reference material for 8-bit device development using the Hardware Configurator.
- Building and Flashing describes how to compile and flash images to various device types.
- Companion IDEs: Visual Studio Code discusses how to generate project code in Simplicity Studio 5 to then be used in the VS Code IDE.
- Testing and Debugging outlines tools and strategies for testing your applications on Silicon Labs devices.
- Using the Tools is a reference guide to the various tools packaged with SSv5. Not all tools are applicable to every development path. Their specific use is described in the individual **Developing for** pages.

New Features

New Features

Simplicity Studio® 5 version 5.8.0.0 released December 13, 2023

This release includes the following key features:

- Two new project generators have been officially released.
  - Visual Studio Code Project (utilizing CMake and Ninja)
  - CMake
- Matter Provisioning Tool 1.0.0 (Beta) has been released.
  - Requires GSDK 4.4 and Matter Extension 2.2 or newer
  - Documentation will be coming in a future patch release
- Improvements for computers using MAC M1 Hardware.

Simplicity Studio® 5 version 5.7.0.0 released June 7, 2023

This release includes the following key features:

- Amazon Sidewalk: Can now be installed as an extension directly in Simplicity Studio.
- Amazon Sidewalk: Protocol Configurator released.
- Pin Tool: Now supports Analog pin configuration.

Simplicity Studio® 5 version 5.6.3.0 released March 10, 2023

This release includes the following key features:

- Solutions: Enhanced support for TrustZone applications.
- Post-Build Editor: Removes the requirements for manually editing the yaml output files.

Simplicity Studio® 5 version 5.6.2.0 released February 14, 2023

This release includes the following key features:

- Pin Tool: The user interface is redesigned.
- Bluetooth NCP Commander: Updates in support of Bluetooth mesh functionality.

Simplicity Studio® 5 version 5.6.0.0 released December 14, 2022

This release includes the following key features:

- Project Configuration Report Generator: Provides a report including
  - Target hardware device
  - Target hardware device pin mapping
  - Target SDK
  - Software extensions used in the build
  - Installed application, utility, and platform software components
- Post-Build Editor: Allows for ease of post-build image creation:
  - YAML file is modified to include steps.
  - YAML file can be provided to Simplicity Commander to create an OTA upgrade image.
- Support for Visual Studio Code as a Companion IDE: Basic IDE functionality within VS Code using a Silicon Labs Extension.
  - Build
  - Flash

  - Debug
- Extended Matter Enablement Package support to cover example project configuration, build, and debug.

Simplicity Studio® 5 version 5.5.0.0 released September 7, 2022

This release includes the following key features:

- Introduction of Matter Support:
  - Pre-built Matter demos now supported within the Simplicity Studio Launcher perspective.
  - Matter Enablement Package now available from the Installation Manager.
- Integrated Wireshark Launching from the Debug Adapters view.
- Usability improvements to the Package Manager.

Simplicity Studio® 5 version 5.4.2.0 released August 17, 2022

- Support for the EFR32xG24 Explorer Kit.

Simplicity Studio® 5 version 5.4.0.0 released June 8, 2022

This release includes the following key features:

- Eclipse upgraded to v4.23, featuring several bug fixes and improvements, including improved support for the latest MacOS versions.
- Logic Analyzer view added to Energy Profiler for use with the general-purpose logic analyzer input pins of the Wireless Pro Kit mainboard (e.g. xG24-PK6009A featuring BRD4002A).
- New Wi-SUN Configurator tool available, providing easy graphical configuration of the main Wi-SUN application settings.
- New filters added to Launcher -> Example Projects & Demos view to help users filter by wireless technology, device type, level/difficulty, quality, and more.
- New "Quick Links" view added to the SLC Project Configurator.
- Bluetooth Direction Finding (AoA) tool suite update with new playback and visualization features.
- Support for SDK Extensions as defined by the SLC Specification.
- Improvements to the Linux installation dependencies and scripts.

Simplicity Studio® 5 version 5.3.0.0 released December 15, 2021

This release includes the following key features:

- NCP Commander enhancements
- Bluetooth LE Direction Finding tools suite v1.0
- Installation of GSDKs published on GitHub
- Dynamic GATT Configuration
- IDE Dark Theme improvements
- Multi-Project Solutions

Simplicity Studio® 5 version 5.2.0.0 released June 16, 2021

This release includes the following key features:

- Support for upgrading SLC projects to a newer SDK
- NCP Commander updates:
  - Advertiser improvements including ability to build custom advertisement data packets, fine tune the TX Power, and export to incorporate settings into firmware.
  - Devices discovery improvements: ability to favorite devices and new filter by device name, address, RSSI, raw advertising data or favorite devices.
  - RF Regulatory Tests: ability to add CTE to DTM packets
  - Standalone version with access to all system COM ports may be launched through Tools dialog or Compatible Tools tab (see the Bluetooth NCP Commander page for more information)
- Bluetooth Mesh Configurator updates
  - New warnings if Device Composition Data does not comply with the specification

  - Various bug fixes and improvements to validate model rules
- Upgrade to support integrations with IAR Embedded Workbench v8.50.9
- Upgraded default GCC compiler to version Arm GCC v10.2

Simplicity Studio® 5 version 5.1.0.0 released December 9, 2020

This release includes the following key features:

- Beta release of GNU Debugger (GDB) support
  Simplicity Studio 5.1 includes Beta-level support for a GDB client and SEGGER's GDB server.
- GitHub integration for access to software example repositories
  Starting in Simplicity Studio 5.1, developers may clone sample projects from within the Gecko SDK or within compatible GitHub repos.
- Bluetooth Mesh support
  Starting in Simplicity Studio 5.1 and Gecko SDK 3.1, Bluetooth Mesh is supported in the new Project Configurator workflow. Studio 5.1 introduces the all-new Bluetooth Mesh Configurator tool for configuring a Bluetooth Mesh node's Device Composition Data (DCD).
- Bluetooth NCP Commander
  Bluetooth NCP Commander is a new Simplicity Studio 5 interactive console tool for sending BGAPI commands to a Bluetooth NCP application connected to a development host machine.

Simplicity Studio® 5 version 5.0.0.0 released July 29, 2020

The initial release of SSv5 includes the following key features:

- Launcher
  - Fresh, clean user interface
  - Automatic detection of connected development boards
  - Context-aware development board and target device developer resources
  - SDK download and update manager
  - Easy programming of pre-built demo apps
  - Simple cloning of software examples
  - Quick access to software and hardware documentation with search and filter capabilities
- Software creation & management tools
  - Support for Gecko SDK Suite 3.0 and later
  - Creation of projects for
    - The integrated Simplicity Studio IDE
    - IAR Embedded Workbench
    - Command-line GNU toolchain (GNU makefiles)
  - Searchable library of device-relevant software making it easy to add software components to projects
  - Configure software components in a GUI or text (C source) editor
    - Configurable peripheral initialization, drivers, middleware, kernels and utilities
  - Built-in software configuration error-checking
  - Software dependency manager simplifies porting from
    - SDK to SDK
    - Silicon Labs development kits to custom hardware designs
  - Graphical configuration tools
    - Pin tool to assign pin and peripheral hardware resources
    - Proprietary radio configurator tool
    - Editor for Bluetooth LE Generic Attribute Profile (GATT)
  - Source/Project management and revision control
    - Option to copy or link SDK source
    - No hard-coded paths
    - Clear separation between SDK and customer source code
    - Easy transfer of projects (share/import/export of projects)
    - Source code management and software engineering workflows
- Development Tools
  - Powerful integrated development environment (IDE)
    - Built on latest Eclipse framework (v4.14 and newer)
    - C/C++ Development Tooling (CDT, v9.10 and newer)

        - Eclipse marketplace to enhance and customize
    - Valuable insights with Silicon Labs analysis tools
        - Wireless network traffic capture and analysis
            - Test and analyze the traffic of your wireless devices and networks
            - Data collected directly from devices under test
            - Includes signal strength, LQI (Link Quality Indicator) and filtering information (unique to Silicon Labs)
        - Capture and display energy usage
            - Monitor single device or multiple nodes on a wireless network
            - Code-correlated energy data – identify where your embedded application is consuming energy and design power-optimized applications
            - Advanced trigger and search functions locate power events

Known Issues

Known Issues

Simplicity Studio® Version 5.8.0

ID	Issue	Workaround
758963	The Navigation Buttons (back, forward, etc) will not be displayed if Simplicity Studio starts up on the Simplicity IDE perspective.	Change to a different perspective such as the Launcher perspective and then back to the Simplicity IDE perspective.
1095792	Changing the Target Part in Project Configurator does not work correctly on customer hardware.	Either create a new software example for the desired target part and port in code changes or manually change the component ID in the .slcp and .slps files with a text editor and then open the Project configurator and do a Force Generation.
1113960	GDB Debug Session Launch fails with "Error in final launch sequence" popup.	Typically, the debug session will launch successfully the next time or after a Simplicity Studio restart. If the issue persists, edit the debug configuration **Run > Debug configurations...** and uncheck the 'Continue' box on the Startup tab. The debug session will stop before `main()`, but clicking **Resume** will run to `main()`.
1156776	Custom Network Analyzer Decoders no longer work starting with version 5.7.0.0.	JavaScript custom decoders need to be converted to LUA decoders. See Custom Decoders.
1209674	IAR Toolchain linker command line length can exceed Windows line limit.	When the IAR toolchain is used to compile projects from the Simplicity Studio IDE under Windows, it is possible to get linker errors if the linker line length exceeds the Windows limit of 30,000 characters. The work around is to edit the project properties [C/C++ Build] > [Settings] > [IAR Linker for ARM] [Expert settings] and to use this command line: `@echo "$(OBJS) $(USER_OBJS)" > linker_file.xcl; ${COMMAND} -f linker_file.xcl $(LIBS) ${OUTPUT_FLAG}${OUTPUT_PREFIX}${OUTPUT} ${FLAGS}`
1225683	Blank Project Wizard or Log In windows seen with MacOS Ventura or later.	When running Simplicity Studio under MacOS Ventura or later, if blank windows are seen for the new project wizard, login page or other windows, the internal browser used by Simplicity Studio might not have permissions to make changes to the system. The work around is to go to the MacOS System Settings -> Privacy & Security -> App Management and to add Simplicity Studio as an application that can update or delete other applications.

For Users of Previous Versions

For Users of Previous Versions

If you are migrating from Simplicity Studio 4 to SSv5, or if you are transitioning from AppBuilder projects to the component-based architecture, see Project Migration for additional information.

Getting Started

Getting Started

To get started with Simplicity Studio® 5 (SSv5):

1. Install SSv5 and development software
2. Explore the main features of SSv5
3. Start a project

You do not need hardware to install SSv5 and the relevant software packages or, for 32-bit devices, to explore the project configuration interface for a particular SDK. However, having your target hardware connected during installation ensures that your SSv5 installation is configured precisely for your environment.

You should also have an account set up in the Silicon Labs Customer Support Portal. Access to some software packages is controlled by your customer profile.

See Prerequisites for more information.

Prerequisites

Prerequisites

To streamline your Simplicity Studio® 5 (SSv5) installation:

- Connect development hardware to your PC
- Log in to your customer account

SSv5 System Requirements

Operating Systems

Operating System	Version
Windows	Windows 10 (64-bit), Windows 11
macOS	10.14 Mojave, 10.15 Catalina, 11.x Big Sur, 12.x Monterey
Linux	Ubuntu 20.04 LTS

Hardware

Hardware Component	Item
CPU	1 GHZ or better
Memory	1 GB RAM minimum, 8 GB recommended for Wireless Protocol development
Disk Space	600 MB disk space for minimum FFD installation; 7 GB for Wireless Dynamic Protocol support

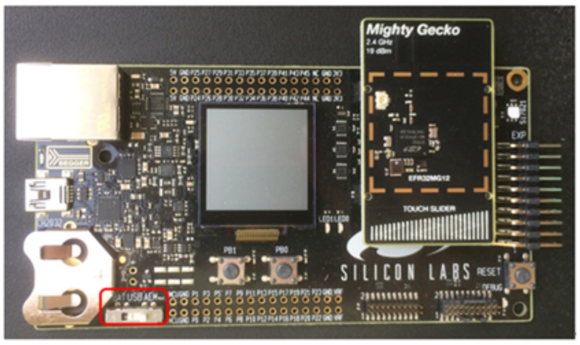
Hardware

A Silicon Labs development kit is not required to get started with Simplicity Studio. However, if Simplicity Studio detects an official development kit, it can be used to help select the appropriate software and tools for installation. If you don't have a development kit, you can explore the available options here.

EFR32 Kits

If you have a Silicon Labs Wireless Starter Kit (WSTK) or Pro Kit, mount a radio board on the mainboard and connect the mainboard to your PC using the USB cable supplied with the kit.

Note: For best performance in SSv5, be sure that the power switch on the mainboard is in the Advanced Energy Monitoring or “AEM” position.



Customer Account

If you do not already have one, create an account on the Silicon Labs Customer Support portal. Be sure to record your account username and password as you will use it to log in through SSv5. Your account properties determine not only what software components you can download, but also what notifications you will receive.

To review or change your notification 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. Silicon Labs strongly recommends that you receive these notifications. Click **Save** to save any changes.

Install SSv5 and Software

Install SSv5 and Software

This section discusses how to install SSv5 and, once SSv5 is installed, how to install software.

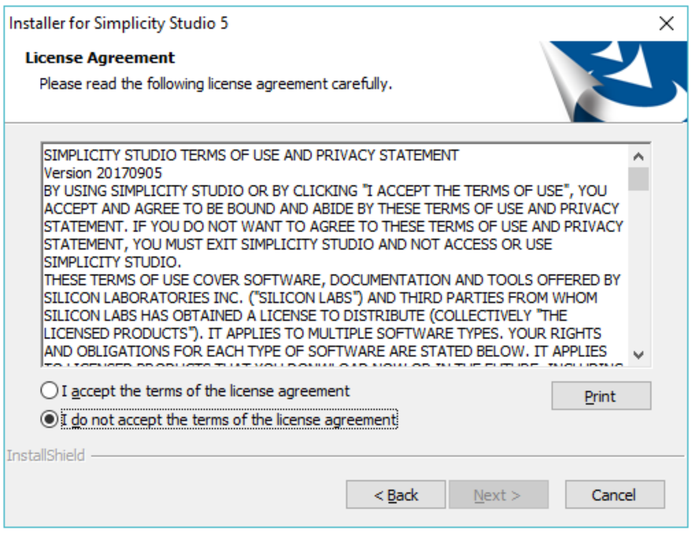
NOTE: Depending on your corporate environment, you may encounter installation problems. For example, if you cannot log in to Simplicity Studio® 5 (SSv5) but know you are entering your Customer Account username and password correctly, or if you get an error about connecting to the update server and your computer is definitely connected to the Internet, see SDK installation or updates are not working for possible causes and solutions.

Install SSv5

Instructions for installation on Windows and MacOS and on Linux are provided.

On Windows and MacOS

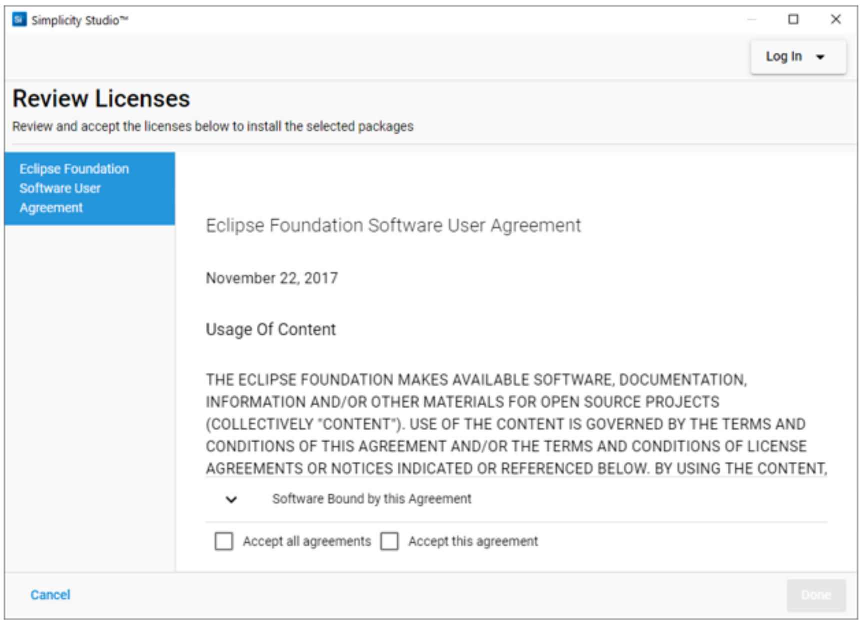
1. Download the SSv5 installation package from the Silicon Labs website. The Windows package is an '.iso' disk image. After the package finishes downloading, double-click it to mount the iso image as a drive and then double-click the setup.exe file inside the drive to launch the installer.
2. When the SSv5 installer first launches, it presents a Simplicity Studio License Agreement dialog. Accept the terms of the agreement and click **Next >**.


Installer for Simplicity Studio 5
License Agreement
Please read the following license agreement carefully.

SIMPLICITY STUDIO TERMS OF USE AND PRIVACY STATEMENT
Version 20170905
BY USING SIMPLICITY STUDIO OR BY CLICKING "I ACCEPT THE TERMS OF USE", YOU ACCEPT AND AGREE TO BE BOUND AND ABIDE BY THESE TERMS OF USE AND PRIVACY STATEMENT. IF YOU DO NOT WANT TO AGREE TO THESE TERMS OF USE AND PRIVACY STATEMENT, YOU MUST EXIT SIMPLICITY STUDIO AND NOT ACCESS OR USE SIMPLICITY STUDIO.
THESE TERMS OF USE COVER SOFTWARE, DOCUMENTATION AND TOOLS OFFERED BY SILICON LABORATORIES INC. ("SILICON LABS") AND THIRD PARTIES FROM WHOM SILICON LABS HAS OBTAINED A LICENSE TO DISTRIBUTE (COLLECTIVELY "THE LICENSED PRODUCTS"). IT APPLIES TO MULTIPLE SOFTWARE TYPES. YOUR RIGHTS AND OBLIGATIONS FOR EACH TYPE OF SOFTWARE ARE STATED BELOW. IT APPLIES

I accept the terms of the license agreement [✓ I do not accept the terms of the license agreement]

3. Choose a destination location, click **Next >**, and then click **Install**.
4. When SSv5 launches, you are presented with one or more additional license agreements. You can accept them individually or all at once. Click **DONE** when finished.



Review Licenses

Review and accept the licenses below to install the selected packages

Eclipse Foundation Software User Agreement

Eclipse Foundation Software User Agreement

November 22, 2017

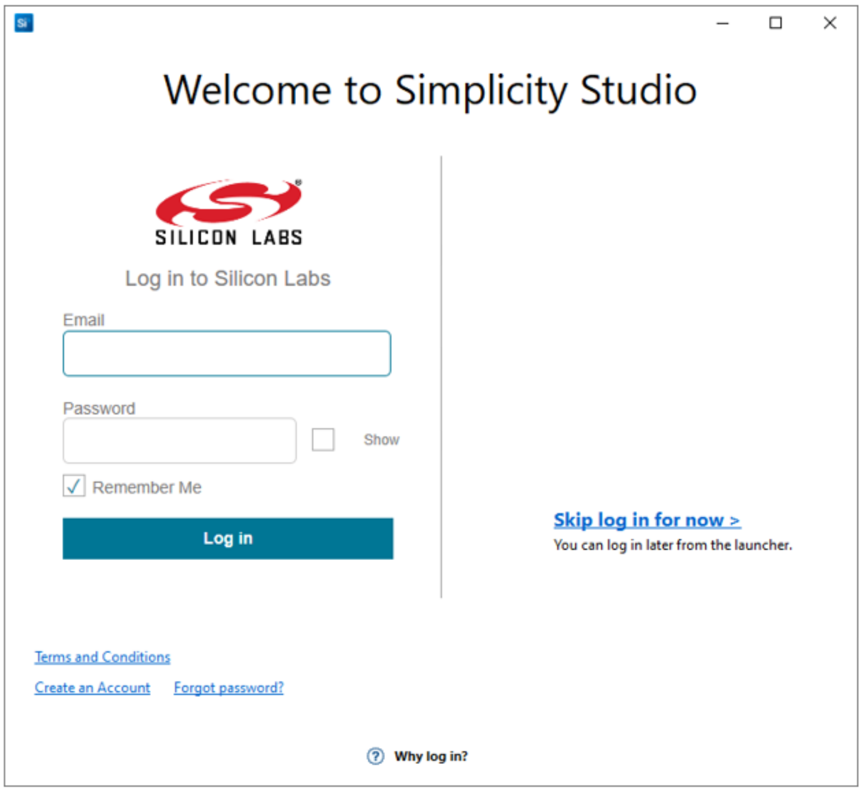
Usage Of Content

THE ECLIPSE FOUNDATION MAKES AVAILABLE SOFTWARE, DOCUMENTATION, INFORMATION AND/OR OTHER MATERIALS FOR OPEN SOURCE PROJECTS (COLLECTIVELY "CONTENT"). USE OF THE CONTENT IS GOVERNED BY THE TERMS AND CONDITIONS OF THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. BY USING THE CONTENT,

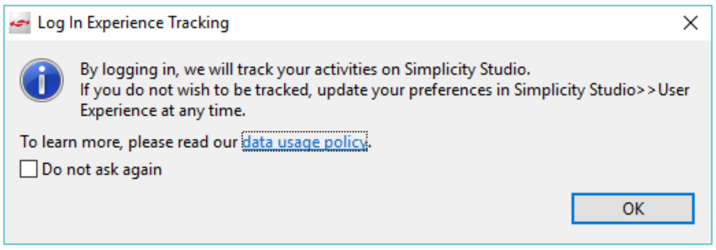
Software Bound by this Agreement

☐ Accept all agreements ☐ Accept this agreement

5. Next, you are invited to log in. Log in using your Silicon Labs account username and password. **NOTE**: Although you can skip log in here, you must be logged in to access some restricted access software packages.



6. Before login completes, you must agree to Experience Tracking. You can change the status later in SSv5 Preferences >> Simplicity Studio >> User Experience.

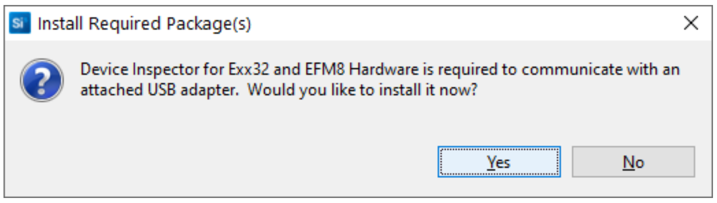


Log In Experience Tracking

By logging in, we will track your activities on Simplicity Studio.
If you do not wish to be tracked, update your preferences in Simplicity Studio>>User Experience at any time.

To learn more, please read our data usage policy.

7. If you have hardware connected, depending on your operating system and security settings, you may need to allow SSv5 to make changes to your system. SSv5 prompts you to install the Device Inspector. Click **Yes**.



Install Required Package(s)

Device Inspector for Exx32 and EFM8 Hardware is required to communicate with an attached USB adapter. Would you like to install it now?

On Linux

Simplicity Studio for Linux® OS is only officially supported on Ubuntu® LTS distributions. Simplicity Studio might work on other distributions, but it is only tested and verified to work with Ubuntu® LTS.

The base Simplicity Studio platform (not an installer) is delivered in a tar file that should be expanded into the desired installation directory. The recommended installation directory is the user's root directory (`/home/USERNAME/SimplicityStudio_v5`). This directory must be one that can be updated without root permissions.

Before launching Simplicity Studio the first time, execute the following:

```
sudo apt-get update
sudo apt-get upgrade
cd SimplicityStudio_v5
sudo ./setup.sh
```

See Notices for more information on the setup.sh script.

When the Simplicity Studio executable 'studio' is launched, it starts a first-time installation process to add support either by a connected debug adapter or by technology type.

Notices

setup.sh will:

1. Attempt to install dev rules for USB device connectivity
2. Install missing packages (libraries) through your system's package manager.

setup.sh will only install required external Linux® packages on Ubuntu® distributions. For other distributions, the setup.sh script should be examined for a list of possible packages that will need to be installed for Simplicity Studio to work.

If **the Wayland display protocol** is being used, start Simplicity Studio from a terminal window with `./studiowayland.sh`.

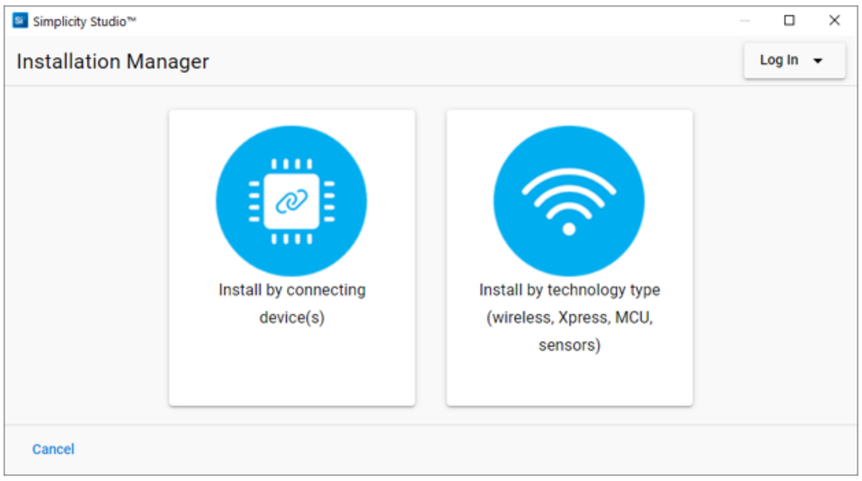
JxBrowser: Simplicity Studio uses the third party product JxBrowser to render several windows, including the installation window, Launcher perspective and Project Configurator windows. By default JxBrowser expands the Chromium browser into the /tmp folder. If this is not a good option because of file permissions of the /tmp folder, the location can be changed by adding a line to the studio.ini file specifying a different path such as: `-Djxbrowser.chromium.dir=/home/USERNAME/.jxbrowser`

Install Software

After SSv5 installation is complete, you are prompted to install software packages, such as the Gecko SDK (GSDK). SSv5 offers two options:

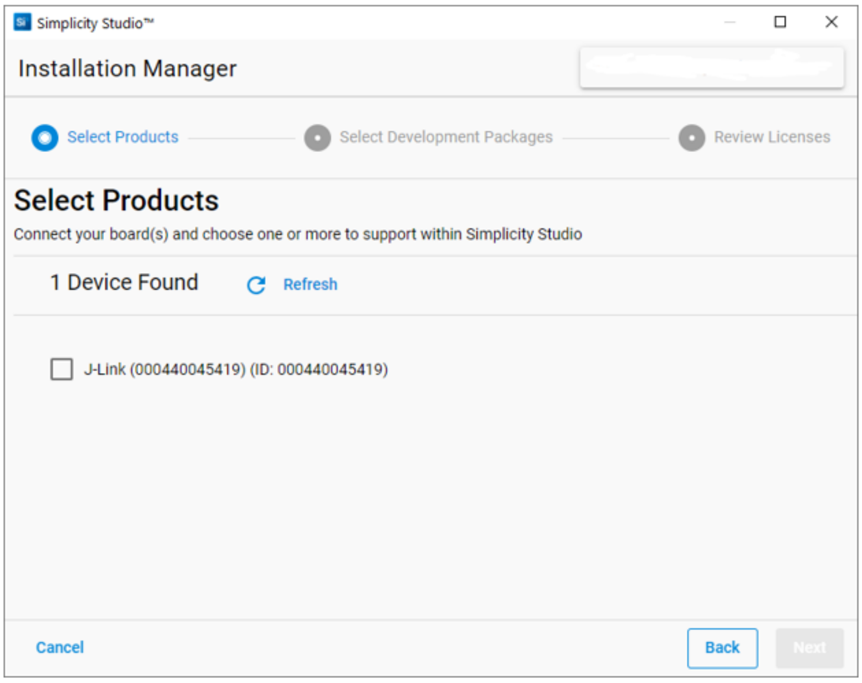
- Install by Connecting Device(s)
- Install by Technology Type

You do not need to have a target device connected to install by Technology Type.


Simplicity Studio™
Installation Manager
Log In
Install by connecting device(s)
Install by technology type (wireless, Xpress, MCU, sensors)
Cancel

Install Software by Connecting Devices

1. Make sure you have a device connected and click **Install by Connecting Devices**. The connected devices are shown.



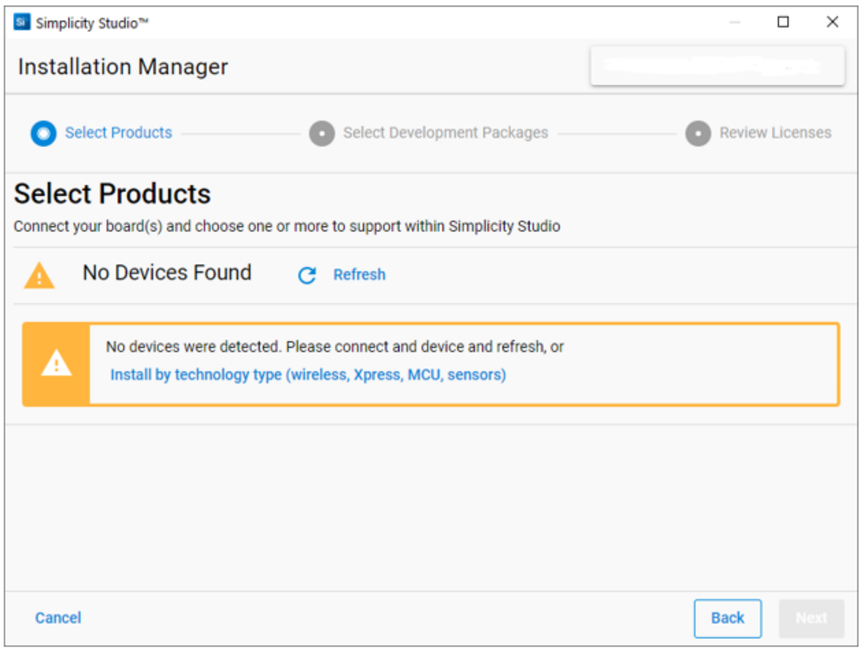
Select Products

Connect your board(s) and choose one or more to support within Simplicity Studio

1 Device Found

J-Link (000440045419) (ID: 000440045419)

If you do not see a connected device, or if you connect a device after this dialog is presented, click **Refresh**. If you do not have a device connected you cannot continue.



Select Products

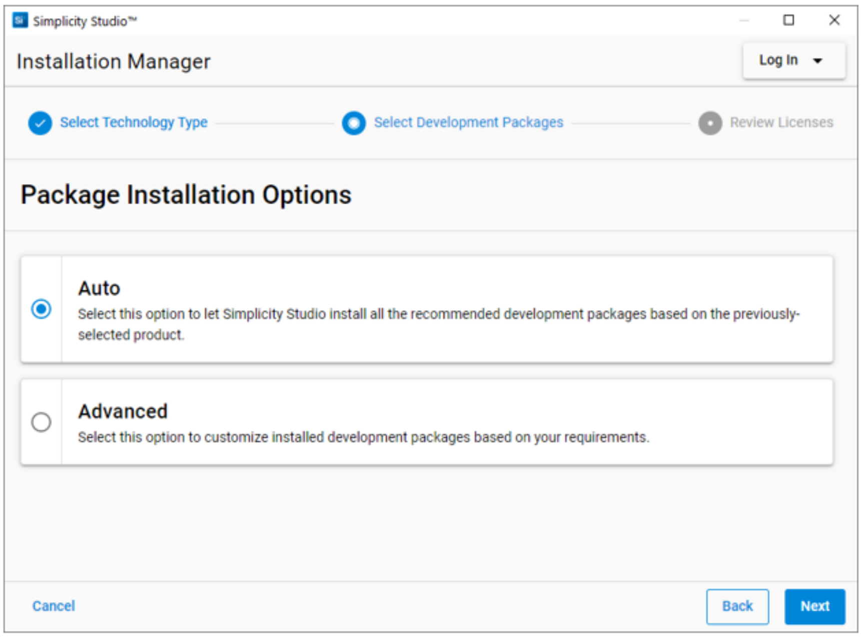
Connect your board(s) and choose one or more to support within Simplicity Studio

No Devices Found   Refresh

No devices were detected. Please connect and device and refresh, or

Install by technology type (wireless, Xpress, MCU, sensors)

2. Select the device(s) to use. If you select more than one device with different software compatibilities, Studio will download any package that is compatible with any of the devices. Click **NEXT**.
3. You have two installation options, **Auto** and **Advanced**.



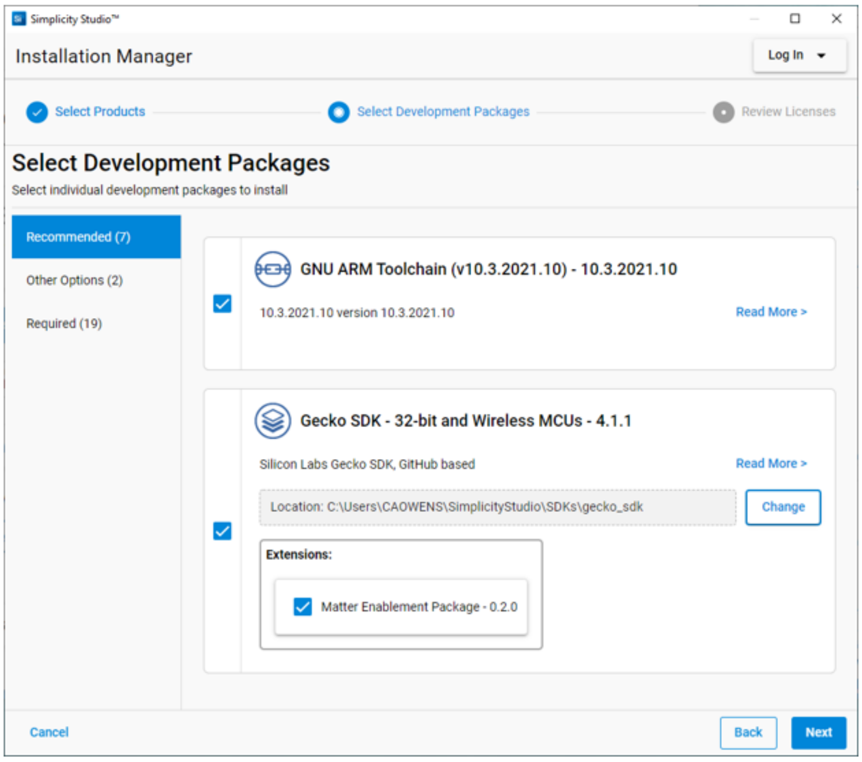
Package Installation Options

[✓ Auto] Select this option to let Simplicity Studio install all the recommended development packages based on the previously-selected product.

Advanced: Select this option to customize installed development packages based on your requirements.

   - Select **Auto** to have SSv5 automatically download all packages that are compatible with the connected hardware and to which you have permissions based on your login.
   - Select **Advanced** to select which packages you want to install or to specify an installation location for the Gecko SDK.

If you selected **Advanced**, all 'Required' packages must be installed. **NOTE**: Do not exclude packages unless you understand the full effect of the decision. Click **Read More** to see release notes for each item. Optionally change the default installation location for the Gecko SDK (GSDK). The GSDK installs as a clone of a GitHub repository. Do not select an existing folder with other content already in it, as your cloned repository will no longer be clean. The Matter Enablement Package is automatically selected when you select the GSDK.



Select Development Packages

Select individual development packages to install

☑ GNU ARM Toolchain (v10.3.2021.10) - 10.3.2021.10

10.3.2021.10 version 10.3.2021.10

☑ Gecko SDK - 32-bit and Wireless MCUs - 4.1.1

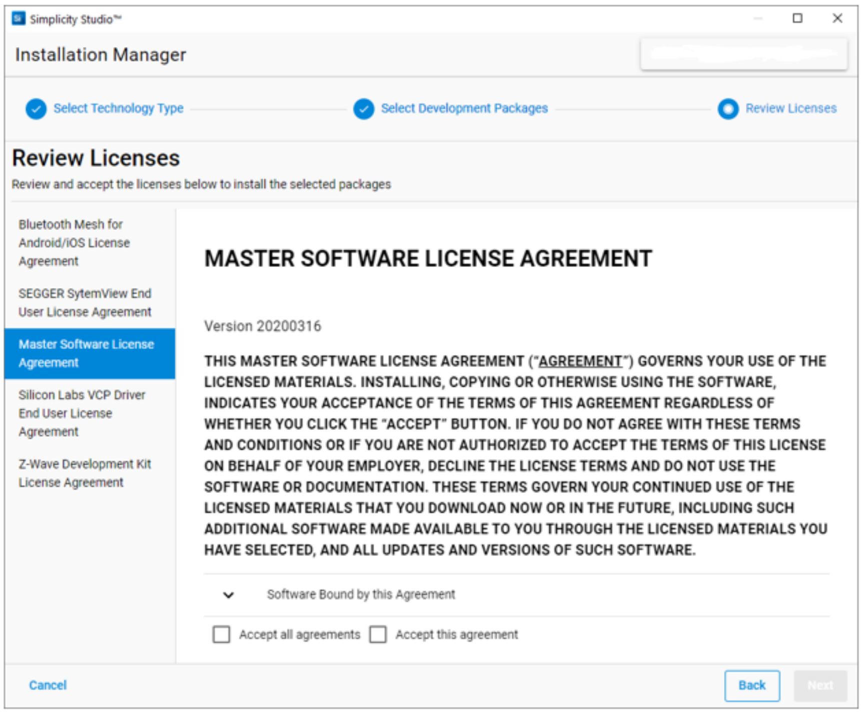
Silicon Labs Gecko SDK, GitHub based

Location: C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk

Extensions:

☑ Matter Enablement Package - 0.2.0

4. Before package installation begins, accept one or more software license agreements.


Simplicity Studio™
Installation Manager
Select Technology Type
Select Development Packages
Review Licenses

Review Licenses

Review and accept the licenses below to install the selected packages

Bluetooth Mesh for Android/iOS License Agreement
SEGGER SytemView End User License Agreement
Master Software License Agreement
Silicon Labs VCP Driver End User License Agreement
Z-Wave Development Kit License Agreement

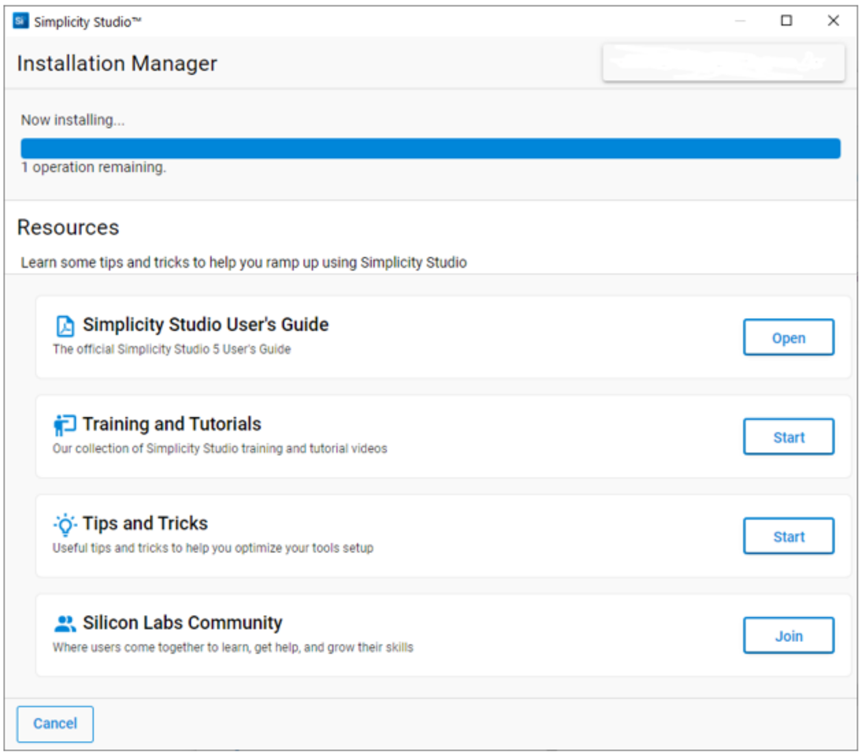
MASTER SOFTWARE LICENSE AGREEMENT

Version 20200316

THIS MASTER SOFTWARE LICENSE AGREEMENT ("AGREEMENT") GOVERNS YOUR USE OF THE LICENSED MATERIALS. INSTALLING, COPYING OR OTHERWISE USING THE SOFTWARE, INDICATES YOUR ACCEPTANCE OF THE TERMS OF THIS AGREEMENT REGARDLESS OF WHETHER YOU CLICK THE "ACCEPT" BUTTON. IF YOU DO NOT AGREE WITH THESE TERMS AND CONDITIONS OR IF YOU ARE NOT AUTHORIZED TO ACCEPT THE TERMS OF THIS LICENSE ON BEHALF OF YOUR EMPLOYER, DECLINE THE LICENSE TERMS AND DO NOT USE THE SOFTWARE OR DOCUMENTATION. THESE TERMS GOVERN YOUR CONTINUED USE OF THE LICENSED MATERIALS THAT YOU DOWNLOAD NOW OR IN THE FUTURE, INCLUDING SUCH ADDITIONAL SOFTWARE MADE AVAILABLE TO YOU THROUGH THE LICENSED MATERIALS YOU HAVE SELECTED, AND ALL UPDATES AND VERSIONS OF SUCH SOFTWARE.

Software Bound by this Agreement
☐ Accept all agreements ☐ Accept this agreement
Cancel
Back
Next

5. Once you click **NEXT**, installation begins.


Simplicity Studio™
Installation Manager

Now installing...

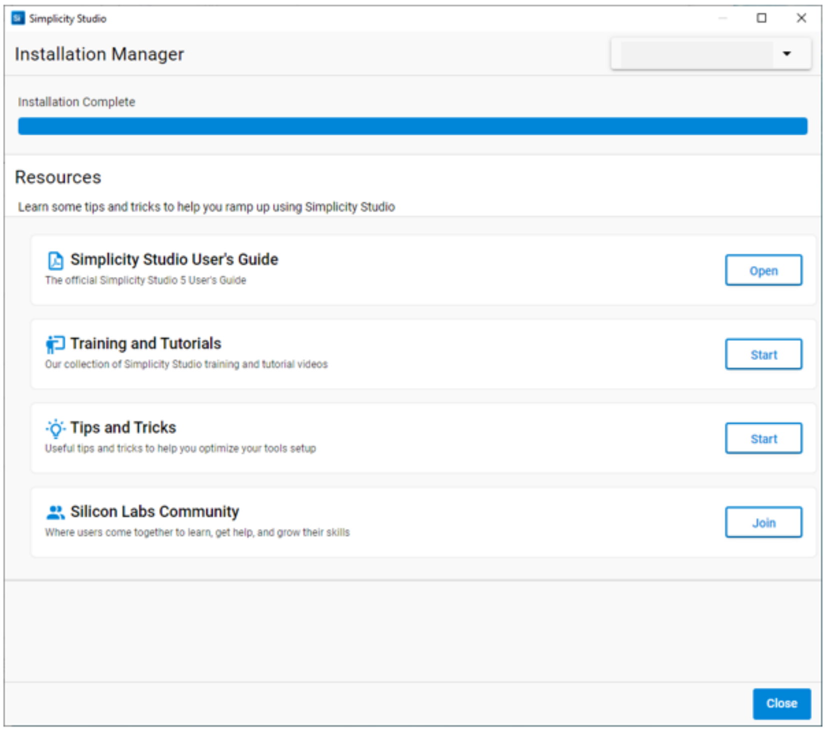
1 operation remaining.

Learn some tips and tricks to help you ramp up using Simplicity Studio

Simplicity Studio User's Guide
The official Simplicity Studio 5 User's Guide
Open
Training and Tutorials
Our collection of Simplicity Studio training and tutorial videos
Start
Tips and Tricks
Useful tips and tricks to help you optimize your tools setup
Start
Silicon Labs Community
Where users come together to learn, get help, and grow their skills
Join
Cancel

Depending upon your selections, installation may take some time. The progress bar may pause at points where large files are being installed, but will resume as they complete. If there is a problem during installation, SSv5 displays an error. You must restart SSv5 to resume.

6. When installation is complete, click **CLOSE**.

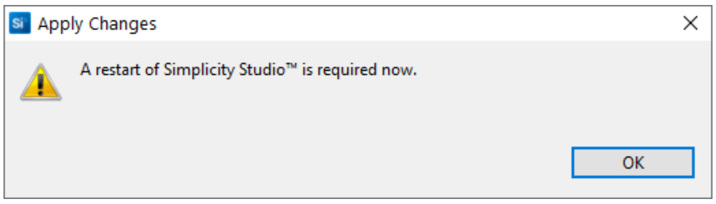

Simplicity Studio
Installation Manager

Installation Complete

Learn some tips and tricks to help you ramp up using Simplicity Studio

Simplicity Studio User's Guide
The official Simplicity Studio 5 User's Guide
Open
Training and Tutorials
Our collection of Simplicity Studio training and tutorial videos
Start
Tips and Tricks
Useful tips and tricks to help you optimize your tools setup
Start
Silicon Labs Community
Where users come together to learn, get help, and grow their skills
Join
Close

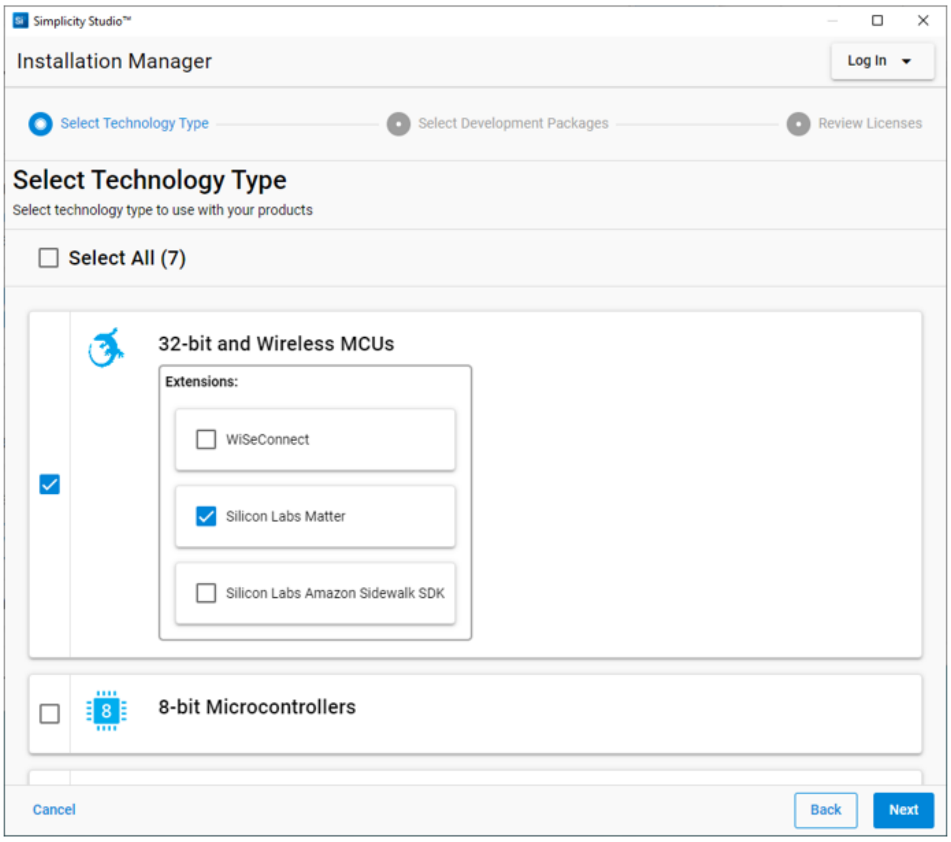
7. Click **RESTART** to begin working with projects.



A restart of Simplicity Studio™ is required now.

Install Software by Technology Type

1. Click **Install by Technology Type**
2. Select the technology(s) to install and click **NEXT**. To install the Gecko SDK Suite, select **32-Bit and Wireless MCUs**. The Matter Enablement Package is automatically selected when you select the GSDK.

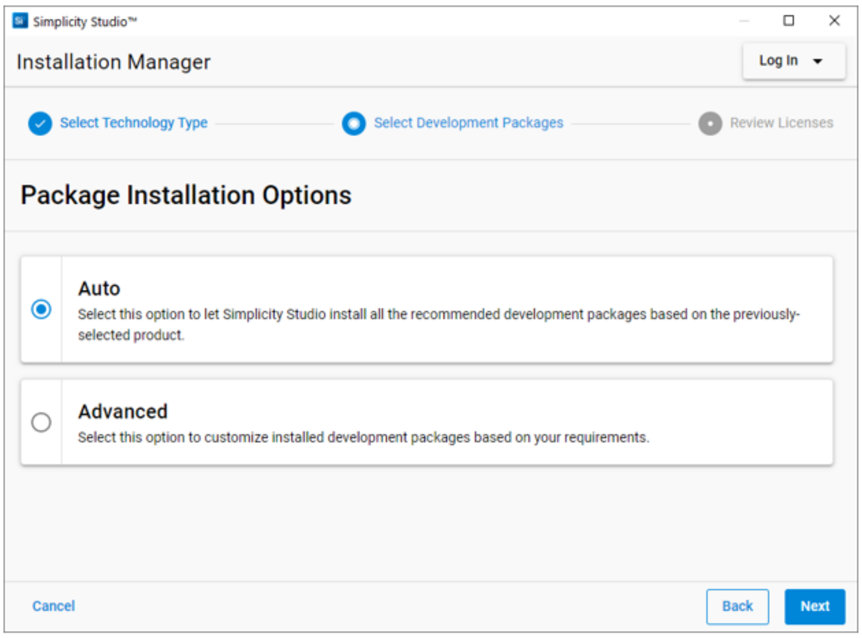


Select Technology Type

Select technology type to use with your products

- [ ] Select All (7)
- [✓] 32-bit and Wireless MCUs
  - Extensions:
    - [ ] WiSeConnect
    - [✓] Silicon Labs Matter
    - [ ] Silicon Labs Amazon Sidewalk SDK
- [ ] 8-bit Microcontrollers

3. You have two installation options, **Auto** and **Advanced**.


Simplicity Studio™

Installation Manager

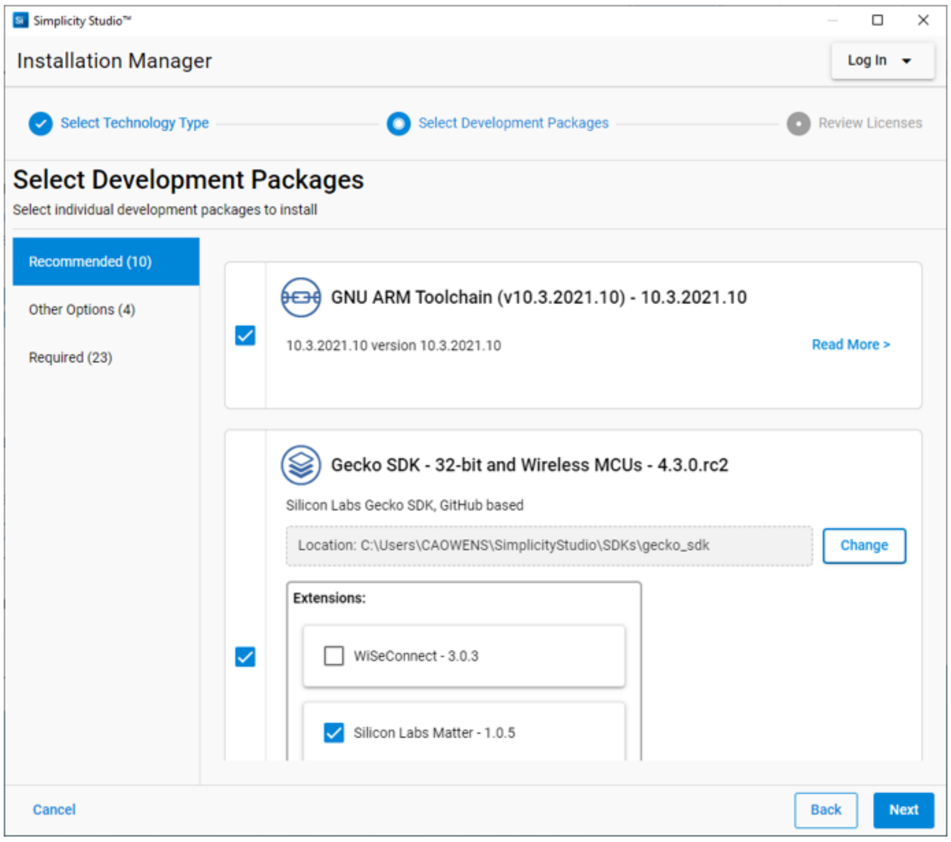
Package Installation Options

[✓ Auto] Select this option to let Simplicity Studio install all the recommended development packages based on the previously-selected product.

[ ] Advanced Select this option to customize installed development packages based on your requirements.

- Select **Auto** to have SSv5 automatically download all packages related to the selected technologies and to which you have permissions based on your login.
- Select **Advanced** to select which packages you want to install.

If you selected **Advanced**, all 'Required' packages must be installed. **NOTE**: Do not exclude packages unless you understand the full effect of the decision. Click **Read More** to see release notes for each item. Optionally change the default installation location for the Gecko SDK. The GSDK installs as a clone of a GitHub repository. Do not select an existing folder with other content already in it, as your cloned repository will no longer be clean. The Matter Enablement Package is automatically selected when you select the GSDK.


Simplicity Studio™

Installation Manager

Select Development Packages

Select individual development packages to install

[✓] GNU ARM Toolchain (v10.3.2021.10) - 10.3.2021.10

10.3.2021.10 version 10.3.2021.10 — Read More >

[✓] Gecko SDK - 32-bit and Wireless MCUs - 4.3.0.rc2

Silicon Labs Gecko SDK, GitHub based

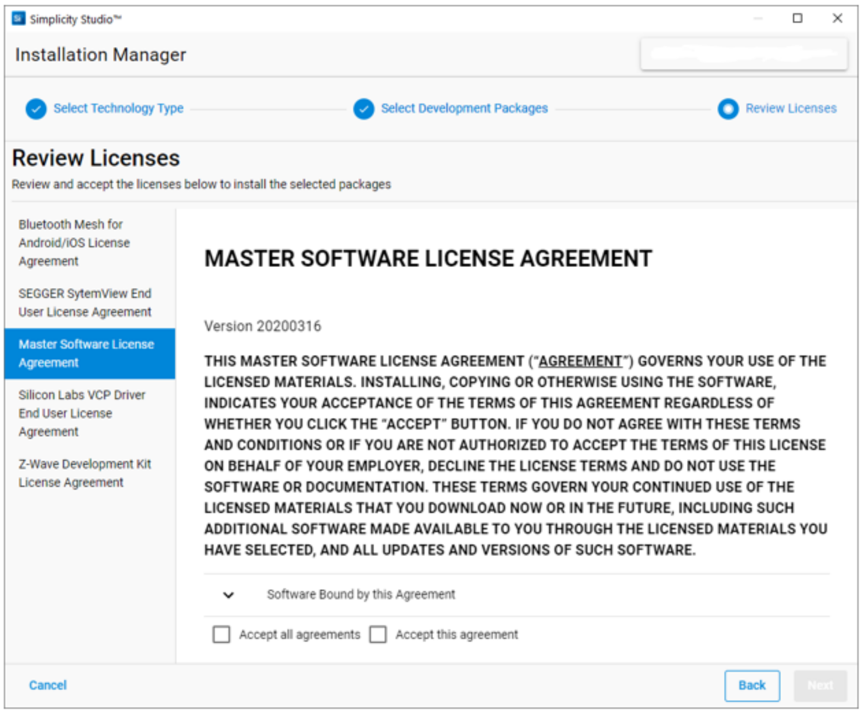
Location: C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk — Change

Extensions:

[ ] WiSeConnect - 3.0.3

[✓] Silicon Labs Matter - 1.0.5

4. Before installation begins, accept one or more software license agreements.



Simplicity Studio™

Installation Manager

Select Technology Type — Select Development Packages — Review Licenses

Review Licenses

Review and accept the licenses below to install the selected packages

- Bluetooth Mesh for Android/iOS License Agreement
- SEGGER SytemView End User License Agreement
- Master Software License Agreement
- Silicon Labs VCP Driver End User License Agreement
- Z-Wave Development Kit License Agreement

MASTER SOFTWARE LICENSE AGREEMENT

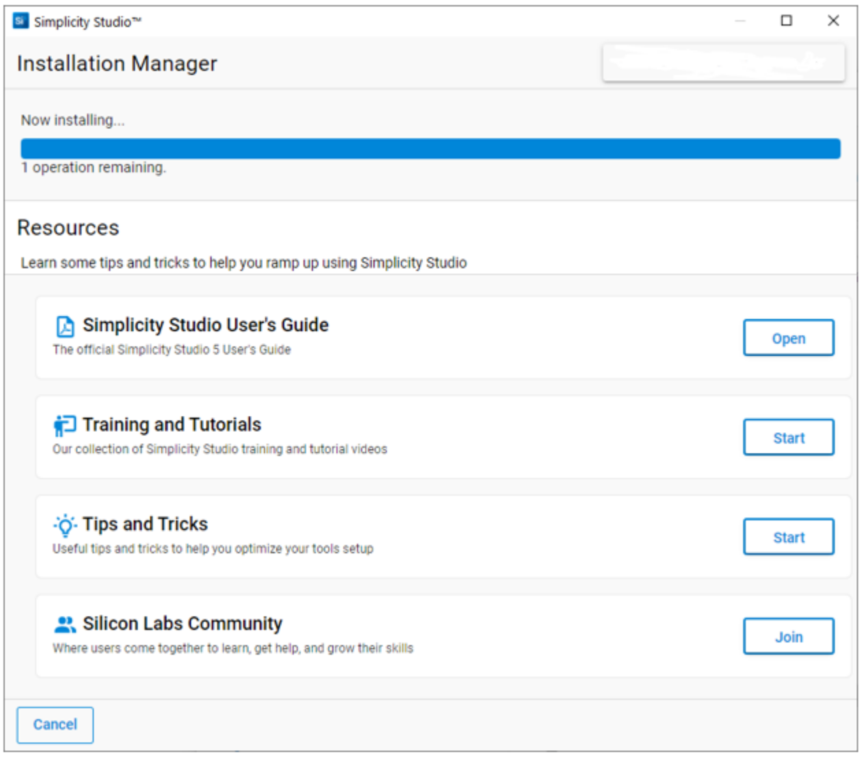
Version 20200316

THIS MASTER SOFTWARE LICENSE AGREEMENT ("AGREEMENT") GOVERNS YOUR USE OF THE LICENSED MATERIALS. INSTALLING, COPYING OR OTHERWISE USING THE SOFTWARE, INDICATES YOUR ACCEPTANCE OF THE TERMS OF THIS AGREEMENT REGARDLESS OF WHETHER YOU CLICK THE "ACCEPT" BUTTON. IF YOU DO NOT AGREE WITH THESE TERMS AND CONDITIONS OR IF YOU ARE NOT AUTHORIZED TO ACCEPT THE TERMS OF THIS LICENSE ON BEHALF OF YOUR EMPLOYER, DECLINE THE LICENSE TERMS AND DO NOT USE THE SOFTWARE OR DOCUMENTATION. THESE TERMS GOVERN YOUR CONTINUED USE OF THE LICENSED MATERIALS THAT YOU DOWNLOAD NOW OR IN THE FUTURE, INCLUDING SUCH ADDITIONAL SOFTWARE MADE AVAILABLE TO YOU THROUGH THE LICENSED MATERIALS YOU HAVE SELECTED, AND ALL UPDATES AND VERSIONS OF SUCH SOFTWARE.

Software Bound by this Agreement

☐ Accept all agreements ☐ Accept this agreement

5. Once you click **NEXT**, installation begins.



Simplicity Studio™

Installation Manager

Now installing...

1 operation remaining.

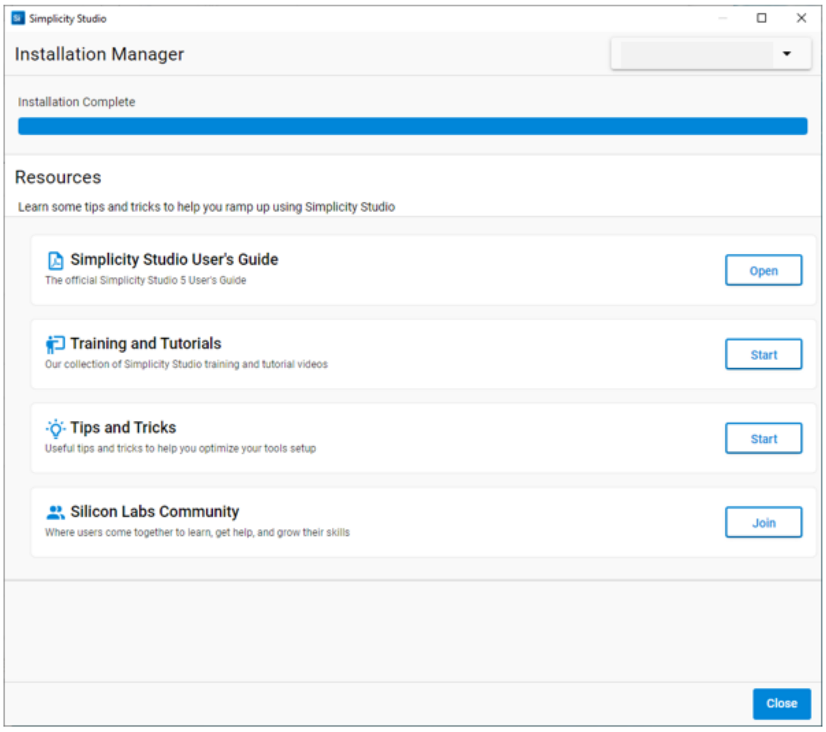
Resources

Learn some tips and tricks to help you ramp up using Simplicity Studio

Simplicity Studio User's Guide	The official Simplicity Studio 5 User's Guide	Open
Training and Tutorials	Our collection of Simplicity Studio training and tutorial videos	Start
Tips and Tricks	Useful tips and tricks to help you optimize your tools setup	Start
Silicon Labs Community	Where users come together to learn, get help, and grow their skills	Join

Depending upon your selections, installation may take some time. The progress bar may pause at points where large files are being installed, but will resume as they complete. If there is a problem during installation, SSv5 displays an error. You must restart SSv5 to resume.

6. When installation is complete, click **CLOSE**.


Simplicity Studio

Installation Manager

Installation Complete

Resources

Learn some tips and tricks to help you ramp up using Simplicity Studio

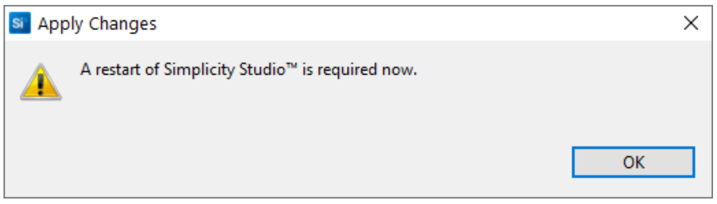
Simplicity Studio User's Guide
The official Simplicity Studio 5 User's Guide

Training and Tutorials
Our collection of Simplicity Studio training and tutorial videos

Tips and Tricks
Useful tips and tricks to help you optimize your tools setup

Silicon Labs Community
Where users come together to learn, get help, and grow their skills

7. Click **RESTART** to begin working with projects.

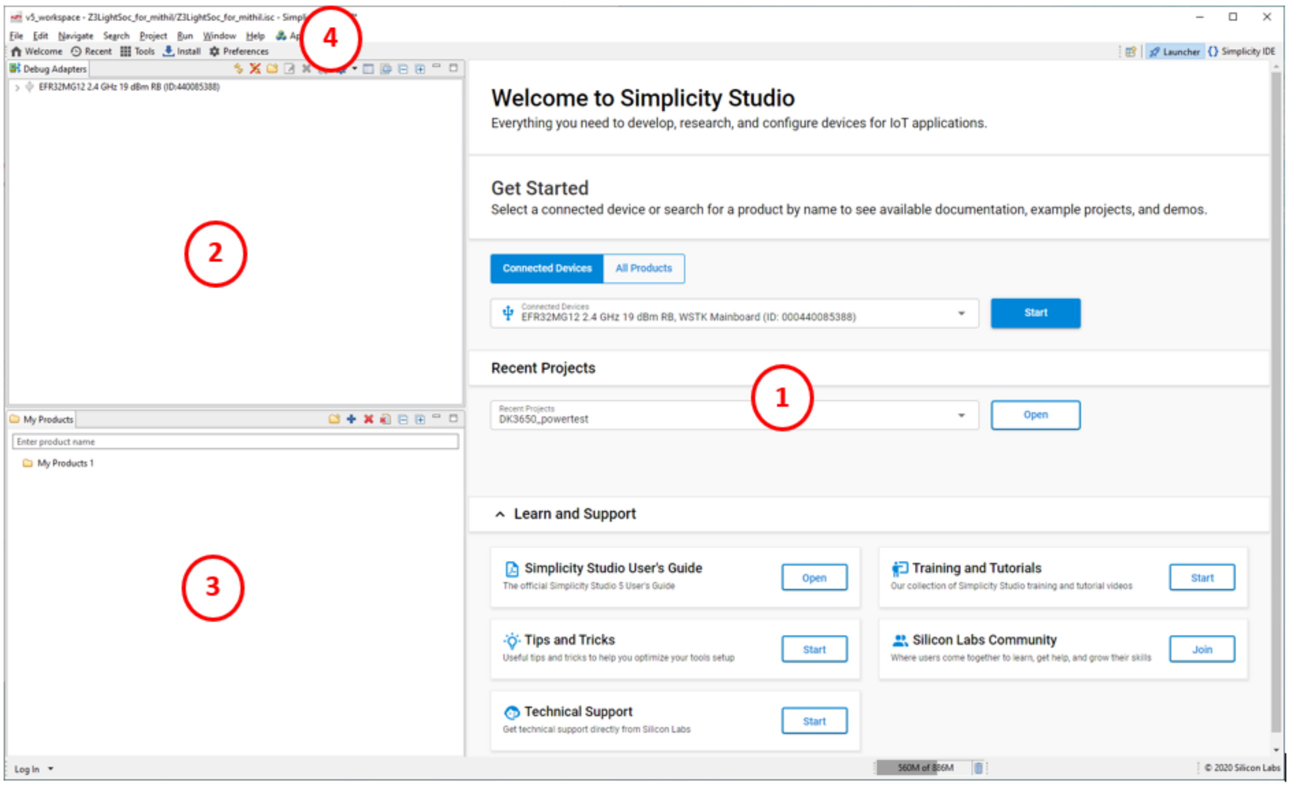

Apply Changes

A restart of Simplicity Studio™ is required now.

Explore SSv5

Explore SSv5

When Simplicity Studio® 5 (SSv5) opens, the first screen is the Welcome page within the Launcher perspective. A "perspective" is an Eclipse term for an arrangement of views and an editor area. This page provides an overview of the Launcher perspective's main parts. A detailed reference to all of the functions accessed through the Launcher perspective can be found in the About the Launcher reference section.


v5_workspace - Z3LightSoc_for_mithil/Z3LightSoc_for_mithil.isc - Simpl
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Debug Adapters
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
My Products
Enter product name
My Products 1
Connected Devices
All Products
Start
Open
Log In
560M of 886M
© 2020 Silicon Labs
1
2
3
4

Welcome to Simplicity Studio

Everything you need to develop, research, and configure devices for IoT applications.

Get Started

Select a connected device or search for a product by name to see available documentation, example projects, and demos.

Connected Devices: EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440085388)

Recent Projects

Recent Projects: DK3650_powertest

Learn and Support

Simplicity Studio User's Guide — The official Simplicity Studio 5 User's Guide — Open

Training and Tutorials — Our collection of Simplicity Studio training and tutorial videos — Start

Tips and Tricks — Useful tips and tricks to help you optimize your tools setup — Start

Silicon Labs Community — Where users come together to learn, get help, and grow their skills — Join

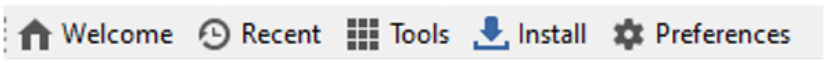
Technical Support — Get technical support directly from Silicon Labs — Start

1 - **Editor**. The editor begins in Welcome mode. Here you get started by selecting a part, and then find resources and create projects based on that part. Note that, when you drop down the Learn and Support area, you have access to technical support, the Silicon Labs Community, and educational resources.

2 - **Debug Adapters view**: Shows devices physically connected to your computer with a debug adapter, or detected on a local network. Select a device to begin a project

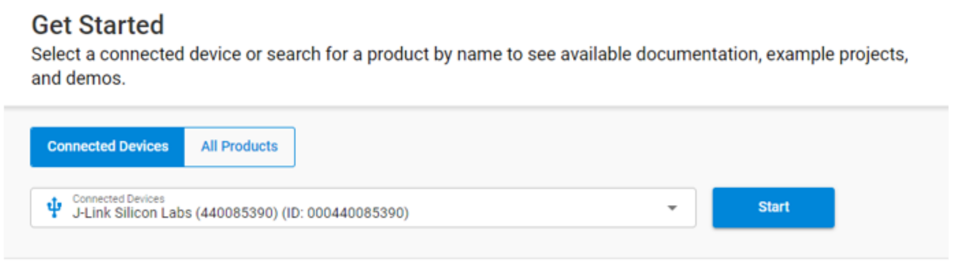
3 - **My Product view**: Here you can add devices, boards, or kits and select them just like a connected kit. You can then explore resources or create and configure projects for the selected (target) device.

4 - **Menu** and **Toolbar**: Many of the main functions of interest are provided on the toolbar.



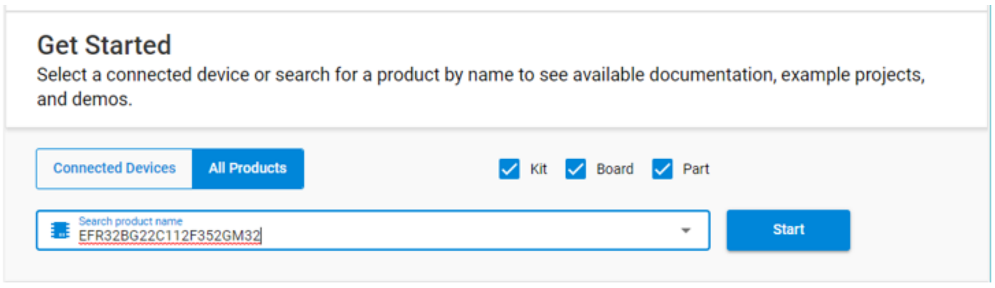
- **Welcome** returns you to the Launcher Welcome page.
- **Recent** shows you a list of recent projects. Select one to go to that project in the Simplicity IDE Project Explorer view.
- **Tools** provides a list of available tools.
- **Install** brings you to a menu where you can install or uninstall software packages and tools, or review available updates.
- **Preferences** is a shortcut to the list of preferences also available through the menu selection Window > Preferences.

Throughout SSv5, the contents of the launcher and other perspectives depends on the device you have selected as your development target, as well as your software environment. When you first open SSv5, the Launcher perspective does not have a target device context. When you connect a kit, it appears both in the the Debug Adapters view and in the device selector in the **Get Started** editor area.


Get Started

Select a connected device or search for a product by name to see available documentation, example projects, and demos.

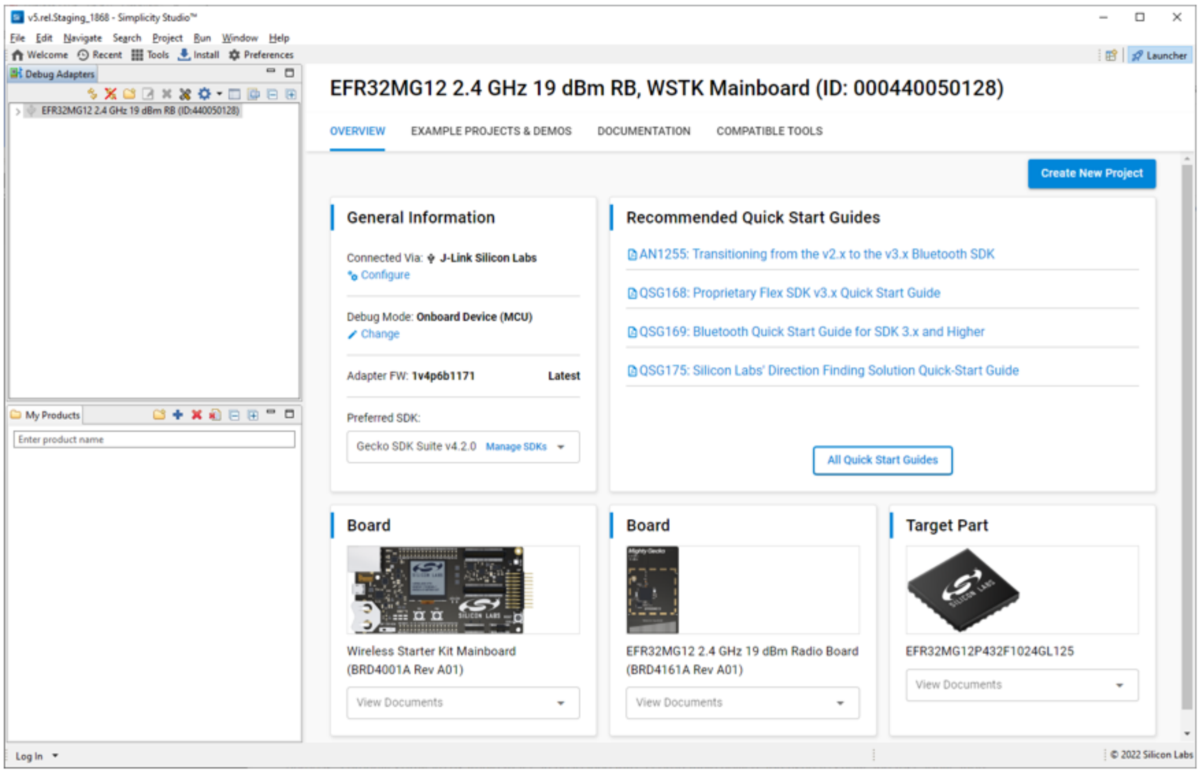
If you don't have a Silicon Labs kit to connect, but would like to explore more of SSv5's functions, click **All Products**, and enter a part number for a kit, board, or device. As you type you will be presented with a list. Use the checkboxes to filter the list.


Get Started

Select a connected device or search for a product by name to see available documentation, example projects, and demos.

You can also add products to the My Products view, and select a target device there. If you have added a product under **Get Started**, once you click **Start** the selected product is added to the My Products view for easy access in the Welcome page or other Launcher perspective views.

Once you have connected or selected a target device, click **Start** to move to the Launcher perspective's OVERVIEW tab, showing the details about that device.



EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440050128)

General Information

Connected Via: J-Link Silicon Labs

Debug Mode: Onboard Device (MCU)

Adapter FW: 1v4p6b1171 Latest

Preferred SDK: Gecko SDK Suite v4.2.0

Recommended Quick Start Guides

AN1255: Transitioning from the v2.x to the v3.x Bluetooth SDK

QSG168: Proprietary Flex SDK v3.x Quick Start Guide

QSG169: Bluetooth Quick Start Guide for SDK 3.x and Higher

QSG175: Silicon Labs' Direction Finding Solution Quick-Start Guide

Board: Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Board: EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)

Target Part: EFR32MG12P432F1024GL125

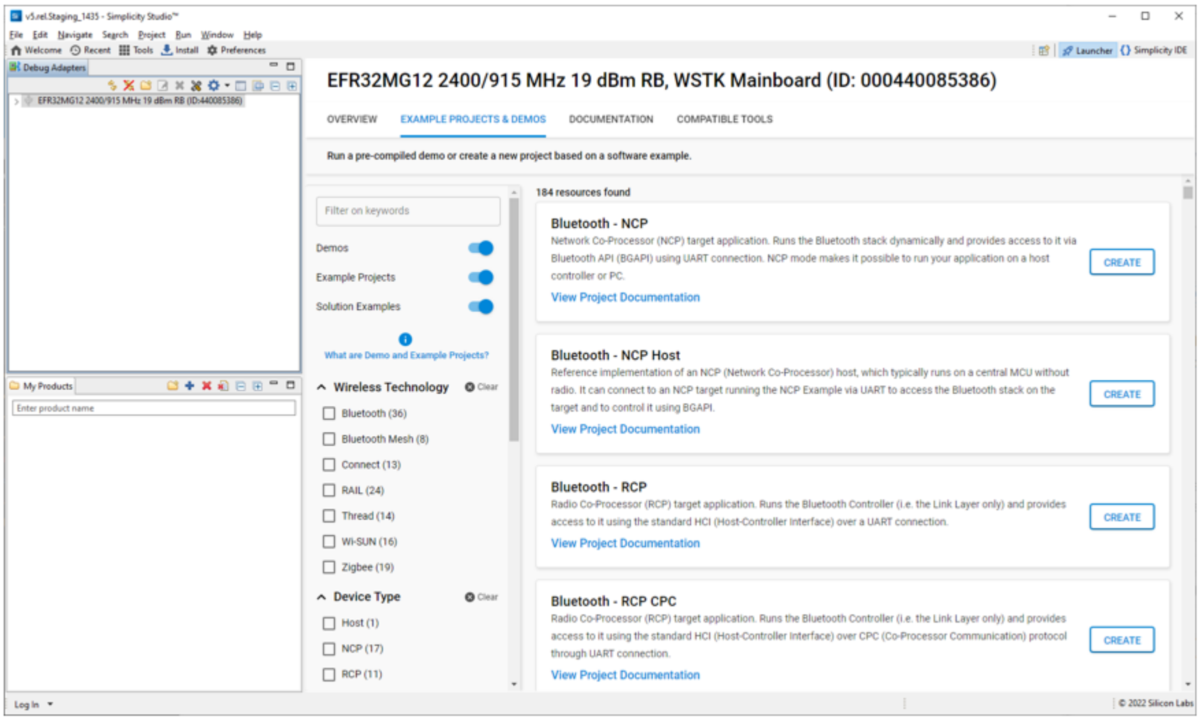
You'll find several "cards" on the OVERVIEW tab. The first is the **General Information** card where you can update the debug adapter firmware of your Silicon Labs kits, change the mode of the debug adapter, manage the security settings of the target device (if applicable), and change the active/preferred SDK.

The **Recommended Quick Start Guides** card highlights some of the getting-started resources available for the selected target device. Click **All Quick-Start Guides** to quickly go to the DOCUMENTATION tab, pre-filtered for quick-start guides. Cards are also available for each board and device of the selected target. The **View Documents** drop-down list on each hardware card provides a filtered view of hardware documentation for that item.

You can start a project with the **Create New Project** control. See Start a Project for an introduction to creating projects.

The EXAMPLE PROJECTS & DEMOS tab shows a list of example projects and demos compatible with the selected device. A demo is a prebuilt software example that can be loaded into a compatible device and used to show and test application functionality. Every demo comes with an associated example project. See the Quick Start Guide for your SDK for more information about the examples and demos provided.

Use the checkboxes and search box to filter the list. A number of different filter categories are available. The categories and filters displayed depend on your selected device. Click **RUN** on any demo to install it on a target device. Click **CREATE** on any project to create it. This is equivalent to creating a project from the OVERVIEW tab, except that the project is already selected.



EFR32MG12 2400/915 MHz 19 dBm RB, WSTK Mainboard (ID: 000440085386)

Run a pre-compiled demo or create a new project based on a software example.

184 resources found

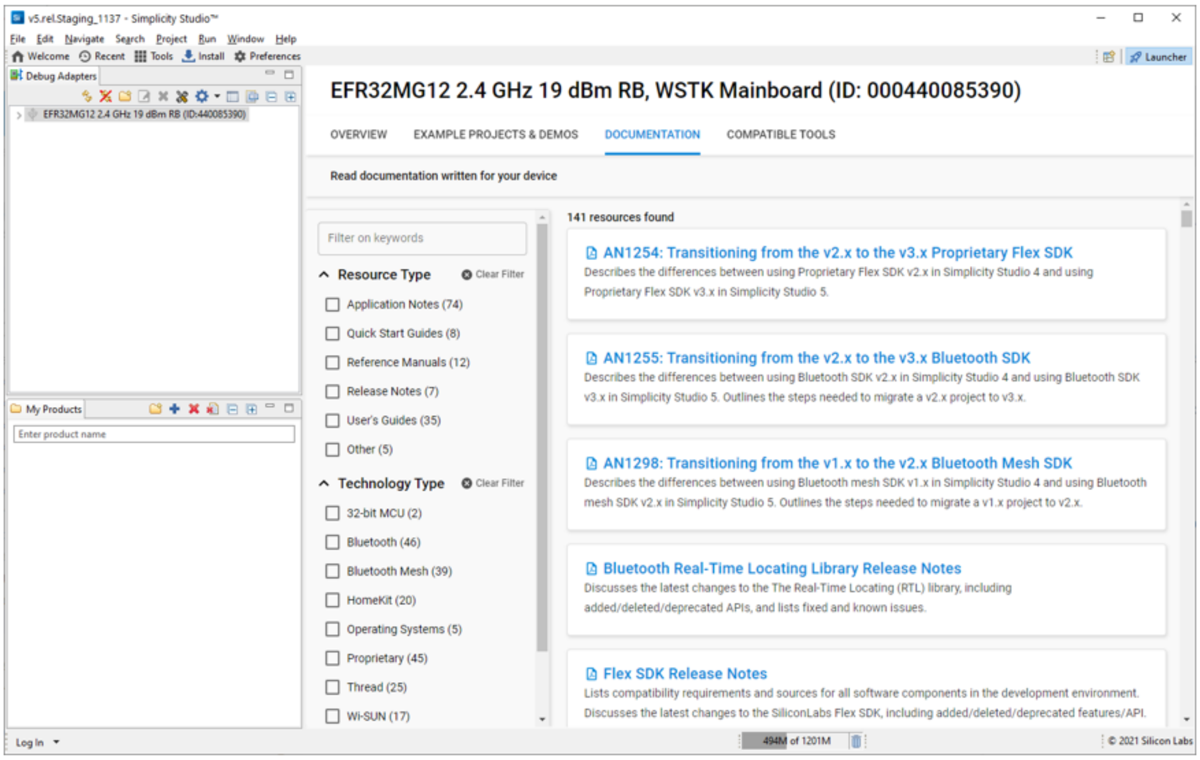
Bluetooth - NCP
Network Co-Processor (NCP) target application. Runs the Bluetooth stack dynamically and provides access to it via Bluetooth API (BGAPI) using UART connection. NCP mode makes it possible to run your application on a host controller or PC.

Bluetooth - NCP Host
Reference implementation of an NCP (Network Co-Processor) host, which typically runs on a central MCU without radio. It can connect to an NCP target running the NCP Example via UART to access the Bluetooth stack on the target and to control it using BGAPI.

Bluetooth - RCP
Radio Co-Processor (RCP) target application. Runs the Bluetooth Controller (i.e. the Link Layer only) and provides access to it using the standard HCI (Host-Controller Interface) over a UART connection.

Bluetooth - RCP CPC
Radio Co-Processor (RCP) target application. Runs the Bluetooth Controller (i.e. the Link Layer only) and provides access to it using the standard HCI (Host-Controller Interface) over CPC (Co-Processor Communication) protocol through UART connection.

The DOCUMENTATION tab shows all documentation compatible with the selected device. Use the checkboxes or text filter field to find a resource of interest. The technology filter corresponding to your development environment will show you most software documents relevant to that environment.



EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440085390)

Read documentation written for your device

141 resources found

AN1254: Transitioning from the v2.x to the v3.x Proprietary Flex SDK
Describes the differences between using Proprietary Flex SDK v2.x in Simplicity Studio 4 and using Proprietary Flex SDK v3.x in Simplicity Studio 5.

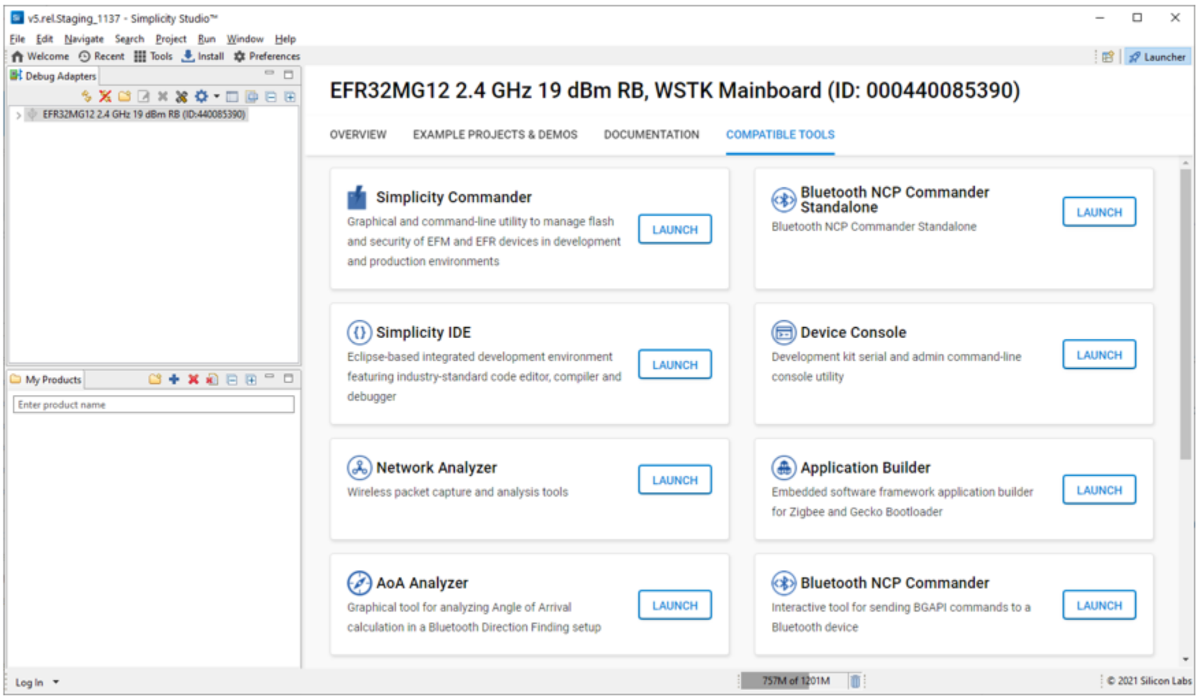
AN1255: Transitioning from the v2.x to the v3.x Bluetooth SDK
Describes the differences between using Bluetooth SDK v2.x in Simplicity Studio 4 and using Bluetooth SDK v3.x in Simplicity Studio 5. Outlines the steps needed to migrate a v2.x project to v3.x.

AN1298: Transitioning from the v1.x to the v2.x Bluetooth Mesh SDK
Describes the differences between using Bluetooth mesh SDK v1.x in Simplicity Studio 4 and using Bluetooth mesh SDK v2.x in Simplicity Studio 5. Outlines the steps needed to migrate a v1.x project to v2.x.

Bluetooth Real-Time Locating Library Release Notes
Discusses the latest changes to the The Real-Time Locating (RTL) library, including added/deleted/deprecated APIs, and lists fixed and known issues.

Flex SDK Release Notes
Lists compatibility requirements and sources for all software components in the development environment. Discusses the latest changes to the SiliconLabs Flex SDK, including added/deleted/deprecated features/API.

The COMPATIBLE TOOLS tab shows the tools compatible with the selected device. The Tools button on the toolbar shows all tools unfiltered.


v5.rel.Staging_1137 - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher
Debug Adapters
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085390)
My Products
Enter product name
OVERVIEW
EXAMPLE PROJECTS & DEMOS
DOCUMENTATION
COMPATIBLE TOOLS
LAUNCH
Log In
757M of 1201M
© 2021 Silicon Labs

EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440085390)

Simplicity Commander
Graphical and command-line utility to manage flash and security of EFM and EFR devices in development and production environments

Bluetooth NCP Commander Standalone
Bluetooth NCP Commander Standalone

Simplicity IDE
Eclipse-based integrated development environment featuring industry-standard code editor, compiler and debugger

Device Console
Development kit serial and admin command-line console utility

Network Analyzer
Wireless packet capture and analysis tools

Application Builder
Embedded software framework application builder for Zigbee and Gecko Bootloader

AoA Analyzer
Graphical tool for analyzing Angle of Arrival calculation in a Bluetooth Direction Finding setup

Bluetooth NCP Commander
Interactive tool for sending BGAPI commands to a Bluetooth device

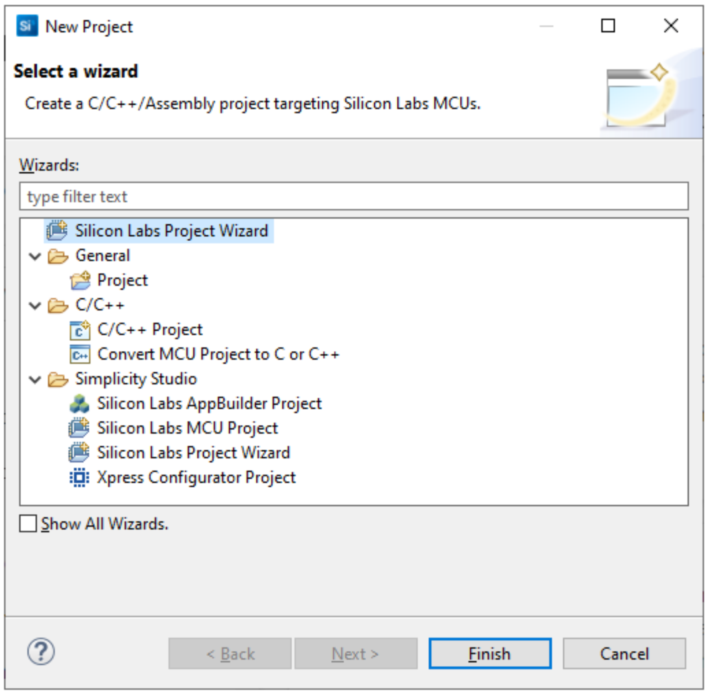
Start a Project

Start a Project

Simplicity Studio® 5 (SSv5) supports several different project types, which can be created through **File > New**. These include:

- Silicon Labs Project Wizard (creates Project Configurator projects, as described in this section)
- Solution... (creates a combination of two or more projects)
- Project... (opens the New Project Wizards dialog)
- Other (combines all of the above selections with other, rarely used options)

Select **Files > New > Project** to open the New Project Wizards dialog.


New Project

Select a wizard

Create a C/C++/Assembly project targeting Silicon Labs MCUs.

Wizards:

type filter text

- Silicon Labs Project Wizard
- General
  - Project
- C/C++
  - C/C++ Project
  - Convert MCU Project to C or C++
- Simplicity Studio
  - Silicon Labs AppBuilder Project
  - Silicon Labs MCU Project
  - Silicon Labs Project Wizard
  - Xpress Configurator Project

Show All Wizards.

Most notably:

- Silicon Labs Project Wizard (two instances) and also Silicon Labs MCU Project Wizard: Creates Project Configurator projects, as described in this section.
- AppBuilder Project Wizard: Creates a legacy AppBuilder Project

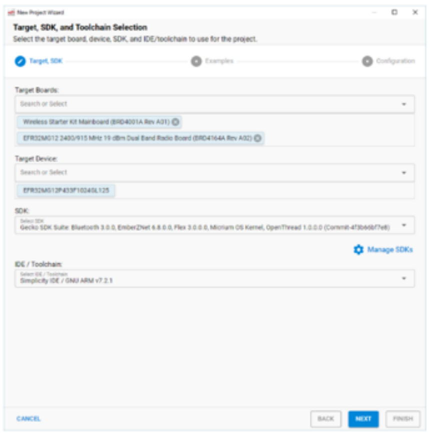
This page focuses on Project Configurator (*.slcp) projects, as well as Solutions, or combinations of Project Configurator projects. On this page:

- Proprietary Radio Configurator
- Wi-SUN Configurator
- ZCL Advanced Platform
- About Solutions
- About AppBuilder Projects

Project Creation

New Simplicity Studio® 5 (SSv5) projects are created through a sequence of three dialogs:

- Target, SDK, and Toolchain
- Examples
- Configuration



New Project Wizard

Target, SDK, and Toolchain Selection

Select the target board, device, SDK, and IDE/toolchain to use for the project.

Target, SDK — Examples — Configuration

Target Boards: Search or Select
Wireless Starter Kit Mainboard (BRD4001A Rev A01)
EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)

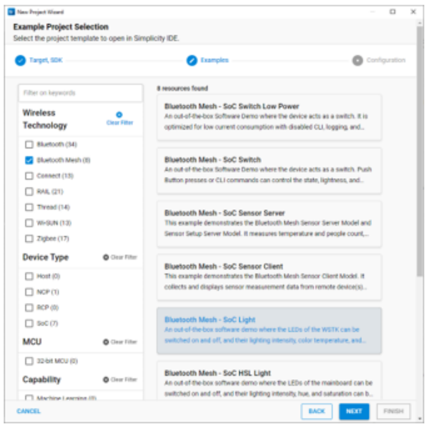
Target Device: Search or Select
EFR32MG12P433F1024GL125

SDK: Select SDK
Gecko SDK Suite: Bluetooth 3.0.0, EmberZNet 6.8.0.0, Flex 3.0.0.0, Micrium OS Kernel, OpenThread 1.0.0.0 (Commit-4f3b6dbf7e8)

Manage SDKs

IDE / Toolchain: Select IDE / Toolchain
Simplicity IDE / GNU ARM v7.2.1

CANCEL BACK NEXT FINISH



New Project Wizard

Example Project Selection

Select the project template to open in Simplicity IDE.

Target, SDK — Examples — Configuration

Filter on keywords

Wireless Technology (Clear Filter)
- [ ] Bluetooth (34)
- [x] Bluetooth Mesh (8)
- [ ] Connect (13)
- [ ] RAIL (21)
- [ ] Thread (14)
- [ ] Wi-SUN (13)
- [ ] Zigbee (17)

Device Type (Clear Filter)
- [ ] Host (0)
- [ ] NCP (1)
- [ ] RCP (0)
- [ ] SoC (7)

MCU (Clear Filter)
- [ ] 32-bit MCU (0)

Capability (Clear Filter)
- [ ] Machine Learning (0)

8 resources found

Bluetooth Mesh - SoC Switch Low Power
An out-of-the-box Software Demo where the device acts as a switch. It is optimized for low current consumption with disabled CLI, logging, and...

Bluetooth Mesh - SoC Switch
An out-of-the-box Software Demo where the device acts as a switch. Push Button presses or CLI commands can control the state, lightness, and...

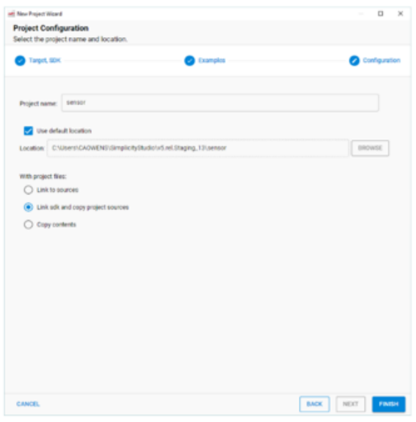
Bluetooth Mesh - SoC Sensor Server
This example demonstrates the Bluetooth Mesh Sensor Server Model and Sensor Setup Server Model. It measures temperature and people count,...

Bluetooth Mesh - SoC Sensor Client
This example demonstrates the Bluetooth Mesh Sensor Client Model. It collects and displays sensor measurement data from remote device(s)...

Bluetooth Mesh - SoC Light
An out-of-the-box software demo where the LEDs of the WSTK can be switched on and off, and their lighting intensity, color temperature, and...

Bluetooth Mesh - SoC HSL Light
An out-of-the-box software demo where the LEDs of the mainboard can be switched on and off, and their lighting intensity, hue, and saturation can b...

CANCEL BACK NEXT FINISH



New Project Wizard

Project Configuration

Select the project name and location.

Target, SDK — Examples — Configuration

Project name: sensor

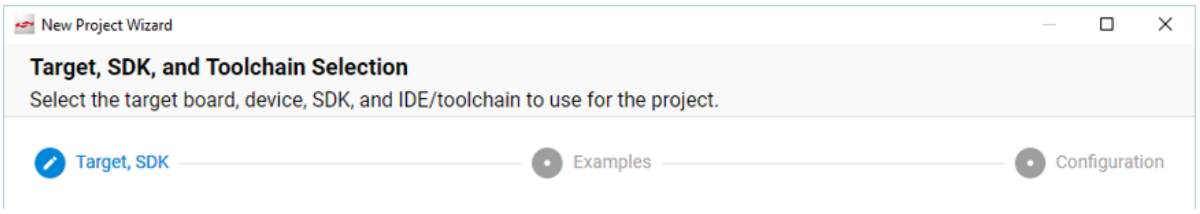
- [x] Use default location

Location: C:\Users\CAOWENS\SimplicityStudio\v5.rel.Staging_13\sensor BROWSE

With project files:
- ○ Link to sources
- ● Link sdk and copy project sources
- ○ Copy contents

CANCEL BACK NEXT FINISH

An indicator at the top of the dialog shows you where you are.



New Project Wizard

Target, SDK, and Toolchain Selection

Select the target board, device, SDK, and IDE/toolchain to use for the project.

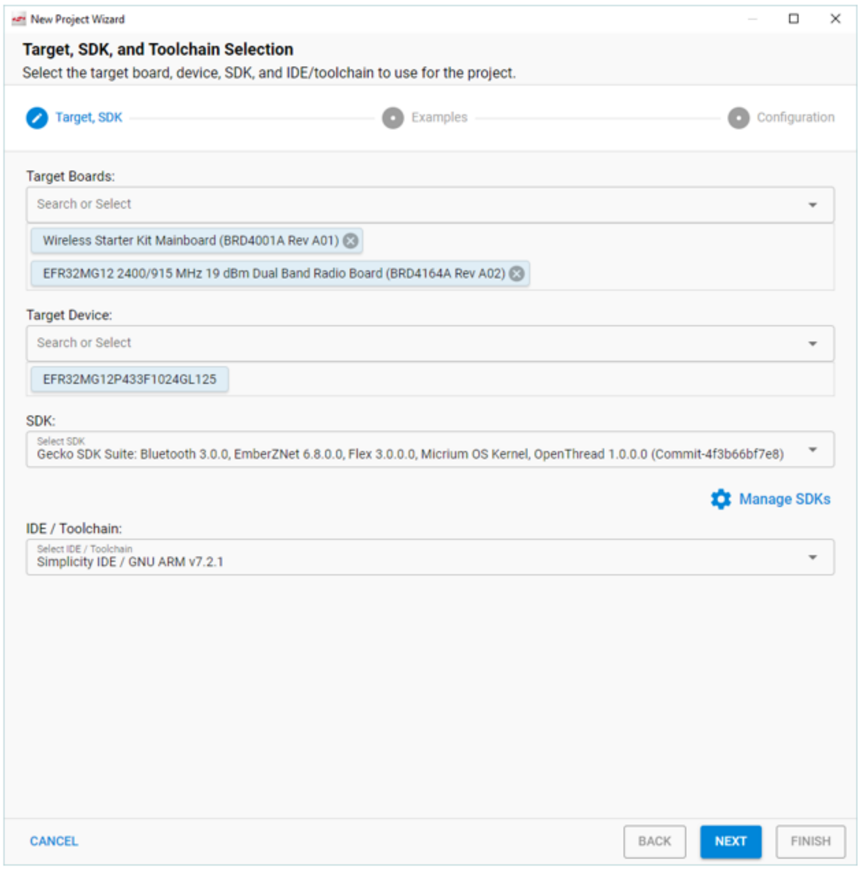
You can start a project from three different locations in the Launcher Perspective. Where you start determines which of the three dialogs you will land on. Click **BACK** to move to an earlier dialog, if you need to make a change.

- **From the OVERVIEW tab:** Click **Create New Project**. Starts on the Examples dialog.
- **From the EXAMPLE PROJECTS tab:** Use the filters as needed, select a project, and click **CREATE**. Starts on the Project Configuration dialog.
- **From the file menu**: Select New >> Silicon Labs Project Wizard. Starts on the Target, SDK, and Toolchain Selection dialog.

While you are getting started, you can leave the default values in place.

Target, SDK, and Toolchain Selection

If you have connected or selected a target, all information is pre-populated. Otherwise you can select target parts here. Click **NEXT**.



New Project Wizard

Target, SDK, and Toolchain Selection

Select the target board, device, SDK, and IDE/toolchain to use for the project.

Target Boards: Wireless Starter Kit Mainboard (BRD4001A Rev A01), EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)

Target Device: EFR32MG12P433F1024GL125

SDK: Gecko SDK Suite: Bluetooth 3.0.0, EmberZNet 6.8.0.0, Flex 3.0.0.0, Micrium OS Kernel, OpenThread 1.0.0.0 (Commit-4f3b66bf7e8)

IDE / Toolchain: Simplicity IDE / GNU ARM v7.2.1

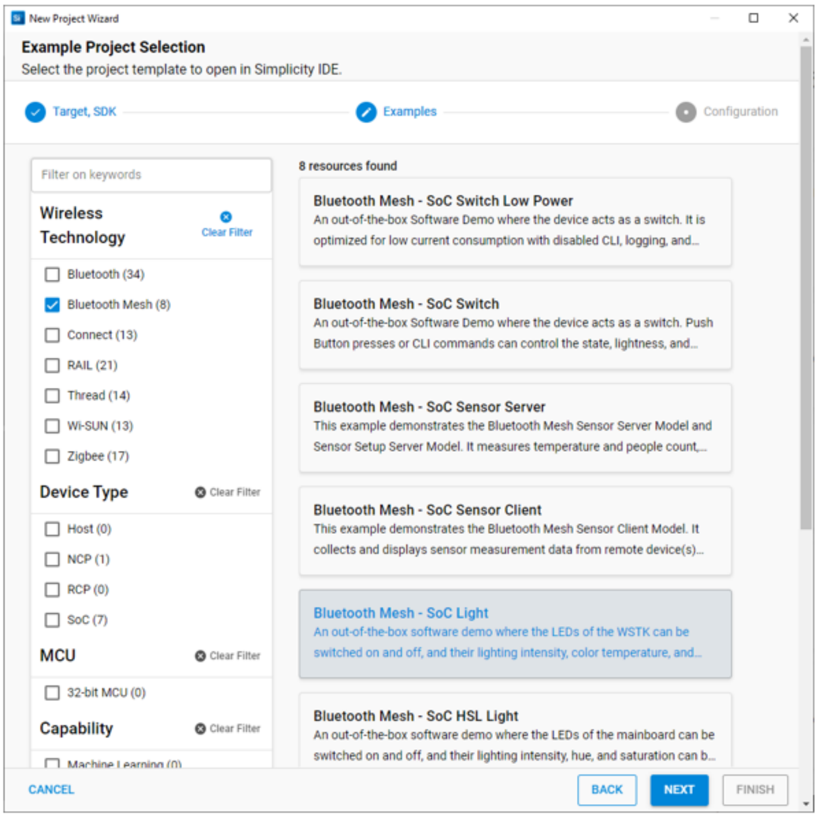
IDE / Toolchain has several options, not all of which are used within the Simplicity IDE. In those cases the project still shows up in the Simplicity IDE Project Explorer view, but you use the IDE for which the project was created to build and debug the project.

- Simplicity IDE / GNU ARM vn.n.n - This is the default toolchain used when Simplicity Studio is first installed. The Simplicity IDE is used for all project operations - editing source files, building the project, debugging the project. If multiple version of the GNU ARM Toolchain are installed, there are options for each version.
- Simplicity IDE / IAR ARM vn.n.n.n - Similar to the first option, but builds use the IAR ARM toolchain. The IAR ARM toolchain requires installation of IAR Embedded Workbench (EW) external to Simplicity Studio and a license for IAR EW must be purchased from IAR. All project operations are still performed inside the Simplicity IDE.
- Makefile IDE (use Simplicity Studio tools to configure and, with a manual step, to debug the project but use 'make' to build the project) - This option creates a project without a Simplicity IDE build configuration and [No toolchain] will be reported. Instead, PROJECTNAME.Makefile and PROJECTNAME.project.mak files are created. They are used from a command prompt or terminal prompt with 'make -f PROJECTNAME.Makefile' to build the project. The Project Configurator (.slcp file) can be used to add / configure / remove software components and the various Advanced Configurators, such as Pintool, Bluetooth Gatt Configurator, or ZCL Advanced Platform, can also be used to configure and generate the project. It just cannot be built from within Simplicity Studio. Also the project cannot be automatically debugged within Simplicity Studio. You must manually create a debug configuration that points to the build .out file.
- Visual Studio Code - This option allows you to use Visual Studio Code in combination with Simplicity Studio. Use Simplicity Studio to create the initial project and to make project changes with the Component Editor and various Advanced Configurators. Then do all editing, building, and debugging of the project in VS Code. See the Visual Studio Code page for more information.
- CMake - This option creates both the Simplicity Studio .slcp file and a CMake file during project generation. The Project Configurator (.slcp file) can be used to add / configure / remove software components and the various Advanced Configurators, such as Pintool, Bluetooth Gatt Configurator, or ZCL Advanced Platform, can also be used to configure and generate the project.
- IAR Embedded Workbench (use Simplicity Studio tools to configure the project but IAR EW to build and debug the project) - This option creates a project with the IAR EW projects files but without a Simplicity IDE build configuration and [No toolchain] will be reported. As with the Makefile IDE, the Project Configurator (.slcp file) can be used to add / configure /

remove software components and the various Advanced Configurators can also be used to configure and generate the project. After that, double-click the .eww file to launch IAR EW for building and debugging the project. Project options should be set in IAR EW. You cannot use the Simplicity IDE and IAR EW interchangeably. All operations except project configuration should be done in IAR EW.

Examples

Use the checkboxes or keywords to find the example of interest, then select it and click **NEXT**.


New Project Wizard

Example Project Selection

Select the project template to open in Simplicity IDE.

Target, SDK — Examples — Configuration

Filter on keywords

Wireless Technology Clear Filter

- [ ] Bluetooth (34)
- [x] Bluetooth Mesh (8)
- [ ] Connect (13)
- [ ] RAIL (21)
- [ ] Thread (14)
- [ ] Wi-SUN (13)
- [ ] Zigbee (17)

Device Type Clear Filter

- [ ] Host (0)
- [ ] NCP (1)
- [ ] RCP (0)
- [ ] SoC (7)

MCU Clear Filter

- [ ] 32-bit MCU (0)

Capability Clear Filter

- [ ] Machine Learning (0)

8 resources found

Bluetooth Mesh - SoC Switch Low Power
An out-of-the-box Software Demo where the device acts as a switch. It is optimized for low current consumption with disabled CLI, logging, and...

Bluetooth Mesh - SoC Switch
An out-of-the-box Software Demo where the device acts as a switch. Push Button presses or CLI commands can control the state, lightness, and...

Bluetooth Mesh - SoC Sensor Server
This example demonstrates the Bluetooth Mesh Sensor Server Model and Sensor Setup Server Model. It measures temperature and people count,...

Bluetooth Mesh - SoC Sensor Client
This example demonstrates the Bluetooth Mesh Sensor Client Model. It collects and displays sensor measurement data from remote device(s)...

Bluetooth Mesh - SoC Light
An out-of-the-box software demo where the LEDs of the WSTK can be switched on and off, and their lighting intensity, color temperature, and...

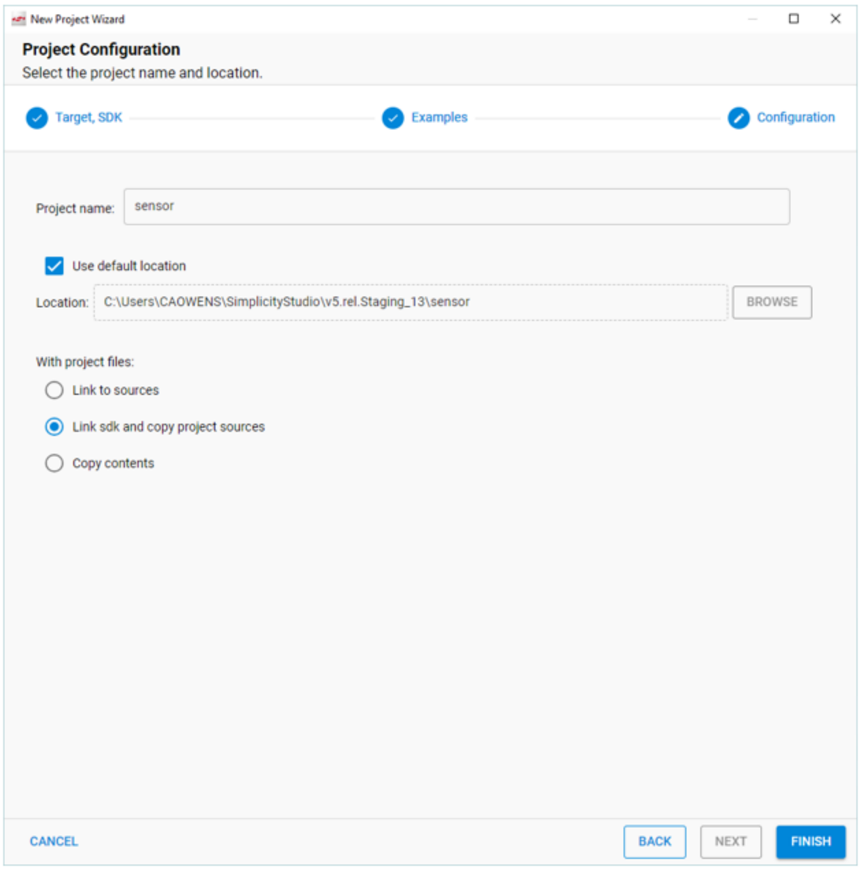
Bluetooth Mesh - SoC HSL Light
An out-of-the-box software demo where the LEDs of the mainboard can be switched on and off, and their lighting intensity, hue, and saturation can b...

Configuration

Rename your project if you want. The three selections under "With project files" control which sources are copied and which are linked. If a linked source is modified, the changes apply to any other project linking to that source.

- Link to sources: The project links both to the SDK files and to project files such as app.c.
- Link SDK and copy project sources (default): Project sources are copied to your workspace. Note that the SDK folder structure is created but if you drill down you will see that all folders are empty.
- Copy contents: Both project files and the SDK files are copied to your workspace.

Click **FINISH**.



New Project Wizard

Project Configuration

Select the project name and location.

Target, SDK — Examples — Configuration

Project name: sensor

[✓] Use default location

Location: C:\Users\CAOWENS\SimplicityStudio\v5.rel.Staging_13\sensor

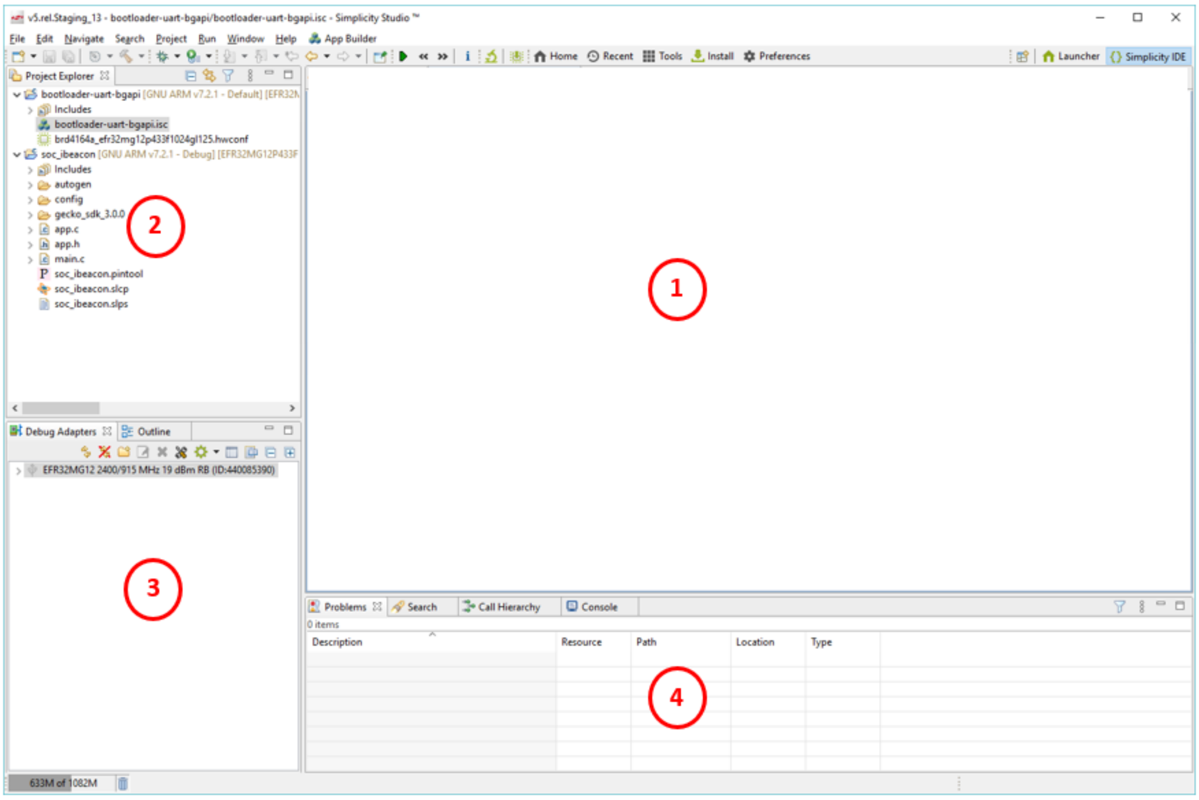
BROWSE

With project files:

- ( ) Link to sources
- (●) Link sdk and copy project sources
- ( ) Copy contents

CANCEL — BACK — NEXT — FINISH

Once you finish project creation, the Simplicity IDE perspective opens. There may be a slight delay for initial configuration. For details on all the features and functions available in this perspective, see About the Simplicity IDE.


v5.rel.Staging_13 - bootloader-uart-bgapi/bootloader-uart-bgapi.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Home Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
bootloader-uart-bgapi [GNU ARM v7.2.1 - Default] [EFR32M
Includes
bootloader-uart-bgapi.isc
brd4164a_efr32mg12p433f1024gl125.hwconf
soc_ibeacon [GNU ARM v7.2.1 - Debug] [EFR32MG12P433F
Includes
autogen
config
gecko_sdk_3.0.0
app.c
app.h
main.c
soc_ibeacon.pintool
soc_ibeacon.slcp
soc_ibeacon.slps
Debug Adapters
Outline
EFR32MG12 2400/915 MHz 19 dBm RB (ID:440085390)
Problems
Search
Call Hierarchy
Console
0 items
Description
Resource
Path
Location
Type
633M of 1082M
1
2
3
4

1 - **Editor area** (depends on the project).

2 - **Project Explorer** view: Lists the projects and solutions available in your workspace.

3 - **Debug Adapters** view: Lists the kits or SEGGER J-Links connected to your computer via USB or detected on a local network.

4 - **Developer** views: A set of views of use during the development process.

The editor in the Simplicity IDE perspective depends on the project:

- Project Configurator: Used for all Gecko SDK protocols and tools beginning with Gecko SDK 4.0; Project files end in .slcp.
- 8-bit Hardware Configurator: Used for 8-bit device applications.
- AppBuilder: Legacy tool used for Zigbee EmberZNet and Gecko Bootloader in Gecko SDK 3.2 and lower; Project files end in .isc.

Project Configurator Projects

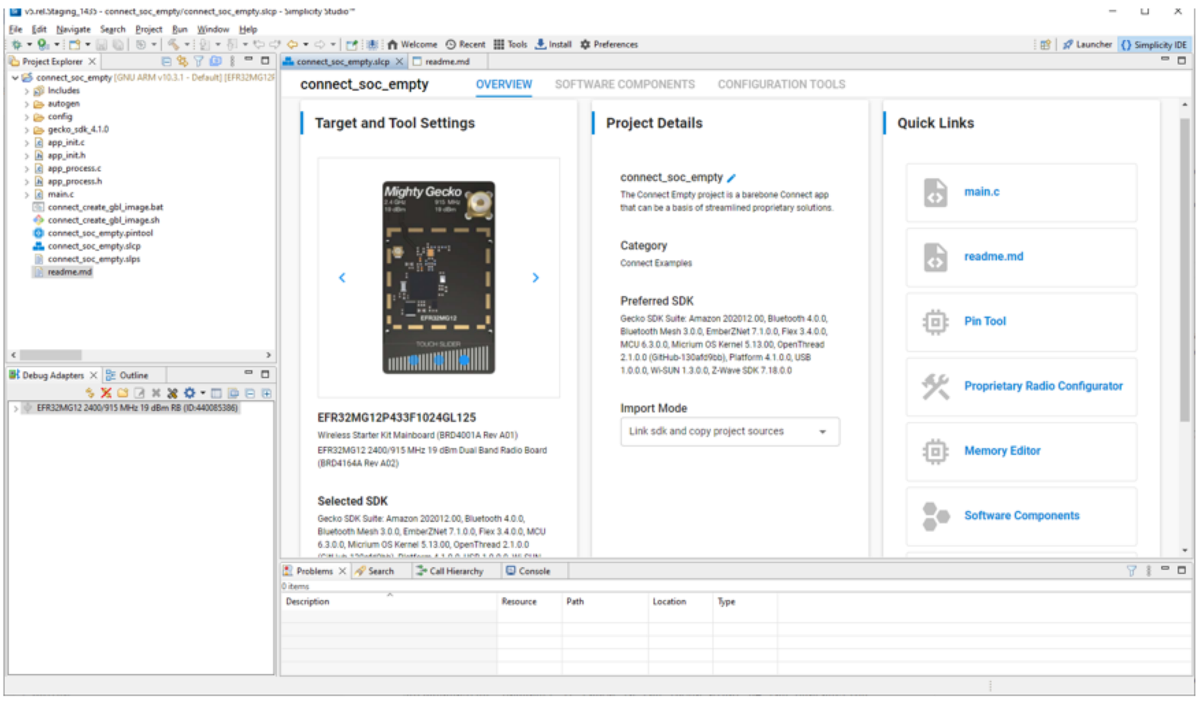
As well as introducing Project Configurator projects, this page also introduces

- The Pin Tool
- Bluetooth and Bluetooth Mesh SDK's GATT Configurator
- Bluetooth Mesh SDK's Bluetooth Mesh Configurator
- Proprietary SDK's Radio Configurator
- Zigbee EmberZNet and Matter SDK's ZCL Advanced Platform
- Wi-SUN SDK's Wi-SUN Configurator

Project Configurator projects are defined in **.slcp** (Silicon Labs Configurator project) files. Users can modify the project by adding, removing, and configuring components on the Software Components tab. See Developing with Project Configurator for details.

NOTE: Beginning with Simplicity Studio version 5.3, you can create a project in the Simplicity IDE by importing an .slcp file. See Import and Export for more information.

The project opens, generally either on a README tab containing an example project description, or on an OVERVIEW tab.



connect_soc_empty

Target and Tool Settings

EFR32MG12P433F1024GL125

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)

Selected SDK

Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.0.0, Bluetooth Mesh 3.0.0, EmberZNet 7.1.0.0, Flex 3.4.0.0, MCU 6.3.0.0, Micrium OS Kernel 5.13.00, OpenThread 2.1.0.0

Project Details

connect_soc_empty

The Connect Empty project is a barebone Connect app that can be a basis of streamlined proprietary solutions.

Category

Connect Examples

Preferred SDK

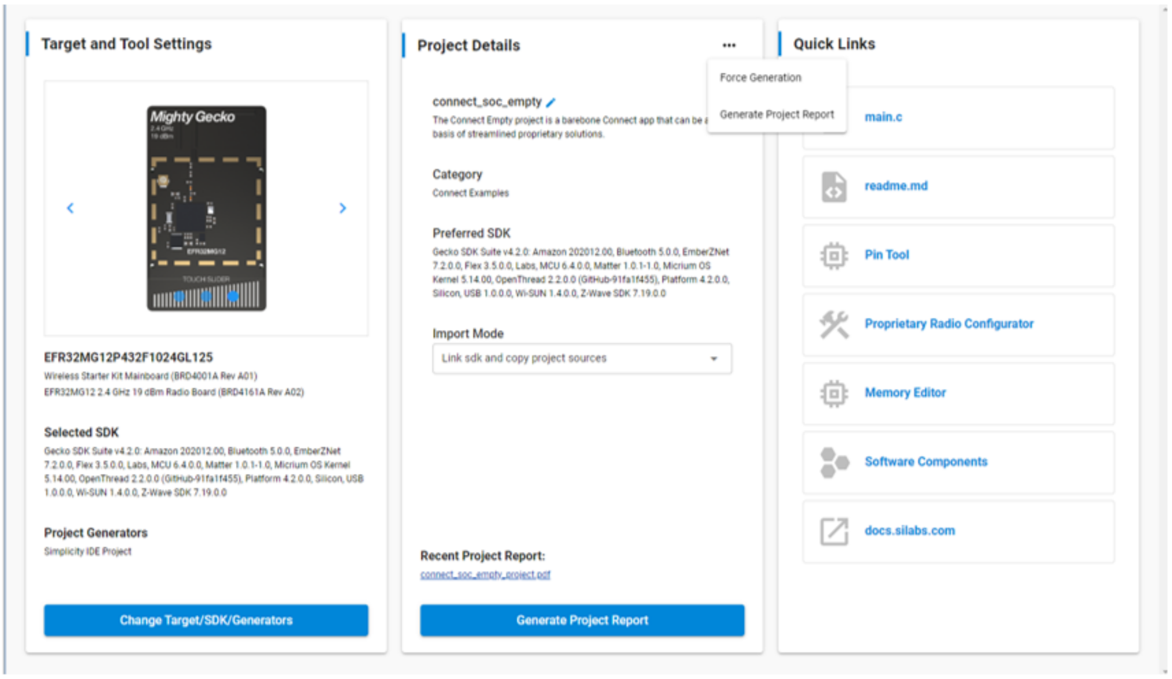
Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.0.0, Bluetooth Mesh 3.0.0, EmberZNet 7.1.0.0, Flex 3.4.0.0, MCU 6.3.0.0, Micrium OS Kernel 5.13.00, OpenThread 2.1.0.0 (GitHub-130afd9bb), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.0.0, Z-Wave SDK 7.18.0.0

Import Mode

Link sdk and copy project sources

The OVERVIEW tab has three cards with information, some with settings that may be changed:

- **Target and Tool Settings**, where you can change your development target, SDK, and project generators. Scroll down and click **Change Target/SDK/Generators** to edit these settings. The Project Generators configuration determines the IDE or build system project files that SSv5 generates as you configure your project. (Note: The compiler / toolchain used by Simplicity IDE is configurable in **Project > Build Configurations**. The default IDE is configurable in **Preferences > Simplicity Studio > Preferred IDE**.)
- **Project Details**, where you can rename the project, change the project source import mode, generate a project report and, if necessary, force the generation of project and source files (in the *autogen* folder).
  - **Generate Project Report** creates a PDF, available through a link immediately above the control, that provides details on:
    - Target hardware device
    - Target hardware device pin mapping
    - Target SDK
    - Software extensions used in the build
    - Installed application, utility, and platform software components
  - **Import mode** controls which resources are copied and which are linked. If you modify a linked source, your changes apply to any other project linking to that source.
  - **Force Generation** (available through the **...** drop-down menu) can be used in rare cases when auto-generation is not triggered, usually because of some change made outside of SSv5 such as editing the .slcp file.
- **Quick Links** provides links to the tools commonly used to modify the project or lower-level configurations such as radio or peripheral connections. The links vary based on the SDK and target device.



EFR32MG12P432F1024GL125
Wireless Starter Kit Mainboard (BRD4001A Rev A01)
EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A02)

Selected SDK
Gecko SDK Suite v4.2.0: Amazon 202012.00, Bluetooth 5.0.0, EmberZNet 7.2.0.0, Flex 3.5.0.0, Labs, MCU 6.4.0.0, Matter 1.0.1-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.0.0 (GitHub-91fa1f455), Platform 4.2.0.0, Silicon, USB 1.0.0.0, Wi-SUN 1.4.0.0, Z-Wave SDK 7.19.0.0

Project Generators
Simplicity IDE Project

connect_soc_empty
The Connect Empty project is a barebone Connect app that can be a basis of streamlined proprietary solutions.

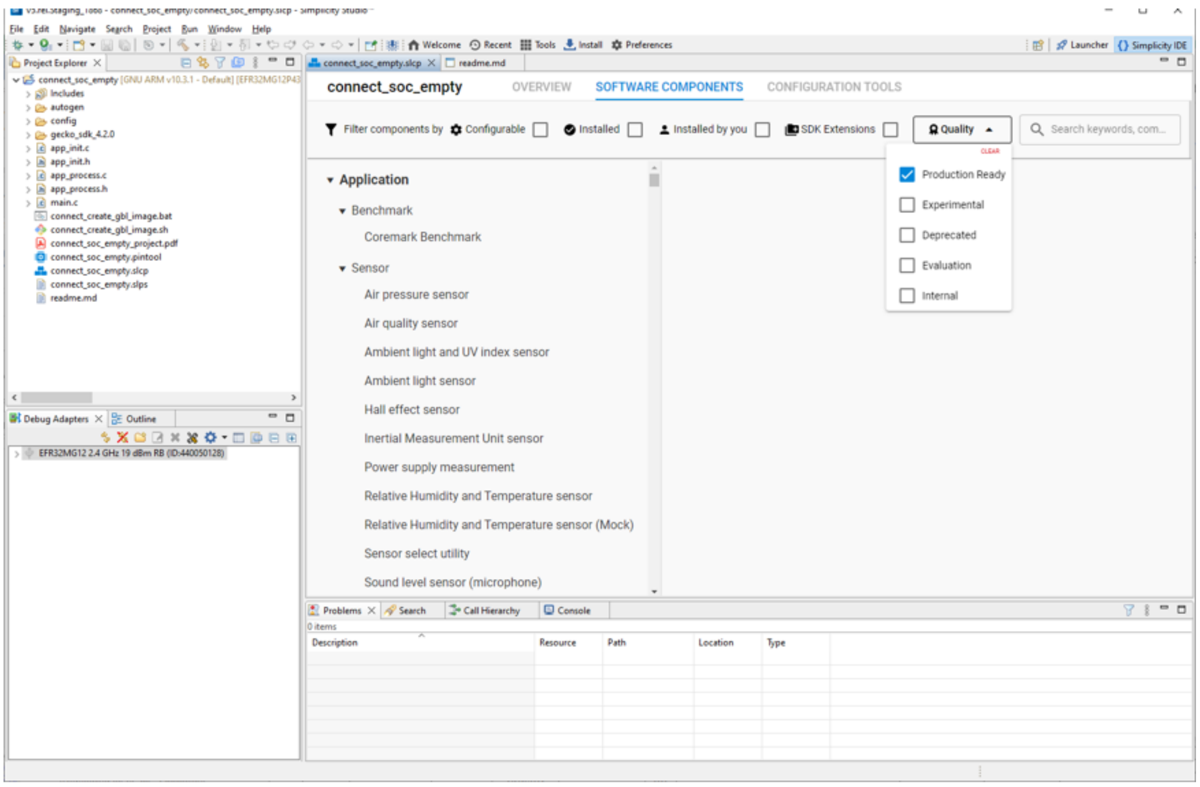
Category
Connect Examples

Preferred SDK
Gecko SDK Suite v4.2.0: Amazon 202012.00, Bluetooth 5.0.0, EmberZNet 7.2.0.0, Flex 3.5.0.0, Labs, MCU 6.4.0.0, Matter 1.0.1-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.0.0 (GitHub-91fa1f455), Platform 4.2.0.0, Silicon, USB 1.0.0.0, Wi-SUN 1.4.0.0, Z-Wave SDK 7.19.0.0

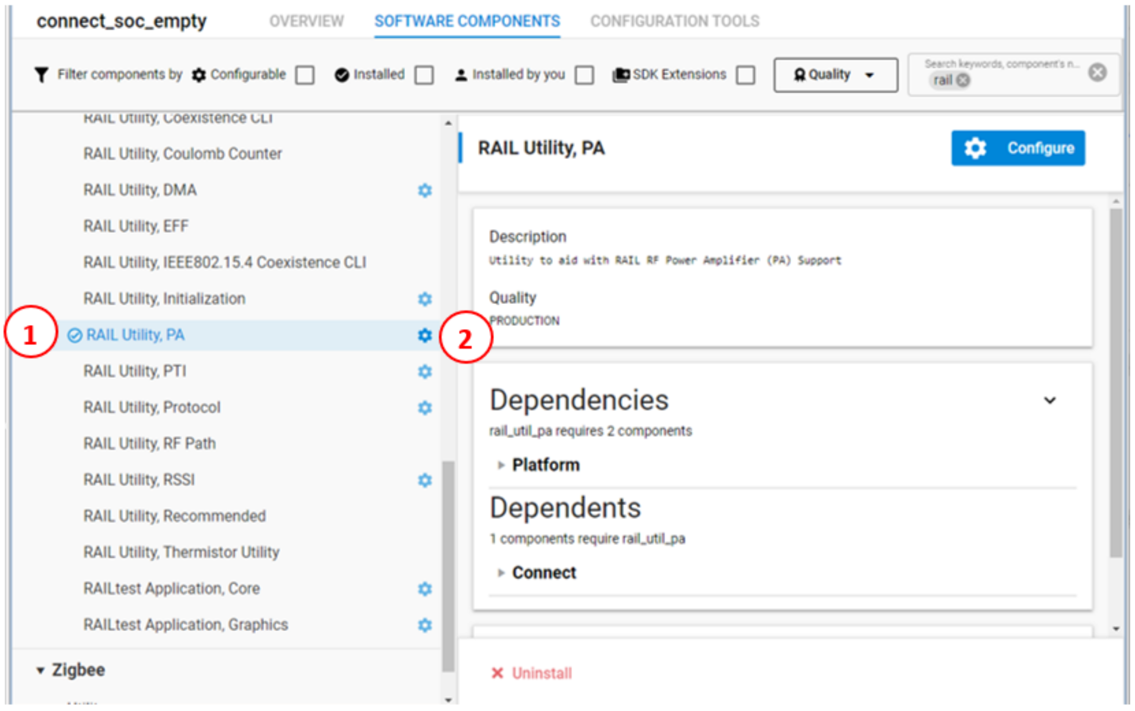
Import Mode
Link sdk and copy project sources

Recent Project Report:
connect_soc_empty_project.pdf

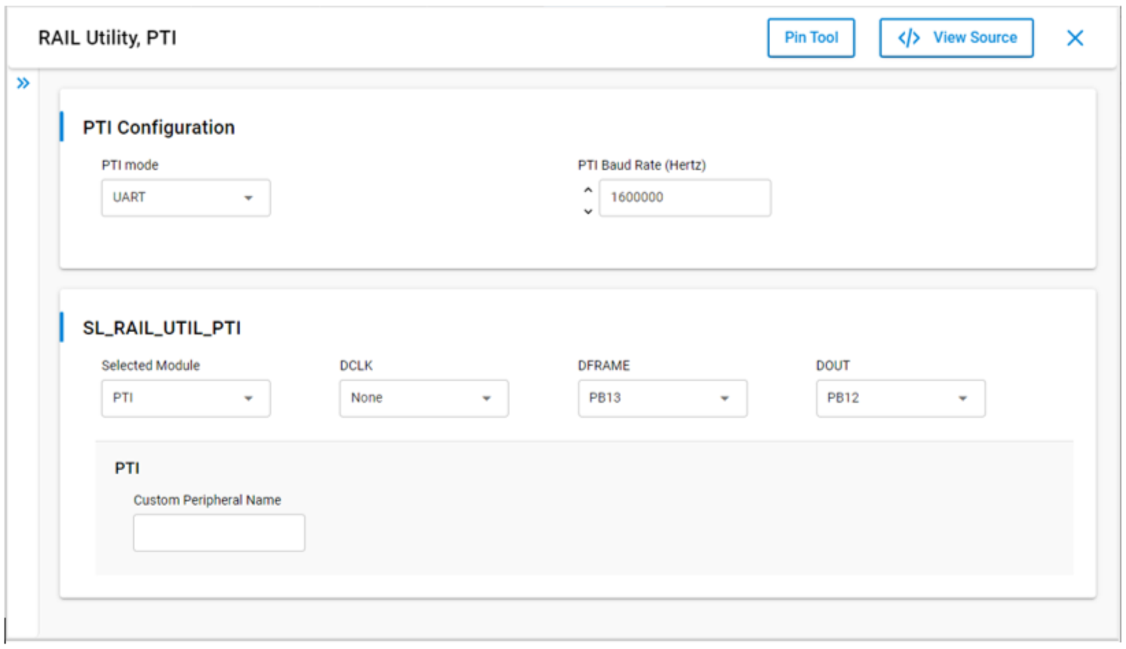
To configure the project through the component library, click the SOFTWARE COMPONENTS tab. A number of filters as well as a keyword search are available to help you explore the various component categories. The **SDK Extensions** filter is only enabled if you have an extension installed. Note that components for all installed SDKs are presented.



Expand a component category\subcategory to see individual components. Components installed in the project are checked (1), and can be uninstalled. Configurable components are indicated by a gear symbol (2).



Click the gear symbol next to the component name or **Configure** in the configurable component description to open the Component Editor. Here you can change parameters or edit the component source directly. You can also go from the component to the Pin Tool



RAIL Utility, PTI

Pin Tool | View Source

PTI Configuration

PTI mode: UART

PTI Baud Rate (Hertz): 1600000

SL_RAIL_UTIL_PTI

Selected Module: PTI

DCLK: None

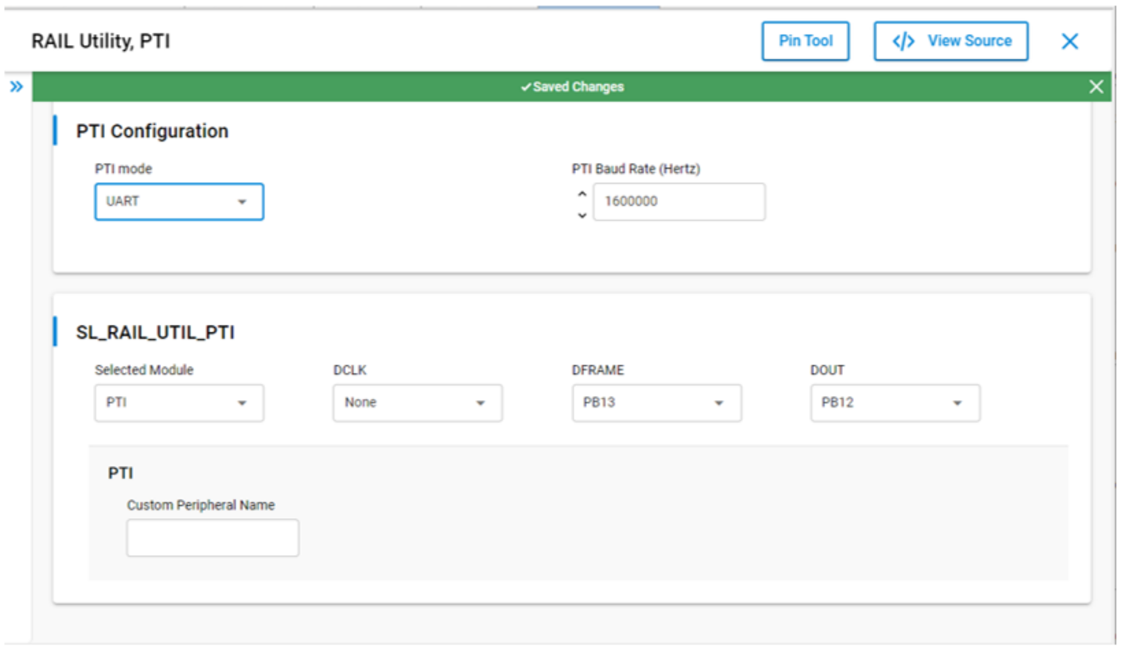
DFRAME: PB13

DOUT: PB12

PTI

Custom Peripheral Name:

Changes are autosaved in the Component Editor.



RAIL Utility, PTI

Pin Tool | View Source

✓Saved Changes

PTI Configuration

PTI mode: UART

PTI Baud Rate (Hertz): 1600000

SL_RAIL_UTIL_PTI

Selected Module: PTI

DCLK: None

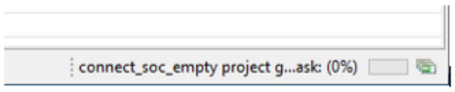
DFRAME: PB13

DOUT: PB12

PTI

Custom Peripheral Name:

As you make changes in the Project Configurator, for example installing or uninstalling a component, project output files are autogenerated. Progress is shown in the lower right of the Perspective.

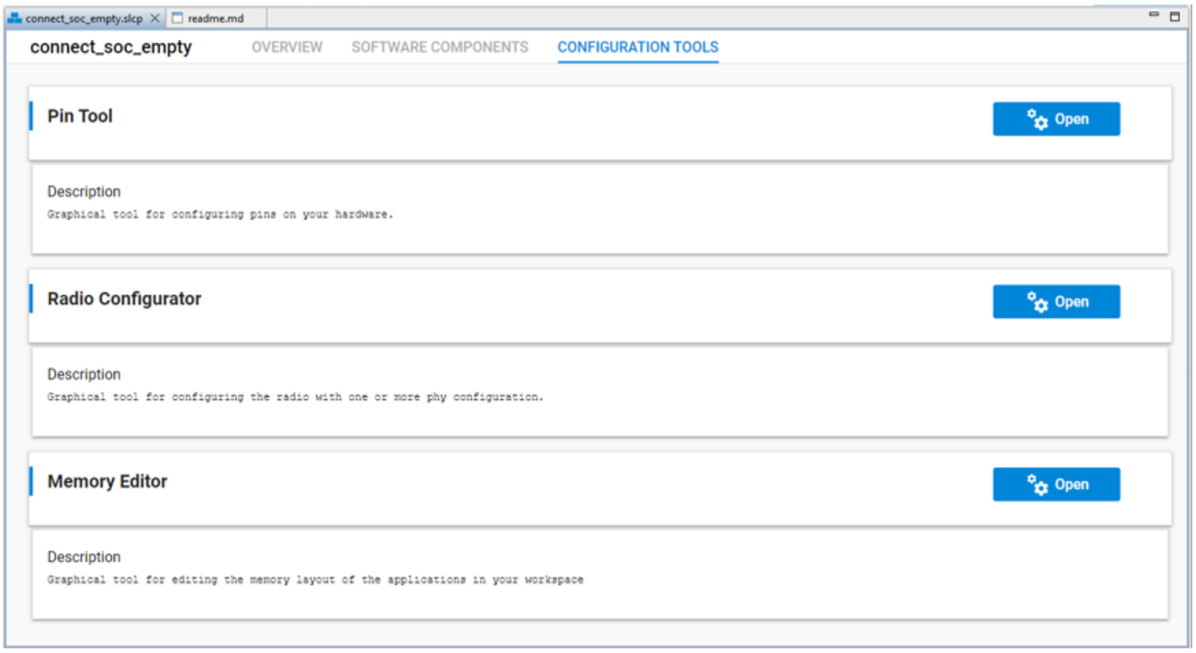


connect_soc_empty project g...ask: (0%)

Speed varies depending on your system. Be sure that generation is complete before building the application image.

Build the application image and flash it to your target device as described in Building and Flashing.

A CONFIGURATION TOOLS tab provides an easy way to open a tool when the tool's tab is not already open, as an alternative to the Quick Links card on the general tab. It shows configuration tools relevant to the project type. For example, a Bluetooth Mesh project shows a number of tools, while an OpenThread project might only show the Pin Tool. Click **Open** on the tool's card to open it in a separate tab.



Pin Tool

Description
Graphical tool for configuring pins on your hardware.

Radio Configurator

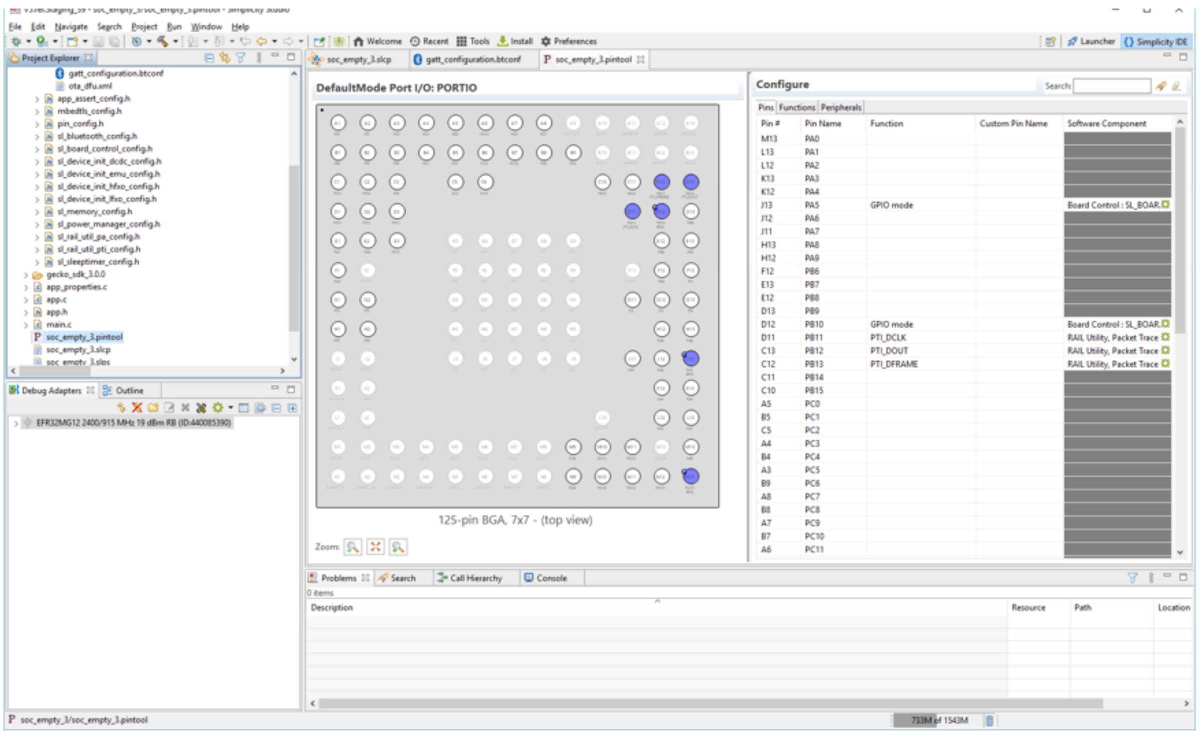
Description
Graphical tool for configuring the radio with one or more phy configuration.

Memory Editor

Description
Graphical tool for editing the memory layout of the applications in your workspace

Pin Tool

The Pin Tool lets you modify the target device's pin use and parameters. As well as opening the Pin Tool through the CONFIGURATION TOOLS tab, you can also double-click the <project>.pintool file in the Project Explorer view.

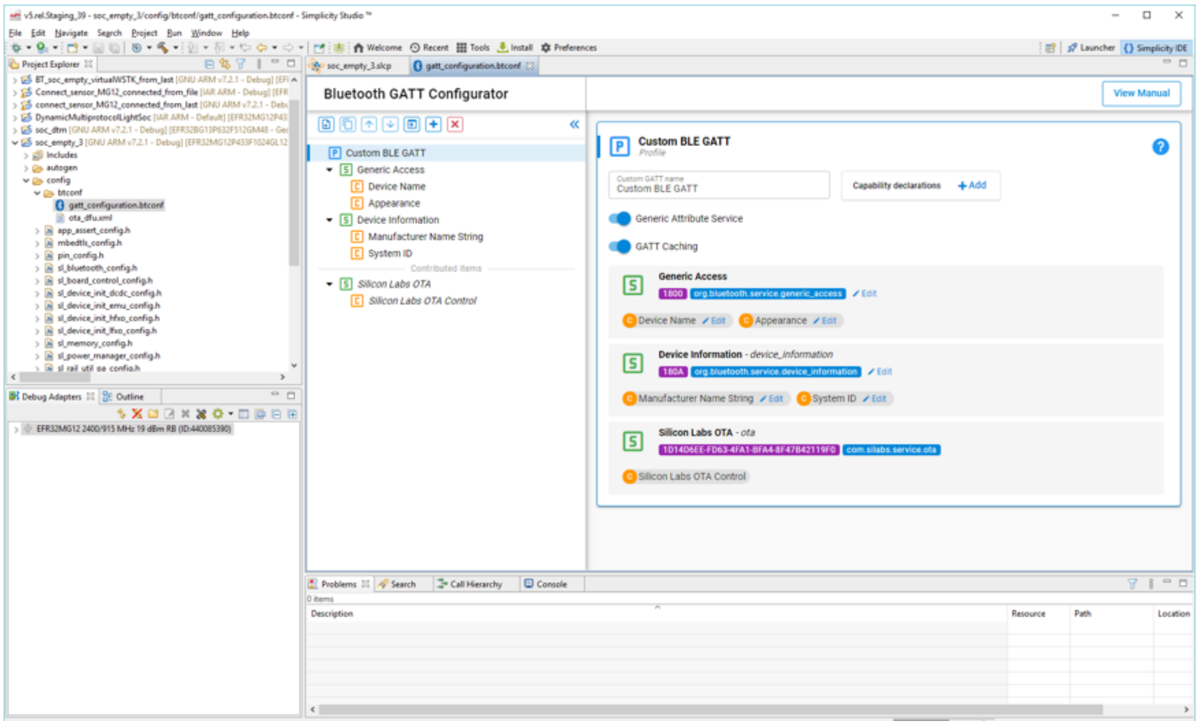

File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
soc_empty_3.slcp
gatt_configuration.btconf
soc_empty_3.pintool
DefaultMode Port I/O: PORTIO
125-pin BGA, 7x7 - (top view)
Zoom:
Configure
Search:
Pins Functions Peripherals
Debug Adapters
Outline
Problems Search Call Hierarchy Console
0 items
Description
Resource
Path
Location
soc_empty_3/soc_empty_3.pintool

Pin #	Pin Name	Function	Custom Pin Name	Software Component
M13	PA0			
L13	PA1			
L12	PA2			
K13	PA3			
K12	PA4			
J13	PA5	GPIO mode		Board Control : SL_BOAR.
J12	PA6			
J11	PA7			
H13	PA8			
H12	PA9			
F12	PB6			
E13	PB7			
E12	PB8			
D13	PB9			
D12	PB10	GPIO mode		Board Control : SL_BOAR.
D11	PB11	PTI_DCLK		RAIL Utility, Packet Trace
C13	PB12	PTI_DOUT		RAIL Utility, Packet Trace
C12	PB13	PTI_DFRAME		RAIL Utility, Packet Trace
C11	PB14			
C10	PB15			
A5	PC0			
B5	PC1			
C5	PC2			
A4	PC3			
B4	PC4			
A3	PC5			
B9	PC6			
A8	PC7			
B8	PC8			
A7	PC9			
B7	PC10			
A6	PC11			

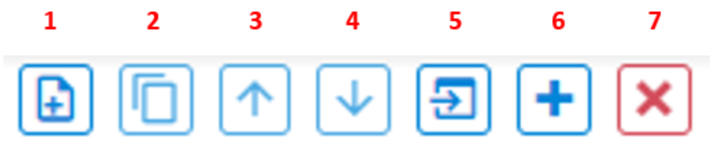
Double-click a Software Component to open the Component Editor and configure that function. Pin Tool does not autosave.

Bluetooth GATT Configurator

Bluetooth and Bluetooth Mesh projects are also configured with the Bluetooth GATT Configurator.



The Bluetooth GATT configurator menu allows you to add and remove services and characteristics.

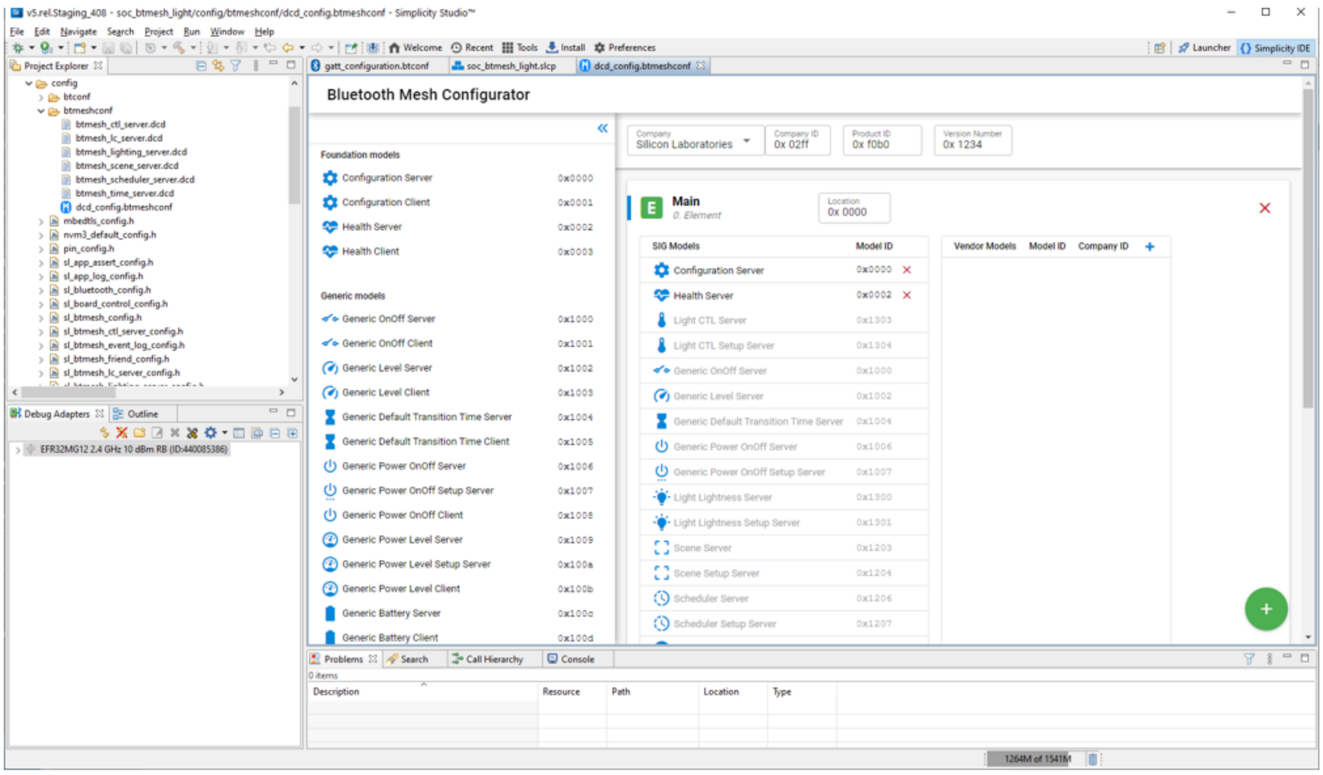


1. Add an item.
2. Duplicate the selected item.
3. Move the selected item up.
4. Move the selected item down.
5. Import a Bluetooth GATT database.
6. Add Predefined.
7. Delete the selected item.

To add a custom service, click **Profile (Custom BLE GATT)**, and then click **Add** (1). To add a custom characteristic, select a service and then click **Add** (1). To add a predefined service/characteristic click **Add Predefined** (6).

Bluetooth Mesh Configurator

Bluetooth Mesh project Device Composition Data is configured through the Bluetooth Mesh Configurator. The Device Composition Data is presented in three areas: device information, elements, and models.



Bluetooth Mesh Configurator

Company	Company ID	Product ID	Version Number
Silicon Laboratories	0x 02ff	0x f0b0	0x 1234

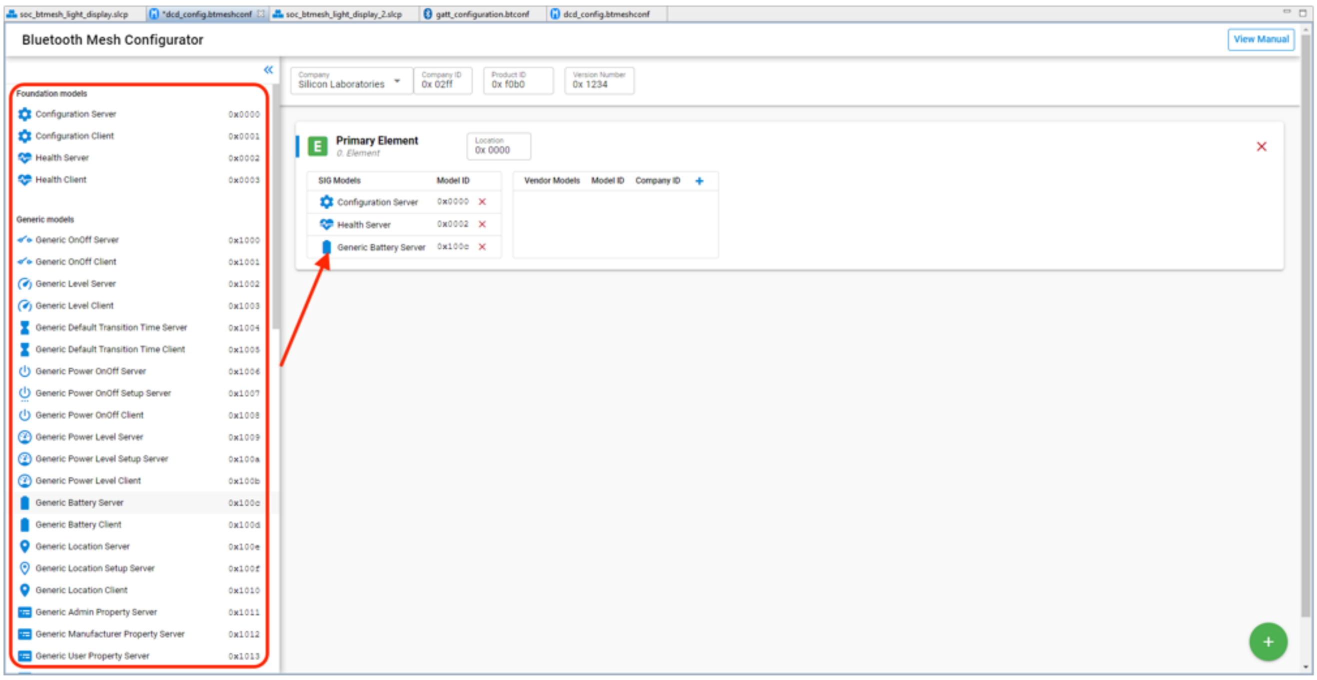
Main, 0. Element, Location 0x 0000

SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Light CTL Server	0x1303
Light CTL Setup Server	0x1304
Generic OnOff Server	0x1000
Generic Level Server	0x1002
Generic Default Transition Time Server	0x1004
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Light Lightness Server	0x1300
Light Lightness Setup Server	0x1301
Scene Server	0x1203
Scene Setup Server	0x1204
Scheduler Server	0x1206
Scheduler Setup Server	0x1207

Device information is determined by the company selected in the first field.

Each node has at minimum a primary element. To add more elements click the green + symbol in the bottom right.

The Bluetooth Mesh Configurator has editors for both SIG-adopted models and vendor models. SIG-adopted model components are provided but cannot be edited. You can, however, delete those models, and then add models to meet your needs. To delete a model, select it and click the red X symbol. To add a SIG-adopted model, drag the model from the left model pool to the SIG Models table in the correct element. A list of all the SIG-adopted models is displayed, and you can choose the one you want.



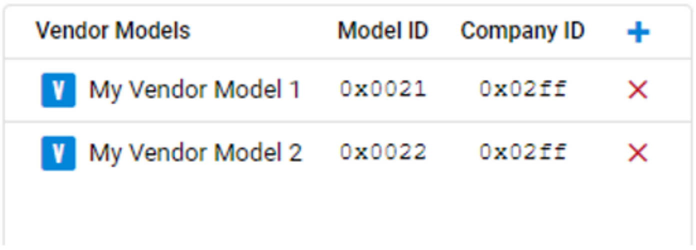
Bluetooth Mesh Configurator

Company	Company ID	Product ID	Version Number
Silicon Laboratories	0x 02ff	0x f0b0	0x 1234

Primary Element, 0. Element, Location 0x 0000

SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Generic Battery Server	0x100c

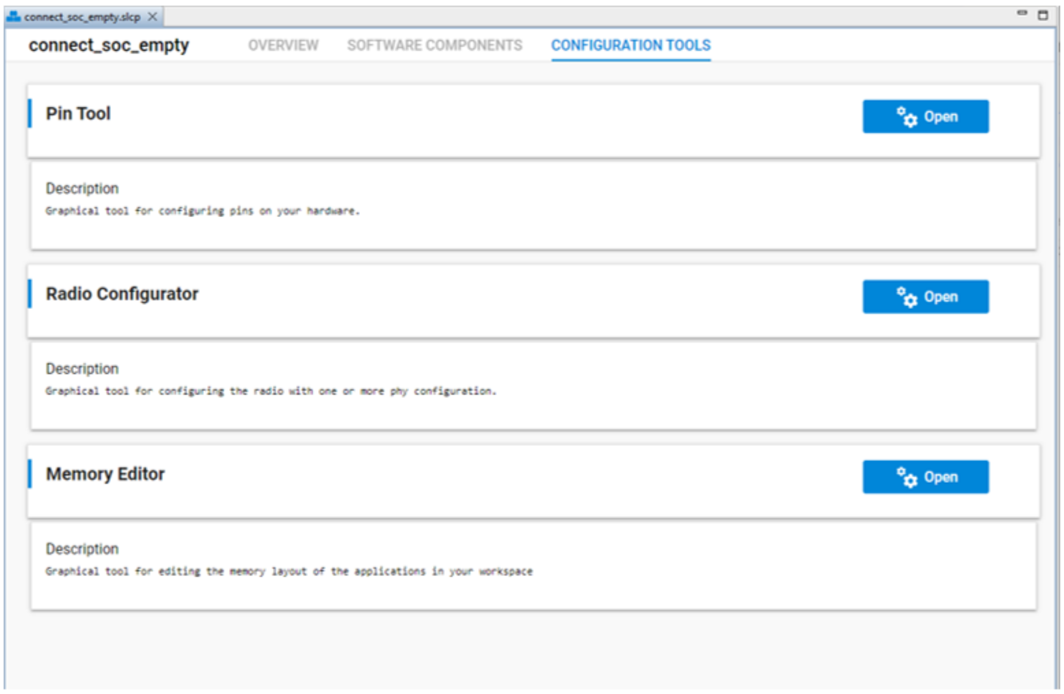
Vendor models give you more flexibility when developing products not covered by the SIG-adopted models. Vendors can define their own specification in these vendor models, including states, messages, and the associated behaviors. The vendor model editor is shown in the following figure.



Vendor Models	Model ID	Company ID
My Vendor Model 1	0x0021	0x02ff
My Vendor Model 2	0x0022	0x02ff

Proprietary Radio Configurator

The Radio Configurator is provided as part of the Proprietary SDK. Use the Radio Configurator to create standard or custom radio configurations for your RAIL-based radio applications.



Pin Tool

Description
Graphical tool for configuring pins on your hardware.

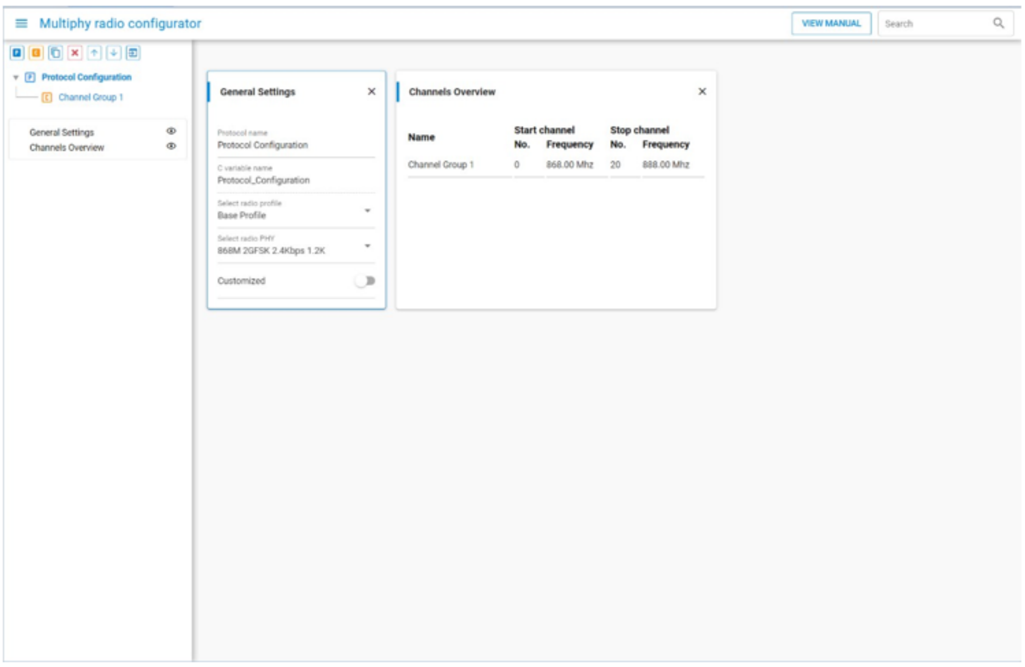
Radio Configurator

Description
Graphical tool for configuring the radio with one or more phy configuration.

Memory Editor

Description
Graphical tool for editing the memory layout of the applications in your workspace

The parameters in the Radio Configurator are arranged in cards, some of which are grouped together. Different radio profiles offer different views and parameter sets.



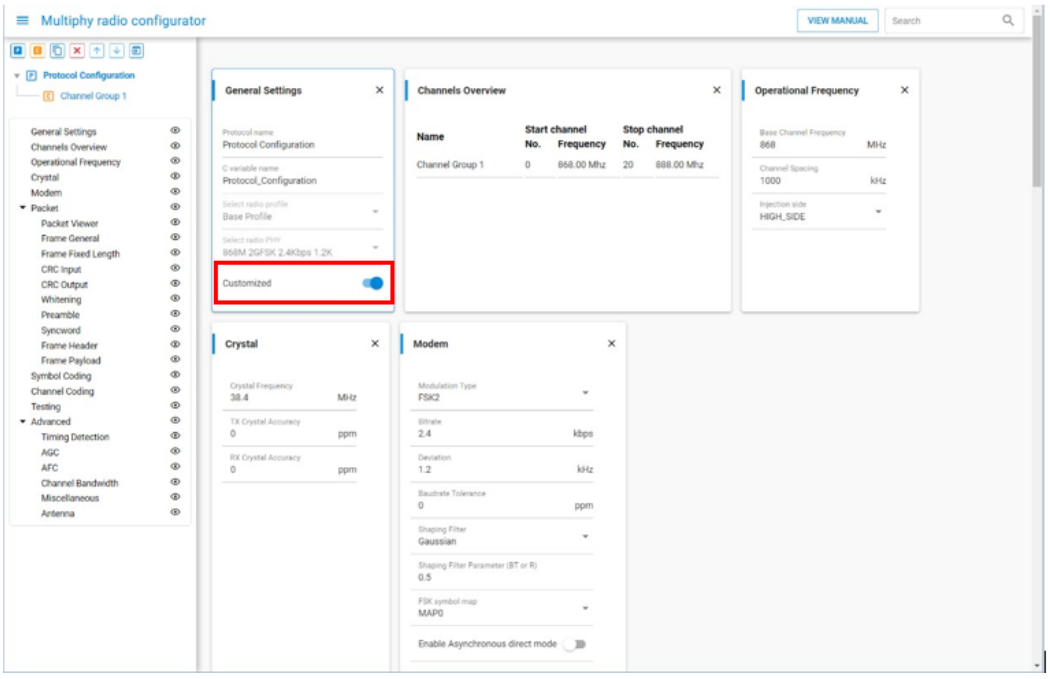
General Settings

Protocol name: Protocol Configuration
C variable name: Protocol_Configuration
Select radio profile: Base Profile
Select radio PHY: 868M 2GFSK 2.4Kbps 1.2K
Customized

Channels Overview

Name	Start channel No.	Start channel Frequency	Stop channel No.	Stop channel Frequency
Channel Group 1	0	868.00 Mhz	20	888.00 Mhz

1. In the General Settings card, select a radio profile in the **Select radio profile** drop-down menu. A radio profile may be any supported radio link technology. These technologies can be bound by standards (for example the Sigfox or WMBus protocols) or can be fully customized. The fully customizable profile is called the "Base Profile".
2. Select a radio PHY (radio configuration) in the **Select a radio PHY** dropdown list. Each profile has "built-in" configurations ready to use.
3. Review and update the profile options. By default, no changes are allowed; fields are grayed out. To enable customization, use the **Customized** switch on the General Settings card. This allows access to all the parameters defined by the profile.



General Settings

Protocol name: Protocol Configuration
C variable name: Protocol_Configuration
Select radio profile: Base Profile
Select radio PHY: 868M 2GFSK 2.4Kbps 1.2K
Customized

Channels Overview

Name	Start channel No.	Start channel Frequency	Stop channel No.	Stop channel Frequency
Channel Group 1	0	868.00 Mhz	20	888.00 Mhz

Operational Frequency

Base Channel Frequency: 868 MHz
Channel Spacing: 1000 kHz
Injection side: HIGH_SIDE

Crystal

Crystal Frequency: 38.4 MHz
TX Crystal Accuracy: 0 ppm
RX Crystal Accuracy: 0 ppm

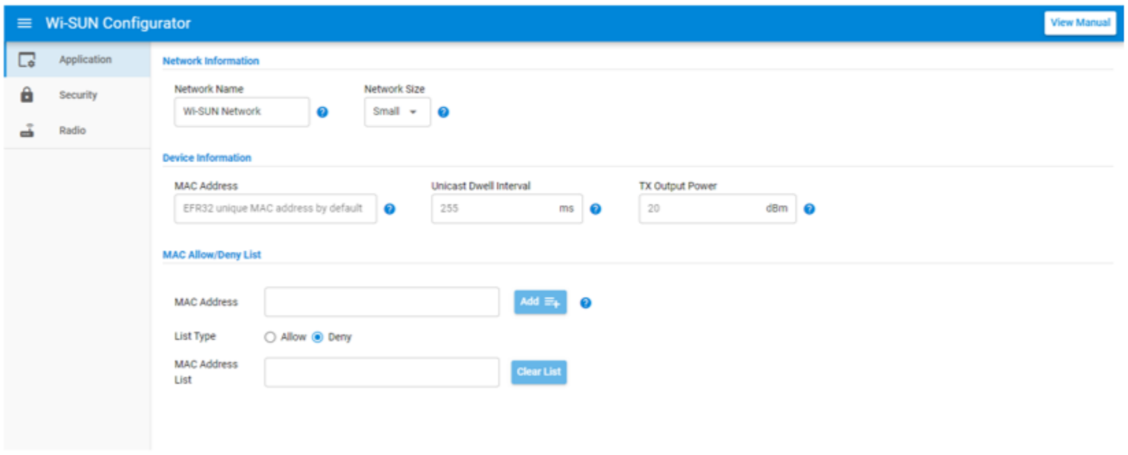
Modem

Modulation Type: FSK2
Bitrate: 2.4 kbps
Deviation: 1.2 kHz
Baudrate Tolerance: 0 ppm
Shaping Filter: Gaussian
Shaping Filter Parameter (BT or R): 0.5
FSK symbol map: MAP0
Enable Asynchronous direct mode

Wi-SUN Configurator

The Wi-SUN Configurator is provided as part of the Wi-SUN SDK. Use it to configure the main settings of the Wi-SUN application through three panels: Application, Security, and Radio. For some projects only the Radio panel is available. The

Wi-SUN Configurator tab is available when a project is created or can be displayed by opening the project file /config/wisun/wisun_settings.wisunconf.


Wi-SUN Configurator

View Manual

Application | Security | Radio

Network Information

Network Name: Wi-SUN Network

Network Size: Small

Device Information

MAC Address: EFR32 unique MAC address by default

Unicast Dwell Interval: 255 ms

TX Output Power: 20 dBm

MAC Allow/Deny List

MAC Address

Add

List Type: ( ) Allow [✓ Deny]

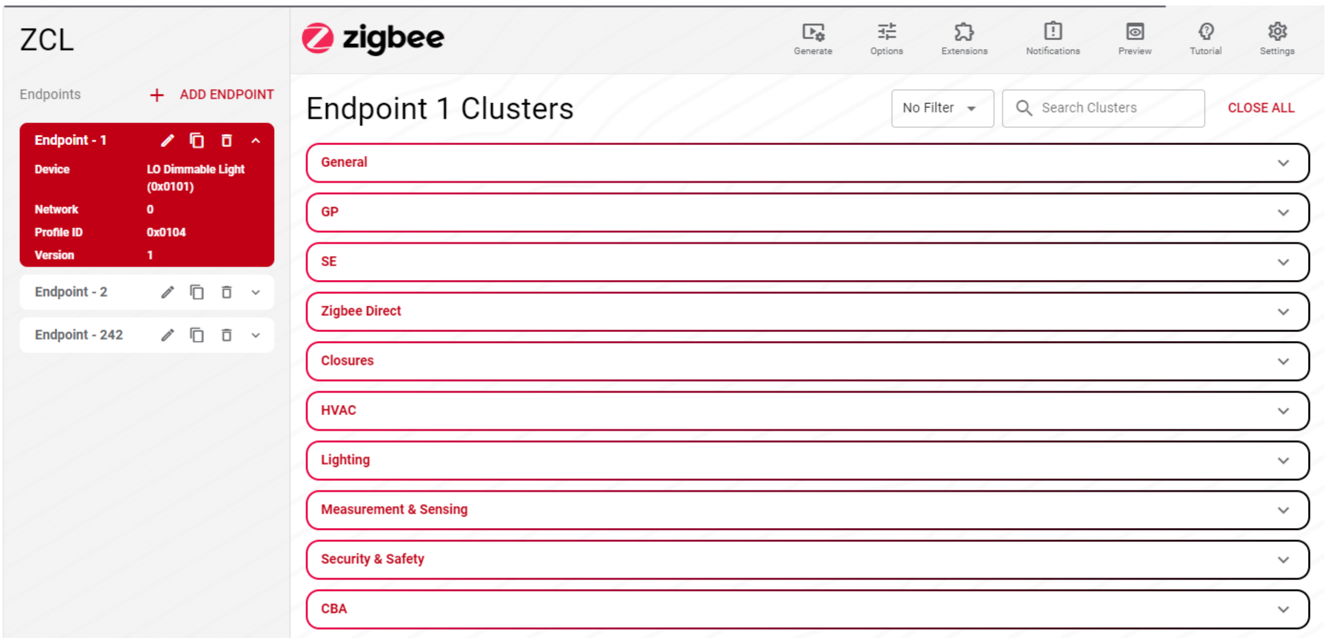
MAC Address List

Clear List

ZCL Advanced Platform

The ZCL Advanced Platform (ZAP) is provided as a configuration tool for both Zigbee and Matter projects. Use it to manage endpoints, clusters, and commands.

The interface, which differs slightly for Matter and Zigbee, is based on adding or modifying endpoints. Click an endpoint to open the configuration editor.



ZCL

Endpoints — ADD ENDPOINT

Endpoint - 1	
Device	LO Dimmable Light (0x0101)
Network	0
Profile ID	0x0104
Version	1

Endpoint - 2

Endpoint - 242

Endpoint 1 Clusters — No Filter — Search Clusters — CLOSE ALL

- General
- GP
- SE
- Zigbee Direct
- Closures
- HVAC
- Lighting
- Measurement & Sensing
- Security & Safety
- CBA

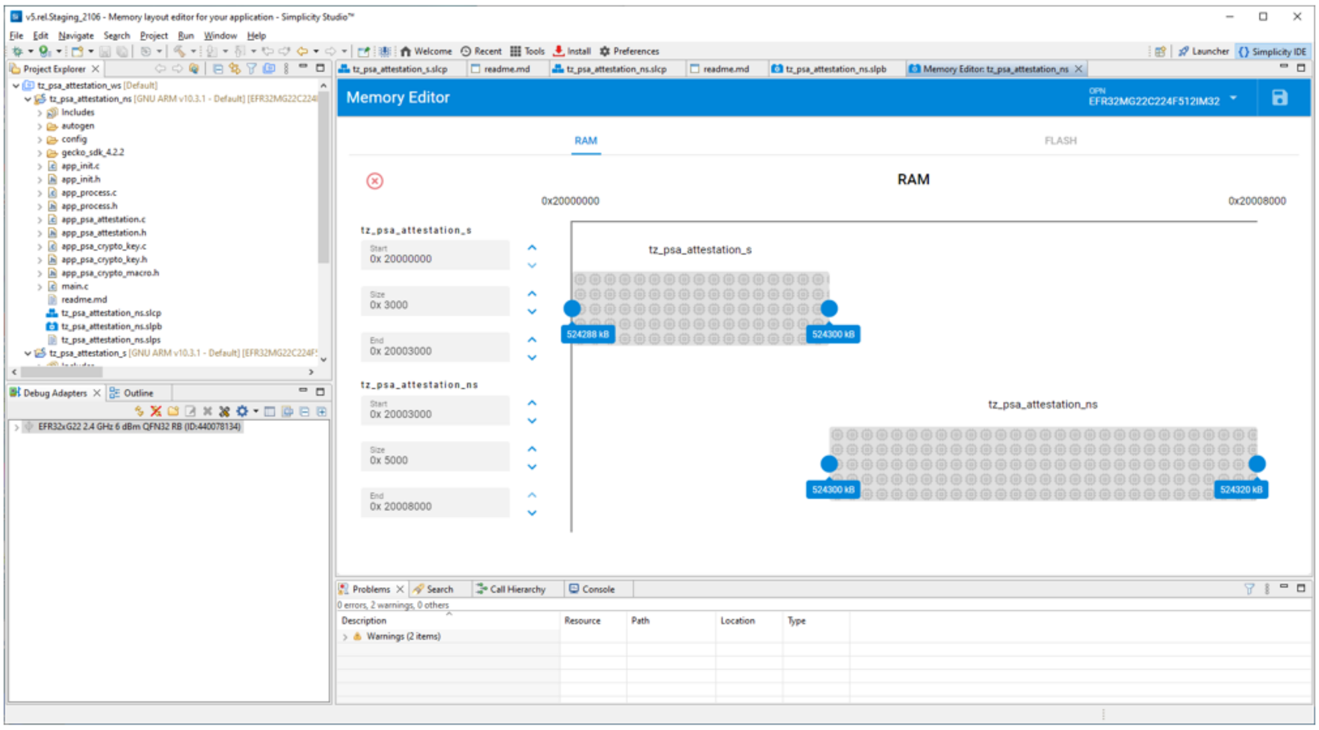
You can add and configure clusters as needed. Configuration changes made through the Zigbee Cluster Configurator for a Zigbee project are saved to <project-folder>\config\zcl\zcl_config.zap. For a Matter project the configuration is saved to <project-folder>\config\zap\<name>.zap. When you save the file, the Zigbee Cluster Configurator not only saves the .zap file into your project, but also automatically generates all the files required for your application.

Solutions

A Solution is a combinations of projects that can be compiled, debugged, and flashed together. Solutions are also called workspaces in some contexts. The projects in a solution can be configured individually. When the solution is built, the Post-Build Editor is used to combine the project images into a single downloadable .s37 image. Applications that make use of the

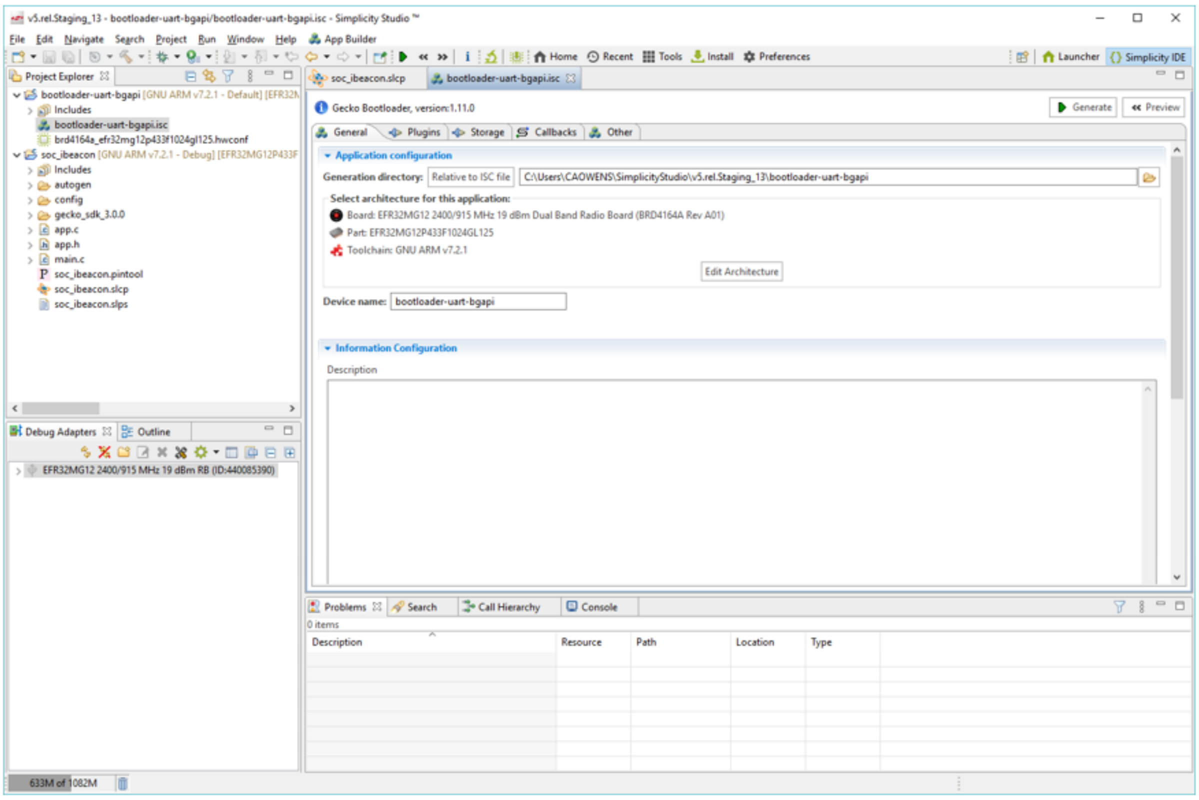
Series 2 TrustZone feature use solutions to combined the secure and non-secure TrustZone projects. Solutions can also be used to debug an application and its bootloader together.

The Memory Editor is a graphical tool for editing the flash and RAM memory layout of the applications in a solution. Not all example projects have default settings included at this time, which results in a warning when you open the editor. Save the default values that the tool calculates as your starting point.



AppBuilder Projects

AppBuilder Projects are configured by modifying parameters in various tabs, especially the PLUGINS tab. See Developing with AppBuilder for details.


Gecko Bootloader, version:1.11.0

Application configuration

Generation directory: Relative to ISC file C:\Users\CAOWENS\SimplicityStudio\v5.rel.Staging_13\bootloader-uart-bgapi

Select architecture for this application:

Board: EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A01)

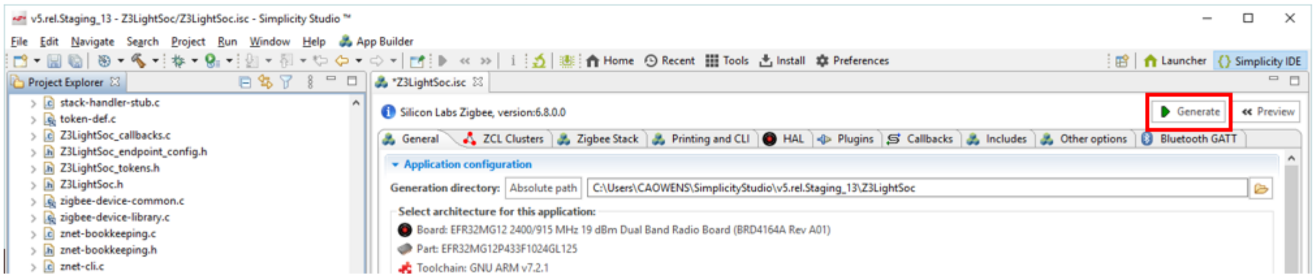
Part: EFR32MG12P433F1024GL125

Toolchain: GNU ARM v7.2.1

Device name: bootloader-uart-bgapi

Information Configuration

When you have configured the project, click **Generate** to create project files. Build the application image and flash it to your target device as in Building and Flashing.


Silicon Labs Zigbee, version:6.8.0.0

Application configuration

Generation directory: Absolute path C:\Users\CAOWENS\SimplicityStudio\v5.rel.Staging_13\Z3LightSoc

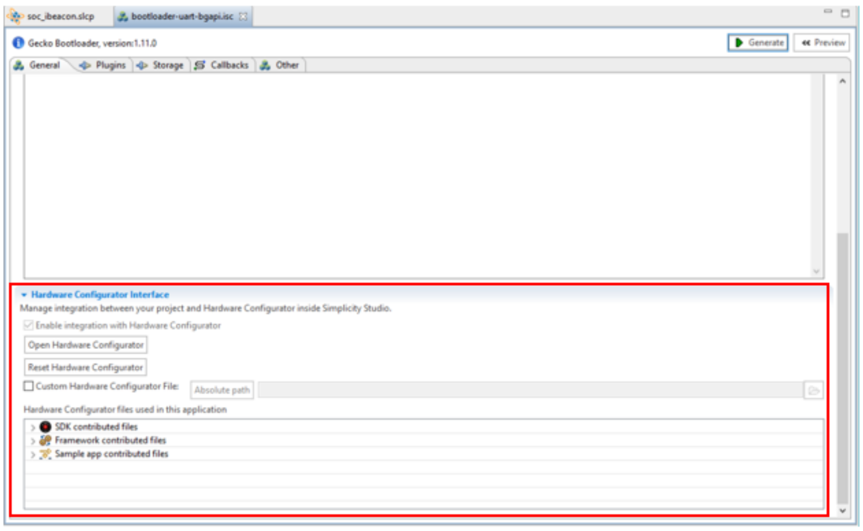
Select architecture for this application:

Board: EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A01)

Part: EFR32MG12P433F1024GL125

Toolchain: GNU ARM v7.2.1

To modify the target device's pin use and parameters, use the Hardware Configurator available on the HAL tab. Note that, although the interface is similar to the 8-bit Hardware Configurator, this is a different tool.


soc_ibeacon.slcp
bootloader-uart-bgapi.isc
Gecko Bootloader, version:1.11.0
Generate
Preview
General
Plugins
Storage
Callbacks
Other
Hardware Configurator Interface
Manage integration between your project and Hardware Configurator inside Simplicity Studio.
Enable integration with Hardware Configurator
Open Hardware Configurator
Reset Hardware Configurator
Custom Hardware Configurator File:
Absolute path
Hardware Configurator files used in this application
SDK contributed files
Framework contributed files
Sample app contributed files

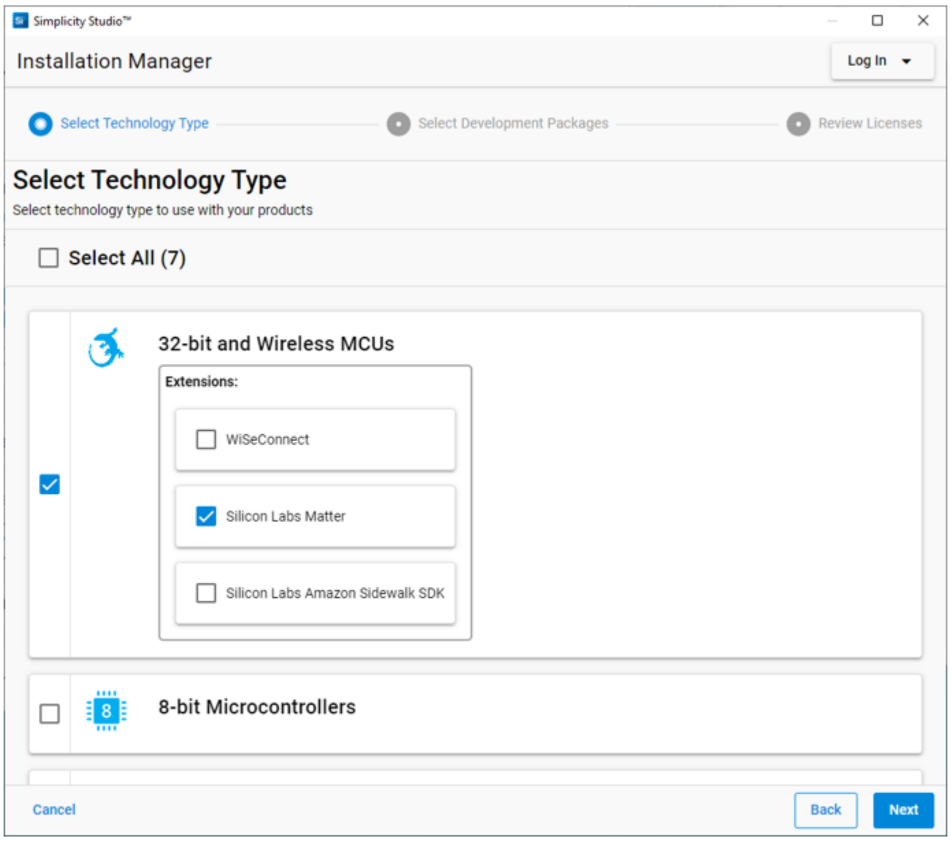
Changes made through the Hardware Configurator are stored in a board-specific .hwconf file.

Install SDK Extensions

Install SDK Extensions

An SDK Extension is an entity specific to developing for 32-bit devices using Project Configurator and other Silicon Labs Configurator (SLC)-based tools. It is a collection of components and other items, such as example files. The SDK extension has dependencies on the parent SDK, which must be installed first. SDK extensions can be used to control access to certain functions, or to contain customer-created components and other items to be maintained separately from the Silicon Labs SDK.

SDK extensions are installed as part of the GSDK through the standard installation dialog.



Select Technology Type

Select technology type to use with your products

☐ Select All (7)

☑ 32-bit and Wireless MCUs

Extensions:

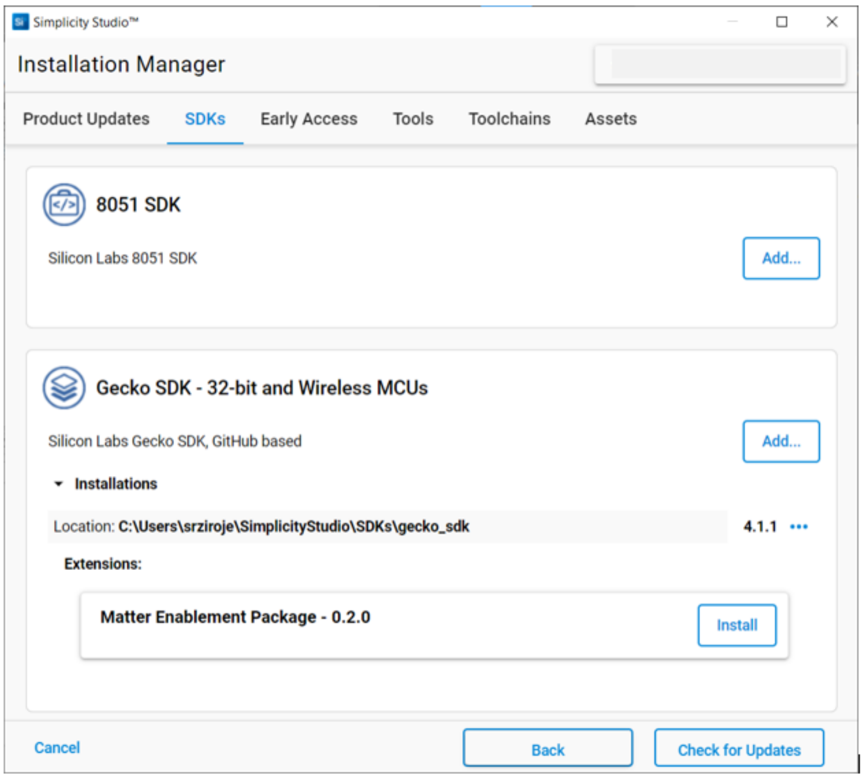
☐ WiSeConnect

☑ Silicon Labs Matter

☐ Silicon Labs Amazon Sidewalk SDK

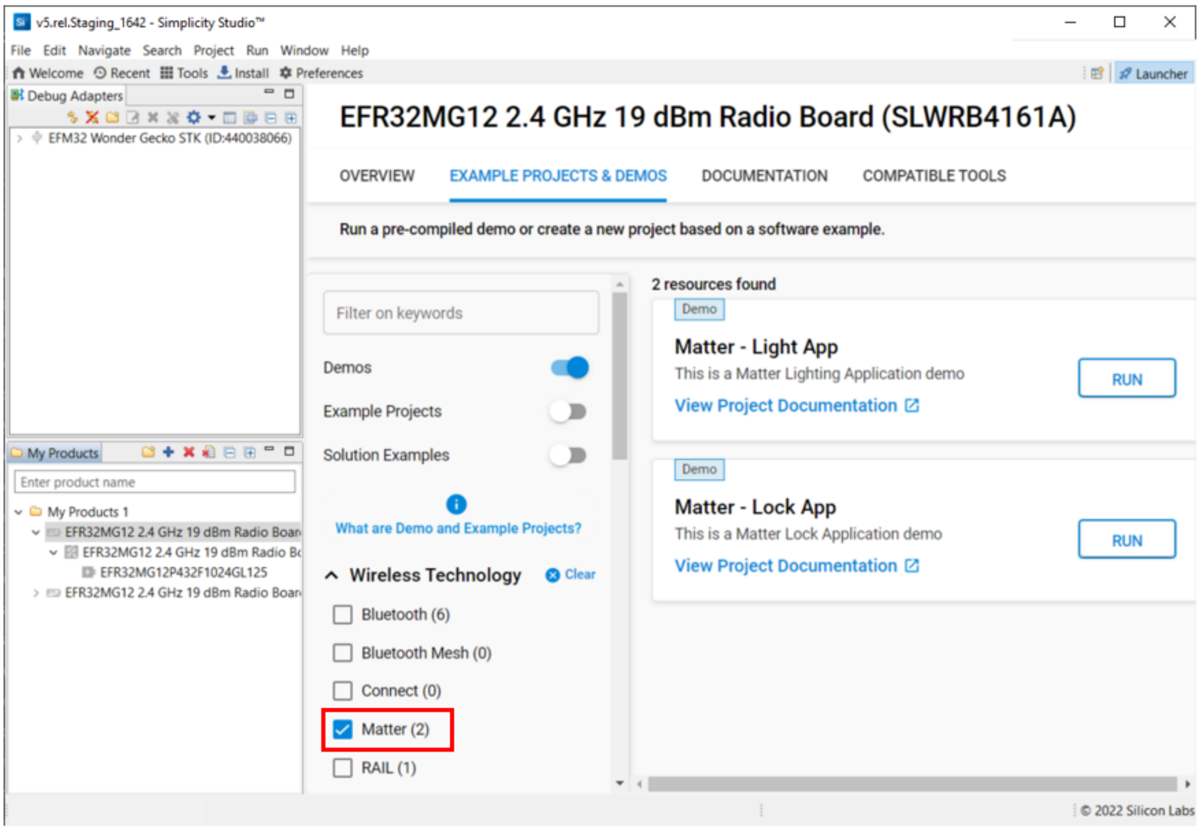
☐ 8-bit Microcontrollers

If you already have a GSDK installed but did not installed the extension, you can install it by clicking **Install** on the toolbar, and then clicking **Manage Installed Packages**. Open the SDK tab, and click **Install** next to the extension name.



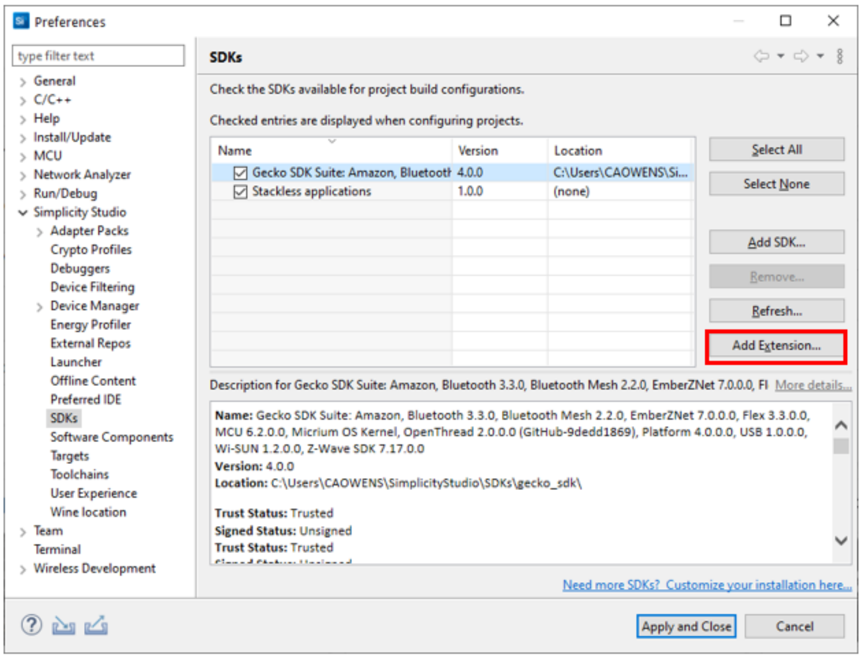
See update information in About the Launcher for additional details.

Once you have added an SDK extension, SSv5 treats it like any other SDK, such as showing it in the filter for example projects.


v5.rel.Staging_1642 - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher
Debug Adapters
EFM32 Wonder Gecko STK (ID:440038066)
My Products
Enter product name
My Products 1
EFR32MG12 2.4 GHz 19 dBm Radio Boar
EFR32MG12 2.4 GHz 19 dBm Radio Bo
EFR32MG12P432F1024GL125
EFR32MG12 2.4 GHz 19 dBm Radio Boar
EFR32MG12 2.4 GHz 19 dBm Radio Board (SLWRB4161A)
OVERVIEW
EXAMPLE PROJECTS & DEMOS
DOCUMENTATION
COMPATIBLE TOOLS
Run a pre-compiled demo or create a new project based on a software example.
Filter on keywords
Demos
Example Projects
Solution Examples
What are Demo and Example Projects?
Wireless Technology
Clear
Bluetooth (6)
Bluetooth Mesh (0)
Connect (0)
Matter (2)
RAIL (1)
2 resources found
Demo
Matter - Light App
This is a Matter Lighting Application demo
View Project Documentation
RUN
Demo
Matter - Lock App
This is a Matter Lock Application demo
View Project Documentation
RUN
© 2022 Silicon Labs

If you have been given an extension package that is not yet part of the standard installation, you can install it using using the following procedure.

1. Open Preferences > Simplicity Studio > SDKs either from Preferences on the toolbar or by selecting **Manage SDKs** from the Launcher perspective OVERVIEW tab. Select the parent SDK and click **Add Extensions**.



SDKs

Check the SDKs available for project build configurations.

Checked entries are displayed when configuring projects.

Name	Version	Location
☑ Gecko SDK Suite: Amazon, Bluetooth	4.0.0	C:\Users\CAOWENS\Si...
☑ Stackless applications	1.0.0	(none)

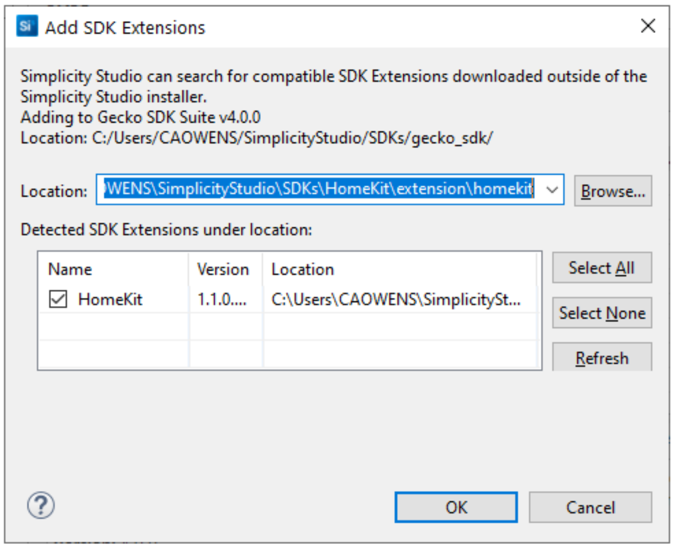
Description for Gecko SDK Suite: Amazon, Bluetooth 3.3.0, Bluetooth Mesh 2.2.0, EmberZNet 7.0.0.0, Fl More details...

Name: Gecko SDK Suite: Amazon, Bluetooth 3.3.0, Bluetooth Mesh 2.2.0, EmberZNet 7.0.0.0, Flex 3.3.0.0, MCU 6.2.0.0, Micrium OS Kernel, OpenThread 2.0.0.0 (GitHub-9dedd1869), Platform 4.0.0.0, USB 1.0.0.0, Wi-SUN 1.2.0.0, Z-Wave SDK 7.17.0.0
Version: 4.0.0
Location: C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk\

Trust Status: Trusted
Signed Status: Unsigned
Trust Status: Trusted

Need more SDKs? Customize your installation here...

2. In the Add SDK Extensions dialog, browse to the extension directory. If it has a valid SDK extension, SSv5 detects it. Click **OK**.



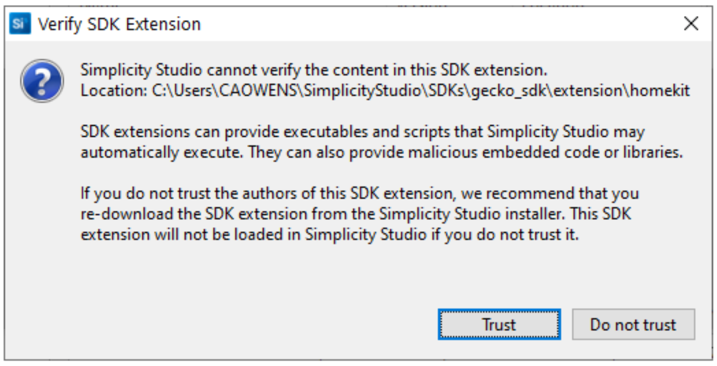
Simplicity Studio can search for compatible SDK Extensions downloaded outside of the Simplicity Studio installer.
Adding to Gecko SDK Suite v4.0.0
Location: C:/Users/CAOWENS/SimplicityStudio/SDKs/gecko_sdk/

Location: WENS\SimplicityStudio\SDKs\HomeKit\extension\homekit

Detected SDK Extensions under location:

Name	Version	Location
☑ HomeKit	1.1.0....	C:\Users\CAOWENS\SimplicitySt...

3. You may be asked to trust the SDK extension. If you do, click **Trust**.

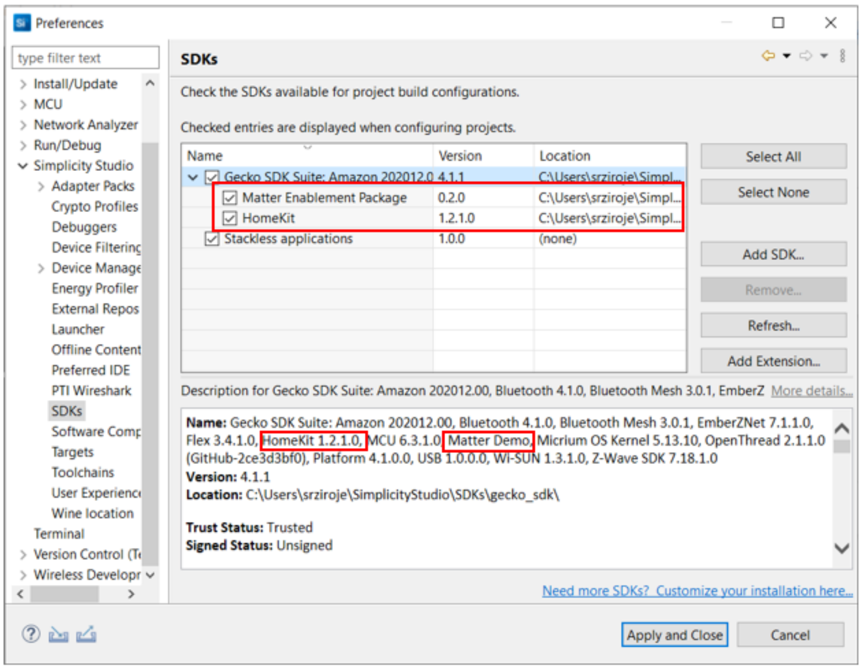


Simplicity Studio cannot verify the content in this SDK extension.
Location: C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk\extension\homekit

SDK extensions can provide executables and scripts that Simplicity Studio may automatically execute. They can also provide malicious embedded code or libraries.

If you do not trust the authors of this SDK extension, we recommend that you re-download the SDK extension from the Simplicity Studio installer. This SDK extension will not be loaded in Simplicity Studio if you do not trust it.

4. The extension is now displayed under the GSDK, and in the list of GSDK components. Click **Apply and Close**.



SDKs

Check the SDKs available for project build configurations.

Checked entries are displayed when configuring projects.

Name	Version	Location
☑ Gecko SDK Suite: Amazon 202012.0	4.1.1	C:\Users\srziroje\Simpl...
☑ Matter Enablement Package	0.2.0	C:\Users\srziroje\Simpl...
☑ HomeKit	1.2.1.0	C:\Users\srziroje\Simpl...
☑ Stackless applications	1.0.0	(none)

Description for Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.1.0, Bluetooth Mesh 3.0.1, EmberZ More details...

Name: Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.1.0, Bluetooth Mesh 3.0.1, EmberZNet 7.1.1.0, Flex 3.4.1.0, HomeKit 1.2.1.0, MCU 6.3.1.0 Matter Demo, Micrium OS Kernel 5.13.10, OpenThread 2.1.1.0 (GitHub-2ce3d3bf0), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.1.0, Z-Wave SDK 7.18.1.0
Version: 4.1.1
Location: C:\Users\srziroje\SimplicityStudio\SDKs\gecko_sdk\

Trust Status: Trusted
Signed Status: Unsigned

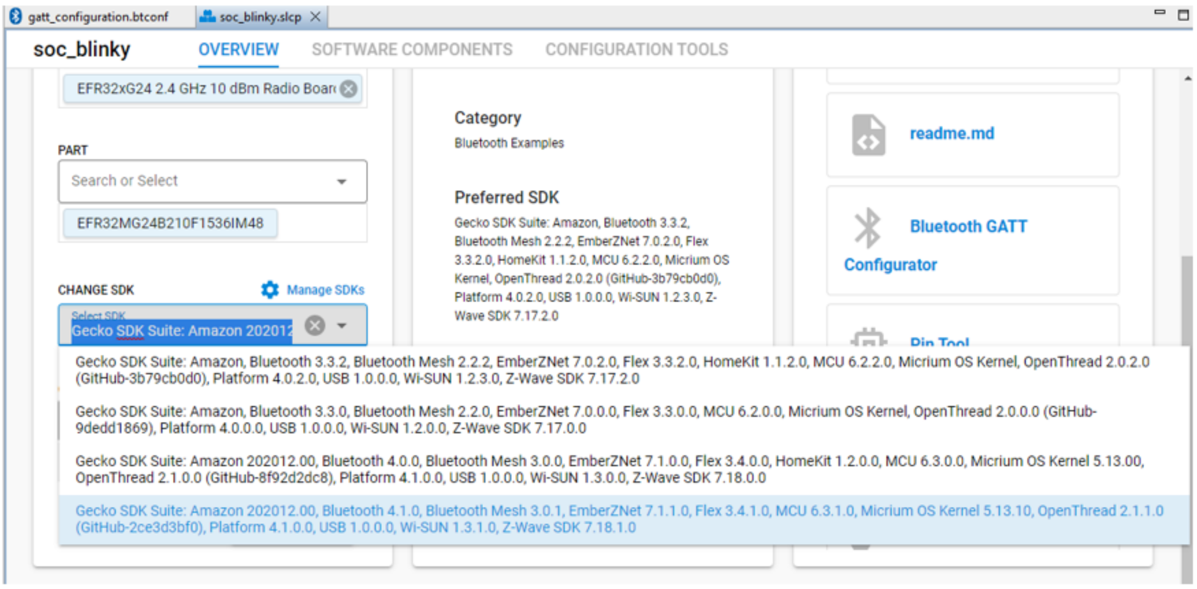
Upgrade a Project to a New Software Version

Upgrading a Project to a New Software Version

Upgrading to a New GSDK

If a project was created with an older GSDK, and a new GSDK version is installed, the upgrade process starts differently depending on if the existing GSDK itself was upgraded to the latest version or if the latest GSDK was installed as a separate GSDK instance.

1. If a parallel instance of the new GSDK was installed, then the project upgrade must be started from the Project Configurator Overview tab, Target and Tool Settings card. Click **Change Target/SDK/Generators**. Click the **CHANGE SDK** drop down and select the desired GSDK.



EFR32xG24 2.4 GHz 10 dBm Radio Boar

PART

Search or Select

EFR32MG24B210F1536IM48

CHANGE SDK

Select SDK

Gecko SDK Suite: Amazon 202012

Category

Bluetooth Examples

Preferred SDK

Gecko SDK Suite: Amazon, Bluetooth 3.3.2, Bluetooth Mesh 2.2.2, EmberZNet 7.0.2.0, Flex 3.3.2.0, HomeKit 1.1.2.0, MCU 6.2.2.0, Micrium OS Kernel, OpenThread 2.0.2.0 (GitHub-3b79cb0d0), Platform 4.0.2.0, USB 1.0.0.0, Wi-SUN 1.2.3.0, Z-Wave SDK 7.17.2.0

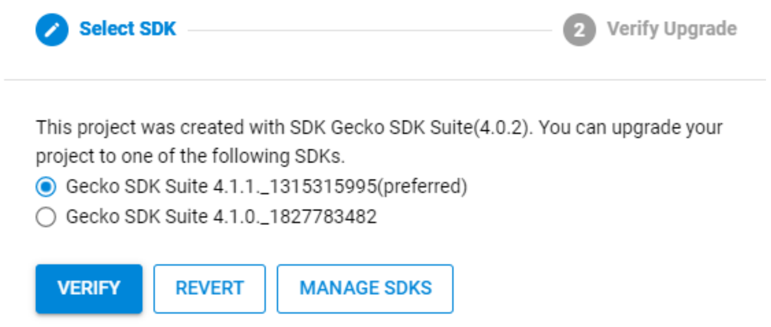
Gecko SDK Suite: Amazon, Bluetooth 3.3.2, Bluetooth Mesh 2.2.2, EmberZNet 7.0.2.0, Flex 3.3.2.0, HomeKit 1.1.2.0, MCU 6.2.2.0, Micrium OS Kernel, OpenThread 2.0.2.0 (GitHub-3b79cb0d0), Platform 4.0.2.0, USB 1.0.0.0, Wi-SUN 1.2.3.0, Z-Wave SDK 7.17.2.0

Gecko SDK Suite: Amazon, Bluetooth 3.3.0, Bluetooth Mesh 2.2.0, EmberZNet 7.0.0.0, Flex 3.3.0.0, MCU 6.2.0.0, Micrium OS Kernel, OpenThread 2.0.0.0 (GitHub-9dedd1869), Platform 4.0.0.0, USB 1.0.0.0, Wi-SUN 1.2.0.0, Z-Wave SDK 7.17.0.0

Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.0.0, Bluetooth Mesh 3.0.0, EmberZNet 7.1.0.0, Flex 3.4.0.0, HomeKit 1.2.0.0, MCU 6.3.0.0, Micrium OS Kernel 5.13.00, OpenThread 2.1.0.0 (GitHub-8f92d2dc8), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.0.0, Z-Wave SDK 7.18.0.0

Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.1.0, Bluetooth Mesh 3.0.1, EmberZNet 7.1.1.0, Flex 3.4.1.0, MCU 6.3.1.0, Micrium OS Kernel 5.13.10, OpenThread 2.1.1.0 (GitHub-2ce3d3bf0), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.1.0, Z-Wave SDK 7.18.1.0

A Verify SDK dialog is displayed.

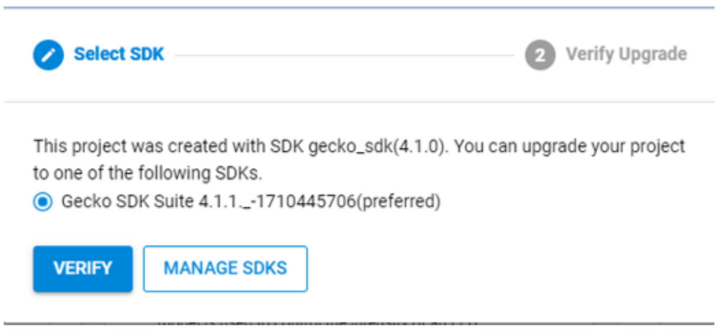


This project was created with SDK Gecko SDK Suite(4.0.2). You can upgrade your project to one of the following SDKs.

[✓ Gecko SDK Suite 4.1.1._1315315995(preferred)]

Gecko SDK Suite 4.1.0._1827783482

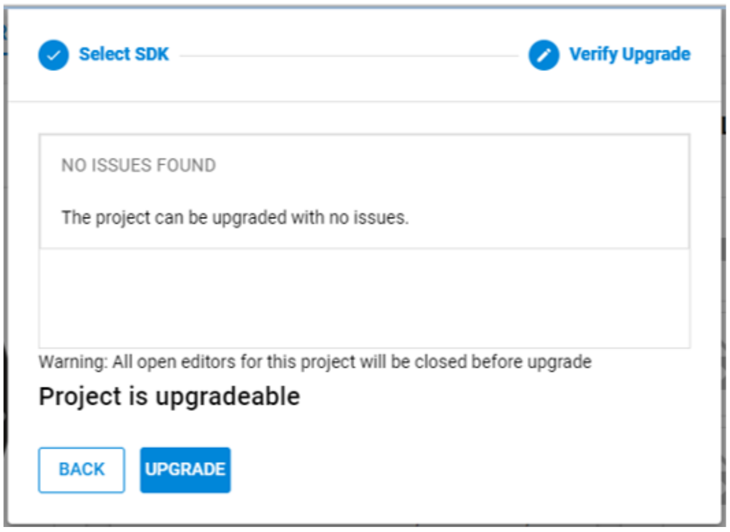
2. If the GSDK was upgraded to the latest version, when the project's .slcp file is opened the Verify SDK dialog is displayed immediately.



This project was created with SDK gecko_sdk(4.1.0). You can upgrade your project to one of the following SDKs.

(●) Gecko SDK Suite 4.1.1._-1710445706(preferred)

From this point the process is the same. The displayed preferred Gecko SDK is the latest version. Click **Verify**. The Gecko SDK and Advanced configurator upgrade scripts (if any exist) are run and then a dialog something like the following is displayed. The dialog may include additional messages and warnings.



NO ISSUES FOUND

The project can be upgraded with no issues.

Warning: All open editors for this project will be closed before upgrade

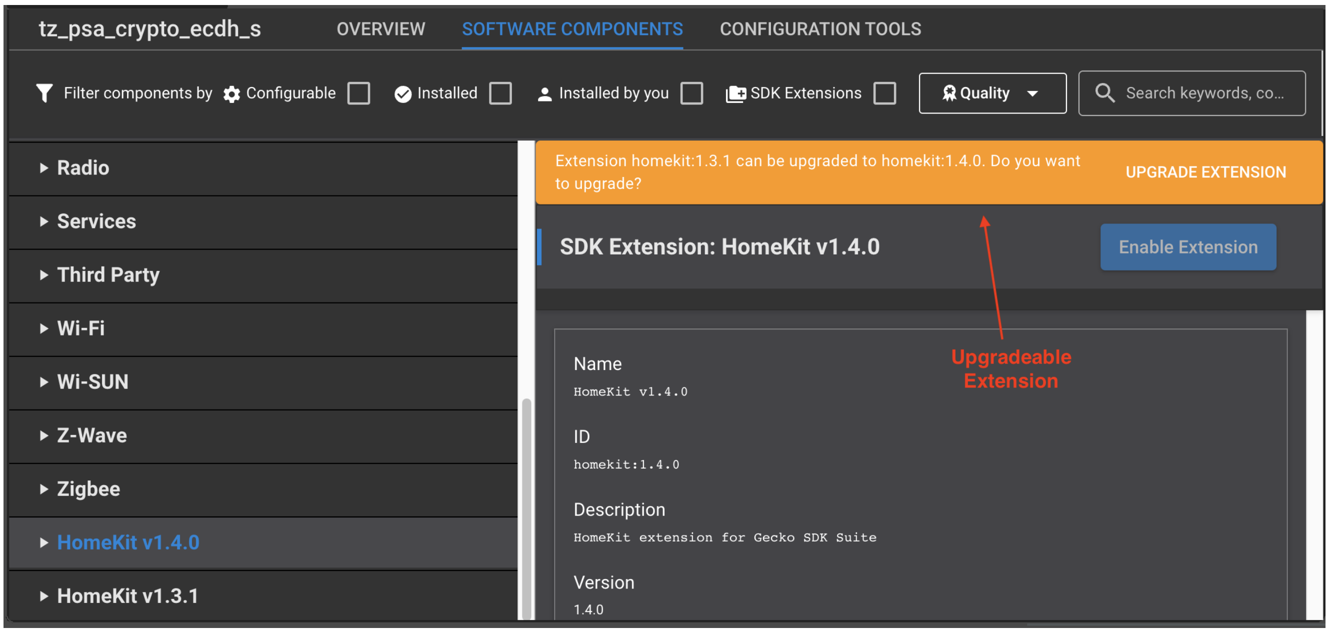
Project is upgradeable

Click **Upgrade**. The results of the upgrade are displayed. After the upgrade finishes, build the project and then manually resolve any issues that occur due to SDK changes. Reviewing the release notes for the SDK and Gecko Platform is recommended to see what new features are available. These are available through the Launcher perspective's Documentation tab (filter on Release Notes) or on silabs.com at https://www.silabs.com/developers/gecko-software-development-kit.

Upgrading to a New Extension

Simplicity Studio also checks if there are any missing extensions required by the project. If so, Simplicity Studio presents a similar dialog to the GSDK verify dialog with either an available upgradeable extension or an error if no upgradeable extension can be found.

Extensions can be upgraded independently of the GSDK. All available extensions will appear in the Software Components tab. When viewing a newer extension, Simplicity Studio will check if you can upgrade to it and presents an Upgrade Extension notification instead of the standard Install Extension. If selected, this will launch the Extension Upgrade dialog.



Extension homekit:1.3.1 can be upgraded to homekit:1.4.0. Do you want to upgrade?

SDK Extension: HomeKit v1.4.0

Name
HomeKit v1.4.0

ID
homekit:1.4.0

Description
HomeKit extension for Gecko SDK Suite

Version
1.4.0

If you click **UPGRADE EXTENSION**, a verification dialog is displayed similar to that for the SDK.

Project Migration

Project Migration

Moving to the Component-Based Architecture

In SSv5 version 5.3, the Gecko Bootloader, Zigbee, and Z-Wave changed to the Silicon Labs Configurator component-based architecture. For more information on transitioning these projects from earlier versions of SSv5 see:

- Zigbee AN1301: Transitioning from Zigbee EmberZNet SDK 6.x to SDK 7.x
- Gecko Bootloader AN1326: Transitioning to the Updated Gecko Bootloader in GSDK 4.0 and Higher

Migrate from Simplicity Studio 4 to Simplicity Studio 5

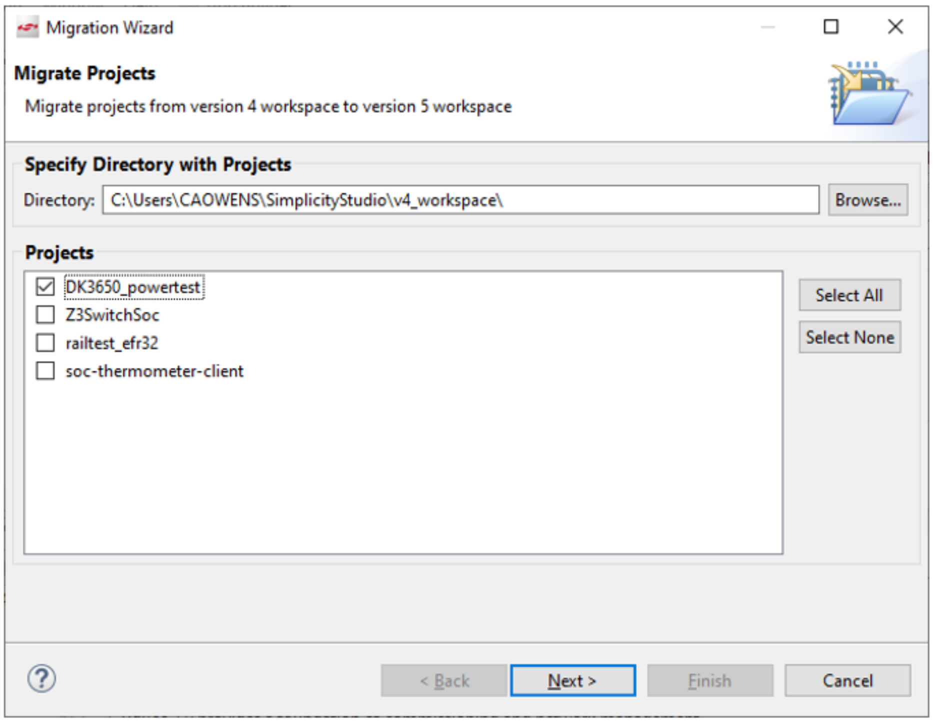
The process to migrate a project from Simplicity Studio® 4 (SSv4) to SSv5 depends on the type of project.

If you are migrating a Bootloader, EFM32, or EFM8 project, use the **Migrate Project** tool. If you are migrating a Z-Wave, Bluetooth/Bluetooth Mesh, Proprietary Flex, or Zigbee project, follow the instructions in the specified documents.

Bootloader, EFM32, and EFM8 Projects

Bootloader note: Use this procedure as a first step in migration.

Click the **Tools** toolbar button to open the Tools dialog. Select **Migrate Projects** and click **OK**. Select the project to be migrated and click **Next**.



Migrate projects from version 4 workspace to version 5 workspace

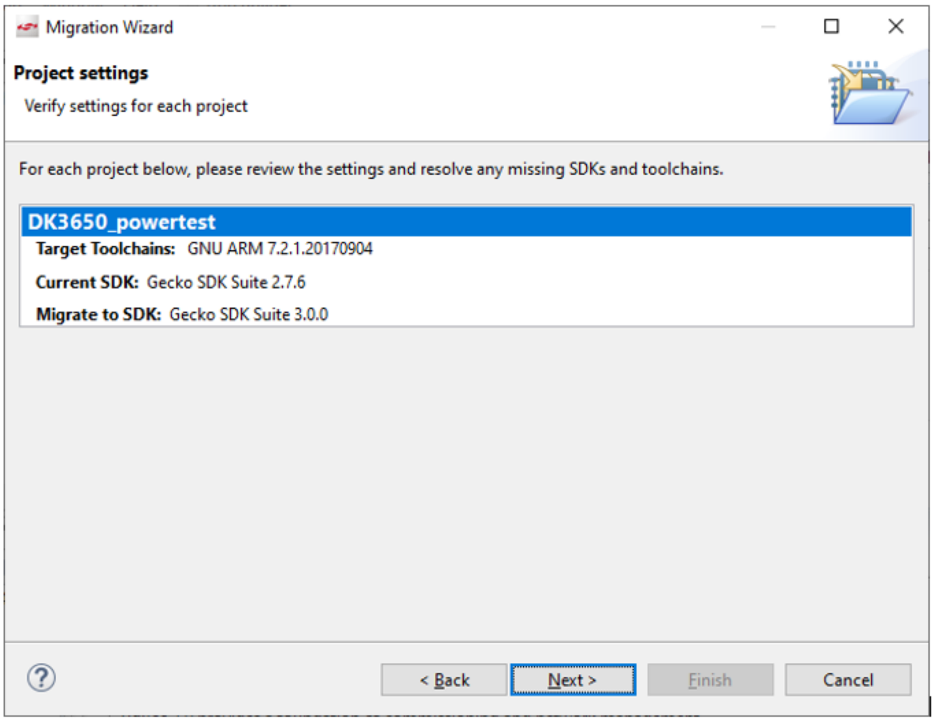
Specify Directory with Projects

Directory: C:\Users\CAOWENS\SimplicityStudio\v4_workspace\

Projects

- [✓] DK3650_powertest
- [ ] Z3SwitchSoc
- [ ] railtest_efr32
- [ ] soc-thermometer-client

Verify the information displayed and click **Next**.


Migration Wizard

Project settings

Verify settings for each project

For each project below, please review the settings and resolve any missing SDKs and toolchains.

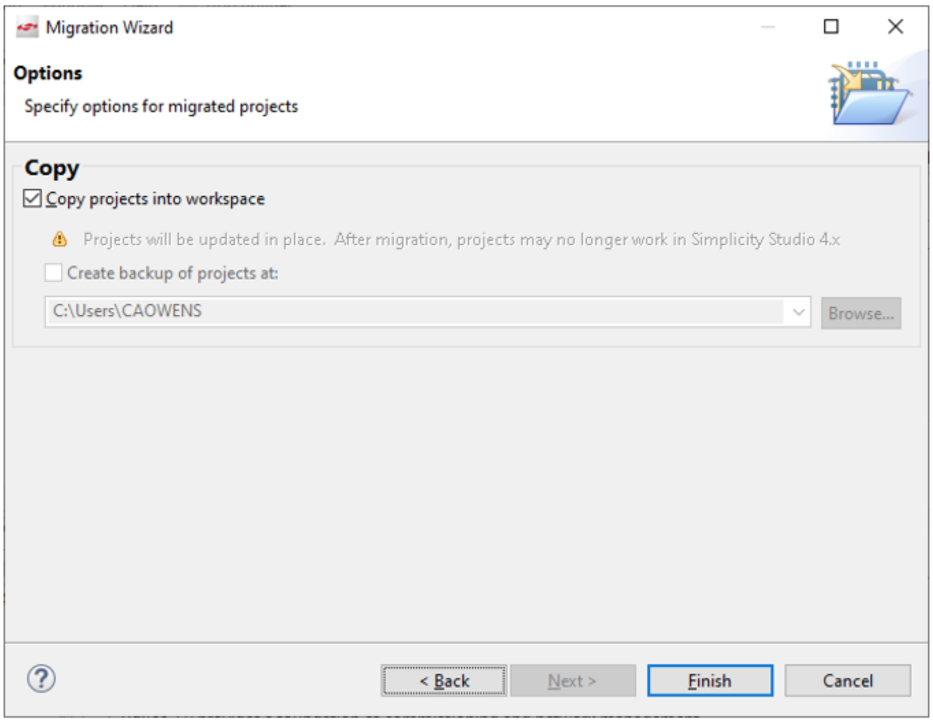
DK3650_powertest

Target Toolchains: GNU ARM 7.2.1.20170904

Current SDK: Gecko SDK Suite 2.7.6

Migrate to SDK: Gecko SDK Suite 3.0.0

Decide if you want to copy the project (recommended) and click **Finish**.


Migration Wizard

Options

Specify options for migrated projects

Copy

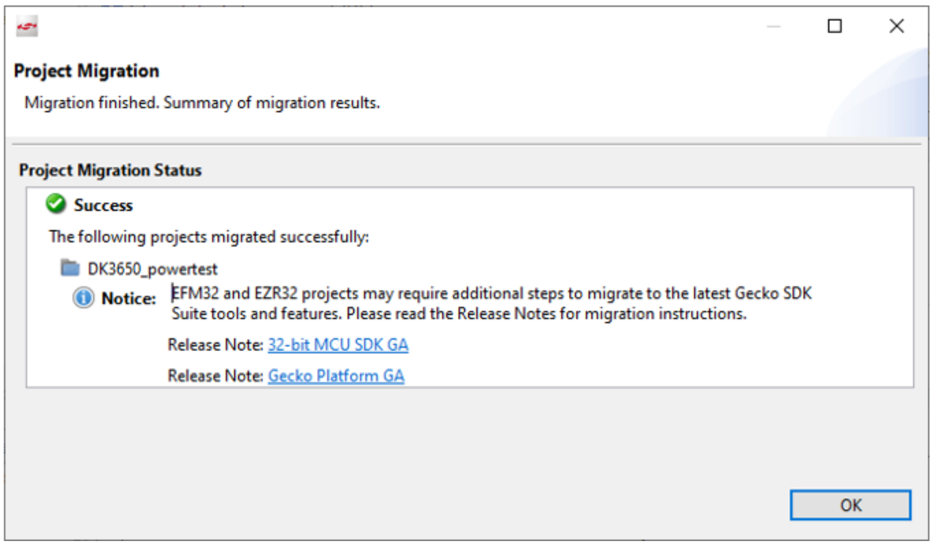
[✓] Copy projects into workspace

Projects will be updated in place. After migration, projects may no longer work in Simplicity Studio 4.x

[ ] Create backup of projects at:

C:\Users\CAOWENS

The project is migrated from SSv4 to SSv5.


Project Migration

Migration finished. Summary of migration results.

Project Migration Status

Success

The following projects migrated successfully:

DK3650_powertest

Notice: EFM32 and EZR32 projects may require additional steps to migrate to the latest Gecko SDK Suite tools and features. Please read the Release Notes for migration instructions.

Release Note: 32-bit MCU SDK GA

Release Note: Gecko Platform GA

Z-Wave Projects

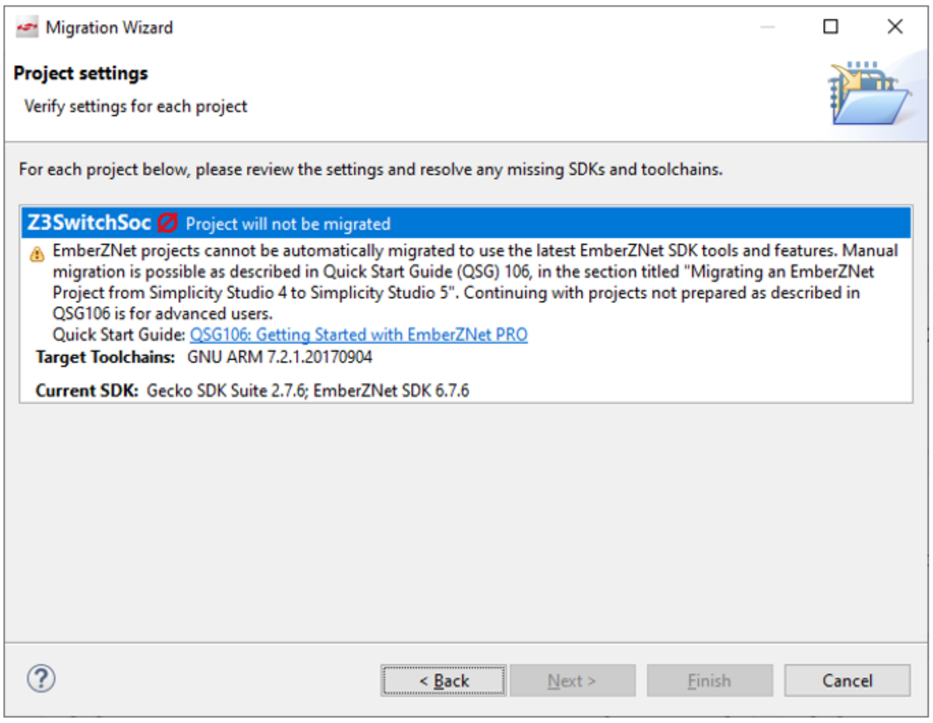
(Through Simplicity Studio 5.2) Follow the instructions in the knowledge base article Migrating a Z-Wave project from GSDK 2.7.6 to GSDK 3.0.0.

Zigbee, Flex, and Bluetooth/Bluetooth Mesh Projects

These projects cannot be migrated using the tool. Instead, refer to the following documents:

- Zigbee (Through Simplicity Studio 5.2): QSG106: Getting Started with EmberZNet PRO
- Flex: AN1254: Transitioning from the v2.x to the v3.x Proprietary Flex SDK
- Bluetooth: AN1255: Transitioning from the v2.x to the v3.x Bluetooth SDK
- Bluetooth Mesh: AN1298: Transitioning from the v1.x to the v2.x Bluetooth Mesh SDK

If you try using the tool, in most cases the tool will point you to the documentation for the migration process.


Migration Wizard

Project settings

Verify settings for each project

For each project below, please review the settings and resolve any missing SDKs and toolchains.

Z3SwitchSoc Project will not be migrated

EmberZNet projects cannot be automatically migrated to use the latest EmberZNet SDK tools and features. Manual migration is possible as described in Quick Start Guide (QSG) 106, in the section titled "Migrating an EmberZNet Project from Simplicity Studio 4 to Simplicity Studio 5". Continuing with projects not prepared as described in QSG106 is for advanced users.

Quick Start Guide: QSG106: Getting Started with EmberZNet PRO

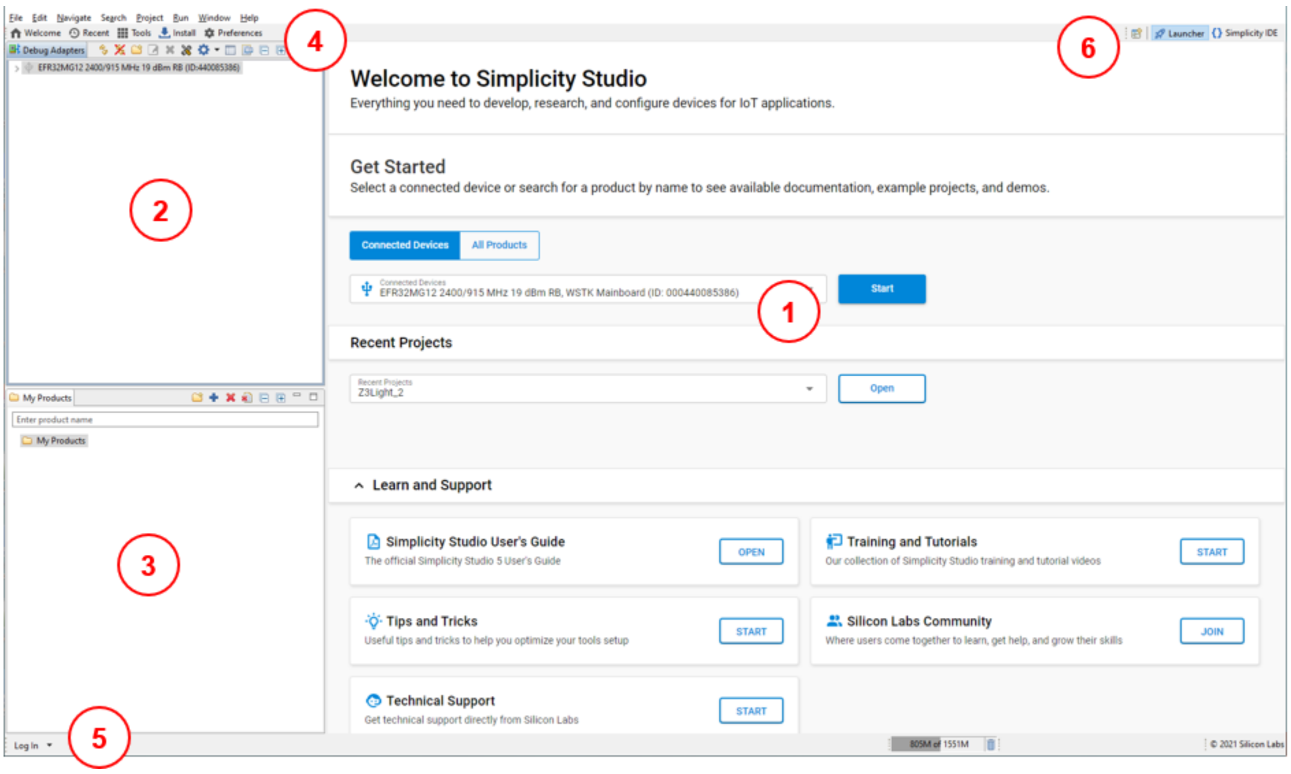
Target Toolchains: GNU ARM 7.2.1.20170904

Current SDK: Gecko SDK Suite 2.7.6; EmberZNet SDK 6.7.6

About the Launcher

About the Launcher

This section provides a reference to the functionality available in Simplicity Studio® 5 (SSv5) when you first open the application. This first perspective is called the **Launcher perspective**. A "perspective" is an Eclipse term for an initial arrangement of views and an editor area.



Welcome to Simplicity Studio

Everything you need to develop, research, and configure devices for IoT applications.

Get Started

Select a connected device or search for a product by name to see available documentation, example projects, and demos.

Connected Devices: EFR32MG12 2400/915 MHz 19 dBm RB, WSTK Mainboard (ID: 000440085386)

Simplicity Studio User's Guide — The official Simplicity Studio 5 User's Guide

Training and Tutorials — Our collection of Simplicity Studio training and tutorial videos

Tips and Tricks — Useful tips and tricks to help you optimize your tools setup

Silicon Labs Community — Where users come together to learn, get help, and grow their skills

Technical Support — Get technical support directly from Silicon Labs

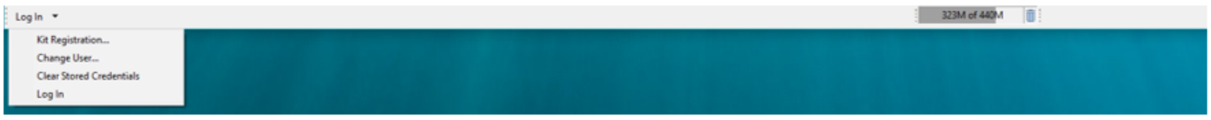
1 - Welcome and Device-Specific Tabs: The Welcome page includes a Get Started section to help with target kit, board, or device selection. The Learn and Support section may be expanded to show some of the reference and support resources available. Once a device is connected and/or selected, device-specific tabs provide access to example projects, documentation, and so on.

2 - Debug Adapters: Silicon Labs kits and supported debug adapters (for example, SEGGER J-Link products). The icon to the left of the debug adapter item indicates if it is connected over USB or Ethernet.

3 - My Products: Shows an editable list of products that you may want to use as target devices to set the Launcher's context. Use this view to select target devices when you don't have a kit available.

4 - A top-level menu provides access to a number of functions, including configuration options and help. A toolbar offers access to tools and functions on the left and shortcuts to different perspectives on the right.

5 - The Launcher perspective also provides additional functions across the bottom of the perspective:

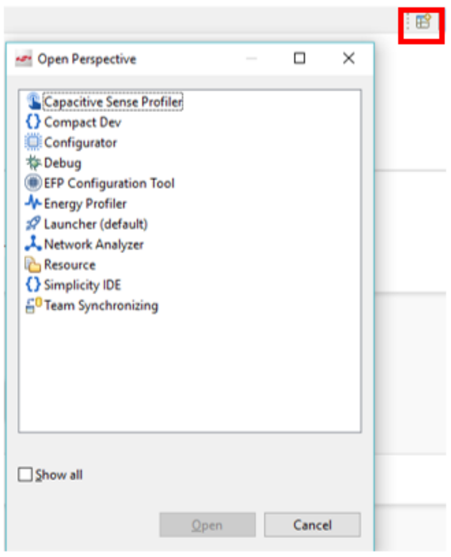


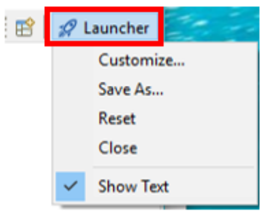
- Log in menu. Here you can:
  - Log in or out
  - Register a software development kit.
  - Change users
  - Clear stored credentials
- Garbage collection status and clear function. Click the trash can to run the garbage collector. This releases memory that may not have been released automatically. You should not need to use this function during normal operation. To turn the display off, go to **Preferences > General** and uncheck **Show heap status**.

6 - A perspectives tool bar in the top right shows open perspectives and allows you to open others.

Click **Open Perspective** to see a list of available perspectives. The perspectives list is similar to that available through the **Window > Perspective** menu selection.

Right-click an open perspective to see the context menu. While you can customize the perspective through the **Customize** menu option (see **Help > Help Contents > Workbench User Guide** for more information), you can also customize a perspective by dragging views and opening and closing views. These changes are persistent and survive closing and reopening a perspective. Click **Reset** to restore the perspective to its default settings.





Not all functions available through the Launcher perspective are relevant to developing for all target devices. The **Developing for ...** pages highlight the key pieces for each device category.

Welcome and Device Tabs

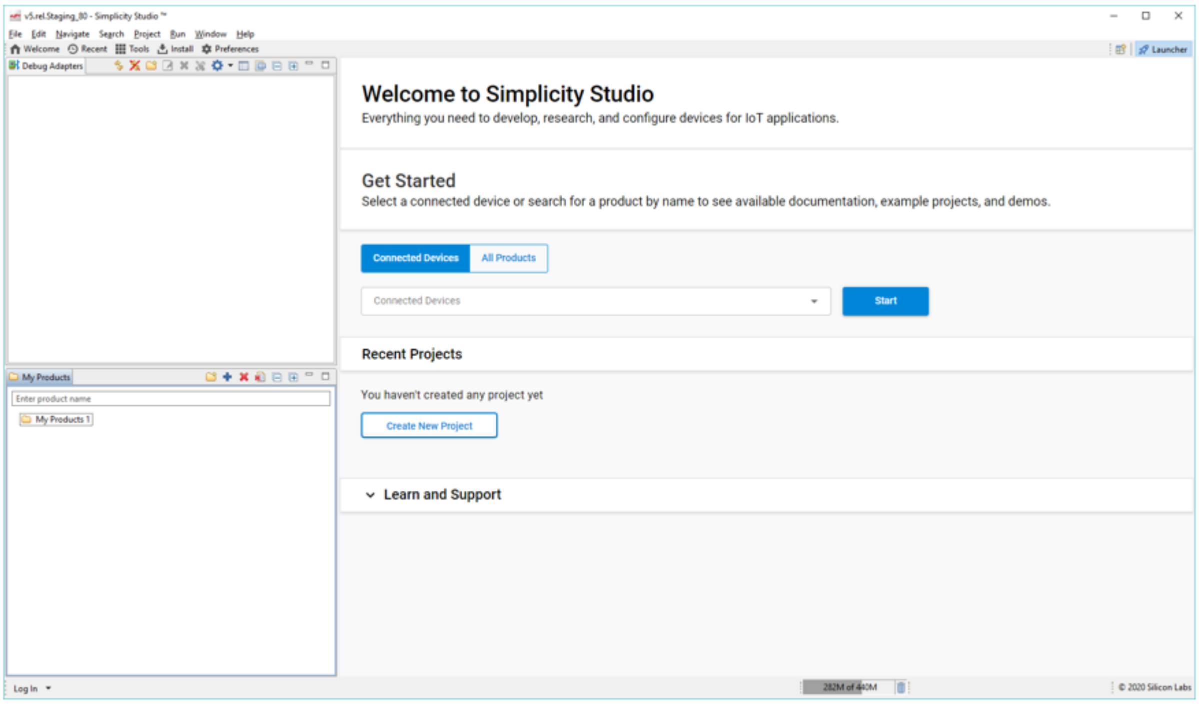
Welcome and Device Tabs

This page describes the functionality available on the Welcome page and the tabbed interface that is displayed once you connect or selected a devices. Tabs include:

- Overview
- Example Projects & Demos
- Documentation
- Compatible Tools

Welcome

Simplicity Studio® 5 (SSv5) opens to a Welcome page.



Welcome to Simplicity Studio

Everything you need to develop, research, and configure devices for IoT applications.

Get Started

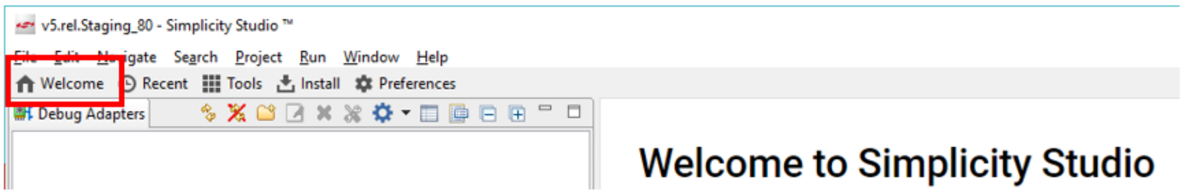
Select a connected device or search for a product by name to see available documentation, example projects, and demos.

Recent Projects

You haven't created any project yet

Learn and Support

To return to the Welcome page at any time, click **Welcome** on the tool bar.


v5.rel.Staging_80 - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Debug Adapters
Welcome to Simplicity Studio

On the Welcome Page you can:

- Start a new project
- Access educational and support resources

Select a Target Device

SSv5's purpose is to provide a development environment directed toward a specific target device. Therefore, one of the first things to do is to define that target device. Once a target device is selected, a tabbed interface to features specific to that device is available, starting on the Overview tab. The device can be a physical piece of hardware, or a virtual part.

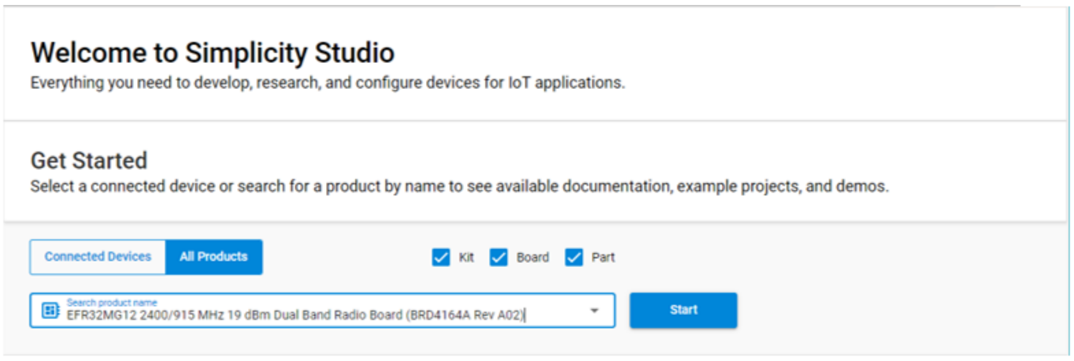
Physical If you have one or more devices physically connected, either on a development kit or on customer hardware with a supported debug adapter, they are displayed in the Debug Adapters view, where you can get started simply by clicking a device to selected it.

They are also displayed in the Get Started area's **Connected Devices** drop down. Clicking a device to select it, then click **Start**.

Virtual If you do not have a physical device, but would like to explore some of SSv5's functions, or get started with developing for a part you will receive later, select a virtual device either in the My Products view or in the Get Started area.

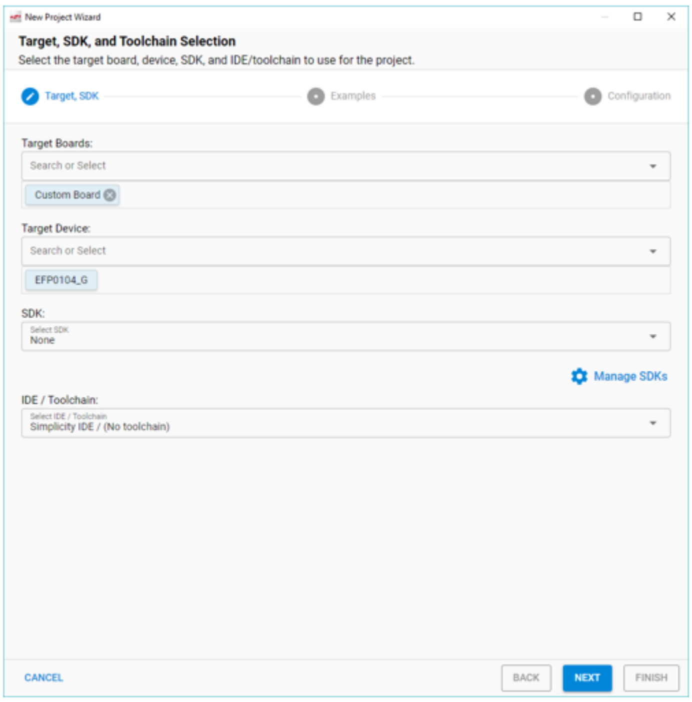
In the My Products view, start typing a product name and select the product of interest.

Under Get Started, click **All Products**. Use the checkboxes to limit the search list to kits, boards, or parts. Click in the search products field and start typing. When you see the target, select it and click **Start**. The next time you return to the Welcome page, that device will be shown in the My Products view.


Welcome to Simplicity Studio
Everything you need to develop, research, and configure devices for IoT applications.
Get Started
Select a connected device or search for a product by name to see available documentation, example projects, and demos.
Connected Devices
All Products
Kit
Board
Part
Search product name
EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)
Start

Start a New Project

You can start a new project from the Welcome page, but you must immediately select a target device. See Start a Project for more information.


New Project Wizard

Target, SDK, and Toolchain Selection

Select the target board, device, SDK, and IDE/toolchain to use for the project.

Target Boards: Search or Select — Custom Board

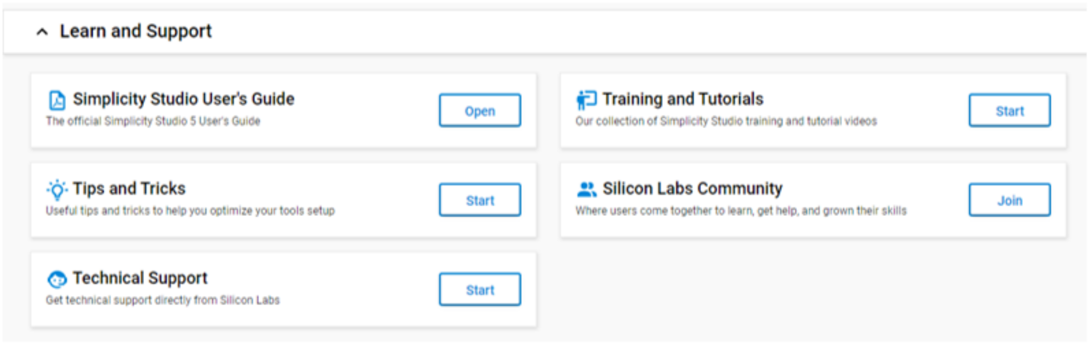
Target Device: Search or Select — EFP0104_G

SDK: Select SDK — None

IDE / Toolchain: Select IDE / Toolchain — Simplicity IDE / (No toolchain)

Learn and Support

Expand the Learn and Support section for access to a variety of resources related to developing for a Silicon Labs target.


Learn and Support

Simplicity Studio User's Guide — The official Simplicity Studio 5 User's Guide — Open

Training and Tutorials — Our collection of Simplicity Studio training and tutorial videos — Start

Tips and Tricks — Useful tips and tricks to help you optimize your tools setup — Start

Silicon Labs Community — Where users come together to learn, get help, and grown their skills — Join

Technical Support — Get technical support directly from Silicon Labs — Start

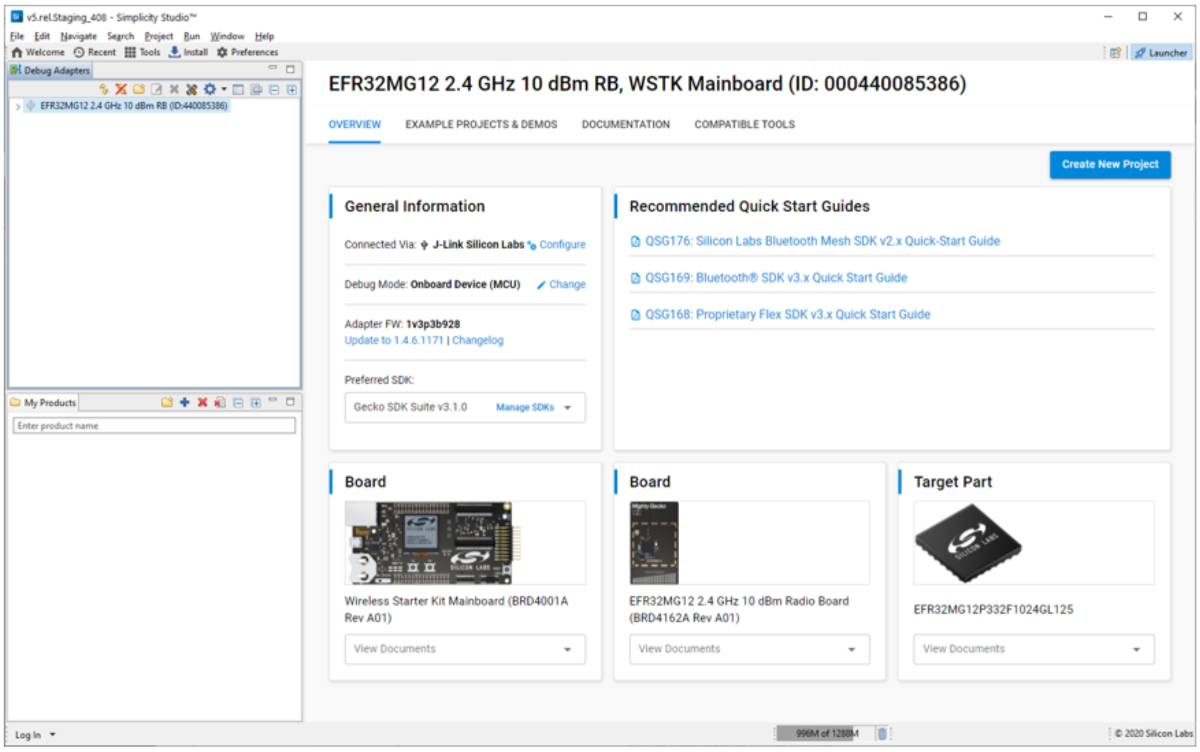
Device-Specific Tabs

Device-Specific tabs include:

- Overview
- Example Projects
- Documentation
- Compatible Tools

Overview Tab

Once you have selected a target device, the Launcher perspective editor area changes to the OVERVIEW tab specific to that part. For a physically-connected device you have general device information, as well as details about the hardware components. Each hardware component is pictured in a card and has a **View Documents** drop down where you can see related hardware documentation. Finally, links to recommended quick start guides from compatible protocol SDKs are provided.


v5.rel.Staging_408 - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher
Debug Adapters
EFR32MG12 2.4 GHz 10 dBm RB (ID:440085386)
My Products
Enter product name
Log In
996M of 1288M
© 2020 Silicon Labs

EFR32MG12 2.4 GHz 10 dBm RB, WSTK Mainboard (ID: 000440085386)

OVERVIEW | EXAMPLE PROJECTS & DEMOS | DOCUMENTATION | COMPATIBLE TOOLS

Create New Project

General Information

Connected Via: J-Link Silicon Labs Configure

Debug Mode: Onboard Device (MCU) Change

Adapter FW: 1v3p3b928
Update to 1.4.6.1171 | Changelog

Preferred SDK:
Gecko SDK Suite v3.1.0 Manage SDKs

Recommended Quick Start Guides

- QSG176: Silicon Labs Bluetooth Mesh SDK v2.x Quick-Start Guide
- QSG169: Bluetooth® SDK v3.x Quick Start Guide
- QSG168: Proprietary Flex SDK v3.x Quick Start Guide

Board

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

View Documents

Board

EFR32MG12 2.4 GHz 10 dBm Radio Board (BRD4162A Rev A01)

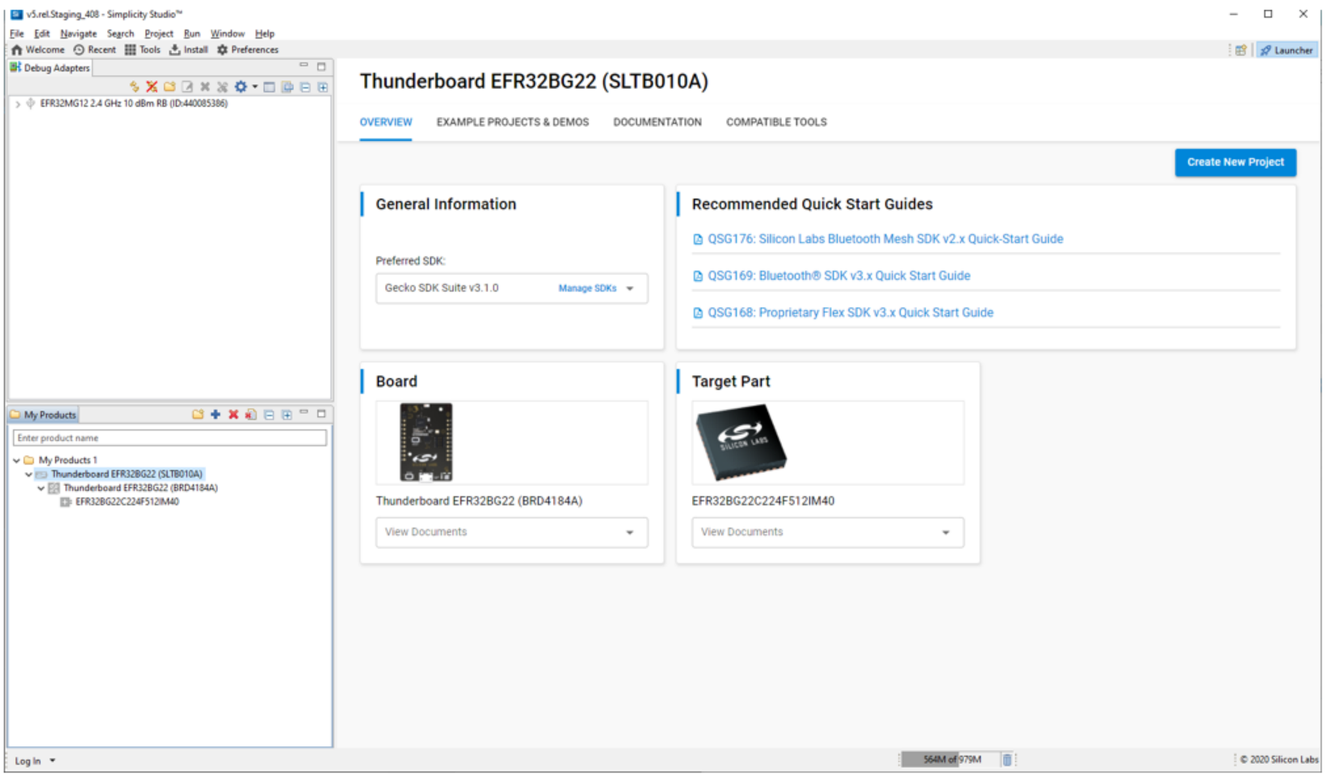
View Documents

Target Part

EFR32MG12P332F1024GL125

View Documents

SSv5 displays similar information for virtual devices (selected in the My Products view) and devices connected to a supported debug adapter (for example, SEGGER J-Link or a Wireless Starter Kit mainboard in debug OUT mode). The settings in the General Information card vary depending on the target device.


v5.rel.Staging_408 - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher
Debug Adapters
EFR32MG12 2.4 GHz 10 dBm RB (ID:440085386)
My Products
Enter product name
My Products 1
Thunderboard EFR32BG22 (SLTB010A)
Thunderboard EFR32BG22 (BRD4184A)
EFR32BG22C224F512IM40
Log In
564M of 979M
© 2020 Silicon Labs

Thunderboard EFR32BG22 (SLTB010A)

OVERVIEW | EXAMPLE PROJECTS & DEMOS | DOCUMENTATION | COMPATIBLE TOOLS

Create New Project

General Information

Preferred SDK:
Gecko SDK Suite v3.1.0 Manage SDKs

Recommended Quick Start Guides

- QSG176: Silicon Labs Bluetooth Mesh SDK v2.x Quick-Start Guide
- QSG169: Bluetooth® SDK v3.x Quick Start Guide
- QSG168: Proprietary Flex SDK v3.x Quick Start Guide

Board

Thunderboard EFR32BG22 (BRD4184A)

View Documents

Target Part

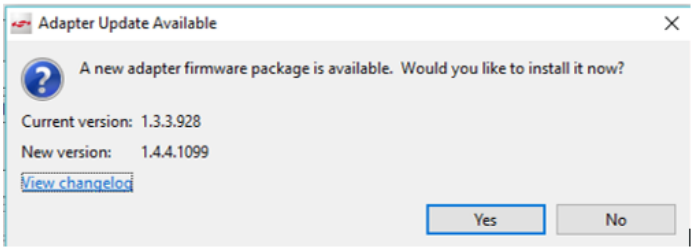
EFR32BG22C224F512IM40

View Documents

General Device Information

Configure Connection

Device Connection shows how the device is connected to SSv5. Click **Configure** to explore or modify connection parameters. If your device firmware is not up to date, you will be invited to update it.

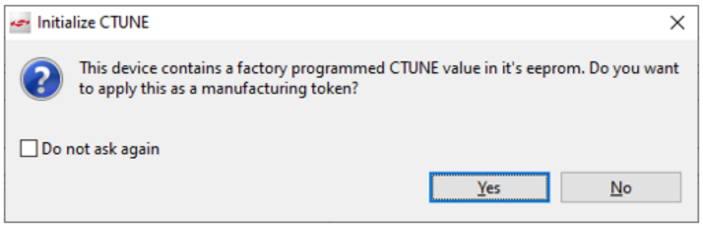

Adapter Update Available

A new adapter firmware package is available. Would you like to install it now?

Current version: 1.3.3.928

New version: 1.4.4.1099

If you are targeting an EFR32-based Silicon Labs kit, you will almost certainly also see the following question:


Initialize CTUNE

This device contains a factory programmed CTUNE value in it's eeprom. Do you want to apply this as a manufacturing token?

☐ Do not ask again

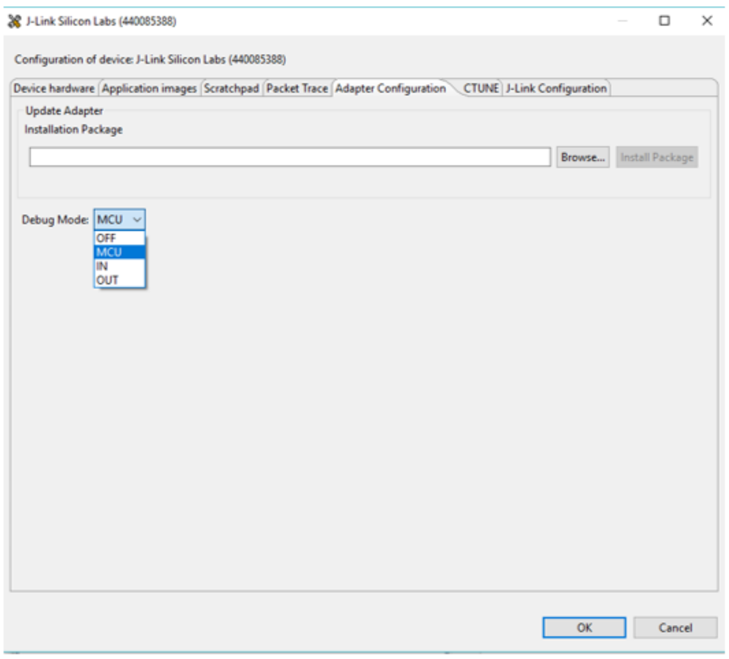
The CTUNE value is used to tune the external crystal capacitors to hit the exact frequency they are intended to hit. Because this varies from board to board, in production Silicon Labs measures it during board tests and programs a unique CTUNE value for each board into the EEPROM. Each SDK also has a default CTUNE value programmed in a manufacturing token. This message asks if you want to overwrite the default CTUNE value with the one found in the EEPROM. Because the EEPROM value is more accurate, Silicon Labs recommends you click **Yes**. The default CTUNE value will also work, but could under some circumstances, such as temperature extremes, bring the radio frequency out of spec.

Debug Mode

The Debug Mode controls the interface to the wireless starter kit mainboard onboard debugger. Changing Debug Mode opens the Adapter Configuration tab of the J-Link Configuration tool. The debug modes are:

- **Onboard Device (MCU)** (default): The debugger built into the development board is connected to the on-board target device.
- **External Device (OUT)**: The on-board debugger is configured for connection to an external device such as your custom hardware.
- **External Debugger (IN)**: An external debugger is connected to the device on the development board.

See your kit's User's Guide for more information on the debug modes available.



Configuration of device: J-Link Silicon Labs (440085388)

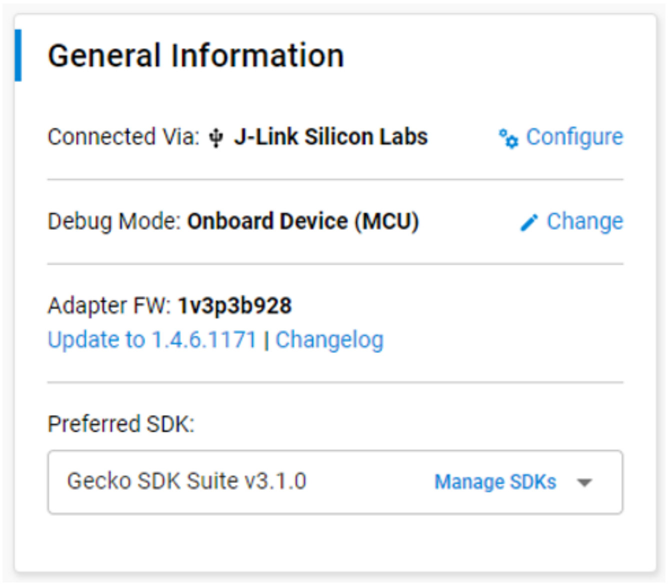
Update Adapter

Installation Package

Debug Mode: MCU

Adapter Firmware

This shows the firmware version running on the debug controller of your Silicon Labs development kit and whether or not an update is available. Silicon Labs strongly recommends that you update adapter firmware to the current version. The Changelog shows the firmware release notes.



General Information

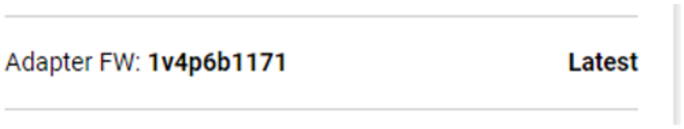
Connected Via: J-Link Silicon Labs

Debug Mode: Onboard Device (MCU)

Adapter FW: 1v3p3b928
Update to 1.4.6.1171 | Changelog

Preferred SDK:
Gecko SDK Suite v3.1.0

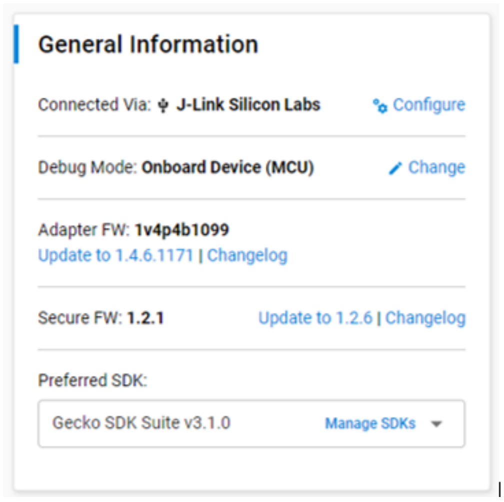
When you update adapter firmware SSv5 will ask you to confirm before proceeding. When firmware is up to date the interface says **Latest**.



Adapter FW: 1v4p6b1171 Latest

Secure Firmware

Series 2 devices contain a Secure Element, a tamper-resistant component used to securely store sensitive data and keys, and to execute cryptographic functions and secure services. The Secure Element firmware can also be updated. When you have a Series 2 device connected, the General Information card includes a **Secure FW** line. This shows the firmware version running on the Secure Element and whether or not an update is available. Silicon Labs strongly recommends that you update firmware to the current version. The Changelog shows the firmware release notes.



General Information

Connected Via: J-Link Silicon Labs — Configure

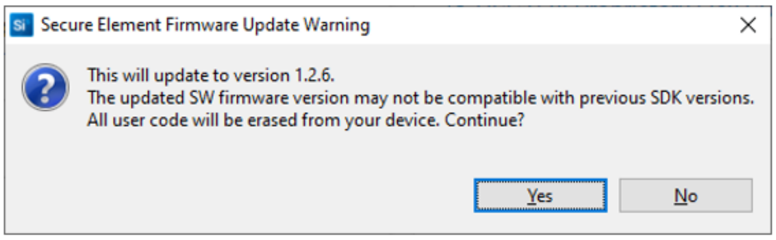
Debug Mode: **Onboard Device (MCU)** — Change

Adapter FW: **1v4p4b1099**
Update to 1.4.6.1171 | Changelog

Secure FW: **1.2.1** — Update to 1.2.6 | Changelog

Preferred SDK:
Gecko SDK Suite v3.1.0 — Manage SDKs

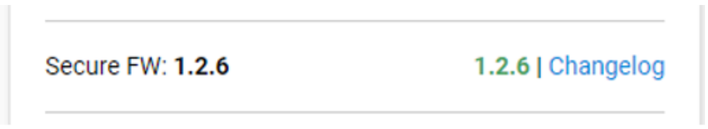
When you update, you are warned that user software, including any factory-installed applications such as RangeTest, will be deleted.



Secure Element Firmware Update Warning

This will update to version 1.2.6.
The updated SW firmware version may not be compatible with previous SDK versions.
All user code will be erased from your device. Continue?

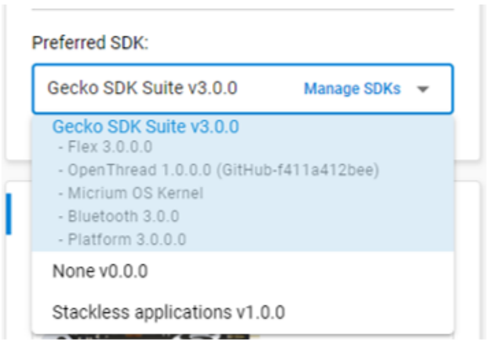
After upgrade, the installed and available versions are the same.



Secure FW: **1.2.6** — 1.2.6 | Changelog

Preferred SDK

Some developers may have more than one GSDK version installed. The preferred SDK shows the currently selected SDK. Click **Manage SDKs** to see other options.


Preferred SDK:

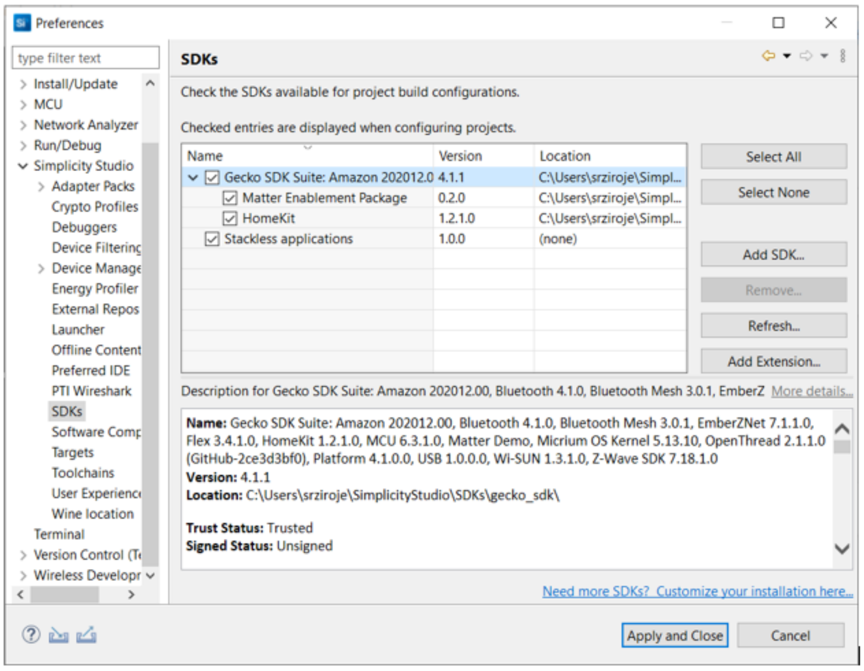
Gecko SDK Suite v3.0.0 — Manage SDKs

Gecko SDK Suite v3.0.0
- Flex 3.0.0.0
- OpenThread 1.0.0.0 (GitHub-f411a412bee)
- Micrium OS Kernel
- Bluetooth 3.0.0
- Platform 3.0.0.0

None v0.0.0

Stackless applications v1.0.0

The options shown are those selected in Preferences > Simplicity Studio > SDKs. To add an installed SDK to the list, click **Add SDK** and browse to the installed location. SSv5 autodetects SDKs in the location folder. Select the SDK to add, and click **OK**. Make sure the new SDK is selected in the list, and click **Apply and Close**. You should now be able to select it in the Preferred SDK list.



SDKs

Check the SDKs available for project build configurations.

Checked entries are displayed when configuring projects.

Name	Version	Location
☑ Gecko SDK Suite: Amazon 202012.0	4.1.1	C:\Users\srziroje\Simpl...
☑ Matter Enablement Package	0.2.0	C:\Users\srziroje\Simpl...
☑ HomeKit	1.2.1.0	C:\Users\srziroje\Simpl...
☑ Stackless applications	1.0.0	(none)

Description for Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.1.0, Bluetooth Mesh 3.0.1, EmberZ More details...

Name: Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.1.0, Bluetooth Mesh 3.0.1, EmberZNet 7.1.1.0, Flex 3.4.1.0, HomeKit 1.2.1.0, MCU 6.3.1.0, Matter Demo, Micrium OS Kernel 5.13.10, OpenThread 2.1.1.0 (GitHub-2ce3d3bf0), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.1.0, Z-Wave SDK 7.18.1.0
Version: 4.1.1
Location: C:\Users\srziroje\SimplicityStudio\SDKs\gecko_sdk\

Trust Status: Trusted
Signed Status: Unsigned

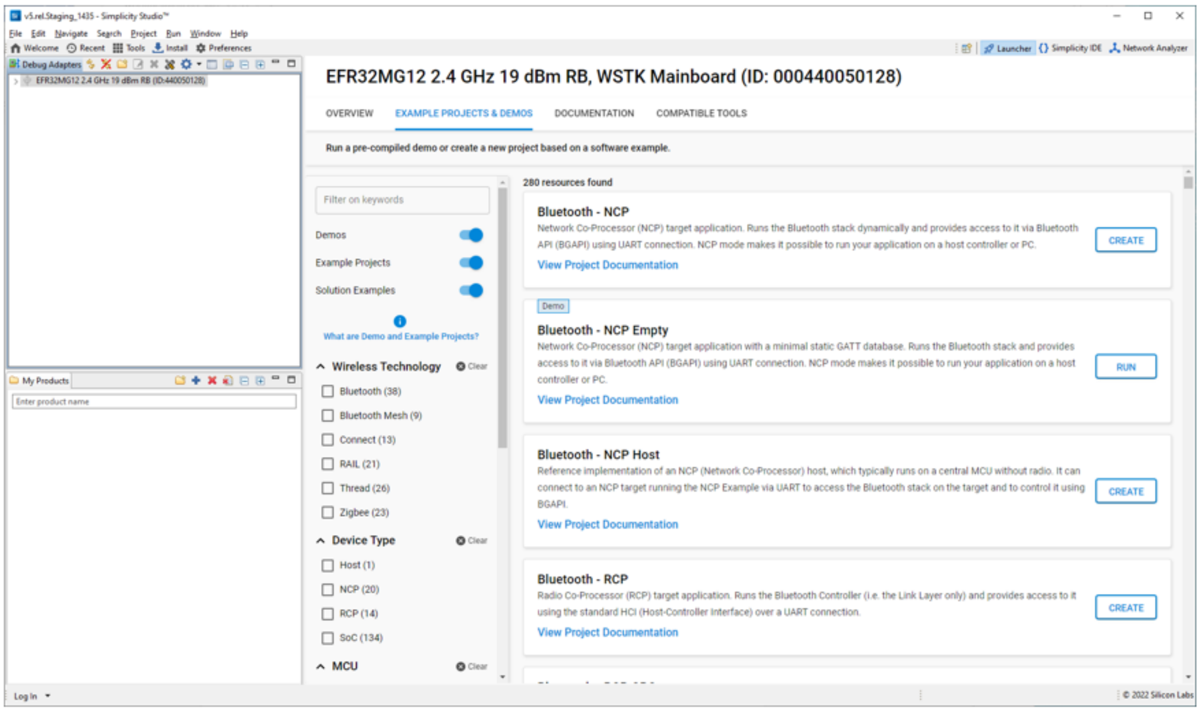
Need more SDKs? Customize your installation here...

The same dialog is used to install SDK extensions.

Example Projects & Demos Tab

The EXAMPLE PROJECTS & DEMOS tab shows a list of example projects and demos compatible with the selected device. A demo is a prebuilt software example that can be loaded into a compatible device and used to show and test application functionality. Every demo comes with an associated example project. See the Quick Start Guide for your SDK for more information about the examples and demos provided.

Click **RUN** on any demo to install it on a target device. Click **CREATE** on any project to create it. This is equivalent to creating a project from the OVERVIEW tab, except that the project is already selected. By default, the tab enables showing both demos and examples. Demos have a blue tag in the upper left of the card. To see only one type of file, disable the other type.



EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440050128)

Run a pre-compiled demo or create a new project based on a software example.

280 resources found

Bluetooth - NCP
Network Co-Processor (NCP) target application. Runs the Bluetooth stack dynamically and provides access to it via Bluetooth API (BGAPI) using UART connection. NCP mode makes it possible to run your application on a host controller or PC.

Bluetooth - NCP Empty
Network Co-Processor (NCP) target application with a minimal static GATT database. Runs the Bluetooth stack and provides access to it via Bluetooth API (BGAPI) using UART connection. NCP mode makes it possible to run your application on a host controller or PC.

Bluetooth - NCP Host
Reference implementation of an NCP (Network Co-Processor) host, which typically runs on a central MCU without radio. It can connect to an NCP target running the NCP Example via UART to access the Bluetooth stack on the target and to control it using BGAPI.

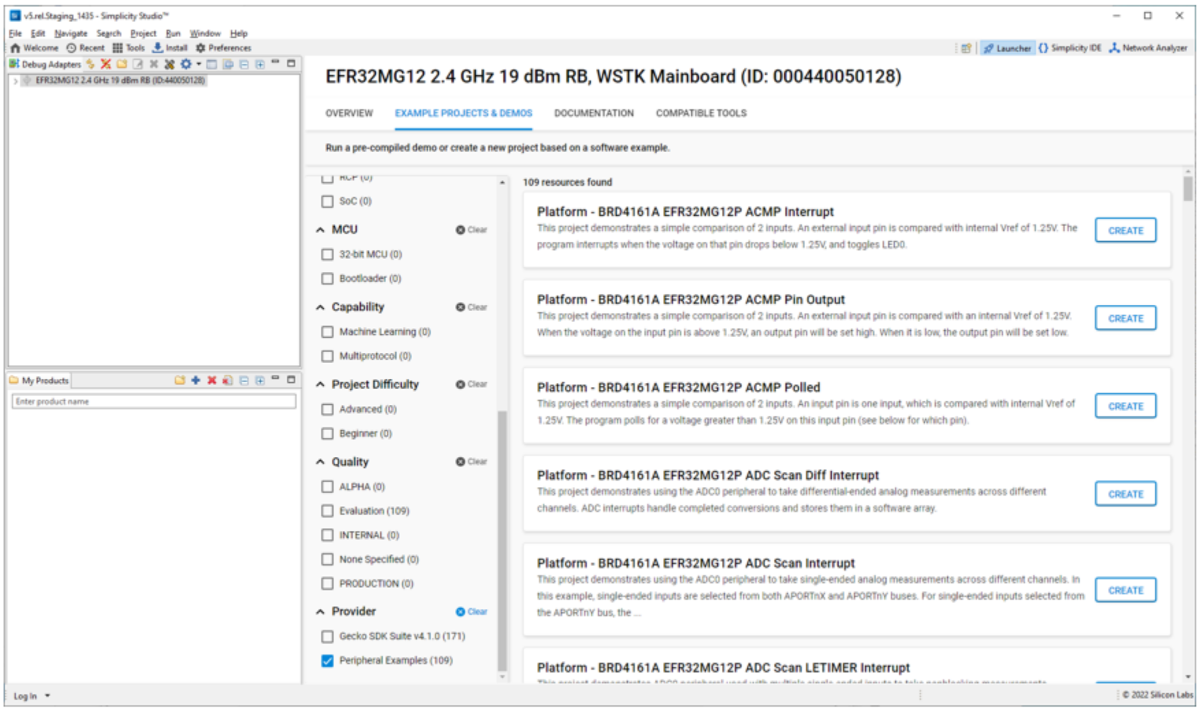
Bluetooth - RCP
Radio Co-Processor (RCP) target application. Runs the Bluetooth Controller (i.e. the Link Layer only) and provides access to it using the standard HCI (Host-Controller Interface) over a UART connection.

Use the switches, checkboxes and search box to filter the list. Note: Solutions are combinations of projects on which certain operations can be performed at the same time. This is a new feature and support is still being added for it. No example solutions are provided in this release, but you can create your own. See Solutions for more information.

The checkboxes displayed depend on your selected device. Checkbox categories include:

- Wireless Technology
- Device Type
- MCU
- Capability
- Project Difficulty
- Quality
- Provider

Provider includes the checkbox **Peripheral Examples**. These examples are provided from the Silicon Labs GitHub repository **peripheral-examples**, are specific to your connected or selected board, and allow you to exercise various peripheral functions.



EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440050128)

Run a pre-compiled demo or create a new project based on a software example.

- SoC (0)
- MCU: 32-bit MCU (0), Bootloader (0)
- Capability: Machine Learning (0), Multiprotocol (0)
- Project Difficulty: Advanced (0), Beginner (0)
- Quality: ALPHA (0), Evaluation (109), INTERNAL (0), None Specified (0), PRODUCTION (0)
- Provider: Gecko SDK Suite v4.1.0 (171), ✓ Peripheral Examples (109)

109 resources found

Platform - BRD4161A EFR32MG12P ACMP Interrupt
This project demonstrates a simple comparison of 2 inputs. An external input pin is compared with internal Vref of 1.25V. The program interrupts when the voltage on that pin drops below 1.25V, and toggles LED0.

Platform - BRD4161A EFR32MG12P ACMP Pin Output
This project demonstrates a simple comparison of 2 inputs. An external input pin is compared with an internal Vref of 1.25V. When the voltage on the input pin is above 1.25V, an output pin will be set high. When it is low, the output pin will be set low.

Platform - BRD4161A EFR32MG12P ACMP Polled
This project demonstrates a simple comparison of 2 inputs. An input pin is one input, which is compared with internal Vref of 1.25V. The program polls for a voltage greater than 1.25V on this input pin (see below for which pin).

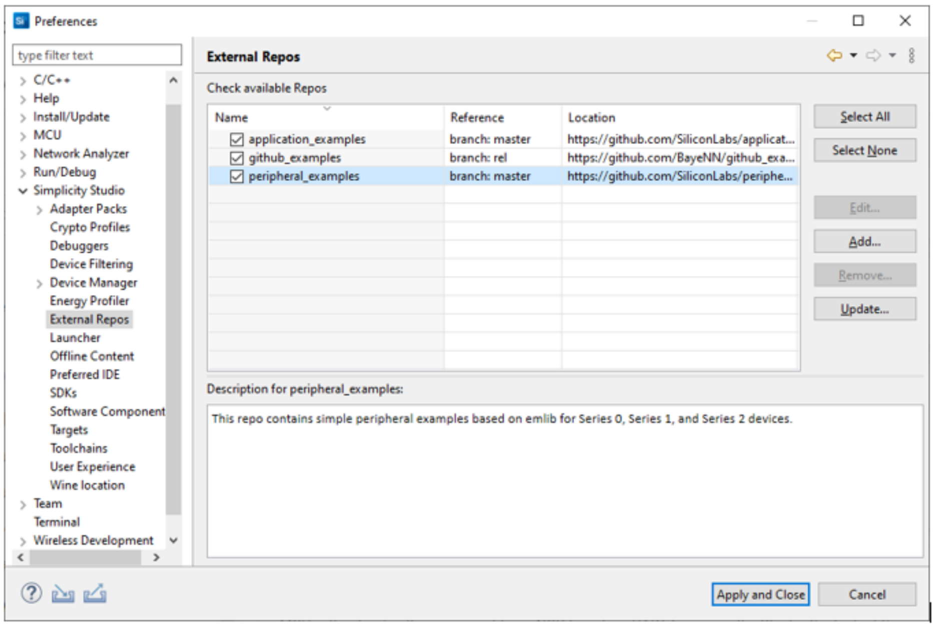
Platform - BRD4161A EFR32MG12P ADC Scan Diff Interrupt
This project demonstrates using the ADC0 peripheral to take differential-ended analog measurements across different channels. ADC interrupts handle completed conversions and stores them in a software array.

Platform - BRD4161A EFR32MG12P ADC Scan Interrupt
This project demonstrates using the ADC0 peripheral to take single-ended analog measurements across different channels. In this example, single-ended inputs are selected from both APORTnX and APORTnY buses. For single-ended inputs selected from the APORTnY bus, the ...

Platform - BRD4161A EFR32MG12P ADC Scan LETIMER Interrupt

© 2022 Silicon Labs

SSv5 allows you to add other GitHub repositories containing examples. To add a GitHub repository, go to **Preferences > Simplicity Studio > External Repos**. Here you can add, edit, and delete repos, and select from repos that are already added. Adding a repo is done in two steps: cloning and then selecting the branch, tag, or commit to add. The default branch is Master. **application_examples** and **peripheral_examples** are official Silicon Labs repos and may not be edited or deleted.



External Repos

Check available Repos

Name	Reference	Location
✓ application_examples	branch: master	https://github.com/SiliconLabs/applicat...
✓ github_examples	branch: rel	https://github.com/BayeNN/github_exa...
✓ peripheral_examples	branch: master	https://github.com/SiliconLabs/periphe...

Description for peripheral_examples:

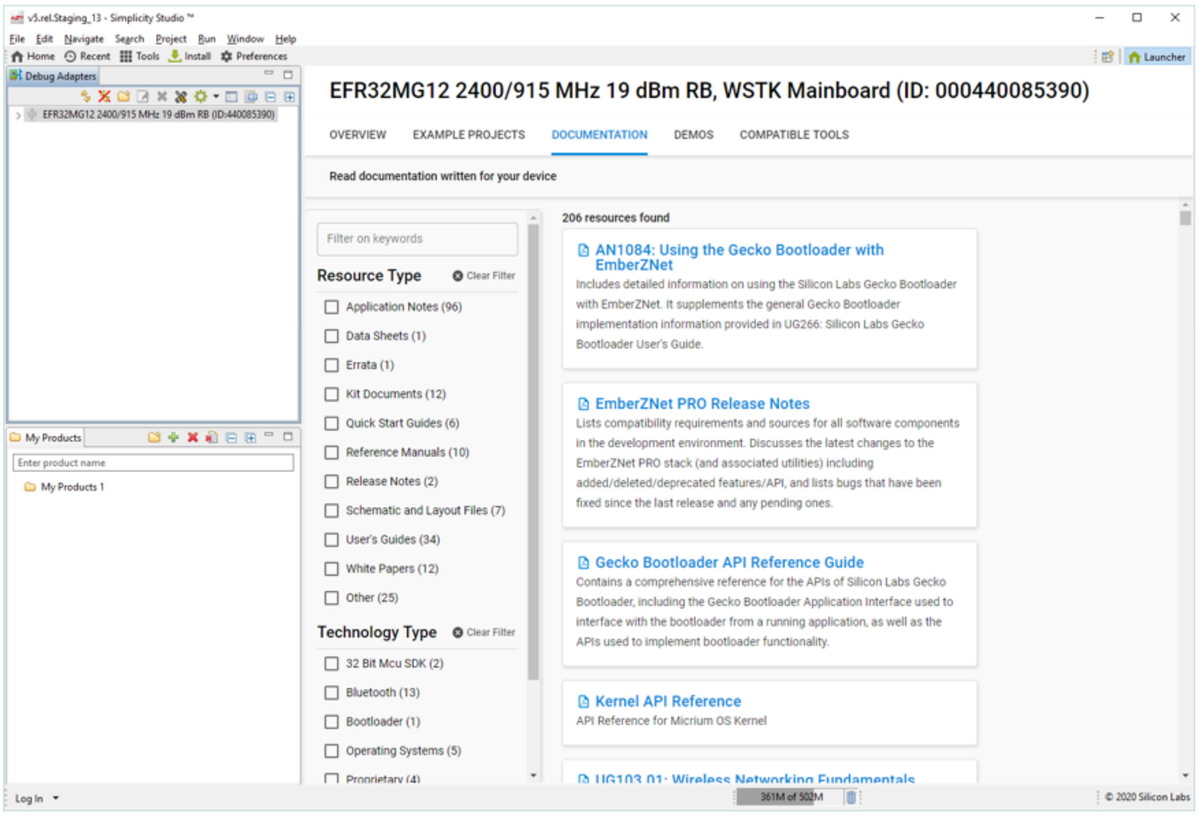
This repo contains simple peripheral examples based on emlib for Series 0, Series 1, and Series 2 devices.

You must be connected to the Internet to create a project from an example in a remote GitHub repository. Projects created from GitHub repositories are created in the location specified in the project creation dialog. The cloned project does not have any Git-related information in it. You cannot sync/checkout/pull code from Git.

If you specify a repository installed locally, SSv5 will not synchronize the repo and all Git-related interface items will not be presented.

Documentation Tab

The DOCUMENTATION tab shows all documentation compatible with the selected part. Use the checkboxes or text filter field to find a resource of interest. The technology filter corresponding to your development environment will show you most software documents relevant to that environment.


v5.rel.Staging_13 - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help
Home Recent Tools Install Preferences
Launcher
Debug Adapters
EFR32MG12 2400/915 MHz 19 dBm RB (ID:440085390)
My Products
Enter product name
My Products 1
Log In
361M of 502M
© 2020 Silicon Labs

EFR32MG12 2400/915 MHz 19 dBm RB, WSTK Mainboard (ID: 000440085390)

OVERVIEW EXAMPLE PROJECTS DOCUMENTATION DEMOS COMPATIBLE TOOLS

Read documentation written for your device

Filter on keywords

Resource Type — Clear Filter

- ☐ Application Notes (96)
- ☐ Data Sheets (1)
- ☐ Errata (1)
- ☐ Kit Documents (12)
- ☐ Quick Start Guides (6)
- ☐ Reference Manuals (10)
- ☐ Release Notes (2)
- ☐ Schematic and Layout Files (7)
- ☐ User's Guides (34)
- ☐ White Papers (12)
- ☐ Other (25)

Technology Type — Clear Filter

- ☐ 32 Bit Mcu SDK (2)
- ☐ Bluetooth (13)
- ☐ Bootloader (1)
- ☐ Operating Systems (5)
- ☐ Proprietary (4)

206 resources found

AN1084: Using the Gecko Bootloader with EmberZNet
Includes detailed information on using the Silicon Labs Gecko Bootloader with EmberZNet. It supplements the general Gecko Bootloader implementation information provided in UG266: Silicon Labs Gecko Bootloader User's Guide.

EmberZNet PRO Release Notes
Lists compatibility requirements and sources for all software components in the development environment. Discusses the latest changes to the EmberZNet PRO stack (and associated utilities) including added/deleted/deprecated features/API, and lists bugs that have been fixed since the last release and any pending ones.

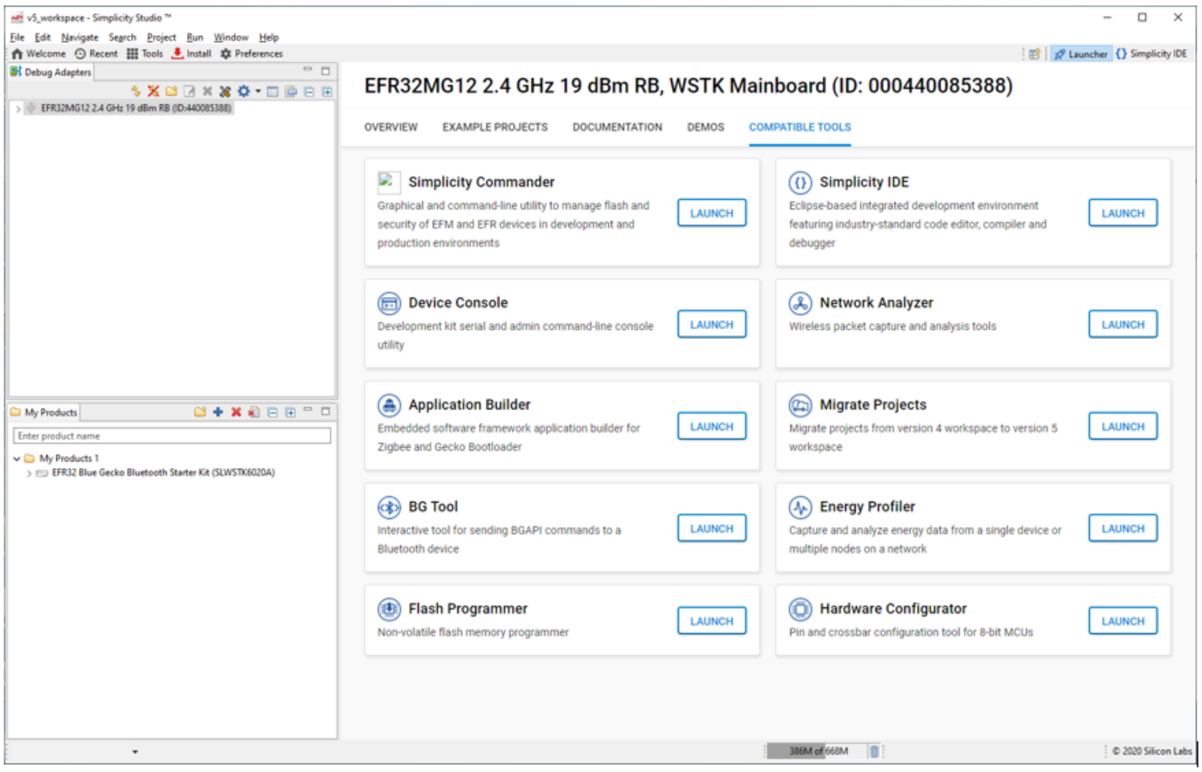
Gecko Bootloader API Reference Guide
Contains a comprehensive reference for the APIs of Silicon Labs Gecko Bootloader, including the Gecko Bootloader Application Interface used to interface with the bootloader from a running application, as well as the APIs used to implement bootloader functionality.

Kernel API Reference
API Reference for Micrium OS Kernel

UG103.01: Wireless Networking Fundamentals

Compatible Tools Tab

The COMPATIBLE TOOLS tab shows the tools compatible with the selected product. The Tools button on the toolbar shows all tools unfiltered.


v5_workspace - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Debug Adapters
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
My Products
Enter product name
My Products 1
EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020A)
OVERVIEW
EXAMPLE PROJECTS
DOCUMENTATION
DEMOS
COMPATIBLE TOOLS
LAUNCH
386M of 668M
© 2020 Silicon Labs

EFR32MG12 2.4 GHz 19 dBm RB, WSTK Mainboard (ID: 000440085388)

Simplicity Commander
Graphical and command-line utility to manage flash and security of EFM and EFR devices in development and production environments

Simplicity IDE
Eclipse-based integrated development environment featuring industry-standard code editor, compiler and debugger

Device Console
Development kit serial and admin command-line console utility

Network Analyzer
Wireless packet capture and analysis tools

Application Builder
Embedded software framework application builder for Zigbee and Gecko Bootloader

Migrate Projects
Migrate projects from version 4 workspace to version 5 workspace

BG Tool
Interactive tool for sending BGAPI commands to a Bluetooth device

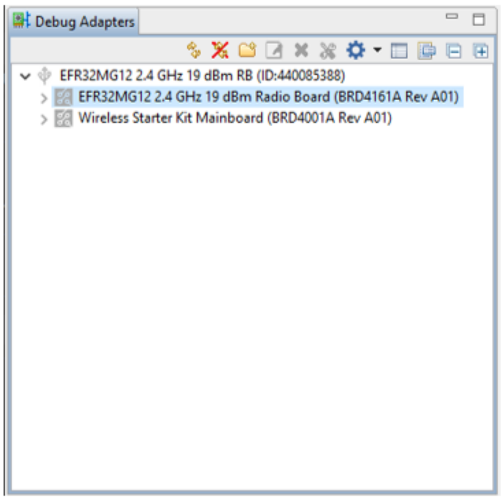
Energy Profiler
Capture and analyze energy data from a single device or multiple nodes on a network

Flash Programmer
Non-volatile flash memory programmer

Hardware Configurator
Pin and crossbar configuration tool for 8-bit MCUs

Debug Adapters

The Simplicity Studio® 5 (SSv5) Debug Adapters view shows connected hardware adapters and displays kits, board, and devices.

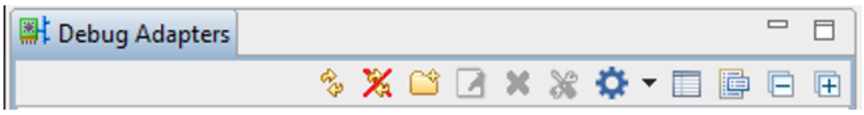

Debug Adapters

EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)

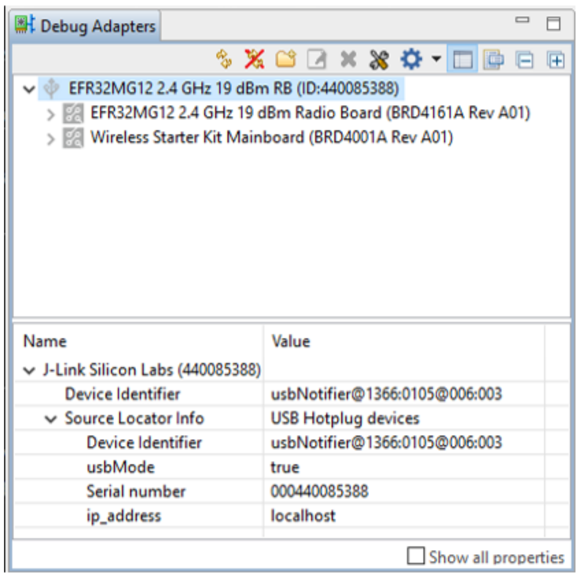
EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

The Debug Adapters view menu functions are:



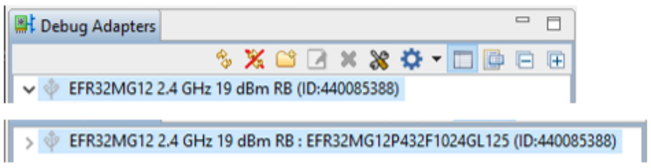
- Refresh the list of detected kits and debuggers
- Clear and refresh the list of detected kits and debuggers
- Create a group to help organize multiple adapters
- Rename a group
- Delete a group
- Device Configuration (equivalent to Configure Connection)
- Preferences (provides shortcuts to three of the options available in Preferences from the toolbar or **Window > Preferences**.
    - Device Manager (Simplicity Studio > Device Manager)
    - Discovery (Simplicity Studio > Device Manager > TCP/IP Adapters)
    - Stack (Network Analyzer > Decoding > Stack Versions)
- Toggle properties (shows or hides the adapter properties)



EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)
Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Name	Value
J-Link Silicon Labs (440085388)	
Device Identifier	usbNotifier@1366:0105@006:003
Source Locator Info	USB Hotplug devices
Device Identifier	usbNotifier@1366:0105@006:003
usbMode	true
Serial number	000440085388
ip_address	localhost

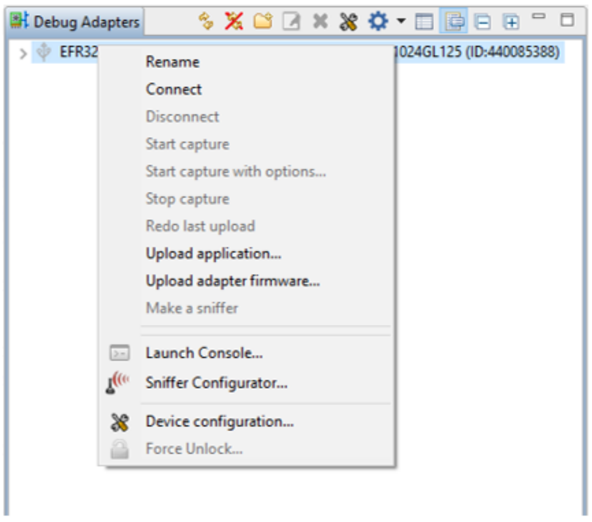
- Toggle device label detail



EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
EFR32MG12 2.4 GHz 19 dBm RB : EFR32MG12P432F1024GL125 (ID:440085388)

- Collapse all
- Expand all

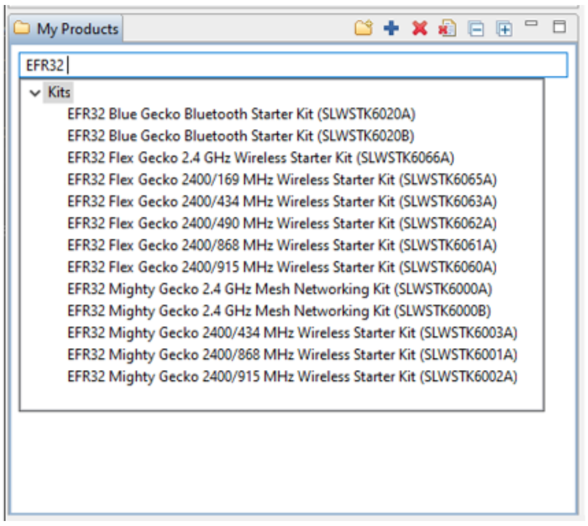
Finally, right-clicking on a debug adapter brings up a menu of functions. The availability varies depending on the context.



My Products

My Products

The Simplicity Studio® 5 (SSv5) My Products view provides you with a convenient way to save and organize a list of kits, boards, and devices for quick access. Use the search function to locate and add products to the view. Products selected here set the target device context and allow you to explore the relevant resources and to start projects.

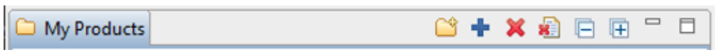

My Products

EFR32

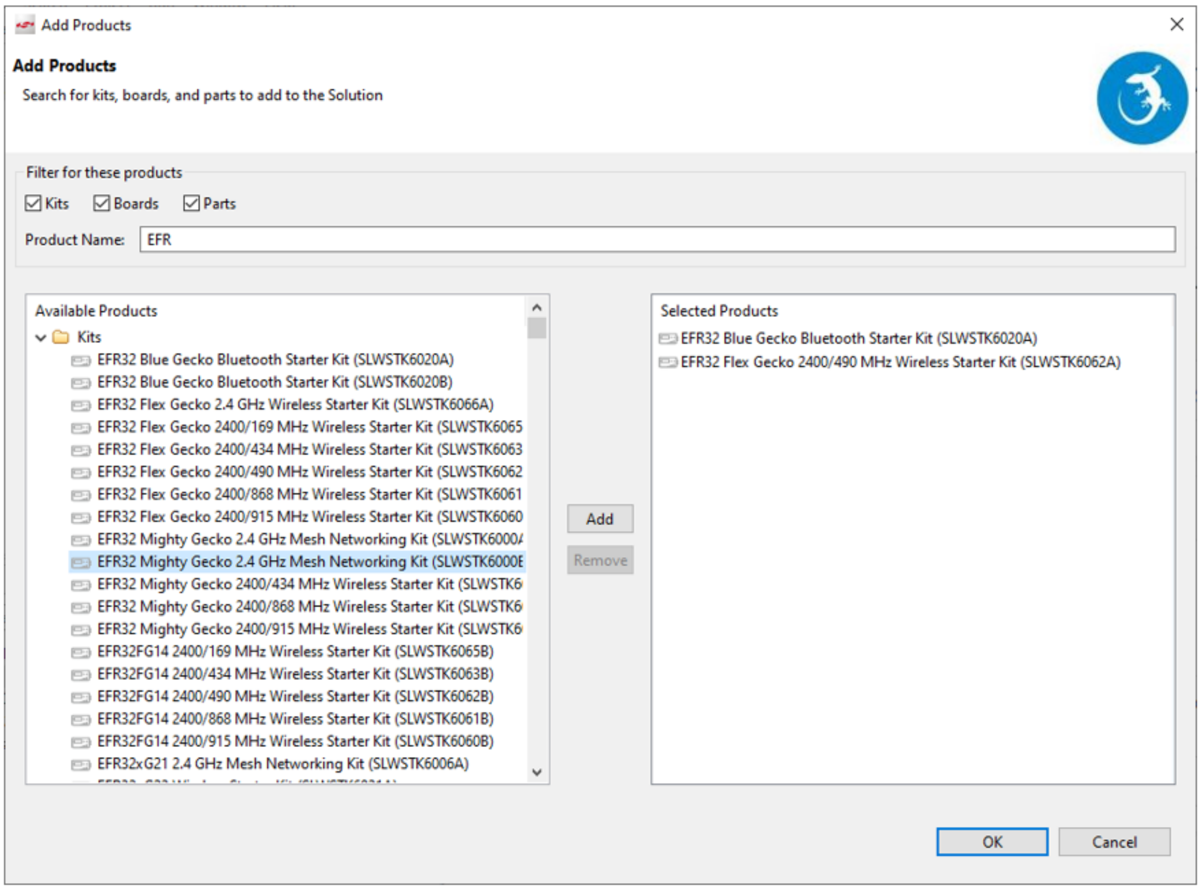
Kits
- EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020A)
- EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020B)
- EFR32 Flex Gecko 2.4 GHz Wireless Starter Kit (SLWSTK6066A)
- EFR32 Flex Gecko 2400/169 MHz Wireless Starter Kit (SLWSTK6065A)
- EFR32 Flex Gecko 2400/434 MHz Wireless Starter Kit (SLWSTK6063A)
- EFR32 Flex Gecko 2400/490 MHz Wireless Starter Kit (SLWSTK6062A)
- EFR32 Flex Gecko 2400/868 MHz Wireless Starter Kit (SLWSTK6061A)
- EFR32 Flex Gecko 2400/915 MHz Wireless Starter Kit (SLWSTK6060A)
- EFR32 Mighty Gecko 2.4 GHz Mesh Networking Kit (SLWSTK6000A)
- EFR32 Mighty Gecko 2.4 GHz Mesh Networking Kit (SLWSTK6000B)
- EFR32 Mighty Gecko 2400/434 MHz Wireless Starter Kit (SLWSTK6003A)
- EFR32 Mighty Gecko 2400/868 MHz Wireless Starter Kit (SLWSTK6001A)
- EFR32 Mighty Gecko 2400/915 MHz Wireless Starter Kit (SLWSTK6002A)

Start typing a part number or key word to see kits, boards, and parts. Once you select an item, it is added to a folder called 'My Products 1'.

The My Products view menu provides functions to help manage parts selected in this view.



- Make a new folder (rename a folder by right-clicking it)
- Add a product (opens a dialog where you can add and remove more than one part at a time)



Add Products

Add Products

Search for kits, boards, and parts to add to the Solution

Filter for these products

☑ Kits ☑ Boards ☑ Parts

Product Name: EFR

Available Products

Kits

- EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020A)
- EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020B)
- EFR32 Flex Gecko 2.4 GHz Wireless Starter Kit (SLWSTK6066A)
- EFR32 Flex Gecko 2400/169 MHz Wireless Starter Kit (SLWSTK6065
- EFR32 Flex Gecko 2400/434 MHz Wireless Starter Kit (SLWSTK6063
- EFR32 Flex Gecko 2400/490 MHz Wireless Starter Kit (SLWSTK6062
- EFR32 Flex Gecko 2400/868 MHz Wireless Starter Kit (SLWSTK6061
- EFR32 Flex Gecko 2400/915 MHz Wireless Starter Kit (SLWSTK6060
- EFR32 Mighty Gecko 2.4 GHz Mesh Networking Kit (SLWSTK6000
- EFR32 Mighty Gecko 2.4 GHz Mesh Networking Kit (SLWSTK6000
- EFR32 Mighty Gecko 2400/434 MHz Wireless Starter Kit (SLWSTK6
- EFR32 Mighty Gecko 2400/868 MHz Wireless Starter Kit (SLWSTK6
- EFR32 Mighty Gecko 2400/915 MHz Wireless Starter Kit (SLWSTK6
- EFR32FG14 2400/169 MHz Wireless Starter Kit (SLWSTK6065B)
- EFR32FG14 2400/434 MHz Wireless Starter Kit (SLWSTK6063B)
- EFR32FG14 2400/490 MHz Wireless Starter Kit (SLWSTK6062B)
- EFR32FG14 2400/868 MHz Wireless Starter Kit (SLWSTK6061B)
- EFR32FG14 2400/915 MHz Wireless Starter Kit (SLWSTK6060B)
- EFR32xG21 2.4 GHz Mesh Networking Kit (SLWSTK6006A)

Selected Products

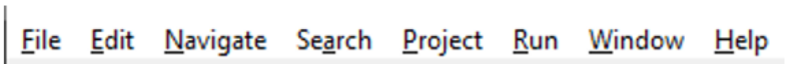
- EFR32 Blue Gecko Bluetooth Starter Kit (SLWSTK6020A)
- EFR32 Flex Gecko 2400/490 MHz Wireless Starter Kit (SLWSTK6062A)

- Remove the selected part
- Remove all content
- Collapse all
- Expand all

Menu

Menu

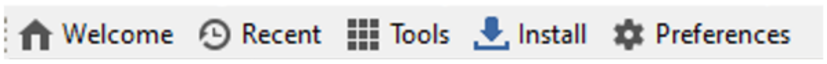
The perspective menu is available on all Simplicity Studio® 5 (SSv5) perspectives. The options available under each vary depending on the context.



Toolbar

Toolbar

The Simplicity Studio toolbar contains functions that are generally useful throughout Simplicity Studio® 5 (SSv5).

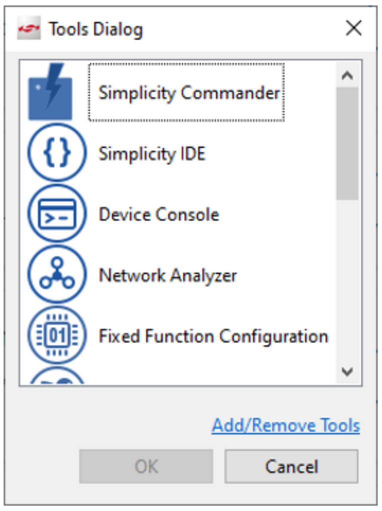


- Welcome returns you to the Launcher Welcome configuration.
- Recent shows you a list of recent projects. Select one to go to that project in the Simplicity IDE Project Explorer view.
- Tools provides a list of available tools. This list may have more selections than are available in the Launcher perspective Tools tab.
- Install brings you to a menu where you can install or uninstall software packages and review available updates.
- Preferences is a shortcut to the list of preferences also available through the menu selection Window > Preferences.

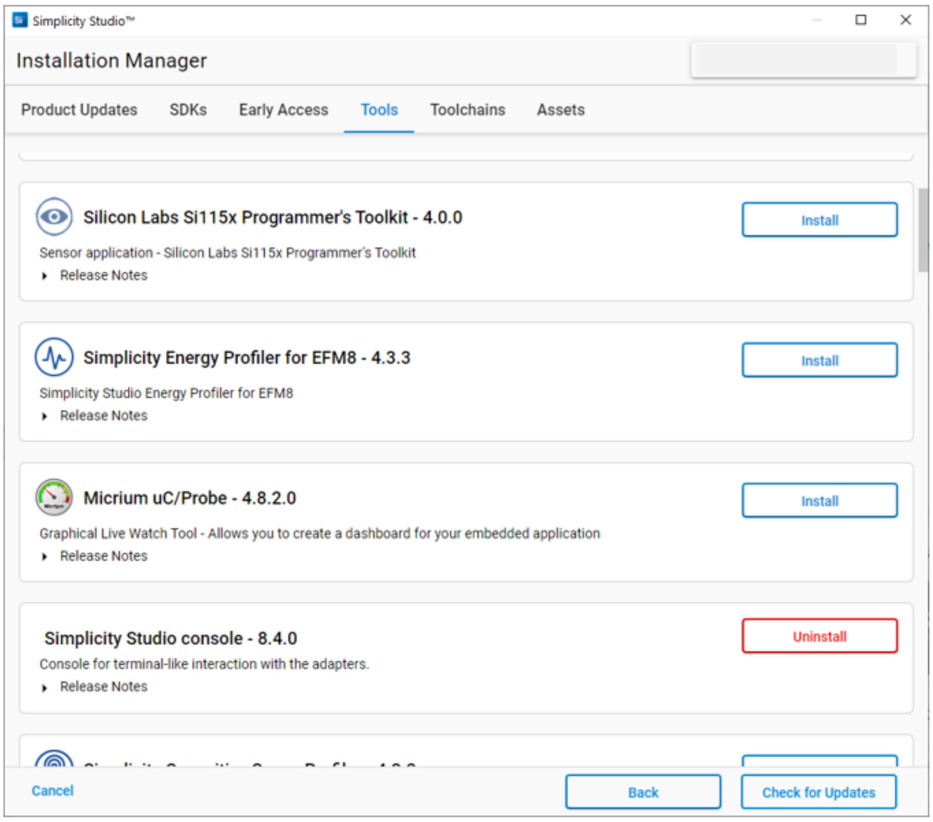
In addition, the section About Update Frequency describes how to manage the frequency with which Simplicity Studio checks for updates.

Tools

The Tools button opens a dialog where you can select a tool to open it, and also add and remove tools.

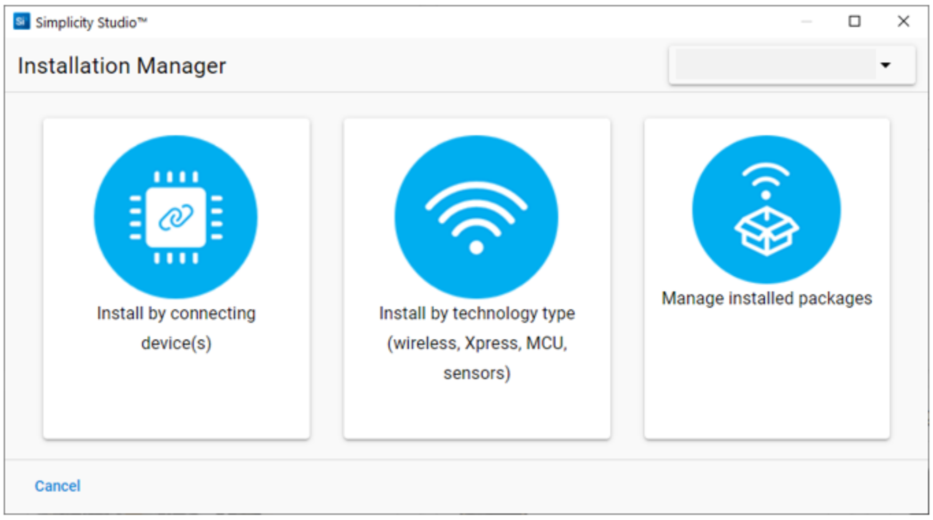


The Add/Remove Tools link opens the Installation Manager. Click **Manage Installed Packages** and go to the Tools tab, where you can add and remove tools.


Simplicity Studio™
Installation Manager
Product Updates
SDKs
Early Access
Tools
Toolchains
Assets
Silicon Labs Si115x Programmer's Toolkit - 4.0.0
Sensor application - Silicon Labs Si115x Programmer's Toolkit
Release Notes
Install
Simplicity Energy Profiler for EFM8 - 4.3.3
Simplicity Studio Energy Profiler for EFM8
Release Notes
Install
Micrium uC/Probe - 4.8.2.0
Graphical Live Watch Tool - Allows you to create a dashboard for your embedded application
Release Notes
Install
Simplicity Studio console - 8.4.0
Console for terminal-like interaction with the adapters.
Release Notes
Uninstall
Cancel
Back
Check for Updates

Install

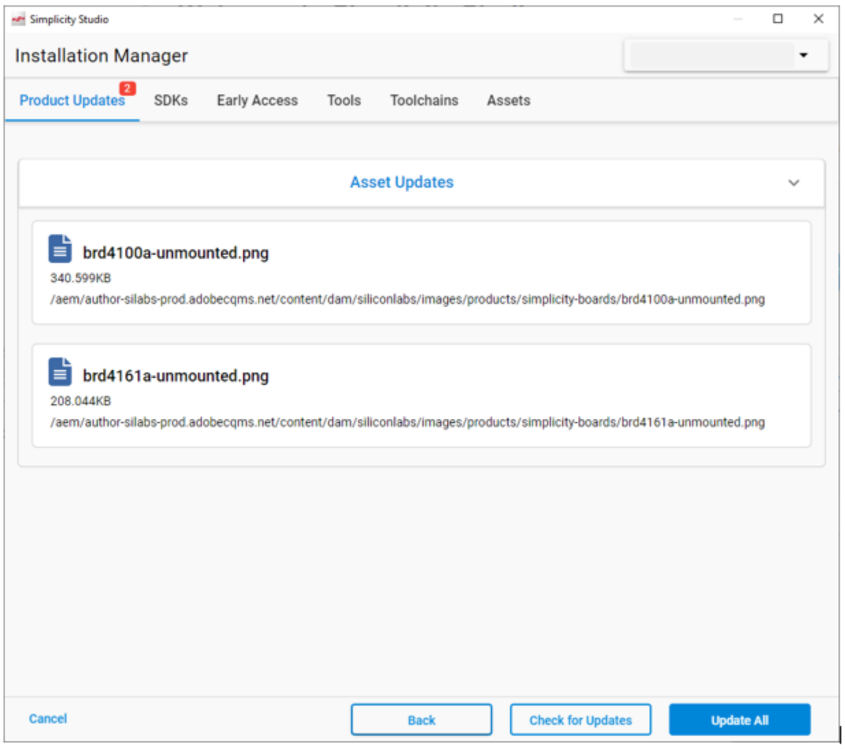
The Install button (red when updates are available) opens the Installation Manager.


Simplicity Studio™
Installation Manager
Install by connecting device(s)
Install by technology type (wireless, Xpress, MCU, sensors)
Manage installed packages
Cancel

Because available items are often controlled through your Silicon Labs account, it is recommended that you be logged in while using the Installation Manager. **Install by connecting device(s)** and **Install by technology type** are both described in Install SSv5 and Software. This section describes the **Manage installed packages** functionality.

Click **Manage installed packages** to open a tabbed interface where you can install and uninstall a variety of SSv5 components. Note that you can have more than one instance of some items, such as SDKs. You can use the Installation Manager to install additional instances as well as updates.

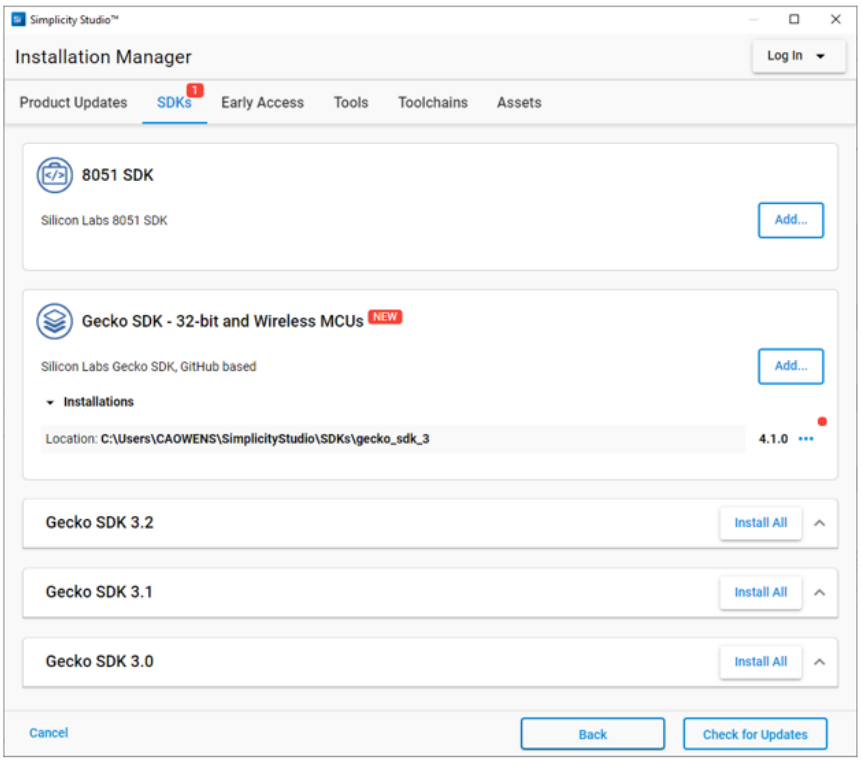
The interface opens on the Product Updates tab. Updates are grouped by type. If Product Updates are available these typically must be installed before you can install a new SDK.



brd4100a-unmounted.png
340.599KB
/aem/author-silabs-prod.adobecqms.net/content/dam/siliconlabs/images/products/simplicity-boards/brd4100a-unmounted.png

brd4161a-unmounted.png
208.044KB
/aem/author-silabs-prod.adobecqms.net/content/dam/siliconlabs/images/products/simplicity-boards/brd4161a-unmounted.png

The SDKs tab supports installing and uninstalling SDKs and extensions.



8051 SDK
Silicon Labs 8051 SDK

Gecko SDK - 32-bit and Wireless MCUs
Silicon Labs Gecko SDK, GitHub based

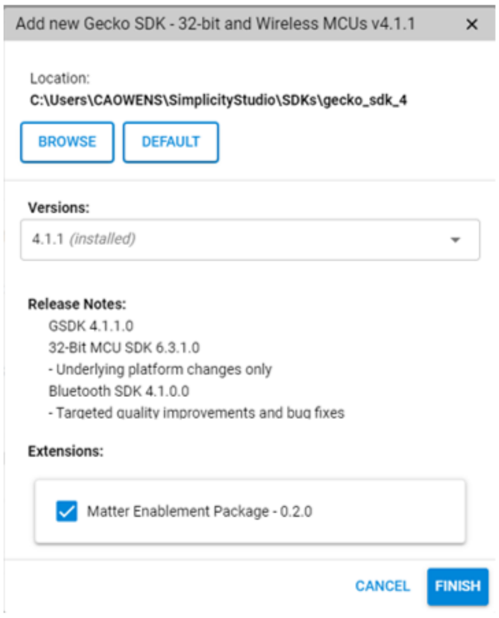
Location: C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk_3 4.1.0

Gecko SDK 3.2

Gecko SDK 3.1

Gecko SDK 3.0

Click **Add ...** to install another version or instance of the SDK. The dialog displays what is available as well as what is currently installed. Click **Browse** to select an installation location other than the default. Click **Default** to select the default location (\Users\USERNAME\SimplicityStudio\SDKs\gecko_sdk<_n>, where _n is added for the second and incremented for subsequent installations). Hover over the release notes to get a scroll control.



Add new Gecko SDK - 32-bit and Wireless MCUs v4.1.1

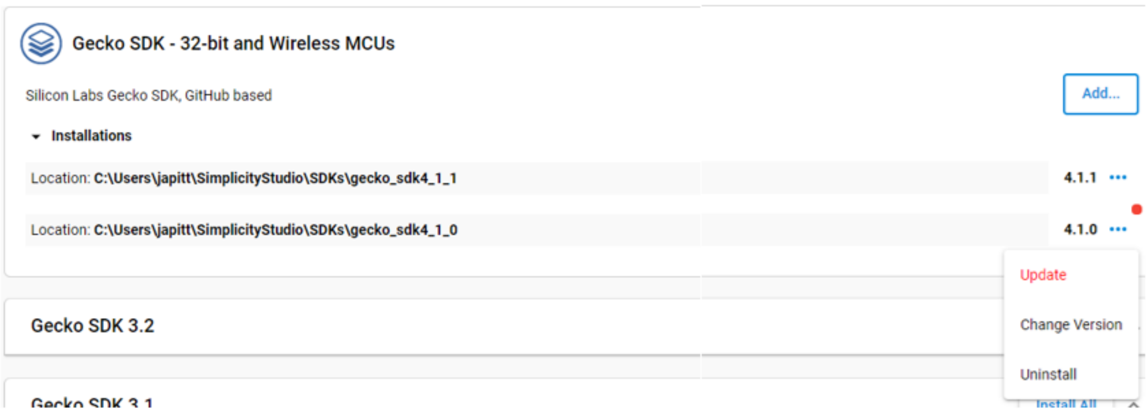
Location:
C:\Users\CAOWENS\SimplicityStudio\SDKs\gecko_sdk_4

Versions: 4.1.1 *(installed)*

Release Notes:
GSDK 4.1.1.0
32-Bit MCU SDK 6.3.1.0
- Underlying platform changes only
Bluetooth SDK 4.1.0.0
- Targeted quality improvements and bug fixes

Extensions:
☑ Matter Enablement Package - 0.2.0

Click **...** next to an SDK version to install an update to it, change the version, or uninstall that version.



Gecko SDK - 32-bit and Wireless MCUs

Silicon Labs Gecko SDK, GitHub based

Installations

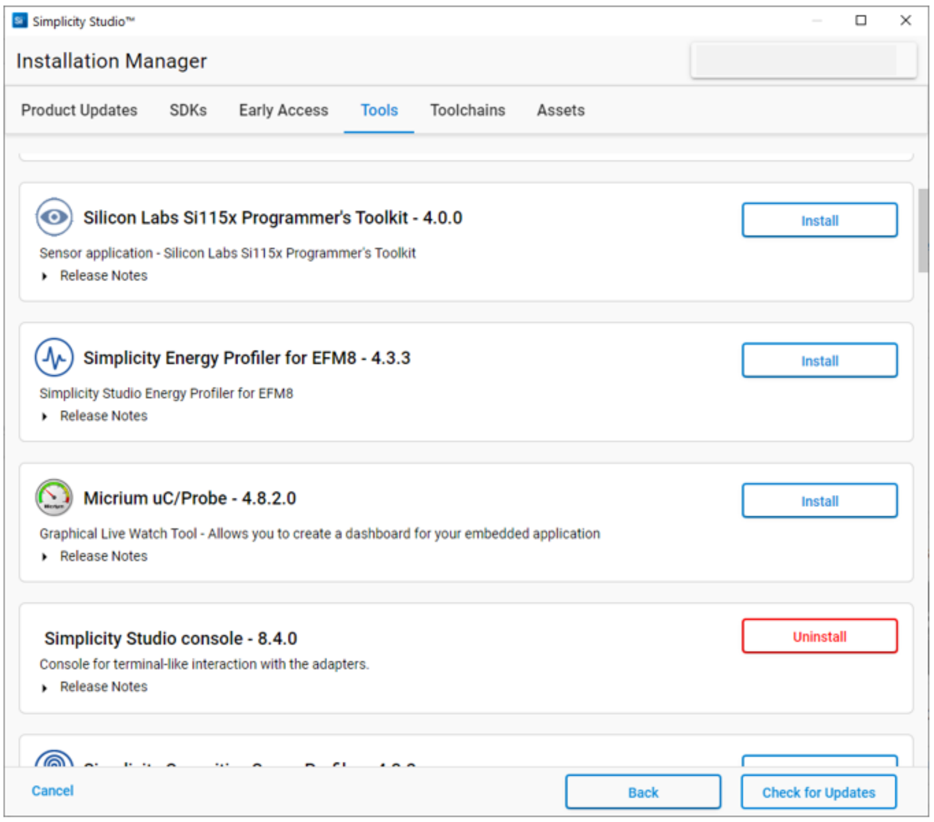
Location: C:\Users\japitt\SimplicityStudio\SDKs\gecko_sdk4_1_1 4.1.1

Location: C:\Users\japitt\SimplicityStudio\SDKs\gecko_sdk4_1_0 4.1.0

Gecko SDK 3.2

The Early Access tab is used to provide limited access for some customers, usually for beta testing.

The Tools tab shows available and installed tools, regardless of your connected or selected hardware. Summary release notes are provided.



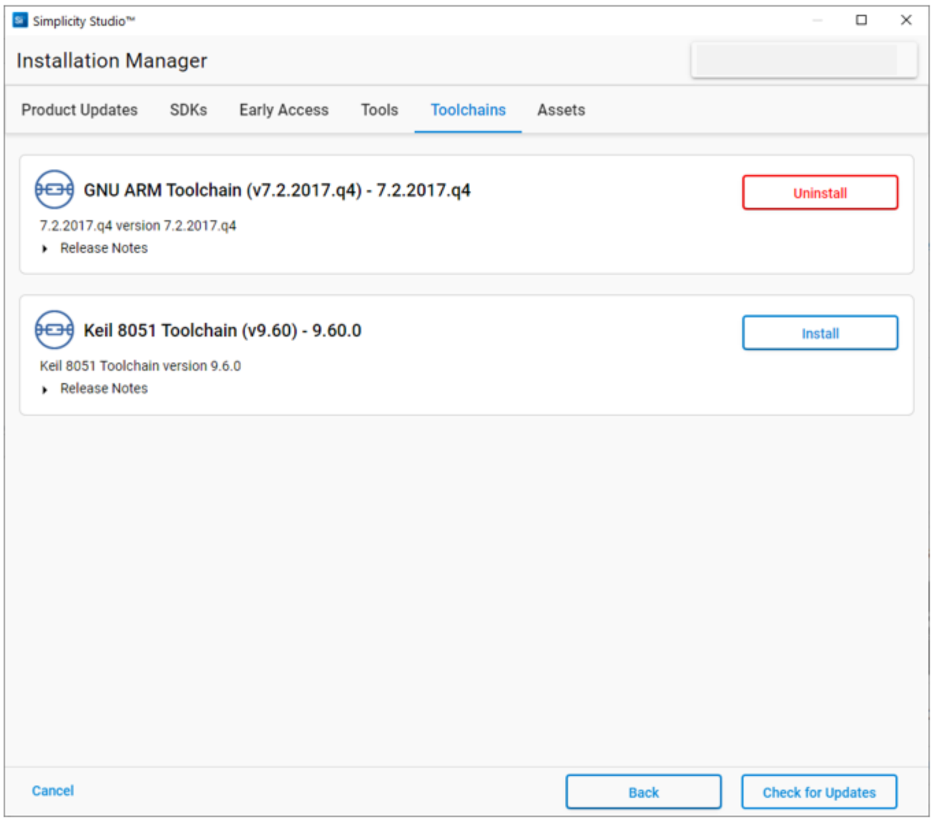
Silicon Labs Si115x Programmer's Toolkit - 4.0.0
Sensor application - Silicon Labs Si115x Programmer's Toolkit
Release Notes

Simplicity Energy Profiler for EFM8 - 4.3.3
Simplicity Studio Energy Profiler for EFM8
Release Notes

Micrium uC/Probe - 4.8.2.0
Graphical Live Watch Tool - Allows you to create a dashboard for your embedded application
Release Notes

Simplicity Studio console - 8.4.0
Console for terminal-like interaction with the adapters.
Release Notes

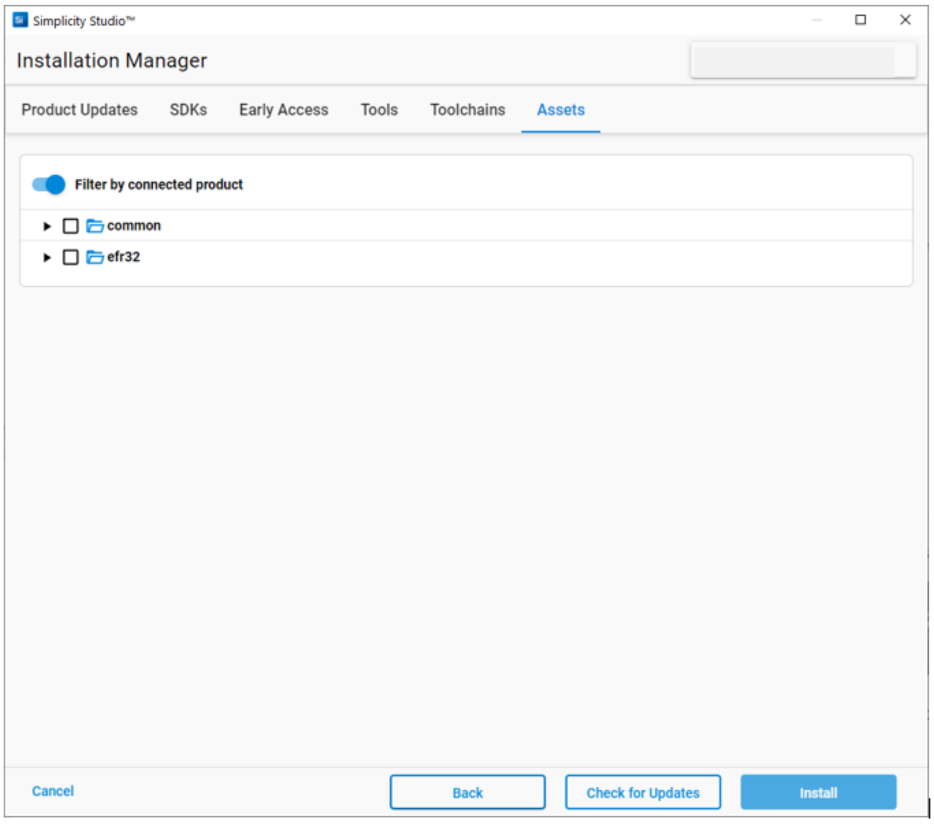
The Toolchains tab shows available and installed toolchains.



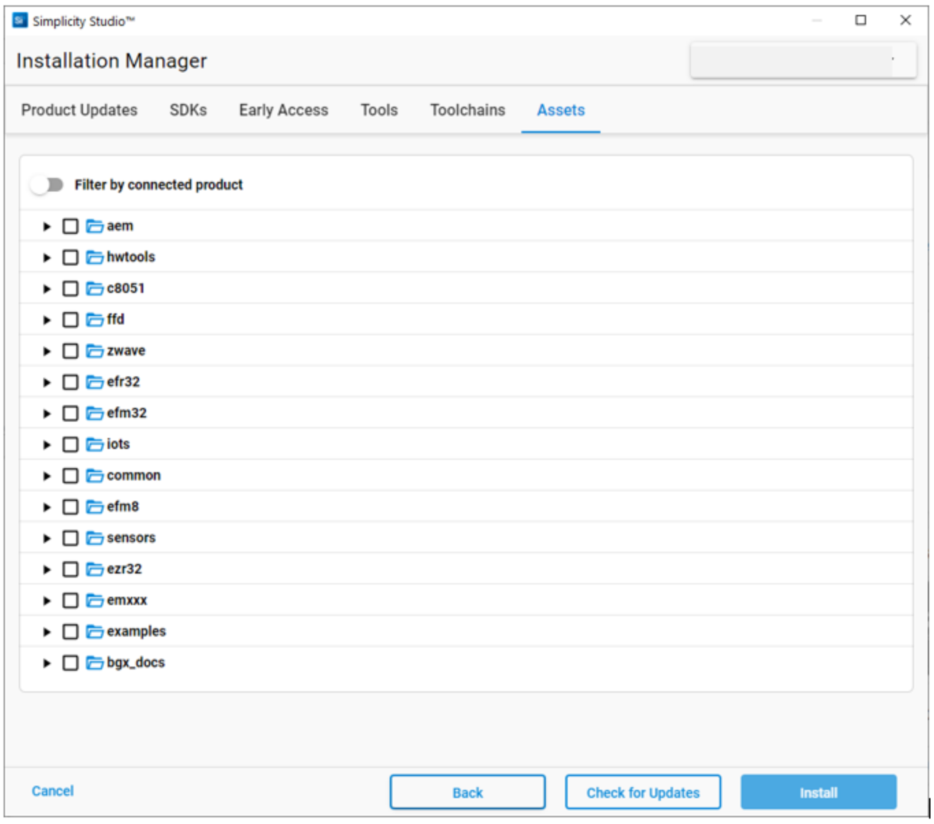
GNU ARM Toolchain (v7.2.2017.q4) - 7.2.2017.q4
7.2.2017.q4 version 7.2.2017.q4
Release Notes

Keil 8051 Toolchain (v9.60) - 9.60.0
Keil 8051 Toolchain version 9.6.0
Release Notes

The Assets tab shows available assets. Assets include items such as documents, photographs, code, and so on. Check the item(s) of interest and click **Install**. The default view is filtered by connected product.



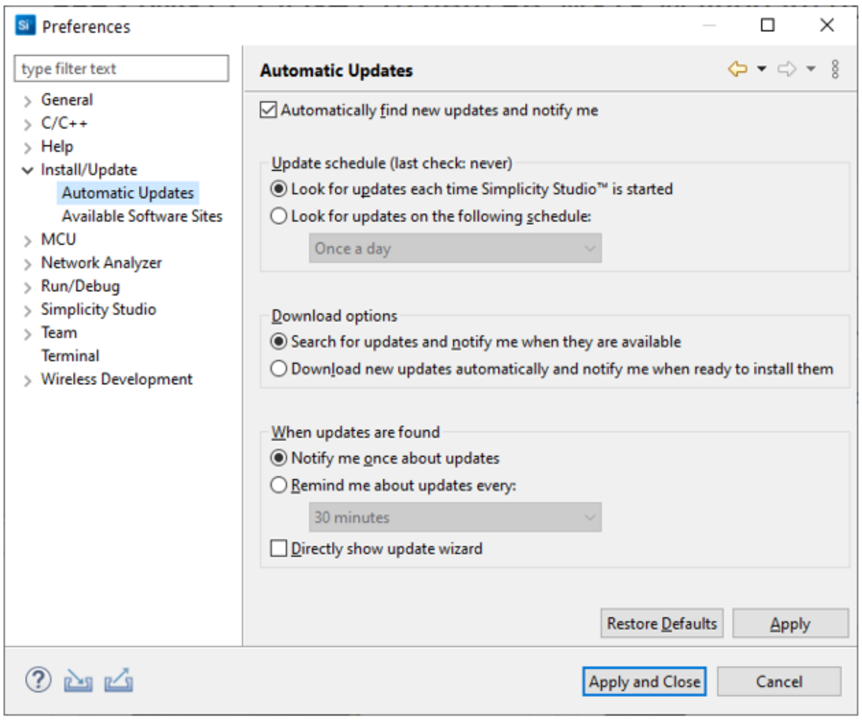
The unfiltered view provides a much larger set of options.



After you install or uninstall items, for example as a result of an update or because you want to manage your SDK installations, SSv5 determines if a restart is required.

About Update Frequency

By default, SSv5 checks for updates when you first open it. You can manage update frequency through Preferences > Install/Update > Automatic Updates.



Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
  - Automatic Updates
  - Available Software Sites
- MCU
- Network Analyzer
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

Automatic Updates

[✓] Automatically find new updates and notify me

Update schedule (last check: never)
- (●) Look for updates each time Simplicity Studio™ is started
- ( ) Look for updates on the following schedule:
  Once a day

Download options
- (●) Search for updates and notify me when they are available
- ( ) Download new updates automatically and notify me when ready to install them

When updates are found
- (●) Notify me once about updates
- ( ) Remind me about updates every:
  30 minutes

[ ] Directly show update wizard

Restore Defaults | Apply

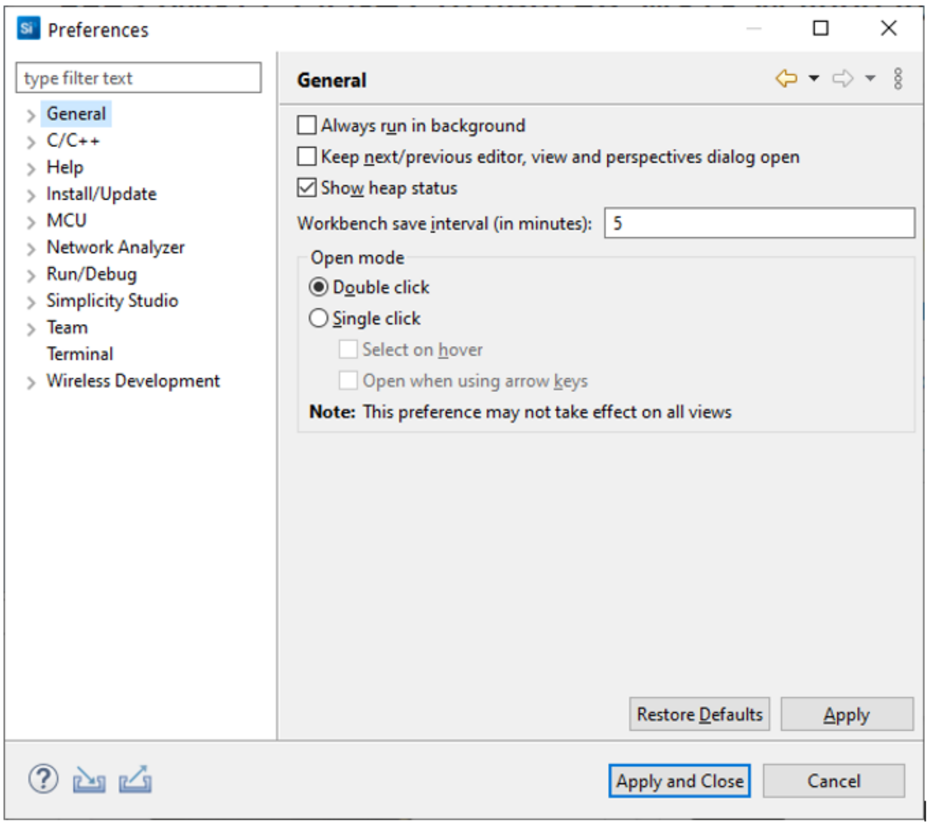
Apply and Close | Cancel

- Automatically find new updates and notify me (default: On): When selected, SSv5 automatically searches for update, as defined by the update schedule.
- Update schedule (default: On startup): Look for updates on each startup, or once a day or some day a week, at a predefined time.
- Download options (default: Search and Notify): Choose if SSv5 will search for updates and notify you of them once they are available or automatically download new updates and ask you to install them.
- When updates are found (default: Once): Choose to be notified about new updates only once, or to receive reminders until the updates are installed.

If you have turned automatic updates off, you can check for updates by clicking **Install** on the toolbar, then **Manage Installed Packages**, and then **Check for Updates** on the Product Updates tab.

Preferences

The Preferences button is a shortcut to the Windows > Preferences dialog (App menu on macOS).



Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

General

- [ ] Always run in background
- [ ] Keep next/previous editor, view and perspectives dialog open
- [x] Show heap status

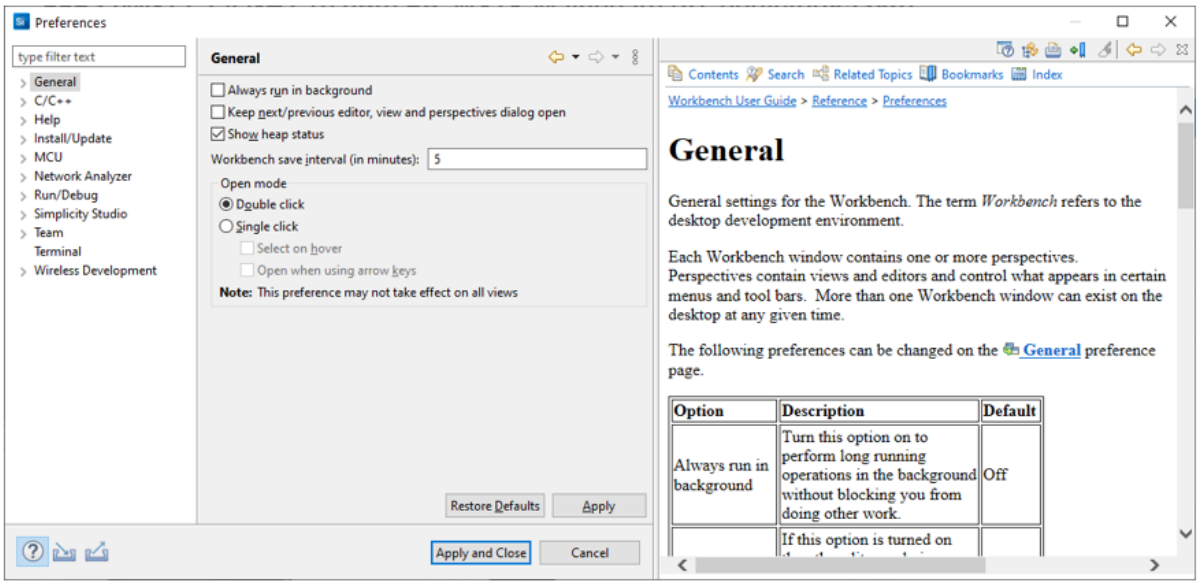
Workbench save interval (in minutes): 5

Open mode

- [✓ Double click]
- Single click
  - [ ] Select on hover
  - [ ] Open when using arrow keys

Note: This preference may not take effect on all views

Frequently used preferences are discussed in context of their use cases. Context-sensitive help is available for most preferences.



Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

General

- [ ] Always run in background
- [ ] Keep next/previous editor, view and perspectives dialog open
- [x] Show heap status

Workbench save interval (in minutes): 5

Open mode

- [✓ Double click]
- Single click
  - [ ] Select on hover
  - [ ] Open when using arrow keys

Note: This preference may not take effect on all views

Workbench User Guide > Reference > Preferences

General

General settings for the Workbench. The term *Workbench* refers to the desktop development environment.

Each Workbench window contains one or more perspectives. Perspectives contain views and editors and control what appears in certain menus and tool bars. More than one Workbench window can exist on the desktop at any given time.

The following preferences can be changed on the **General** preference page.

Option	Description	Default
Always run in background	Turn this option on to perform long running operations in the background without blocking you from doing other work.	Off
	If this option is turned on	

The arrow controls in the upper-right enable navigation through previously viewed pages. To return to a page after viewing several pages, click the drop-down arrow to display a list of recently viewed pages.

If you have made changes in preferences and wish to move them to a different installation, use the Import and Export controls in the lower left.

About the Simplicity IDE

About the Simplicity IDE

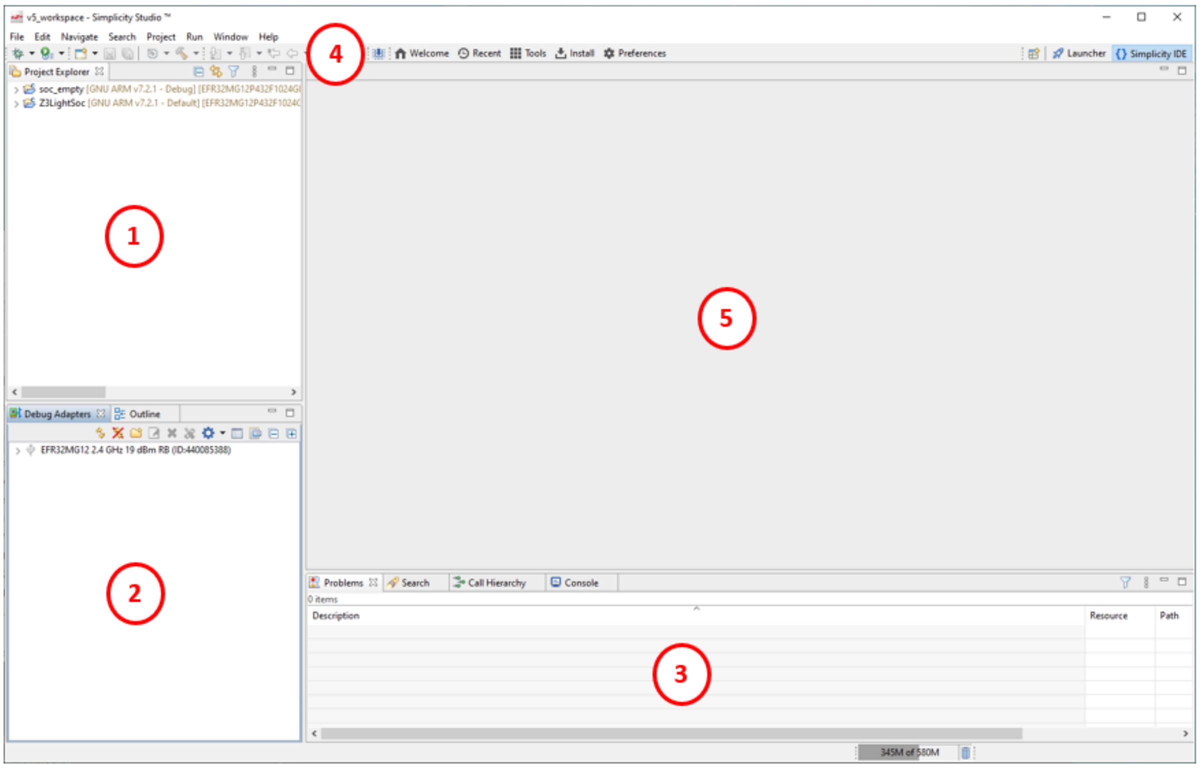
The Simplicity Studio® 5 (SSv5) **Simplicity Integrated Development Environment (IDE)** perspective is designed to support code editing, downloading, and debugging for EFR32 devices and modules and EFM32, EFM8, and 8051 devices.

These pages review the Simplicity IDE User Interface, discuss how to import and export projects, and explore the features supporting code editing.

User Interface Review

User Interface Review

The Simplicity IDE perspective is made up of three views areas, the editor area, and a toolbar.

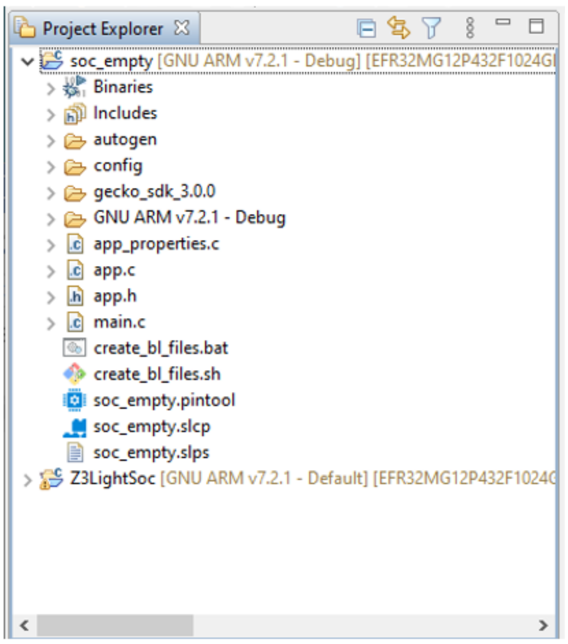

v5_workspace - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
soc_empty [GNU ARM v7.2.1 - Debug] [EFR32MG12P432F1024G
Z3LightSoc [GNU ARM v7.2.1 - Default] [EFR32MG12P432F1024
Debug Adapters
Outline
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
Problems
Search
Call Hierarchy
Console
0 items
Description
Resource
Path
345M of 580M
1
2
3
4
5

1. Project Explorer View
2. Debug Adapters View
3. Other Views
4. Toolbar
5. Editor area

The Editor area is populated with whatever tool is appropriate for the device and project, such as Project Configurator or the 8-bit device Hardware Configurator.

Project Explorer View

The Project Explorer view shows project files in the current workspace.


Project Explorer

soc_empty [GNU ARM v7.2.1 - Debug] [EFR32MG12P432F1024G
- Binaries
- Includes
- autogen
- config
- gecko_sdk_3.0.0
- GNU ARM v7.2.1 - Debug
- app_properties.c
- app.c
- app.h
- main.c
- create_bl_files.bat
- create_bl_files.sh
- soc_empty.pintool
- soc_empty.slcp
- soc_empty.slps

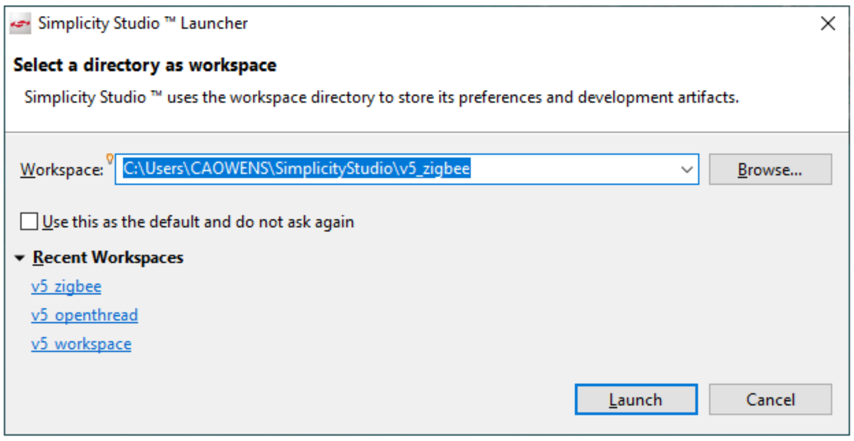
Z3LightSoc [GNU ARM v7.2.1 - Default] [EFR32MG12P432F1024G

A workspace is a directory that includes some hidden metadata, and must be created through SSv5. The default Windows workspace is:

\Users\<username>\SimplicityStudio\v5_workspace\

Maintaining separate workspaces for different GSDK versions or compiler versions may be necessary in some cases or may help improve SSv5 Performance. See Tips and Tricks for more information.

Each SSv5 session has a single default workspace. To change workspaces or to create a new workspace, use the **File > Switch Workspace** function. To create a new workspace click **Browse** and add a new directory. After you add or switch, SSv5 restarts.


Simplicity Studio ™ Launcher

Select a directory as workspace

Simplicity Studio ™ uses the workspace directory to store its preferences and development artifacts.

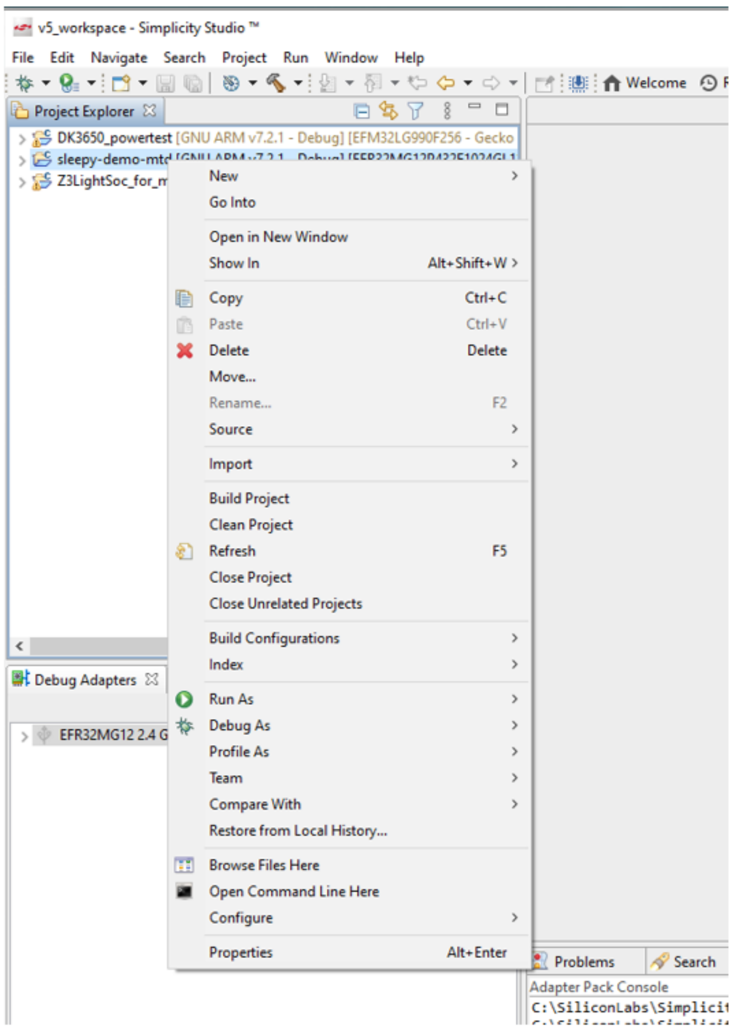
Workspace: C:\Users\CAOWENS\SimplicityStudio\v5_zigbee   Browse...

☐ Use this as the default and do not ask again

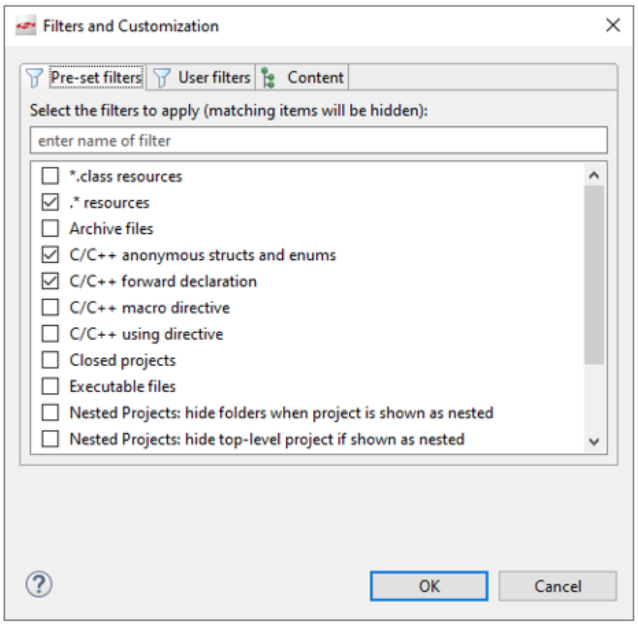
Recent Workspaces
- v5_zigbee
- v5_openthread
- v5_workspace

Launch   Cancel

Right-click a project directory for a context menu of project-level functions.



The Filter functionality available through the view toolbar can be used to limit the files shown.



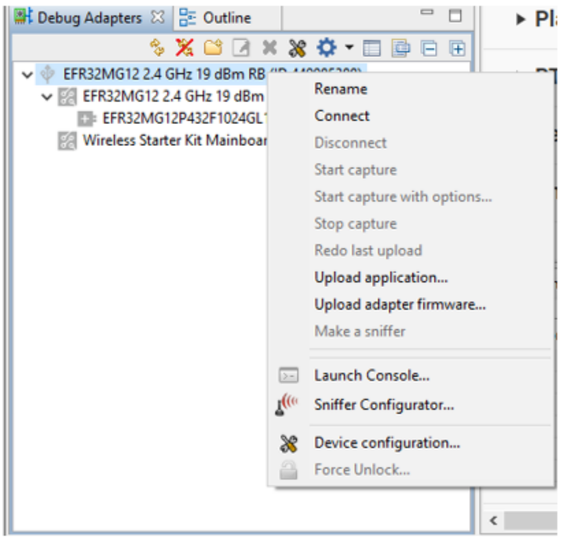
Select the filters to apply (matching items will be hidden):

enter name of filter

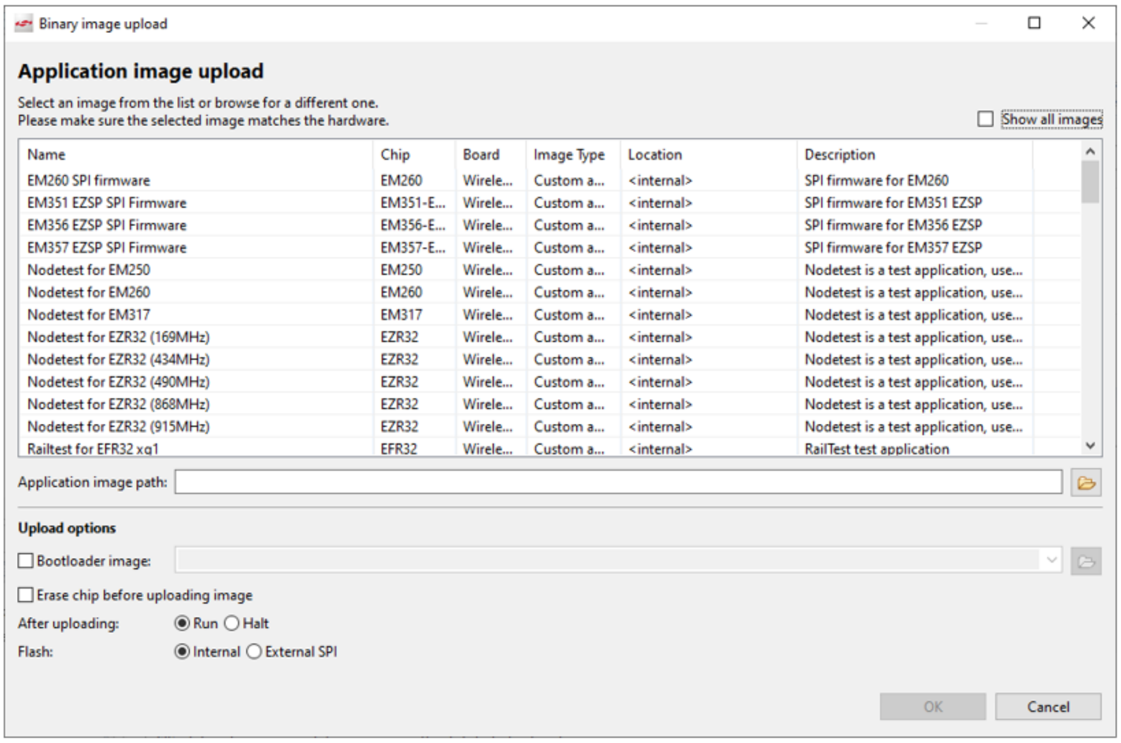
- ☐ *.class resources
- ☑ .* resources
- ☐ Archive files
- ☑ C/C++ anonymous structs and enums
- ☑ C/C++ forward declaration
- ☐ C/C++ macro directive
- ☐ C/C++ using directive
- ☐ Closed projects
- ☐ Executable files
- ☐ Nested Projects: hide folders when project is shown as nested
- ☐ Nested Projects: hide top-level project if shown as nested

Debug Adapters View

The debug adapters tab in this view is the same as that in the Launcher perspective.



When the adapter is disconnected, an **Upload application** option is available. This has the advantage of allowing you to flash both an application image and a bootloader image at the same time, and also to see the various pre-compiled images available based on the installed SDKs, along with their compatible hardware.


Binary image upload

Application image upload

Select an image from the list or browse for a different one.
Please make sure the selected image matches the hardware.

☐ Show all images

Name	Chip	Board	Image Type	Location	Description
EM260 SPI firmware	EM260	Wirele...	Custom a...	<internal>	SPI firmware for EM260
EM351 EZSP SPI Firmware	EM351-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM351 EZSP
EM356 EZSP SPI Firmware	EM356-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM356 EZSP
EM357 EZSP SPI Firmware	EM357-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM357 EZSP
Nodetest for EM250	EM250	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM260	EM260	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM317	EM317	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (169MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (434MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (490MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (868MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (915MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Railtest for EFR32 xg1	EFR32	Wirele...	Custom a...	<internal>	RailTest test application

Application image path:

Upload options

☐ Bootloader image:

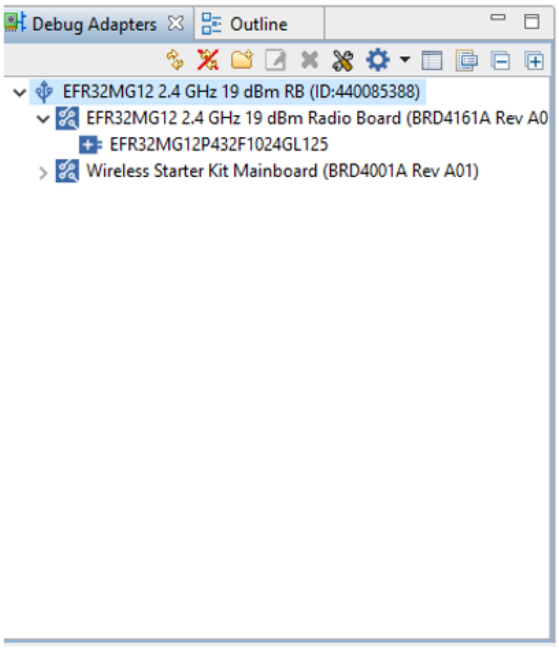
☐ Erase chip before uploading image

After uploading: [✓ Run] Halt

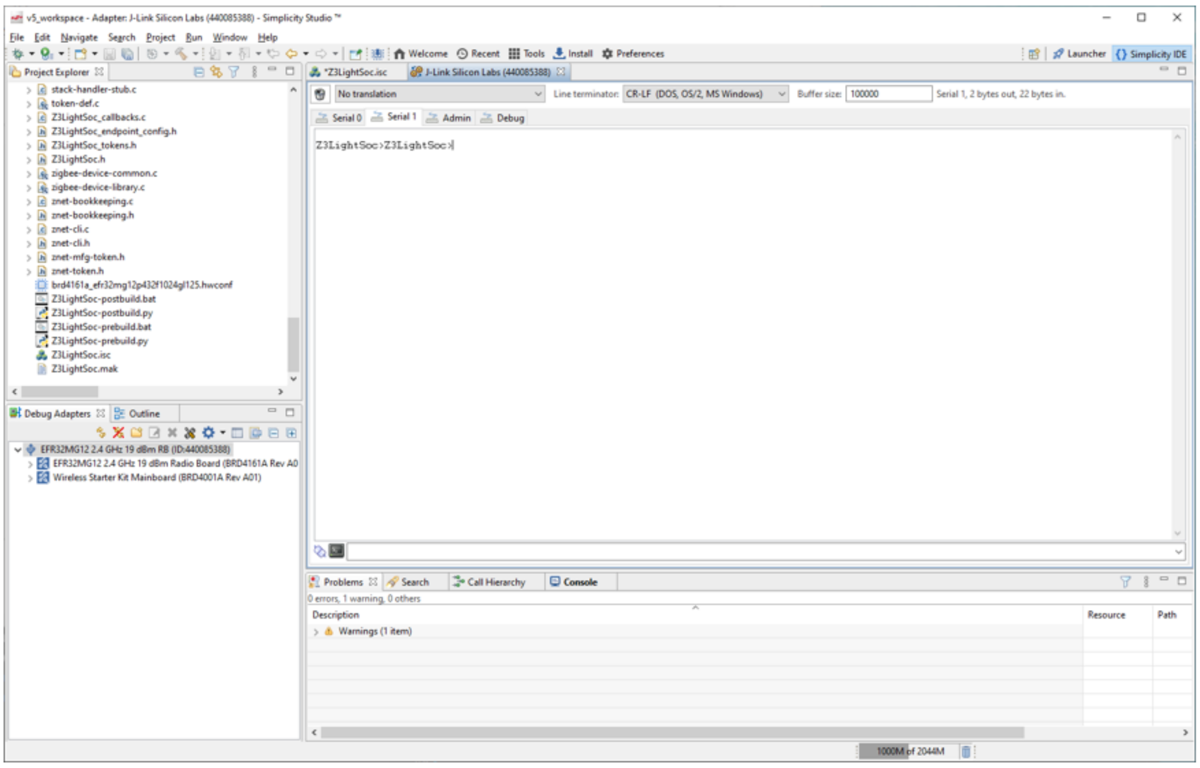
Flash: [✓ Internal] External SPI

OK Cancel

When you connect to an adapter, the icons turn blue.


Debug Adapters
Outline
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A0
EFR32MG12P432F1024GL125
Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Once you have connected, **Start capture** and **Start capture with options** open the Network Analyzer interface for that device. For many devices, **Launch console** opens up a console in the editor area, where you can interact with the device. This option is not available with EFM8 / C8051 starter or development kits (STKs or DKs).



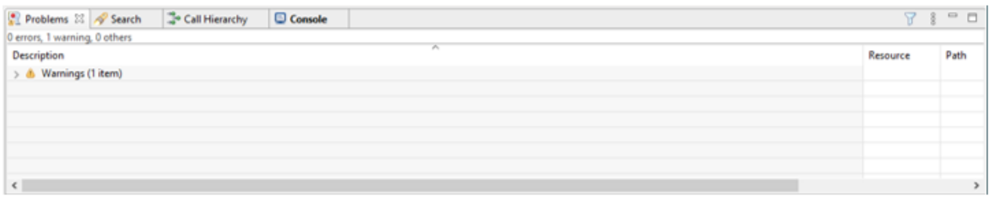
No translation
Line terminator: CR-LF (DOS, OS/2, MS Windows)
Buffer size: 100000
Serial 1, 2 bytes out, 22 bytes in.

```
Z3LightSoc>Z3LightSoc>
```

0 errors, 1 warning, 0 others

Other Views

Other views are available in the pane below the editor area.



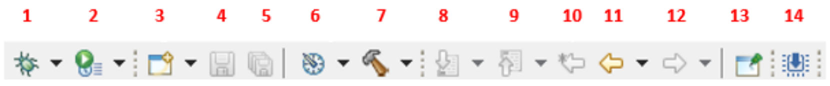
0 errors, 1 warning, 0 others

Description	Resource	Path
> Warnings (1 item)		

- **Problems** shows any problems detected in an open file or a process.
- **Search** shows the results of the most recent search. The toolbar allows you to retrieve previous searches.
- **Call Hierarchy** shows the results of Open Call Hierarchy when using the IDE for code development.
- **Console** shows activity such as flashing an image to the device and build results.

Toolbar

The Simplicity IDE toolbar offers debug and other code and file options.



1 **Debug** builds an image in debug mode, flashes it, and opens the debugger.

2 **Profile as** opens the Energy Profiler.

3 **New** opens a menu where you can create a variety of items, including a new project (equivalent to **File > New > Project**).

4 **Save** saves a changed file.

5 **Save all** saves all changed files.

6 **Mode** allows change between debug and release mode for the applicable toolchain.

7 **Build** builds the selected project file in the default mode (debug or release).

When working with code files, **8 Next** and **9 Previous annotation** allow you to select the type of annotation and move to them.

10 Edit last location opens the last file you had selected.

11 Back and 12 Forward to file allow you to move to the previous or next file, or to select a file from a list.

13 Pin editor links the selection in the Project Explorer view to its editor tab.

14 Flash programmer opens the Flash programmer tool, also available through the Tools menu.

Import and Export

Import and Export

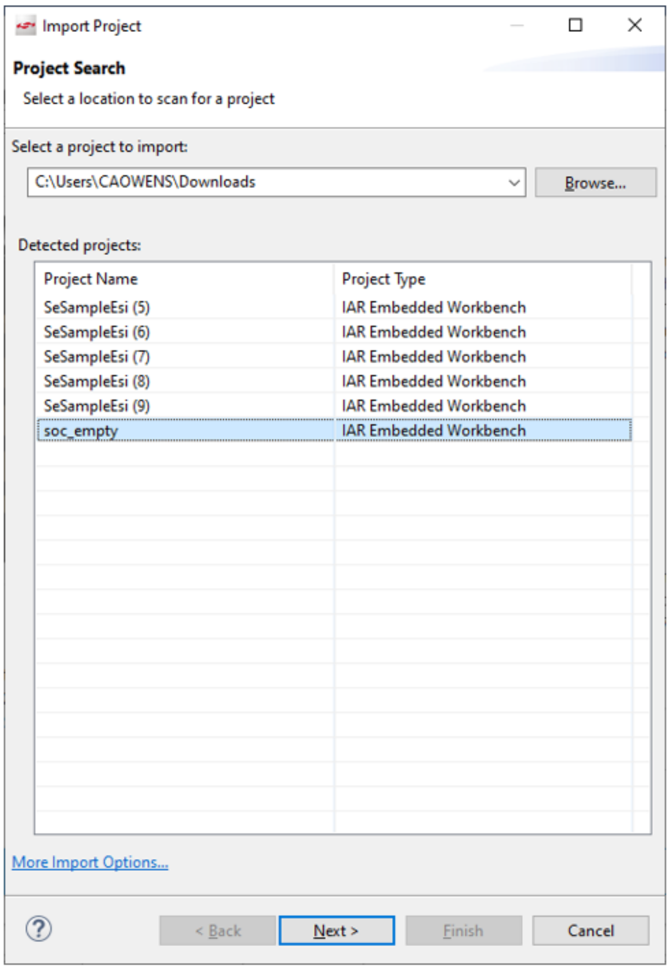
Simplicity Studio® 5 (SSv5) File > Import allows you to import a project from Keil® µVision4®, IAR Embedded Workbench, or the 8-bit Silicon Labs IDE into Simplicity IDE. You can also import a Silicon Labs Configurator project (.slcp) file to create a project in the Simplicity IDE, or a Silicon Labs export (.sls) file to import a project or a solution.

File > Export allows you to export a project or solution in a variety of formats.

Import

Place the file to be imported in a folder.

Select **File > Import**, and locate the folder with the project or solution to be imported.

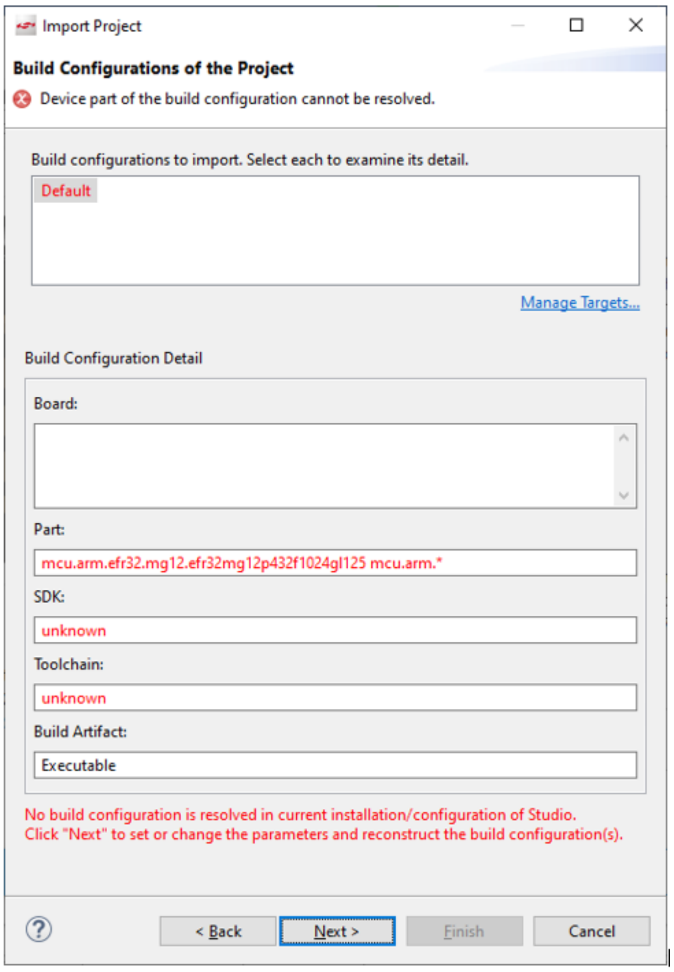


Select a project to import: C:\Users\CAOWENS\Downloads

Detected projects:

Project Name	Project Type
SeSampleEsi (5)	IAR Embedded Workbench
SeSampleEsi (6)	IAR Embedded Workbench
SeSampleEsi (7)	IAR Embedded Workbench
SeSampleEsi (8)	IAR Embedded Workbench
SeSampleEsi (9)	IAR Embedded Workbench
soc_empty	IAR Embedded Workbench

Select the project and click **Next**. If anything about the project is unresolved you can resolve it.



Import Project

Build Configurations of the Project

Device part of the build configuration cannot be resolved.

Build configurations to import. Select each to examine its detail.

Default

Manage Targets...

Build Configuration Detail

Board:

Part:

mcu.arm.efr32.mg12.efr32mg12p432f1024gl125 mcu.arm.*

SDK:

unknown

Toolchain:

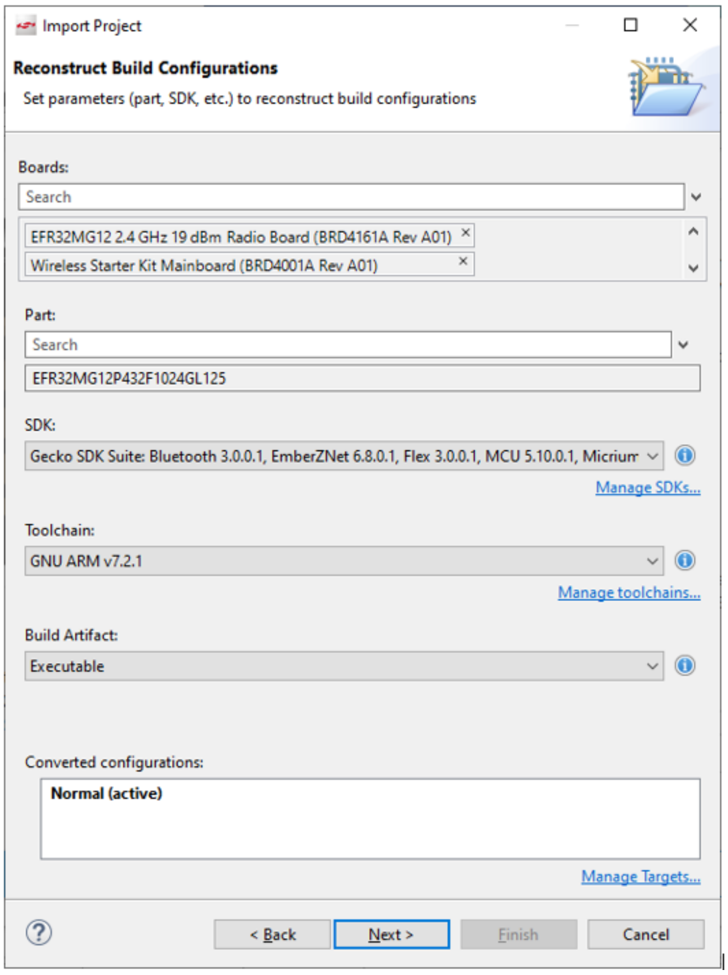
unknown

Build Artifact:

Executable

No build configuration is resolved in current installation/configuration of Studio.
Click "Next" to set or change the parameters and reconstruct the build configuration(s).

< Back | Next > | Finish | Cancel


Import Project

Reconstruct Build Configurations

Set parameters (part, SDK, etc.) to reconstruct build configurations

Boards:

Search

EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Part:

Search

EFR32MG12P432F1024GL125

SDK:

Gecko SDK Suite: Bluetooth 3.0.0.1, EmberZNet 6.8.0.1, Flex 3.0.0.1, MCU 5.10.0.1, Micrium

Manage SDKs...

Toolchain:

GNU ARM v7.2.1

Manage toolchains...

Build Artifact:

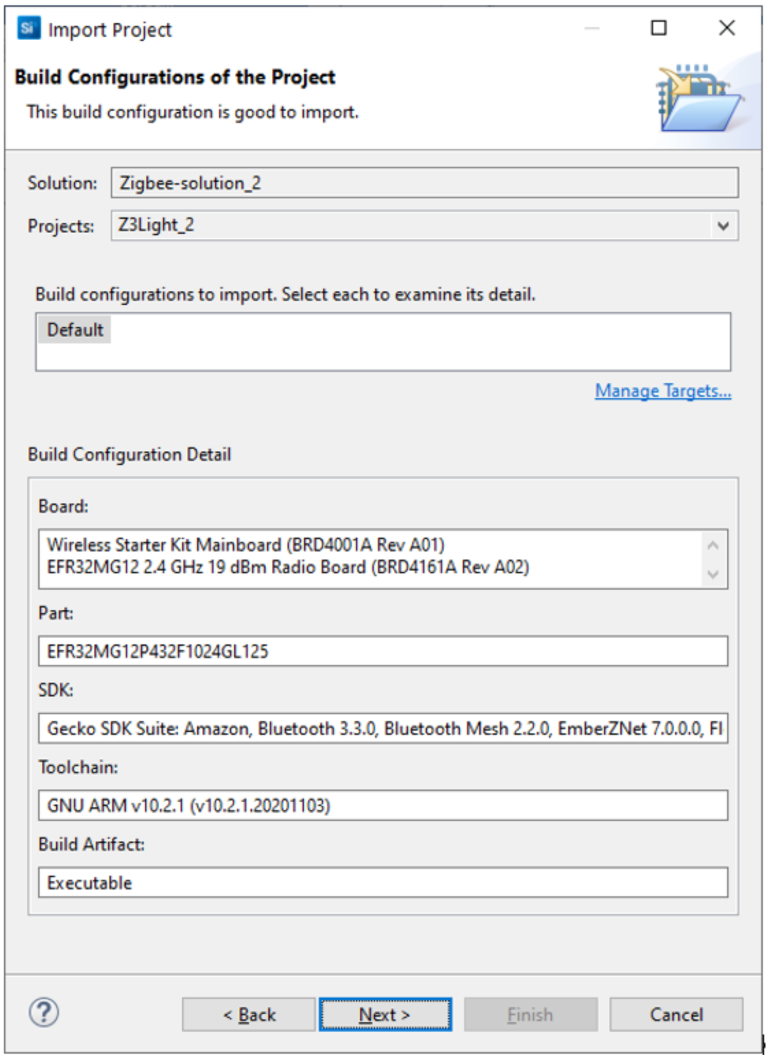
Executable

Converted configurations:

Normal (active)

Manage Targets...

If you are importing a solution, you can see the projects that are included in the solution.



Build Configurations of the Project

This build configuration is good to import.

Solution: Zigbee-solution_2

Projects: Z3Light_2

Build configurations to import. Select each to examine its detail.

Default

Manage Targets...

Build Configuration Detail

Board:

Wireless Starter Kit Mainboard (BRD4001A Rev A01)
EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A02)

Part:

EFR32MG12P432F1024GL125

SDK:

Gecko SDK Suite: Amazon, Bluetooth 3.3.0, Bluetooth Mesh 2.2.0, EmberZNet 7.0.0.0, Fl

Toolchain:

GNU ARM v10.2.1 (v10.2.1.20201103)

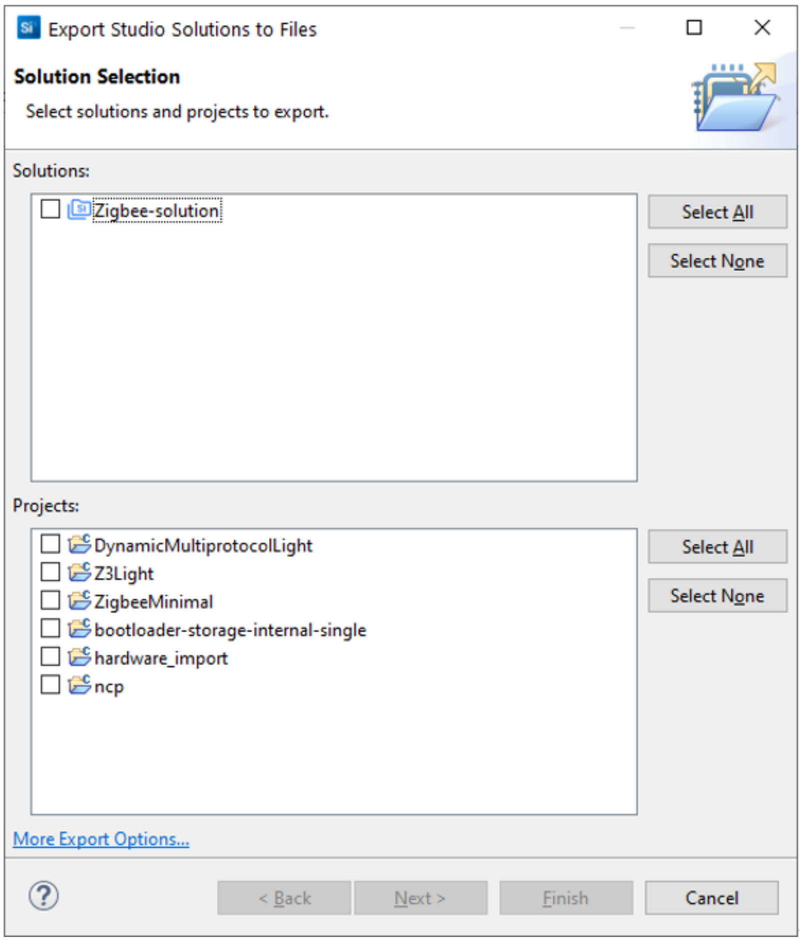
Build Artifact:

Executable

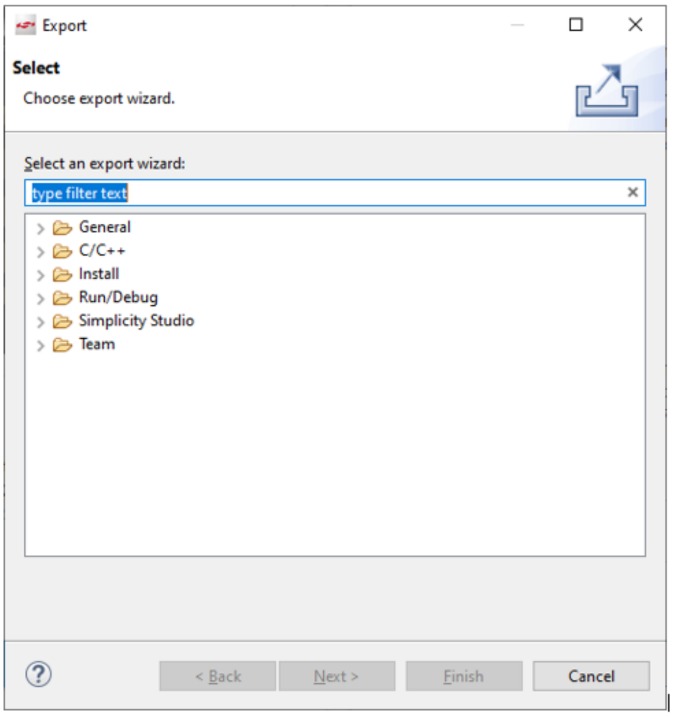
Click **Next**. Name the project or solution and click **Finish**.

Export

Select **File > Export**, and select the file(s) and/or solution(s) to be exported.



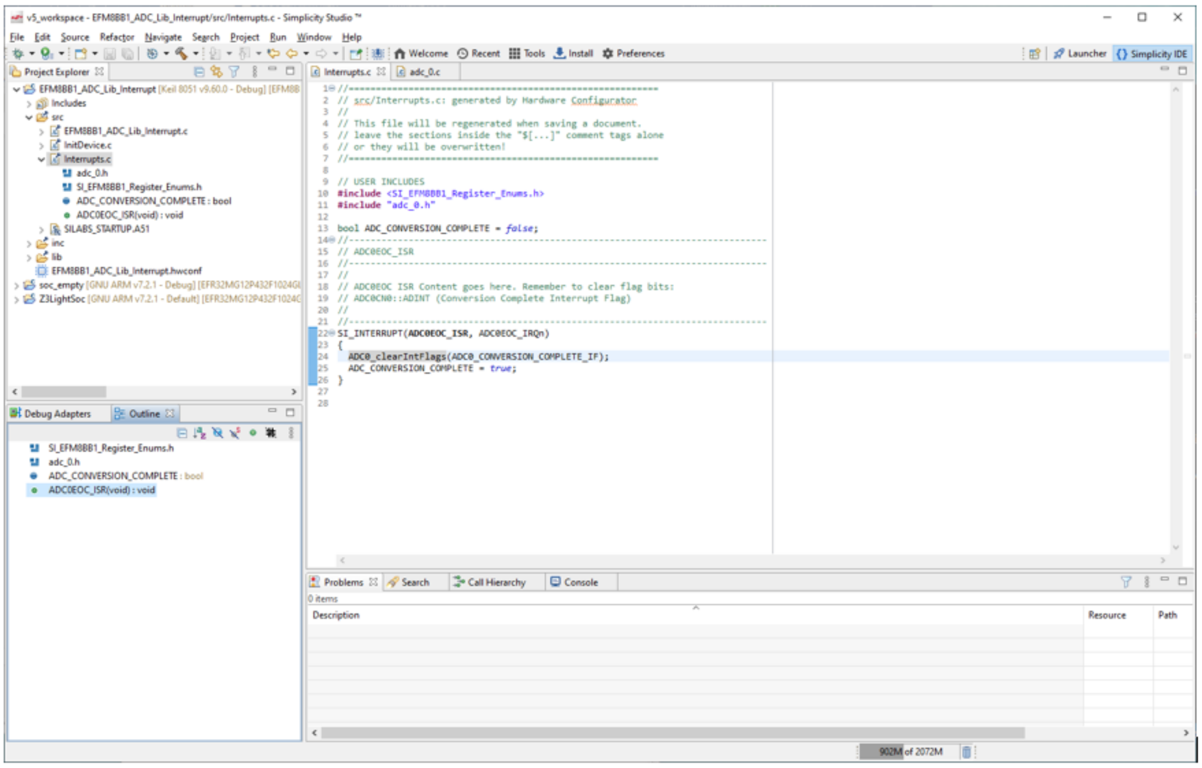
The default export format is an .sls file, which can easily be re-imported into SSv5. However, other options are available. Click **More export options...** and select a wizard.



Code Editing

Code Editing

Simplicity Studio® 5 (SSv5)'s Simplicity IDE is a code editing and development environment. The editor includes context highlighting, reference searching, and standard features found in any modern editor.



```
//==============================================================
// src/Interrupts.c: generated by Hardware Configurator
//
// This file will be regenerated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!
//==============================================================

// USER INCLUDES
#include <SI_EFM8BB1_Register_Enums.h>
#include "adc_0.h"

bool ADC_CONVERSION_COMPLETE = false;
//-----------------------------------------------------------------------------
// ADC0EOC_ISR
//-----------------------------------------------------------------------------
//
// ADC0EOC ISR Content goes here. Remember to clear flag bits:
// ADC0CN0::ADINT (Conversion Complete Interrupt Flag)
//
//-----------------------------------------------------------------------------
SI_INTERRUPT(ADC0EOC_ISR, ADC0EOC_IRQn)
{
  ADC0_clearIntFlags(ADC0_CONVERSION_COMPLETE_IF);
  ADC_CONVERSION_COMPLETE = true;
}
```

Note that the Outline view tab in the Debug Adapters area provides a code outline and a toolbar to control what shows in the Outline view.

This page includes information on:

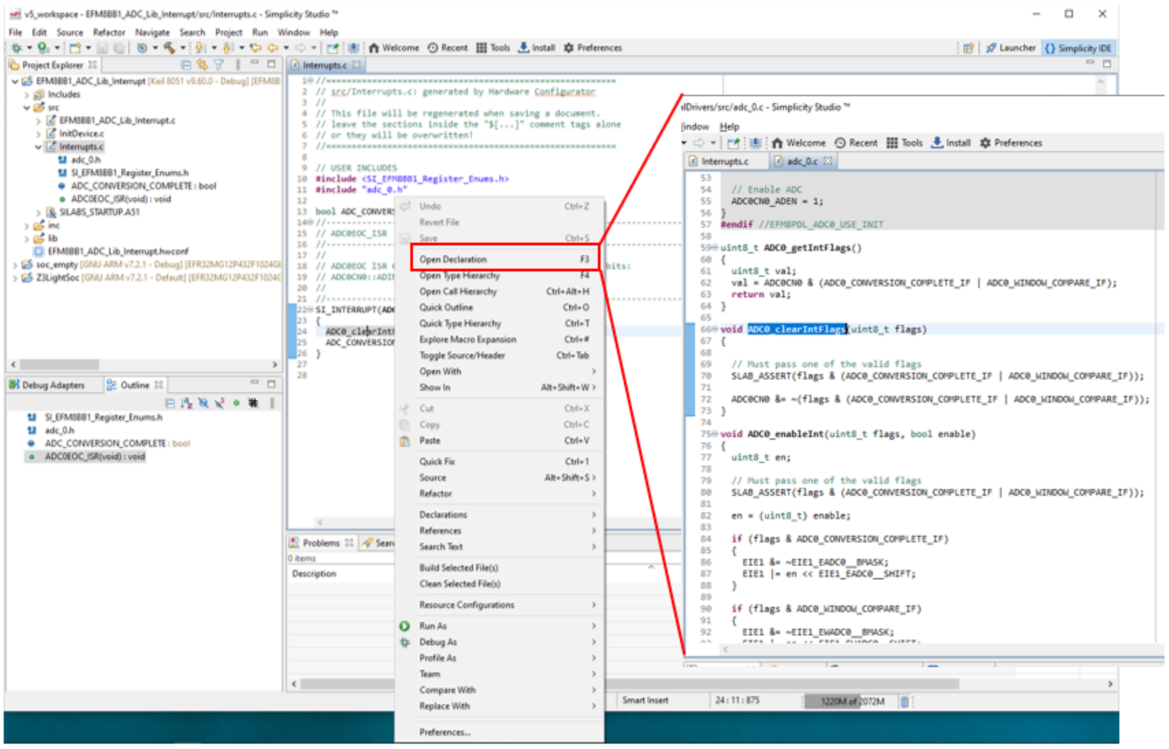
- Open Declaration
- Content Assist
- Link with Editor
- Symbol Expansion
- Task View
- Quick-Access Console
- Call Hierarchy

Open Declaration

In addition to the basic features, Simplicity IDE supports many advanced code-editing features. For example, the IDE automatically indexes all code within the project to support symbol lookup. The code does not have to build completely for the indexer to work, though certain features may not be available if, for example, the `main()` routine is not declared.

In this example, the **Open Declaration** (F3 shortcut) feature quickly finds symbol declarations.

1. Open the file of interest by double-clicking it in the Project Explorer view.
2. Right-click the desired symbol to display the context menu.
3. Click **Open Declaration** to quickly navigate to the definition of the symbol.
4. SSv5 automatically opens the file and highlights the line containing the declaration of the symbol.



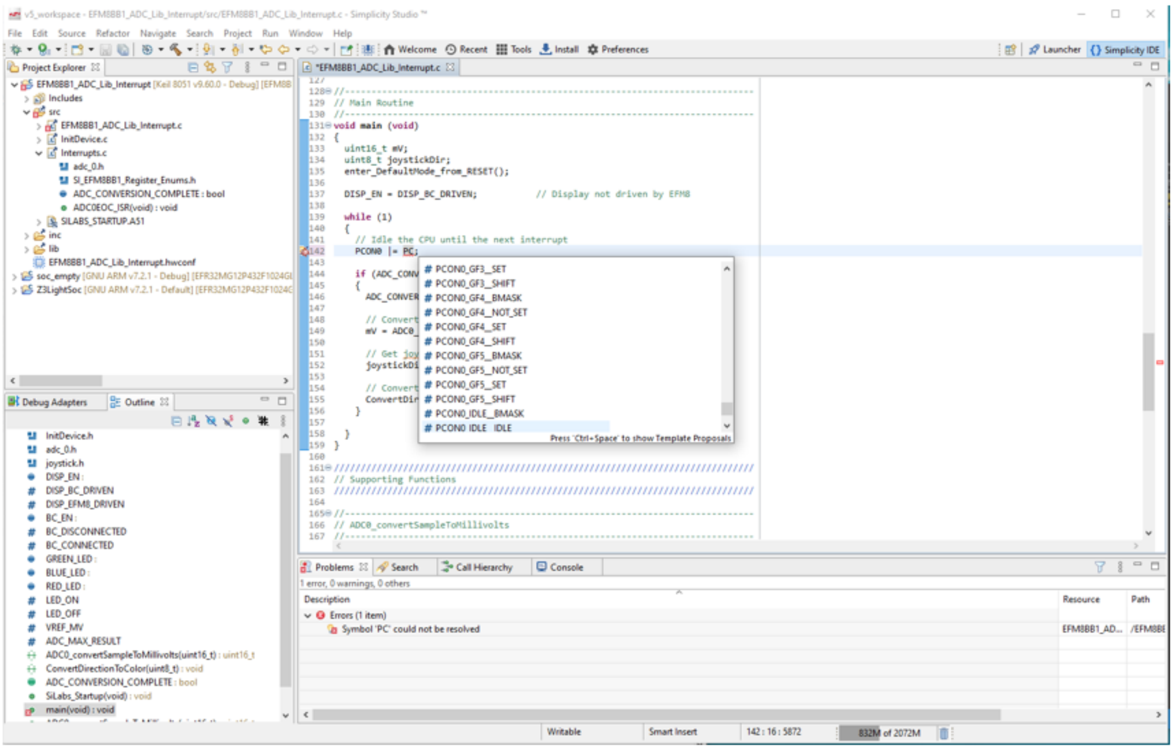
```
//=========================================================
// src/Interrupts.c: generated by Hardware Configurator
//
// This file will be regenerated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!
//=========================================================

// USER INCLUDES
#include <SI_EFM8BB1_Register_Enums.h>
#include "adc_0.h"

bool ADC_CONVERS
//-------------
// ADC0EOC_ISR
//-------------
//
// ADC0EOC ISR
// ADC0CN0::ADIN
//
//-------------
SI_INTERRUPT(ADC
{
  ADC0_clearIntF
  ADC_CONVERSION
}
```

```
  // Enable ADC
  ADC0CN0_ADEN = 1;
}
#endif //EFM8PDL_ADC0_USE_INIT

uint8_t ADC0_getIntFlags()
{
  uint8_t val;
  val = ADC0CN0 & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF);
  return val;
}

void ADC0_clearIntFlags(uint8_t flags)
{

  // Must pass one of the valid flags
  SLAB_ASSERT(flags & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF));

  ADC0CN0 &= ~(flags & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF));
}

void ADC0_enableInt(uint8_t flags, bool enable)
{
  uint8_t en;

  // Must pass one of the valid flags
  SLAB_ASSERT(flags & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF));

  en = (uint8_t) enable;

  if (flags & ADC0_CONVERSION_COMPLETE_IF)
  {
    EIE1 &= ~EIE1_EADC0__BMASK;
    EIE1 |= en << EIE1_EADC0__SHIFT;
  }

  if (flags & ADC0_WINDOW_COMPARE_IF)
  {
    EIE1 &= ~EIE1_EWADC0__BMASK;
```

Content Assist

Simplicity Studio also supports code completion, a feature called **Content Assist**. Content Assist requires that the appropriate header files be included in the file so that the symbols are available. To use Content Assist, type the first few letters of a symbol or include file and press **Ctrl+Space** to display a list of symbols that match. For example, to use [Content Assist] to display a list of symbols starting with the characters PC:

1. Type `PC` and press **Ctrl-Space** to display the Content Assist list.
2. Use the arrow keys or page up and down keys to look through the list of matching symbols.
3. Press **Enter** to replace the typed characters with the selected symbol.



```
//-----------------------------------------------------------------------------
// Main Routine
//-----------------------------------------------------------------------------
void main (void)
{
  uint16_t mV;
  uint8_t joystickDir;
  enter_DefaultMode_from_RESET();

  DISP_EN = DISP_BC_DRIVEN;           // Display not driven by EFM8

  while (1)
  {
    // Idle the CPU until the next interrupt
    PCON0 |= PC;

    if (ADC_CONV
    {
      ADC_CONVER

      // Convert
      mV = ADC0_

      // Get joy
      joystickDi

      // Convert
      ConvertDir
    }

  }
}

///////////////////////////////////////////////////////////////////////////////
// Supporting Functions
///////////////////////////////////////////////////////////////////////////////

//-----------------------------------------------------------------------------
// ADC0_convertSampleToMillivolts
//-----------------------------------------------------------------------------
```

```
# PCON0_GF3__SET
# PCON0_GF3__SHIFT
# PCON0_GF4__BMASK
# PCON0_GF4__NOT_SET
# PCON0_GF4__SET
# PCON0_GF4__SHIFT
# PCON0_GF5__BMASK
# PCON0_GF5__NOT_SET
# PCON0_GF5__SET
# PCON0_GF5__SHIFT
# PCON0_IDLE__BMASK
# PCON0 IDLE  IDLE
Press 'Ctrl+Space' to show Template Proposals
```

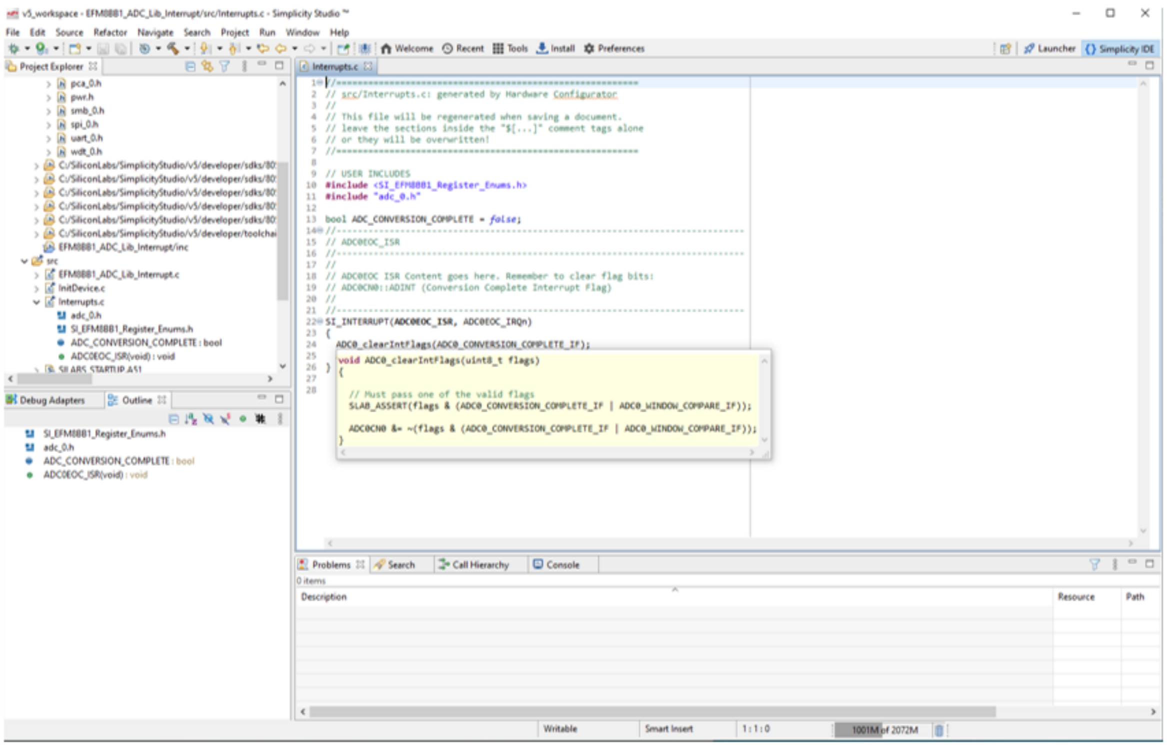
```
1 error, 0 warnings, 0 others
Errors (1 item)
  Symbol 'PC' could not be resolved
```

Link with Editor

In the Project Explorer view, click **Link with Editor** in the toolbar. Then, if you have multiple files open, selecting a file in Project Explorer makes the associated editor tab active.

Symbol Expansion

Hover over a function or macro in the editor to display a hover window with expanded information.



```
//============================================================
// src/Interrupts.c: generated by Hardware Configurator
//
// This file will be regenerated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!
//============================================================

// USER INCLUDES
#include <SI_EFM8BB1_Register_Enums.h>
#include "adc_0.h"

bool ADC_CONVERSION_COMPLETE = false;
//-----------------------------------------------------------------------------
// ADC0EOC_ISR
//-----------------------------------------------------------------------------
//
// ADC0EOC ISR Content goes here. Remember to clear flag bits:
// ADC0CN0::ADINT (Conversion Complete Interrupt Flag)
//
//-----------------------------------------------------------------------------
SI_INTERRUPT(ADC0EOC_ISR, ADC0EOC_IRQn)
{
  ADC0_clearIntFlags(ADC0_CONVERSION_COMPLETE_IF);

}
```

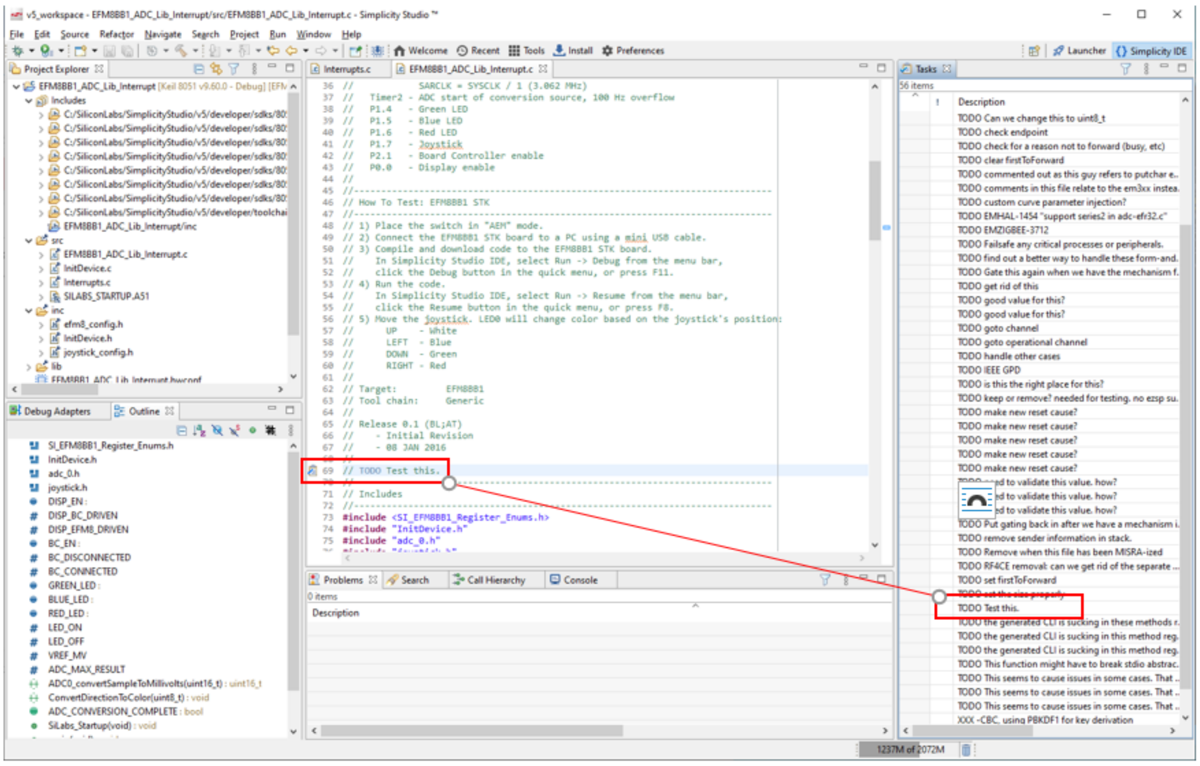
```
void ADC0_clearIntFlags(uint8_t flags)
{

  // Must pass one of the valid flags
  SLAB_ASSERT(flags & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF));

  ADC0CN0 &= ~(flags & (ADC0_CONVERSION_COMPLETE_IF | ADC0_WINDOW_COMPARE_IF));
}
```

Task View

The Tasks view automatically picks up any comments with `TODO` or `FIXME` in the line. For example:

1. Type `TODO` in a comment line with the desired text and save the file
2. Select **Window > Show View > Other** , type `Tasks` , and press **Open**.

The TODO or FIXME lines are highlighted by a clipboard icon to the left of the line in the editor. Double-click on a line in the Tasks view to open the relevant file with that line selected.



```
//              SARCLK = SYSCLK / 1 (3.062 MHz)
//   Timer2 - ADC start of conversion source, 100 Hz overflow
//   P1.4   - Green LED
//   P1.5   - Blue LED
//   P1.6   - Red LED
//   P1.7   - Joystick
//   P2.1   - Board Controller enable
//   P0.0   - Display enable
//
//-----------------------------------------------------------------------------
// How To Test: EFM8BB1 STK
//-----------------------------------------------------------------------------
// 1) Place the switch in "AEM" mode.
// 2) Connect the EFM8BB1 STK board to a PC using a mini USB cable.
// 3) Compile and download code to the EFM8BB1 STK board.
//    In Simplicity Studio IDE, select Run -> Debug from the menu bar,
//    click the Debug button in the quick menu, or press F11.
// 4) Run the code.
//    In Simplicity Studio IDE, select Run -> Resume from the menu bar,
//    click the Resume button in the quick menu, or press F8.
// 5) Move the joystick. LED0 will change color based on the joystick's position:
//      UP    - White
//      LEFT  - Blue
//      DOWN  - Green
//      RIGHT - Red
//
// Target:         EFM8BB1
// Tool chain:     Generic
//
// Release 0.1 (BL;AT)
//    - Initial Revision
//    - 08 JAN 2016
//
// TODO Test this.
//-----------------------------------------------------------------------------
// Includes
//-----------------------------------------------------------------------------
#include <SI_EFM8BB1_Register_Enums.h>
#include "InitDevice.h"
#include "adc_0.h"
```

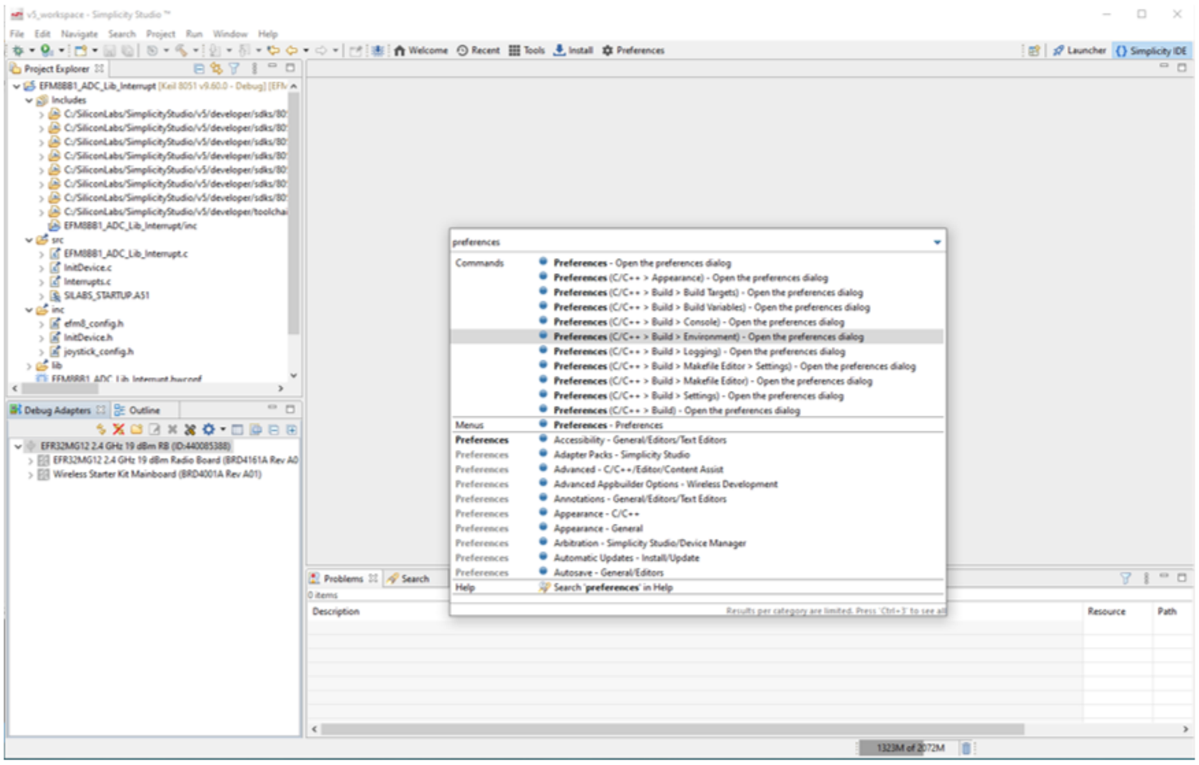
56 items

Description
TODO Can we change this to uint8_t
TODO check endpoint
TODO check for a reason not to forward (busy, etc)
TODO clear firstToForward
TODO commented out as this guy refers to putchar e..
TODO comments in this file relate to the em3xx instea.
TODO custom curve parameter injection?
TODO EMHAL-1454 "support series2 in adc-efr32.c"
TODO EMZIGBEE-3712
TODO Failsafe any critical processes or peripherals.
TODO find out a better way to handle these form-and.
TODO Gate this again when we have the mechanism f.
TODO get rid of this
TODO good value for this?
TODO good value for this?
TODO goto channel
TODO goto operational channel
TODO handle other cases
TODO IEEE GPD
TODO is this the right place for this?
TODO keep or remove? needed for testing. no ezsp su.
TODO make new reset cause?
TODO make new reset cause?
TODO make new reset cause?
TODO make new reset cause?
TODO make new reset cause?
TODO need to validate this value. how?
ed to validate this value. how?
ed to validate this value. how?
TODO Put gating back in after we have a mechanism i.
TODO remove sender information in stack.
TODO Remove when this file has been MISRA-ized
TODO RF4CE removal: can we get rid of the separate ...
TODO set firstToForward
TODO set the size properly
TODO Test this.
TODO the generated CLI is sucking in these methods r.
TODO the generated CLI is sucking in this method reg.
TODO the generated CLI is sucking in this method reg.
TODO This function might have to break stdio abstrac.
TODO This seems to cause issues in some cases. That ...
TODO This seems to cause issues in some cases. That ...
TODO This seems to cause issues in some cases. That ...
XXX -CBC, using PBKDF1 for key derivation

Note: To reset the perspective to its original layout, right-click the perspective button in the toolbar and select **Reset**.

Quick-Access Console

Press **Ctrl+3** to open a quick-access console for locating any menu or view within the IDE. For example, press **Ctrl+3** and type `Preferences` . This lists all of the Preferences menus available within SSv5. Then, select an option to open the menu.



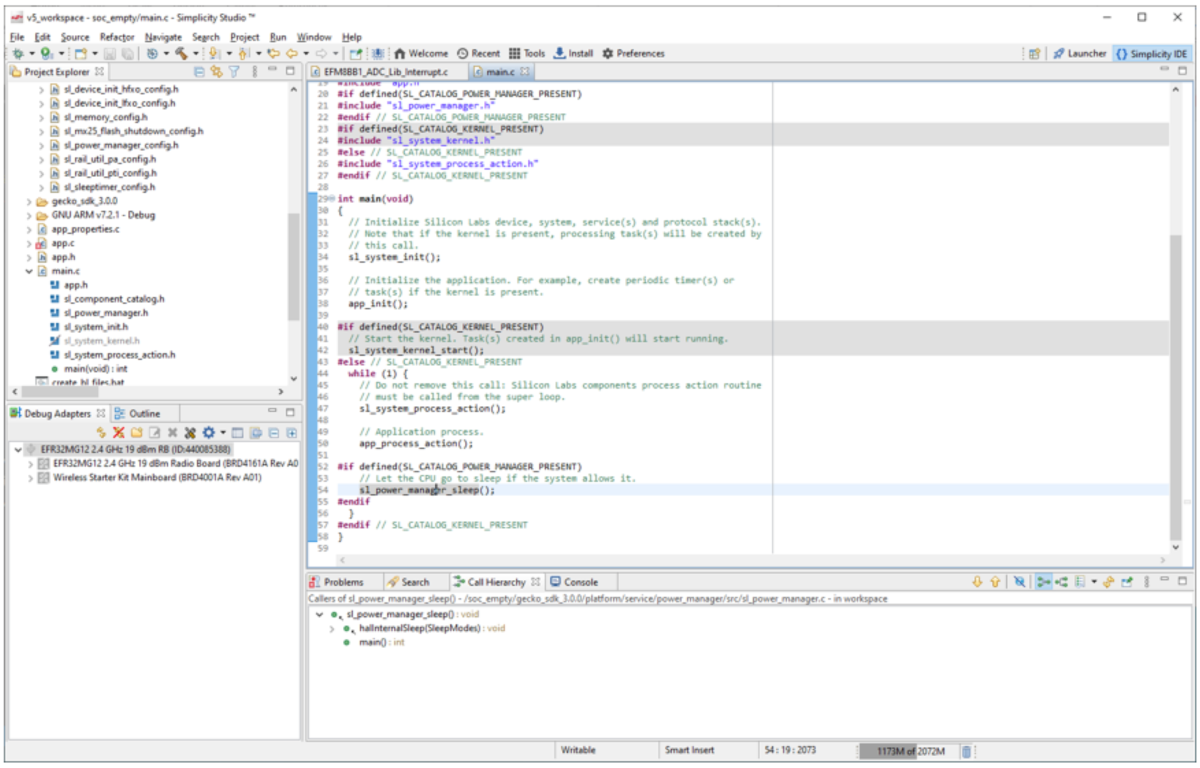
preferences

Commands	**Preferences** - Open the preferences dialog
	Preferences (C/C++ > Appearance) - Open the preferences dialog
	Preferences (C/C++ > Build > Build Targets) - Open the preferences dialog
	Preferences (C/C++ > Build > Build Variables) - Open the preferences dialog
	Preferences (C/C++ > Build > Console) - Open the preferences dialog
	Preferences (C/C++ > Build > Environment) - Open the preferences dialog
	Preferences (C/C++ > Build > Logging) - Open the preferences dialog
	Preferences (C/C++ > Build > Makefile Editor > Settings) - Open the preferences dialog
	Preferences (C/C++ > Build > Makefile Editor) - Open the preferences dialog
	Preferences (C/C++ > Build > Settings) - Open the preferences dialog
	Preferences (C/C++ > Build) - Open the preferences dialog
Menus	**Preferences** - Preferences
Preferences	Accessibility - General/Editors/Text Editors
Preferences	Adapter Packs - Simplicity Studio
Preferences	Advanced - C/C++/Editor/Content Assist
Preferences	Advanced Appbuilder Options - Wireless Development
Preferences	Annotations - General/Editors/Text Editors
Preferences	Appearance - C/C++
Preferences	Appearance - General
Preferences	Arbitration - Simplicity Studio/Device Manager
Preferences	Automatic Updates - Install/Update
Preferences	Autosave - General/Editors
Help	Search '**preferences**' in Help

Results per category are limited. Press 'Ctrl+3' to see all

Call Hierarchy

The IDE includes a call hierarchy that can help find where functions are called. To find the call hierarchy for a function in development mode, right-click the function and select **Open Call Hierarchy**.



```
#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
#include "sl_power_manager.h"
#endif // SL_CATALOG_POWER_MANAGER_PRESENT
#if defined(SL_CATALOG_KERNEL_PRESENT)
#include "sl_system_kernel.h"
#else // SL_CATALOG_KERNEL_PRESENT
#include "sl_system_process_action.h"
#endif // SL_CATALOG_KERNEL_PRESENT

int main(void)
{
  // Initialize Silicon Labs device, system, service(s) and protocol stack(s).
  // Note that if the kernel is present, processing task(s) will be created by
  // this call.
  sl_system_init();

  // Initialize the application. For example, create periodic timer(s) or
  // task(s) if the kernel is present.
  app_init();

#if defined(SL_CATALOG_KERNEL_PRESENT)
  // Start the kernel. Task(s) created in app_init() will start running.
  sl_system_kernel_start();
#else // SL_CATALOG_KERNEL_PRESENT
  while (1) {
    // Do not remove this call: Silicon Labs components process action routine
    // must be called from the super loop.
    sl_system_process_action();

    // Application process.
    app_process_action();

#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
    // Let the CPU go to sleep if the system allows it.
    sl_power_manager_sleep();
#endif
  }
#endif // SL_CATALOG_KERNEL_PRESENT
}
```

Callers of sl_power_manager_sleep() - /soc_empty/gecko_sdk_3.0.0/platform/service/power_manager/src/sl_power_manager.c - in workspace

- sl_power_manager_sleep() : void
  - halInternalSleep(SleepModes) : void
  - main() : int

Overview

Developing For 32-Bit Devices - Overview

These pages describe how to customize projects developed in Simplicity Studio® 5 (SSv5) for 32-bit target devices.

If you have not yet installed SSv5 or are not familiar with its features, Getting Started guides you through installing SSv5 and your Silicon Labs protocol software development kit (SDK). This section assumes you have installed studio, have your development hardware connected, and are generally familiar with the SSv5 Launcher Perspective and with the Simplicity IDE.

Each Silicon Labs SDK comes with example projects. The easiest way to start development is to modify the configuration of an example project. The various paths to creating a project based on an example are described in Start a Project.

The development tools you will use depend on the example chosen as a starting point. The most commonly used tools are described in Developing with Project Configurator. In general, Project Configurator is used with Gecko Bootloader and all other protocol projects created with GSDK version 4.0 and higher, as well as Bluetooth, Bluetooth Mesh, OpenThread, Wi-SUN, and Proprietary applications created with Gecko SDK 3.x. These projects share a common infrastructure, called Silicon Labs Configurator (SLC). As of SSv5 version 5.3, a legacy tool called AppBuilder is only used with Zigbee EmberZNet and Gecko Bootloader projects that were created with GSDKs version 3.2 and lower.

After you have created your project files, compile an application image and flash it to your device as described in Building and Flashing.

When you are ready to test your project, whether on Silicon Labs hardware or your own custom device, SSv5 provides a number of tools to help with testing and debugging. An overview is provided in Testing and Debugging.

Developing with Project Configurator

Developing with Project Configurator

Simplicity Studio® 5 (SSv5) introduced new and improved tools for configuring project code, designed to work on the project's component-based architecture. SSv5 includes project configuration tools that provide an enhanced level of software component discoverability, configurability, and dependency management. These include:

- Project Configurator to install and uninstall components,
- Component Editor to change component parameters,
- Pin Tool to configure peripherals.

Other tools are protocol-specific:

- For Bluetooth and Bluetooth Mesh projects, use the Bluetooth GATT Configurator to customize the Bluetooth GATT database.
- For Bluetooth Mesh projects, use the Bluetooth Mesh Configurator to customize a node's Device Configuration Data (DCD).
- For Proprietary RAIL projects, use the Radio Configurator to define and manage proprietary radio protocols and channel groups
- For Wi-SUN projects, use the Wi-SUN Configurator to configure application settings, security, and radio.
- For Zigbee and Matter projects, use the ZCL Advanced Platform (ZAP) to manage and configure endpoints.

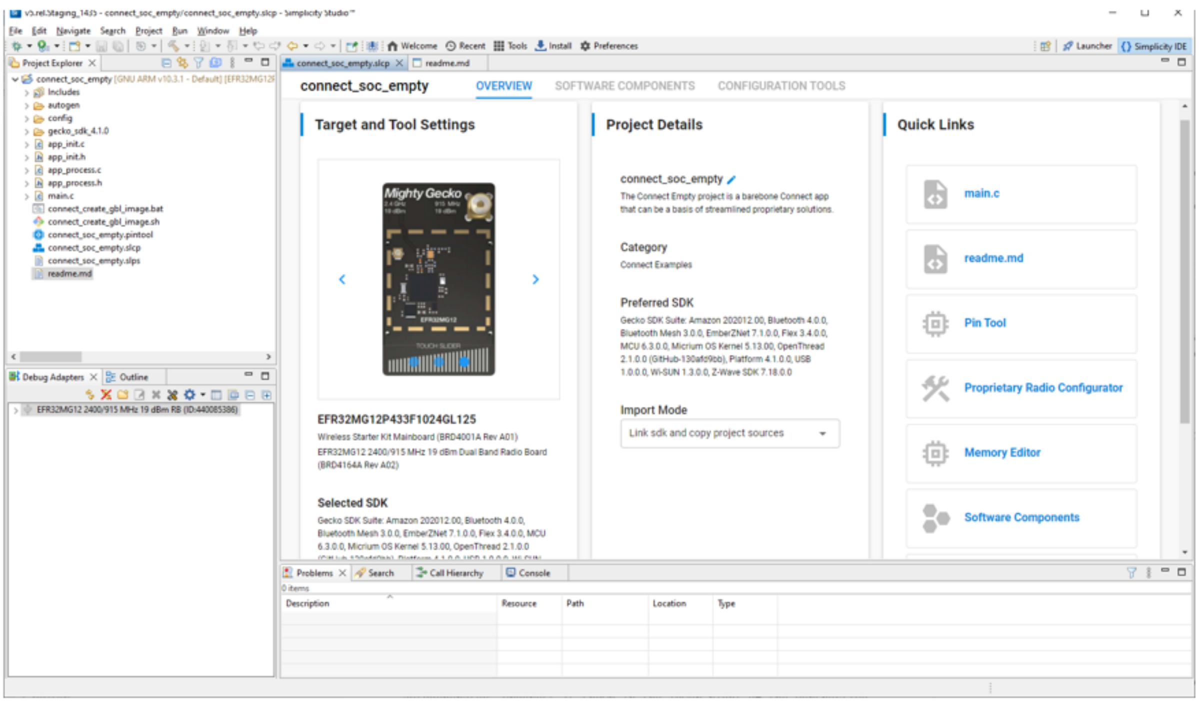
Once you have finishing customizing your project, build and flash it and test and debug it.

Beginning with Simplicity Studio 5.3, you can combine projects into solutions. Solutions can be built, debugged, and flashed together.

Project Configurator

When you create a component-based project (.slcp), the Project Configurator automatically opens an OVERVIEW tab in the Editor area of the Simplicity IDE perspective. A SOFTWARE COMPONENTS tab provides access to a library of software components and their configuration functions. A CONFIGURATION TOOLS tab allows for quick access to project-specific tools.

Open the Project Configurator for an existing project by double-clicking the *<project>.slcp* file in the Project Explorer view.


v5.rel.Staging_1435 - connect_soc_empty/connect_soc_empty.slcp - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
Debug Adapters Outline
connect_soc_empty.slcp readme.md
OVERVIEW SOFTWARE COMPONENTS CONFIGURATION TOOLS
Problems Search Call Hierarchy Console
Mighty Gecko
EFR32MG12
TOUCH SLIDER

connect_soc_empty [GNU ARM v10.3.1 - Default] [EFR32MG12
- Includes
- autogen
- config
- gecko_sdk_4.1.0
- app_init.c
- app_init.h
- app_process.c
- app_process.h
- main.c
- connect_create_gbl_image.bat
- connect_create_gbl_image.sh
- connect_soc_empty.pintool
- connect_soc_empty.slcp
- connect_soc_empty.slps
- readme.md

EFR32MG12 2400/915 MHz 19 dBm RB (ID:440085386)

connect_soc_empty

Target and Tool Settings

EFR32MG12P433F1024GL125

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)

Selected SDK

Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.0.0, Bluetooth Mesh 3.0.0, EmberZNet 7.1.0.0, Flex 3.4.0.0, MCU 6.3.0.0, Micrium OS Kernel 5.13.00, OpenThread 2.1.0.0

Project Details

connect_soc_empty

The Connect Empty project is a barebone Connect app that can be a basis of streamlined proprietary solutions.

Category

Connect Examples

Preferred SDK

Gecko SDK Suite: Amazon 202012.00, Bluetooth 4.0.0, Bluetooth Mesh 3.0.0, EmberZNet 7.1.0.0, Flex 3.4.0.0, MCU 6.3.0.0, Micrium OS Kernel 5.13.00, OpenThread 2.1.0.0 (GitHub-130afd9bb), Platform 4.1.0.0, USB 1.0.0.0, Wi-SUN 1.3.0.0, Z-Wave SDK 7.18.0.0

Import Mode

Link sdk and copy project sources

Quick Links

- main.c
- readme.md
- Pin Tool
- Proprietary Radio Configurator
- Memory Editor
- Software Components

0 items

Description	Resource	Path	Location	Type

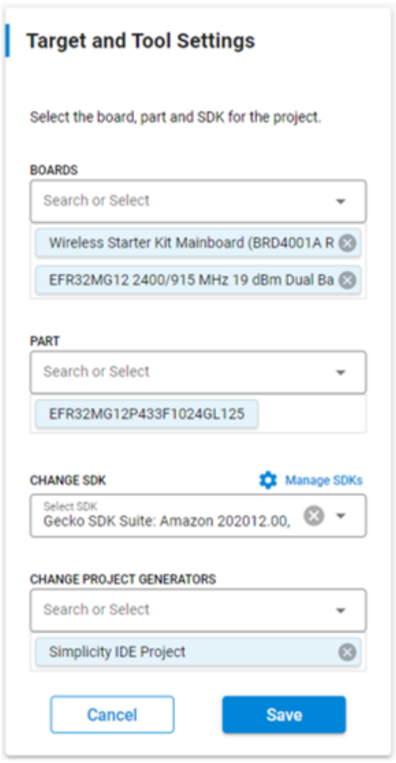
Overview Tab

The OVERVIEW tab has three cards:

- Target and Tool Settings, where you can change your development target, SDK, and project generators.
- Project Details, where you can rename the project, change the Import mode and, in rare cases, force generation.
- Quick Links, where you can change the files SSv5 generates for your project.

Target and Tool Settings

Click **Change Target/SDK/Generators** to change the device for which the project is developed, the SDK version to be used in development, or the files that will be generated for the project. Changes made here do not apply to the project until you save changes.



Target and Tool Settings

Select the board, part and SDK for the project.

BOARDS: Wireless Starter Kit Mainboard (BRD4001A R; EFR32MG12 2400/915 MHz 19 dBm Dual Ba

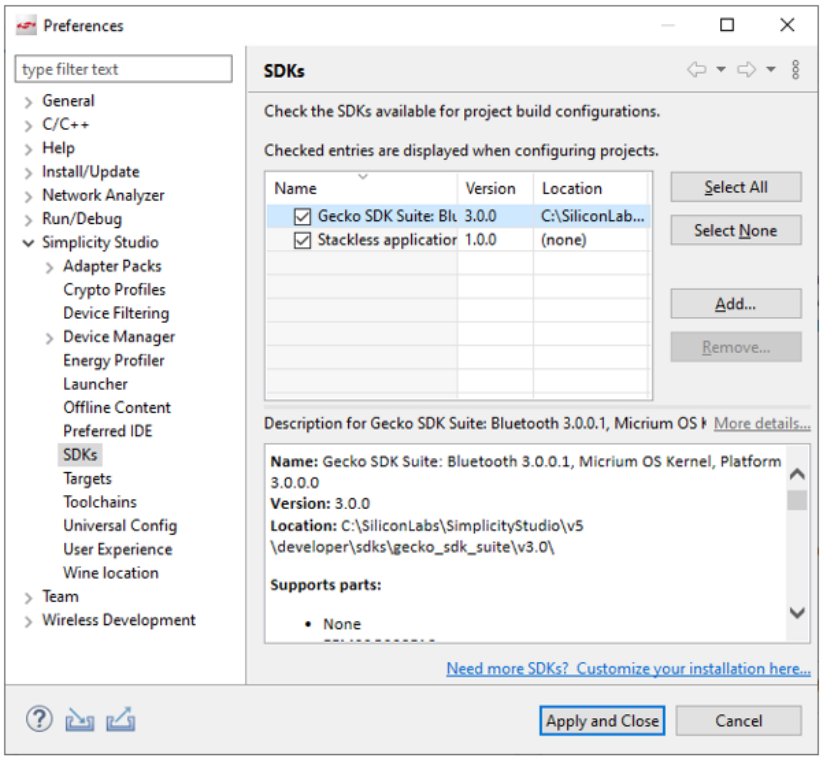
PART: EFR32MG12P433F1024GL125

CHANGE SDK: Gecko SDK Suite: Amazon 202012.00,

CHANGE PROJECT GENERATORS: Simplicity IDE Project

Search for target hardware by part number. If you select a different part, it replaces the part already selected, or you can delete the part and then add a new one.

If you have more than one Gecko SDK Suite (GSDK) version installed, you can select it here. If you do not see a version that you expect to see, click **Manage SDKs** to open the SDK Preferences.



SDKs

Check the SDKs available for project build configurations.

Checked entries are displayed when configuring projects.

Name	Version	Location
☑ Gecko SDK Suite: Blu	3.0.0	C:\SiliconLab...
☑ Stackless applicatior	1.0.0	(none)

Description for Gecko SDK Suite: Bluetooth 3.0.0.1, Micrium OS K

Name: Gecko SDK Suite: Bluetooth 3.0.0.1, Micrium OS Kernel, Platform 3.0.0.0
Version: 3.0.0
Location: C:\SiliconLabs\SimplicityStudio\v5 \developer\sdks\gecko_sdk_suite\v3.0\

Supports parts:

- None

Here you can search for SDKs that might be installed outside of SSv5. Note that, if you have more than one SDK protocol installed in the same GSDK version, for example Bluetooth and Zigbee, you cannot change between them. You can only choose among GSDK versions.

The options for the files that are generated by Project Configurator are:

- GCC Makefile
- Visual Studio Code Compatible Project (Beta)
- IAR Embedded Workbench Project
- Simplicity IDE project

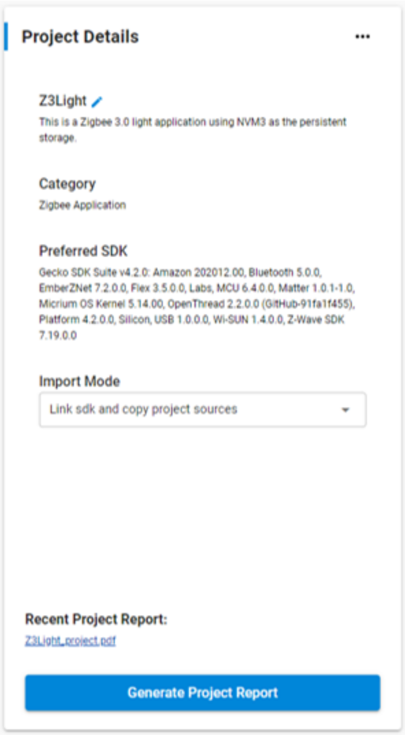
The options that are available for selection and the default vary depending on the SDK and toolchain you selected when you created the project.

Note that changing the project generator does not change the project toolchain. The compiler / toolchain used by Simplicity IDE is configurable in **Project > Build Configurations**. The default IDE is configurable in **Preferences > Simplicity Studio > Preferred IDE**.

Project Details

Project details include a description of the example on which the project is based, the category of the example, the versions of SDK included in the currently selected GSDK version, and the import mode. Here you can:

- Edit the project name
- Change the import mode
- Generate a project report
- (If necessary) Force project generation



Project Details

Z3Light
This is a Zigbee 3.0 light application using NVM3 as the persistent storage.

Category
Zigbee Application

Preferred SDK
Gecko SDK Suite v4.2.0: Amazon 202012.00, Bluetooth 5.0.0, EmberZNet 7.2.0.0, Flex 3.5.0.0, Labs, MCU 6.4.0.0, Matter 1.0.1-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.0.0 (GitHub-91fa1f455), Platform 4.2.0.0, Silicon, USB 1.0.0.0, Wi-SUN 1.4.0.0, Z-Wave SDK 7.19.0.0

Import Mode
Link sdk and copy project sources

Recent Project Report:
Z3Light_project.pdf

Click the pen icon to edit the project name.

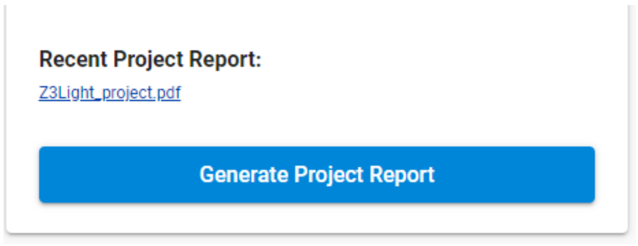
The import mode determines what resources are copied into your project and what are linked. If you change the import mode, the change is autosaved and your project files are regenerated.

- **Link to sources** means all project and SDK resources are linked. Only generated files are saved with the project.

- **Link SDK and copy project sources** (default) means that all the example project sources are copied, but SDK libraries are linked. If you update the SDK and regenerate the project files, the project will change.
- **Copy contents** means that all resources are copied with the project. You can update SDK or load an updated example project and your project will not change.

Generate Project Report creates a PDF, available through a link immediately above the control, that provides details on:

- Target hardware device
- Target hardware device pin mapping
- Target SDK
- Software extensions used in the build
- Installed application, utility, and platform software components

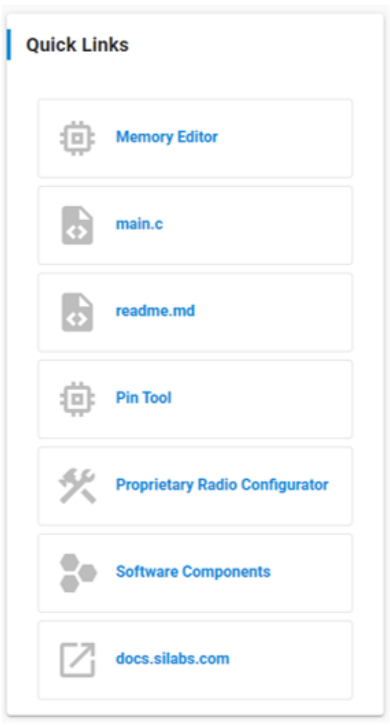


Force Generation, available through the **...** menu, is used only in rare cases when autogeneration is not triggered, usually because of some change made outside of SSv5. It will run all available generators (slcp, radio config, gatt, and so on).

Note for Simplicity Studio 4/AppBuilder users: Because many project configurator files are autogenerated, Project Configurator does not include a **Generate** control like the one in AppBuilder. **Force Generation** is not a replacement for that control.

Quick Links

Quick links are shortcuts to tools and other information of interest while developing a project of this type. The links available vary based on the protocol and device. For example, a Connect project includes a Radio Configurator quick link.

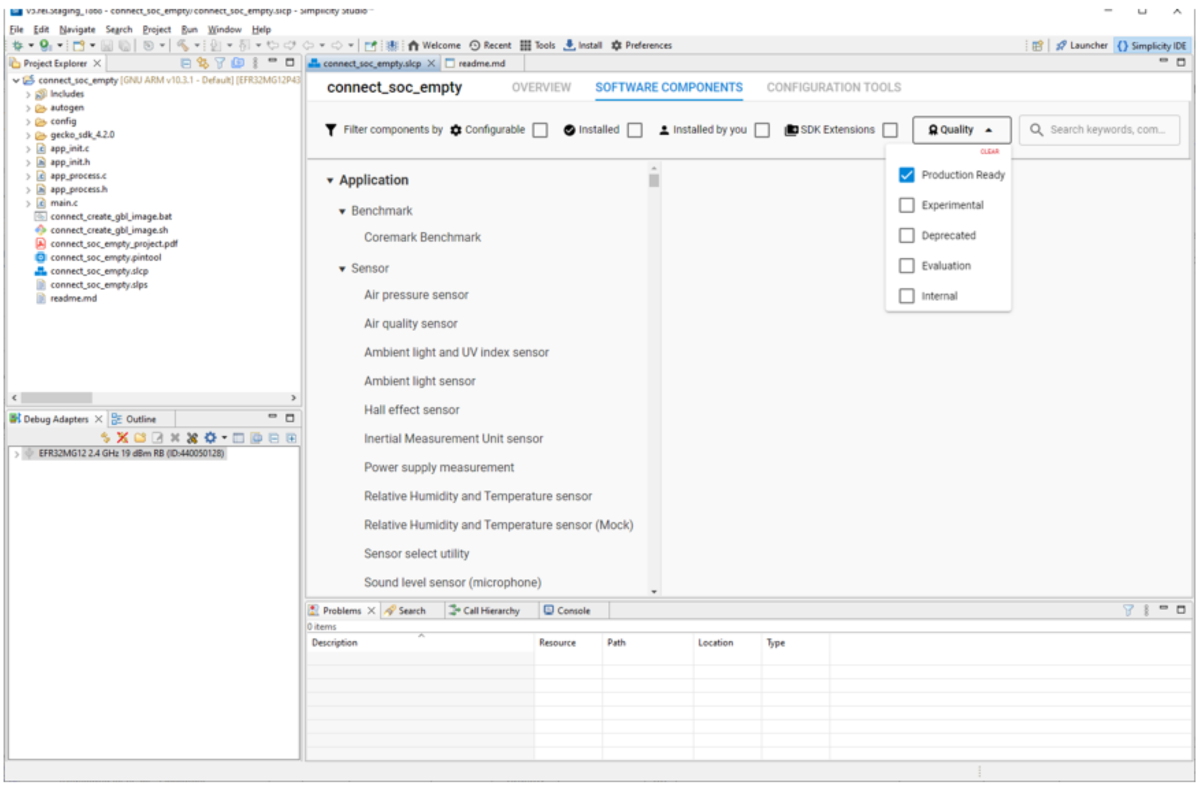


Quick Links

- Memory Editor
- main.c
- readme.md
- Pin Tool
- Proprietary Radio Configurator
- Software Components
- docs.silabs.com

Software Components Tab

Projects are configured by installing and uninstalling components, and configuring installed components. The SOFTWARE COMPONENTS tab displays categories of components on the left, and details about the selected component on the right.


File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
Debug Adapters Outline
Problems Search Call Hierarchy Console
Description Resource Path Location Type

connect_soc_empty [GNU ARM v10.3.1 - Default] [EFR32MG12P43
- Includes
- autogen
- config
- gecko_sdk_4.2.0
- app_init.c
- app_init.h
- app_process.c
- app_process.h
- main.c
- connect_create_gbl_image.bat
- connect_create_gbl_image.sh
- connect_soc_empty_project.pdf
- connect_soc_empty.pintool
- connect_soc_empty.slcp
- connect_soc_empty.slps
- readme.md

EFR32MG12 2.4 GHz 19 dBm RB (ID:440050128)

connect_soc_empty.slcp | readme.md

connect_soc_empty — OVERVIEW | SOFTWARE COMPONENTS | CONFIGURATION TOOLS

Filter components by: Configurable ☐ Installed ☐ Installed by you ☐ SDK Extensions ☐ Quality

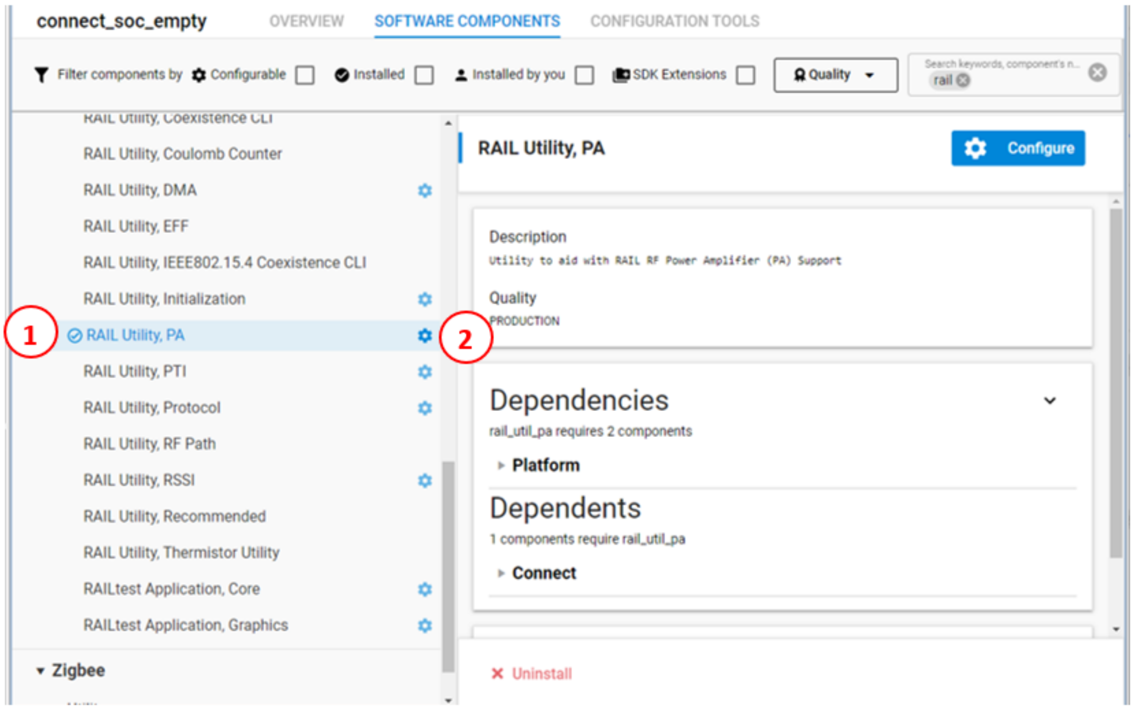
Quality: CLEAR
- ☑ Production Ready
- ☐ Experimental
- ☐ Deprecated
- ☐ Evaluation
- ☐ Internal

Search keywords, com...

- Application
  - Benchmark
    - Coremark Benchmark
  - Sensor
    - Air pressure sensor
    - Air quality sensor
    - Ambient light and UV index sensor
    - Ambient light sensor
    - Hall effect sensor
    - Inertial Measurement Unit sensor
    - Power supply measurement
    - Relative Humidity and Temperature sensor
    - Relative Humidity and Temperature sensor (Mock)
    - Sensor select utility
    - Sound level sensor (microphone)

0 items

A number of filters as well as a keyword search are available to help you explore the various component categories. Note that components for all installed SDKs are presented. Expand a component category\subcategory to see individual components. Select a component to see its details. Components that were included in the original project or installed by the user are checked (1), and can be uninstalled. Configurable components are indicated by a gear symbol (2). Configurable components must be installed before their configuration can be changed.


connect_soc_empty
OVERVIEW
SOFTWARE COMPONENTS
CONFIGURATION TOOLS
Filter components by
Configurable
Installed
Installed by you
SDK Extensions
Quality
Search keywords, component's n...
rail
RAIL Utility, Coexistence CLI
RAIL Utility, Coulomb Counter
RAIL Utility, DMA
RAIL Utility, EFF
RAIL Utility, IEEE802.15.4 Coexistence CLI
RAIL Utility, Initialization
1
RAIL Utility, PA
2
RAIL Utility, PTI
RAIL Utility, Protocol
RAIL Utility, RF Path
RAIL Utility, RSSI
RAIL Utility, Recommended
RAIL Utility, Thermistor Utility
RAILtest Application, Core
RAILtest Application, Graphics
Zigbee
RAIL Utility, PA
Configure
Description
Utility to aid with RAIL RF Power Amplifier (PA) Support
Quality
PRODUCTION
Dependencies
rail_util_pa requires 2 components
Platform
Dependents
1 components require rail_util_pa
Connect
Uninstall

For any component you can see see the components on which this component depends, and components dependent on it. You can also link to additional documentation, if available.

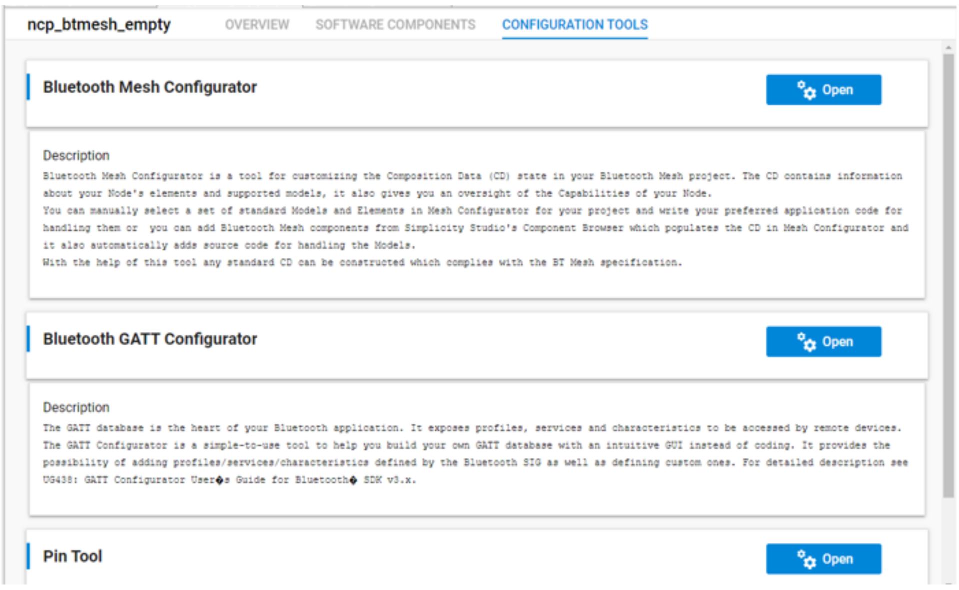
When you install a component, the installation process will:

1. Copy the corresponding SDK files from the SDK folder into the project folder.
2. Copy all the dependencies of the given component into the project folder.
3. Add new include directories to the project settings.
4. Copy the configurations files into the /config folder.
5. Modify the corresponding auto-generated files (in the *autogen* folder) to integrate the component into the application.

Additionally, "init" type software components will implement the initialization code for a given component, utilizing their corresponding configuration file as input. Some software components will fully integrate into the application to perform a specific task without the need of any additional code, while other components provide an API to be used in the application.

Configuration Tools Tab

This tab provides an easy way to open a tool when the tool's tab is not open. The tab shows configuration tools relevant to the project type. A Bluetooth Mesh project shows a number of tools, while an OpenThread project might only show the Pin Tool. Click **Open** on the tool's card to open it in a separate tab.



Bluetooth Mesh Configurator

Description

Bluetooth Mesh Configurator is a tool for customizing the Composition Data (CD) state in your Bluetooth Mesh project. The CD contains information about your Node's elements and supported models, it also gives you an oversight of the Capabilities of your Node.
You can manually select a set of standard Models and Elements in Mesh Configurator for your project and write your preferred application code for handling them or you can add Bluetooth Mesh components from Simplicity Studio's Component Browser which populates the CD in Mesh Configurator and it also automatically adds source code for handling the Models.
With the help of this tool any standard CD can be constructed which complies with the BT Mesh specification.

Bluetooth GATT Configurator

Description

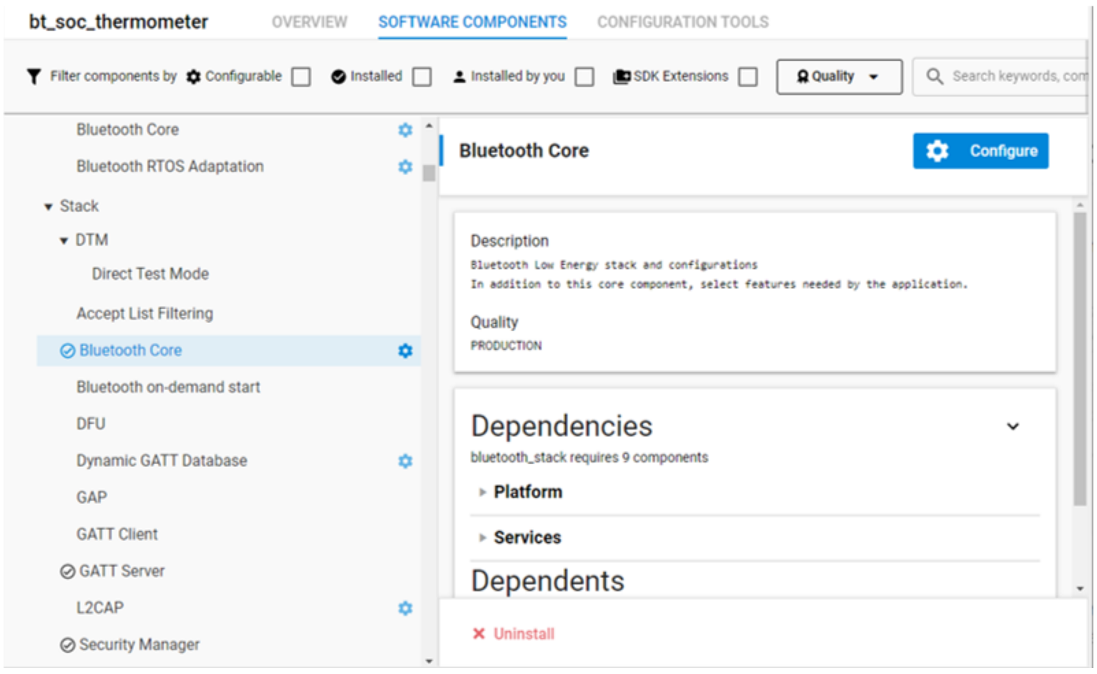
The GATT database is the heart of your Bluetooth application. It exposes profiles, services and characteristics to be accessed by remote devices. The GATT Configurator is a simple-to-use tool to help you build your own GATT database with an intuitive GUI instead of coding. It provides the possibility of adding profiles/services/characteristics defined by the Bluetooth SIG as well as defining custom ones. For detailed description see UG438: GATT Configurator User�s Guide for Bluetooth� SDK v3.x.

Pin Tool

Component Editor

Component Editor

Click the gear symbol next to an installed configurable component name, or **Configure** on the component description to open the Component Editor.

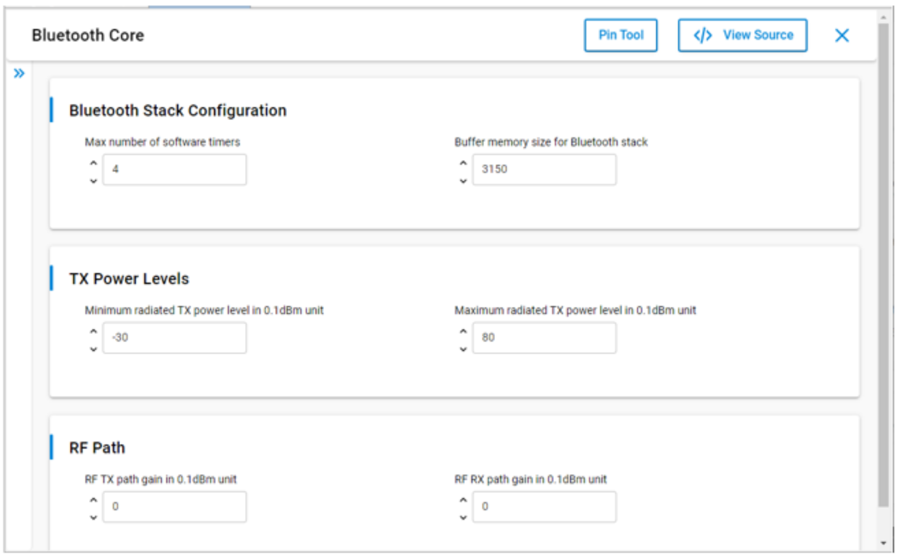


Description
Bluetooth Low Energy stack and configurations
In addition to this core component, select features needed by the application.

Quality
PRODUCTION

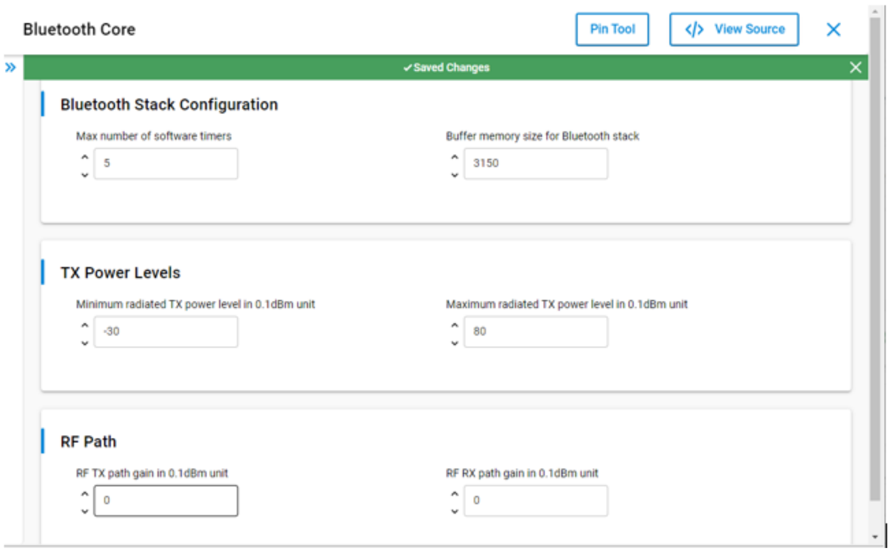
bluetooth_stack requires 9 components

The Component Editor opens in a new tab. The changes you can make depend on the component.



Max number of software timers: 4
Buffer memory size for Bluetooth stack: 3150
Minimum radiated TX power level in 0.1dBm unit: -30
Maximum radiated TX power level in 0.1dBm unit: 80
RF TX path gain in 0.1dBm unit: 0
RF RX path gain in 0.1dBm unit: 0

Changes made here are autosaved in native source format in the associated source files.



Bluetooth Stack Configuration

Max number of software timers	5
Buffer memory size for Bluetooth stack	3150

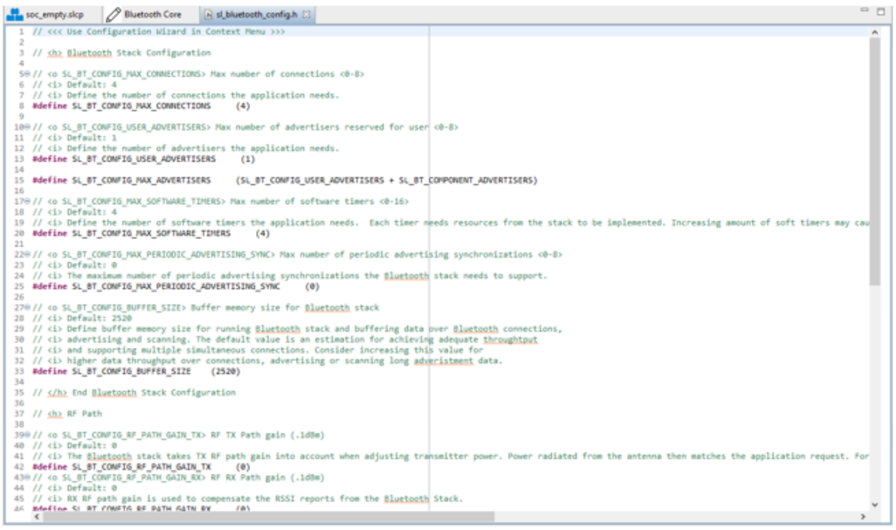
TX Power Levels

Minimum radiated TX power level in 0.1dBm unit	-30
Maximum radiated TX power level in 0.1dBm unit	80

RF Path

RF TX path gain in 0.1dBm unit	0
RF RX path gain in 0.1dBm unit	0

Click **View Source** to open an editor on the configuration file. If a component has multiple configuration files, the control is **View Source Files**

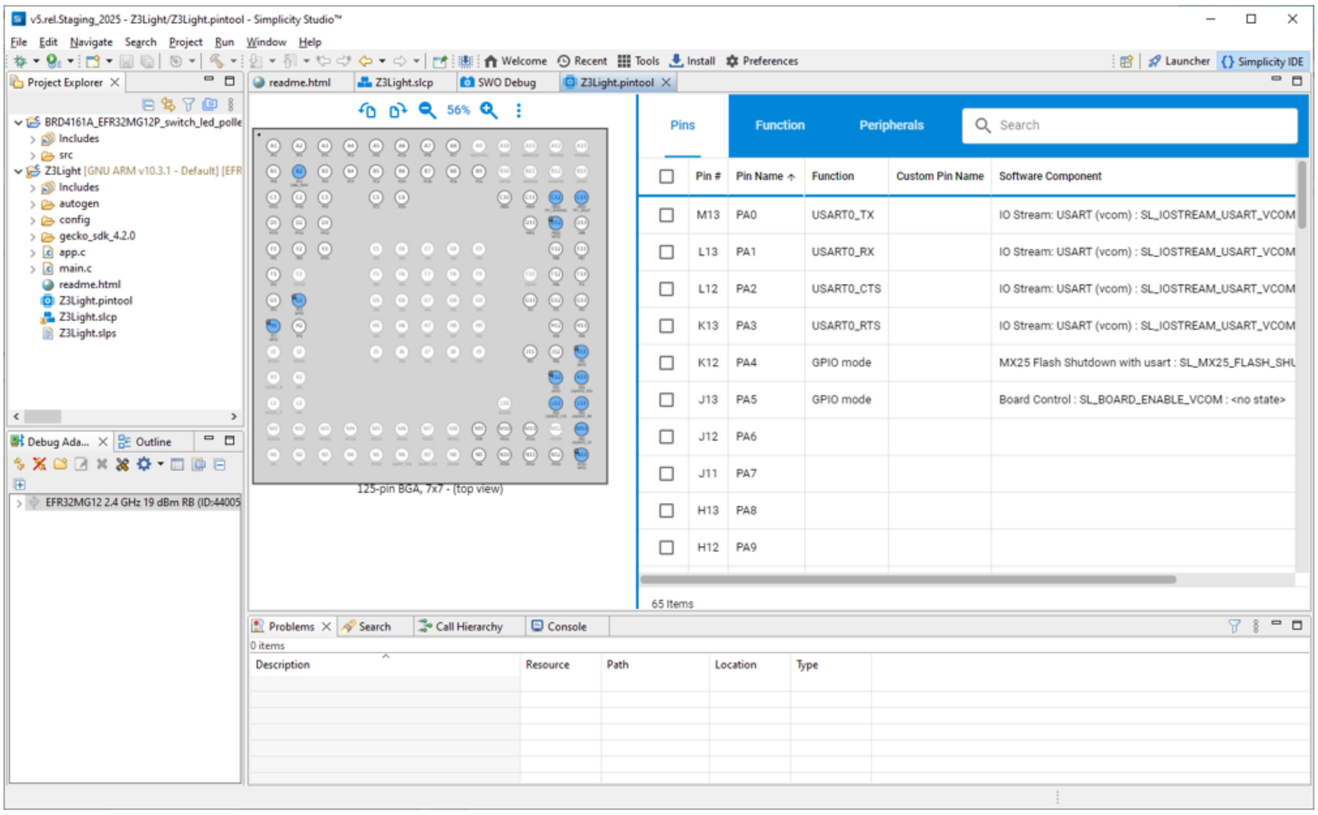


```
// <<< Use Configuration Wizard in Context Menu >>>

// <h> Bluetooth Stack Configuration

// <o SL_BT_CONFIG_MAX_CONNECTIONS> Max number of connections <0-8>
// <i> Default: 4
// <i> Define the number of connections the application needs.
#define SL_BT_CONFIG_MAX_CONNECTIONS     (4)

// <o SL_BT_CONFIG_USER_ADVERTISERS> Max number of advertisers reserved for user <0-8>
// <i> Default: 1
// <i> Define the number of advertisers the application needs.
#define SL_BT_CONFIG_USER_ADVERTISERS     (1)

#define SL_BT_CONFIG_MAX_ADVERTISERS     (SL_BT_CONFIG_USER_ADVERTISERS + SL_BT_COMPONENT_ADVERTISERS)

// <o SL_BT_CONFIG_MAX_SOFTWARE_TIMERS> Max number of software timers <0-16>
// <i> Default: 4
// <i> Define the number of software timers the application needs.  Each timer needs resources from the stack to be implemented. Increasing amount of soft timers may cau
#define SL_BT_CONFIG_MAX_SOFTWARE_TIMERS     (4)

// <o SL_BT_CONFIG_MAX_PERIODIC_ADVERTISING_SYNC> Max number of periodic advertising synchronizations <0-8>
// <i> Default: 0
// <i> The maximum number of periodic advertising synchronizations the Bluetooth stack needs to support.
#define SL_BT_CONFIG_MAX_PERIODIC_ADVERTISING_SYNC     (0)

// <o SL_BT_CONFIG_BUFFER_SIZE> Buffer memory size for Bluetooth stack
// <i> Default: 2520
// <i> Define buffer memory size for running Bluetooth stack and buffering data over Bluetooth connections,
// <i> advertising and scanning. The default value is an estimation for achieving adequate throughtput
// <i> and supporting multiple simultaneous connections. Consider increasing this value for
// <i> higher data throughput over connections, advertising or scanning long adveristment data.
#define SL_BT_CONFIG_BUFFER_SIZE    (2520)

// </h> End Bluetooth Stack Configuration

// <h> RF Path

// <o SL_BT_CONFIG_RF_PATH_GAIN_TX> RF TX Path gain (.1dBm)
// <i> Default: 0
// <i> The Bluetooth stack takes TX RF path gain into account when adjusting transmitter power. Power radiated from the antenna then matches the application request. For
#define SL_BT_CONFIG_RF_PATH_GAIN_TX     (0)
// <o SL_BT_CONFIG_RF_PATH_GAIN_RX> RF RX Path gain (.1dBm)
// <i> Default: 0
// <i> RX RF path gain is used to compensate the RSSI reports from the Bluetooth Stack.
#define SL_BT_CONFIG_RF_PATH_GAIN_RX     (0)
```

Pin Tool

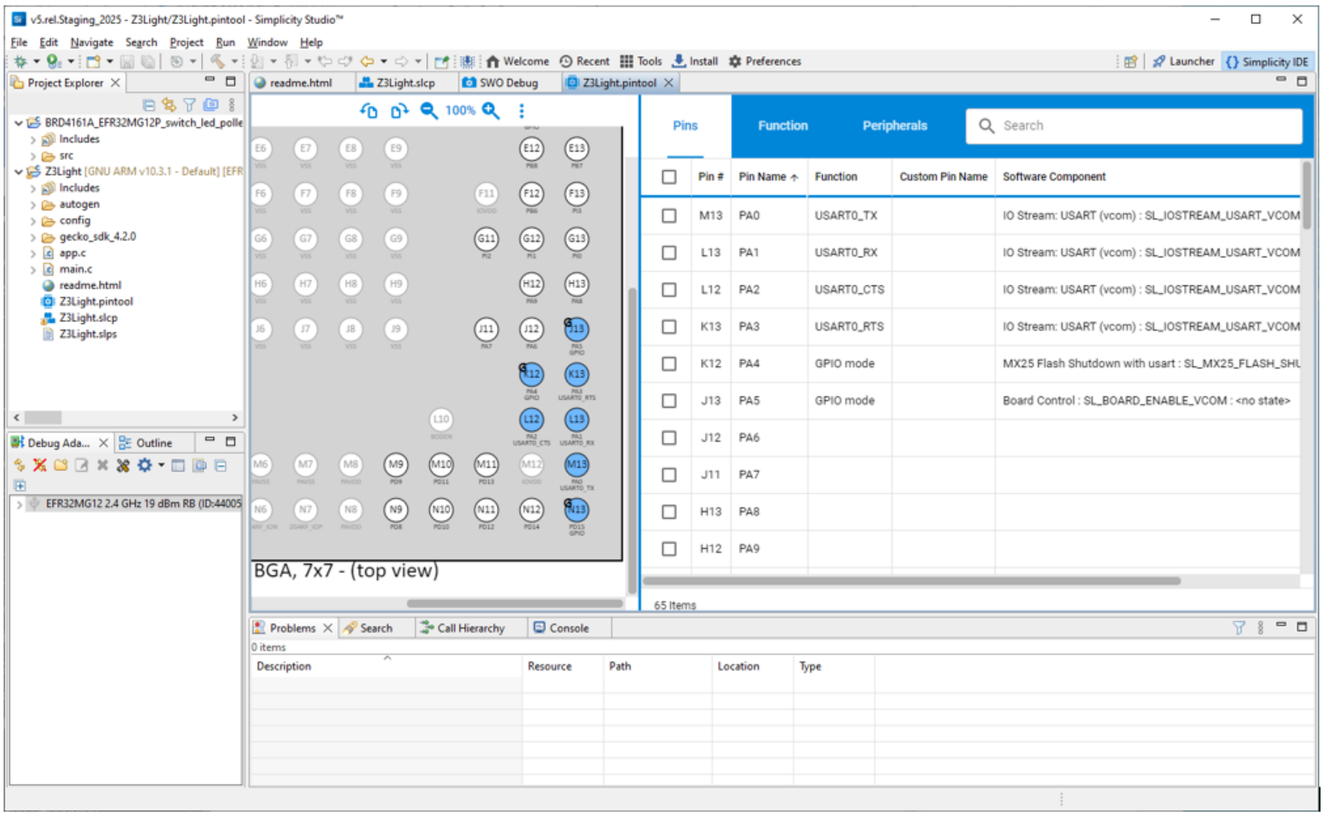
Pin Tool

Simplicity Studio® 5 (SSv5) offers a Pin tool that allows you to easily configure new peripherals or change the properties of existing ones. In the Project Configurator, open the CONFIGURATION TOOLS tab and open the Pin Tool. Alternatively, double-click the file <project>.pintool in the Project Explorer view. The graphical view differs based on the chip.


v5.rel.Staging_2025 - Z3Light/Z3Light.pintool - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences Launcher Simplicity IDE
Project Explorer
BRD4161A_EFR32MG12P_switch_led_polle
Includes
src
Z3Light [GNU ARM v10.3.1 - Default] [EFR
Includes
autogen
config
gecko_sdk_4.2.0
app.c
main.c
readme.html
Z3Light.pintool
Z3Light.slcp
Z3Light.slps
Debug Ada... Outline
EFR32MG12 2.4 GHz 19 dBm RB (ID:44005
readme.html Z3Light.slcp SWO Debug Z3Light.pintool
56%
125-pin BGA, 7x7 - (top view)
Pins Function Peripherals Search
65 Items
Problems Search Call Hierarchy Console
0 items
Description Resource Path Location Type

	Pin #	Pin Name	Function	Custom Pin Name	Software Component
☐	M13	PA0	USART0_TX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	L13	PA1	USART0_RX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	L12	PA2	USART0_CTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	K13	PA3	USART0_RTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	K12	PA4	GPIO mode		MX25 Flash Shutdown with usart : SL_MX25_FLASH_SHU
☐	J13	PA5	GPIO mode		Board Control : SL_BOARD_ENABLE_VCOM : <no state>
☐	J12	PA6			
☐	J11	PA7			
☐	H13	PA8			
☐	H12	PA9			

For more complex layouts, you can use the zoom controls above the graphic to see detail and the rotate controls to rotate the image.



	Pin #	Pin Name	Function	Custom Pin Name	Software Component
☐	M13	PA0	USART0_TX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	L13	PA1	USART0_RX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	L12	PA2	USART0_CTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	K13	PA3	USART0_RTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
☐	K12	PA4	GPIO mode		MX25 Flash Shutdown with usart : SL_MX25_FLASH_SHU
☐	J13	PA5	GPIO mode		Board Control : SL_BOARD_ENABLE_VCOM : <no state>
☐	J12	PA6			
☐	J11	PA7			
☐	H13	PA8			
☐	H12	PA9			

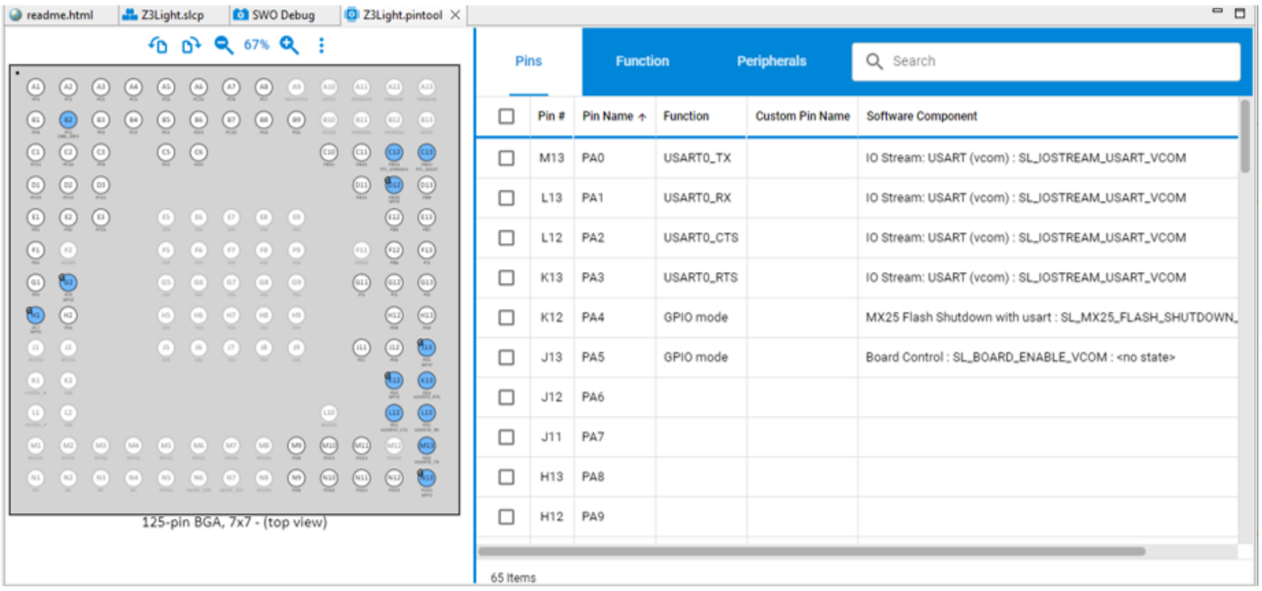
65 Items

Right-click anywhere on the image or click the Additional Functions icon in the upper right of the image to open a menu where you can:

- Save Diagram as Picture: Saves the graphic as a .png file
- Module Configuration Report: Produces an .htm file showing modules with signals assigned to pins. If a module has a signal assigned to a pin, the module and signal/pin assignment appear in the report.
- Pin Configuration Report: Produces an .htm file showing all the user modifiable pins of the chip and the current signal assignment, if any.

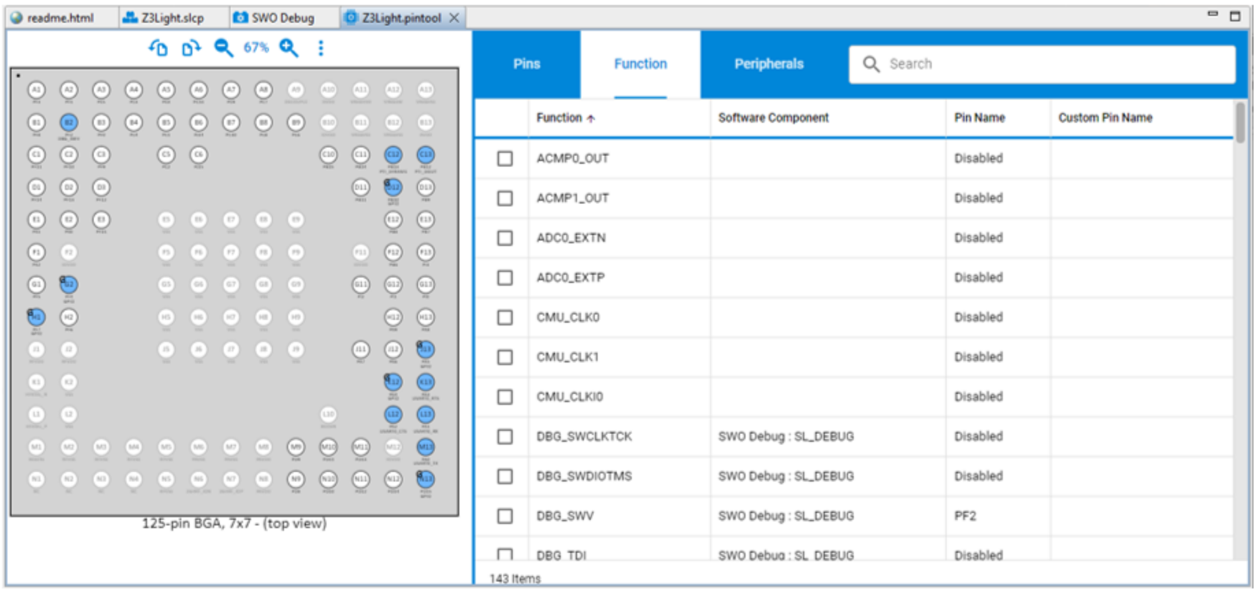
Click a pin on the graphic to select it. The corresponding row is checked in the right pane's tabbed interface.

The pin, function, and peripheral tabs in the configuration pane provide different modes of access. A search function also provided



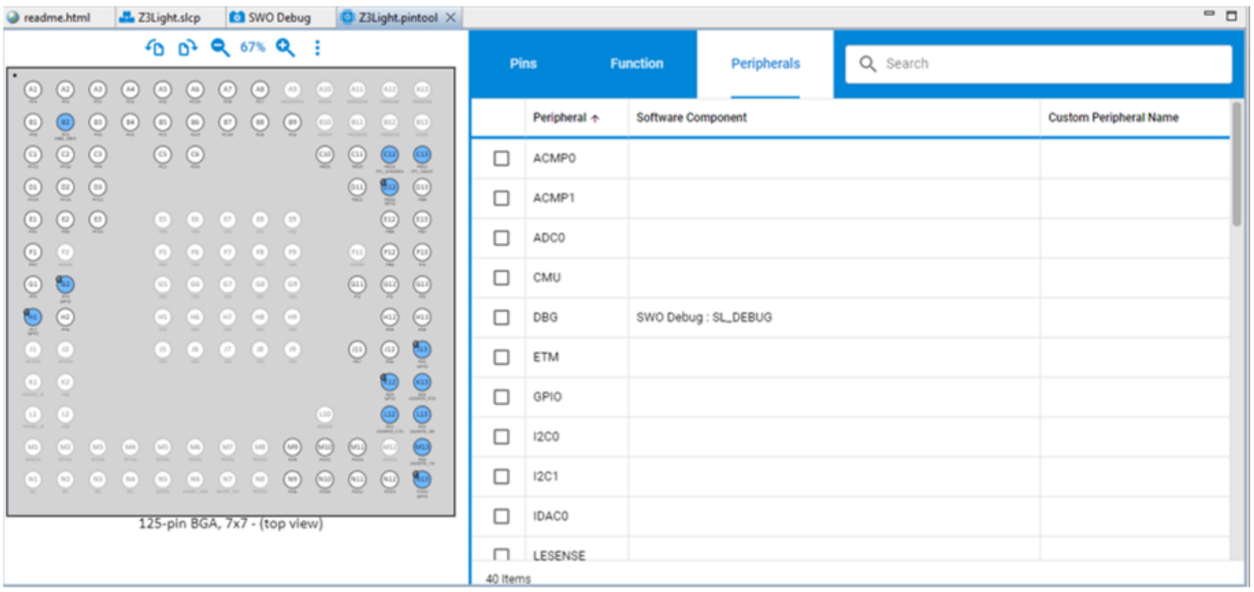
Pin #	Pin Name	Function	Custom Pin Name	Software Component
M13	PA0	USART0_TX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
L13	PA1	USART0_RX		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
L12	PA2	USART0_CTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
K13	PA3	USART0_RTS		IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM
K12	PA4	GPIO mode		MX25 Flash Shutdown with usart : SL_MX25_FLASH_SHUTDOWN_
J13	PA5	GPIO mode		Board Control : SL_BOARD_ENABLE_VCOM : <no state>
J12	PA6			
J11	PA7			
H13	PA8			
H12	PA9			

65 Items



Function	Software Component	Pin Name	Custom Pin Name
ACMP0_OUT		Disabled	
ACMP1_OUT		Disabled	
ADC0_EXTN		Disabled	
ADC0_EXTP		Disabled	
CMU_CLK0		Disabled	
CMU_CLK1		Disabled	
CMU_CLKI0		Disabled	
DBG_SWCLKTCK	SWO Debug : SL_DEBUG	Disabled	
DBG_SWDIOTMS	SWO Debug : SL_DEBUG	Disabled	
DBG_SWV	SWO Debug : SL_DEBUG	PF2	
DBG_TDI	SWO Debug : SL_DEBUG	Disabled	

143 Items



	Peripheral	Software Component	Custom Peripheral Name
☐	ACMP0		
☐	ACMP1		
☐	ADC0		
☐	CMU		
☐	DBG	SWO Debug : SL_DEBUG	
☐	ETM		
☐	GPIO		
☐	I2C0		
☐	I2C1		
☐	IDAC0		
☐	LESENSE		

40 Items

Modifying Pin Configurations

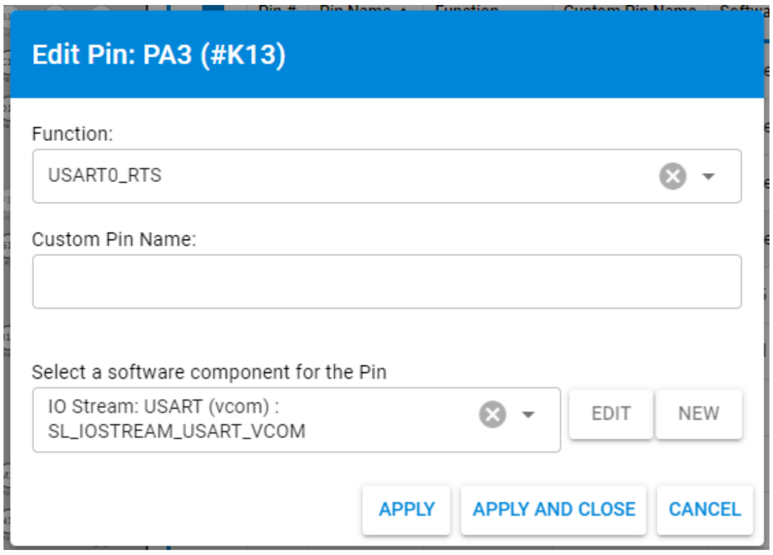
Use the Pin Tool to modify the pin configuration of the device. Software components control behavior in the project but must be associated with a peripheral, and generally need pin or function assignments. These pin or function assignments are most easily edited in the Component Editor for that component. The Pin Tool allows you to assign functions to pins. On all three dialogs, click **EDIT** next to a software component to go directly to the Component Editor for that component. Click **NEW** to go to the Project Configurator's SOFTWARE COMPONENTS tab, where you can install a component in the project so that it can be selected in the dialog.

When you are finished with the configuration, click **APPLY** or **APPLY AND CLOSE** to save changes. Configuration code changes are automatically generated.

The Pins Tab

Click on any pin's row to open the Edit Pin dialog.

- The function assigned to the pin: To apply a function or change a function, drop down the function list and select.
- The Custom Pin Name (optional): Add or edit the custom pin name.
- The Software Component: The component associated with the function is shown.



Edit Pin: PA3 (#K13)

Function:
USART0_RTS

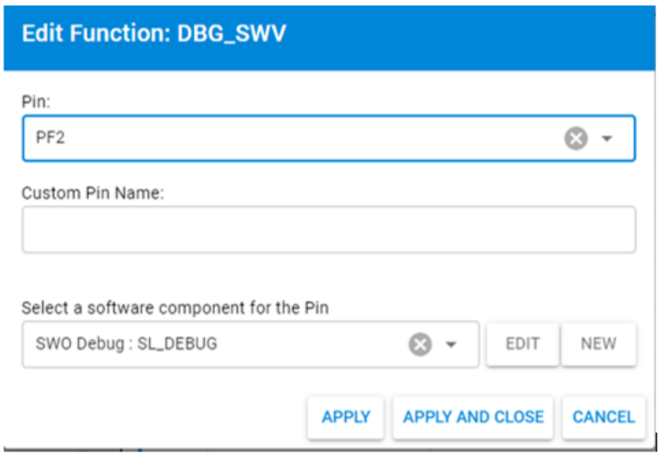
Custom Pin Name:

Select a software component for the Pin
IO Stream: USART (vcom) : SL_IOSTREAM_USART_VCOM

The Functions Tab

Click on any function's row to open the Edit Function dialog.

- The pin to which the function is assigned: Select a pin, or disable.
- The Custom Pin Name (optional): Add or edit the custom pin name.
- The Software Component: The component associated with the function is shown. A pin must be selected.



Edit Function: DBG_SWV

Pin: PF2

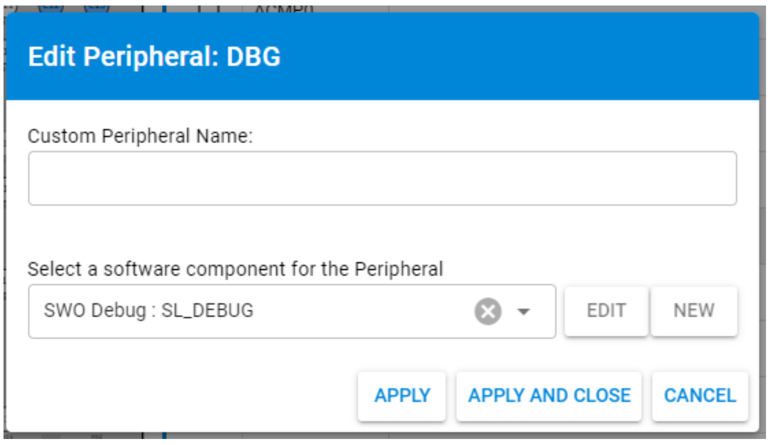
Custom Pin Name:

Select a software component for the Pin: SWO Debug : SL_DEBUG

The Peripherals Tab

Click on any peripheral's row to open the Edit Peripherals dialog.

- The Custom Peripheral Name: Add or edit the custom peripheral name.
- The Software Component: The component associated with the peripheral.



Edit Peripheral: DBG

Custom Peripheral Name:

Select a software component for the Peripheral: SWO Debug : SL_DEBUG

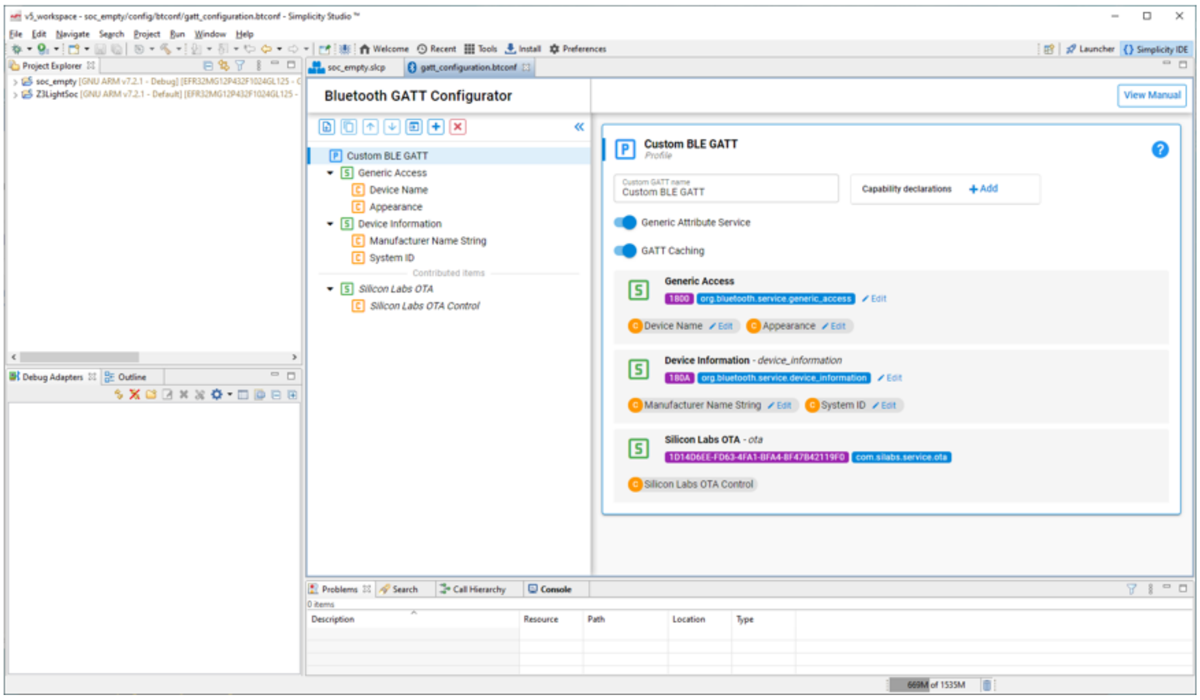
Bluetooth GATT Configurator

Bluetooth GATT Configurator

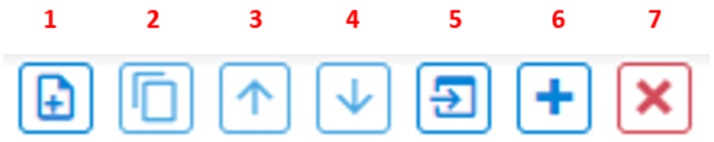
The Simplicity Studio® 5 (SSv5) Bluetooth GATT Configurator is a simple-to-use tool to help build a customized Bluetooth GATT database for Bluetooth projects. It is accessed through the Advanced Configurators software component group on the Project Configurator SOFTWARE COMPONENTS tab, or by double-clicking the project file *config > btconf > gatt_configuration.btconf*.

The Bluetooth GATT Configurator is composed of a Custom GATT editor on the left, showing a list of project Profiles/Services/Characteristics/Descriptors, and a Settings editor on the right. A SIG selector allows you to add standard elements to the profile. An options menu is provided at the top of the Custom GATT editor.

The Custom GATT editor is always visible, and the Settings editor opens by default.



The Bluetooth GATT Configurator menu is:

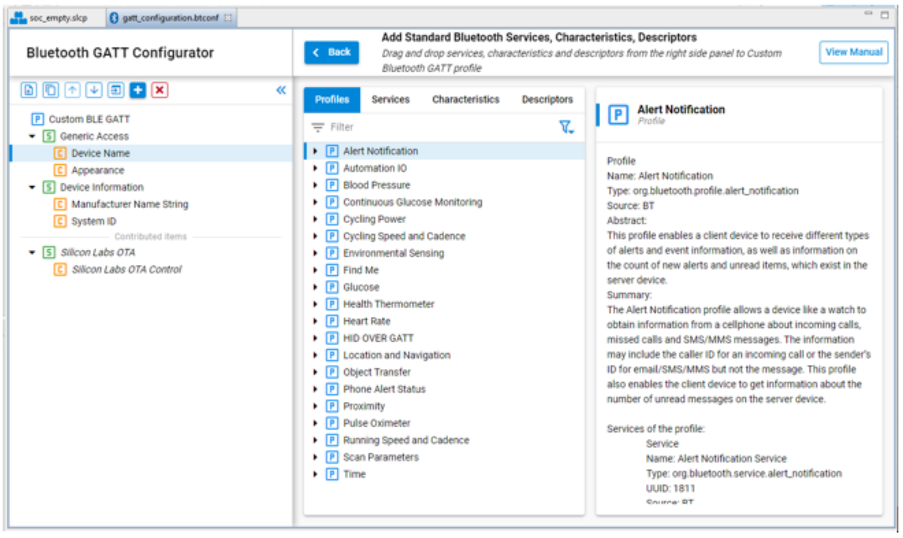


1. Add an item.
2. Duplicate the selected item.

3. Move the selected item up.
4. Move the selected item down.
5. Import a Bluetooth GATT database.
6. Add Predefined (opens the SIG editor).
7. Delete the selected item.

SIG Selector

Click the Add Predefined menu button (6) to open the SIG selector. The SIG Selector displays a list of predefined Profiles, Services, Characteristics, and Descriptors. These items can be filtered, using the filter pane. Tabs allow you to switch between different lists. As shown in the following figure, the pane on the right side of the list displays textual information about the latest selection. To add an item to the Custom GATT editor, mouse over it and click **+** on the right. The item can then be edited in the Settings section. The selected SIG service/characteristic/descriptor will be added under the highlighted profile/service/characteristic. Click **< BACK** to return to the Setting Editor. Click **View Manual** to see more information about Bluetooth GATT database items.



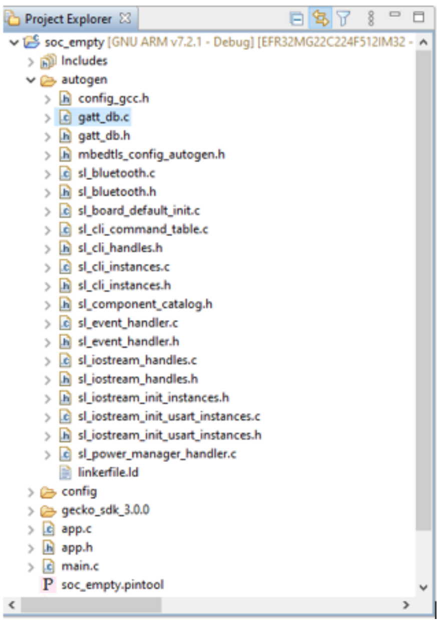
Add Standard Bluetooth Services, Characteristics, Descriptors

Drag and drop services, characteristics and descriptors from the right side panel to Custom Bluetooth GATT profile

Profile
Name: Alert Notification
Type: org.bluetooth.profile.alert_notification
Source: BT
Abstract:
This profile enables a client device to receive different types of alerts and event information, as well as information on the count of new alerts and unread items, which exist in the server device.
Summary:
The Alert Notification profile allows a device like a watch to obtain information from a cellphone about incoming calls, missed calls and SMS/MMS messages. The information may include the caller ID for an incoming call or the sender's ID for email/SMS/MMS but not the message. This profile also enables the client device to get information about the number of unread messages on the server device.

Services of the profile:
Service
Name: Alert Notification Service
Type: org.bluetooth.service.alert_notification
UUID: 1811
Source: BT

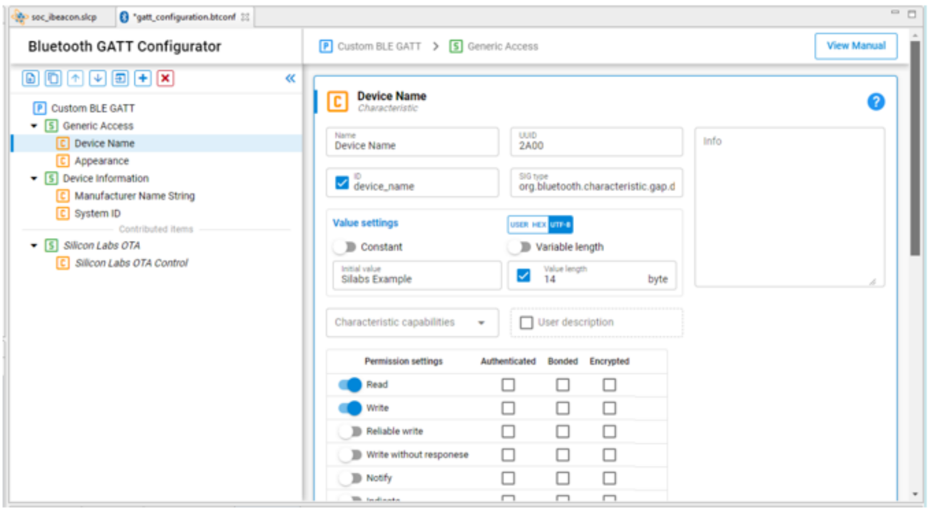
The GATT Configurator does not automatically save changes. An asterisk next to the configuration file name indicates unsaved changes in the configuration. Database generation happens automatically when the configuration is saved. The generated source files can be found in the *autogen* directory:



```
Project Explorer
soc_empty [GNU ARM v7.2.1 - Debug] [EFR32MG22C224F512IM32 -
  Includes
  autogen
    config_gcc.h
    gatt_db.c
    gatt_db.h
    mbedtls_config_autogen.h
    sl_bluetooth.c
    sl_bluetooth.h
    sl_board_default_init.c
    sl_cli_command_table.c
    sl_cli_handles.h
    sl_cli_instances.c
    sl_cli_instances.h
    sl_component_catalog.h
    sl_event_handler.c
    sl_event_handler.h
    sl_iostream_handles.c
    sl_iostream_handles.h
    sl_iostream_init_instances.h
    sl_iostream_init_usart_instances.c
    sl_iostream_init_usart_instances.h
    sl_power_manager_handler.c
    linkerfile.ld
  config
  gecko_sdk_3.0.0
  app.c
  app.h
  main.c
  soc_empty.pintool
```

Custom GATT Editor

The Custom GATT Editor in the right pane of the GATT Configurator displays the items present in the current configuration file. This includes a Custom GATT Profile, Services, Characteristics, and Descriptors displayed as a hierarchical list. The order of items shown reflects the order in which they exist in the GATT database. When the Settings Editor is open, you can select an item to see its properties and configuration.



soc_ibeacon.slcp | *gatt_configuration.btconf

Bluetooth GATT Configurator

- Custom BLE GATT
  - Generic Access
    - Device Name
    - Appearance
  - Device Information
    - Manufacturer Name String
    - System ID
  - Contributed items
  - *Silicon Labs OTA*
    - *Silicon Labs OTA Control*

Custom BLE GATT > Generic Access — View Manual

Device Name *Characteristic*

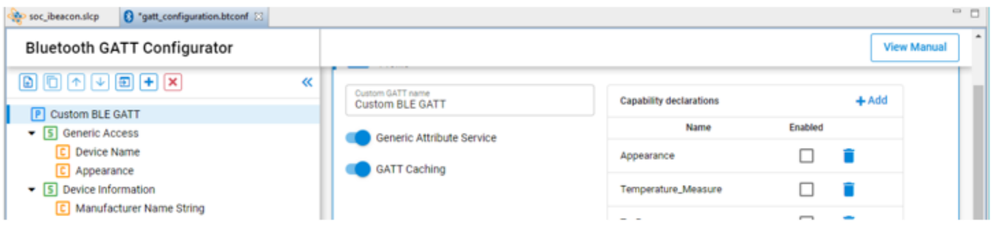
Name: Device Name
UUID: 2A00
ID: ✓ device_name
SIG type: org.bluetooth.characteristic.gap.d
Info

Value settings (USER / HEX / UTF-8)
Constant — Variable length
Initial value: Silabs Example
Value length: ✓ 14 byte

Characteristic capabilities
User description

Permission settings	Authenticated	Bonded	Encrypted
Read (on)	☐	☐	☐
Write (on)	☐	☐	☐
Reliable write	☐	☐	☐
Write without responese	☐	☐	☐
Notify	☐	☐	☐
Indicate	☐	☐	☐

Note: The Generic Attribute Service is not listed in the Custom GATT database structure. This is a special service that is maintained by the stack, and can be added by enabling the Generic Attribute Service slider in the settings of the Custom BLE GATT profile. Once enabled, the service will be part of the database. It still will not appear in the Custom GATT database structure or on iOS devices, as iOS hides this service, but you may see it on Android devices for example.


soc_ibeacon.slcp | *gatt_configuration.btconf

Bluetooth GATT Configurator

View Manual

- Custom BLE GATT
  - Generic Access
    - Device Name
    - Appearance
  - Device Information
    - Manufacturer Name String

Custom GATT name: Custom BLE GATT

Generic Attribute Service

GATT Caching

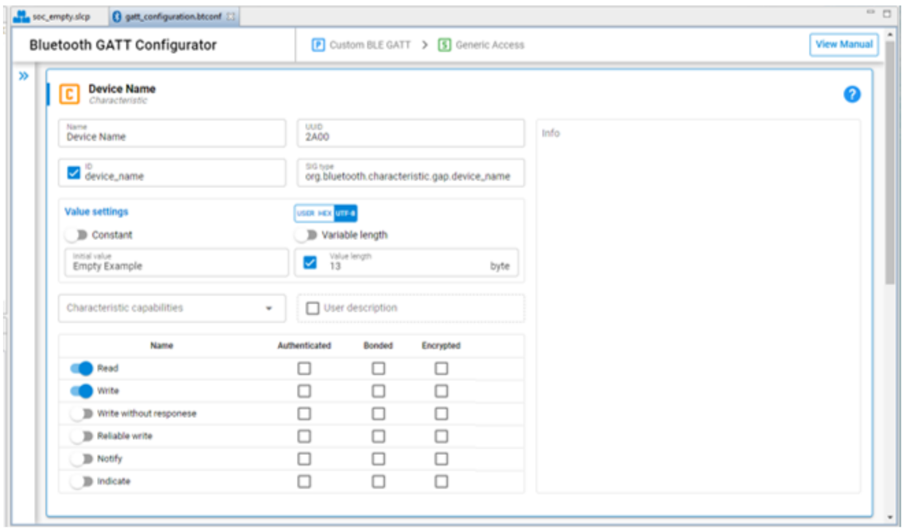
Capability declarations + Add

Name	Enabled
Appearance	☐
Temperature_Measure	☐

Some services are listed in the configurator as "contributed items". This means that their content is defined in other components, and they cannot be edited from this view.

Settings Editor

The Settings editor allows you to configure the properties of items such as Profiles, Services, Characteristics and Descriptors that are present in the Custom GATT editor. Selecting an item populates the relevant configuration options such as the name, ID, properties and capabilities. Any changes made in this section are reflected immediately for the selected item. You can minimize the Custom GATT editor while working in the Settings editor, as shown in the following figure. All Characteristics for a Service are included in the same Settings editor pane.


soc_empty.slcp | gatt_configuration.btconf

Bluetooth GATT Configurator

Custom BLE GATT > Generic Access

View Manual

Device Name
Characteristic

Name: Device Name

UUID: 2A00

ID: ✓ device_name

SIG type: org.bluetooth.characteristic.gap.device_name

Value settings: USER HEX [✓ UTF-8]

Constant

Variable length

Initial value: Empty Example

Value length: ✓ 13 byte

Characteristic capabilities

☐ User description

Info

Name	Authenticated	Bonded	Encrypted
● Read	☐	☐	☐
● Write	☐	☐	☐
Write without responese	☐	☐	☐
Reliable write	☐	☐	☐
Notify	☐	☐	☐
Indicate	☐	☐	☐

Additional Information

For more information on specific use cases in the GATT Configurator, see *UG438: GATT Configurator User's Guide in Bluetooth SDK 3.x.*

Bluetooth Mesh Configurator

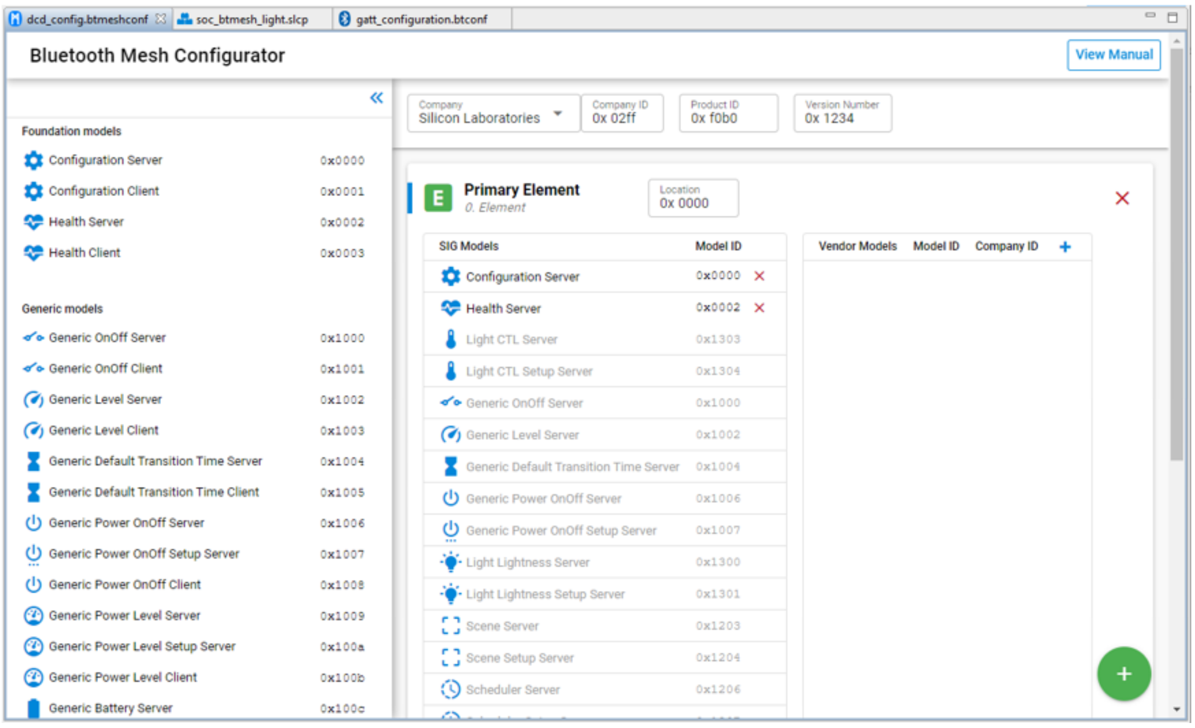
Bluetooth Mesh Configurator

The Bluetooth Mesh Configurator provides access to a Bluetooth Mesh node's Device Composition Data (DCD). This contains information about the elements it includes and the supported models. DCD exposes the node information to a configuration client so that it knows the potential functionalities the node supports and, based on that, can configure the node.

Configuring DCD through the Bluetooth Mesh Configurator is only one part of configuring a Bluetooth Mesh node. For details see *UG472: Bluetooth® Mesh Node Configurator User's Guide for SDK v2.x.*

When you create a Bluetooth Mesh project in Simplicity Studio 5, three tabs open automatically: the GATT Configurator (gatt_configuration.btconf), .the slcp or Project Configurator <projectname>.slcp, and the Bluetooth Mesh Configurator (dcd_config.btmeshconf). If the example has documentation, the project opens on a readme tab.

Click the dcd_config.btmeshconf tab to open the Bluetooth Mesh Configurator. If the tab is closed, you can open it from the Project Configurator's CONFIGURATION TOOLS tab. The Device Composition Data is presented in three areas: device information, elements, and models.



Foundation models

Configuration Server	0x0000
Configuration Client	0x0001
Health Server	0x0002
Health Client	0x0003

Generic models

Generic OnOff Server	0x1000
Generic OnOff Client	0x1001
Generic Level Server	0x1002
Generic Level Client	0x1003
Generic Default Transition Time Server	0x1004
Generic Default Transition Time Client	0x1005
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Generic Power OnOff Client	0x1008
Generic Power Level Server	0x1009
Generic Power Level Setup Server	0x100a
Generic Power Level Client	0x100b
Generic Battery Server	0x100c

Company: Silicon Laboratories
Company ID: 0x 02ff
Product ID: 0x f0b0
Version Number: 0x 1234

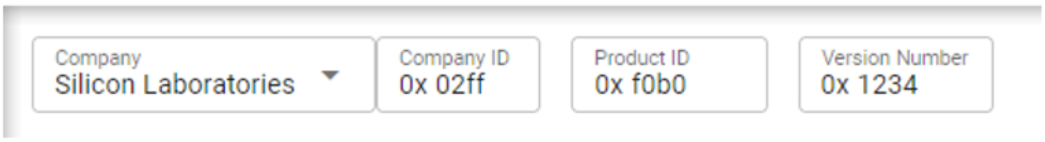
Primary Element
0. Element
Location: 0x 0000

SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Light CTL Server	0x1303
Light CTL Setup Server	0x1304
Generic OnOff Server	0x1000
Generic Level Server	0x1002
Generic Default Transition Time Server	0x1004
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Light Lightness Server	0x1300
Light Lightness Setup Server	0x1301
Scene Server	0x1203
Scene Setup Server	0x1204
Scheduler Server	0x1206

Vendor Models	Model ID	Company ID

Device Information

The device information card contains four fields. Changing the Company changes the next three fields.



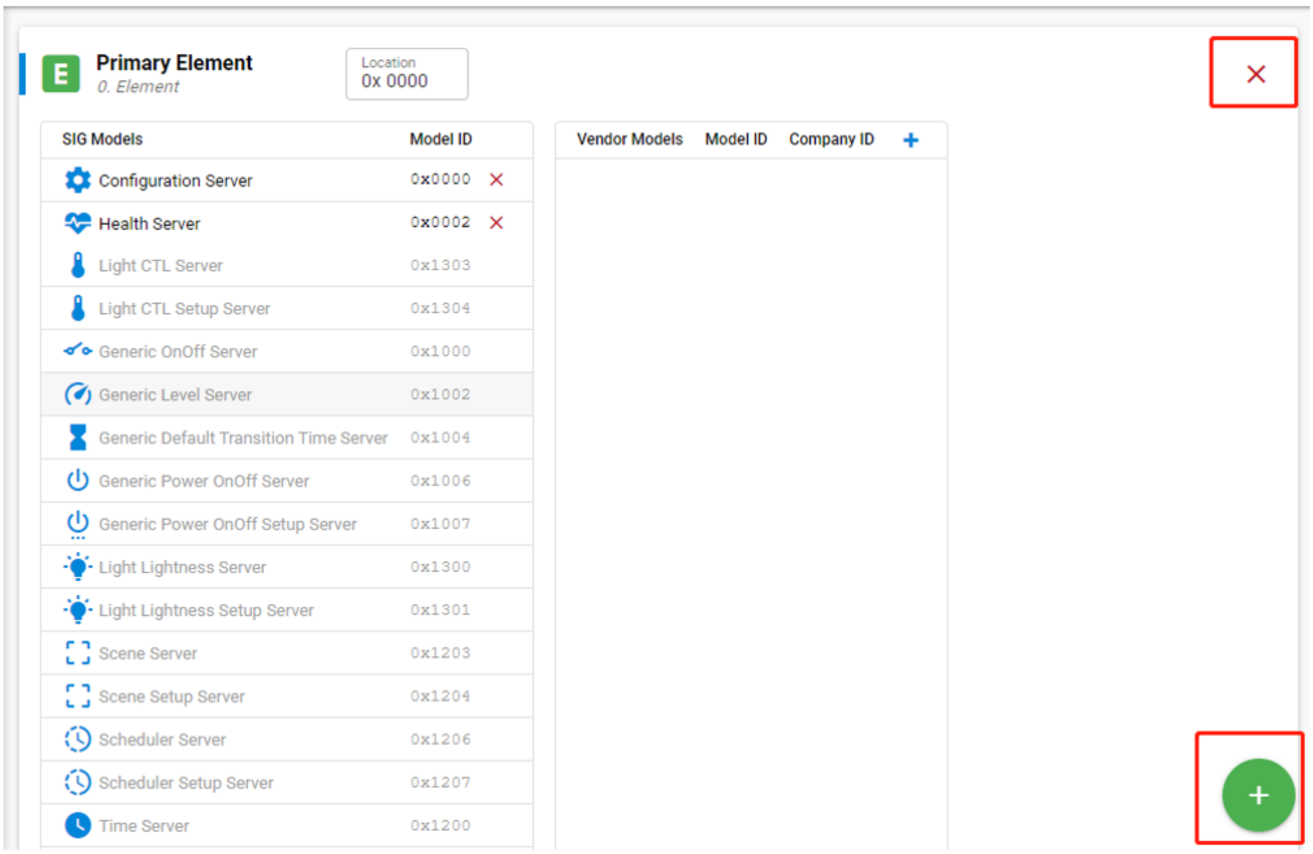
Company	Company ID	Product ID	Version Number
Silicon Laboratories	0x 02ff	0x f0b0	0x 1234

Device configuration information includes:

- Company: Selected from a list of company names.
- Company ID: 16-bit company identifier assigned by the Bluetooth SIG. A list of companies and their identifiers may be found on the Bluetooth SIG site.
- Product ID: 16-bit vendor-assigned product identifier, vendor-specific.
- Version Number: 16-bit vendor-assigned product version identifier, vendor-specific.

Elements

An element is an addressable entity within a node. Each node can have one or more elements, the first called the primary element and the others called secondary elements. Each element is assigned a unicast address during provisioning so that it can be used to identify which node is transmitting or receiving a message. The primary element is addressed using the first unicast address assigned to the node, and the secondary elements are addressed using the subsequent addresses. Both primary and secondary elements have a dedicated card, such as that shown in the following figure, through which they can be configured. Click the green plus symbol to add an element or select an element and click the red X symbol to remove it.



Primary Element
0. Element

Location
0x 0000

SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Light CTL Server	0x1303
Light CTL Setup Server	0x1304
Generic OnOff Server	0x1000
Generic Level Server	0x1002
Generic Default Transition Time Server	0x1004
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Light Lightness Server	0x1300
Light Lightness Setup Server	0x1301
Scene Server	0x1203
Scene Setup Server	0x1204
Scheduler Server	0x1206
Scheduler Setup Server	0x1207
Time Server	0x1200

Vendor Models	Model ID	Company ID

Models

A model defines the basic functionality of a node, and a node may include multiple models. A model defines the required states, the messages that act upon those states, and any associated behaviors.

Models may be defined and adopted by the Bluetooth SIG and may also be defined by vendors. Models defined by the Bluetooth SIG are known as SIG-adopted models, and models defined by vendors are known as vendor models. SIG-adopted models are identified by a 16-bit model identifier and vendor models are identified by a 16-bit vendor identifier and a 16-bit model identifier.

Model specifications are designed to be very small and self-contained. At specification definition time, a model can require other models that must also be instantiated within the same node. This is called extending, which means a model adds functionality on top of other models.

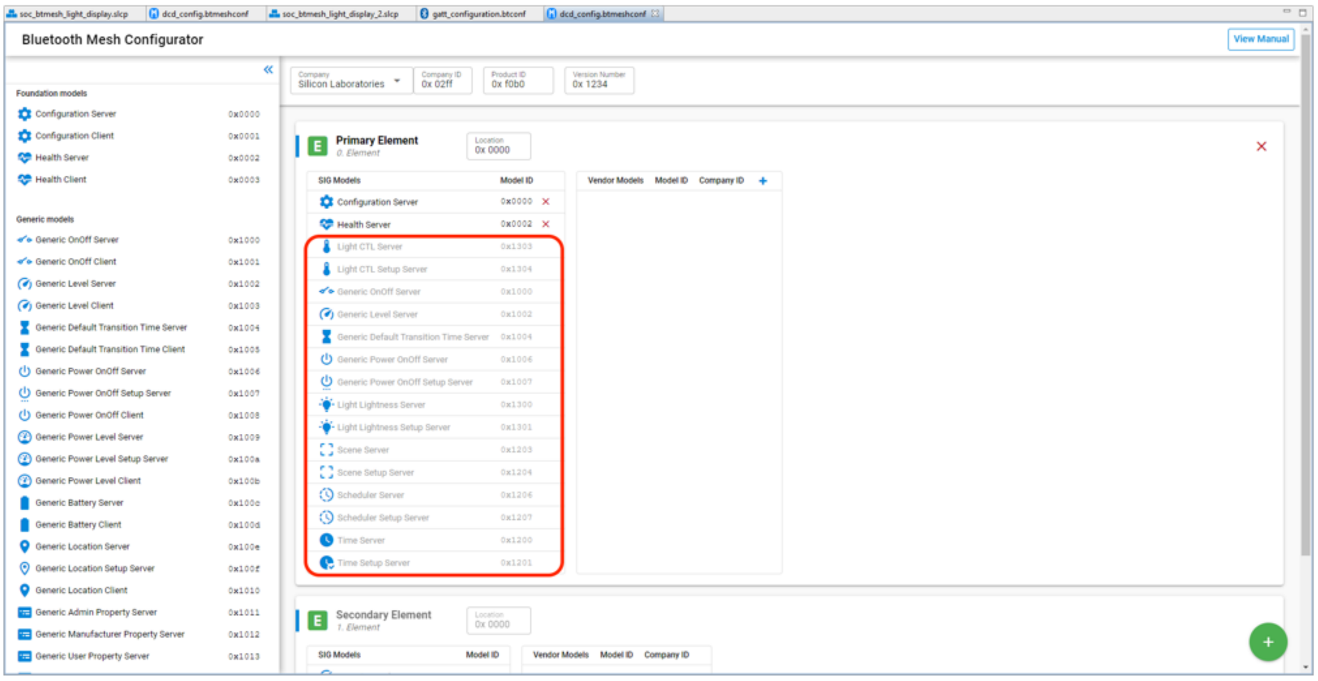
Models that do not extend other models are referred to as root models. Model specifications are immutable. In other words, it is not possible to remove or add behavior to a model, whether the desired behavior is optional behavior or mandatory. Models are not versioned and have no feature bits. If additional behavior is required in a model, then a new extended model is defined that exposes the required behavior and can be implemented alongside the original model.

Therefore, knowledge of the models supported by an element determines the exact behavior exposed by that element.

The Bluetooth Mesh Configurator supports configuring both SIG-adopted models and vendor models through separate editors.

SIG-Adopted Model Editor

If you are using the provided model components that automatically bring in the source/header files, libraries, and configurations to the project, and also contribute the model to the DCD, you cannot edit or delete the model from the DCD manually. The model is greyed out, as shown in the following figure. In this case, all the model implementations will be generated to the project. You can modify the callbacks to adjust the application to your use case.


soc_btmesh_light_display.slcp
dcd_config.btmeshconf
soc_btmesh_light_display_2.slcp
gatt_configuration.btconf
dcd_config.btmeshconf
View Manual

Bluetooth Mesh Configurator

Company: Silicon Laboratories
Company ID: 0x 02ff
Product ID: 0x f0b0
Version Number: 0x 1234

Foundation models

Configuration Server	0x0000
Configuration Client	0x0001
Health Server	0x0002
Health Client	0x0003

Generic models

Generic OnOff Server	0x1000
Generic OnOff Client	0x1001
Generic Level Server	0x1002
Generic Level Client	0x1003
Generic Default Transition Time Server	0x1004
Generic Default Transition Time Client	0x1005
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Generic Power OnOff Client	0x1008
Generic Power Level Server	0x1009
Generic Power Level Setup Server	0x100a
Generic Power Level Client	0x100b
Generic Battery Server	0x100c
Generic Battery Client	0x100d
Generic Location Server	0x100e
Generic Location Setup Server	0x100f
Generic Location Client	0x1010
Generic Admin Property Server	0x1011
Generic Manufacturer Property Server	0x1012
Generic User Property Server	0x1013

Primary Element
0. Element
Location: 0x 0000

SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Light CTL Server	0x1303
Light CTL Setup Server	0x1304
Generic OnOff Server	0x1000
Generic Level Server	0x1002
Generic Default Transition Time Server	0x1004
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Light Lightness Server	0x1300
Light Lightness Setup Server	0x1301
Scene Server	0x1203
Scene Setup Server	0x1204
Scheduler Server	0x1206
Scheduler Setup Server	0x1207
Time Server	0x1200
Time Setup Server	0x1201

Vendor Models	Model ID	Company ID

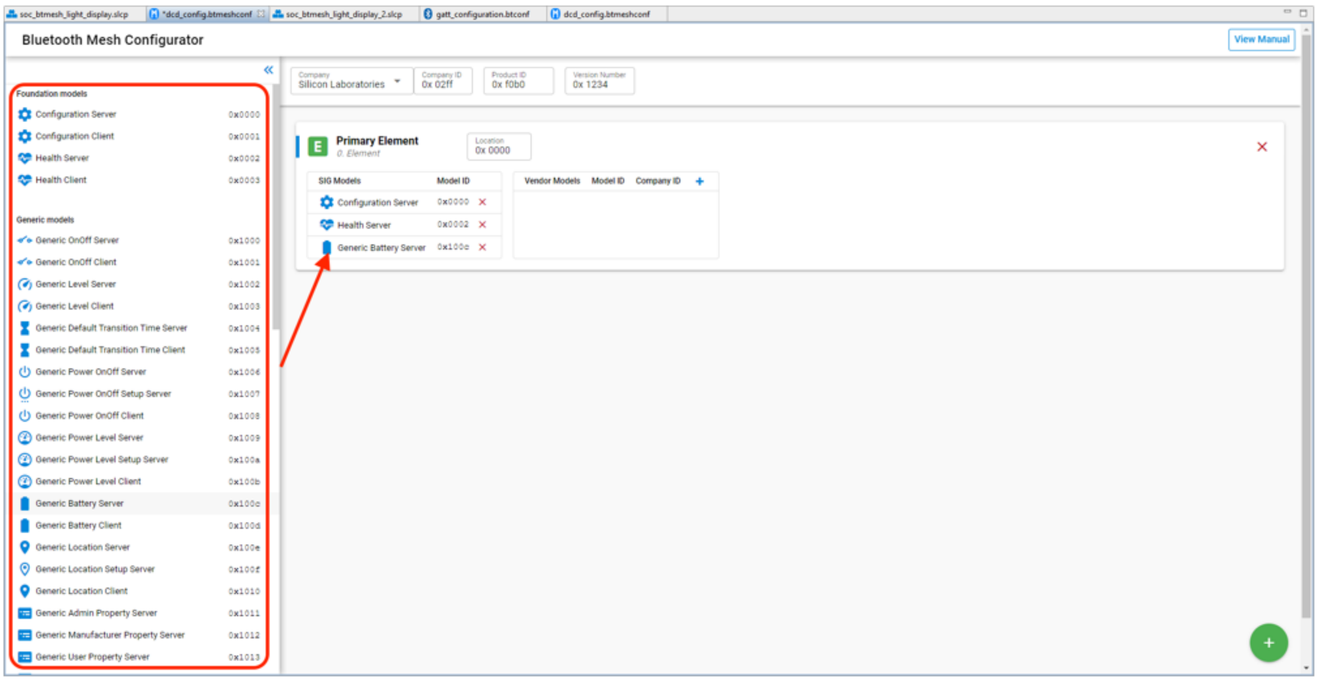
Secondary Element
1. Element
Location: 0x 0000

SIG Models	Model ID

Vendor Models	Model ID	Company ID

The drawback to using components is that you cannot edit the DCD and model information, because the models added by components are greyed out. If you want to build the DCD from scratch, for example to add a specific model to an element, uninstall all the model components. Then you can edit the DCD manually.

To delete a model, select it and click the red X symbol. To add a SIG-adopted model, drag the model from the left model pool to the SIG Models table in the correct element. A list of all the SIG-adopted models is displayed, and you can choose the one you want, as shown in the following figure. Note that, although all the SIG-adopted models are listed, not all of them are currently supported by the Bluetooth Mesh SDK. For the information on the supported models, see the SDK release notes.


soc_btmesh_light_display.slcp
*dcd_config.btmeshconf
soc_btmesh_light_display_2.slcp
gatt_configuration.btconf
dcd_config.btmeshconf
Bluetooth Mesh Configurator
View Manual

Foundation models	
Configuration Server	0x0000
Configuration Client	0x0001
Health Server	0x0002
Health Client	0x0003

Generic models	
Generic OnOff Server	0x1000
Generic OnOff Client	0x1001
Generic Level Server	0x1002
Generic Level Client	0x1003
Generic Default Transition Time Server	0x1004
Generic Default Transition Time Client	0x1005
Generic Power OnOff Server	0x1006
Generic Power OnOff Setup Server	0x1007
Generic Power OnOff Client	0x1008
Generic Power Level Server	0x1009
Generic Power Level Setup Server	0x100a
Generic Power Level Client	0x100b
Generic Battery Server	0x100c
Generic Battery Client	0x100d
Generic Location Server	0x100e
Generic Location Setup Server	0x100f
Generic Location Client	0x1010
Generic Admin Property Server	0x1011
Generic Manufacturer Property Server	0x1012
Generic User Property Server	0x1013

Company: Silicon Laboratories
Company ID: 0x 02ff
Product ID: 0x f0b0
Version Number: 0x 1234

Primary Element
0. Element
Location: 0x 0000

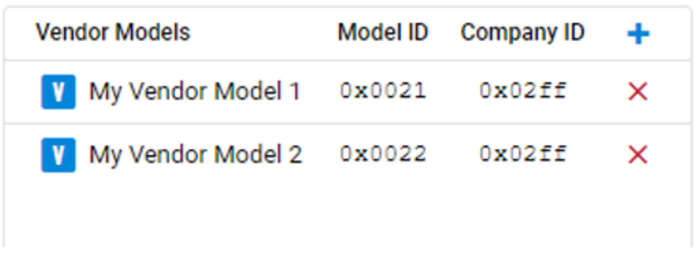
SIG Models	Model ID
Configuration Server	0x0000
Health Server	0x0002
Generic Battery Server	0x100c

Vendor Models	Model ID	Company ID

Due to the extension mechanism of models mentioned above, attention is needed when adding models to your project. See *UG472: Bluetooth® Mesh Node Configurator User's Guide for SDK v2.x* for more information.

Vendor Models Editor

Vendor models give you more flexibility when developing products not covered by the SIG-adopted models. Vendors can define their own specification in these vendor models, including states, messages, and the associated behaviors. The vendor model editor is shown in the following figure. The ID field contains the 32-bit vendor identifier and model identifier. The two least significant bytes of the ID are the vendor ID and the two most significant bytes are the model ID. In the following figure, 0x02FF is the vendor ID for Silicon Labs, and 0x0021 and 0x0022 is the model ID. Click the plus symbol to add a vendor model, or select a model and click the red X symbol to remove it.

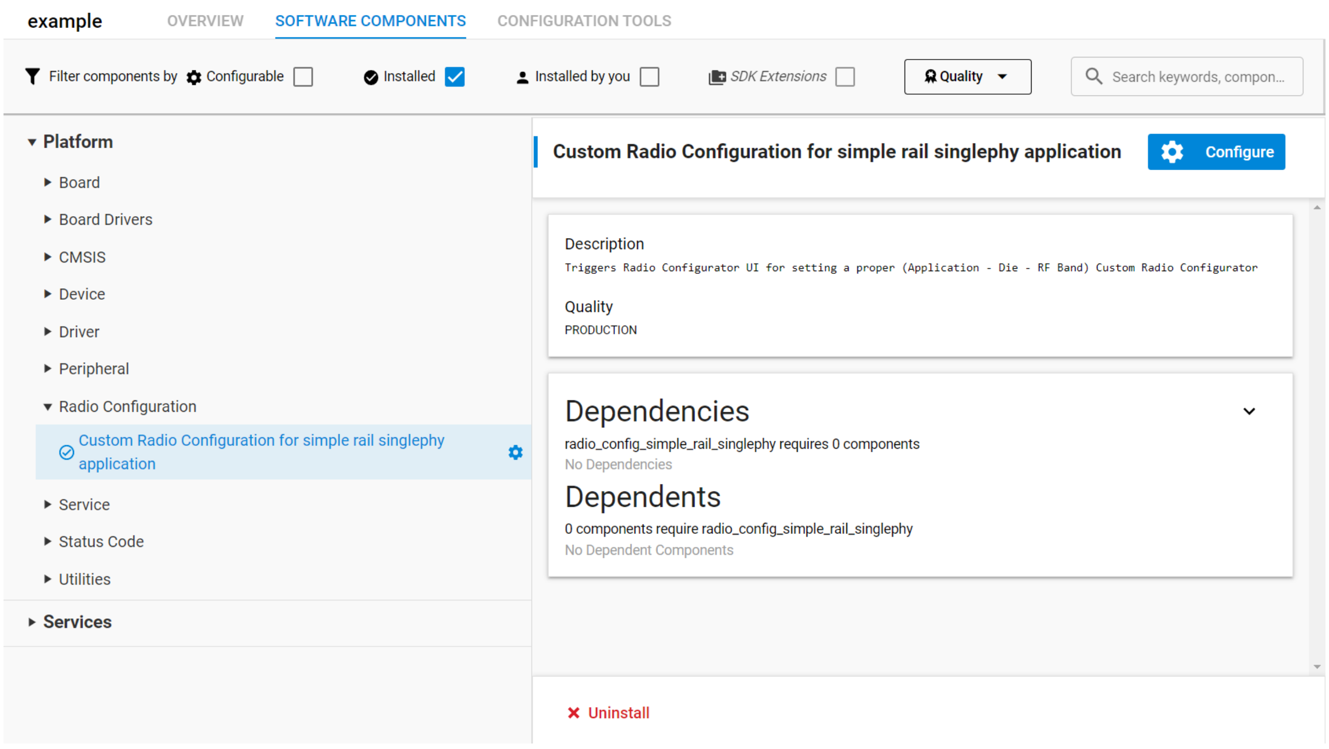


Vendor Models	Model ID	Company ID
My Vendor Model 1	0x0021	0x02ff
My Vendor Model 2	0x0022	0x02ff

Proprietary Radio Configurator

Proprietary Radio Configurator

The Simplicity Studio® 5 (SSv5) Radio Configurator is provided as part of the Proprietary SDK. Use the Radio Configurator to create standard or custom radio configurations for your RAIL-based radio applications. The Radio Configurator is accessed through the through the Platform group's **Custom Radio Configuration for simple rail singlephy application** component, on the Project Configurator SOFTWARE COMPONENTS tab and can be accessed on the CONFIGURATION TOOLS tab. (For some examples, the Radio Configurator might open on project creation). All radio configurator settings are stored in *config/rail/radio_settings.radioconf*. Changes made with Radio Configurator are not automatically saved.



Custom Radio Configuration for simple rail singlephy application

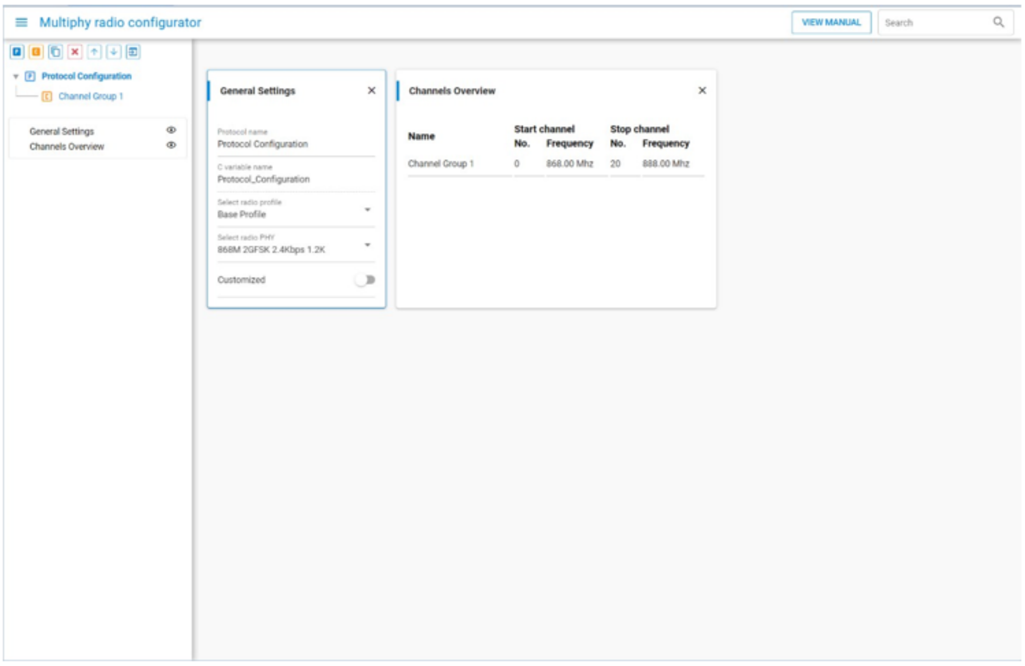
Description
Triggers Radio Configurator UI for setting a proper (Application - Die - RF Band) Custom Radio Configurator

Quality
PRODUCTION

Dependencies
radio_config_simple_rail_singlephy requires 0 components
No Dependencies

Dependents
0 components require radio_config_simple_rail_singlephy
No Dependent Components

All the parameters in the Radio Configurator are arranged in cards, some of which are grouped together. Each card contains entries that logically go together. Different radio profiles (as described under Protocol) offer different views and parameter sets, as a profile is a high-level view of the parameter set valid for and describing a given radio link.



General Settings

Protocol name: Protocol Configuration

C variable name: Protocol_Configuration

Select radio profile: Base Profile

Select radio PHY: 868M 2GFSK 2.4Kbps 1.2K

Customized

Channels Overview

Name	Start channel		Stop channel	
	No.	Frequency	No.	Frequency
Channel Group 1	0	868.00 Mhz	20	888.00 Mhz

A radio configuration has two hierarchical levels: Protocol level and Channel Group level. A radio configuration can contain multiple protocols and a protocol can have multiple channel groups defined.

Protocol

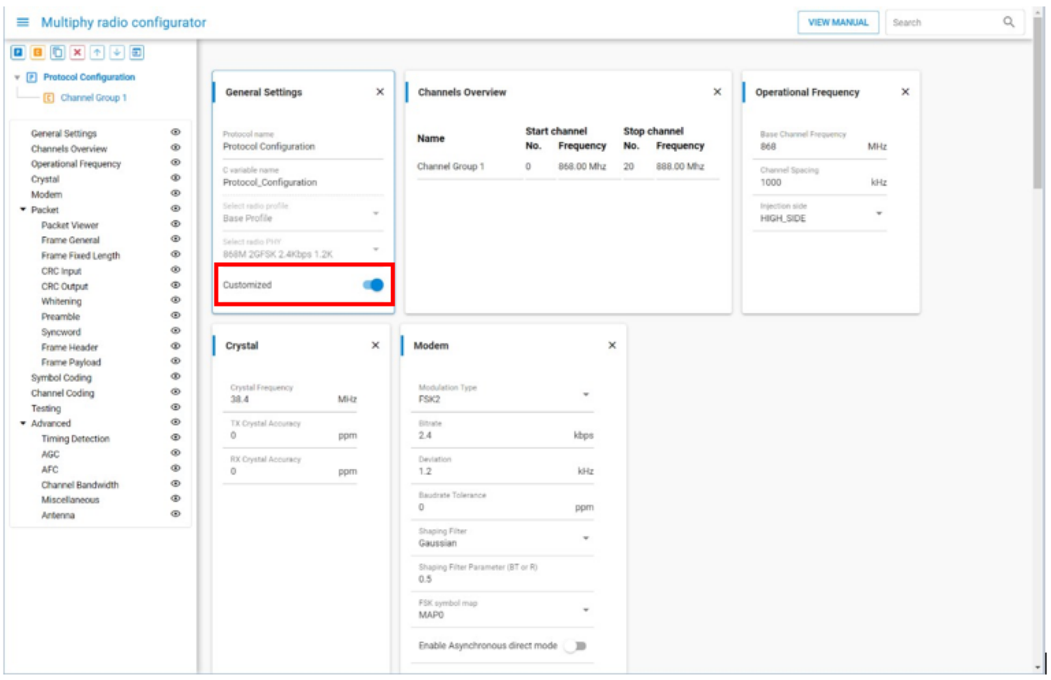
Protocols are complete radio configurations that can be switched using the `RAIL_ConfigChannels()` API, or can be used in Dynamic Multiprotocol applications. For Channel Group definitions see Channel Groups.

To configure a protocol, first **select a predefined PHY** configuration, then **customize** it to meet your needs.

First, look at the **General Settings** card. Select a radio profile in the **Select radio profile** drop-down menu. A radio profile may be any supported radio link technology. These technologies can be bound by standards (for example the Sigfox or WMBus protocols) or can be fully customized. The fully customizable profile is called the **"Base Profile"**.

Once the radio profile has been selected, the next step is to select a radio PHY (radio configuration) in the **Select a radio PHY** dropdown list. Each profile has "built-in" configurations ready to use.

Once the radio profile and radio PHY have been selected, users can review the **profile options**. By default, no changes are allowed, fields are grayed out. To enable customization, use the **Customized** switch on the **General Settings** card. This allows access to all the parameters defined by the profile.



General Settings
Protocol name: Protocol Configuration
C variable name: Protocol_Configuration
Select radio profile: Base Profile
Select radio PHY: 868M 2GFSK 2.4Kbps 1.2K
Customized: on

Channels Overview

Name	Start channel No.	Start channel Frequency	Stop channel No.	Stop channel Frequency
Channel Group 1	0	868.00 Mhz	20	888.00 Mhz

Operational Frequency
Base Channel Frequency: 868 MHz
Channel Spacing: 1000 kHz
Injection side: HIGH_SIDE

Crystal
Crystal Frequency: 38.4 MHz
TX Crystal Accuracy: 0 ppm
RX Crystal Accuracy: 0 ppm

Modem
Modulation Type: FSK2
Bitrate: 2.4 kbps
Deviation: 1.2 kHz
Baudrate Tolerance: 0 ppm
Shaping Filter: Gaussian
Shaping Filter Parameter (BT or R): 0.5
FSK symbol map: MAP0
Enable Asynchronous direct mode: off

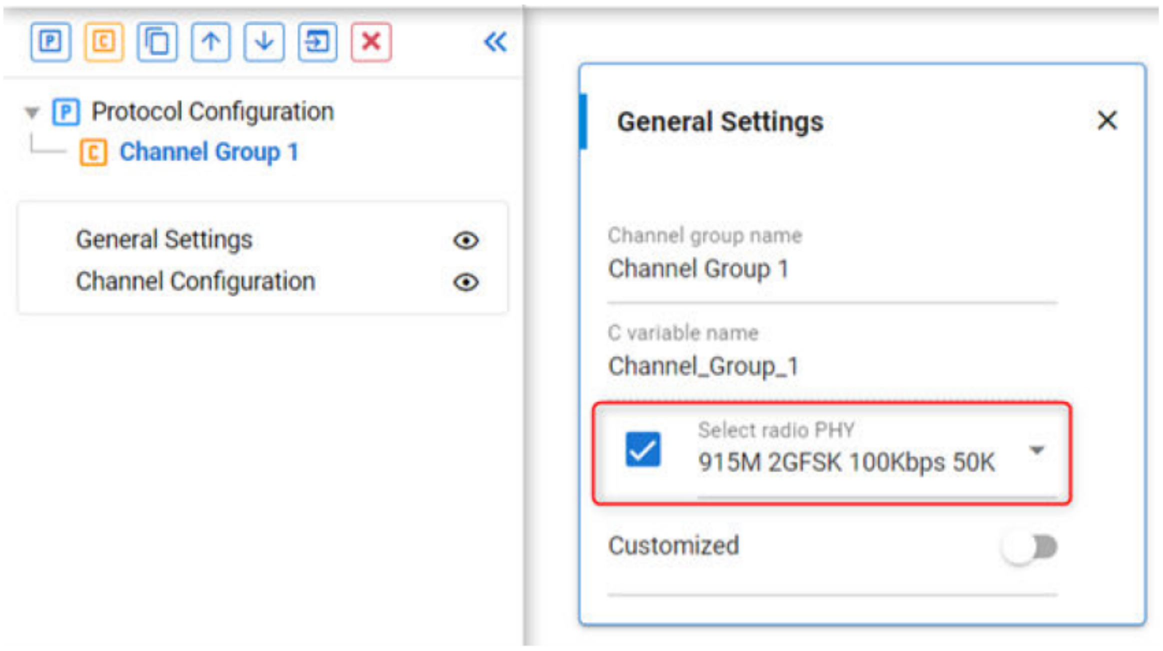
Important notes:

1. If you select a "built-in" PHY, and then switch to "Customized", the Radio Configurator retains the property values of the "built-in" PHY. You can edit the values, but can also revert to the defaults.
2. If you switch to “Customized” mode, Silicon Labs **recommends unchecking all "Advanced"** properties, as those are fine-tuned for the "built-in" PHY, and may not be the optimal choice for the modified PHY. This way those parameters get auto-calculated, and you can experiment with the fine tuning starting with the calculated values. To keep the improved performance achieved by the original optimization, only minor changes should be made. For example, <100 MHz change in carrier frequency, or a different frame length configuration.
3. If you switch customization off, your **modifications will be reverted** to the property values of the "built-in" PHY.
4. Each menu item in the Navigation pane (on the left) is represented by a card in the main editor panel (on the right). **Cards can be hidden** by clicking the corresponding "eye" icon on the Navigation panel.

Based on the selected radio profile, customizable options may be restricted. For example, if a radio profile is selected that is bound by a standard, the profile options only allow users to set the base frequency. All other options are preset according to the standard.

Channel Groups

Each protocol configuration includes one or more channel group configurations. Channel groups define one or more (sequential) channels, with a constant channel spacing between them. Channel groups can differ in the radio configuration both from each other and from the parent protocol. By default, a channel group configuration includes only the **General Settings** and **Channel Configuration** cards. Additional parameters defined by the Protocol can be accessed for customization on a channel group basis by sliding the **Customized** switch on the corresponding card. Channel groups can also be completely overridden to a predefined PHY by checking the **Select radio PHY** box.



Channel group name: Channel Group 1

C variable name: Channel_Group_1

Select radio PHY: 915M 2GFSK 100Kbps 50K

Customized

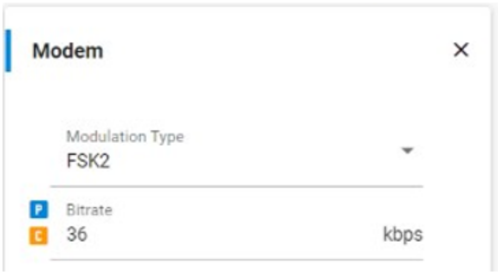
RAIL automatically detects when hopping to a new channel requires hopping between channel groups. The configured property values defined by the channel group will be applied automatically for the new channel. This enables users to define virtual channels to the same physical frequency, but with different configuration settings.

The order of the channel groups will be the same in the `RAIL_ChannelConfigEntry_t` array as shown in the Radio Configurator. Channel group order can be used to override channels in channel groups, as RAIL will always load the channel from the first channel group in which it is defined.

For more on Multi-PHY configuration, see the example in AN1253: EFR32 Radio Configurator Guide for Simplicity Studio 5.

Finalizing a Configuration

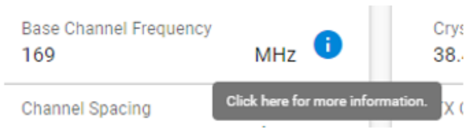
When a parameter is modified from its pre-loaded value, small pictograms show up next to the property field on the card. These pictograms symbolize the differences against the originally selected PHY configurations. A "C" means a difference from the original Channel Group property, and a "P" means a difference from the original Protocol Configuration.



Modulation Type: FSK2

Bitrate: 36 kbps

Each input field has an **Information** icon next to it, which opens the embedded version of AN1253: EFR32 Radio Configurator Guide for Simplicity Studio 5 at the relevant section. You can also reach the documentation by clicking **View Manual** in the top right corner of the perspective.



On save the Radio Configurator generates *rail_config.c* and *rail_config.h* in the project's *autogen* directory. These files are also generated when a project is created.

Wi-SUN Configurator

Wi-SUN Configurator

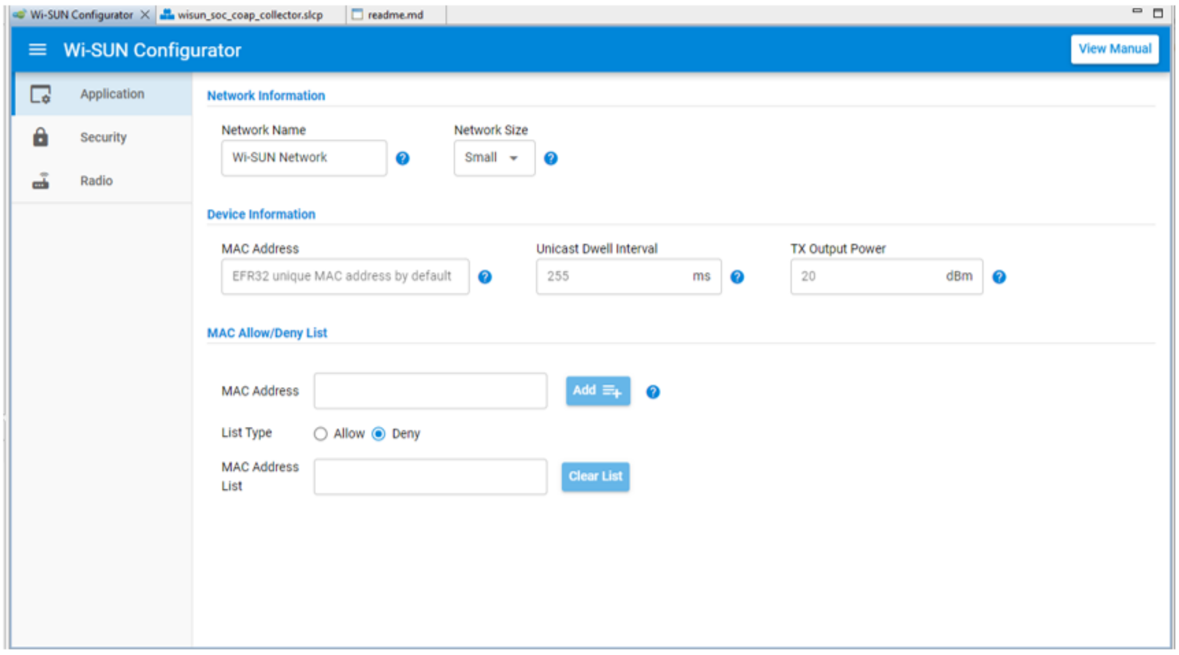
When you create a new Wi-SUN project, a Wi-SUN Configurator is added to it by default. The Wi-SUN Configurator provides an interface to the Wi-SUN application's main settings through three panels: Application, Security, and Radio. For some examples, the Wi-SUN Configurator only displays the Radio panel. These examples do not have the application and security infrastructure.

The Wi-SUN Configurator tab can be displayed by opening /config/wisun/wisun_settings.wisunconf. For more information, click **View Manual**.

Application Panel

The Application panel exposes multiple Wi-SUN stack settings associated with the application, such as:

- The network name the device will try to connect to
- The network size setting
- The device’s TX output power
- The unicast dwell interval

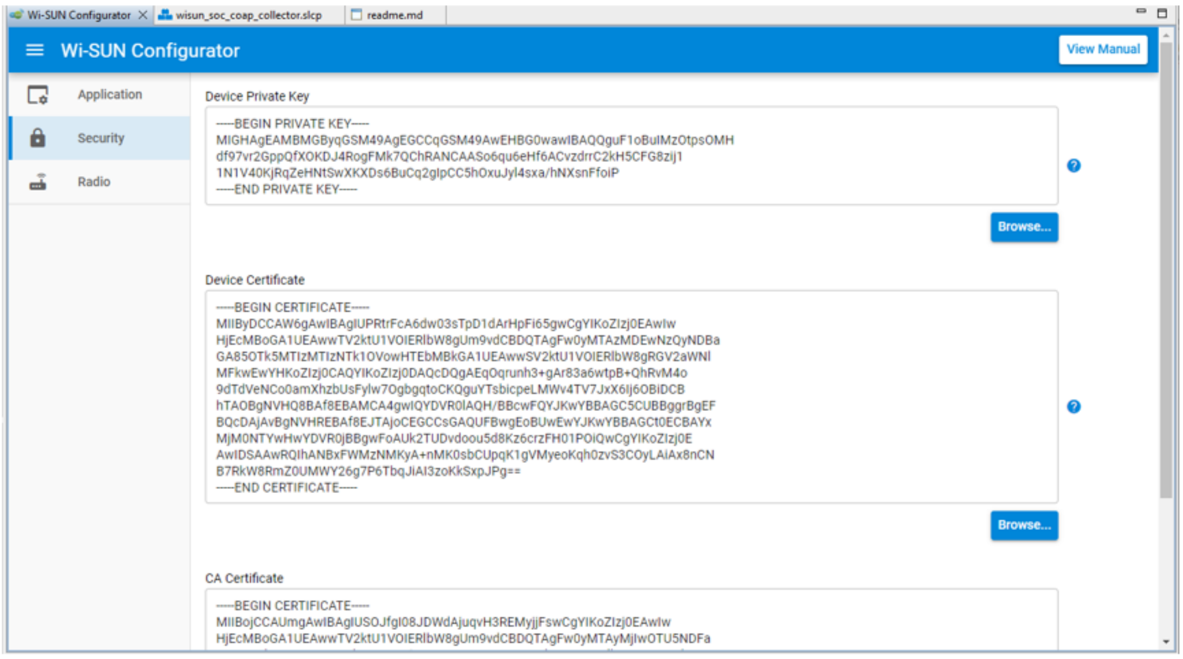


Network Name: Wi-SUN Network
Network Size: Small
MAC Address: EFR32 unique MAC address by default
Unicast Dwell Interval: 255 ms
TX Output Power: 20 dBm
MAC Address:
List Type: Allow [✓ Deny]
MAC Address List:

The MAC address filtering feature provides a way to force a topology on a Wi-SUN network. It is a simpler alternative to spacing out the Wi-SUN devices to achieve the desired topology. A device either does or does not interact with other Wi-SUN devices on the list, depending on the allow/deny option.

Security Panel

The Security panel displays the private key and certificates used by the device to authenticate itself when connecting to a Wi-SUN network. By default, it uses the Silicon Labs demonstration samples. They can be modified to use a distinct certificate infrastructure aligned with the border router or authenticator certificate.



Device Private Key

```
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQguF1oBulMzOtpsOMH
df97vr2GppQfXOKDJ4RogFMk7QChRANCAASo6qu6eHf6ACvzdrrC2kH5CFG8zij1
1N1V40KjRqZeHNtSwXKXDs6BuCq2glpCC5hOxuJyl4sxa/hNXsnFfoiP
-----END PRIVATE KEY-----
```

Device Certificate

```
-----BEGIN CERTIFICATE-----
MIIByDCCAW6gAwIBAgIUPRtrFcA6dw03sTpD1dArHpFi65gwCgYIKoZIzj0EAwIw
HjEcMBoGA1UEAwwTV2ktU1VOIERlbW8gUm9vdCBDQTAgFw0yMTAzMDEwNzQyNDBa
GA85OTk5MTIzMTIzNTk1OVowHTEbMBkGA1UEAwwSV2ktU1VOIERlbW8gRGV2aWNl
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEqOqrunh3+gAr83a6wtpB+QhRvM4o
9dTdVeNCo0amXhzbUsFylw7OgbgqtoCKQguYTsbicpeLMWv4TV7JxX6Ij6OBiDCB
hTAOBgNVHQ8BAf8EBAMCA4gwIQYDVR0lAQH/BBcwFQYJKwYBBAGC5CUBBggrBgEF
BQcDAjAvBgNVHREBAf8EJTAjoCEGCCsGAQUFBwgEoBUwEwYJKwYBBAGCt0ECBAYx
MjM0NTYwHwYDVR0jBBgwFoAUk2TUDvdoou5d8Kz6crzFH01POiQwCgYIKoZIzj0E
AwIDSAAwRQIhANBxFWMzNMKyA+nMK0sbCUpqK1gVMyeoKqh0zvS3COyLAiAx8nCN
B7RkW8RmZ0UMWY26g7P6TbqJiAI3zoKkSxpJPg==
-----END CERTIFICATE-----
```

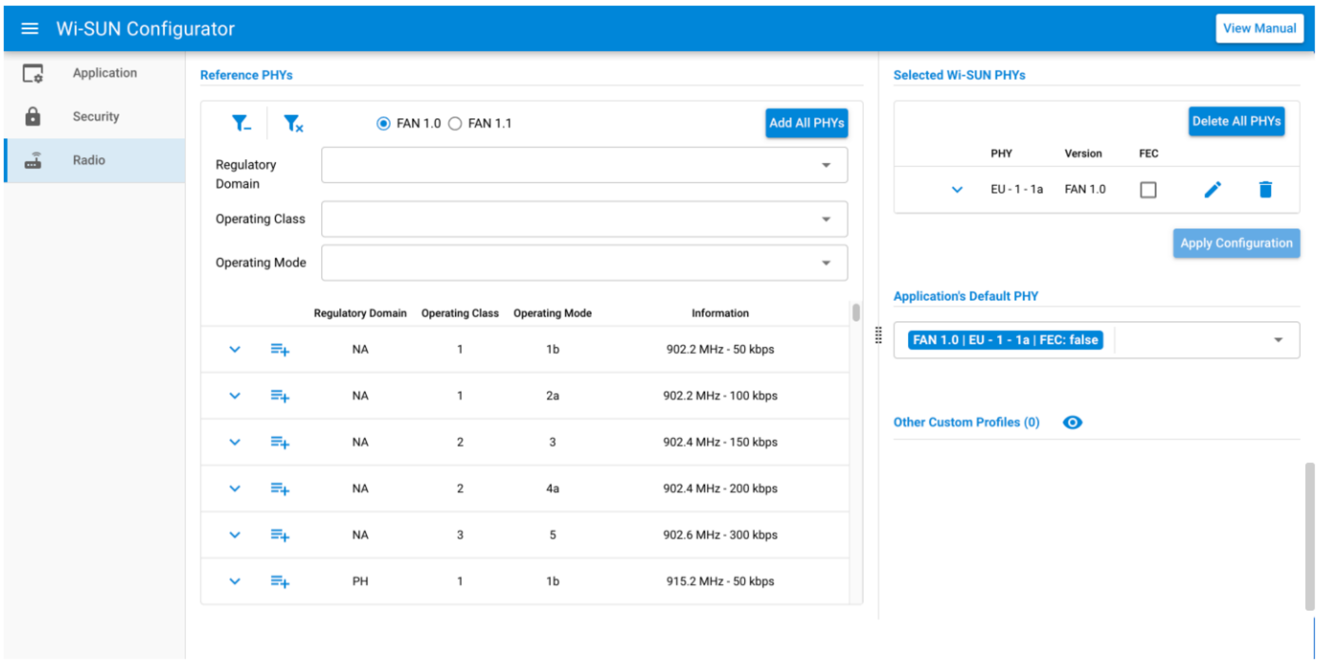
CA Certificate

```
-----BEGIN CERTIFICATE-----
MIIBojCCAUmgAwIBAgIUSOJfgI08JDWdAjuqvH3REMyjjFswCgYIKoZIzj0EAwIw
HjEcMBoGA1UEAwwTV2ktU1VOIERlbW8gUm9vdCBDQTAgFw0yMTAyMjIwOTU5NDFa
```

Radio Panel

The Radio panel is an interface to configure the radio profiles included in a Wi-SUN application. It provides a user interface to access any specified Wi-SUN FAN 1.0 or FAN 1.1 PHY. A radio button allows the user to choose between the FAN 1.0 or FAN 1.1 context.

The complete list can be filtered to help you find the right PHY configuration. An application can embed several PHYs from different regions and different specification versions.



Reference PHYs

(● FAN 1.0) ○ FAN 1.1

Regulatory Domain

Operating Class

Operating Mode

Regulatory Domain	Operating Class	Operating Mode	Information
NA	1	1b	902.2 MHz - 50 kbps
NA	1	2a	902.2 MHz - 100 kbps
NA	2	3	902.4 MHz - 150 kbps
NA	2	4a	902.4 MHz - 200 kbps
NA	3	5	902.6 MHz - 300 kbps
PH	1	1b	915.2 MHz - 50 kbps

Selected Wi-SUN PHYs

PHY	Version	FEC
EU - 1 - 1a	FAN 1.0	☐

Application's Default PHY

FAN 1.0 \| EU - 1 - 1a \| FEC: false

Other Custom Profiles (0)

The **Application's Default PHY** input defines the PHY that the application starts with. The default value depends on the EFR32 radio board used to create the project. For example, a BRD4163A radio board supporting the 868 MHz band defaults to the mandatory Wi-SUN PHY for the European region (that is, Wi-SUN FAN EU - 1 – 1a). On the other hand, a BRD4164A radio board supporting the 915 MHz band defaults to the North America mandatory Wi-SUN PHY (that is, Wi-SUN FAN NA - 1 – 1b). The user can always open the dropdown list and select another default PHY.

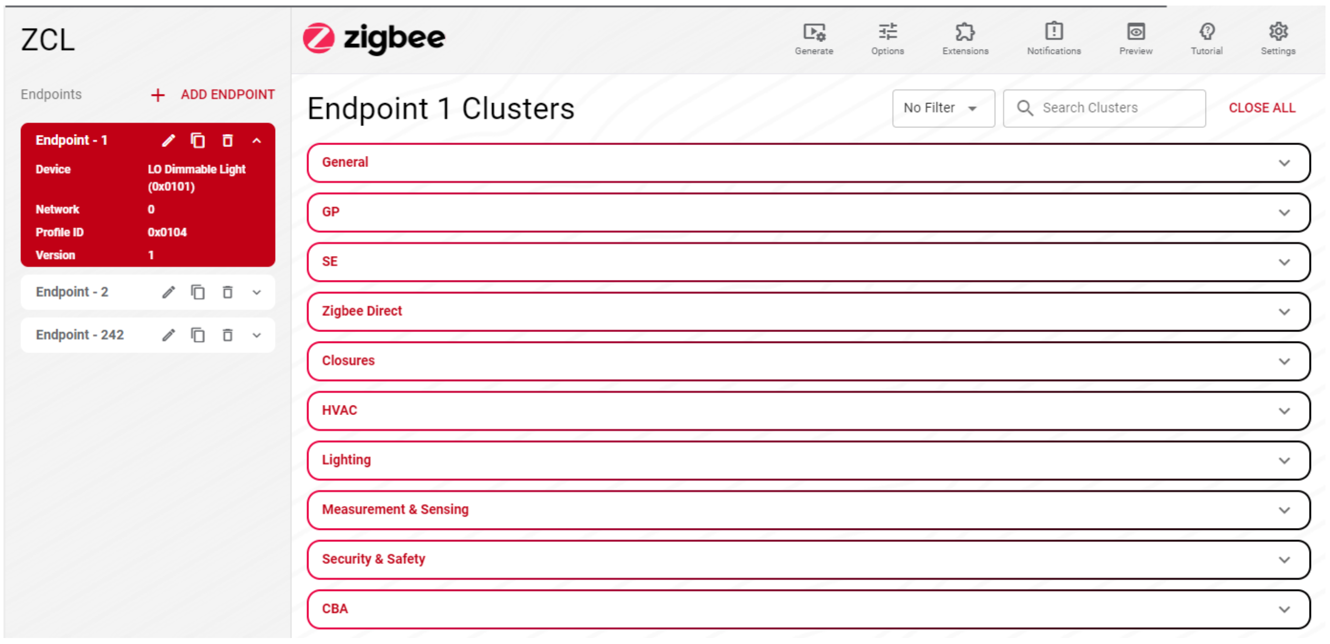
Every selected Wi-SUN PHY can be edited in the Radio Configurator by clicking the pen icon. This action opens the Radio Configurator user interface on the selected Wi-SUN PHY. Moreover, non Wi-SUN FAN PHY are listed under "Other Custom Profile" for information.

ZCL Advanced Platform

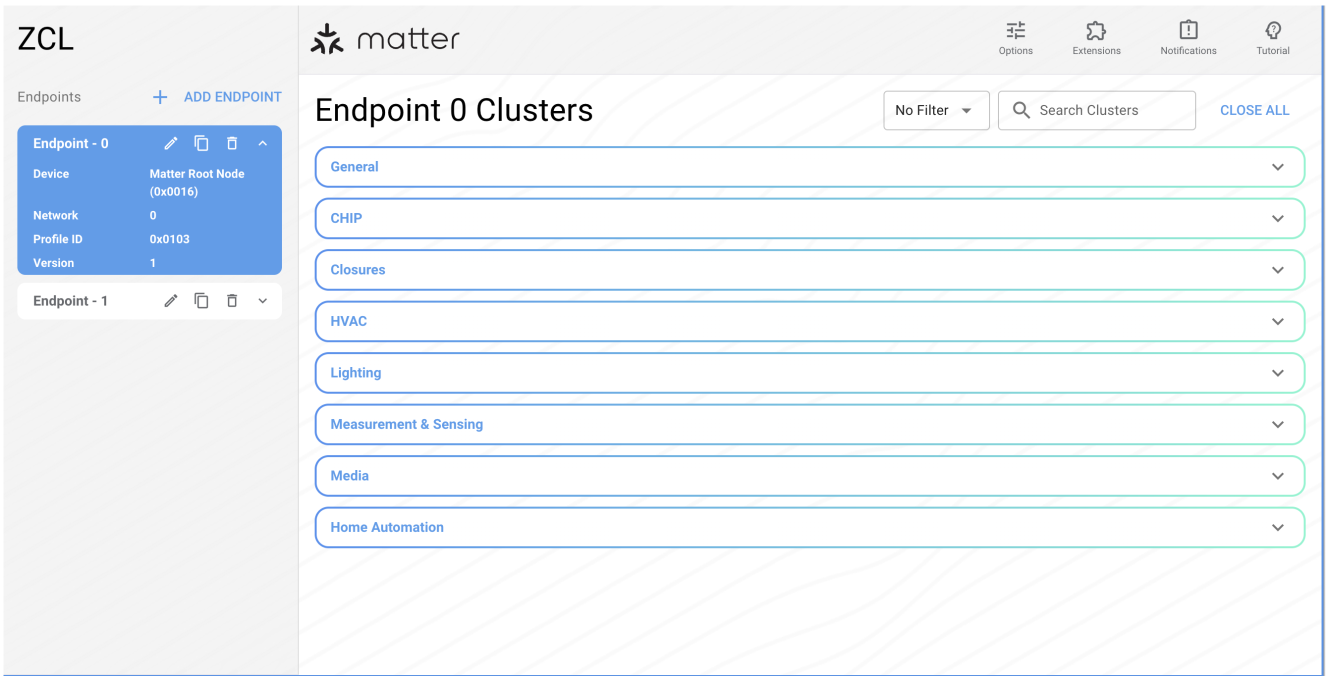
ZCL Advanced Platform (ZAP)

The ZCL Advanced Platform (ZAP) is a tool used to configure endpoints for Zigbee and Matter. When you open ZAP, it launches in a new tab next to the <project>.slcp tab. A Zigbee or Matter application can have multiple endpoints. Each endpoint contains a device configuration made up of Clusters on that end-point. The cluster groups differ depending on whether you re configuring a Zigbee or a Matter project, and the interface differs slightly.

ZAP for Zigbee:



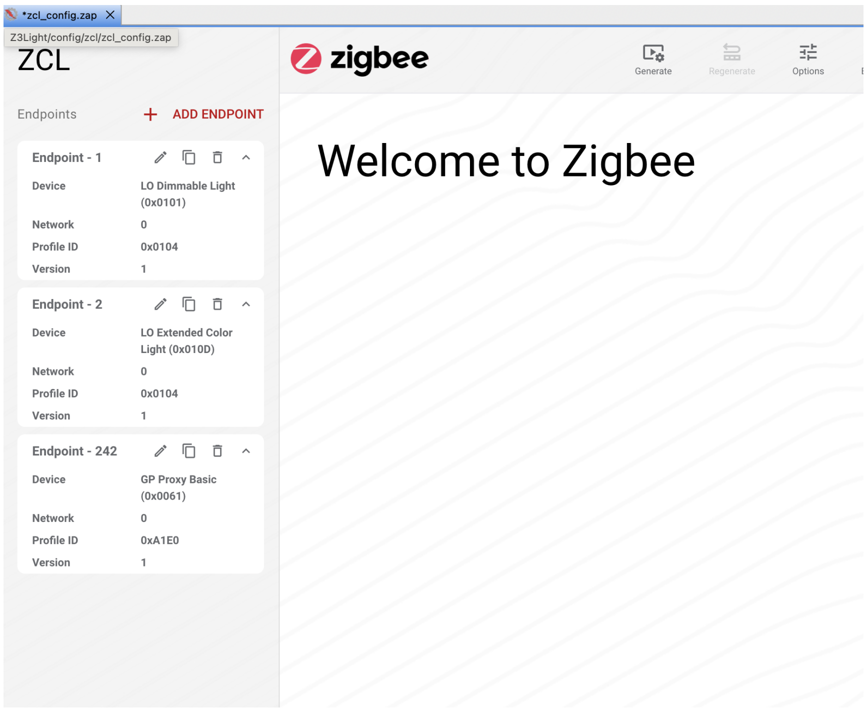
ZAP for Matter:



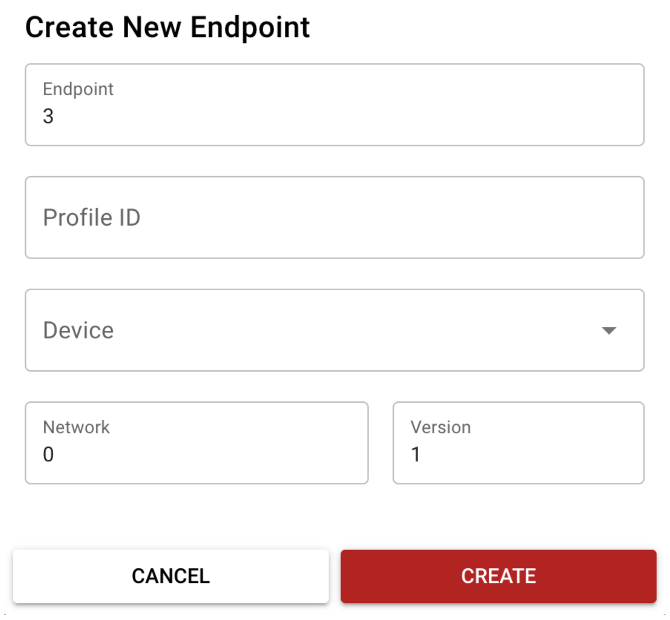
This section is illustrated with a Zigbee project, but the information applies to Matter as well. For more details about using ZAP with Matter, see Matter References: Using ZAP, the ZCL Advanced Platform.

At any point click Tutorial in the menu for an interactive tutorial.

Click **ADD ENDPOINT** to add a new endpoint.



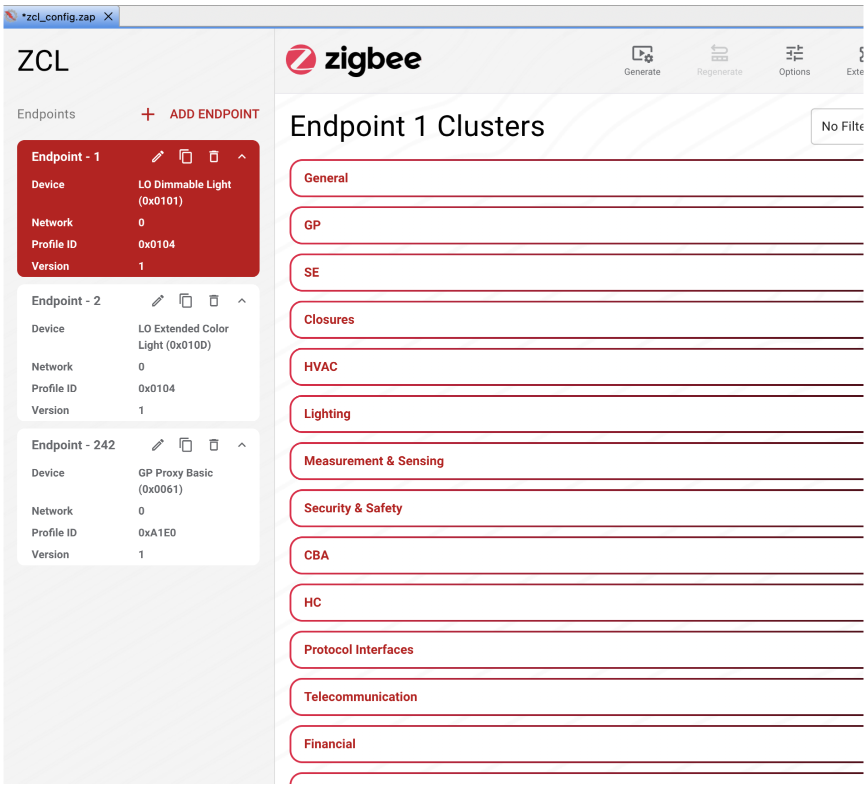
In the next dialog, select a device type for the endpoint, and optionally change the endpoint number. Click **CREATE** to create the endpoint.



Create New Endpoint

Endpoint	3
Profile ID	
Device	
Network	0
Version	1

To modify an endpoint, select it in the left panel.

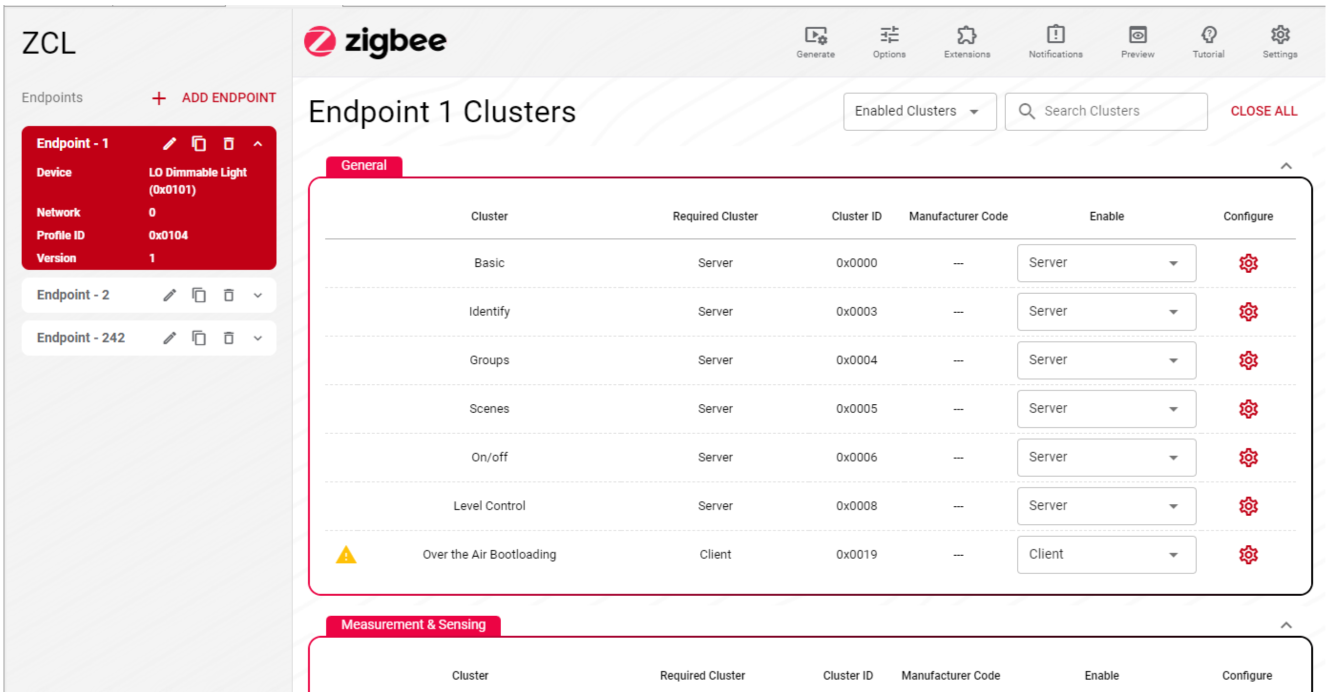


Endpoint 1 Clusters

Endpoint - 1	
Device	LO Dimmable Light (0x0101)
Network	0
Profile ID	0x0104
Version	1
Endpoint - 2	
Device	LO Extended Color Light (0x010D)
Network	0
Profile ID	0x0104
Version	1
Endpoint - 242	
Device	GP Proxy Basic (0x0061)
Network	0
Profile ID	0xA1E0
Version	1

- General
- GP
- SE
- Closures
- HVAC
- Lighting
- Measurement & Sensing
- Security & Safety
- CBA
- HC
- Protocol Interfaces
- Telecommunication
- Financial

The Filter dropdown in the upper right filters the clusters shown. To see only clusters that are enabled on the endpoint, select 'Enabled Clusters'.



Endpoint - 1

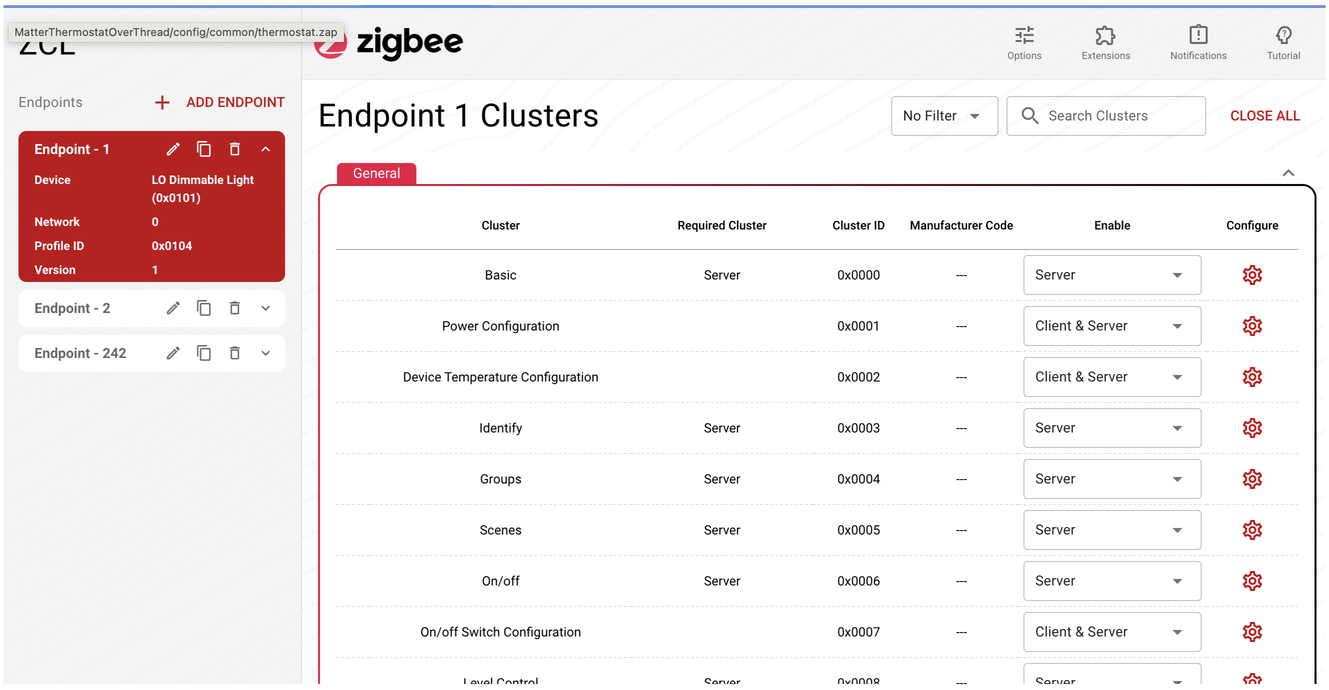
Device	LO Dimmable Light (0x0101)
Network	0
Profile ID	0x0104
Version	1

Endpoint 1 Clusters

Cluster	Required Cluster	Cluster ID	Manufacturer Code	Enable	Configure
Basic	Server	0x0000	---	Server	
Identify	Server	0x0003	---	Server	
Groups	Server	0x0004	---	Server	
Scenes	Server	0x0005	---	Server	
On/off	Server	0x0006	---	Server	
Level Control	Server	0x0008	---	Server	
⚠ Over the Air Bootloading	Client	0x0019	---	Client	

Cluster	Required Cluster	Cluster ID	Manufacturer Code	Enable	Configure

Settings in the Enable column enable either the Client or Server (or both) sides of a cluster. Depending on the changes you make, you may be notified that components have been added to your project. To remove the cluster entirely from the configuration, select 'Not Enabled'.



Endpoint - 1

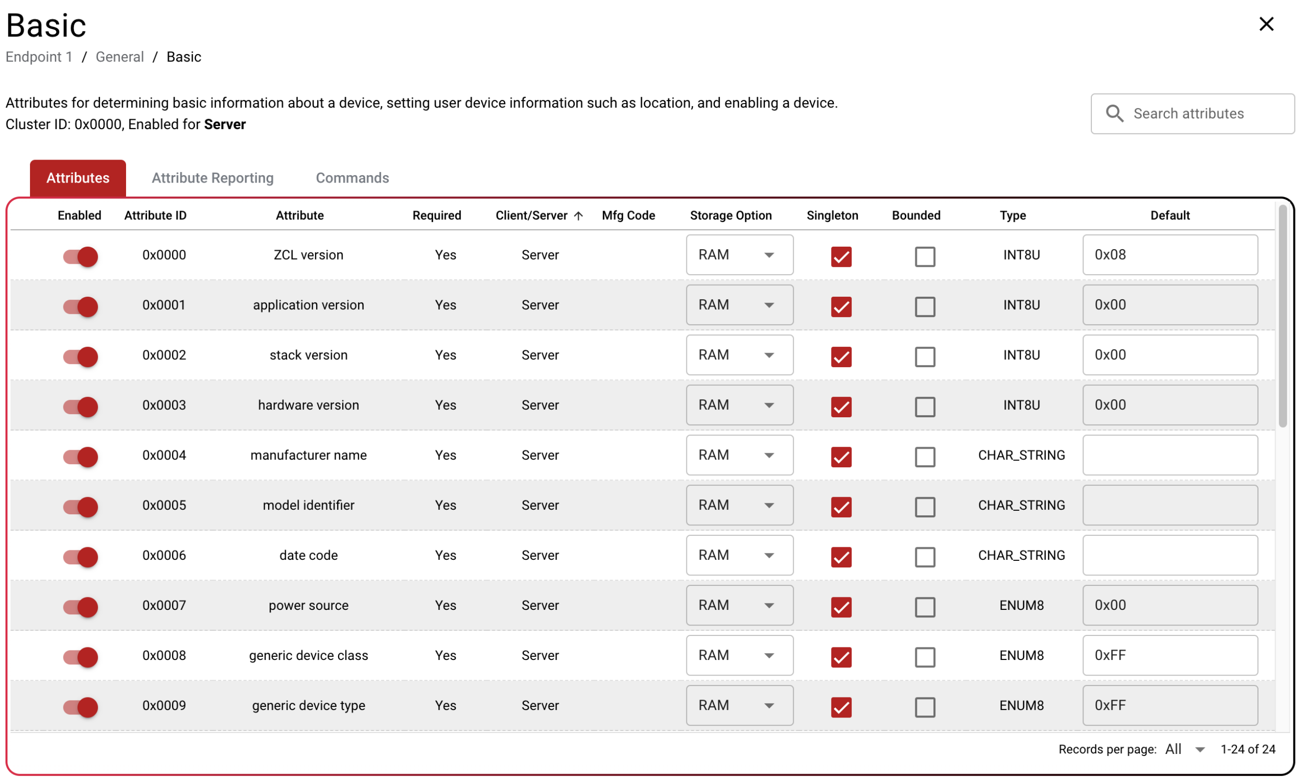
Device	LO Dimmable Light (0x0101)
Network	0
Profile ID	0x0104
Version	1

Endpoint 1 Clusters

Cluster	Required Cluster	Cluster ID	Manufacturer Code	Enable	Configure
Basic	Server	0x0000	---	Server	
Power Configuration		0x0001	---	Client & Server	
Device Temperature Configuration		0x0002	---	Client & Server	
Identify	Server	0x0003	---	Server	
Groups	Server	0x0004	---	Server	
Scenes	Server	0x0005	---	Server	
On/off	Server	0x0006	---	Server	
On/off Switch Configuration		0x0007	---	Client & Server	
Level Control	Server	0x0008	---	Server	

Click the Gear icon next to a cluster to enable or disable attributes, manage how they are stored, manage attribute reporting, and also manage the handling of commands on that cluster. The cluster configuration interface consists of three tabs:

- Attributes
- Attribute Reporting
- Commands



Basic

Endpoint 1 / General / Basic

Attributes for determining basic information about a device, setting user device information such as location, and enabling a device.
Cluster ID: 0x0000, Enabled for **Server**

Search attributes

Attributes | Attribute Reporting | Commands

Enabled	Attribute ID	Attribute	Required	Client/Server ↑	Mfg Code	Storage Option	Singleton	Bounded	Type	Default
●	0x0000	ZCL version	Yes	Server		RAM	✓	☐	INT8U	0x08
●	0x0001	application version	Yes	Server		RAM	✓	☐	INT8U	0x00
●	0x0002	stack version	Yes	Server		RAM	✓	☐	INT8U	0x00
●	0x0003	hardware version	Yes	Server		RAM	✓	☐	INT8U	0x00
●	0x0004	manufacturer name	Yes	Server		RAM	✓	☐	CHAR_STRING	
●	0x0005	model identifier	Yes	Server		RAM	✓	☐	CHAR_STRING	
●	0x0006	date code	Yes	Server		RAM	✓	☐	CHAR_STRING	
●	0x0007	power source	Yes	Server		RAM	✓	☐	ENUM8	0x00
●	0x0008	generic device class	Yes	Server		RAM	✓	☐	ENUM8	0xFF
●	0x0009	generic device type	Yes	Server		RAM	✓	☐	ENUM8	0xFF

Records per page: All 1-24 of 24

Configuration changes made through the Zigbee Cluster Configurator for a Zigbee project are saved to <project-folder>\config\zcl\zcl_config.zap. For a Matter project the configuration is saved to <project-folder>\config\zap\<name>.zap. The .zap file is the backing data file for the Zigbee Cluster Configurator configuration for your application. When you save the file, the Zigbee Cluster Configurator not only saves the .zap file into your project, but also automatically generates all the files required for your application. For Zigbee, the files are placed in the project's *autogen* folder and all start with the 'zap' prefix. For Matter, the files are placed in the project's *autogen > zap-generated* folder.

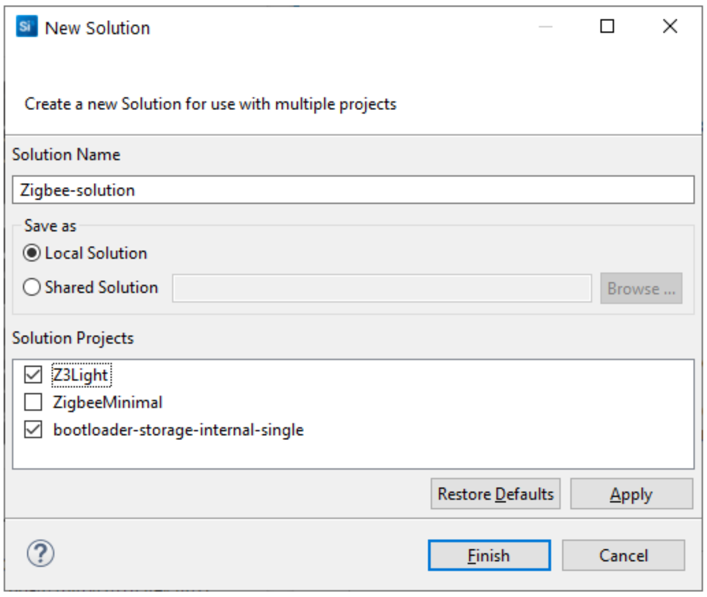
For additional details about cluster configuration and adding a custom cluster, see the Zigbee-focused *AN1325: Zigbee Cluster Configurator User's Guide.*

Solutions

Solutions

A solution is a construct that combines two or more projects such that they can be compiled, debugged, and flashed together. The solution is defined by a metadata .solutions file. Solutions can be local or shared. In a local solution, the .solutions file is stored in the Simplicity Studio Workspace. In a shared solution, the file is stored in the project specified when the shared solution is created. Shared solutions are easier to place under source control, and they can be exported to other Workspaces.

Solution examples are available on the Example Projects and Demos tab or you can create your own through **File > New > Solution** or **Projects > Manage Solutions > New**. The resulting dialog shows available projects in the current Workspace. Name the solution, select the projects to be included in it, and click **Finish**.


New Solution

Create a new Solution for use with multiple projects

Solution Name: Zigbee-solution

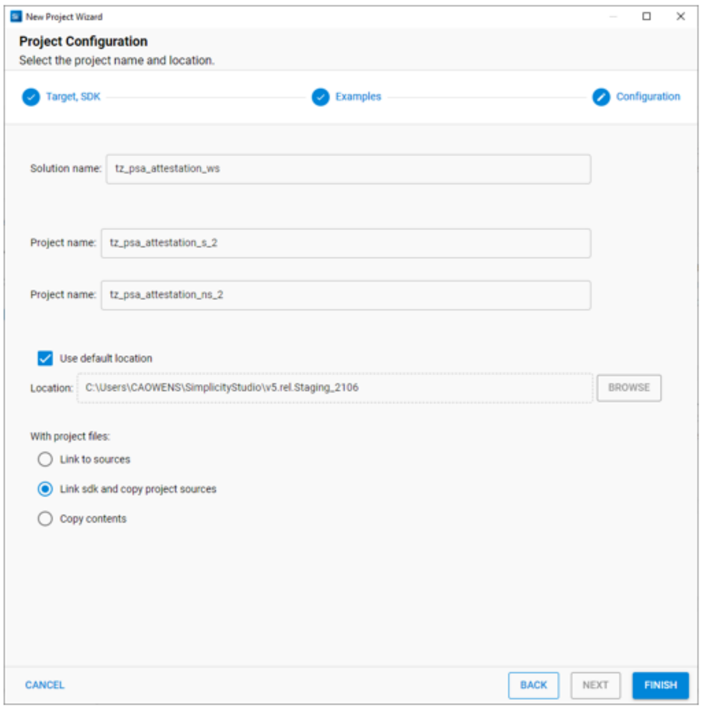
Save as: [✓ Local Solution] Shared Solution

Solution Projects:
- [✓] Z3Light
- [ ] ZigbeeMinimal
- [✓] bootloader-storage-internal-single

Note: The generic term for the construct that combines multiple projects is a workspace. However, in Simplicity Studio a Workspace is a special directory into which projects (and solutions) are created. Therefore, whenever possible, Simplicity Studio documentation uses the term 'solution' for this construct.

Create a Solution from an Example

When you create a solution based on an example, the configuration dialog allows you to edit the name of the projects in the solution as well as the solution.


New Project Wizard

Project Configuration

Select the project name and location.

Solution name: tz_psa_attestation_ws

Project name: tz_psa_attestation_s_2

Project name: tz_psa_attestation_ns_2

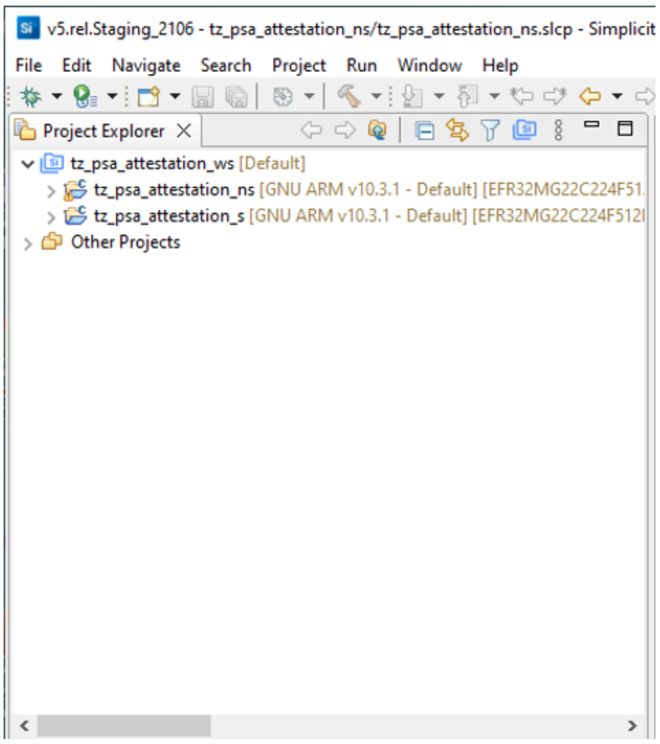
[✓] Use default location

Location: C:\Users\CAOWENS\SimplicityStudio\v5.rel.Staging_2106

With project files:

- Link to sources
- [✓ Link sdk and copy project sources]
- Copy contents

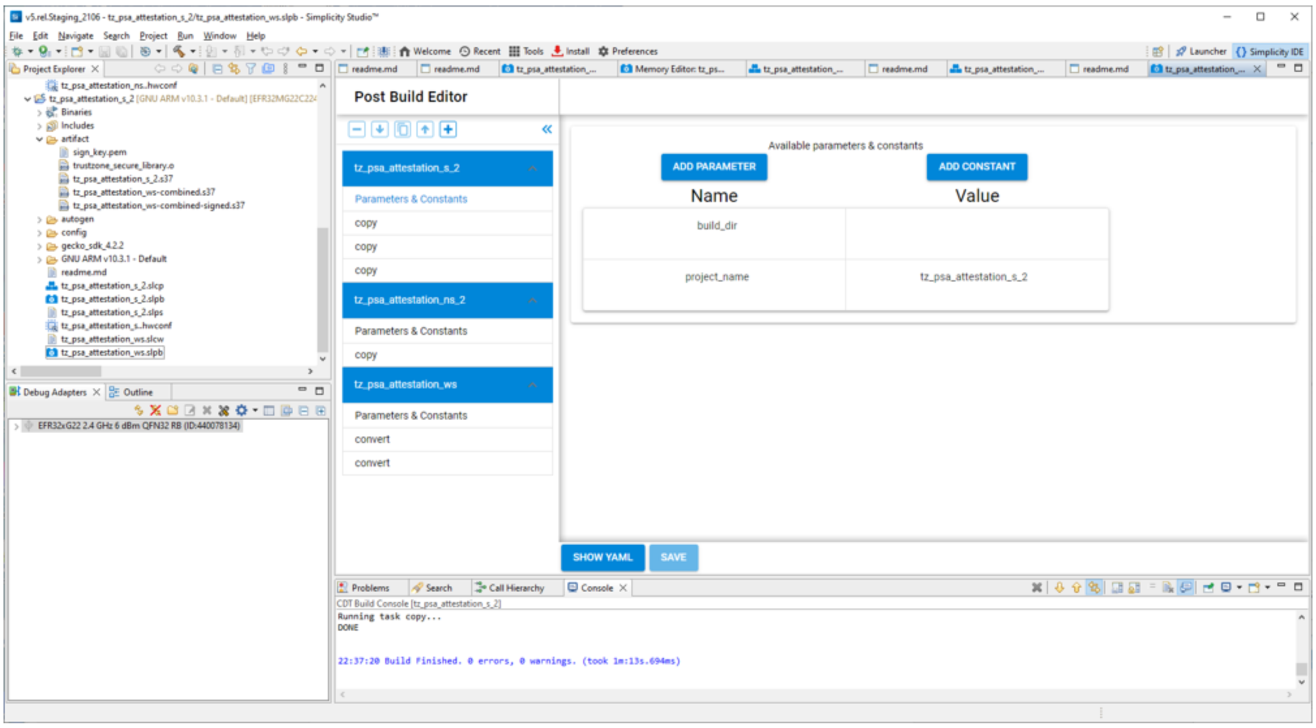
Each project in the solution then opens individually. Project Explorer view shows the solution as a separate folder, and projects in the Workspace folder that are not included in the solution as 'Other Projects'.


v5.rel.Staging_2106 - tz_psa_attestation_ns/tz_psa_attestation_ns.slcp - Simplicit

tz_psa_attestation_ws [Default]
- tz_psa_attestation_ns [GNU ARM v10.3.1 - Default] [EFR32MG22C224F51
- tz_psa_attestation_s [GNU ARM v10.3.1 - Default] [EFR32MG22C224F512I

Other Projects

Configure the projects in the solution as you would other project. Select the solution folder and click **Build** to build all projects in the solution. Project- and solution-level post-build script files (.slpb) are provided with the examples, and will combine the individual projects into a downloadable .s37 image. Double-click an *.slpb file to open the Post-Build Editor to see how the script is configured.



Name	Value
build_dir	
project_name	tz_psa_attestation_s_2

```
CDT Build Console [tz_psa_attestation_s_2]
Running task copy...
DONE


22:37:20 Build Finished. 0 errors, 0 warnings. (took 1m:13s.694ms)
```

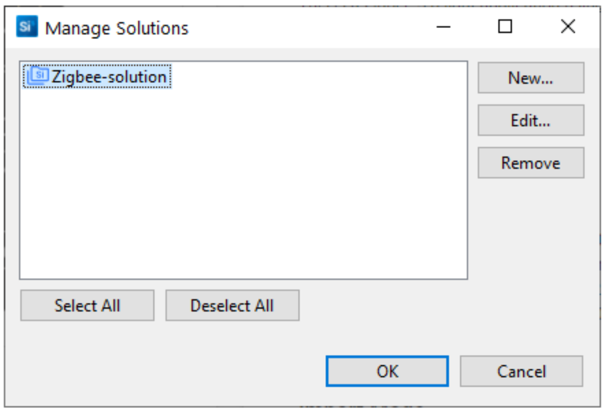
While the default memory use in the example solutions is fine for development and debugging, you can use Memory Editor to finalize the application memory layouts. Rebuild the solution to update the memory configurations.

Solutions are required for applications to take advantage of the EFR32 Series 2 TrustZone security feature. See AN1374: Series 2 TrustZone for a discussion of both the feature and how solutions are used to implement it.

Manage Solutions

Right-click the Solution in Project Explorer view, and click **Solutions** on the context menu to open the Manage Solutions dialog, set the active Configuration or sync solution configurations.

With **Manage Solutions** (also available on the Project menu) you can create, edit, and remove solutions.



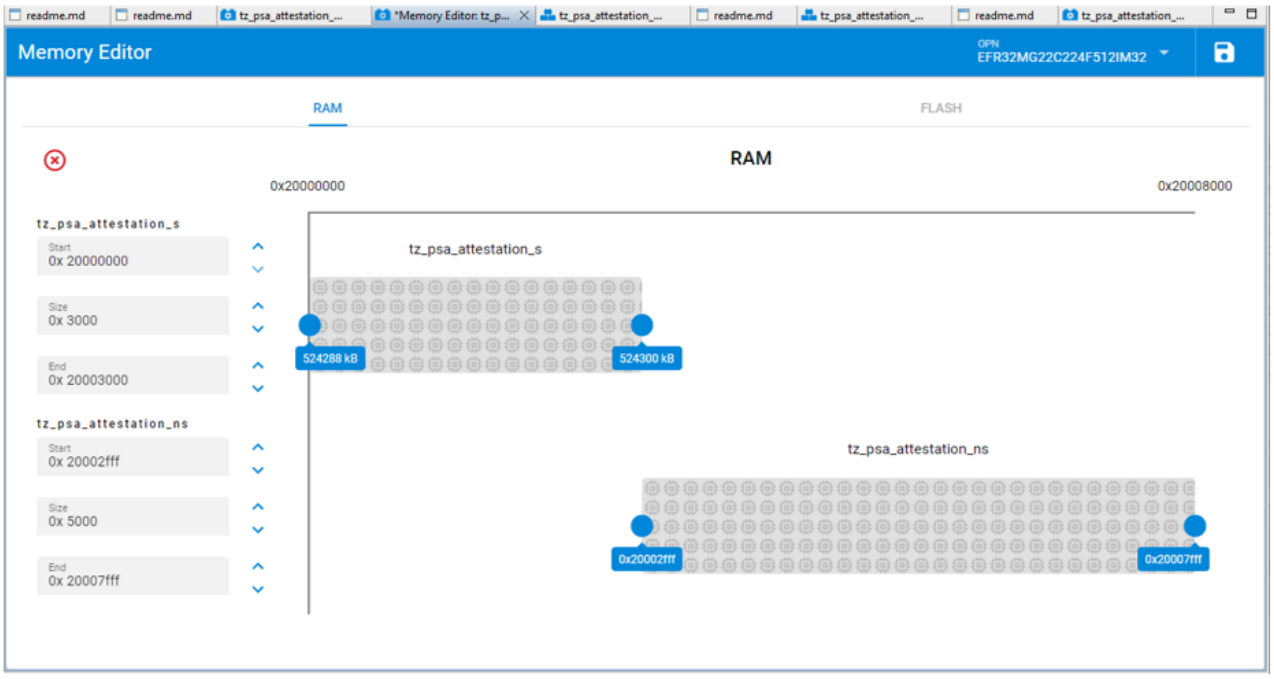
Click **Edit...** to rename a solution, add or remove projects from a solution, or save a local solution as a shared solution. You must specify the project to contain the solution metadata.

If the projects have multiple build configurations and the solution also has multiple build configurations, click **Sync Solution Configuration** to set the active build configuration of each project to match the solution's active build configuration.

Memory Editor

Memory Editor

The Memory Editor is a graphical tool for editing the flash and RAM memory layout of the applications in a solution. It can be opened from the Project Configurator Configuration Tools tab or the Quick Links card on the Overview tab. Memory Editor shows the Flash and RAM memory map for the projects in the solution. The default memory settings in the example solutions are good enough for software development and debugging, but with the Memory Editor you can fine-tune memory use based on your needs.



OPN
EFR32MG22C224F512IM32

tz_psa_attestation_s

Start
0x 20000000

Size
0x 3000

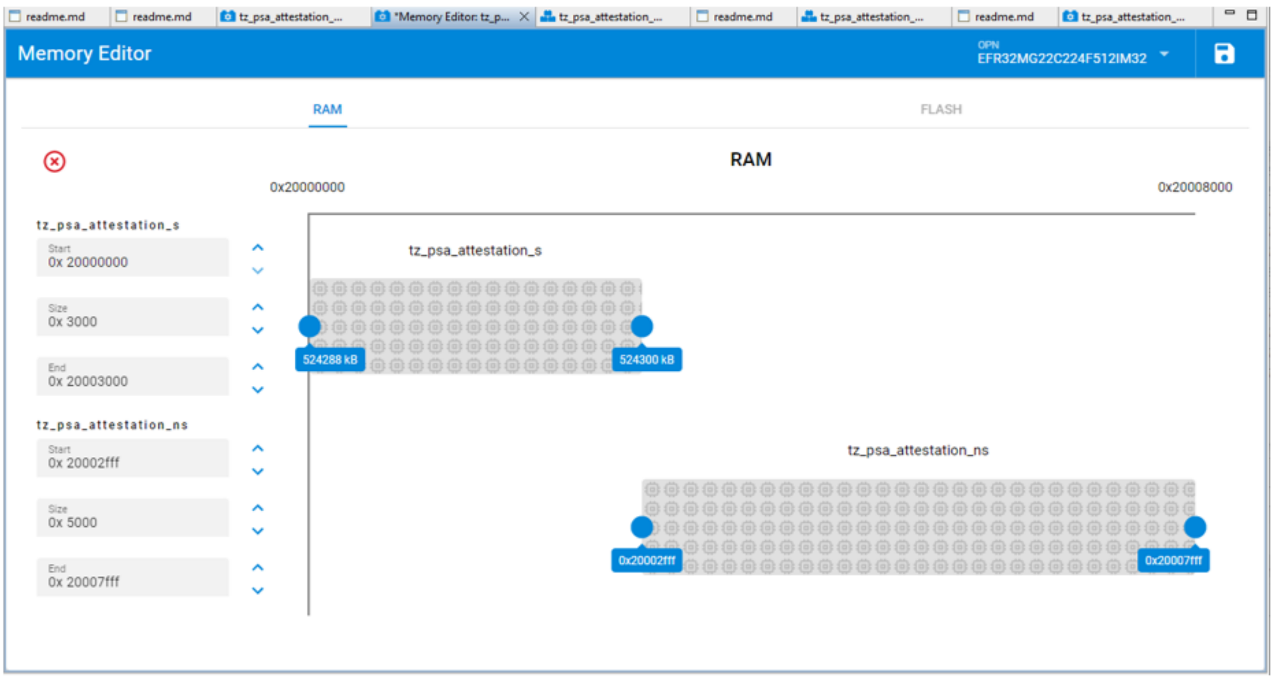
End
0x 20003000

tz_psa_attestation_ns

Start
0x 20002fff

Size
0x 5000

End
0x 20007fff



When you alter memory used by changing either or both applications' start or end location or size and save the changes, Memory Editor updates the linker file in the project autogen folder with the custom settings. Rebuild the projects to use the new memory configurations in the linker files.

Memory Editor displays a warning if your changes cause the applications' memory use to overlap.

This is a beta implementation. Not all example projects have default settings included at this time, which results in a warning when you open the editor. Save the default values that the tool calculates as your starting point.

Developing with AppBuilder

Developing with AppBuilder

Simplicity Studio® 5 (SSv5)'s AppBuilder is a graphical tool used to create configurations and build Application Framework files. The AppBuilder configuration files indicate which features and functions you would like for your compiled binary image. By using AppBuilder along with an application framework, you can quickly create an application that includes all of the required functionality for the image's purpose.

AppBuilder must be used in conjunction with one of the Silicon Labs application frameworks, which are shipped as part of the SDK. All of the code that will be compiled into the binary image is included in the SDK distribution. AppBuilder creates configuration and build files that tell the application framework which portions of the code to include in, and which to exclude from, the compiled binary image.

With the exception of a few header (.h) files, AppBuilder does not generate the C source code for the application. All of the source code that ultimately will be included in the binary image is provided in the Application Framework.

In general, to develop an application using AppBuilder:

1. Create an application based on one of the AppBuilder examples, such as Z3Switch.
2. Customize the application on the various AppBuilder tabs. Functions can be added by enabling and configuring plugins.
3. Click **Generate** to create application files.
4. Optionally, add your own code to the application.

Once you have finishing customizing your project, build and flash it, and then test and debug it.

Configuring a Project

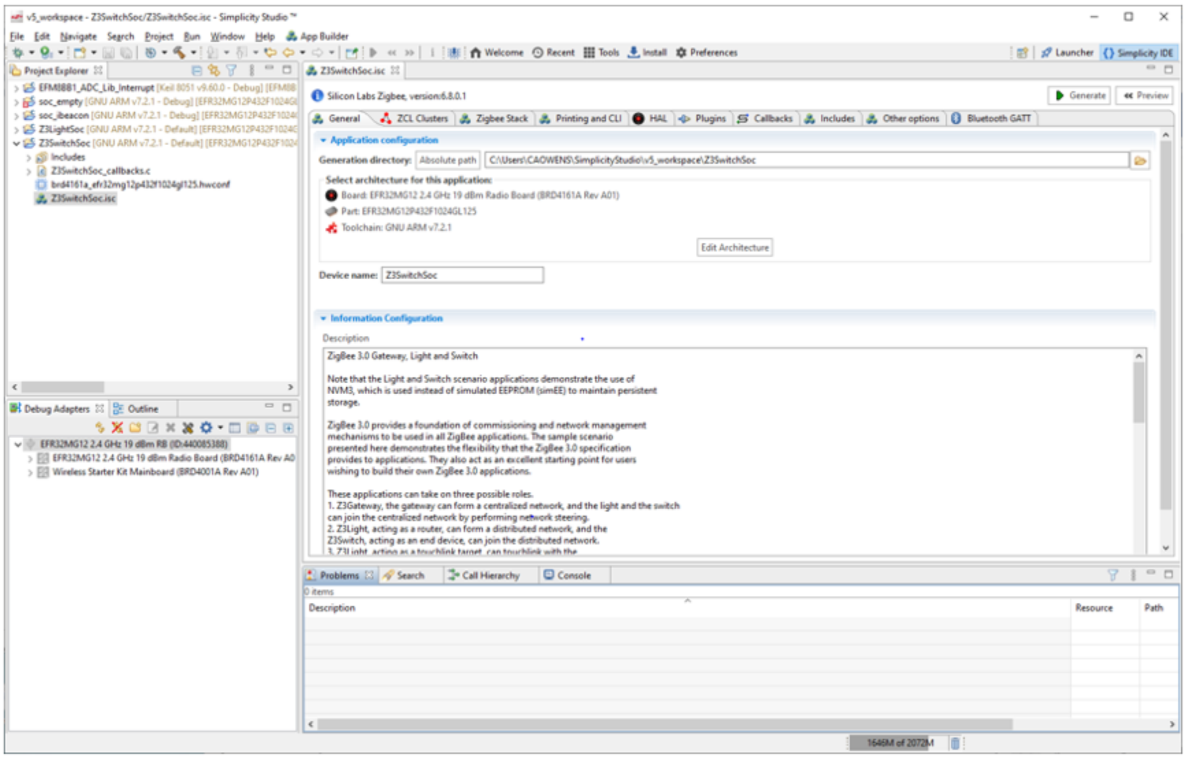
Configuring a Project

Simplicity Studio® 5 (SSv5)'s AppBuilder provides a tabbed interface. Each tab provides customization options.

- General - Configure global values for your application like the Device Name and Generation Directory
- ZCL Clusters - Choose the specific combination of clusters to include
- Zigbee Stack - Configure stack-specific settings
- Printing and CLI - Configure debug printing and CLI options
- HAL - Configure bootloader and board header file options, and access the Hardware Configurator tool to manage peripheral configurations
- Plugins - Include or exclude plugins, where a plugin is a specific implementation of a cluster or some other functionality
- Callbacks - Select from a set of predefined callbacks
- Includes - Include any external files into the generated project
- Other Options - Configure PHY support, plugin services, and other options
- Bluetooth GATT - For use with dynamic multiprotocol applications, provides a direct interface to the GATT configurator

General

The General tab provides information about the example used as a foundation for the project, and lets you change project level information, such as the toolchain used.


v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences Launcher Simplicity IDE
Project Explorer
Z3SwitchSoc.isc
Generate
Preview
General ZCL Clusters Zigbee Stack Printing and CLI HAL Plugins Callbacks Includes Other options Bluetooth GATT
Application configuration
Edit Architecture
Information Configuration
Debug Adapters
Outline
Problems Search Call Hierarchy Console
Resource
Path
1646M of 2072M

Silicon Labs Zigbee, version:6.8.0.1

Generation directory: Absolute path C:\Users\CAOWENS\SimplicityStudio\v5_workspace\Z3SwitchSoc

Select architecture for this application:
Board: EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)
Part: EFR32MG12P432F1024GL125
Toolchain: GNU ARM v7.2.1

Device name: Z3SwitchSoc

Description

ZigBee 3.0 Gateway, Light and Switch

Note that the Light and Switch scenario applications demonstrate the use of NVM3, which is used instead of simulated EEPROM (simEE) to maintain persistent storage.

ZigBee 3.0 provides a foundation of commissioning and network management mechanisms to be used in all ZigBee applications. The sample scenario presented here demonstrates the flexibility that the ZigBee 3.0 specification provides to applications. They also act as an excellent starting point for users wishing to build their own ZigBee 3.0 applications.

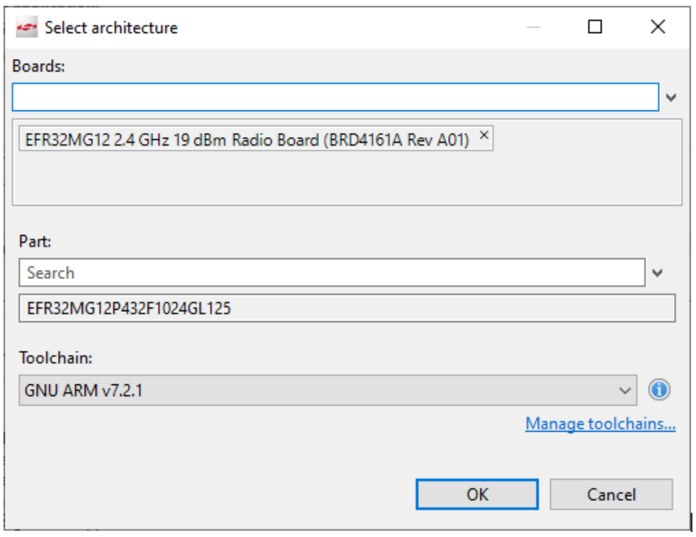
These applications can take on three possible roles.
1. Z3Gateway, the gateway can form a centralized network, and the light and the switch can join the centralized network by performing network steering.
2. Z3Light, acting as a router, can form a distributed network, and the Z3Switch, acting as an end device, can join the distributed network.
3. Z3Light, acting as a touchlink target, can touchlink with the

0 items

Description

Generation directory shows where project files will be generated. By default this is a <project name> directory in the default SSv5 workspace.

Click **Edit Architecture** to change the target device or the toolchain.



Select architecture

Boards:

EFR32MG12 2.4 GHz 19 dBm Radio Board (BRD4161A Rev A01)

Part:

Search

EFR32MG12P432F1024GL125

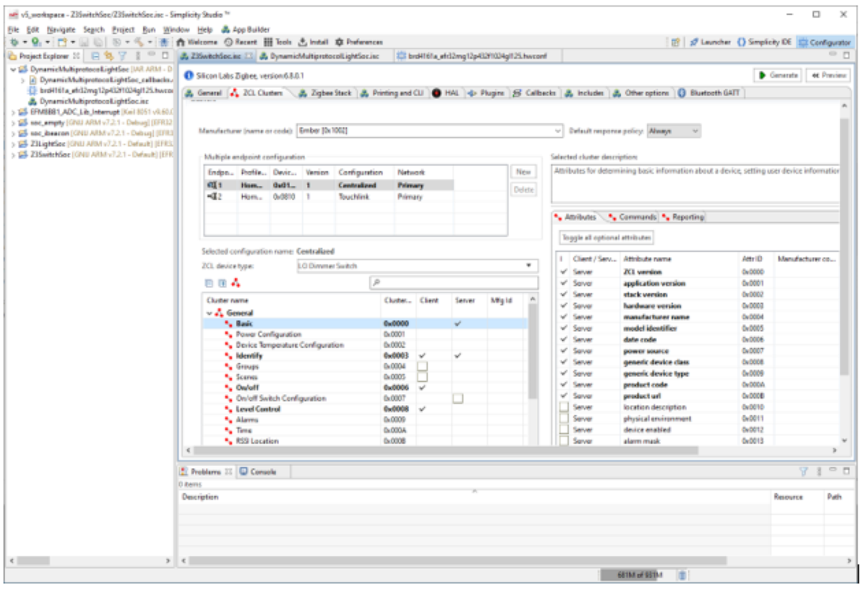
Toolchain:

GNU ARM v7.2.1

Manage toolchains...

ZCL Clusters

The ZCL Clusters tab provides functionality to manage endpoints and their cluster configuration.



Silicon Labs Zigbee, version 6.8.0.1

Manufacturer (name or code): Ember [0x1002]

Default response policy: Always

Multiple endpoint configuration

Endpo...	Profile...	Devic...	Version	Configuration	Network
1	Hom...	0x01...	1	Centralized	Primary
2	Hom...	0x0810	1	Touchlink	Primary

Selected configuration name: Centralized

ZCL device type: LO Dimmer Switch

Cluster name	Cluster...	Client	Server	Mfg Id
General				
Basic	0x0000		✓	
Power Configuration	0x0001			
Device Temperature Configuration	0x0002			
Identify	0x0003	✓	✓	
Groups	0x0004	☐		
Scenes	0x0005	☐		
On/off	0x0006	✓		
On/off Switch Configuration	0x0007		☐	
Level Control	0x0008	✓		
Alarms	0x0009			
Time	0x000A			
RSSI Location	0x000B			

Selected cluster description:

Attributes for determining basic information about a device, setting user device informatio

I	Client / Serv...	Attribute name	Attr ID	Manufacturer co...
✓	Server	ZCL version	0x0000	
✓	Server	application version	0x0001	
✓	Server	stack version	0x0002	
✓	Server	hardware version	0x0003	
✓	Server	manufacturer name	0x0004	
✓	Server	model identifier	0x0005	
✓	Server	date code	0x0006	
✓	Server	power source	0x0007	
✓	Server	generic device class	0x0008	
✓	Server	generic device type	0x0009	
✓	Server	product code	0x000A	
✓	Server	product url	0x000B	
☐	Server	location description	0x0010	
☐	Server	physical environment	0x0011	
☐	Server	device enabled	0x0012	
☐	Server	alarm mask	0x0013	

Multiple Endpoint Configuration Table

The Multiple Endpoint Configuration Table allows you to configure the endpoints on your device. Click **New** to add an endpoint.

Endpoint: Change the endpoint on which your device will be hosted by clicking the number in the Endpoint column. The number field becomes editable. Change the endpoint to any of the valid Zigbee endpoints available.

Configuration: Each endpoint implements a specific configuration. The configuration is not a Zigbee construct. It is a name created for Simplicity Studio AppBuilder so that multiple endpoints that implement the same application can share the metadata associated with that application. This conserves the use of flash within the device. By default, each new endpoint will implement the configuration named "Primary." Change the configuration by selecting the endpoint row and then clicking on the "Configuration" column. A button appears to the right of the Configuration name. Click the button to launch the configuration dialog. Once you have created a new configuration, the application for that configuration can be implemented on as many endpoints as you like without using any more memory on your device for metadata storage. Each endpoint still needs to have its own attribute storage to contain the endpoint-specific values. AppBuilder and the application framework take care of this for you.

Network: By default, the first endpoint belongs to the network named "Primary." When new endpoints are created, they will belong to the same network as the first endpoint. On platforms that support multi-network, you may choose the network for a specific endpoint. New networks can be created and configured in the Zigbee Stack tab.

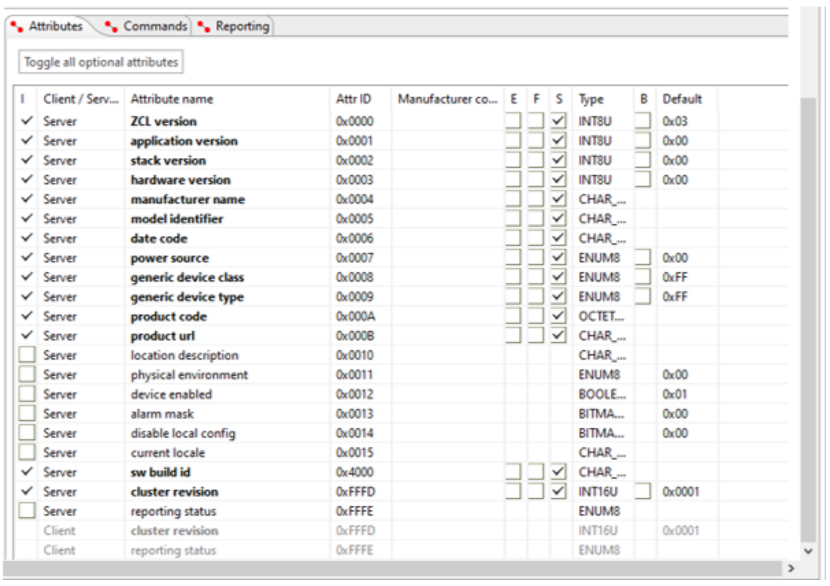
ZCL Device Type: Select from a set of pre-defined ZCL Device Types. AppBuilder then populates the cluster table with the clusters appropriate to that device type as defined by the Zigbee Cluster Library specification. In addition to these pre-defined device types, you also have the option to create a custom device type. With this option, you can choose from any of the available Zigbee clusters included within the Application Framework.

Cluster Table

The Cluster Table displays which clusters will be included as part of each endpoint type.In the case of a custom device implemented on any endpoint, you have the option to choose the clusters to include and their client/server role.

Attribute Table

When you click a specific cluster, its attributes display in the Attribute Table. Configure the device's behavior by configuring the attributes in this table.



I	Client / Serv...	Attribute name	Attr ID	Manufacturer co...	E	F	S	Type	B	Default
✓	Server	**ZCL version**	0x0000		☐	☐	✓	INT8U	☐	0x03
✓	Server	**application version**	0x0001		☐	☐	✓	INT8U	☐	0x00
✓	Server	**stack version**	0x0002		☐	☐	✓	INT8U	☐	0x00
✓	Server	**hardware version**	0x0003		☐	☐	✓	INT8U	☐	0x00
✓	Server	**manufacturer name**	0x0004		☐	☐	✓	CHAR_...		
✓	Server	**model identifier**	0x0005		☐	☐	✓	CHAR_...		
✓	Server	**date code**	0x0006		☐	☐	✓	CHAR_...		
✓	Server	**power source**	0x0007		☐	☐	✓	ENUM8	☐	0x00
✓	Server	**generic device class**	0x0008		☐	☐	✓	ENUM8	☐	0xFF
✓	Server	**generic device type**	0x0009		☐	☐	✓	ENUM8	☐	0xFF
✓	Server	**product code**	0x000A		☐	☐	✓	OCTET...		
✓	Server	**product url**	0x000B		☐	☐	✓	CHAR_...		
☐	Server	location description	0x0010					CHAR_...		
☐	Server	physical environment	0x0011					ENUM8		0x00
☐	Server	device enabled	0x0012					BOOLE...		0x01
☐	Server	alarm mask	0x0013					BITMA...		0x00
☐	Server	disable local config	0x0014					BITMA...		0x00
☐	Server	current locale	0x0015					CHAR_...		
✓	Server	**sw build id**	0x4000		☐	☐	✓	CHAR_...		
✓	Server	**cluster revision**	0xFFFD		☐	☐	✓	INT16U	☐	0x0001
☐	Server	reporting status	0xFFFE					ENUM8		
	Client	cluster revision	0xFFFD					INT16U		0x0001
	Client	reporting status	0xFFFE					ENUM8		

External Attribute storage: By default, all attributes are stored in a RAM buffer provided by the application framework. Storing some attributes may not make sense, however, since they are either already stored somewhere else in the system or are not stored at all but read from some external piece of hardware. In this case you may indicate that the attribute is externally stored. When the application framework needs to interact with the attribute it will use the external attribute callbacks. By implementing these callbacks and marking the attribute as externally stored, the attribute storage is removed from the framework and placed into the domain of the application. The metadata used to represent the attribute on the network is still stored inside the framework so that the framework can respond to ZDO requests and so on.

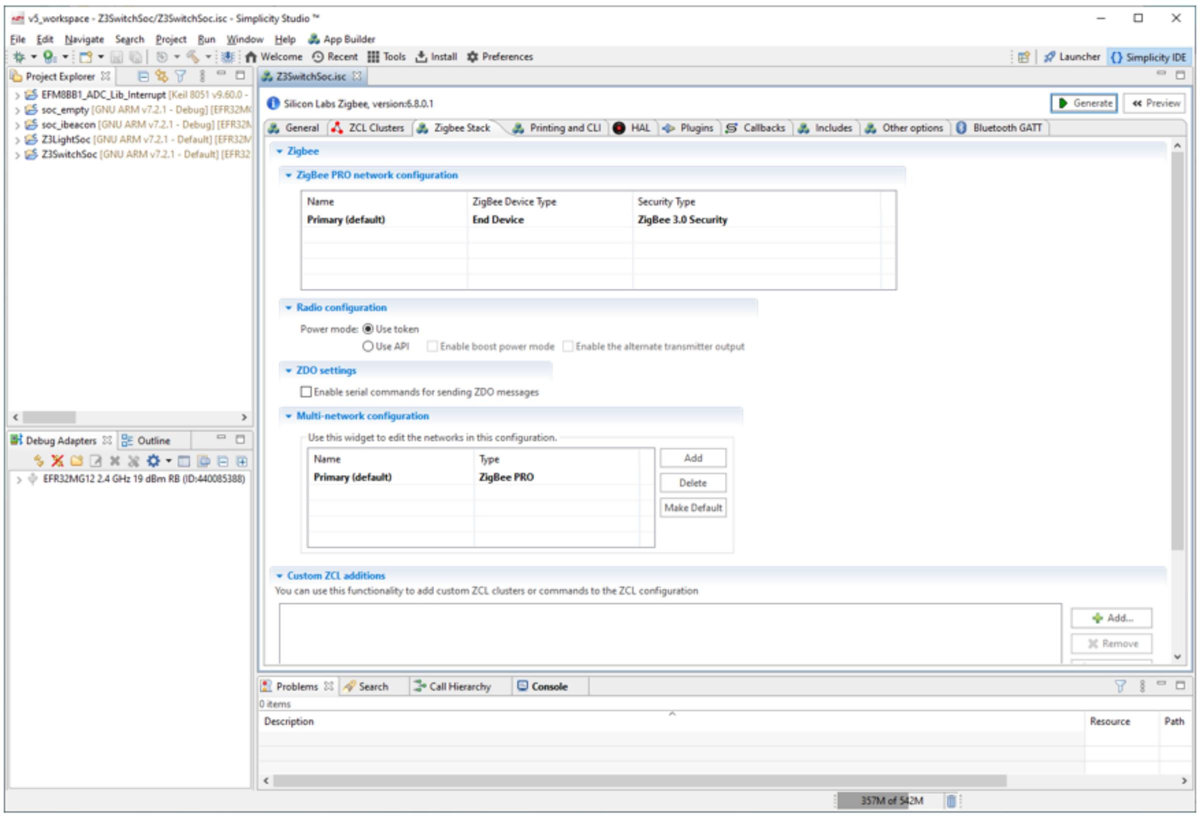
Persistent attribute storage: By default, all attributes are stored in RAM. As a result, attribute data does not survive a reboot of the device. If you want to maintain state for an attribute across reboots, the attribute must be stored in persistent memory on the device. To store a specific attribute in persistent memory, check the checkbox in the F column.

Singleton attribute storage: Most attributes are stored as a separate value for each endpoint on which they are implemented. This does not make sense for some attributes that may exist across endpoints. For instance, it makes little sense to store multiple copies of the Basic Cluster's ZCL Version since the version pertains to the entire application and not to each individual endpoint. Attribute values may be shared across endpoints by indicating that the attribute is a "singleton."

Bounding attributes to their Min and Max values: By default, the application framework allows you to write any value (within the limits of the size of the attribute) into the attribute table. However, you may request that the application framework reject any attribute value outside the Min and Max defined by the Zigbee specification by checking the checkbox in the B column. The application framework then stores the min and max values for the chosen attribute in the attribute table and rejects any write request, whether from the CLI or an external device, that is outside the attribute value range.

Zigbee Stack

Here you can configure networks and add custom clusters.


Silicon Labs Zigbee, version:6.8.0.1

Zigbee

ZigBee PRO network configuration

Name	ZigBee Device Type	Security Type
Primary (default)	End Device	ZigBee 3.0 Security

Radio configuration

Power mode: [✓ Use token] Use API; Enable boost power mode ☐; Enable the alternate transmitter output ☐

ZDO settings

☐ Enable serial commands for sending ZDO messages

Multi-network configuration

Use this widget to edit the networks in this configuration.

Name	Type
Primary (default)	ZigBee PRO

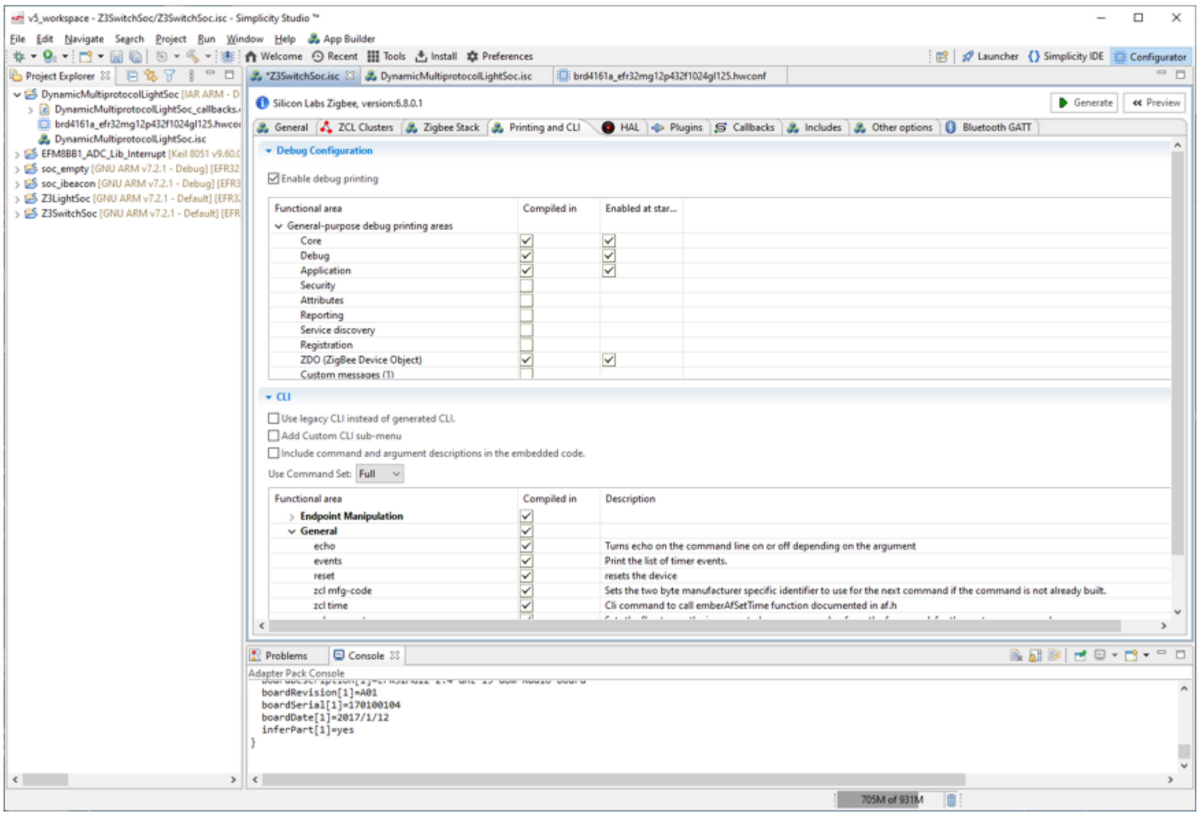
Custom ZCL additions

You can use this functionality to add custom ZCL clusters or commands to the ZCL configuration

v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences Launcher Simplicity IDE
Project Explorer
Z3SwitchSoc.isc
Generate
Preview
General ZCL Clusters Zigbee Stack Printing and CLI HAL Plugins Callbacks Includes Other options Bluetooth GATT
Add Delete Make Default
Add... Remove
Debug Adapters Outline
Problems Search Call Hierarchy Console
0 items
Description Resource Path
357M of 542M

Printing and CLI

Here you can enable and configure debug printing, and enable/disable CLI functionality.


v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE Configurator
Project Explorer
*Z3SwitchSoc.isc
DynamicMultiprotocolLightSoc.isc
brd4161a_efr32mg12p432f1024gl125.hwconf
Generate
Preview
General ZCL Clusters Zigbee Stack Printing and CLI HAL Plugins Callbacks Includes Other options Bluetooth GATT
Problems Console
705M of 931M

Silicon Labs Zigbee, version:6.8.0.1

Debug Configuration

☑ Enable debug printing

Functional area	Compiled in	Enabled at star...
General-purpose debug printing areas		
Core	✓	✓
Debug	✓	✓
Application	✓	✓
Security	☐	
Attributes	☐	
Reporting	☐	
Service discovery	☐	
Registration	☐	
ZDO (ZigBee Device Object)	✓	✓
Custom messages (1)	☐	

CLI

☐ Use legacy CLI instead of generated CLI.
☐ Add Custom CLI sub-menu
☐ Include command and argument descriptions in the embedded code.

Use Command Set: Full

Functional area	Compiled in	Description
Endpoint Manipulation	✓	
General	✓	
echo	✓	Turns echo on the command line on or off depending on the argument
events	✓	Print the list of timer events.
reset	✓	resets the device
zcl mfg-code	✓	Sets the two byte manufacturer specific identifier to use for the next command if the command is not already built.
zcl time	✓	Cli command to call emberAfSetTime function documented in af.h

Adapter Pack Console

```
  boardRevision[1]=A01
  boardSerial[1]=170100104
  boardDate[1]=2017/1/12
  inferPart[1]=yes
}
```

Debug printing

The application framework can output application and stack debug information over the serial port. To choose the information to include in the debug output, select from the serial printing checkboxes. If you are not concerned with debug output, and/or would prefer to conserve the flash and RAM associated with debugging, turn off the serial printing options by deselecting them.

Command Line Interface (CLI) Options

AppBuilder can output either a "Legacy" or "Generated" command line interface.

Use Legacy CLI: If checked, AppBuilder uses the hard coded CLI interface included in the application framework. Otherwise AppBuilder generates a Command Line Interface along with other generated files based on the CLI interface options selected in the CLI configuration window.

Add Custom CLI Sub-menu: Adds the ability to include user-defined CLI commands into the Command Line Interface.

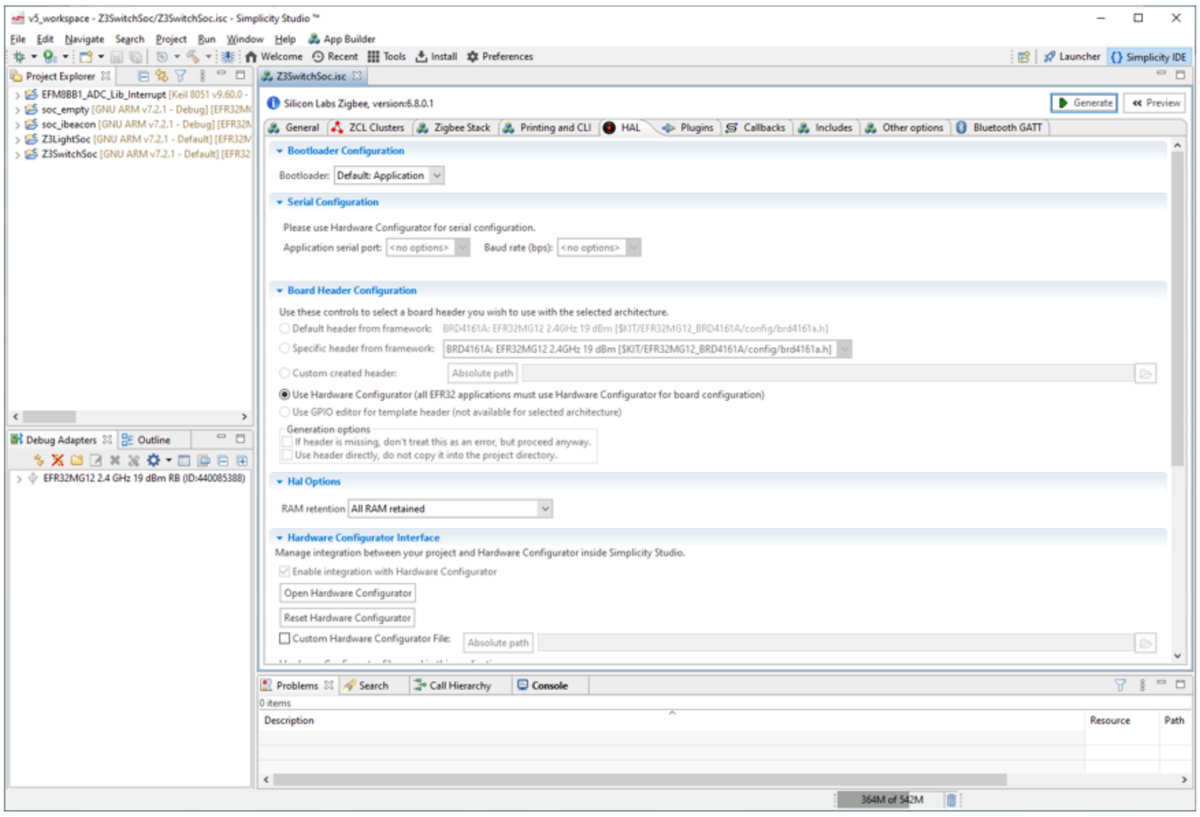
Include Command and ...: Includes a description for each command or argument into the Command Line Interface. Turning this on will increase the size of your application but will make it easier to use the Command Line Interface since all commands will be self-described when a command is not recognized by the application.

Use Command Set: Makes it possible to turn the command line interface on and off and choose a pre-defined subset of the CLI for use in your application

CLI Configuration Table: Select exactly what CLI commands are supported in the application. By default, the Full command line interface command set selects all General commands and all cluster and plugin commands supported by clusters and plugins included in the application.

HAL

Use the HAL tab to configure hardware-specific options for your device.


v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
Z3SwitchSoc.isc
Generate
Preview
General ZCL Clusters Zigbee Stack Printing and CLI HAL Plugins Callbacks Includes Other options Bluetooth GATT
Debug Adapters Outline
Problems Search Call Hierarchy Console
364M of 542M

Silicon Labs Zigbee, version:6.8.0.1

Bootloader Configuration

Bootloader: Default: Application

Serial Configuration

Please use Hardware Configurator for serial configuration.

Application serial port: <no options> Baud rate (bps): <no options>

Board Header Configuration

Use these controls to select a board header you wish to use with the selected architecture.

- Default header from framework: BRD4161A: EFR32MG12 2.4GHz 19 dBm [$KIT/EFR32MG12_BRD4161A/config/brd4161a.h]
- Specific header from framework: BRD4161A: EFR32MG12 2.4GHz 19 dBm [$KIT/EFR32MG12_BRD4161A/config/brd4161a.h]
- Custom created header: Absolute path
- [✓ Use Hardware Configurator (all EFR32 applications must use Hardware Configurator for board configuration)]
- Use GPIO editor for template header (not available for selected architecture)

Generation options
- If header is missing, don't treat this as an error, but proceed anyway.
- Use header directly, do not copy it into the project directory.

Hal Options

RAM retention All RAM retained

Hardware Configurator Interface

Manage integration between your project and Hardware Configurator inside Simplicity Studio.

- [✓] Enable integration with Hardware Configurator

Open Hardware Configurator

Reset Hardware Configurator

- Custom Hardware Configurator File: Absolute path

0 items

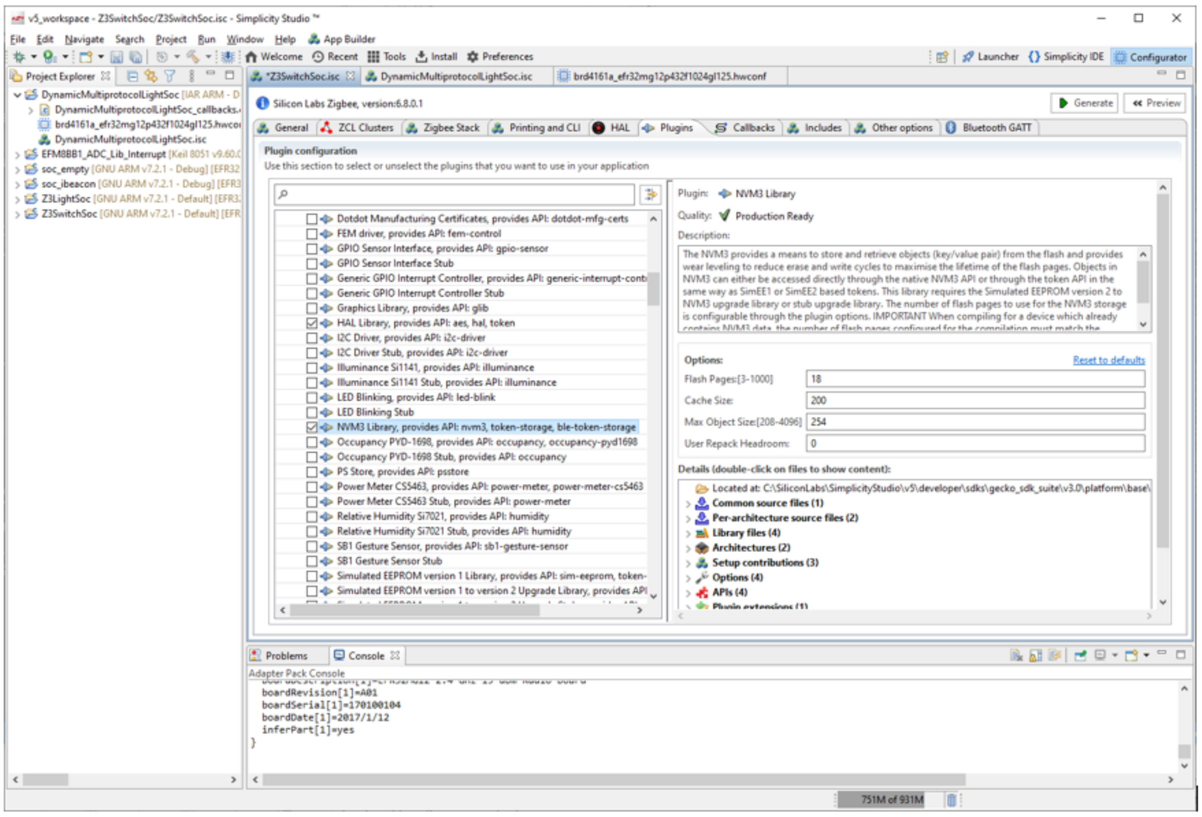
Description	Resource	Path

For EFR32 devices, most hardware configuration is done through the Hardware Configurator. Under **Hardware Configurator Interface** click **Open Hardware Configurator**.

Bootloader - Select the type of bootloader to use. See UG103.6: Bootloader Fundamentals as a starting point to learn more about bootloaders.

Plugins

Use the Plugins tab to include, exclude, and configure the functional implementations called plugins.


v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE Configurator
Project Explorer
*Z3SwitchSoc.isc
DynamicMultiprotocolLightSoc.isc
brd4161a_efr32mg12p432f1024gl125.hwconf
Generate
Preview
General ZCL Clusters Zigbee Stack Printing and CLI HAL Plugins Callbacks Includes Other options Bluetooth GATT
Problems Console
751M of 931M

Silicon Labs Zigbee, version:6.8.0.1

Plugin configuration

Use this section to select or unselect the plugins that you want to use in your application

- [ ] Dotdot Manufacturing Certificates, provides API: dotdot-mfg-certs
- [ ] FEM driver, provides API: fem-control
- [ ] GPIO Sensor Interface, provides API: gpio-sensor
- [ ] GPIO Sensor Interface Stub
- [ ] Generic GPIO Interrupt Controller, provides API: generic-interrupt-cont
- [ ] Generic GPIO Interrupt Controller Stub
- [ ] Graphics Library, provides API: glib
- [x] HAL Library, provides API: aes, hal, token
- [ ] I2C Driver, provides API: i2c-driver
- [ ] I2C Driver Stub, provides API: i2c-driver
- [ ] Illuminance Si1141, provides API: illuminance
- [ ] Illuminance Si1141 Stub, provides API: illuminance
- [ ] LED Blinking, provides API: led-blink
- [ ] LED Blinking Stub
- [x] NVM3 Library, provides API: nvm3, token-storage, ble-token-storage
- [ ] Occupancy PYD-1698, provides API: occupancy, occupancy-pyd1698
- [ ] Occupancy PYD-1698 Stub, provides API: occupancy
- [ ] PS Store, provides API: psstore
- [ ] Power Meter CS5463, provides API: power-meter, power-meter-cs5463
- [ ] Power Meter CS5463 Stub, provides API: power-meter
- [ ] Relative Humidity Si7021, provides API: humidity
- [ ] Relative Humidity Si7021 Stub, provides API: humidity
- [ ] SB1 Gesture Sensor, provides API: sb1-gesture-sensor
- [ ] SB1 Gesture Sensor Stub
- [ ] Simulated EEPROM version 1 Library, provides API: sim-eeprom, token-
- [ ] Simulated EEPROM version 1 to version 2 Upgrade Library, provides API

Plugin: NVM3 Library

Quality: ✓ Production Ready

Description:

The NVM3 provides a means to store and retrieve objects (key/value pair) from the flash and provides wear leveling to reduce erase and write cycles to maximise the lifetime of the flash pages. Objects in NVM3 can either be accessed directly through the native NVM3 API or through the token API in the same way as SimEE1 or SimEE2 based tokens. This library requires the Simulated EEPROM version 2 to NVM3 upgrade library or stub upgrade library. The number of flash pages to use for the NVM3 storage is configurable through the plugin options. IMPORTANT When compiling for a device which already

Options: Reset to defaults

Flash Pages:[3-1000]	18
Cache Size:	200
Max Object Size:[208-4096]	254
User Repack Headroom:	0

Details (double-click on files to show content):

- Located at: C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\platform\base\
- Common source files (1)
- Per-architecture source files (2)
- Library files (4)
- Architectures (2)
- Setup contributions (3)
- Options (4)
- APIs (4)
- Plugin extensions (1)

Adapter Pack Console

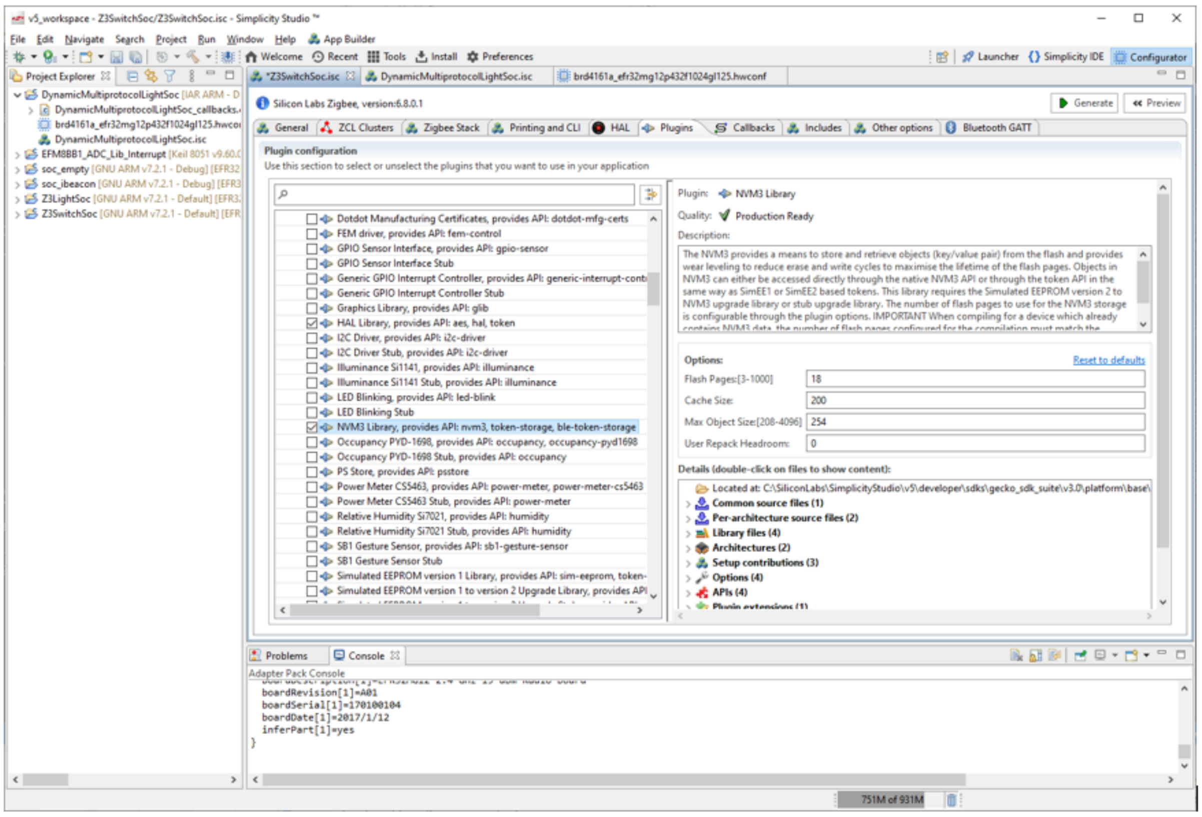
```
  boardRevision[1]=A01
  boardSerial[1]=170100104
  boardDate[1]=2017/1/12
  inferPart[1]=yes
}
```

Include/exclude plugins by checking/unchecking them. Select a plugin to see information about it and configurable parameters, if any.

Some plugins provide access to the Hardware Configurator peripheral settings. These plugins have a Hardware Configurator Dependent Options pane. You may need to scroll down in that pane to see all options. Enabling/disabling plugins can also enable and disable Hardware Configurator modules and their dependent peripherals. When a plugin with Hardware Configurator dependencies is enabled/disabled, a pop-up dialog box shows which Hardware Configurator modules and peripherals are being enabled/disabled. Also, when a plugin that shares Hardware Configurator modules/peripherals with another plugin is enabled/disabled, a similar warning shows which peripherals are being affected behind the scenes in the Hardware Configurator. See Configuring Peripherals for information on using the Hardware Configurator directly. For more details on both plugin and tool access, see AN1115: Configuring Peripherals for 32-Bit Devices using Hardware Configurator.

Callbacks

Use the Callbacks tab to add or remove callbacks, and to see the callbacks that are required by plugins or other code. Select a callback to see its code and other information about it.



Silicon Labs Zigbee, version:6.8.0.1

Plugin configuration

Use this section to select or unselect the plugins that you want to use in your application

- [ ] Dotdot Manufacturing Certificates, provides API: dotdot-mfg-certs
- [ ] FEM driver, provides API: fem-control
- [ ] GPIO Sensor Interface, provides API: gpio-sensor
- [ ] GPIO Sensor Interface Stub
- [ ] Generic GPIO Interrupt Controller, provides API: generic-interrupt-cont
- [ ] Generic GPIO Interrupt Controller Stub
- [ ] Graphics Library, provides API: glib
- [x] HAL Library, provides API: aes, hal, token
- [ ] I2C Driver, provides API: i2c-driver
- [ ] I2C Driver Stub, provides API: i2c-driver
- [ ] Illuminance Si1141, provides API: illuminance
- [ ] Illuminance Si1141 Stub, provides API: illuminance
- [ ] LED Blinking, provides API: led-blink
- [ ] LED Blinking Stub
- [x] NVM3 Library, provides API: nvm3, token-storage, ble-token-storage
- [ ] Occupancy PYD-1698, provides API: occupancy, occupancy-pyd1698
- [ ] Occupancy PYD-1698 Stub, provides API: occupancy
- [ ] PS Store, provides API: psstore
- [ ] Power Meter CS5463, provides API: power-meter, power-meter-cs5463
- [ ] Power Meter CS5463 Stub, provides API: power-meter
- [ ] Relative Humidity Si7021, provides API: humidity
- [ ] Relative Humidity Si7021 Stub, provides API: humidity
- [ ] SB1 Gesture Sensor, provides API: sb1-gesture-sensor
- [ ] SB1 Gesture Sensor Stub
- [ ] Simulated EEPROM version 1 Library, provides API: sim-eeprom, token-
- [ ] Simulated EEPROM version 1 to version 2 Upgrade Library, provides API

Plugin: NVM3 Library

Quality: Production Ready

Description:

The NVM3 provides a means to store and retrieve objects (key/value pair) from the flash and provides wear leveling to reduce erase and write cycles to maximise the lifetime of the flash pages. Objects in NVM3 can either be accessed directly through the native NVM3 API or through the token API in the same way as SimEE1 or SimEE2 based tokens. This library requires the Simulated EEPROM version 2 to NVM3 upgrade library or stub upgrade library. The number of flash pages to use for the NVM3 storage is configurable through the plugin options. IMPORTANT When compiling for a device which already

Options:

Flash Pages:[3-1000]	18
Cache Size:	200
Max Object Size:[208-4096]	254
User Repack Headroom:	0

Details (double-click on files to show content):

Located at: C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\platform\base\

- Common source files (1)
- Per-architecture source files (2)
- Library files (4)
- Architectures (2)
- Setup contributions (3)
- Options (4)
- APIs (4)

Adapter Pack Console

```
boardRevision[1]=A01
boardSerial[1]=170100104
boardDate[1]=2017/1/12
inferPart[1]=yes
}
```

v5_workspace - Z3SwitchSoc/Z3SwitchSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE Configurator
Project Explorer
DynamicMultiprotocolLightSoc [IAR ARM - D
DynamicMultiprotocolLightSoc_callbacks.
brd4161a_efr32mg12p432f1024gl125.hwcon
DynamicMultiprotocolLightSoc.isc
EFM8BB1_ADC_Lib_Interrupt [Keil 8051 v9.60.
soc_empty [GNU ARM v7.2.1 - Debug] [EFR32
soc_ibeacon [GNU ARM v7.2.1 - Debug] [EFR3
Z3LightSoc [GNU ARM v7.2.1 - Default] [EFR3
Z3SwitchSoc [GNU ARM v7.2.1 - Default] [EFR
*Z3SwitchSoc.isc
DynamicMultiprotocolLightSoc.isc
brd4161a_efr32mg12p432f1024gl125.hwconf
Generate
Preview
General
ZCL Clusters
Zigbee Stack
Printing and CLI
HAL
Plugins
Callbacks
Includes
Other options
Bluetooth GATT
Reset to defaults
Problems
Console
751M of 931M

Enabled callbacks must be implemented in application code, otherwise you will get linker errors.

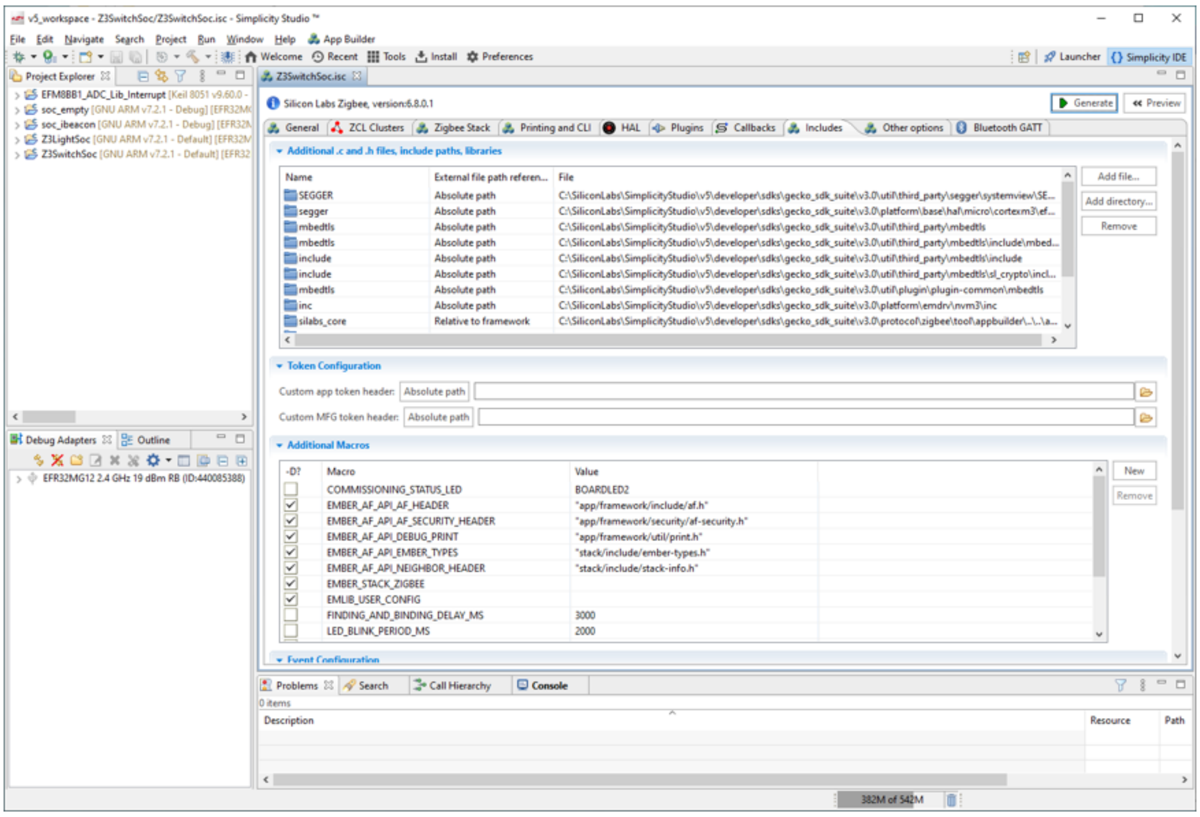
Callbacks are grouped into logical sections in the callbacks tab for ease of navigation. Callbacks implemented by an included plugin have the plug symbol. All of the custom application framework callbacks are included in the "Non-cluster related" section. All of the other callbacks are grouped by cluster. Many of the cluster-specific callbacks are intended for command handling.

When a command comes in, it is passed off to a cluster-specific callback for processing. If the callback returns TRUE then the internal handler for that command will not be called. Cluster-related callbacks are also only implemented for those client and server clusters that are included in the application. If a cluster is not included, the application is not expected to parse that cluster's commands. The application framework returns a default response of UNSUPPORTED_COMMAND.

Callback generation: The first time you generate your application from AppBuilder, it automatically generates a callbacks.c file with the name <project name>_callbacks.c. When you regenerate files in the future, AppBuilder protects your generated callbacks file from being overwritten by asking if you want to overwrite it. By default, AppBuilder will not overwrite any previously created callbacks file. If you choose to overwrite the file, AppBuilder backs up the previous version to the file <project name>_callbacks.bak.

Includes

Use the includes tab to manage information included in the application.



Silicon Labs Zigbee, version:6.8.0.1

Additional .c and .h files, include paths, libraries

Name	External file path referen...	File
SEGGER	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\third_party\segger\systemview\SE...
segger	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\platform\base\hal\micro\cortexm3\ef...
mbedtls	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\third_party\mbedtls
mbedtls	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\third_party\mbedtls\include\mbed...
include	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\third_party\mbedtls\include
include	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\third_party\mbedtls\sl_crypto\incl...
mbedtls	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\util\plugin\plugin-common\mbedtls
inc	Absolute path	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\platform\emdrv\nvm3\inc
silabs_core	Relative to framework	C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\protocol\zigbee\tool\appbuilder\..\..\a...

Token Configuration

Custom app token header: Absolute path

Custom MFG token header: Absolute path

Additional Macros

-D?	Macro	Value
	COMMISSIONING_STATUS_LED	BOARDLED2
✓	EMBER_AF_API_AF_HEADER	"app/framework/include/af.h"
✓	EMBER_AF_API_AF_SECURITY_HEADER	"app/framework/security/af-security.h"
✓	EMBER_AF_API_DEBUG_PRINT	"app/framework/util/print.h"
✓	EMBER_AF_API_EMBER_TYPES	"stack/include/ember-types.h"
✓	EMBER_AF_API_NEIGHBOR_HEADER	"stack/include/stack-info.h"
✓	EMBER_STACK_ZIGBEE	
✓	EMLIB_USER_CONFIG	
	FINDING_AND_BINDING_DELAY_MS	3000
	LED_BLINK_PERIOD_MS	2000

Additional .c and .h files

When you generate a project file from AppBuilder, you often need to modify the project to include your own source files into the application image. This is a problem if the project file needs to be re-generated, since the newly generated project file will overwrite the old one, thereby removing any files that have been included.

Any files that you would like to compile along with your application can be included in the file include table. To include a new file, click **New**. Navigate to the location of the file you wish to include and click **OK**. The file will be included inside the generated project file with an absolute path to the location of the file.

Token Configuration

To implement attributes in tokens, the application framework places its own token file into the generated APPLICATION_TOKEN_HEADER #define within the generated project file.

The application framework's token header file, located at app/framework/util/tokens.h, includes a spot where you may place a relative path to your own token header. The token header files are chained off one another with the application framework's token header at the top of the chain. To include your own token header within your application, provide absolute or relative path inside the appropriate Token Configuration path field.

Additional Macros

Much of the configuration for the application framework and the stack is done using macros documented within the API. Any additional macros that you wish to define, either for compile time or for run time, should be provided here.

Compile-time macros should include the "-D" option. These macros will be included in the compiler and linker definition sections in the generated project file. Any macros which do not include the "-D" option will be included in the generated application header file.

Event Configuration

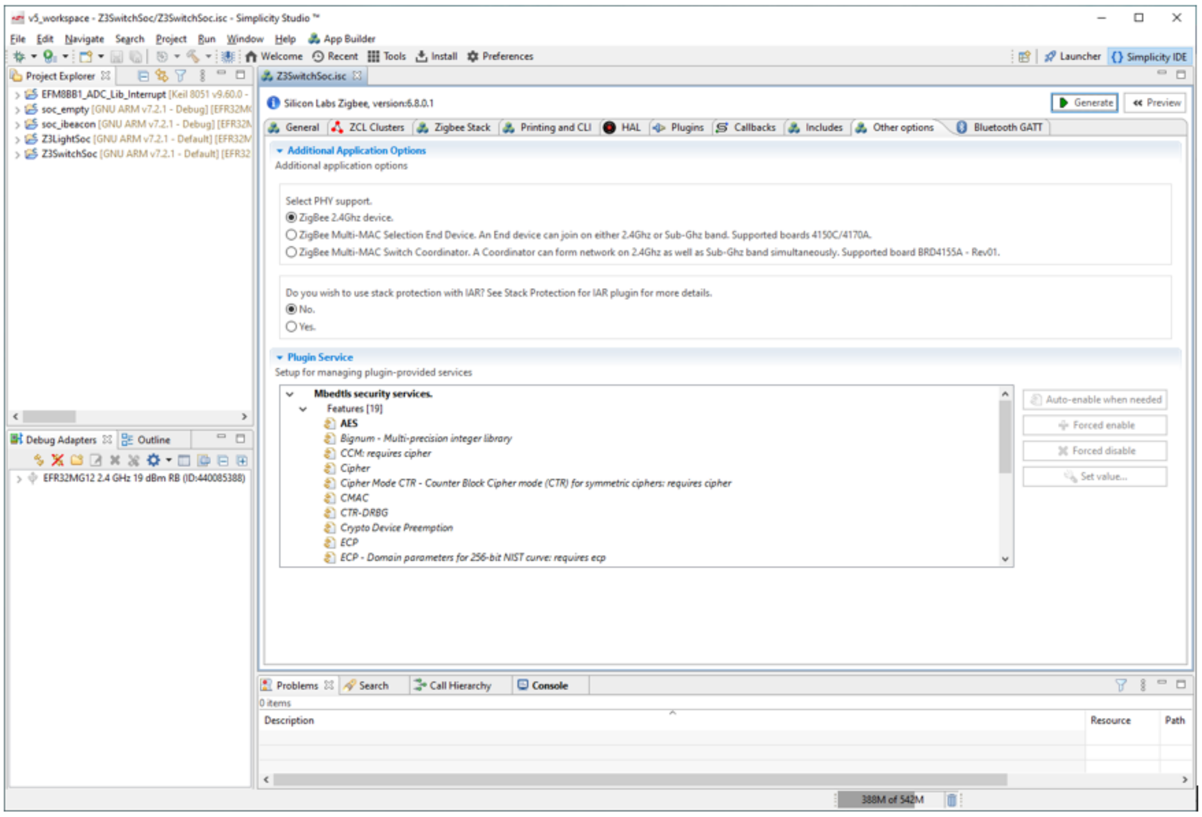
You can include custom application events into the application framework's event system. These events are run in the same manner as the application framework's own events as well as all cluster events implemented by the cluster plugins.

To include a custom event, click **Add New**. Your custom event will be automatically included in the event configuration section of the generated *endpoint_configuration.c* file. Stubs for your event function and event control will be generated into the *callbacks.c* file. You may also change the name of the generated event control value and function to whatever you desire.

For more information on how events work UG391: Zigbee Application Framework Developer's Guide .

Other Options

Use the Other Options tab to configure PHY support, stack protection, and to manage plugin-provided services.



Additional Application Options
Additional application options

Select PHY support.
[✓ ZigBee 2.4Ghz device.]
ZigBee Multi-MAC Selection End Device. An End device can join on either 2.4Ghz or Sub-Ghz band. Supported boards 4150C/4170A.
ZigBee Multi-MAC Switch Coordinator. A Coordinator can form network on 2.4Ghz as well as Sub-Ghz band simultaneously. Supported board BRD4155A - Rev01.

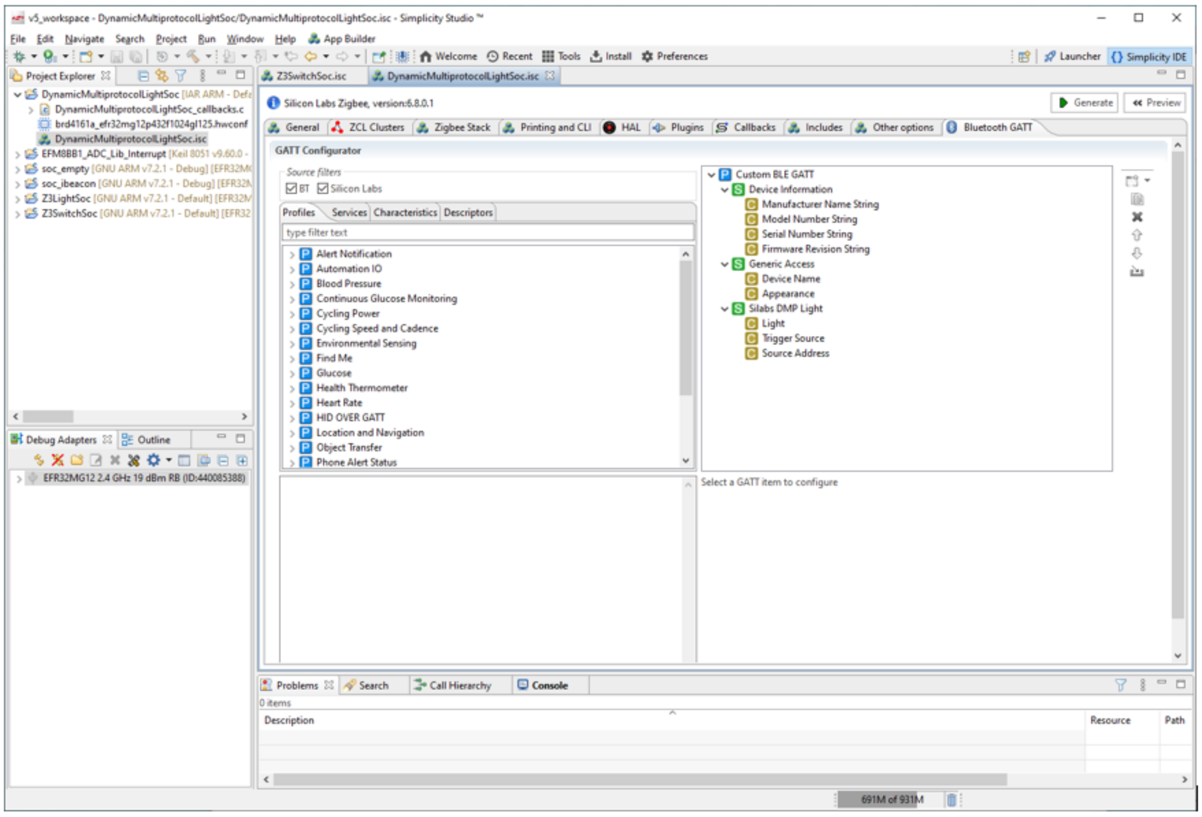
Do you wish to use stack protection with IAR? See Stack Protection for IAR plugin for more details.
[✓ No.]
Yes.

Plugin Service
Setup for managing plugin-provided services

- **Mbedtls security services.**
  - Features [19]
    - **AES**
    - *Bignum - Multi-precision integer library*
    - *CCM: requires cipher*
    - *Cipher*
    - *Cipher Mode CTR - Counter Block Cipher mode (CTR) for symmetric ciphers: requires cipher*
    - *CMAC*
    - *CTR-DRBG*
    - *Crypto Device Preemption*
    - *ECP*
    - *ECP - Domain parameters for 256-bit NIST curve: requires ecp*

Bluetooth GATT

For Dynamic Multiprotocol applications, use the Bluetooth GATT tab to configure the GATT database for the Bluetooth application.


v5_workspace - DynamicMultiprotocolLightSoc/DynamicMultiprotocolLightSoc.isc - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help App Builder
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Project Explorer
DynamicMultiprotocolLightSoc [IAR ARM - Def
DynamicMultiprotocolLightSoc_callbacks.c
brd4161a_efr32mg12p432f1024gl125.hwconf
DynamicMultiprotocolLightSoc.isc
EFM8BB1_ADC_Lib_Interrupt [Keil 8051 v9.60.0 -
soc_empty [GNU ARM v7.2.1 - Debug] [EFR32M
soc_ibeacon [GNU ARM v7.2.1 - Debug] [EFR32M
Z3LightSoc [GNU ARM v7.2.1 - Default] [EFR32M
Z3SwitchSoc [GNU ARM v7.2.1 - Default] [EFR32
Debug Adapters
Outline
EFR32MG12 2.4 GHz 19 dBm RB (ID:440085388)
Z3SwitchSoc.isc
DynamicMultiprotocolLightSoc.isc
Silicon Labs Zigbee, version:6.8.0.1
Generate
Preview
General
ZCL Clusters
Zigbee Stack
Printing and CLI
HAL
Plugins
Callbacks
Includes
Other options
Bluetooth GATT
GATT Configurator
Source filters
BT
Silicon Labs
Profiles
Services
Characteristics
Descriptors
type filter text
Alert Notification
Automation IO
Blood Pressure
Continuous Glucose Monitoring
Cycling Power
Cycling Speed and Cadence
Environmental Sensing
Find Me
Glucose
Health Thermometer
Heart Rate
HID OVER GATT
Location and Navigation
Object Transfer
Phone Alert Status
Custom BLE GATT
Device Information
Manufacturer Name String
Model Number String
Serial Number String
Firmware Revision String
Generic Access
Device Name
Appearance
Silabs DMP Light
Light
Trigger Source
Source Address
Select a GATT item to configure
Problems
Search
Call Hierarchy
Console
0 items
Description
Resource
Path
691M of 931M

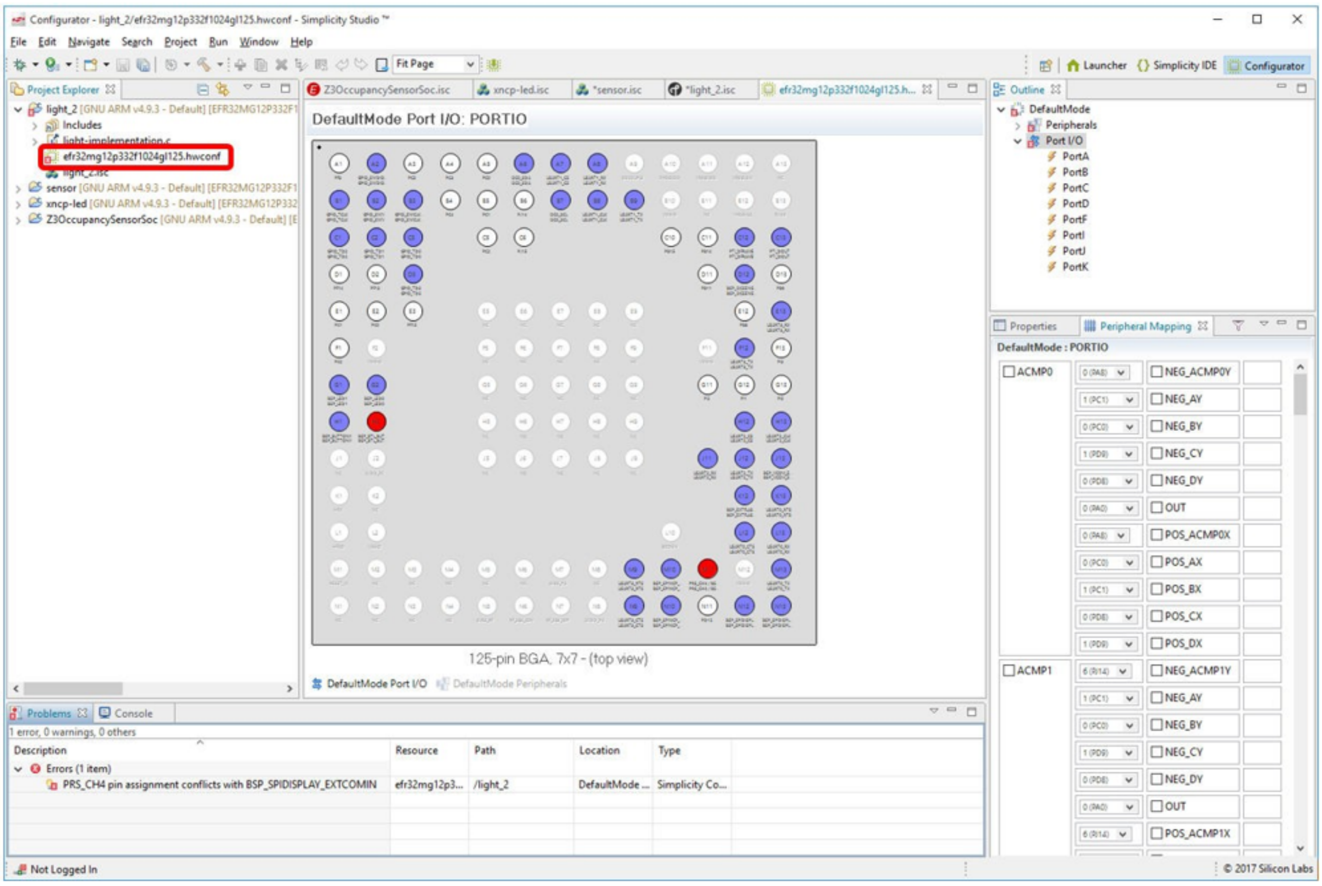
For more information about Bluetooth GATT database configuration see UG365: GATT Configurator User's Guide for Bluetooth SDK v2.x.

Configuring Peripherals

Simplicity Studio® 5 (SSv5)'s AppBuilder HAL tab provides access to a **Hardware Configurator** that you can use to configure pins and peripherals. Access to specific configuration options is also available through various peripheral plugins.

To open the Hardware Configurator for an open project, double-click the *.hwconf* file in Project Explorer, click **Open Hardware Configurator** on the HAL tab, or select **Hardware Configurator** from the Tools menu and then select the target project. The Hardware Configurator has two views, accessed through the two DefaultMode tabs at the bottom of the center pane:

- Port I/O: Used to configure the pin locations and port pins
- Peripherals: Used to configure modules such as Timers, USARTs, and HAL peripherals. Most configuration changes can be made through this tab.



Configurator - light_2/efr32mg12p332f1024gl125.hwconf - Simplicity Studio ™
File Edit Navigate Search Project Run Window Help
Fit Page
Launcher
Simplicity IDE
Configurator
Project Explorer
light_2 [GNU ARM v4.9.3 - Default] [EFR32MG12P332F1
Includes
light-implementation.c
efr32mg12p332f1024gl125.hwconf
light_2.isc
sensor [GNU ARM v4.9.3 - Default] [EFR32MG12P332F1
xncp-led [GNU ARM v4.9.3 - Default] [EFR32MG12P332
Z3OccupancySensorSoc [GNU ARM v4.9.3 - Default] [E
Z3OccupancySensorSoc.isc
xncp-led.isc
*sensor.isc
*light_2.isc
efr32mg12p332f1024gl125.h...
DefaultMode Port I/O: PORTIO
125-pin BGA, 7x7 - (top view)
DefaultMode Port I/O
DefaultMode Peripherals
Outline
DefaultMode
Peripherals
Port I/O
PortA
PortB
PortC
PortD
PortF
PortI
PortJ
PortK
Properties
Peripheral Mapping
Problems
Console
Not Logged In
© 2017 Silicon Labs

1 error, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (1 item)				
PRS_CH4 pin assignment conflicts with BSP_SPIDISPLAY_EXTCOMIN	efr32mg12p3...	/light_2	DefaultMode ...	Simplicity Co...

DefaultMode : PORTIO

ACMP0	0 (PA8)	NEG_ACMP0Y	
	1 (PC1)	NEG_AY	
	0 (PC0)	NEG_BY	
	1 (PD9)	NEG_CY	
	0 (PD8)	NEG_DY	
	0 (PA0)	OUT	
	0 (PA8)	POS_ACMP0X	
	0 (PC0)	POS_AX	
	1 (PC1)	POS_BX	
	0 (PD8)	POS_CX	
	1 (PD9)	POS_DX	
ACMP1	6 (R14)	NEG_ACMP1Y	
	1 (PC1)	NEG_AY	
	0 (PC0)	NEG_BY	
	1 (PD9)	NEG_CY	
	0 (PD8)	NEG_DY	
	0 (PA0)	OUT	
	6 (R14)	POS_ACMP1X	

Port I/O

The Port I/O tab displays a package drawing for the selected device. This drawing updates to display changed pin assignments. Hover the cursor over a pin to see its definition. White = unused, blue = assigned, orange = two or more signals going to the pin, conflict allowed, and red = two or more signals going to the pin, conflict not allowed. While you can make some changes to the peripheral associated with a given pin, Silicon Labs recommends changing peripheral configuration through the Peripheral tab. Use this configuration for pin customizations such as renaming the pin.

Some changes must be made in both the Port I/O and Peripherals tab. For example:

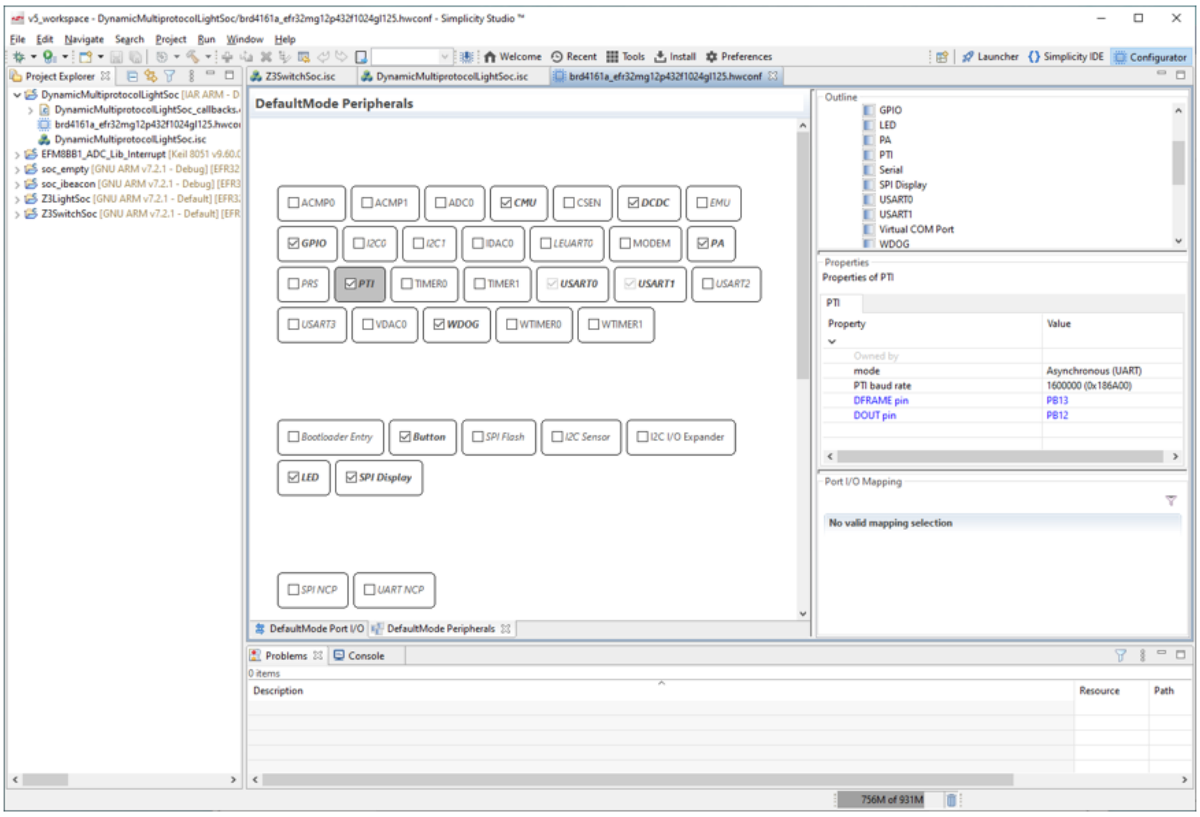
- A route can be selected in Port I/O and an APORT channel can be selected in the peripherals properties.
- ACMP output can be selected on the Port IO route selection and ACMP inputs can be configured in the Peripherals section.

A printable report can be generated by right-clicking on the pinout diagram and selecting Pin Configuration Report. This opens a report as a webpage in a browser that can be saved, printed, or archived. The Module Configuration Report option generates a similar set of tables organized by module rather than by pin order.

Note: To share information about the hardware configurations for a device Silicon Labs recommends sharing the .hwconf file itself.

Peripherals

The Peripherals tab contains boxes, each of which represents either a hardware peripheral or a software concept that is HAL-related but not physically on the chip. Peripheral boxes are organized into groups. Collapse or expand a group by clicking the arrow icon on the upper right of the group. Hover your cursor over a box to see information about the peripheral.

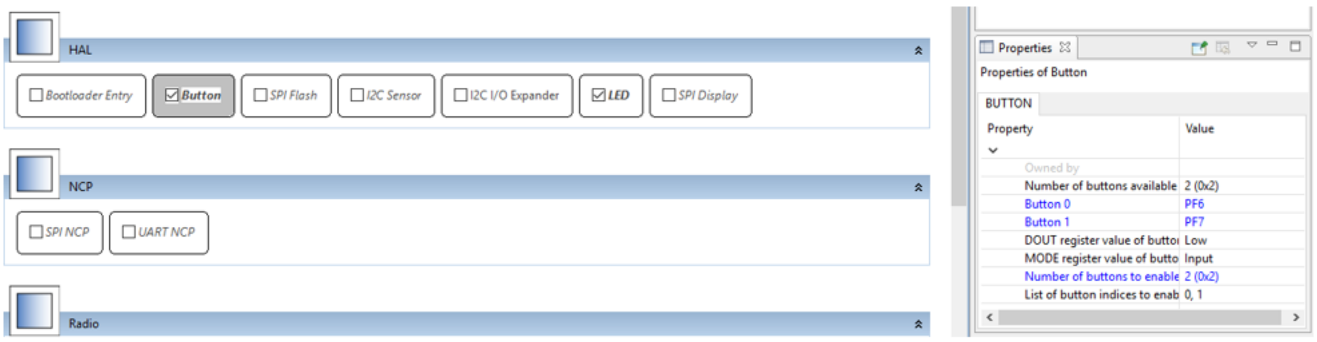


Properties of PTI

Property	Value
Owned by	
mode	Asynchronous (UART)
PTI baud rate	1600000 (0x186A00)
DFRAME pin	PB13
DOUT pin	PB12

Click on a box to see its properties in the Properties pane. Properties can either have a drop-down menu with the available selections or a text input box for a numeric or text field. After making a selection for a property, click away from the property or press **Enter** to ensure the property value change occurs. If you cannot disable a peripheral it may be owned by another peripheral. Go to the settings for that peripheral to free up any dependent peripherals.

Note: The defines for the peripheral are only generated if the checkbox on the peripheral box is checked. For example, the interface for a project based on the Silicon Labs Zigbee Z3LightSoc example has a HAL group with a Button peripheral. Click on that peripheral to see its properties.



Properties of Button

BUTTON

Property	Value
Owned by	
Number of buttons available	2 (0x2)
Button 0	PF6
Button 1	PF7
DOUT register value of buttoı	Low
MODE register value of butto	Input
Number of buttons to enable	2 (0x2)
List of button indices to enab	0, 1

Developing for 8-Bit Devices

Developing for 8-Bit Devices

Hardware Configurator (Configurator) for 8-bit devices is part of Simplicity Studio® 5 (SSv5) and greatly simplifies EFM8 and C8051 MCU peripheral initialization by presenting peripherals and peripheral properties in a graphical user interface.

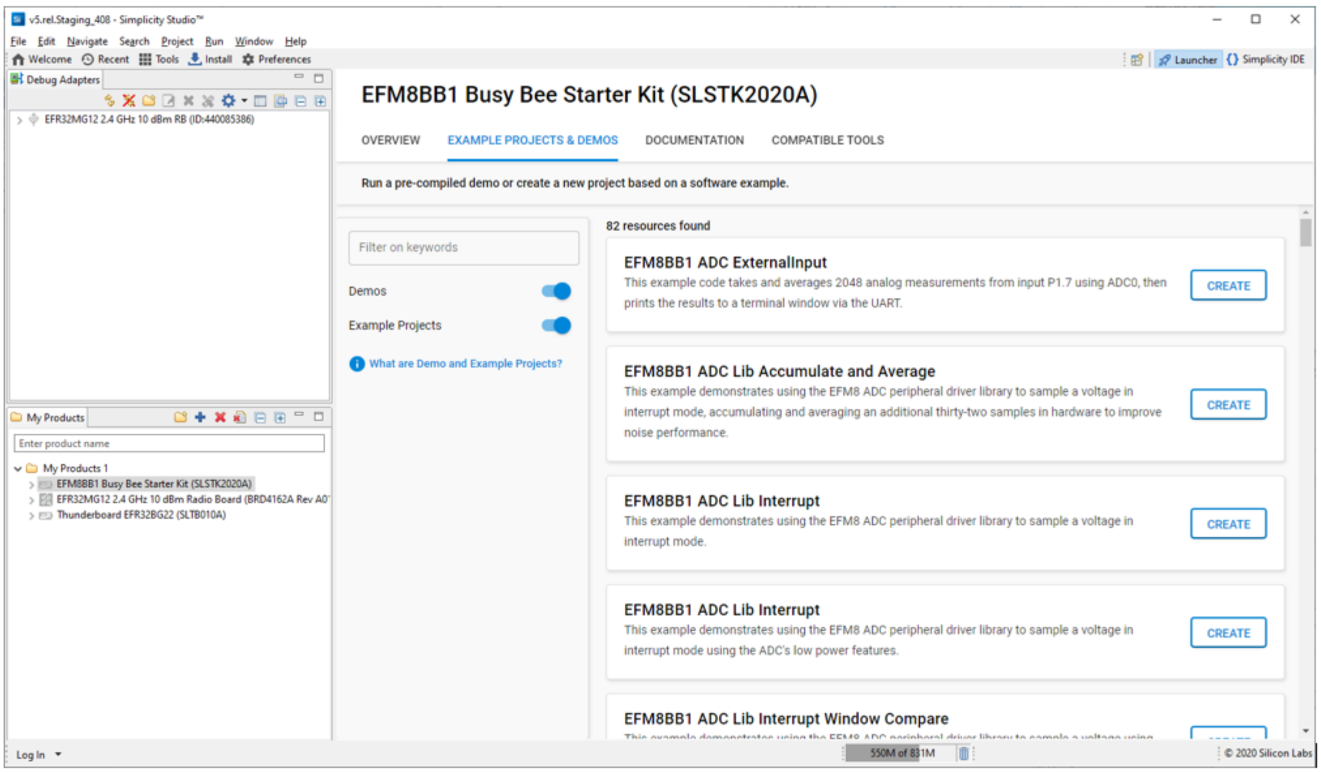
The 8051 SDK contains an extensive and nearly comprehensive set of examples for 8-bit MCU peripherals for each supported device family. These examples show simple use cases for each peripheral, and can be used as building blocks for larger systems incorporating multiple peripherals, or as a starting place for applications requiring each peripheral. Projects based on these examples are customized by changing code as described in About Projects and by updating peripheral configuration using the 8-bit Hardware Configurator.

In Hardware Configurator, most of the initialization firmware can be generated by selecting peripherals and property values from combo boxes or entering register values in text boxes. Some peripherals provide calculators, such as baud rate, timer overflow rate, and SPI clock rate, that can be used to automatically confirm the necessary reload register value to generate the specified clock rate. Hardware Configurator also provides real-time property validation to ensure that a configuration is valid before downloading code to the MCU.

About Projects

About Projects

Create a Simplicity Studio® 5 (SSv5) project for an 8-bit target by selecting a target device, and then selecting an example, as described in Project Creation. An extensive list of examples is provided for most Silicon Labs EFM8 and C8051 devices. These examples contain configurations specific to each target device.


v5.rel.Staging_408 - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Debug Adapters
EFR32MG12 2.4 GHz 10 dBm RB (ID:440085386)
My Products
Enter product name
My Products 1
EFM8BB1 Busy Bee Starter Kit (SLSTK2020A)
EFR32MG12 2.4 GHz 10 dBm Radio Board (BRD4162A Rev A0
Thunderboard EFR32BG22 (SLTB010A)
OVERVIEW
EXAMPLE PROJECTS & DEMOS
DOCUMENTATION
COMPATIBLE TOOLS
Filter on keywords
Demos
Example Projects
What are Demo and Example Projects?
CREATE
Log In
550M of 831M
© 2020 Silicon Labs

EFM8BB1 Busy Bee Starter Kit (SLSTK2020A)

Run a pre-compiled demo or create a new project based on a software example.

82 resources found

EFM8BB1 ADC ExternalInput
This example code takes and averages 2048 analog measurements from input P1.7 using ADC0, then prints the results to a terminal window via the UART.

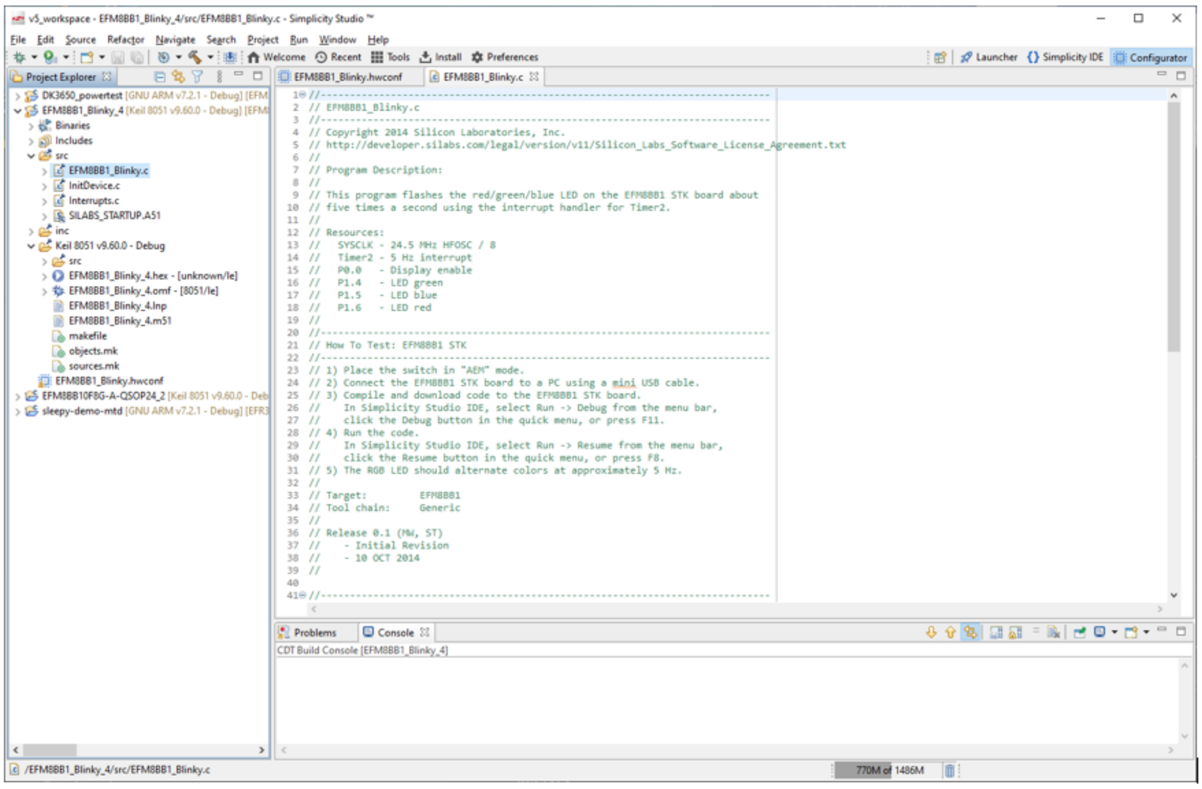
EFM8BB1 ADC Lib Accumulate and Average
This example demonstrates using the EFM8 ADC peripheral driver library to sample a voltage in interrupt mode, accumulating and averaging an additional thirty-two samples in hardware to improve noise performance.

EFM8BB1 ADC Lib Interrupt
This example demonstrates using the EFM8 ADC peripheral driver library to sample a voltage in interrupt mode.

EFM8BB1 ADC Lib Interrupt
This example demonstrates using the EFM8 ADC peripheral driver library to sample a voltage in interrupt mode using the ADC's low power features.

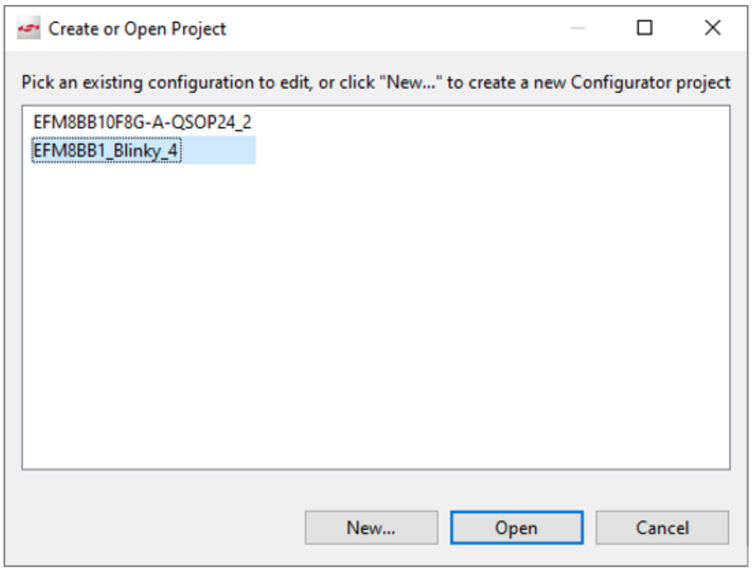
EFM8BB1 ADC Lib Interrupt Window Compare

The Simplicity IDE perspective opens with the project directory highlighted in Project Explorer view. Expand the directory and double-click the project source to see or modify the code.



```
//-----------------------------------------------------------------------------
// EFM8BB1_Blinky.c
//-----------------------------------------------------------------------------
// Copyright 2014 Silicon Laboratories, Inc.
// http://developer.silabs.com/legal/version/v11/Silicon_Labs_Software_License_Agreement.txt
//
// Program Description:
//
// This program flashes the red/green/blue LED on the EFM8BB1 STK board about
// five times a second using the interrupt handler for Timer2.
//
// Resources:
//   SYSCLK - 24.5 MHz HFOSC / 8
//   Timer2 - 5 Hz interrupt
//   P0.0   - Display enable
//   P1.4   - LED green
//   P1.5   - LED blue
//   P1.6   - LED red
//
//-----------------------------------------------------------------------------
// How To Test: EFM8BB1 STK
//-----------------------------------------------------------------------------
// 1) Place the switch in "AEM" mode.
// 2) Connect the EFM8BB1 STK board to a PC using a mini USB cable.
// 3) Compile and download code to the EFM8BB1 STK board.
//    In Simplicity Studio IDE, select Run -> Debug from the menu bar,
//    click the Debug button in the quick menu, or press F11.
// 4) Run the code.
//    In Simplicity Studio IDE, select Run -> Resume from the menu bar,
//    click the Resume button in the quick menu, or press F8.
// 5) The RGB LED should alternate colors at approximately 5 Hz.
//
// Target:         EFM8BB1
// Tool chain:     Generic
//
// Release 0.1 (MW, ST)
//    - Initial Revision
//    - 10 OCT 2014
//

//-----------------------------------------------------------------------------
```

Open Hardware Configurator by double-clicking the *.hwconf* file in Project Explorer, or by selecting **Hardware Configurator** from the Tools menu, and selecting the project.



Create or Open Project

Pick an existing configuration to edit, or click "New..." to create a new Configurator project

EFM8BB10F8G-A-QSOP24_2
EFM8BB1_Blinky_4

Use Hardware Configurator to customize the configuration and functionality as desired. Modifications to the device configuration can be saved to the *.hwconf* file. Hardware Configurator generates output in the form of generated code in multiple files, as described in Using Hardware Configurator.

Build and debug the application as described in Building and Using the Debugger.

Using Hardware Configurator

Using Hardware Configurator

Simplicity Studio® 5 (SSv5)'s Hardware Configurator for 8-bit devices uses the **Configurator** perspective. This perspective is tailored specifically for use with Hardware Configurator to initialize MCU peripherals. Special views in this perspective allow peripheral registers to be configured.

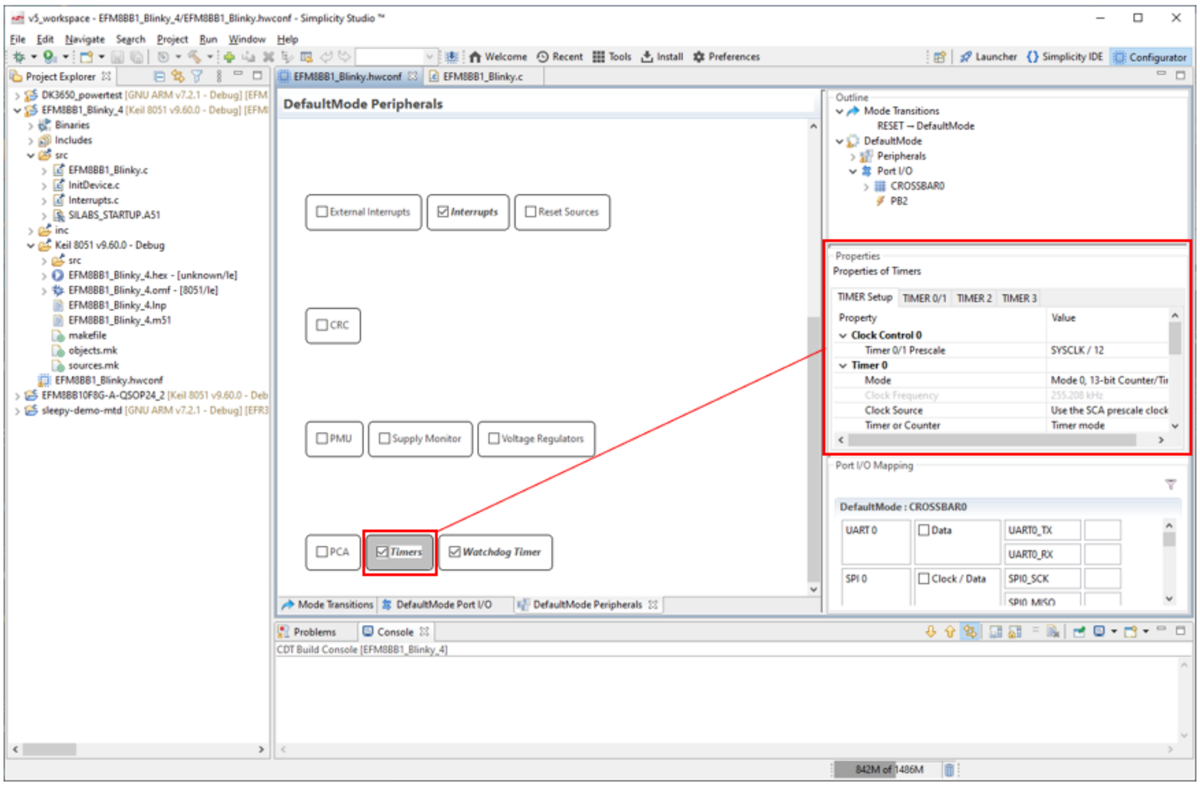
The available editors in a Configurator project are as follows:

- Peripherals: Used to configure hardware peripherals such as ADCs, Timers, LEUART, PCAs, and so on.
- Port I/O: Used to configure the pin locations, crossbar, and port pins. This editor is open by default for a new project.
- Mode Transitions: Used to define different MCU peripheral initialization states, such as DefaultMode, RESET, and other optional modes (example: LowPower mode). This editor can also be used to define transitions between states.

The Hardware Configurator automatically validates register configuration values. Occasionally changes to the configuration can cause errors or warnings. These clear as soon as the cause is addressed.

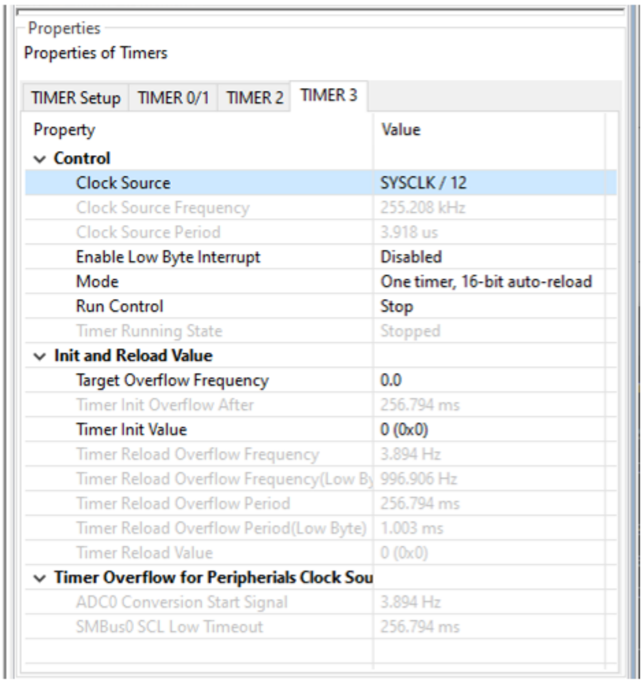
Peripherals

Use the Peripherals tab to configure hardware peripherals like the ADC, comparators, or timers. To configure a peripheral, click the peripheral icon. This selects the peripheral and opens it in the Properties view. Properties can either have a drop-down menu with the available selections or a text input box for a numeric or text field. Properties that are grayed out are read-only. These properties typically provide more information about the configuration of a peripheral. After making a selection for a property, click away from the property or press **Enter** to ensure the property value change occurs.



Property	Value
Clock Control 0	
Timer 0/1 Prescale	SYSCLK / 12
Timer 0	
Mode	Mode 0, 13-bit Counter/Ti
Clock Frequency	255.208 kHz
Clock Source	Use the SCA prescale clock
Timer or Counter	Timer mode

For example, click the Timers peripheral to display the timer properties in a tabbed view. The Timer 3 tab has several properties that can be modified, like **Clock Source**, **Mode**, and **Target Overflow Frequency**. Modifying any of these properties will update the **Timer Reload Overflow Period** and **Timer Reload Overflow Frequency** read-only properties, which display the calculated overflow time based on the **Target Overflow Frequency** and the **Clock Source** settings. These read-only fields and calculators depend on the Mode and Run Control state. Calculators for reload value are for auto-reload mode.

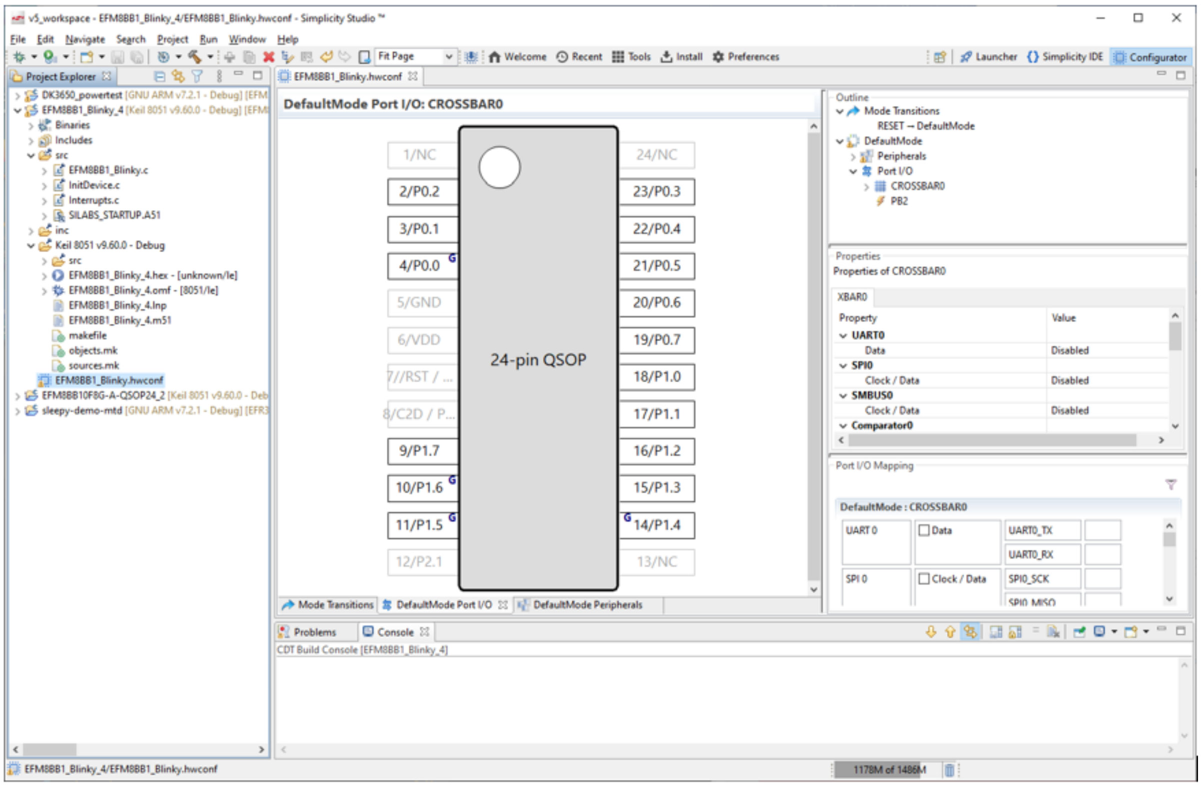


Property	Value
Control	
Clock Source	SYSCLK / 12
Clock Source Frequency	255.208 kHz
Clock Source Period	3.918 us
Enable Low Byte Interrupt	Disabled
Mode	One timer, 16-bit auto-reload
Run Control	Stop
Timer Running State	Stopped
Init and Reload Value	
Target Overflow Frequency	0.0
Timer Init Overflow After	256.794 ms
Timer Init Value	0 (0x0)
Timer Reload Overflow Frequency	3.894 Hz
Timer Reload Overflow Frequency(Low By	996.906 Hz
Timer Reload Overflow Period	256.794 ms
Timer Reload Overflow Period(Low Byte)	1.003 ms
Timer Reload Value	0 (0x0)
Timer Overflow for Peripherials Clock Sou	
ADC0 Conversion Start Signal	3.894 Hz
SMBus0 SCL Low Timeout	256.794 ms

Note: Peripherals will not have code generation enabled unless the peripheral checkbox is checked.

Port I/O

The Port I/O tab displays a package drawing for the selected device. This drawing updates to display pin assignments as peripherals are enabled on the crossbar or peripherals with fixed pin assignments are enabled. For fixed pin assignments, the associated peripheral must be enabled in order for the fixed pin function to appear on the Port I/O tab. Changes are shown in blue until the file is saved.



Properties

Properties of CROSSBAR0

XBAR0

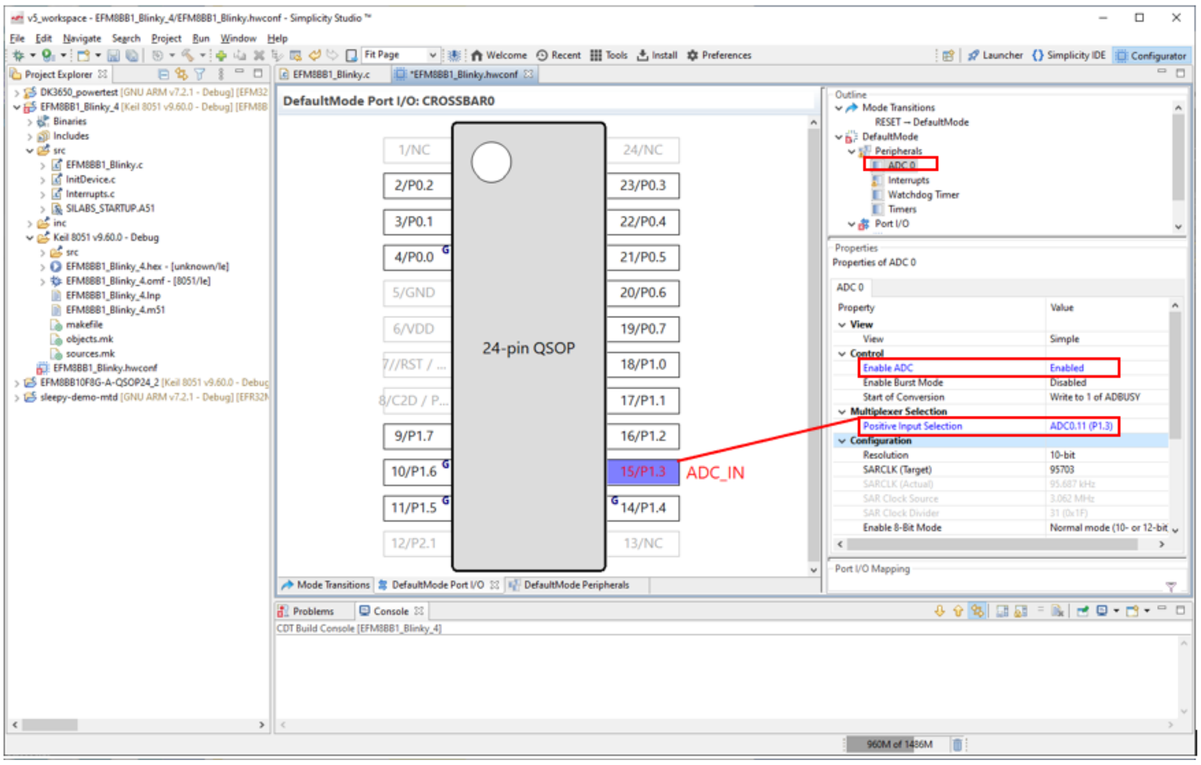
Property	Value
UART0	
Data	Disabled
SPI0	
Clock / Data	Disabled
SMBUS0	
Clock / Data	Disabled
Comparator0	

Port I/O Mapping

DefaultMode : CROSSBAR0

UART 0	☐ Data	UART0_TX	
		UART0_RX	
SPI 0	☐ Clock / Data	SPI0_SCK	
		SPI0_MISO	

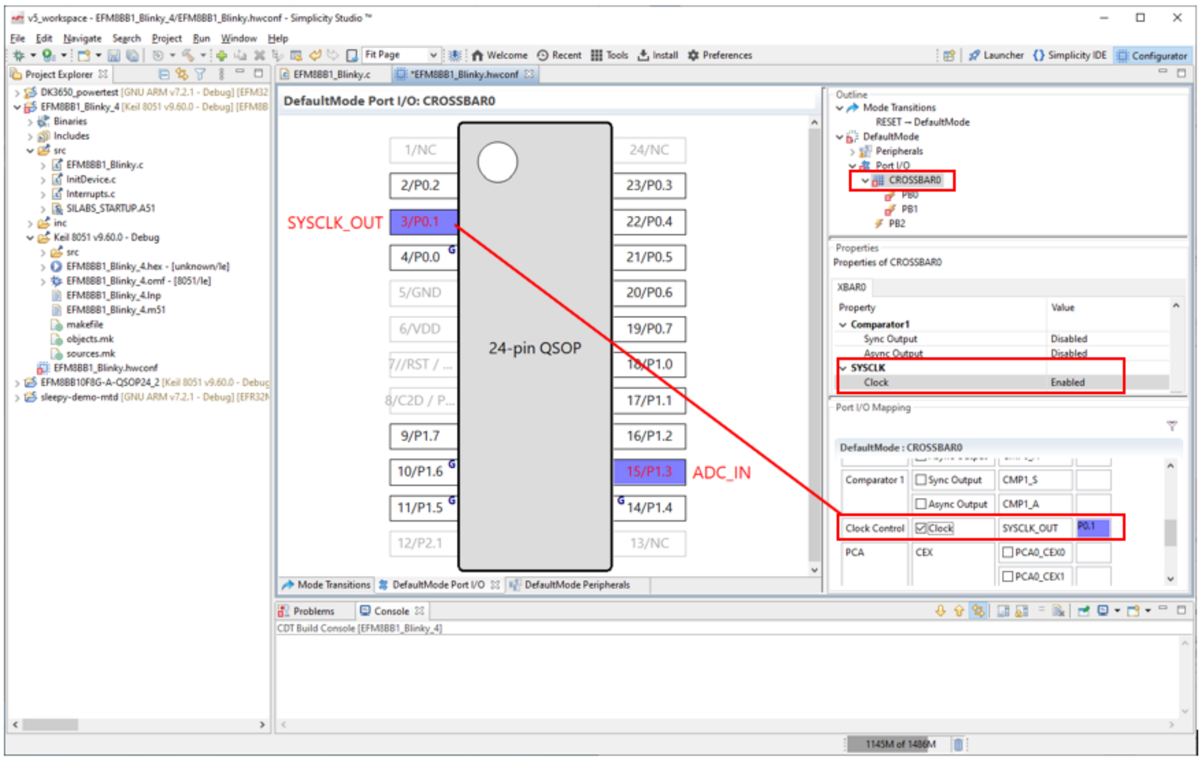
For example, on the Peripherals tab make sure the ADC0 module is checked. On the port I/O tab, in the Outline view select ADC0, in Properties change **Enable ADC** to `Enabled`, and select a **Positive Input pin** (in this example P1.3). This displays the ADC_IN fixed signal on the selected pin in the drawing.



Properties of ADC 0

Property	Value
View	
View	Simple
Control	
Enable ADC	Enabled
Enable Burst Mode	Disabled
Start of Conversion	Write to 1 of ADBUSY
Multiplexer Selection	
Positive Input Selection	ADC0.11 (P1.3)
Configuration	
Resolution	10-bit
SARCLK (Target)	95703
SARCLK (Actual)	95.687 kHz
SAR Clock Source	3.062 MHz
SAR Clock Divider	31 (0x1F)
Enable 8-Bit Mode	Normal mode (10- or 12-bit

To enable a peripheral on the crossbar, on the Peripherals tab enable a peripheral for code generation (in this case Clock Control). On the Port I/O tab, in the Outline view click CROSSBAR0, and in the Port I/O Mapping view check the corresponding signal. This changes the corresponding property to `Enabled`.



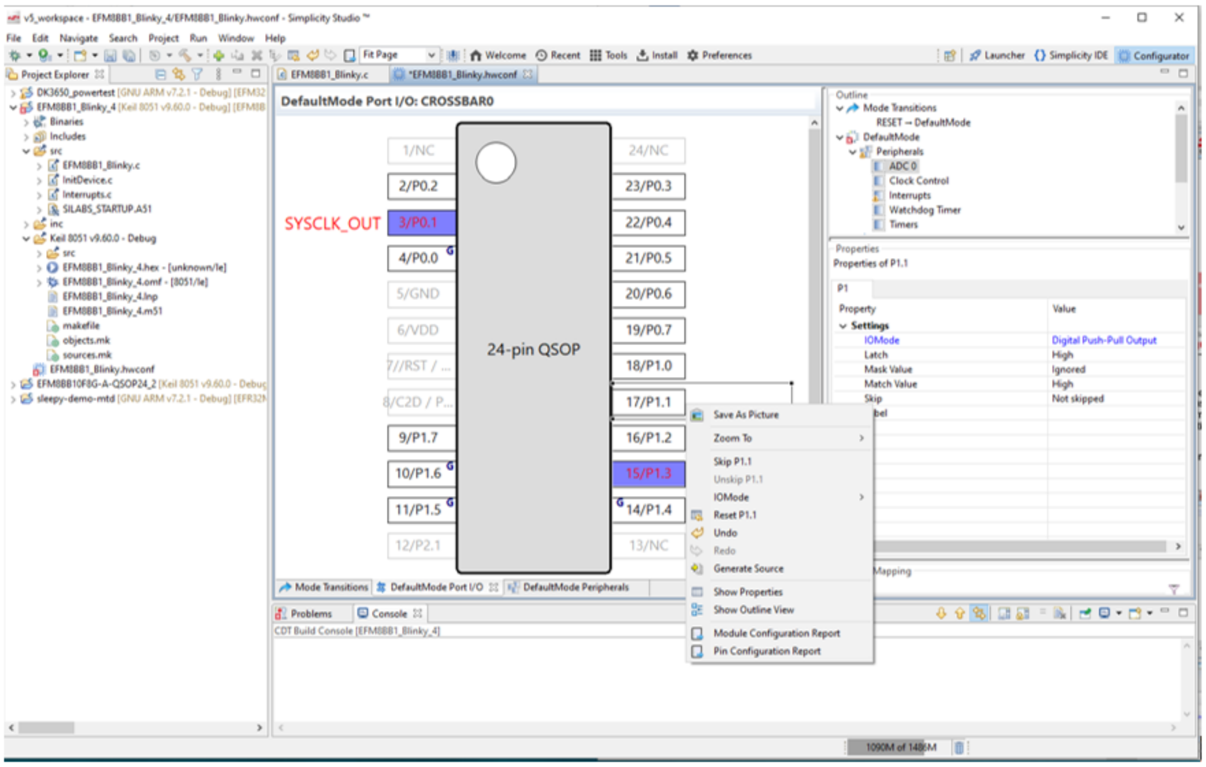
Properties of CROSSBAR0

Property	Value
Comparator1	
Sync Output	Disabled
Async Output	Disabled
SYSCLK	
Clock	Enabled

DefaultMode : CROSSBAR0

Comparator 1	☐ Sync Output	CMP1_S	
	☐ Async Output	CMP1_A	
Clock Control	☑ Clock	SYSCLK_OUT	P0.1
PCA	CEX	☐ PCA0_CEX0	
		☐ PCA0_CEX1	

To configure a pin's properties, click the pin in the package drawing and select the desired pin property settings. If configuring a device with a crossbar, a pin can be skipped by right-clicking the pin and selecting **Skip** in the context menu or changing the Skip property setting.



Properties of P1.1

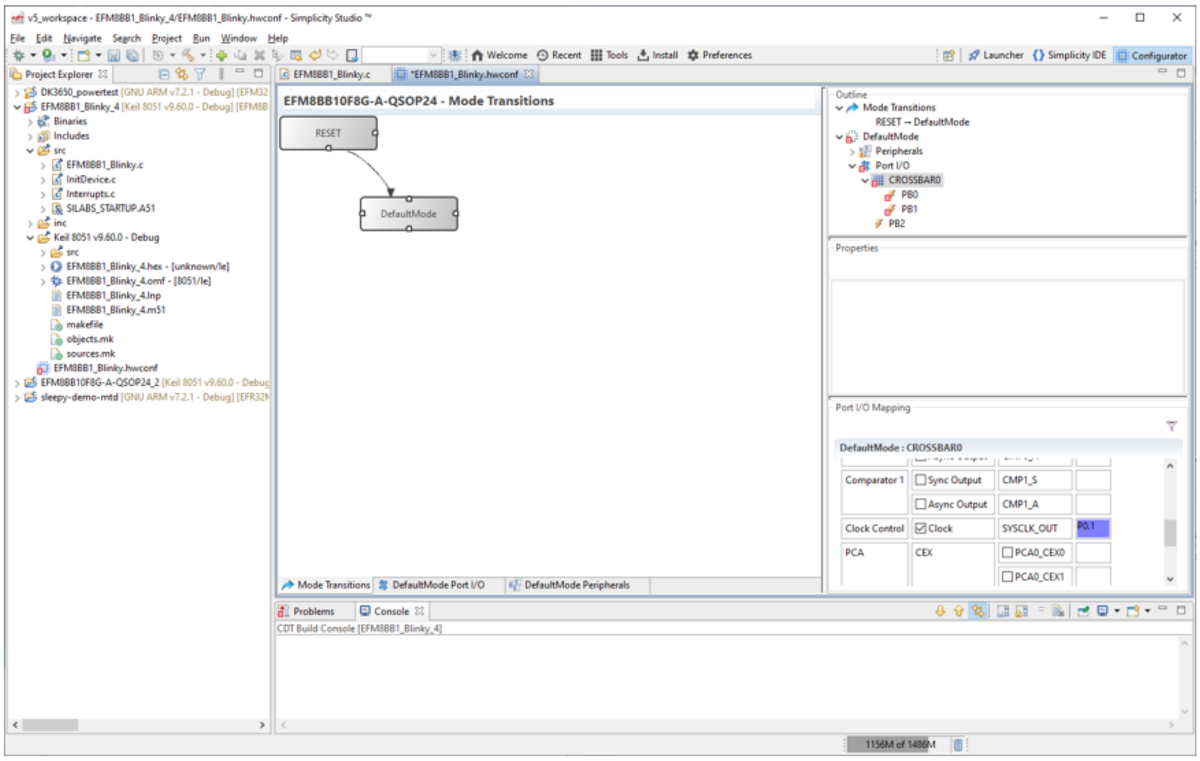
Property	Value
Settings	
IOMode	Digital Push-Pull Output
Latch	High
Mask Value	Ignored
Match Value	High
Skip	Not skipped

Similarly, I/O mode can be selected by right-clicking the pin and selecting **IOMode** or changing the IOMode property setting.

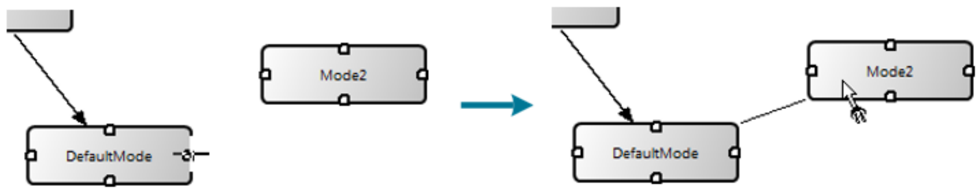
On all devices, multiple pins can be selected by holding Ctrl or Shift and selecting the desired pins or holding the left mouse button and dragging over a group of pins. With multiple pins selected, the property changes made in the Properties view or context menu apply to all pins. To reset a pin to its default state, right-click the pin and select **Reset**.

Mode Transitions

Modes define different states of peripheral configurations. For example, an application may use two configurations of the ADC and Timer 2 with different input pins and sampling rates. These two configurations can be treated as separate modes with separate initialization sequences. The movements between these modes are called transitions. By default, each project contains one **DefaultMode** mode. The reset state exists only to indicate the transition from RESET to DefaultMode and is not a full mode.



To add a new mode, right-click either the Mode Transitions tab or the Outline view and select **Create Mode from Reset**. Right-clicking an existing mode also enables cloning that mode with the **Clone Mode** option. To add a transition from one mode to another, hover over one of the four white handles on a mode until the mouse cursor changes, hold down left mouse button, and drag to the end mode of the transition. A Peripherals and Port I/O pair of tabs are added for each mode in the project.

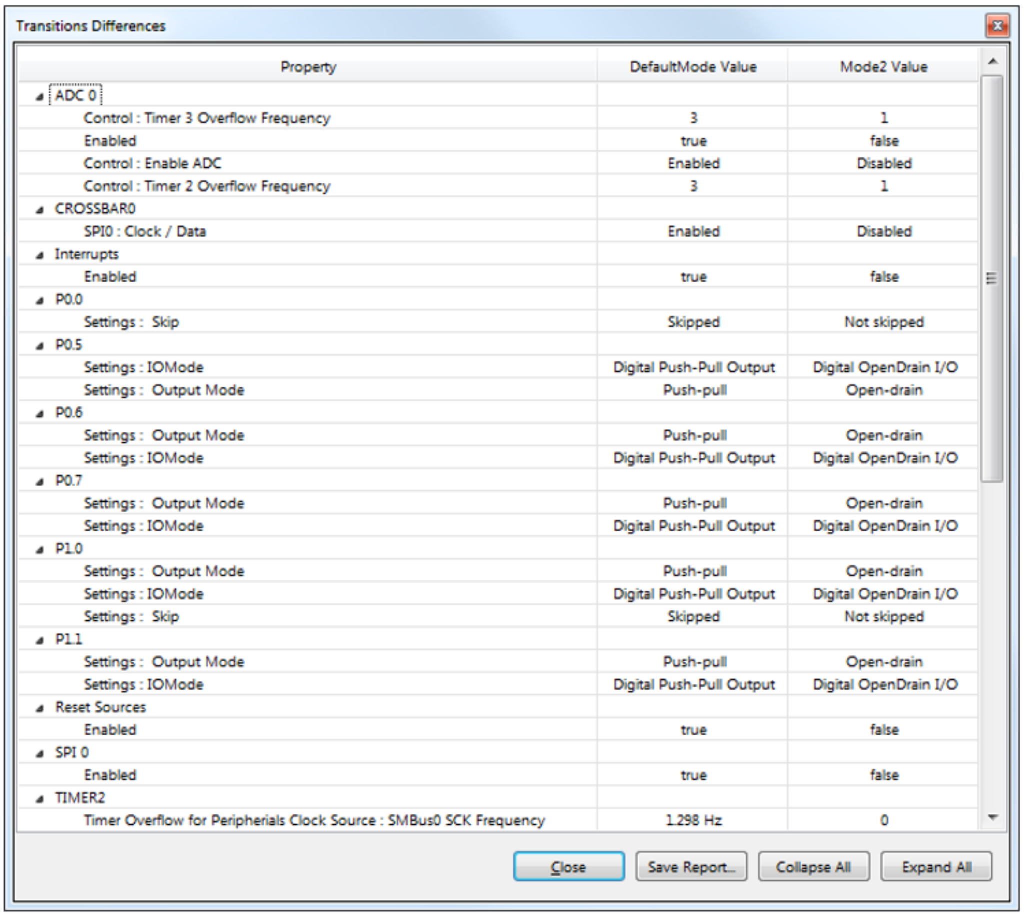


After reset, call the mode transition `enter_DefaultMode_from_RESET()` to apply the DefaultMode settings. In most cases this will be the first transition in the application code.

Adding a transition generates a corresponding function that application firmware can call to initiate the transition between modes. For example, with DefaultMode as the originating mode going to Mode2:

`enter_Mode2_from_DefaultMode(void)`

This function then calls all of the functions that configure the peripherals for the mode. Hardware Configurator generates only the transitional code from peripheral to peripheral between modes. For example, if only the timer reload value and ADC input mux selection differ, then those are the only two properties touched by the transition code. To see the differences between two modes, right-click the mode transition and select **Show Differences**. This provides a report of all the different property values between the modes on a module-by-module basis.



Property	DefaultMode Value	Mode2 Value
ADC 0		
Control : Timer 3 Overflow Frequency	3	1
Enabled	true	false
Control : Enable ADC	Enabled	Disabled
Control : Timer 2 Overflow Frequency	3	1
CROSSBAR0		
SPI0 : Clock / Data	Enabled	Disabled
Interrupts		
Enabled	true	false
P0.0		
Settings : Skip	Skipped	Not skipped
P0.5		
Settings : IOMode	Digital Push-Pull Output	Digital OpenDrain I/O
Settings : Output Mode	Push-pull	Open-drain
P0.6		
Settings : Output Mode	Push-pull	Open-drain
Settings : IOMode	Digital Push-Pull Output	Digital OpenDrain I/O
P0.7		
Settings : Output Mode	Push-pull	Open-drain
Settings : IOMode	Digital Push-Pull Output	Digital OpenDrain I/O
P1.0		
Settings : Output Mode	Push-pull	Open-drain
Settings : IOMode	Digital Push-Pull Output	Digital OpenDrain I/O
Settings : Skip	Skipped	Not skipped
P1.1		
Settings : Output Mode	Push-pull	Open-drain
Settings : IOMode	Digital Push-Pull Output	Digital OpenDrain I/O
Reset Sources		
Enabled	true	false
SPI 0		
Enabled	true	false
TIMER2		
Timer Overflow for Peripherals Clock Source : SMBus0 SCK Frequency	1.298 Hz	0

Transitions can be added in both directions using the same method as the original mode transition.

To delete a mode transition, select the arrow and press Delete or right-click the arrow and select **Delete transition**.

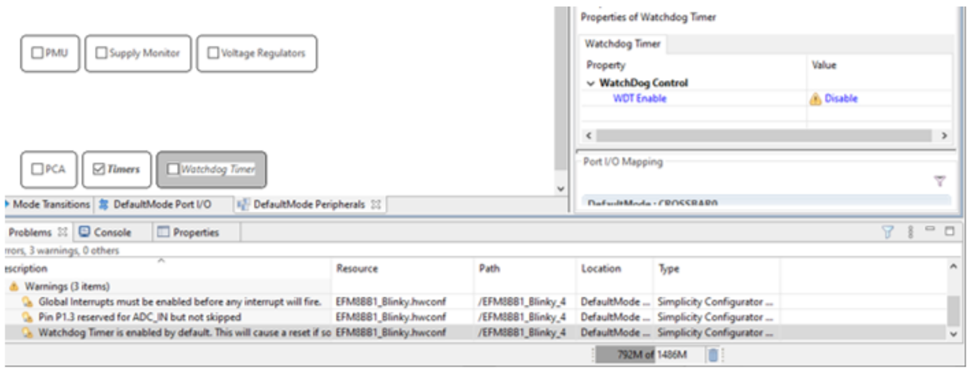
Transition functions will not modify the state of the global interrupt enable (EA) on EFM8 or C8051 devices. Transitions assume that interrupts are disabled (that is, global interrupt enable bit EA = 0) or that any enabled interrupts will not interact with any changes made during the transition. Transition functions do not clear interrupt flags, to ensure that no information is lost during reconfiguration. If having these flags set will cause an issue in firmware operation, firmware must clear these interrupt flags before calling the transition function.

Transition functions also assume that any peripherals being updated are not active. For example, firmware should stop a timer before calling a transition function that changes the timer configuration and restart the timer after the transition completes.

It is possible to change the configuration order of a set of peripherals by creating multiple modes with multiple transitions. For example, to configure Timer 2 before the ADC during a mode transition that would normally put the ADC first, create a mode to handle the Timer 2 changes and a mode to handle the ADC changes. Firmware can then call the Timer2 mode transition function first, followed by the ADC transition function.

Errors and Warnings

The Hardware Configurator automatically validates register configuration values and displays any errors, warnings, or information items in the Problems view. Double-clicking an entry displays the property that raised the problem in the Properties view. Any peripherals that contain a warning or error are highlighted in yellow (warning) or red (error), and the associated property has a corresponding warning or error icon.



Properties of Watchdog Timer

Watchdog Timer

Property	Value
WatchDog Control	
WDT Enable	Disable

rrors, 3 warnings, 0 others

escription	Resource	Path	Location	Type
Warnings (3 items)				
Global Interrupts must be enabled before any interrupt will fire.	EFM8BB1_Blinky.hwconf	/EFM8BB1_Blinky_4	DefaultMode ...	Simplicity Configurator ...
Pin P1.3 reserved for ADC_IN but not skipped	EFM8BB1_Blinky.hwconf	/EFM8BB1_Blinky_4	DefaultMode ...	Simplicity Configurator ...
Watchdog Timer is enabled by default. This will cause a reset if so	EFM8BB1_Blinky.hwconf	/EFM8BB1_Blinky_4	DefaultMode ...	Simplicity Configurator ...

Once the highlighted property is updated, the associated warning or error will disappear from the Problems view, indicating that the problem has been solved.

All newly created projects will have a warning regarding the Watchdog Timer, since it's enabled after a reset and must be handled in application code or disabled. If a warning or error doesn't apply to the project, it can be temporarily deleted from the project by right-clicking the entry in the Problems view and selecting **Delete**. Closing and reopening the *.hwconf* file regenerates any deleted warnings or errors.

Code Generation

To generate code for a peripheral, check the checkbox for that peripheral in the Peripherals tab. Saving the project automatically generates code for all selected peripherals and port pins. This code generates into the *InitDevice.c*, *InitDevice.h*, and *Interrupts.c* files.

Each hardware register has a section in the *InitDevice.c* file tagged with the register name and register description. For example, the P1 register area is as follows:

```
// $[P1 - Port 1 Pin Latch]
// [P1 - Port 1 Pin Latch]$
```

These tags indicate that everything in between is automatically generated, so any code added manually between these tags will be deleted on the next Hardware Configurator project save. Code can be added in between these tags, however, and this code will not be overwritten.

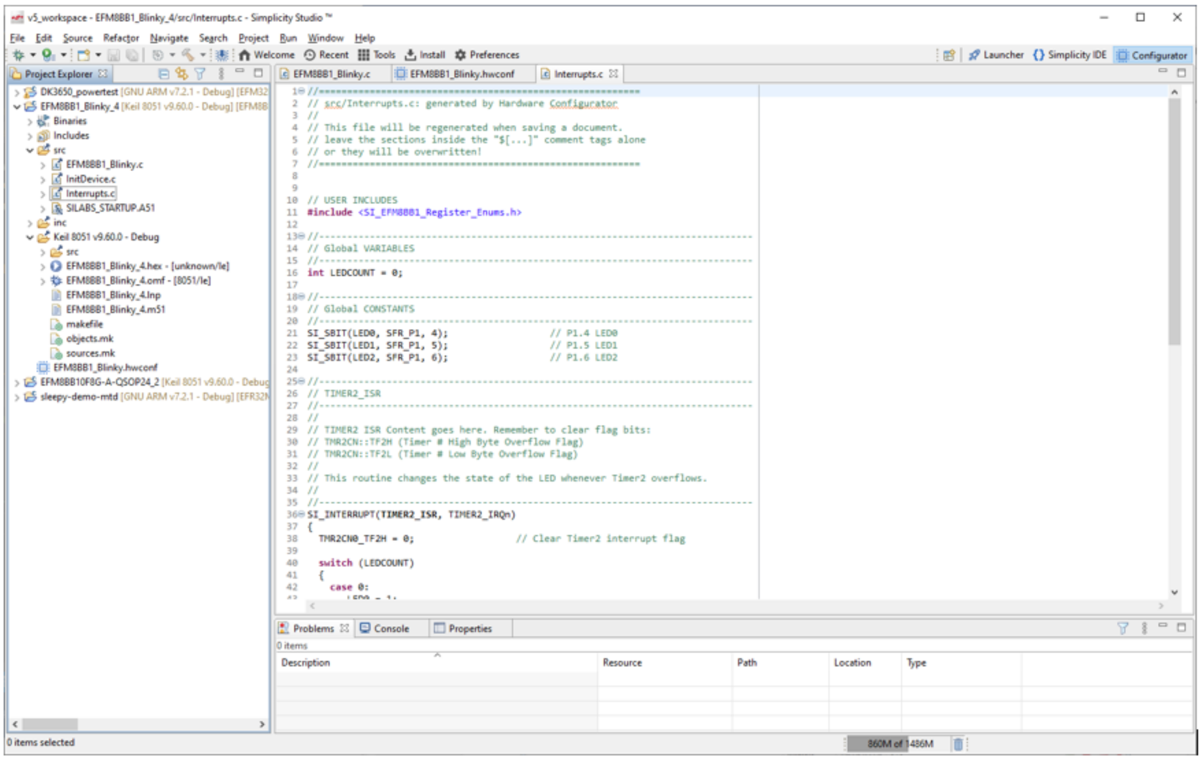
```
// $[P1 - Port 1 Pin Latch]
P1 = 0×00; // This code is not safe, and Configurator will overwrite it
// [P1 - Port 1 Pin Latch]$

P1 = 0×00; // This code is safe from Configurator automatic code generation

// $[P1MASK - Port 1 Mask]
// [P1MASK - Port 1 Mask]$
```

For registers that are modified from their reset value or the value in the previous mode, the tags include comments that update based on the settings for the register. For example, setting P1.0 to push-pull on an EFM8 or 8-bit MCU generates the following:

```
// $[P1MDOUT - Port 1 Output Mode]
/*
// B0 (Port 1 Bit 0 Output Mode) = PUSH_PULL
//  (P1.0 output is push-pull.)
// B1 (Port 1 Bit 1 Output Mode) = OPEN_DRAIN
//  (P1.1 output is open-drain.)
// B2 (Port 1 Bit 2 Output Mode) = OPEN_DRAIN
//  (P1.2 output is open-drain.)
// B3 (Port 1 Bit 3 Output Mode) = OPEN_DRAIN
//  (P1.3 output is open-drain.)
// B4 (Port 1 Bit 4 Output Mode) = OPEN_DRAIN
//  (P1.4 output is open-drain.)
// B5 (Port 1 Bit 5 Output Mode) = OPEN_DRAIN
//  (P1.5 output is open-drain.)
// B6 (Port 1 Bit 6 Output Mode) = OPEN_DRAIN
//  (P1.6 output is open-drain.)
// B7 (Port 1 Bit 7 Output Mode) = OPEN_DRAIN
//  (P1.7 output is open-drain.)
*/
P1MDOUT = P1MDOUT_B0 PUSH_PULL | P1MDOUT_B1 OPEN_DRAIN | P1MDOUT_B2 OPEN_DRAIN
   | P1MDOUT_B3 OPEN_DRAIN | P1MDOUT_B4 OPEN_DRAIN | P1MDOUT_B5 OPEN_DRAIN | P1MDOUT_B6 OPEN_DRAIN | P1MDOUT_B7
OPEN_DRAIN;
// [P1MDOUT - Port 1 Output Mode]$
```

Interrupts

The Hardware Configurator creates an *interrupts.c* file containing interrupt prototypes whenever interrupts are enabled and the project is saved. Hardware Configurator generates the prototype if it is not present, but does not overwrite existing prototypes. This protects any application code written in the prototypes from being accidentally modified or deleted. The comments above the prototypes list any flags in the modules that may need to be cleared by application code after the interrupt triggers.


v5_workspace - EFM8BB1_Blinky_4/src/Interrupts.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE Configurator
Project Explorer
EFM8BB1_Blinky.c
EFM8BB1_Blinky.hwconf
Interrupts.c
Problems Console Properties
0 items
Description Resource Path Location Type
0 items selected

```
//==============================================================
// src/Interrupts.c: generated by Hardware Configurator
//
// This file will be regenerated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!
//==============================================================


// USER INCLUDES
#include <SI_EFM8BB1_Register_Enums.h>

//-----------------------------------------------------------------------------
// Global VARIABLES
//-----------------------------------------------------------------------------
int LEDCOUNT = 0;

//-----------------------------------------------------------------------------
// Global CONSTANTS
//-----------------------------------------------------------------------------
SI_SBIT(LED0, SFR_P1, 4);                  // P1.4 LED0
SI_SBIT(LED1, SFR_P1, 5);                  // P1.5 LED1
SI_SBIT(LED2, SFR_P1, 6);                  // P1.6 LED2

//-----------------------------------------------------------------------------
// TIMER2_ISR
//-----------------------------------------------------------------------------
//
// TIMER2 ISR Content goes here. Remember to clear flag bits:
// TMR2CN::TF2H (Timer # High Byte Overflow Flag)
// TMR2CN::TF2L (Timer # Low Byte Overflow Flag)
//
// This routine changes the state of the LED whenever Timer2 overflows.
//
//-----------------------------------------------------------------------------
SI_INTERRUPT(TIMER2_ISR, TIMER2_IRQn)
{
  TMR2CN0_TF2H = 0;                  // Clear Timer2 interrupt flag

  switch (LEDCOUNT)
  {
    case 0:
```

Building and Flashing

Building and Flashing

These pages describe:

- How to compile or build software into a binary image
- How to configure additional build steps using the Post-Build Editor to make customized binary images
- How to load or flash that application image onto a connected device

Building

Building

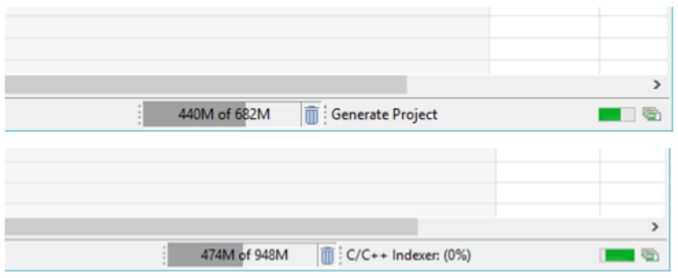
Simplicity Studio® 5 (SSv5) offers two convenient ways to compile or 'build' projects:

- Simple build
- Debug build and flash

Projects are built with the toolchain defined when the project was created, and using the active configuration. Project files must be generated before you can build the project.

- For 32-bit device development using AppBuilder, click **Generate** in the Simplicity IDE.
- Files are automatically generated if the project uses Project Configurator.

Progress is shown in the lower right of the Simplicity IDE perspective.



Speed varies depending on your system, number of projects, and other factors. Be sure that generation and indexing complete before building the application image.

The result of the build is a compiled application known as a firmware image or binary. The binaries are physically located in a directory named for the compiler used to generate them, and are also shown as a 'binaries' group in project explorer. Right-click a binary for a context menu from which you can debug, flash, and perform other functions.

The binaries generated depend on both the SDK and the target device. For example, the Keil® 8051 compiler for an 8-bit device generates:

- *.hex (application image)
- *.omf (contains additional debug information)

The GNU ARM® compiler for a 32-bit device using the Zigbee EmberZNet SDK generates:

- *.axf (image that can be read on a Microsoft Windows device)
- *.bin (binary image file, can be flashed to any address)
- *.gbl (Gecko Bootloader file, specialized firmware file for use with the Gecko bootloader)
- *.hex (application image file)
- *.s37 (similar to a binary file but contains the target memory location to flash to - an application image or a bootloader image, but not both)

Note that if you need a .gbl file but do not see one in the binary directory, you can use Simplicity Commander to create one. Commands invoked from Simplicity Commander's CLI can be used to:

- Generate key files for signing and encryption

- Sign application images for Secure Boot
- Create GBL images (encrypted or unencrypted, signed or unsigned)
- Parse GBL images

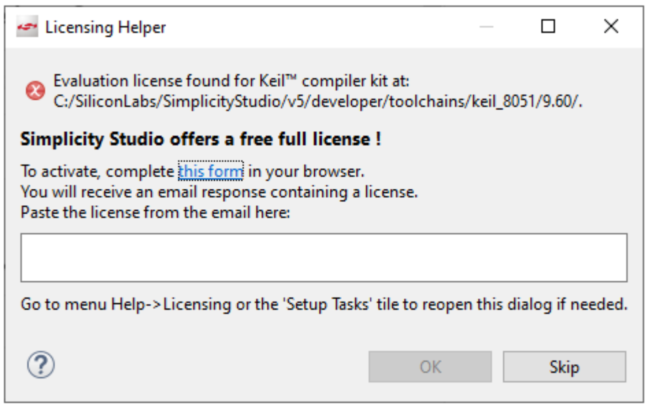
For details about using Simplicity Commander see UG162: Simplicity Commander Reference Guide.

For details about using the Gecko Bootloader, including its enhanced security features, see UG266: Silicon Labs Gecko Bootloader User's Guide.

See your SDK's quick-start guide for more stack-specific information about building and flashing.

About Toolchains

SSv5 provides a GNU ARM® toolchain to build projects for 32-bit devices and a 30-day evaluation license to the Keil® 8051 toolchain to build projects for 8-bit devices. You can apply for a free full license when you build an 8-bit project.



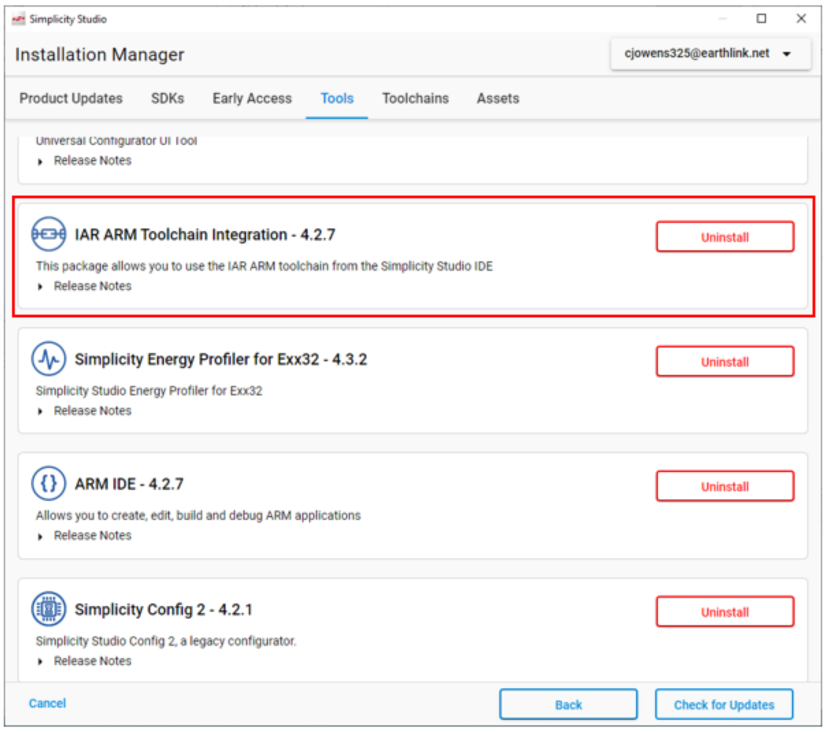
Evaluation license found for Keil™ compiler kit at:
C:/SiliconLabs/SimplicityStudio/v5/developer/toolchains/keil_8051/9.60/.

Simplicity Studio offers a free full license !

To activate, complete this form in your browser.
You will receive an email response containing a license.
Paste the license from the email here:

Go to menu Help->Licensing or the 'Setup Tasks' tile to reopen this dialog if needed.

You can also add your own toolchain, for example IAR-EWARM. IAR is required for some projects, such as the Zigbee Dynamic Multiprotocol and Micrium OS examples. However, you must use the version specified in the SDK release notes front page under compatible tools. See your SDK's quick-start guide for information on how to load a compatible version and obtain a free evaluation license. SSv5 provides an integration package for IAR, so that you can use that tool within SSv5.



Simplicity Studio

Installation Manager

cjowens325@earthlink.net

Universal Configurator UI Tool
Release Notes

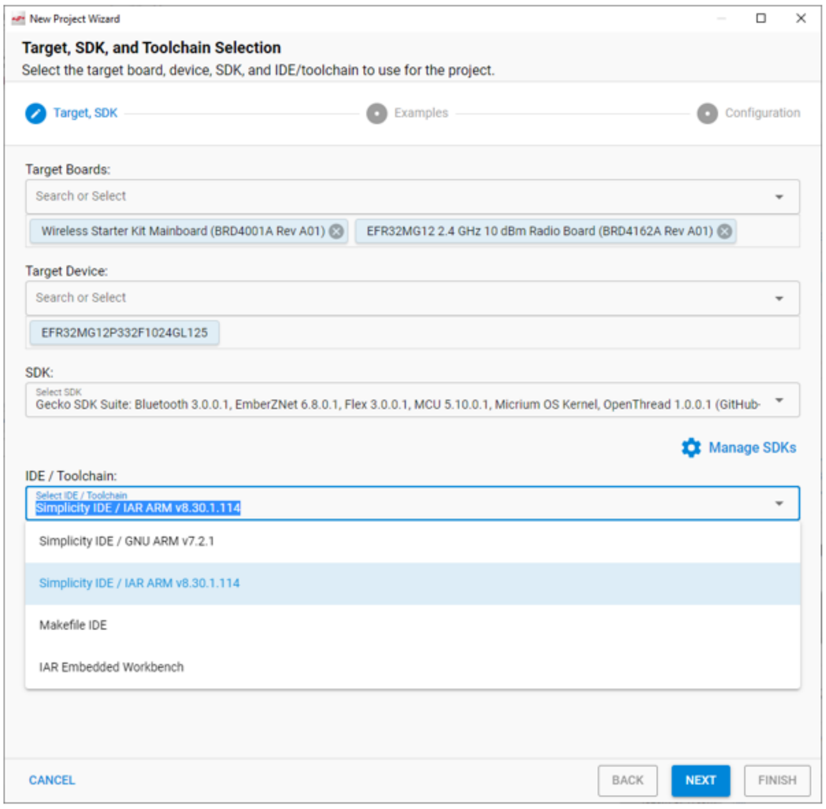
IAR ARM Toolchain Integration - 4.2.7
This package allows you to use the IAR ARM toolchain from the Simplicity Studio IDE
Release Notes

Simplicity Energy Profiler for Exx32 - 4.3.2
Simplicity Studio Energy Profiler for Exx32
Release Notes

ARM IDE - 4.2.7
Allows you to create, edit, build and debug ARM applications
Release Notes

Simplicity Config 2 - 4.2.1
Simplicity Studio Config 2, a legacy configurator.
Release Notes

Projects use the toolchain selected at the time of project creation. The selection is available on the first of the three New Project Wizard dialogs.



New Project Wizard

Target, SDK, and Toolchain Selection
Select the target board, device, SDK, and IDE/toolchain to use for the project.

Target Boards: Search or Select
Wireless Starter Kit Mainboard (BRD4001A Rev A01) | EFR32MG12 2.4 GHz 10 dBm Radio Board (BRD4162A Rev A01)

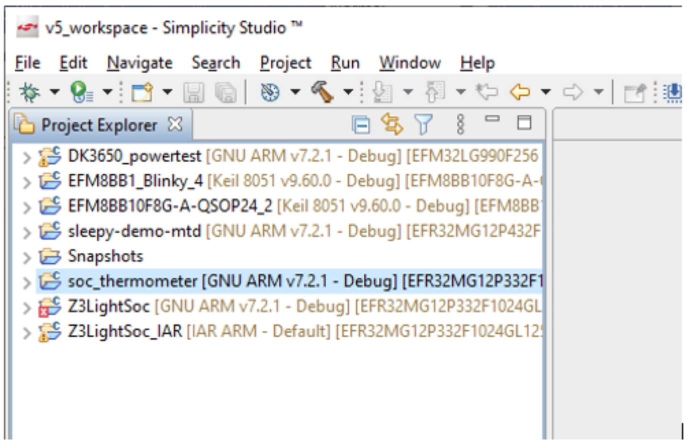
Target Device: Search or Select
EFR32MG12P332F1024GL125

SDK: Select SDK
Gecko SDK Suite: Bluetooth 3.0.0.1, EmberZNet 6.8.0.1, Flex 3.0.0.1, MCU 5.10.0.1, Micrium OS Kernel, OpenThread 1.0.0.1 (GitHub-

IDE / Toolchain: Select IDE / Toolchain
Simplicity IDE / IAR ARM v8.30.1.114

- Simplicity IDE / GNU ARM v7.2.1
- Simplicity IDE / IAR ARM v8.30.1.114
- Makefile IDE
- IAR Embedded Workbench

The toolchain and build configuration are displayed in Project Explorer view with the project directory. The build configuration determines whether the image is built for debugging or release. Default is typically equivalent to a debug build.



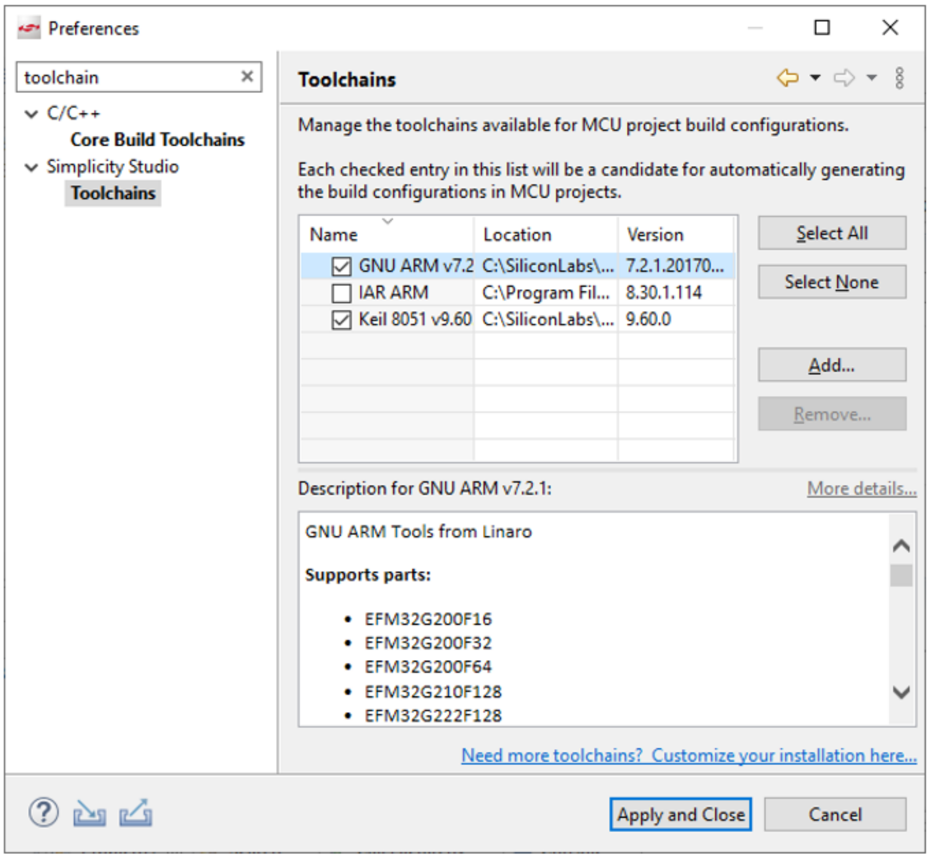
v5_workspace - Simplicity Studio ™

File Edit Navigate Search Project Run Window Help

Project Explorer

- DK3650_powertest [GNU ARM v7.2.1 - Debug] [EFM32LG990F256
- EFM8BB1_Blinky_4 [Keil 8051 v9.60.0 - Debug] [EFM8BB10F8G-A-
- EFM8BB10F8G-A-QSOP24_2 [Keil 8051 v9.60.0 - Debug] [EFM8BB
- sleepy-demo-mtd [GNU ARM v7.2.1 - Debug] [EFR32MG12P432F
- Snapshots
- soc_thermometer [GNU ARM v7.2.1 - Debug] [EFR32MG12P332F1
- Z3LightSoc [GNU ARM v7.2.1 - Debug] [EFR32MG12P332F1024GL
- Z3LightSoc_IAR [IAR ARM - Default] [EFR32MG12P332F1024GL12

If more than one toolchain is available, you can configure which are presented as options when creating a new project. In the **Preferences** filter field type `toolchain`. Uncheck the ones you do not wish to display. You can also completely remove toolchains, or add new toolchains through this dialog.



Preferences

toolchain

- C/C++
  - Core Build Toolchains
- Simplicity Studio
  - Toolchains

Toolchains

Manage the toolchains available for MCU project build configurations.

Each checked entry in this list will be a candidate for automatically generating the build configurations in MCU projects.

Name	Location	Version
✓ GNU ARM v7.2	C:\SiliconLabs\...	7.2.1.20170...
☐ IAR ARM	C:\Program Fil...	8.30.1.114
✓ Keil 8051 v9.60	C:\SiliconLabs\...	9.60.0

Description for GNU ARM v7.2.1: More details...

GNU ARM Tools from Linaro

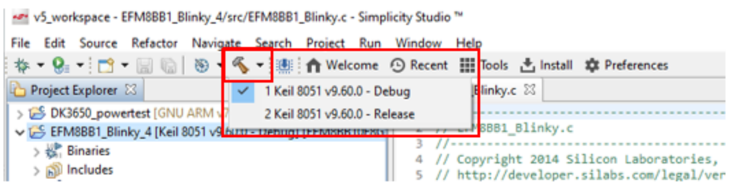
Supports parts:

- EFM32G200F16
- EFM32G200F32
- EFM32G200F64
- EFM32G210F128
- EFM32G222F128

Need more toolchains? Customize your installation here...

Simple Build

Right-click the project directory and in the context menu select **Build**, or click the **Build (hammer)** button on the Simplicity IDE perspective toolbar to build a project with its toolchain and the active build configuration.

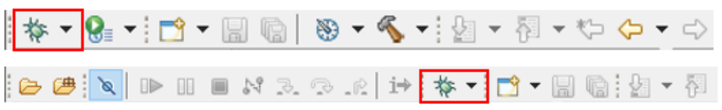

v5_workspace - EFM8BB1_Blinky_4/src/EFM8BB1_Blinky.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Project Explorer
1 Keil 8051 v9.60.0 - Debug
2 Keil 8051 v9.60.0 - Release
DK3650_powertest [GNU ARM v
EFM8BB1_Blinky_4 [Keil 8051 v9
Binaries
Includes

While the Build menu selection always uses the active configuration, the **Build** button's drop-down menu allows you to select a different configuration. This also changes the active configuration for the project to the one selected.

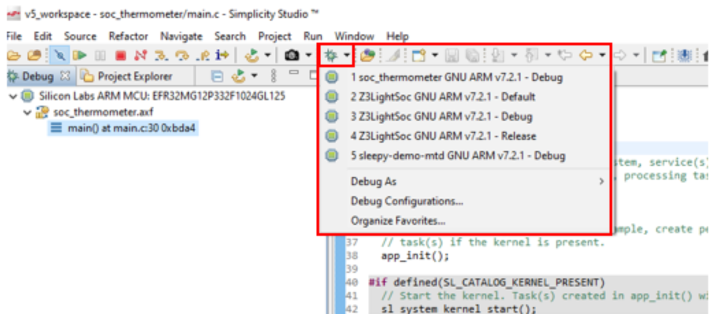
If the **Build** button is not enabled, make sure you have a project directory or file in the directory selected.

Debug Build and Flash

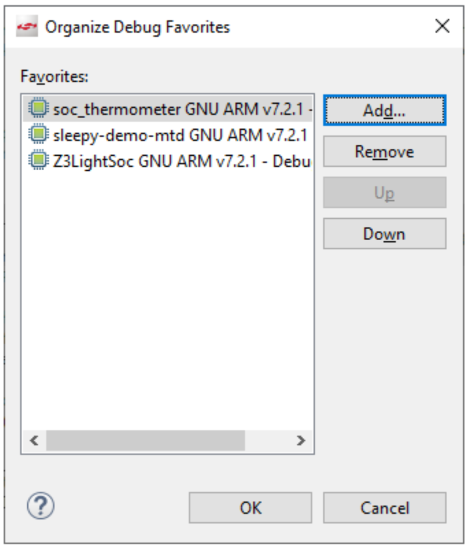
Click the **Debug (bug)** button in either the Simplicity IDE or Debug perspective to build an application image, flash it to the device, and open the Debug perspective (if it is not already open).



The **Debug** button's drop-down menu allows you to select the project in your workspace you want to debug.


v5_workspace - soc_thermometer/main.c - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Project Explorer
Silicon Labs ARM MCU: EFR32MG12P332F1024GL125
soc_thermometer.axf
main() at main.c:30 0xbda4
1 soc_thermometer GNU ARM v7.2.1 - Debug
2 Z3LightSoc GNU ARM v7.2.1 - Default
3 Z3LightSoc GNU ARM v7.2.1 - Debug
4 Z3LightSoc GNU ARM v7.2.1 - Release
5 sleepy-demo-mtd GNU ARM v7.2.1 - Debug
Debug As
Debug Configurations...
Organize Favorites...

If you have a lot of projects use **Organize Favorites** to select favorites and organize them. These projects then appear at the top of the selection list with other projects below a demarcation line.

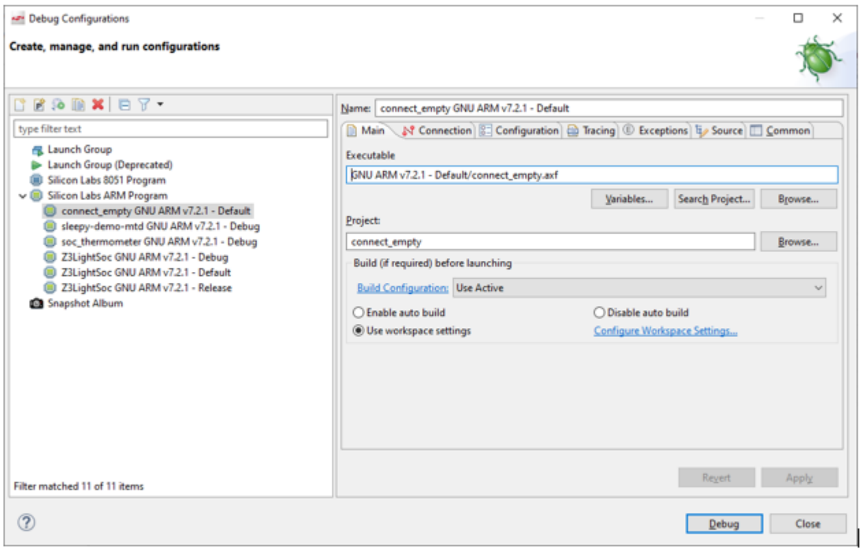


Favorites:

- soc_thermometer GNU ARM v7.2.1 -
- sleepy-demo-mtd GNU ARM v7.2.1
- Z3LightSoc GNU ARM v7.2.1 - Debu

Select **Debug Configurations** to open a tabbed set of configuration option dialogs. If you click **Debug**, the build and flash process will proceed with the options you selected.

See Using the Debugger for more information.



Create, manage, and run configurations

type filter text

- Launch Group
- Launch Group (Deprecated)
- Silicon Labs 8051 Program
- Silicon Labs ARM Program
  - connect_empty GNU ARM v7.2.1 - Default
  - sleepy-demo-mtd GNU ARM v7.2.1 - Debug
  - soc_thermometer GNU ARM v7.2.1 - Debug
  - Z3LightSoc GNU ARM v7.2.1 - Debug
  - Z3LightSoc GNU ARM v7.2.1 - Default
  - Z3LightSoc GNU ARM v7.2.1 - Release
- Snapshot Album

Filter matched 11 of 11 items

Name: connect_empty GNU ARM v7.2.1 - Default

Executable: GNU ARM v7.2.1 - Default/connect_empty.axf

Project: connect_empty

Build (if required) before launching

Build Configuration: Use Active

Enable auto build [✓ Use workspace settings] Disable auto build

Configure Workspace Settings...

Post-Build Editor

Post-Build Editor (PBE)

The Post-Build Editor (PBE) is provided to make it easy for the user to specify actions required on the project output binary to create various types of image files. The tool provides the following capabilities:

- Adding signatures or checksums to application or bootloader binaries
- Merging applications and other data into production images
- Packaging applications and other data into upgrade files, including GBL and Zigbee OTA.

Workflow

The PBE-defined post-build steps are stored in a post-build YAML file with an '.slpb' extension that can be used either from within Simplicity Studio 5 (SSv5) or by running Simplicity Commander on the command line. Simplicity Studio IDE projects and SLC exported projects will execute the post-build steps automatically using the post-build command step to call the command line version of Simplicity Commander. Also, the various project configurator generators (GCC Makefile, Visual Studio Code Compatible (beta), and IAR Embedded Workbench) will include the necessary Simplicity Commander call to automatically perform the post build steps when the project is built in those environments.

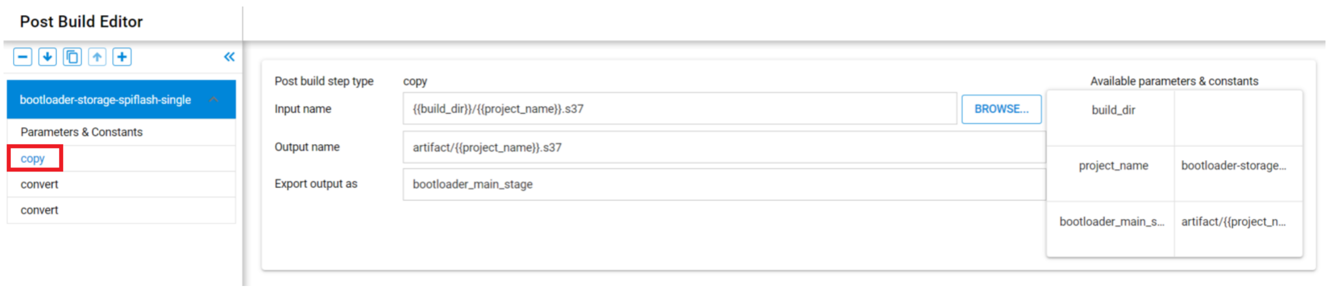
Several example .slpb files are included in the Gecko SDK Suite (GSDK) starting with version 4.2.0.0. For examples of projects that use the .slpb feature, see the TrustZone examples. Refer to UG162: Simplicity Commander User's Guide for details on the signing and secure boot information in the following tasks.

Tasks

The PBE supports four different tasks: copy, convert, create_gbl, and create_ota.

Copy

The `copy` task is used to copy a file from one folder to another and it is also the starting point for specifying the post-build sequence.



Post build step type: copy

Input name: {{build_dir}}/{{project_name}}.s37

Output name: artifact/{{project_name}}.s37

Export output as: bootloader_main_stage

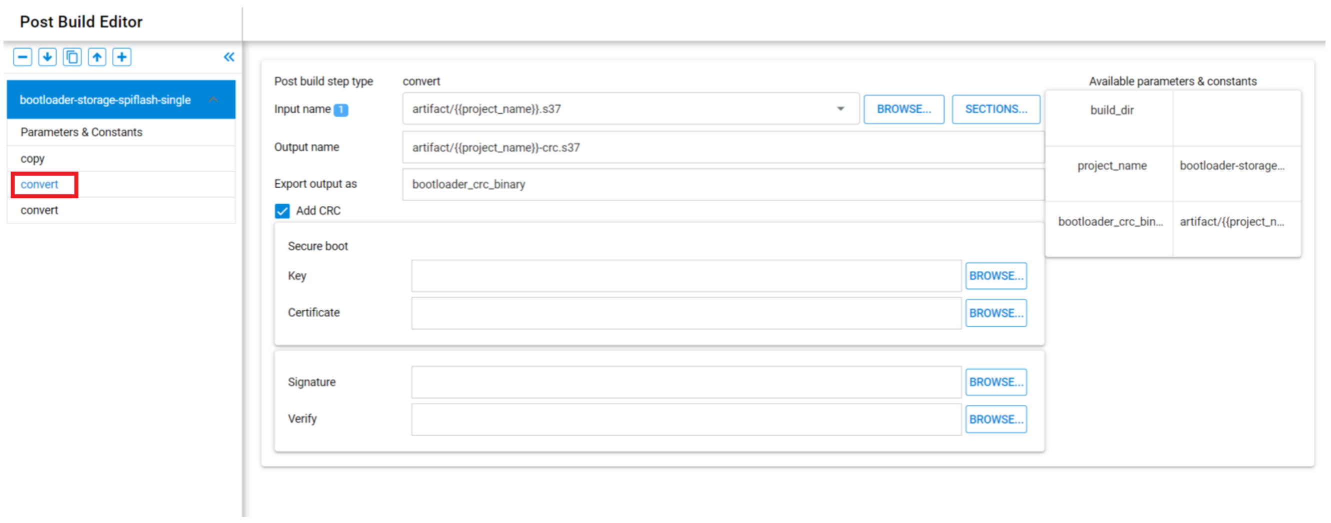
Available parameters & constants

build_dir	
project_name	bootloader-storage...
bootloader_main_s...	artifact/{{project_n...

- Required Arguments:
  - Input name: The project relative path and filename to copy.
  - Output name: The destination project relative path and filename. Folders will be created as needed.
- Optional Arguments:
  - Export output as: The parameter name to assign to the output file.

Convert

The `convert` task performs various actions on the binary to convert it to another form, such as adding a CRC or a signature. See the *Simplicity Commander User's Guide* 'Convert and Modify File Commands' section for details.



bootloader-storage-spiflash-single
- Parameters & Constants
- copy
- convert
- convert

Post build step type: convert
Input name: artifact/{{project_name}}.s37
Output name: artifact/{{project_name}}-crc.s37
Export output as: bootloader_crc_binary
[✓] Add CRC
Key:
Certificate:
Signature:
Verify:

Available parameters & constants

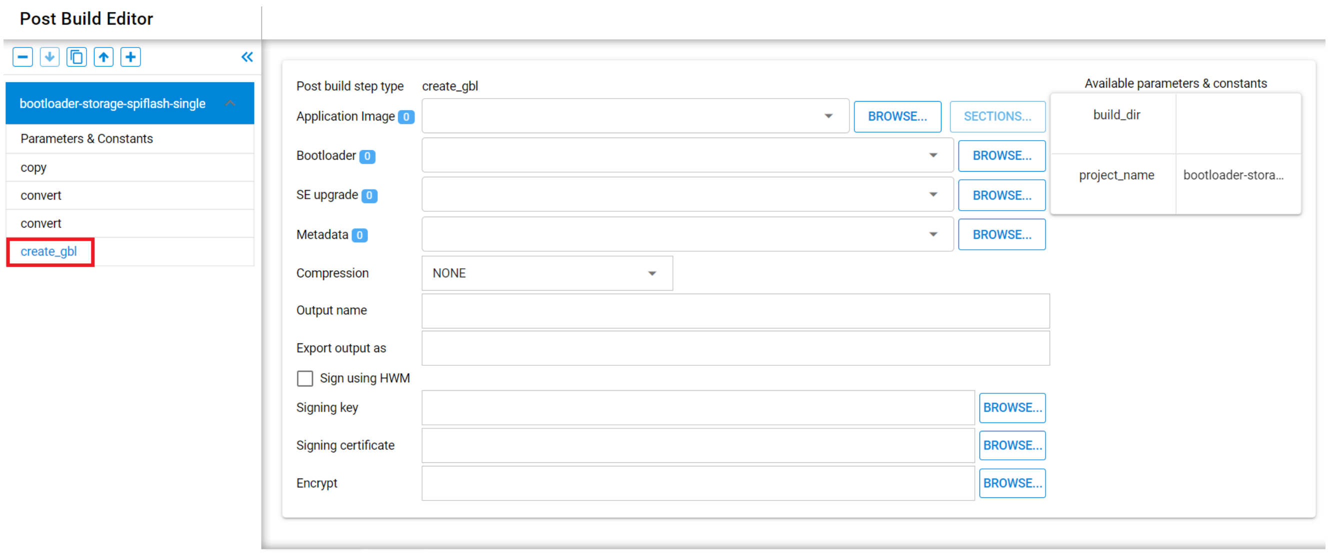
build_dir	
project_name	bootloader-storage...
bootloader_crc_bin...	artifact/{{project_n...

- Required Arguments:
  - Input name: The project relative path and filename to convert.
  - Output name: The destination project relative path and filename. Folders will be created as needed.
- Optional Arguments:
  - Export output as: The parameter name to assign to the output file.
  - Add CRC checkbox: To calculate and store the CRC32 on the bootloader image. The Gecko Bootloader can use the CRC to ensure image integrity when Secure Boot is not used.
  - Secure boot: Private Key and Intermediary Certificate to use to sign the secure boot image.
  - Signature file to sign the image
  - Include-section: <ELF section to include in image>
  - Exclude-section: <ELF section to exclude from image>
  - Verify: Key file

The `Include-section` and `Exclude-section` optional arguments can be used if the input file is an Executable and Linkable Format (ELF) file. If neither the Include-section nor the Exclude-section arguments are used, all .text sections will be extracted, as well as sections of type `progbits` with address not equal to 0x0.

Create_gbl

The `create_gbl` task creates a Gecko Bootloader file (GBL) from an application and/or bootloader image that optionally includes a Secure Element (SE) upgrade image. See the *Simplicity Commander User's Guide* "GBL Commands" section for details.



Post build step type: create_gbl
Application Image 0
Bootloader 0
SE upgrade 0
Metadata 0
Compression: NONE
Output name
Export output as
Sign using HWM
Signing key
Signing certificate
Encrypt

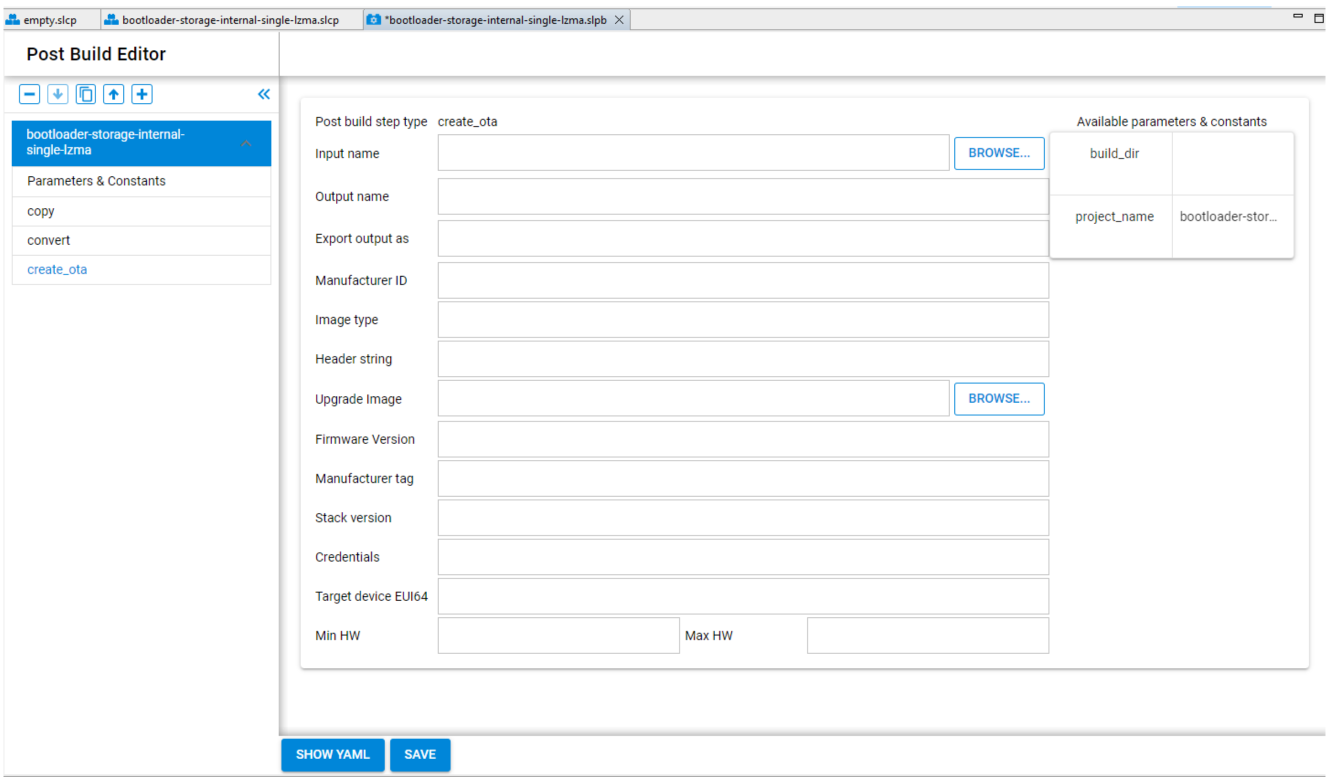
build_dir	
project_name	bootloader-stora...

- Required Arguments:
  - Output name: The destination project relative path and filename. Folders will be created as needed.
- Optional Arguments:
  - Export output as: The parameter name to assign to the output file.
  - Application Image: Application image file.
  - Bootloader: Bootloader image file.
  - SE upgrade: Secure Element upgrade image file.
  - Metadata: Metadata bin file to include in the image.
  - Compression: Compression method for output (none, lzma, lz4).
  - Sign using HWM checkbox: Check if the signature will be created using Hardware Security Module.
  - Signing key
  - Signing certificate
  - Encrypt: Encryption key used to encrypt image.
  - Include-section: <ELF section to include in image>.
  - Exclude-section: <ELF section to exclude from image>.

Create_ota

The `create_ota` task creates a Zigbee Over-the-air (OTA) bootloader file from one or more GBL files. See the *Simplicity Commander User's Guide* "OTA Commands" section for details.

NOTE: The "Input name" field should be ignored; it will be removed in a future release. The "Input name" is a duplicate of the "Upgrade Image". Using both will add two copies of the file to the output.



empty.slcp | bootloader-storage-internal-single-lzma.slcp | *bootloader-storage-internal-single-lzma.slpb

Post Build Editor

- bootloader-storage-internal-single-lzma
- Parameters & Constants
- copy
- convert
- create_ota

Post build step type: create_ota

Input name	
Output name	
Export output as	
Manufacturer ID	
Image type	
Header string	
Upgrade Image	
Firmware Version	
Manufacturer tag	
Stack version	
Credentials	
Target device EUI64	
Min HW	
Max HW	

Available parameters & constants

build_dir	
project_name	bootloader-stor...

- Required Arguments:
  - Upgrade Image: The project relative path of the gbl file.
  - Output name: The destination project relative path and filename. Folders will be created as needed.
  - Manufacturer ID
  - Image type
  - Header string
  - Firmware Version
- Optional Arguments:
  - Export output as: The parameter name to assign to the output file
  - Manufacturer tag: Tag ID.
  - Stack version
  - Credentials
  - Target device EUI64
  - Minimum HW version
  - Maximum HW version

Example

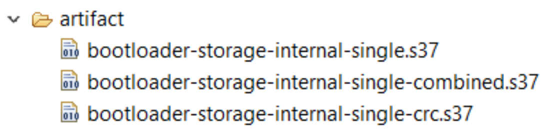
Series 1 Combined Bootloader

An example of the PBE can be seen by creating a Series 1 bootloader for an EFR32MG12 target part. The PBE file copies the output .s37 of the main bootloader and adds a CRC to it. Then it copies the pre-built first stage bootloader image into a combined bootloader file that can be programmed to the device. To create this example:

1. Add BRD4161A to the Launcher Perspective [My Products] window and select it.
2. Make sure GSDK 4.2.0 or later is the Preferred SDK.
3. Click the Example Projects and Demos tab and click the Bootloaders filter checkbox
4. Click [Create] for “Bootloader – SoC Internal Storage (single image on 1MB device)
5. Build the project. In the build console output will be a call to commander.exe with the postbuild argument and the .slpb file:

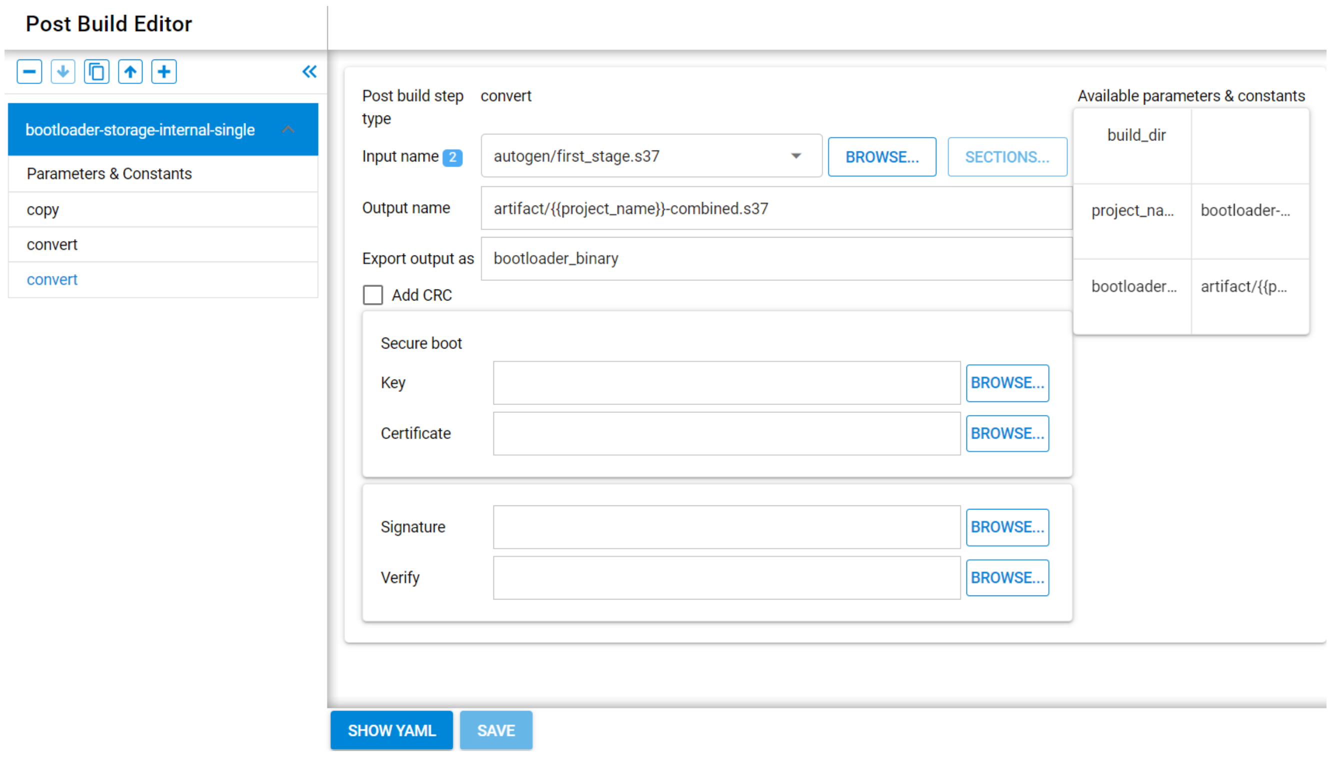
```
"C:\SiliconLabs\SimplicityStudio\v5\developer\adapter_packs\commander\commander.exe" postbuild
"C:\Users\USERNAME\SimplicityStudio\v5_workspace\bootloader-storage-internal-single/bootloader-storage-internal-single.slpb" --parameter
build_dir:"C:\Users\USERNAME\SimplicityStudio\v5_workspace\bootloader-storage-internal-single\GNU ARM v10.3.1 - Default"
Parsing file C:\Users\japitt\SimplicityStudio\v5.rel.Staging_1865\bootloader-storage-internal-single/bootloader-storage-internal-single.slpb...
Running task copy...
Running task convert...
Running task convert...
DONE
```

The 'artifact' folder is created with three files in it:



- artifact
  - bootloader-storage-internal-single.s37
  - bootloader-storage-internal-single-combined.s37
  - bootloader-storage-internal-single-crc.s37

Opening the .slpb file shows three operations with the last one being a convert that takes the 'autogen/first_stage.s37' file and adds it to the main bootloader image in a new file called 'bootloader-storage-internal-single-combined.s37.



Post Build Editor

- bootloader-storage-internal-single
  - Parameters & Constants
  - copy
  - convert
  - convert

Post build step type	convert
Input name 2	autogen/first_stage.s37
Output name	artifact/{{project_name}}-combined.s37
Export output as	bootloader_binary
Add CRC	☐

Secure boot

Key	
Certificate	
Signature	
Verify	

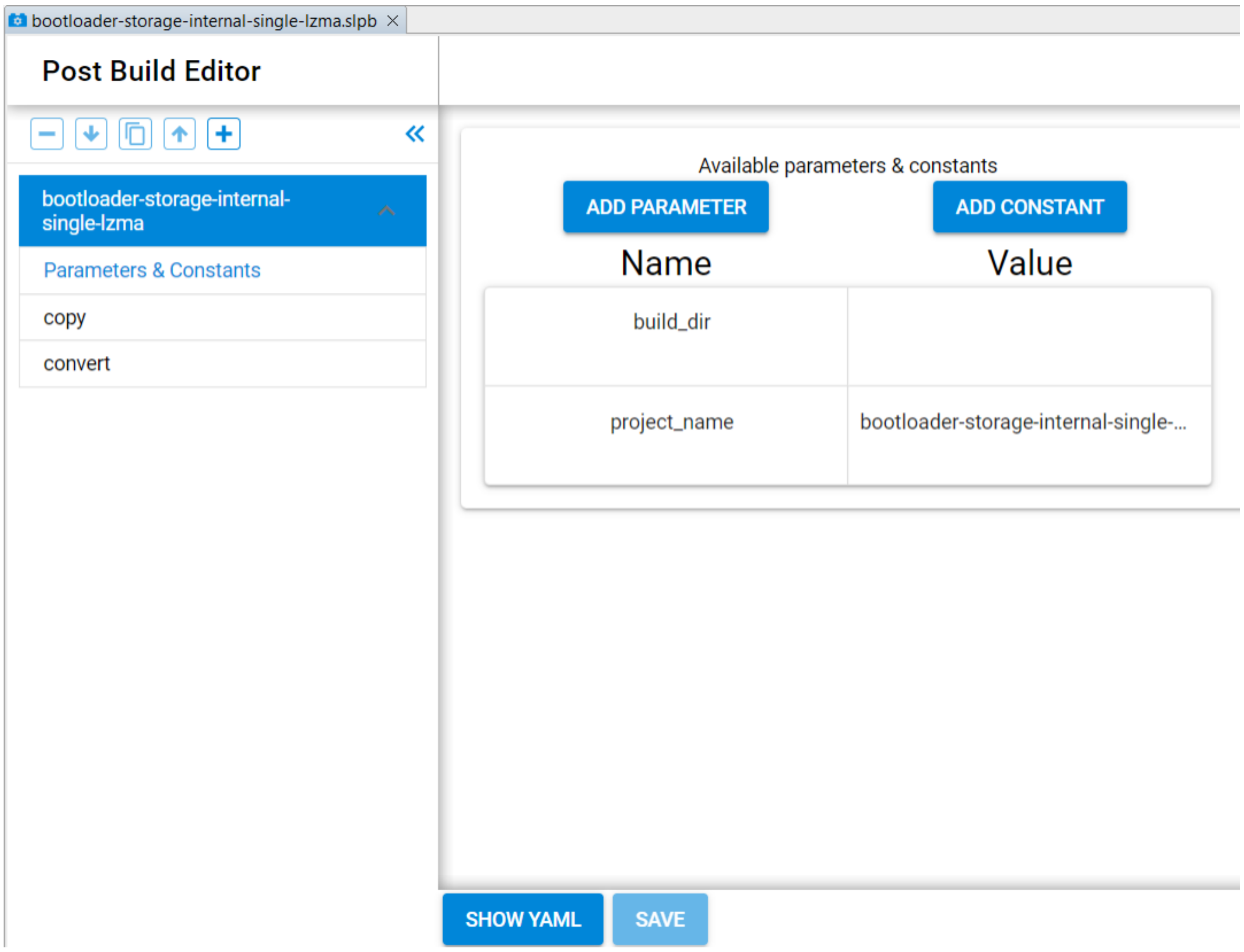
Available parameters & constants

build_dir	
project_na...	bootloader-...
bootloader...	artifact/{{p...

Series 2 OTA Image Creation

The following example was created using an EFR32MG24B210F1536IM48 target on a BRD4186C radio board. It includes all four tasks and can be created with the following steps:

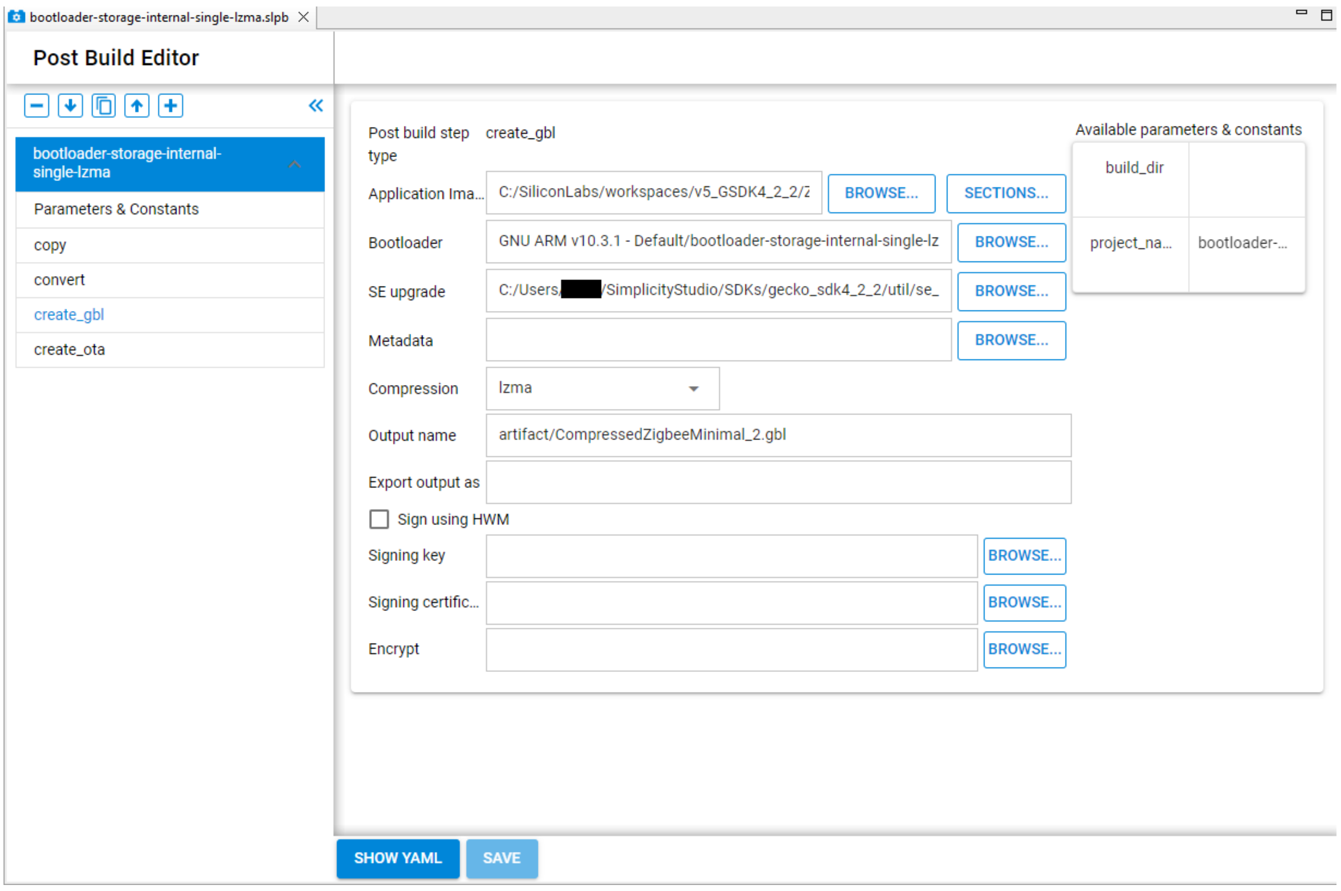
1. Create and build a "Bootloader - SoC Internal Storage single image with LZMA compression, 1MB Flash" (bootloader-storage-internal-single-lzma) project.
2. Create and build a "Zigbee SoC ZigbeeMinimal" (ZigbeeMinimal) project.
3. Open the bootloader .slpb file in the PBE. It will have copy and convert tasks already defined:


bootloader-storage-internal-single-lzma.slpb
Post Build Editor
bootloader-storage-internal-single-lzma
Parameters & Constants
copy
convert
Available parameters & constants
ADD PARAMETER
ADD CONSTANT
SHOW YAML
SAVE

Name	Value
build_dir	
project_name	bootloader-storage-internal-single-...

Add a Create_gbl Task

1. Select the Convert task and then click the "+" icon and select "create_gbl".
2. Next to 'Application Image', click **BROWSE** and select the ZigbeeMinimal.s37 file. Multiple application images could be entered in this field if needed.
3. Next to 'Bootloader', click **BROWSE** and select the bootloader .s37 file.
4. Next to 'SE Upgrade', click **BROWSE** and browse to the Secure Element upgrade image file "s2c4_se_fw_upgrade_2v2p1.seu".
5. Click the 'Compression' dropdown and select "lzma".
6. Click the 'Output name' field and enter "artifact/CompressedZigbeeMinimal.gbl".
7. Click **SAVE**. The GBL task should look like this:


bootloader-storage-internal-single-lzma.slpb

Post Build Editor

- bootloader-storage-internal-single-lzma
- Parameters & Constants
- copy
- convert
- create_gbl
- create_ota

Post build step type: create_gbl

Application Ima...: C:/SiliconLabs/workspaces/v5_GSDK4_2_2/Z

Bootloader: GNU ARM v10.3.1 - Default/bootloader-storage-internal-single-lz

SE upgrade: C:/Users/[redacted]/SimplicityStudio/SDKs/gecko_sdk4_2_2/util/se_

Metadata:

Compression: lzma

Output name: artifact/CompressedZigbeeMinimal_2.gbl

Export output as:

☐ Sign using HWM

Signing key:

Signing certific...:

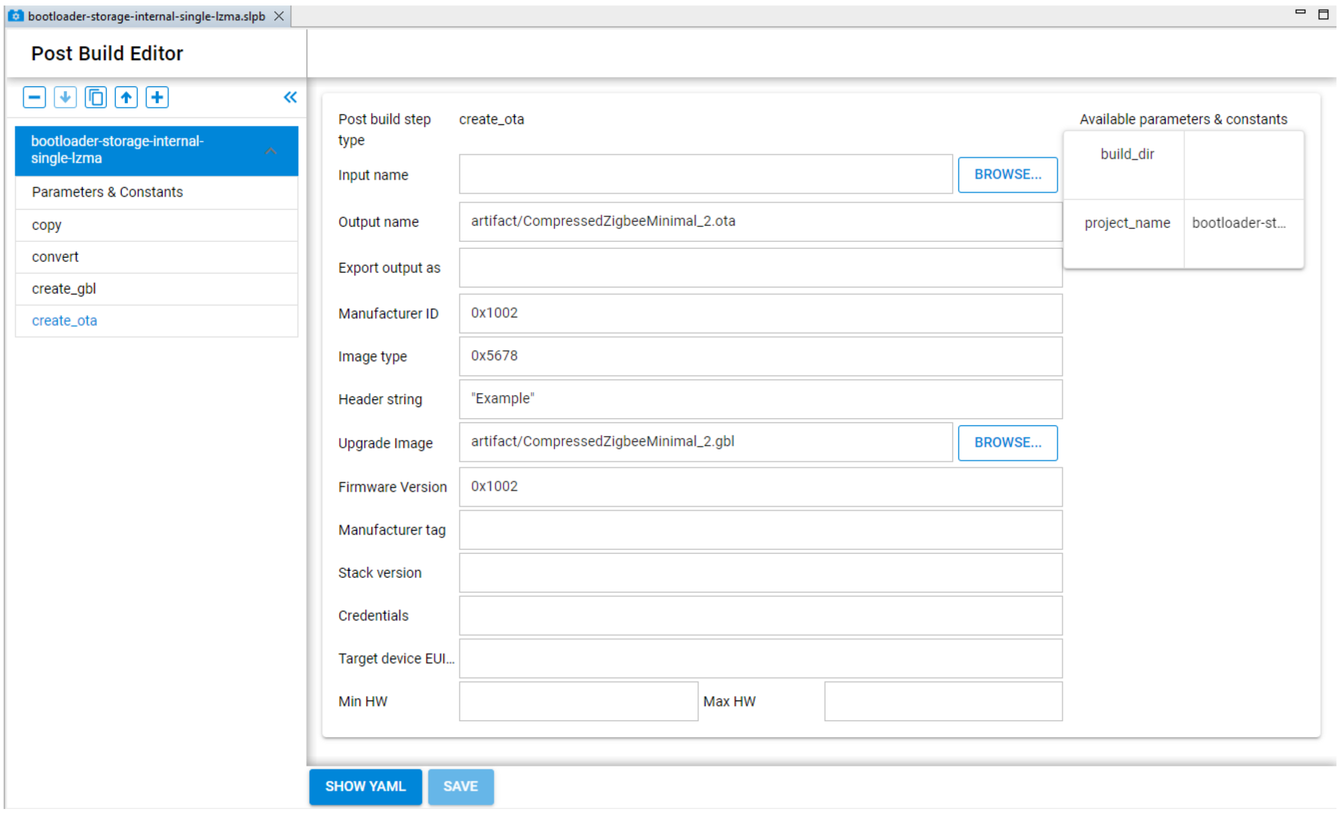
Encrypt:

Available parameters & constants

build_dir	
project_na...	bootloader-...

Add a Create_ota Task

1. Click the "+" icon and select "create_ota".
2. Click the 'Output name' field and enter "artifact/CompressedZigbeeMinimal.ota".
3. Click the 'Manufacturer ID' field and enter your manufacturer ID. A dummy value of 0x1002 is used in this example.
4. Click the 'Image type' field and enter a hex number for the image type. A dummy value of 0x5678 is used in this example.
5. Click the 'Header string' field and enter "Example".
6. Click the 'Upgrade Image' field and enter the path and .gbl filename from the create_gbl task - artifact/CompressedZigbeeMinimal.gbl. Note multiple .gbl files could be entered in this field if needed.
7. Click the 'Firmware Version' field and enter an unsigned integer value for the version. A dummy value of 0x1002 is used in this example.
8. Click **SAVE**. The OTA task should look like this:



Post build step type	create_ota
Input name	
Output name	artifact/CompressedZigbeeMinimal_2.ota
Export output as	
Manufacturer ID	0x1002
Image type	0x5678
Header string	"Example"
Upgrade Image	artifact/CompressedZigbeeMinimal_2.gbl
Firmware Version	0x1002
Manufacturer tag	
Stack version	
Credentials	
Target device EUI...	
Min HW	
Max HW	

Available parameters & constants

build_dir	
project_name	bootloader-st...

Test the Post-Build File

Build the project and at the end of the build commander will be called with the .slpb file and each of the tasks will be run:

```
"C:\SiliconLabs\SimplicityStudio\v5\developer\adapter_packs\commander\commander.exe" postbuild
"C:\SiliconLabs\workspaces\v5_workspace\bootloader-storage-internal-single-lzma/bootloader-storage-internal-single-lzma.slpb" --parameter
build_dir:"C:\SiliconLabs\workspaces\v5_workspace\bootloader-storage-internal-single-lzma\GNU ARM v10.3.1 - Default"
Parsing file C:\SiliconLabs\workspaces\v5_workspace\bootloader-storage-internal-single-lzma/bootloader-storage-internal-single-lzma.slpb...
Running task copy...
Running task convert...
Running task GBL create...
Running task OTA create...
DONE
```

Flashing

Flashing

Simplicity Studio® 5 (SSv5) offers several ways to load (flash) a firmware image to your device.

- With the **Debug** button (see Debug build and flash)
- Through the debug adapter's context menu, Upload Application... selection
- Using the Flash Programmer tool

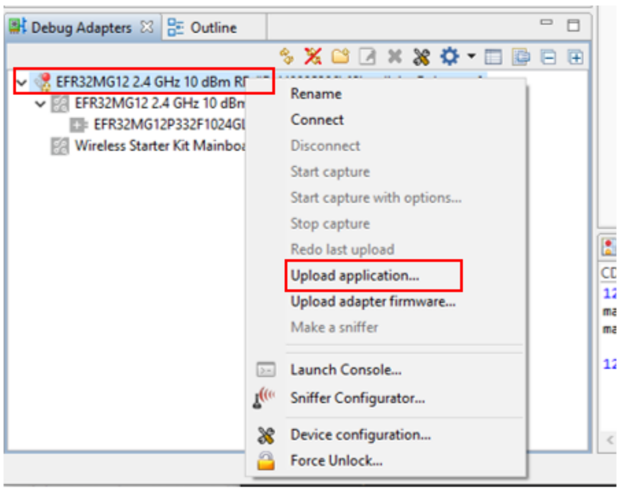
See your SDK's quick start guide for more stack-specific information about building and flashing.

Upload Application

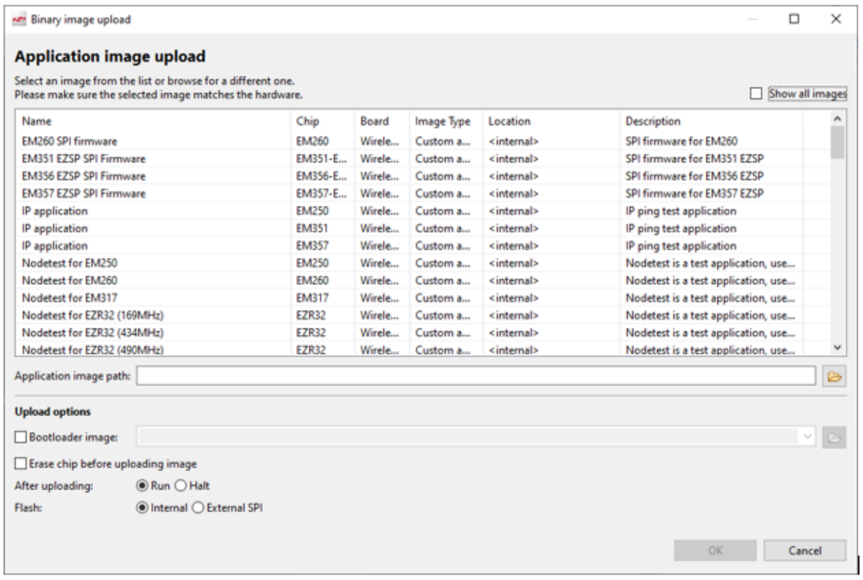
This option is useful if you want to load a bootloader image and an application image in a single step. To use this option, you will need to know the radio board part number (shown in the Debug Adapters view), and the location and name of the binary image you wish to load. The default workspace locations are:

Windows 10 workspace: C:\Users\<user>\SimplicityStudio Mac workspace: /Users/<user>/SimplicityStudio

In the Debug Adapters view, right-click the adapter (top line) and select **Upload Application...** from the debug adapter context menu.



The Application Image Upload dialog is displayed.



Binary image upload

Application image upload

Select an image from the list or browse for a different one.
Please make sure the selected image matches the hardware.

☐ Show all images

Name	Chip	Board	Image Type	Location	Description
EM260 SPI firmware	EM260	Wirele...	Custom a...	<internal>	SPI firmware for EM260
EM351 EZSP SPI Firmware	EM351-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM351 EZSP
EM356 EZSP SPI Firmware	EM356-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM356 EZSP
EM357 EZSP SPI Firmware	EM357-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM357 EZSP
IP application	EM250	Wirele...	Custom a...	<internal>	IP ping test application
IP application	EM351	Wirele...	Custom a...	<internal>	IP ping test application
IP application	EM357	Wirele...	Custom a...	<internal>	IP ping test application
Nodetest for EM250	EM250	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM260	EM260	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM317	EM317	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (169MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (434MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (490MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...

Application image path:

Upload options

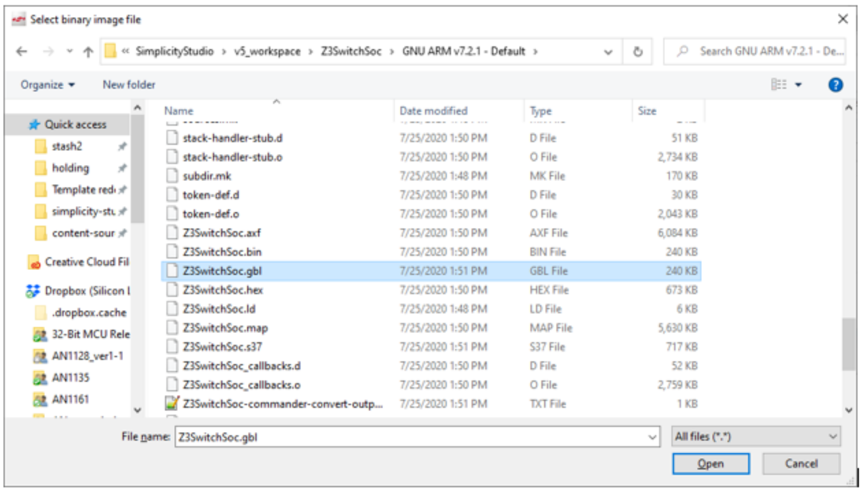
☐ Bootloader image:

☐ Erase chip before uploading image

After uploading: [✓ Run] Halt

Flash: [✓ Internal] External SPI

Browse to the project directory, go to the directory corresponding to the compiler toolchain, and select an image file. This example uses the .GBL file, as it assumes you are also loading a Gecko Bootloader image. (Start with UG103.6: Bootloader Fundamentals if you are not familiar with bootloaders.) Click **Open**.



Select binary image file

« SimplicityStudio › v5_workspace › Z3SwitchSoc › GNU ARM v7.2.1 - Default ›

Name	Date modified	Type	Size
stack-handler-stub.d	7/25/2020 1:50 PM	D File	51 KB
stack-handler-stub.o	7/25/2020 1:50 PM	O File	2,734 KB
subdir.mk	7/25/2020 1:48 PM	MK File	170 KB
token-def.d	7/25/2020 1:50 PM	D File	30 KB
token-def.o	7/25/2020 1:50 PM	O File	2,043 KB
Z3SwitchSoc.axf	7/25/2020 1:50 PM	AXF File	6,084 KB
Z3SwitchSoc.bin	7/25/2020 1:50 PM	BIN File	240 KB
Z3SwitchSoc.gbl	7/25/2020 1:51 PM	GBL File	240 KB
Z3SwitchSoc.hex	7/25/2020 1:50 PM	HEX File	673 KB
Z3SwitchSoc.ld	7/25/2020 1:48 PM	LD File	6 KB
Z3SwitchSoc.map	7/25/2020 1:50 PM	MAP File	5,630 KB
Z3SwitchSoc.s37	7/25/2020 1:51 PM	S37 File	717 KB
Z3SwitchSoc_callbacks.d	7/25/2020 1:50 PM	D File	52 KB
Z3SwitchSoc_callbacks.o	7/25/2020 1:50 PM	O File	2,759 KB
Z3SwitchSoc-commander-convert-outp...	7/25/2020 1:51 PM	TXT File	1 KB

File name: Z3SwitchSoc.gbl

All files (*.*)

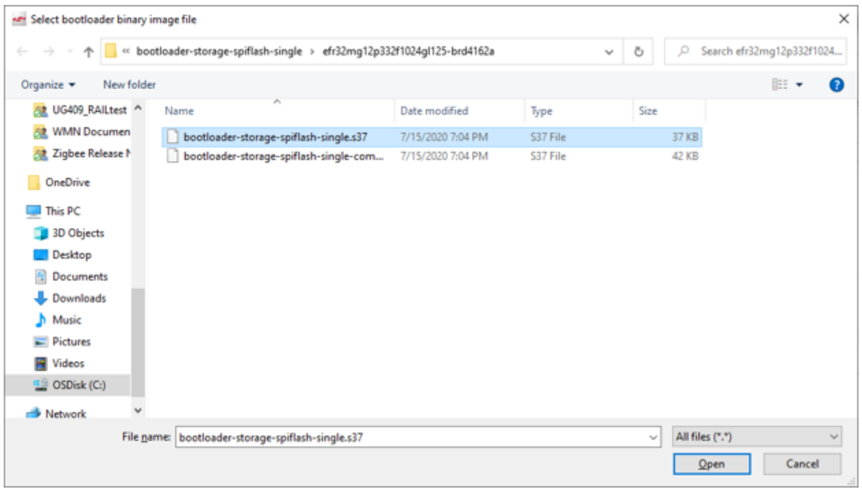
Check **Bootloader image**, then browse to the folder containing a prebuilt bootloader image. Images are located in the Simplicity Studio platform bootloader folder under sample apps. In this case open the SPI Flash Single folder, for example:

C:\SiliconLabs\SimplicityStudio\<version>\developer\sdks\gecko_sdk_suite\<version>\platform\bootloader\sample-apps\bootloader-storage-spiflash-single

Open the folder that corresponds to your board and part number and select the .s37 file, for example:

efr32mg12p332f1024gl125-brd4162a\bootloader-storage-spiflash-single-combined.s37

Click **Open**.



Select bootloader binary image file

« bootloader-storage-spiflash-single › efr32mg12p332f1024gl125-brd4162a

Search efr32mg12p332f1024...

Name	Date modified	Type	Size
bootloader-storage-spiflash-single.s37	7/15/2020 7:04 PM	S37 File	37 KB
bootloader-storage-spiflash-single-com...	7/15/2020 7:04 PM	S37 File	42 KB

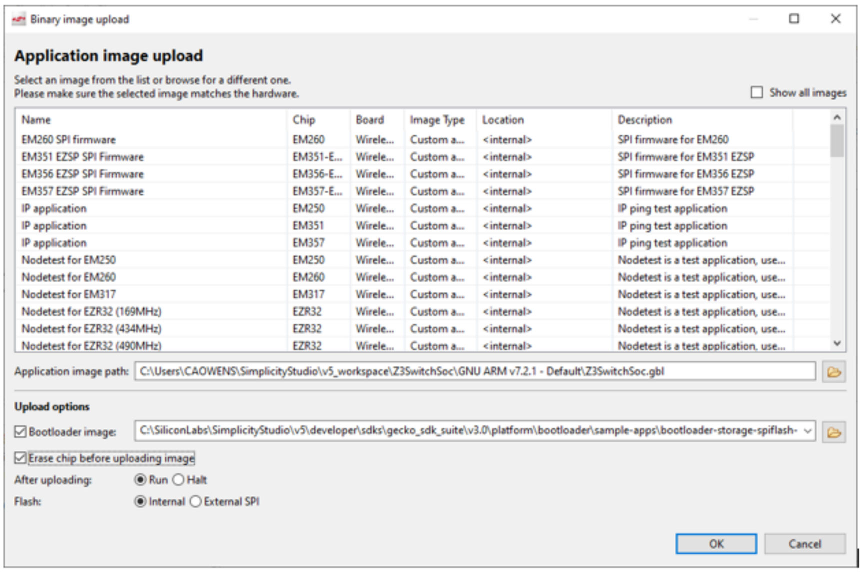
File name: bootloader-storage-spiflash-single.s37

All files (*.*)

Now that both the application image and the bootloader are selected, check **Erase Chip**, to make sure that the main flash block is erased before your new image is uploaded. New users will typically always check this.

- The **After uploading** options are **Run** (begin executing the code immediately) and **Halt** (wait for an event, such as a debugger to connect or manual initiation of a boot sequence). During initial development you will typically leave this set to **Run**.
- The **Flash** options determine the storage location, and are Internal and External SPI. Leave the option set to **Internal**.

The completed dialog should resemble the following:



Binary image upload

Application image upload

Select an image from the list or browse for a different one.
Please make sure the selected image matches the hardware.

☐ Show all images

Name	Chip	Board	Image Type	Location	Description
EM260 SPI firmware	EM260	Wirele...	Custom a...	<internal>	SPI firmware for EM260
EM351 EZSP SPI Firmware	EM351-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM351 EZSP
EM356 EZSP SPI Firmware	EM356-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM356 EZSP
EM357 EZSP SPI Firmware	EM357-E...	Wirele...	Custom a...	<internal>	SPI firmware for EM357 EZSP
IP application	EM250	Wirele...	Custom a...	<internal>	IP ping test application
IP application	EM351	Wirele...	Custom a...	<internal>	IP ping test application
IP application	EM357	Wirele...	Custom a...	<internal>	IP ping test application
Nodetest for EM250	EM250	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM260	EM260	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EM317	EM317	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (169MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (434MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...
Nodetest for EZR32 (490MHz)	EZR32	Wirele...	Custom a...	<internal>	Nodetest is a test application, use...

Application image path: C:\Users\CAOWENS\SimplicityStudio\v5_workspace\Z3SwitchSoc\GNU ARM v7.2.1 - Default\Z3SwitchSoc.gbl

Upload options

☑ Bootloader image: C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.0\platform\bootloader\sample-apps\bootloader-storage-spiflash-

☑ Erase chip before uploading image

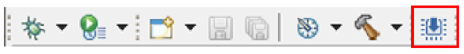
After uploading: [✓ Run] Halt

Flash: [✓ Internal] External SPI

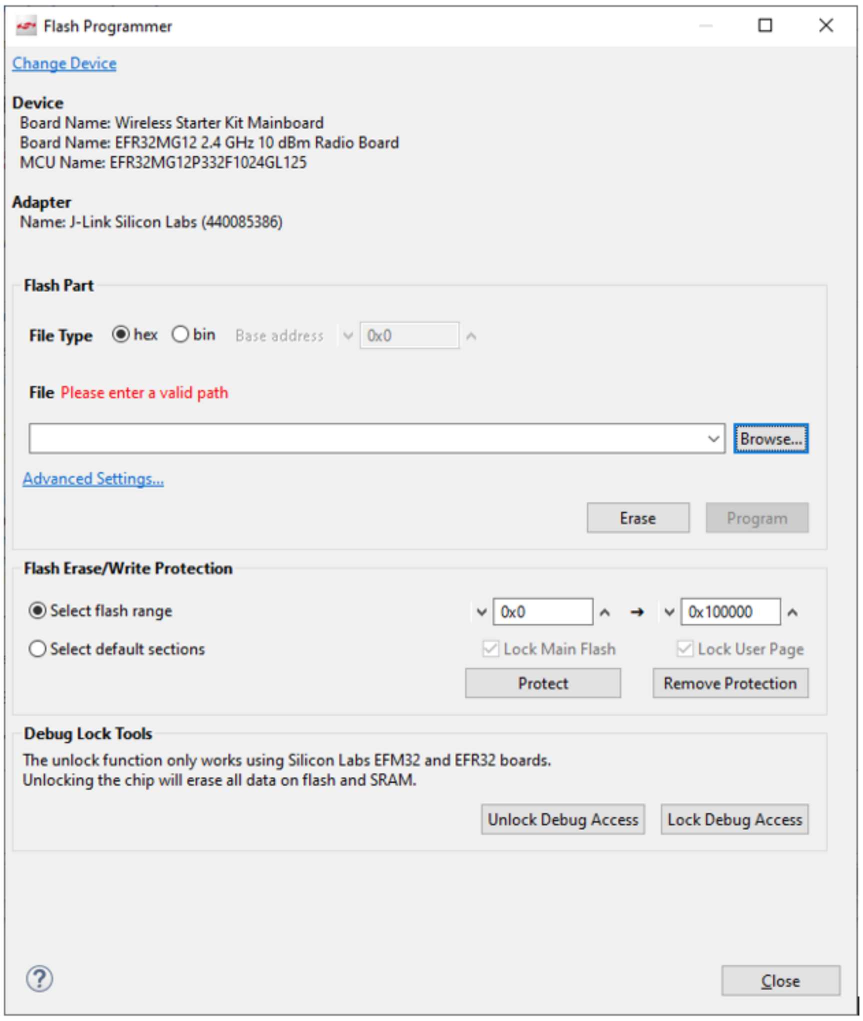
Click **OK**.

Flash Programmer

The Flash Programmer is a tool that provides a number of options for use when flashing images to a device. Select the Flash Programmer from the **Tools** button on the toolbar. Some perspectives have a button specifically for the Flash Programmer.



The Flash Programmer provides basic flash and erase functionality, but also allows you to lock parts of memory and enable or disable debug access.



Flash Programmer

Change Device

Device
Board Name: Wireless Starter Kit Mainboard
Board Name: EFR32MG12 2.4 GHz 10 dBm Radio Board
MCU Name: EFR32MG12P332F1024GL125

Adapter
Name: J-Link Silicon Labs (440085386)

Flash Part

File Type [✓ hex] bin Base address 0x0

File Please enter a valid path

Advanced Settings...

Flash Erase/Write Protection

[✓ Select flash range] 0x0 → 0x100000

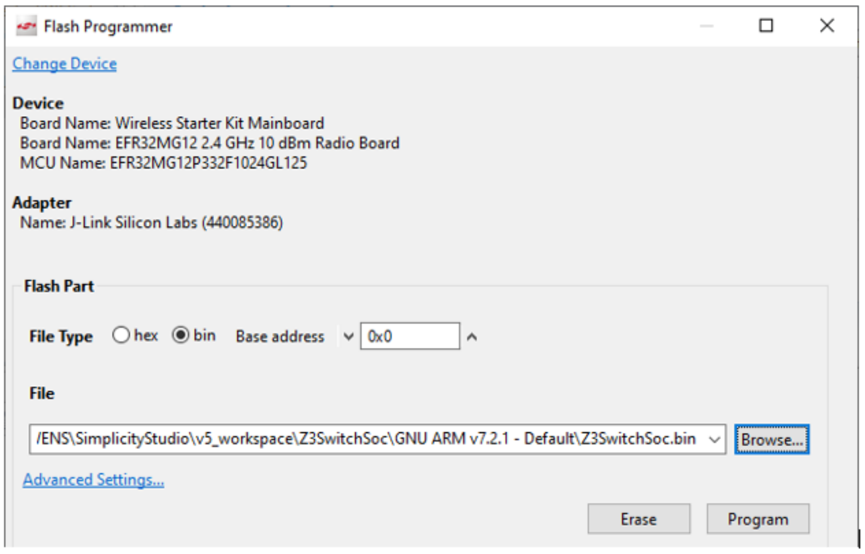
Select default sections Lock Main Flash Lock User Page

Debug Lock Tools

The unlock function only works using Silicon Labs EFM32 and EFR32 boards.
Unlocking the chip will erase all data on flash and SRAM.

Flash Part

The Flash Programmer is configured to facilitate flashing .hex or .bin files.



Flash Programmer

Change Device

Device
Board Name: Wireless Starter Kit Mainboard
Board Name: EFR32MG12 2.4 GHz 10 dBm Radio Board
MCU Name: EFR32MG12P332F1024GL125

Adapter
Name: J-Link Silicon Labs (440085386)

Flash Part

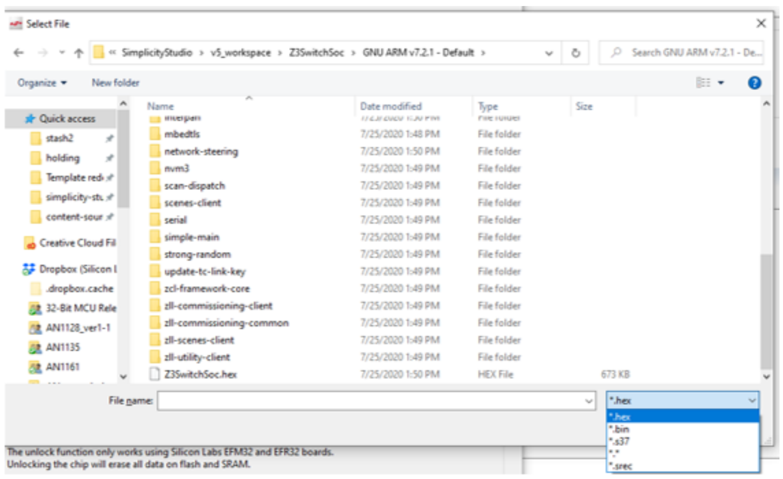
File Type hex [✓ bin] Base address 0x0

File

/ENS\SimplicityStudio\v5_workspace\Z3SwitchSoc\GNU ARM v7.2.1 - Default\Z3SwitchSoc.bin

Advanced Settings...

To flash an image, browse to the image location. Note that the images you see are filtered by the extension drop-down to the right of the file name field.


Select File

SimplicityStudio › v5_workspace › Z3SwitchSoc › GNU ARM v7.2.1 - Default ›

Name	Date modified	Type	Size
mbedtls	7/25/2020 1:48 PM	File folder	
network-steering	7/25/2020 1:50 PM	File folder	
nvm3	7/25/2020 1:49 PM	File folder	
scan-dispatch	7/25/2020 1:49 PM	File folder	
scenes-client	7/25/2020 1:49 PM	File folder	
serial	7/25/2020 1:49 PM	File folder	
simple-main	7/25/2020 1:49 PM	File folder	
strong-random	7/25/2020 1:49 PM	File folder	
update-tc-link-key	7/25/2020 1:49 PM	File folder	
zcl-framework-core	7/25/2020 1:49 PM	File folder	
zll-commissioning-client	7/25/2020 1:49 PM	File folder	
zll-commissioning-common	7/25/2020 1:49 PM	File folder	
zll-scenes-client	7/25/2020 1:49 PM	File folder	
zll-utility-client	7/25/2020 1:49 PM	File folder	
Z3SwitchSoc.hex	7/25/2020 1:50 PM	HEX File	673 KB

File name: [ ] *.hex (*.hex, *.bin, *.s37, *.*, *.srec)

The unlock function only works using Silicon Labs EFM32 and EFR32 boards.
Unlocking the chip will erase all data on flash and SRAM.

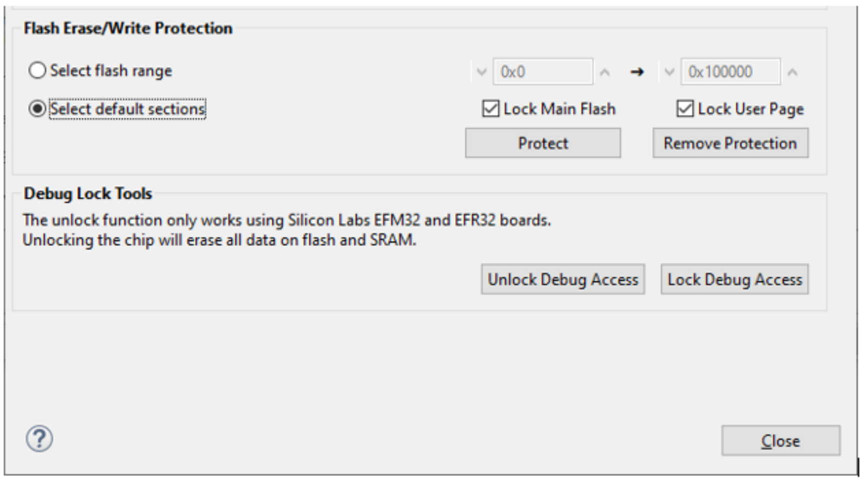
Click **Open** and then click **Program** to flash the image.

By default, the **Erase** function erases the main page. Click **Advanced Settings** to change what is erased.

Finally, if you have more than one device connected, click **Change Device** to select the target.

Flash Erase/Write Protection

Use these functions to protect or remove protection from a custom range, or default pages.


Flash Erase/Write Protection

( ) Select flash range   0x0 → 0x100000

(●) Select default sections   [✓] Lock Main Flash   [✓] Lock User Page

Debug Lock Tools

The unlock function only works using Silicon Labs EFM32 and EFR32 boards.
Unlocking the chip will erase all data on flash and SRAM.

Debug Lock Tools

Use these functions to unlock or lock debug access.

Testing and Debugging

Testing and Debugging

This section provides information on testing and debugging application firmware.

- Overview and Resources reviews the tools that are available and additional documentation.
- Using the Debugger provides details on Simplicity Studio's built-in debuggers.
- Using Wireshark describes how to connect a device to the Wireshark packet analyzer.

Overview and Resources

Resources

Simplicity Studio® 5 (SSv5) offers a number of tools that can be used to test and debug application functionality, power consumption, and network behavior.

Tools include:

- A built-in debugger, either the GNU Debugger (GDB) client and SEGGER's GDB server or a Simplicity Studio Debugger.
- Connectivity to the Wireshark packet analyzer.
- Network Analyzer
- Energy Profiler

In addition, a number of SDKs provide example applications that can be used for testing, such as:

- NodeTest (Zigbee EmberZNet SDK) (See AN1019: Using the NodeTest Application) - Deprecated in Zigbee EmberZNet SDK 6.10.0.0
- RAILtest (Proprietary Flex SDK) (see UG409: RAILtest User's Guide)

Finally, Silicon Labs provides a number of other documentation resources related to testing such as:

- AN1267: Radio Frequency Physical Layer Evaluation in Bluetooth® SDK v3.x
- AN1142: Mesh Network Performance Comparison
- AN718: Manufacturing Test Overview
- UG104: Testing and Debugging Applications for the Silicon Labs EM35x and EFR32MG Platforms (Zigbee)

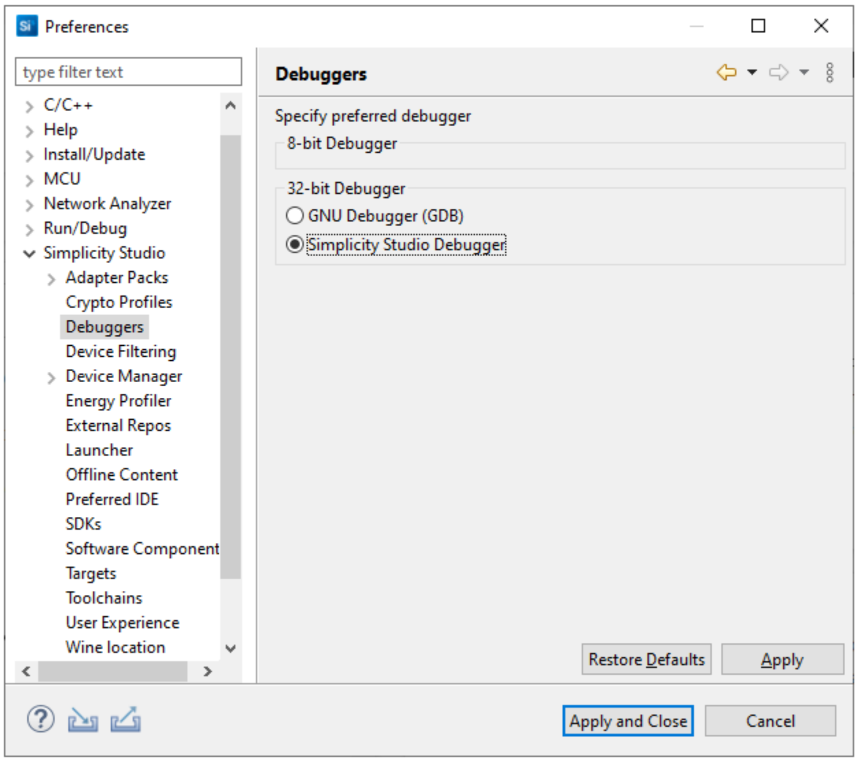
Using the Debugger

Using the Debuggers

Simplicity Studio® 5 (SSv5) supports two debuggers:

- For 32-bit device users, SSv5 includes support for a GNU Debugger (GDB) client and SEGGER's GDB server. The GNU Debugger (GDB) enables thread-aware debugging, and offers features such as unlimited flash breakpoints. This is the default debugger
- The classic Simplicity Studio debugger is a full featured debugger that offers the ability to step through code, set breakpoints, examine memory, variables and registers.

To change from the GNU debugger, on the toolbar click Preferences and select **Simplicity Studio > Debuggers**. Select **Simplicity Studio Debugger** and click **Apply and Close**.



Debuggers

Specify preferred debugger

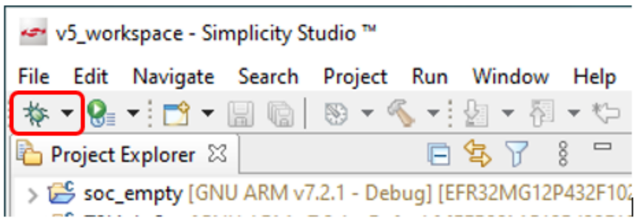
8-bit Debugger

32-bit Debugger

GNU Debugger (GDB) [✓ Simplicity Studio Debugger]

Silicon Labs supplies the debug models for each of their devices so that all of the registers can be examined and modified using either debugger.

The debugger runs in the Debug perspective. The debug perspective can be opened in a variety of ways, but most commonly by clicking the **Debug** button in the Simplicity IDE perspective once a project has reached a point where it will build and link correctly.


v5_workspace - Simplicity Studio ™
File
Edit
Navigate
Search
Project
Run
Window
Help
Project Explorer
soc_empty [GNU ARM v7.2.1 - Debug] [EFR32MG12P432F10

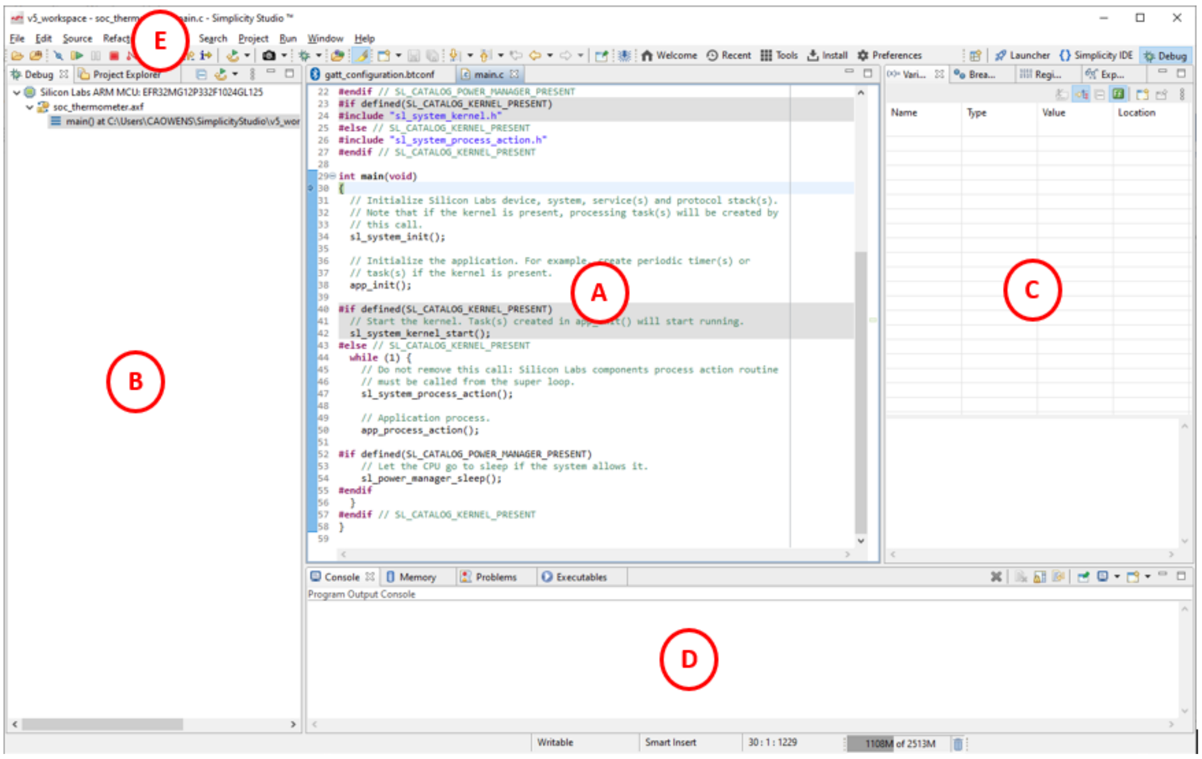
If a debug session has never been started before, a warning that a debug configuration does not exist might appear. Click the down arrow next to the debug icon, select **Debug as...** and then select either **Silicon Labs ARM Program** for 32-bit parts or **Silicon Labs 8051 Program** for EFM8 and C8051 parts.

If a single debug adapter is available, SSv5 automatically downloads the code to that device. If more than one debug adapter is available, SSv5 will prompt to select one.

With the Simplicity Studio debugger you can also save a snapshot of register values on a device, which can be used to compare the states of two different devices, or the state of the device at two different points in time.

Debug Perspective

The Debug Perspective by default is composed of the editor (A) and three tabbed view panes.

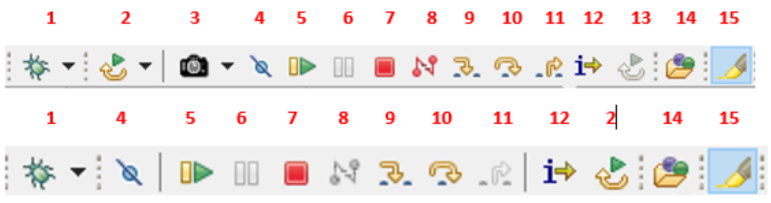


```
#endif // SL_CATALOG_POWER_MANAGER_PRESENT
#if defined(SL_CATALOG_KERNEL_PRESENT)
#include "sl_system_kernel.h"
#else // SL_CATALOG_KERNEL_PRESENT
#include "sl_system_process_action.h"
#endif // SL_CATALOG_KERNEL_PRESENT

int main(void)
{
  // Initialize Silicon Labs device, system, service(s) and protocol stack(s).
  // Note that if the kernel is present, processing task(s) will be created by
  // this call.
  sl_system_init();

  // Initialize the application. For example, create periodic timer(s) or
  // task(s) if the kernel is present.
  app_init();

#if defined(SL_CATALOG_KERNEL_PRESENT)
  // Start the kernel. Task(s) created in app_init() will start running.
  sl_system_kernel_start();
#else // SL_CATALOG_KERNEL_PRESENT
  while (1) {
    // Do not remove this call: Silicon Labs components process action routine
    // must be called from the super loop.
    sl_system_process_action();

    // Application process.
    app_process_action();

#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
    // Let the CPU go to sleep if the system allows it.
    sl_power_manager_sleep();
#endif
  }
#endif // SL_CATALOG_KERNEL_PRESENT
}
```

The Debug view (B) shows the call stack and call hierarchy that can help debug and find where functions are called. The Project Explorer view is available as a tab.

The right pane (C) and the bottom pane (D) contains tabbed views into data and consoles. These vary between the two debuggers. Depending on your actions, additional views may open in either the right or bottom pane.

Buttons in the Debug perspective toolbar (E) for the Simplicity Studio debugger and the GDB are shown below. The toolbar is similar for both debuggers.



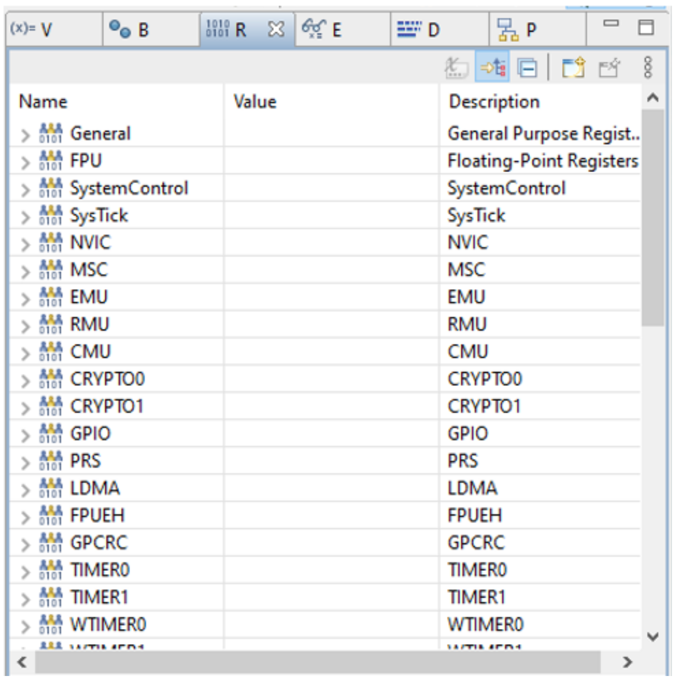
Number	Command	Description
1	Debug	Starts a new debug session. An active debug session must be disconnected before starting a new session using the same debug adapter.
2	Reset the device	Performs a hardware reset.
3	Create snapshot	(Simplicity Studio Debugger only) Creates a snapshot when clicked (default). Drop down the menu to select **Create when Stopped** or **Create at Breakpoints**. See About Snapshots for more information.
4	Skip	Skips all breakpoints.
5	Resume	Runs the application after reset or hitting a breakpoint.
6	Suspend	Halts the application.
7	Terminate	Terminates the session.
8	Disconnect	(Simplicity Studio Debugger only, gray for GDB) Terminates the session and disconnects the debug adapter. SSv5 automatically switches back to the Simplicity IDE perspective.
9	Step into	Single steps into the first line of a function.
10	Step over	Single steps over a function, executing the entire function.
11	Step return	Steps out of a function, executing the rest of the function.
12	Instruction Stepping Mode	Toggles assembly single stepping. When enabled, single stepping executes a single assembly instruction at a time. If clicked, opens a Disassembly view in the right pane that shows the assembly code corresponding to the source code at the current line of execution.
13	Restart	(Simplicity Studio debugger only) Restarts a process or target without terminating and relaunching
14	Open Element	Opens a dialog where you can see the qualified name and location for an element. Start typing to bring up a list of matching elements.
15	Toggle Mark Occurrences	Turns the mark occurrences function on and off.

A debug adapter can only support a single debug session at a time. An active debug session must be disconnected before code can be recompiled and a new debug session started.

To set a breakpoint, double-click in the blue bar to the left of the code editor or right-click on a line of code and select **Add Breakpoint**.

Simplicity Studio Debugger

This section describe the various views in the Simplicity Studio debugger.



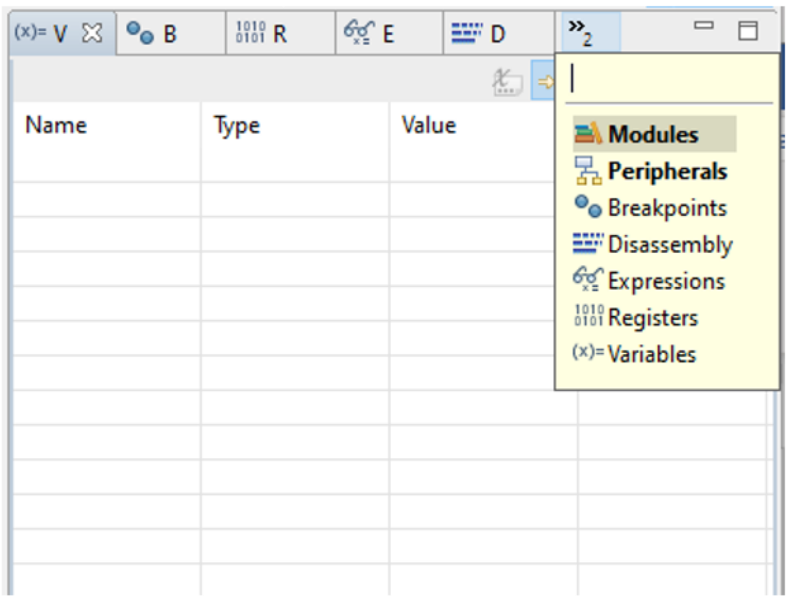
Name	Value	Description
General		General Purpose Regist..
FPU		Floating-Point Registers
SystemControl		SystemControl
SysTick		SysTick
NVIC		NVIC
MSC		MSC
EMU		EMU
RMU		RMU
CMU		CMU
CRYPTO0		CRYPTO0
CRYPTO1		CRYPTO1
GPIO		GPIO
PRS		PRS
LDMA		LDMA
FPUEH		FPUEH
GPCRC		GPCRC
TIMER0		TIMER0
TIMER1		TIMER1
WTIMER0		WTIMER0

- The **Variables** view shows the values of variables.
- The **Breakpoints** view provides functionality to manage breakpoints.
- The **Registers** view displays and allows you to change register contents. Registers with bitfields are split into individual fields with the enumeration value decoded. Registers that have changed between breakpoints are highlighted in yellow.
- The **Expressions** view allows you to add and change expressions. If possible, SSv5 evaluates the expression and displays its current value when the application is halted. If an expression is out of scope, SSv5 cannot evaluate the expression until the application is halted in a section of code where the expression is in scope.
- The **Disassembly** view provides an interleaved display of the original source code with the assembler instructions use to execute the source code.

The Memory view in the bottom pane displays the contents of the CODE, RAM, and XRAM memory. To experiment, in the Monitors menu click **+**. Select **Enter memory space and address**, select a memory space such as CORE and enter an address to view, such as 0x0000. Click **OK**. A memory monitor is added to the Memory view that displays the contents of the space starting at the address you provided.

GDB

This section describe the various views in the GNU Debugger.



- The **Variables** view shows the values of variables.
- The **Breakpoints** view provides functionality to manage breakpoints.
- The **Registers** view displays and allows you to change general register contents. Registers that have changed between breakpoints are highlighted in yellow. To see other registers, select a peripheral in the Peripherals view.
- The **Expressions** view allows you to add and change expressions. If possible, SSv5 evaluates the expression and displays its current value when the application is halted. If an expression is out of scope, SSv5 cannot evaluate the expression until the application is halted in a section of code where the expression is in scope.
- The **Disassembly** view provides an interleaved display of the original source code with the assembler instructions use to execute the source code.
- The **Peripherals** view, after a short delay, shows peripherals with bitfields that are split into individual fields shown as a hex value. Their values cannot be modified. When a peripheral is selected, the **Memory** tab in the bottom pane shows associated registers and their values.

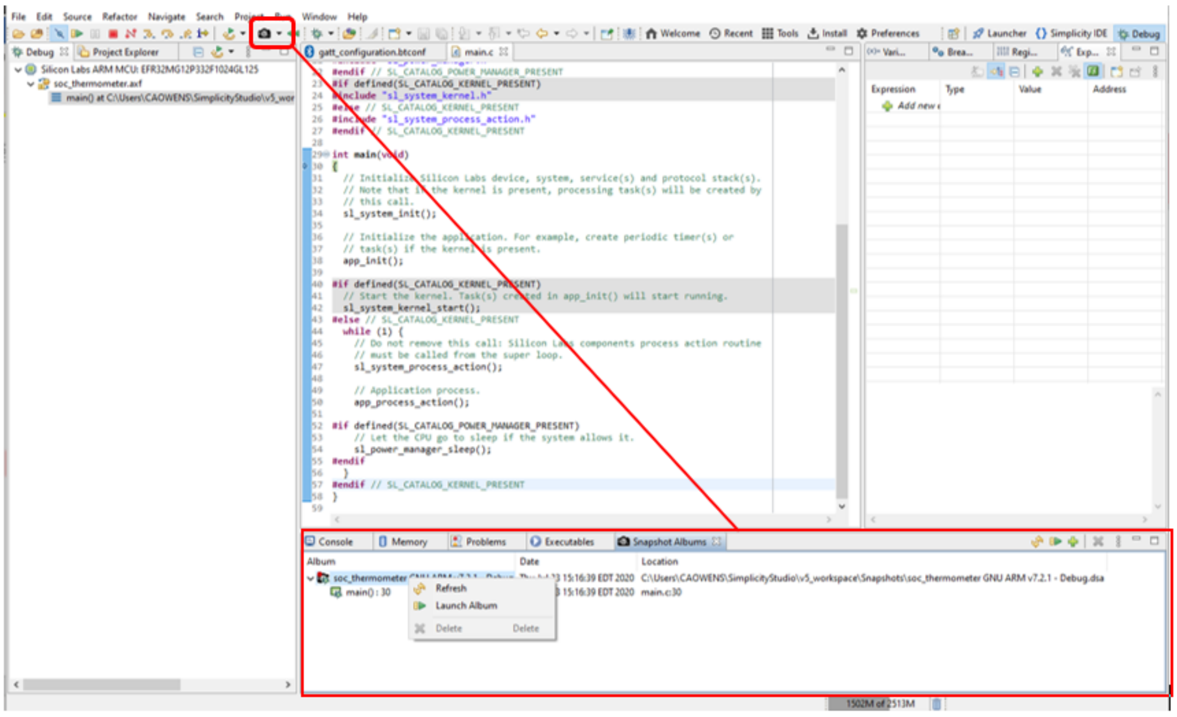
The Memory view in the bottom pane displays the contents of the CODE, RAM, and XRAM memory. To experiment, in the Monitors menu click **+** and enter an address or an expression to monitor. A monitor for that address or expression is added.

The bottom pane also includes a Debugger Console, with output specific to GDB Debug SEGGER J-Link Debugging.

About Snapshots

The Snapshot feature is available in the Simplicity Studio debugger only. A snapshot saves the values of the registers on a device at a particular point in time. This feature is useful when comparing the states of two systems next to each other or looking at the state of the hardware at separate points in time.

Once you take a snapshot, the Snapshot Album view opens in the bottom view pane.



```
#endif // SL_CATALOG_POWER_MANAGER_PRESENT
#if defined(SL_CATALOG_KERNEL_PRESENT)
#include "sl_system_kernel.h"
#else // SL_CATALOG_KERNEL_PRESENT
#include "sl_system_process_action.h"
#endif // SL_CATALOG_KERNEL_PRESENT

int main(void)
{
  // Initialize Silicon Labs device, system, service(s) and protocol stack(s).
  // Note that if the kernel is present, processing task(s) will be created by
  // this call.
  sl_system_init();

  // Initialize the application. For example, create periodic timer(s) or
  // task(s) if the kernel is present.
  app_init();

#if defined(SL_CATALOG_KERNEL_PRESENT)
  // Start the kernel. Task(s) created in app_init() will start running.
  sl_system_kernel_start();
#else // SL_CATALOG_KERNEL_PRESENT
  while (1) {
    // Do not remove this call: Silicon Labs components process action routine
    // must be called from the super loop.
    sl_system_process_action();

    // Application process.
    app_process_action();

#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
    // Let the CPU go to sleep if the system allows it.
    sl_power_manager_sleep();
#endif
  }
#endif // SL_CATALOG_KERNEL_PRESENT
}
```

Album	Date	Location
soc_thermometer GNU ARM v7.2.1 - Debug	Thu Jul 23 15:16:39 EDT 2020	C:\Users\CAOWENS\SimplicityStudio\v5_workspace\Snapshots\soc_thermometer GNU ARM v7.2.1 - Debug.dsa
main() : 30	3 15:16:39 EDT 2020	main.c:30

In the Debug perspective, if you do not see the Snapshot Album view, select **Window > Show View > Other > Snapshot Albums**.

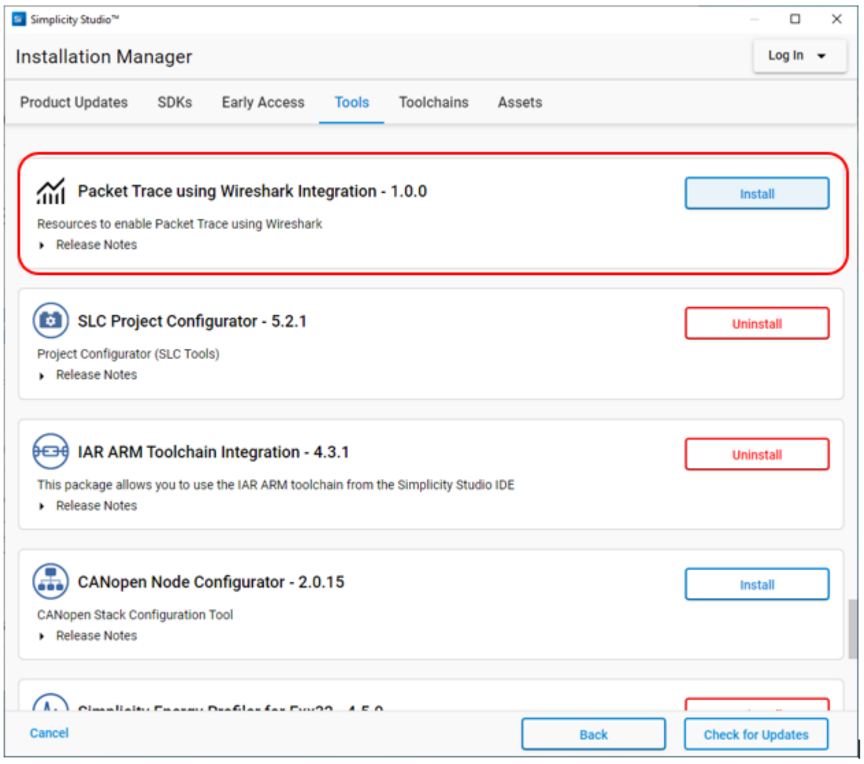
To view a snapshot, right-click a snapshot in the list and select **Launch Snapshot**. This adds a debug session in the Debug window as it was at the time of the snapshot, with a Snapshot label at the start of the debug configuration. However, it is not an active debug session on hardware. To switch back to a debug session on hardware, select the Silicon Labs ARM debug configuration in the Debug window if a hardware debug session was active or else select **Run > Debug** if a session is not active.

Using Wireshark

Using Wireshark

Beginning with version 5.5.0, Simplicity Studio® 5 (SSv5) supports live interaction between the application running on a Silicon Labs device and the Wireshark packet analyzer. Windows/Mac users can download Wireshark from https://www.wireshark.org/download.html. Linux users can find instructions at https://www.wireshark.org/docs/wsug_html_chunked/ChBuildInstallUnixInstallBins.html.

If you installed the GSDK in an earlier version of Simplicity Studio, click **Install** on the toolbar. Click **Manage Installed Packages** and go to the Tools tab. Find the "Packet Trace Using Wireshark Integration" tool and click **Install**.



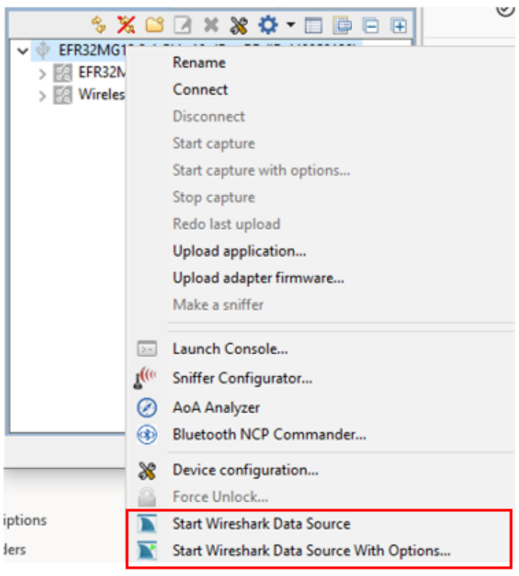
Packet Trace using Wireshark Integration - 1.0.0
Resources to enable Packet Trace using Wireshark
Release Notes

SLC Project Configurator - 5.2.1
Project Configurator (SLC Tools)
Release Notes

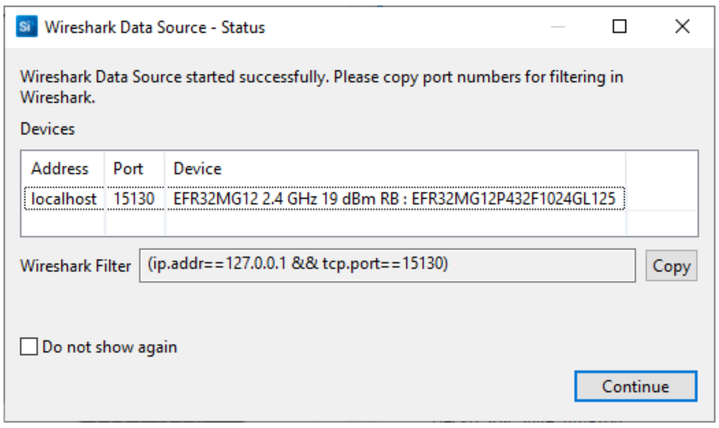
IAR ARM Toolchain Integration - 4.3.1
This package allows you to use the IAR ARM toolchain from the Simplicity Studio IDE
Release Notes

CANopen Node Configurator - 2.0.15
CANopen Stack Configuration Tool
Release Notes

1. Make sure that debug output is enabled in your project, if it is available. This is enabled by default in most GSDK examples and demos.
2. Build and flash the project to your device. If you are working in Simplicity Studio, do not start a debug session to build and flash the project, but rather build it and then flash the binary separately.
3. In Simplicity Studio's Debug Adapters view, select one or more devices and right-click one of the devices. Two Wireshark options are available, 'Start Wireshark Data Source' and 'Start Wireshark Data Source with Options'.



Start Wireshark Data Source: If all selected devices are connected through a network, the Wireshark Data Source starts. Devices connected by USB use the application Silink to communicate with the network. The network ports that Silink uses are assigned dynamically and are different on each launch. Knowing the port numbers can be useful when filtering captured packets by TCP port using Wireshark's display filters functionality. If you select at least one device connected by USB, a dialog is presented showing the device's IP address and port number.


Wireshark Data Source - Status

Wireshark Data Source started successfully. Please copy port numbers for filtering in Wireshark.

Devices

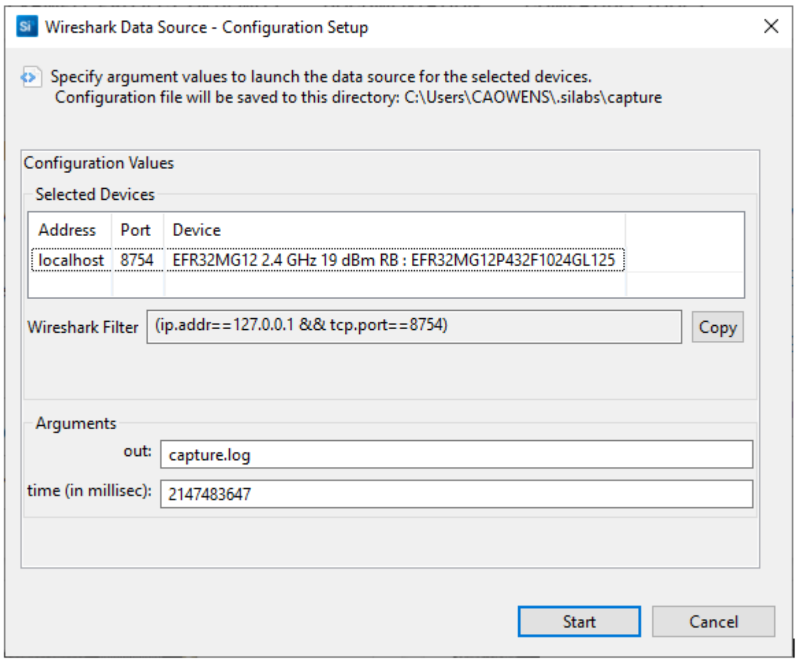
Address	Port	Device
localhost	15130	EFR32MG12 2.4 GHz 19 dBm RB : EFR32MG12P432F1024GL125

Wireshark Filter (ip.addr==127.0.0.1 && tcp.port==15130) Copy

☐ Do not show again

Continue

Start Wireshark Data Source with Options: A dialog is displayed in which you can configure the output log name and the capture duration in milliseconds. It also shows the IP address and port number.



Wireshark Data Source - Configuration Setup

Specify argument values to launch the data source for the selected devices.
Configuration file will be saved to this directory: C:\Users\CAOWENS\.silabs\capture

Configuration Values

Selected Devices

Address	Port	Device
localhost	8754	EFR32MG12 2.4 GHz 19 dBm RB : EFR32MG12P432F1024GL125

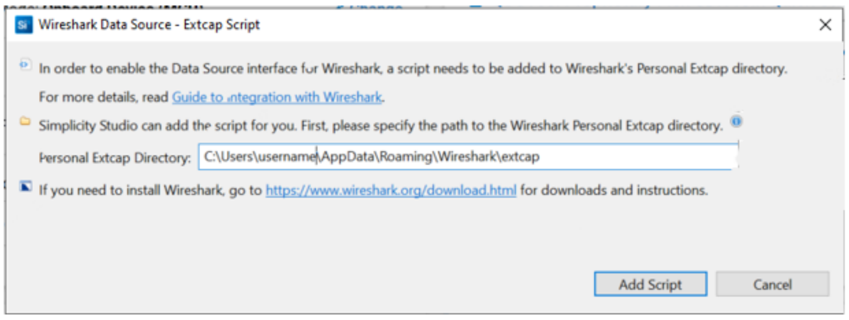
Wireshark Filter: (ip.addr==127.0.0.1 && tcp.port==8754)

Arguments

out: capture.log

time (in millisec): 2147483647

When Wireshark Data Source starts, Simplicity Studio writes a script to Wireshark's Personal Extcap directory. If this fails, a dialog where you can specify the target location is displayed. To find the location, launch Wireshark, select **Help > About Wireshark**, and go to the Folders tab.



Wireshark Data Source - Extcap Script

In order to enable the Data Source interface for Wireshark, a script needs to be added to Wireshark's Personal Extcap directory.

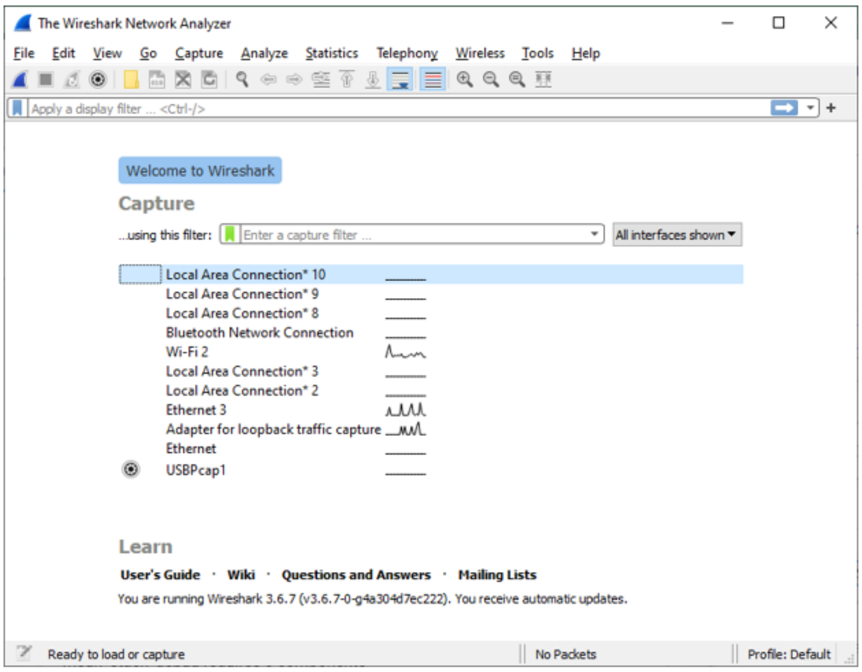
For more details, read Guide to integration with Wireshark.

Simplicity Studio can add the script for you. First, please specify the path to the Wireshark Personal Extcap directory.

Personal Extcap Directory: C:\Users\username\AppData\Roaming\Wireshark\extcap

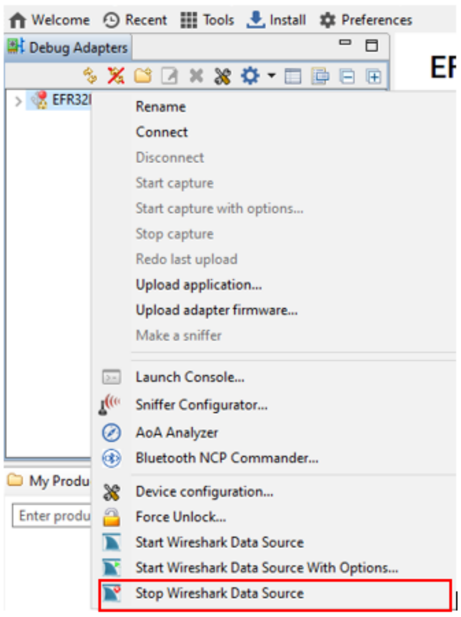
If you need to install Wireshark, go to https://www.wireshark.org/download.html for downloads and instructions.

4. After the Wireshark Data Source starts successfully, Simplicity Studio launches Wireshark automatically, and locks the device. In the Wireshark startup dialog, select a network interface to use and configure the capture and display filters. If the device is connected by USB, select the 'loopback' network interface. For networked devices, try Wi-Fi or Ethernet. See the Wireshark documentation for details on using Wireshark.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Local Area Connection* 10
Local Area Connection* 9
Local Area Connection* 8
Bluetooth Network Connection
Wi-Fi 2
Local Area Connection* 3
Local Area Connection* 2
Ethernet 3
Adapter for loopback traffic capture
Ethernet
USBPcap1
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 3.6.7 (v3.6.7-0-g4a304d7ec222). You receive automatic updates.
Ready to load or capture
No Packets
Profile: Default

Note: Wireshark is launched as a child process of Simplicity Studio. If you close Simplicity Studio, Wireshark also closes.

After you have completed the packet analysis, in Simplicity Studio right-click any of the devices connected to Wireshark, and click 'Stop Wireshark Data Source'. This stops Wireshark Data Source for all connected devices.



Note: Stopping Wireshark Data Source does not close Wireshark and closing Wireshark does not stop Wireshark Data Source.

Companion IDEs

Visual Studio Code Enablement

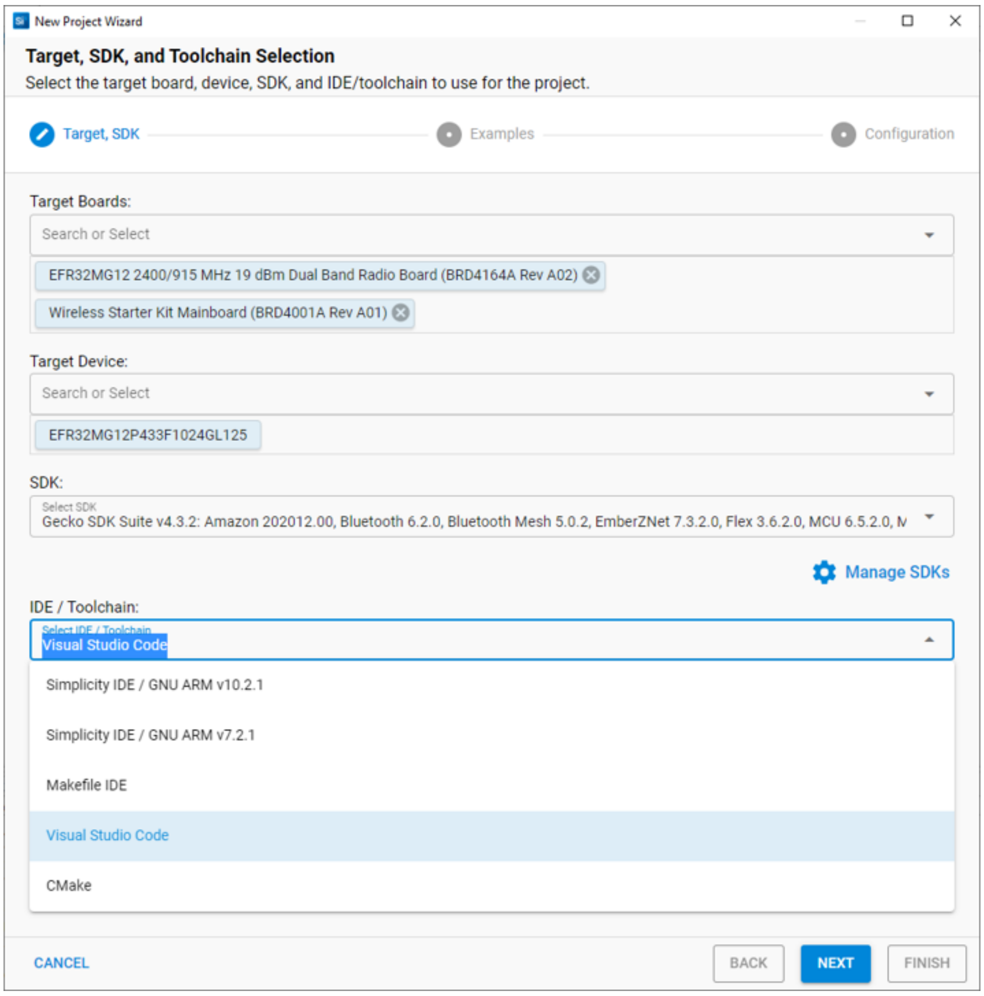
With the Simplicity Studio 5 (SSv5) 5.6.0.0 release, Silicon Labs introduced support for using Visual Studio Code (VS Code) in combination with Simplicity Studio. To enable this support, a VS Code extension is required. It can be either downloaded directly from the VS Code marketplace or with this link to marketplace. The preferred method of installation is to download it from the marketplace within the VS Code application itself.

Simplicity Studio is still needed to create the initial project and to make project changes with the Project Configurator GUIs. However, when you are using the Visual Studio Code compatible project generator, all editing, building, and debugging of the project should be done in VS Code. In other words, do not try to build the project in both Simplicity Studio and in VS Code.

Enabling a Project for Use Within VS Code

In order to operate on a Silicon Labs project from within VS Code, the correct project generator must be used. This can be set from within an existing project or at initial project creation.

For new projects, you can set the target generator in the new project wizard on step 1 as shown below.


New Project Wizard

Target, SDK, and Toolchain Selection

Select the target board, device, SDK, and IDE/toolchain to use for the project.

Target Boards:

Search or Select

EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)

Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Target Device:

Search or Select

EFR32MG12P433F1024GL125

SDK:

Select SDK

Gecko SDK Suite v4.3.2: Amazon 202012.00, Bluetooth 6.2.0, Bluetooth Mesh 5.0.2, EmberZNet 7.3.2.0, Flex 3.6.2.0, MCU 6.5.2.0, M

IDE / Toolchain:

Select IDE / Toolchain

Visual Studio Code

Simplicity IDE / GNU ARM v10.2.1

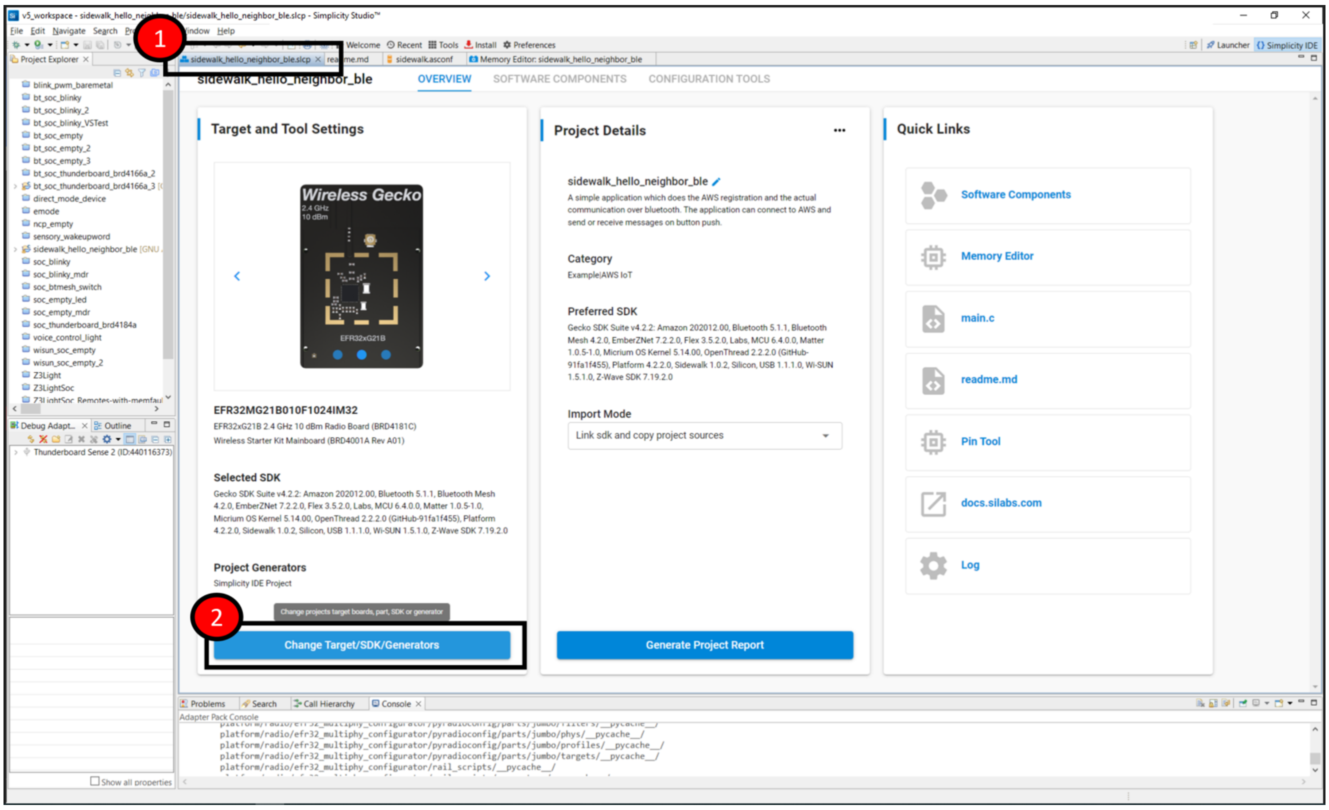
Simplicity IDE / GNU ARM v7.2.1

Makefile IDE

Visual Studio Code

CMake

For existing projects, from the Project Configurator Overview tab, click **Change Target/SDK/Generators** and select the VS Code generator.



sidewalk_hello_neighbor_ble

Target and Tool Settings

EFR32MG21B010F1024IM32
EFR32xG21B 2.4 GHz 10 dBm Radio Board (BRD4181C)
Wireless Starter Kit Mainboard (BRD4001A Rev A01)

Selected SDK
Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth 5.1.1, Bluetooth Mesh 4.2.0, EmberZNet 7.2.2.0, Flex 3.5.2.0, Labs, MCU 6.4.0.0, Matter 1.0.5-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.2.0 (GitHub-91fa1f455), Platform 4.2.2.0, Sidewalk 1.0.2, Silicon, USB 1.1.1.0, Wi-SUN 1.5.1.0, Z-Wave SDK 7.19.2.0

Project Generators
Simplicity IDE Project

Project Details

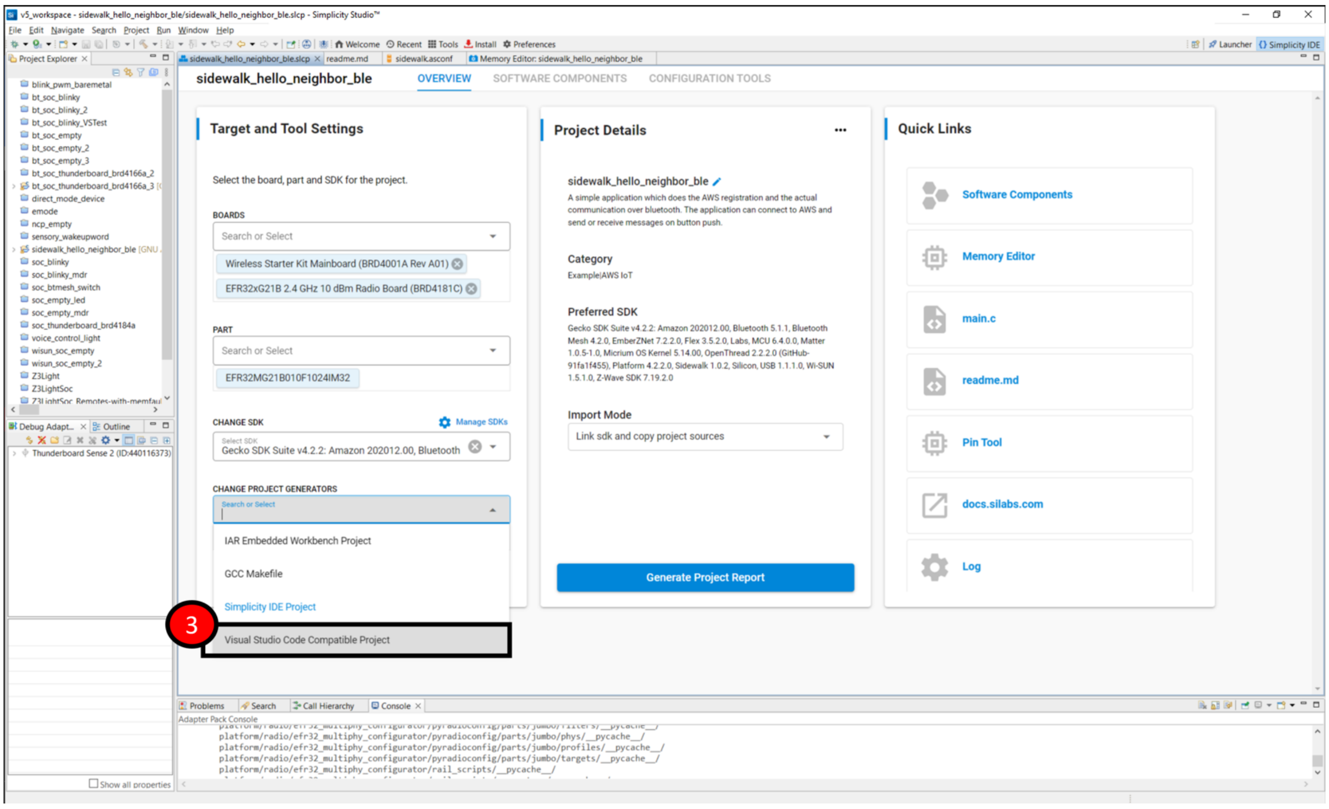
sidewalk_hello_neighbor_ble
A simple application which does the AWS registration and the actual communication over bluetooth. The application can connect to AWS and send or receive messages on button push.

Category
Example|AWS IoT

Preferred SDK
Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth 5.1.1, Bluetooth Mesh 4.2.0, EmberZNet 7.2.2.0, Flex 3.5.2.0, Labs, MCU 6.4.0.0, Matter 1.0.5-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.2.0 (GitHub-91fa1f455), Platform 4.2.2.0, Sidewalk 1.0.2, Silicon, USB 1.1.1.0, Wi-SUN 1.5.1.0, Z-Wave SDK 7.19.2.0

Import Mode
Link sdk and copy project sources

Adapter Pack Console
```
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/phys/__pycache__/
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/profiles/__pycache__/
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/targets/__pycache__/
    platform/radio/efr32_multiphy_configurator/rail_scripts/__pycache__/
```



sidewalk_hello_neighbor_ble

Target and Tool Settings

Select the board, part and SDK for the project.

BOARDS
Search or Select
Wireless Starter Kit Mainboard (BRD4001A Rev A01)
EFR32xG21B 2.4 GHz 10 dBm Radio Board (BRD4181C)

PART
Search or Select
EFR32MG21B010F1024IM32

CHANGE SDK
Select SDK
Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth

CHANGE PROJECT GENERATORS
Search or Select
- IAR Embedded Workbench Project
- GCC Makefile
- Simplicity IDE Project
- Visual Studio Code Compatible Project

Project Details

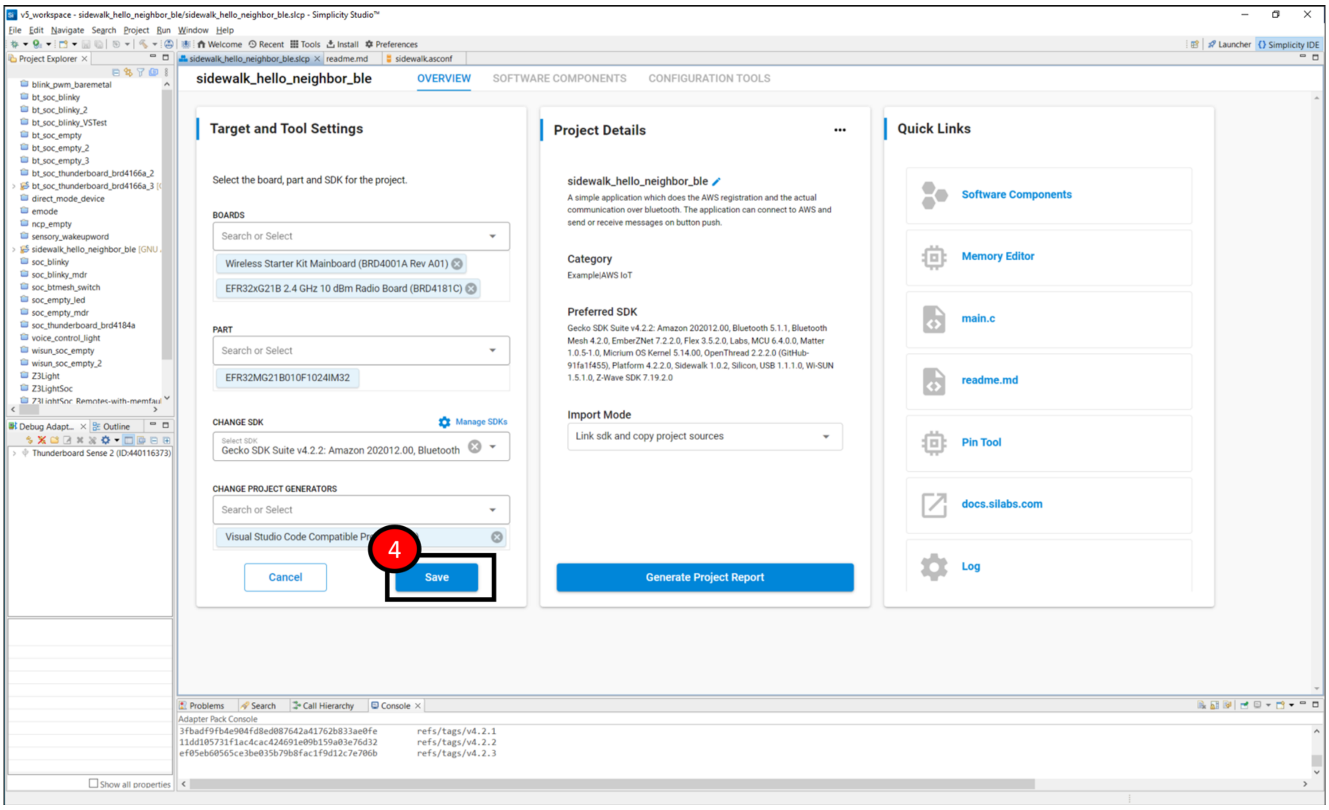
sidewalk_hello_neighbor_ble
A simple application which does the AWS registration and the actual communication over bluetooth. The application can connect to AWS and send or receive messages on button push.

Category
Example|AWS IoT

Preferred SDK
Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth 5.1.1, Bluetooth Mesh 4.2.0, EmberZNet 7.2.2.0, Flex 3.5.2.0, Labs, MCU 6.4.0.0, Matter 1.0.5-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.2.0 (GitHub-91fa1f455), Platform 4.2.2.0, Sidewalk 1.0.2, Silicon, USB 1.1.1.0, Wi-SUN 1.5.1.0, Z-Wave SDK 7.19.2.0

Import Mode
Link sdk and copy project sources

Adapter Pack Console
```
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/phys/__pycache__/
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/profiles/__pycache__/
    platform/radio/efr32_multiphy_configurator/pyradioconfig/parts/jumbo/targets/__pycache__/
    platform/radio/efr32_multiphy_configurator/rail_scripts/__pycache__/
```



sidewalk_hello_neighbor_ble

Target and Tool Settings

Select the board, part and SDK for the project.

BOARDS: Wireless Starter Kit Mainboard (BRD4001A Rev A01); EFR32xG21B 2.4 GHz 10 dBm Radio Board (BRD4181C)

PART: EFR32MG21B010F1024IM32

CHANGE SDK: Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth

CHANGE PROJECT GENERATORS: Visual Studio Code Compatible Pr

Project Details

sidewalk_hello_neighbor_ble

A simple application which does the AWS registration and the actual communication over bluetooth. The application can connect to AWS and send or receive messages on button push.

Category: Example|AWS IoT

Preferred SDK: Gecko SDK Suite v4.2.2: Amazon 202012.00, Bluetooth 5.1.1, Bluetooth Mesh 4.2.0, EmberZNet 7.2.2.0, Flex 3.5.2.0, Labs, MCU 6.4.0.0, Matter 1.0.5-1.0, Micrium OS Kernel 5.14.00, OpenThread 2.2.2.0 (GitHub-91fa1f455), Platform 4.2.2.0, Sidewalk 1.0.2, Silicon, USB 1.1.1.0, Wi-SUN 1.5.1.0, Z-Wave SDK 7.19.2.0

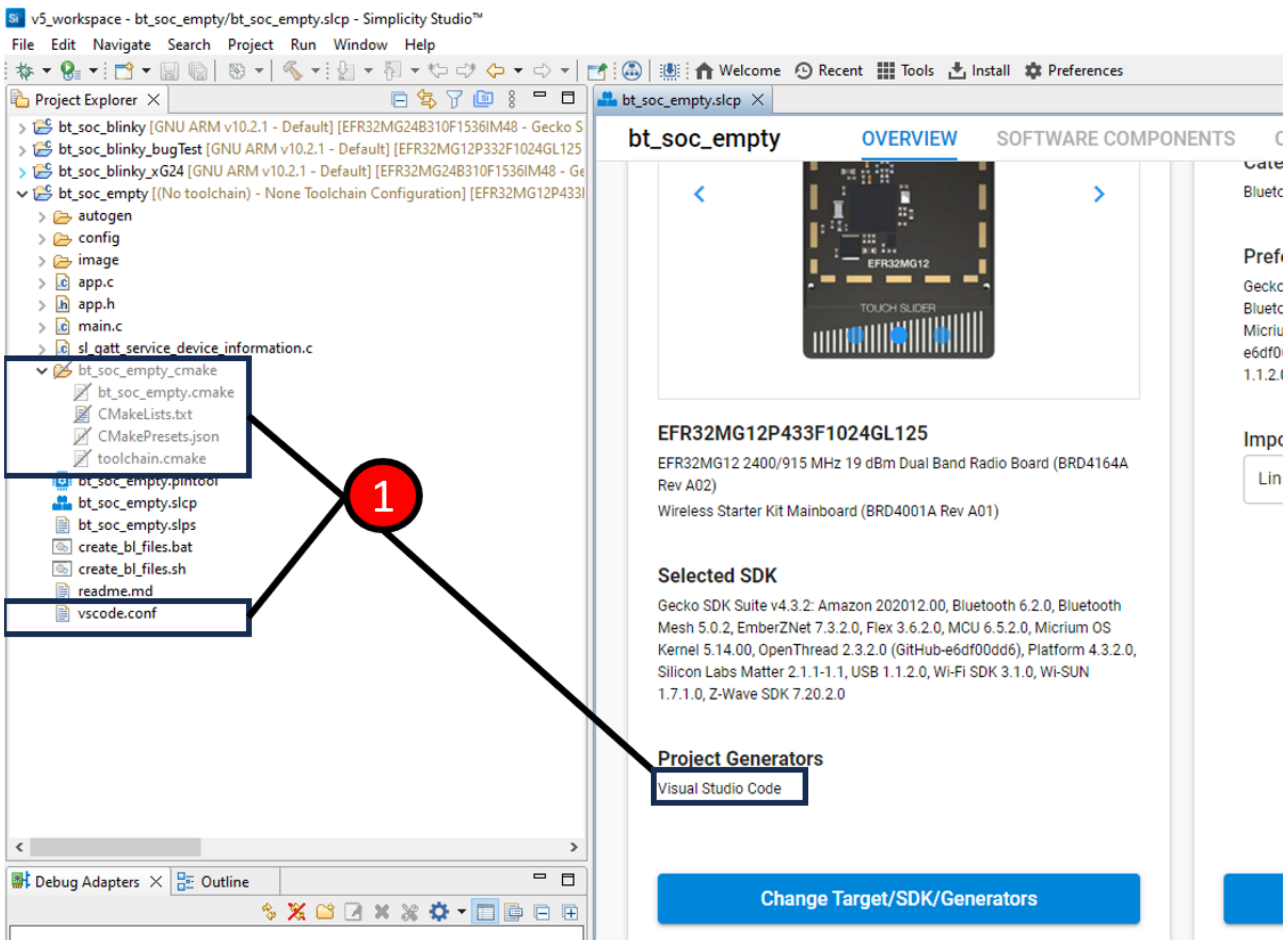
Import Mode: Link sdk and copy project sources

```
Adapter Pack Console
3fbadf9fb4e904fd8ed087642a41762b833ae0fe       refs/tags/v4.2.1
11dd105731f1ac4cac424691e09b159a03e76d32       refs/tags/v4.2.2
ef05eb60565ce3be035b79b8fac1f9d12c7e706b       refs/tags/v4.2.3
```

Once you have set your project to generate for VS Code, and generation is complete, you will notice new files in the Project Explorer view. This includes new files to support development in VS Code and enable CMake and Ninja. If a popup dialog appears asking what to do with some project configuration files, leave the setting at the default "Keep my file" and click **OK**.

The new VS Code support files are:

- Folder: <projectname>_cmake
  - <projectname>.cmake
  - CMakeLists.txt
  - CMakePresets.json
  - toolchain.cmake
- vscode.conf.


v5_workspace - bt_soc_empty/bt_soc_empty.slcp - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Project Explorer
bt_soc_blinky [GNU ARM v10.2.1 - Default] [EFR32MG24B310F1536IM48 - Gecko S
bt_soc_blinky_bugTest [GNU ARM v10.2.1 - Default] [EFR32MG12P332F1024GL125
bt_soc_blinky_xG24 [GNU ARM v10.2.1 - Default] [EFR32MG24B310F1536IM48 - Ge
bt_soc_empty [(No toolchain) - None Toolchain Configuration] [EFR32MG12P433I
autogen
config
image
app.c
app.h
main.c
sl_gatt_service_device_information.c
bt_soc_empty_cmake
bt_soc_empty.cmake
CMakeLists.txt
CMakePresets.json
toolchain.cmake
bt_soc_empty.pintool
bt_soc_empty.slcp
bt_soc_empty.slps
create_bl_files.bat
create_bl_files.sh
readme.md
vscode.conf
1
Debug Adapters
Outline
bt_soc_empty.slcp
bt_soc_empty
OVERVIEW
SOFTWARE COMPONENTS
EFR32MG12
TOUCH SLIDER
EFR32MG12P433F1024GL125
EFR32MG12 2400/915 MHz 19 dBm Dual Band Radio Board (BRD4164A Rev A02)
Wireless Starter Kit Mainboard (BRD4001A Rev A01)
Selected SDK
Gecko SDK Suite v4.3.2: Amazon 202012.00, Bluetooth 6.2.0, Bluetooth Mesh 5.0.2, EmberZNet 7.3.2.0, Flex 3.6.2.0, MCU 6.5.2.0, Micrium OS Kernel 5.14.00, OpenThread 2.3.2.0 (GitHub-e6df00dd6), Platform 4.3.2.0, Silicon Labs Matter 2.1.1-1.1, USB 1.1.2.0, Wi-Fi SDK 3.1.0, Wi-SUN 1.7.1.0, Z-Wave SDK 7.20.2.0
Project Generators
Visual Studio Code
Change Target/SDK/Generators

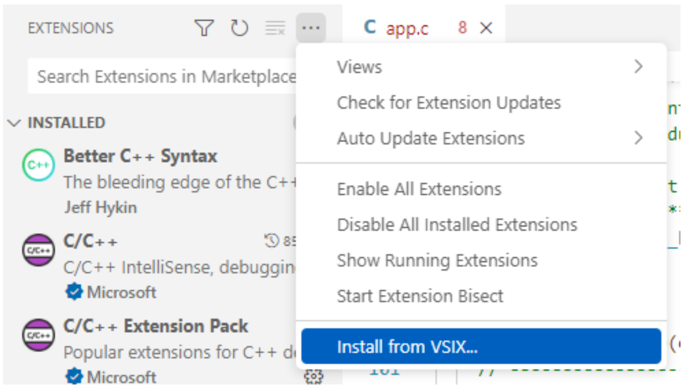
You can make any other project configurator changes, such as adding a component, to the project from within Simplicity Studio. From there, the project can be continued in VS Code. Once you begin using VS Code for development, it is not recommended to build the project in Simplicity Studio. However, before any changes have been made in VS Code, you can verify that it builds in Simplicity Studio before continuing development in VS Code.

The VS Code generator is Gecko SDK Suite (GSDK)-independent. This means it can be used with any 4.x.x GSDK.

Using the Project in Visual Studio Code

Preparing VS Code for Silicon Labs Support

To use VS Code for development with Silicon Labs projects, VS Code must first be downloaded and installed from the Visual Studio website (Download Visual Studio Code). Then the Silicon Labs extension must be installed either from within VS Code by clicking on the Extensions icon and then searching the Marketplace for “SiLabs” or by downloading the extension as a .vsix file from the marketplace and then adding it to VS Code by clicking the three dots next to EXTENSIONS and selecting **'Install from vsix...'**.



Installing the Simplicity Studio for VS Code extension will also install other extensions it depends on: C++ Extension Pack and Cortex-Debug.

VS Code is now ready to add the Silicon Labs project from the earlier step.

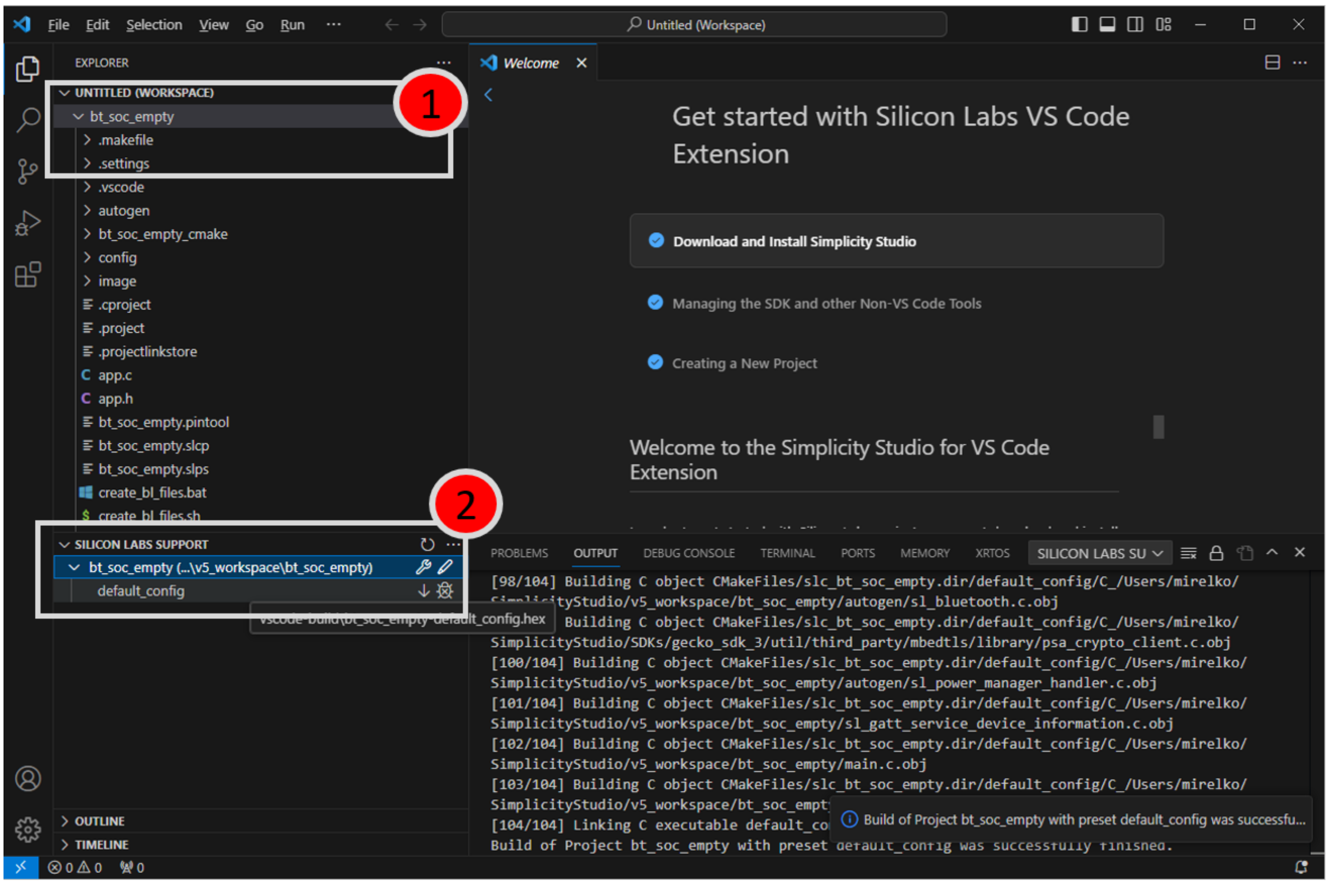
Adding a VS Code-Enabled Simplicity Studio Project to VS Code

You can add a project to VS Code in two ways.

1. Use **File > Open Folder...** and browse to the project folder in the Simplicity Studio workspace (default workspace is v5_workspace) or the source-controlled project folder.
2. Use **File > Add Folder to Workspace...** and again browse to the project folder. This second method is recommended, especially if working with multiple projects. This will place the folder in an untitled workspace. Select **File > Save Workspace As...** and select a folder and name for the workspace. The VS Code workspace should be saved outside of the Simplicity Studio workspace folder in a folder dedicated to VS Code workspaces. Subsequent projects can be added to this open workspace using **File > Add Folder to Workspace...**.

After adding projects to a workspace, you should notice that the project appears in the explorer and the 'SILICON LABS SUPPORT' section is now populated with project related actions as shown below.

Upon initial project addition, you will only see the “Build” icon. Once built, the flash and debug icons will appear as shown below.

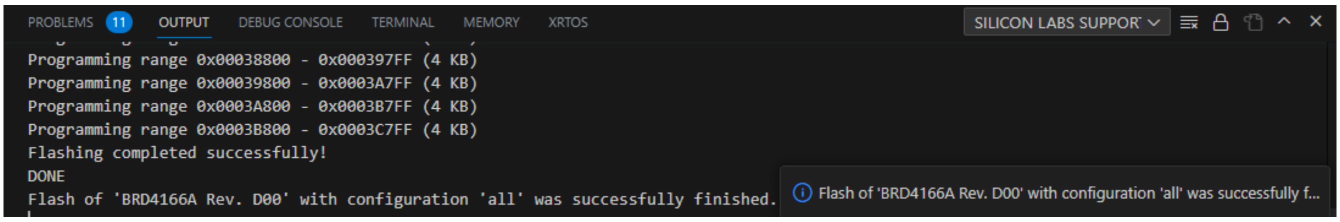

File Edit Selection View Go Run
Untitled (Workspace)
EXPLORER
UNTITLED (WORKSPACE)
bt_soc_empty
.makefile
.settings
.vscode
autogen
bt_soc_empty_cmake
config
image
.cproject
.project
.projectlinkstore
app.c
app.h
bt_soc_empty.pintool
bt_soc_empty.slcp
bt_soc_empty.slps
create_bl_files.bat
create_bl_files.sh
SILICON LABS SUPPORT
bt_soc_empty (...\v5_workspace\bt_soc_empty)
default_config
vscode-build\bt_soc_empty-default_config.hex
OUTLINE
TIMELINE
1
2
Welcome
Get started with Silicon Labs VS Code Extension
Download and Install Simplicity Studio
Managing the SDK and other Non-VS Code Tools
Creating a New Project
Welcome to the Simplicity Studio for VS Code Extension
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS MEMORY XRTOS
SILICON LABS SU

```
[98/104] Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/v5_workspace/bt_soc_empty/autogen/sl_bluetooth.c.obj
         Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/SDKs/gecko_sdk_3/util/third_party/mbedtls/library/psa_crypto_client.c.obj
[100/104] Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/v5_workspace/bt_soc_empty/autogen/sl_power_manager_handler.c.obj
[101/104] Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/v5_workspace/bt_soc_empty/sl_gatt_service_device_information.c.obj
[102/104] Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/v5_workspace/bt_soc_empty/main.c.obj
[103/104] Building C object CMakeFiles/slc_bt_soc_empty.dir/default_config/C_/Users/mirelko/
SimplicityStudio/v5_workspace/bt_soc_empt
[104/104] Linking C executable default_co
Build of Project bt_soc_empty with preset default_config was successfully finished.
```

Build of Project bt_soc_empty with preset default_config was successfu...

The source files can be edited and additional files may be added to the projects. When ready to build the project, select the project build configuration ('all' is the default for now) in the SILICON LABS SUPPORT section and the icons for three support options are displayed.

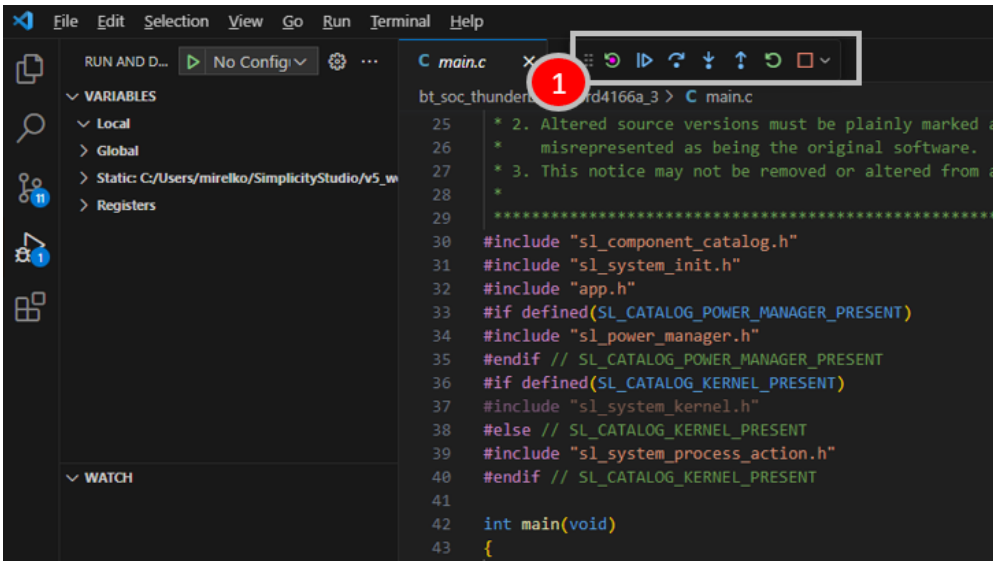
- **Build** will use 'make' along with the two generated make files and the GNU ARM GCC toolchain specified when the project was created in Simplicity Studio. All files and folders in the project folder will be compiled (including user-added source files).
- **Flash** will flash the project .hex file to the target board using Simplicity Commander. The Output console window will display the progress of the flash operation.



```
Programming range 0x00038800 - 0x000397FF (4 KB)
Programming range 0x00039800 - 0x0003A7FF (4 KB)
Programming range 0x0003A800 - 0x0003B7FF (4 KB)
Programming range 0x0003B800 - 0x0003C7FF (4 KB)
Flashing completed successfully!
DONE
Flash of 'BRD4166A Rev. D00' with configuration 'all' was successfully finished.
```

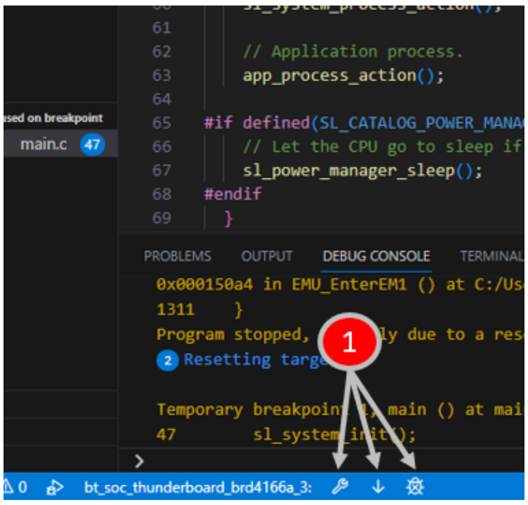
Flash of 'BRD4166A Rev. D00' with configuration 'all' was successfully f...

- **Debug** will start a debug session using the Segger GDB Server and the GNU ARM GDB Client. The normal options to step through the program, set breakpoints, and examine variables and registers are all supported. The debug control icons are highlighted in the red box in the following figure.



```
 * 2. Altered source versions must be plainly marked a
 *    misrepresented as being the original software.
 * 3. This notice may not be removed or altered from a
 *
 ******************************************************
#include "sl_component_catalog.h"
#include "sl_system_init.h"
#include "app.h"
#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
#include "sl_power_manager.h"
#endif // SL_CATALOG_POWER_MANAGER_PRESENT
#if defined(SL_CATALOG_KERNEL_PRESENT)
#include "sl_system_kernel.h"
#else // SL_CATALOG_KERNEL_PRESENT
#include "sl_system_process_action.h"
#endif // SL_CATALOG_KERNEL_PRESENT

int main(void)
{
```

Additionally, you can access these functions from the bottom ribbon as shown below.



```
      // Application process.
      app_process_action();

  #if defined(SL_CATALOG_POWER_MANA
      // Let the CPU go to sleep if
      sl_power_manager_sleep();
  #endif
    }
```

```
0x000150a4 in EMU_EnterEM1 () at C:/Us
1311    }
Program stopped,         ly due to a res
2 Resetting targe

Temporary breakpoint 1, main () at mai
47          sl_system_init();
```

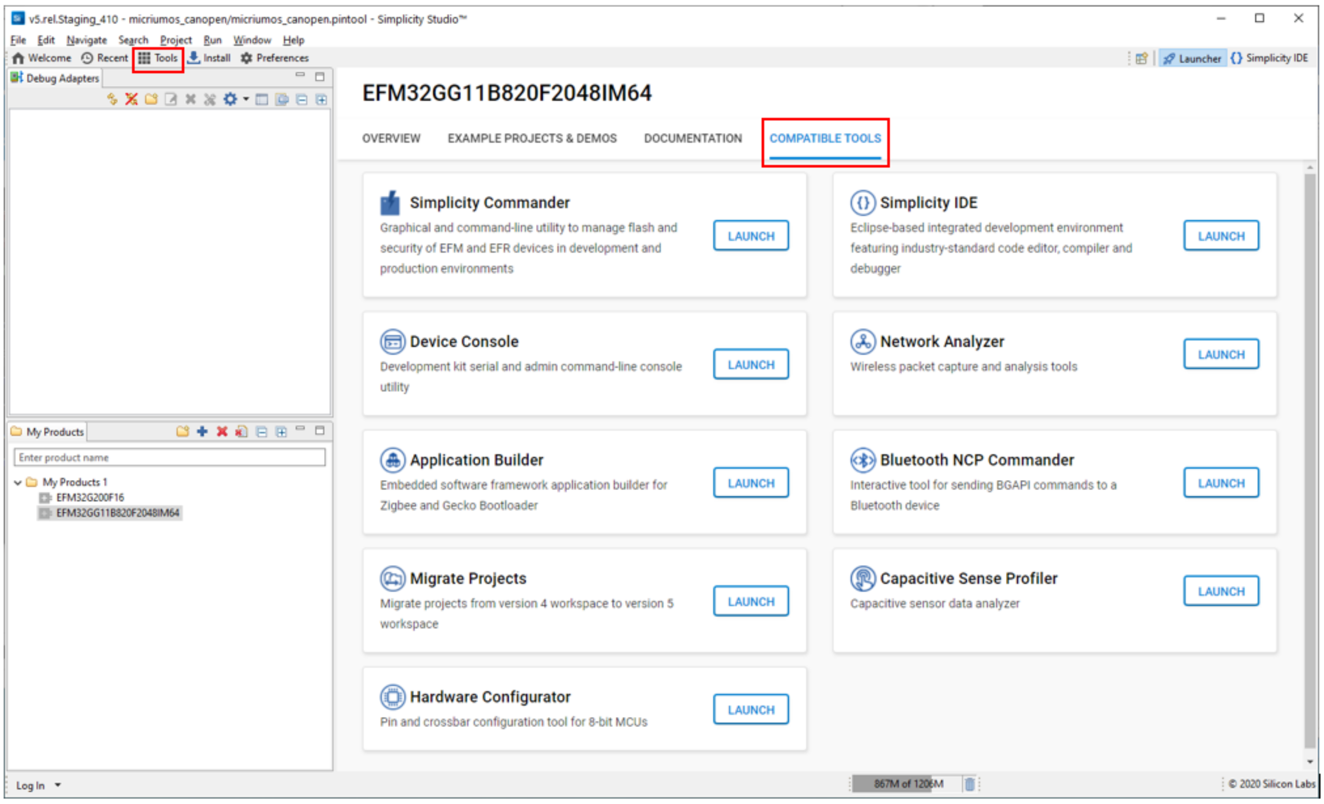
Note: The *vscode.conf* file is used as the 'glue' file between the project configuration of the Simplicity Studio-installed resources and the Silicon Labs VS Code extension. The links in this file are used by the Silicon Labs VS Code extension to find the associated tools necessary for building, flashing, and debugging the project. It is not meant to be manually edited.

Note for Matter developers: After you have generated your project in Simplicity Studio and added it to VS Code Workspace, Silicon Labs has provided four VS Code tasks to facilitate developing Matter projects in VS Code. See the GitHub Matter documentation for more information.

Tools Overview

Using the Tools - Overview

Simplicity Studio® 5 (SSv5) provides a number of tools to assist with development activities. Use the **Tools** button to see a list of available tools. The Project COMPATIBLE TOOLS tab provides a list filtered by the project's target device.


v5.rel.Staging_410 - micriumos_canopen/micriumos_canopen.pintool - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Debug Adapters
My Products
Enter product name
My Products 1
EFM32G200F16
EFM32GG11B820F2048IM64
OVERVIEW EXAMPLE PROJECTS & DEMOS DOCUMENTATION COMPATIBLE TOOLS
LAUNCH
Log In
867M of 1206M
© 2020 Silicon Labs

EFM32GG11B820F2048IM64

Simplicity Commander
Graphical and command-line utility to manage flash and security of EFM and EFR devices in development and production environments

Simplicity IDE
Eclipse-based integrated development environment featuring industry-standard code editor, compiler and debugger

Device Console
Development kit serial and admin command-line console utility

Network Analyzer
Wireless packet capture and analysis tools

Application Builder
Embedded software framework application builder for Zigbee and Gecko Bootloader

Bluetooth NCP Commander
Interactive tool for sending BGAPI commands to a Bluetooth device

Migrate Projects
Migrate projects from version 4 workspace to version 5 workspace

Capacitive Sense Profiler
Capacitive sensor data analyzer

Hardware Configurator
Pin and crossbar configuration tool for 8-bit MCUs

Many of these tools are discussed on other pages in this guide, for example:

- Simplicity IDE
- Application Builder
- Migrate Projects
- Flash Programmer
- Hardware Configurator

These pages discuss:

- Bluetooth NCP Commander
- Energy Profiler
- Network Analyzer
- Bluetooth Direction Finding tools suite

More tools will be documented in future releases of the *Simplicity Studio 5 User's Guide*.

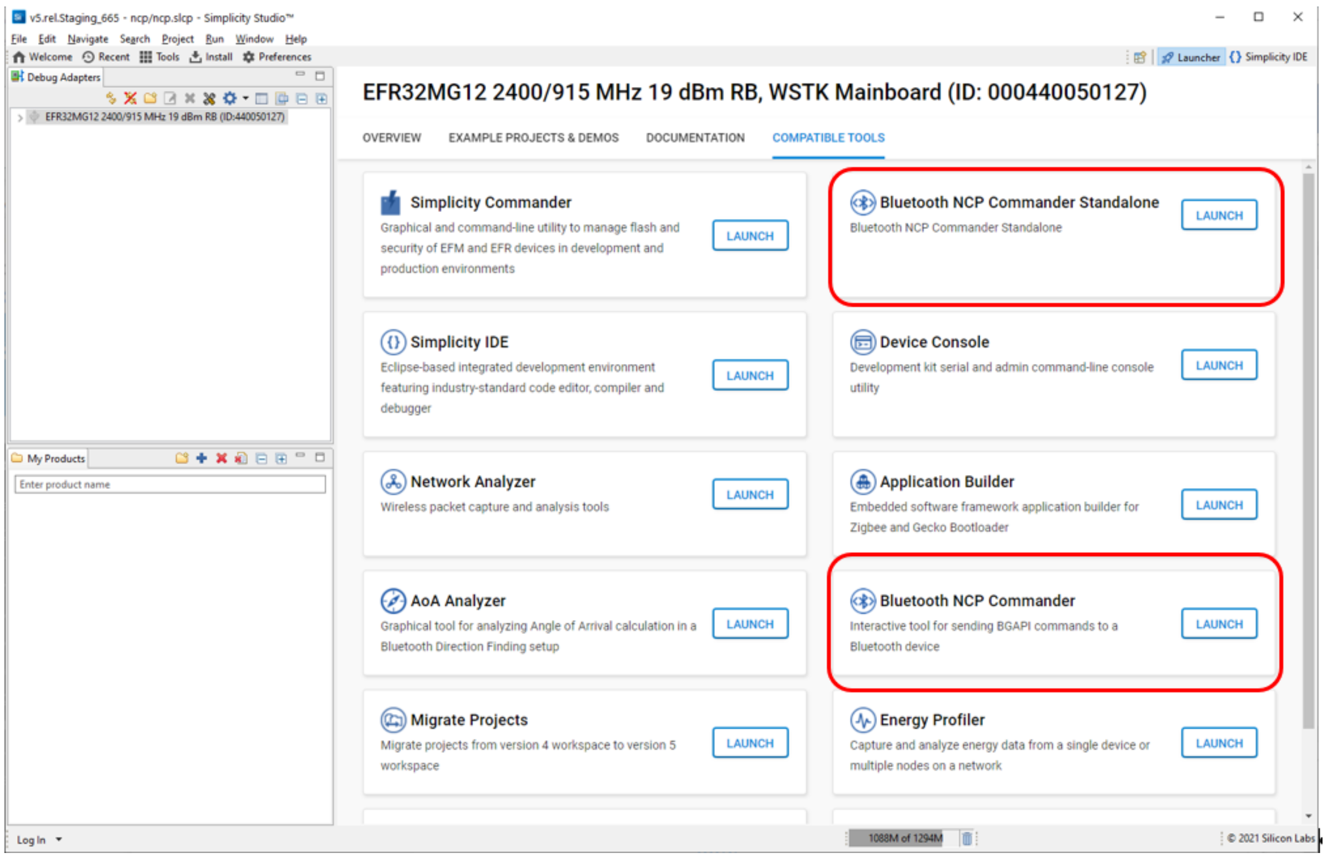
Bluetooth NCP Commander

Bluetooth NCP Commander

Bluetooth NCP Commander is a Simplicity Studio® 5 (SSv5) tool for sending BGAPI commands to an NCP target application during development.

Two versions of the tool are available in SSv5:

- Bluetooth NCP Commander Standalone
- Bluetooth NCP Commander


v5.rel.Staging_665 - ncp/ncp.slcp - Simplicity Studio™
File Edit Navigate Search Project Run Window Help
Welcome Recent Tools Install Preferences
Launcher Simplicity IDE
Debug Adapters
EFR32MG12 2400/915 MHz 19 dBm RB (ID:440050127)
My Products
Enter product name
OVERVIEW
EXAMPLE PROJECTS & DEMOS
DOCUMENTATION
COMPATIBLE TOOLS
LAUNCH
Log In
1088M of 1294M
© 2021 Silicon Labs

EFR32MG12 2400/915 MHz 19 dBm RB, WSTK Mainboard (ID: 000440050127)

Simplicity Commander
Graphical and command-line utility to manage flash and security of EFM and EFR devices in development and production environments

Bluetooth NCP Commander Standalone
Bluetooth NCP Commander Standalone

Simplicity IDE
Eclipse-based integrated development environment featuring industry-standard code editor, compiler and debugger

Device Console
Development kit serial and admin command-line console utility

Network Analyzer
Wireless packet capture and analysis tools

Application Builder
Embedded software framework application builder for Zigbee and Gecko Bootloader

AoA Analyzer
Graphical tool for analyzing Angle of Arrival calculation in a Bluetooth Direction Finding setup

Bluetooth NCP Commander
Interactive tool for sending BGAPI commands to a Bluetooth device

Migrate Projects
Migrate projects from version 4 workspace to version 5 workspace

Energy Profiler
Capture and analyze energy data from a single device or multiple nodes on a network

For detailed information about using NCP Commander and developing Bluetooth NCP-Host applications, see AN1259: Using the Silicon Labs v3.x Bluetooth Stack in Network Co-Processor Mode.

Bluetooth NCP Commander Standalone

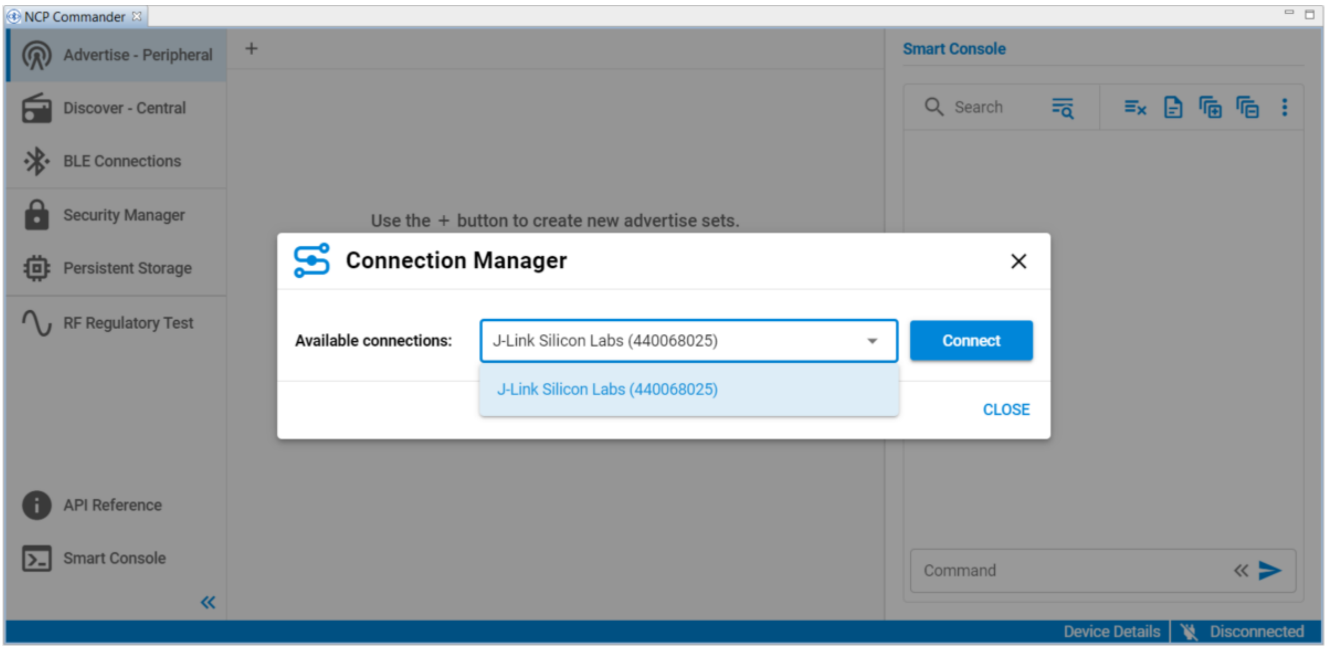
The Standalone version is provided for customers who need to control an NCP device on custom hardware. The Standalone version can access the system COM ports that are not exposed in SSv5. Users who wish to work outside of the SSv5 environment can do so by copying the directory <SSv5-install>/developer/adapter_packs/ncp_commander.

The Standalone version cannot access parts over Ethernet, which the integrated SSv5 version can.

Bluetooth NCP Commander

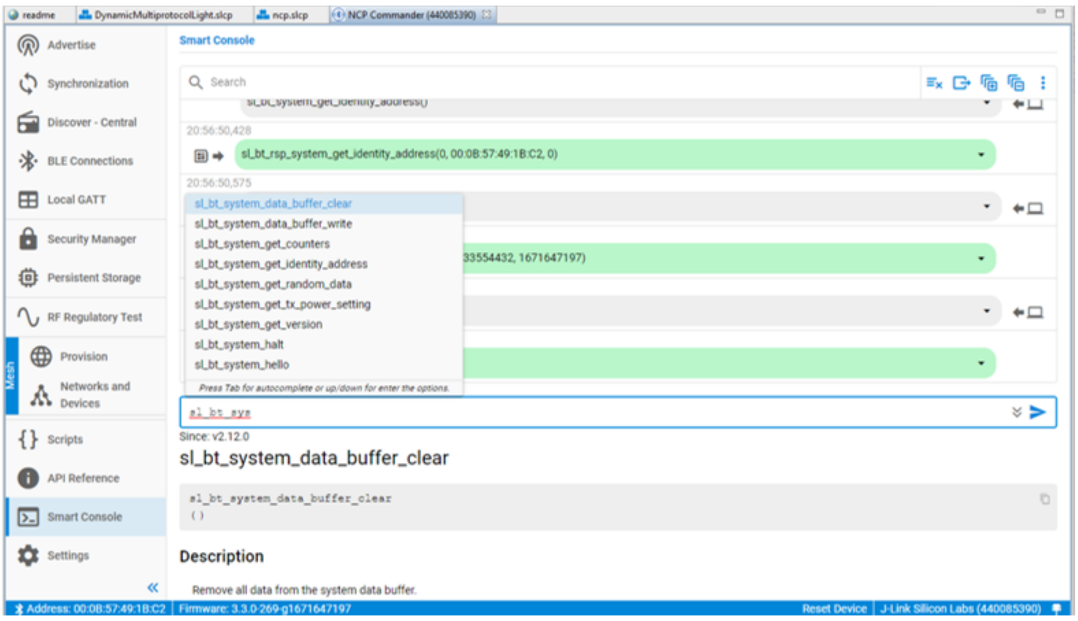
To open Bluetooth NCP Commander, make sure that the correct board is connected. If you are just getting started, build and flash a project based on **Bluetooth - NCP**.

In the Project Configurator COMPATIBLE TOOLS tab, click **Launch** on the Bluetooth NCP Commander card. In the Connection Manager dialog, select the target device and click **CONNECT**.



Once the UART connection to the mainboard is established, an Interactive view opens, which you can use to issue BGAPI commands. Note that the connected device is shown in the lower right. You can change target devices without leaving Bluetooth NCP Commander by clicking that area. It will show "Disconnected" if not connected to any device. Check the log for the NCP target response and status messages.

You can also issue commands manually. For example, you can issue the 'system hello' command at any time to verify that communication between the host and the device is working. The Smart Console provides auto completion and documentation for the possible commands.



```
sl_bt_system_get_identity_address()
20:56:50,428
sl_bt_rsp_system_get_identity_address(0, 00:0B:57:49:1B:C2, 0)
20:56:50,575
33554432, 1671647197)
```

```
sl_bt_system_data_buffer_clear
sl_bt_system_data_buffer_write
sl_bt_system_get_counters
sl_bt_system_get_identity_address
sl_bt_system_get_random_data
sl_bt_system_get_tx_power_setting
sl_bt_system_get_version
sl_bt_system_halt
sl_bt_system_hello
Press Tab for autocomplete or up/down for enter the options.
```

```
sl_bt_sys
```

Since: v2.12.0

sl_bt_system_data_buffer_clear

```
sl_bt_system_data_buffer_clear
()
```

Description

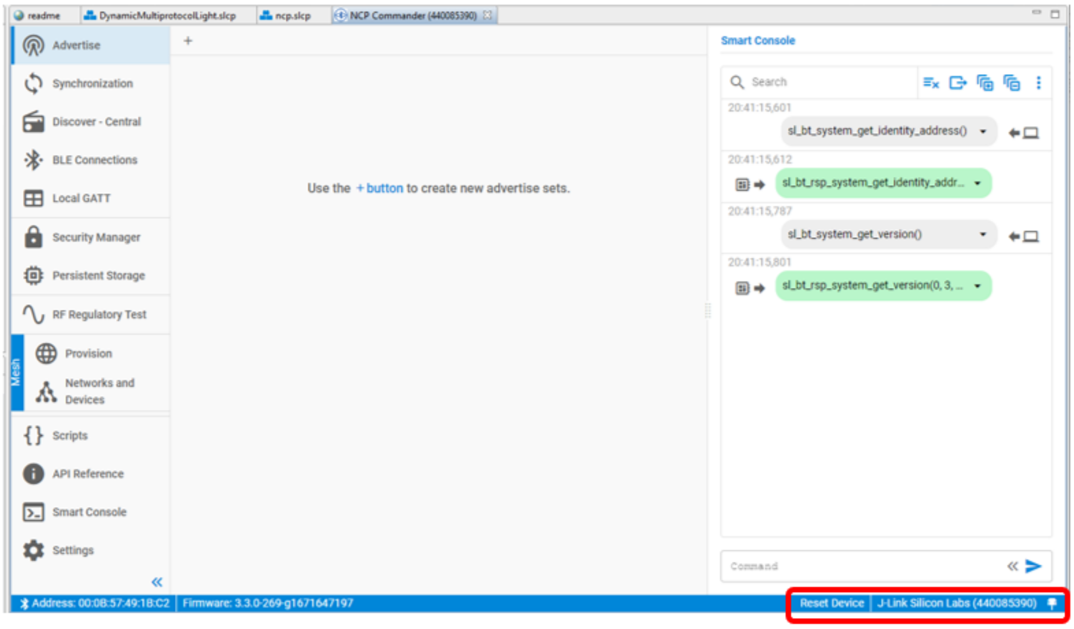
Remove all data from the system data buffer.

Address: 00:0B:57:49:1B:C2 | Firmware: 3.3.0-269-g1671647197 | Reset Device | J-Link Silicon Labs (440085390)

NCP Commander provides a simple scripting feature. Create or import an existing script using the controls in the top right corner. You can use any BGAPI commands in the script, but it has no additional features, such as branching or error handling. Click **Export** to save the commands sent through the console to a file that can be imported back as a script.

Advertising

To start advertising, click "+" next to "Advertise – Peripheral" to create an advertiser set.

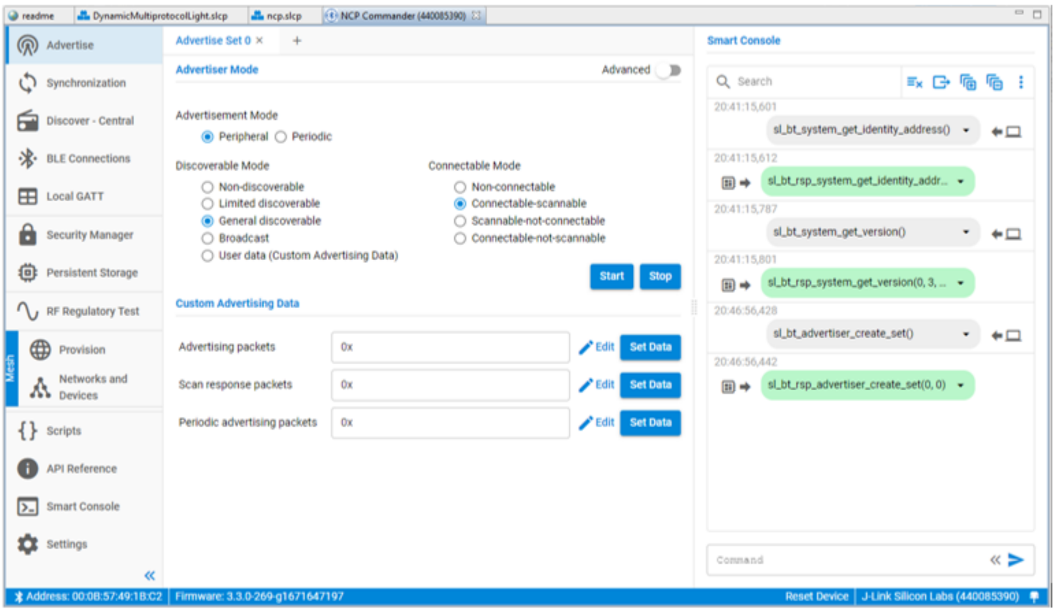


Use the + button to create new advertise sets.

```
20:41:15,601
sl_bt_system_get_identity_address()
20:41:15,612
sl_bt_rsp_system_get_identity_addr...
20:41:15,787
sl_bt_system_get_version()
20:41:15,801
sl_bt_rsp_system_get_version(0, 3, ...
```

Address: 00:0B:57:49:1B:C2 | Firmware: 3.3.0-269-g1671647197 | Reset Device | J-Link Silicon Labs (440085390)

Select the desired advertising mode, create custom advertising packets if desired, and click **Start**.



Advertisement Mode
[✓ Peripheral] Periodic

Discoverable Mode
Non-discoverable
Limited discoverable
[✓ General discoverable]
Broadcast
User data (Custom Advertising Data)

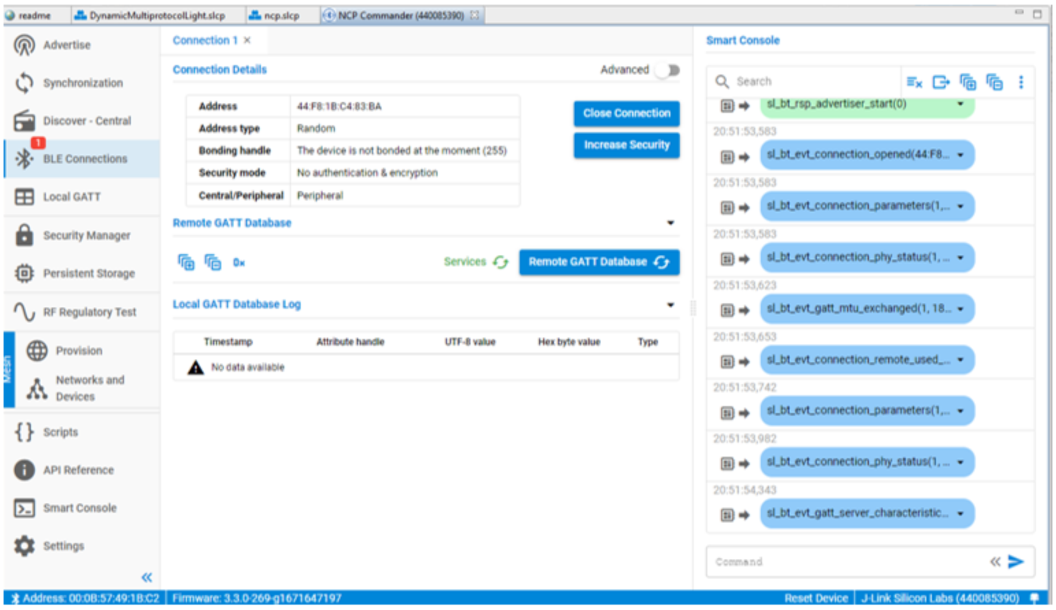
Connectable Mode
Non-connectable
[✓ Connectable-scannable]
Scannable-not-connectable
Connectable-not-scannable

Advertising packets	0x
Scan response packets	0x
Periodic advertising packets	0x

```
20:41:15,601  sl_bt_system_get_identity_address()
20:41:15,612  sl_bt_rsp_system_get_identity_addr...
20:41:15,787  sl_bt_system_get_version()
20:41:15,801  sl_bt_rsp_system_get_version(0, 3, ...
20:46:56,428  sl_bt_advertiser_create_set()
20:46:56,442  sl_bt_rsp_advertiser_create_set(0, 0)
```

Address: 00:0B:57:49:1B:C2 | Firmware: 3.3.0-269-g1671647197

When advertising, the NCP target example accepts Bluetooth connections. If you connect to a WSTK or with another central device (for example with your phone), you can see the events and commands on the log.



Address	44:F8:1B:C4:83:BA
Address type	Random
Bonding handle	The device is not bonded at the moment (255)
Security mode	No authentication & encryption
Central/Peripheral	Peripheral

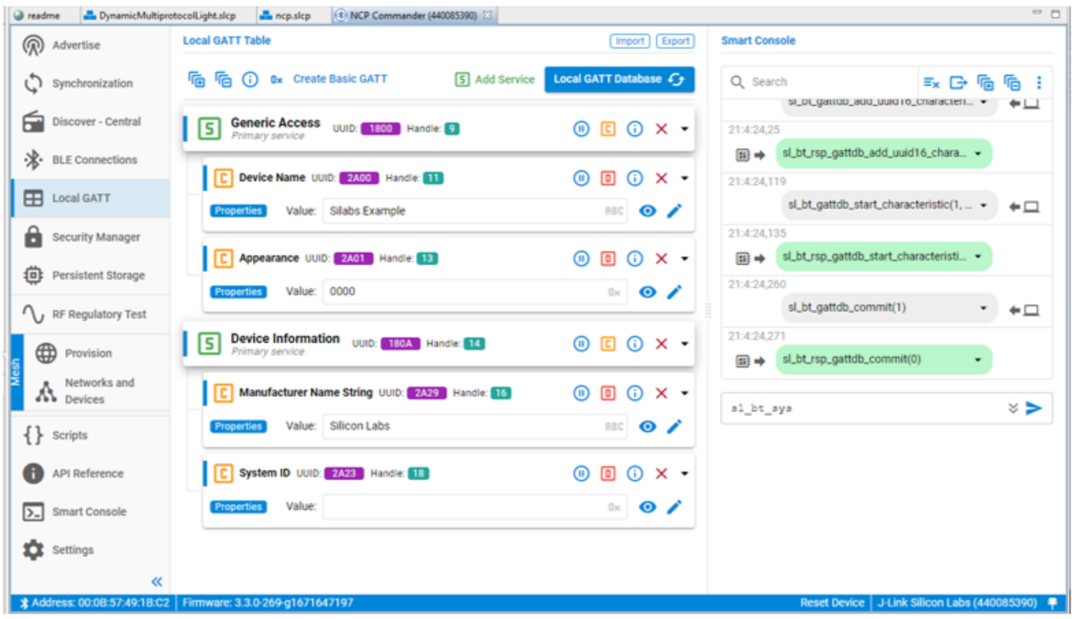
Timestamp	Attribute handle	UTF-8 value	Hex byte value	Type
No data available				

```
             sl_bt_rsp_advertiser_start(0)
20:51:53,583  sl_bt_evt_connection_opened(44:F8...
20:51:53,583  sl_bt_evt_connection_parameters(1,...
20:51:53,583  sl_bt_evt_connection_phy_status(1,...
20:51:53,623  sl_bt_evt_gatt_mtu_exchanged(1, 18...
20:51:53,653  sl_bt_evt_connection_remote_used_...
20:51:53,742  sl_bt_evt_connection_parameters(1,...
20:51:53,982  sl_bt_evt_connection_phy_status(1,...
20:51:54,343  sl_bt_evt_gatt_server_characteristic...
```

Address: 00:0B:57:49:1B:C2 | Firmware: 3.3.0-269-g1671647197

GATT Database

NCP commander supports creating a GATT database. To generate a basic GATT database, open the Local GATT view, and click **Create Basic GATT**. The following database is generated.

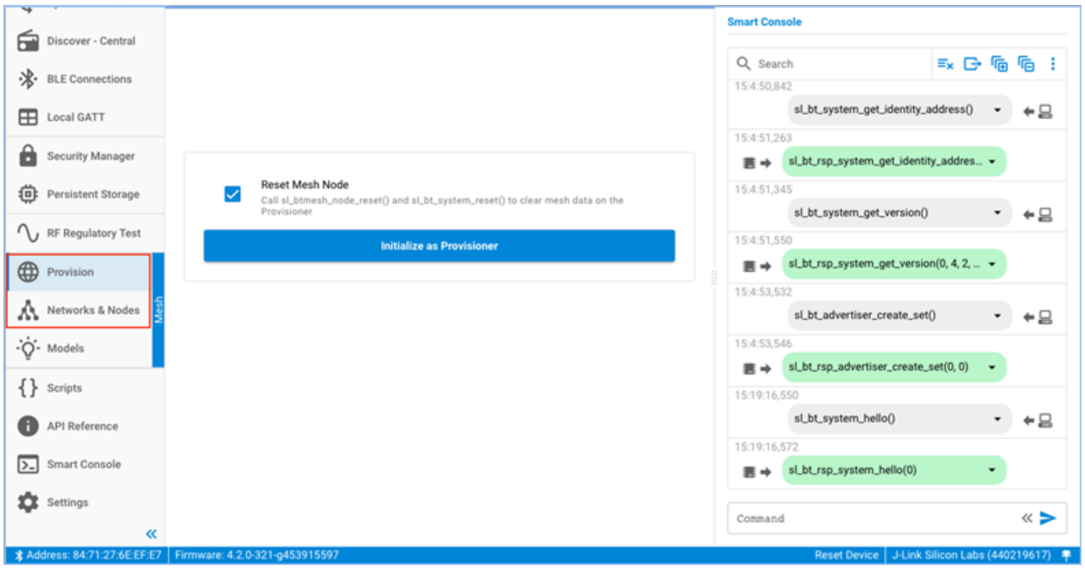


Here you can add services and characteristics. You can also read out the local GATT database from the device.

Bluetooth mesh

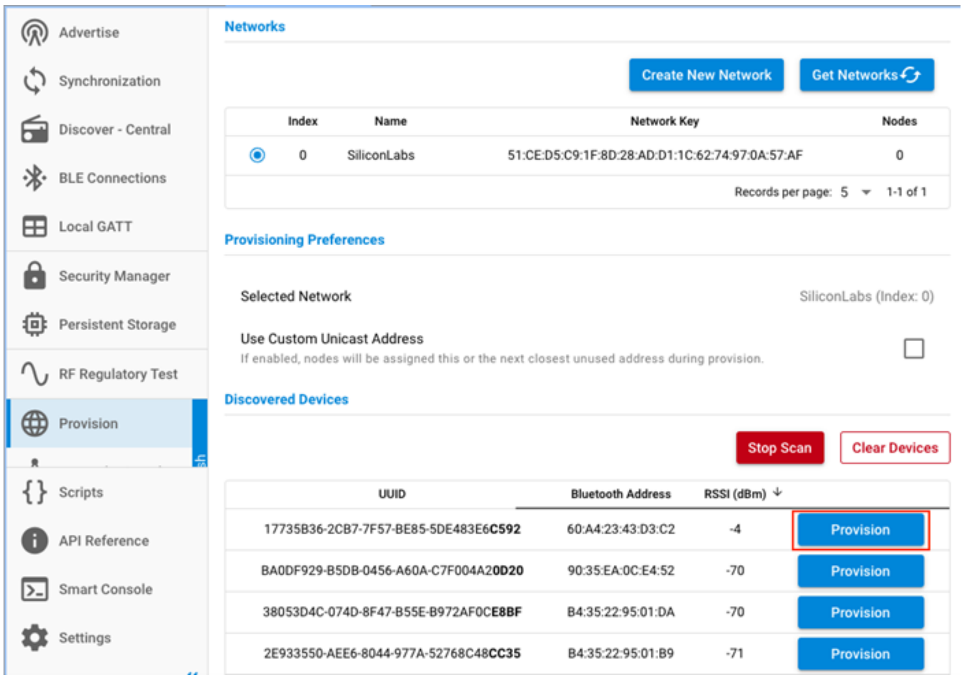
NCP Commander supports Bluetooth mesh features. You can issue Bluetooth mesh commands manually in the Smart Console, or use the interactive host provisioner feature to provision and configure mesh nodes and manage mesh networks, instead of using a Bluetooth mesh mobile application. To use the Bluetooth Mesh features, create, build, and flash the device with an NCP example supporting Mesh features. This section provides a summary of host provisioning. For details see AN1259: Using the Silicon Labs Bluetooth® Stack v3.x and Higher in Network Co-Processor Mode .

To start using the host provisioner, select either **Provision** or **Networks and Devices** on the left menu, and click **Initialize for Provision**.



If you do not have a network from a previous configuration or have reset the provisioner node, you must create a new network with **Create New Network**.

To provision devices, select **Provision** on the left menu, and click **Start Scan** in the right panel. The devices that are transmitting unprovisioned beacons are shown in the **Discovered Devices** section. Click **Provision** next to the device you want to provision.



Networks

	Index	Name	Network Key	Nodes
●	0	SiliconLabs	51:CE:D5:C9:1F:8D:28:AD:D1:1C:62:74:97:0A:57:AF	0

Records per page: 5 1-1 of 1

Provisioning Preferences

Selected Network SiliconLabs (Index: 0)

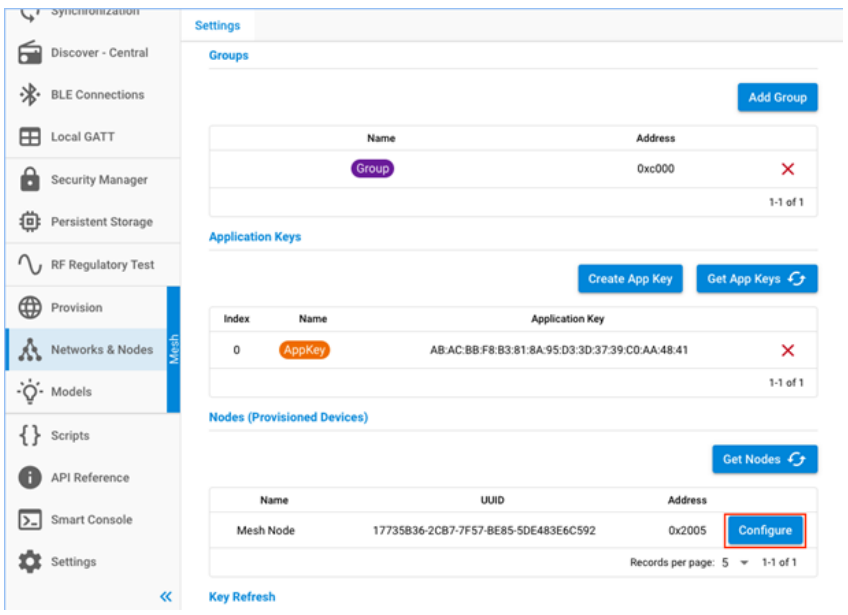
Use Custom Unicast Address
If enabled, nodes will be assigned this or the next closest unused address during provision.

Discovered Devices

UUID	Bluetooth Address	RSSI (dBm)	
17735B36-2CB7-7F57-BE85-5DE483E6**C592**	60:A4:23:43:D3:C2	-4	Provision
BA0DF929-B5DB-0456-A60A-C7F004A2**0D20**	90:35:EA:0C:E4:52	-70	Provision
38053D4C-074D-8F47-B55E-B972AF0C**E8BF**	B4:35:22:95:01:DA	-70	Provision
2E933550-AEE6-8044-977A-52768C48**CC35**	B4:35:22:95:01:B9	-71	Provision

Before configuring provisioned devices, you may need to create application keys and groups. Application keys, groups and other network settings can be managed in the **Settings** tab of the **Networks and Devices** menu item.

To configure provisioned devices, select **Networks and Devices** on the left menu. Provisioned devices are shown in the **Provisioned Devices** section of the **Settings** tab. Click **Configure**.



Groups

Name	Address	
Group	0xc000	✕

1-1 of 1

Application Keys

Index	Name	Application Key	
0	AppKey	AB:AC:BB:F8:B3:81:8A:95:D3:3D:37:39:C0:AA:48:41	✕

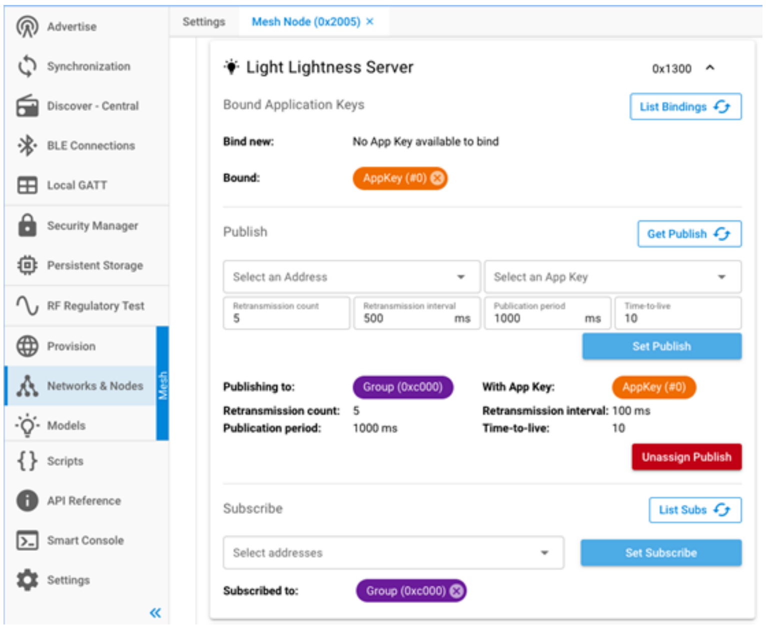
1-1 of 1

Nodes (Provisioned Devices)

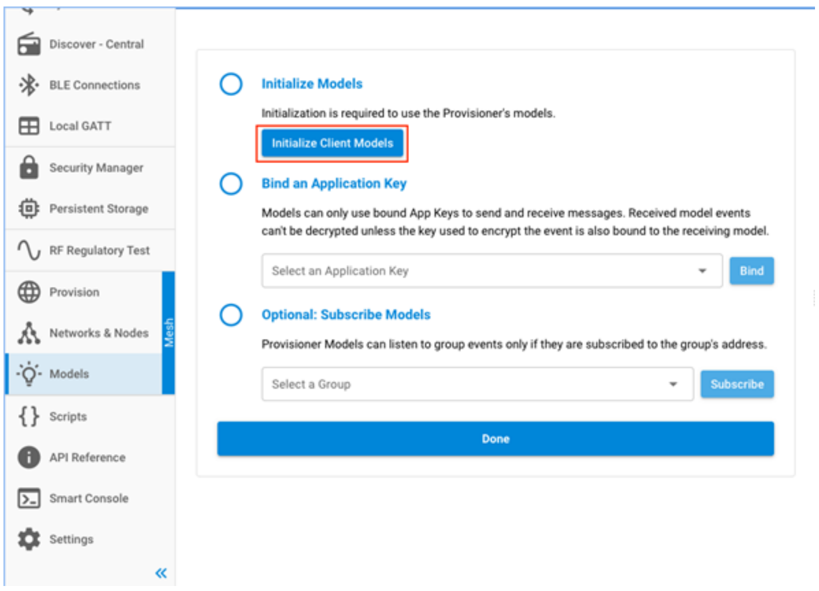
Name	UUID	Address	
Mesh Node	17735B36-2CB7-7F57-BE85-5DE483E6C592	0x2005	Configure

Records per page: 5 1-1 of 1

This opens a **Mesh Node** tab in which you can add an application key. Once you have added the key, the interface changes to include a **Get DCD** control (not shown). Click **Get DCD** to to configure all the Models available on your node(s), bind to app keys, set publishing or subscription to groups, fine tune parameters, and so on.

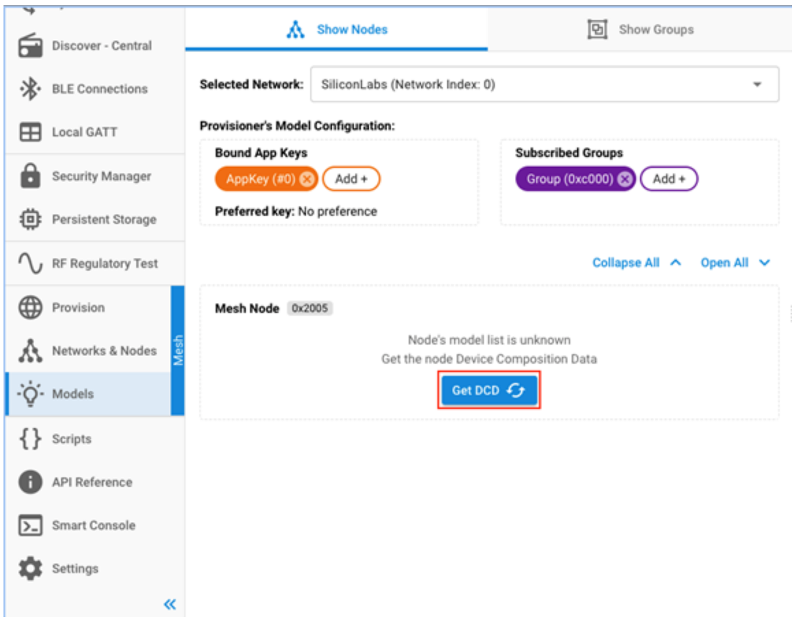


To configure the host provisioner, first initialize models using **Initialize Client Models**.


Discover - Central
BLE Connections
Local GATT
Security Manager
Persistent Storage
RF Regulatory Test
Provision
Networks & Nodes
Models
Scripts
API Reference
Smart Console
Settings
Mesh
Initialize Models
Initialization is required to use the Provisioner's models.
Initialize Client Models
Bind an Application Key
Models can only use bound App Keys to send and receive messages. Received model events can't be decrypted unless the key used to encrypt the event is also bound to the receiving model.
Select an Application Key
Bind
Optional: Subscribe Models
Provisioner Models can listen to group events only if they are subscribed to the group's address.
Select a Group
Subscribe
Done

Bind an application key to the models and, optionally, subscribe the models to a group. When configuration is complete, click **Done**

The Show Nodes tab is displayed. Click **Get DCD** to configure all the Models available on your node(s), bind to app keys, set publishing or subscription to groups, fine-tune parameters, and so on.



Selected Network: SiliconLabs (Network Index: 0)

Provisioner's Model Configuration:

Bound App Keys

Preferred key: No preference

Subscribed Groups

Mesh Node 0x2005

Node's model list is unknown
Get the node Device Composition Data

Energy Profiler

Energy Profiler

Simplicity Studio® 5 (SSv5)'s Energy Profiler enables you to visualize the energy consumption of individual devices, multiple devices on one target system, or a network of interacting wireless devices to analyze and improve the power performance of these systems. Real-time information on current consumption is correlated with the program counter providing advanced energy software monitoring capabilities. It also provides a basic level of integration with the Network Analyzer network analysis tool.

Energy saving and efficiency are at the top of developers' priorities for an ever-growing number of embedded systems applications. These constraints can be due to government regulations (for example, metering applications), a requirement to increase battery life, or simply a need to lower the electricity bill. In battery-operated systems, energy efficiency often plays the most important role. In cases where developers are satisfied with their system's battery life, increasing the energy efficiency means they can switch to a smaller and cheaper battery, which will lower the overall cost. There are also situations where the operating life must be extended to the absolute maximum, such as products where the battery cannot be replaced or replacement involves very high costs.

Having a low-power MCU by itself does not mean you will have lower energy consumption. The trick is to optimize your software not just in terms of functionality, but also with respect to energy efficiency. Having full control of the hardware surrounding the MCU and optimizing the overall software and peripheral usage are crucial factors for reducing system energy consumption. Software is not usually seen as an energy drain, yet every clock cycle consumes energy. Minimizing the number of clock cycles becomes a key challenge to reduce overall system consumption.

The Energy Profiler enables you to visualize the energy consumption of individual devices, multiple devices on one target system, or a network of interacting wireless devices to analyze and improve the power performance of these systems. Real-time information on current consumption is correlated with the program counter providing advanced energy software monitoring capabilities. It also provides a basic level of integration with the Network Analyzer network analysis tool.

In these pages the following terms will be used:

AEM: Advanced Energy Monitor

ISD: The file extension used with Multi-Node Energy Profiler and Network Analyzer files.

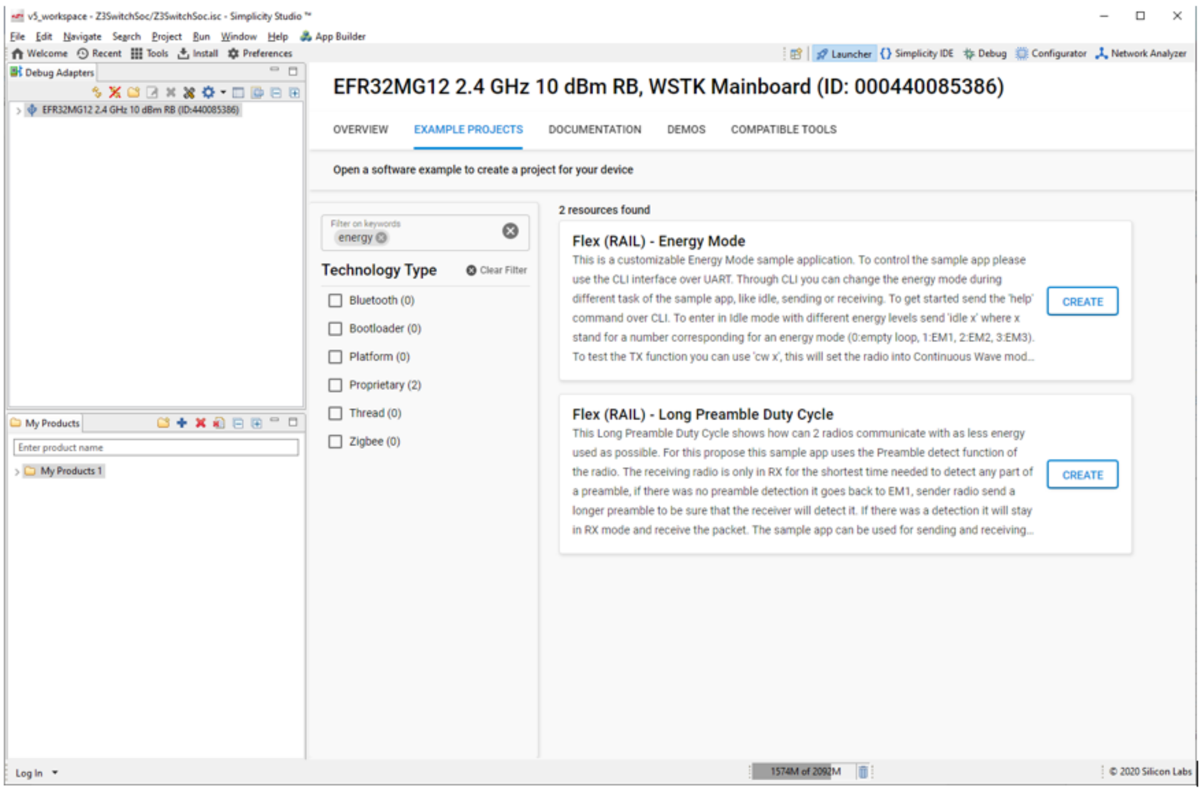
PTI: Packet Trace Information

Starting an Energy Analysis Session

An energy analysis session can be started from either the Simplicity IDE Perspective or from the Energy Profiler.

From the Simplicity IDE Perspective

Some Silicon Labs SDKs contain examples that are pre-configured to deliver energy profiling data to the Energy Profiler. In the Simplicity Studio Launcher perspective, you can find example titles such as "Powertest" or "Energy modes" or "Emode" for many development kits. In the figure below, the selected device is an EFR32 development kit. The Flex SDK has been installed.



EFR32MG12 2.4 GHz 10 dBm RB, WSTK Mainboard (ID: 000440085386)

Open a software example to create a project for your device

2 resources found

Flex (RAIL) - Energy Mode

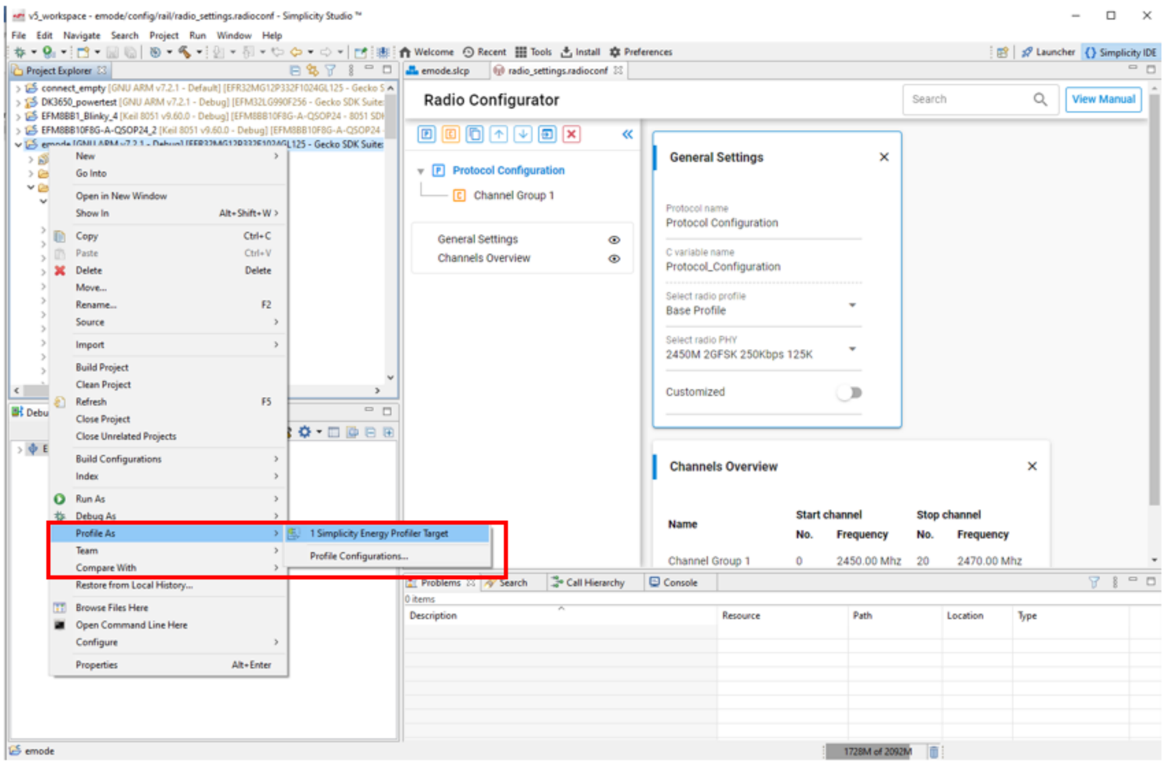
This is a customizable Energy Mode sample application. To control the sample app please use the CLI interface over UART. Through CLI you can change the energy mode during different task of the sample app, like idle, sending or receiving. To get started send the 'help' command over CLI. To enter in Idle mode with different energy levels send 'idle x' where x stand for a number corresponding for an energy mode (0:empty loop, 1:EM1, 2:EM2, 3:EM3). To test the TX function you can use 'cw x', this will set the radio into Continuous Wave mod...

Flex (RAIL) - Long Preamble Duty Cycle

This Long Preamble Duty Cycle shows how can 2 radios communicate with as less energy used as possible. For this propose this sample app uses the Preamble detect function of the radio. The receiving radio is only in RX for the shortest time needed to detect any part of a preamble, if there was no preamble detection it goes back to EM1, sender radio send a longer preamble to be sure that the receiver will detect it. If there was a detection it will stay in RX mode and receive the packet. The sample app can be used for sending and receiving...

Create an example (in this case **Flex(RAIL) - Energy Mode**). Because this is a RAIL example, the Simplicity IDE perspective opens with the Radio Configurator open. If this is an AppBuilder project, click **Generate**.

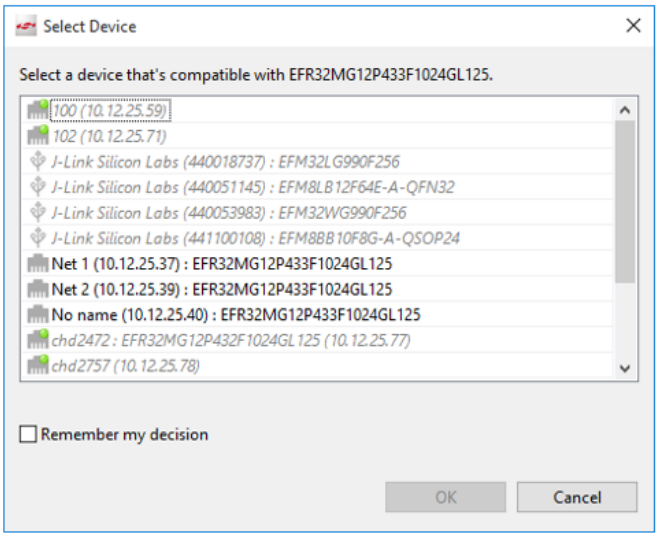
Right-click the project directory. In the context menu select **Profile As > Simplicity Energy Profiler Target**.



SSv5 then:

1. Rebuilds the entire project.
2. Flashes the application to compatible hardware.
3. Starts energy capture for that device.
4. Switches to the Energy Profiler perspective.
5. Displays the device data on Single-Node View, Multi-Node View, and Scope View.

In step 2 above, if more than one connected device is compatible with the application being profiled, SSv5 prompts you to select a device. Select the target of interest and click **OK**.


Select Device

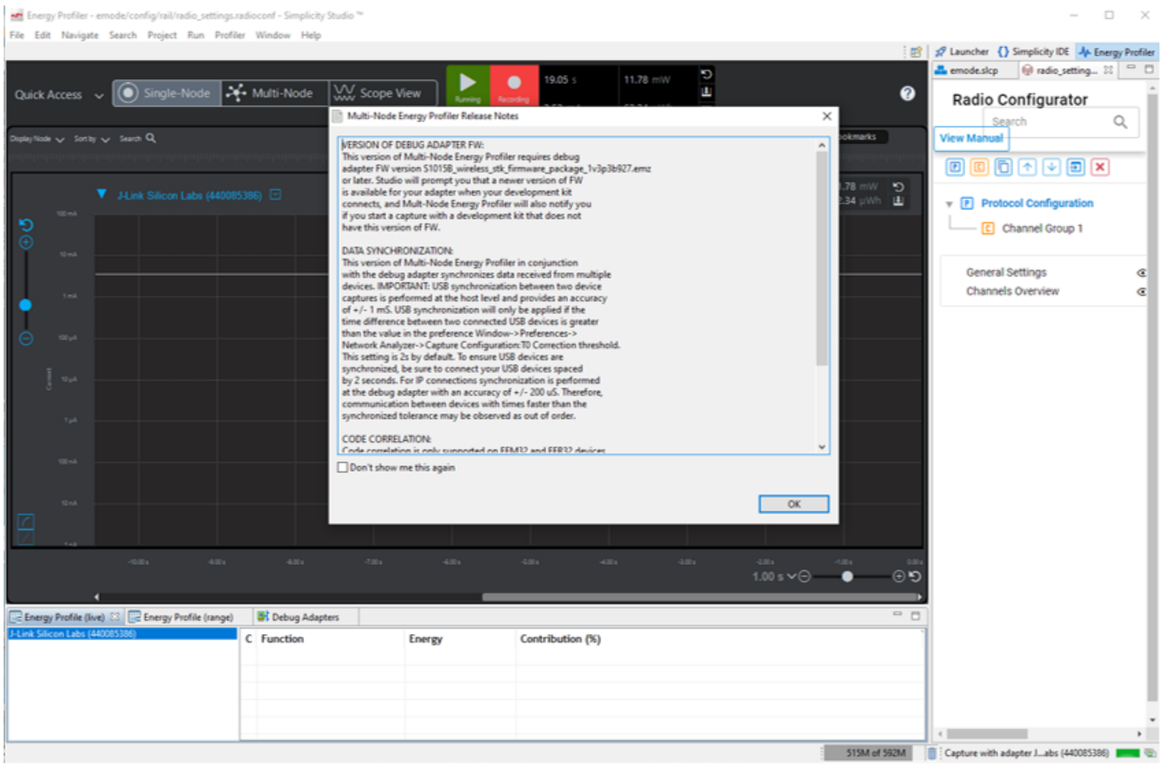
Select a device that's compatible with EFR32MG12P433F1024GL125.

100 (10.12.25.59)
102 (10.12.25.71)
J-Link Silicon Labs (440018737) : EFM32LG990F256
J-Link Silicon Labs (440051145) : EFM8LB12F64E-A-QFN32
J-Link Silicon Labs (440053983) : EFM32WG990F256
J-Link Silicon Labs (441100108) : EFM8BB10F8G-A-QSOP24
Net 1 (10.12.25.37) : EFR32MG12P433F1024GL125
Net 2 (10.12.25.39) : EFR32MG12P433F1024GL125
No name (10.12.25.40) : EFR32MG12P433F1024GL125
chd2472 : EFR32MG12P432F1024GL125 (10.12.25.77)
chd2757 (10.12.25.78)

☐ Remember my decision

OK Cancel

After the application is built and the firmware flashed, the Energy Profiler perspective is displayed, with a release notes dialog. Click **OK**.



Multi-Node Energy Profiler Release Notes

VERSION OF DEBUG ADAPTER FW:
This version of Multi-Node Energy Profiler requires debug adapter FW version S1015B_wireless_stk_firmware_package_1v3p3b927.emz or later. Studio will prompt you that a newer version of FW is available for your adapter when your development kit connects, and Mult-Node Energy Profiler will also notify you if you start a capture with a development kit that does not have this version of FW.

DATA SYNCHRONIZATION:
This version of Multi-Node Energy Profiler in conjunction with the debug adapter synchronizes data received from multiple devices. IMPORTANT: USB synchronization between two device captures is performed at the host level and provides an accuracy of +/- 1 mS. USB synchronization will only be applied if the time difference between two connected USB devices is greater than the value in the preference Window->Preferences->Network Analyzer->Capture Configuration:T0 Correction threshold. This setting is 2s by default. To ensure USB devices are synchronized, be sure to connect your USB devices spaced by 2 seconds. For IP connections synchronization is performed at the debug adapter with an accuracy of +/- 200 uS. Therefore, communication between devices with times faster than the synchronized tolerance may be observed as out of order.

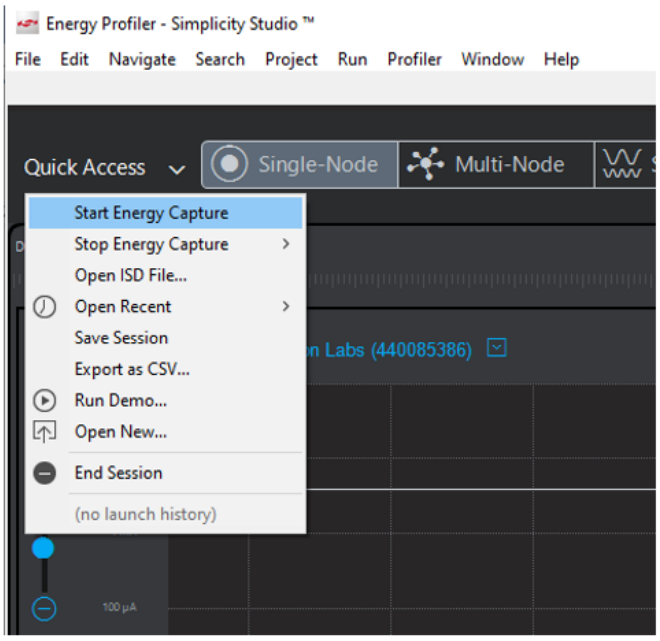
CODE CORRELATION:

☐ Don't show me this again

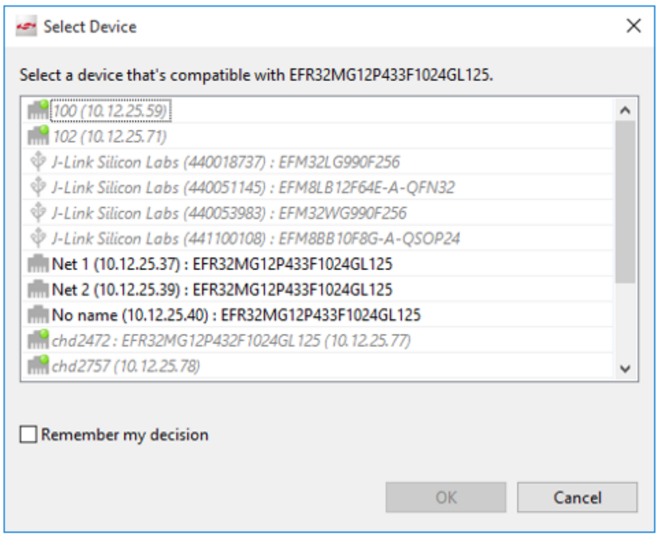
OK

From the Energy Profiler

If application firmware is already running on a device, you can connect to it from the Energy Profiler tool. Open the Energy Profiler perspective. Click **Quick Access** to display its menu and select **Start Energy Capture**.



If more than one connected device is compatible with the application being profiled, SSv5 prompts you to select a device. Select the target of interest and click **OK**.


Select Device

Select a device that's compatible with EFR32MG12P433F1024GL125.

100 (10.12.25.59)
102 (10.12.25.71)
J-Link Silicon Labs (440018737) : EFM32LG990F256
J-Link Silicon Labs (440051145) : EFM8LB12F64E-A-QFN32
J-Link Silicon Labs (440053983) : EFM32WG990F256
J-Link Silicon Labs (441100108) : EFM8BB10F8G-A-QSOP24
Net 1 (10.12.25.37) : EFR32MG12P433F1024GL125
Net 2 (10.12.25.39) : EFR32MG12P433F1024GL125
No name (10.12.25.40) : EFR32MG12P433F1024GL125
chd2472 : EFR32MG12P432F1024GL125 (10.12.25.77)
chd2757 (10.12.25.78)

☐ Remember my decision

Energy Profiler then:

- Starts energy capture for that device.
- Depending upon which view is currently active, displays the device’s energy data on Single-Node view, Multi-Node view, or Scope view.

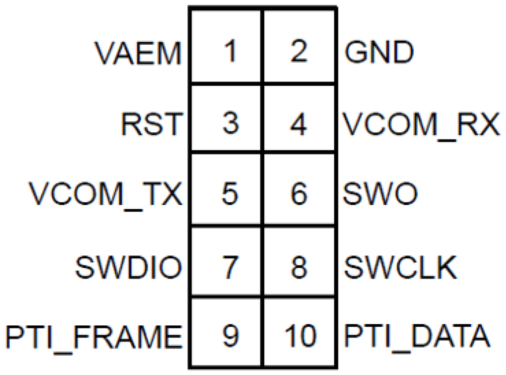
Customer Hardware and Software Design Information

This section discusses design requirements in order for a device to be able to use Energy Profiler features. Among these features is Code Correlation, which connects power consumption and executed code. Using code correlation is discussed in more detail in Profiling with Code Correlation

Hardware Design

EFM8 Hardware Interface

To use the Energy Profiler functionality on a board design, the board needs to include a debug interface that can be connected to a Silicon Labs starter kit (STK) or wireless starter kit (WSTK). For the basic Energy Profiler current measurements, the board must be powered from the VAEM supply of a Silicon Labs STK or WSTK. Both the current measurement and EFM8 C2 Debug interface can be obtained by using the Silicon Labs Mini Simplicity 10 pin connector. This connector is detailed in AN958: Debugging and Programming Interfaces for Custom Designs. The EFM8 C2 interface signals are on SWDIO (Pin 7 C2D) and SWCLK (Pin 8 C2CK) pins shown in the following figure.

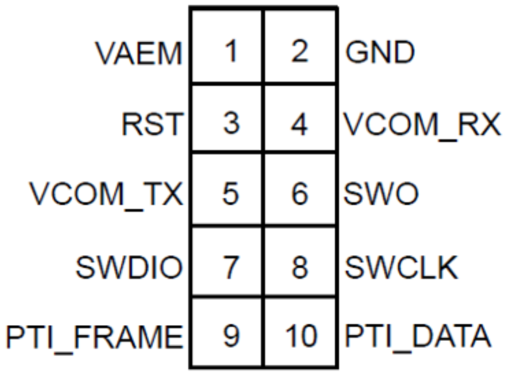


Code correlation is not possible with the EFM8 parts as they do not include the SWO pin that is used to transmit the program counter information. As mentioned in AN958: Debugging and Programming Interfaces for Custom Designs, using the

Silicon Labs debug adapter board (BRD8010A) is the easiest way to get the Mini Simplicity pinout from a Silicon Labs STK or WSTK development kit.

EFM32 and EFR32 Hardware Interface

To use the Energy Profiler functionality on a board design, the board needs to include a debug interface that can be connected to a Silicon Labs STK or WSTK. For the basic Energy Profiler current measurements, the board must be powered from the VAEM supply of a Silicon Labs STK or WSTK. To also include code correlation, the debug interface must include the SWD interface. Both the current measurement and the code correlation (SWD) can be obtained by using the Silicon Labs Mini Simplicity 10 pin connector. This connector is detailed in AN958: Debugging and Programming Interfaces for Custom Designs. The Mini Simplicity 10 pin connector can be used with all EFM32 and EFR32 parts. The pinout is shown in the following figure.



As mentioned in AN958: Debugging and Programming Interfaces for Custom Designs, using the Silicon Labs debug adapter board (BRD8010A) is the easiest way to get the Mini Simplicity pinout from a Silicon Labs STK or WSTK development kit. The debug adapter board is not compatible with the older EFM32 Development Kits (DKs) and some of the older EFM32 starter kits (STK) that have a different debug connector on them (Gecko, Giant Gecko (EFM32GG-STK3700), Leopard Gecko, Tiny Gecko, Wonder Gecko, Zero Gecko).

Software Design

To use the current monitoring functionality provided by the AEM interface, no software changes or setup are required. To use the code correlation functionality, the SWD interface must be configured to output periodic program counter information. Currently, code correlation is not possible with the EFM8 family. The Multi-Node Energy Profiler can still be used to monitor the overall energy use of the parts over time / usage scenarios.

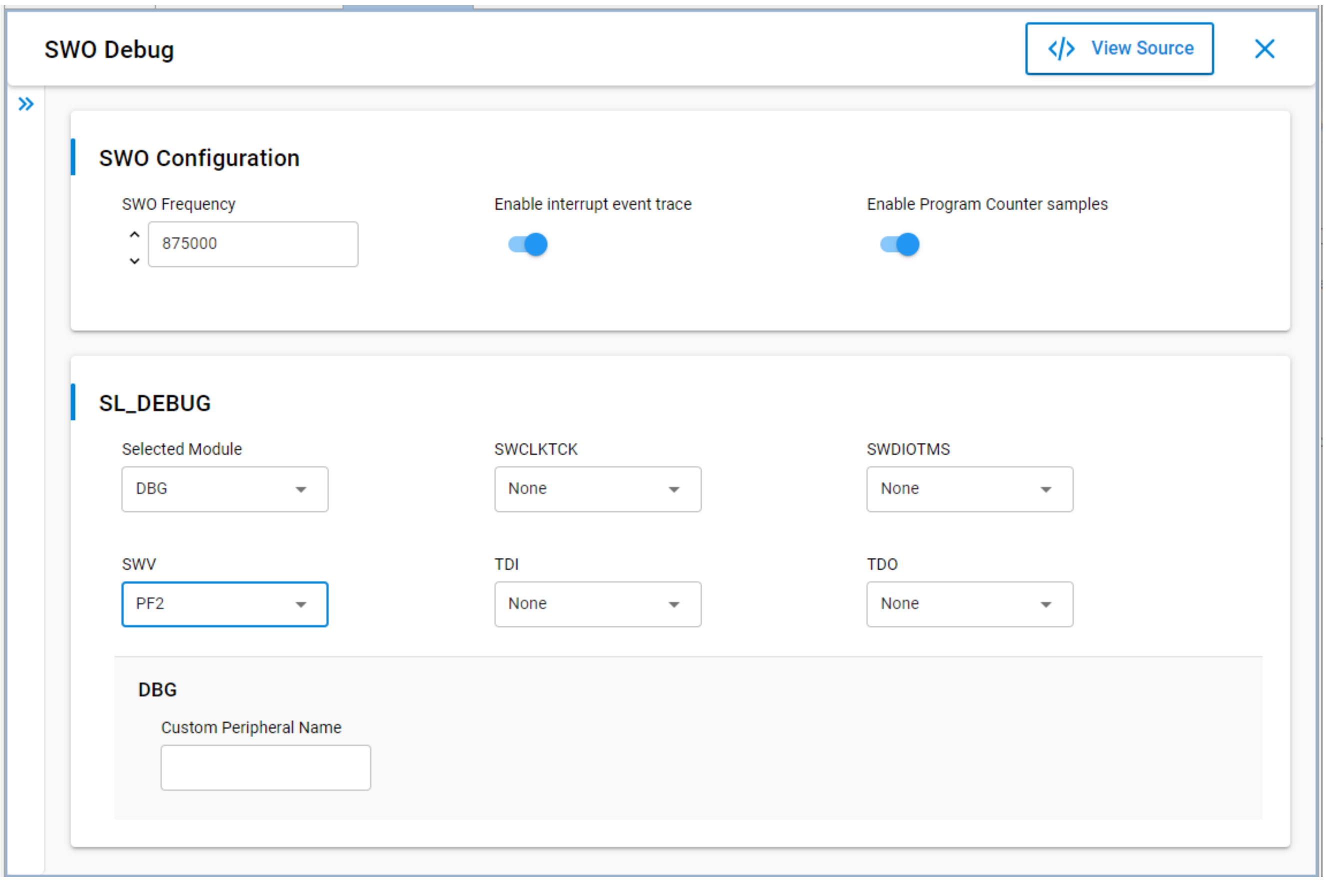
Software can be configured in two ways:

- Project configurator method (using software components; preferred for SSv5 projects)
- Legacy method (by manually adding the necessary source code)

Project Configurator Method

This method of enabling code correlation is preferred in SSv5, as it leverages the software components and thereby reduces the chance of error.

1. In the Project Configurator, SOFTWARE COMPONENTS tab, find and install the **SWO Debug** component.
2. Click Configure to open the Component editor.
3. Open its configuration file and turn on **Enable interrupt event trace** and **Enable program counter samples**.
4. If the device allows SWO functionality on multiple GPIO pins, in the SL_Debug card's **SWV** field, select the pin to be used as the SWO output.



SWO Debug

SWO Configuration

SWO Frequency: 875000

Enable interrupt event trace: on

Enable Program Counter samples: on

SL_DEBUG

Selected Module: DBG

SWCLKTCK: None

SWDIOTMS: None

SWV: PF2

TDI: None

TDO: None

DBG

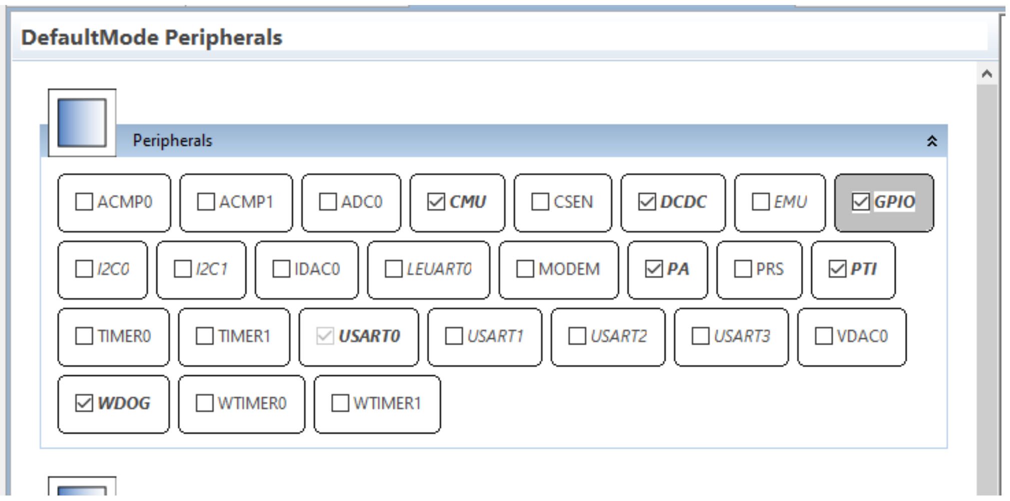
Custom Peripheral Name:

Further initialization of the software component in code is not required provided that the **System Init** component is installed in the project.

Legacy Method

This method should be used for Appbuilder project and projects not created in the Project Configurator.

1. The following paths must be included in your project, as they provide access to relevant header files used to enable code correlation. These header files are readily available for Silicon Labs parts. For custom design boards use bsp_trace.h and traceconfig.h as a reference instead.
   1. <SSv5 install>/developer/sdks/gecko_sdk_suite/<version>/hardware/kit/common/bsp
      - Grants access to bsp_trace.h
   2. (not required for AppBuilder projects) <SSv5 install>/developer/sdks/gecko_sdk_suite/<version>/hardware/kit/kitname/config
      - Grants access to traceconfig.h
      - For AppBuilder projects, instead see the next step.
2. (Only for Appbuilder projects) In the Hardware Configurator file (.hwconf) in the "DefaultMode" peripheral tab, enable the GPIO module. This adds the **BSP_TRACE_SWO_** macros to the hal-config.h file.


DefaultMode Peripherals

Peripherals

☐ ACMP0 ☐ ACMP1 ☐ ADC0 ☑ CMU ☐ CSEN ☑ DCDC ☐ EMU ☑ GPIO
☐ I2C0 ☐ I2C1 ☐ IDAC0 ☐ LEUART0 ☐ MODEM ☑ PA ☐ PRS ☑ PTI
☐ TIMER0 ☐ TIMER1 ☑ USART0 ☐ USART1 ☐ USART2 ☐ USART3 ☐ VDAC0
☑ WDOG ☐ WTIMER0 ☐ WTIMER1

3. Add bsp_trace.c to the project (from <SSv5install>/developer/sdks/gecko_sdk_suite/<version>/hardware/kit/common/bsp)
4. The code must enable SWO output from the MCU. To enable this output, add `#include bsp_trace.h` to the appropriate module and call `BSP_TraceSwoSetup()` during initialization as indicated in the following sections, based on the SDK being used.

Stackless project MCU SDK: Place `BSP_TraceSwoSetup()` after the `EMU_DCDCInit()` call in main().

```
#include "bsp_trace.h"
.
.
.
/* Initialize DCDC. Always start in low-noise mode. */
dcdcInit.dcdcMode = emuDcdcMode_LowNoise;
EMU_DCDCInit(&dcdcInit);
// Setup SWD for code correlation
BSP_TraceSwoSetup();
```

Bluetooth LE SDK: Place `BSP_TraceProfilerSetup()` in main.c after the `initApp()` call.

```
#include "bsp_trace.h"
.
.
.
// Initialize application
initApp();

// Setup SWD for code correlation
BSP_TraceProfilerSetup();

// Initialize LEDs
BSP_LedsInit();
```

Flex SDK: Place `BSP_TraceProfilerSetup()` in main.c after the `BSP_Init()` call.

```
#include "hal_common.h"
#include "bsp_trace.h"
.
.
.
// Initialize the BSP
BSP_Init(BSP_INIT_BCC);

// Setup SWD for code correlation
BSP_TraceProfilerSetup();
```

EmberZNet SDK: Place `BSP_TraceProfilerSetup()` in af-main-soc.c after the `emberInit()` call.

```
#include "afv2-bookkeeping.h"

#if defined(CORTEXM3_EFR32_MICRO) || defined (CORTEXM3_EMBER_MICRO)
  #define EXTENDED_RESET_INFO
  #include "hal/micro/cortexm3/ diagnostic.h"
#endif
#include "bsp_trace.h"
.
.
.
int emberAfMain (MAIN_FUNCTION_PARAMETERS)
{
  EmberStatus status;

    int returnCode;
    if (emberAfMainStartCallback(& returnCode, APP_FRAMEWORK_MAIN_ARGUMENTS)) {
      return returnCode;
  }
}

// Initialize the Ember Stack.
status = emberInit();

if (status != EMBER_SUCCESS) {
  emberAfCorePrintln("%pemberInit  0x%x", "ERROR: ", status);

  // The app can choose what to do  here. If the app is running
  // another device then it could  stay running and report the
  // error visually for example. This  app asserts.
  assert(false);
} else {
  emberAfDebugPrintln("init pass");
  }
// Setup SWD for code correlation
BSP_TraceProfilerSetup();
```

The program must be built with debug information enabled so that source code lookup is possible. If you create your project in Simplicity Studio, this is enabled by default. If you import a project into Simplicity Studio, check compiler options in project context menu **Properties...-> C/C++ Build -> Settings**. If you build the program outside of Simplicity Studio, check compiler options of the build tools.

EFM32 Software Configuration

1. Add bsp_trace.c to the project. (from <SSv5 install>\developer\sdks\gecko_sdk_suite&lt;version>\hardware\kit\common\bsp)
2. The macros BSP_TRACE_SWO_PIN, BSP_TRACE_SWO_PORT and BSP_TRACE_SWO_LOC must be defined. For EFM32 parts this is done by default for Silicon Labs development kits in <SSv5 install>\developer\sdks\gecko_sdk_suite\<version>\hardware\kit\<board>/hal-config-board.h. For a custom board design the Silicon Labs hal-config-board.h can be used as an example for defining the values. The following is an example for the Pearl Gecko Starter Kit (SLSTK3401A):

```
// $[GPIO]
#define   PORTIO_GPIO_SWV_PIN                    (2)
#define   PORTIO_GPIO_SWV_PORT                   (gpioPortF)
#define   PORTIO_GPIO_SWV_LOC                    (0)

#define   BSP_TRACE_SWO_PIN                      (2)
#define   BSP_TRACE_SWO_PORT                     (gpioPortF)
#define   BSP_TRACE_SWO_LOC                      (0)
// [GPIO]$
```

3. The code must enable SWO output from the MCU. To enable this output, add `#include bsp_trace.h` to main.c and call `BSP_TraceProfilerSetup()` during initialization after the `EMU_DCDCInit()` call in `main()`.

```
#include
#include "bsp_trace.h"
.
.
.
/* Initialize DCDC. Always start in   low-noise mode. */
dcdcInit.dcdcMode = emuDcdcMode_LowNoise;
EMU_DCDCInit(&dcdcInit);

// Setup SWD for code correlation
BSP_TraceProfilerSetup();
```

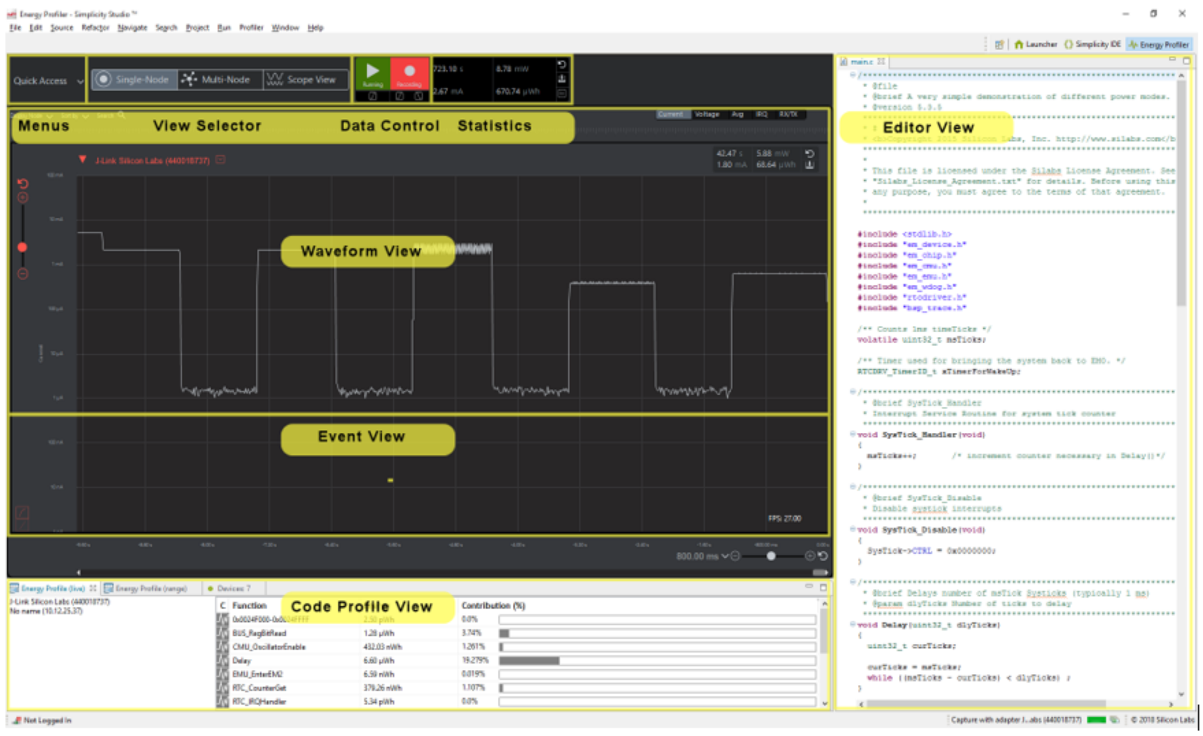
4. The program must be built with debug information enabled so that source code lookup is possible. If you create your project in SSv5, this is enabled by default. If you import a project into SSv5, check compiler options in project context menu **Properties... > C/C++ Build > Settings**. If you build your program outside of SSv5, check compiler options of the build tools.

User Interface

Energy Profiler User Interface

The Energy Profiler perspective is divided into six primary sections. The following figure shows Single-Node view.

- Quick Access Menus - The most frequently used menus for Multi-Node Energy Profiler are available on the user interface itself and also from the Profiler menu at the top menu bar.
- View Selector - Three views for Multi-Node Energy Profiler can be selected by clicking the View Selector bar.
- Data Control - Controls in this area are used to enable and disable data flow to the user interface and to disk.
- Statistics - Energy data for the waveform is shown in the statistics area.
- Code Correlation - Function execution related to power consumption is shown in the Code Correlation view.
- Editor - Source code for the program is shown here and works in conjunction with the Code Correlation view.



main.c

```
/***************************************************************************
 * @file
 * @brief A very simple demonstration of different power modes.
 * @version 5.3.5
 ***************************************************************************
 * # License
 * <b>Copyright 2015 Silicon Labs, Inc. http://www.silabs.com</b
 ***************************************************************************
 *
 * This file is licensed under the Silabs License Agreement. See
 * "Silabs_License_Agreement.txt" for details. Before using this
 * any purpose, you must agree to the terms of that agreement.
 *
 ***************************************************************************

#include <stdlib.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "em_wdog.h"
#include "rtcdriver.h"
#include "bsp_trace.h"

/** Counts 1ms timeTicks */
volatile uint32_t msTicks;

/** Timer used for bringing the system back to EM0. */
RTCDRV_TimerID_t xTimerForWakeUp;

/***************************************************************************
 * @brief SysTick_Handler
 * Interrupt Service Routine for system tick counter
 ***************************************************************************
void SysTick_Handler(void)
{
  msTicks++;       /* increment counter necessary in Delay()*/
}

/***************************************************************************
 * @brief SysTick_Disable
 * Disable systick interrupts
 ***************************************************************************
void SysTick_Disable(void)
{
  SysTick->CTRL = 0x0000000;
}

/***************************************************************************
 * @brief Delays number of msTick Systicks (typically 1 ms)
 * @param dlyTicks Number of ticks to delay
 ***************************************************************************
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}
```

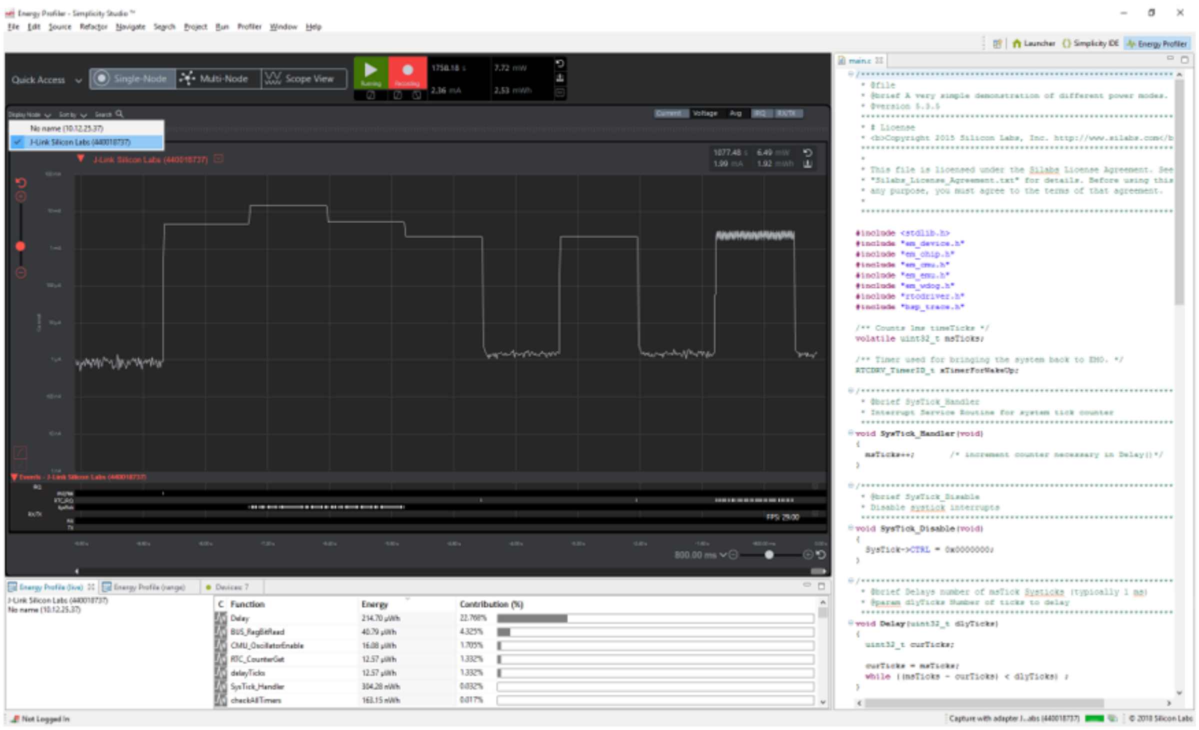
Function		Contribution (%)
0x0024F000-0x0024FFFF	2.50 pWh	0.0%
BUS_RegBitRead	1.28 µWh	3.74%
CMU_OscillatorEnable	432.03 nWh	1.261%
Delay	6.60 µWh	19.279%
EMU_EnterEM2	6.59 nWh	0.019%
RTC_CounterGet	379.26 nWh	1.107%
RTC_IRQHandler	5.34 pWh	0.0%

Three views are available for evaluating power performance:

- Single-Node View
- Multi-Node View
- Scope View

Single-Node View

Single-Node view is used to display a single device’s waveform and events. It displays a high-detail waveform and ensures that all events rows are visible, without the need to scroll the UI. The Energy Profiler perspective combines this Single-Node view, the Code Correlation view, and the source code Editor view, allowing you to perform in-depth analysis of a single device from many viewpoints. Any active device may be selected through the **Display Nodes** menu.



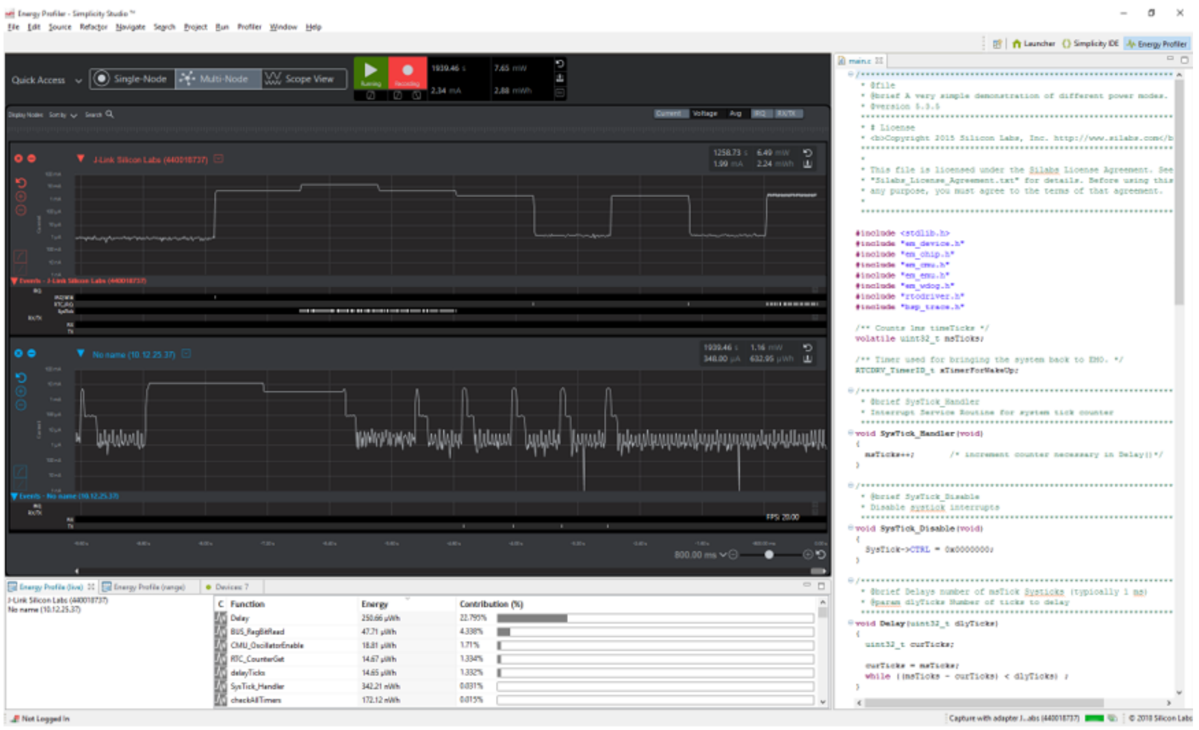
```
/*******************************************************************
 * @file
 * @brief A very simple demonstration of different power modes.
 * @version 5.3.5
 *******************************************************************
 * # License
 * <b>Copyright 2015 Silicon Labs, Inc. http://www.silabs.com</b
 *******************************************************************
 *
 * This file is licensed under the Silabs License Agreement. See
 * "Silabs_License_Agreement.txt" for details. Before using this
 * any purpose, you must agree to the terms of that agreement.
 *
 *******************************************************************

#include <stdlib.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "em_wdog.h"
#include "rtcdriver.h"
#include "bsp_trace.h"

/** Counts 1ms timeTicks */
volatile uint32_t msTicks;

/** Timer used for bringing the system back to EM0. */
RTCDRV_TimerID_t xTimerForWakeUp;

/*******************************************************************
 * @brief SysTick_Handler
 * Interrupt Service Routine for system tick counter
 *******************************************************************
void SysTick_Handler(void)
{
  msTicks++;       /* increment counter necessary in Delay()*/
}

/*******************************************************************
 * @brief SysTick_Disable
 * Disable systick interrupts
 *******************************************************************
void SysTick_Disable(void)
{
  SysTick->CTRL = 0x0000000;
}

/*******************************************************************
 * @brief Delays number of msTick Systicks (typically 1 ms)
 * @param dlyTicks Number of ticks to delay
 *******************************************************************
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}
```

J-Link Silicon Labs (440018737)
No name (10.12.25.37)

C	Function	Energy	Contribution (%)
	Delay	214.70 µWh	22.768%
	BUS_RegBitRead	40.79 µWh	4.325%
	CMU_OscillatorEnable	16.08 µWh	1.705%
	RTC_CounterGet	12.57 µWh	1.332%
	delayTicks	12.57 µWh	1.332%
	SysTick_Handler	304.28 nWh	0.032%
	checkAllTimers	163.15 nWh	0.017%

Multi-Node View

Multi-Node view is used to display multiple device waveform and events. It allows you to investigate system behavior and interactions between devices. Two devices are visible in the main portion of the UI, while others are viewed by scrolling. The Multi-Node view groups each individual device's waveform and events together. The Energy Profiler perspective combines this Multi-Node view, the Code Correlation view, and the source code Editor view, allowing you to perform in-depth analysis of multiple devices from many viewpoints.



J-Link Silicon Labs (440018737)
No name (10.12.25.37)

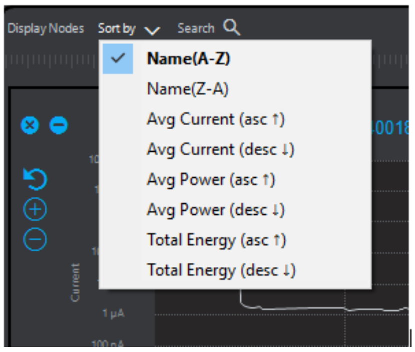
C	Function	Energy	Contribution (%)
	Delay	250.66 µWh	22.795%
	BUS_RegBitRead	47.71 µWh	4.338%
	CMU_OscillatorEnable	18.81 µWh	1.71%
	RTC_CounterGet	14.67 µWh	1.334%
	delayTicks	14.65 µWh	1.332%
	SysTick_Handler	342.21 nWh	0.031%
	checkAllTimers	172.12 nWh	0.015%

Multi-Node view offers the following capabilities:

- Sorting
- Node display control

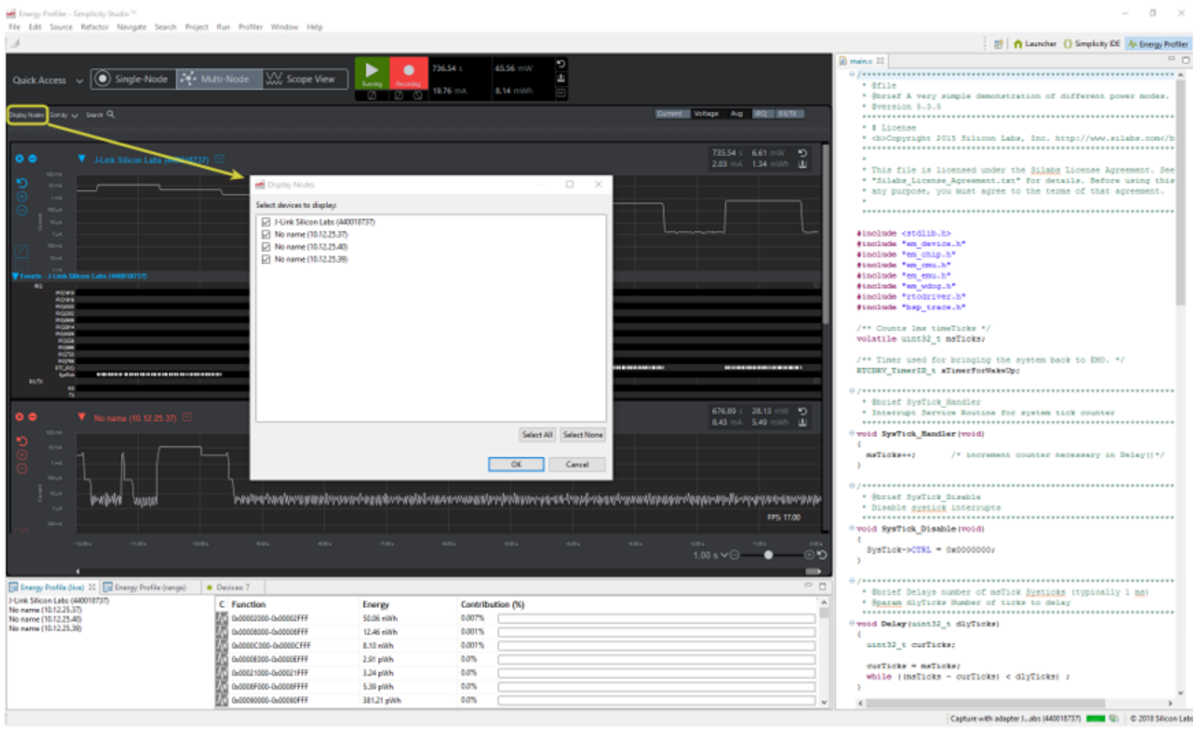
Sorting

Sorting in Multi-Node view allows you to order waveforms based upon several criteria. For example, it may be useful to sort descending by average power, such that the highest power consumers are easily identified at the top. As shown in the following figure, devices may be sorted ascending and descending by name, average current, average power, or total energy.



Displaying Nodes

Enabling capture on any device automatically adds it to the Multi-Node view. However, as the number of devices displayed increases, UI performance may be impacted. Click **Display Nodes** to select the devices that will be displayed. This selection only affects the waveform display. If recording is enabled, data for all actively capturing devices is streamed to disk regardless of whether or not they are displayed.



Display Nodes

Select devices to display:

- [✓] J-Link Silicon Labs (440018737)
- [✓] No name (10.12.25.37)
- [✓] No name (10.12.25.40)
- [✓] No name (10.12.25.38)

Select All | Select None

OK | Cancel

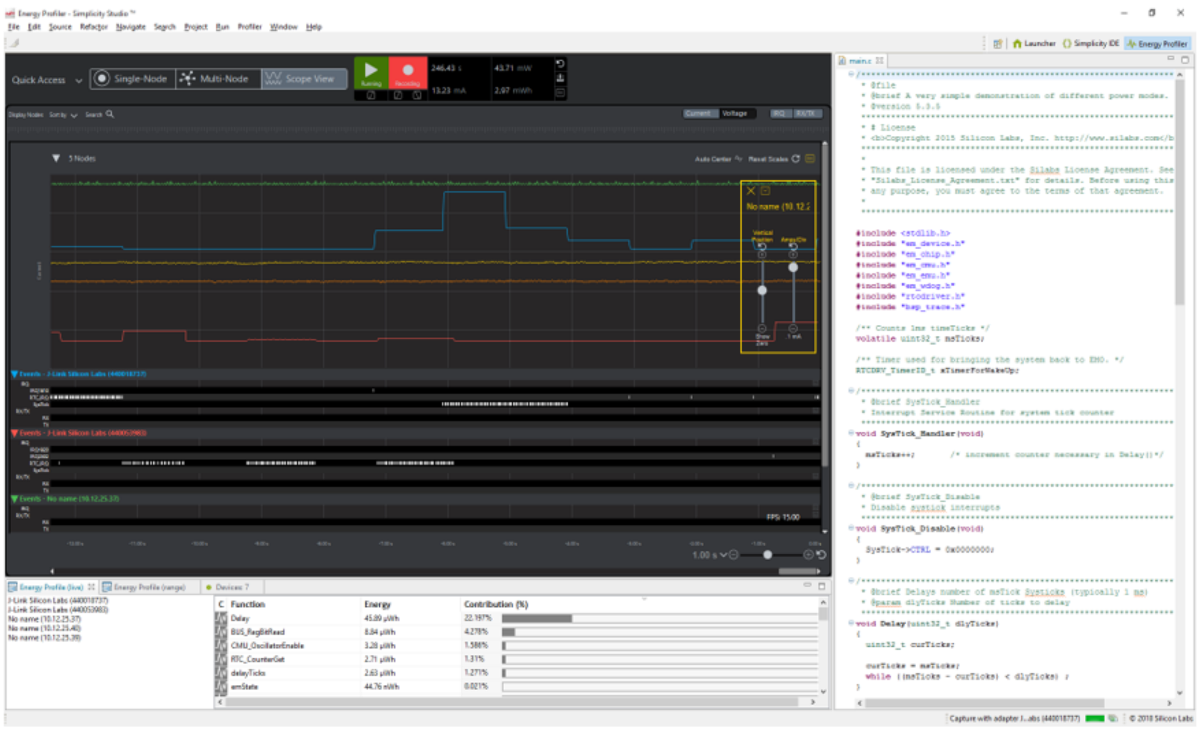
J-Link Silicon Labs (440018737)
No name (10.12.25.37)
No name (10.12.25.40)
No name (10.12.25.38)

C	Function	Energy	Contribution (%)
	0x00002000-0x00002FFF	50.06 nWh	0.007%
	0x00008000-0x00008FFF	12.46 nWh	0.001%
	0x0000C000-0x0000CFFF	8.10 nWh	0.001%
	0x0000E000-0x0000EFFF	2.91 pWh	0.0%
	0x00021000-0x00021FFF	3.24 pWh	0.0%
	0x0000F000-0x0000FFFF	5.39 pWh	0.0%
	0x00000000-0x00000FFF	381.21 pWh	0.0%

```
/***************************************************************************
 * @file
 * @brief A very simple demonstration of different power modes.
 * @version 5.3.5
 ***************************************************************************
 * # License
 * <b>Copyright 2015 Silicon Labs, Inc. http://www.silabs.com</b
 ***************************************************************************
 *
 * This file is licensed under the Silabs License Agreement. See
 * "Silabs_License_Agreement.txt" for details. Before using this
 * any purpose, you must agree to the terms of that agreement.
 *
 ***************************************************************************

#include <stdlib.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "em_wdog.h"
#include "rtcdriver.h"
#include "bsp_trace.h"

/** Counts 1ms timeTicks */
volatile uint32_t msTicks;

/** Timer used for bringing the system back to EM0. */
RTCDRV_TimerID_t xTimerForWakeUp;

/***************************************************************************
 * @brief SysTick_Handler
 * Interrupt Service Routine for system tick counter
 ***************************************************************************
void SysTick_Handler(void)
{
  msTicks++;       /* increment counter necessary in Delay()*/
}

/***************************************************************************
 * @brief SysTick_Disable
 * Disable systick interrupts
 ***************************************************************************
void SysTick_Disable(void)
{
  SysTick->CTRL = 0x0000000;
}

/***************************************************************************
 * @brief Delays number of msTick Systicks (typically 1 ms)
 * @param dlyTicks Number of ticks to delay
 ***************************************************************************
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}
```

Scope View

Scope view is used to display multiple device waveforms and events, allowing you to investigate system behavior and interactions between devices. In Scope view, all waveforms are displayed in one waveform area, so that you can see all waveforms without needing to scroll the UI. It also enables the user overlay and align traces like an oscilloscope. All event traces are grouped below the waveform view. The Energy Profiler perspective combines this Scope view, the Code Correlation view, and the source code Editor view allowing you to perform in-depth analysis of a single device from many viewpoints.


Energy Profiler - Simplicity Studio™
File Edit Source Refactor Navigate Search Project Run Profiler Window Help
Launcher
Simplicity IDE
Energy Profiler
Quick Access
Single-Node
Multi-Node
Scope View
Current
Voltage
IRQ
RX/TX
5 Nodes
Auto Center
Rescal Scales
No name (10.12.2
Events - J-Link Silicon Labs (440018737)
Events - J-Link Silicon Labs (440053983)
Events - No name (10.12.25.37)
FPS: 15.00
1.00 s
Energy Profile (live)
Energy Profile (range)
Devices: 7
main.c
Capture with adapter J...abs (440018737)
© 2018 Silicon Labs

```
/***************************************************************
 * @file
 * @brief A very simple demonstration of different power modes.
 * @version 5.3.5
 ***************************************************************
 * # License
 * <b>Copyright 2015 Silicon Labs, Inc. http://www.silabs.com</b
 ***************************************************************
 *
 * This file is licensed under the Silabs License Agreement. See
 * "Silabs_License_Agreement.txt" for details. Before using this
 * any purpose, you must agree to the terms of that agreement.
 *
 ***************************************************************

#include <stdlib.h>
#include "em_device.h"
#include "em_chip.h"
#include "em_cmu.h"
#include "em_emu.h"
#include "em_wdog.h"
#include "rtcdriver.h"
#include "bsp_trace.h"

/** Counts 1ms timeTicks */
volatile uint32_t msTicks;

/** Timer used for bringing the system back to EM0. */
RTCDRV_TimerID_t xTimerForWakeUp;

/***************************************************************
 * @brief SysTick_Handler
 * Interrupt Service Routine for system tick counter
 ***************************************************************
void SysTick_Handler(void)
{
  msTicks++;       /* increment counter necessary in Delay()*/
}

/***************************************************************
 * @brief SysTick_Disable
 * Disable systick interrupts
 ***************************************************************
void SysTick_Disable(void)
{
  SysTick->CTRL = 0x0000000;
}

/***************************************************************
 * @brief Delays number of msTick Systicks (typically 1 ms)
 * @param dlyTicks Number of ticks to delay
 ***************************************************************
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}
```

J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440053983)
No name (10.12.25.37)
No name (10.12.25.40)
No name (10.12.25.39)

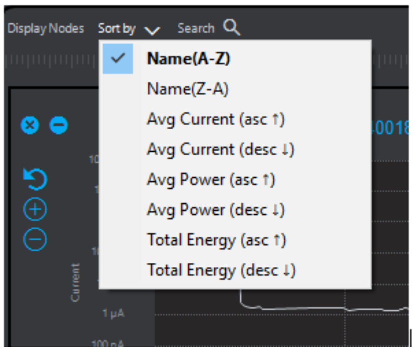
C	Function	Energy	Contribution (%)
	Delay	45.89 µWh	22.197%
	BUS_RegBitRead	8.84 µWh	4.278%
	CMU_OscillatorEnable	3.28 µWh	1.586%
	RTC_CounterGet	2.71 µWh	1.31%
	delayTicks	2.63 µWh	1.271%
	emState	44.76 nWh	0.021%

Scope view offers the following capabilities:

- Sorting
- Node display control
- Waveform display control

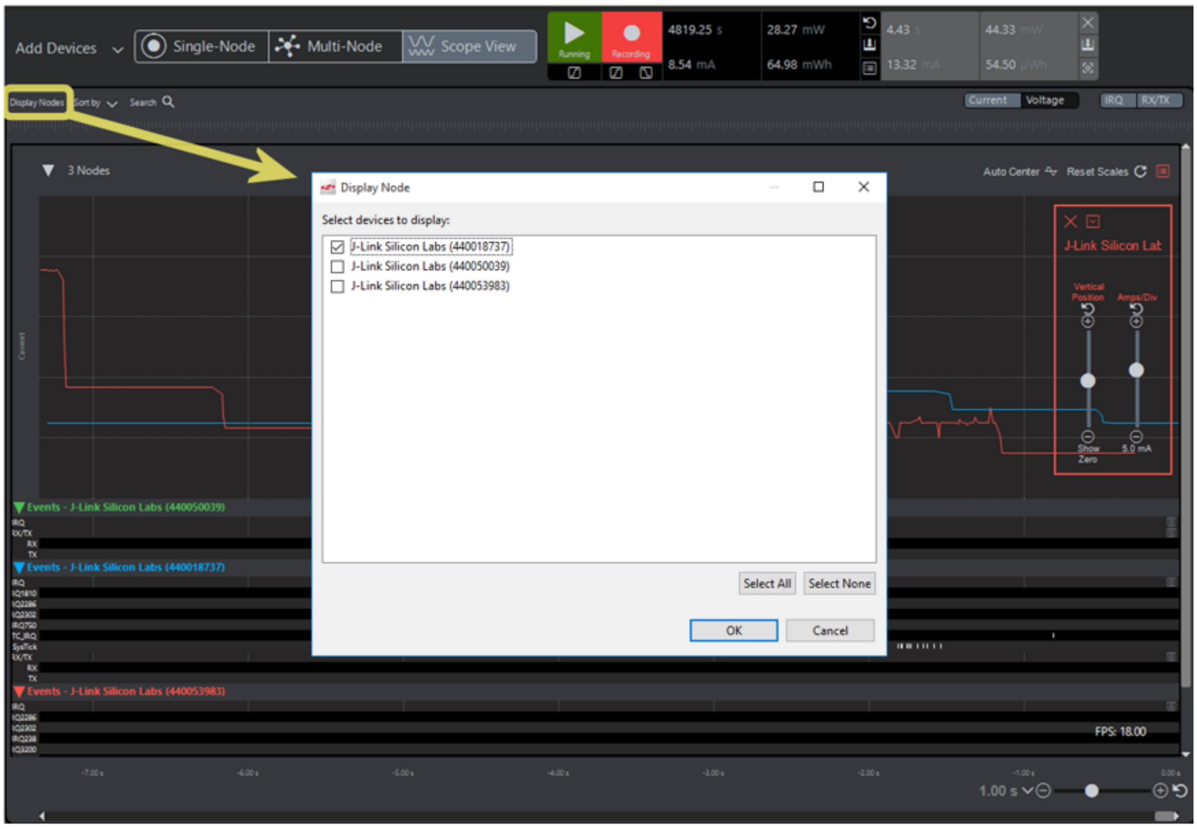
Sorting

Sorting in Scope view allows you to order events based upon several criteria. For example, it may be useful to sort events descending by average power, such that the events for the highest power consumers are easily identified at the top.



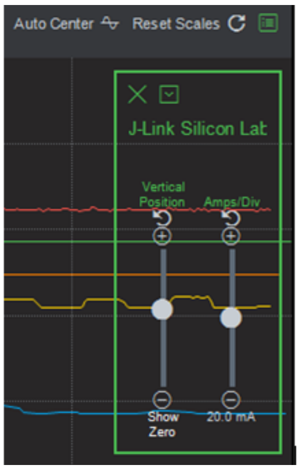
Displaying Nodes

Enabling capture on any device automatically adds it to scope view. However, as the number of devices displayed increases, UI performance may be impacted. Click **Display Nodes** to select which devices will be displayed. This selection only enables or disables display of waveform and event data in the UI. If recording is enabled, data for all actively capturing devices is streamed to disk regardless of whether they are displayed.

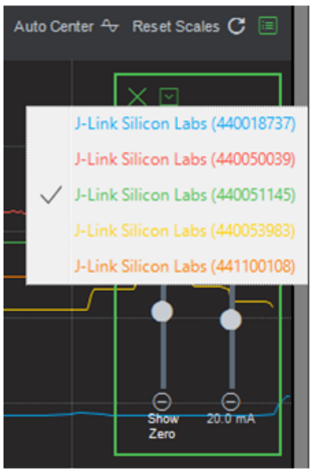


Displaying Waveforms

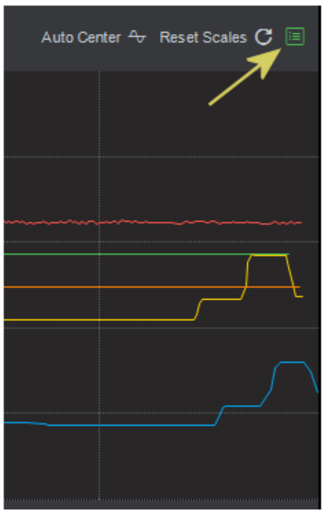
One of the Scope view's key benefits is the ability to view waveforms in a single view at difference scales, enabling you to compare waveform characteristics that would not be possible for two waveforms on the same scale. The waveform scale/offset controls are available by default at the upper right corner of the Scope waveform view. The color of the waveform scale/offset control indicates which waveform will be adjusted when the sliders are moved.



The drop-down menu at the top of the control allows you to select the device to be adjusted by the waveform scale and offset control. The debug adapter name colors match the colors in the waveform portion of Scope view.

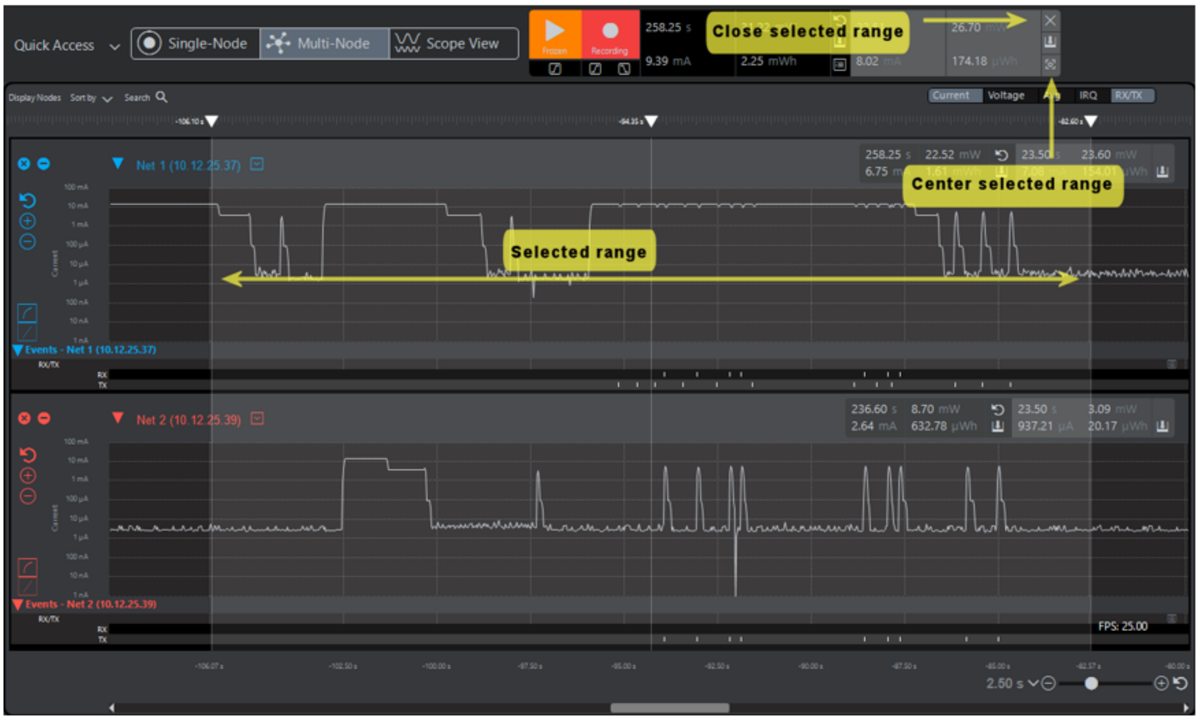

Auto Center
Reset Scales
J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440050039)
J-Link Silicon Labs (440051145)
J-Link Silicon Labs (440053983)
J-Link Silicon Labs (441100108)
Show Zero
20.0 mA

Toggle wave scale/offset control display using the show/hide button above it.



Selecting Ranges

Energy Profiler provides the ability to select a range of data by clicking and dragging across the waveforms or events. This is useful for determining power statistics for regions of interest. Once a range is selected, a light gray section summarizing the energy statistics for the selected region is displayed. That section also allows you to re-center the selection in the waveform view if it has scrolled out of view and includes a button to close/deselect the region.



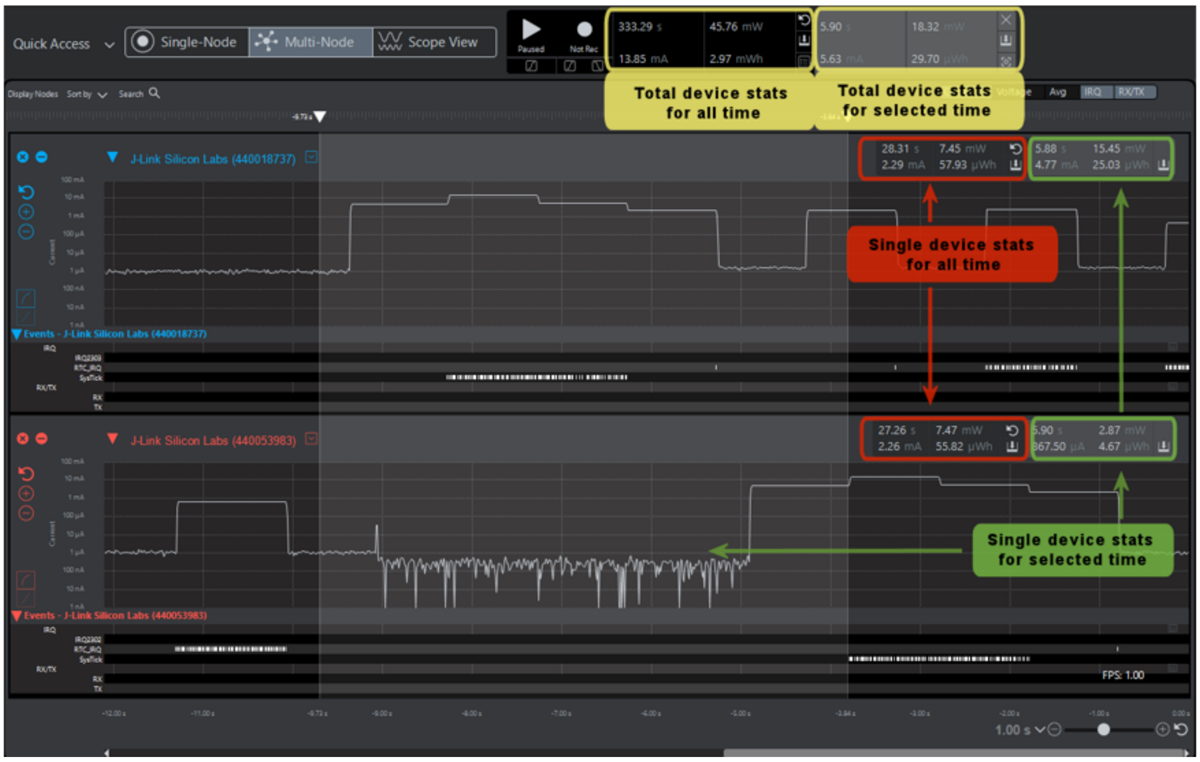
Energy Statistics

Energy Statistics

Energy Profiler calculates the following device and system statistics:

- Average current
- Average power
- Total energy

Statistics are provided for each individual device in the waveform's top bar on both Single-Node and Multi-Node views. The sum for all devices is provided in the top bar of Multi-Node Energy Profiler for all three views. Single-Node and Multi-Node views provide individual device statistics, while Scope view does not.

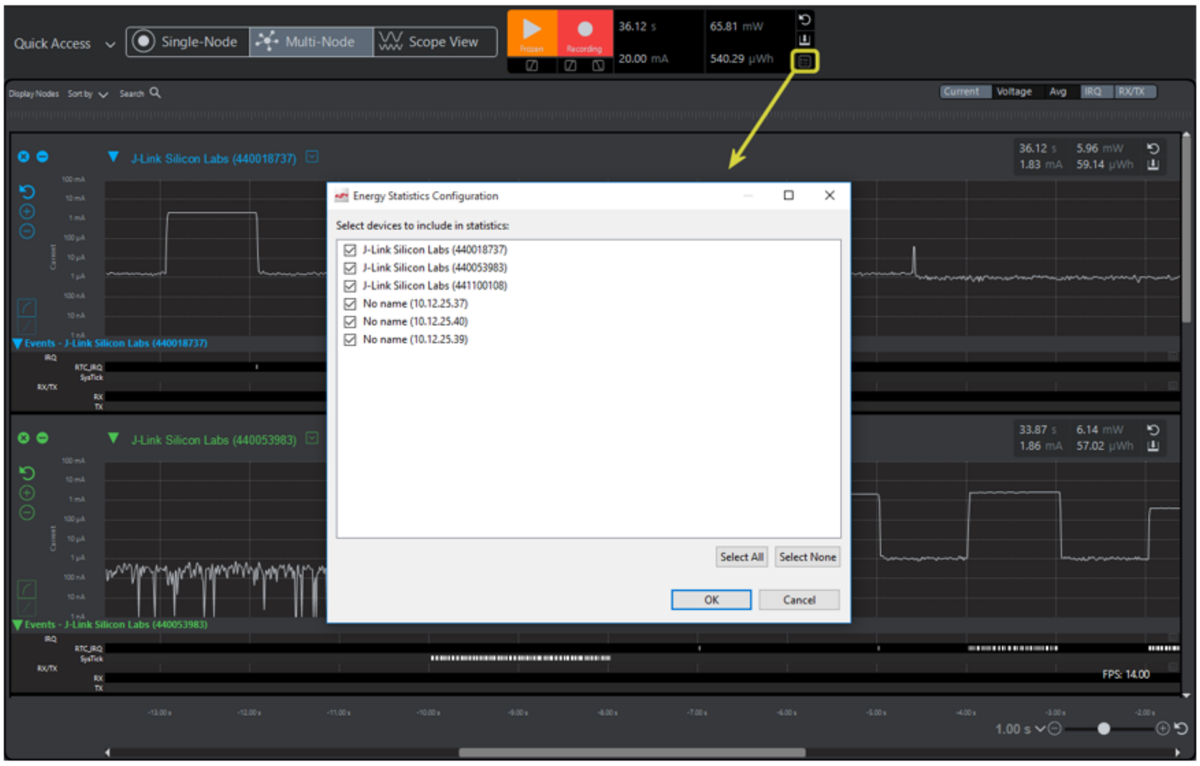

Quick Access
Single-Node
Multi-Node
Scope View
Total device stats for all time
Total device stats for selected time
Single device stats for all time
Single device stats for selected time
J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440053983)

This page reviews how to:

- Configure energy statistics
- Save a snapshot for future reference
- Review saved statistics

Energy Statistics Configuration

Energy Profiler allows you to select which of the actively capturing devices are included in the total system energy statistics calculation displayed in the top bar. Click the **Energy Statistics Configuration** icon and select the devices to be included in the calculation. It is important to note that the devices available for selection are those that are actively capturing data, which may be different from the devices that are selected for display in the UI.


Quick Access
Single-Node
Multi-Node
Scope View
J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440053983)
Events - J-Link Silicon Labs (440018737)
Events - J-Link Silicon Labs (440053983)
FPS: 14.00

Energy Statistics Configuration

Select devices to include in statistics:

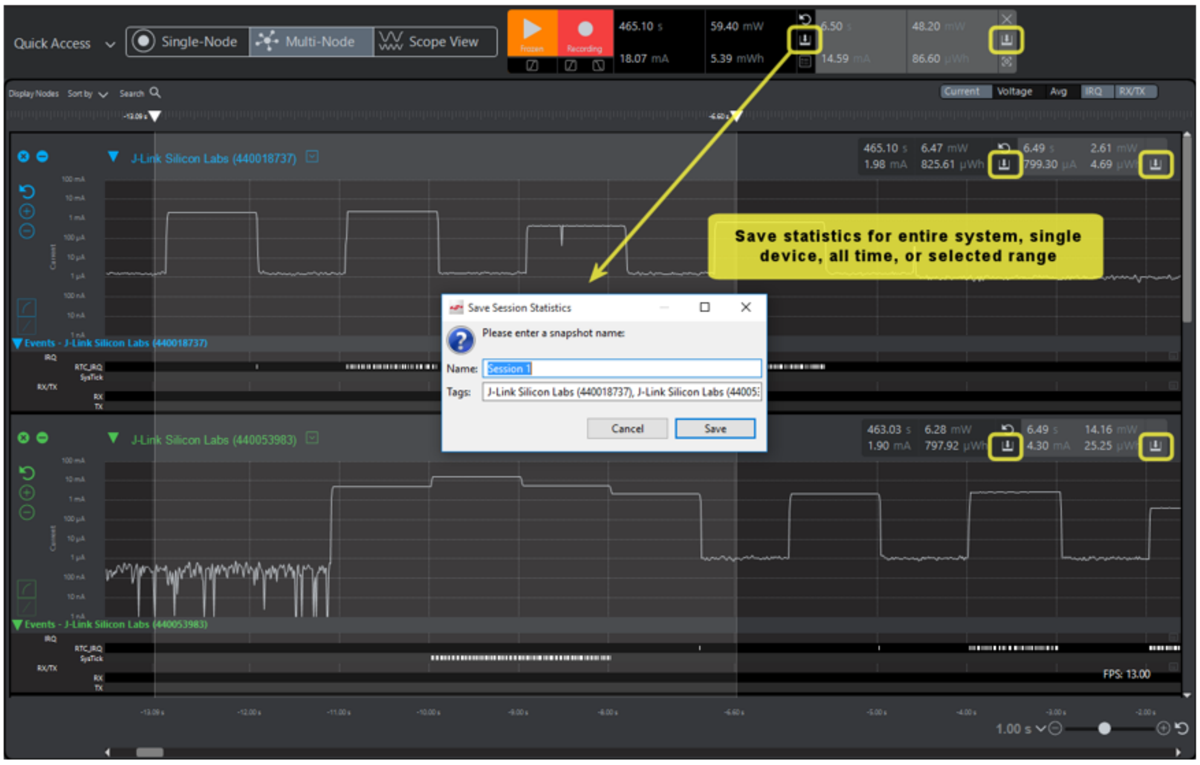
- ☑ J-Link Silicon Labs (440018737)
- ☑ J-Link Silicon Labs (440053983)
- ☑ J-Link Silicon Labs (441100108)
- ☑ No name (10.12.25.37)
- ☑ No name (10.12.25.40)
- ☑ No name (10.12.25.39)

Select All | Select None

OK | Cancel

Energy Statistics Snapshot

Energy Profiler allows you to save statistics for future reference and comparison. Click the **Save Snapshot** icon on any of the energy statistics panels and be presented with a dialog to provide a name and tag for future reference.


Quick Access
Single-Node
Multi-Node
Scope View
J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440053983)
Save statistics for entire system, single device, all time, or selected range
FPS: 13.00

Save Session Statistics

Please enter a snapshot name:

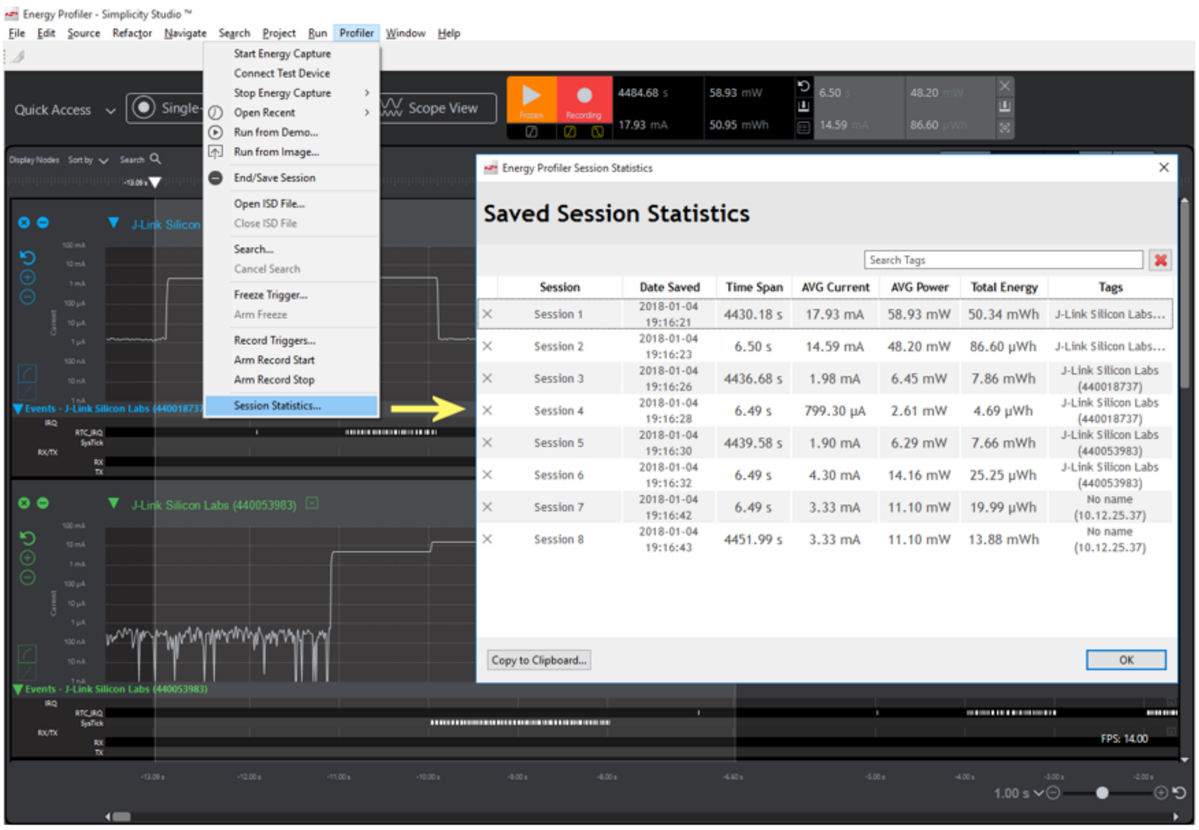
Name: Session 1

Tags: J-Link Silicon Labs (440018737), J-Link Silicon Labs (44005

Cancel | Save

Review Saved Statistics

Once you have saved multiple snapshots, you may want to review them, or to copy them to the clipboard for use in other applications. This is possible through the **Profiler->Session Statistics...** menu.



Saved Session Statistics

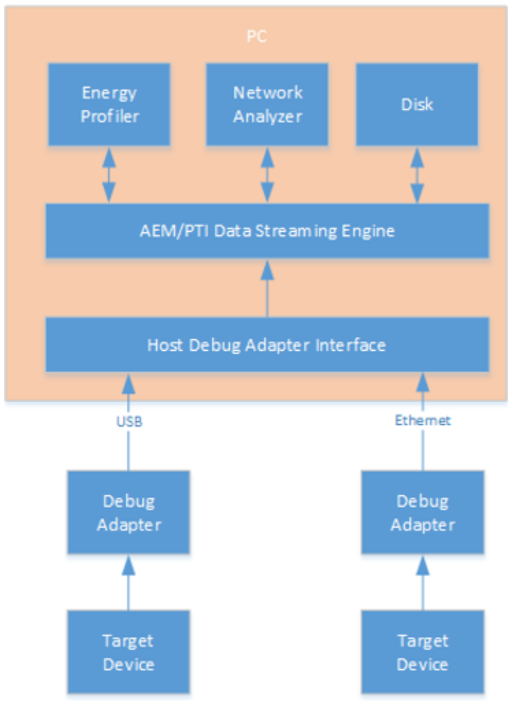
Search Tags

	Session	Date Saved	Time Span	AVG Current	AVG Power	Total Energy	Tags
×	Session 1	2018-01-04 19:16:21	4430.18 s	17.93 mA	58.93 mW	50.34 mWh	J-Link Silicon Labs...
×	Session 2	2018-01-04 19:16:23	6.50 s	14.59 mA	48.20 mW	86.60 µWh	J-Link Silicon Labs...
×	Session 3	2018-01-04 19:16:26	4436.68 s	1.98 mA	6.45 mW	7.86 mWh	J-Link Silicon Labs (440018737)
×	Session 4	2018-01-04 19:16:28	6.49 s	799.30 µA	2.61 mW	4.69 µWh	J-Link Silicon Labs (440018737)
×	Session 5	2018-01-04 19:16:30	4439.58 s	1.90 mA	6.29 mW	7.66 mWh	J-Link Silicon Labs (440053983)
×	Session 6	2018-01-04 19:16:32	6.49 s	4.30 mA	14.16 mW	25.25 µWh	J-Link Silicon Labs (440053983)
×	Session 7	2018-01-04 19:16:42	6.49 s	3.33 mA	11.10 mW	19.99 µWh	No name (10.12.25.37)
×	Session 8	2018-01-04 19:16:43	4451.99 s	3.33 mA	11.10 mW	13.88 mWh	No name (10.12.25.37)

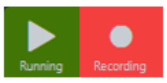
Play and Record Data Control

Play and Record Data Control

The following diagram provides a general description of data flow from target devices into SSv5 profiling applications. Data is sourced from target devices from the debug adapter either through USB or Ethernet connectivity to the host PC. The host debug adapter software layer provides data to the AEM/PTI data streaming engine, which in turn provides data to analysis applications such as Multi-Node Energy Profiler or Network Analyzer. The AEM/PTI data streaming engine saves data to disk in temporary files during a capture session. When the user ends the session, Energy Profiler prompts the user with the option to save the data to file.



Play and record controls in the top bar of the Energy Profiler perspective provide control over data being captured from connected devices.

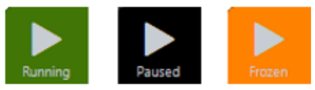


Play Control

The Play control determines whether data being captured is displayed in the Multi-Node Energy Profiler UI. The Play control has three states:

- Running (green)
- Paused (black)

- Frozen (orange)

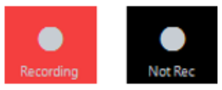


- **Running**: The "Running" state is entered when a capture is first started. When the Play control is green and displays "Running", it means energy capture data is actively arriving and being displayed on the UI. The visible portion of the waveform and events is the most recently captured data as it is arriving to Multi-Node Energy Profiler. The time position scroll bar is all the way to the right.
- **Paused**: The "Paused" state is entered by clicking **Play** while it is in the green "Running" state. The Play control turns black and displays "Paused", indicating energy capture data is actively arriving but it is not being displayed on the UI. The visible portion of the waveform and events may be anywhere on the timeline of previously captured data. Pausing the display has no effect on whether data is saved to disk. Save to disk is determined by the state of the Record control.
- **Frozen**: The "Frozen" state means energy capture data is actively arriving for display, but the visible portion of the waveform and events in the UI is previously captured data. The time position scroll bar may be anywhere on the timeline of previously captured data. The "Frozen" state is entered when the UI is in the play state and the time position scroll bar is scrolled or moved to the left. The "Frozen" state may also be entered as the result of a "Freeze trigger" condition. The "Frozen" Play control state has no effect on whether data is saved to disk. Save to disk is determined by the state of the Record control.

Record Control

The Record control manages data streaming to disk. There are two states to the Record control:

- Recording (red)
- Not recording (black)



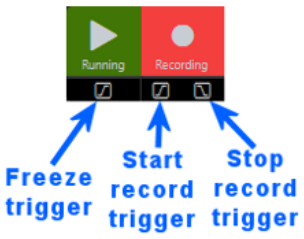
- **Recording**: When the Record control is red and displays "Recording", it means energy capture data is actively arriving and is being saved to temporary files to disk. When an energy capture is started, the "Recording" state is entered automatically. The state of the Record control has no effect on the Play control or the data being displayed in the UI. Clicking the Record control while in the "Recording" state transitions to the "Not Rec" state (that is, not recording). The "Recording" state may also be entered by use of Record triggers.
- **Not Recording**: When the Record control is black and displays "Not Rec", it means energy capture data is actively arriving but it is not being saved to disk. The "Not Rec" state is entered by clicking the Record control when it is in the "Recording" state. The state of the Record control has no effect on the Play control or the data being displayed in the UI. Clicking the Record control while in the "Not Rec" state will transition to the "Recording" state. The "Not Rec" state may also be entered by use of record triggers.

Freeze and Record Triggers

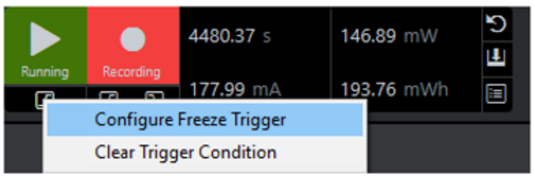
Freeze and Record Triggers

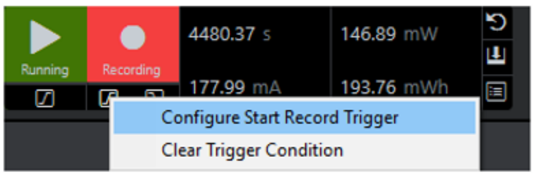
Triggers provide a method to automate freezing the display or recording to disk. Trigger icons are found below the Play and Record controls. Three triggers are available:

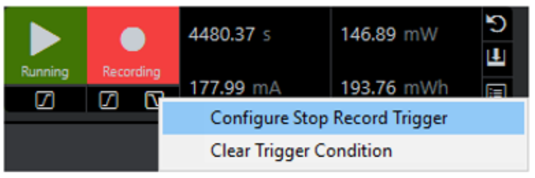
- Freeze trigger
- Record start trigger
- Record stop trigger



Triggers are driven by criteria set by the user. To set the criteria, use the context menu available from each trigger icon. The context menu also provides the ability to clear all conditions, which also disables the trigger.







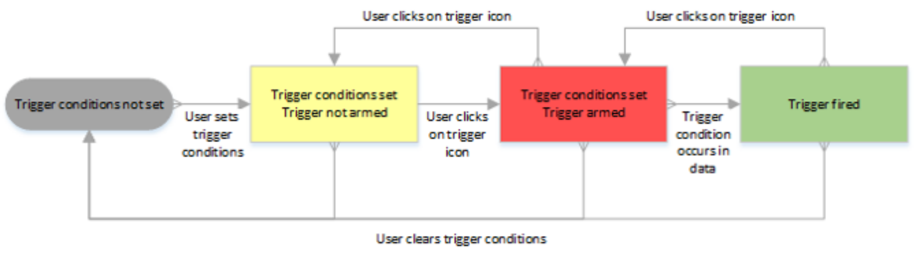
Trigger States

The four trigger states are each designated by a color:

- Grey - Trigger conditions are not set
- Yellow - Trigger conditions are set, trigger is not armed
- Red - Trigger conditions are set, trigger is armed

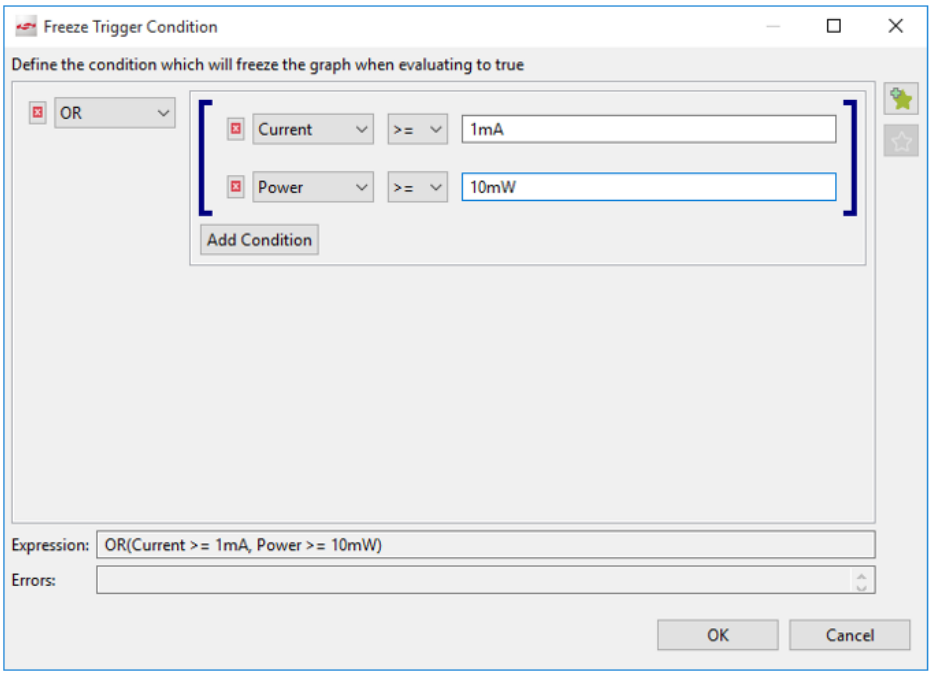
- Green - Trigger has fired

When Energy Profiler first starts up in a new workspace, all trigger icons are grey, indicating trigger conditions have not yet been set. Clicking a grey trigger icon opens the trigger configuration dialog, allowing you to specify trigger conditions. Once trigger conditions have been set, the trigger icon turns yellow, indicating the trigger conditions are set but the trigger is not armed. Clicking a yellow trigger icon arms the trigger as indicated by the trigger icon turning red. Incoming data is now evaluated against the trigger condition. Once the trigger condition is satisfied, the trigger icon turns green, indicating the armed trigger has fired. The relevant action (freeze, record start, or record stop) will have been executed. The following diagram shows the trigger state behavior.


User clicks on trigger icon
User clicks on trigger icon
Trigger conditions not set
User sets trigger conditions
Trigger conditions set Trigger not armed
User clicks on trigger icon
Trigger conditions set Trigger armed
Trigger condition occurs in data
Trigger fired
User clears trigger conditions

Freeze Trigger

The freeze trigger transitions the user interface from play to freeze mode automatically based upon search criteria. The freeze criteria are set through the Freeze Trigger Conditions dialog. This is invoked either through the **Profiler > Freeze Trigger...** menu selection or by right-clicking the **Freeze Trigger** icon and selecting **Configure Freeze Trigger".


Freeze Trigger Condition

Define the condition which will freeze the graph when evaluating to true

OR

Current >= 1mA

Power >= 10mW

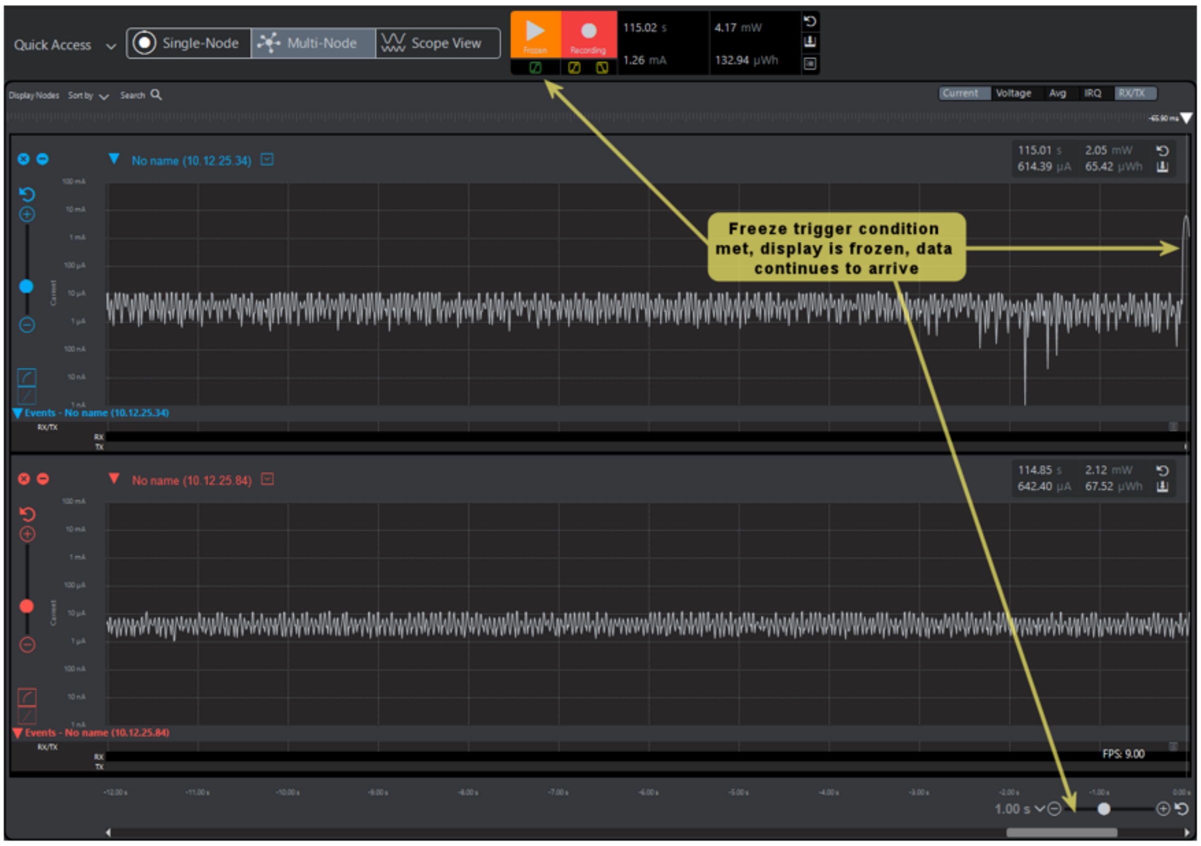
Add Condition

Expression: OR(Current >= 1mA, Power >= 10mW)

Errors:

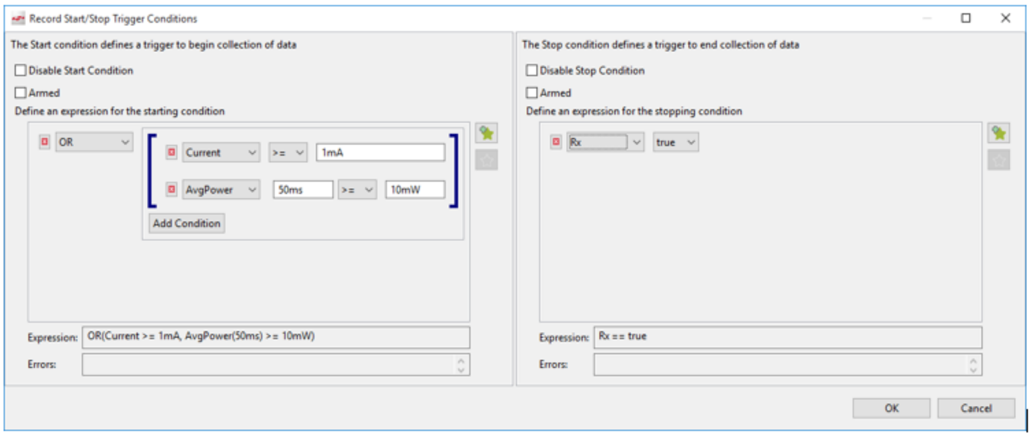
OK Cancel

In the following diagram, the play freeze trigger condition has occurred in the incoming data. Freeze mode is automatically entered for the waveform display and data continues to arrive as indicated by the scroll bar automatically moving to the left as new data arrives. Notice that the freeze trigger icon is green, indicating the trigger condition has occurred in arriving data.



Record Start/Stop Triggers

The record start/stop triggers automatically start and stop data record to disk based upon search criteria. This may be beneficial for capturing rare conditions rather than recording large amounts of data followed by search. The record start criteria are set through the Record Start/Stop Trigger Conditions dialog, which is invoked through the **Profiler > Record Triggers...** selection" menu or by right-clicking the **Record Start** or **Record Stop** icon and selecting the **Configure** option. The record start/stop trigger conditions are set together in a single dialog as shown in the following figure.



Record Start/Stop Trigger Conditions

The Start condition defines a trigger to begin collection of data

☐ Disable Start Condition

☐ Armed

Define an expression for the starting condition

OR [Current >= 1mA ; AvgPower 50ms >= 10mW] Add Condition

Expression: OR(Current >= 1mA, AvgPower(50ms) >= 10mW)

Errors:

The Stop condition defines a trigger to end collection of data

☐ Disable Stop Condition

☐ Armed

Define an expression for the stopping condition

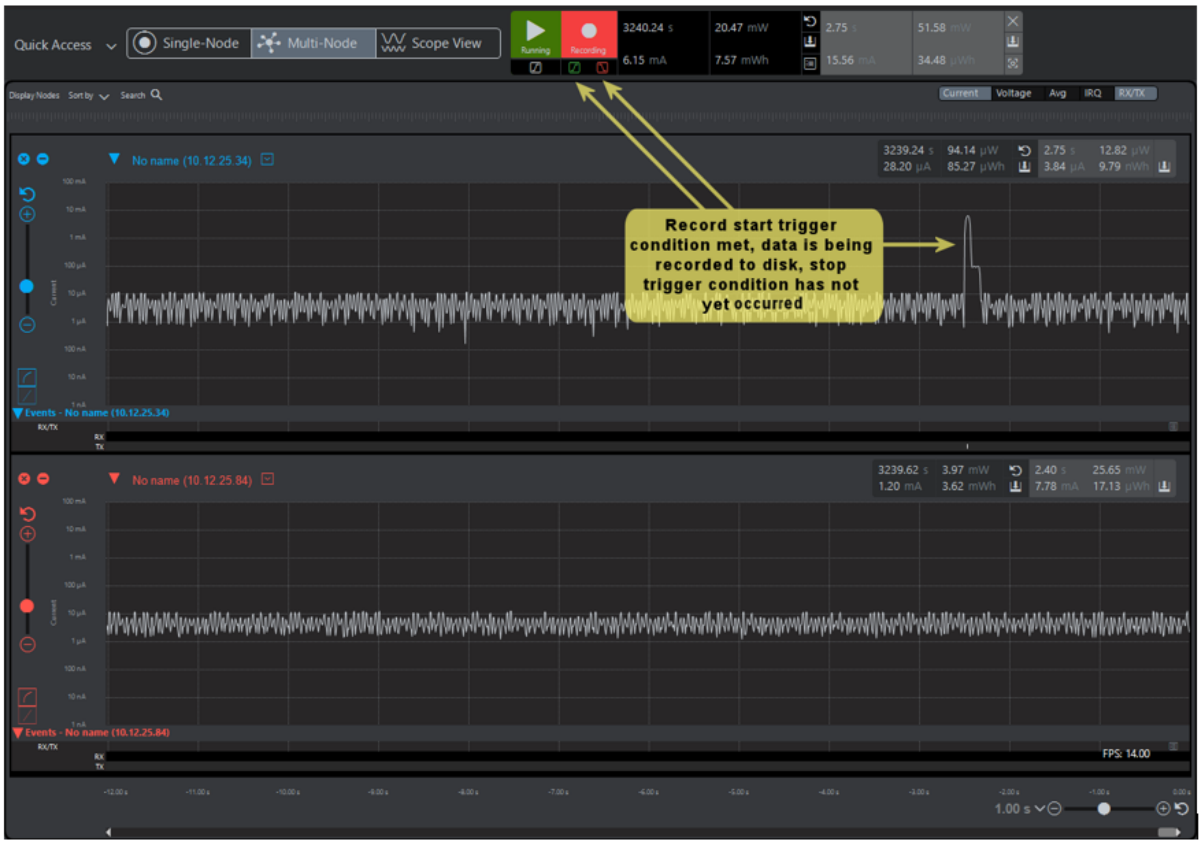
Rx true

Expression: Rx == true

Errors:

OK Cancel

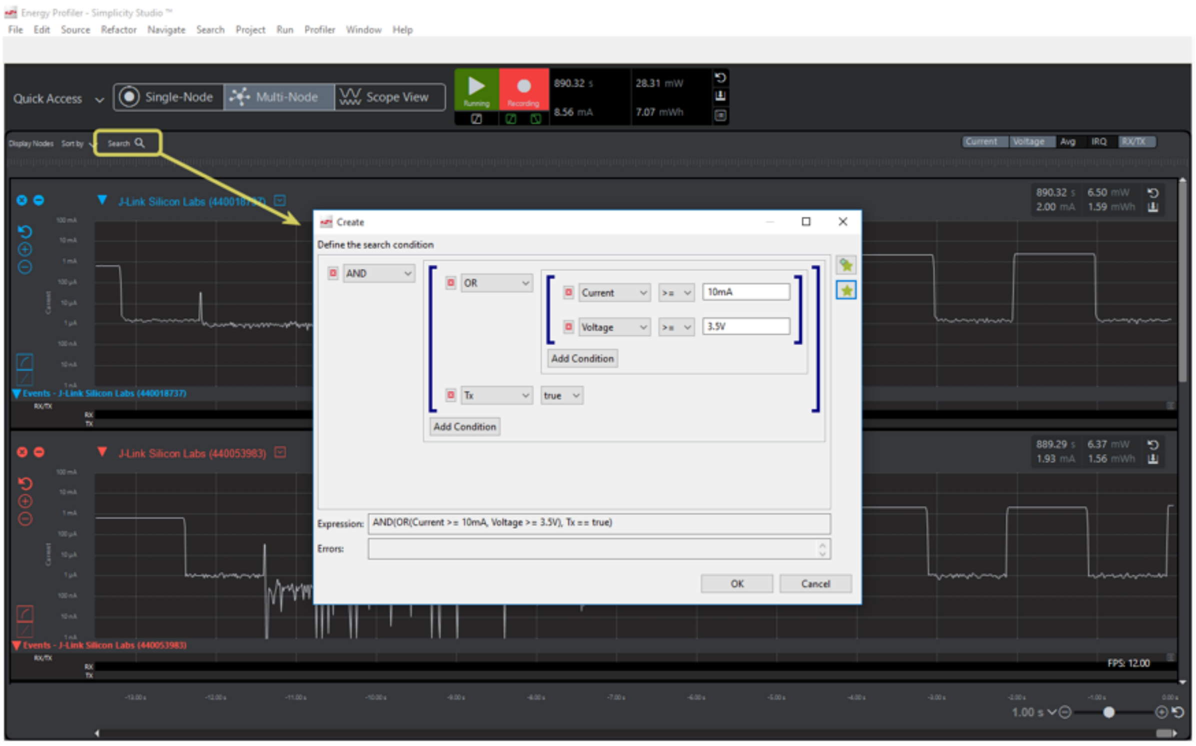
In the following figure, the record start trigger condition has occurred in the incoming data as indicated by the green record start trigger icon. The record stop trigger condition has not yet occurred and data will be written to disk until it does.


Quick Access
Single-Node
Multi-Node
Scope View
Running
Recording
3240.24 s
6.15 mA
20.47 mW
7.57 mWh
2.75 s
15.56 mA
51.58 mW
34.48 µWh
Display Nodes
Sort by
Search
Current
Voltage
Avg
IRQ
RX/TX
No name (10.12.25.34)
3239.24 s
28.20 µA
94.14 µW
85.27 µWh
2.75 s
3.84 µA
12.82 µW
9.79 nWh
Record start trigger condition met, data is being recorded to disk, stop trigger condition has not yet occurred
Events - No name (10.12.25.34)
No name (10.12.25.84)
3239.62 s
1.20 mA
3.97 mW
3.62 mWh
2.40 s
7.78 mA
25.65 mW
17.13 µWh
Events - No name (10.12.25.84)
FPS: 14.00
1.00 s

Search Capability

Search Capability

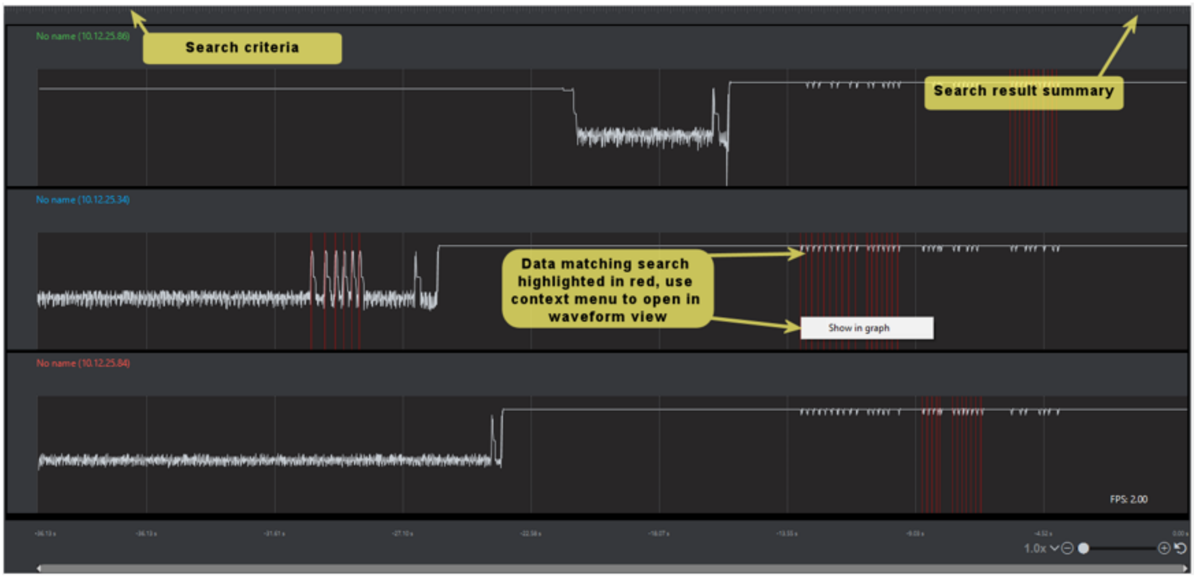
One of Energy Profiler's most powerful aspects is the ability to search data. Search is available both during a live capture and with an offline ISD file. Click **Search** to open the Search Conditions dialog.


Energy Profiler - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Profiler Window Help
Quick Access
Single-Node
Multi-Node
Scope View
Running
Recording
Display Nodes
Sort by
Search
Current
Voltage
Avg
IRQ
RX/TX
J-Link Silicon Labs (440053983)
Events - J-Link Silicon Labs (440018737)
Events - J-Link Silicon Labs (440053983)
Create
Define the search condition
AND
OR
Current
>=
10mA
Voltage
>=
3.5V
Add Condition
Tx
true
Add Condition
OK
Cancel
FPS: 12.00

Expression: AND(OR(Current >= 10mA, Voltage >= 3.5V), Tx == true)

Errors:

When you click **OK**, a search is performed on all data currently available, whether from the currently active capture or in offline mode from an open file. The search results window is displayed on completion. Sections where the data matched the criteria are shown in red. A summary of the count and % coverage of the matching regions is shown at the upper right. Use the context menu to jump to the matching in the waveform view from the search results window.


No name (10.12.25.86)
Search criteria
Search result summary
No name (10.12.25.34)
Data matching search highlighted in red, use context menu to open in waveform view
Show in graph
No name (10.12.25.84)
FPS: 2.00
1.0x

Profiling with Code Correlation

Profiling with Code Correlation

Code correlation is one of the most powerful features in Energy Profiler. Energy Profiler captures program execution in conjunction with the energy data. This allows Energy Profiler to calculate the power consumed by each function executed in the application. This data may then be sorted to highlight the portions of an application that consume the most power. This enables the application developer to know where they should concentrate their software development efforts to reduce power consumption. Additionally, Energy Profiler can color-code sections of the current waveform to represent which functions were executing when a given section of the current waveform occurred.

This page reviews:

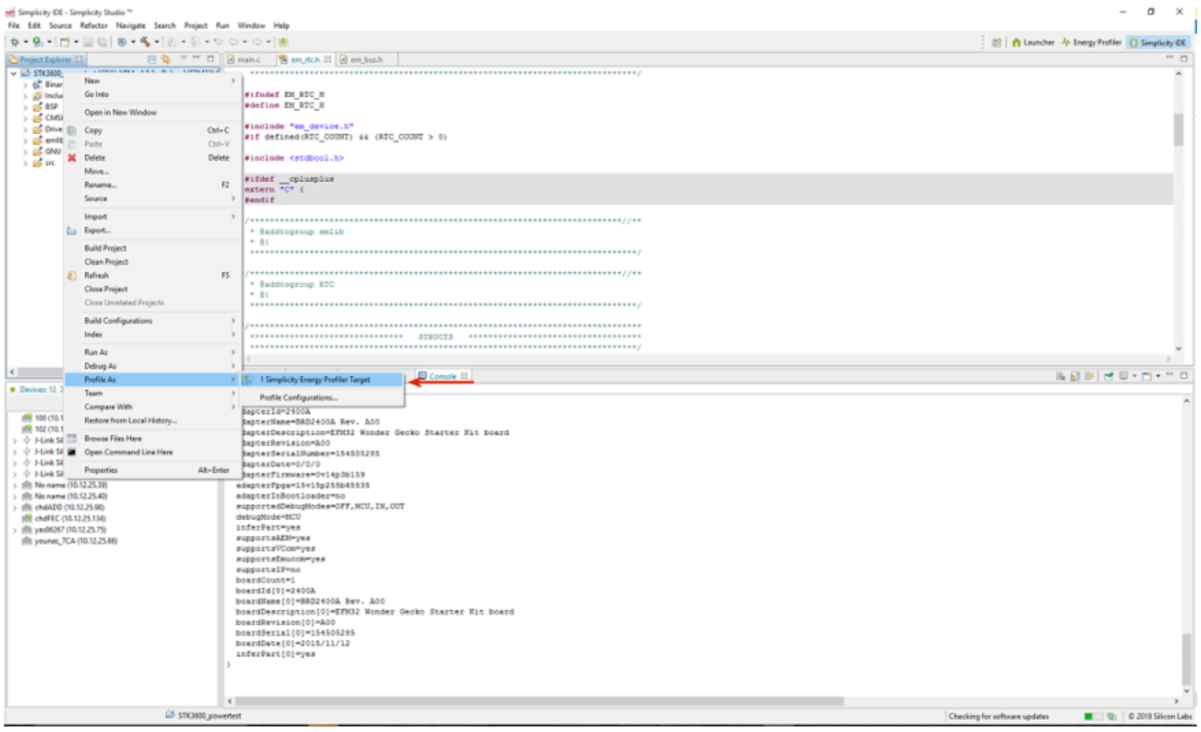
- Enabling code correlation
- The Code Correlation view
- Color coding

Enabling Code Correlation

This section assumes an embedded application is already configured to produce AEM data through a debug adapter (see Customer Hardware and Software Design Information for details). For code correlation to operate, Energy Profiler needs access to debug information associated with the embedded application that is currently running. This information is available in the AXF file output from the build. Code correlation can be enabled when an application is started from the Simplicity IDE perspective through context menu commands for a given project, or it can be associated with an application already running on an embedded device that is being profiled in Energy Profiler.

From the Simplicity IDE Perspective

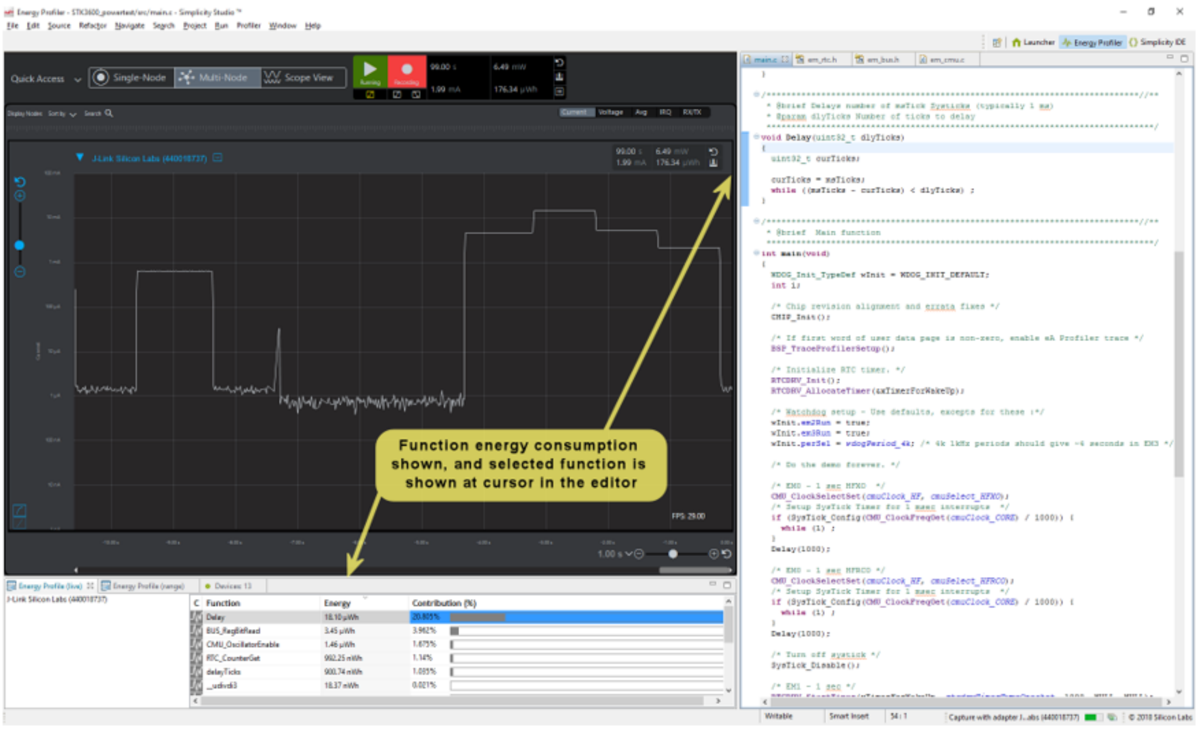
One simple way to enable the code correlation view is to start the energy profiling session by using the context menu on the project from the Simplicity IDE perspective, as shown below. The associated debug file will be provided to Energy Profiler when the session is started.


Profile As
1 Simplicity Energy Profiler Target
Profile Configurations...
Console

```
#ifndef EM_RTC_H
#define EM_RTC_H

#include "em_device.h"
#if defined(RTC_COUNT) && (RTC_COUNT > 0)

#include <stdbool.h>

#ifdef __cplusplus
extern "C" {
#endif
```

```
adapterId=2400A
adapterName=BRD2400A Rev. A00
adapterDescription=EFM32 Wonder Gecko Starter Kit board
adapterRevision=A00
adapterSerialNumber=154505285
adapterDate=0/0/0
adapterFirmware=0v14p3b159
adapterFpga=15v15p255b65535
adapterInBootloader=no
supportedDebugModes=OFF,MCU,IN,OUT
debugMode=MCU
inferPart=yes
supportsAEM=yes
supportsVCom=yes
supportsEmucom=yes
supportsIP=no
boardCount=1
boardId[0]=2400A
boardName[0]=BRD2400A Rev. A00
boardDescription[0]=EFM32 Wonder Gecko Starter Kit board
boardRevision[0]=A00
boardSerial[0]=154505285
boardDate[0]=2015/11/12
inferPart[0]=yes
}
```

Once Energy Profiler starts, the following view is displayed. Notice in the Code Correlation view, energy consumed by each function is updated in real time. Clicking any of the functions opens the associated source code file and jumps to that function in the file.

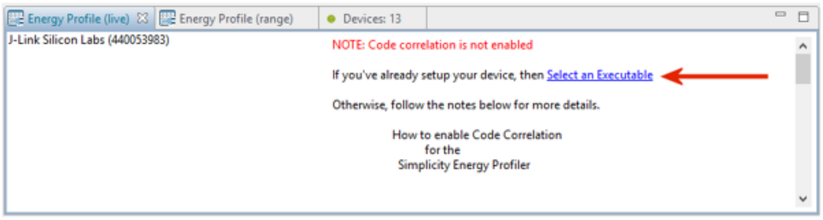

Function energy consumption shown, and selected function is shown at cursor in the editor

C	Function	Energy	Contribution (%)
	Delay	18.10 µWh	20.805%
	BUS_RegBitRead	3.45 µWh	3.962%
	CMU_OscillatorEnable	1.46 µWh	1.675%
	RTC_CounterGet	992.25 nWh	1.14%
	delayTicks	900.74 nWh	1.035%
	__udivdi3	18.37 nWh	0.021%

```
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}
```

From Energy Profiler

If you start energy capture with a running embedded application, Energy Profiler does not have the debug information it needs to associate program execution with source code. In these cases, the Code Correlation view is as shown in the following figure.



Energy Profile (live) | Energy Profile (range) | Devices: 13

J-Link Silicon Labs (440053983)

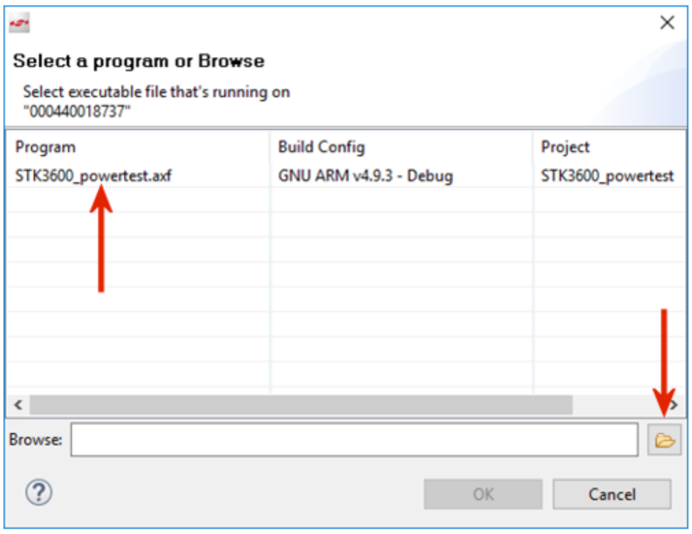
NOTE: Code correlation is not enabled

If you've already setup your device, then Select an Executable

Otherwise, follow the notes below for more details.

How to enable Code Correlation
for the
Simplicity Energy Profiler

Click the **elect an Executable** link and the program selection dialog is displayed. Debug files from open projects are provided in the list. Alternatively, you may browse to select the debug file.



Select a program or Browse

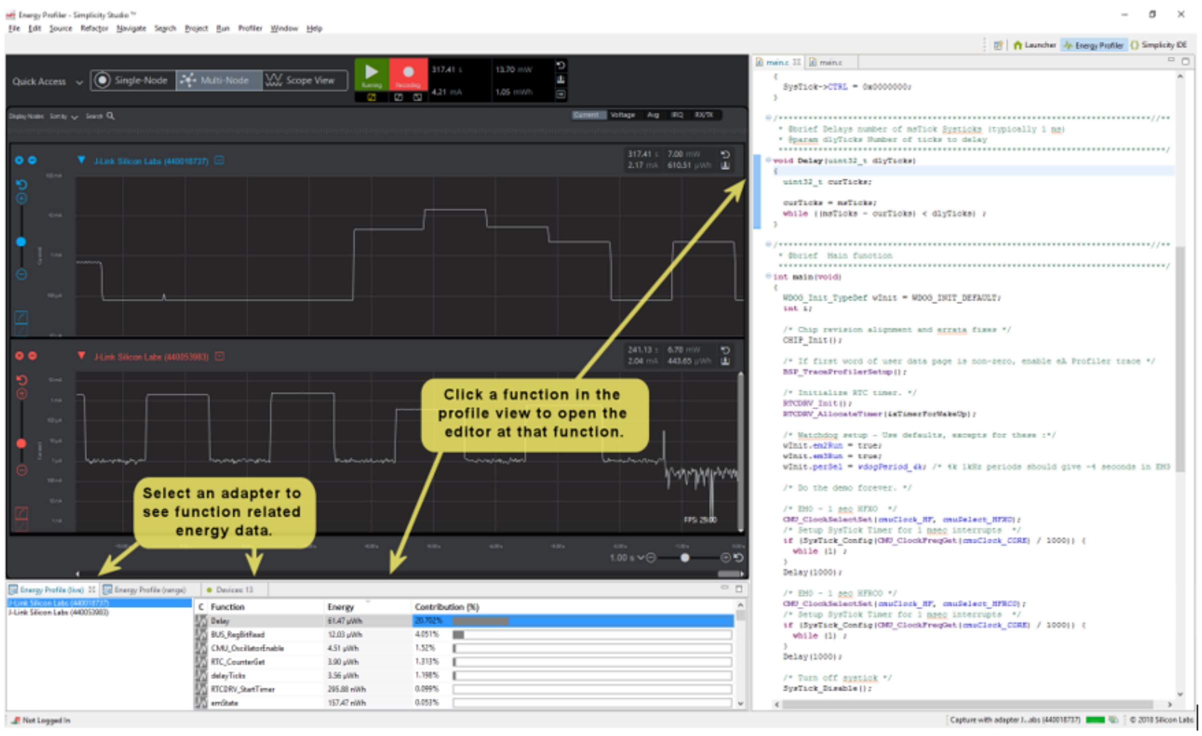
Select executable file that's running on "000440018737"

Program	Build Config	Project
STK3600_powertest.axf	GNU ARM v4.9.3 - Debug	STK3600_powertest

Browse:

Code Correlation View

The Code Correlation view is shown below the waveform portion of the Multi-Node Energy Profiler Perspective, and works in conjunction with the Editor view. When multiple devices are displayed in the Multi-Node view, you can click on each adapter to view the function profile data associated with that device. The function data may be sorted as desired. Clicking on any function opens the associated source file in the Editor view with the start of scope for that function selected.

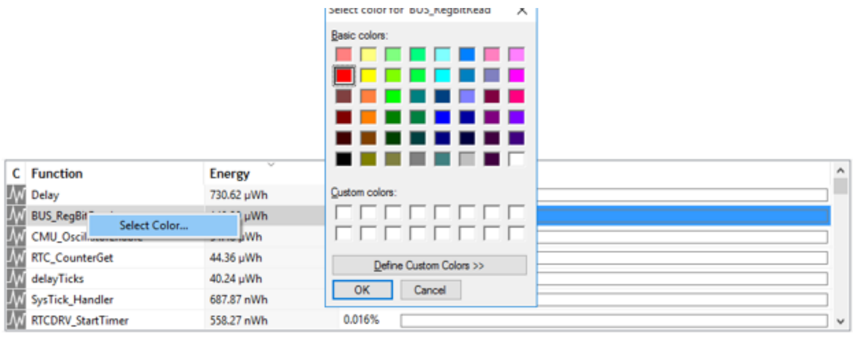

Energy Profiler - Simplicity Studio ™
File Edit Source Refactor Navigate Search Project Run Profiler Window Help
Quick Access
Single-Node
Multi-Node
Scope View
Launcher
Energy Profiler
Simplicity IDE
main.c
J-Link Silicon Labs (440018737)
J-Link Silicon Labs (440053983)
Select an adapter to see function related energy data.
Click a function in the profile view to open the editor at that function.
Energy Profile (live)
Energy Profile (range)
Devices: 13
Not Logged In
Capture with adapter J...abs (440018737)
© 2018 Silicon Labs

C	Function	Energy	Contribution (%)
	Delay	61.47 µWh	20.702%
	BUS_RegBitRead	12.03 µWh	4.051%
	CMU_OscillatorEnable	4.51 µWh	1.52%
	RTC_CounterGet	3.90 µWh	1.313%
	delayTicks	3.56 µWh	1.198%
	RTCDRV_StartTimer	295.88 nWh	0.099%
	emState	157.47 nWh	0.053%

```
{
  SysTick->CTRL = 0x0000000;
}

/***************************************************************************//**
 * @brief Delays number of msTick Systicks (typically 1 ms)
 * @param dlyTicks Number of ticks to delay
 ******************************************************************************/
void Delay(uint32_t dlyTicks)
{
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < dlyTicks) ;
}

/***************************************************************************//**
 * @brief  Main function
 ******************************************************************************/
int main(void)
{
  WDOG_Init_TypeDef wInit = WDOG_INIT_DEFAULT;
  int i;

  /* Chip revision alignment and errata fixes */
  CHIP_Init();

  /* If first word of user data page is non-zero, enable eA Profiler trace */
  BSP_TraceProfilerSetup();

  /* Initialize RTC timer. */
  RTCDRV_Init();
  RTCDRV_AllocateTimer(&xTimerForWakeUp);

  /* Watchdog setup - Use defaults, excepts for these :*/
  wInit.em2Run = true;
  wInit.em3Run = true;
  wInit.perSel = wdogPeriod_4k; /* 4k 1kHz periods should give ~4 seconds in EM3

  /* Do the demo forever. */

  /* EM0 - 1 sec HFXO  */
  CMU_ClockSelectSet(cmuClock_HF, cmuSelect_HFXO);
  /* Setup SysTick Timer for 1 msec interrupts  */
  if (SysTick_Config(CMU_ClockFreqGet(cmuClock_CORE) / 1000)) {
    while (1) ;
  }
  Delay(1000);

  /* EM0 - 1 sec HFRCO */
  CMU_ClockSelectSet(cmuClock_HF, cmuSelect_HFRCO);
  /* Setup SysTick Timer for 1 msec interrupts  */
  if (SysTick_Config(CMU_ClockFreqGet(cmuClock_CORE) / 1000)) {
    while (1) ;
  }
  Delay(1000);

  /* Turn off systick */
  SysTick_Disable();
```

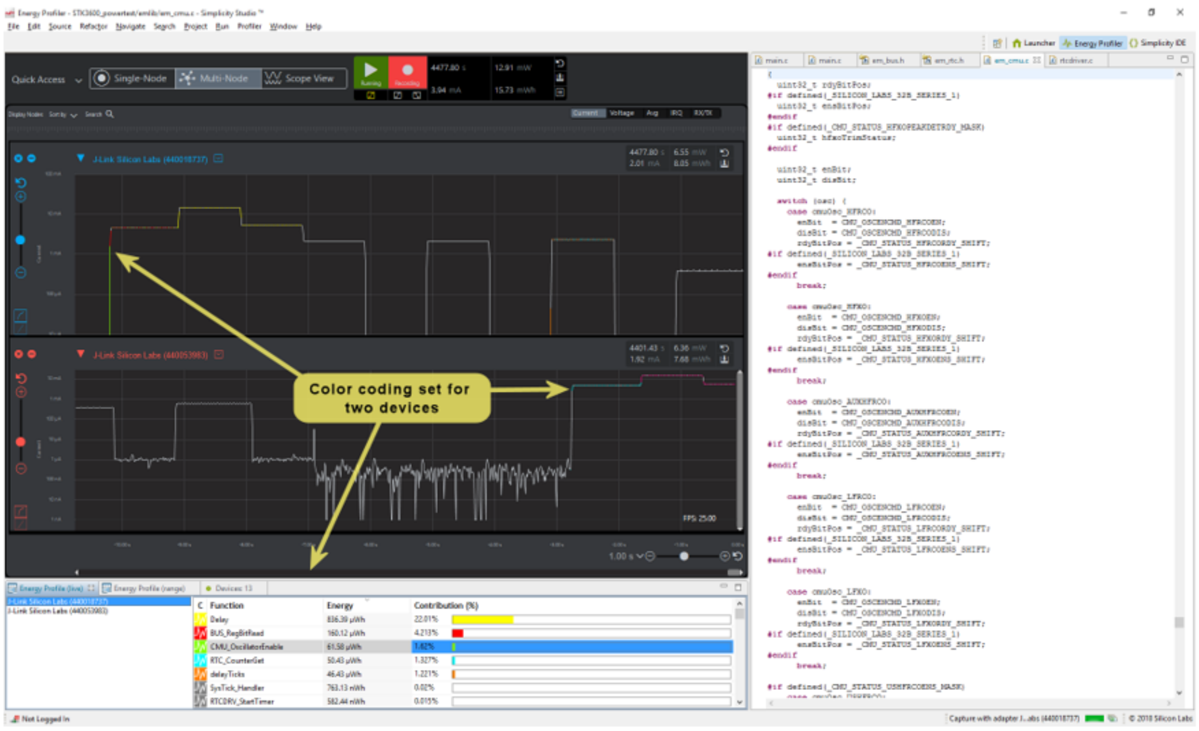
Color Coding and Code Correlation

Energy Profiler provides the ability to color-code the current waveform based on program execution. You can set the color for each function using the context menus in the Code Correlation view.



C	Function	Energy	
	Delay	730.62 µWh	
	BUS_RegBit	µWh	
	CMU_Oscil	Wh	
	RTC_CounterGet	44.36 µWh	
	delayTicks	40.24 µWh	
	SysTick_Handler	687.87 nWh	
	RTCDRV_StartTimer	558.27 nWh	0.016%

The following figure shows the current waveform view with several functions set to different colors.



```
{
  uint32_t rdyBitPos;
#if defined(_SILICON_LABS_32B_SERIES_1)
  uint32_t ensBitPos;
#endif
#if defined(_CMU_STATUS_HFXOPEAKDETRDY_MASK)
  uint32_t hfxoTrimStatus;
#endif

  uint32_t enBit;
  uint32_t disBit;

  switch (osc) {
    case cmuOsc_HFRCO:
      enBit  = CMU_OSCENCMD_HFRCOEN;
      disBit = CMU_OSCENCMD_HFRCODIS;
      rdyBitPos = _CMU_STATUS_HFRCORDY_SHIFT;
#if defined(_SILICON_LABS_32B_SERIES_1)
      ensBitPos = _CMU_STATUS_HFRCOENS_SHIFT;
#endif
      break;

    case cmuOsc_HFXO:
      enBit  = CMU_OSCENCMD_HFXOEN;
      disBit = CMU_OSCENCMD_HFXODIS;
      rdyBitPos = _CMU_STATUS_HFXORDY_SHIFT;
#if defined(_SILICON_LABS_32B_SERIES_1)
      ensBitPos = _CMU_STATUS_HFXOENS_SHIFT;
#endif
      break;

    case cmuOsc_AUXHFRCO:
      enBit  = CMU_OSCENCMD_AUXHFRCOEN;
      disBit = CMU_OSCENCMD_AUXHFRCODIS;
      rdyBitPos = _CMU_STATUS_AUXHFRCORDY_SHIFT;
#if defined(_SILICON_LABS_32B_SERIES_1)
      ensBitPos = _CMU_STATUS_AUXHFRCOENS_SHIFT;
#endif
      break;

    case cmuOsc_LFRCO:
      enBit  = CMU_OSCENCMD_LFRCOEN;
      disBit = CMU_OSCENCMD_LFRCODIS;
      rdyBitPos = _CMU_STATUS_LFRCORDY_SHIFT;
#if defined(_SILICON_LABS_32B_SERIES_1)
      ensBitPos = _CMU_STATUS_LFRCOENS_SHIFT;
#endif
      break;

    case cmuOsc_LFXO:
      enBit  = CMU_OSCENCMD_LFXOEN;
      disBit = CMU_OSCENCMD_LFXODIS;
      rdyBitPos = _CMU_STATUS_LFXORDY_SHIFT;
#if defined(_SILICON_LABS_32B_SERIES_1)
      ensBitPos = _CMU_STATUS_LFXOENS_SHIFT;
#endif
      break;

#if defined(_CMU_STATUS_USHFRCOENS_MASK)
    case cmuOsc_USHFRCO:
```

C	Function	Energy	Contribution (%)
🟨	Delay	836.39 µWh	22.01%
🟥	BUS_RegBitRead	160.12 µWh	4.213%
🟩	CMU_OscillatorEnable	61.58 µWh	1.62%
🟦	RTC_CounterGet	50.43 µWh	1.327%
🟧	delayTicks	46.43 µWh	1.221%
⬜	SysTick_Handler	763.13 nWh	0.02%
⬜	RTCDRV_StartTimer	582.44 nWh	0.015%

Debugging with the Logic Analyzer

Debugging with the Logic Analyzer

Note: Energy Profiler's integrated Logic Analyzer can only be used with Wireless Pro Kit Mainboards, such as BRD4002A. The features described in this section are enabled only when supported hardware is connected.

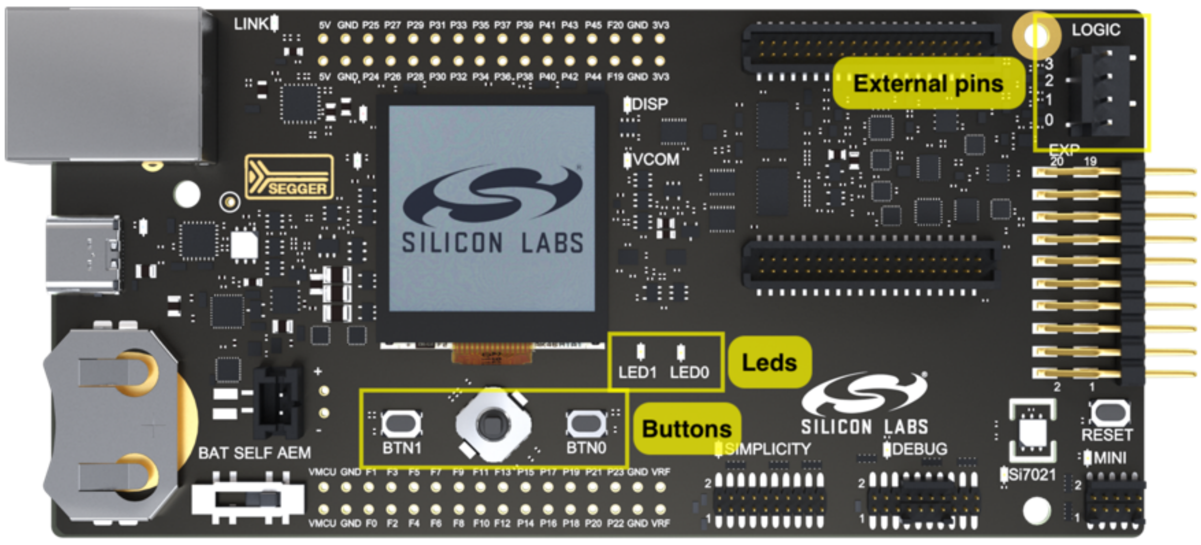
The Logic Analyzer displays the profiles from up to eight channels corresponding to the mainboard's four external pins, two LEDs, and two buttons:

External channels: 0-3

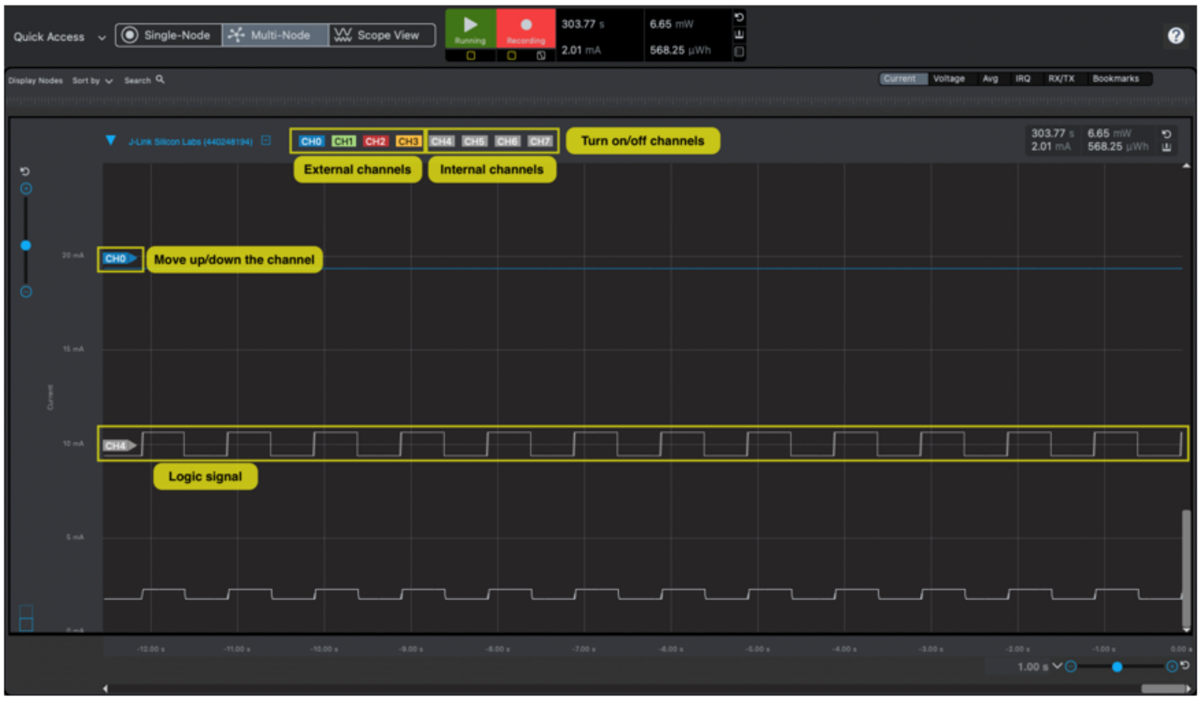
- CH0
- CH1
- CH2
- CH3

Internal channels: 4-7

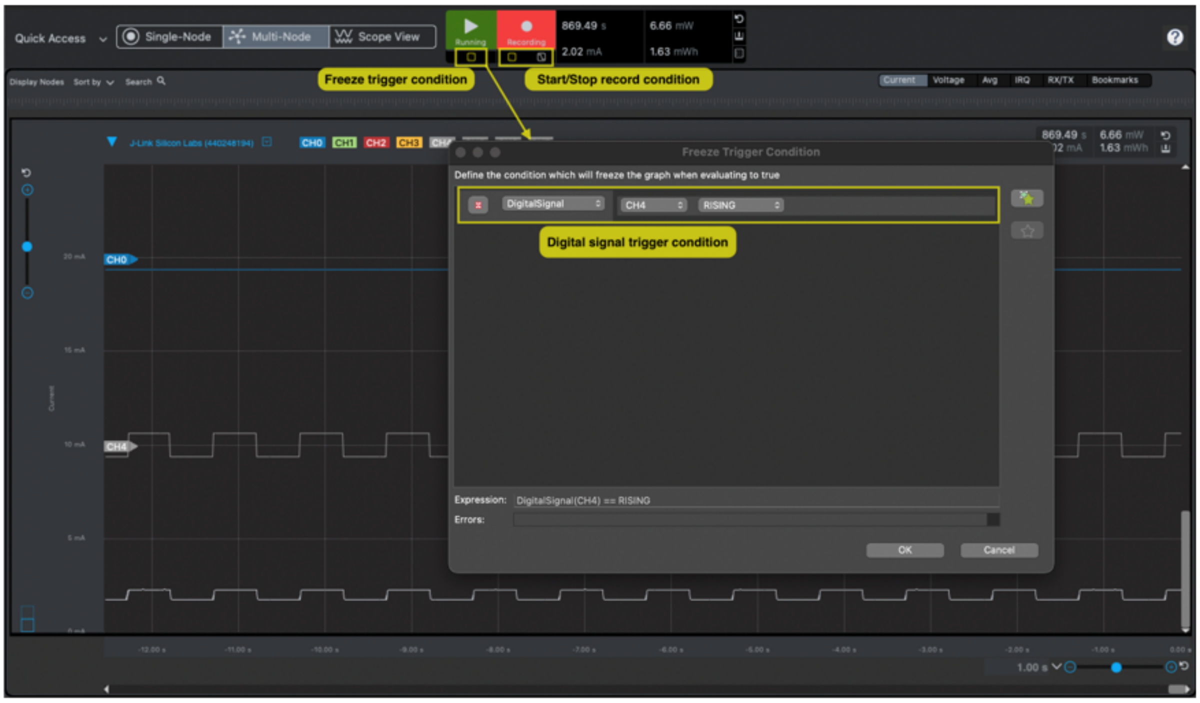
- CH4: LED0
- CH5: LED1
- CH6: BTN0
- CH7: BTN1



The Logic Analyzer's channel buttons act as toggles: click a button to display the logic signal, click again to turn it off. The channels that are being displayed can be dragged and positioned closer to or farther away from other channels or the energy profile.



Freeze triggers can be set for the logic signals as for other features, as described in Freeze and Record Triggers. Click the Freeze trigger to open the configuration dialog, then select **DigitalSignal**, the target channel, and the condition. Logic channel data recording will be enabled in a future release.


Freeze trigger condition
Start/Stop record condition
Freeze Trigger Condition
Define the condition which will freeze the graph when evaluating to true
DigitalSignal
CH4
RISING
Digital signal trigger condition
Expression: DigitalSignal(CH4) == RISING
Errors:
OK
Cancel

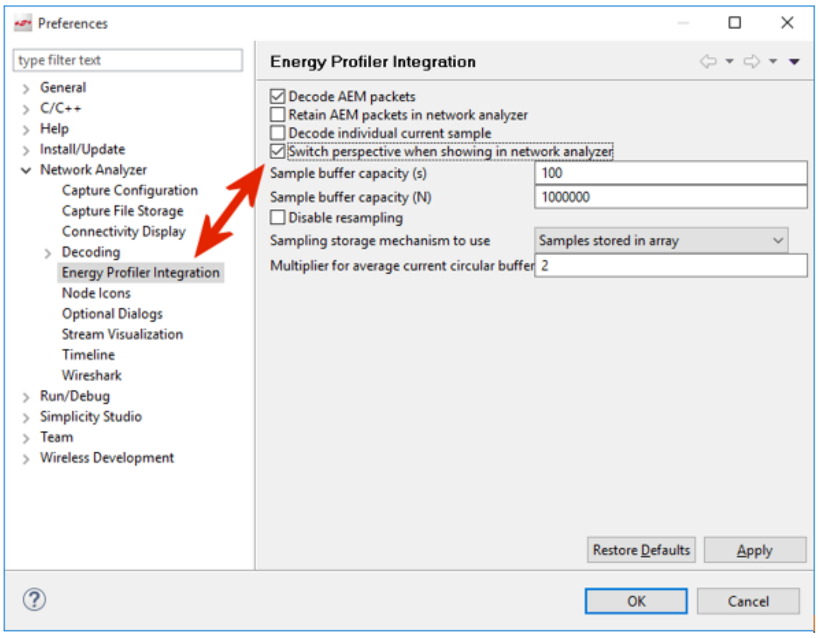
Energy Profiler and Network Analyzer

Energy Profiler and Network Analyzer Integration

Note: This functionality is only available in offline mode with a previously saved ISD file.

Energy Profiler and Network Analyzer now provide a basic level of integration when trace data from a debug adapter includes both AEM and PTI data. This integration allows you to easily move between the two applications by selecting an event in either one and selecting **Show in Network Analyzer** or **Show in Energy Profiler**.

An SSv5 preference determines whether moving between applications leaves the Energy Profiler perspective and opens the Network Analyzer perspective, or stays in the Energy Profiler perspective.



Energy Profiler Integration

- [✓] Decode AEM packets
- [ ] Retain AEM packets in network analyzer
- [ ] Decode individual current sample
- [✓] Switch perspective when showing in network analyzer

Sample buffer capacity (s): 100

Sample buffer capacity (N): 1000000

- [ ] Disable resampling

Sampling storage mechanism to use: Samples stored in array

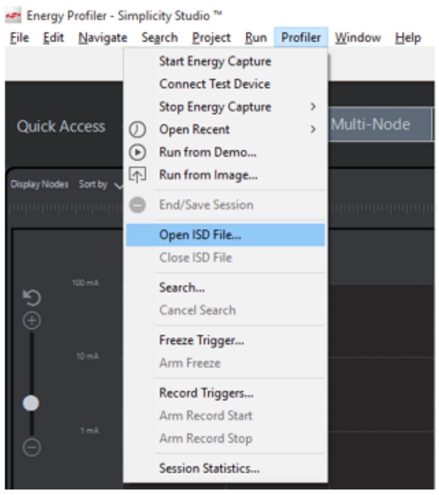
Multiplier for average current circular buffer: 2

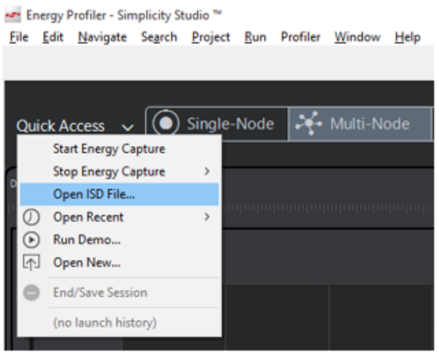
This page reviews:

- How to use the integrated tools
- Use Cases for Multi-Node Energy Profiler and Network Analyzer
- Post analysis using an ISD file

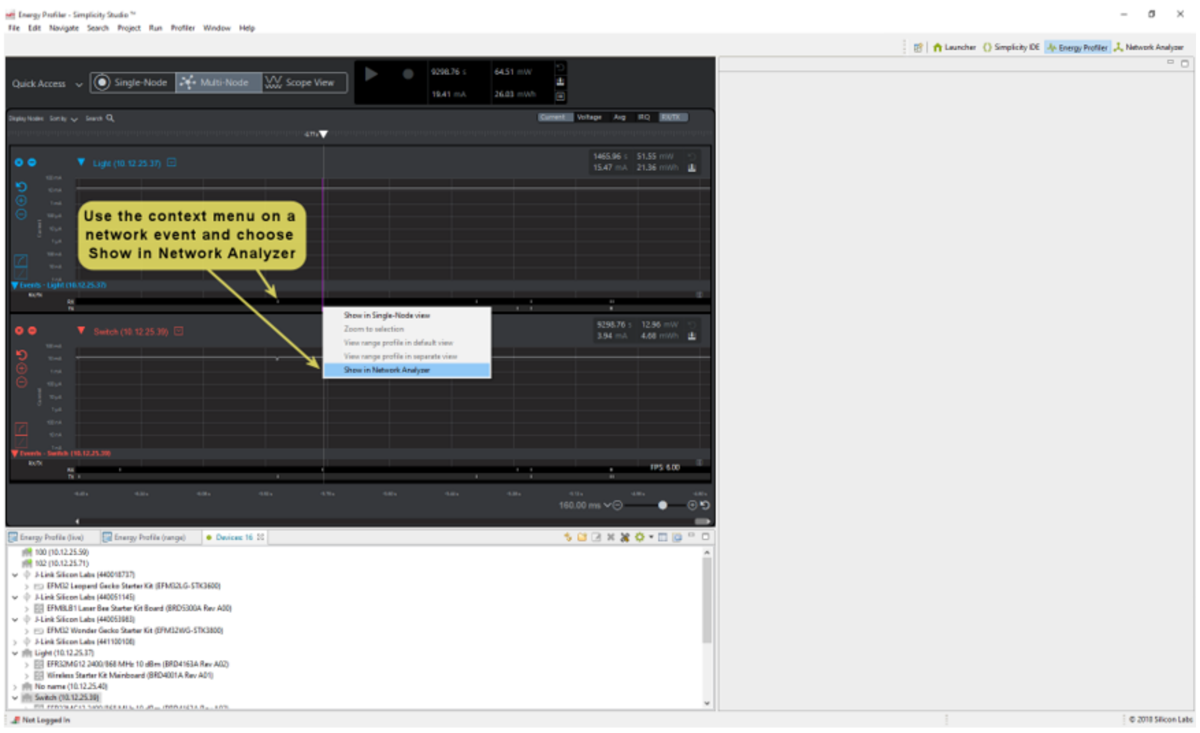
Using the Integrated Tools

To open a previously saved ISD file, either use the top menu **Profiler > Open ISD File...** menu or the **Quick Access > Open ISD File...** menu, as shown in the following figure.

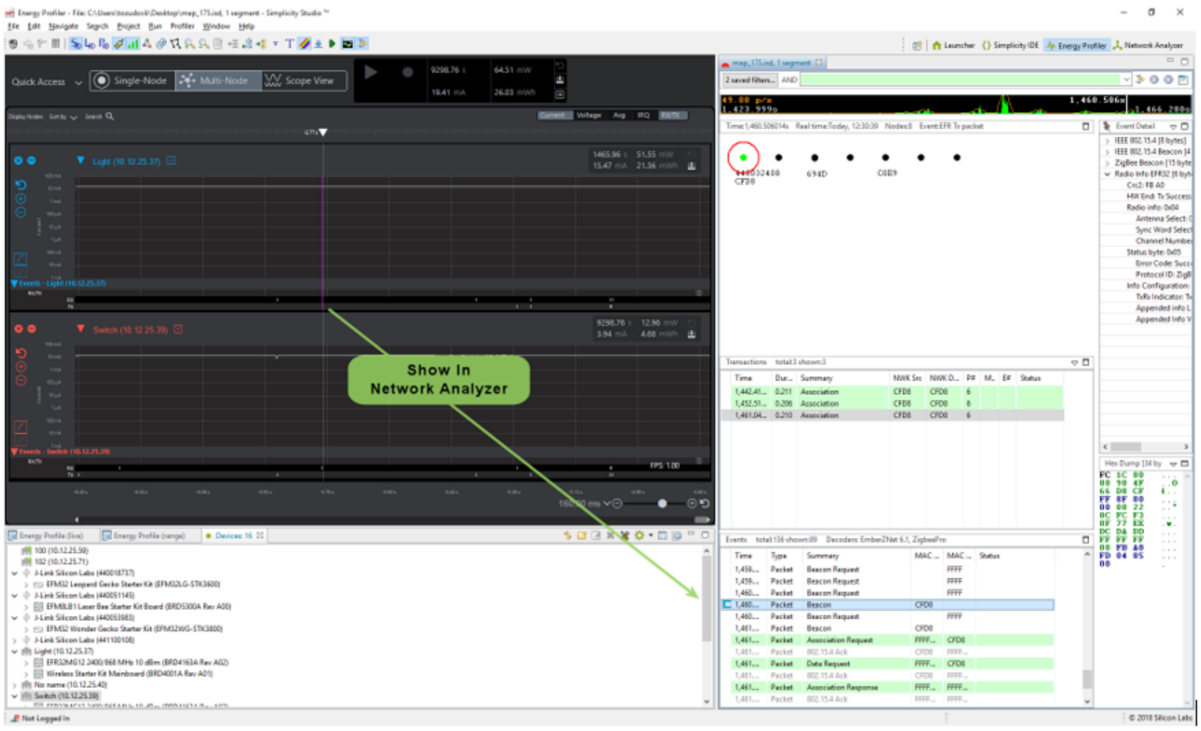




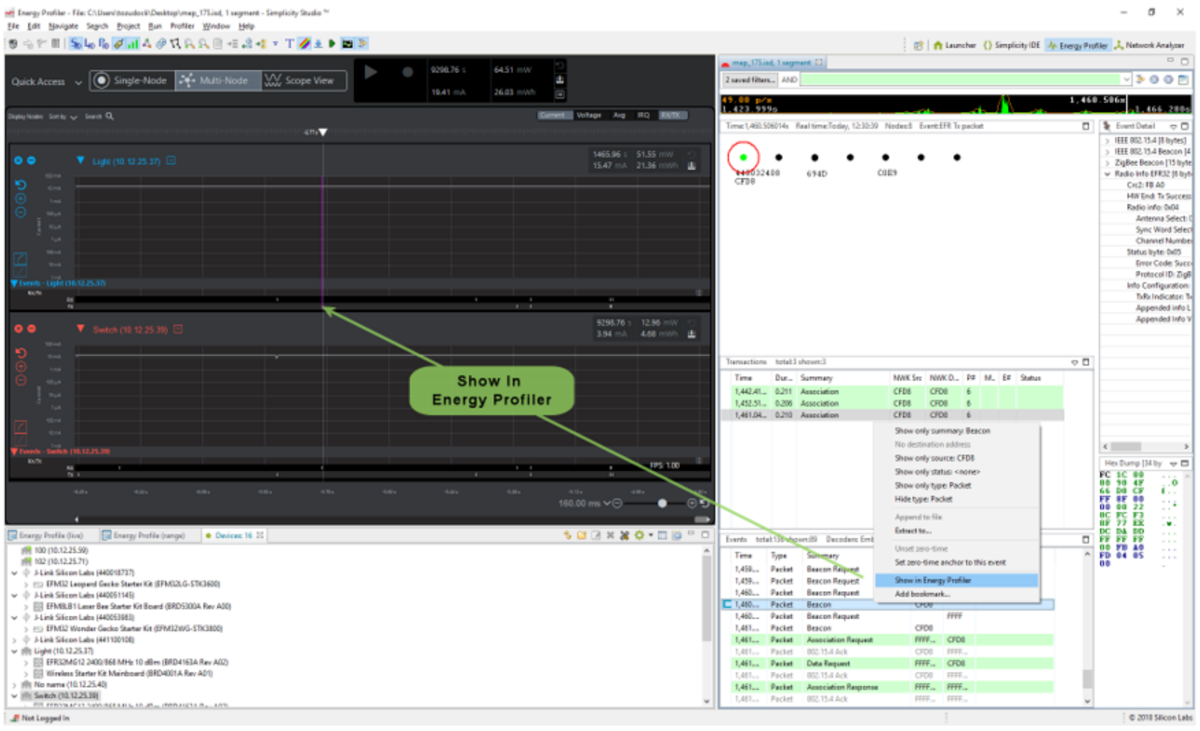
Once the file is open, you can examine previously recorded details. In the following figure, notice two devices transmitting and receiving packets. Select a TX or RX event and display the context menu. One of the options is **Show in Network Analyzer**.



Select that option and the Network Analyzer view opens either in the Energy Profiler perspective or the Network Analyzer perspective, based upon the preference setting.



Likewise, you can select an event in Network Analyzer and use the context menu to navigate back to Energy Profiler.



Use Cases for Multi-Node Energy Profiler and Network Analyzer

Depending upon your goals for an analysis session, there will be times when you will primarily use Network Analyzer, times when you will primarily use Energy Profiler, and times when you will find it advantageous to use them together.

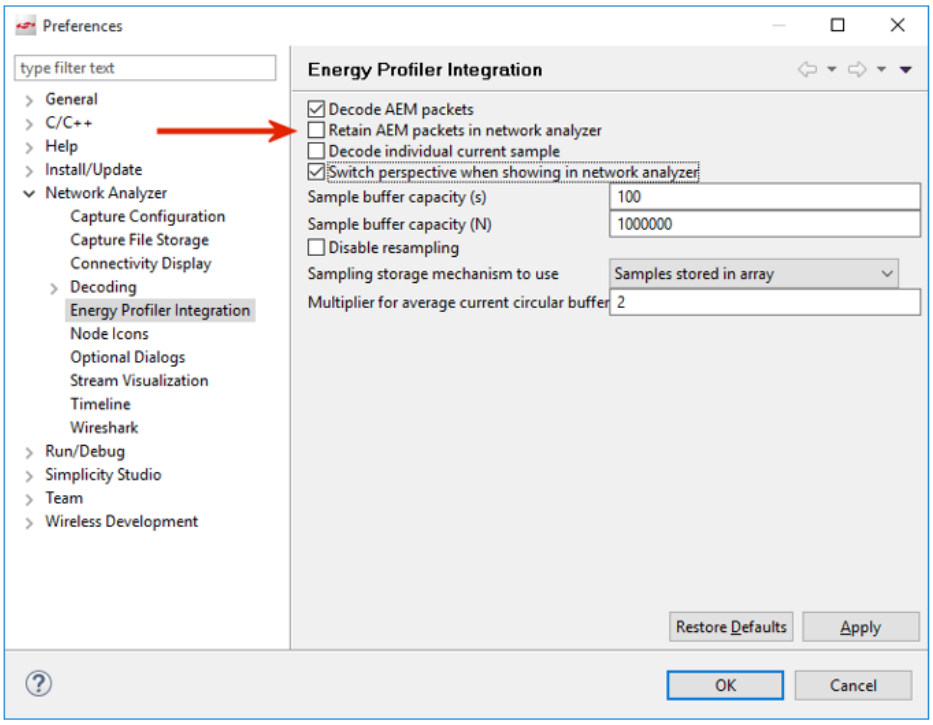
When you are interested in energy consumption of a single or multiple devices, Energy Profiler is where you will focus your time. From a network communication perspective, it provides only the start time of a TX or RX packet, and does not provide any decoding of the packets themselves. Still, this information can be revealing in your application's consumption of power relative to network activity. For example, by using the search tool, you can set search criteria that identify all points in the data where a packet was transmitted and the power exceeded a given value. If you have an expected power performance based upon your design, you can quickly identify when your system is operating outside its specification.

If you are interested in packet contents, node activity, and network interaction in a wireless network, Network Analyzer is where you will focus your time. Network Analyzer's multiple editor panes provide tiered displays of network activity, letting you drill down from a high-level map of node interactions to the details of each packet. Customizable filters enable you to specify exactly which network activities to display, allowing you to sift out information unrelated to a given task. These features allow you to determine when your network is behaving as expected, but are not available in Energy Profiler.

You will find yourself benefitting from the integration of these applications where these two information spaces intersect. For example, the Energy Profiler use case mentioned above ended at finding a transmit packet that exceeded expected power consumption. To investigate further, you could select **Show in Network Analyzer** to investigate that specific node's network activity and packet contents, in hopes of determining why the power was higher for that particular network transaction. In contrast, you may suspect that packets with specific content are the root cause of high power consumption. In Network Analyzer you could search for that data, and use **Show in Energy Profiler** to determine if indeed those packet contents are the root cause of high power consumption.

Post Analysis Using an ISD File

Both Network Analyzer and Energy Profiler capture data in the same format, which are saved in the ISD file format. Both applications can open an ISD file that has been save by either application. For example, if you have completed an Energy Profiler session and saved the ISD file, only to realize later that you are interested in the packet trace information, you would open that file from Network Analyzer to further investigate it. It is important to note, however, that captures started in Energy Profiler include packet trace data by default, but captures started in Network Analyzer only contain energy data if the preference to include it has been selected, as shown in the following figure.



Energy Profiler Integration

- [✓] Decode AEM packets
- [ ] Retain AEM packets in network analyzer
- [ ] Decode individual current sample
- [✓] Switch perspective when showing in network analyzer

Sample buffer capacity (s)	100
Sample buffer capacity (N)	1000000

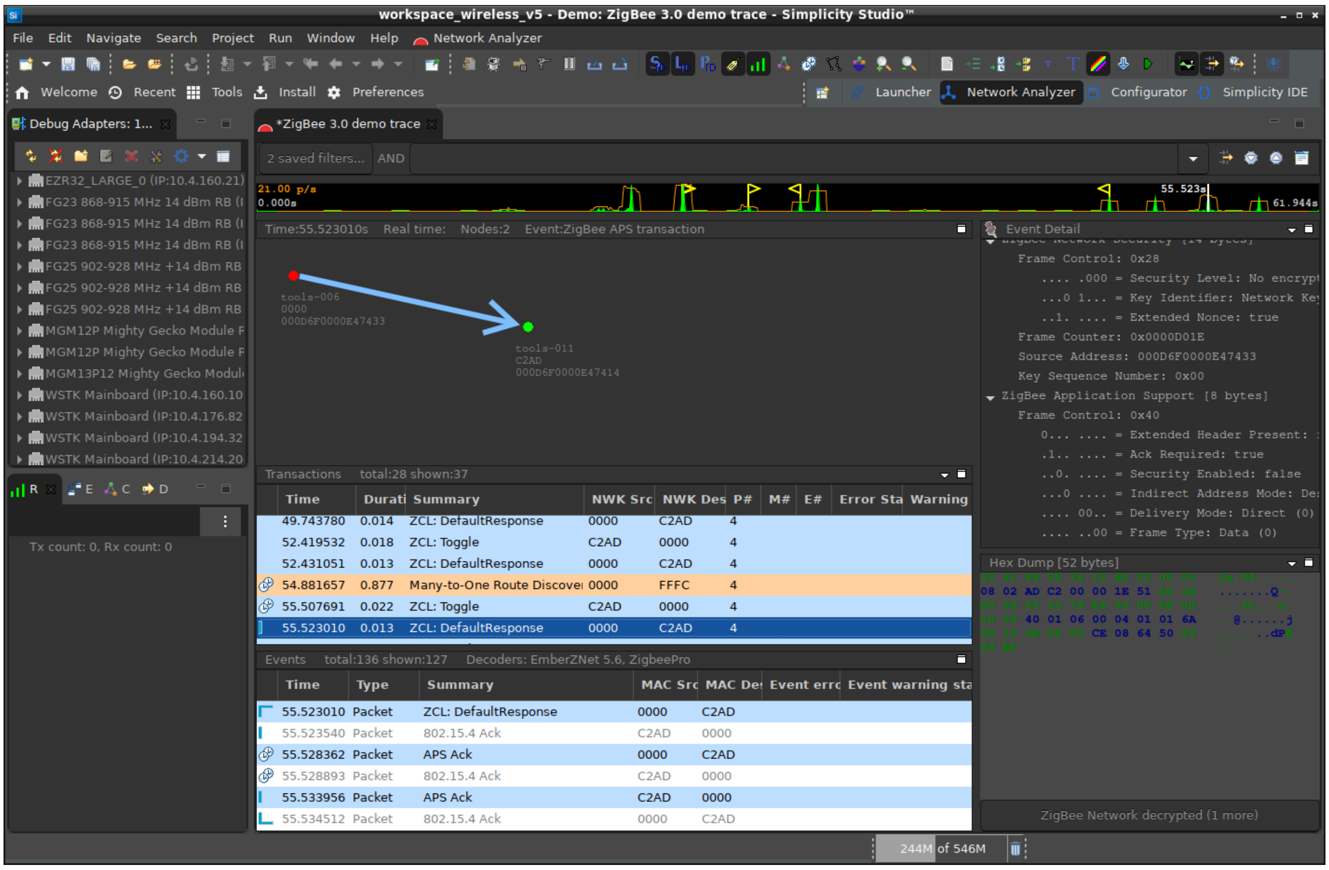
- [ ] Disable resampling

Sampling storage mechanism to use	Samples stored in array
Multiplier for average current circular buffer	2

Network Analyzer

Network Analyzer

Simplicity Studio® 5 (SSv5)'s Network Analyzer enables debugging of complex wireless systems. This tool captures a trace of wireless network activity that can be examined in detail live or at a later time.


workspace_wireless_v5 - Demo: ZigBee 3.0 demo trace - Simplicity Studio™
File Edit Navigate Search Project Run Window Help Network Analyzer
Welcome Recent Tools Install Preferences
Launcher Network Analyzer Configurator Simplicity IDE
Debug Adapters: 1...
*ZigBee 3.0 demo trace
2 saved filters... AND
21.00 p/s
0.000s
55.523s
61.944s
tools-006
0000
000D6F0000E47433
tools-011
C2AD
000D6F0000E47414
Tx count: 0, Rx count: 0
ZigBee Network decrypted (1 more)
244M of 546M

EZR32_LARGE_0 (IP:10.4.160.21)
FG23 868-915 MHz 14 dBm RB (I
FG23 868-915 MHz 14 dBm RB (I
FG23 868-915 MHz 14 dBm RB (I
FG25 902-928 MHz +14 dBm RB
FG25 902-928 MHz +14 dBm RB
FG25 902-928 MHz +14 dBm RB
MGM12P Mighty Gecko Module F
MGM12P Mighty Gecko Module F
MGM13P12 Mighty Gecko Modul
WSTK Mainboard (IP:10.4.160.10
WSTK Mainboard (IP:10.4.176.82
WSTK Mainboard (IP:10.4.194.32
WSTK Mainboard (IP:10.4.214.20

Time:55.523010s Real time: Nodes:2 Event:ZigBee APS transaction

Transactions total:28 shown:37

Time	Durati	Summary	NWK Src	NWK Des	P#	M#	E#	Error Sta	Warning
49.743780	0.014	ZCL: DefaultResponse	0000	C2AD	4				
52.419532	0.018	ZCL: Toggle	C2AD	0000	4				
52.431051	0.013	ZCL: DefaultResponse	0000	C2AD	4				
54.881657	0.877	Many-to-One Route Discove	0000	FFFC	4				
55.507691	0.022	ZCL: Toggle	C2AD	0000	4				
55.523010	0.013	ZCL: DefaultResponse	0000	C2AD	4				

Events total:136 shown:127 Decoders: EmberZNet 5.6, ZigbeePro

Time	Type	Summary	MAC Src	MAC Des	Event erro	Event warning sta
55.523010	Packet	ZCL: DefaultResponse	0000	C2AD		
55.523540	Packet	802.15.4 Ack	C2AD	0000		
55.528362	Packet	APS Ack	0000	C2AD		
55.528893	Packet	802.15.4 Ack	C2AD	0000		
55.533956	Packet	APS Ack	C2AD	0000		
55.534512	Packet	802.15.4 Ack	0000	C2AD		

Event Detail

```
    Frame Control: 0x28
        .... .000 = Security Level: No encrypt
        ...0 1... = Key Identifier: Network Key
        ..1. .... = Extended Nonce: true
    Frame Counter: 0x0000D01E
    Source Address: 000D6F0000E47433
    Key Sequence Number: 0x00
ZigBee Application Support [8 bytes]
    Frame Control: 0x40
        0... .... = Extended Header Present:
        .1.. .... = Ack Required: true
        ..0. .... = Security Enabled: false
        ...0 .... = Indirect Address Mode: De
        .... 00.. = Delivery Mode: Direct (0)
        .... ..00 = Frame Type: Data (0)
```

Hex Dump [52 bytes]

```
32 61 88 55 34 12 AD C2 00 00   2a.U4.....
08 02 AD C2 00 00 1E 51 2D 1E   .......Q-.
D0 00 00 33 74 E4 00 00 6F 0D   ...3t...o.
00 00 40 01 06 00 04 01 01 6A   ..@......j
08 10 0B 02 00 CE 08 64 50 23   .......dP#
99 EF                           ..
```

More than simply a packet sniffer, the Network Analyzer works with the data sniffer interface on the Silicon Labs wireless chips to provide direct feedback from the baseband radio of each device, allowing any supported radio to report detailed packet transmission and reception data, such as timestamps, link quality (or LQI), receive sensitivity (or RSSI), and CRC pass/fail results, all without any software overhead.

With SSv5, any PTI-enabled Silicon Labs platform can record the radio activity regardless of the application firmware that is being used, so there's no need to have a dedicated sniffer device installed to catch the traffic. The Network Analyzer also enables capture from multiple sources simultaneously into the same log file without falsely duplicating packets. This enables the developer to compare how well different radios in the network heard the same transmission.

In cases when detail is not desired, Network Analyzer makes it easier to understand the workings of a complex wireless protocol. Related packet events are automatically grouped into a Transactions pane within the capture view, allowing for quicker parsing of what's happening during that portion of the traffic log. Quickly access statistics like total duration, number of related packets, number of point-to-point and end-to-end retries, and unexpected conditions like requests with missing responses or deliveries where expected acknowledgments are missing.

In addition to capturing packet events, Network Analyzer also captures asserts, debug prints, and many other events.

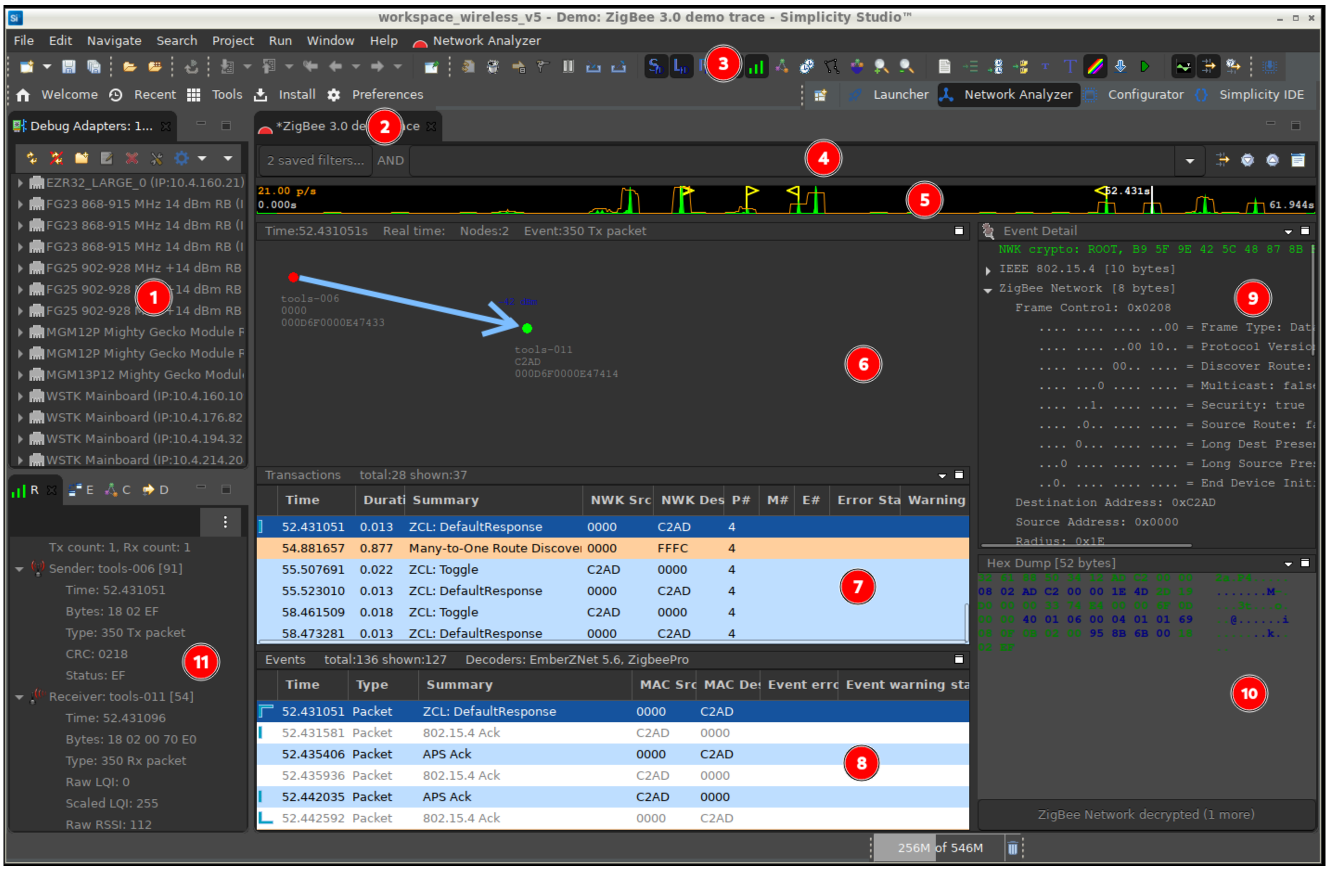
This guide contains the following sections:

- Network Analyzer Interface: Provides a guided tour of all elements of the Network Analyzer Perspective.
- Capturing Data and Managing Sessions: Describes how to perform live captures, and save and manage the resulting data.
- Viewing Data in Editors: Goes into detail about working with the data presented in both the Stream Editor and the Large File Editor.
- Filtering Captured Data: Explains how to focus on exactly what you need from a Network Analyzer session.
- Multinetwork Considerations: Discusses how to manage nodes that belong to more than one network.
- Custom Decoders: Explains the basics of how to create custom decoders.
- Network Analyzer Preferences: Provides a reference for the many ways you can customize Network Analyzer to meet your needs.

Network Analyzer Interface

Network Analyzer Interface

The Network Analyzer Interface is presented as a Network Analyzer perspective.



EZR32_LARGE_0 (IP:10.4.160.21)
FG23 868-915 MHz 14 dBm RB (I
FG23 868-915 MHz 14 dBm RB (I
FG23 868-915 MHz 14 dBm RB (I
FG25 902-928 MHz +14 dBm RB
FG25 902-928 +14 dBm RB
FG25 902-928 +14 dBm RB
MGM12P Mighty Gecko Module F
MGM12P Mighty Gecko Module F
MGM13P12 Mighty Gecko Modul
WSTK Mainboard (IP:10.4.160.10
WSTK Mainboard (IP:10.4.176.82
WSTK Mainboard (IP:10.4.194.32
WSTK Mainboard (IP:10.4.214.20

Time:52.431051s Real time: Nodes:2 Event:350 Tx packet

Tx count: 1, Rx count: 1
Sender: tools-006 [91]
Time: 52.431051
Bytes: 18 02 EF
Type: 350 Tx packet
CRC: 0218
Status: EF
Receiver: tools-011 [54]
Time: 52.431096
Bytes: 18 02 00 70 E0
Type: 350 Rx packet
Raw LQI: 0
Scaled LQI: 255
Raw RSSI: 112

```
NWK crypto: ROOT, B9 5F 9E 42 5C 48 87 8B
IEEE 802.15.4 [10 bytes]
ZigBee Network [8 bytes]
  Frame Control: 0x0208
    .... .... .... ..00 = Frame Type: Dat
    .... .... ..00 10.. = Protocol Versio
    .... .... 00.. .... = Discover Route:
    .... ...0 .... .... = Multicast: false
    .... ..1. .... .... = Security: true
    .... .0.. .... .... = Source Route: fa
    .... 0... .... .... = Long Dest Prese
    ...0 .... .... .... = Long Source Pre
    ..0. .... .... .... = End Device Init
  Destination Address: 0xC2AD
  Source Address: 0x0000
  Radius: 0x1E
```

Transactions total:28 shown:37

Time	Durati	Summary	NWK Src	NWK Des	P#	M#	E#	Error Sta	Warning
52.431051	0.013	ZCL: DefaultResponse	0000	C2AD	4				
54.881657	0.877	Many-to-One Route Discove	0000	FFFC	4				
55.507691	0.022	ZCL: Toggle	C2AD	0000	4				
55.523010	0.013	ZCL: DefaultResponse	0000	C2AD	4				
58.461509	0.018	ZCL: Toggle	C2AD	0000	4				
58.473281	0.013	ZCL: DefaultResponse	0000	C2AD	4				

Events total:136 shown:127 Decoders: EmberZNet 5.6, ZigbeePro

Time	Type	Summary	MAC Src	MAC Des	Event erro	Event warning sta
52.431051	Packet	ZCL: DefaultResponse	0000	C2AD		
52.431581	Packet	802.15.4 Ack	C2AD	0000		
52.435406	Packet	APS Ack	0000	C2AD		
52.435936	Packet	802.15.4 Ack	C2AD	0000		
52.442035	Packet	APS Ack	C2AD	0000		
52.442592	Packet	802.15.4 Ack	0000	C2AD		

The Network Analyzer perspective contains the following work areas:

- Debug Adapters view (1) - Lists all debug adapters and their connected hardware that are accessible to Network Analyzer. See Simplicity IDE User Interface Review
- Capture sessions (2) - Contain data captured from a node or set of nodes, with each session shown as a tab, and a live session shown in a ***Live** tab.
- Editor Panes - Display the data of a selected capture session in the editor area with up to five different panes. Network Analyzer supports two types of capture sessions:
  - Live sessions - Display data as it is captured; the display is continuously refreshed as new data arrives.
  - Saved sessions - Contain captured data that has been saved to permanent storage in a named .isd file. You can load a saved session and play it back at any time.
- Toolbar and Menu (3) and Supporting Views (11) - Provide options to control data capture and display.

A filter Bar (4) allows you to enter data filters, as described in Filtering Captured Data.

A Timeline Bar (5) displays the statistics of the traffic over time, as discussed in Viewing Data in editors

The Views area (11) displays additional views in a tabbed interface.

Editor Panes

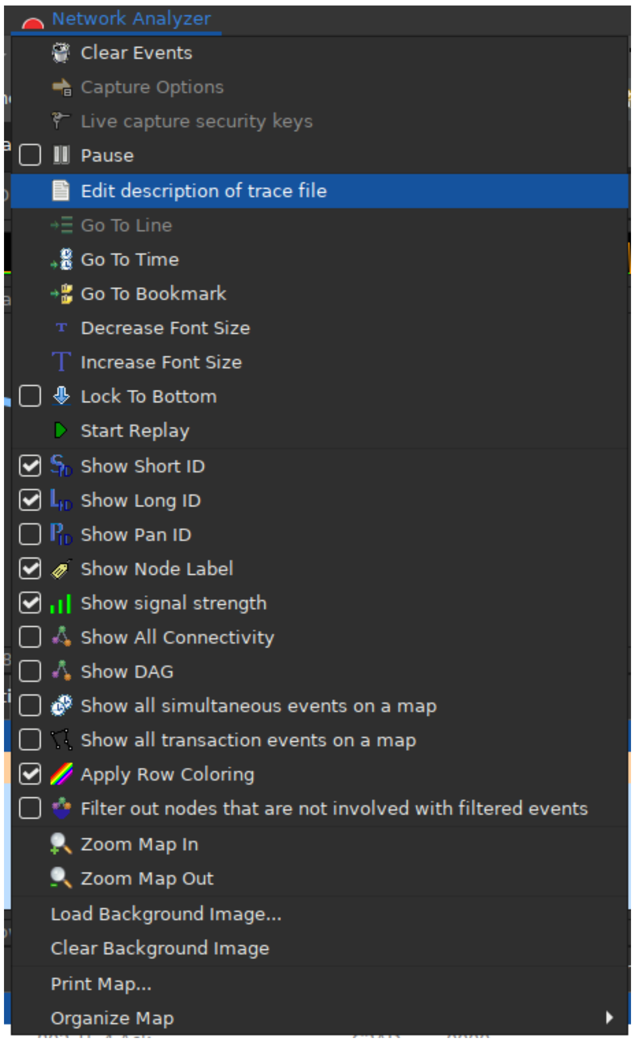
Editor panes provide different aspects of capture session data. Up to five editor panes can be open at any one time:

- **Map** (6) provides a graphical view of the network, where nodes are displayed with their network identifiers. The map also displays network activity.
- **Transactions** (7) displays high-level node interactions that might comprise multiple events.
- **Events** (8) displays information about all events transmitted and received during a capture session.
- **Event Detail** (9) displays the decoded contents of the event that is currently selected in the Events pane.
- **Hex Dump** (10) displays the data of the selected event in raw bytes. Network Analyzer highlights bytes that map to the data currently selected in the Event Detail pane. It shows multiple "layers", so if the packet is decrypted, the "raw" layer shows encrypted data, but the higher-level layers show this data progressively decrypted.

Using the editor is discussed in more detail in Viewing Data in Editors

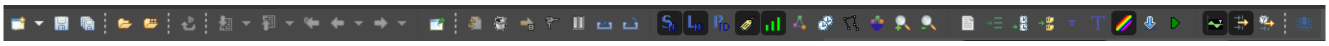
Toolbar and Menu

The Network Analyzer Toolbar and Menu provides shortcuts to frequently-used features. Selections and controls are either enabled or disabled depending on what the user is doing.



On the toolbar, hover over any control to see a short description.

The following describes the controls in order from left to right. Where the function is also available on the Network Analyzer menu, it is so noted.



Network Analyzer toolbar controls are added to the controls already present in the Simplicity IDE. See About the Simplicity IDE for more information. This section provides Network Analyzer-specific information, if any, about those controls.

Open File - Opens a file dialog from which the user may select one of the applicable extensions, such as .isd, or .log. Equivalent to File > Open File (Network Analyzer Trace, Energy Profiler, etc.).

Open Capture Directory - Opens the folder to which an output file has previously been saved. Equivalent to File > Other Network Analyzer Actions > Open Network Analyzer Capture Directory.

Restart a Process - Simplicity IDE function.

New - Creates a new project or other file. Equivalent to File > Other Network Analyzer Functions > New Live Capture Session.

Save (Ctrl+S) - Saves any changes to the currently opened file to disk. If the file has never been saved before, the user can say where they want it saved and what extension they want to give it. If the file was previously saved, Network Analyzer saves over the old file.

Save All (Ctrl+Shift+S) - Saves all files that have been edited, including any files open elsewhere in Simplicity Studio. As with Save, the user has the option to indicate where and how to save any files that have never been saved.

Next Annotation and **Previous Annotation** - Simplicity IDE function.

Last Edit Location - Simplicity IDE function.

Back and **Forward** - Simplicity IDE function.

Pin Editor - Simplicity IDE function.

Reopen Editor - Closes and reopens the editor. When you have changed or added new decryption keys to the preferences, it is useful to be able to reopen the file and run it through the decryptors and decoders again.

Clear Events - Clears the editor of all events. This is useful when you are capturing on a network and are waiting for some set of events to occur but do not want to keep everything else around.

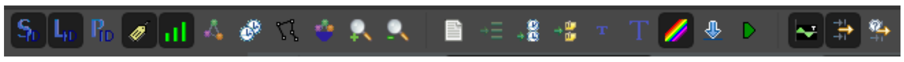
Live Capture Options - Opens the Capture Options dialog, used to tailor the condition under which events will be captured.

View and Modify Security Keys (Toolbar and Menu) - Opens the **Active live capture keys** dialog, where you can add security keys to be used during the current capture in progress. Any keys that are added will also be added to the list of security keys in the Network Analyzer preferences.

Pause Stream (Toolbar and Menu) - Pauses a capture without stopping it. This is useful to stop capturing for period of time, and then later continue as if nothing happened. Live events occurring during a pause are not retrievable.

Import - Select files to import and the type of import. Additional fields allow you to specify characteristics of the import. Equivalent to File > Network Analyzer Import.

Export - Select the exporter to use, and the output file name and location. Other options depend on the exporter. Equivalent to File > Network Analyzer Export.



Show Short Id (Toolbar and Menu) - Displays a node's shortId on the map, if one is known for the currently selected time.

Show Long Id (Toolbar and Menu) - Displays a node's long Id (64 bit identifier) on the map, if one is known.

Show Pan Id (Toolbar and Menu) - Displays the pan Id on the map, if one is know for the currently selected time.

Show Node Label (Toolbar and Menu) - Displays the node's label. If the node is connected to Network Analyzer over the backchannel, this value will be the host name of the adapter connected to the node.

Show Signal Strength (Toolbar and Menu)

Show All Connectivity (Toolbar and Menu) - Shows the quality of connections between nodes on the map.

Show all Simultaneous Events on Map (Toolbar and Menu) - During periods of heavy traffic, several simultaneous transactions may overlap. When enabled, the map shows all traffic. When it is disabled (default), the map shows only the traffic related to the currently selected transaction.

Show All Transactions on Map (Toolbar and Menu) - When enabled, all transactions that overlap with the currently selected one are shown on the map.

Filter Nodes (Toolbar and Menu) - Filters out nodes that are not involved with the filtered event.

Map Zoom In and **Map Zoom Out** (Toolbar and Menu) - Increases and decreases the size of the map in the map page. Zooming in can be useful if you are looking at a very large network where a large number of nodes are positioned very close together.

Edit Trace File Description (Toolbar and Menu) - Each saved Network Analyzer trace file contains a description. The description can be used to store information that is important to the trace but may not be included by default. Generally the description is used to provide context for the trace and any other information that may be of help to the viewer. A checkbox on the dialog determines if the description is shown when the file is loaded.

Go To Line (Toolbar and Menu) - Moves the event cursor directly to the event number entered. This is only enabled if the Stream preference "Show event numbers" is selected. To turn this feature on go to: Window > Preferences > Network Analyzer > Capture Configuration > Show event numbers.

Go to Time (Toolbar and Menu) - Moves the cursor to the transaction and event that match or immediately follow the specified time.

Go To Bookmark (Toolbar and Menu) - Moves the event or transaction cursor to the bookmark selected in the bookmark dialog. Assign bookmarks to events or transactions by right-clicking the event or transaction and selecting **Add Bookmark**.

Decrease Font Size and **Increase Font Size** (Toolbar and Menu) - Decreases and increases the size of the font used in the Event, Transaction and Detail panes.

Apply Row Coloring (Toolbar and Menu) - Turns coloring on and off in the Transaction and Event panes. Row colors are applied based on the pre-defined filters included in the Filter Manager.

Lock To Bottom (Toolbar and Menu) - Locks the event cursor to the bottom of the Event pane. During a live capture, this causes the Map and Details panes to always show the latest event. To remove the lock, select any event or transaction during a live session, which causes a view to scroll as the events are captured.

Start Replay (Toolbar and Menu) - Begins scrolling forward through events from the current event selected. If no event is selected, the scrolling will begin from the start of the current trace file. Once replay has started, converts to a **Stop Replay** function.

Timeline Bar - Toggles the timeline shown at the top of the currently opened Stream Editor.

Toggle Filter Bar - Toggles the Filter Bar at the top of the currently opened Stream Editor.

Show Filter Manager View - Toggles the Filter Manager View.

The following are selections on the Network Analyzer menu that are not on the toolbar:

Show DAG - Shows the connectivity DAG from the captured neighbor DAG events. Applies only in cases when the networking stack is instrumented with the correct abilities.

Load Background Image and **Clear Background Image** - Loads and clears a background image on the map pane. Also available on the Map pane context menu.

Print map - Prints the nodes as currently displayed in the map pane.

Organize map - Organizes the nodes on the map into a Default, Random, Square, or Hexagonal placement. Also available on the Map pane context menu.

Other Views

Network Analyzer provides access to a wide range of functionality through the use of views. The views are all accessible in the menu **Window > Show View**. For a complete listing of all available Views, select **Window > Show View > Other...**.

The following lists some of the most helpful views. The first four are open in the Views area by default.

Radio Info - The Radio Info View shows data captured for each event, as discussed in Radio Info View.

Event Difference - Event Difference view is a helper view that displays the specific differences between two packets, as discussed in Event Difference View.

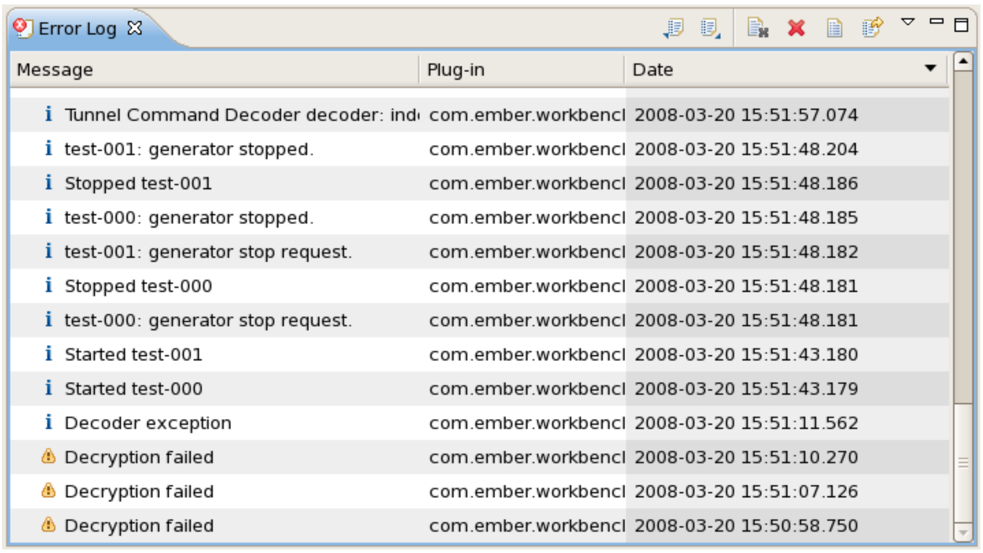
Connectivity view - (15.4 captures only) Displays a graph of network connectivity, using the neighbor information from the nodes.

Data capture view

Adapters - (deprecated)

Application Controller view - (deprecated)

Error Log - Decoder, decryption, and other types of errors are displayed in a tabular format in the Error Log View. Each error is shown in a single row with its summary message, the plugin or component that reported the error, and the date and time the error was encountered. To view detailed information about the error, double-click it. The error will be displayed in an Event Details dialog that includes the date, severity, message, and, if available, stack trace. The navigation at the top of the view allows you to perform basic functions on the Error log itself, including Export, Import, Clear, Delete, Open, and Restore.



Message	Plug-in	Date
ℹ Tunnel Command Decoder decoder: ind	com.ember.workbencl	2008-03-20 15:51:57.074
ℹ test-001: generator stopped.	com.ember.workbencl	2008-03-20 15:51:48.204
ℹ Stopped test-001	com.ember.workbencl	2008-03-20 15:51:48.186
ℹ test-000: generator stopped.	com.ember.workbencl	2008-03-20 15:51:48.185
ℹ test-001: generator stop request.	com.ember.workbencl	2008-03-20 15:51:48.182
ℹ Stopped test-000	com.ember.workbencl	2008-03-20 15:51:48.181
ℹ test-000: generator stop request.	com.ember.workbencl	2008-03-20 15:51:48.181
ℹ Started test-001	com.ember.workbencl	2008-03-20 15:51:43.180
ℹ Started test-000	com.ember.workbencl	2008-03-20 15:51:43.179
ℹ Decoder exception	com.ember.workbencl	2008-03-20 15:51:11.562
⚠ Decryption failed	com.ember.workbencl	2008-03-20 15:51:10.270
⚠ Decryption failed	com.ember.workbencl	2008-03-20 15:51:07.126
⚠ Decryption failed	com.ember.workbencl	2008-03-20 15:50:58.750

Expression Manager - Also known as the Filter Manager, this tool is used to compose and edit custom filtering expressions.

Progress - The Progress View shows the progress of user actions. Actions that take a long time to execute may be managed within the Progress View. For example, the application upload action can take several seconds. The Progress View provides a user interface for managing this action. If you wish to stop an action, you may do so in the Progress View.

Sample Application Wizard - (deprecated)

Other views of interest are:

Event Detail - Event details are normally shown in the EventDetail Pane within the Stream Editor. If you want to see the event details in a separate window, you can open the Event Detail View. This view shows the details for the currently selected Event in the same format as the Event Detail Pane, but in a view that you can pull outside of Network Analyzer and resize to your liking.

Hex Dump - Similar to event details, the hex dump information is shown the Hex Dump Pane within the Stream Editor. If you want to see the hex dump in a separate window, you can open the Hex Dump View. This view shows the hex dump of the currently selected Event in the same format as the Hex Dump Pane, but in a view that you can pull outside of Network Analyzer and resize to your liking.

Search - Search data is displayed in the Search Pane within the Large File Editor. However, as with the Event Details and Hex Dump Panes, you may wish to see this data in a separate view. The Search View allows you to view search results in a window that you can pull outside of Network Analyzer and resize to your liking.

Shell - The Shell View provides command line access to the scripting capabilities of Network Analyzer. For a list of all commands available within the shell, simply enter `help` at the command line.

Capturing Data and Managing Sessions

Capturing Data and Managing Sessions

Network Analyzer can display one or more network capture sessions. Each capture session displays the transaction and event data captured from one or more nodes. The captured data is shown in editor panes. In general, Network Analyzer captures all incoming and outgoing packet data via the selected adapters, regardless of whether the host nodes have sniffer applications. The captured data includes failed transmissions, as well as debug messages from node applications that are compiled in debug mode.

This kind of capture is called a perfect trace session. The capture nodes of a perfect trace session are not sniffers but nodes that might be running your own application that you are trying to debug. The perfect trace session compiles all incoming and outgoing data from each node in chronological real time, providing a richly-layered display of all activity within a network. A perfect trace session can be especially useful for debugging a network in development as it allows you to see every packet on the network.

Network Analyzer supports two types of capture sessions:

- Live sessions display data as it is captured. The display is continuously refreshed as new data arrives. When a live session starts, it is unnamed and its data is maintained in temporary storage until you save it to a named file. Network Analyzer can capture live session data from multiple sets of adapters into the same session. You cannot, however, capture from one adapter into different sessions at the same time. You can run more than one live session at the same time.
- Saved sessions contain captured data that has been saved to permanent storage in a named .isd file. You can load a saved session and analyze it at any time.

When you start Network Analyzer, it opens a new live session. Network Analyzer displays each session in an editor with its own tab. The tab of each saved session is labeled with the session's file name; the first live session is labeled **Live**, the second **Live1**, the third **Live2**, and so on.

You can capture data from the node of any connected Debug Adapter, from one node at a time or multiple nodes simultaneously. You can also capture all network traffic over the current channel by capturing data from a connected sniffer node.

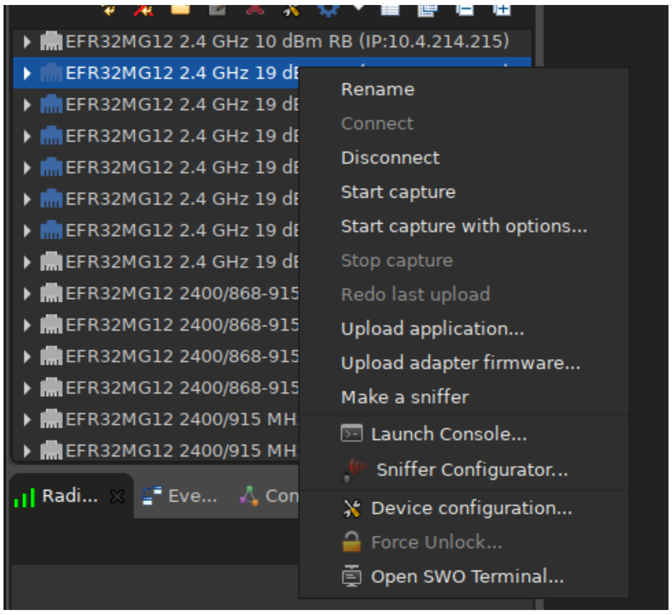
Note: The types of data captured from a node depend primarily on the software protocol running on the node, and also the capabilities of the node's radio chip and Debug Adapter.

Starting a Capture

Before starting a capture, open **Preferences > Network Analyzer > Decoding > Stack Versions** and make sure that the protocol running on the adapters is selected. If you are working in a multiprotocol environment, select 'Auto-detecting decoder stack'. A change here will not affect any active capture sessions but will apply to the next capture you start.

To start a capture:

1. Select one or more connected adapters. Multiple adapters are usually on the same network.
2. Right-click the selected adapters.
3. On the context menu, select **Start Capture**.



Alternatively, select **File > Other Network Analyzer Actions > New Live Capture Session**. This creates a new live capture session and puts it on top of the editor list.

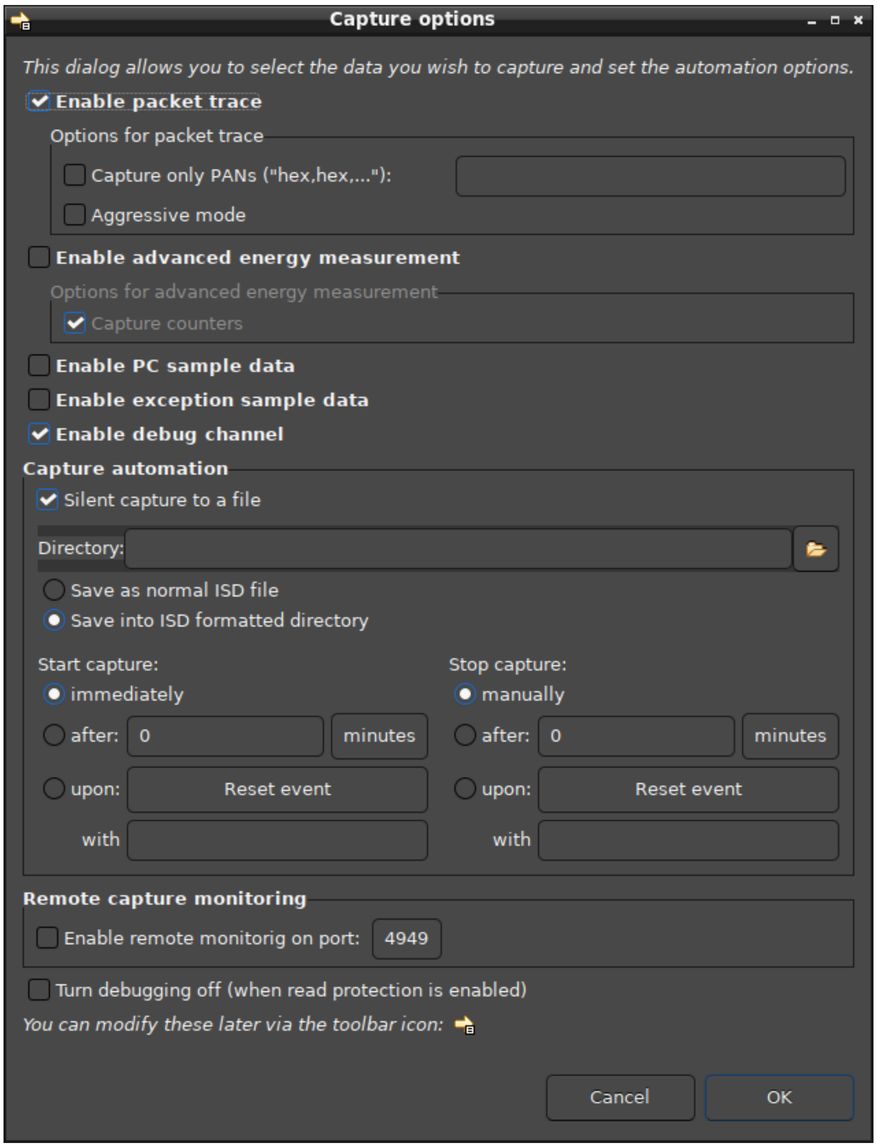
When you start a capture on an adapter, or on multiple adapters at the same time, the live session used for the capture is assigned in one of the following ways:

- If no live sessions are currently active, a new live session is created and used.
- If live sessions are active, but they are not on top of the editor stack (for example, another file is opened and currently on top), a new live session is created and used.
- If a live session is active and it is also on top of the editor stack, then this session will be used for capture.

Capturing with Options

You can filter packets out of the stream during a capture. For example, you can choose to see only packets from a certain PAN ID and drop all other packets.

1. Select one or more connected adapters.
2. Right-click the selected adapters.
3. On the menu, select **Start capture with options**.
4. Configure options in the resulting dialog, and click **OK**.


Capture options

This dialog allows you to select the data you wish to capture and set the automation options.

[✓] **Enable packet trace**

Options for packet trace
- [ ] Capture only PANs ("hex,hex,..."):
- [ ] Aggressive mode

[ ] **Enable advanced energy measurement**

Options for advanced energy measurement
- [✓] Capture counters

[ ] **Enable PC sample data**

[ ] **Enable exception sample data**

[✓] **Enable debug channel**

Capture automation

[✓] Silent capture to a file

Directory:

○ Save as normal ISD file
● Save into ISD formatted directory

Start capture:	Stop capture:
● immediately	● manually
○ after: 0 minutes	○ after: 0 minutes
○ upon: Reset event	○ upon: Reset event
with	with

Remote capture monitoring

[ ] Enable remote monitorig on port: 4949

[ ] Turn debugging off (when read protection is enabled)

You can modify these later via the toolbar icon:

Cancel OK

In the capture options dialog, you can set the following:

- Capture only PANs: (15.4 protocols only) This filter, based on PAN ID, allows a comma-separated list of hexadecimal values.
- Aggressive mode: Filters out all events that may not be or definitely are not packets.
- Enable advanced energy measurement: This enables AEM packet filtering. AEM packets are used with Energy Profiler. If you turn this option on and later look at the data in Energy Profiler, you will not see any packets.
- Enable PC sample data: If selected, captures diagnostic events.
- Enable exception sample data: If selected, captures Java framework exceptions, which are used for deep diagnostics of possible Network Analyzer bugs.
- Enable debug channel: If selected, enables capturing all other non-packet events.
- Silent capture to file: If selected, performs a lengthy capture as a background task. Traffic will not show in the GUI until the capture is stopped and the file is opened. The silent capture enables you to run a capture over several days, as it does not consume memory resources, only disk space.
- The start capture options specify the trigger for delayed start of capture.
  - immediately: No automation, capture starts right away.
  - after: Capture starts after a certain time or after a certain number of events.
  - upon: Capture starts upon Node reset, or upon an event containing a specified ASCII or byte pattern.

  The triggers for starting and stopping capture on node reset work only if you have DEBUG level NORMAL turned on for the node that you are capturing from. With the debug level set to Normal, the chip will send debug information, including node resets, over the back channel to Network Analyzer. Network Analyzer uses these reset and other commands to trigger the start-and-stop capture process. If an image does not have debug turned on, it will not send reset information to Network Analyzer, and Network Analyzer has no basis for triggering start or stop capture.

You can also modify some, but not all, capture options through the Live Capture Options toolbar control.

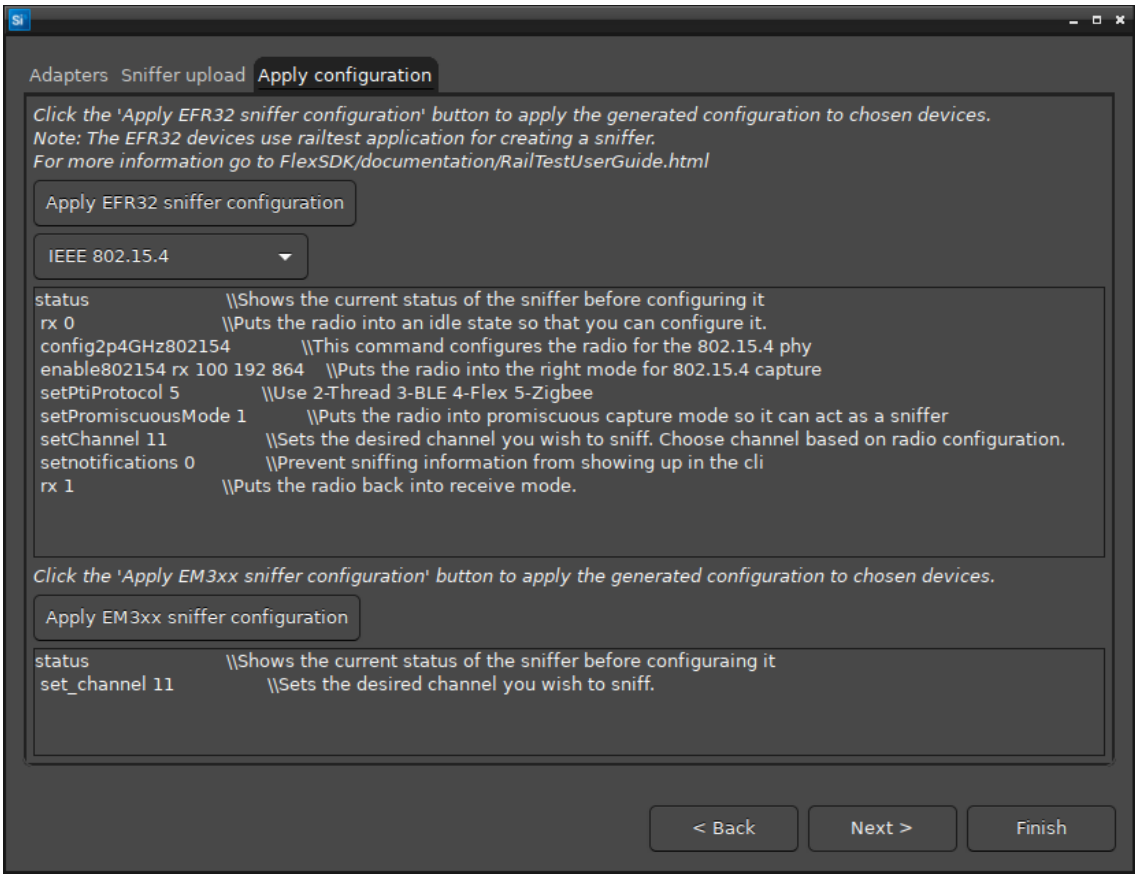
Capturing from a Sniffer Node

Any adapter's node that connects to Network Analyzer can be designated as a sniffer. A sniffer node is capable of capturing all data that is transmitted among nodes over the designated channel.

A sniffer node must have a sniffer application loaded. The sniffer application enables the node to capture over-the-air transmissions between nodes over the designated channel. When you start capturing from a sniffer node, the sniffer node captures all packets that are exchanged by the nodes on the designated channel.

If no sniffer application is currently loaded, load one in either of the following ways:

- In the Debug Adapters view, right-click the adapter and select **Sniffer Configurator**.
  - On the Adapters tab, select the adapter.
  - On the Upload tab, select the target adapter.
  - On the Apply Configuration tab, apply the appropriate configuration.



Click the 'Apply EFR32 sniffer configuration' button to apply the generated configuration to chosen devices.
Note: The EFR32 devices use railtest application for creating a sniffer.
For more information go to FlexSDK/documentation/RailTestUserGuide.html

```
status                        \\Shows the current status of the sniffer before configuring it
 rx 0                         \\Puts the radio into an idle state so that you can configure it.
 config2p4GHz802154           \\This command configures the radio for the 802.15.4 phy
 enable802154 rx 100 192 864   \\Puts the radio into the right mode for 802.15.4 capture
 setPtiProtocol 5             \\Use 2-Thread 3-BLE 4-Flex 5-Zigbee
 setPromiscuousMode 1         \\Puts the radio into promiscuous capture mode so it can act as a sniffer
 setChannel 11                \\Sets the desired channel you wish to sniff. Choose channel based on radio configuration.
 setnotifications 0           \\Prevent sniffing information from showing up in the cli
 rx 1                         \\Puts the radio back into receive mode.
```

Click the 'Apply EM3xx sniffer configuration' button to apply the generated configuration to chosen devices.

```
status                        \\Shows the current status of the sniffer before configuraing it
 set_channel 11               \\Sets the desired channel you wish to sniff.
```

- In the Debug Adapters view, right-click the adapter and select **Make Sniffer**. Network Analyzer will first check whether the node has a Sniffer image on it before attempting to upload the new Sniffer image.

Stopping a Capture

1. Select one or more connected adapters.
2. Right-click the selected adapter(s).
3. On the context menu, select **Stop Capture**.

To resume capture, select **Start Capture**.

Pausing a Capture

Pause a capture at any time by selecting **Pause** on the Network Analyzer menu or the toolbar. This is a convenient way to stop capturing from a device or devices without having to start a new capture at a later time. Events that occur during the pause are lost, and cannot be retrieved.

Clearing Session Events

Click **Clear Events** on the Network Analyzer menu or toolbar to purge all events and their associated transactions from the current session. **Caution!** You cannot retrieve cleared events.

This is mostly used if you are working on a scenario on an embedded node, where you control some activity through command-line actions. You then simply "Clear" events between each retry, instead of having to do a complete "Start Capture / Stop Capture".

Note: Network Analyzer is designed to stay connected and continue capturing, even when the firmware on the target node is uploaded and the node resets. That does not stop the capture session.

Saving a Session

When you start a capture, it is initially written to an unnamed live session. At any point during a live session, you can save the data thus far captured to a file by selecting File > Save or clicking the toolbar Save control. After you save a session file, Network Analyzer continues to append capture data to it; however, you must save again in order to retain this data in the session file.

Network Analyzer saves session data to an .isd file, which is a compressed file that stores session data and the network state. Network state includes display settings such as map modifications, which Network Analyzer restores when you reload the session file.

Network Analyzer closes a saved session from further captures after you explicitly stop the capture, or when you start another live session. After a saved session is closed, it cannot be reopened to capture more data.

If you modify a saved session file - for example, set bookmarks or reposition icons in the Map pane - Network Analyzer asks whether to save or discard those changes before you close the session.

Note: For security reasons, Security Keys that you may use to decrypt captured data are not included in saved .isd files by default. If you wish to share security keys with your files, you should turn on the option "Save decryption keys in Network Analyzer files" on the Security Keys preference page, which you can access by selecting Window > Preferences > Network Analyzer > Decoding > Security Keys

Saving Multiple Sessions

If multiple open capture sessions have unsaved data, you can save all of them at once by selecting **File > Save All**, or using the toolbar **Save All** control.

Exporting to Other File Formats

To export a capture session to another file format:

1. Select **File > Network Analyzer Export** or click the toolbar **Network Analyzer Export** control.
2. Select the export format
3. Name the output file.
4. Click **Save**.

Extracting Individual Events

You can extract specific events from the Transactions pane, Events pane, or Hex Dump pane, and save them into a separate text log file.

To extract specific events to a text file:

1. Right-click the event in the Transactions, Events, or Hex Dump panes that you want to extract.
2. On the menu, select **Extract to**.
3. Name the output file with a text extension.
4. Click **Save**.

Once you specify a file, you can append additional events to it by right-clicking the event that you want to append and selecting **Append to <file>**.

Open and Close Options

To open a capture file, select **File > Open Recent File (Network Analyzer trace ...)** or click the toolbar control.

To close a single capture session, close the session tab.

To close all capture sessions, right-click any session tab and select **Close All**, or select **File > Close All**.

To close all but the current capture session, right-click its session tab and select **Close Others**.

Replaying a Session

(Rarely used) Replay events of the current session, whether live or saved, by selecting **Start Replay** on the Network Analyzer menu or the toolbar. Network Analyzer replays the session from the selected event at a constant speed. Replaying events in a live session has no effect on the capture in progress.

Once replay has started, both the toolbar control and the menu selection convert to a **Stop Replay** function.

Viewing Data in Editors

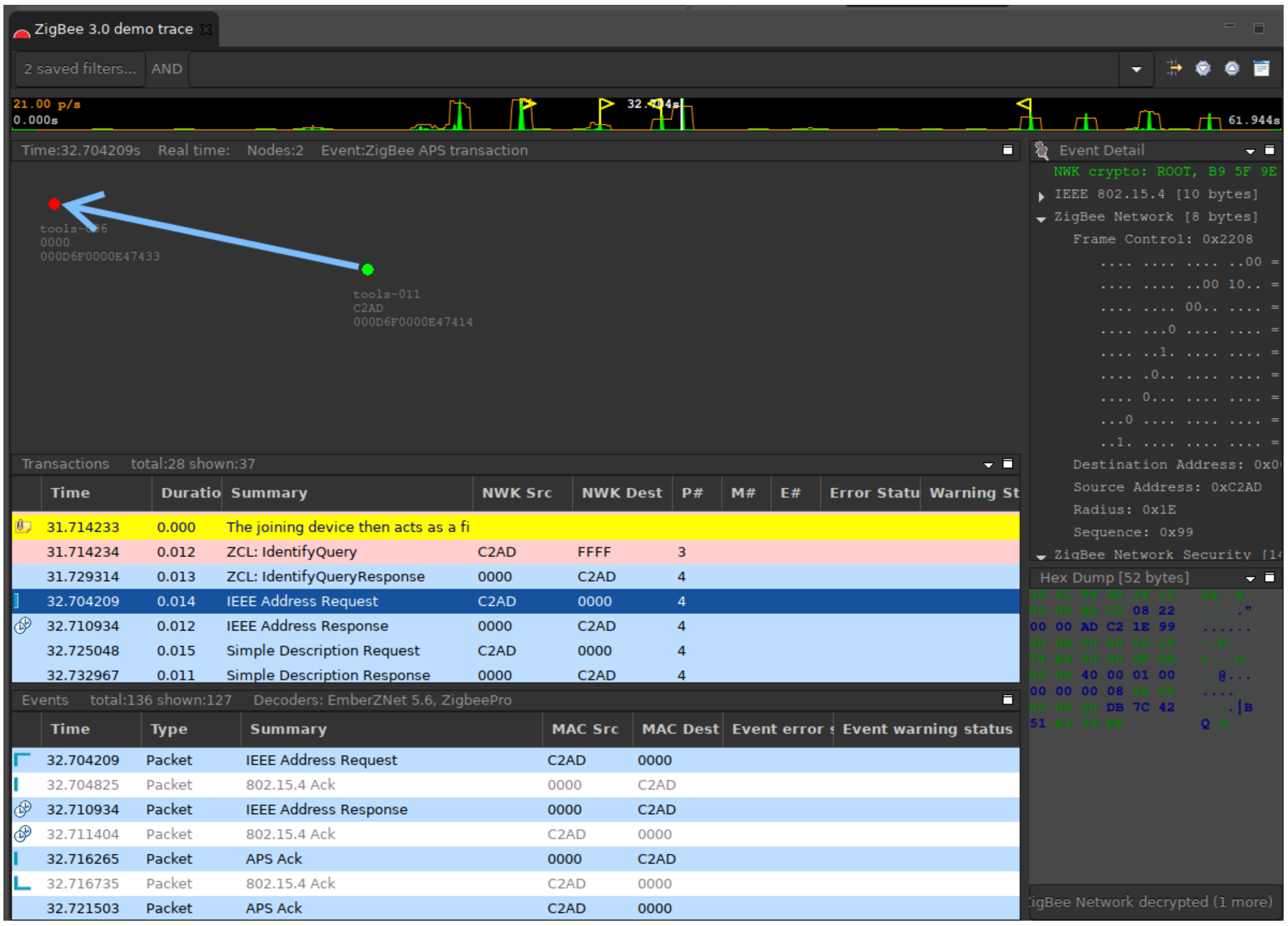
Viewing Data in Editors

On the File menu, select **Open File (Network Analyzer trace ..)** or **Open Recent File (Network Analyzer trace ...)** to open data in an editor. If the file is smaller than the size set in Preferences > Network Analyzer > Capture File Storage for a file to be considered large, Network Analyzer opens it in the Stream Editor. Otherwise, Network Analyzer opens it in the Large File editor.

- Stream Editor: The Stream Editor decrypts, decodes, and displays details of individual events.
- Large File Editor: The Large File Editor does not offer any detailed decoding and presentation of events. Instead, it provides a high-level overview of a file and allows users to open their points of interest in the Stream Editor.

The Stream Editor provides details about individual events. However, in the case of really large captures, this may tax system resources. The Large File Editor shows an overall timeline and node statistics. It allows you to scan very large captures for areas of interest, which you can then open in Stream Editor.

Stream Editor



Time:32.704209s   Real time: Nodes:2   Event:ZigBee APS transaction

Transactions   total:28 shown:37

	Time	Duratio	Summary	NWK Src	NWK Dest	P#	M#	E#	Error Statu	Warning St
	31.714233	0.000	The joining device then acts as a fi							
	31.714234	0.012	ZCL: IdentifyQuery	C2AD	FFFF	3				
	31.729314	0.013	ZCL: IdentifyQueryResponse	0000	C2AD	4				
	32.704209	0.014	IEEE Address Request	C2AD	0000	4				
	32.710934	0.012	IEEE Address Response	0000	C2AD	4				
	32.725048	0.015	Simple Description Request	C2AD	0000	4				
	32.732967	0.011	Simple Description Response	0000	C2AD	4				

Events   total:136 shown:127   Decoders: EmberZNet 5.6, ZigbeePro

	Time	Type	Summary	MAC Src	MAC Dest	Event error s	Event warning status
	32.704209	Packet	IEEE Address Request	C2AD	0000		
	32.704825	Packet	802.15.4 Ack	0000	C2AD		
	32.710934	Packet	IEEE Address Response	0000	C2AD		
	32.711404	Packet	802.15.4 Ack	C2AD	0000		
	32.716265	Packet	APS Ack	0000	C2AD		
	32.716735	Packet	802.15.4 Ack	C2AD	0000		
	32.721503	Packet	APS Ack	C2AD	0000		

```
  NWK crypto: ROOT, B9 5F 9E
▸ IEEE 802.15.4 [10 bytes]
▾ ZigBee Network [8 bytes]
    Frame Control: 0x2208
        .... .... .... ..00 =
        .... .... ..00 10.. =
        .... .... 00.. .... =
        .... ...0 .... .... =
        .... ..1. .... .... =
        .... .0.. .... .... =
        .... 0... .... .... =
        ...0 .... .... .... =
        ..1. .... .... .... =
    Destination Address: 0x0
    Source Address: 0xC2AD
    Radius: 0x1E
    Sequence: 0x99
▾ ZigBee Network Security [14
```

```
32 61 88 9D 34 12    2a..4.
00 00 AD C2 08 22    ....."
00 00 AD C2 1E 99    ......
2D 0A 50 00 00 14    -.P...
74 E4 00 00 6F 0D    t...o.
00 00 40 00 01 00    ..@...
00 00 00 08 86 00    ......
00 00 00 DB 7C 42    ...|B
51 A3 39 EF          Q.9.
```

The Stream Editor contains five editor panes, each of which provides a different view of the captured session data:

- Map pane: Provides a map of the network, with nodes displayed with their network identifiers. The map also displays network activity.
- Transactions pane: Displays high-level node interactions that might comprise multiple events.
- Events pane: Displays information about all packets transmitted and received during a capture session.
- Event Detail pane: Displays the decoded contents of the packet that is currently selected in the Events pane.
- Hex Dump pane: Displays the data of the selected event in raw bytes. Network Analyzer highlights bytes that map to the data currently selected in the Event Detail pane.

All five editor panes may be open at once. Live captured data is continuously updated and displayed in the editor panes.

A Timeline Bar displays the statistics of the traffic over time.

Views are presented in a tabbed interface in the lower left of the default Network Analyzer perspective.

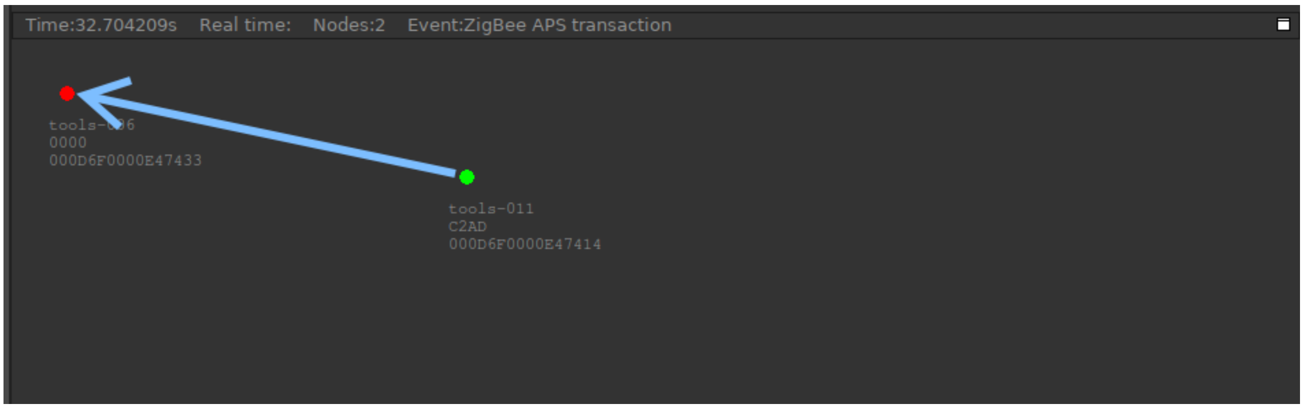
- Radio Info View: Shows the information from the radio of all the receivers in the network that have heard the currently selected event.
- Event Difference View: Displays the differences between two packets.
- Connectivity View: (15.4 captures only) Displays a graph of network connectivity, using the neighbor information from the nodes.

For more information about using the Stream Editor panes, see the Editor navigation tools.

Map Pane

The following information is applicable to 15.4 networks only.

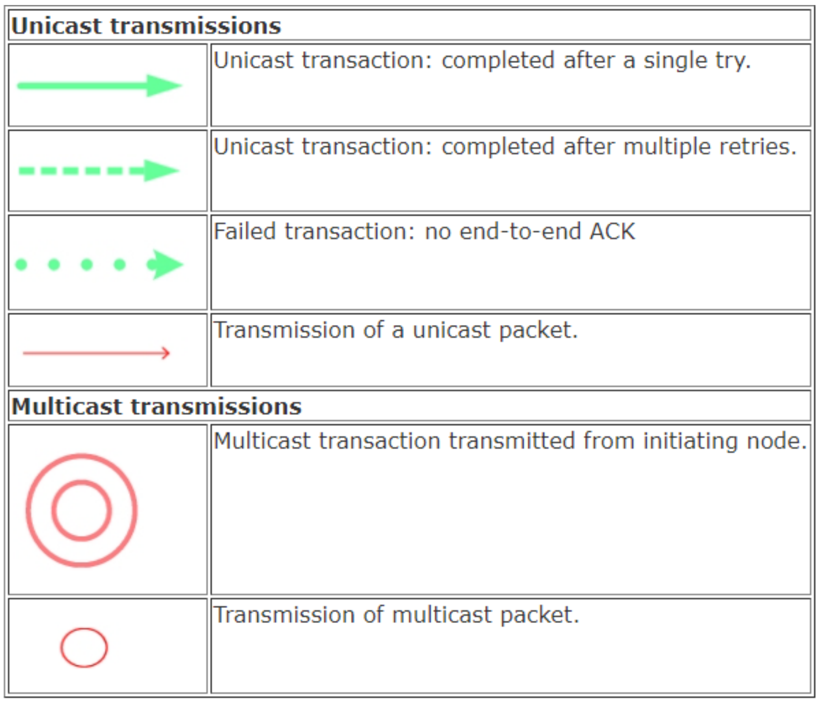
The Map pane shows all interaction between nodes at a high level. As events occur or are replayed, the Map pane refreshes to show the pattern of network communication. Debug messages issued from a node also display next to the node.


Time:32.704209s   Real time: Nodes:2   Event:ZigBee APS transaction

Each node in the map pane is given a different color depending on its capabilities within the network as they are understood by Network Analyzer based on captured data.

- RED: The node is a network coordinator.
- BLACK: The node is a router.
- GREEN: Default color for network nodes.

The following figure shows the graphical elements that appear in the Map pane to depict network activity. Thick lines depict transactions, while thin lines depict single packets.



Unicast transmissions	
→	Unicast transaction: completed after a single try.
⇢	Unicast transaction: completed after multiple retries.
⋯→	Failed transaction: no end-to-end ACK
→	Transmission of a unicast packet.
Multicast transmissions	
◎	Multicast transaction transmitted from initiating node.
○	Transmission of multicast packet.

Note: The colors shown vary according to the transaction or event type, and can be configured through the Filter Manager.

The data that is shown for each node is managed through menu/toolbar options:

- **Show Short ID** toggles display of the node's 16-bit address that is unique within the personal area network (PAN).
- **Show EUI64** toggles display of the node's unique 64-bit IEEE address.
- **Show PAN ID** toggles display of the PAN identifier of the node's network. This label can be useful when the map displays multiple networks.
- **Show Node Label** displays the custom label that you create for map display only.
- **Show LQI** toggles display of link quality data that pertains to the quality of connection between nodes. This is available with perfect trace captures, but not with sniffer captures.
- **Show Connectivity** shows the neighbor relationships between nodes in the network.
- **Simultaneous Events** displays on the Map pane all events that occurred at the same time as the transaction or event that is currently selected. The currently selected event is in color and any other events display in gray.
- **Zoom Map In** and **Zoom Map Out** enlarge and shrink the space that the map uses to display nodes. Zoom options have no effect on the size of node icons.

You can move node icons within the Map pane display. This has no effect on network functionality. However, it can help to highlight certain node interactions and relationships. When you move node icons in a session, Network Analyzer asks whether to save those changes before you close the session.

Right-click anywhere in the map pane to bring up a context menu.

- **Organize Map** establishes the layout of all nodes on a map. You can also modify individual node positions as needed. The following layouts are available:
  - Default Placement aligns nodes in a linear pattern.
  - Random Placement scatters the nodes across the map randomly.
  - Square Grid aligns the nodes in a grid.
  - Hexagonal Grid aligns the nodes in a hexagonal, offset pattern.
- **Load Background Image** and **Clear Background Image** manage the display of a background image in the Map pane.

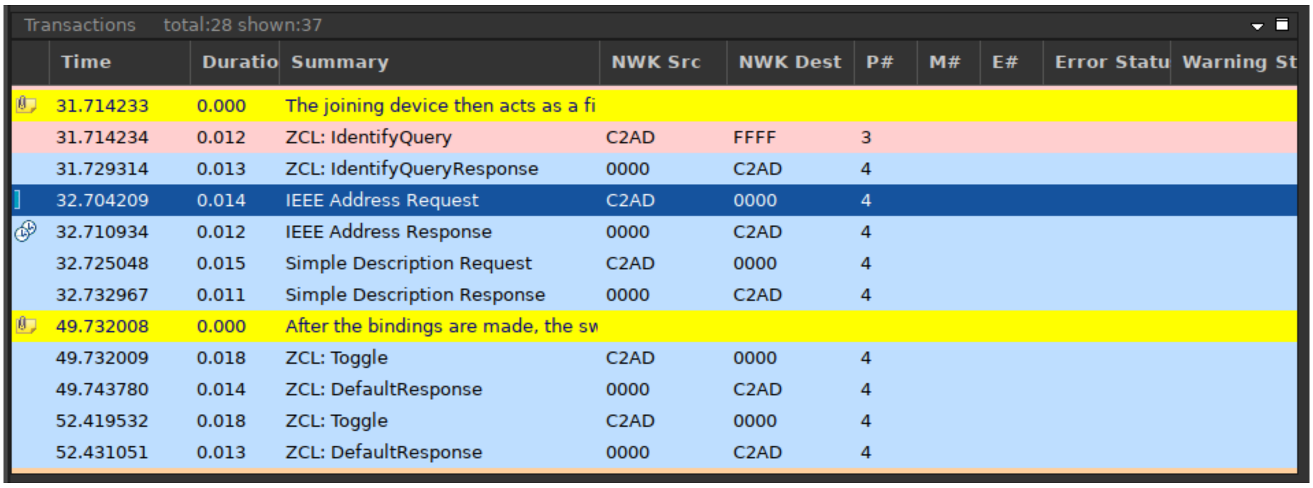
Right-click on a node to bring up a context menu.

- **Assign EUI64** lets you assign a EUI64 to a node (not available if Network Analyzer obtains the EUI64). Network Analyzer obtains a node's EUI64 only when that node associates with a network. If the node already belongs to a network when a session begins, its EUI64 is unknown. This option lets you display a known EUI64 for a node; the node's actual EUI64 is unaffected by this label. The Multinetwork checkbox can be used to indicate that the node is operating on multiple networks. See Multinetwork Considerations for more information.

- **Multinetwork** toggles the multinetwork property. See Multinetwork Considerations for more information.
- **Label** lets you customize the node's adapter (device) label with any string up to 25 characters long. This string appears in brackets after the node's device name. (By default, the Map pane labels each node that is undergoing capture with its device name.) You can also make the labels time-dependent by entering a start time. This lets you supply multiple names for the same node. This can be useful while debugging applications, by indicating the node's current state.
- **Icon** and **More Icons** allow you to display the node as an icon.

Transactions Pane

The Transactions Pane displays higher-layer protocol events that consist of multiple packet transmissions. For example, a Zigbee broadcast is retransmitted by every node in the network. By analyzing packet headers, Network Analyzer determines which packets belong to the same transaction and groups them accordingly.



Transactions total:28 shown:37

	Time	Duratio	Summary	NWK Src	NWK Dest	P#	M#	E#	Error Statu	Warning St
	31.714233	0.000	The joining device then acts as a fi							
	31.714234	0.012	ZCL: IdentifyQuery	C2AD	FFFF	3				
	31.729314	0.013	ZCL: IdentifyQueryResponse	0000	C2AD	4				
	32.704209	0.014	IEEE Address Request	C2AD	0000	4				
	32.710934	0.012	IEEE Address Response	0000	C2AD	4				
	32.725048	0.015	Simple Description Request	C2AD	0000	4				
	32.732967	0.011	Simple Description Response	0000	C2AD	4				
	49.732008	0.000	After the bindings are made, the sw							
	49.732009	0.018	ZCL: Toggle	C2AD	0000	4				
	49.743780	0.014	ZCL: DefaultResponse	0000	C2AD	4				
	52.419532	0.018	ZCL: Toggle	C2AD	0000	4				
	52.431051	0.013	ZCL: DefaultResponse	0000	C2AD	4				

Typical 15.4 transactions include:

- **802.15.4 association**: Involves a request-response protocol that consists of at least 6 packet transmissions.
- **APS unicast**: Can contain the following events:
  - A MAC layer unicast packet and its MAC retries
  - Acknowledgements for each hop along the route
  - An end-to-end APS acknowledgement message, which itself consists of multiple MAC unicast packets
  - Multiple end-to-end APS retries
- **Zigbee route discovery**: Involves a broadcast route request followed by unicast route-reply packets across multiple hops.

In the case of Bluetooth Low Energy (and Bluetooth mesh), a “transaction” refers to an actual Bluetooth Low Energy transaction as defined in the core specification. This corresponds most of the time to a Bluetooth Low Energy procedure. Equally, the event pane displays the actual Bluetooth Low Energy events corresponding to the transaction or procedure. For more details, refer to the Bluetooth Core specification document.

Network Analyzer understands the protocol semantics for many transaction types. Therefore it can group multiple packets in real time to facilitate high-level analysis.

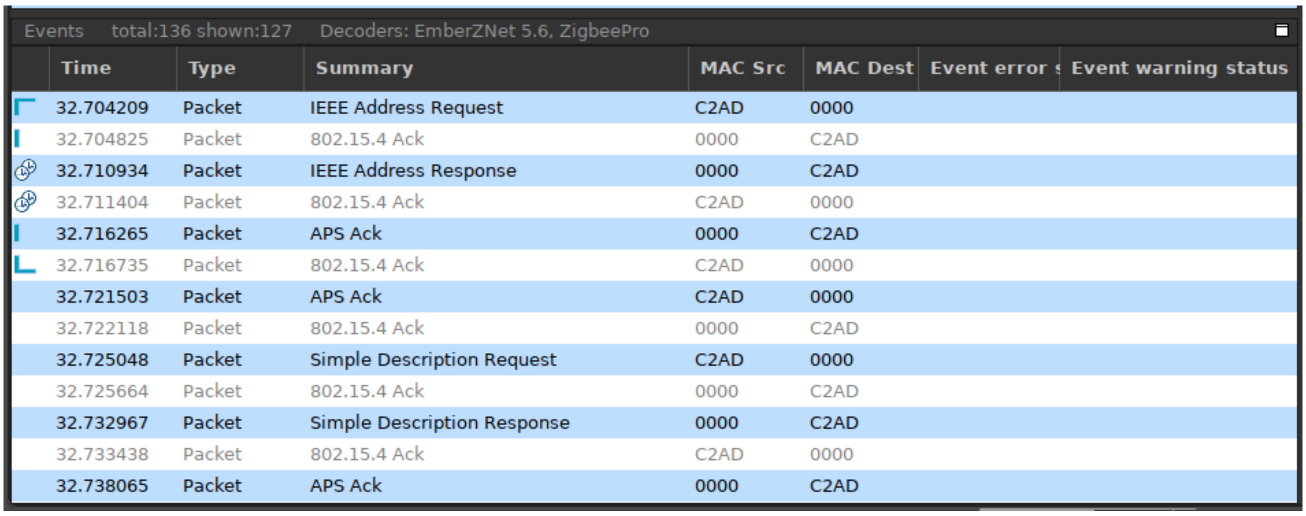
All transactions are listed in chronological order, using transaction start times. Each selection maps to one or more events in the Events Pane, which are marked accordingly. Clock icons indicate concurrent transactions with the current selection.

All transactions and their events are uniquely numbered. However, the transaction numbers may not be in sequence, and various factors will result in number gaps. For example, only top-level transactions and the lowest-level packets are shown. Intermediate transactions are not shown. Also, number gaps are likely to occur if filters are turned on.

When you click on a transaction, the information shown in the Event Detail Pane and the Hex Dump Pane corresponds to the first packet in the transaction. However, if filters are turned on, the first transaction might not be shown in the Event Pane. In that case, the event detail information in the transaction display will not be consistent with the first packet shown in the Event Pane. In fact, with a filter expression such as `show(transaction.summary != null, SELF)` only transactions are displayed and the Event Pane will be blank. In that case, click the transaction to see the first events in the transaction in the Event Detail and Hex Dump Panes.

Events Pane

The Events pane displays information about packets received by the current session. All events are displayed in chronological order.



Events total:136 shown:127 Decoders: EmberZNet 5.6, ZigbeePro

	Time	Type	Summary	MAC Src	MAC Dest	Event error	Event warning status
⌜	32.704209	Packet	IEEE Address Request	C2AD	0000		
▏	32.704825	Packet	802.15.4 Ack	0000	C2AD		
🕓	32.710934	Packet	IEEE Address Response	0000	C2AD		
🕓	32.711404	Packet	802.15.4 Ack	C2AD	0000		
▏	32.716265	Packet	APS Ack	0000	C2AD		
⌞	32.716735	Packet	802.15.4 Ack	C2AD	0000		
	32.721503	Packet	APS Ack	C2AD	0000		
	32.722118	Packet	802.15.4 Ack	0000	C2AD		
	32.725048	Packet	Simple Description Request	C2AD	0000		
	32.725664	Packet	802.15.4 Ack	0000	C2AD		
	32.732967	Packet	Simple Description Response	0000	C2AD		
	32.733438	Packet	802.15.4 Ack	C2AD	0000		
	32.738065	Packet	APS Ack	0000	C2AD		

Events that belong to the currently selected transaction in the Transactions pane are marked by one of the following icons:

Transaction icon	Meaning
⌜	Start of transaction
▏	Intermediate event
⌞	End of transaction
⊏	Single-event transaction

Clock icons mark unrelated events that are concurrent with the selected transaction in the Transactions pane.

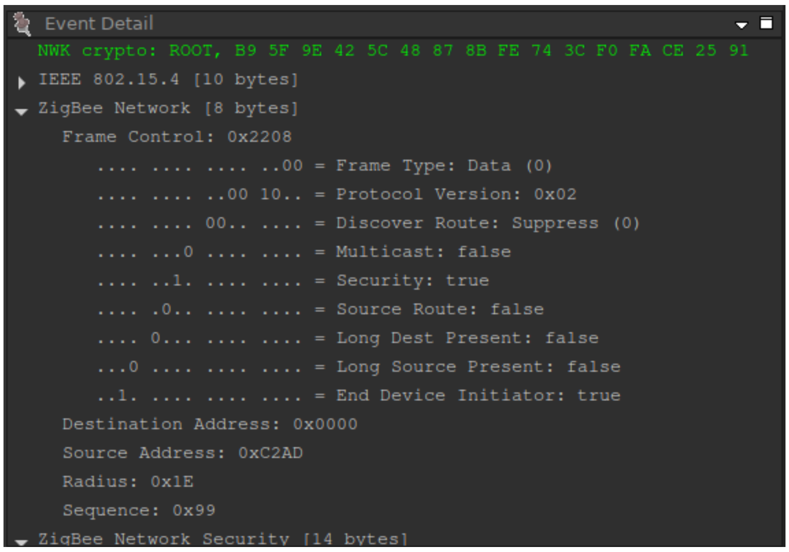
Event Detail Pane

The Event Detail pane displays the decoded contents of the event that is currently selected in the Events pane. The content of this pane varies according to the event type. If a transaction is selected on the Transactions pane, the Event Detail pane shows the details of the first event in the transaction.

Pane options include:

- **Expand Bitfield**: Shows the bitfields in an expanded mode, like Wireshark.
- **Use Fixed Fonts**: Can improve readability as information is presented aligned.

When capturing from multiple devices, Network Analyzer may capture the same packet as heard by several different sources. In order to reduce confusion, Network Analyzer automatically performs duplicate detection on all packets captured. If the transmission is captured over the backchannel, only the transmitted packet is kept. Otherwise the first receive packet is kept. All duplicate packets are dropped after extracting their RadioInfo data. Only the radio info frame for each duplicate packet is kept. The radio info for each individual instance of a packet captured by Network Analyzer is visible in the Radio Info View

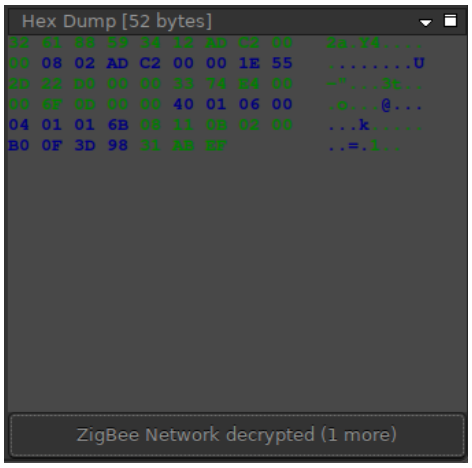


```
NWK crypto: ROOT, B9 5F 9E 42 5C 48 87 8B FE 74 3C F0 FA CE 25 91
IEEE 802.15.4 [10 bytes]
ZigBee Network [8 bytes]
  Frame Control: 0x2208
     .... .... .... ..00 = Frame Type: Data (0)
     .... .... ..00 10.. = Protocol Version: 0x02
     .... .... 00.. .... = Discover Route: Suppress (0)
     .... ...0 .... .... = Multicast: false
     .... ..1. .... .... = Security: true
     .... .0.. .... .... = Source Route: false
     .... 0... .... .... = Long Dest Present: false
     ...0 .... .... .... = Long Source Present: false
     ..1. .... .... .... = End Device Initiator: true
  Destination Address: 0x0000
  Source Address: 0xC2AD
  Radius: 0x1E
  Sequence: 0x99
ZigBee Network Security [14 bytes]
```

Pinning a field: The Event Detail pane has the ability to "pin" a field into view. When you double-click on a specific field, the Pin icon in the top left of the pane turns bright red, indicating that it is active. Now, as you move through events, this field is always visible when it is present in the currently selected packet. This is useful if you are interested in a specific field across multiple events in a trace file. In the above figure, the Zigbee Application Support Delivery Mode is "pinned" into view. The pin can be deactivated at any time by either double-clicking on the pinned field, or by clicking the Pin icon itself.

Hex Dump Pane

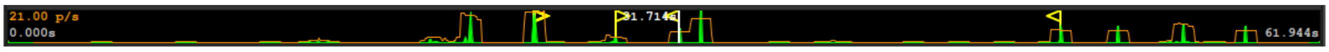
The Hex Dump pane displays data in raw bytes of a selected event in the Events pane. Clicking on bytes in the Hex Dump pane selects the corresponding field in the Event Detail pane. Alternatively, selecting a field or a frame in the Event Detail pane highlights the corresponding bytes in the Hex Dump pane. The pane shows multiple "layers", so if the packet is decrypted, the "raw" layer shows encrypted data, but the higher-level layers show this data progressively decrypted.



```
32 61 88 59 34 12 AD C2 00    2a.Y4....
00 08 02 AD C2 00 00 1E 55    ........U
2D 22 D0 00 00 33 74 E4 00    -"...3t..
00 6F 0D 00 00 40 01 06 00    .o...@...
04 01 01 6B 08 11 0B 02 00    ...k.....
B0 0F 3D 98 31 AB EF          ..=.1..
```

Timeline Bar

The Timeline bar displays the statistics of the traffic over time. The Timeline bar function on the Network Analyzer toolbar toggles the Timeline Bar on and off.



Available actions are:

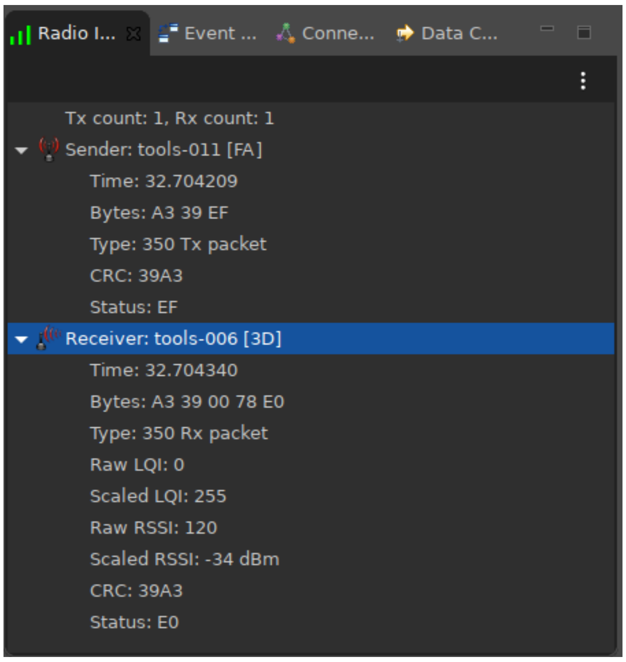
- Click on the Timeline bar to move the cursor to the event closest to the time selected.
- Click and drag on the Timeline bar to filter the display to only the time within the selected area.
- Right-click to display a timeline menu.

The Timeline bar shows bookmarks as yellow flags. You can click a bookmark to jump to it in the Transaction and Event panes. It shows red flags for errors, such as out-of-sequence problems.

Radio Info View

The Radio Info view is a helper view that shows the information from the radio of all the receivers in the network that have heard the currently selected event. It is available through the tabbed Views interface in the lower left of the default Network Analyzer perspective. If not, add it through **Window > Show View**.

The view displays in a tree all the information that has been gathered from the receiver nodes. Displayed information includes LQI value, CRC value, and the status bits that show several states of the radio.

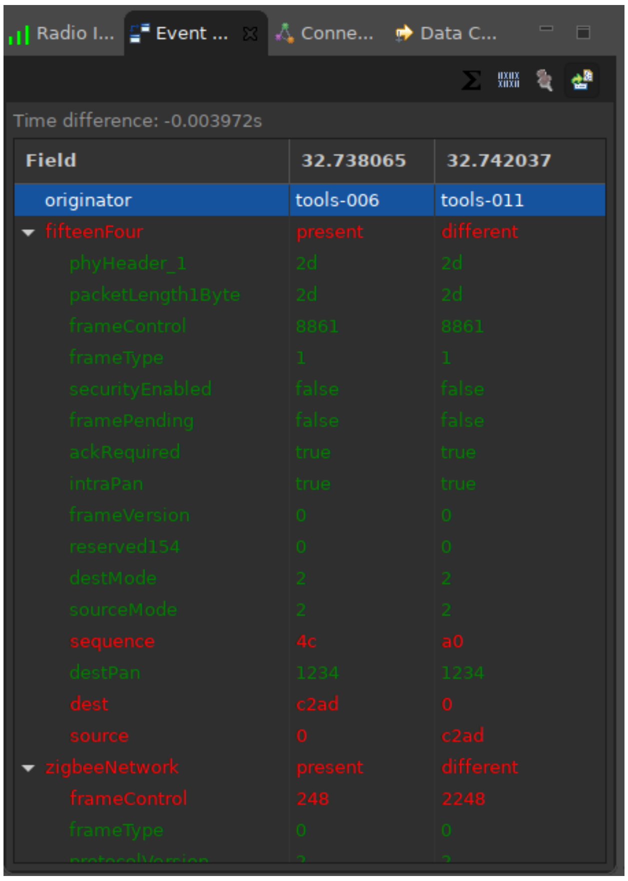


```
Tx count: 1, Rx count: 1
Sender: tools-011 [FA]
    Time: 32.704209
    Bytes: A3 39 EF
    Type: 350 Tx packet
    CRC: 39A3
    Status: EF
Receiver: tools-006 [3D]
    Time: 32.704340
    Bytes: A3 39 00 78 E0
    Type: 350 Rx packet
    Raw LQI: 0
    Scaled LQI: 255
    Raw RSSI: 120
    Scaled RSSI: -34 dBm
    CRC: 39A3
    Status: E0
```

The event that supplied radio information in the figure above was captured from both the sending and receiving nodes. This is possible because the trace that contains this event was created by capturing from both the sending and receiving nodes simultaneously using Network Analyzer's Perfect Trace capability. While the original events were merged into a single event by Network Analyzer's duplicate detection mechanism, the radio information was retained for each event and is shown in the Radio Info view with the time that the event was captured by Network Analyzer.

Event Difference View

Event Difference view is a helper view that displays the specific differences between two packets. It is available through the tabbed Views interface in the lower left of the default Network Analyzer perspective. If not, add it through **Window > Show View**.



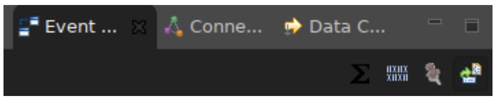
Time difference: -0.003972s

Field	32.738065	32.742037
originator	tools-006	tools-011
fifteenFour	present	different
phyHeader_1	2d	2d
packetLength1Byte	2d	2d
frameControl	8861	8861
frameType	1	1
securityEnabled	false	false
framePending	false	false
ackRequired	true	true
intraPan	true	true
frameVersion	0	0
reserved154	0	0
destMode	2	2
sourceMode	2	2
sequence	4c	a0
destPan	1234	1234
dest	c2ad	0
source	0	c2ad
zigbeeNetwork	present	different
frameControl	248	2248
frameType	0	0

Once the view is shown, it tracks the selected events. The view will by default show the difference between the last two events selected. If you select event 1, and then click on event 2, the view shows the difference between those two events. If later you select event 3, the view shows the differences between event 2 and 3.

Packet frames that do not have any differences are shown in green. Frames that contain differences are shown in red. Expand the frame to see which portions of the frames are different.

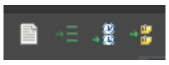
The menu at the top of the view supports additional functions.



- **Show traffic counts**: Opens a window showing the statistics for the events between, but not including, the two selected event.
- **Show byte differences**: Enables viewing of individual bytes in the view.
- **Pin last selected event**: Changes the way events are tracked. If this is enabled, then the first event for diffing stays the same, and only the second event changes. You can use this if you wish to always differentiate events against a certain static event, rather than always viewing last two selected events.
- **Include fields that are same**: Enables filtering out fields that are same in both events.

Editor Navigation Tools

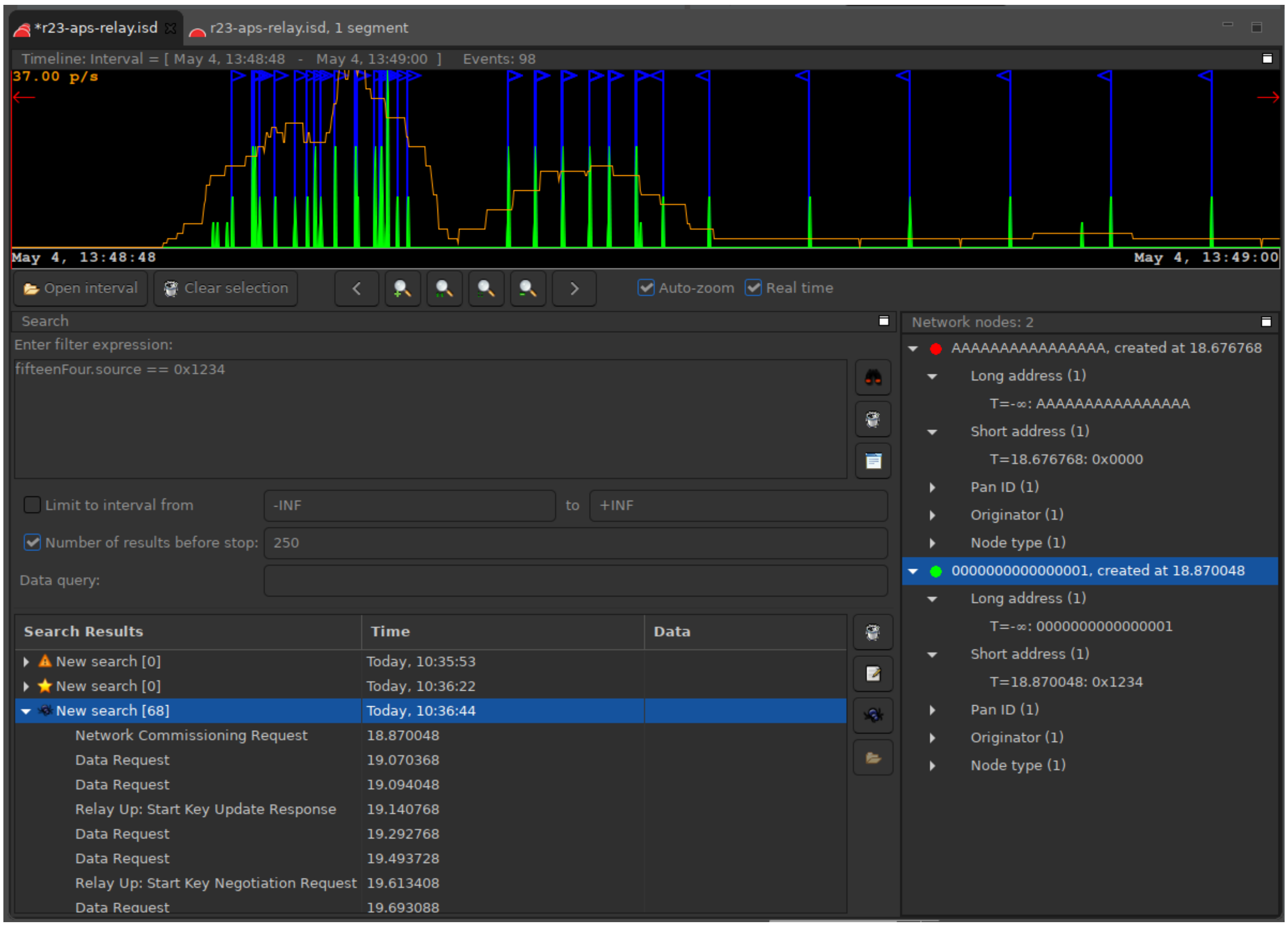
Some of the Network Analyzer Toolbar functions are specific to working in the Stream Editor.



- **Edit description of trace file**: Opens a simple dialog which allows you to view and edit overall description of the captured data. This is helpful if you need to pass on some information others for analyzing the contents of the trace file.
- **Go to Line**: Moves the cursor to the event or transaction having the specified event number. This is only enabled if the Stream preference "Show event numbers" is selected. To turn this feature on go to: Window > Preferences > Network Analyzer > Capture Configuration > Show event numbers.
- **Go to Time**: Moves the cursor to the transaction and event that match or immediately follow the specified time.
- **Go to Bookmark**: Moves the cursor to the selected bookmark. Assign bookmarks to events or transactions by right-clicking the event or transaction and selecting **Add Bookmark**.
- **Lock to Bottom**: Locks the cursor on the latest event during a live session. To remove the lock, select any event or transaction during a live session, which causes a view to scroll as the events are captured.

Large File Editor

If the file is larger than the size set in Preferences > Network Analyzer > Capture File Storage, Network Analyzer opens it in the Large File editor. The Large File Editor allows you to find and select a region of interest, which you can then open and analyze with the Stream Editor.



Enter filter expression:

```
fifteenFour.source == 0x1234
```

Limit to interval from -INF to +INF

Number of results before stop: 250

Data query:

Search Results	Time	Data
New search [0]	Today, 10:35:53	
New search [0]	Today, 10:36:22	
New search [68]	Today, 10:36:44	
Network Commissioning Request	18.870048	
Data Request	19.070368	
Data Request	19.094048	
Relay Up: Start Key Update Response	19.140768	
Data Request	19.292768	
Data Request	19.493728	
Relay Up: Start Key Negotiation Request	19.613408	
Data Request	19.693088	

- AAAAAAAAAAAAAAAA, created at 18.676768
  - Long address (1)
    - T=-∞: AAAAAAAAAAAAAAAA
  - Short address (1)
    - T=18.676768: 0x0000
  - Pan ID (1)
  - Originator (1)
  - Node type (1)
- 0000000000000001, created at 18.870048
  - Long address (1)
    - T=-∞: 0000000000000001
  - Short address (1)
    - T=18.870048: 0x1234
  - Pan ID (1)
  - Originator (1)
  - Node type (1)

The Large File Editor consists of three component panes:

- Large File Timeline: Shows a high-level view of a large file's traffic over time.
- Large File Search pane: Provides a mechanism for searching across very large files.
- Large File Network Nodes pane: Shows all of the network devices included in a large file.

Large File Timeline

The Large File Timeline shows a high-level view of a large file's traffic over time. It works similarly to the Stream Editor's timeline. In fact, the Large File and the Stream Editor timelines use the same widget.

Large File Timeline Segments: The entire set of events shown in the Large File Timeline is broken into segments. By default, each segment includes up to 5,000 events. Segments boundaries are shown in the Large File Timeline by horizontal grey lines.

Large File Timeline Time Markers: The Large File Timeline shows the actual time during which a file was captured. The capture start-time appears in the bottom left corner of the Timeline. The capture end-time appears in the bottom right corner.

Moving the cursor to any point on the timeline displays the time for that point.

Large File Intervals: A Large File Interval is a subset of an entire trace. You can create an interval by clicking and dragging. Click on the Large File Timeline at the desired start-point, and then drag the cursor along the timeline to the desired end-point.

The click and drag operation creates the interval you defined, and zooms the timeline view into that interval. To clear the interval, click the Clear Selection button in the toolbar under the timeline, or right-click on the timeline and select **Clear interval**.

Once you have created an interval, you can open that interval in the Stream Editor by clicking the Open Interval button, or right-clicking in the timeline and selecting **Open Interval**.

Timeline Flags: Search results and errors are displayed in the Large File Timeline by flags. Search results are displayed as a yellow flag. Errors are displayed as a red flag. When you move the mouse over the flag, the Timeline displays the Summary of the Event or Transaction that is associated with the flag.

Large File Search

Use the Large File Search mechanism to search for events across very large files. The entire filter language is supported in the Large File Search. For more information, see Filter Language.

To run a search in the Large File Editor, enter the filter expression into the Filter Expression text box and click the Start Search control. The search progress is shown in the Large File Timeline. Options allow you to limit the search within a time interval or to limit the number of search results. It is useful to limit the number of search results, as the system can become slow if the search expression matches an extremely large number of events.

Filter results are shown in the Search Results table. Search results are grouped into results trees and labeled with the time and date that the search was performed.

To view search result details, double-click an individual search result. This opens three segments in a Stream Editor: one before the selected event, one that contains the event, and one after the event.

Note: The expressions and search results are saved into the Network Analyzer file. Thus they will be seen by other users who open the same Network Analyzer file.

Controls to the right of the search results allow you to delete, tag, open, and assign decorative icons to the searches and search results.

Large File Network Nodes

The Large File Network Node pane shows all of the network devices included in a large file. The information provided about each node in a trace includes:

- EUI64 address
- Short address
- PAN ID
- Node type

Since each of these values is subject to change over time, the summary also includes the time at which each value was discovered.

Filtering Captured Data

Filtering Captured Data

By default, the Events pane displays all session events. You can build and apply filters that constrain Network Analyzer to show only events that are of interest. By filtering events, you can analyze results more efficiently.

Each capture session has its own filter settings. A filter can be used either to search for the next matching event or to display only the events that match the filter. In the latter mode, when you change a session's filters, Network Analyzer immediately refreshes the display. When you exit Network Analyzer, all session filters are cleared and must be reapplied when you restart. Network Analyzer provides two ways to edit filters:

- Filter Manager: Maintains a set of saved filters that you can review and edit. You can also add new filters. You specify any of the saved filters for display on the Filters menu, accessed through the **n saved filters** button on the filter bar, so that they are available for use in one or more sessions.
- Filter Bar: An editor that attaches to a given session, where you can enter one or more filter expressions on the fly. Network Analyzer discards filter bar expressions for all sessions when it exits.

Quick filters are available when you right-click on events and transactions. They provide an easy way to create common expressions for the filter bar.

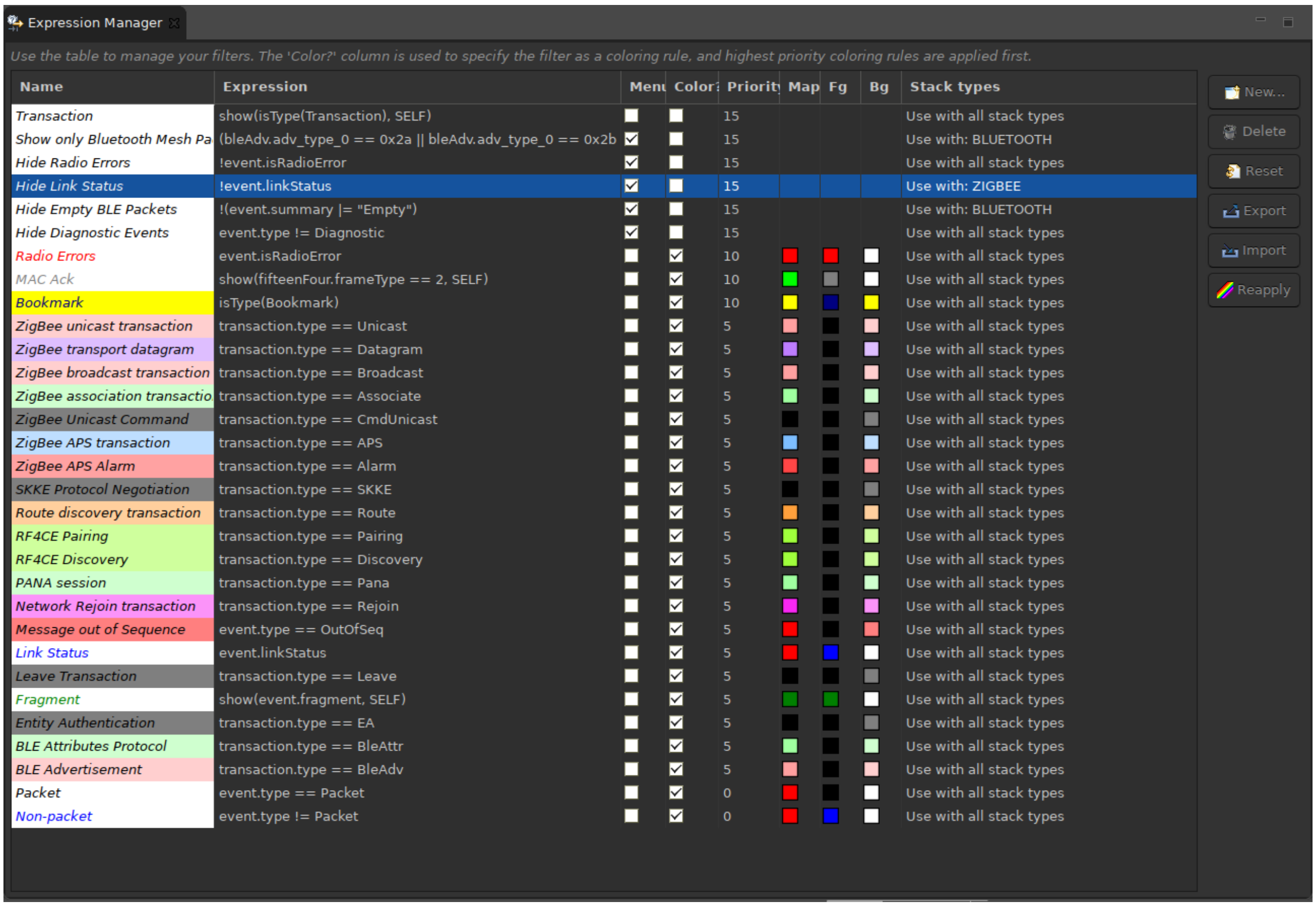
Filter Language is a powerful syntax used to create filters.

Filter Manager

The Filter Manager lets you:

- Use filter expressions to customize display of entries in the Events pane.
- Specify which filters appear in the Filters menu.

The Filter Manager is available through menu and toolbar selections as well as through Window > Show View (as Expression Manager).



Expression Manager

Use the table to manage your filters. The 'Color?' column is used to specify the filter as a coloring rule, and highest priority coloring rules are applied first.

Name	Expression	Menu	Color?	Priority	Map	Fg	Bg	Stack types
Transaction	show(isType(Transaction), SELF)	☐	☐	15				Use with all stack types
Show only Bluetooth Mesh Pa	(bleAdv.adv_type_0 == 0x2a || bleAdv.adv_type_0 == 0x2b	☑	☐	15				Use with: BLUETOOTH
Hide Radio Errors	!event.isRadioError	☑	☐	15				Use with all stack types
Hide Link Status	!event.linkStatus	☑	☐	15				Use with: ZIGBEE
Hide Empty BLE Packets	!(event.summary |= "Empty")	☑	☐	15				Use with: BLUETOOTH
Hide Diagnostic Events	event.type != Diagnostic	☑	☐	15				Use with all stack types
Radio Errors	event.isRadioError	☐	☑	10	🟥	🟥	⬜	Use with all stack types
MAC Ack	show(fifteenFour.frameType == 2, SELF)	☐	☑	10	🟩	▨	⬜	Use with all stack types
Bookmark	isType(Bookmark)	☐	☑	10	🟨	🟦	🟨	Use with all stack types
ZigBee unicast transaction	transaction.type == Unicast	☐	☑	5	🟥	⬛	🟥	Use with all stack types
ZigBee transport datagram	transaction.type == Datagram	☐	☑	5	🟪	⬛	🟪	Use with all stack types
ZigBee broadcast transaction	transaction.type == Broadcast	☐	☑	5	🟥	⬛	🟥	Use with all stack types
ZigBee association transactio	transaction.type == Associate	☐	☑	5	🟩	⬛	🟩	Use with all stack types
ZigBee Unicast Command	transaction.type == CmdUnicast	☐	☑	5	⬛	⬛	▨	Use with all stack types
ZigBee APS transaction	transaction.type == APS	☐	☑	5	🟦	⬛	🟦	Use with all stack types
ZigBee APS Alarm	transaction.type == Alarm	☐	☑	5	🟥	⬛	🟥	Use with all stack types
SKKE Protocol Negotiation	transaction.type == SKKE	☐	☑	5	⬛	⬛	▨	Use with all stack types
Route discovery transaction	transaction.type == Route	☐	☑	5	🟧	⬛	🟧	Use with all stack types
RF4CE Pairing	transaction.type == Pairing	☐	☑	5	🟩	⬛	🟩	Use with all stack types
RF4CE Discovery	transaction.type == Discovery	☐	☑	5	🟩	⬛	🟩	Use with all stack types
PANA session	transaction.type == Pana	☐	☑	5	🟩	⬛	🟩	Use with all stack types
Network Rejoin transaction	transaction.type == Rejoin	☐	☑	5	🟪	⬛	🟪	Use with all stack types
Message out of Sequence	event.type == OutOfSeq	☐	☑	5	🟥	⬛	🟥	Use with all stack types
Link Status	event.linkStatus	☐	☑	5	🟥	🟦	⬜	Use with all stack types
Leave Transaction	transaction.type == Leave	☐	☑	5	⬛	⬛	▨	Use with all stack types
Fragment	show(event.fragment, SELF)	☐	☑	5	🟩	🟩	⬜	Use with all stack types
Entity Authentication	transaction.type == EA	☐	☑	5	⬛	⬛	▨	Use with all stack types
BLE Attributes Protocol	transaction.type == BleAttr	☐	☑	5	🟩	⬛	🟩	Use with all stack types
BLE Advertisement	transaction.type == BleAdv	☐	☑	5	🟥	⬛	🟥	Use with all stack types
Packet	event.type == Packet	☐	☑	0	🟥	⬛	⬜	Use with all stack types
Non-packet	event.type != Packet	☐	☑	0	🟥	🟦	⬜	Use with all stack types

An older feature, the Expression Builder, is deprecated.

Maintaining Filters

To add a filter to the Filters menu, check the Menu checkbox. This makes the filter available to the current sessions.

To restore filters to installation settings, click **Reset**.

To rename a filter, click on the name field and edit.

You can also create new, delete, export and import filters.

Setting Filter Color Schemes

You can associate a color scheme with each filter. If an event evaluates as true for a given filter expression, Network Analyzer applies the filter's color scheme to that event. An event can be configured with a color scheme for two display levels:

- For the Map pane, the color used by the graphic representing the event type.
- For the Events pane, the foreground and background colors used by event type instances.

Note: Setting a color scheme on a filter expression does not specify whether to display events; it only determines how to display certain event types.

Note: If a filter's foreground color is the same as its background color you will not be able to read the text in the event and it will effectively disappear from view.

To set a filter color scheme, in the Filter Manager:

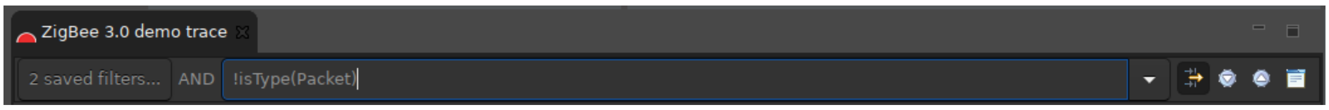
1. Next to the filter of interest, check the Color checkbox to enable the controls for Map (Map color), and Fg (Foreground color) and Bg (background color) for event instances.
2. Click the control you want to set to enable the color selector control.

3. Click the color selector control, set the filter's color, and click **OK**. If you have changed Foreground or Background colors you will see the change on the Name field.
4. For each filter, set the color scheme's Priority level by assigning a positive or negative integer value. If an event evaluates as true for multiple filters, Network Analyzer uses the color scheme with the highest precedence. In the case where there is no clear precedence, Network Analyzer randomly chooses one of the matching color schemes.
5. To refresh the Map pane and Events pane with the new color scheme, click **Reapply**.

A filter whose menu checkbox is selected is shown in the Filters menu. The order in which the filters are displayed in the Filters menu is determined by their order in the filter manager table.

Filter Bar

A session's Filter Bar provides the same level of functionality as the Filter Manager for building expressions. Any filter expression that appears in a session's Filter Bar is combined with the saved filter expressions that are already in effect for that session.



The Filter bar is displayed by default, but can be toggled on and off using the **Show Filter Bar** menu control.

You can compose a Filter Bar expression in the following ways:

1. Enter the expression directly in the filter edit field, and press Enter.
2. Create a Quick filter.

To apply a Filter Bar filter, enter an expression into the filter edit field and press Enter.

To find an event that matches the filter, click the Find icons on the Filter Bar.

To remove a filter, select it and press Enter.

Note: A history of filter expressions is maintained in the drop-down list.

Quick Filters

Network Analyzer provides several filters that are available from the Transactions pane, Events pane, Event Detail pane, and Hex Dump pane.

To access a quick filter, right-click an item and choose a quick filter pop-up menu option.

Hide or Show Events/Transactions

Access these filters by right-clicking an event (Events pane) or transaction (Transactions pane) and selecting a pop-up menu option to hide or show specific information. The filter options that are available depend on the selected transaction or event. You can specify to hide all events/transactions of the selected type, or to show only that type. In addition if you select an event of type APITrace, the pop-up menu displays two filter options:

- Hide type: APITrace
- Show only type: APITrace

Further, if you select an event such as a neighbor exchange that has a source and/or destination address, the pop-up menu also contains these two filter options:

- Show only destination: short-ID
- Show only source: short-ID

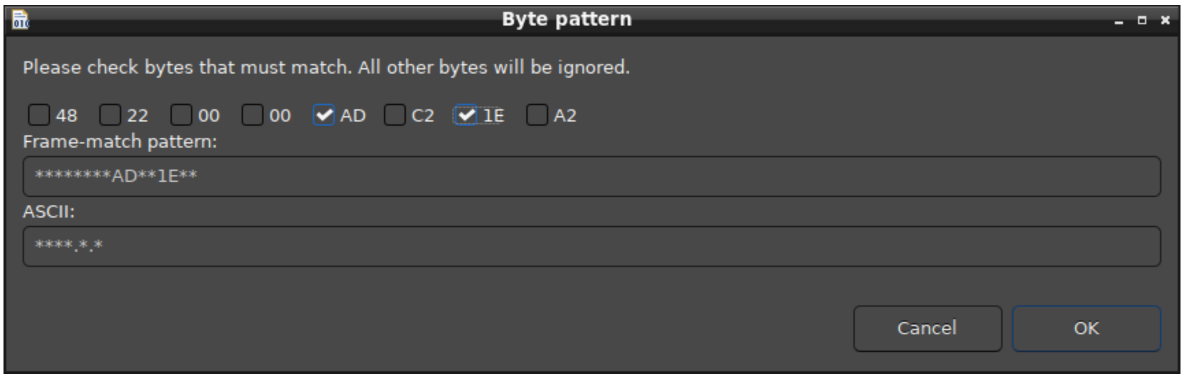
In all cases, Network Analyzer enters the corresponding filter expression in the session's Filter Bar. This can help you to understand the filter language. For example, if you specify to show only route discovery transactions, this expression is set in the filter bar: `isType(Route)`

Frame Byte Pattern Filtering

A frame pattern filter matches a specific byte-array pattern. For example, you could filter for packets in a payload whose frame has the third byte equal to 0x33. (Many more complex combinations are possible.)

To create a frame pattern filter:

1. Right-click a frame in the Event Detail pane or Hex Dump pane.
2. Select Filter by frame pattern from the pop-up menu.
3. In the Byte Pattern dialog, check the byte pattern match desired and click OK. The filter is added to the Filter Bar for this session.



Please check bytes that must match. All other bytes will be ignored.

☐ 48 ☐ 22 ☐ 00 ☐ 00 ✓ AD ☐ C2 ✓ 1E ☐ A2

Frame-match pattern:

```
********AD**1E**
```

ASCII:

```
****.*.*
```

Filter Language

Filter language enables you to construct logical expressions, based on decoded fields in events. The following are some examples:

- `fifteenFour.sequence == 0x52` : Matches events where 15.4 sequence number equals hex 0x52.
- `fifteenFour.ackRequired == true && fifteenFour.source == 0x035f` : Matches events where 15.4 ack required flag is set, and source shortId is 0x035f.
- `isPresent(zigbeeSecurity.frameCounter)` : Matches events that contain the Zigbee security frame, and the frameCounter field within it.
- `event.summary | "string"` : Matches events where a string is a substring of the summary.
- `isType(Packet)` : Matches events that are packets.
- `frameMatch(fifteenFour,"**88**EF/**********")` : Matches events where 15.4 frame contains second byte equal to 0x88 and fourth byte equal to 0xEF.

A good way to learn the filter language is by first using the **Add to filter** context menu option in the Event Detail Pane. This option will add a filter expression for the chosen field.

You can use most standard logical operators (&&, ||) and standard comparison operators (==, !=, |, <, >, <=, >=, etc.) in filter expressions.

Event and Transaction Filter Extensions

In addition to filtering on decoded packet fields, you can filter on several other Event and Transaction values.

Event Extensions:

- event.summary: A String value of the summary shown in the Event Pane.
  - Example: event.summary == "APS Ack"
- event.linkStatus: True if the packet is a Link Status packet.

- Example: event.linkStatus == true
- event.ack: True if the packet is an 802.15.4 ack.
  - Example: event.ack == true
- event.time: The time that the event was transmitted (tx) or received (rx).
  - Example: event.time >= 75.78
- event.originator: The adapter that saw and reported the event.
  - Example: event.originator == "ewb-unit04"
- event.status: The event status, listed in the righthand column of the event status window
  - Example: event.status == "ZCL: ReportEventStatus"
- event.type: The type of event, shown in the Type column of the Event and Transaction Panes.
  - Example: event.type == "Packet"
- event.corrupt: The event corruption string, empty if event is not corrupt
  - Example: event.corrupt < "crypt"

Transaction Extensions:

- transaction.summary: Filters on the transaction summary field shown in the Summary column of the Transaction Pane.
  - Example: transaction.summary == "ZCL: LoadControlEvent"
- transaction.packetCount: Filters on the number of packets in the transaction shown in the P# column of the Transaction Pane
  - Example: transaction.packetCount == 4
- transaction.macRetries: Filters on the number of MAC retries in the transaction shown in the M# column in the Transaction Pane.
  - Example: transaction.macRetries == 2
- transaction.endToEndRetries: Filters on the number of end to end retries shown in the E# column in the Transaction Pane.
  - Example: transaction.endToEndRetries == 3
- transaction.status: Filters on the status of the transaction shown in the Status column of the Transaction Pane.
  - Example: transaction.status == "CRC failed"
- transaction.dest: Filters on the network destination of the transaction shown in the NWK Dest column of the Transaction Pane.
  - Example: transaction.dest == 0x05c7
- transaction.source: Filters on the network source of the transaction shown in the NWK Src column of the Transaction Pane.
  - Example: transaction.source == 0x0000

How Network Analyzer Applies a Filter

When Network Analyzer captures an over-the-air message, it runs the message through a processing stream. The processing stream is made up primarily of Decoders and Groupers.

Decoders: Decoders are responsible for making sense of the message based on its format so that it may be displayed to the user. Each over-the-air message captured becomes a single Event of type Packet. This Event is displayed to the user in the Event view.

Groupers: The groupers are responsible for making sense of a series of packets and grouping them into a hierarchy under a single Transaction. The Transaction is displayed in the Transaction view.

Using `show(expression, SELF|PARENT|CHILD|SIBLING)`

Events exist within a hierarchical structure where Transactions represent the top of the hierarchy and Events are at the bottom. The hierarchical nature of trace data creates something of a problem for filtering. In most cases, you wish to see Transactions associated with filter-matching Events, and vice versa.

For instance, if you use a filter like: "transaction.summary == Association", you probably do not want to see only the transactions in the Transaction Pane. You probably also want to see the events contained within the Association displayed in the Event Pane.

You can solve this problem by using the optional `show(expr, args)` syntax in your filters.

The show syntax allows you to explicitly indicate the conditions under which an event or transaction should match your filter. The arguments for the show syntax are as follows:

SELF - The Event or Transaction matches if it contains data that matches the expression provided.

PARENT - A Transaction should be shown if any one of its child Events matches the filter.

CHILD - An Event should be shown if the Transaction to which it belongs matches the filter.

SIBLING - An Event matches if its PARENT transaction contains another Event which itself matches the filter.

Filter display defaults: Filter expressions that do not explicitly contain the optional `show(expr, args)` syntax are implemented as though they contain one of two default syntaxes. Which default syntax is used depends on where the filter is executed, in the Stream Editor or in the Large File Editor.

- Stream Editor default: `show(expression, SELF|PARENT|CHILD)`
  By default, filter expressions that do not explicitly contain the show(expr, args) syntax are implemented as though they were wrapped in the following syntax: `show(expression, SELF|PARENT|CHILD)` The SELF|PARENT|CHILD arguments provide what Silicon Labs believes a user expects to see when a filter is run in the Stream Editor. The filter display includes the events and transactions that match the filter itself. If an Event matches the filter, you also see an associated Transaction (PARENT) regardless of whether that Transaction matches the filter. Likewise, if a Transaction matches the filter, you also see its associated Events (CHILD) regardless of whether those Events match the filter.
- Large File Editor default: `show(expression, SELF)`
  By default, the search mechanism in the Large File Editor returns only those Events and Transactions which themselves match the filter expression provided. This behavior makes it very easy to run a filter to search for all the Transactions with a given summary without having the search results bogged down with hits for their associated Events.

Expression Validation

Lexical validation - When you enter a filter expression, the filter engine validates whether the expression is lexically correct. If an expression is not lexically correct, Network Analyzer gives you an error message with a suggestion about where there may be a problem in the expression.

Event Key validation - When you enter a lexically correct expression, Network Analyzer also runs an event key validation. It checks that any identifier provided represents a real entity within an Event or Transaction. If the filter engine is not able to find any associated data for an event key within the expression, Network Analyzer will warn you that you are using an unverified identifier.

For example, in the expression

`fifteenFour.dest == 0xffff`

fifteenFour.dest is a verifiable event key in that Network Analyzer knows that it represents real data in an event.

Here are two examples to illustrate the validation of expressions in Network Analyzer. Consider this expression:

`foo == bar`

The filter mechanism has no way of knowing what foo and bar represent, or that they even represent any type of data within an Event or Transaction. While this expression is lexically correct, Network Analyzer will warn the user that foo and bar could not be verified, and that the expression may provide unexpected results. In fact, this expression will not show anything, since foo may very well equal bar, but the filter engine has no way of knowing that.

Consider also this expression:

`foo == foo`

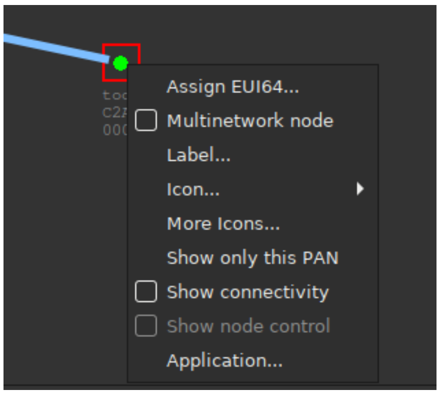
This expression also displays a warning. However, when run, it will return ALL events, because while Network Analyzer does not know what foo is, it knows it definitely equals foo.

Special identifiers - Several special identifiers are not mapped to an event key.

- payload.xxx, which evaluates into the payload bytes for a given layer xxx, for example payload.raw or payload.tcp_stream. You can form expressions like `payload.raw == {001122aabbcc}` to match payloads
- flag.xxx, which evaluates into the value of the event flag for a given event, for example: `flag.neighbor_exchange` or `flag.fragment`.

Multinetwork Considerations

If your application uses nodes operating on multiple networks, this information can be reflected when reviewing capture sessions. Currently, however, Network Analyzer cannot auto-detect multinetwork nodes. At the onset, unless all traditional, conceptual nodes constituting the multinetwork node are assigned EUI64s, each conceptual node shows up as a separate node in the Map editor pane. Each such node must be assigned the same EUI64 by right-clicking on the node in the Map editor pane and selecting **Assign EUI64...** A dialog pops up that facilitates the assignment, and the **Multinetwork EUI64** checkbox must be checked to inform Network Analyzer that the node is indeed a multinetwork node. This sequence is illustrated in the following figure.



Once all the constituent conceptual nodes have been assigned the same EUI64, Network Analyzer coalesces them into one node in the Map editor pane. Alternatively, if the all the conceptual nodes know the EUI64 or all have been coalesced but Network Analyzer has not been informed that the node is multinetwork, the **Multinetwork node** menu item in the figure above can be toggled. Once a node is known to Network Analyzer to be multinetwork, it is indicated as such by being colored magenta in the Map editor pane.

Custom Decoders

Custom decoders

Network Analyzer provides functionality to create custom decoders. This section provides instructions on how to use Lua, a language embedded inside Network Analyzer, to develop custom payload decoders for Simplicity Studio.

Procedure

There are five main steps to leverage this functionality.

Step 1: Writing a decoder

All Lua decoders must implement the following six functions.

- **name():** Returns a string that will be used as the name of your decoder.
- **filterName():** Returns a string that will serve as the filterName that can be used to filter the packets decoded by this decoder later on.
- **description():** Returns a string that will act as the description of the frame in Network Analyzer.
- **enabled():** Returns a boolean value that corresponds to whether the decoder should be active by default when first loaded/when defaults are reapplied.
- **accept(event, packet):** Decides which packets your decoder is interested in. It has two objects, of type Event and Packet. The packet object can be used to figure out the raw bytes of the entire packet or just the bytes of the payload. The event object can be used to identify and read information from prior frames via `get()` and `contain()` functions.
- **decode(event, fieldContext, formatter):** Handles the actual decoding. It has two objects, of type Event and FieldContext. The decode process functions by using the `append()`` and `decode()`` functions to create and register different fields of interest. The **formatter** field is optional and is used for formatting the fields of your decoder.

More detailed decoder examples can be found under Example Decoders.

Step 2: Loading into Simplicity Studio

After the decoder is ready to be tested, load it into Simplicity Studio.

1. Open Simplicity Studio
2. Go to **Window -> Preferences -> Network Analyzer -> Decoding**
3. Click **Add**, select the LUA file that corresponds to our decoder, and then click **Open**.

You should now see the decoder under the table of Decoders.

Step 3: Editing the Decoder

Once the decoder is loaded, you can now edit it in Simplicity Studio. Click **Edit** under the decoding screen to open a text editor where you can modify the file. Note that if you save your decoder while there are syntax errors, then you may need to re-add your decoder.

Step 4: Testing the Decoder

Once the decoder is loaded into Studio, you can test it against specific packets by doing the following:

1. From the **Decoding** page, click decoder in the table of custom decoders.
2. Click **Test**.
3. Set the desired options (for example, Input Format or Payload vs. Full Over The Air)
4. Enter the raw payload or over-the-air packet bytes in the input textbox.

NOTE: The OUT box does not work and will be fixed in a future release.

You see how this works by looking at the example decoders and trying the example packets that are listed. You can also have additional inline debug information printed to screen with print() statements within the decoder.

Step 5: Use with Network Captures

If the decoder is enabled (as indicated by the checkbox next to its entry in the **Preference -> Decoding** page), then you can decode messages using this decoder in Network Analyzer.

Note: To have logging during longer network captures, go to **Preference -> Decoding** and check **Enable Debug Logging**. This will write to file any calls to the following function:

```
log("message")
```

See Example decoders to see how this is done.

Field Formatters

You can enhance the decoded information by implementing custom field formatters. Currently, Network Analyzer supports user-written functions that take in an integer and outputs a string that can be used as a description. Note that these formatters can sometimes get in null information, so these functions should be able to deal with null.

Example of a formatter:

```
function decoder.versionName(version)
  local versions = {
    [0] = "Version 1",
    [1] = "Version 2",
    [2] = "Version 3",
    [3] = "Version 4",
  }

  return versions[version] or "Unknown version"
end
```

In order to use this functionality, include the following line at the top of your custom decoder:

```
decoder.fieldFormatters = { "versionName" }
```

Each of the strings within this array should correspond to the names of the function that you are using as a formatter.

To use these formatters in your decoder, add a third argument to the decode functions that corresponds to a map of formatters that are indexed by the strings given in fieldFormatter.

For example, in function decode(event, fieldcontext, formatter), formatter is a map, indexed by function name, that corresponds to the formatter.

Once this is done, you can use a formatter by adding a third argument to many of fieldContext's append and decode functions.

```
fc.append("versionName", 1, formatter["versionName"]);
```

See Example decoders to see how this is done.

The list of currently supported append/decode functions that allow for a field formatter is provided in the next sections.

Event Object API

`void setStderr(final String err)` - Sets the stderr of the decoder.

- Parameters: `err` —

`void setStdout(final String out)` - Sets the stdout of the decoder.

- **Parameters:** `out` —

`Object get(final String eventKey, final String subkey)` - Wrapper function to access the value associated in the decorated map. Mainly for use with custom user payload decoders.

- **Parameters:**
  - `EventKey` — A string that must corresponds to the name of a specific frame. See EventKey Documentation.
  - `Subkey` — A string that corresponds to field names. See FieldContext API to see how to generate these fields.
- **Returns:** Object Null if it does not exist.

`int getInt(final String eventKey, final String subkey)` - Wrapper function to access the value associated in the decorated map. Mainly for use with custom user payload decoders.

- **Parameters:**
  - `EventKey` — A string that must corresponds to the name of a specific frame. See EventKey Documentation.
  - `Subkey` — A string that corresponds to field names. See FieldContext API to see how to generate these fields.
- **Returns:** int Null if it does not exist.

`boolean getBoolean(final String eventKey, final String subkey)` - Wrapper function to access the value associated in the decorated map. Mainly for use w/ custom user payload decoders.

- **Parameters:**
  - `EventKey` — A string that must corresponds to the name of a specific frame. See EventKey Documentation.
  - `Subkey` — A string that corresponds to field names. See FieldContext API to see how to generate these fields.
- **Returns:** boolean Null if it does not exist.

`boolean containsSubkey(final String eventKey, final String subkey` - Wrapper function so that users can figure out if a frame/subkey combination is known by the decorated map.

- **Parameters:**
  - `EventKey` — A string that must corresponds to the name of a specific frame. See EventKey Documentation.
  - `Subkey` — A string that corresponds to field names. See FieldContext API to see how to generate these fields.
- **Returns:** boolean True if the subkey's corresponding field is found under the frame given by EventKey

`boolean containsKey(final String eventKey)` - Wrapper function so that users can figure out if an frame/eventKey is known by the the event.

- **Parameters:** `Subkey` — A string that corresponds to field names. See FieldContext API to see how to generate these fields.
- **Returns:** boolean True if the event encountered or recognizes the event key in question, false otherwise.

Packet Object API

`byte[] currentLayerBytes()` - Returns raw byte array. Decryptors need this.

`int currentOffset()` - Returns the current offset where the decoders need to pick up.

`void setSummary(final String summary)` - Sets the summary string.

- **Parameters:** `summary` —

`String summary()` - Returns the packet summary string.

`void setCorruption(final String message)` - Backwards compatibility. JC interface needs this.

- **Parameters:** `message` —

`void skip(final int skipBytes)` - Skips bytes. Remove after packet decoders are cleaned up.

`void setEndOffset(final int end)`

- **Parameters:** `end` —

`void setOffset(final int offset)`

- **Parameters:** `offset` —

`int micCount()` - Returns number of all mics

`int endOffset()` - Returns the end of "over the air" packet.

- **Returns:** end of over the air packet

`int flagBits()`

`void setFlagBits(final int bits)`

- **Parameters:** `bits` —

`boolean isFragment()` - Returns true if this packet is a fragment.

- **Parameters:** `` —
- **Returns:** boolean

FieldContext Object API

`void append(String name, int length)` - Appends the number of bytes as a field. This method is more efficient than decode, so if you do not need the value of the field, use this.

Example Input and Output:

0x01234567 -> 0x01234567

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.

`void appendBytes(String name, int length)` - Appends the number of bytes as given by length as a byte field. This method is more efficient than decode, so if you do not needs the value of the field, use this.

Example Input and Output:

0x01234567 -> 01 23 45 67

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.

`void appendEui64(String name)` - Appends the next 8 bytes as the EUID of the sender. This method is more efficient than decode, so if you do not needs the value of the field, use this. The format will be Little Endian here.

Example Input and Output:

0x89ABCDEF01234567 -> 67452301EFCAB89

- **Parameters:** `name` — The name of the field that is to be registered.

`void appendFloat(String name, int length)` - Appends the field as a float. This method is more efficient than decode, so if you do not needs the value of the field, use this.

Example Input and Output:

0x00000001x -> 1.4 E-45

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.

`void appendFloatLE(String name, int length)` - Appends the field as a float in little endian order. This method is more efficient than decode, so if you do not needs the value of the field, use this.

Example Input and Output:

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.
    0x01000001x -> 1.4 E-45

`void appendRemainingBytesLE(String name)` - Appends the remaining bytes in the packet in little endian order. This method is more efficient than decode, so if you do not needs the value of the field, use this.

- **Parameters:** `name` — The name that will be used to refer to the field containing the remaining bytes

`int decode(String name, int length)` - Decodes the field as an int and returns value.

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.
- **Returns:** int The value that was decoded from the payload and copied into the field called name.

`int decodeLE(String name, int length)` - Decodes the field as an int in little endian format and returns value.

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `length` — An integer number of bytes to read from the payload and save as name.
- **Returns:** int The value that was decoded from the payload and copied into the field called name.

`String decodeString(String name, int length)` - Decodes length bytes as a string.

Example Output:

0x68656c6c6f20776f726c64 -> 'hello world'

- **Parameters:**
  - `name` — The name of the field the string will be registered as.
  - `length` — The number of bytes to be decoded as a string.

`void appendStringWithLength(String name)` - Appends a string that first contains a length byte and then the remainder as bytes.

- **Returns:** String

`boolean decodeBit(String name)` - Decodes and returns the value of the next bit in the payload.

- **Parameters:** `name` — The name of the field that is to be registered.
- **Returns:** boolean The value of the decoded bit.

`int decodeBits(String name, int numOfBits)` - Decodes bits as field name. Identical to bits(name.name(), numOfBits);

- **Parameters:**
  - `name` — The name of the field that is to be registered.
  - `numOfBits` — The number of bits that are to be decoded jointly.
- **Returns:** int The value of the decoded bits as an integer.

`void bit(String name)` - This method heavily depends on the last field being decoded. So it's meaningful to be used kinda only in context of last field. You would do this: fc.decode(flags); fc.bit(flag1); fc.bit(flag2); fc.bit(flag3);

As each field take length from flags and does internal bitshifting to calculate masks of flag1 flag2.

- **Parameters:** `name` — The name of the field that the bit is to be decoded as.

`void bits(String name, int numOfBits)` - Appends numOfBits within a bitfield, labeling then name.

- **Parameters:**
  - `name` — The name of the field that the bit is to be decoded as.
  - `numOfBits` — The number of bits that are to be decoded as a group.

`void namelessBitField(int nBytes)` - Declares the start of the nameless bit field.

- **Parameters:** `nBytes` — The number of bytes to be treated as a nameless field.

`void namelessBitFieldLE(int nBytes)` - Declares the start of a nameless bit field.

- **Parameters:** `nBytes` — The number of bytes to be treated as a nameless field.

`void skipBits(int numOfBits)` - Skips given number of bits in decoding.

- **Parameters:** `numOfBits` — The number of bits to be skipped.

`void skipBytes(int n)` - Skips n bytes without decoding them.

`void appendString(String name, int length)` - Appends length bytes as a string. This method is more efficient than decode, so if you do not needs the value of the field, use this.

Example Output:

0x68656c6c6f20776f726c64 -> 'hello world'

- **Parameters:**
  - `name` — The name of the field the string will be registered as.
  - `length` — The number of bytes to be decoded as a string.

`void appendSummary(String append)` - Appends a message to describe the packet in network analyzer. This can be useful for describing what kind of payload the packet is.

- **Parameters:** `append` — A string that will be appended to the packet's summary.

`void setSummary(String summary)` - Sets the summary for what kind of event the payload could be. This will be what is read under the the network analyzer.

- **Parameters:** `summary` — The summary that will be read under network analyzer.

`void setPrefix(String prefix)` - Sets a prefix for the payload.

A prefix will be prepended to the fields that are decoded by this decoder.

- **Parameters:** `prefix` — A string that will be prepended to the names of fields for this decoder.

`void applyPrefixToSummary()` - Sets the summary to be the prefix associated with the protocol. See fc.setPrefix()

`String codeNameSuffix()` - Returns the codeNameSuffix, the suffix that is appeneded to each field.

- **Returns:** String The codeNameSuffix currently set for the decoder.

** `int currentOffset()**` - Returns the current offset for next element to be decoded in the payload.

- **Returns:** int The current offset within the packet.

`boolean hasRemainingBytes()` - Returns true if there are remaining bytes left in the payload that have not been updated.

- **Returns:** boolean True if there are remaining bytes left.

`int remainingBytes()` - Returns the number of bytes left before the end of the payload

- **Returns:** int The remaining bytes that have been not been processed within the payload.

`void setAppendLengthsToFieldCodes(boolean flag)` - Toggles the mode where the lengths of the fields are appended to the names.

If you are decoding field "x" with length, 2, then the actual name of the field will be "x_2" if the length appending mode is true, and only "x" if its false.

- **Parameters:** `flag` — Setting this true will append lengths to the end of fields as stated above.

`void setCodeNameSuffix(String codeNameSuffix)` - Sets the codeNameSuffix

- **Parameters:** `codeNameSuffix` — A string that will be appended to the ends of field names.

EventKey / Subkey Fields

To see the full list of EventKeys that are currently available, go to **Window -> Preferences -> Network Analyzer -> Decoding -> Frames and Fields**.

This window displays the valid frames and associated fields for your selected stack and profile. Note that the string you should use as the EventKey is the text that is located within brackets (e.g., if there is line "IEEE 802.15.4 [fifteenFour]", then you should refer to that frame using "fifteenFour").

If you expand the tree under each frame, then you will be able to see all the valid fields, and the text string that can be used as a subkey to search for them.

Example Decoders

Custom Decoder Skeleton

```
-- Custom Decoder Skeleton (custom-decoder-skeleton.lua)
-- Users should use this as a framework to build their decoders.

-- Create a table to hold all the functions
local decoder = {}

-- This will let you set what the name of the frame should be when viewed in the network analyzer.
decoder.name = "Custom Decoder Skeleton"

-- This will determine what the decoder can be filtered by in the network analyzer when viewing all packets.
decoder.filterName = "SampleDecoder"

-- This is a function that is describing what the frame should have as its description
-- when you are viewing specific information in the network analyzer.
decoder.description = "This is an example of how you can describe a frame"

-- This function will let you set whether the decode should be enabled by default when
-- loading it for the first time or when resetting the settings to default.
decoder.enabled = true

-- This function will do the actual decoding of the payload.
-- You should primarily use the functions as defined for the FieldContext object fc to do this.
-- The functions for event can be used to process information from earlier frames.
function decoder.decode(event, fieldContext)
   -- Your decoding logic here
end

-- This function determines if an event is relevant to this decoder.
-- Note that this should be specific to the payload; if it is too generic,
-- then event payloads belonging to other decoders may be mistakenly read and decoded by this decoder.
function decoder.accept(event, packet)
   return false
end

return decoder
```

Field Formatter Example

```
-- Example Decoder for using a field formatter (field-formatter-example.lua)
-- Create a table to hold all the functions
local decoder = {}

decoder.name = "Field Formatter Example"
decoder.filterName = "fieldFormatterExample"
decoder.description = "This is an example of how you can log debug information and test your decoder"
decoder.enabled = true

-- The list of field formatters
local fieldFormatters = { "decimal", "enuming", "bitFields", "bitDescription" }

-- This function will do the actual decoding of the payload.
-- Note that the function call when using custom decoding requires
-- the addition of the third parameter formatter.
function decoder.decode(event, fc, formatter)
  fc:setAppendLengthsToFieldCodes(true)
  fc:append("append", 4, formatter["decimal"])
  fc:decodeLE("appendLe", 4, formatter["enuming"])
  fc:namelessBitField(4)
  fc:bits("decodeBits", 3, formatter["bitFields"])
  fc:bit("decodedBit", formatter["bitDescription"])
end

-- This function determines if an event is relevant to this decoder.
function decoder.accept(event, packet)
  return true
end

-- These are three examples of field formatters.
-- Note that these functions MUST be able to handle null values.
function decimal(value)
  return tostring(value)
end

function enuming(value)
  if value == 0 then
     return "ZERO"
  elseif value == 1 then
     return "ONE"
  elseif value == 2 then
     return "TWO"
  else
     return "NOT ONE OR ZERO"
  end
end

function bitFields(value)
  if value == tonumber("0001", 2) then
     return "1"
  elseif value == tonumber("0010", 2) then
     return "2"
  elseif value == tonumber("0100", 2) then
     return "3"
  else
     return "invalid"
  end
end

function bitDescription(value)
  return (value and "1" or "0")
end

return decoder
```

Payload Decoder Logging Example

```
-- Sample Logging Examples for testing and debugging (payload-decoder-logging-example.lua)
-- Users should see this for how they may be able to do some initial logging.

-- Create a table to hold all the functions
local decoder = {}

decoder.name = "Payload Decoder Logging Example"
decoder.filterName = "debugExampleDecoder"
decoder.description = "This is an example of how you can log debug information and test your decoder"
decoder.enabled = true

function decoder.decode(event, fieldContext)
  -- This will show up under the test console, but not the log file.
  print("Hello world!")

  -- This will show up under the log file, but not the test console.
  local a = fieldContext:decode("var1", 2)
  log(a)
end

function decoder.accept(event, packet)
  return false
end

return decoder
```

Sensor Sink Example

```
-- Example Packet w/ Length byte for Full Over the Air (sensor-sink.lua)
--
384188E8FF01FFFF00000803111100000AD5C12D0610000084E10A00006F0D0000080101000FC0010184E10A00006F0D000000FB8


-- Create a table to hold all the functions
local decoder = {}

decoder.name = "Sensor Sink Example"
decoder.filterName = "sensor"
decoder.description = "Lorem Ipsum"
decoder.enabled = true

function decoder.decode(event, fc)
  fc:appendEui64("SenderEuid64")
  local clusterId
  if event:containsSubkey("zigbeeApplicationSupport", "clusterId") == true then
     clusterId = event:getInt("zigbeeApplicationSupport", "clusterId")
  else
     clusterId = event:getInt("zigbeeApplicationSupport", "clusterIdV2")
  end
  if clusterId == 0x01 then
     fc:setSummary("Sink Advertise")
     fc:append("senderShortId", 2)
  elseif clusterId == 0x02 then
     fc:setSummary("Sensor Select Sink")
  elseif clusterId == 0x03 then
     fc:setSummary("Sink Ready")
  elseif clusterId == 0x04 then
     fc:setSummary("Sink Query")
  elseif clusterId == 0x0A then
     fc:setSummary("Sensor Data")
     fc:appendBytes("SensorData", 40)
  elseif clusterId == 0x64 then
     fc:setSummary("Sensor Hello")
     fc:appendString("hello", 5)
  end
end

function decoder.accept(event, packet)
  local accept = false
  local clusterId

  if event:containsSubkey("zigbeeApplicationSupport", "clusterId") == true then
     clusterId = event:getInt("zigbeeApplicationSupport", "clusterId")
  else
     clusterId = event:getInt("zigbeeApplicationSupport", "clusterIdV2")
  end

  local profileId = event:getInt("zigbeeApplicationSupport", "profileId")

  if (clusterId == 0x01 or clusterId == 0x02 or clusterId == 0x03 or clusterId == 0x04 or clusterId == 0x0A or clusterId == 68) and
(profileId == 0xC00F) then
     accept = true
  end

  return accept
end

return decoder
```

Payload Decoding Example

```
-- Example Packet w/ no length byte and payload only (payload-decoding-example.lua)
-- 01234567 89abcdef 01234567 89abcdef01234567 00000001 01000000 68656c6c6f20776f726c64 abcd abcd ffcdabcd 0f
00000000ff
-- Similar Example that is corrupted instead:
-- 01234567 89abcdef 01234567 89abcdef01234567 00000001 01000000 68656c6c6f20776f726c64 abcd abcd ffcdabcd ff
00000000ff

-- Create a table to hold all the functions
local decoder = {}

decoder.name = "Payload Decoding Example"
decoder.filterName = "exd"
decoder.description = "Lorem Ipsum"
decoder.enabled = true

function decoder.decode(event, fc)
  fc:setAppendLengthsToFieldCodes(true)
  fc:append("append", 4)
  fc:appendLE("appendLe", 4)
  fc:appendBytes("appendBytes", 4)
  fc:appendEui64("Eui64")
  fc:appendFloat("appendFloat", 4)
  fc:appendFloatLE("AppendFloatLe", 4)
  fc:appendString("AppendString", 11)
  fc:decode("decode", 2)
  fc:setPrefix("Prefix")
  fc:decodeLE("decodeLe", 2)
  fc:decodeBytes("decodeBytes", 4)
  fc:namelessBitField(1)
  fc:bit("bitTest0")
  fc:setSummary("Hello! ")

  if fc:decodeBit("Error") == true then
     fc:bits("Error Code", 3)
     fc:setCorruption("Error was set!")
     fc:setSummary("Failed!")
  else
     fc:decodeBits("Data", 3)
     fc:appendSummary("Success")
  end

  fc:setCodeNameSuffix("suffix")
  fc:skipBytes(4)
  fc:appendRemainingBytesLE("appendRemainingBytesLe")
end

function decoder.accept(event, packet)
  return true
end

return decoder
```

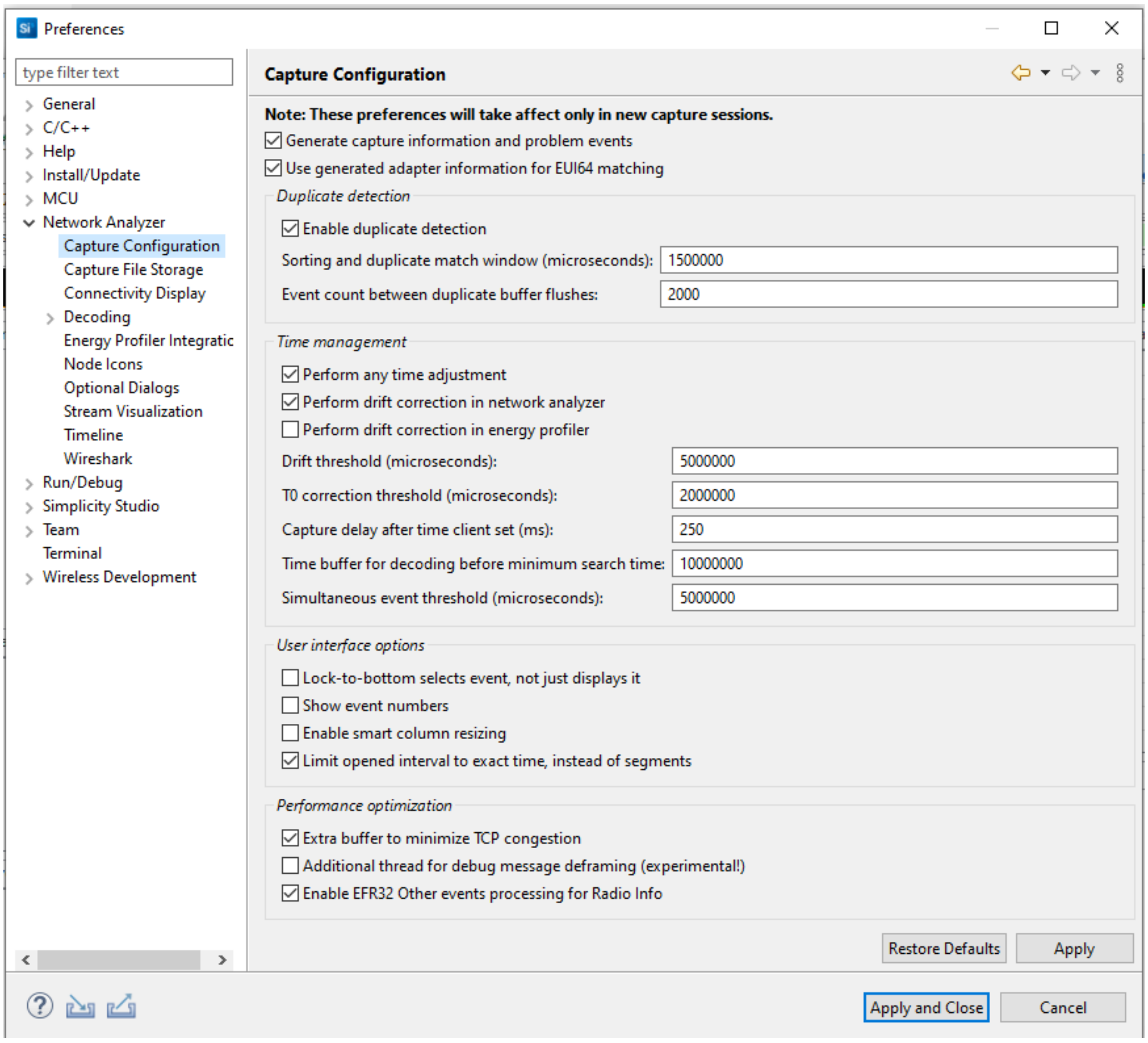
Network Analyzer Preferences

Network Analyzer Preferences

To access Network Analyzer preferences, select Window > Preferences > Network Analyzer. The preferences you set apply to all capture sessions. Network Analyzer saves your preferences and uses them each time Network Analyzer restarts. Preference categories include:

- Capture Configuration
- Capture File Storage
- Connectivity Display
- Decoding
- Energy Profiler Integration
- Node Icons
- Optional Dialogs
- Stream Visualization (deprecated function)
- Timeline
- Wireshark

Capture Configuration Preferences



Capture Configuration

Note: These preferences will take affect only in new capture sessions.

☑ Generate capture information and problem events

☑ Use generated adapter information for EUI64 matching

Duplicate detection

☑ Enable duplicate detection

Sorting and duplicate match window (microseconds):	1500000
Event count between duplicate buffer flushes:	2000

Time management

☑ Perform any time adjustment

☑ Perform drift correction in network analyzer

☐ Perform drift correction in energy profiler

Drift threshold (microseconds):	5000000
T0 correction threshold (microseconds):	2000000
Capture delay after time client set (ms):	250
Time buffer for decoding before minimum search time:	10000000
Simultaneous event threshold (microseconds):	5000000

User interface options

☐ Lock-to-bottom selects event, not just displays it

☐ Show event numbers

☐ Enable smart column resizing

☑ Limit opened interval to exact time, instead of segments

Performance optimization

☑ Extra buffer to minimize TCP congestion

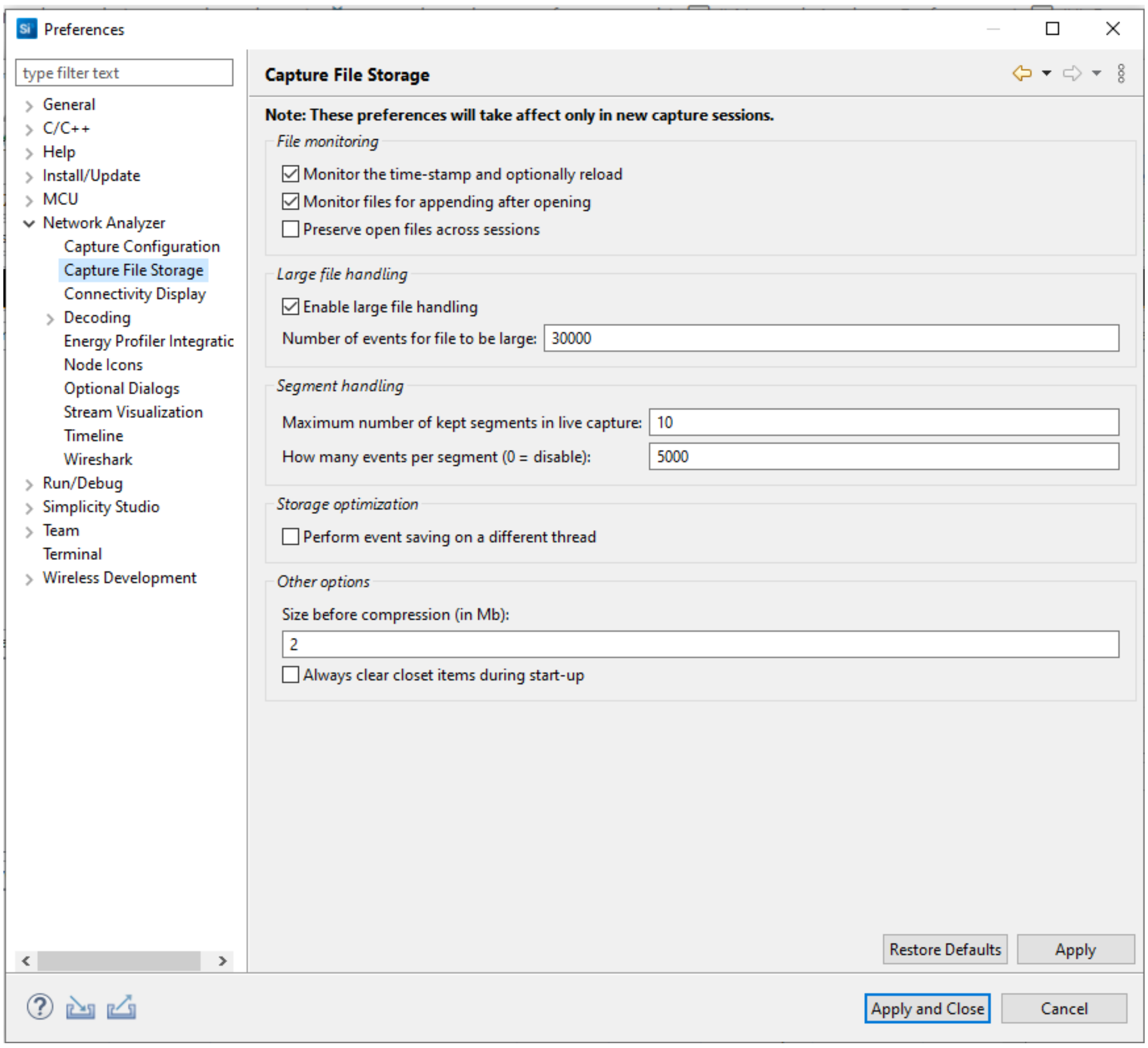
☐ Additional thread for debug message deframing (experimental!)

☑ Enable EFR32 Other events processing for Radio Info

These preferences offer general, time management, and user interface configuration options. Changes take effect in the next new capture session. Some options are:

- Sorting and duplicate match window: Specifies in microseconds a time span in which duplicate packets can be detected. If identical packets arrive within the specified time span, Network Analyzer detects the duplication and allows only one to display.
- Perform drift correction in network Analyzer: Allows drift for autocorrection.
- Drift Threshold: Specifies the amount of time-drift that Network Analyzer will tolerate between the absolute PC clock and the adapter clocks before it resets its "time-correction" factor. In most situations you should not change this value. In a typical situation, adapter clocks are more precise than PC clocks, but they do not provide absolute time, just relative time against an arbitrary reference.

Capture File Storage Preferences



Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
  - Capture Configuration
  - Capture File Storage
  - Connectivity Display
  - Decoding
  - Energy Profiler Integratic
  - Node Icons
  - Optional Dialogs
  - Stream Visualization
  - Timeline
  - Wireshark
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

Capture File Storage

Note: These preferences will take affect only in new capture sessions.

File monitoring

- [✓] Monitor the time-stamp and optionally reload
- [✓] Monitor files for appending after opening
- [ ] Preserve open files across sessions

Large file handling

- [✓] Enable large file handling

Number of events for file to be large: 30000

Segment handling

Maximum number of kept segments in live capture: 10

How many events per segment (0 = disable): 5000

Storage optimization

- [ ] Perform event saving on a different thread

Other options

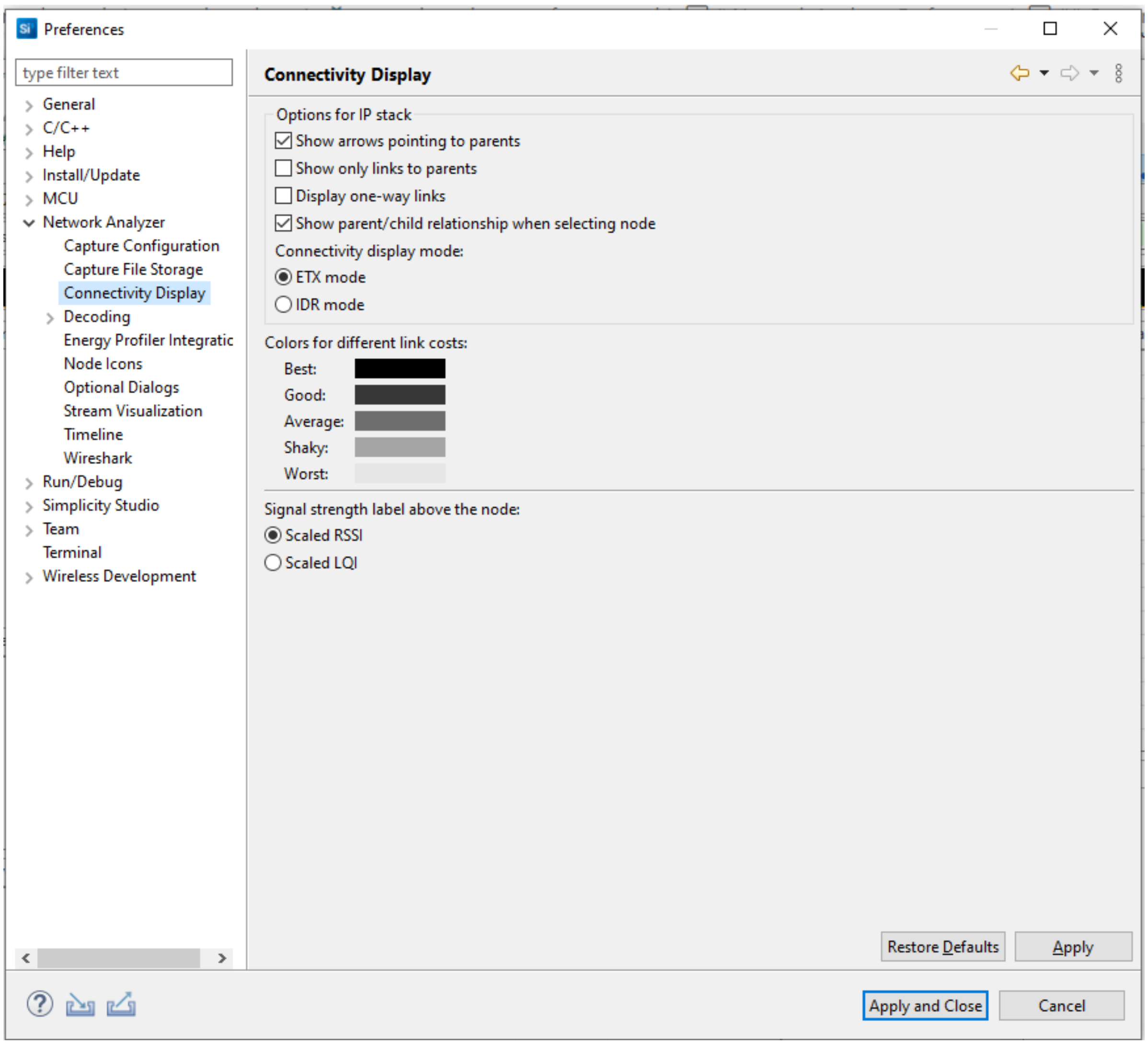
Size before compression (in Mb): 2

- [ ] Always clear closet items during start-up

These preferences determine, among other things, what is considered a large file, which in turn determines the default editor used. Changes take effect in the next new capture session. Some of the options are:

- Monitor the timestamp and optionally reload: Monitors whether the file has changed on the disk, and prompts to reload if so.
- Monitor files for appending after opening: Enables Network Analyzer to detect new packet data appended by an external program to an open session file, and read it in for display.
- Preserve open files across sessions: Instructs Network Analyzer to automatically reopen files from previous session
- Enable large file handling: Enables Network Analyzer to handle large files in the Large File Editor. Files are considered large if they contain more events than the number entered in "Number of events for a file to be large."

Connectivity Display Preferences



Connectivity Display

Options for IP stack

- [✓] Show arrows pointing to parents
- [ ] Show only links to parents
- [ ] Display one-way links
- [✓] Show parent/child relationship when selecting node

Connectivity display mode: [✓ ETX mode] IDR mode

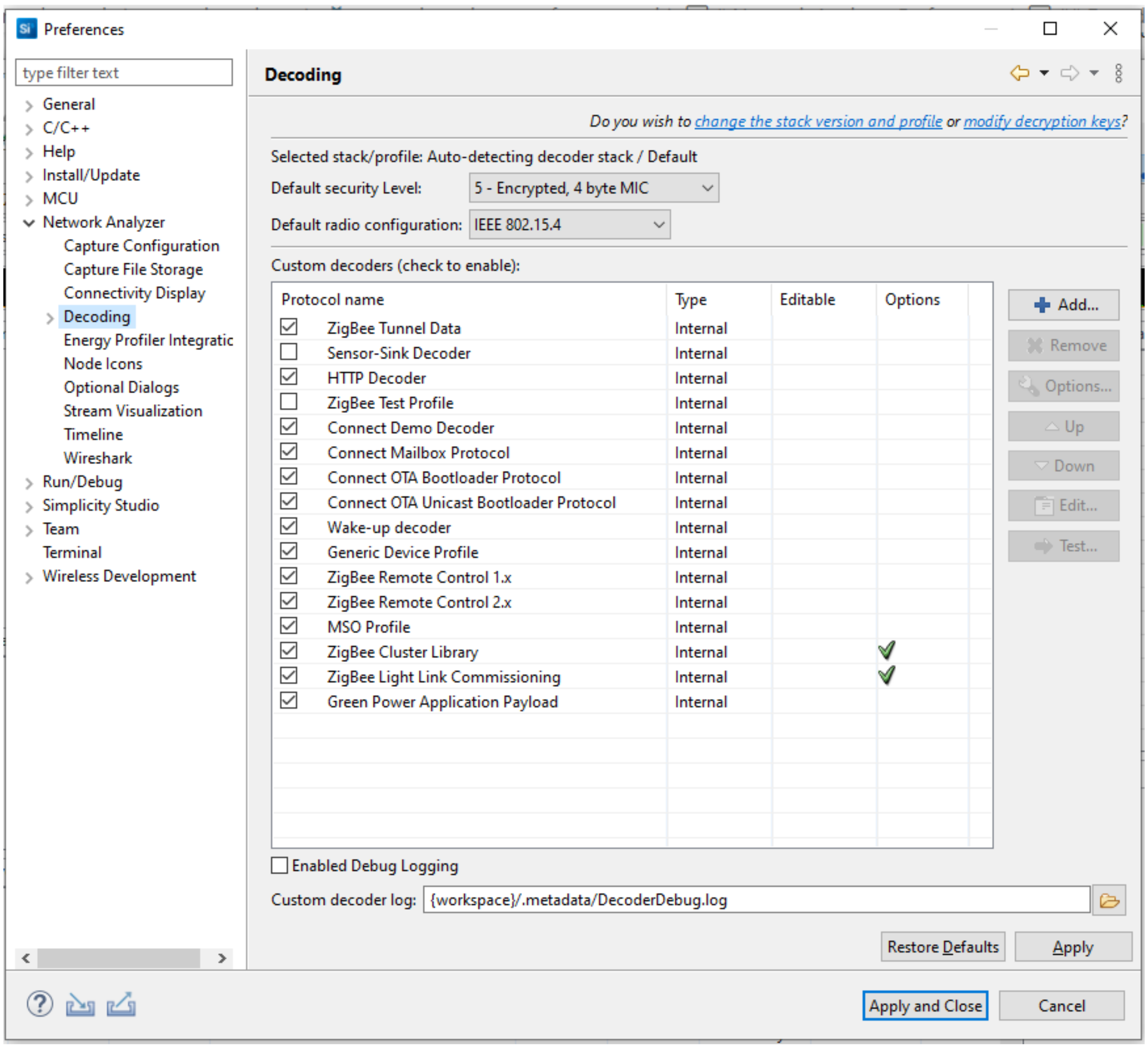
Colors for different link costs:

Best:	■
Good:	■
Average:	▨
Shaky:	▨
Worst:	□

Signal strength label above the node: [✓ Scaled RSSI] Scaled LQI

These preferences customize how device connectivity is displayed.

Decoding Preferences



Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
  - Capture Configuration
  - Capture File Storage
  - Connectivity Display
  - Decoding
  - Energy Profiler Integratic
  - Node Icons
  - Optional Dialogs
  - Stream Visualization
  - Timeline
  - Wireshark
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

Decoding

Do you wish to change the stack version and profile or modify decryption keys?

Selected stack/profile: Auto-detecting decoder stack / Default

Default security Level: 5 - Encrypted, 4 byte MIC

Default radio configuration: IEEE 802.15.4

Custom decoders (check to enable):

	Protocol name	Type	Editable	Options
☑	ZigBee Tunnel Data	Internal		
☐	Sensor-Sink Decoder	Internal		
☑	HTTP Decoder	Internal		
☐	ZigBee Test Profile	Internal		
☑	Connect Demo Decoder	Internal		
☑	Connect Mailbox Protocol	Internal		
☑	Connect OTA Bootloader Protocol	Internal		
☑	Connect OTA Unicast Bootloader Protocol	Internal		
☑	Wake-up decoder	Internal		
☑	Generic Device Profile	Internal		
☑	ZigBee Remote Control 1.x	Internal		
☑	ZigBee Remote Control 2.x	Internal		
☑	MSO Profile	Internal		
☑	ZigBee Cluster Library	Internal		✓
☑	ZigBee Light Link Commissioning	Internal		✓
☑	Green Power Application Payload	Internal		

Add... | Remove | Options... | Up | Down | Edit... | Test...

☐ Enabled Debug Logging

Custom decoder log: {workspace}/.metadata/DecoderDebug.log

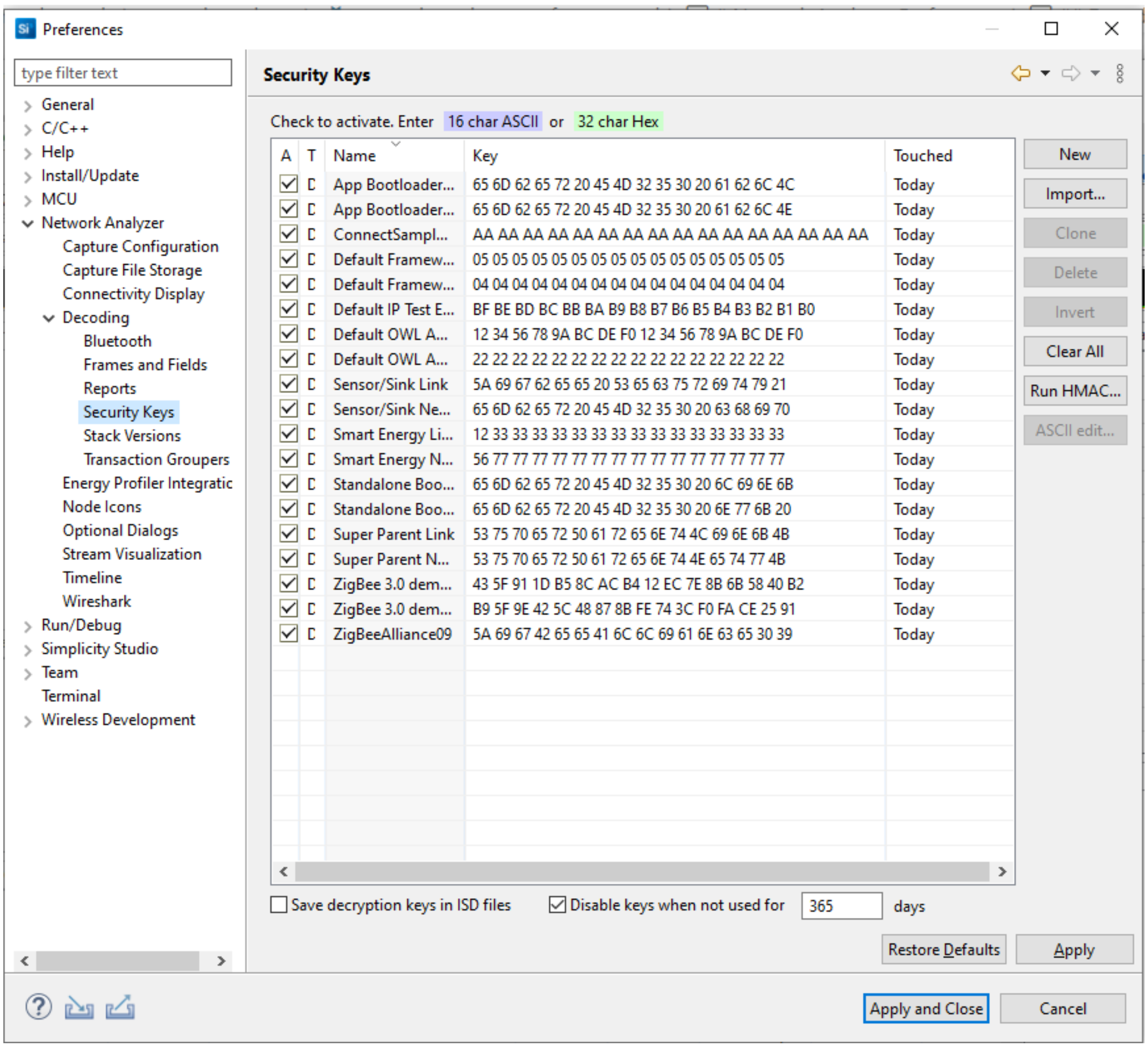
Restore Defaults | Apply

Apply and Close | Cancel

The **Decoding** preferences group affect how decoding is handled. Changes take effect in the next new capture session. On the Decoding dialog, some options are:

- Selected stack: The stack selected for decoding newly captured data. Links allow you to make changes in other preference interfaces. This setting has no effect on a previously captured trace. The decoding stack for a trace file is stored in the file itself.
- Security level: The security level set for the MAC and network layers. Valid values include 1 through 5.

Security Keys



Security Keys

Check to activate. Enter 16 char ASCII or 32 char Hex

A	T	Name	Key	Touched
✓	D	App Bootloader...	65 6D 62 65 72 20 45 4D 32 35 30 20 61 62 6C 4C	Today
✓	D	App Bootloader...	65 6D 62 65 72 20 45 4D 32 35 30 20 61 62 6C 4E	Today
✓	D	ConnectSampl...	AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA	Today
✓	D	Default Framew...	05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05	Today
✓	D	Default Framew...	04 04 04 04 04 04 04 04 04 04 04 04 04 04 04 04	Today
✓	D	Default IP Test E...	BF BE BD BC BB BA B9 B8 B7 B6 B5 B4 B3 B2 B1 B0	Today
✓	D	Default OWL A...	12 34 56 78 9A BC DE F0 12 34 56 78 9A BC DE F0	Today
✓	D	Default OWL A...	22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22	Today
✓	D	Sensor/Sink Link	5A 69 67 62 65 65 20 53 65 63 75 72 69 74 79 21	Today
✓	D	Sensor/Sink Ne...	65 6D 62 65 72 20 45 4D 32 35 30 20 63 68 69 70	Today
✓	D	Smart Energy Li...	12 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33	Today
✓	D	Smart Energy N...	56 77 77 77 77 77 77 77 77 77 77 77 77 77 77 77	Today
✓	D	Standalone Boo...	65 6D 62 65 72 20 45 4D 32 35 30 20 6C 69 6E 6B	Today
✓	D	Standalone Boo...	65 6D 62 65 72 20 45 4D 32 35 30 20 6E 77 6B 20	Today
✓	D	Super Parent Link	53 75 70 65 72 50 61 72 65 6E 74 4C 69 6E 6B 4B	Today
✓	D	Super Parent N...	53 75 70 65 72 50 61 72 65 6E 74 4E 65 74 77 4B	Today
✓	D	ZigBee 3.0 dem...	43 5F 91 1D B5 8C AC B4 12 EC 7E 8B 6B 58 40 B2	Today
✓	D	ZigBee 3.0 dem...	B9 5F 9E 42 5C 48 87 8B FE 74 3C F0 FA CE 25 91	Today
✓	D	ZigBeeAlliance09	5A 69 67 42 65 65 41 6C 6C 69 61 6E 63 65 30 39	Today

☐ Save decryption keys in ISD files ☑ Disable keys when not used for 365 days

Specifies the security decryption keys. To enable a key, select the Decryption key checkbox. To edit a key, select its name or key values. All enabled decryption keys are run against each incoming packet until one is successful. Successful keys are automatically moved to the top of the key list to improve performance. Decryption keys can also be obtained from network traffic for future use.

Buttons on the right of the dialog provide the following features:

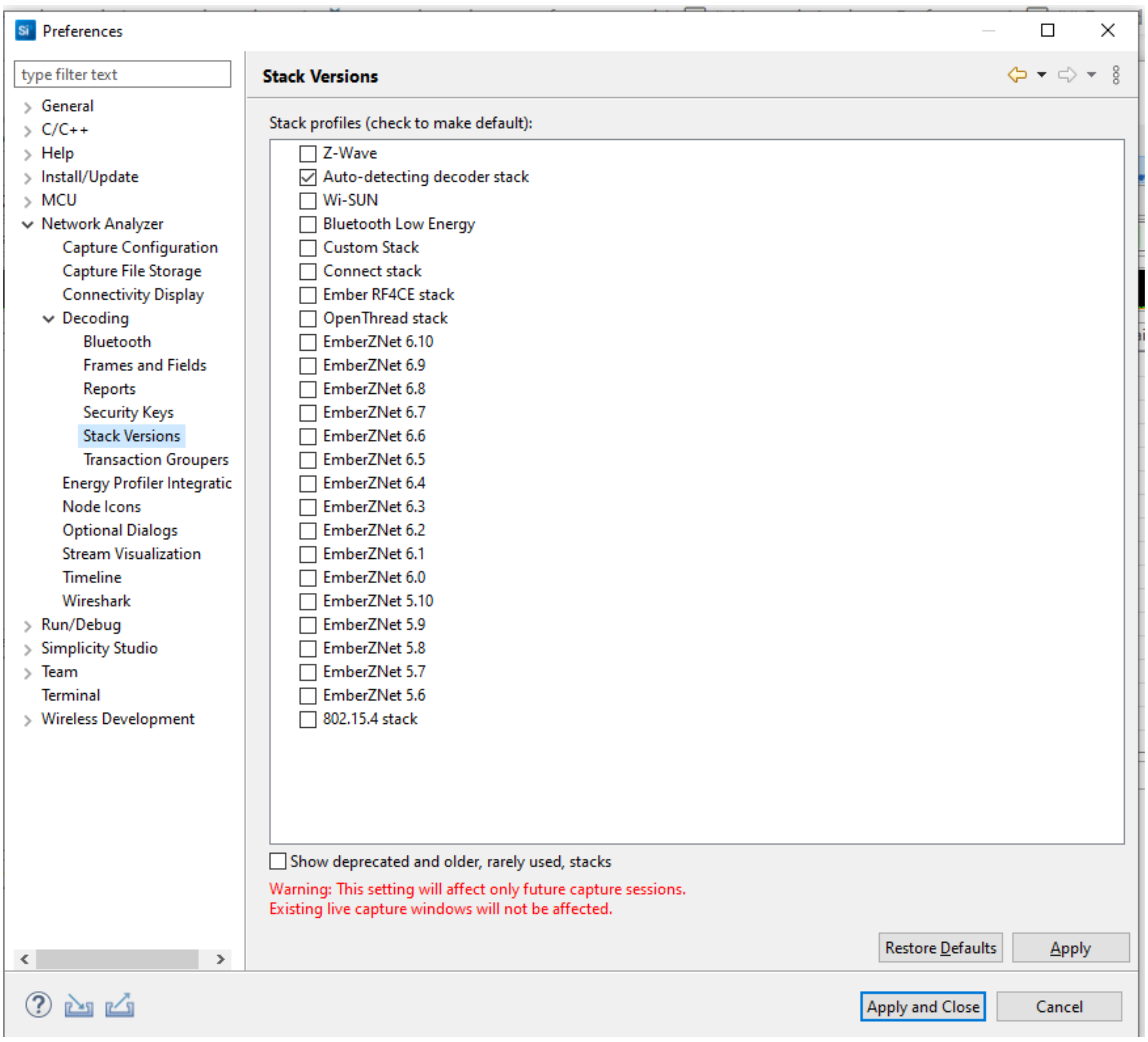
- New: Creates a new key for editing.
- Import: Opens a dialog from which you can import a key file.
- Clone: Creates a copy of the currently selected key.
- Delete: Removes the currently selected key.
- Invert swaps the order of the key values.
- Clear All: Removes all security keys.
- Run HMAC: Opens a dialog that allows you to manually calculate the HMAC authenticated key from trust key and IEEE EUI64 address.
- ASCII edit...: Converts a human-readable ASCII string into a binary security key.

The checkboxes at the bottom of the keys list provide the following options:

- Save decryption keys in ISD files (unchecked by default): Saves the security keys that you have specified into the capture file along with your traffic. Note: If you are sharing the Network Analyzer file with users who have a right to know your key, this may make the opening of the file easier for them, as they will not have to separately enter the key. However, by doing this, you create a security risk. Never enable this option, if your keys must remain secure and known to you only.

- Disable keys when not used for n days (checked by default): You can configure the number of days unused keys should be kept.

Stack Versions



Stack profiles (check to make default):

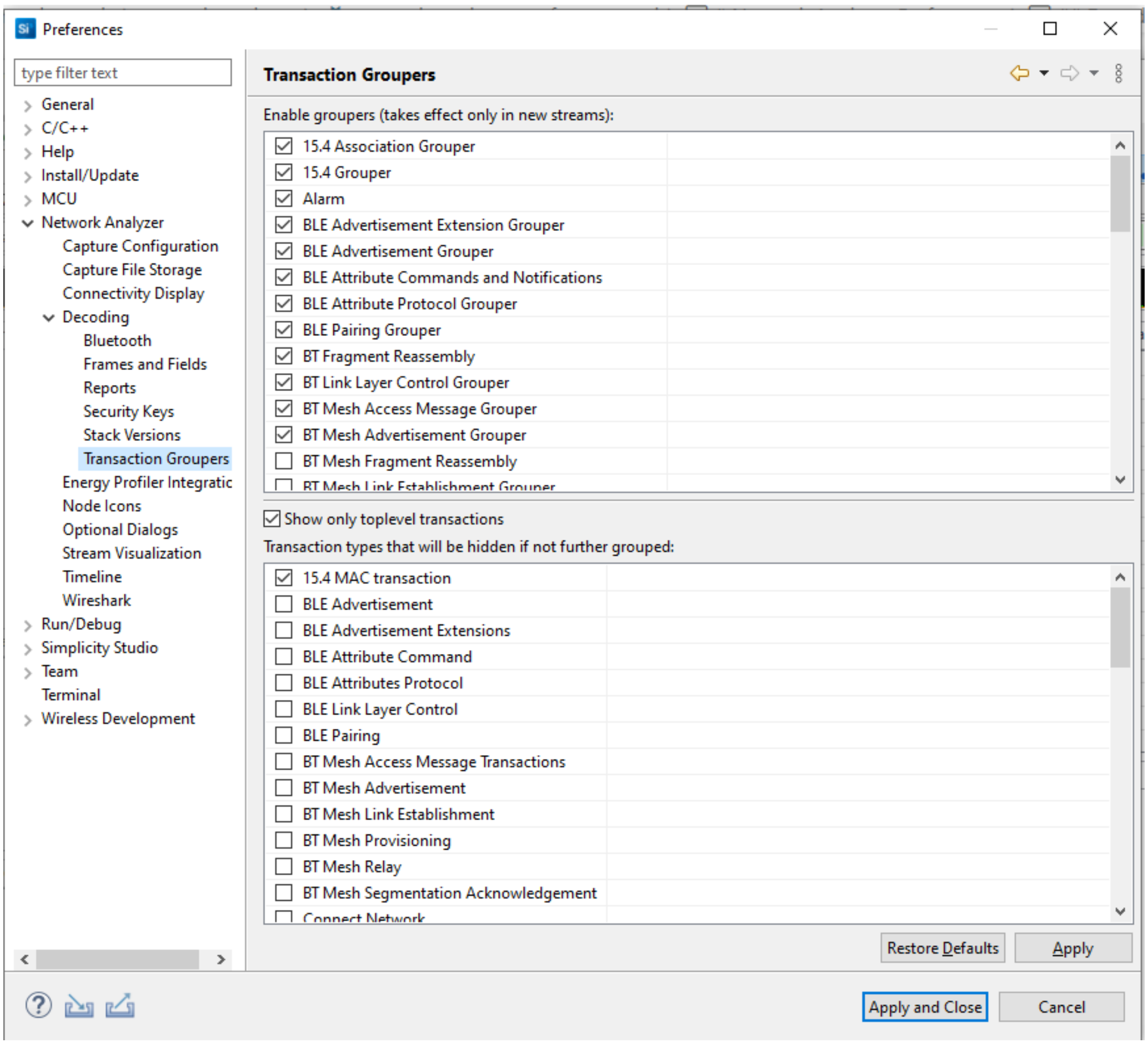
- [ ] Z-Wave
- [✓] Auto-detecting decoder stack
- [ ] Wi-SUN
- [ ] Bluetooth Low Energy
- [ ] Custom Stack
- [ ] Connect stack
- [ ] Ember RF4CE stack
- [ ] OpenThread stack
- [ ] EmberZNet 6.10
- [ ] EmberZNet 6.9
- [ ] EmberZNet 6.8
- [ ] EmberZNet 6.7
- [ ] EmberZNet 6.6
- [ ] EmberZNet 6.5
- [ ] EmberZNet 6.4
- [ ] EmberZNet 6.3
- [ ] EmberZNet 6.2
- [ ] EmberZNet 6.1
- [ ] EmberZNet 6.0
- [ ] EmberZNet 5.10
- [ ] EmberZNet 5.9
- [ ] EmberZNet 5.8
- [ ] EmberZNet 5.7
- [ ] EmberZNet 5.6
- [ ] 802.15.4 stack

- [ ] Show deprecated and older, rarely used, stacks

Warning: This setting will affect only future capture sessions.
Existing live capture windows will not be affected.

A list of profiles representing Gecko SDK Suite stack versions. Check the stack that is deployed on the network you are using. If you are working with Dynamic Multiprotocol projects, select 'Auto-detecting decoder stack'.

Transaction Groupers


Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
  - Capture Configuration
  - Capture File Storage
  - Connectivity Display
  - Decoding
    - Bluetooth
    - Frames and Fields
    - Reports
    - Security Keys
    - Stack Versions
    - Transaction Groupers
  - Energy Profiler Integratic
  - Node Icons
  - Optional Dialogs
  - Stream Visualization
  - Timeline
  - Wireshark
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

Transaction Groupers

Enable groupers (takes effect only in new streams):

- [✓] 15.4 Association Grouper
- [✓] 15.4 Grouper
- [✓] Alarm
- [✓] BLE Advertisement Extension Grouper
- [✓] BLE Advertisement Grouper
- [✓] BLE Attribute Commands and Notifications
- [✓] BLE Attribute Protocol Grouper
- [✓] BLE Pairing Grouper
- [✓] BT Fragment Reassembly
- [✓] BT Link Layer Control Grouper
- [✓] BT Mesh Access Message Grouper
- [✓] BT Mesh Advertisement Grouper
- [ ] BT Mesh Fragment Reassembly
- [ ] BT Mesh Link Establishment Grouper

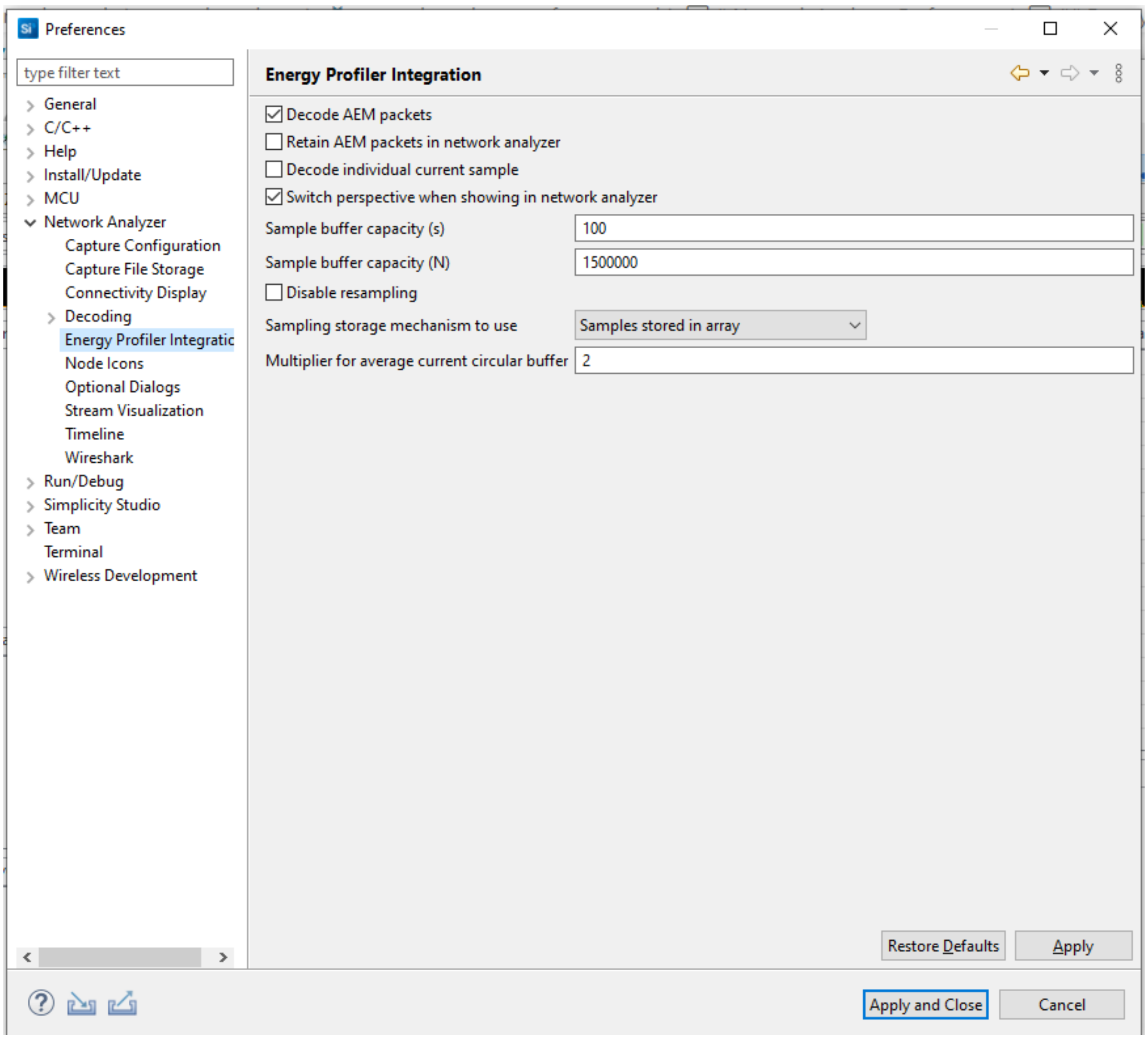
[✓] Show only toplevel transactions

Transaction types that will be hidden if not further grouped:

- [✓] 15.4 MAC transaction
- [ ] BLE Advertisement
- [ ] BLE Advertisement Extensions
- [ ] BLE Attribute Command
- [ ] BLE Attributes Protocol
- [ ] BLE Link Layer Control
- [ ] BLE Pairing
- [ ] BT Mesh Access Message Transactions
- [ ] BT Mesh Advertisement
- [ ] BT Mesh Link Establishment
- [ ] BT Mesh Provisioning
- [ ] BT Mesh Relay
- [ ] BT Mesh Segmentation Acknowledgement
- [ ] Connect Network

A table of all the groupers loaded into Network Analyzer. Transaction groupers are responsible for making transactional sense out of a trace of network data. Groupers watch for batches of events and record them as a transaction.

Energy Profiler Integration Preferences


Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
- MCU
- Network Analyzer
  - Capture Configuration
  - Capture File Storage
  - Connectivity Display
  - Decoding
  - Energy Profiler Integratic
  - Node Icons
  - Optional Dialogs
  - Stream Visualization
  - Timeline
  - Wireshark
- Run/Debug
- Simplicity Studio
- Team
- Terminal
- Wireless Development

Energy Profiler Integration

- [x] Decode AEM packets
- [ ] Retain AEM packets in network analyzer
- [ ] Decode individual current sample
- [x] Switch perspective when showing in network analyzer

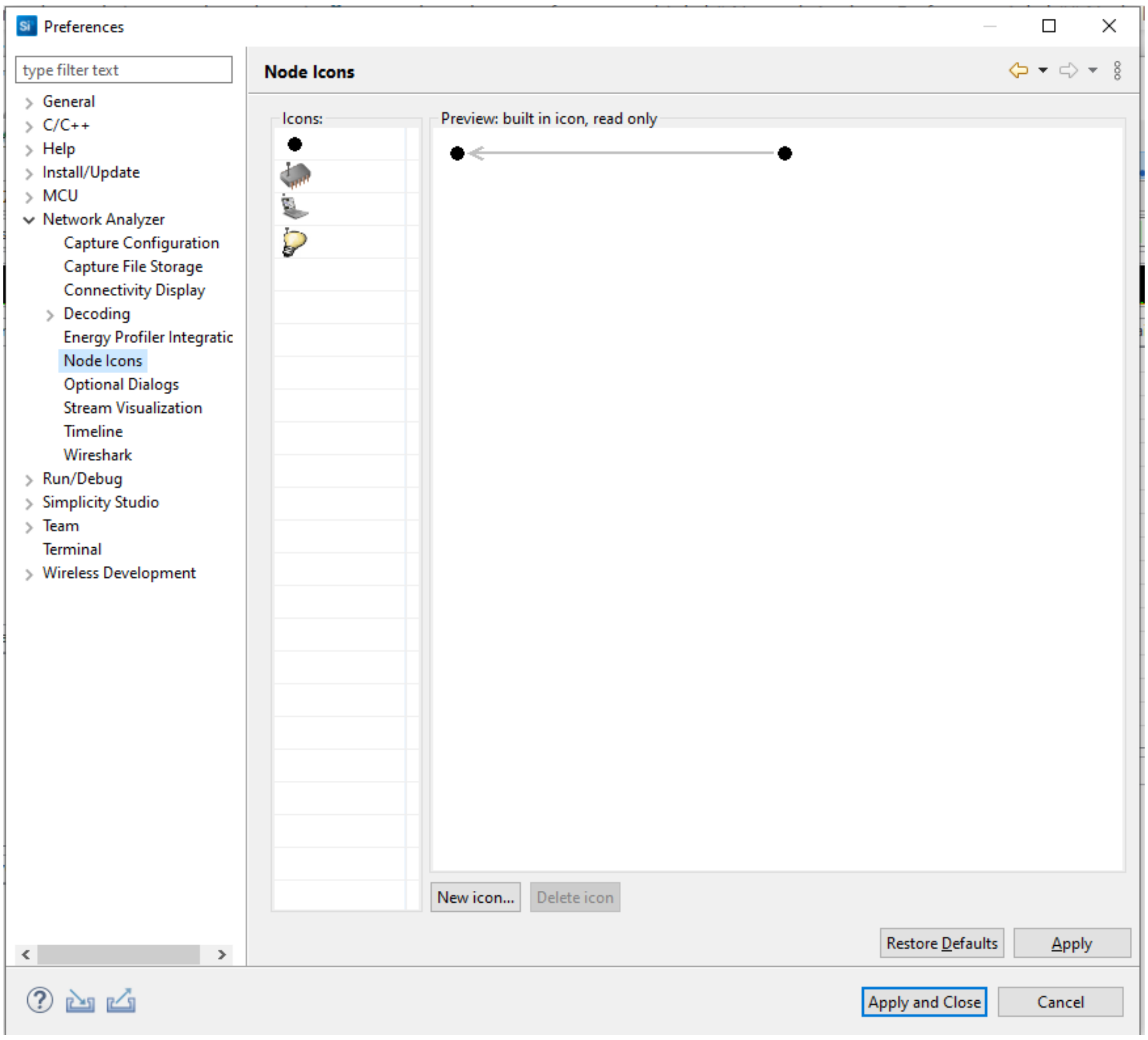
Sample buffer capacity (s)	100
Sample buffer capacity (N)	1500000

- [ ] Disable resampling

Sampling storage mechanism to use	Samples stored in array
Multiplier for average current circular buffer	2

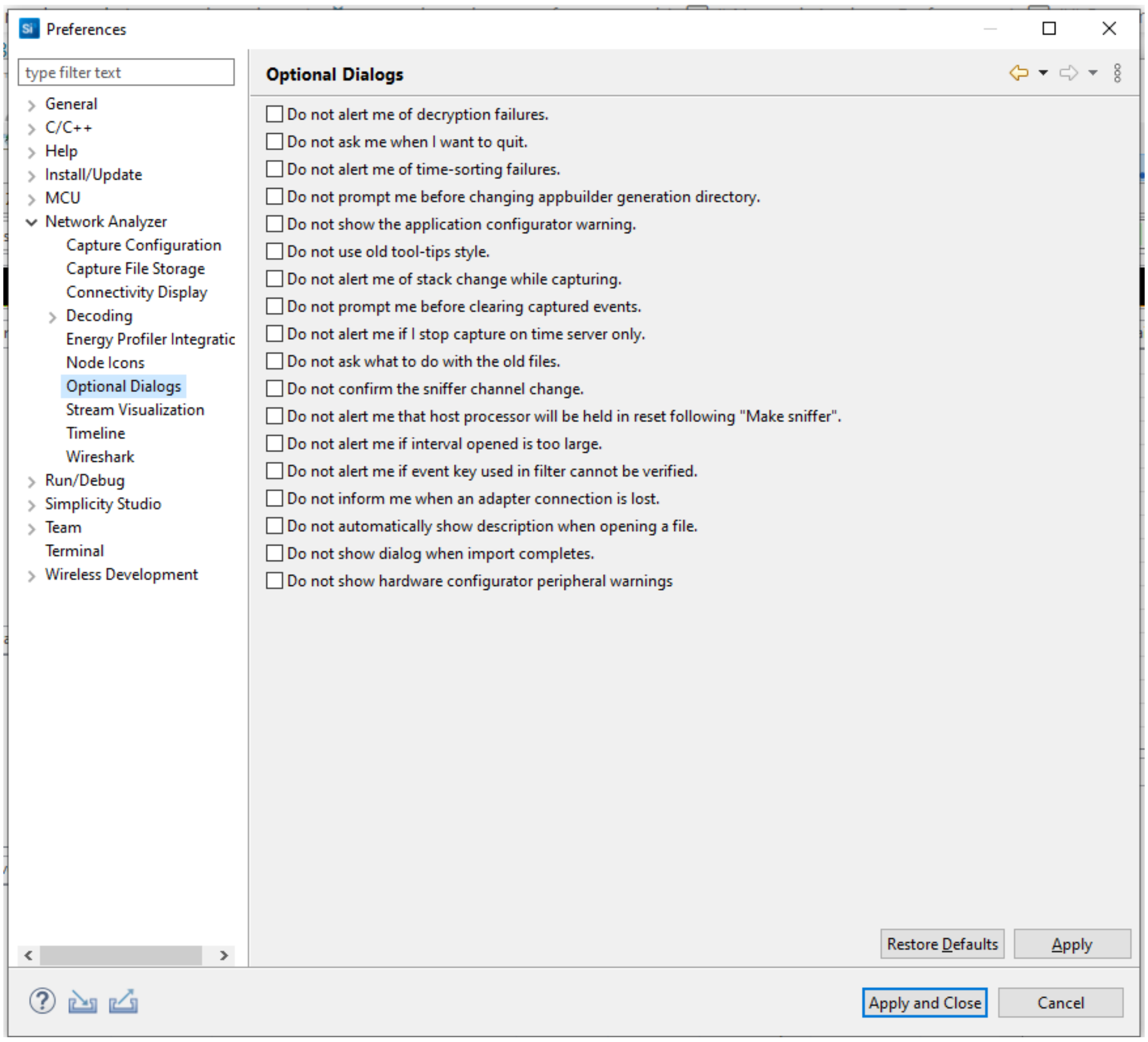
See Energy Profiler and Network Analyzer in the Energy Profiler section for more information about using the two tools together and the effect of these preferences.

Node Icon Preferences



Network Analyzer provides several predefined icons. The Node Icons dialog displays all icons that are available for customizing display of nodes in the map pane. You can also add icons of your own.

Optional Dialog Preferences

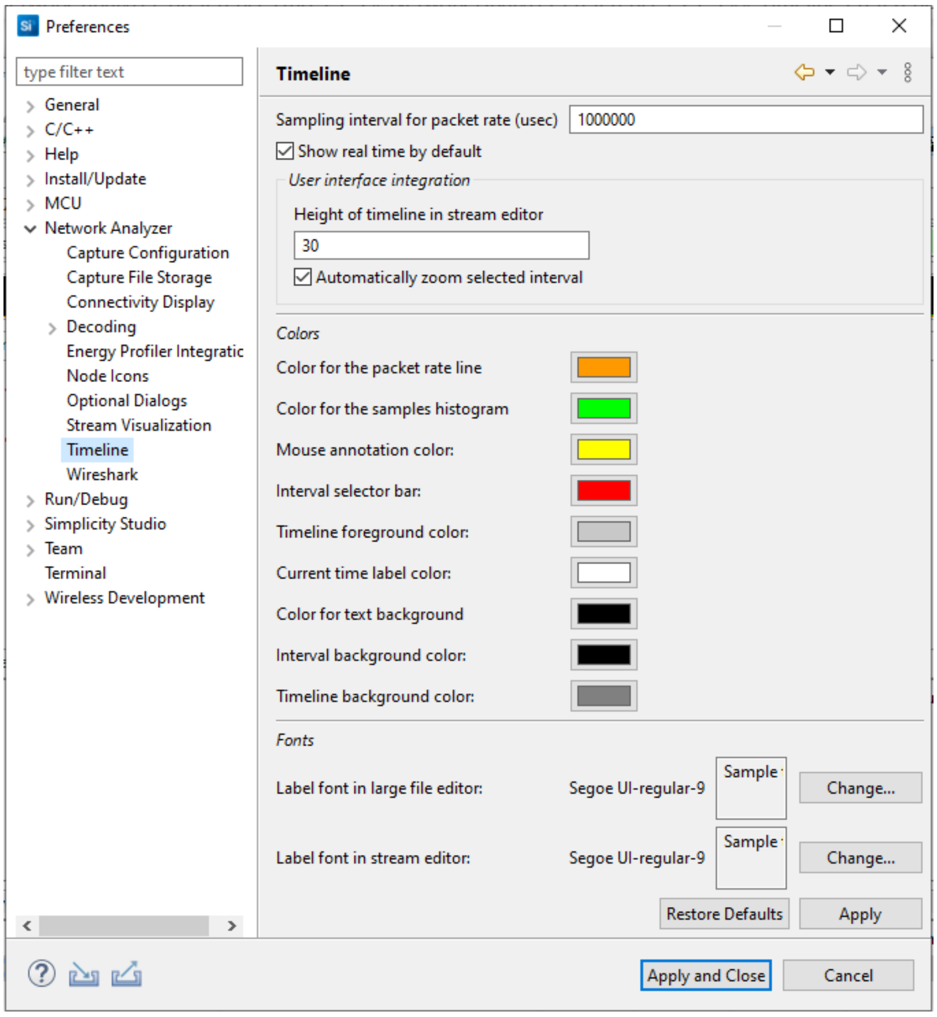


Optional Dialogs

- ☐ Do not alert me of decryption failures.
- ☐ Do not ask me when I want to quit.
- ☐ Do not alert me of time-sorting failures.
- ☐ Do not prompt me before changing appbuilder generation directory.
- ☐ Do not show the application configurator warning.
- ☐ Do not use old tool-tips style.
- ☐ Do not alert me of stack change while capturing.
- ☐ Do not prompt me before clearing captured events.
- ☐ Do not alert me if I stop capture on time server only.
- ☐ Do not ask what to do with the old files.
- ☐ Do not confirm the sniffer channel change.
- ☐ Do not alert me that host processor will be held in reset following "Make sniffer".
- ☐ Do not alert me if interval opened is too large.
- ☐ Do not alert me if event key used in filter cannot be verified.
- ☐ Do not inform me when an adapter connection is lost.
- ☐ Do not automatically show description when opening a file.
- ☐ Do not show dialog when import completes.
- ☐ Do not show hardware configurator peripheral warnings

Some dialogs can be turned off if you prefer not to interact with them.

Timeline Preferences



Timeline

Sampling interval for packet rate (usec) 1000000

☑ Show real time by default

User interface integration

Height of timeline in stream editor

30

☑ Automatically zoom selected interval

Colors

Color for the packet rate line	🟧
Color for the samples histogram	🟩
Mouse annotation color:	🟨
Interval selector bar:	🟥
Timeline foreground color:	⬜
Current time label color:	⬜
Color for text background	⬛
Interval background color:	⬛
Timeline background color:	⬛

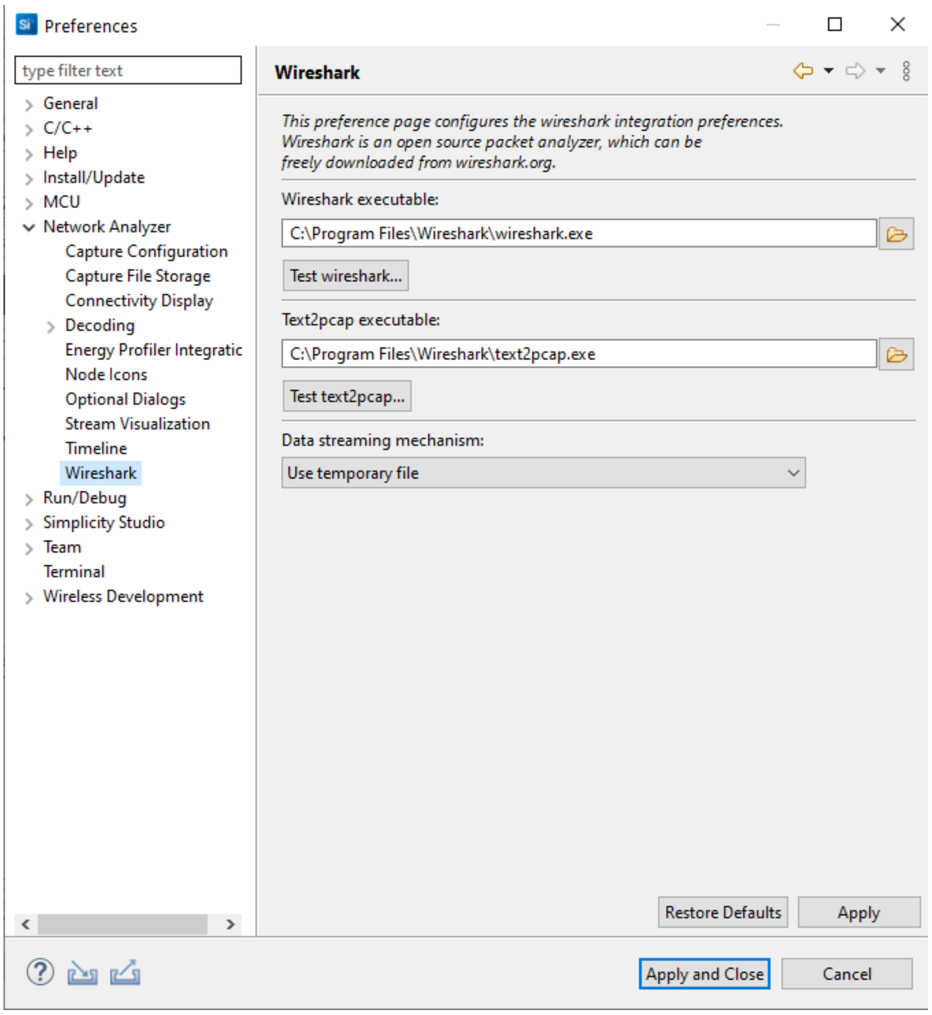
Fonts

Label font in large file editor:	Segoe UI-regular-9	Sample	Change...
Label font in stream editor:	Segoe UI-regular-9	Sample	Change...

Among other options, you can choose the colors and fonts for features of the Timeline.

Wireshark Preferences

If you are using the Wireshark open source packet analyzer, these options configure the integration.



Wireshark

This preference page configures the wireshark integration preferences. Wireshark is an open source packet analyzer, which can be freely downloaded from wireshark.org.

Wireshark executable: C:\Program Files\Wireshark\wireshark.exe

Text2pcap executable: C:\Program Files\Wireshark\text2pcap.exe

Data streaming mechanism: Use temporary file

Bluetooth Direction Finding Tools

Bluetooth Direction Finding Tool Suite

A Real Time Locating System (RTLS) consists of many locators that pick up the signals of asset tags to locate them in 3D space. To calculate the positions of the asset tags, you must know each locator's position, orientation, and configuration (such as antenna array type, switching pattern, and so on). The Silicon Labs' Real Time Locating (RTL) library also needs configuration parameters, such as estimation mode, azimuth, elevation constraints, and so on, to provide the best estimation for a given environment. The Bluetooth Direction Finding Tool Suite is meant to ease development with the Silicon Labs' RTL library. It provides tools to configure the system, and also helps development with analyzer tools that calculate many output parameters from the observed IQ samples.

Note: The tools were created for demonstration purposes, and not to be used in production. For development purposes, start with the sample application described in AN1296: Application Development with Silicon Labs RTL Library.

The implementation described in AN1296 requires you to configure and build both individual locators and a locator host application in order to test the system. While the SDK examples provide the openness to customize and add new features to your direction find-ing project using RTL library APIs directly, the manual configuration of the locators and their topologies may not be a trivial task. It is simpler to configure a single locator and multi-locator setup through interactive graphical interfaces available through the Direction Finding Tools Suite. The Suite is composed of:

- AoA Analyzer
- Positioning Tool

Both are available through the Simplicity Studio Tools menu.

AoA Analyzer

AoA Analyzer can:

- Create a locator configuration file without the need to manually edit the .json file, to ease the configurations process.
- Apply the configuration for a single locator board and run the RTL library to estimate angles from the incoming data (without the need to build and run any host sample application).

Positioning Tool

While it is easy to configure a single locator and get angle information of asset tags, configuring and running a multi-locator setup to get position information can be rather complicated. To calculate position, the RTL needs to know the coordinates and orientation of the locator so that it can translate the angle feeds from each one of them with respect to the coordinate system of a given topology. Simplicity Studio provides the Positioning tool, which makes the configuration and testing of a multi-locator setup easy and quick.

Positioning Tool can:

- Create a multi-locator topology without the need to manually edit the configuration .json file.
- Add locators to the topology and configure their positioning RTL parameters (such coordinates and orientation) easily.
- Apply the configuration and locate asset tags in 3D.
- Import/Export a topology from/to a configuration .json file.

For details about the tools and references to other direction-finding documentation, see UG514: Using the Bluetooth® Direction Finding Tool Suite

Tips and Tricks

Many of these tips rely on changes in Preferences. This can be accessed through the **Window > Preferences** selection, or the **Preferences** button on the toolbar.

Useful Windows 10 paths

Default Simplicity Studio Install Path: C:\SiliconLabs\SimplicityStudio

Downloaded location for kit resources:
C:\SiliconLabs\SimplicityStudio\v5\offline\aem\www.silabs.com\documents\public\schematic-files

Command line or direct GUI access to Simplicity Commander:
C:\SiliconLabs\SimplicityStudio\v5\developer\adapter_packs\commander

Protocol SDK locations for all GSDK SDKs:

- GSDK 4.x: C:\Users\&ltNAME>\SimplicityStudio\SDKs\gecko_sdk\protocol\<protocol>
- GSDK 3.x: C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\<GSDK version>\<protocol>

Beginning with GSDK 4.0, content such as documentation or demos that was previously installed with the SDK is now downloaded after being accessed through Simplicity Studio:
C:\SiliconLabs\SimplicityStudio\v5\offline\com.silabs.sdk.stack.super_<version>

Useful Mac OS paths

Default Simplicity Studio Install Path: /Applications/SimplicityStudio.app/Contents/Eclipse (recommend changing to one word SimplicityStudio.app)

Downloaded location for kit resources:
/Applications/SimplicityStudio.app/Contents/Eclipse/offline/aem/www.silabs.com/documents/public/schematic-files

Command line or direct GUI access to Simplicity Commander:
/Applications/SimplicityStudio.app/Contents/Eclipse/developer/adapter_packs/commander

Protocol SDK locations for all GSDK SDKs:

- GSDK 4.x: /Users/&ltNAME>/SimplicityStudio/SDKs/gecko_sdk/protocol/<protocol>
- GSDK 3.x: /Applications/SimplicityStudio.app/Contents/Eclipse/developer/sdks/gecko_sdk_suite/<GSDK version>/protocol

Beginning with GSDK 4.0, content such as documentation or demos that was previously installed with the SDK is now downloaded after being accessed through Simplicity Studio:
/Applications/SimplicityStudio.app/Contents/Eclipse/offline/com.silabs.sdk.stack.super_<version>

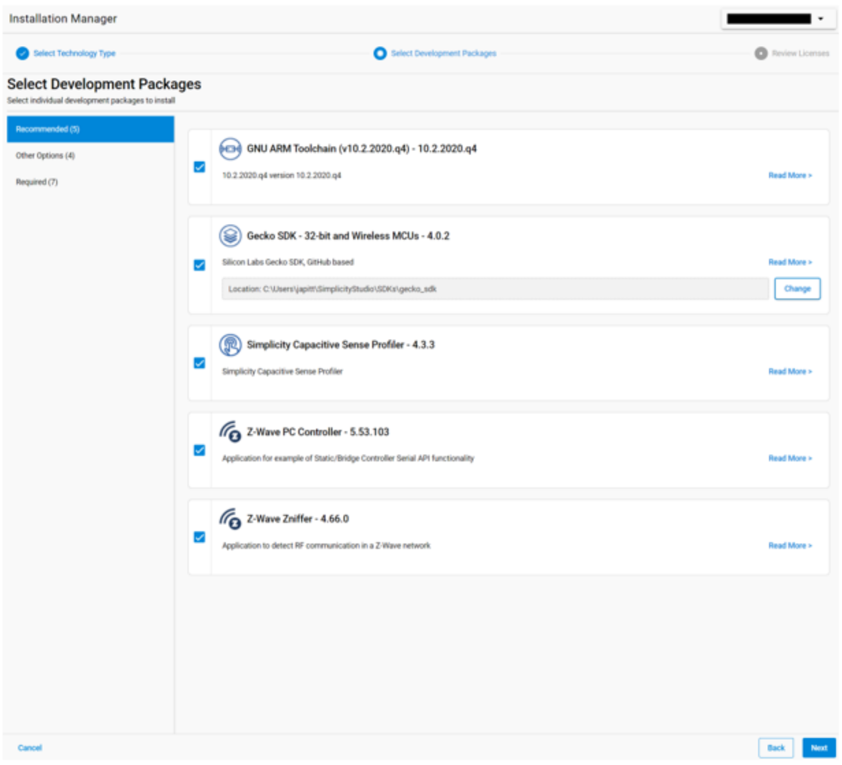
Default Project Locations

Default project locations in your Windows 10 workspace C:\Users&lt;user>\SimplicityStudio Ex:
C:\Users\daseymou\SimplicityStudio

Default project locations in your Mac workspace /Users/<user>/SimplicityStudio Ex: /Users/mahallam/SimplicityStudio

Get All Parts of an Update

When you update Simplicity Studio, whether in response to an update notification or in the Installation Manager, Simplicity Studio installs all Product Updates. However, other updates may also be available, for example for the GSDK or compiler support. The best practice after Simplicity Studio installs an update is to open the Installation Manager and select **Install by Technology Type**. Check the boxes for each technology type being used and click **Next**. Click **Advanced** and then **Next**. The next dialog shows you if there are any required or recommended packages to be installed. If yes, install them, otherwise click **Cancel** to leave the Installation Manager. For example, in an update from Simplicity Studio version 5.1 to 5.3.2, the following recommended updates, including GCC toolchain and the GSDK, were available.



Select Development Packages

Select individual development packages to install

- ✓ GNU ARM Toolchain (v10.2.2020.q4) - 10.2.2020.q4 — 10.2.2020.q4 version 10.2.2020.q4
- ✓ Gecko SDK - 32-bit and Wireless MCUs - 4.0.2 — Silicon Labs Gecko SDK, GitHub based — Location: C:\Users\japitt\SimplicityStudio\SDKs\gecko_sdk
- ✓ Simplicity Capacitive Sense Profiler - 4.3.3 — Simplicity Capacitive Sense Profiler
- ✓ Z-Wave PC Controller - 5.53.103 — Application for example of Static/Bridge Controller Serial API functionality
- ✓ Z-Wave Zniffer - 4.66.0 — Application to detect RF communication in a Z-Wave network

How to Update the Internal Simplicity Studio Copy of Simplicity Commander

Simplicity Commander, included with Simplicity Studio, is regularly updated with other Simplicity Studio updates. In general, to get the latest Simplicity Commander update, update Simplicity Studio. However, sometimes the release of a new Simplicity Commander to the Silicon Labs website (https://www.silabs.com/mcu/programming-options) lags behind the release of a new Simplicity Studio™ version. If the new version of Simplicity Commander includes a feature that is required for development then it is possible to replace the Simplicity Studio version with the following procedure.

First exit all running instances of Simplicity Studio and then download the appropriate version of Simplicity Commander from the above website.

The procedure for updating Simplicity Commander on the MacOS is slightly different and is described after the Windows and Linux instructions.

Windows and Linux (verified on Windows 10 and Ubuntu 18.04 LTS)

Two Simplicity Commander files in the current installation must be backed up as they have to be added back after the update. The default paths for the Simplicity Studio version of Simplicity Commander are:

Windows: `C:\SiliconLabs\SimplicityStudio\v4\developer\adapter_packs\commander`

Linux: `~/SimplicityStudio_v4/developer/adapter_packs/commander`

Back up the following two files:

1. apack.info
2. commander40x40.png

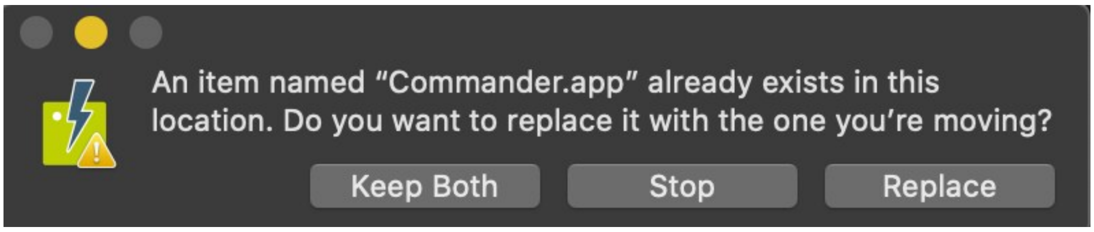
When the downloaded version of Simplicity Commander is expanded on Windows it is placed in a [Simplicity Commander] folder and on Linux in a [SimplicityCommander-Linux] folder. The Linux version will contain an additional tarball that needs to be expanded into a [commander] folder. The contents of that folder ([Simplicity Commander] on Windows or [commander] on Linux) should be used to replace the internal Simplicity Studio™ commander located at the above paths. Then copy in the two files that were backed up (apack.info and commander40x40.png) replacing the apack.info file.

MacOS Instructions (verified on MacOS Catalina version 10.15.4)

After the SimplicityCommander-Mac.zip is expanded it will be placed in a folder with a release note text file, a README.txt and the Commander .dmg file. When the .dmg file is opened it mounts the Commander application as a volume. The Commander.app can then be executed from that volume, dragged into the Applications folder or, in this case, used to replace the internal Simplicity Studio installation. Drag Commander.app into the folder containing the existing Commander.app:

```
/Applications/Simplicity\ Studio.app/Contents/Eclipse/developer/adapter_packs/commander
```

A popup will be displayed saying an item named "Commander.app" already exists with various options:


An item named “Commander.app” already exists in this location. Do you want to replace it with the one you’re moving?

Click **Replace**. Note: If Commander.app is dragged from the Applications folder it will move it out of Applications and not create a copy.

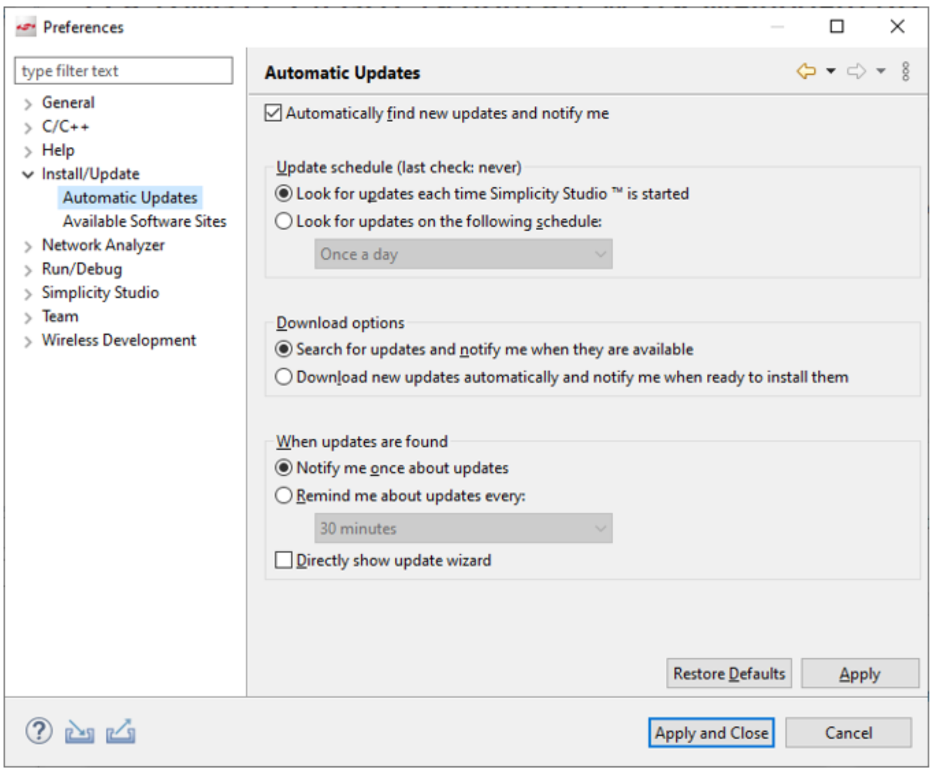
Conclusion

To verify the new version is being used, start Simplicity Studio and launch Simplicity Commander (from the Launcher perspective Tools icon (green wrench)) and select [Help] > [About Simplicity Commander] and the version will match the name of the .zip (Windows), .tgz (Linux) or .dmg (MacOS) file. Commander is also used behind the scenes for several operations including flashing images and various security commands.

Note there has been at least one report on a Mac that it had to be rebooted for this to work correctly. That should not normally be necessary, but it might if Simplicity Commander or Simplicity Studio did not shut down cleanly for some reason so it is worth trying.

Optionally Disable Automatic Updates

By default, SSv5 checks for updates when you first open it. You can manage update frequency through **Preferences > Install/Update > Automatic Updates**.


Preferences

type filter text

- General
- C/C++
- Help
- Install/Update
  - Automatic Updates
  - Available Software Sites
- Network Analyzer
- Run/Debug
- Simplicity Studio
- Team
- Wireless Development

Automatic Updates

[✓] Automatically find new updates and notify me

Update schedule (last check: never)
- [✓ Look for updates each time Simplicity Studio ™ is started]
- Look for updates on the following schedule:
  - Once a day

Download options
- [✓ Search for updates and notify me when they are available]
- Download new updates automatically and notify me when ready to install them

When updates are found
- [✓ Notify me once about updates]
- Remind me about updates every:
  - 30 minutes
- [ ] Directly show update wizard

Either uncheck **Automatically find new updates and notify me** or change the **Update Schedule** selection to use a schedule such as once a week.

If you have turned automatic updates off, you can check for updates by clicking **Install** on the toolbar, then **Manage Installed Packages**, and then **Check for Updates** on the Product Updates tab.

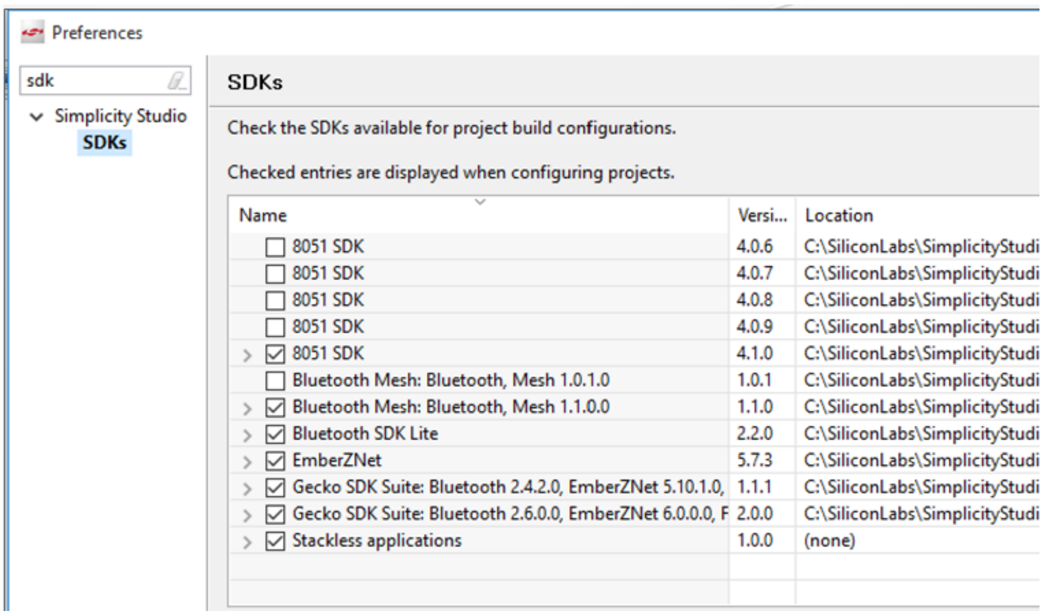
See About Update Frequency for details on the selections available.

Performance Enhancements

To improve SSv5 performance, try one or more of the following suggestions.

Disable SDKs not being used

If you have installed many different versions of different SDKs and GSDKs over time, go to **Preferences**, type `SDK` in the filter field, and uncheck the ones not currently being used.


Preferences

sdk

Simplicity Studio
SDKs

SDKs

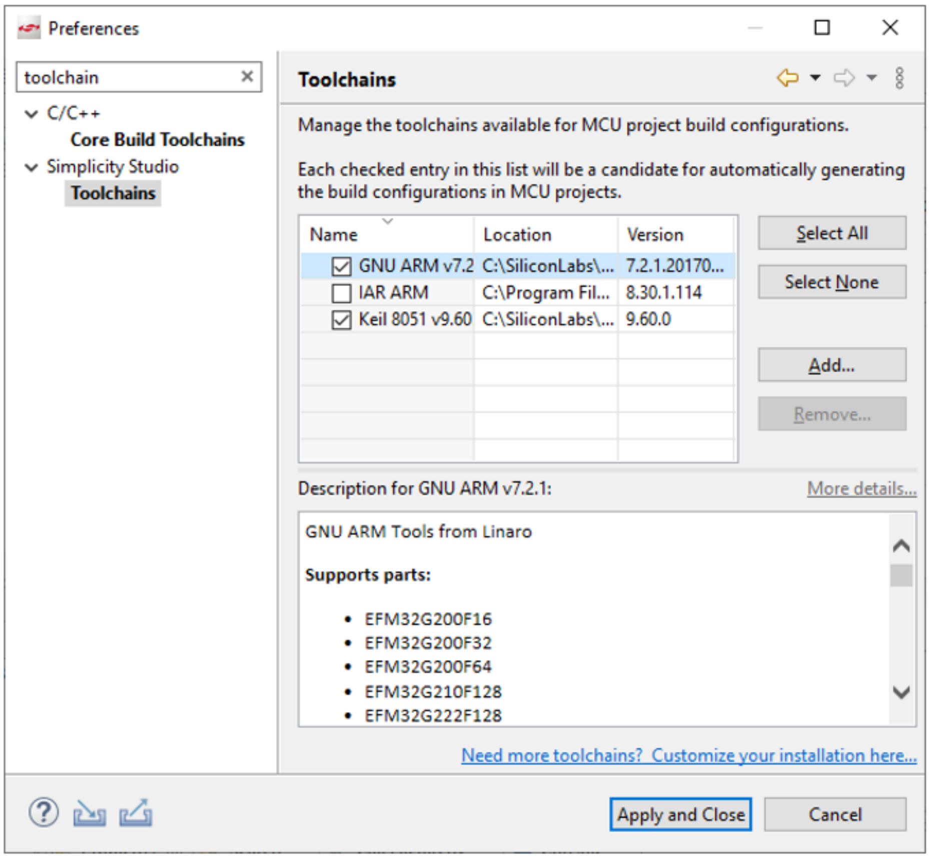
Check the SDKs available for project build configurations.

Checked entries are displayed when configuring projects.

Name	Versi...	Location
☐ 8051 SDK	4.0.6	C:\SiliconLabs\SimplicityStudi
☐ 8051 SDK	4.0.7	C:\SiliconLabs\SimplicityStudi
☐ 8051 SDK	4.0.8	C:\SiliconLabs\SimplicityStudi
☐ 8051 SDK	4.0.9	C:\SiliconLabs\SimplicityStudi
☑ 8051 SDK	4.1.0	C:\SiliconLabs\SimplicityStudi
☐ Bluetooth Mesh: Bluetooth, Mesh 1.0.1.0	1.0.1	C:\SiliconLabs\SimplicityStudi
☑ Bluetooth Mesh: Bluetooth, Mesh 1.1.0.0	1.1.0	C:\SiliconLabs\SimplicityStudi
☑ Bluetooth SDK Lite	2.2.0	C:\SiliconLabs\SimplicityStudi
☑ EmberZNet	5.7.3	C:\SiliconLabs\SimplicityStudi
☑ Gecko SDK Suite: Bluetooth 2.4.2.0, EmberZNet 5.10.1.0,	1.1.1	C:\SiliconLabs\SimplicityStudi
☑ Gecko SDK Suite: Bluetooth 2.6.0.0, EmberZNet 6.0.0.0, F	2.0.0	C:\SiliconLabs\SimplicityStudi
☑ Stackless applications	1.0.0	(none)

Disable unused targets or toolchains

If you have installed multiple toolchains but are not using all of them, or are not using all of the targets, disable the unused ones. Go to **Preferences** and type `toolchain` or `target` in the filter field, and uncheck any unused items.


Preferences

toolchain

C/C++
Core Build Toolchains
Simplicity Studio
Toolchains

Toolchains

Manage the toolchains available for MCU project build configurations.

Each checked entry in this list will be a candidate for automatically generating the build configurations in MCU projects.

Name	Location	Version
☑ GNU ARM v7.2	C:\SiliconLabs\...	7.2.1.20170...
☐ IAR ARM	C:\Program Fil...	8.30.1.114
☑ Keil 8051 v9.60	C:\SiliconLabs\...	9.60.0

Description for GNU ARM v7.2.1:

GNU ARM Tools from Linaro

Supports parts:

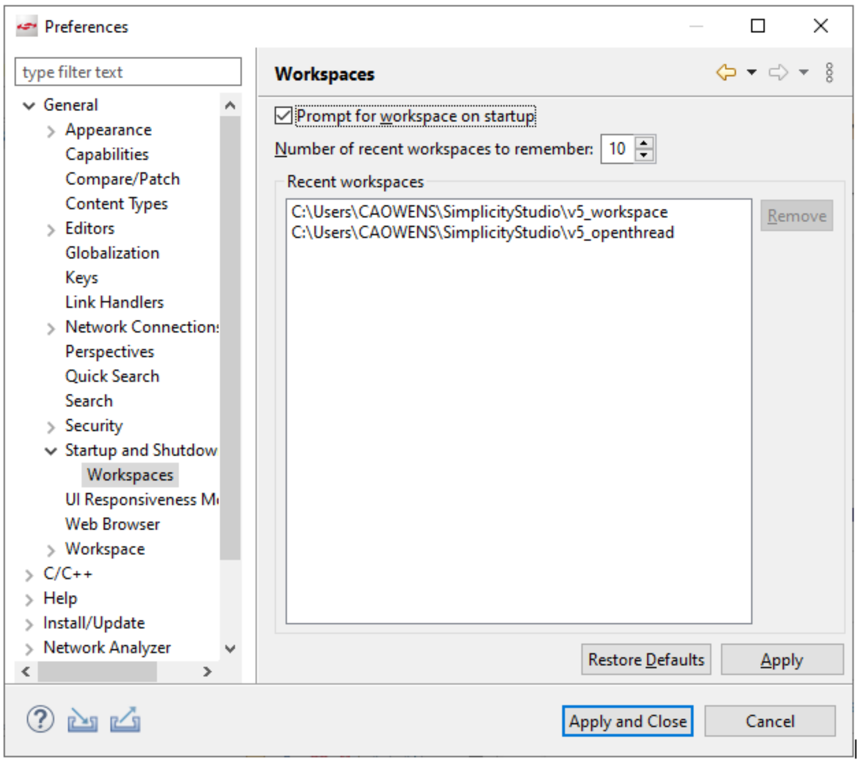
- EFM32G200F16
- EFM32G200F32
- EFM32G200F64
- EFM32G210F128
- EFM32G222F128

Need more toolchains? Customize your installation here...

Use different workspaces for different GSDK versions or compilers

Overloading a workspace with projects and different versions of SDKs can slow down SSv5 operations. Keep separate workspaces for specific project and/or SDK versions and/or toolchains. Note that this is necessary if you are using different versions of the IAR compiler to match different versions of the GSDK.

While you can use **File > Switch Workspaces** to change workspaces as described in Project Explorer View, if you will be working with different workspaces, you may wish to have SSv5 prompt you on startup about which workspace to use. Go to **Preferences > General > Startup and Shutdown > Workspaces** and check **Prompt for workspace on startup**.



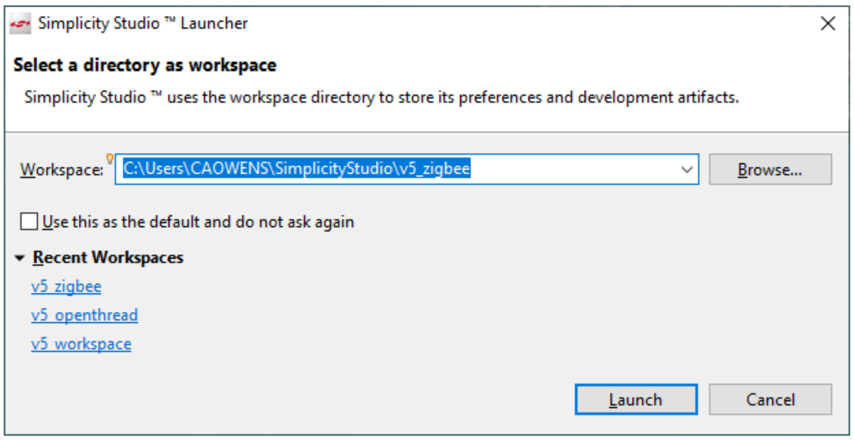
☑ Prompt for workspace on startup

Number of recent workspaces to remember: 10

Recent workspaces

C:\Users\CAOWENS\SimplicityStudio\v5_workspace
C:\Users\CAOWENS\SimplicityStudio\v5_openthread

Then on startup you can select a workspace or create a new workspace by clicking **Browse** and adding a directory.



Select a directory as workspace

Simplicity Studio ™ uses the workspace directory to store its preferences and development artifacts.

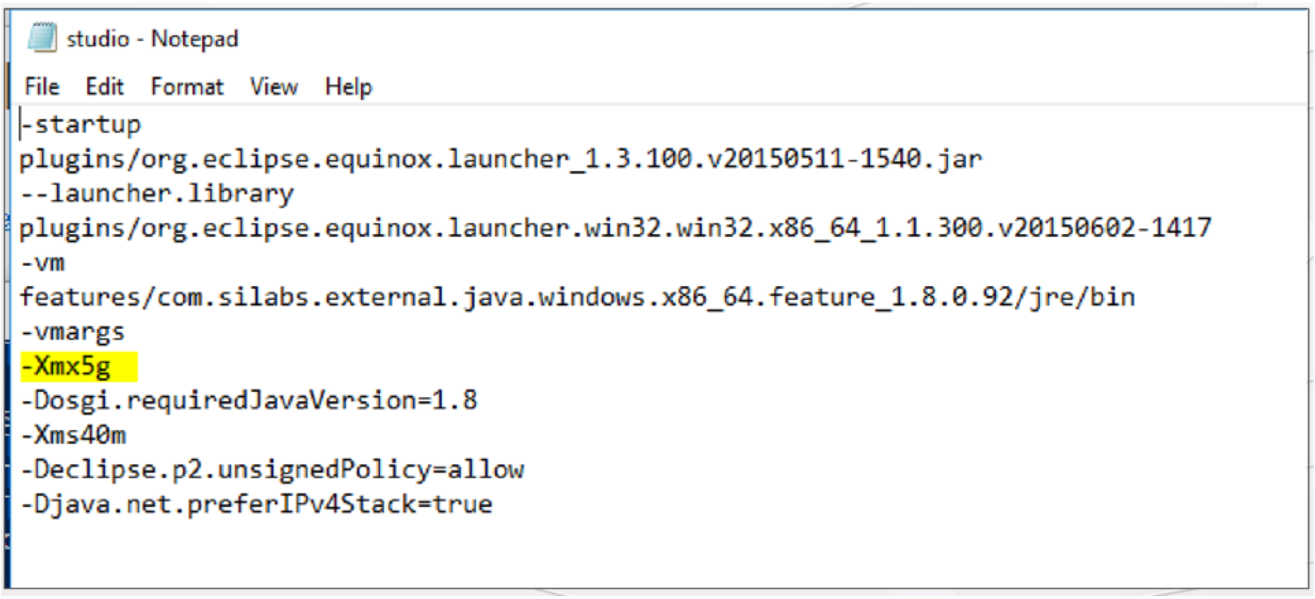
Workspace: C:\Users\CAOWENS\SimplicityStudio\v5_zigbee

☐ Use this as the default and do not ask again

Increase heap size

Close SSv5. Use a text editor to open following file within the Simplicity Studio root path, for example: *C:\SiliconLabs\SimplicityStudio\v5\studio.ini*

In the following figure, the line `Xmx5g` was added to increase the maximum Java heap SSv5 will use to 5 GBytes. Choose larger or smaller depending on your system resources. Without that line SSv5 defaults to a maximum of one quarter of the system RAM. On a system with 8 GBytes of RAM that would be 2 GBytes and SSv5 might run out of Java Heap space on memory-intensive operations.



```
-startup
plugins/org.eclipse.equinox.launcher_1.3.100.v20150511-1540.jar
--launcher.library
plugins/org.eclipse.equinox.launcher.win32.win32.x86_64_1.1.300.v20150602-1417
-vm
features/com.silabs.external.java.windows.x86_64.feature_1.8.0.92/jre/bin
-vmargs
-Xmx5g
-Dosgi.requiredJavaVersion=1.8
-Xms40m
-Declipse.p2.unsignedPolicy=allow
-Djava.net.preferIPv4Stack=true
```

SDK installation or updates are not working

The cause might be your IT department using proxy servers, and potentially also man-in-the-middle security monitoring. See the following knowledge base articles for more information on how to work around these situations:

Configuring Simplicity Studio to Work with Proxy Servers

Installing SSL Security Certificates for Simplicity Studio Updates

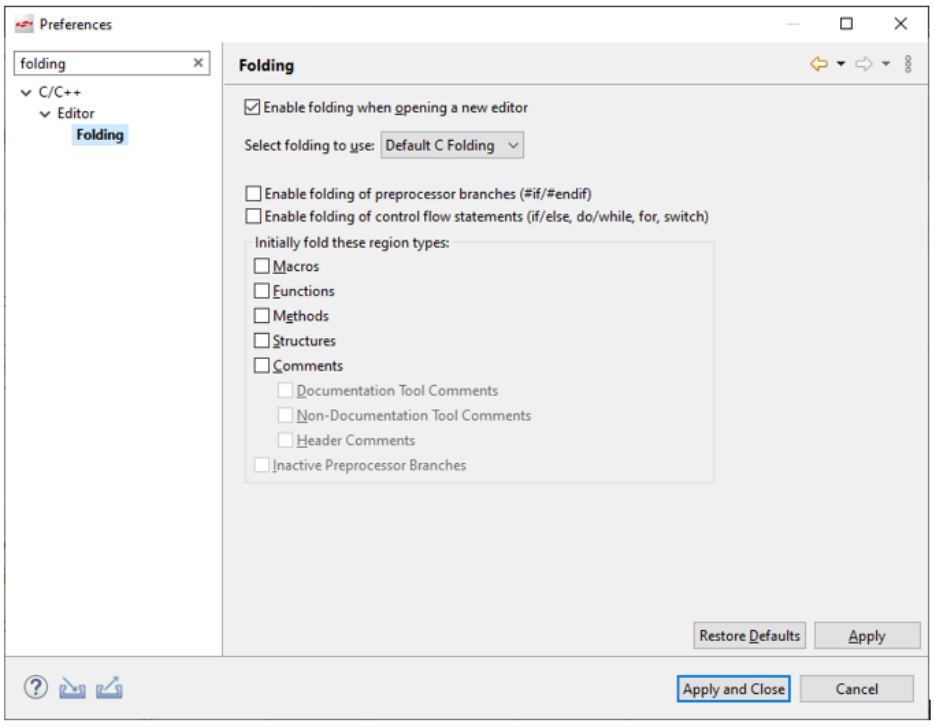
Speed up debug session startup

Go to Preferences > Run/Debug > Launching, and under General Options uncheck **Build (if required) before launching**.

This will interfere with your workflow if you are accustomed to making a change and clicking the Debug button to build the project before launching the project.

Speed up reading large source files

Enable folding of #if/#ifdef's and if/else, do/while, for, and switch statements in source code. Go to **Preferences** and type `folding` in the filter field, and check the three "Enable folding ... " options.



Preferences

folding

C/C++
Editor
Folding

Folding

- [✓] Enable folding when opening a new editor

Select folding to use: Default C Folding

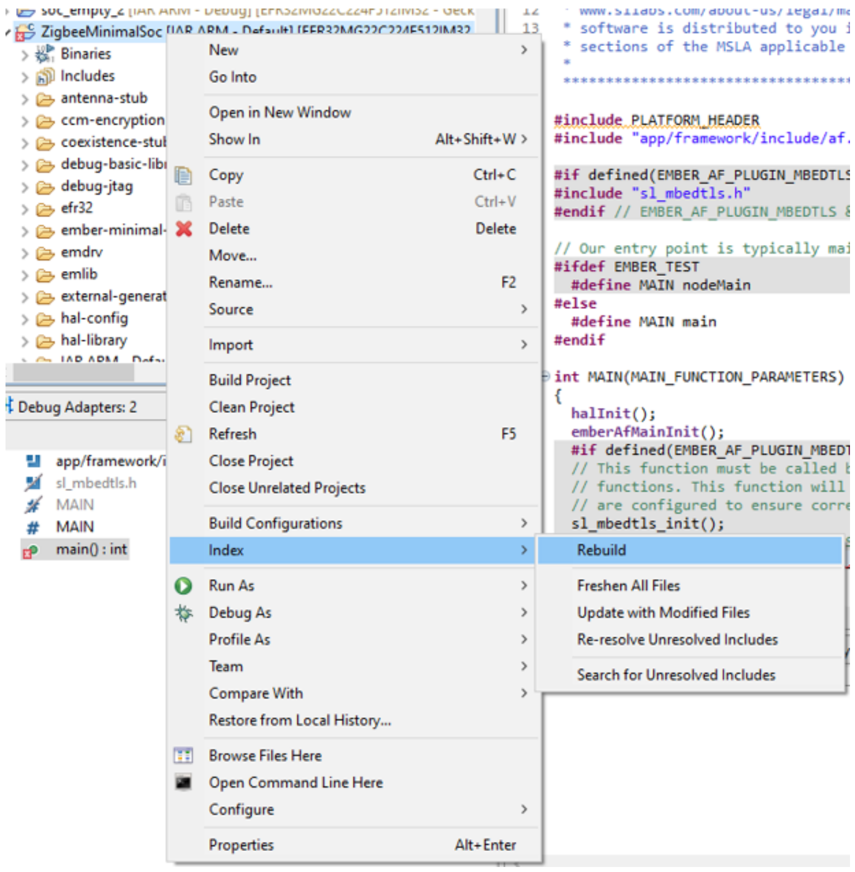
- [ ] Enable folding of preprocessor branches (#if/#endif)
- [ ] Enable folding of control flow statements (if/else, do/while, for, switch)

Initially fold these region types:

- [ ] Macros
- [ ] Functions
- [ ] Methods
- [ ] Structures
- [ ] Comments
  - [ ] Documentation Tool Comments
  - [ ] Non-Documentation Tool Comments
  - [ ] Header Comments
- [ ] Inactive Preprocessor Branches

Speed up the indexer

If the Indexer is taking a long time to run, close other projects. If still slow, in the Project Explorer view right-click the project directory and select **Index > Rebuild**.



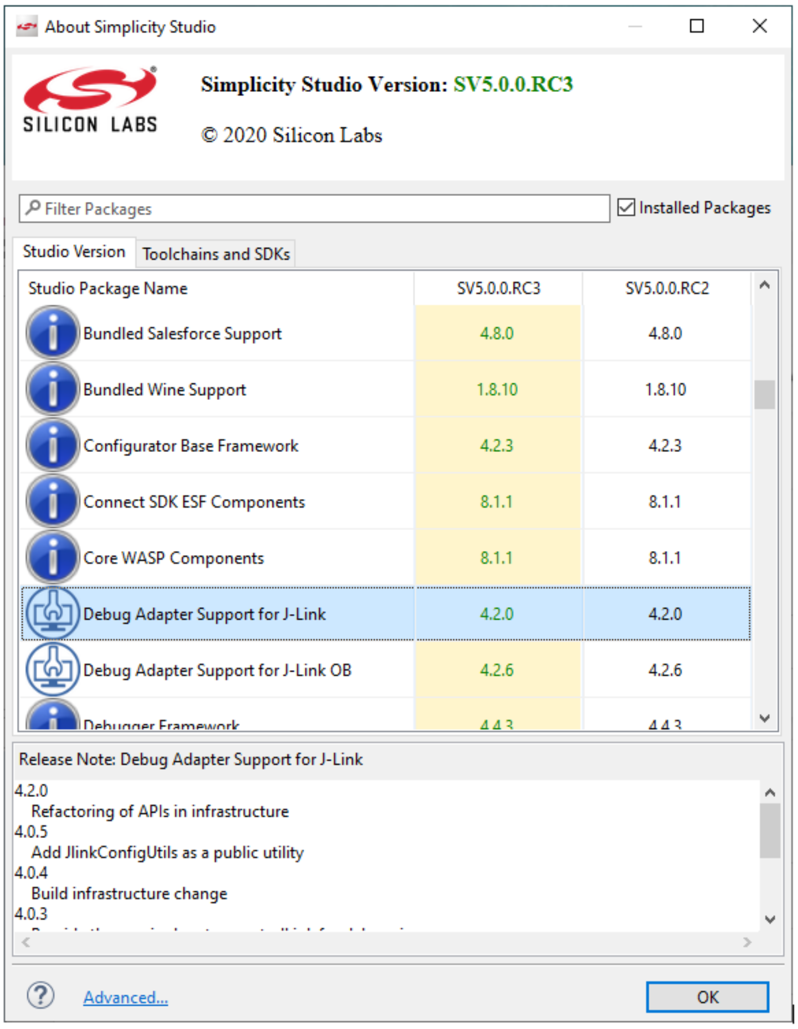
```
 * www.silabs.com/about-us/legal/ma
 * software is distributed to you i
 * sections of the MSLA applicable
 *
 ***********************************

#include PLATFORM_HEADER
#include "app/framework/include/af.

#if defined(EMBER_AF_PLUGIN_MBEDTLS
#include "sl_mbedtls.h"
#endif // EMBER_AF_PLUGIN_MBEDTLS &

// Our entry point is typically mai
#ifdef EMBER_TEST
  #define MAIN nodeMain
#else
  #define MAIN main
#endif

int MAIN(MAIN_FUNCTION_PARAMETERS)
{
  halInit();
  emberAfMainInit();
  #if defined(EMBER_AF_PLUGIN_MBEDT
  // This function must be called b
  // functions. This function will
  // are configured to ensure corre
  sl_mbedtls_init();
```

Restore a perspective layout

To reset a perspective to its original layout, right-click the perspective button in the toolbar and select **Reset**.

Find a Version

Go to **Help > About Simplicity Studio** for version information. The overall SSv5 version is at the top of the dialog. The Studio Version tab lists version information for all the components that make up SSv5. The Toolchains and SDKs tab lists version information for the installed SDKs and toolchains.



About Simplicity Studio

Simplicity Studio Version: SV5.0.0.RC3

© 2020 Silicon Labs

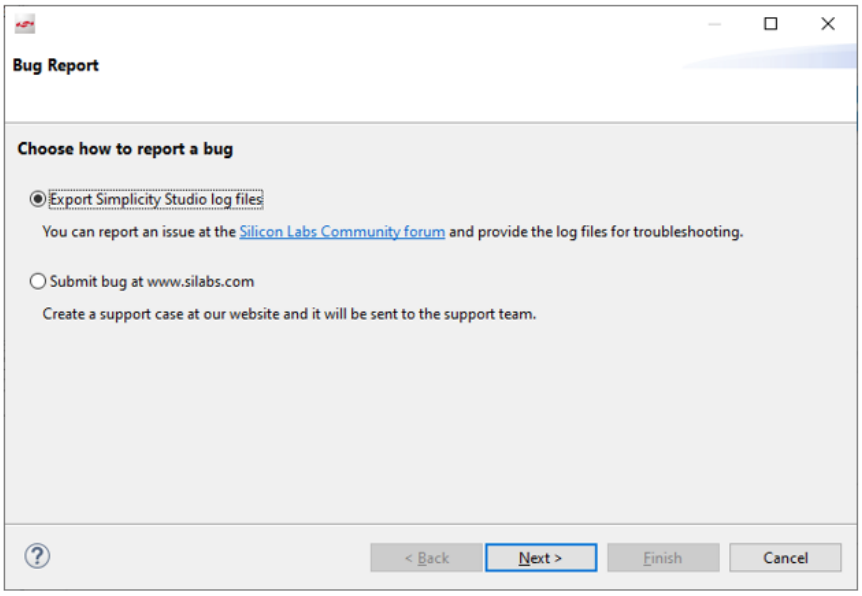
Studio Package Name	SV5.0.0.RC3	SV5.0.0.RC2
Bundled Salesforce Support	4.8.0	4.8.0
Bundled Wine Support	1.8.10	1.8.10
Configurator Base Framework	4.2.3	4.2.3
Connect SDK ESF Components	8.1.1	8.1.1
Core WASP Components	8.1.1	8.1.1
Debug Adapter Support for J-Link	4.2.0	4.2.0
Debug Adapter Support for J-Link OB	4.2.6	4.2.6
Debugger Framework	4.4.3	4.4.3

Release Note: Debug Adapter Support for J-Link

```
4.2.0
  Refactoring of APIs in infrastructure
4.0.5
  Add JlinkConfigUtils as a public utility
4.0.4
  Build infrastructure change
4.0.3
```

Report a Bug

Go to **Help > Report bug ...**. The default selection generates a log file to the location you specify. This is useful if you need to attach a log to a case that's already created.



Bug Report

Choose how to report a bug

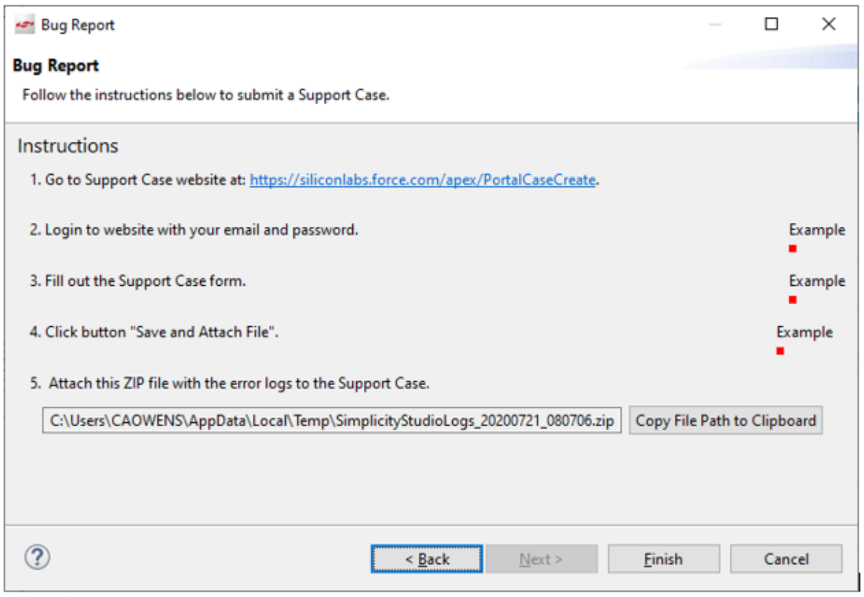
● Export Simplicity Studio log files

You can report an issue at the Silicon Labs Community forum and provide the log files for troubleshooting.

○ Submit bug at www.silabs.com

Create a support case at our website and it will be sent to the support team.

Select **Submit bug at Silabs.com** if you are creating a new ticket. This generates the log file and provides instructions on creating a ticket.



Bug Report

Bug Report

Follow the instructions below to submit a Support Case.

Instructions

1. Go to Support Case website at: https://siliconlabs.force.com/apex/PortalCaseCreate.
2. Login to website with your email and password. Example
3. Fill out the Support Case form. Example
4. Click button "Save and Attach File". Example
5. Attach this ZIP file with the error logs to the Support Case.

C:\Users\CAOWENS\AppData\Local\Temp\SimplicityStudioLogs_20200721_080706.zip

Capture a Simplicity Studio 5 Thread Dump

If Simplicity Studio 5 happens to hang (become unresponsive for several minutes) during an operation, the best way to report this to Silicon Labs is to use a Java tool called jstack to capture a thread dump while the program is still in the hung state. This thread dump can be used by the Silicon Labs team to analyze the hang. If Simplicity Studio 5 is shut down and restarted, a thread dump at that point will not be useful. The jstack tool is included in the Simplicity Studio 5 installed Java Runtime Environment (JRE).

Once you have generated the thread dump file according to one of the following procedures, attach it to a Silicon Labs support case along with other details of what operation was being performed right before the hang as well as any screenshots taken of what was shown in the application. A support case can be created from the Silicon Labs Simplicity Studio Community home page or with this url: https://siliconlabs.force.com/s/contactsupport.

The procedure to use the jstack tool for Windows, MacOS and Linux is described in the following sections.

Windows

1. Open Windows Task Manager and go to the Details tab to find the PID for Simplicity Studio (studio.exe).
2. Open a command prompt and issue the following command, adjusting it based on the Simplicity Studio installation path:

`[SIMPLICITY_STUDIO_INSTALLATION]\features\com.silabs.external.java.windows.x86_64.feature_11.0.5\jre\bin\jstack.exe PID > [PATH]\SSv5ThreadDump.txt`

For example, with the default installation path:

`C:\SiliconLabs\SimplicityStudio\v5\features\com.silabs.external.java.windows.x86_64.feature_11.0.5\jre\bin\jstack.exe -l -e 41068 > C:\temp\SSv5ThreadDump.txt`

MacOS

1. Open a terminal window.
2. Use the following command to get the PID:

`ps aux | grep studio`

This is an example of the output with the pid **76698**:

```
USERNAME@mac0010128 ~ % ps aux | grep studio
USERNAME 76698   5.6  9.7 12491892 1633396   ??  S    Tue09AM  92:52.70 /Applications/SimplicityStudio5.app/Contents/MacOS/studio
USERNAME 19420   0.0  0.0  4268300    680 s000  R+    3:26PM   0:00.00 grep studio
```

1. Use the PID with the jstack command

`[SIMPLICITY_STUDIO_INSTALLATION]/Contents/Eclipse/jre/Contents/Home/bin/jstack -l -e PID > [PATH]/SSv5ThreadDump.txt`

For example

`/Applications/Simplicity\ Studio.app/Contents/Eclipse/jre/Contents/Home/bin/jstack -l -e 76698 > ~/Documents/SSv5ThreadDump.txt`

Linux

1. Open a terminal window.
2. Use the following command to get the PID:

`ps aux | grep studio`

This is an example of the output with the pid **10233**:

```
ThinkPad-W530:~$ ps aux | grep studio
USERNAME     10233 10.1 27.3 110505488 2115904 ?    Sl   12:55  10:58 /home/USERNAME/SimplicityStudio_v5//jre/bin/java -
Dosgi.requiredJavaVersion=11.0.5 -Xms40m -Declipse.p2.unsignedPolicy=allow -Djava.net.preferIPv4Stack=true -Djxbrowser.ipc.external=true -
jar /home/USERNAME/SimplicityStudio_v5//plugins/org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar -os linux -ws gtk -arch x86_64 -
showsplash -launcher /home/USERNAME/SimplicityStudio_v5/studio -name Studio --launcher.library
/home/USERNAME/SimplicityStudio_v5//plugins/org.eclipse.equinox.launcher.gtk.linux.x86_64_1.1.1100.v20190907-0426/eclipse_1801.so -startup
/home/USERNAME/SimplicityStudio_v5//plugins/org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar --launcher.overrideVmargs -exitdata
a78007 -vm /home/USERNAME/SimplicityStudio_v5//jre/bin/java -vmargs -Dosgi.requiredJavaVersion=11.0.5 -Xms40m -
Declipse.p2.unsignedPolicy=allow -Djava.net.preferIPv4Stack=true -Djxbrowser.ipc.external=true -jar
/home/USERNAME/SimplicityStudio_v5//plugins/org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar
```

1. Use the PID with the jstack command:

`[SIMPLICITY_STUDIO_INSTALLATION]/jre/bin/jstack -l -e PID > [PATH]/SSv5ThreadDump.txt`

For example:

`~/SimplicityStudio_v5/jre/bin/jstack -l -e 10233 > ~/Documents/SSv5ThreadDump.txt`

Convert an Existing Simplicity Studio 5 Project from GCC to IAR

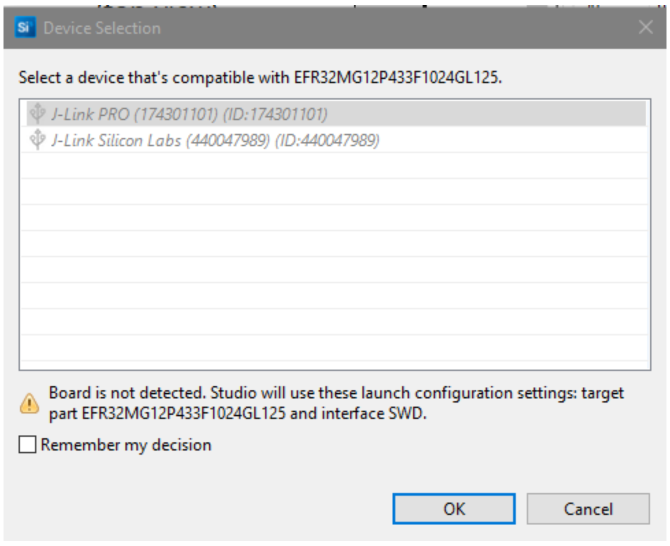
1. Create a new IAR build configuration:
    - Right-click the project folder and select Build Configurations > Manage.
    - In the resulting Manage Build Configurations dialog, click **New**.
    - Select the IAR ARM build configuration to use (IAR ARM Default recommended) and click **OK**.
2. Still in the Manage Build Configurations dialog, select the new build configuration and click **Set Active**.
3. Open the project's .slcp file and, on the OVERVIEW tab, click **Edit** on the Project Generators card.
4. Select IAR EMBEDDED WORKBENCH PROJECT and click **Save**.
   For Bluetooth projects steps 5 through 7 are also necessary. For Platform and Flex projects proceed to step 8.
5. In the Project Explorer view:
    - Right-click the project folder and select Properties.
    - In the resulting Properties dialog, expand C/C++ Build and select Settings.
    - Under Tool Settings select IAR Linker for ARM > Library and select all of the libraries with GCC or gcc in the name or path and also the 'gcc', 'c', 'm' and 'nosys' libraries.
    - Click **Delete** in the dialog menu.
    - Click **Apply and Close**.
6. In the Project Explorer view:
    - Expand gecko_sdk_n.n.n > platform > Device > SiliconLabs > <target-part> > Source.
    - Right-click the GCC folder and select Resource Configurations > Exclude from Build...
    - In the resulting dialog, check the box for the IAR ARM ... build configuration.
    - Click **OK**
7. Check if the project contains gecko_sdk_n.n.n > service > udelay > src > sl_udelay_armv6m_gcc.s. If it does, exclude it from the build with the same command as in the previous step.
8. Build the project. It should be successful. If not, check if any other GCC-specific files exist that need to be excluded from the build configuration.

Use Custom Boards

Connect the board either using an external Segger J-Link Debug Adapter or a Silicon Labs mainboard, with Debug Mode set to OUT. (In the Debug Adapters view, right-click the device, select **Device Configuration**, and change the Debug Mode on the Adapter Configuration tab). Create the project with the chip part number set in the Target, SDK, and Toolchain Selection dialog.

You cannot use Flash Programmer until the debug adapter's device configuration has been configured with the part. Use Simplicity Commander instead.

Projects based on custom boards can be built and flashed as part of a debug session. Simplicity Studio then displays and allows you to confirm the target settings to use for the debug session.


Device Selection

Select a device that's compatible with EFR32MG12P433F1024GL125.

J-Link PRO (174301101) (ID:174301101)
J-Link Silicon Labs (440047989) (ID:440047989)

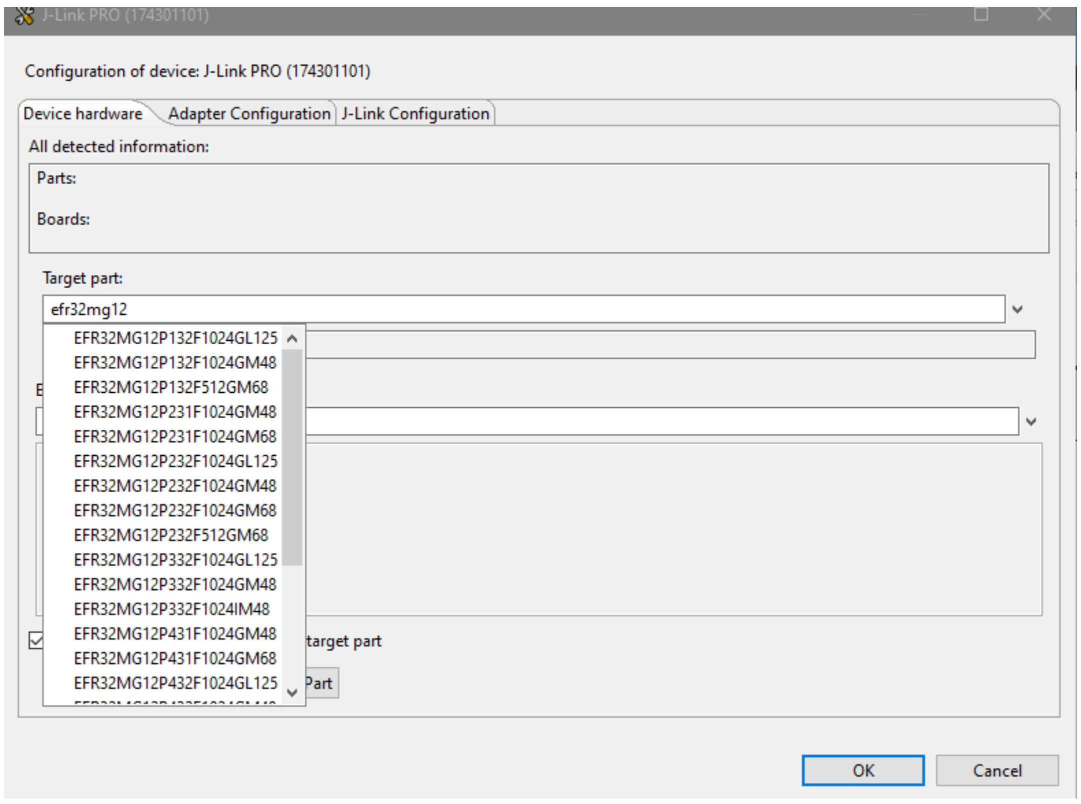
Board is not detected. Studio will use these launch configuration settings: target part EFR32MG12P433F1024GL125 and interface SWD.

Remember my decision

OK Cancel

You can change the debug adapter's device configuration by right-clicking the device in the Debug Adapters view, selecting **Device Configuration**, and making either of these changes:

1. On the Device Hardware tab, enter the target part number.


J-Link PRO (174301101)

Configuration of device: J-Link PRO (174301101)

Device hardware | Adapter Configuration | J-Link Configuration

All detected information:

Parts:

Boards:

Target part:

efr32mg12

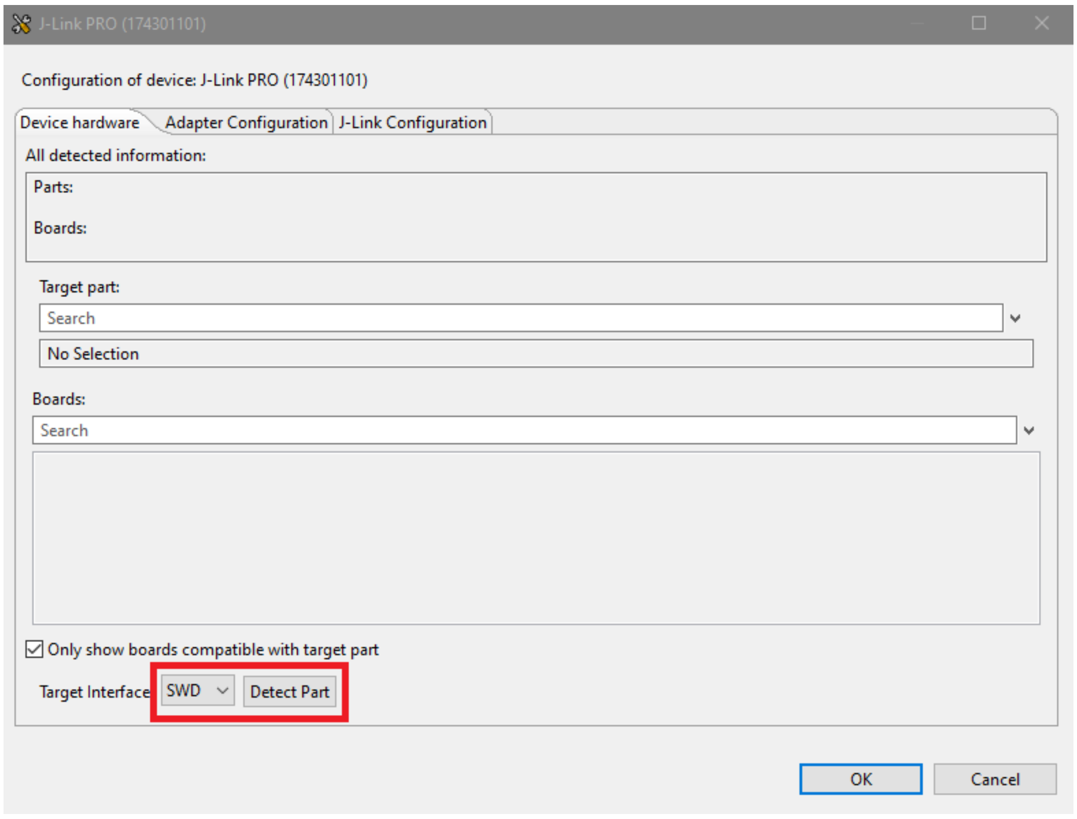
EFR32MG12P132F1024GL125
EFR32MG12P132F1024GM48
EFR32MG12P132F512GM68
EFR32MG12P231F1024GM48
EFR32MG12P231F1024GM68
EFR32MG12P232F1024GL125
EFR32MG12P232F1024GM48
EFR32MG12P232F1024GM68
EFR32MG12P232F512GM68
EFR32MG12P332F1024GL125
EFR32MG12P332F1024GM48
EFR32MG12P332F1024IM48
EFR32MG12P431F1024GM48
EFR32MG12P431F1024GM68
EFR32MG12P432F1024GL125

target part

Part

OK Cancel

1. On the Device Hardware tab, change the Target Interface to **SWD**. Click **Detect Part**.



Configuration of device: J-Link PRO (174301101)

All detected information:

Parts:

Boards:

Target part: No Selection

Boards:

☑ Only show boards compatible with target part

Target Interface: SWD

Detect Part

Discover Network Devices

Sometimes Simplicity Studio may fail to discover a device connected on a local network. For devices with EFR32/EM3xx/Si917 MCUs, this can be addressed through the **Preferences > Simplicity Studio > Device Manager > TCP/IP Adapters** settings. Enabling more network scanning, for example by selecting the **Include all network interfaces. (Use when connected to VPN)** checkbox on that dialog, can flood the network with undesired discovery requests. The preferred method is by discovering the specific device.

- On the TCP/IP Adapters dialog, check **Use TCP discovery for individual addresses**.
- In the **Discovery Subnet Configuration** text box, enter the individual IP address.

Place a Project Under Source Control

Note that not all of these folders will be present in all projects. Which ones are present depends on if the project is a Project Configurator-based project (has a .slcp file) as well as what operations have been done with the project, especially in the Project Configurator.

Hidden Folders in SSv5 IDE Projects:

Files / Folder	Purpose	Under Source Control?
.pdm	Tracks project changes compared to the original project template used during project creation, normally from a software example.	Yes - very important
.cproject & .project files	Eclipse project files, contains project settings for the different build configurations such as Includes, Defines, and compiler options.	Yes

Files / Folder	Purpose	Under Source Control?
.settings	Tracks project-level preferences for Eclipse and Simplicity Studio.	Probably - but some might only apply to the local developer
.trash	Keeps a copy of Project Configurator files or folders for software components that were deleted from the project and had been changed from the default values. The files are kept in case the component is added back, in which case the last used settings for the component are restored. Can also contains user-copied/supplied source files, for example if the user adds their own source files to the project and then deletes them.	No
.uceditor	Keeps a list of software components that were edited on this computer.	No
.projectlinkstore	Contains quick links for commonly used files or resources. The listed links are updated based on how often a user opens a file or resource.	No

General Source Control Information for Project Configurator Projects:

File / Folder	Purpose	Under Source Control?
.slcp, .slps and .pintool	The main Project Configurator files.	Yes
config	Contains the configuration header files for all of the software components used in the project. Changes made in the header files are reflected in the Project Configurator and vice versa. Also contains subfolders with Advanced Configurator (AC) configuration files. For example, Bluetooth projects have a 'btconf' subfolder with the gatt_configuration.btconf file and possibly 'in_place_ota_dfu.xml' file.	Yes - Cannot be recreated
autogen	Contains files created by the Project Configurator based on the installed software components and the files in the 'config' folder. The files are necessary to build the project, but they will be recreated on project generation.	Normally yes, but with some flexibility depending on the workflow of the team (that is, do they always do a generation after updating the project).
gecko_sdk_x.y.z	Created by the Project Configurator based on the installed components. It will be recreated on project generation.	Same as for the 'autogen' folder
Build output folders ('GNU ARM...', 'IAR ARM...' or 'Keil 8051...')	Created when the project is built.	Not normally

Revision History

Revision History

Revision 5.8.0 released December 13, 2023

- Overview: Update new features and known issues
- Getting Started: Add Extension upgrades to Upgrade an Existing Project
- Companion IDEs: Update VSCode Project Generation
- Tools: Network Analyzer: Update custom decoders

Revision 5.7.2 released October 9, 2023

- Overview: Update known issues
- Getting Started: Update the Zigbee Cluster Configurator to ZCL Advanced Platform
- Developing with Project Configurator: Zigbee Cluster Configurator
  - Rename to ZCL Advanced Platform
  - Update user interface illustrations
  - Add information on use with Matter

Revision 5.7.1 released August 17, 2023

- Overview: Update known issues

Revision 5.7.0 released June 7, 2023

- Overview: Update known issues and new features
- Getting Started: Updated first page to reflect that extensions are now installed as part of the normal install process, also update images and the Extensions section
- Developing with Project Configurator: Update Proprietary Radio Configurator
- Companion IDEs - VS Code: Update for current interface and functionality

Revision 5.6.3 released March 16, 2023

- Overview:
  - Add release dates to New Features
  - Update Known Issues
- Getting Started: Restructure and update Solutions and Memory Editor content
- Developing with Project Configurator:
  - Update Pin Tool to reflect new interface
  - Update Solutions section
  - Add Memory Editor section
- Using the Tools: NCP Commander - Update Bluetooth mesh section
- Tips and Tricks:
  - Place a Project Under Source Control (update)
  - Discover Networked Devices (new)

Revision 5.6.0 released December 14, 2022

- Getting Started:
  - Generate project report
  - Software Components tab: quality dropdown, PIN tool button

  - Configurator updates
- Developing with Project Configurator:
  - Update Zigbee Cluster Configurator to include Matter projects
  - Update Wi-SUN Configurator for Fan 1.1 changes
- Building and Flashing: Add new post-build editor section
- Add new *Companion IDEs: Visual Studio Code* section.
- Tips & Tricks: Added a new tip about getting an updated Simplicity Commander tool.

Revision 5.5.0 released September 7, 2022

- Getting Started:
  - Changed Update and Install SDK Software to reflect the modification for the Matter extension.
  - Updated for the new approach to installing extensions.
  - Added a section on updating a project to a newer GSDK version.
- About the Launcher:
  - Revised the Toolbar > Install section, SDK tab to show the new interface for updating or adding SDKs.
  - Changed Preferred SDK information in the Welcome and Device tabs section.
- Testing and Debugging:
  - Changed to reflect that the GNU Debugger is now the default.
  - Added a section on using the Wireshark Packet Analyzer.

Revision 5.4.2 released August 17, 2022

- Getting Started: Updated the New Project Wizard interface, removed outdated content; documented the IDE / toolchain options when creating a new project.
- Using the Tools:
  - Updated the Bluetooth Direction Finding Tool Suite section.
  - Added a logic analyzer section to Energy Profiler
- Tips and Tricks: Added instructions for placing a project under source control
- Throughout - Clarified that AppBuilder is a legacy tool.

Revision 5.4.0 released June 8, 2022

- Getting Started: Updated Project Configurator Overview tab; added summary sections for Wi-SUN Configurator and beta Memory Editor.
- About the Launcher: Documented new filter options for example applications.
- Project Configurator: Updated Overview tab; added new Wi-SUN Configurator section.
- Testing and Debugging: GNU Debugger support is now GA.
- Tips and Tricks: Added a tip for installing all parts of an upgrade.

Revision 5.3.2 released March 9, 2022

- Getting Started: Added more information about linking sources in a new project.
- About the Launcher: Clarified the interface overview, with new information about the perspectives menu; added instructions on adding an SDK to the list shown on the General Information card.
- NCP Commander: Added information about the scripting feature.
- Tips and Tricks: Added a tip about working with custom radio boards.

Revision 5.3.1 released January 26, 2022

- Getting Started: Updated Start a Project to reflect other project types available
- NCP Commander: Updated to include Bluetooth mesh functionality
- Using the tools: New section on the Bluetooth Direction Finding Tool Suite
- Various maintenance updates.

Revision 5.3.0 released December 15, 2021

- Installation instructions reflect installing the Gecko SDK from GitHub.
- Project migration text includes newer migration options.
- New SDK Extensions section.
- New Solutions section.
- Project Configurator now supports Zigbee, Bootloader, and Z-Wave.
- New Zigbee Cluster Configurator tool.
- NCP Commander enhanced with dynamic GATT database interactions.
- Tips and Tricks updated with new installation information.

Revision 5.2.0 released June 16, 2021

- Version changed to correlate with Simplicity Studio 5.
- Tools: Standalone NCP Commander documented.
- Getting Started: Linux installation instructions added.
- Energy Profiler: Code correlation software setup instructions updated.
- Tips and Tricks: Links added at the top of the page; tip on changing from GCC to IAR compiler added.
- Various maintenance updates.

Revision 1.1.0 released December 9, 2020

- General updates for Simplicity Studio 5.1.
- Launcher perspective (Getting Started, About the Launcher):
  - Software Examples and Demos tabs combined into a single tab, enhanced filtering options.
  - Examples from GitHub repositories.
  - New security firmware update option for Series 2 devices.
- Developing for 32-Bit Devices:
  - Project Configurator now supports Bluetooth Mesh and 32-Bit MCU SDKs.
  - New Project Configurator Configuration Tools tab.
  - New Bluetooth Mesh Node Configurator tool.
  - Pin Tool, new 'Add Component' functionality.
- Testing & Debugging:
  - New Beta quality GNU Debugger.
- Tools:
  - New Bluetooth NCP Commander tool.
- Tips & Tricks:
  - Update protocol locations for Bluetooth Mesh.
  - New 'Capture a Simplicity Studio 5 Thread Dump' tip.

Revision 1.0.0 released July 29, 2020

- Initial release