



# QSG106: Getting Started with EmberZNet PRO

---

This quick start guide provides basic information on configuring, building, and installing applications for the EM35x and Mighty Gecko (EFR32MG) family of SoCs using the EmberZNet PRO stack v. 5.7 and higher and Simplicity Studio.

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

## KEY FEATURES

---

- Product overview
- Setting up your development environment
- Discovering your SDK
- Working with example applications.

# 1 Product Overview

Before following the procedures in this guide you must have

- Purchased your development hardware:
  - Mighty Gecko (EFR32MG) Mesh Networking Kit
  - or
  - EM35x Development Kit
- Registered your kit on the Silicon Labs website. This gives you access to a portal from which you can download the EmberZNet PRO stack and other Silicon Labs software, and obtain support.
- Downloaded the required software components. A card included in your development hardware kit contains a link to a Getting Started page, which will direct you to links for the Silicon Labs software products.

## 1.1 Software Components

See the stack release notes for version restrictions and compatibility constraints for the stack and these component. To develop EmberZNet PRO applications, you will need the following:

- The Simplicity Studio development environment, which incorporates AppBuilder. AppBuilder is an interactive GUI tool that allows you to configure a body of Silicon Labs-supplied code to implement applications. Online help for AppBuilder and other Simplicity Studio modules is provided.
  - Simplicity Studio operates on top of an application framework, a working software design that you can modify to meet your needs. It provides common basic services and, when combined with device-specific code, provide you with most of the over-the-air behavior for your device. Some common usage case examples based on the relevant application framework are provided along with a pre-built application, Node Test, for functional testing of RF modules.
  - The hardware abstraction layer (HAL) acts as a conduit between the network stack and the node processor and radio. Separating network stack functionality from the specific hardware implementation enables easy portability. HAL code is provided as a combination of pre-built libraries for complex, stack-critical functionality and C source code that you can alter in order to customize, extend, or reduce device functionality across various hardware platforms. The HAL API is documented in the online API reference.
- The EmberZNet PRO stack, an advanced implementation of a ZigBee stack. The stack API is documented in online API reference as well as in other documents installed with the stack installer, or available through the development environment. The stack is delivered as a collection of libraries that you can link to your applications. A description of each library is provided in the development environment. The release notes contain details on the folders installed along with their contents.
- IAR Embedded Workbench for ARM (IAR-EWARM) 7.30, used as a compiler in the Simplicity Studio development environment. Download the supported version from the Silicon Labs Support Portal. Refer to the “QuickStart Installation Information” section of the IAR installer for additional information about the installation process and how to configure your license.

Although you will not need them for the tasks in this Getting Started guide, you may wish to become familiar with the manufacturing utilities available for your environment. The utilities are hardware-dependent.

- For both the EM35x Development Kit and the Mighty Gecko Mesh Networking Kit, the ISA3 utilities, downloaded from the support portal. The installer modifies your PATH environment variable so that the command line utilities can be easily executed from a Windows Command Prompt. See UG107, *EM3x Utilities Guide: For the EM3x Platform*, for detailed information on the EM3x utilities.
- For the Mighty Gecko Mesh Networking Kit, Simplicity Commander, installed along with Simplicity Studio in a subfolder of the install folder. See UG162, *Simplicity Commander Reference Guide*, for more information.

Finally, if you are working with an EM35x development kit and you want to use the USB interface of the breakout board for UART connectivity, download a driver for the FTDI USB <-> Serial converter from <http://www.ftdichip.com/Drivers/VCP.htm>

## 1.2 Support

Users can access the Silicon Labs support portal at <https://www.silabs.com/support>. Use the support portal to contact Customer Support for any questions you might have during the development process.

## 1.3 Documentation

The stack installer provides a documentation index (in documentation/index.htm and also linked from a Start Menu entry) that contains links to documentation locations and brief descriptions of each document's purpose. Simplicity Studio provides links to hardware documentation and other application notes. See the release notes for details on other documentation available.

## 2 Setting Up Your Development Environment

### 2.1 Install Third-Party Tools

Install third-party tools, such as IAR Embedded Workbench for ARM (see section Software Components for the list).

### 2.2 Install your Silicon Labs Stack or Software Development Kit

Install your Silicon Labs software (see section Software Components for more information).

### 2.3 Connect your Hardware

Connect your development hardware to the PC on which you will install Simplicity Studio. By having it connected when Simplicity Studio installs, Simplicity Studio will automatically obtain the relevant additional resources it needs.

#### 2.3.1 EFR32 Wireless Starter Kit (WSTK)

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

**Note:** For best performance in Simplicity Studio, be sure that the power switch on your WSTK is in the Advanced Energy Monitoring or “AEM” position (see figure below).

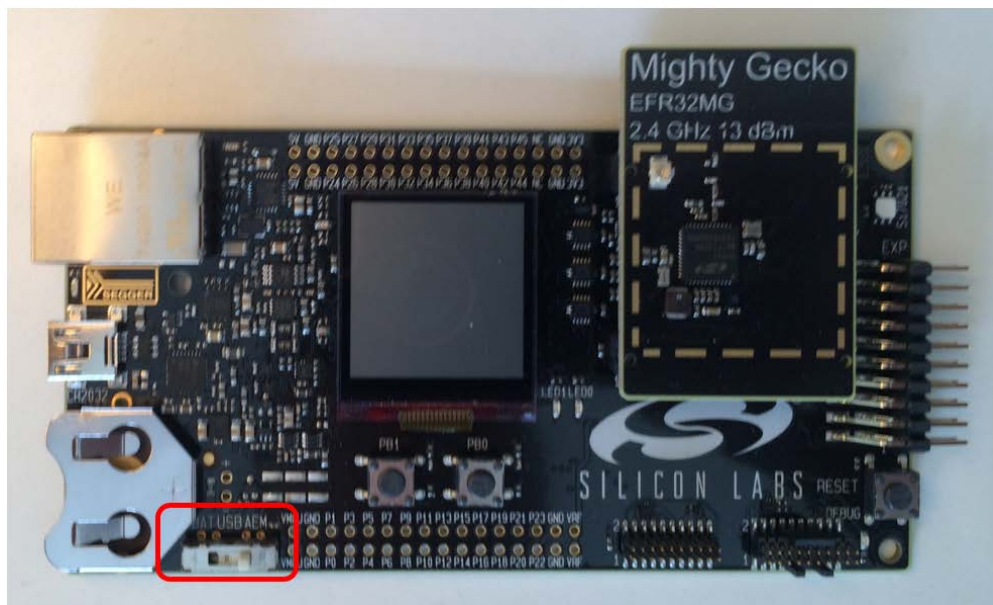


Figure 1. EFR32MG on a WSTK

#### 2.3.2 EM35x Development Kit

Follow the instructions on the Quick Start Guide included in the development kit to set up a development environment connected to your computer.

## 2.4 Install Simplicity Studio

During installation, Simplicity Studio obtains updates and additional packages specific to your connected hardware.

1. As soon as Simplicity Studio launches, it searches for updates. This operation can take several minutes.

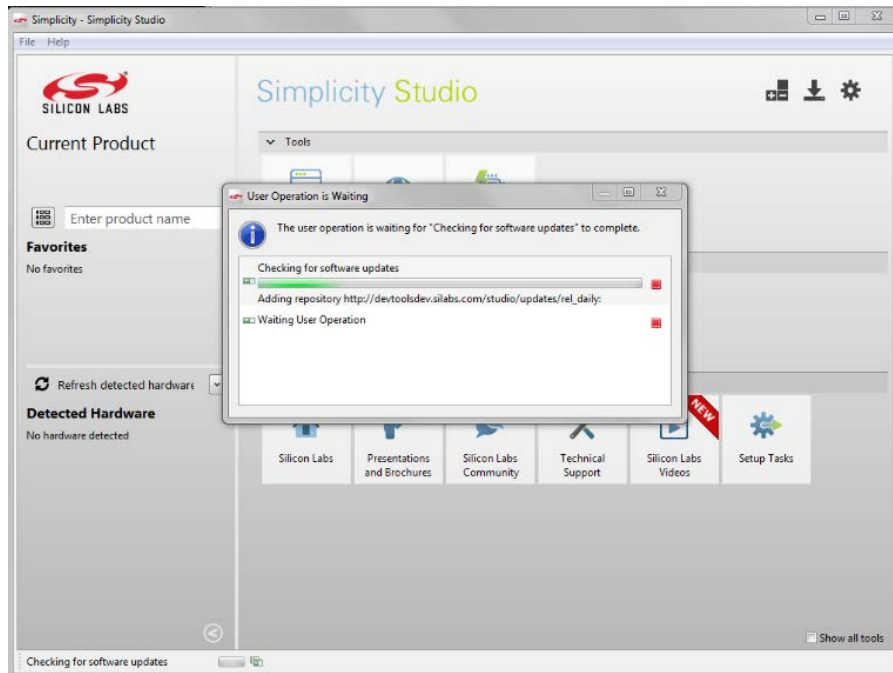


Figure 2. Checking for Software Updates

2. Once software update is complete, Simplicity Studio checks for connected hardware. If you have not connected your WSTK, you are prompted to do so. Connect your hardware and, when the screen changes to show that the hardware has been found, click **[Finish]**.

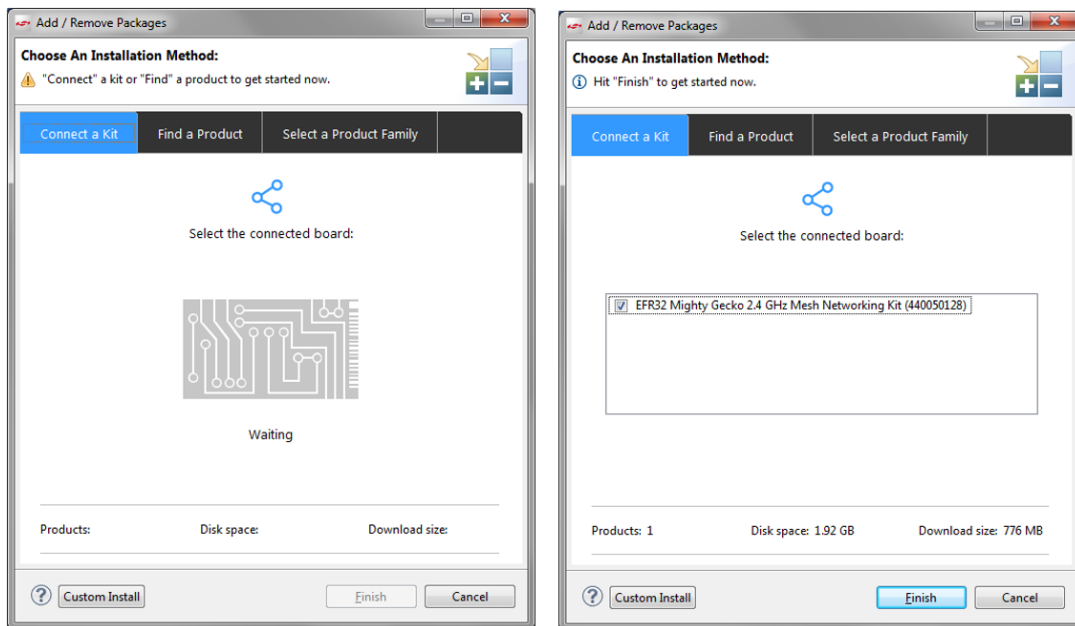


Figure 3. Connect a Kit Before and After Connection

3. Simplicity Studio then installs software packages related to your connected hardware, as shown in the figure below. This procedure can take some time, during which the green progress indicator may appear stationary. However, the update steps above the progress indicator are continuously refreshed.

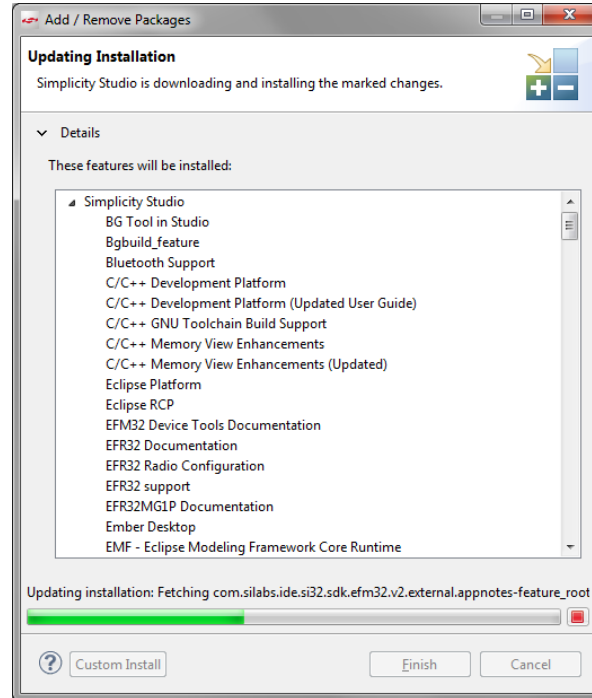


Figure 4. Installation Update

4. After update is complete, restart Simplicity Studio.
5. Once restart is complete, a menu of setup tasks is displayed. Select **Initial Setup** and then click **[Launch setup]**.

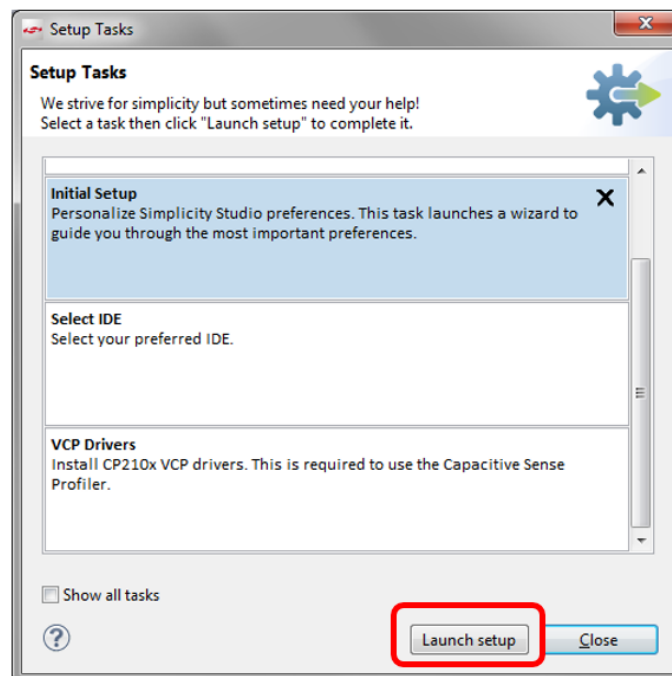


Figure 5. Setup Tasks

6. A kit selection dialog is displayed. Your connected kit should be selected. Click **[Next >]**.

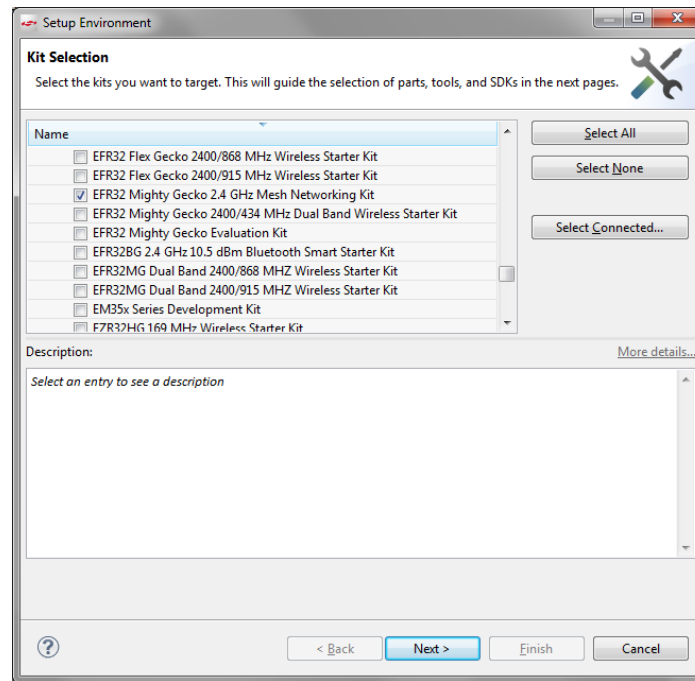


Figure 6. Kit Selection Dialog

7. A part selection dialog is displayed. Your connected part should be selected. Click **[Next >]**.

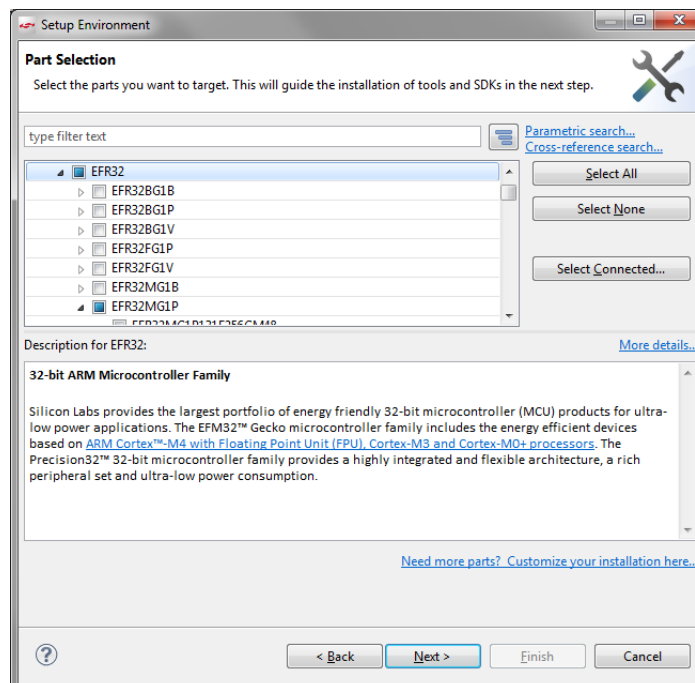


Figure 7. Part Selection Dialog

8. A Build Environment dialog (shown below) is displayed that shows detected items. If a Toolchain or SDK is not shown, you can click **Add...** to configure it now, or configure it later from the Settings control. Adding the EmberZNet PRO stack SDK is described in section Discovering the EmberZNet PRO Stack. Click **[Finish]**. The Simplicity perspective, described in the next section, is displayed.

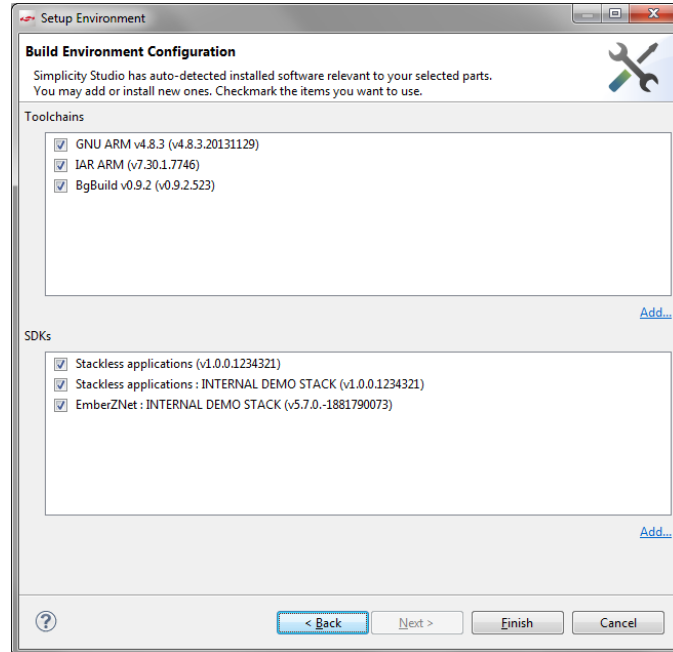


Figure 8. Build Environment Configuration

9. If you plan to work with NCP software communicating with a host MCU/PC, you should upgrade your WSTK board controller firmware to build 435 or later. Click the Kit Manager tile in the Simplicity perspective and click **[Yes]** in the confirmation dialog.

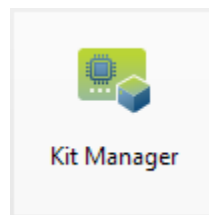


Figure 9. Kit Manager Tile

10. Your firmware is automatically updated to the latest version and displayed in the Kit Manager window once download is complete. Click **[Close]** to return to the Simplicity perspective.

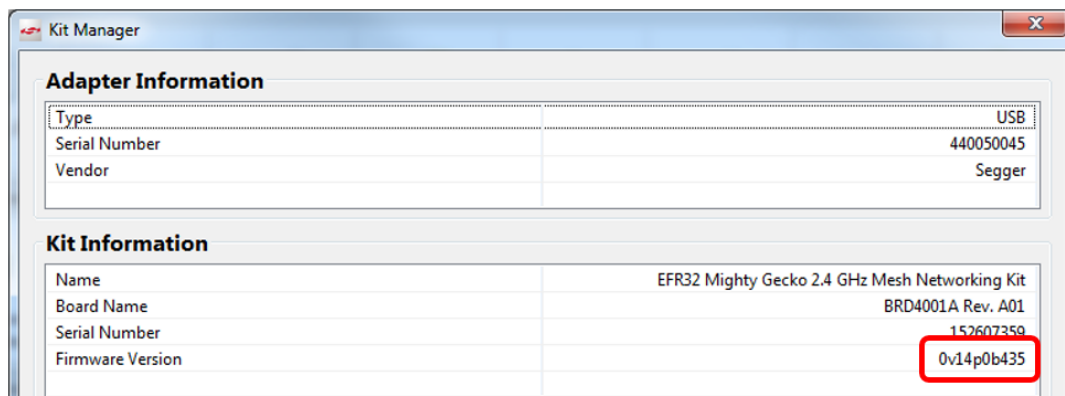


Figure 10. Kit Manager with Latest Board Controller Firmware

## 2.5 Navigation in Simplicity Studio

Simplicity Studio is built on the Eclipse platform. As such, it is broken up into different “perspectives,” each of which allows access to a specific set of functionality. Simplicity Studio starts up in the “Simplicity perspective,” sometimes referred to as the “Home Screen”.

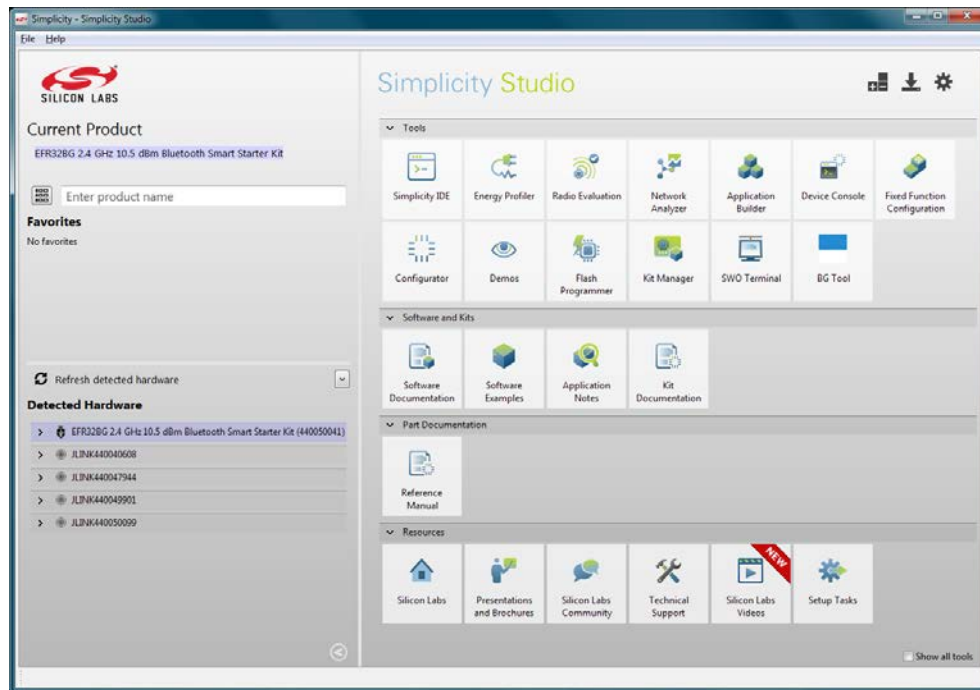


Figure 11. Simplicity Studio's Simplicity Perspective

From the Simplicity perspective, you can discover devices, configure Simplicity Studio, or navigate to another perspective for application development.

The Simplicity perspective shows large tile icons that represent the various sets of functionality within Simplicity Studio. Clicking a tile opens a different perspective. When you are in a different perspective, you can always return to the Simplicity perspective or any other perspective at any time by clicking on one of the tile icons in the top right-hand corner of your screen.

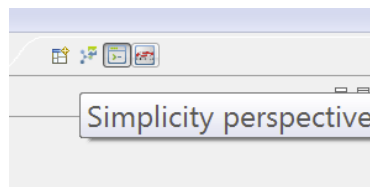


Figure 12. Navigation Tile Icons

Your detected hardware is shown in the **Detected Hardware** panel in the lower left of the Simplicity perspective. If you change connected hardware and it does not auto-detect, click **[Refresh Detected Hardware]**.

Three controls in the upper right allow you to maintain part-specific packages, install updates to the core Simplicity Studio software, and change configuration settings.



Figure 13. Maintenance Tools



### 3 Discovering the EmberZNet PRO Stack

If you are discovering from the initial setup process **Add ...** step (Figure 8), go to step 4. If you are discovering the SDK after initial setup, begin with step 1.

1. Click the Settings icon on the top right-hand corner of the Simplicity perspective to open the Preferences window.



Figure 14. Settings Icon

2. In the Preferences window's left frame, click **Simplicity Studio > SDKs** to open the configuration dialog.

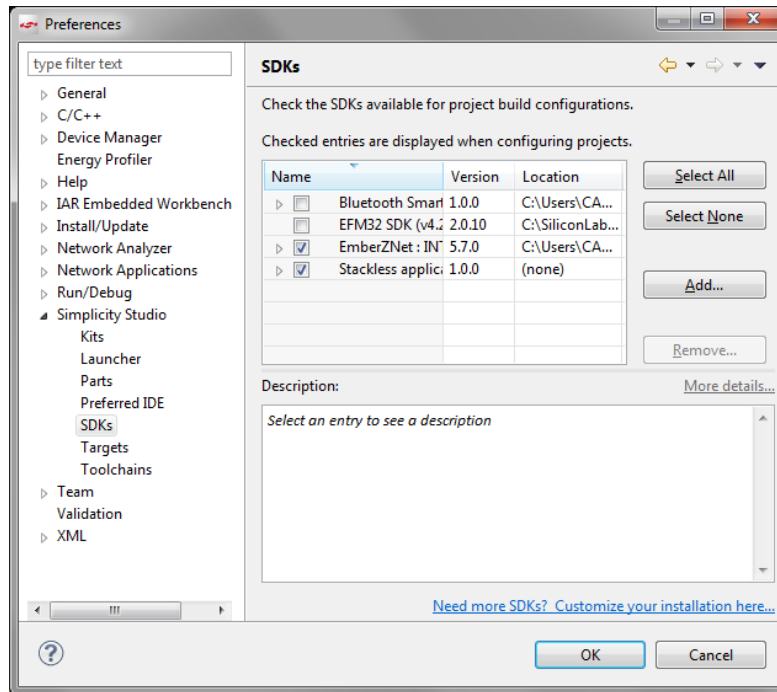
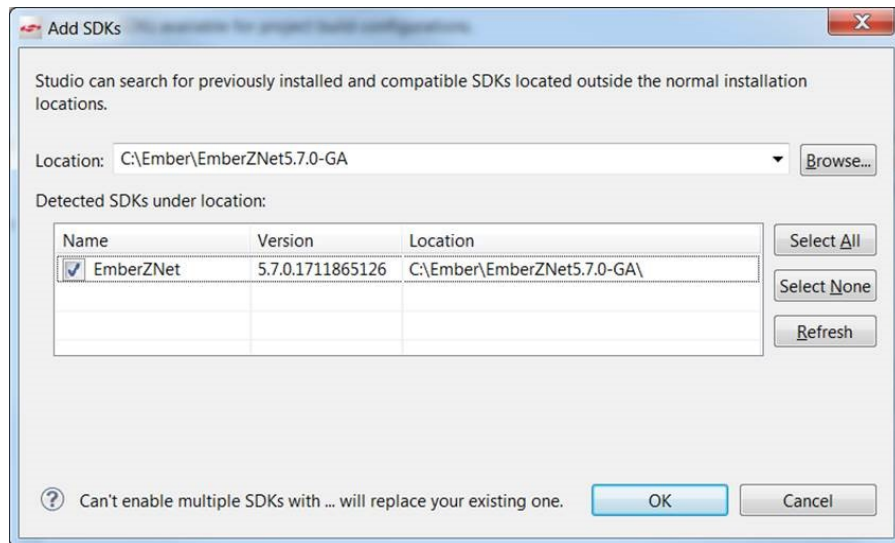


Figure 15. Silicon Labs SDK Configuration

3. Click **[Add]**.
4. Browse to the folder where you have installed the stack.
5. Click **[OK]**. The folder is scanned for required information.

6. Verify the software has detected the EmberZNet SDK and click **[OK]**.



**Figure 16. Add SDKs**

7. Click **[OK]** to return to the starting configuration page.
8. Click **[Finish]** (from initial configuration) or **[OK]** (from the settings icon).

## 4 Working with Example Applications

When working with example applications in Simplicity Studio, you will execute the following steps:

1. Select an example application. To demonstrate the connectivity features of a network, you may need to build two or more different example applications, such as a client and a server application or the Home Automation example scenario involving a gateway, a light, and a switch.
2. Generate application files.
3. Compile and flash the application to the radio board.
4. Interact with the application.

These steps are described in detail in the following sections.

### 4.1 Selecting an Example Application

1. In the Simplicity perspective, click the Software Examples tile.

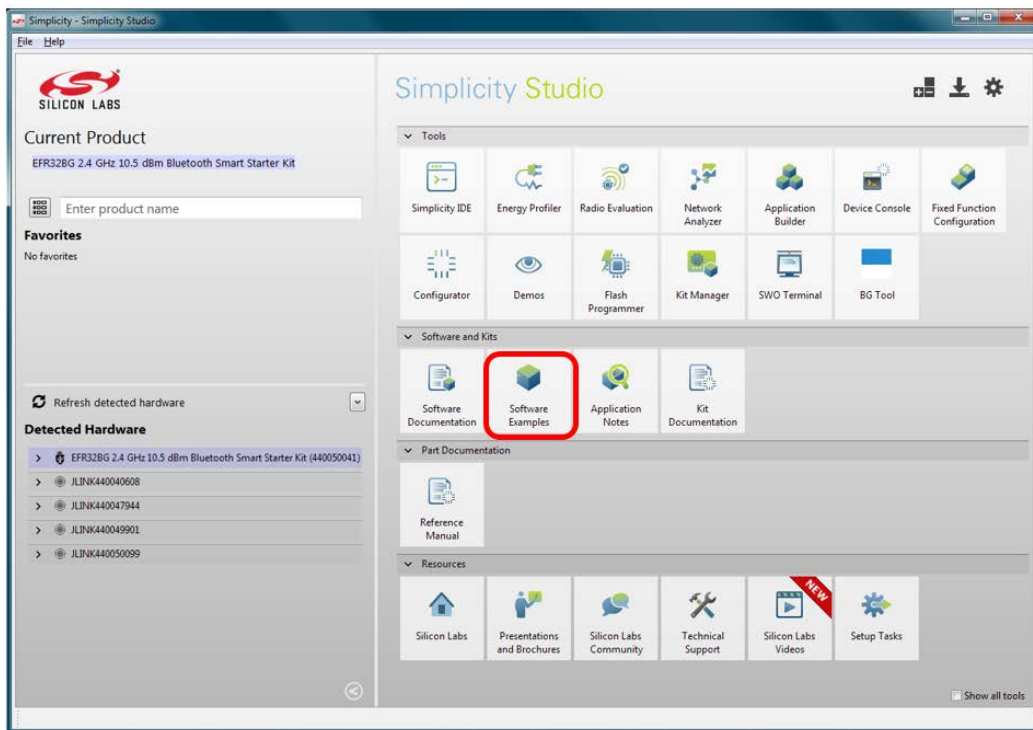


Figure 17. Software Examples Tile

- In the New Example dialog shown below, enter the kit, part, and SDK. Kit and part will be filled by Simplicity Studio if the hardware was detected and displayed in the Simplicity perspective (Figure 11).

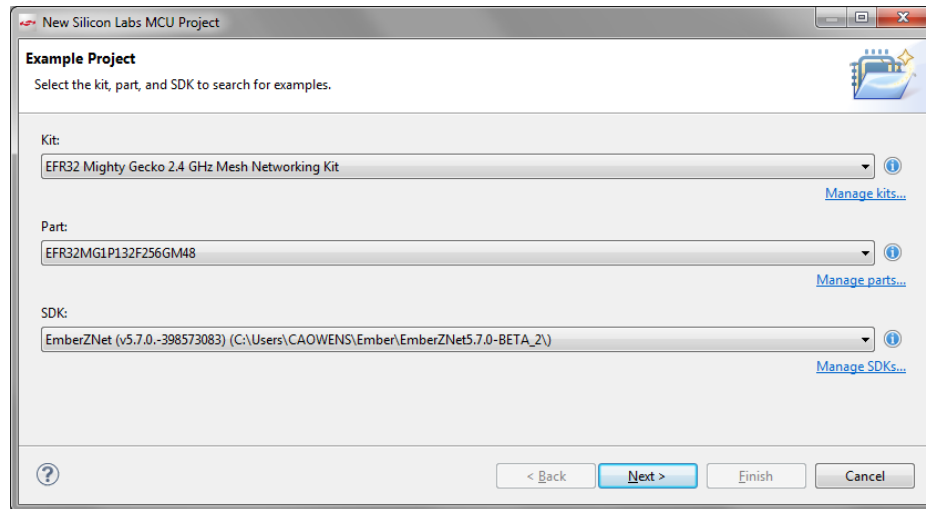


Figure 18. New Example Project Dialog

- Click **[Next]**.
- Select an example from the list. For the purposes of this guide, select HASampleLight. Click **[Next >]**.

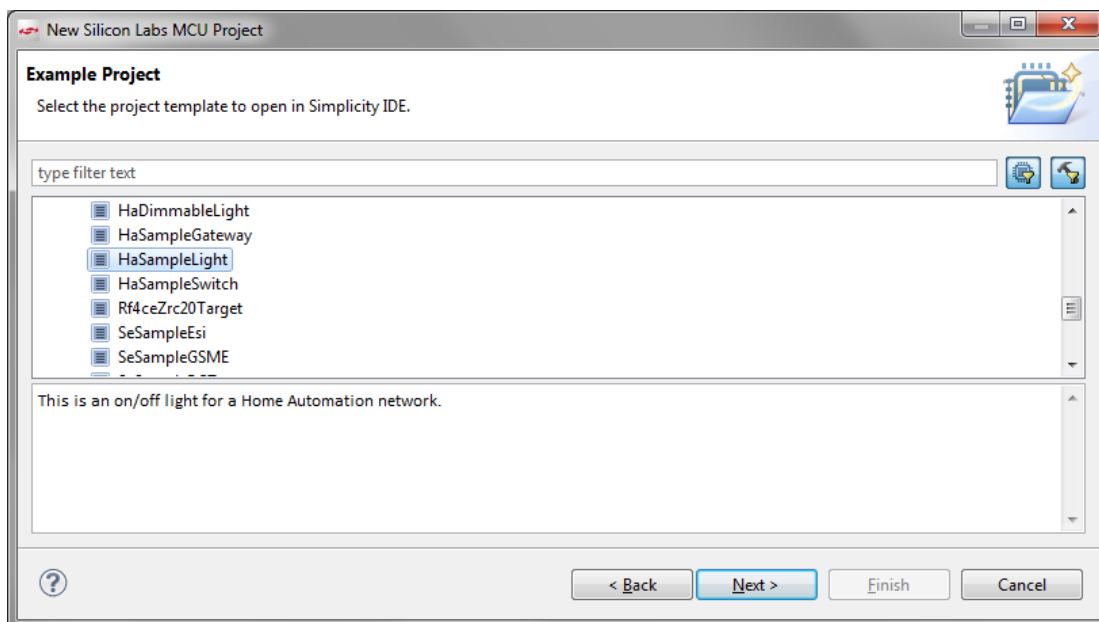


Figure 19. Example Project List

5. In the Project Configuration dialog, change the name of the project if you want, and click **[Finish]**. AppBuilder creates an example project and opens with the project in the AppBuilder General Tab (shown later in Figure 23).

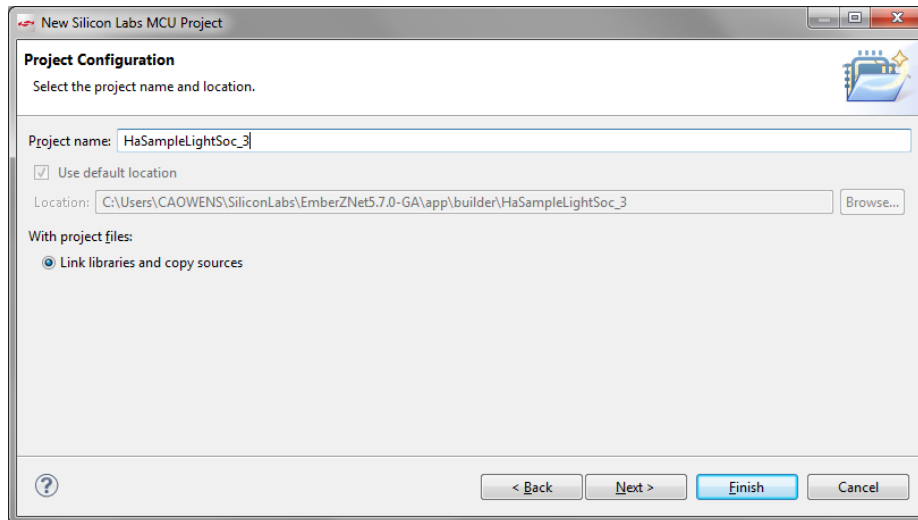


Figure 20. Project Configuration

6. (Optional) If you would like to interact with your application through the serial console as instructed in section Interacting with your Example Application, do the following two steps:
  1. On the HAL configuration tab, change **Debug Level** from Off to Normal.

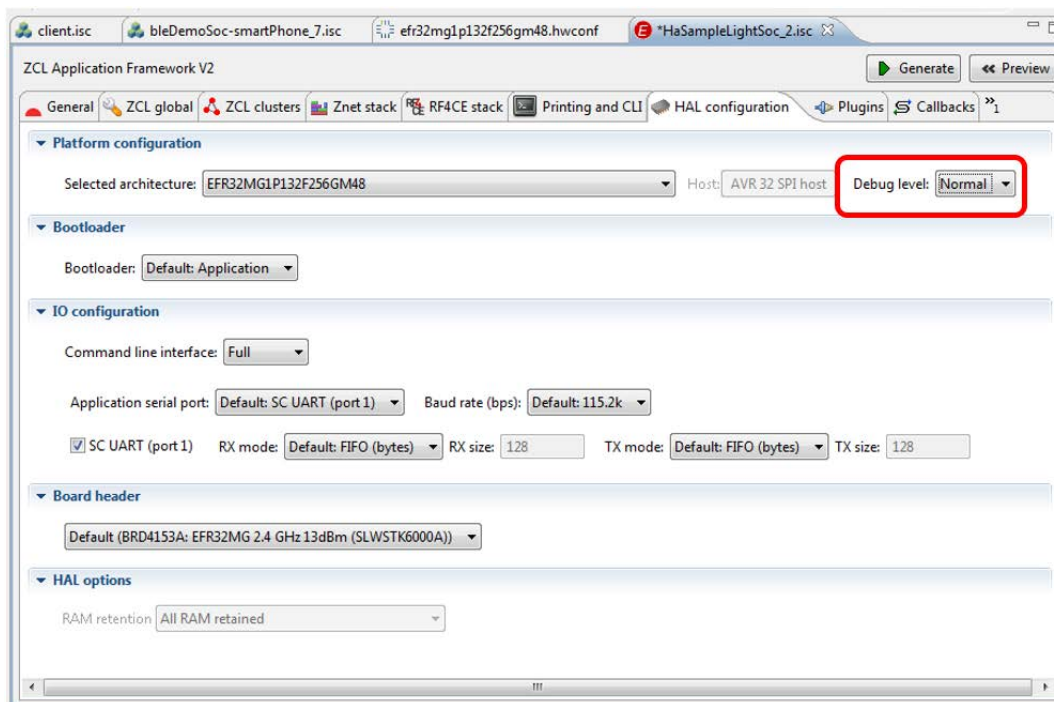


Figure 21. HAL Configuration Tab

- On the Plugins tab, check **Debug Basic Library**. Use the field above the plugins list to filter the list.

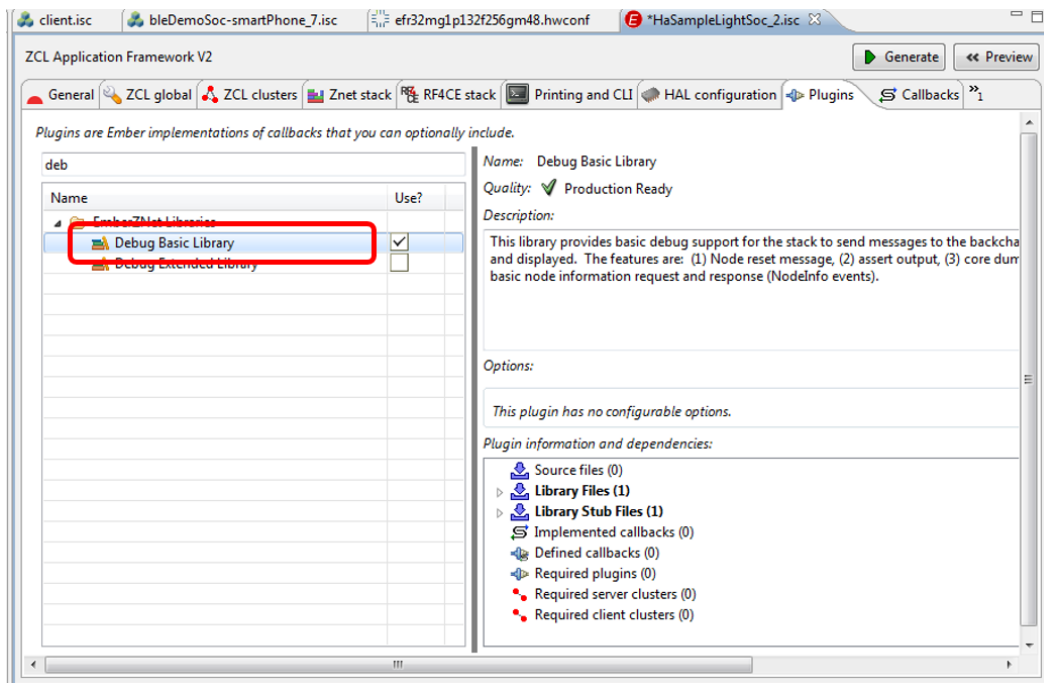


Figure 22. Plugins Tab

## 4.2 Generating the Application Source Files

- When you finish creating your example project, an AppBuilder General tab opens. Click **[Generate]**.

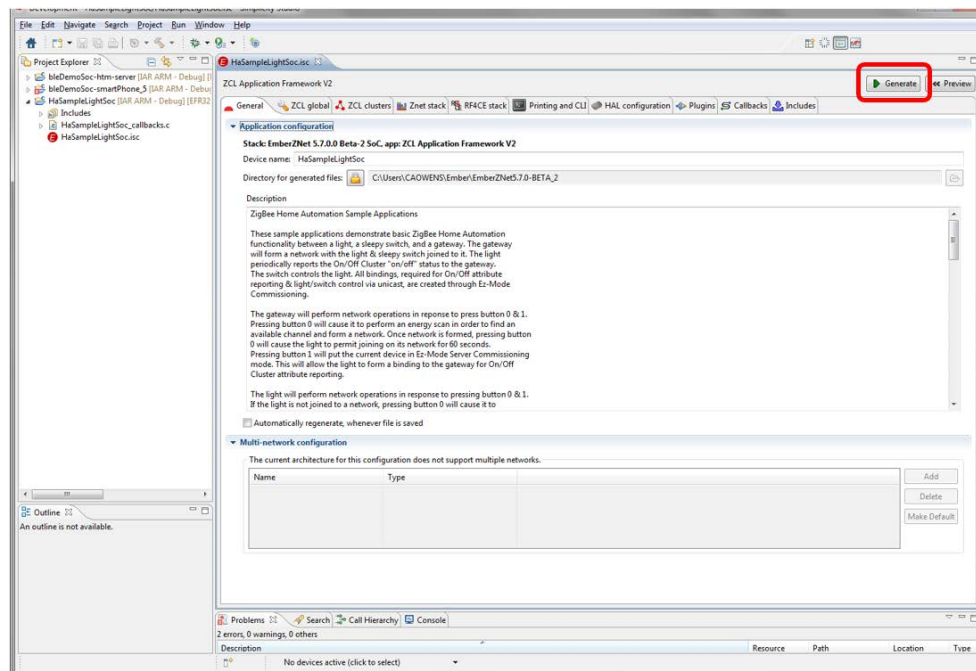
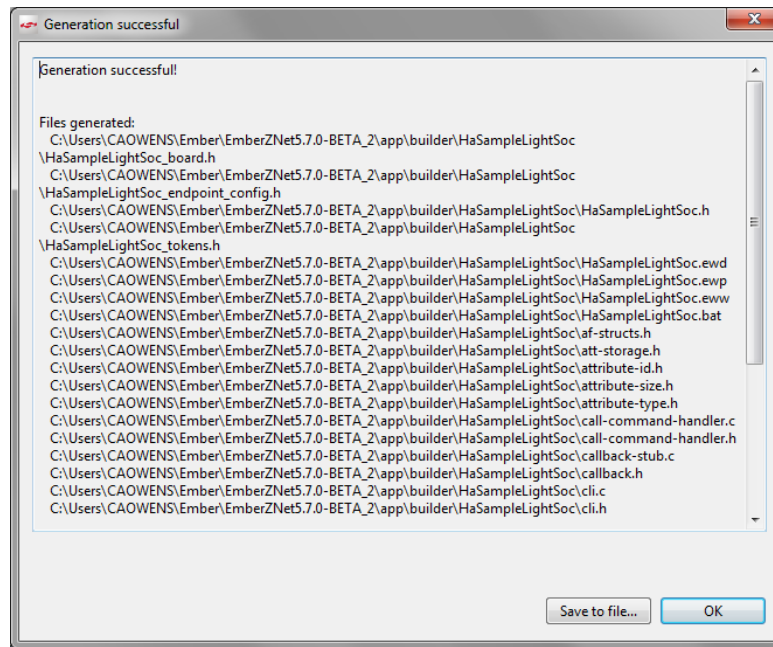


Figure 23. AppBuilder General Tab

2. Once generation is complete, a dialog reporting results is displayed. Click **[OK]**.



**Figure 24 Generation Confirmation Dialog**

### 4.3 Compiling and Flashing the Application

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

### 4.3.1 Automatically Compile and Flash

1. AppBuilder can automatically compile and flash the application to your connected development hardware. After you click **[OK]** on the Generation Confirmation dialog, the AppBuilder General tab returns. Click the **Debug** control.

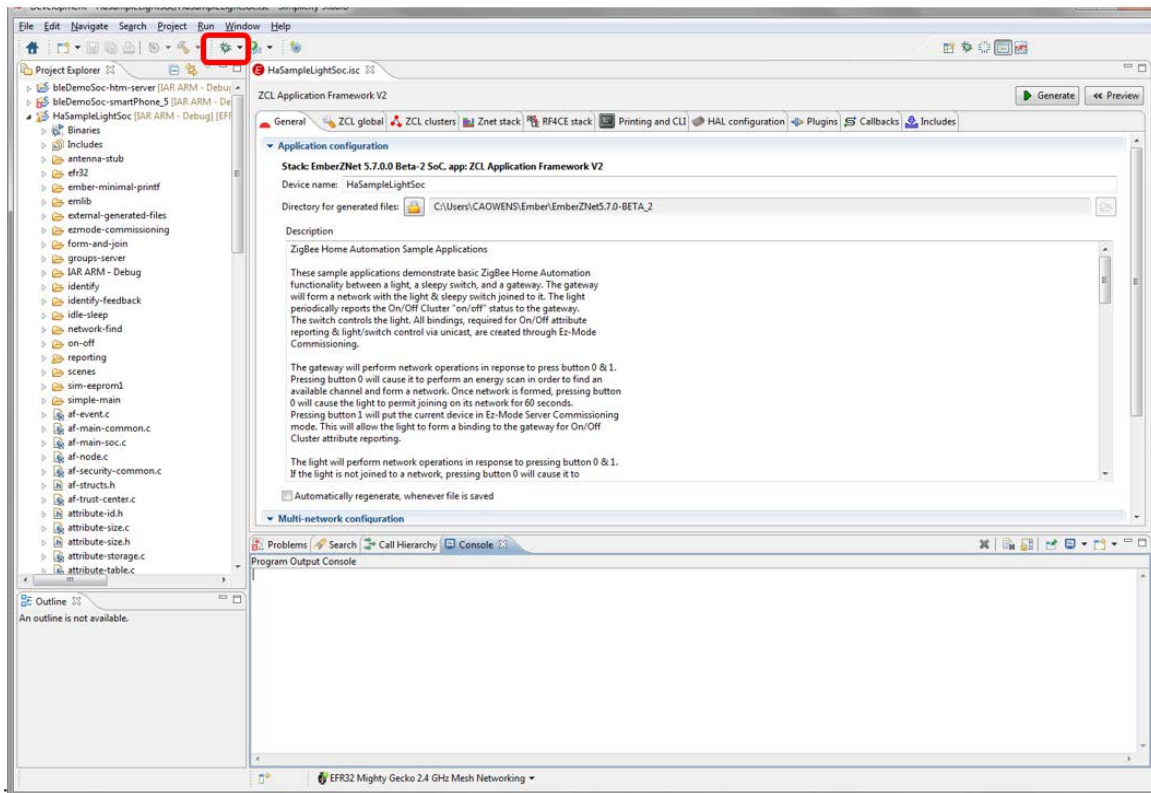


Figure 25. AppBuilder Debug Control



2. Progress is displayed in the lower left. Wait until flashing has completed and a debug window is displayed.

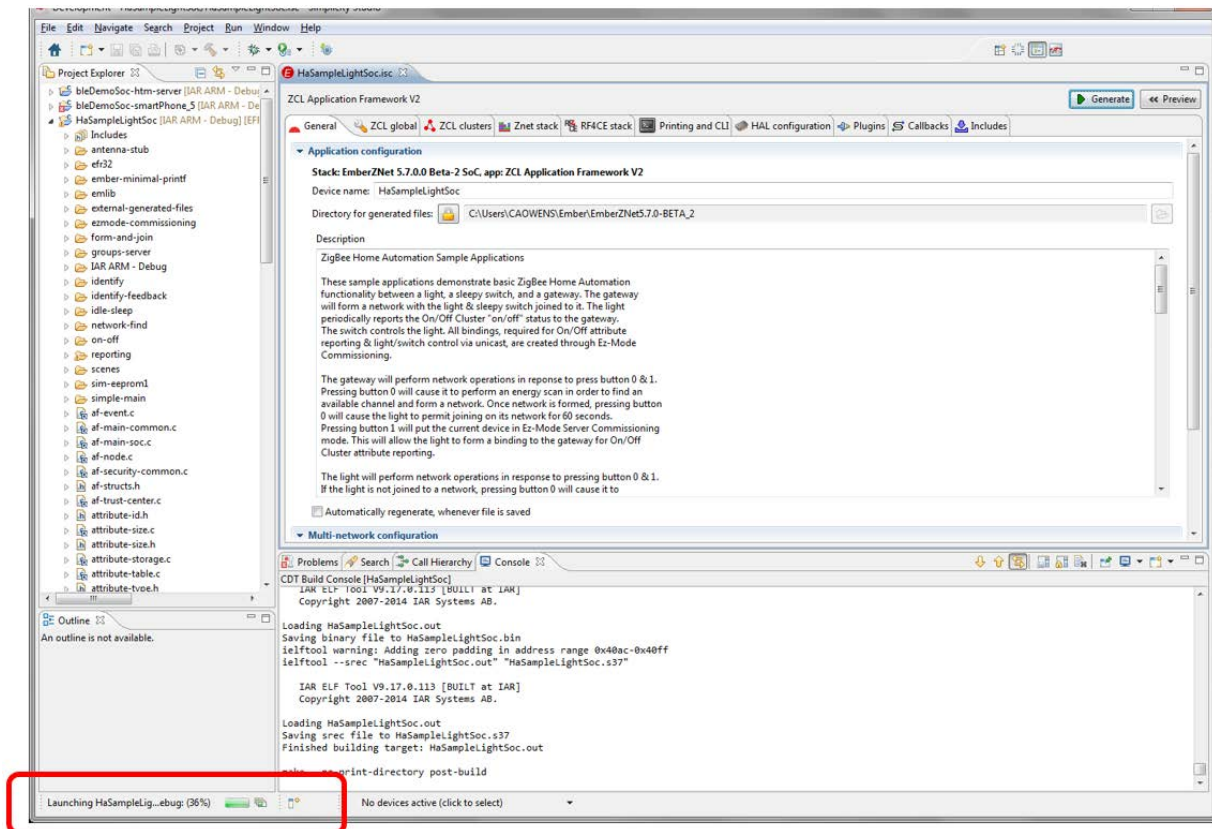


Figure 26. Compile Progress

3. In the Debug window, click the **Resume** control to start the application running on the WSTK.

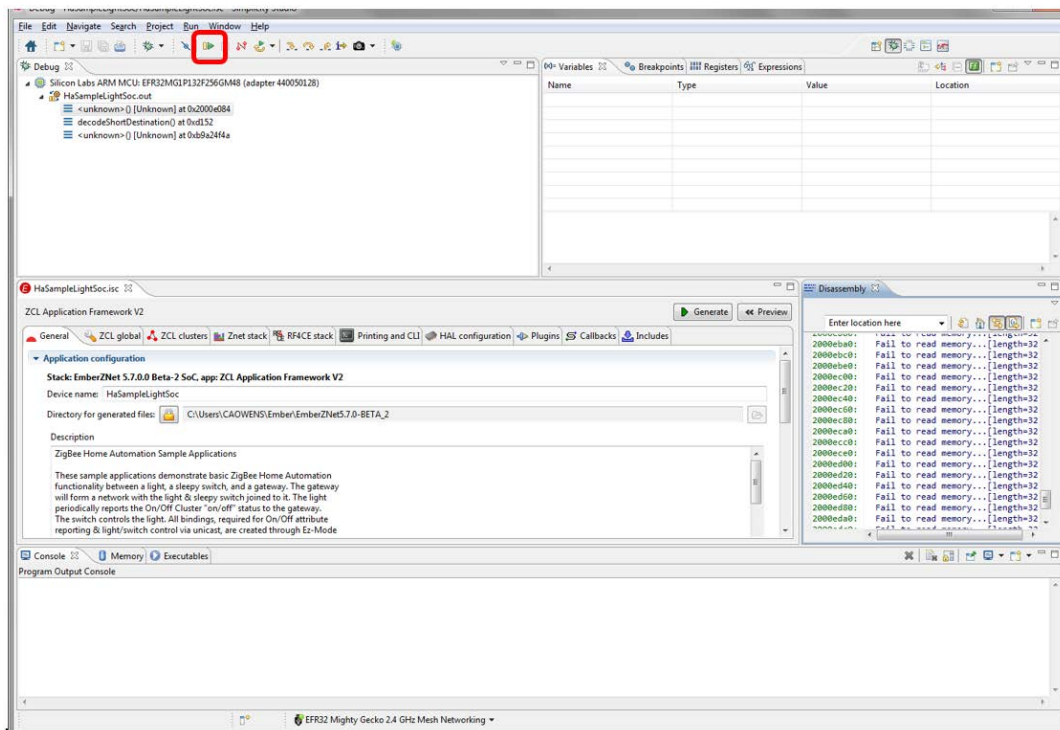


Figure 27. Debug Window - Resume Control

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



Figure 28. Disconnect Control

### 4.3.2 Manually Compile and Flash

After you generate your project files and are in AppBuilder, instead of clicking Debug, click the **Build** control in the top tool bar. Your sample application will compile based on its build configuration. You can change the build configuration at any time in the Project Explorer View by right clicking on the project and going to **Build Configurations > Set Active**.



Figure 29. Build Control

You can also build your application directly in IAR-EWARM by opening IAR-EWARM and opening the generated project file inside IAR.

1. Open IAR-EWARM.
2. Select File > Open > Workspace and navigate to the location you selected for your sample application.
3. Select the application .eww file and click [Open].
4. Select Project > Make or press F7. If the application builds without errors, you are ready to install the image on a device.

The Network Analyzer perspective provides an alternate method of loading any binary image onto your device through the Adapters View.

1. Connect your hardware and make sure it is detected in the Simplicity perspective.
2. Click the Network Analyzer tile.

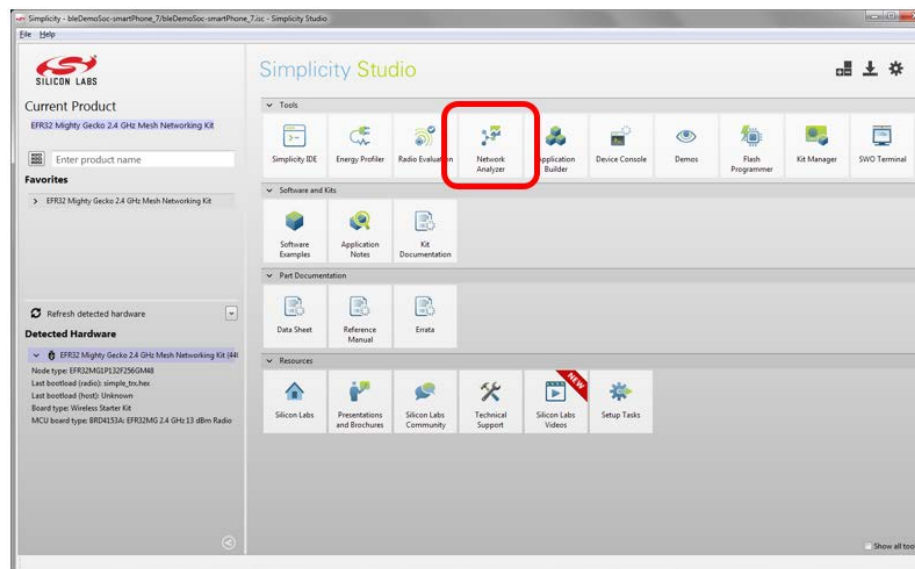


Figure 30. Network Analyzer Tile

3. The Network Analyzer perspective opens. This perspective includes an Adapters View on the left-hand side of the screen. Right-click on your device of choice, and select Connect.

- Right-click on the device in the Adapters view again and select **Flash / Upload**. The Select Binary Image dialog is displayed.

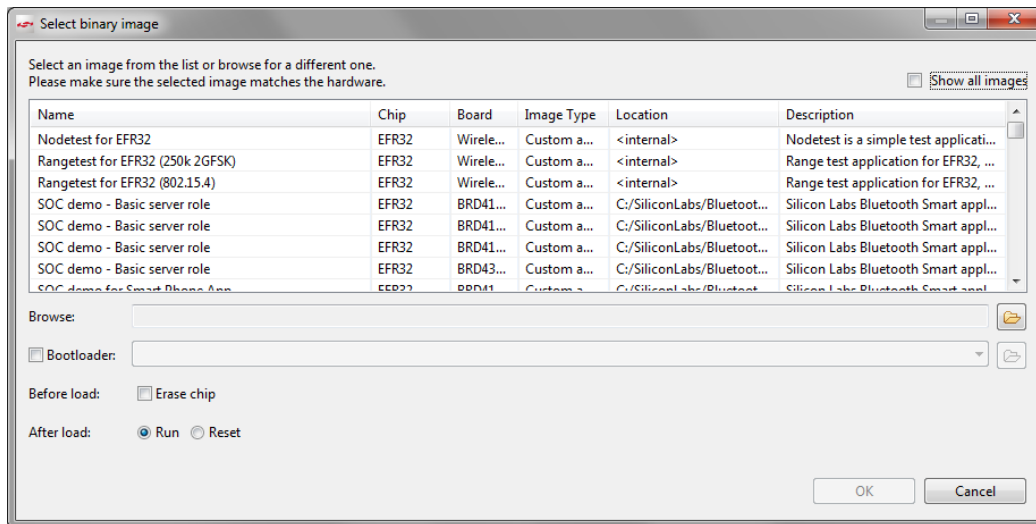


Figure 31. Select Binary Image

- Navigate to the .s37, .bin or .hex image you wish to upload.
- Optionally, check **Erase Chip**, to make sure that any previous bootloader or other non-volatile data is erased before your new image is uploaded. New users will typically check this.
- The **After Load** options are **Run** (begin executing the code immediately) and **Reset** (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**.
- Click **[OK]**. You will be notified once the upload is complete.

## 5 Interacting with your Example Application

Depending on the example application, you may be able to interact with it through your development environment's Console interface using a CLI (command line interpreter). The console interface allows you to form a network and send data.

To launch the Console interface, click the Device Console tile in the Simplicity perspective. Alternatively, in the Network Analyzer perspective right-click on your device in the Adapters View. Choose **Connect** (if you are not already connected) and then **Launch Console**. To get a prompt on the Console Serial 1 tab, press Enter.

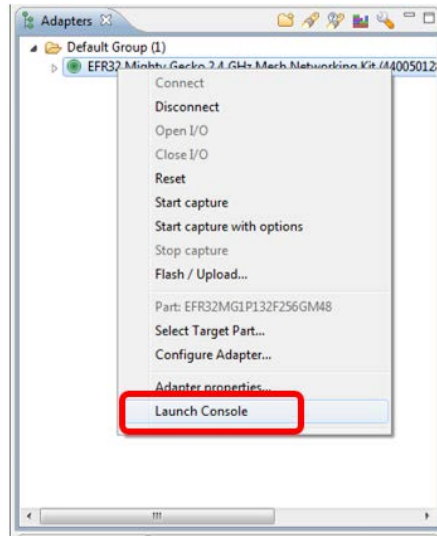
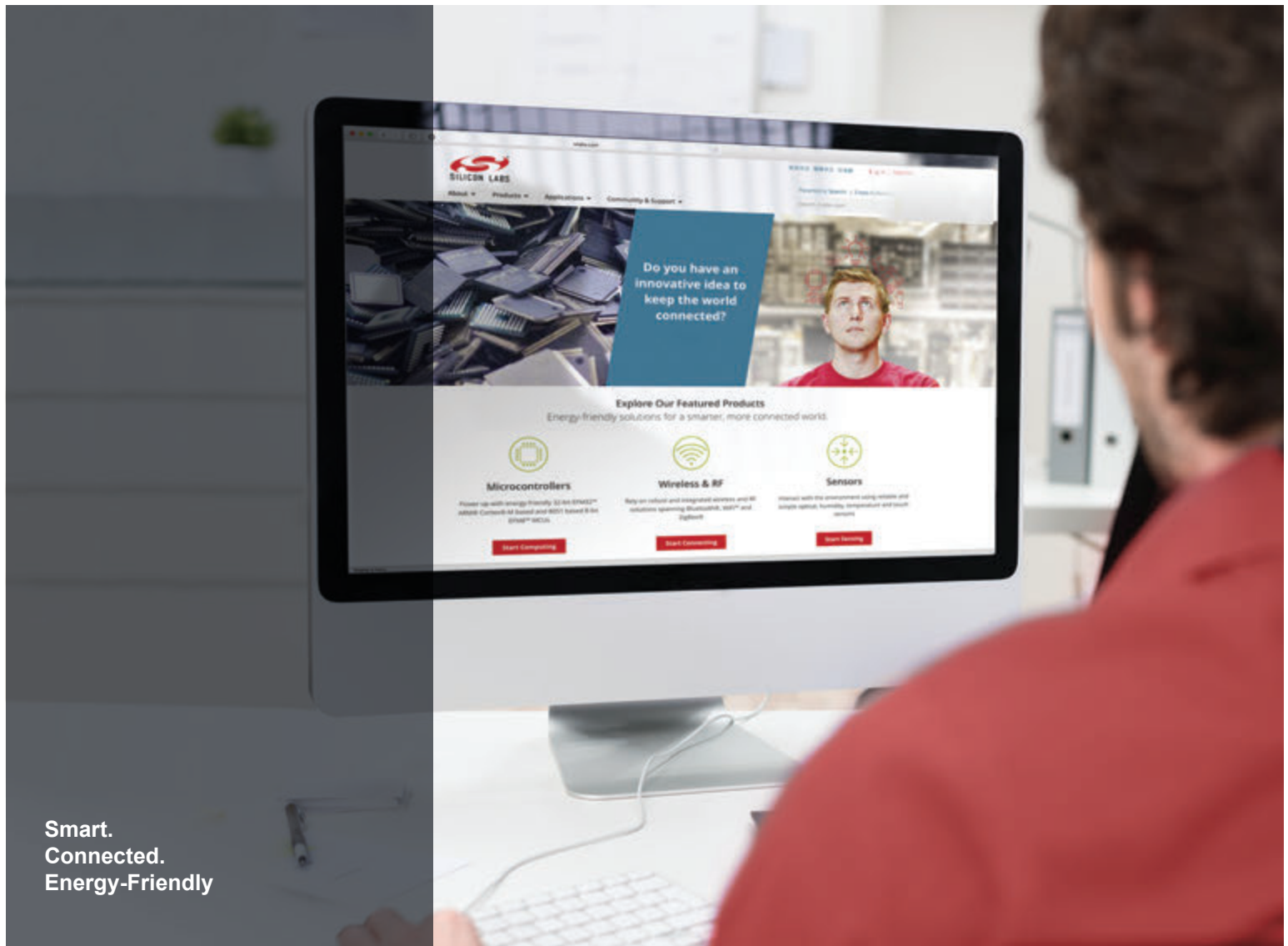


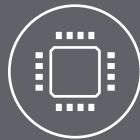
Figure 32. Launch Console



Smart.  
Connected.  
Energy-Friendly



**Products**  
[www.silabs.com/products](http://www.silabs.com/products)



**Quality**  
[www.silabs.com/quality](http://www.silabs.com/quality)



**Support and Community**  
[community.silabs.com](http://community.silabs.com)

#### Disclaimer

Silicon Laboratories intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Laboratories products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Laboratories reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Laboratories shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Laboratories. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Laboratories products are not designed or authorized for military applications. Silicon Laboratories products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

#### Trademark Information

Silicon Laboratories Inc.®, Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEEMS®, DSPLL®, EFM®, EFM32®, EFR®, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOModem®, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress® and others are trademarks or registered trademarks of Silicon Laboratories Inc. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.



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

<http://www.silabs.com>