



QSG155: Using the Silicon Labs Dynamic Multiprotocol Demonstrations

This document shows how to use the dynamic protocol lighting demonstrations. Two demonstrations may be used, one incorporating Zigbee 3.0 and Bluetooth LE functionality and the other RAIL and Bluetooth LE functionality. RAIL (Radio Abstraction Interface Layer) is Silicon Labs' intuitive, easily-customizable radio interface layer that supports proprietary or standards-based wireless protocols. In both demonstrations, a dynamic multiprotocol light application can be controlled either from a protocol-specific switch application or from a Bluetooth LE-enabled smartphone app.

KEY POINTS

- Prerequisite for the demos
- Demo firmware installation
- Instructions for using the demos

1 Prerequisites

The Silicon Labs dynamic multiprotocol demonstrations are designed to illustrate dynamic multiprotocol operation without any need to configure or compile software. This document assumes that you have already obtained your hardware and installed the required software. You will need an EFR32 Mighty Gecko Wireless Starter Kit (SLWSTK6000B). For the demo you will use two WSTK main boards with demonstration-specific radio boards installed. If you already have the WSTK main boards, you can purchase the required radio boards [here](#). Installing Simplicity Studio and the required SDKs are described in their respective quick start guides, listed below.

- Zigbee/Bluetooth Demonstration
 - Radio boards: EFR32MG12 SoC radio boards (SLWRB4162A).
 - SDK: Version 6.0.0.0 or higher of the EmberZNet SDK, as described in *QSG106: Getting Started with EmberZNet PRO*.
- RAIL/Bluetooth Demonstration
 - Radio boards: EFR32MG12 2.4 GHz/915 MHz dual band radio boards (SLWRB4164A)
 - SDKs: Version 2.7.0.0 or higher of the Bluetooth SDK (*QSG139: Bluetooth Development with Simplicity Studio*), which contains the light application, and version 2.1.0 or higher of the Flex SDK (*QSG138: Getting Started with the Silicon Labs Flex SDK for the Wireless Gecko (EFR32™) Portfolio*), which contains the RAIL switch application.

Note: This document describes using the precompiled images supplied with the SDKs. See section **Next Steps** for information on continuing with customizing the building the source code for these demonstration images.

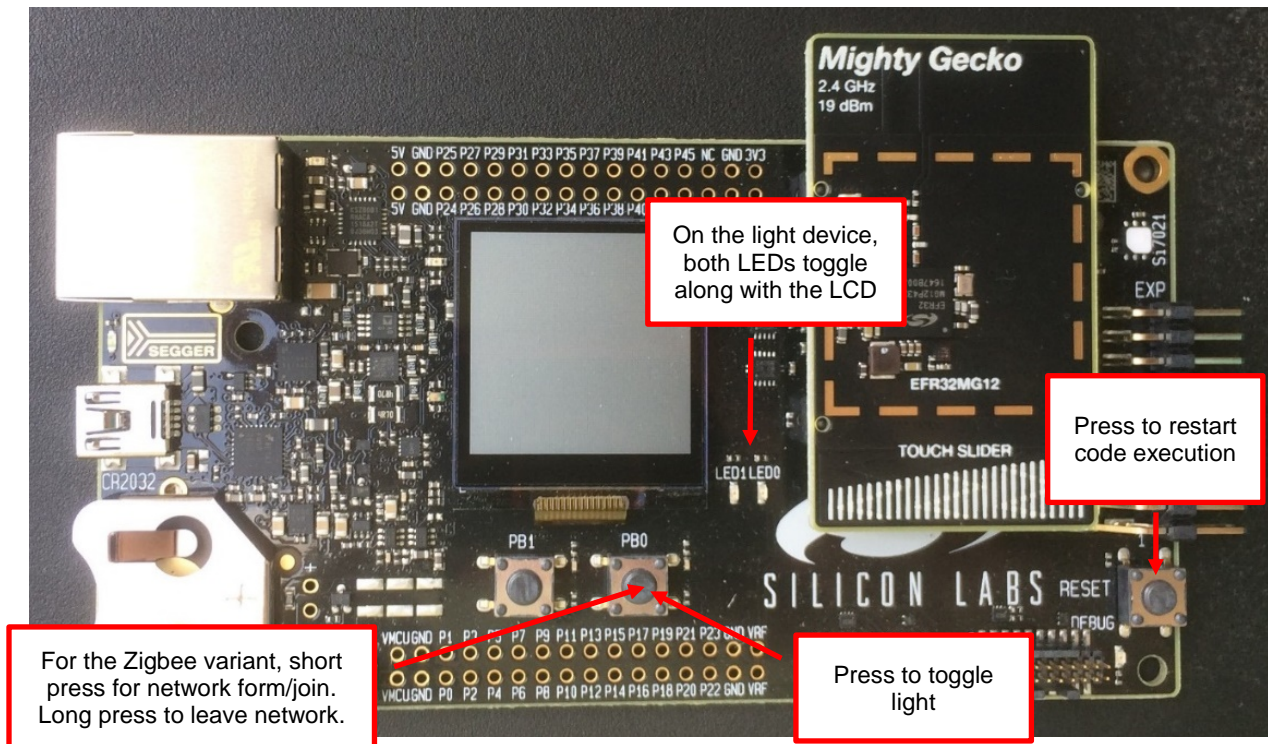
You should also download the **Wireless Gecko** by Silicon Labs smartphone application for Android or iOS. Note that there are several Silicon Labs apps - be sure to download **Wireless Gecko**, not Blue Gecko.

- **Note:** The minimum requirement for the smartphone is Android 6 (API23).

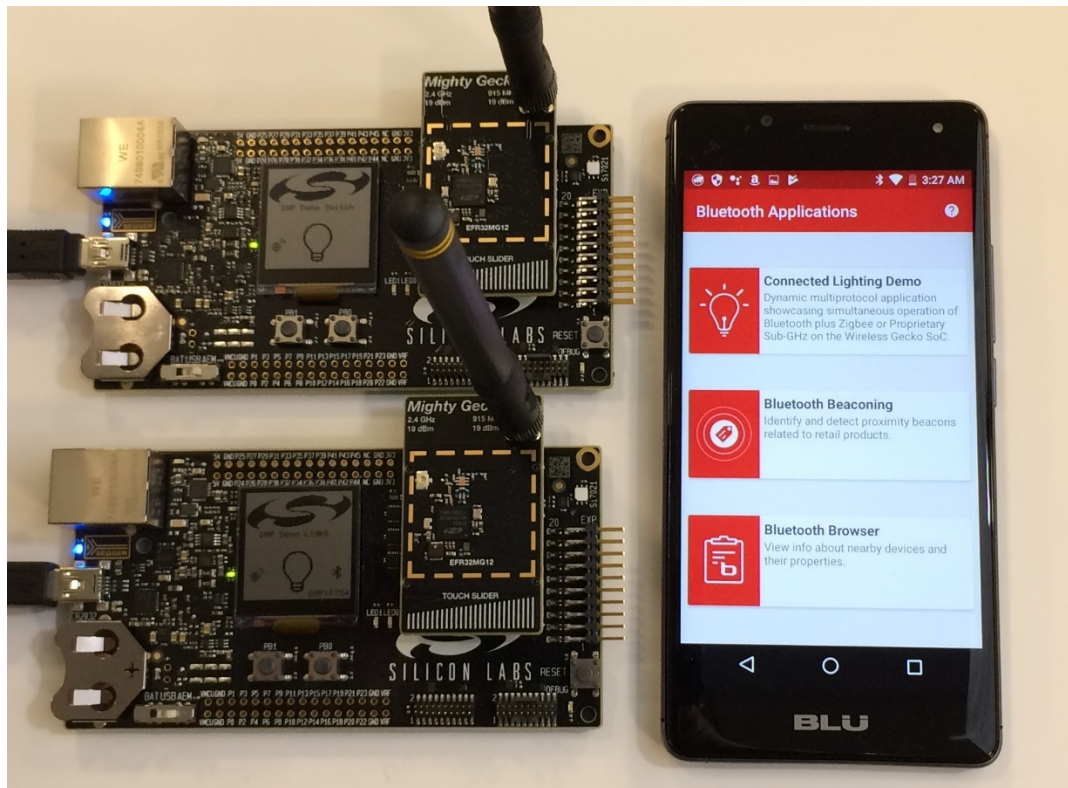
Use the support portal to contact Customer Support for any questions you might have about the demonstrations. You can access the Silicon Labs support portal at <https://www.silabs.com/support> through Simplicity Studio Resources. Click the “Email-Support” link and log in with your self-registered credentials.

2 Demonstration Devices

The following features on the WSTK are used in the demos.

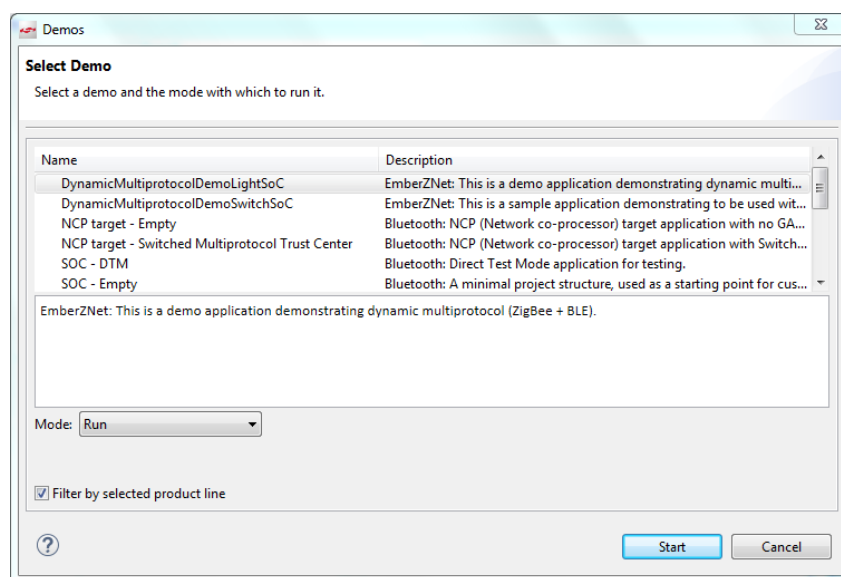


The following is a typical demo setup with two WSTKs and a smart phone (the sub-Gigahertz setup for RAIL/Bluetooth shown).



3 Install the Demonstration Applications

Open Simplicity Studio. In general, to load a demo, click a device in the Devices pane, then click the demo. The Select Demo dialog opens. In the **Mode** drop-down, select **Run**. Click **Start**. The demo software downloads automatically to the selected device.



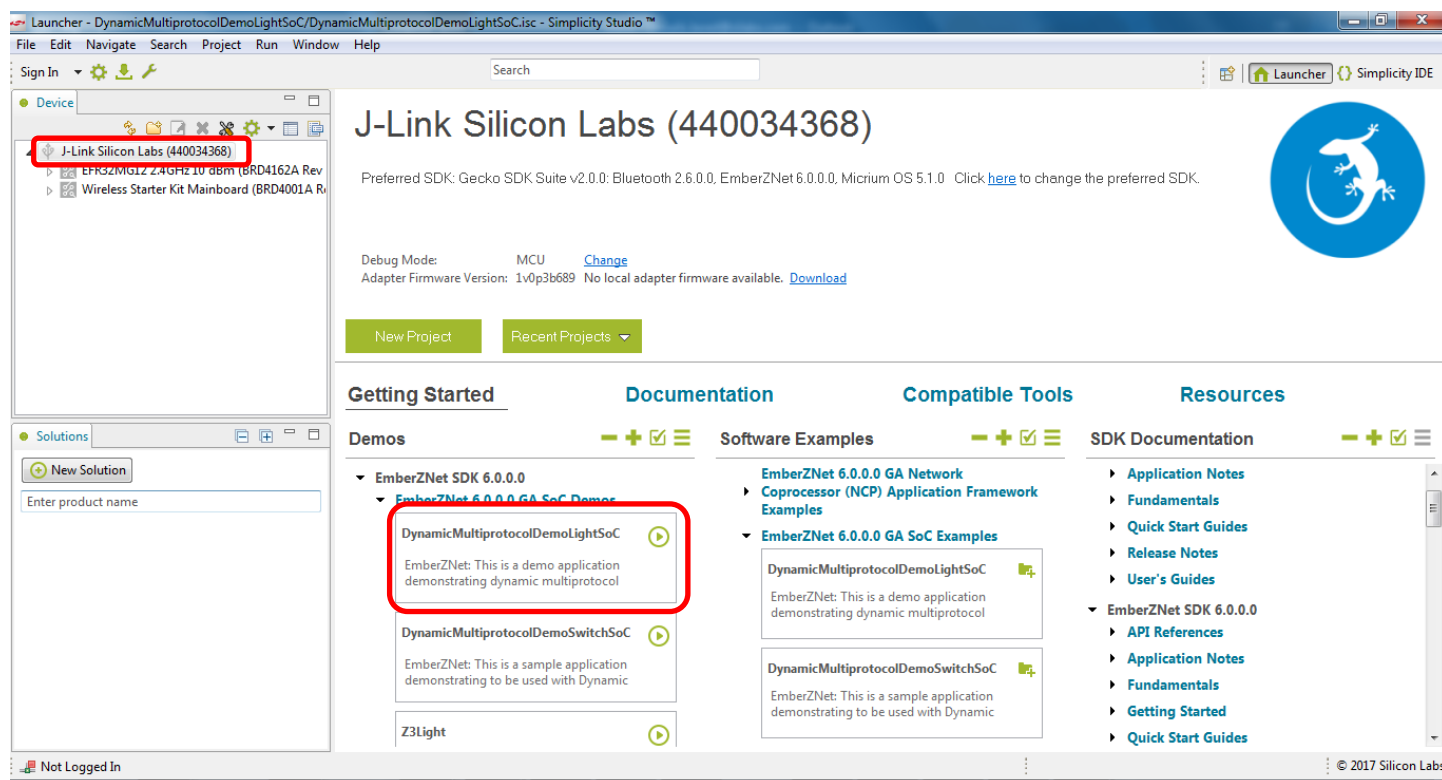
RAIL/Bluetooth

To run the RAIL/Bluetooth demo, in the Flex SDK, load one device with **RAIL: Switch**. Then go to the Bluetooth SDK and load the other device with **SOC - Light -RAIL - DMP**.

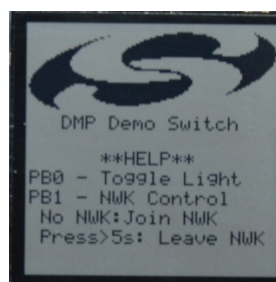
Note: The Light application in the Flex SDK is not the multiprotocol application.

Zigbee/Bluetooth

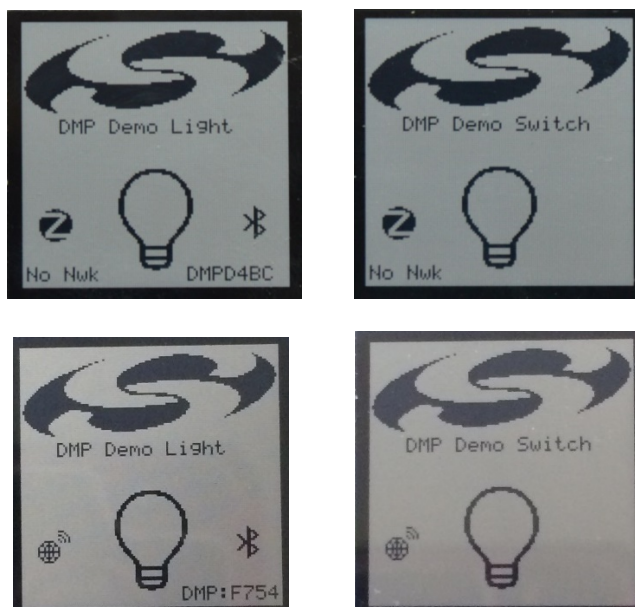
To run the Zigbee/Bluetooth demo, load one device with **DynamicMultiprotocolDemoLightSoC** and one device with **DynamicMultiprotocolDemoSwitchSoC**.



When the Zigbee/Bluetooth demo applications download, they first show a help menu on the LCD, and then show the main display. The help menus are also displayed for approximately 10 seconds if you restart code execution by pressing the Reset button. Note that resetting the application breaks the Bluetooth connection between devices, if one has been formed, and interrupts communication on the device network, if one has been formed. The Help menus are:



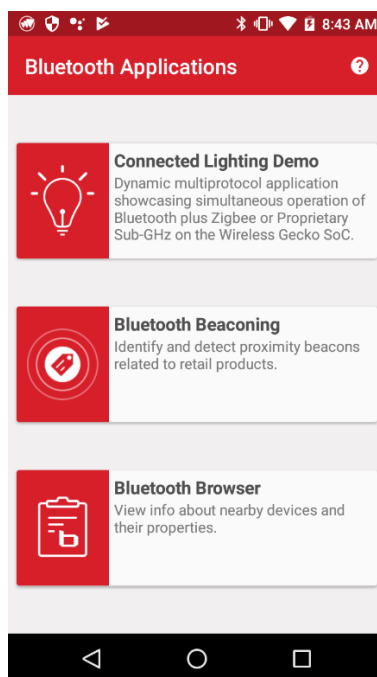
After installation is complete, the LCDs show the application “Light Bulb” display. Note the logos of the different protocols.



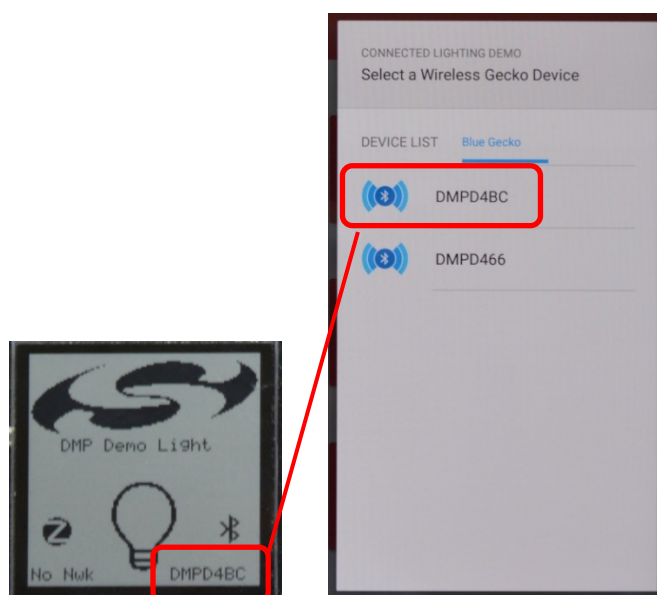
4 Use the Zigbee/Bluetooth Demonstration

4.1 With the Bluetooth Smartphone App Alone

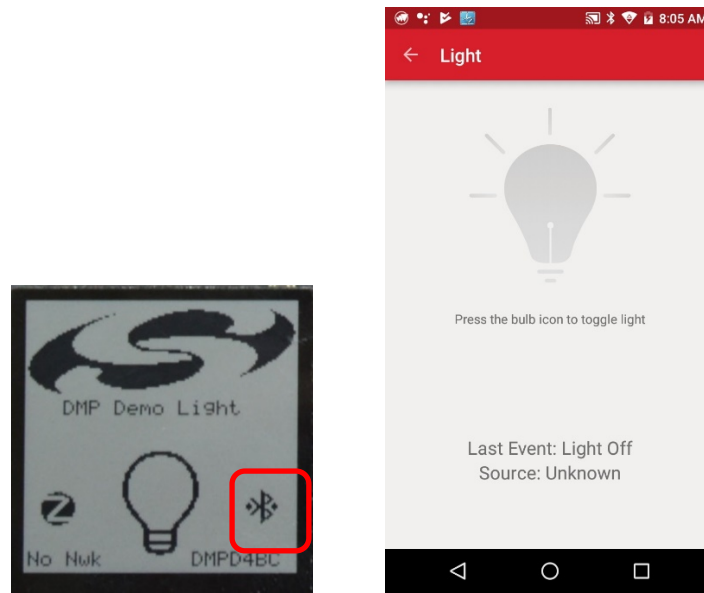
When both devices are ready, open the app and tap **Connected Lighting Demo**.



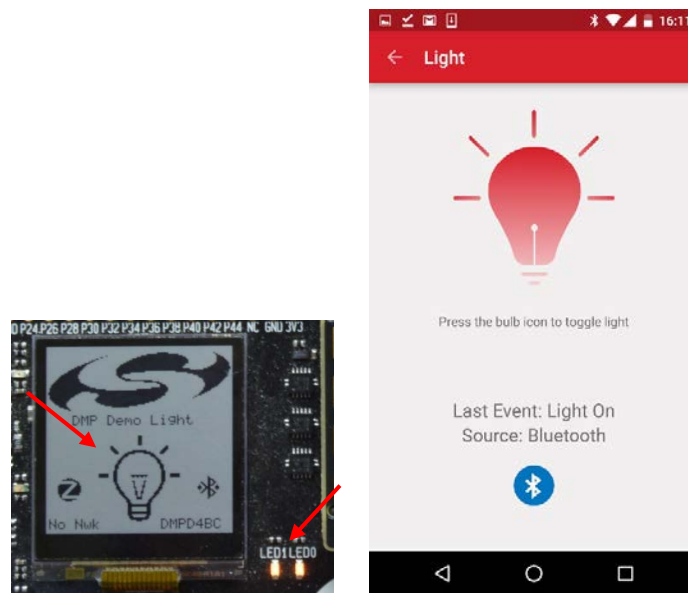
Tap the Bluetooth light device. If you see more than one, tap the one with the matching ID.



The app display changes to the Light Bulb, and the Bluetooth icon on the light's LCD changes as well, indicating a connection.



Tap the bulb icon on the smartphone app to toggle the light. The app display, the LCD display, and the LEDs all turn on. The app shows **Last Event: Light On** and **Source: Bluetooth**.



Tap again to turn the light off. The app shows **Last Event: Light Off** and **Source: Bluetooth**.

As you toggle the light with the app, an arrow is briefly displayed on the light LCD to indicate that the source of the command is Bluetooth.



On the light device, press PB0 to toggle the light. The app display now shows **Source: Button**.

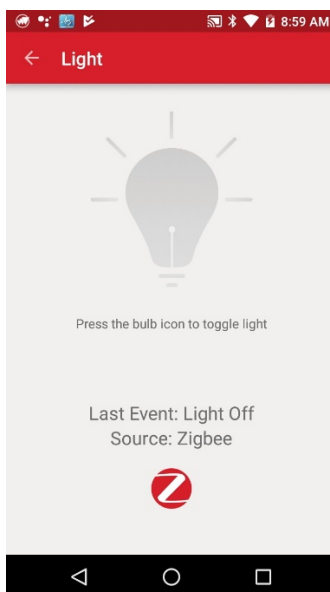
4.2 With the Switch

In order to operate the light from the Zigbee switch application, you need to form a Zigbee network. Press PB1 on the light WSTK. The display changes from **No Nwk** to **Forming** and then to a flashing PAN ID. While the PAN ID is flashing the light is in permit join mode. Press PB1 on the switch. The display changes to **Joining** and, after a brief delay, to the network's PAN ID.



Note: If you are in a busy environment, the switch might join to another network. Make sure the PAN IDs on both the light and the switch are the same. If not, press and hold PB1 on the switch for more than 5 seconds to take it off the network. Press PB1 on the light to put it back into permit join mode. Once the PAN ID is flashing, press PB1 on the switch.

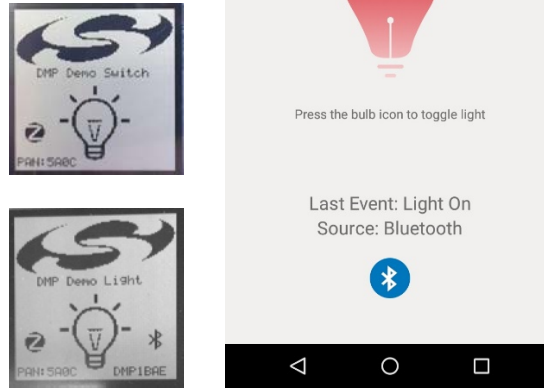
Once the switch is connected, press PB0 on the switch to toggle the light. The app shows **Source: Zigbee**.



Again, a briefly-displayed arrow shows the source of the command on the Light LCD.



Now if you press the bulb icon on the app, both the light and the switch displays change.

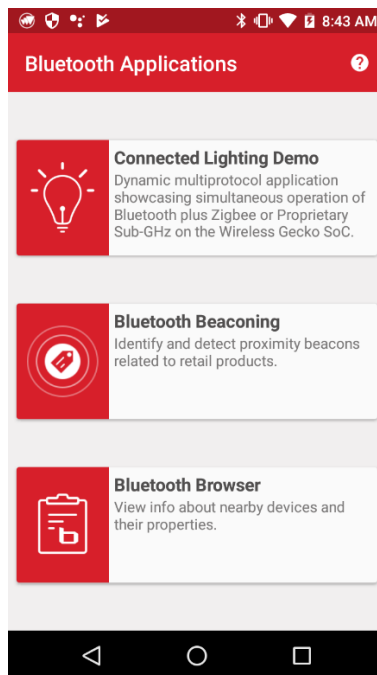


5 Use the RAIL/Bluetooth Demonstration

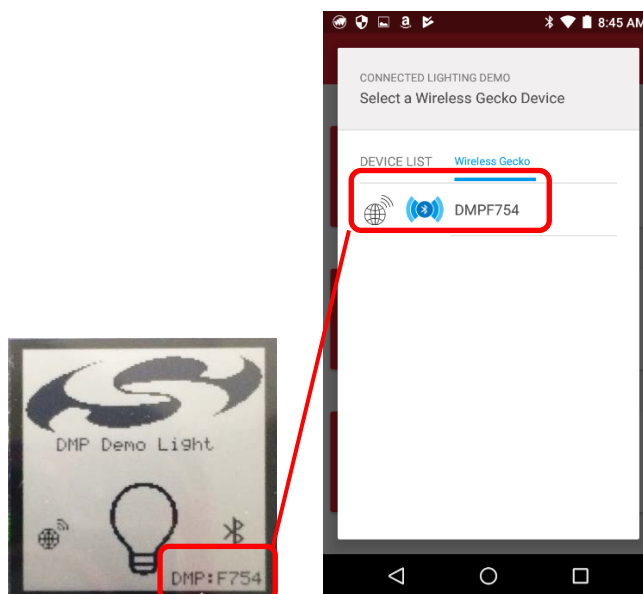
Note: In a demonstration configuration with multiple RAIL/Bluetooth dynamical protocol light devices and a single switch device, unpredictable behavior may occur. We recommend demonstrating with a single light device and a single switch device.

5.1 With the Bluetooth Smartphone App Alone

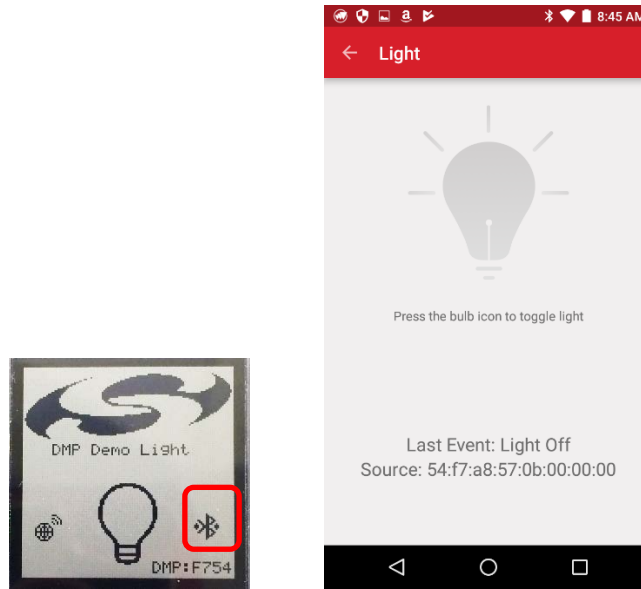
When both devices are ready, open the app and tap **Connected Lighting Demo**.



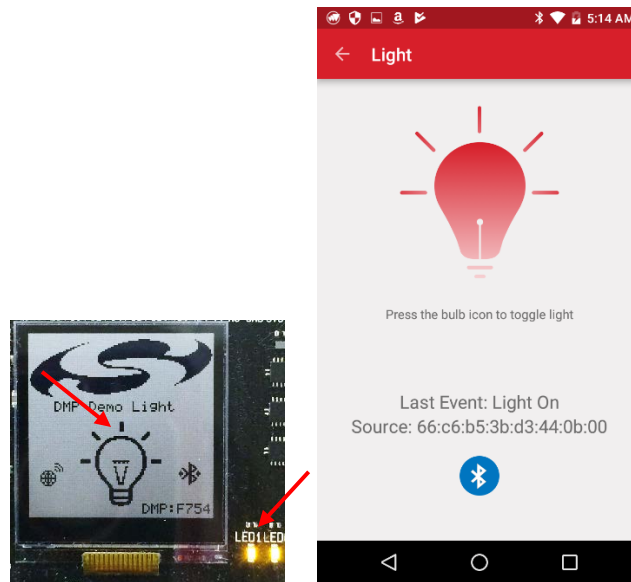
Tap the Bluetooth light device. If you see more than one, tap the one with the matching ID.



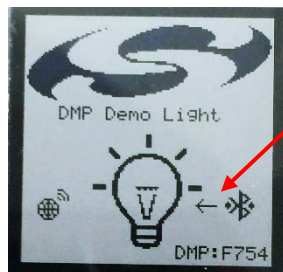
The app display changes to the Light Bulb, and the Bluetooth icon on the light's LCD changes as well, indicating a connection.



Tap the bulb icon on the smartphone app to toggle the light. The app display, the LCD display, and the LEDs all turn on. The app shows **Last Event: Light On. Source** is the MAC address of the device sending the command, in this case the smartphone.



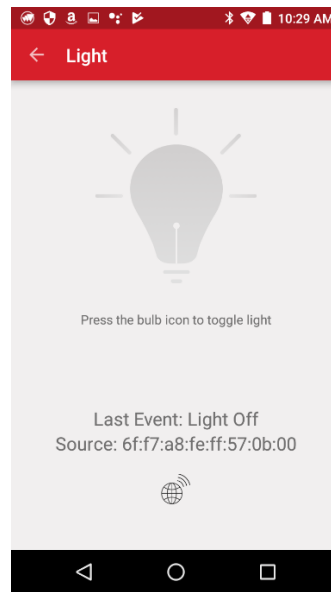
Tap again to turn the light off. The app shows **Last Event: Light Off** and **Source: Smartphone** MAC address. As you toggle the light with the app, an arrow is briefly displayed on the light LCD to indicate that the source of the command is Bluetooth.



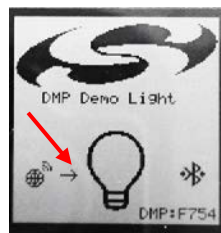
On the light device, press PB0 to toggle the light. The app display now shows the source as the MAC address of the light device.

5.2 With the Switch

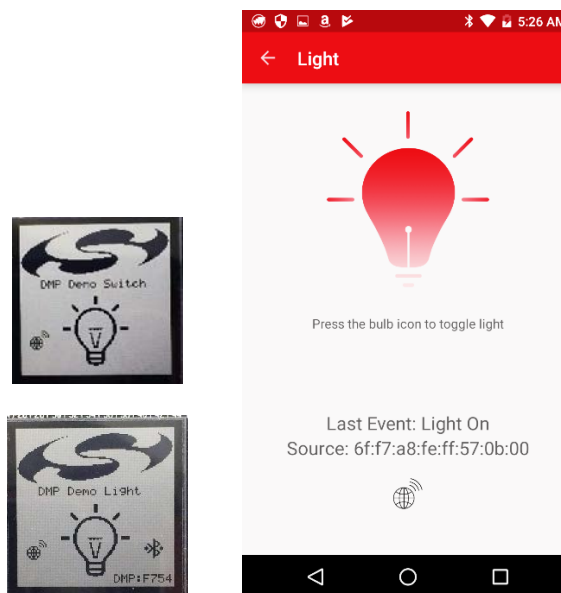
The RAIL switch application joins the RAIL/Bluetooth example devices without the need for a special operation. Once the switch is connected, press PB0 on the switch to toggle the light. The app shows the source as the MAC address of the switch.



Again, a briefly-displayed arrow shows the source of the command on the Light LCD.



Now if you press the bulb icon on the app, both the light and the switch displays change.



6 Next Steps

UG305: Dynamic Multiprotocol User's Guide contains details on the functionality underlying this demo. It also describes how the Radio Scheduler, a key component of the dynamic multiprotocol solution, works.

The source code for the smartphone application is also available. Contact Technical Support if you are interested.

6.1 Zigbee/Bluetooth

The EmberZNet SDK contains the code used to produce the demo images. *UG305: Dynamic Multiprotocol User's Guide* contains a summary procedure for these examples, and *QSG106: Getting Started with EmberZNet* describes in detail how to generate, build, and flash example code. *UG305* also describes how to change configuration settings to make an EmberZNet application into a dynamic multiprotocol application, and contains details on the functionality underlying this demo..

Note that you must download not only the EmberZNet SDK (version 6.0.0.0 or higher) but also the Bluetooth SDK (version 2.6.0.0 or higher) and the Micrium OS kernel in order to fully explore the examples. You must also install and use IAR-EWARM as your compiler.

Finally, it contains instructions on building a dynamic multiprotocol example, and how to.

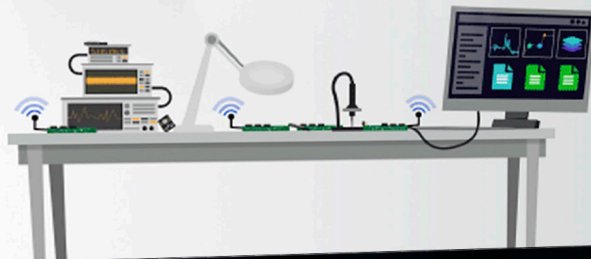
6.2 RAIL/Bluetooth

The Flex SDK (version 2.1.0.0 or higher) contains the code used to produce the Switch demo image. The Bluetooth SDK (version 2.7.0.0 or higher) contains the code used to produce the RAIL/Bluetooth dynamic multiprotocol light example. *UG305: Dynamic Multiprotocol User's Guide* contains summary procedures for these examples, and *QSG138: Getting Started with the Silicon Labs Flex SDK for the Wireless Gecko (EFR32™) Portfolio* and *QSG139: Bluetooth Development with Simplicity Studio*, respectively, describe in detail how to make and flash example applications.

Note that you must download not only the Flex and Bluetooth SDKs, but also the Micrium OS kernel in order to fully explore the examples.

Silicon Labs

Simplicity Studio™4



Simplicity Studio

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



IoT Portfolio
www.silabs.com/IoT



SW/HW
www.silabs.com/simplicity



Quality
www.silabs.com/quality



Support and Community
community.silabs.com

Disclaimer

Silicon Labs 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 Labs 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 Labs 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 Labs 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 Labs. 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 Labs products are not designed or authorized for military applications. Silicon Labs 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®, CMEMS®, 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®, Micrium, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri and others are trademarks or registered trademarks of Silicon Labs. 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>