Getting Started with RSL10 Bluetooth® Low Energy Mesh

1. INTRODUCTION

RSL10 is a multi-protocol, Bluetooth® 5 certified, radio System on Chip (SoC), with the lowest power consumption in the industry. It is designed to be used in devices that require advanced wireless features, with minimal system size and maximized battery life. Bluetooth mesh networking was introduced in July 2017 by the Bluetooth Special Interest Group (SIG) as a means of supporting large-scale networks requiring "many to many" node connections. The RSL10 mesh package provides support for using RSL10 with Bluetooth mesh networking.

This document helps you to get started with the RSL10 mesh CMSIS-Pack. It guides you through the process of installing the mesh package alongside the RSL10 CMSIS-Pack, configuring your environment, and building and debugging your first RSL10 mesh network.

The RSL10 mesh package, along with the RSL10 CMSIS-Pack, allows for rapid development of ultra-low power Bluetooth Low Energy mesh applications. Sample code is provided for easy configuration and deployment of mesh networks with any combination of Proxy Nodes (to connect a mobile phone), Relay Nodes, Friend Nodes, and Low Power Nodes (LPN), with support for multiple LPNs per Friend. For more sample code information, see the *RSL10 Bluetooth Low Energy Mesh Sample Code User's Guide*.

1.1 Basic Terminology

Unprovisioned device

A device that has not been provisioned and is not a member of a mesh network. It cannot send nor receive mesh messages; however, it advertises its presence to Provisioners.

Node

A device that has been provisioned and is a member of a mesh network. It can send and receive mesh messages.

Provisioner

A device that performs all required operations to convert an Unprovisioned device into a Node. The Provisioner plays a key role in setting up and operating a mesh network.

Provisioning

A process that is managed by a Provisioner. It provides an Unprovisioned device with data (including a network key, unicast address, etc.) required for communication in a mesh network.

Provisioning Bearer

A transport layer used by a Provisioner and an Unprovisioned device to exchange messages with each other during the provisioning procedure. Two Provisioning Bearers are defined: Advertising Bearer (PB-ADV) and Generic Attribute Profile Bearer (PB-GATT).

CLI application

An RSL10 Evaluation and Development Board can be programmed with a CLI application, which allows you to use command lines to configure the board to fulfill any role in the network, and to send and receive mesh messages. CLI commands are sent from a computer to the board over an UART interface. A Terminal emulator software (such as Tera Term or YAT) is required on your computer for this.

Stand-alone application

An RSL10 board can be programmed with a specific stand-alone application, which is designed to configure the board into a given role in the network, and to send and receive mesh messages associated with given models.

2. Prerequisites

To use the RSL10 mesh networking package, you need:

- The RSL10 ON Semiconductor IDE installed on a computer
- Terminal emulator software, such as Tera Term or Yet Another Terminal (YAT), installed on the computer
- One RSL10 Evaluation and Development Board for each Provisioner, Server Node and Client Node in the network

Mesh applications can be executed on any RSL10 Evaluation and Development Board. The UART interface is used to connect the board and the computer for issuing user commands, and for tracing activities performed by, or completed in, each mesh peer device. The onboard micro USB-B connector is used as the UART connection without additional hardware.

The computer can use any terminal emulator software to send CLI commands to, and receive responses from, the RSL10 Evaluation and Development Board. Use Windows® Device Manager to search for `JLink CDC UART Port (COMXX)`, and identify the COM port number your serial terminal software connects to. Set up the serial port at a baud rate of 115200 bps, no parity, 1 stop bit, and enable the option that echoes typing. This ensures that the terminal software displays what you enter in the terminal window; otherwise you cannot see what you are typing. To do this in Tera Term, select **Local echo** in **Setup > Terminal.** Alternatively, YAT is a user-friendly and feature-rich terminal. You can use YAT to define a set of commonly-used CLI commands, then execute a desirable sequence of commands to set up a mesh network for your needs.

3. Installation Instructions

The installation of the RSL10 mesh CMSIS-Pack adds Bluetooth Low Energy mesh network support to the ON Semiconductor IDE CMSIS-Pack. To install this pack, follow the steps below:

1. Download the **Mesh CMSIS-Pack** from www.onsemi.com/RSL10 and save it in any temporary folder.
2. Open the ON Semiconductor IDE and choose the desired location for your new workspace — for example, *c:\workspace* — and click **OK**.
3. Follow the same instructions as in the *RSL10 Getting Started Guide* to install a CMSIS-Pack in the ON Semiconductor IDE.
4. The **Mesh CMSIS-Pack** now appears in the list of installed packs.You can manage your installed packs in the **Packs** tab. Expanding **ONSemiconductor.RSL10Mesh** makes the **Pack Properties** tab display the details of the **Mesh CMSIS-Pack** and the version of the RSL10 CMSIS-Pack that needs to be installed. Figure 1 on page 3 illustrates what the **Pack Manager** perspective looks like after installation. It is possible to start by installing the RSL10 CMSIS-Pack if you know the right version required for the Mesh CMSIS-pack.

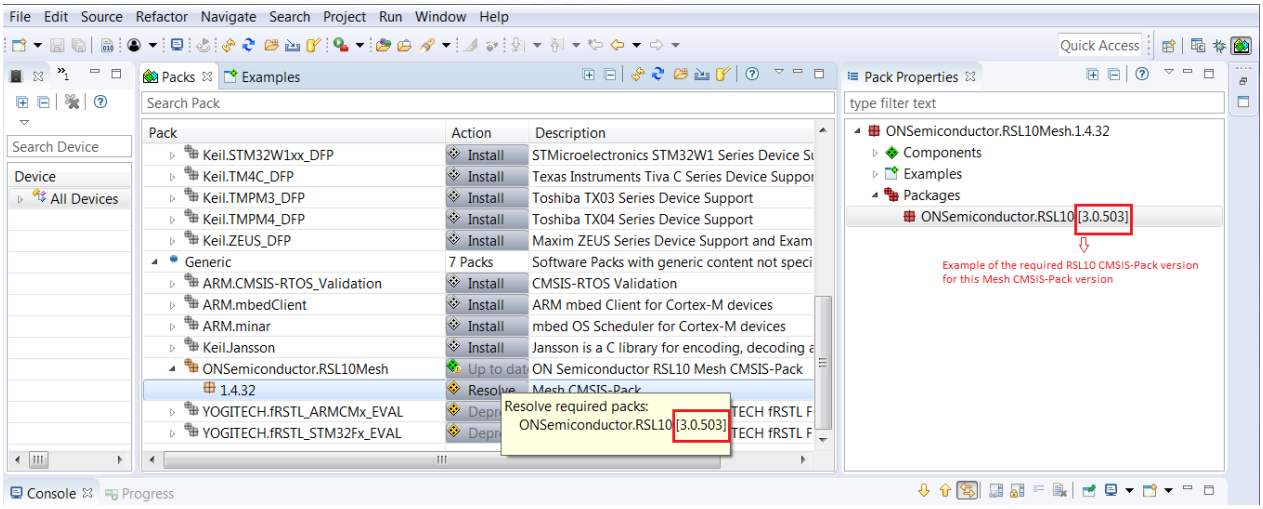


Figure 1. Pack Manager Perspective After Installing Mesh CMSIS-Pack

After installing the required RSL10 CMSIS-Pack version for the installed Mesh CMSIS-Pack, the Mesh application is added to the **Examples** tab of the RSL10 CMSIS-Pack. In the **Devices** tab, expand **All Devices > ONSemiconductor > RSL10 Series** and select **ONSemiconductor.RSL10** (see Figure 2 on page 3). Click on the **Examples** tab to list all the example projects included in the RSL10 CMSIS-Pack.You can see that mesh sample code is added to the list of examples (see Figure 3 on page 4).

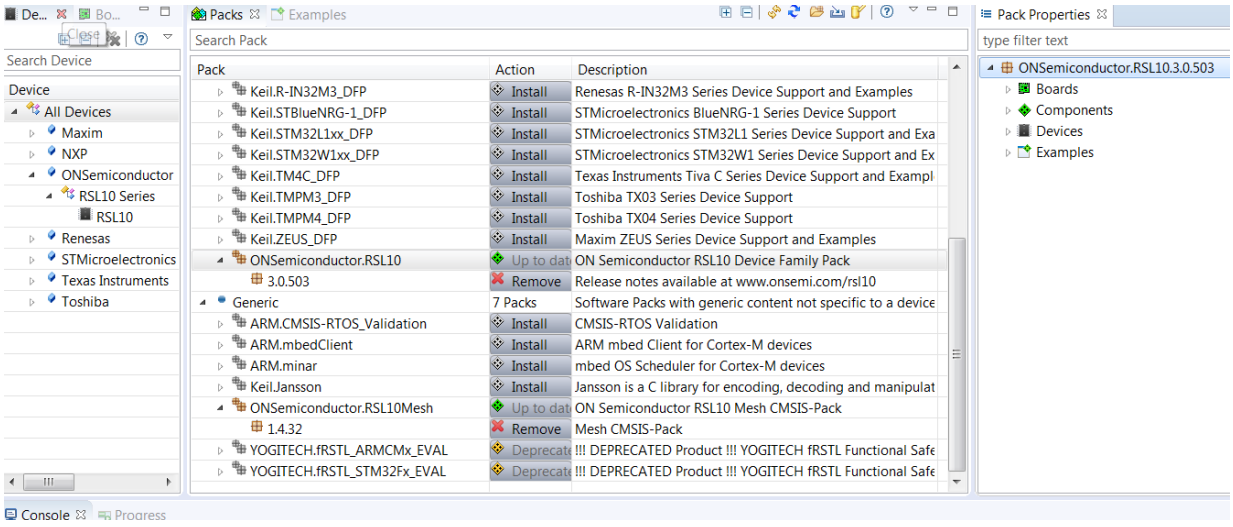


Figure 2. Pack Manager Perspective After Installing Mesh and RSL10 CMSIS-Packs

By default, **Mesh CMSIS-Pack** is installed in *C:\Users*`<user_id>`*\ON_Semiconductor\PACK\ONSemiconductor* and includes the following files:

- The mesh support libraries in *lib\mesh*
- The mesh sample codes in the *source\samples*

- The mesh support documentation in the documentation folder, with documents from Mindtree® in the *documentation\mindtree* subfolder

4. Using the Mesh Sample Project

This procedure assumes that you have installed the RSL10 Bluetooth Low Energy mesh support according to the instructions in Section 3., “Installation Instructions” on page 2.

4.1 Working with the Sample Project in the IDE

Detailed information about using the Integrated Development Environment to import sample code, build it, and debug it can be found in the *Getting Started with RSL10* manual.

To work with the mesh sample project specifically, start the ON Semiconductor IDE and follow the steps below:

1. In the CMSIS Pack Manager, expand the **Example**s tab of RSL10 CMSIS-Pack and choose the **ble_mesh** example project. Click the **Copy** button to import it into your workspace (see Figure 3 on page 4).

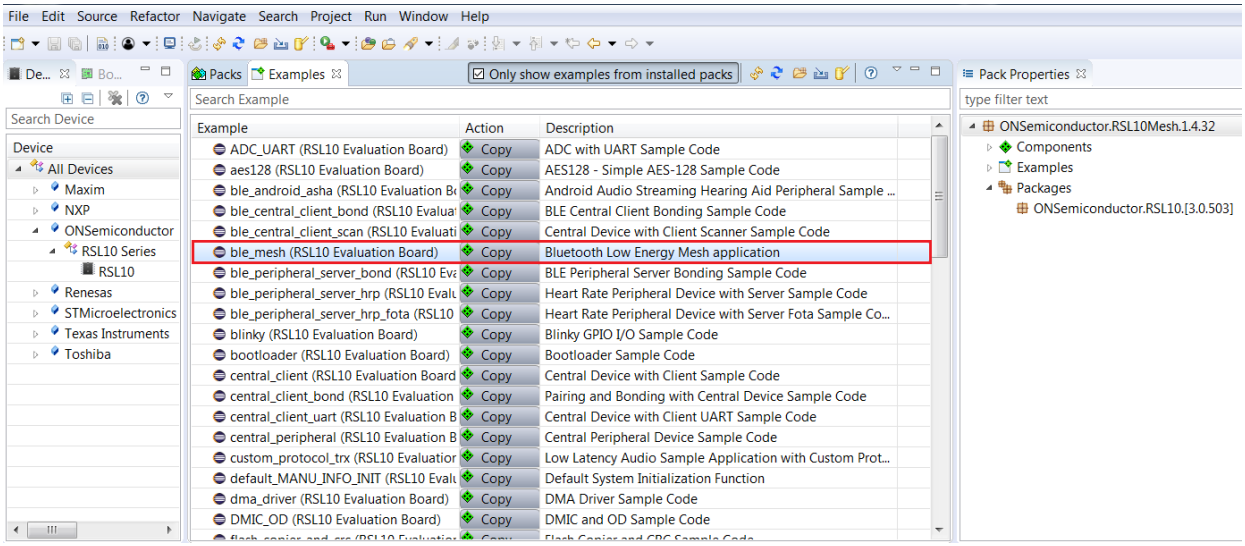


Figure 3. Examples Tab Perspective After Installing Mesh and RSL10 CMSIS-Packs

2. Continue with building the Bluetooth Low Energy mesh project from the IDE C/C++ perspective.

For more details about the *ble_mesh* sample application, continue reading the following sections.

4.2 Mesh Network Setup Using the Stand-alone Provisioner

4.2.1 Goals

These instructions show how to create a mesh network with the stand-alone Provisioner application. The stand-alone Provisioner only applies to PB-ADV. The main goal in this section is to show how to use the stand-alone Provisioner to set up and test a mesh network.

Overview of the process:

1. Flash load three mesh firmware applications into three different RSL10 Evaluation and Development Boards: the mesh Provisioner, the mesh Client, and the mesh Server.
2. Use the Provisioner to provision the Client and Server(s) devices.
3. Use the push button on the Provisioner to set the publish information on the Client, and send OnOff commands to selected Servers.

4.2.2 Stand-Alone Application for Provisioner

To build the stand-alone Provisioner application, define the constant `MESH_APP` with the value `MESH_APP_SA_ADV_PROVISIONER` in the *ble_mesh\include\app_mesh.h* file.

After building this application successfully, use it to program an RSL10 board which then becomes the Provisioner (it provisions a device as soon as it sees the beacon, so we recommend powering it off until everything is ready for testing).

NOTE: In general, you need to reset the provisioning information in a device if you are trying to use it in a new network with a new Provisioner. While you do not need to reset it the first time you set up the network, if you reprogram the device with a different stand-alone application, you must reset the provisioning information before trying to provision it once more.

Refer to Section 4.2.5, “Reset Provisioning Information” on page 5 to reset the provisioning information.

4.2.3 Stand-Alone Application for Generic OnOff Client

To build the stand-alone Client application, define the constant `MESH_APP` with the value `MESH_APP_SA_ADV_CLIENT` in the *ble_mesh\include\app_mesh.h* file.

After building this application successfully, use it to program an RSL10 board which then advertises as an Unprovisioned device.

Refer to Section 4.2.5, “Reset Provisioning Information” on page 5 to reset the provisioning information.

4.2.4 Stand-Alone Application for Generic OnOff Server

To build the stand-alone Server application, define the constant `MESH_APP` with the value `MESH_APP_SA_ADV_SERVER_FRIEND` in the *ble_mesh\include\app_mesh.h* file.

After building this application successfully, use it to program an RSL10 board which then advertises as an Unprovisioned device.

Refer to Section 4.2.5, “Reset Provisioning Information” on page 5 to reset the provisioning information.

This step can be repeated as many times as desired to have more servers for testing.

4.2.5 Reset Provisioning Information

To prepare a device for provisioning:

1. Power on the device.
2. Press and hold the onboard SW1 push button on the device for at least five seconds until the LED has blinked off/on five or more times. Release the button. After the provisioning information from any previous network is cleared, the LED blinks at a slow rate. Power cycle the device.

4.2.6 Provision and Configure devices

The stand-alone Provisioner provisions and configures devices as soon as it sees the beacon. The order in which the devices are powered on to be provisioned does not matter. There are two constraints on provisioning: the maximum number of elements, and the maximum number of models per element. Both of these constants (`MAX_ELEMENTS` and `MAX_MODELS_PER_ELEMENT`) are defined in *appl_adv_provisioner.c*. The constant `MAX_ELEMENTS` only includes the elements of devices that are provisioned (i.e., the element on the stand-alone Provisioner does not count toward the limit). You can make the values higher or lower by changing the constants.

4.2.7 Testing the Stand-alone Provisioner

Testing the stand-alone Provisioner can be summarized as follows:

1. Set up a network by provisioning a Client and a Server Node.
2. On the Provisioner, push the onboard push button which causes the stand-alone Provisioner to search through the stored provisioned information, find a Client, and set its publication information to talk to the selected Server.
3. Press the push button on the Client to send an OnOff command to the Server, which causes the Server's LED to turn on/off.

The details of the test are as follows:

1. Power on the stand-alone Provisioner.
2. Power on a device, such as a Server Node.
3. Wait until you see `MS_ACCESS_CONFIG_MODEL_SUBSCRIPTION_STATUS_OPCODE` (for servers). The server is now part of the mesh network, and configured.
4. Power on the next device, such as a Client Node.
5. Wait until you see `Send Config Model App Bind with Retval - 0x0000` (for clients). The client is now part of the mesh network, and configured.
6. Power on and provision more devices as needed.
7. On the Provisioner, press the onboard SW1 push button to set the publish information for the client. To select the first server you have provisioned, press the button quickly. To select the second server, press and hold the button until the LED blinks once. To select the third server, press and hold the button until the LED blinks twice, and so on.
8. On the Client Node, press the onboard SW1 push button to send an OnOff command to the selected server.
9. Repeat steps 7-8 with different server nodes selected as needed.

This demonstrates that the stand-alone Provisioner can both provision and configure nodes.

5. More Information

For more information about using RSL10 mesh, see:

RSL10 Bluetooth Low Energy Mesh Release Notes
: Lists new features and known issues in the latest release. This file is downloaded with the installer in a zip file.

RSL10 Bluetooth Low Energy Mesh Release Notes History
: A collection of release notes from previous releases that is downloaded with the installer in a zip file.

Additional documentation is installed in *C:\Users*`<user_id>`*\ON_Semiconductor\PACK\ONSemiconductor\RSL10Mesh*`<pack_version>`*\documentation*, where `<user_id>` is your profile name, and `<pack_version>` is a version number (e.g., 3.0.521):

RSL10 Bluetooth Low Energy Mesh Sample Code User's Guide

Describes what the mesh sample application (*ble_mesh*) demonstrates, how to configure the projects to set up different mesh network scenarios, and how to verify correct operation.

EtherMind Mesh Documentation

User and reference manuals from Mindtree:

- *EtherMind Mesh SDK Documentation*: a compiled help file that provides the application programmer interface (API) reference material
- *EtherMind Mesh Application Developer's Guide*: provides guidelines for developing applications using the EtherMind Mesh Core Stack and Models APIs
- *EtherMind User Guide—Mesh Command Line Interface*: how to use commands to configure and control the devices on the mesh network

Bluetooth is a registered trademark of Bluetooth SIG, Inc. Windows is a registered trademark of Microsoft Corporation. All other brand names and product names appearing in this document are trademarks of their respective holders.

ON Semiconductor and ON are trademarks of Semiconductor Components Industries, LLC dba ON Semiconductor or its subsidiaries in the United States and/or other countries. ON Semiconductor owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of ON Semiconductor's product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent-Marking.pdf. ON Semiconductor reserves the right to make changes without further notice to any products herein. ON Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does ON Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using ON Semiconductor products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by ON Semiconductor. "Typical" parameters which may be provided in ON Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. ON Semiconductor does not convey any license under its patent rights nor the rights of others. ON Semiconductor products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use ON Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold ON Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that ON Semiconductor was negligent regarding the design or manufacture of the part. ON Semiconductor is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner.

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Literature Distribution Center for ON Semiconductor
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA
Phone: 303-675-2175 or 800-344-3860 Toll Free USA/Canada
Fax: 303-675-2176 or 800-344-3867 Toll Free USA/Canada
Email: orderlit@onsemi.com

N. American Technical Support: 800-282-9855 Toll Free USA/Canada
Europe, Middle East and Africa Technical Support: Phone: 421 33 790 2910

ON Semiconductor Website: www.onsemi.com

Order Literature: http://www.onsemi.com/orderlit

For additional information, please contact your local Sales Representative