# AN1162: Using the Manufacturing Library for EmberZNet

This document describes how to use the manufacturing library and its associated components in Simplicity Studio for performing RF tests during the characterization and manufacturing phases.

**Note:** Zigbee EmberZNet SDK 7.0 introduced a new component-based architecture, along with a Project Configurator and other tools to replace AppBuilder and plugin configuration. In general, the new software components are comparable to the plugins. Unless otherwise specified, the term 'component' should be understood to refer to the corresponding plugin, if you are working in SDK 6.10.x or lower. When applicable, instructions for both version 7.0 and higher and 6.10.x and lower are provided. For more information, see *AN1301: Transitioning from Zigbee EmberZNet SDK 6.x to SDK 7.x*.

Silicon Labs recommends that you be familiar with creating example applications as described in *the following quick-start guides*

- *QSG106: Zigbee EmberZNet PRO Quick-Start Guide for SDK 6.x and Lower*
- *QSG180: Zigbee EmberZNet PRO Quick-Start Guide for SDK 7.0 and Higher*

**KEY POINTS**

- Describes the manufacturing library functions.
- Provides instructions on enabling the manufacturing library components in Simplicity Studio.
- Includes examples of using the manufacturing library for RF tests.

# 1 Introduction

The manufacturing and functional test library is used for testing and verifying the RF component of products at manufacture time. Customers can optionally include this library in their application code and run at manufacturing time. Any interface to the library is handled by the application.

Silicon Labs recommends using the manufacturing test library for customers who are using mature applications, regardless of the testing phase. However, this library is typically used in the low-volume and high-volume phases of testing where additional programming steps add unnecessary test time. This library allows for accessibility to a test mode within the customer's application and removes the need for multiple application bootloads or multiple programming steps within the manufacturing process.

For an overview of all phases of manufacturing testing, see *AN718: Manufacturing Test Overview*. For details, see *AN700.1: Manufacturing Test Guidelines for the EFR32*.

The specifications for the manufacturing library APIs are detailed in the EmberZNet API reference accessed through Simplicity Studio documentation list or https://docs.silabs.com/.

The EmberZNet SDK demonstrates the use of this library via the manufacturing library components. The following sections provide guidance on using the manufacturing library components for manufacturing tests.

# 2 Using the Manufacturing Library in an EmberZNet Application

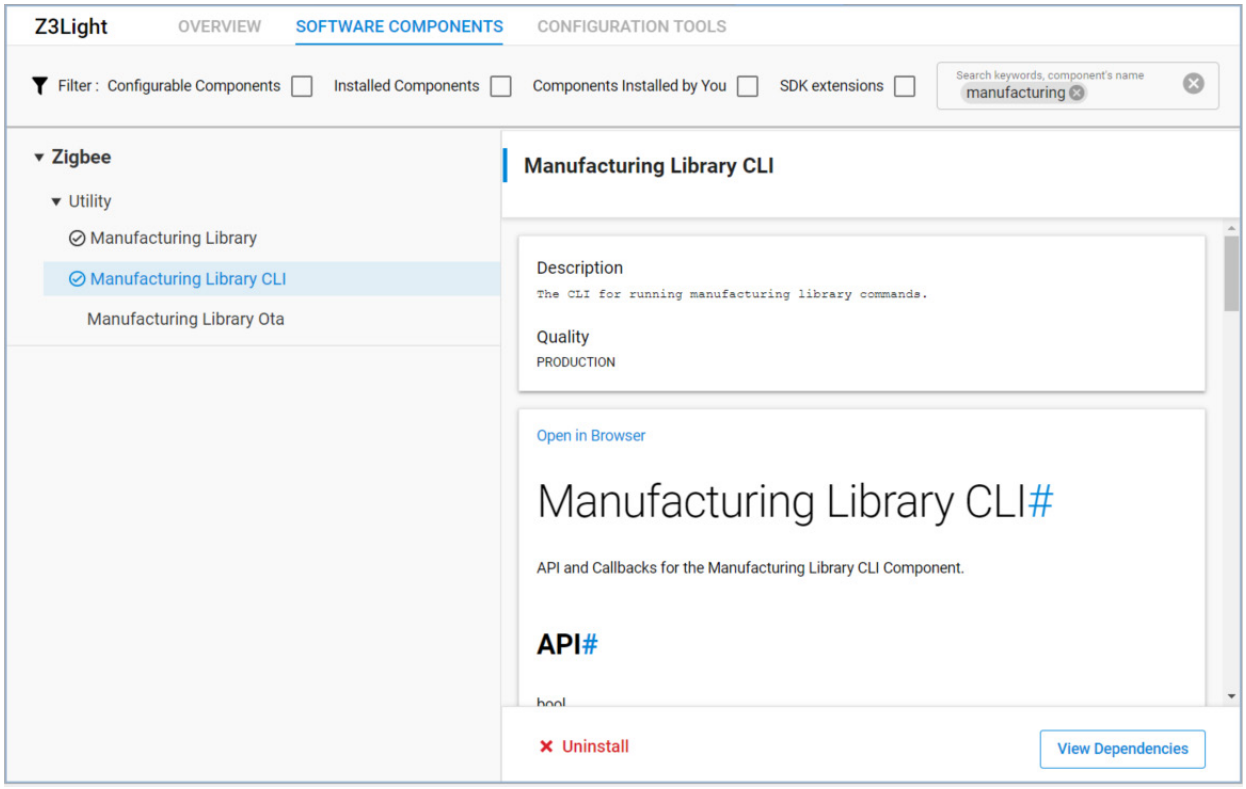
The Manufacturing Library is provided as a component that can be optionally included in the application code. It must be enabled at build time in order to use the library functions in the Command Line Interface (CLI). The Manufacturing Library CLI component uses the Manufacturing Library API functions for testing RF performance and is accessed via serial UART.

For testing unattended devices, the Manufacturing Library Ota component is used where the test commands are sent over-the-air to the device under test (DUT). This is useful in evaluating the radiated RF performance (TRP/TIS) where connecting communication wires to the DUT is not preferred as it could provide interference and affect the RF performance. The Ota component is enabled with a procedure similar to the following. Use of this component is not described in this document.

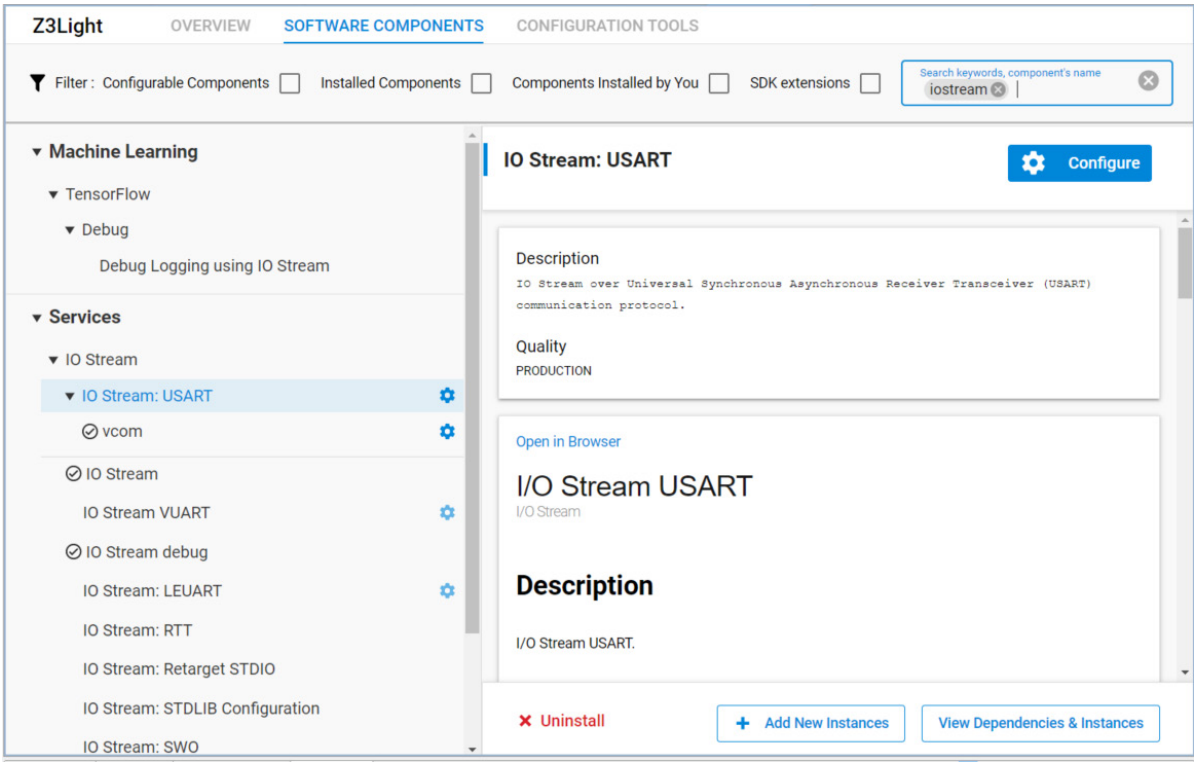
The following instructions assume you have installed Simplicity Studio and the EmberZNet SDK (software development kit), and that you have a SoC Zigbee project open in the Simplicity IDE (integrated development environment). Note that, in case of host/NCP (and Zigbeed/RCP in SDK 7.1.1 and above), the manufacturing library should be enabled on the NCP/Zigbeed side, while the Manufacturing Library CLI component should be enabled on the host application. For guidance on how to build a Zigbee application, refer to the *Zigbee EmberZNet PRO Quick-Start Guide*.

**In SDK 7.0 and higher:**

1. If the *<project>.slcp* file is not open, double-click it in the Project Explorer view to open it. Go to the Software Components tab and enter the keyword 'manufacturing' to search for the Manufacturing Library and Manufacturing CLI components. Install both components.

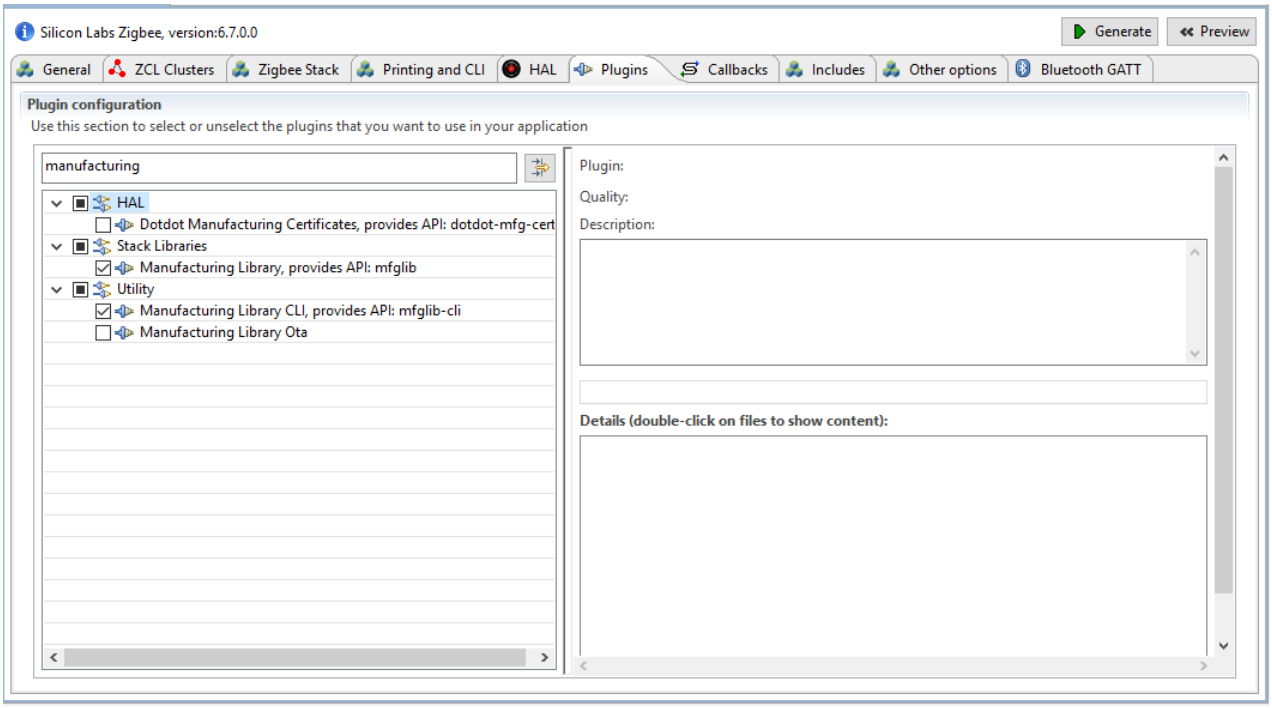


2. Ensure that the I/O Stream USART and associated components are installed as shown below.

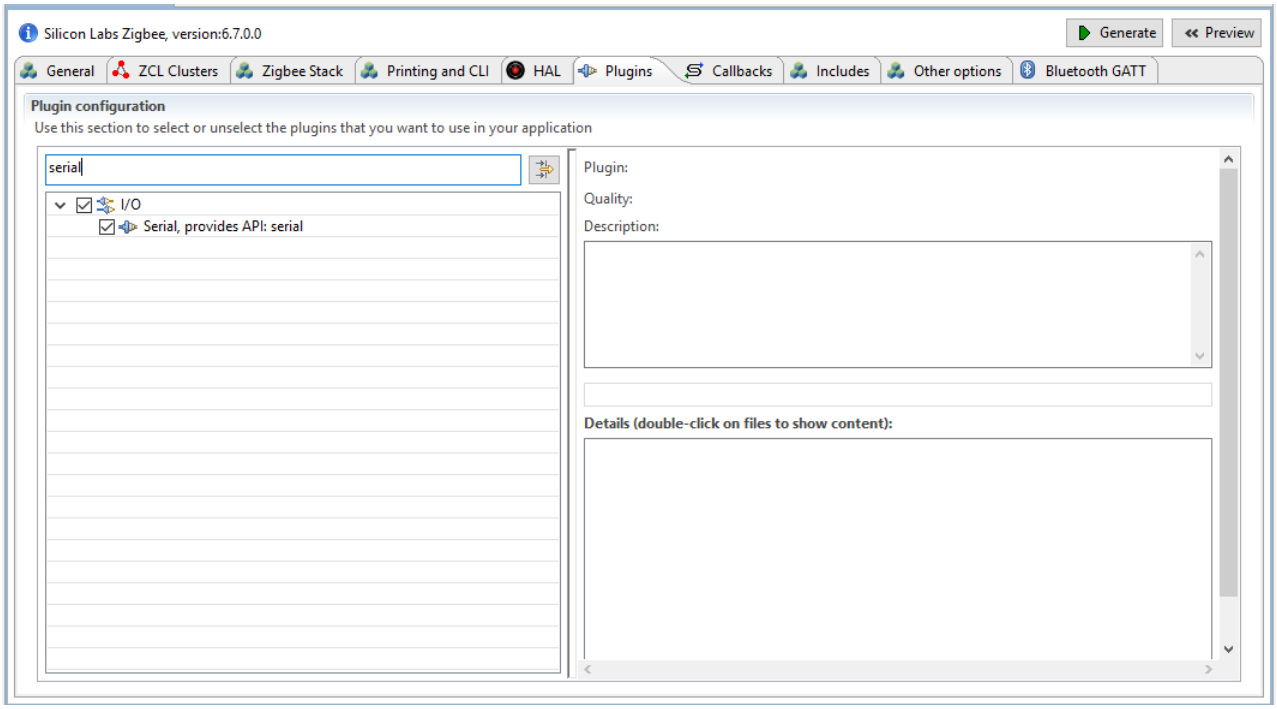



**In SDK 6.10.x and lower:**

1. In the Plugins tab, search for Manufacturing Library. Check (enable) the Manufacturing Library plugin in the Stack Libraries group and the Manufacturing CLI plugin in the utilities group. Note that if you only check the Manufacturing CLI plugin you will see an error when you generate the project files.



2. Ensure that the Serial plugin is enabled.




## 2.1 Command Line Interface

Gecko SDK 2.6.0 and above uses the serial interface to start the manufacturing test mode and test RF parameters, sleep current and other peripherals using the Manufacturing Library CLI plugin.

**Table 2.1. Manufacturing Library CLI Command List**

| Command (1) | Command Description | API Function | Arguments | | |
|---|---|---|---|---|---|
| | | | Name | Type | Description |
| start | Starts mfglib test mode | emAfMfglibStartCommand | useCallback | BOOLEAN | Use the Rx callback. |
| stop | Stops mfglib test mode | emAfMfglibStopCommand | N/A | N/A | |
| tone start | Starts a tone test | emAfMfglibToneStartCommand | N/A | N/A | |
| tone stop | Stops the tone test | emAfMfglibToneStopCommand | N/A | N/A | |
| status | Prints the current status of the manufacturing library | emAfMfglibStatusCommand | N/A | N/A | |
| set-channel | Sets the channel used by the manufacturing library for testing. | emAfMfglibSetChannelCommand | channel | INT8U | The 802.15.4 channel number |
| set-power | Set the power for manufacturing test | emAfMfglibSetPowerAndModeCommand | powerLevel | INT8S | The power level |
| set-power | Set the power for manufacturing test | emAfMfglibSetPowerAndModeCommand (2) | powerMode (2) | INT16U | The power mode |
| stream start | Starts the stream test | emAfMfglibStreamStartCommand | N/A | N/A | |
| stream stop | Stop the stream test | emAfMfglibStreamStopCommand | N/A | N/A | |

| Command (1) | Command Description | API Function | Arguments | | |
|---|---|---|---|---|---|
| | | | Name | Type | Description |
| programEui | Program a custom EUI | emAfMfglibProgramEuiCommand | duration | IEEE_ADDRESS | IEEE address to program |
| set-options | Sets transmission options in the manufacturing library for testing | emAfMfglibSetOptions | options | INT8U | The options bitmask (0 = normal transmit, 1 = CSMA transmit) |
| sleep | Sleep | emAfMfglibSleepCommand | duration | INT16U | Sleep duration (mS). |
| send random | Send a series of test packets with random data | emAfMfglibSendCommand | numPackets | INT16U | The number of packets to send |
| send random | Send a series of test packets with random data | emAfMfglibSendCommand | length | INT8U | The length of the packet to send |
| send test | Send a series of test packets with fixed data | emAfMfglibSendCommand | numPackets | INT16U | The number of packets to send |
| send test | Send a series of test packets with fixed data | emAfMfglibSendCommand | length | INT8U | The length of the packet to send |
| send message | Send a test message with the specified data in it. | emAfMfglibSendMessageCommand | data | OCTET_STRING | The hex data to send, up to 16 bytes |
| send message | Send a test message with the specified data in it. | emAfMfglibSendMessageCommand | numPackets | INT16U | The number of packets to send. |

**Notes**

1. All commands in the table are prepended with `plugin mfglib` in the actual implementation of the command in the CLI.
2. The `powerMode` argument is ignored in the manufacturing test mode for EFR32. Enter **1** when setting the power level using the `set-power` command.

# 3 Test Examples with Manufacturing Library CLI Commands

To conduct tests using the Manufacturing Library CLI commands, the DUT needs to enter manufacturing test mode. The following sections show how to set up the test environment for specific RF tests that are commonly used during the characterization and manufacturing phases, and in test houses for certification.

**Note:** The DUT exits manufacturing test mode upon shutdown, so the manufacturing library start command is not persistent across reboots. The test mode needs to be entered on start up before using the manufacturing library for radio parameter changes or performing any tests. All examples in this section assume that the manufacturing test mode has not been started after DUT startup.

## 3.1 How to Set the Channel and PA Output Power on DUT

The Manufacturing library can be used to set the RF power level of the PA at a specific channel. The ability to adjust the PA output power is required for all transmission tests. In this example, the DUT enters manufacturing test mode, sets the channel and power, and checks the status to verify the parameters to set up the environment for transmission tests. Note that the argument for the `start` command is 0 because the Rx callback is needed only for receiver tests.

```
plugin mfglib start 0           // Enter manufacturing test mode
plugin mfglib set-channel 11    // Set channel to 11 (2405 MHz)
plugin mfglib set-power 18 1    // Set power level to 18 dBm
plugin mfglib status            // Verify radio parameters that were set in previous steps
plugin mfglib stop              // Exit manufacturing test mode
```

## 3.2 How to Set Up DUT to Transmit CW Tone

The command sequence below uses the manufacturing library CLI commands to start a TX tone that can be used to measure the PA output power, phase noise, frequency offset and spurious emissions using a spectrum analyzer. The TX tone is a continuous unmodulated transmission.

```
plugin mfglib start 0             // Enter manufacturing test mode
plugin mfglib set-channel 11       // Set channel to 11 (2405 MHz)
plugin mfglib set-power 18 1       // Set power level to 18 dBm
plugin mfglib status               // Verify radio is in test mode and the parameters that were set
                                      in previous steps
plugin mfglib tone start          // Start TX tone and measure the output power/phase noise/spurious
                                     emissions
plugin mfglib tone stop           // Stop TX tone
plugin mfglib stop                // Exit manufacturing test mode
```

## 3.3 How to Set Up DUT to Transmit Data Stream

Similar to the above example, the command sequence below uses the manufacturing library CLI commands to start a TX stream that can be used to test the power spectral density mask and EVM (Error Vector Magnitude). The TX stream is a continuous transmission of random stream of characters.

```
plugin mfglib start 0             // Enter manufacturing test mode
plugin mfglib set-channel 11       // Set channel to 11 (2405 MHz)
plugin mfglib set-power 18 1       // Set power level to 18 dBm
plugin mfglib status               // Verify radio is in test mode and the parameters that were set
                                      in previous steps
plugin mfglib stream start         // Start TX stream and measure the output power
plugin mfglib stream stop          // Stop TX tone
plugin mfglib stop                 // Exit manufacturing test mode
```

### 3.4 How to Set Up DUT to Send Packets

In order to observe the packets sent by the DUT, manufacturing library CLI commands can be used to send either random data or fixed data in the packet. In the example below 12 packets with random data are sent with 8 bytes packet length on channel 11 at 18 dBm.

```
plugin mfglib start 0              // Enter manufacturing test mode
plugin mfglib set-channel 11        // Set channel to 11
plugin mfglib set-power 18 1        // Set power level to 18 dBm
plugin mfglib status                // Verify radio is in test mode and the parameters that were set
                                       in previous steps
plugin mfglib send random 12 8      // Send 12 packets with random data of 8 bytes packet length
plugin mfglib stop                  // Exit manufacturing test mode
```

### 3.5 How to Set Up DUT to Receive Packets

For receiver tests, the DUT needs to enter the manufacturing test mode, set the channel to match the transmitter frequency, and check the reception of packets periodically using the `plugin mfglib status` command. This test routine assumes that there is a signal generator connected to the DUT or a golden node to transmit Zigbee packets. Note that the argument for the `start` command is 1 in this example because the Rx callback needs to be enabled in order to count the number of packets received.

```
plugin mfglib start 1              // Enter manufacturing test mode
plugin mfglib set-channel 11       // Set channel to 11 (2405 MHz)
plugin mfglib status               // Check the packet count number for any packets received
plugin mfglib stop                 // Exit manufacturing test mode
```

**Note:** The DUT needs to exit manufacturing test mode and enter again in order to reset the packet count to 0.

# 4 Revision History

**Revision 0.4**

December 2022

- Updated to reflect support for RCP.

**Revision 0.3**

June 2022

- Updated to EmberZNet SDK 7.x.

**Revision 0.2**

December 2020

- Deleted references to EM35x.
- Updated Note 2 for Table 2.1.

**Revision 0.1**

Initial release.

# Simplicity Studio

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

**IoT Portfolio**
www.silabs.com/IoT

**SW/HW**
www.silabs.com/simplicity

**Quality**
www.silabs.com/quality

**Support & Community**
www.silabs.com/community



SILICON LABS

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

**www.silabs.com**