Laird™
CONNECTIVITY

User Guide

Sterling-EWB Development Kit – Software

Version 1.0

REVISION HISTORY

Version	Date	Notes	Contributor(s)	Approver
0.1	03 July 2019	Initial release	Dave Neperud Randy Scott	Jay White

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

CONTENTS

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

1 Introduction

1.1 Scope

This document describes the Sterling-EWB STM development kit and how to use it for development of embedded Wi-Fi and/or Bluetooth Low Energy (BLE) applications. It also provides an overview of the hardware along with a description of setting up the software development environment, programming the board, and some sample projects to get you started.

1.2 Related Laird Products

Laird Part Number	Description
455-00030	EWB DVK with integrated chip antenna
455-00031	EWB DVK w/ u.FL antenna connector

2 Sterling-EWB Description

The Sterling-EWB is a Laird Sterling-LWB 802.11b/g/n Bluetooth 4.2 radio with an ST Micro STM32F412 Cortex M4 microprocessor. The development kit takes all of the pins from the Sterling-EWB and brings them out to connectors so that you can develop your product.

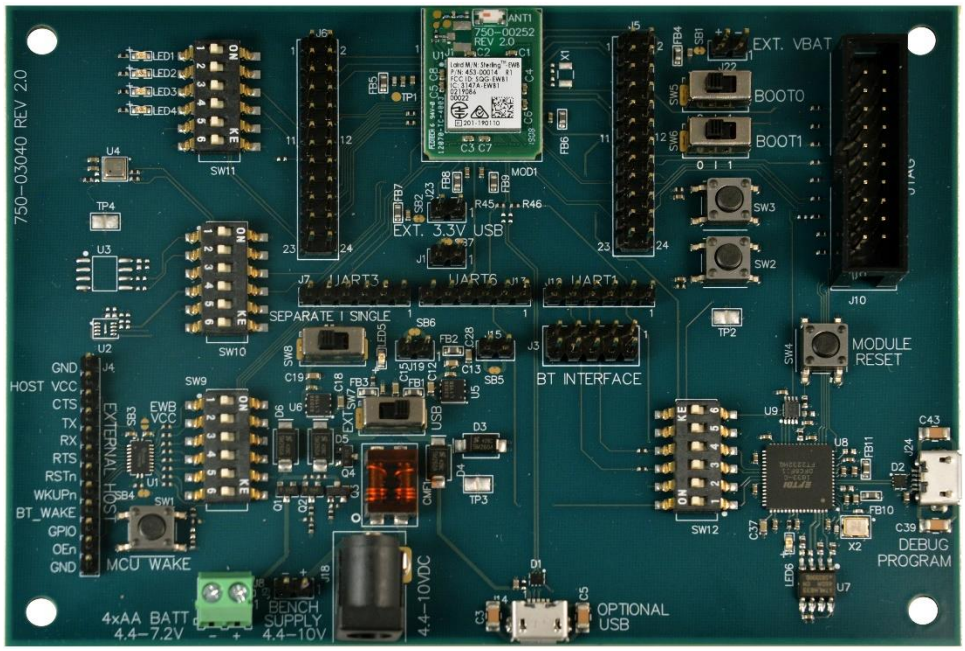


Figure 1: Sterling-EWB development kit

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

2.1 Sterling-EWB Development Kit Hardware

The hardware components on the Sterling-EWB development kit are shown in the following block diagram.

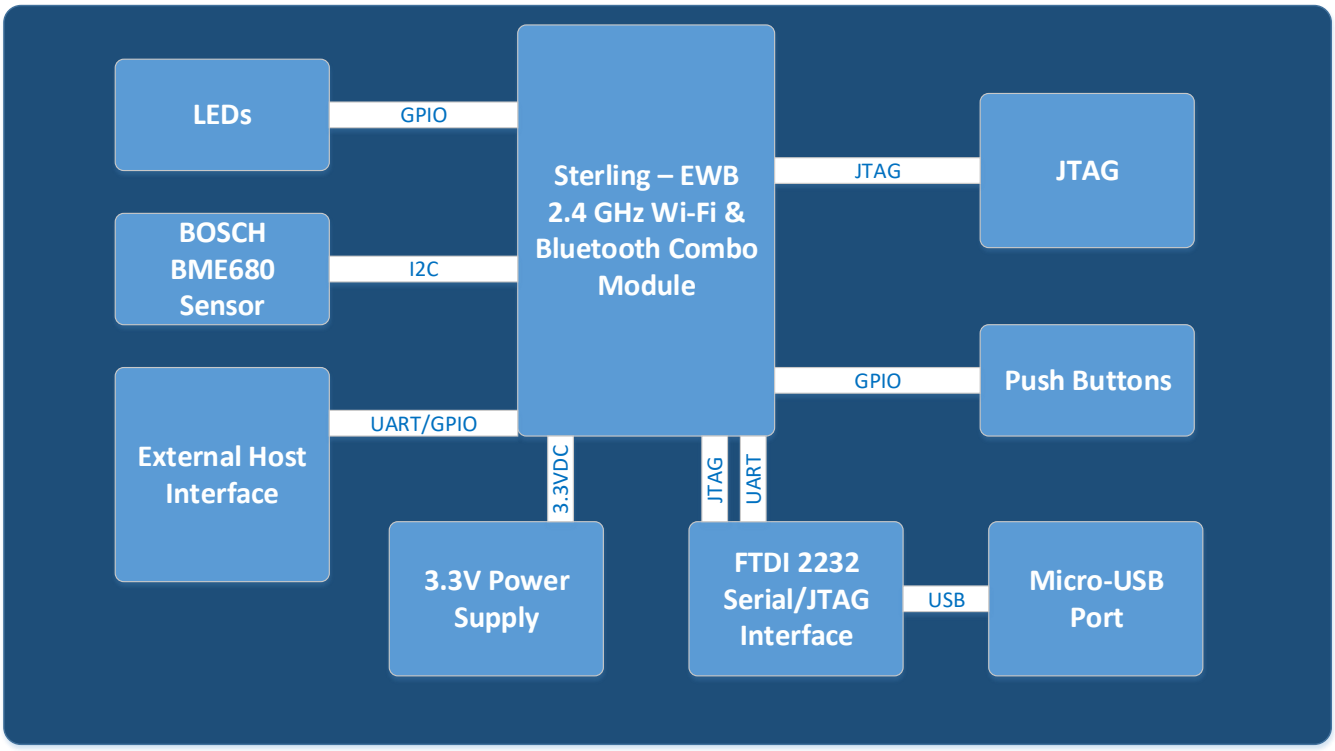


Figure 2: Sterling-EWB block diagram

2.2 Sterling-EWB Wi-Fi + Bluetooth Combo Module

The Sterling-EWB module provides 802.11b/g/n and Bluetooth 4.1 functionality along with an embedded STM32F412 microcontroller in a form factor as small as a 10x10 mm SiP. Along with the SiP, which does not include an antenna, there are two modules that provide an onboard chip antenna or a connector to be used with an external antenna.

2.3 USB JTAG and Serial Interface

The development kit provides a single micro-USB interface for JTAG programming and access to the UART1 serial interface of the STM32F412 MCU. The JTAG interface is provided via a standard FTDI FT2232H IC with its VID and PID updated to be a WICED debugger (VID: 0x0a5c PID: 0x43fa).

2.4 Memory

The STM32F412 included inside the EWB module has one Mbyte of internal flash memory and 256 kbytes of internal RAM. The module also includes an internal two Mbyte SPI flash.

In the default configuration, the internal flash of the STM32 is used for the bootloader, configuration storage blocks (DCT), and the application firmware. The SPI flash is used to store the WLAN firmware required by the radio. The firmware is approximately 400 kbytes in size. The remainder of the SPI flash can be used by an application for additional storage.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

2.5 User I/O

The development board provides two user push-button switches (SW1 and SW2) and six LEDs. The push-button switch SW1 is attached to STM32 pin PD1 and the switch SW2 is attached to STM32 pin PC13.

Four LEDs are controlled by GPIO pins on the STM32: LED1 (green) on pin PB13, LED2 (red) on pin PB15, LED3 (blue) on pin PE7, and LED4 (red) on pin PE8. LED5 is a power LED connected to the 5V power input to the development board. LED6 is a UART activity indicator.

2.6 Bosch Air Quality Sensor

The development board includes the Bosch Sensortec BME680 air quality sensor. Using the Bosch Sensortec Environmental Cluster (BSEC) software, the sensor can report temperature, humidity, atmospheric pressure, and a measure of indoor air quality (IAQ).

3 Software Development with the Sterling-EWB Development Kit

3.1 Required Equipment

The Sterling-EWB is designed to be used directly with the WICED SDK.

The Sterling-EWB development kit can either requires a 1A +5VDC power supply or it can be powered from the USB to operate. The barrel connector (J18) is configured for 5VDC center positive. Power can be provided to the DEBUG PROGRAM USB port. The DEBUG PROGRAM USB port also provides JTAG and USB-to-serial adapter on the Sterling-EWB development kit. You can attach a PC via micro-USB cable to the micro-USB port on the Sterling-EWB Development Kit.

3.2 Required Software

3.2.1 Cypress WICED Studio and WICED SDK

The Sterling-EWB depends on the Cypress WICED SDK for software development. Cypress provides an integrated development environment (IDE) based on Eclipse and the GCC toolchain for ARM. Included in the SDK are a network driver, RTOS, networking utilities, radio firmware, and sample projects demonstrating use of Cypress radios. The Sterling-EWB is based on the Cypress BCM4343W radio and can be used with the WICED SDK by targeting the LAIRD_EWB platform files.

The following are included in the WICED SDK:

- Development environment based on eclipse IDE and the GCC toolchain for ARM
- Tools for programming and debugging embedded code on host MCU targets with several JTAG adapter options
- Makefiles designed to easily build projects based on a “build string” specified in the IDE
- RTOS abstraction layer allowing software developers to choose between ThreadX and FreeRTOS operating systems. The abstraction layer helps move applications between different RTOS configurations with minimal changes to application-level code.
- TCP/IP stack abstraction layer allowing software developers to choose between NetX and LwIP networking stacks for their application. This abstraction layer allows applications to target either stack with minimal changes to application-level code.
- Demos and code snippets showing how to perform common operations for Wi-Fi and Bluetooth applications
- WLAN and Bluetooth firmware and driver files required by Cypress radios

Who would use it?

- The SDK is intended for use by developers experienced in embedded C programming on microcontrollers. Familiarity with software engineering concepts such as make files, RTOS concepts, console debugging, and basic hardware configuration/debugging skills is very helpful.
- Developers should also be familiar with basic Wi-Fi concepts and TCP/IP networking, including Wi-Fi security modes, station vs. access point operation, BSD sockets and networking utilities (DHCP, DNS, ping, telnet, etc.).

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

- When using the BLE features of the WICED SDK, developers should be familiar with Bluetooth concepts including central vs. peripheral roles, components of BLE profiles and services, BLE advertising and tradeoffs associated with BLE radio configuration parameters.

Using the WICED SDK

- The WICED SDK provides tools for developers to create their own RTOS-based embedded software applications that utilize Wi-Fi and/or Bluetooth capabilities by including the appropriate libraries.
- We recommend developers start by building and testing the example “demo” and “snip” projects included with the WICED SDK. This provides a basic understanding of the IDE, build strings, JTAG programming and serial console debugging.

3.2.2 Downloading the Cypress WICED Studio SDK

The WICED Studio SDK can be downloaded from Cypress at its developer community page after signing up for a free developer account. The Cypress Studio SDK can be downloaded from the Cypress website. You must use version 6.2 or greater for the Sterling-EWB.

Note: Laird does not develop the WICED SDK or WICED Studio and does not provide technical support for them. For assistance with the WICED SDK and WICED Studio, please refer to the Cypress community page.

3.2.3 Adding the Sterling-EWB Platform to Your WICED Installation

The WICED SDK ships with many platforms already defined in the *43xxx_Wi-Fi/platforms* folder. Each subfolder provides the configuration and interface code for a particular target platform (Wi-Fi module + processor + development board).

For the Sterling-EWB development Kit, you must install the Sterling-EWB platform files.

Download the WICED Firmware Additions file from the Sterling-EWB product page. Expand the archive and copy the files to the top of the WICED-Studio-6.x.x folder that was created when you installed the WICED SDK. The overwrites some old files and creates the following two new directories:

- 43xxx_Wi-Fi/platforms/LAIRD_EWB
- 43xxx_Wi-Fi/apps/laird

3.2.4 Using the Cypress WICED Studio

To use the Cypress WICED Studio, follow these steps:

1. Launch the WICED Studio from its installed location from the previous step.
2. Read the README.txt file for details on what the installed release of the WICED Studio provides.

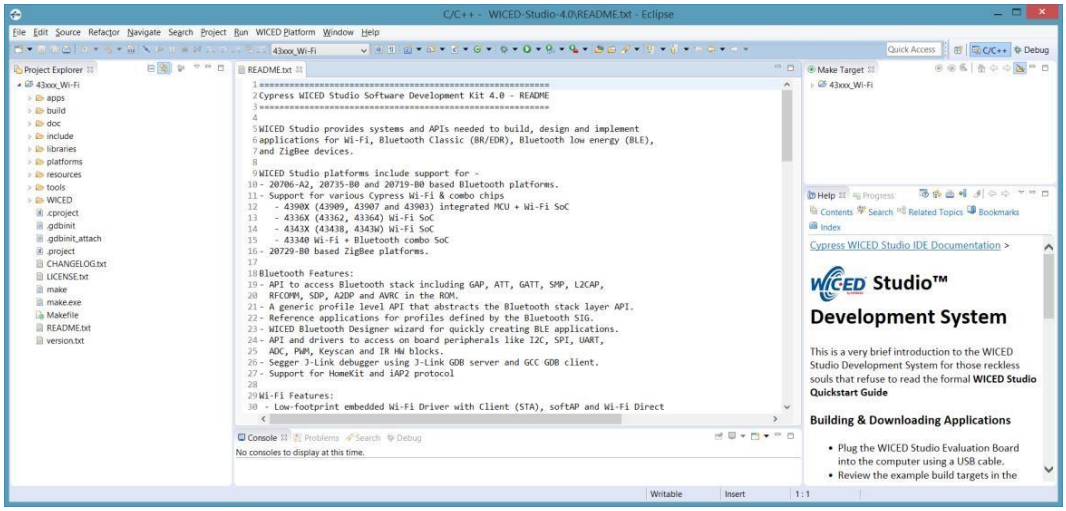


Figure 3: WICED Studio main window

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

The following section of this guide provides a basic workflow for developing a Wi-Fi or Bluetooth application for use with the Sterling-EWB development kit. For more details, please refer to the documents included as part of the WICED SDK installation.

3.2.5 Understanding the WICED SDK Folder Structure

The Project Explorer on the left-hand side of WICED Studio provides a hierarchical folder view rooted at *43xxx_Wi-Fi* within the SDK installation. All SDK files branch from the *43xxx_Wi-Fi* subfolder (Table 1).

Table 1: WICED SDK folder structure

Folder Name	Description
apps	Contains source code for embedded applications that can be built by WICED Studio
doc	Contains documentation included as part of WICED Studio and WICED SDK
include	WICED framework include path for targets being built within WICED Studio
libraries	Contains various peripheral and protocol libraries that can be used by embedded applications built within WICED Studio
platforms	Contains folders for each supported target platform supported by the WICED SDK. These folders are targeted by the build string specified in the Make Target view
Resources	Contains files stored within the flash filesystem for embedded applications, certificates, firmware, and other resource files
Tools	Contains programming and debugging tools such as OpenOCD and other scripts used by WICED Studio during the build process
WICED	Contains the source files for the WICED framework, RTOS, networking, and drivers

3.3 WICED Software Examples

3.3.1 Finding Demo and Code Snippets

The **43xxx_Wi-Fi/apps** folder is the place to go for example code to run with your Sterling-EWB development kit.

Note: Not all demo and snip projects distributed with the WICED SDK are designed for use with all radio modules.

This guide highlights several that are known to work properly with basic Wi-Fi and Bluetooth functionality on the Sterling-EWB (BCM4343W). Some projects may not be supported on the BCM4343W or may require code modifications to either the demo/snip code or the WICED SDK to function properly.

IMPORTANT: For snip or demo projects whose .mk file specifies a VALID_PLATFORMS entry, you must add the following line to the .mk file to allow building the project for the Sterling-EWB platform:

```
VALID_PLATFORMS += LAIRD_EWB
```

3.3.2 Snippets for Learning the WICED Framework

Table 2 shows several snippets that are useful for learning WICED framework concepts and APIs.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

Table 2: Useful snippets for learning WICED framework concepts and APIs

Subfolder of 43xxx_Wi-Fi/apps/snip	Description
stdio	Demonstrates how to use printf() and scanf() c library functions to interact with the debug UART
gpio	Demonstrates use of the WICED APIs for manipulating general purpose I/O (GPIO)
spi_slave	Demonstrates use of SPI for master and slave device configurations via the WICED SDK API
uart	Demonstrates use of UARTs via the WICED SDK API
dct_read_write	Demonstrates use of the Device Configuration Table (DCT) used throughout the WICED SDK

3.3.3 Snippets for Wi-Fi

Table 3 shows several snippets that are useful for learning how to operate the Sterling-EWB WiFi radio for basic networking functionality.

Table 3: Snippets for learning how to operate the EWB Wi-Fi radio for basic network functionalality

Subfolder of 43xxx_Wi-Fi/apps/snip	Description
ping_powersave	Demonstrates Wi-Fi client mode, ICMP ping, MCU powersave, and Wi-Fi powersave. Takes Wi-Fi network configuration from *43xxx_Wi-Fi/include/default_wifi_config_dct.h*
scan	Demonstrates performing a Wi-Fi AP scan. Results of the scan are displayed on the debug UART
apsta	Demonstrates use of Soft AP mode and client mode simultaneously
tcp_client	Basic example of a TCP/IP client
tcp_server	Basic example of a TCP/IP server
udp_transmit	Basic example of transmitting UDP packets
udp_receive	Basic example of receiving UDP packets

3.3.4 Snippets for Bluetooth

The table below shows several snippets that are useful for learning how to operate the Sterling-EWB Bluetooth Low Energy radio in both peripheral and central modes.

Table 4: Snippets for learning how to operate the EWB BLE radio in peripheral and central modes

Subfolder of 43xxx_Wi-Fi/apps/snip	Description
bluetooth/ble_hello_sensor	Demonstrates a simple BLE peripheral that advertises a simple *hello* service for learning the WICED SDK BLE apis. For more information on the BLE APIs, see 43xxx_WiFi/doc/API.html
bluetooth/bt_dualmode_server	Demonstrates simultaneous BLE GATT and Bluetooth Classic RFCOMM servers
<BLE Central Role>	*At the time of publishing this document, there is not a snippet that demonstrates BLE central role even though the SDK supports it. To find API features related to BLE central role, refer to the "43xxx_Wi-Fi/doc/API.html" documentation under the Components>Bluetooth>Device Management>BLE section for details.*

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

3.4 Using WICED Studio to Edit and Build Applications

3.4.1 Application Makefiles and Source Files

Applications can be edited by adding/editing files from the Project Explorer view on the left side of WICED Studio. For example, to view the *snip.scan* source file, expand the following folders: *apps* > *snip* > *scan* to access the *scan.c* and *scan.mk* files. All projects have an associated .mk file to specify the source files and other options to the WICED Studio builder. This file MUST be named with the same name as the folder containing it (e.g., the folder for the application is named *scan*, therefore the build environment looks for a file named *scan.mk* within that when building the scan application).

The .mk file provides at least the NAME of the application and a list of source files required to build it. For example, $(NAME)_SOURCES. See the *scan.mk* file for a simple example.

There are many other options that can be set in the project .mk file but describing these is outside the scope of this document. To learn more about this, take a look at the comments in the top-level *43xxx_Wi-Fi/Makefile* or check out the numerous project.mk files provided with the SDK in *43xxx_WiFi/apps/snip/** and *43xxx_Wi-Fi/apps/demo/*.*

3.4.2 Building an Application

To build an application, you must create a make target that contains a build string that specifies the application name, target platform, operations, and other flags, if necessary.

To create a new make target, follow these steps:

1. In the Make Target tab, right-click *43xxx_Wi-Fi.*
2. Select **New…**

3. For Target name, enter the name of the application starting with the subfolder of *apps* and using dot (.) as the path separator (e.g., *snip.scan*) followed by the target name (e.g., LAIRD_EWB) followed (optionally) by the RTOS (e.g., ThreadX), networking stack (e.g., NetX), and interface (e.g., SDIO) – all separated by dashes.

   Follow this with a space and one or more commands to execute (such as *download*, *download_apps,* and/or *run*).

   For a detailed listing of the possible options for build strings, see the USAGE_TEXT defined in *43xxx_Wi-Fi/Makefile*, near the top of the file.

To build the snip.scan application, follow these steps:

1. Double-click the **clean** make target to clean any previously built target files.

2. Create a make target with the following build string:

   snip.scan-LAIRD_EWB-ThreadX-NetX-SDIO

3. Double-click the make target created in step 2.

   The Console tab near the bottom of the WICED Studio window displays the status of the build. Once the build is complete, the output files are located in the *43xxx_Wi-Fi/build* folder.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

3.5 Downloading an Application to Your Sterling-EWB Development Board

3.5.1 Installing the WICED JTAG Adapter Driver

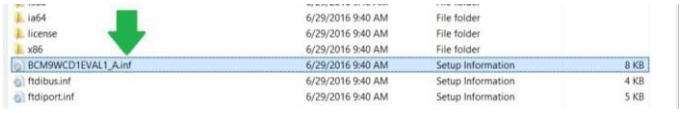
For your Windows PC to recognize the Sterling-EWB development board as a WICED-compatible JTAG adapter, you must first install drivers included with the WICED SDK. To install these drivers, follow these steps:

1. From the Device Manager, right-click **Other devices > USB <-> Serial Converter** and click **Update Driver Software**.

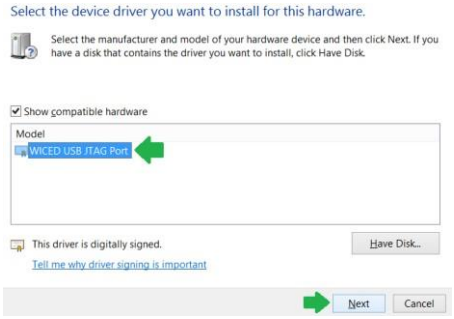
2. Click **Browse my computer for driver software**.

3. Click **Let me pick from a list of device drivers on my computer**.

4. Select **Show All Devices**, then click **Next**.
5. Click **Have Disk**.
6. Use the **Browse…** button to navigate to the location where you installed the WICED SDK.
7. Select the *43xxx_Wi-Fi/tools/drivers/BCM9WCD1EVAL1/BCM9WCD1EVAL1_A.inf* file.
8. Click Open.
9. Click **OK** to select this as the driver for the USB <-> Serial Converter.



10. Select **WICED USB JTAG Port** and click **Next**.



11. Click **Close** to finish installing the WICED USB JTAG Port. If the installation is successful, a WICED USB JTAG Port displays in the device manager.

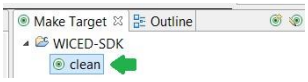
Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

3.5.2 Performing a *Build and Download* from WICED Studio

The programming process depends on the OpenOCD programming utility bundled with the WICED SDK from Cypress. During development, you can start the programming process directly from within WICED Studio. This is done by double-clicking a make target with the *download* directive specified as part of the build string. For more information about build strings within the WICED environment, refer to the *README.txt* in the top level *43xxx_Wi-Fi* folder of your WICED SDK installation.

For example, to build and download the **snip.scan** application to your Sterling-EWB Development Kit, do the following:

1. Double-click the **clean** make target to clean any previously built target files



2. Create a make target with the following build string:

snip.scan-LAIRD_EWB-ThreadX-NetX-SDIO download download_apps run

3. Double-click the make target created in step 2.

   The Console tab near the bottom of the WICED Studio window displays the status of the build and download. Once the build is complete, the output files are located in the *43xxx_Wi-Fi/build* folder.

   If the USB cable is attached to the PC and the JTAG driver is installed as described in 3.5.1, WICED Studio proceeds with programming the STM32F412 MCU after the build.

In this example, we specified the download_apps command. This instructs the IDE to also download the WLAN firmware to the external SPI flash. Once this is done for your board, *download_apps* may be omitted if you are just updating the application and not the WLAN firmware).

3.5.3 Downloading WLAN Firmware as Part of the *Build and Download* Step

As stated in the previous section, if you have never programmed the external SPI flash with WLAN firmware, you must do so at least once before attempting to build and run any applications on the STM32F412 that use the Sterling-EWB module.

To include programming of the WLAN firmware to the SPI flash in your build and download process, add *download_apps* after the *download* command in your build string. This triggers additional steps during the programming process to utilize the *sflash_write* application to write the WLAN firmware to the external SPI flash.

4 TROUBLESHOOTING

4.1 Understanding the Role of OpenOCD

WICED Studio depends on several tools that are bundled as part of the *43xxx_Wi-Fi* folder installed on your system. One of the important tools is OpenOCD which is used to communicate with target hardware for programming and debugging operations.

OpenOCD is located in the *43xxx_Wi-Fi/tools/OpenOCD* folder and provides not only binaries for several platforms but also .cfg and .tcl files that control the programming process. These files are used to inform OpenOCD how to perform each step during the programming and debugging process and are specifically provided for only certain target MCUs and JTAG adapters. If you are using a platform that is not listed in the .cfg files bundled with the OpenOCD utility inside the WICED SDK, you must provide .cfg files and .tcl scripts that work for your platform.

The Sterling-EWB utilizes the *BCM9WCD1EVAL1.cfg* file for the JTAG adapter definitions and *stm32f4x.cfg* for board configuration. These files are chosen for you automatically when your build string is set to use the LAIRD_EWB platform; no modifications to these files are necessary.

If you prefer to program the MCU from the command line, look at the make targets defined in *43xxx_Wi-Fi/tools/Makefiles/standard_platform_targets.mk* to get an idea of how the command line arguments are being sent to the OpenOCD executable. If you execute these command lines from the *43xxx_Wi-Fi* folder, you can perform programming operations outside of WICED Studio. This is useful for situations like manufacturing processes where you don't want to install the entire WICED Studio package to perform a programming operation.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

4.2 OpenOCD Log

During a build and download process initiated in WICED Studio by double-clicking a make target, several files are output into the *43xxx_Wi-Fi/build* folder. One file that is particularly useful for debugging problems with JTAG programming is the *43xxx_Wi-Fi/build/openocd_log.txt* file. This file provides a full dump of all steps that were attempted during the programming process and can be useful for tracking down issues. Check here first if you have trouble programming your board.

5 STERLING-EWB SENSOR DEMO

To demonstrate the use case of a cloud-connected sensor, each EWB development board comes pre-programmed with a demo application. This application uses BLE to configure the Wi-Fi network and provision the device. Once associated and provisioned, the device uses its internal temperature and humidity sensor and starts sending that data via AWS to the Laird Connect demo.

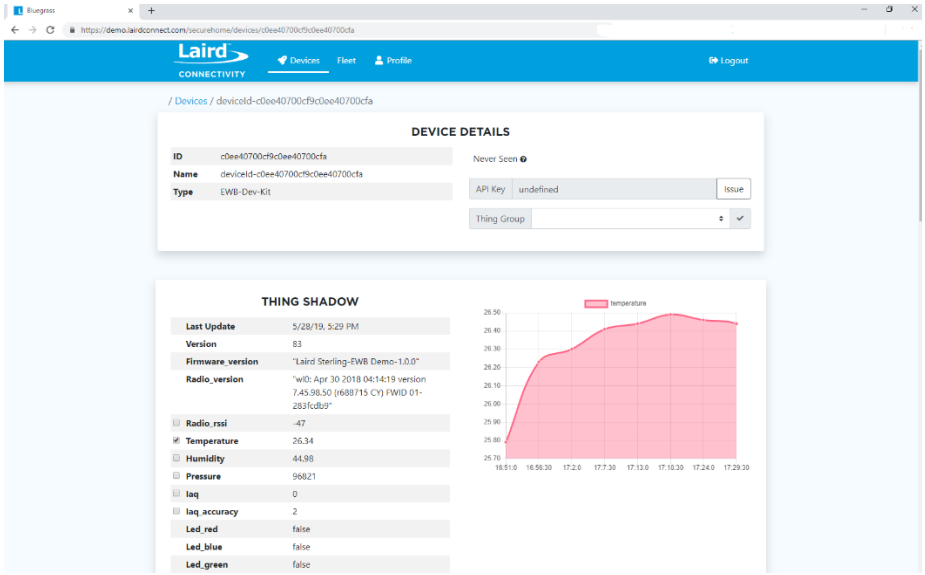


Figure 4: Web application showing sensor data reported from the Sterling-EWB

5.1 Downloading the Demo Source Code

You can download this sample code for use with WICED Studio from the Sterling-EWB Software Downloads section of the Laird Sterling-EWB product page. The project source code provides developers an overview of the steps involved when developing a typical BLE and Wi-Fi-enabled sensor including the following:

- Defining and implementing custom BLE profiles for communicating with a mobile app
  - Simple Sensor Profile
  - Wi-Fi Configuration Profile
  - AWS IoT Configuration Profile
- Steps required to assign Wi-Fi credentials to the device over a BLE connection
- Steps required to assign cloud authentication credentials to the device over a BLE connection
- Basic driver for interfacing with sensors over I2C and SPI bus and LED control
- Steps required to post sensor data to a cloud API via AWS

5.2 Building and Downloading the Demo

The Sterling EWB development board comes preloaded with the demo firmware, but if you would like to build or modify this firmware, the source code is provided. Make sure that you have already installed the Sterling-EWB WICED Firmware Additions as described in the *Adding the Sterling-EWB Platform to Your WICED Installation* section of this document. The demo source code is included in this package.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

The demo firmware includes software from Bosch called the Bosch Sensortec Environmental Cluster (BSEC) that Laird Connectivity is not allowed to distribute. In order to build the demo firmware, you must obtain the software from Bosch and copy a set of files into the demo firmware directory. The steps to do this are:

1. Go to this URL: https://www.bosch-sensortec.com/bst/products/all_products/bsec. Scroll to the bottom to read and agree to the license agreement. Click the Download link that appears.
2. Unzip the downloaded file.
3. Copy the two .h files and one .a file from …\algo\bin\Normal_version\gcc\Cortex_M4F in the BSEC release into …\43xxx_Wi-Fi\apps\laird\demo\sensor\ in the WICED installation.
4. Copy the three bme680.* files from …\API\ in the BSEC release into …\43xxx_Wi-Fi\apps\laird\demo\sensor\ in the WICED installation.
5. Copy the two bsec_integration.* files from …\examples\ in the BSEC release into …\43xxx_Wi-Fi\apps\laird\demo\sensor\ in the WICED installation.

To build, download and run the project, create a new make target: **laird.demo-LAIRD_EWB-ThreadX-NetX-SDIO download download_apps run**

This triggers a series of steps that can be monitored in the WICED Studio console view. After building the source code, if your development kit is attached via USB and drivers are installed, the application is programmed onto the development kit and starts running once the process has completed.

5.3 Configuring the development board

There are several sets of DIP switches on the development board that must be set correctly for the demo firmware to operate. The correct settings are:

- S1: 1 through 5 ON, 6 OFF
- S2: 1 through 3 and 6 ON, 4 and 5 OFF
- S3: All OFF
- S4: All ON

5.4 Monitoring the Debug Output of the Demo

Once the demo is running on your development kit, you can open a serial terminal on the USB serial adapter associated with your development kit (see Device Manager for a list of COM ports on Windows) to view debug output. The serial terminal settings should be set to the following:

- 115200 baud
- 8 data bits
- no parity
- 1 stop bit.

The debug output gives developers a view into the steps the application is taking such as enabling Wi-Fi, enabling BLE, joining a Wi-Fi network, reading sensor data, and reporting to AWS.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

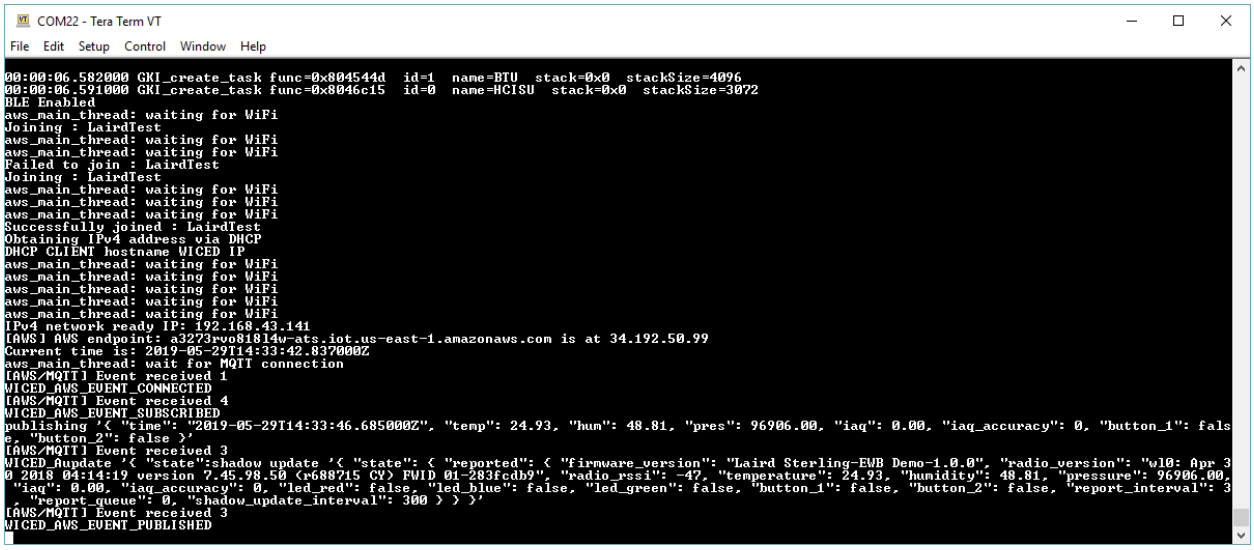


Figure 5: Debug output from the demo application

5.5 Using the EWB Mobile App

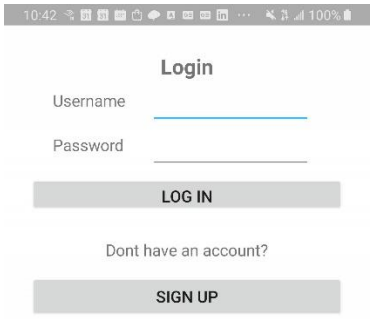
To configure and connect the Sterling-EWB for your local Wi-Fi network, Laird provides the EWB mobile app. This application walks you through the following:

- Creating an account
- Scanning and connecting to your Sterling-EWB over BLE
- Scanning for, selecting, and providing the passphrase for the Wi-Fi network to your Sterling-EWB
- Provisioning the client for use on the demo

Download and install the EWB mobile app to get started.

5.5.1 Creating an Account

Within the application, the first screen asks you to either sign in or sign up for a free account. This account is used to ensure only you have access to your device's sensor data and provide access to future enhancements to the Sterling-EWB demo.



Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

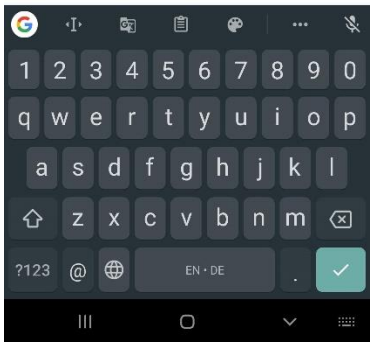


Figure 6: Creating an account

5.5.2 Device List View

Once you've created an account and/or have logged in, the app presents a list containing devices discovered nearby in a BLE scan.

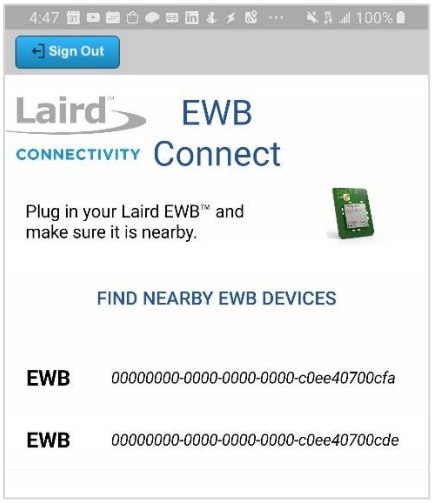


Figure 7: Nearby devices

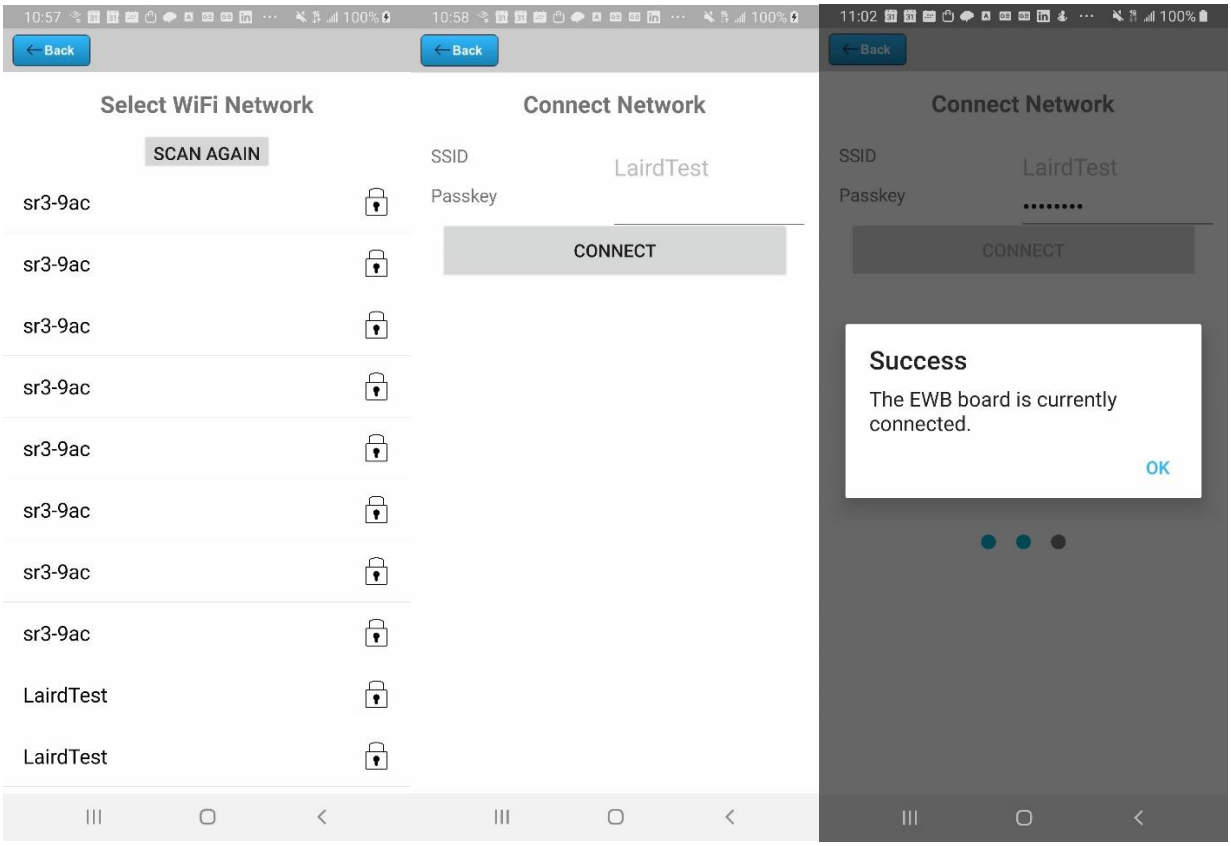
5.5.3 Configuring your Sterling-EWB for Wi-Fi

To configure your Sterling-EWB for Wi-Fi, follow these steps:

1. Click the address of your Sterling-EWB in the list of devices to start the Wi-Fi commissioning process.
   The app requests nearby Wi-Fi networks from the Sterling-EWB and presents them in a list.
2. Select the Wi-Fi network you'd like to use to connect your Sterling-EWB to the Internet.
3. Enter the passkey then press **Save** to commit the Wi-Fi credentials to the module.

   The app then registers the device with the AWS and the Sterling-EWB attempts to connect and begin reporting sensor data on a periodic basis.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610



5.6 Using the Web Application

Once your Sterling-EWB is connected to the Wi-Fi network and reporting its data to the AWS, you can see your data from a web browser.

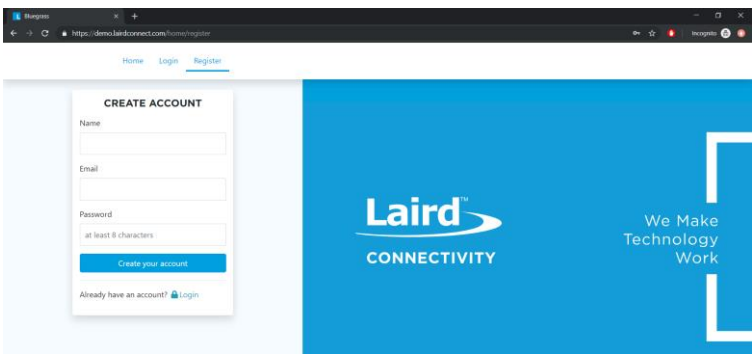
5.6.1 Accessing Web Application

We recommend using the Google Chrome browser to ensure the best experience. To bring up the Demo Web Application, visit the following URL: https://demo.lairdconnect.com.

5.6.2 Using the Demo Web Application

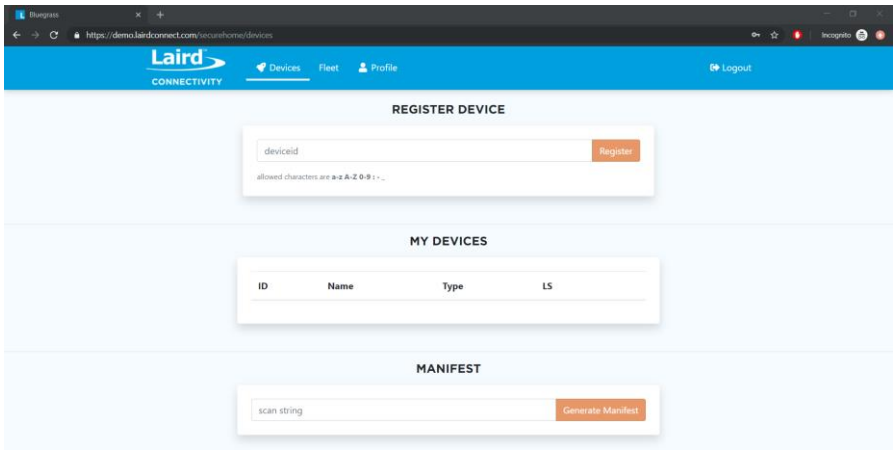
To use the Demo Web application, follow these steps:

1. On first login to the website, you must create an account.



Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

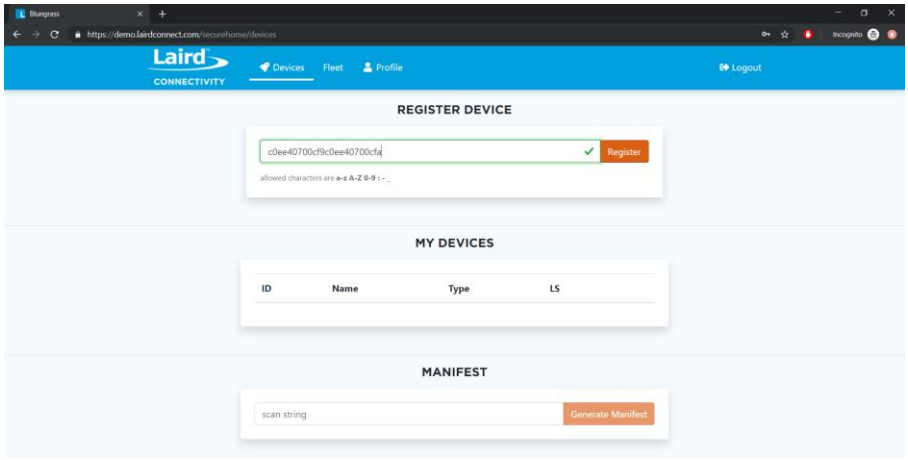
Once you have logged in to the website, a Register Device section displays.



2. Enter the client ID of your device.

   The client ID is the Wi-Fi MAC address and the BT MAC address combined. The BT MAC address can be found in the EWB Mobile App screen. The Wi-Fi MAC address is the BT MAC address minus 1.

   For example, if the EWB mobile app shows the MAC address as *c0ee40700cfa*, the client ID is *c0ee40700cf9c0ee40700cfa*.

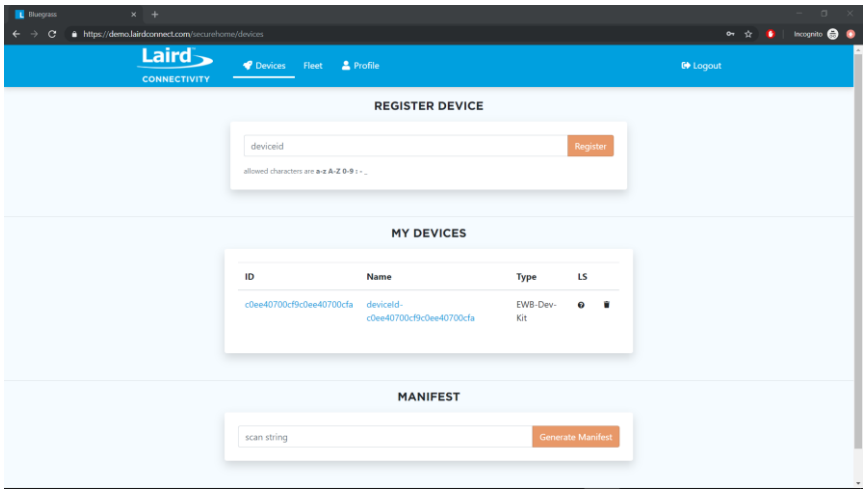


3. Once you have entered the device, click **Register**.

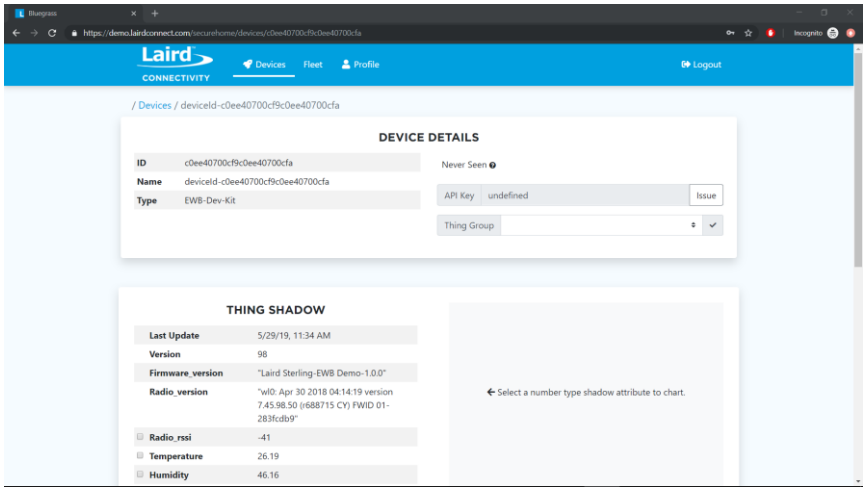
   If you see an error message that the device is already registered, press continue.

   You should now see your Sterling-EWB listed in the My Devices section.

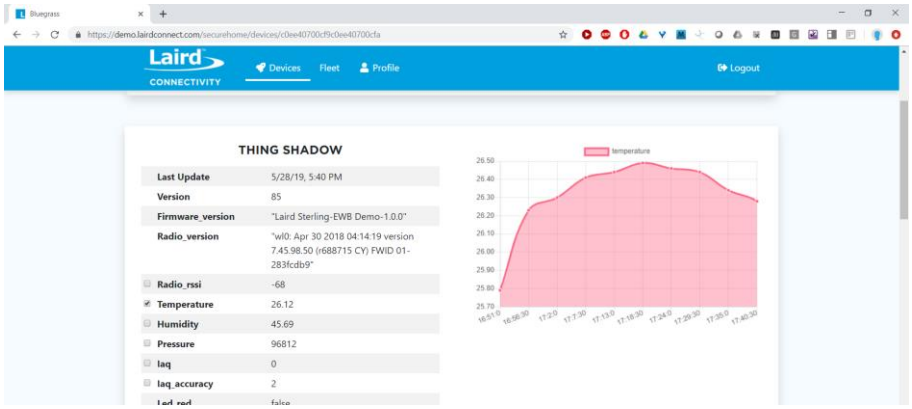
Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610



4. Click the ID of your device to advance to the device monitor screen.



This screen displays the device details and the data stream. There is also a section called Thing Shadow which allows you to graph readings from the sensor. The graph updates every five minutes.



Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610