SoC - Thunderboard

This sample application collects and processes sensor data from the Thunderboard Sense 2 or the Thunderboard EFR32BG22 board, and gives immediate graphical feedback to the user through the Thunderboard iOS/Android application.

Getting Started

To get started with Bluetooth and Simplicity Studio, see QSG169: Bluetooth® SDK v3.x Quick Start Guide.

To run this demo application, you need either a Thunderboard Sense 2 or a Thunderboard EFR32BG22 board, a mobile device, and the Thunderboard mobile application, available for iOS and Android.

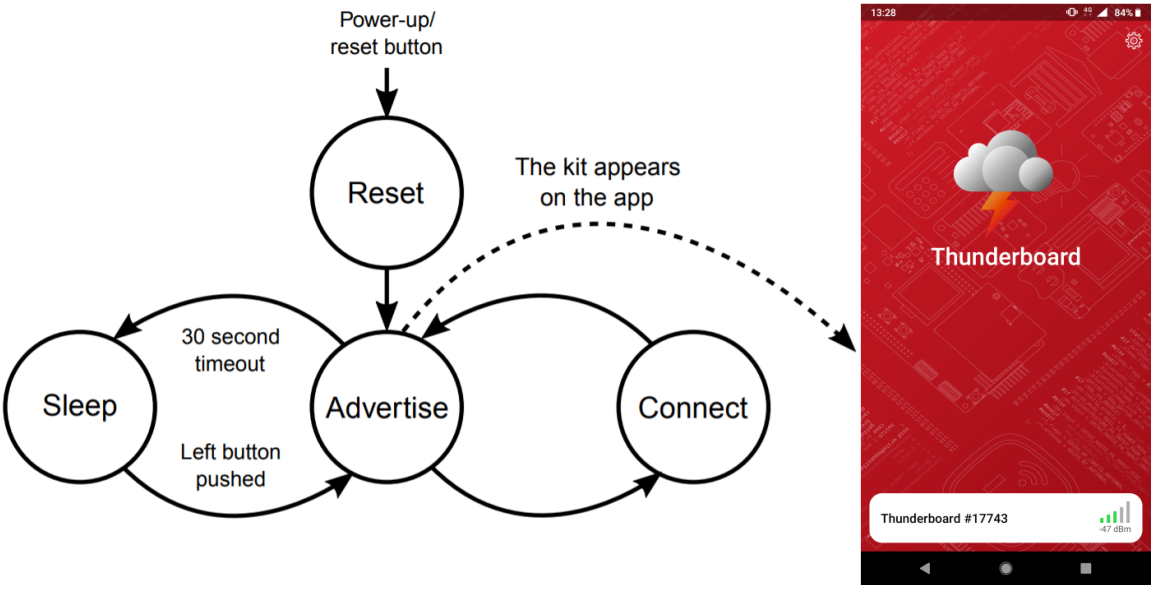
Project Setup

The available sensors are different based on the board you use. For a list of the available features, see the User's Guide for the respective boards:

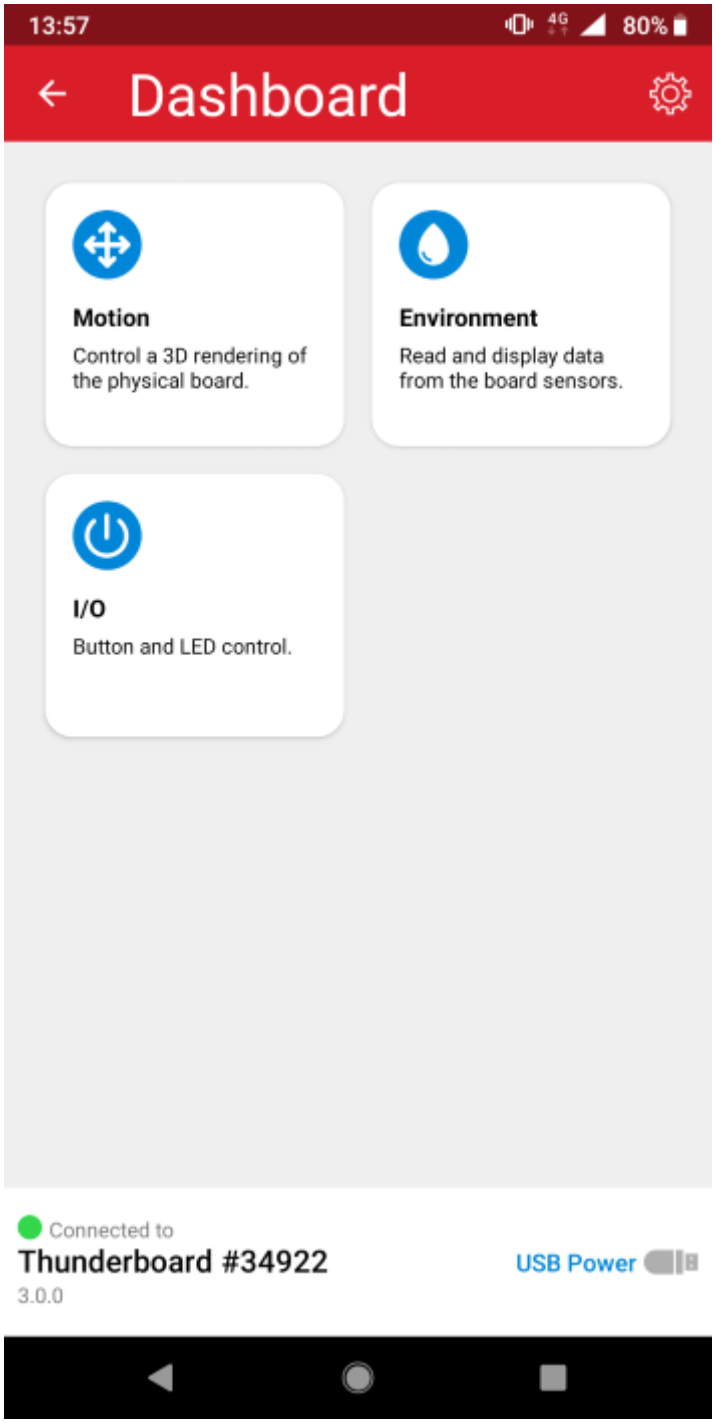
UG415:Thunderboard EFR32BG22 User's guide

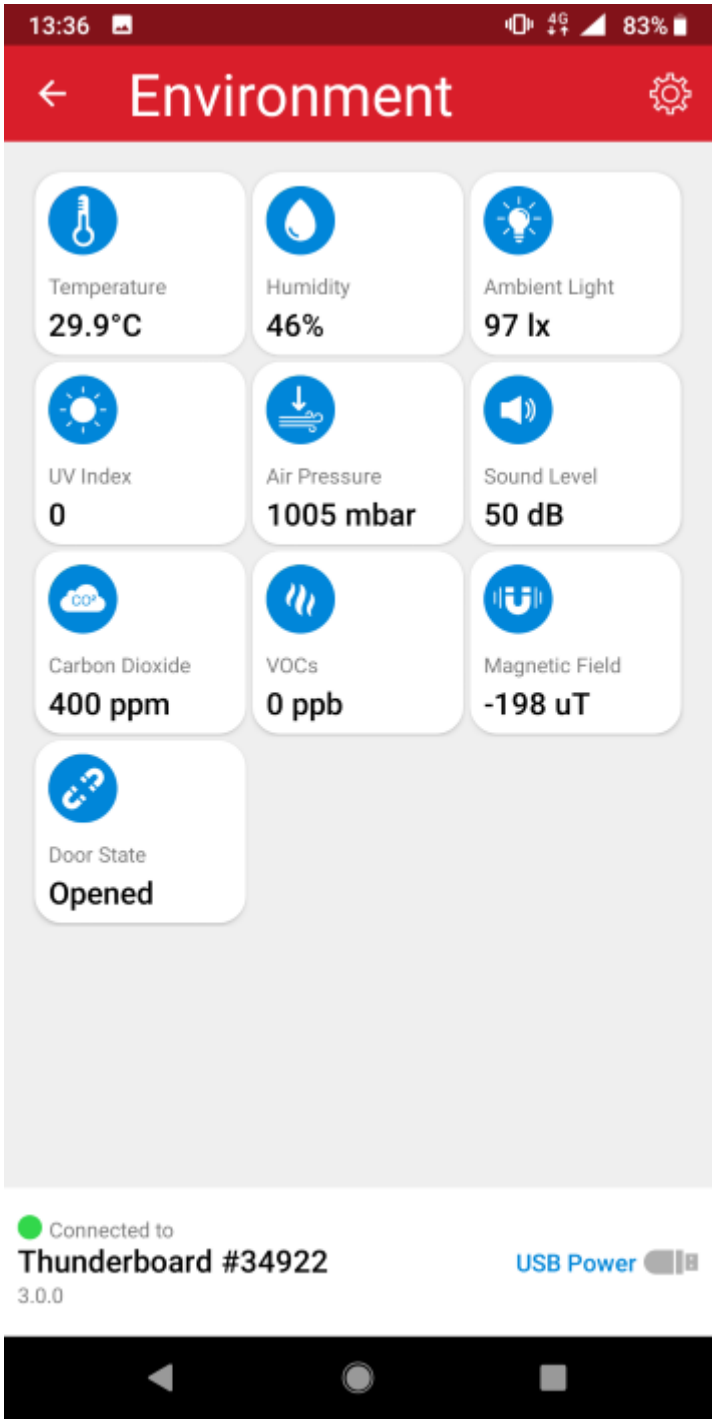
UG309: Thunderboard Sense 2 User's Guide

After flashing the demo, the board will start to advertise, and after a 30 seconds timeout it will go to sleep mode. It will wake up when the left button (BTN0) is pressed. The state diagram of the firmware is shown below.

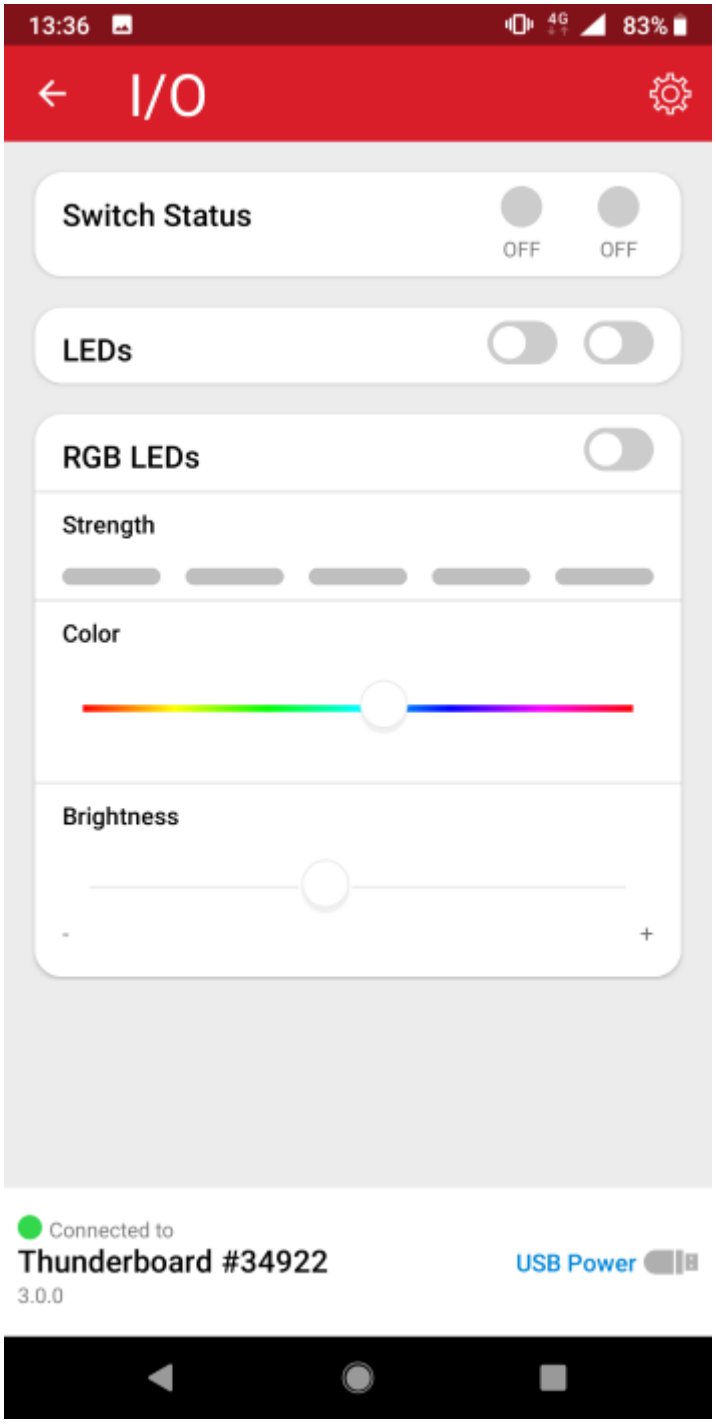


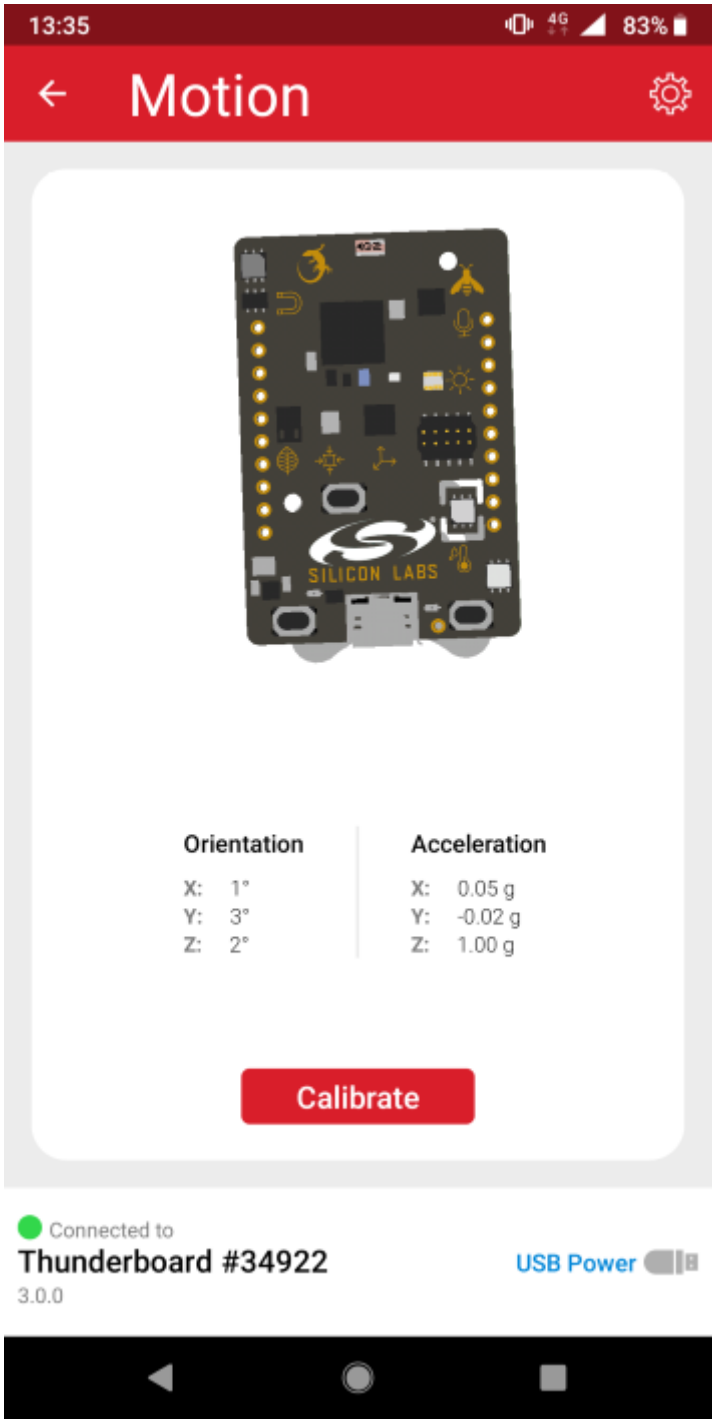
You can connect the device by tapping on the name of it in the Thunderboard app. After the connection is established, you will see the dashboard. There, you can select the categories of the different sensor types and the controls to see.
The screenshots are taken with the Thunderboard Sense 2, for the EFR32BG22 the same categories apply, but with a reduced set of sensors and LEDs.
By selecting the environment data, you can see the values of the different sensors mounted on the board, as shown below:


13:57
80%
Dashboard
Motion
Control a 3D rendering of the physical board.
Environment
Read and display data from the board sensors.
I/O
Button and LED control.
Connected to
Thunderboard #34922
3.0.0
USB Power



Within the I/O you can control the LEDs on the board and see the state of the push buttons. Inside the Motion part, you will see a 3D image of the board. Note that the orientation is changing when you move the board. See the image below.

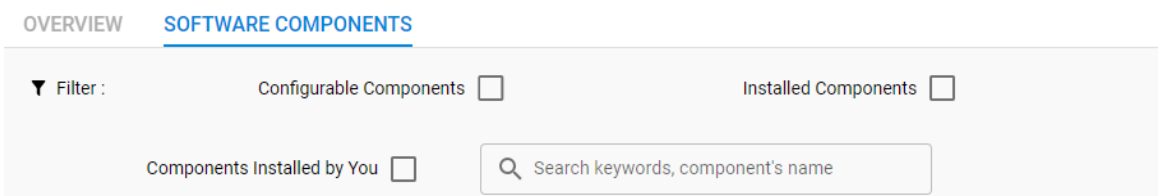

13:36
83%
I/O
Switch Status
OFF
OFF
LEDs
RGB LEDs
Strength
Color
Brightness
-
+
Connected to
Thunderboard #34922
3.0.0
USB Power

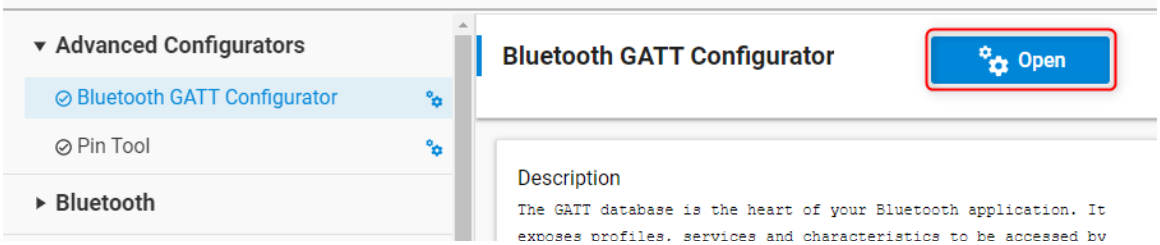


Project Structure

The project code is the same for both boards. The different sensor configurations are set in the automatically generated *sl_component_catalog.h*. The main application file, the *app.c* configures the project accordingly.

The Bluetooth-related event handling is implemented in the function *sl_bt_on_event*.
The projects contain the needed services in the GATT database. GATT definitions can be extended using the GATT Configurator, which can be found under Advanced Configurators in the Software Components tab of the Project Configurator:





To learn how to use the GATT Configurator, see UG438: GATT Configurator User's Guide for Bluetooth® SDK v3.x.

The sensors and I/O are also handled in this file by overriding the default weak implementation of the service handling functions.

Additional functionality can be added to the empty *app_process_action* function.

Troubleshooting

Note that **NO** Bootloader is included in any Software Example projects, but they are configured so, that they expect a bootloader to be present on the device. To get your application work, you should either

- flash a bootloader to the device or
- uninstall the **OTA DFU** and **Bootloader Application Interface** software components.

To flash a bootloader, you should either create a bootloader project or run a precompiled **Demo** on your device from the Launcher view. Precompiled Demos flash both bootloader and application images to your device.

- To flash an OTA DFU capable bootloader to your device, *SoC-Thermometer* demo can be flashed before your application to load the bootloader.
- To flash a UART DFU capable bootloader to your device, *NCP-Empty* demo can be flashed before your application to load the bootloader.
- For your custom application, create your own bootloader project and flash it to your device before flashing your application.
- When you flash your application image to the device, use the *.hex* or *.s37* output file. Flashing *.bin* files may overwrite (erase) the bootloader.
- On Series 1 devices (EFR32xG1x), both first stage and second stage bootloaders have to be flashed. This can be done at once by flashing the **-combined.s37* file found in your bootloader project after building the project.
- For more information, see *UG103: Bootloading fundamentals* and *UG266: Silicon Labs Gecko Bootloader User's Guide*.

Before programming the radio board mounted on the WSTK, make sure the power supply switch the AEM position (right side) as shown below.

Resources

Bluetooth Documentation

UG103.14: Bluetooth® LE Fundamentals

QSG169: Bluetooth® SDK v3.x Quick Start Guide

UG434: Silicon Labs Bluetooth ® C Application Developer's Guide for SDK v3.x

Bluetooth Training

Report Bugs & Get Support

You are always encouraged and welcome to report any issues you found to us via Silicon Labs Community.