

Presentation Will
Begin Shortly

4:00



BLUETOOTH

- FEB 29TH | Small Bluetooth Devices - How to Minimize Size without Compromising Performance and Reliability
- APR 4TH | Bluetooth LE Application Development Journey
- MAY 9TH | Unboxing Silicon Labs' Latest Bluetooth SoC for Energy Harvesting
- JUN 13TH | Explore Bluetooth Channel Sounding

Welcome

Bluetooth LE Application
Development Journey

tech talks



Introduction



Mate Perjesi

- Máté is Director of Software Engineering here at Silicon Labs, leading the Bluetooth SDK development and the Ease-of-Use Technology Vector in the Software R&D organization. He has been with Silicon Labs since 2014, and right from the beginning he has been working on Bluetooth and DX (Developer Experience).



Zoltán Fegyveres

- Zoltán Fegyveres is a Senior Software Engineer at Silicon Labs with 11 years of expertise in embedded software engineering. He has worked on automotive projects, robotics, and now focuses on Bluetooth Low Energy, delivery innovative solutions at the forefront of technological advancement.



Péter Kerekes

- Péter Kerekes joined Silicon Labs after finishing his studies in 2020 and works as a Software Engineer since. He develops Bluetooth Low Energy sample applications and considers his main area to be Certificate-Based Authentication and Pairing (CBAP).

Agenda

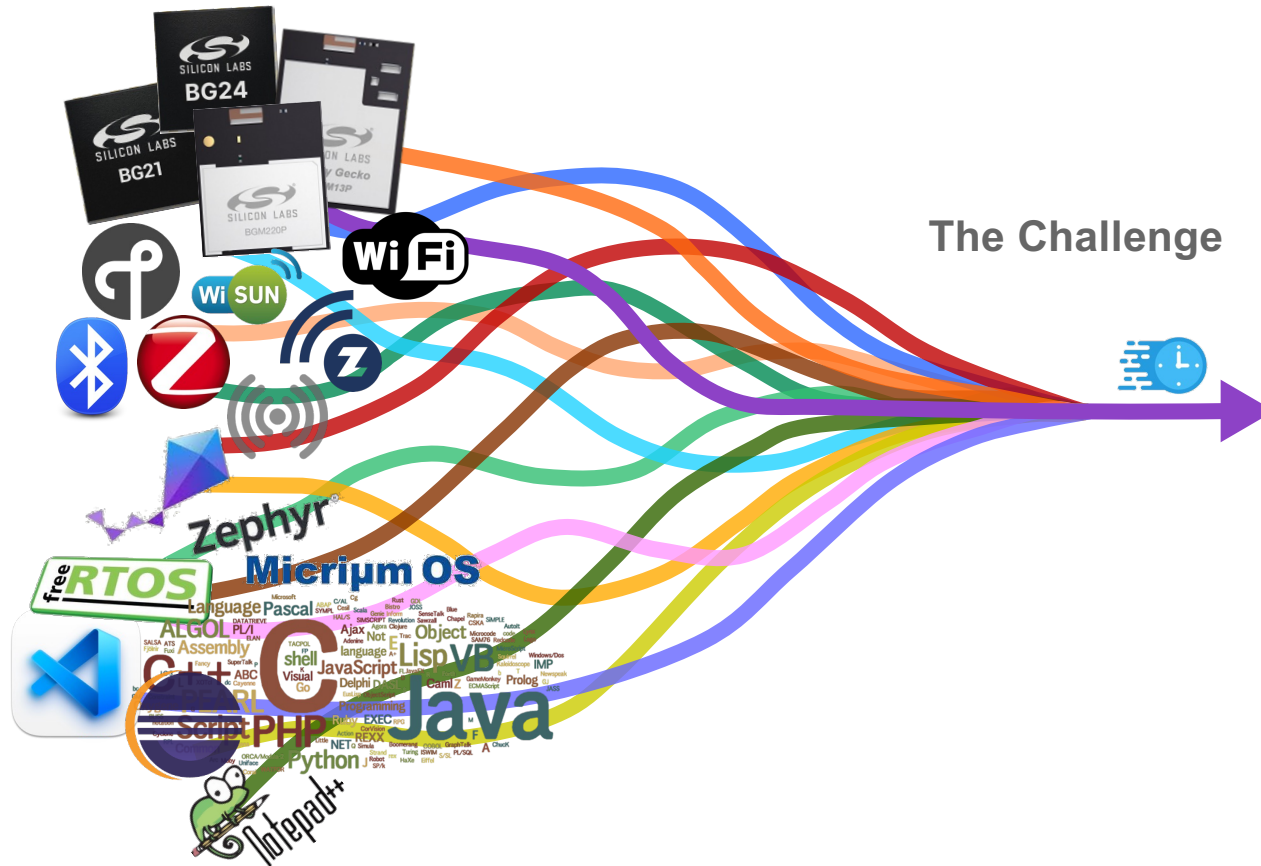
- 01** Welcome & intro - Aashish
- 02** Bluetooth Developer Journey – Máté
- 03** Bluetooth LE App SoC Dev – Zoltán
- 04** Bluetooth LE App pyBGAPI Dev – Péter
- 05** Q&A

Bluetooth Developer Journey with Silicon Labs

Máté Perjési

The Challenge of IoT Product Creation

IoT HW & SW offering



The Challenge

The desired IoT Product

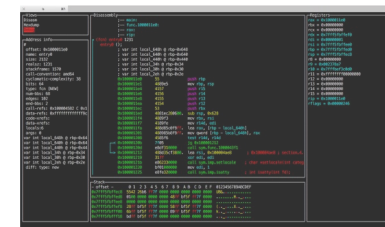
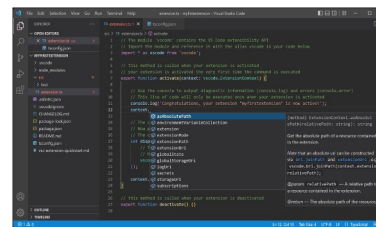
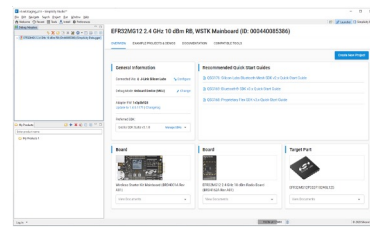
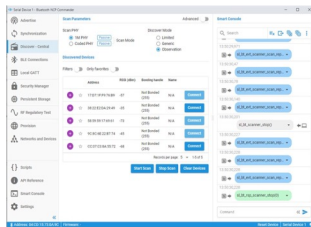


Silicon Labs Code Levels

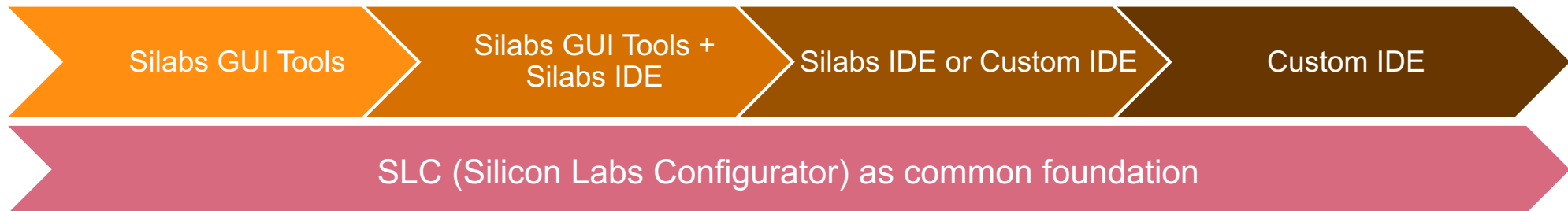
Large variety of customer **expertise levels**:



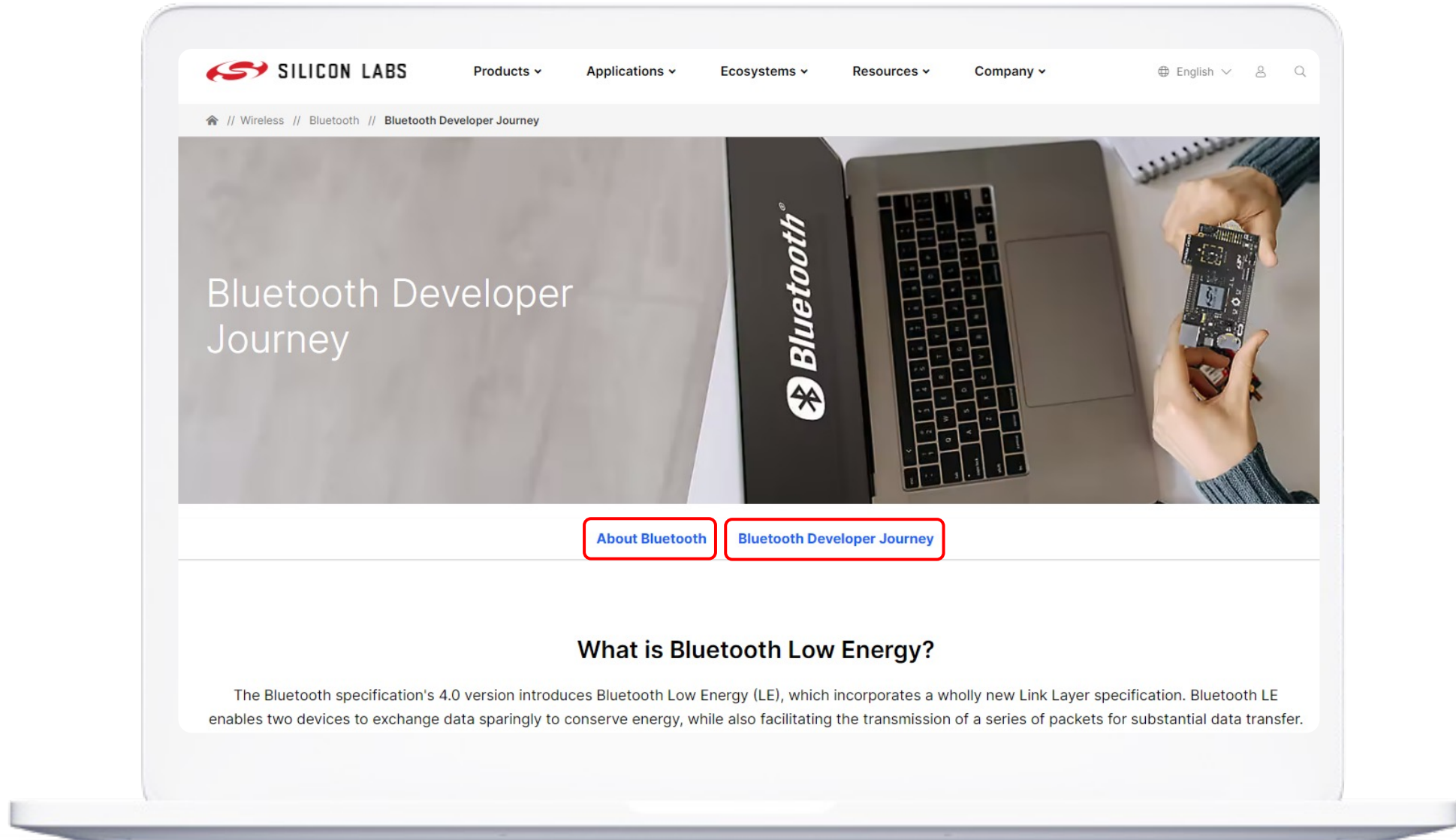
Matching developer flows, the Silicon Labs **Code Levels**:



Typical developer environment and tools:



The new Silicon Labs Bluetooth Developer Journey web page



About Bluetooth – What is Bluetooth Low Energy?

What is Bluetooth Low Energy?

The Bluetooth specification's 4.0 version introduces Bluetooth Low Energy (LE), which incorporates a wholly new Link Layer specification. Bluetooth LE enables two devices to exchange data sparingly to conserve energy, while also facilitating the transmission of a series of packets for substantial data transfer.



Bluetooth Mesh

Even though the Bluetooth specification does not define a network layer, the multi-connection support lets you connect one device with many more, forming a star topology. Bluetooth LE Dual Topology even lets you make an extended star topology. You only have to take care of forwarding data between the nodes. If you need a real Bluetooth LE based network, check out our [Bluetooth Mesh](#) solution.

[Explore Bluetooth Mesh](#)

Bluetooth solution, we have you covered.

[Explore Bluetooth Product Brief](#)

FR and PHY



About Bluetooth – What's New with Bluetooth 5.4

What's New with Bluetooth 5.4

Broadcasting Data (PaWR)

Bluetooth LE advertisements let you send data to an unlimited number of devices. With extended advertisements you can send up to 1650 B at once, and choose to repeat or change the payload at any time. Periodic Advertisements eliminate the need for continuous scanning and Periodic Advertisements with Responses (PAWR) make it possible to transfer data to thousands of devices with acknowledgements.

[Learn About PaWR](#)



Electronic Shelf Labels

Bluetooth 5.4 defines all features that are needed for extra low power data exchange between Access Points and thousands of ESL tags, and it aims to be the de-facto standard for ESLs.

[Explore Electronic Shelf Labels](#)



Bluetooth Developer Journey

Bluetooth Developer Journey with Silicon Labs

Silicon Labs can accelerate the development of Bluetooth devices, starting by outlining each step in the process and helping you along each stage of your project. We are here to simplify your development journey and help you get your devices to market faster and more efficiently.

We have outlined below three key stages of the Bluetooth Developer Journey, along with what is required to successfully complete each stage.

Getting Started

Develop

Deploy Product

1. Buy Kit: Examples &
Hardware

2. EFR Connect Mobile App

3. Create User Accounts

4. Set Up Development
Environment

5. Explore Demos

1. Buy Kit: Hardware and Examples


Silicon Labs offers several Bluetooth development kits ranging from ultra-low-cost, small form factor prototyping platforms to compact, feature-packed development platform kits for robust networks. Based on the demos you would like to explore, select which kit is the best fit for your needs below. For enabling in-depth evaluation and future development, Silicon Labs recommends choosing at least 2 development kits (2 BLE nodes).



Bluetooth Developer Journey – Develop and Deploy

Getting StartedDevelopDeploy Product

1. Qualify Product2. Manufacture Product



2. Manufacture Product

Bluetooth DAC injection is required for end products. Silicon Labs can help simplify the DAC injection process for Bluetooth certification with our [Custom Part Manufacturing Service \(CPMS\)](#), keeping your private keys private – from the factory to the end user's homes.

[Bluetooth RF-PHY Evaluation >](#)




[Bluetooth Qualification \(Step-by-Step Guide\)* >](#)

[Bluetooth SIG Qualifications and Listings >](#)

**Login required*



Getting Started – 1. Buy Kit

			
1. Buy			
1. Buy			
Silicon Labs development enabling			
Kit	BGM220 Explorer Kit	EFR32BG22 Thunderboard Kit	EFR32xG24 Dev Kit
OPN	(BGM220-EK4314A)	(SLTB010A)	(xG24-DK2601B)
Description	The BGM220 Explorer Kit is an ultra-low-cost, small form factor development and evaluation platform for the BGM220P Bluetooth Module.	Thunderboard BG22 is a small form-factor, optimized development platform for adding Bluetooth connectivity to battery-powered IoT products.	The EFR32xG24 Dev Kit is a compact, feature-packed development platform. It provides the fastest path to develop and prototype wireless IoT products.
Price	\$10 USD	\$45 USD	\$79 USD
Flash/RAM	512 kB / 32 kB	512 kB / 32 kB	1536 kB / 256 kB
AI/ML			✓
Microphone			✓
	EFR Connect Mobile App Demo Support/Coverage		
RGB LED			✓
Health Thermometer	✓*	✓	✓
Kit	Blinky	✓	✓
OPN	Environment	✓	✓



Getting Started – 2. EFR Connect Mobile App

1. Buy Kit: Examples & Hardware

2. EFR Connect B

Test and debug Bluetooth e and iOS interoperability. Dow

GET IT ON Google Play

BLE Devices

SCANNER GRAPH (0) ACTIVE CONNECTIONS

Blinky Example CONNECT

-40 dBm 103 ms Unspecifi... Connecti...

★ Not bonded

N/A CONNECT

-45 dBm 27 ms Unspecifi... Connecti...

★ Not bonded

Thermometer Exa... CONNECT

-51 dBm 104 ms

Stop Scanning

Demo

Health Thermomet... View readings from the Health Thermometer service

Connected Lighting Control a Dynamic Multiprotocol application of connected lights and ...

Range Test Evaluate the link budget and range of EFR32

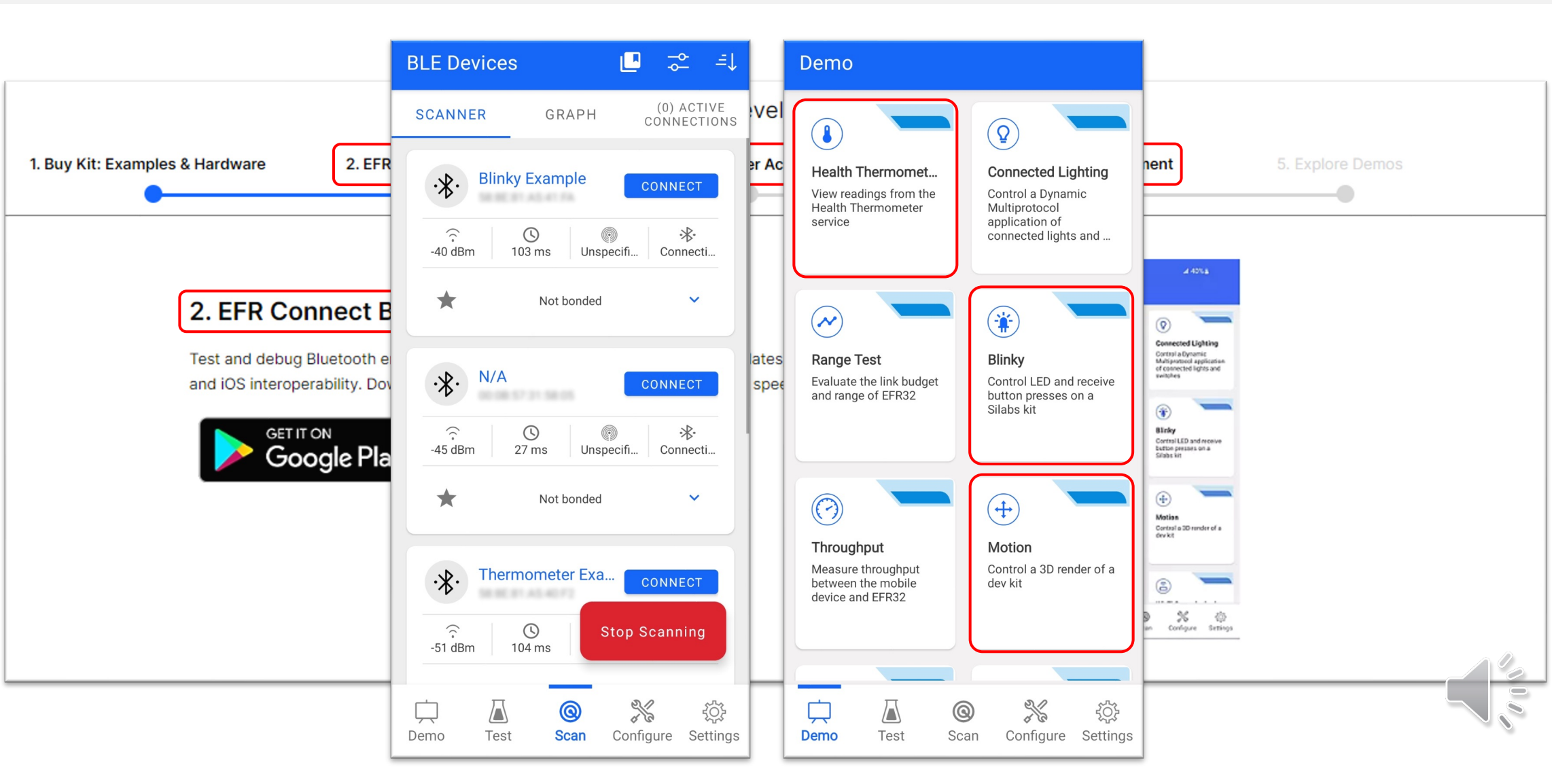
Blinky Control LED and receive button presses on a Silabs kit

Throughput Measure throughput between the mobile device and EFR32

Motion Control a 3D render of a dev kit

Demo Test Scan Configure Settings

5. Explore Demos



The image displays two overlapping screenshots of the EFR Connect Mobile App. The left screenshot shows the 'BLE Devices' scanner interface with three device cards: 'Blinky Example', 'N/A', and 'Thermometer Exa...'. Each card displays signal strength (dBm), latency (ms), and a 'CONNECT' button. A red box highlights the '2. EFR Connect B' step in the background. The right screenshot shows the 'Demo' screen with a grid of demo cards: 'Health Thermomet...', 'Connected Lighting', 'Range Test', 'Blinky', 'Throughput', and 'Motion'. Red boxes highlight the 'Health Thermomet...' and 'Blinky' cards. A navigation bar at the bottom of both screens includes 'Demo', 'Test', 'Scan', 'Configure', and 'Settings' icons. A speaker icon is visible in the bottom right corner of the overall image.

Getting Started – 4. Set Up Development Environment

The screenshot displays the Silicon Labs development environment interface, divided into three main sections:

- Bluetooth Services Panel (Left):** A vertical menu with various Bluetooth-related options including Advertise, Synchronization, Discover - Central, BLE Connections (highlighted with a red '1'), Local GATT, Security Manager, Persistent Storage, RF Regulatory Test, Provision, Networks & Nodes, Models, Scripts, API Reference, Smart Console, and Settings.
- Remote GATT Database (Center):** A list of services and characteristics for a connected device. The services shown are:
 - Generic Attribute (Primary service):** UUID: 1801, Handle: 65544. Includes a Characteristics icon.
 - Service Changed:** UUID: 2A05, Handle: 3. Includes a Descriptors icon. Properties: 32. Value field is empty.
 - Client Characteristic Configuration:** UUID: 2902, Handle: 4. Value field contains '0000'.
 - Database Hash:** UUID: 2B2A, Handle: 6. Includes a Descriptors icon. Properties: 2. Value field contains '624B8830997E7DEC697235682351A31B'.
 - Client Supported Features:** UUID: 2B29, Handle: 8. Includes a Descriptors icon. Properties: 10. Value field contains '00'.
 - Generic Access (Primary service):** UUID: 1800, Handle: 589837. Includes a Characteristics icon.
 - Device Name:** UUID: 2A00, Handle: 11. Includes a Descriptors icon. Properties: 10. Value field contains 'Thermometer Example'.
- Smart Console (Right):** A terminal window showing a search for 'sl_bt_evt_scanner_legacy_advertisement...' and a list of received events with timestamps (e.g., 17:0:1,509, 17:0:1,540, etc.). A 'byte' label is visible on the right side of the console. A 'Command' input field is at the bottom.

Getting Started – 5. Explore Demos

Getting Started

Develop

Deploy Product



NCP CLIENT, THUNDERBOARD SERVER

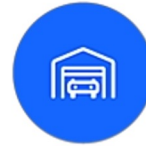
Outside Environment Measurement Tool with Logging Capability to a Web-server on the PC

The [Thermometer example](#) will provide all temperature readings, while the NCP will be connected to the PC host, and [PyBGAPI Thermometer Client example](#) will log all measurements on it.

Suggested Kits:

- [BG22 Thunderboard Kit](#) or [xG24 Dev Kit](#) to use as Thermometer server*
- [BGM220 Bluetooth Module Explorer Kit](#) to use as NCP client connected to the PC - use [PyBGAPI Thermometer Client example](#).

(*any board that has thermometer sensor)



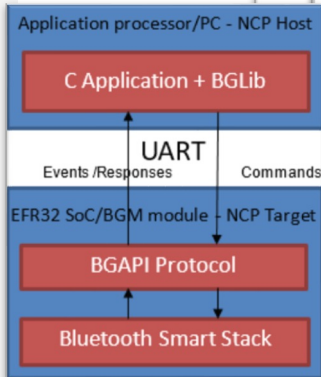
NCP CLIENT, BLINKY SERVER

Garage Door Opener

Use the [Blinky example](#) to drive the garage door opener (motor to be connected through relay), remotely using the EFR Connect Mobile app, while logging the access to the garage on a PC as NCP host (through e.g. [PyBGAPI](#)). Drive the same server/actuator through two means: EFR mobile and PC.

Suggested Kits:

- 2 x [BGM220 Bluetooth Module Explorer Kit](#) to use as Blinky server and NCP client.



- [BGM220 Bluetooth Module Explorer Kit](#) to use as Thermometer client

server* + EFR mobile app as Client Environment or Motion tile



Getting Started – 5. Explore Demos



NCP CLIENT, BLINKY AND THERMOMETER SERVER

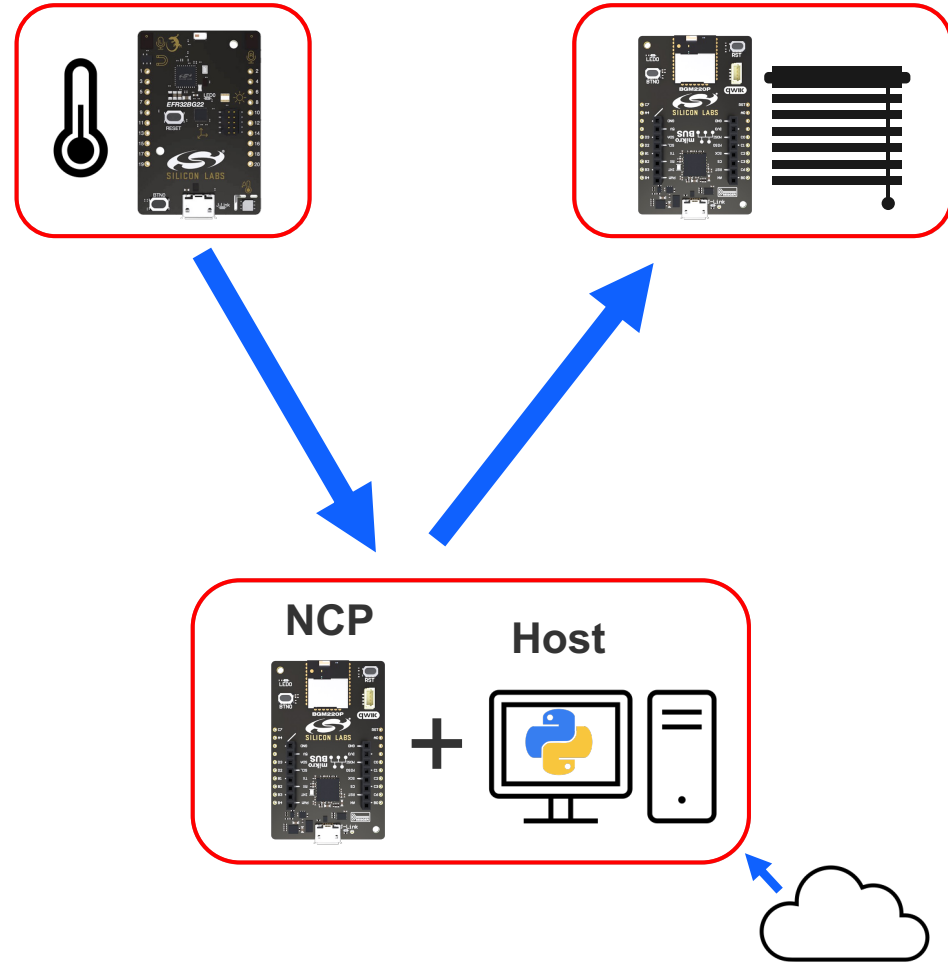
Shade Adjuster Based on Outside Temperature and Weather Forecast

The [Thermometer example](#) will measure outside temperature (if Thunderboard is used, the ambient light can be combined to this using the [Thunderboard example](#)) and send data to NCP. NCP host (through e.g. [PyBGAPI](#)) will command the SOC [Blinky example](#) using pre-defined algorithm that gets data from the internet to adjust the shades on the house. The Blinky will be connected to the motor of the shades. The best tool to evaluate this use-case might be to use [PyBGAPI tool](#).

Suggested Kits:

- [BG22 Thunderboard Kit](#) or [xG24 Dev Kit](#) to use as Thunderboard (or Thermometer) sensor*
- [BGM220 Bluetooth Module Explorer Kit](#) to use as NCP client
- Additional [BGM220 Bluetooth Module Explorer Kit](#) to use as Blinky server if needed.

(*any board that has thermometer sensor)

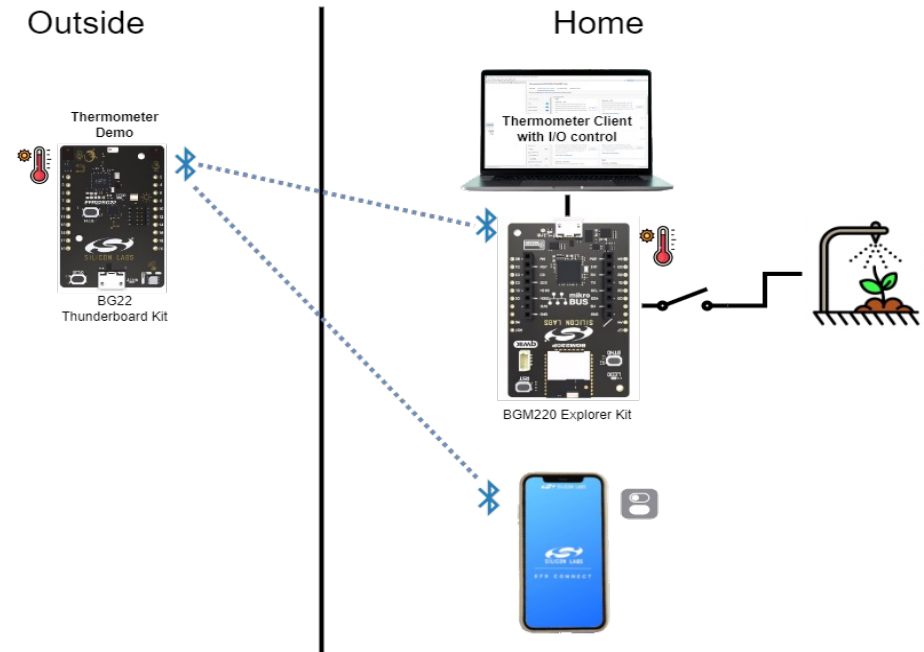


Bluetooth LE SoC Application Development

Zoltán Fegyveres


Bluetooth LE SoC Application Development

- Realize the **water irrigation system**:
 - Irrigate automatically - only below certain temperature
 - Stop automatic irrigation after predefined time (5 seconds)
 - Irrigate manually – from your smartphone
- 1 SoC **prebuilt demo** on the Thunderboard Kit
 - SoC Thermometer:
 - measuring & sending temperature over BLE
- 2 SoC **examples merged** on the BGM220 Explorer Kit
 - SoC Blinky:
 - controlling GPIO
 - connectable with a smartphone
 - data exchange between smartphone or other BLE device
 - SoC Thermometer Client:
 - Auto connect to SoC Thermometer – collect temperature data from BLE
 - **Optionally: additional hardware parts - relay board, aquarium pump, 12V PSU (for the pump)**



Bluetooth LE SoC Application Development – No code

- **BG22 Thunderboard Kit**

- Upload the **SoC Thermometer** demo to the Thunderboard
 - +  Put a coin cell battery into the kit – take it wherever you want

- **BGM220 Explorer Kit**

- Create **SoC Blinky** & **SoC Thermometer Client** examples for the board
- Build & Run the SoC Thermometer Client on the board
- Get necessary components:
 - **BLE**
 - GATT Server
 - Device Information GATT Service
 - Legacy Advertising
 - Static GATT database & configuration (installed with the service)
 - **Peripherals**
 - Simple LED
 - Simple Button
 - Timer
- Get relevant parts of GATT from SoC Blinky

Debug Adapters

BGM220 Explorer Kit Board (ID:440197566)

My Products

Enter product name

BGM220 Explorer Kit Board (ID: 000440197566)

- OVERVIEW
- EXAMPLE PROJECTS & DEMOS
- DOCUMENTATION
- COMPATIBLE TOOLS

Create New Project

General Information

Connected Via: **J-Link Silicon Labs** [Configure](#)

Debug Mode: **Onboard Device (MCU)** [Change](#)

Adapter FW: **1v5p0b37** **Latest**

Secure FW: **1.2.14** [1.2.14 | Changelog](#)

Preferred SDK:
Gecko SDK Suite v4.4.1 [Manage SDKs](#)

Recommended Quick Start Guides

[AN1255: Transitioning from the v2.x to the v3.x Bluetooth SDK](#)

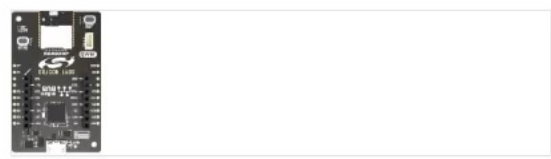
[QSG175: Silicon Labs' Direction Finding Solution Quick-Start Guide](#)

[QSG183: Bluetooth Mesh SDK Quick-Start Guide for SDK v4.x](#)

[QSG169: Bluetooth® Quick-Start Guide for SDK v3.x and Higher](#)

All Quick Start Guides

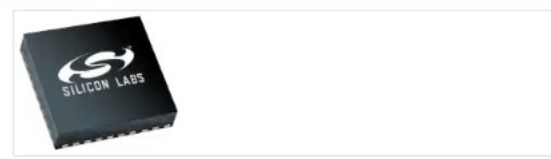
Board



BGM220 Explorer Kit Board (BRD4314A)

[View Documents](#)

Target Part



BGM220PC22HNA

[View Documents](#)

Bluetooth LE SoC Application Development – Low code

▪ Explorer Kit

○ Get code parts from Blinky to Thermometer Client:

▪ BLE features:

- Legacy Advertising
- Connection open / close
- GATT attribute change – LED indication, Button notification (used e.g. in EFR Connect)

▪ Peripherals:

- Button handling
- Onboard LED handling

○ Build & Flash what you have so far!

○ Optional: hook up the relay to the Pin of the LED (**GPIO PA4** – on the BGM220)

▪ Congratulation! 🌱

○ You can control your relay board with the EFR Connect app, if you connect to your Explorer Kit via Bluetooth LE Select Demo / Blinky in EFR Connect 📱

○ You can check the temperature data if you open a serial connection to your board on your PC 🖥️

v5_workspace - bt_soc_thermometer_client_bgm220/bt_soc_thermometer_client_bgm220.sclp - Simplicity Studio™

File Edit Navigate Search Project Run Window Help

gatt_configuration.btconf bt_soc_thermometer_client_bgm220.sclp

Project Explorer

- sl_gatt_service_device_information.c
- bt_soc_blinky_bgm220.pintool
- bt_soc_blinky_bgm220.sclp
- bt_soc_blinky_bgm220.slps
- create_bl_files.bat
- create_bl_files.py
- create_bl_files.sh
- readme.md
- bt_soc_thermometer_client_bgm220 [GNU ARM]
 - Binaries
 - Includes
 - autogen
 - config
 - gecko_sdk_4.4.1
 - GNU ARM v12.2.1 - Default
 - image
 - app.c
 - app.h
 - main.c
 - sl_gatt_service_device_information.c
 - bt_soc_thermometer_client_bgm220.pintool
 - bt_soc_thermometer_client_bgm220.sclp
 - bt_soc_thermometer_client_bgm220.slps
 - readme.md


Debug Adapters

- BGM220 Explorer Kit Board (ID:440197566)

bt_soc_thermometer_client_bgm220

OVERVIEW SOFTWARE COMPONENTS CONFIGURATION TOOLS

Target and Tool Settings



BGM220PC22HNA
BGM220 Explorer Kit Board (BRD4314A)

Selected SDK
Gecko SDK Suite v4.4.1: Amazon 202012.00, Bluetooth 7.0.1, Bluetooth Mesh 6.0.1, EmberZNet 7.4.1.0, Flex 3.7.1.0, MCU 6.6.1.0, Micrium OS Kernel 5.14.00, OpenThread 2.4.1.0 (GitHub-7074a43e4), Platform 4.4.1.0, Sidewalk 2.0.1, Silicon Labs Matter 2.2.0-1.2, USB 1.2.0.0, Wi-SUN 1.9.0.0, Z-Wave SDK 7.21.1.0

Project Details

bt_soc_thermometer_client_bgm220

Implements a GATT Client that discovers and connects with up to 4 BLE devices advertising themselves as Thermometer Servers. It displays the discovery process and the temperature values received via UART.

Category
Bluetooth Examples

Preferred SDK
Gecko SDK Suite v4.4.1: Amazon 202012.00, Bluetooth 7.0.1, Bluetooth Mesh 6.0.1, EmberZNet 7.4.1.0, Flex 3.7.1.0, MCU 6.6.1.0, Micrium OS Kernel 5.14.00, OpenThread 2.4.1.0 (GitHub-7074a43e4), Platform 4.4.1.0, Sidewalk 2.0.1, Silicon Labs Matter 2.2.0-1.2, USB 1.2.0.0, Wi-SUN 1.9.0.0, Z-Wave SDK 7.21.1.0

Import Mode
Copy contents

Quick Links

- Software Components
- app.c
- main.c
- readme.md
- Pin Tool
- Memory Editor

Problems Search Call Hierarchy Console

```
CDT Build Console [bt_soc_thermometer_client_bgm220]
.debug_ranges      18376      0
.debug_line        397590     0
.debug_str         201801     0
.debug_frame       94376      0
.debug_loclists    108090     0
.debug_rnglists    28069      0
Total              2551304
```

08:04:37 Build Finished. 0 errors, 0 warnings. (took 3m:25s.687ms)

©2024 Silicon Laboratories Inc. All rights reserved.

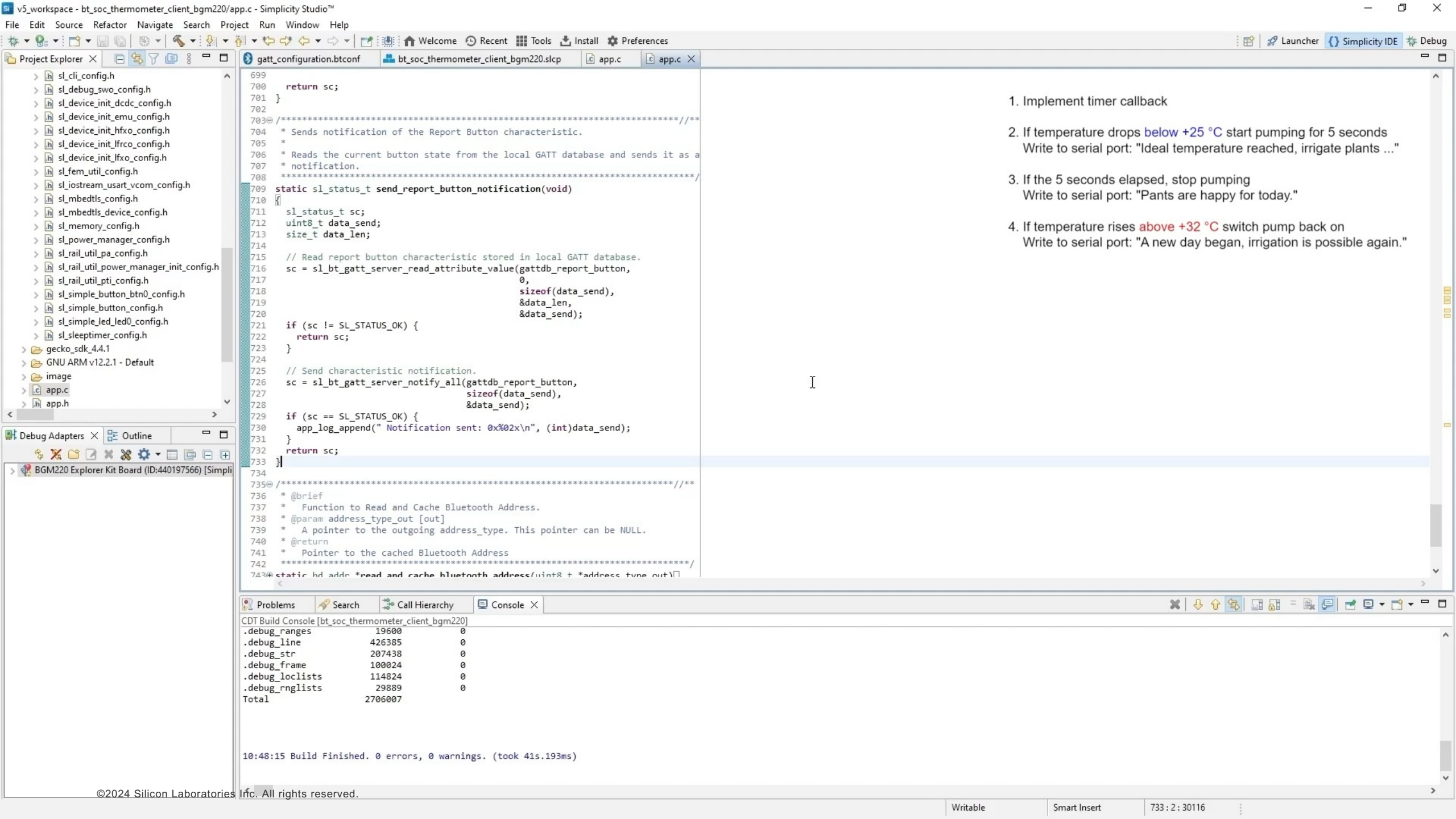
Bluetooth LE SoC Application Development – Custom code

▪ Explorer Kit

- Add the application timer component (Timer)
- Implement the timer callback function

- Check where the example writes its status to the serial output – over there you can realize this simple irrigation logic:
 1. If temperature drops **below +25 degrees (Celsius)** start pumping for 5 seconds 💧
 2. If the timer elapsed stop pumping
 3. If temperature rise **above +32 degrees (Celsius)** switch the pump access back on

- Congratulation your app **evolved!** 🌱
 - Your plant is automatically irrigated on the mornings
 - You can also start/stop irrigation manually with your smartphone in EFR Connect 📱
 - Temperature data and irrigation events can still be checked with a serial connection to the PC 🖥️



```
699
700     return sc;
701 }
702
703 /**
704  * Sends notification of the Report Button characteristic.
705  *
706  * Reads the current button state from the local GATT database and sends it as a
707  * notification.
708  */
709 static sl_status_t send_report_button_notification(void)
710 {
711     sl_status_t sc;
712     uint8_t data_send;
713     size_t data_len;
714
715     // Read report button characteristic stored in local GATT database.
716     sc = sl_bt_gatt_server_read_attribute_value(gattdb_report_button,
717                                               0,
718                                               sizeof(data_send),
719                                               &data_len,
720                                               &data_send);
721
722     if (sc != SL_STATUS_OK) {
723         return sc;
724     }
725
726     // Send characteristic notification.
727     sc = sl_bt_gatt_server_notify_all(gattdb_report_button,
728                                     sizeof(data_send),
729                                     &data_send);
730
731     if (sc == SL_STATUS_OK) {
732         app_log_append(" Notification sent: 0x%02x\n", (int)data_send);
733     }
734     return sc;
735 }
736
737 /**
738  * @brief
739  * Function to Read and Cache Bluetooth Address.
740  * @param address_type_out [out]
741  * A pointer to the outgoing address_type. This pointer can be NULL.
742  * @return
743  * Pointer to the cached Bluetooth Address
744  */
745 static hd_addr_t *read_and_cache_bluetooth_address(uint8_t *address_type_out)
```

1. Implement timer callback
2. If temperature drops **below +25 °C** start pumping for 5 seconds
Write to serial port: "Ideal temperature reached, irrigate plants ..."
3. If the 5 seconds elapsed, stop pumping
Write to serial port: "Pants are happy for today."
4. If temperature rises **above +32 °C** switch pump back on
Write to serial port: "A new day began, irrigation is possible again."

Problems Search Call Hierarchy Console X

CDT Build Console [bt_soc_thermometer_client_bgm220]

.debug_ranges	19600	0
.debug_line	426385	0
.debug_str	207438	0
.debug_frame	100024	0
.debug_loclists	114824	0
.debug_rnglists	29889	0
Total	2706007	

10:48:15 Build Finished. 0 errors, 0 warnings. (took 41s.193ms)

Bluetooth LE Application Development in Python

Péter Kerekes



NCP CLIENT, THUNDERBOARD SERVER

Outside Environment Measurement Tool with Logging Capability to a Web-server on the PC

The [Thermometer example](#) will provide all temperature readings, while the NCP will be connected to the PC host, and [PyBGAPI Thermometer Client example](#) will log all measurements on it.

Suggested Kits:

- [BG22 Thunderboard Kit](#) or [xG24 Dev Kit](#) to use as Thermometer server*
- [BGM220 Bluetooth Module Explorer Kit](#) to use as NCP client connected to the PC - use [PyBGAPI Thermometer Client example](#).

*(*any board that has thermometer sensor)*

Q&A



Thank you



BLUETOOTH

- FEB 29TH | Small Bluetooth Devices - How to Minimize Size without Compromising Performance and Reliability
- APR 4TH | Bluetooth LE Application Development Journey
- MAY 9TH | Unboxing Silicon Labs' Latest Bluetooth SoC for Energy Harvesting
- JUN 13TH | Explore Bluetooth Channel Sounding

tech **talks**



BLUETOOTH