

LAB 6: Bluetooth Low Energy - MESH (BLE-MESH)

1. MỤC TIÊU

- Xây dựng hệ thống BLE-Mesh.
- Sử dụng API của Bluetooth của Silicon Labs.

Thiết bị thực hành: BRD4180B Radio Board (EFR32xG21 2.4 GHz 20 dBm) + Wireless Starter Kit Mainboard (BRD4001A) [1].

Phần mềm: Simplicity Studio 5 + Simplicity SDK Suite v.2024.6.0.

Link mã nguồn:

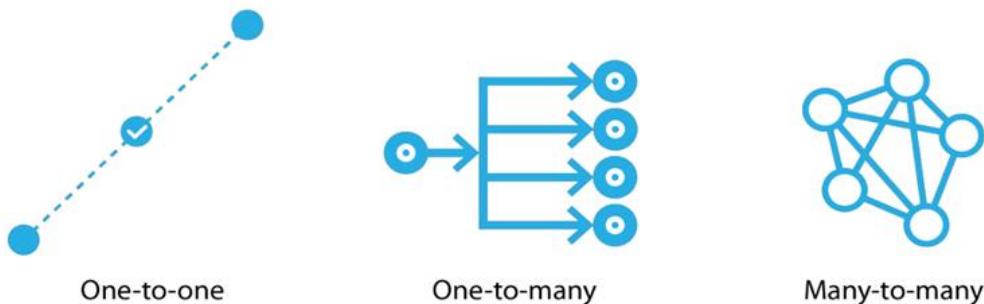
https://drive.google.com/drive/folders/16z2u0c1a6yowdjrDS31W431tDeEg76dK?usp=drive_link

2. NỘI DUNG

2.1. Bluetooth Low Energy – Mesh (BLE-MESH)

Bluetooth Low Energy (BLE), là một công nghệ mạng không dây cá nhân được thiết kế và tiếp thị bởi Bluetooth Special Interest Group (SIG). BLE được tạo ra để thay thế cho các ứng dụng tiêu tốn nhiều năng lượng của Bluetooth truyền thống với mục tiêu cung cấp mức tiêu thụ năng lượng thấp hơn đáng kể trong khi vẫn duy trì phạm vi truyền thông tương đương.

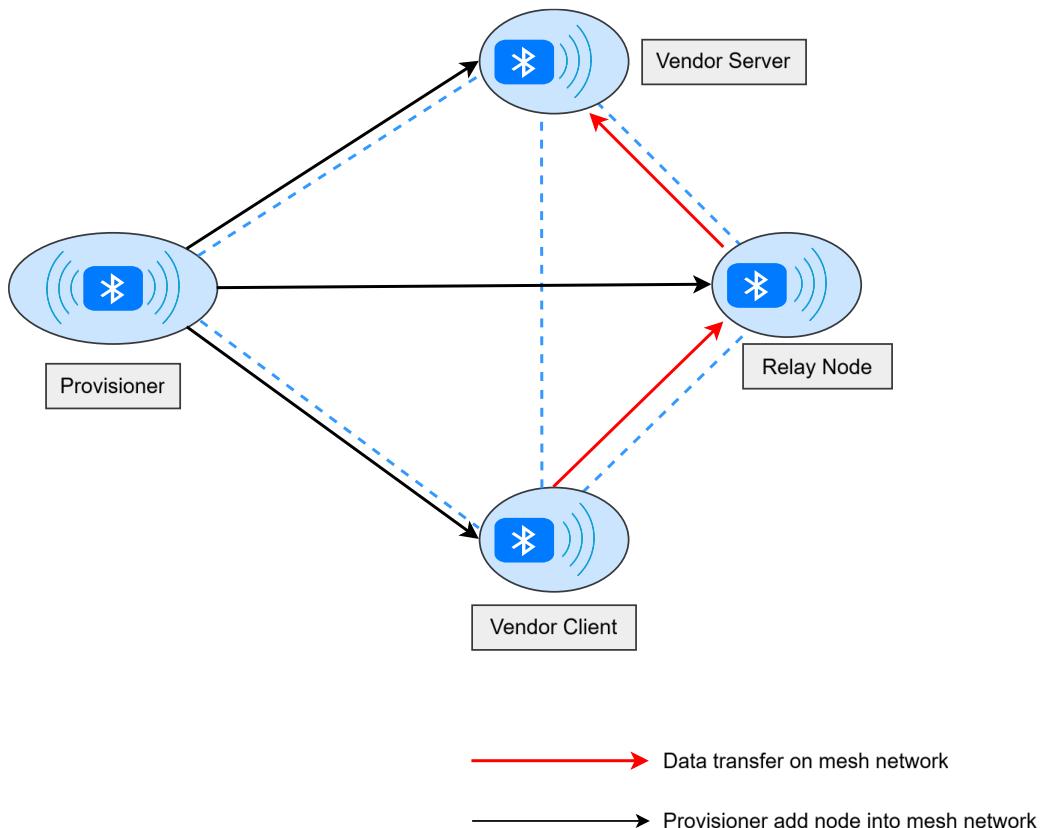
Bluetooth Mesh là một tiêu chuẩn mạng lưới máy tính dựa trên Bluetooth Low Energy cho phép giao tiếp “many to many” qua Bluetooth.



Hình 1 Các kiểu cấu trúc liên kết trong BLE.

Trong một hệ thống Mesh sẽ có 2 phần chính:

- Thiết bị cấp phát (provisioner) có chức năng cấp phát cấu hình cho các thiết bị và thêm các thiết bị chưa được cấp phát vào hệ thống Mesh
- Thiết bị BLE thông thường, khi chưa được cấp phát các thiết bị sẽ thực hiện quảng bá dữ liệu của chính mình ra xung quang (unprovisioner), sau khi được cấp phát và thêm vào hệ thống Mesh thì tùy vào thiết lập của provisioner mà các thiết bị (Node trong Mesh) sẽ có các chức năng khác nhau như “Relay Node, Proxy Node, Friend Node và Low Power Node (LPN), ...”



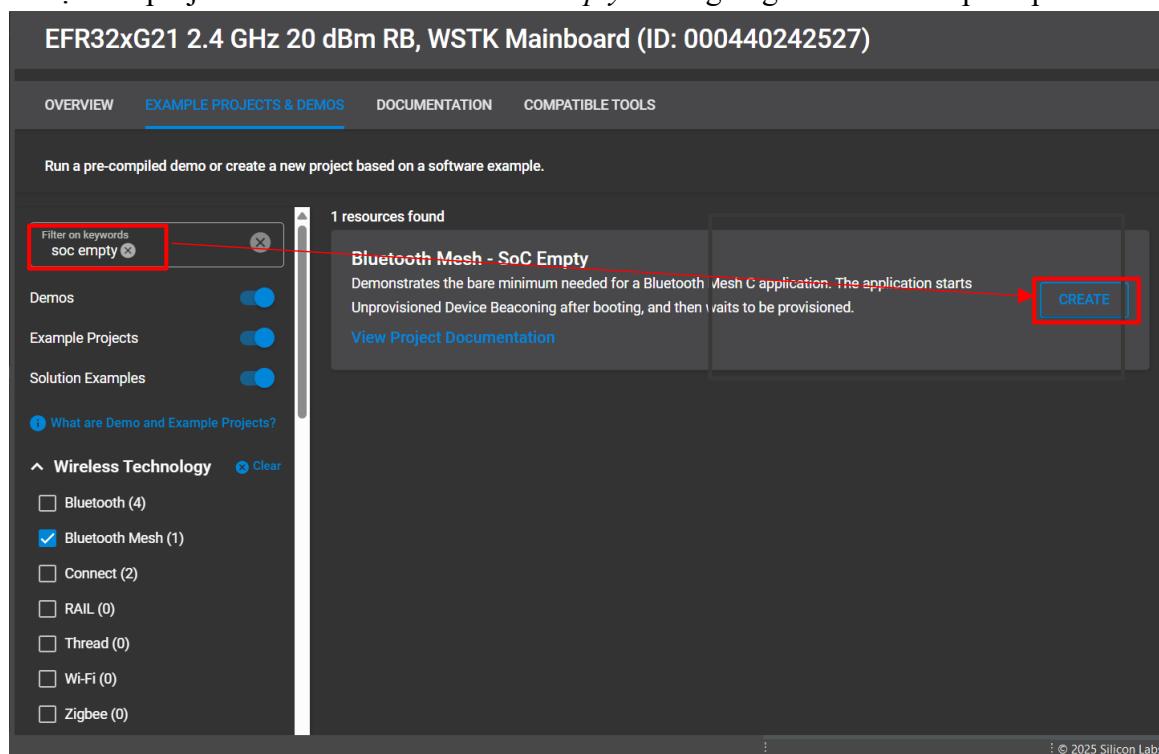
Hình 2 Mô hình BLE-Mesh cơ bản.

2.3. Chương trình demo

1. Mở công cụ Simplicity Studio 5.
2. Chọn thiết bị thực hành: “*EFR32xG21 2.4 GHz 20 dBm Radio Board (BRD4180B)*”.
3. Chọn “*Simplicity SDK Suite v2024.6.0*”.

Với bài lab này ta thực hiện tạo **3 project** với **3 thiết bị chính (Server, Client, Provisioner) cần nạp**

4. Tạo các project “*Bluetooth Mesh - SoC Empty*” tương ứng với các thành phần phía dưới.



4.1 Tạo project “*Bluetooth Mesh - SoC Empty*” Project với tên “*btmesh_soc_vendor_server*”

- Cài đặt **Log** (*Phụ lục, mục 1*).
- Cài đặt Vendor Model , Test, Button Press (*Phụ lục, mục 2,3,4*)
- Cài đặt driver WSTK (*Phụ lục, mục 6*)
- Config model và thêm model vào Main (*Phụ lục, mục 5*), đổi với server thiết lập Model ID là 0x1111
- Sau đó thay thế file các file trong project hiện tại bằng các file trong link bên dưới. (Chọn Copy files -> Overwrite all):

https://drive.google.com/drive/folders/1eis1PXxZrXTu0Bclj86el6EDr4BwLcvw?usp=drive_link

4.2 Tạo project “Bluetooth Mesh - SoC Empty” Project với tên “btmesh_soc_vendor_client”

- Cài đặt **Log** (*Phụ lục, mục 1*).
- Cài đặt Vendor Model , Test, Button Press (*Phụ lục, mục 2,3,4*)
- Cài đặt driver WSTK (*Phụ lục, mục 6*)
- Cài đặt sensor (*Phụ lục, mục 7*)
- Config model và thêm model vào Main (*Phụ lục, mục 5*) đối với client thiết lập Model ID là 0x2222
- Sau đó thay thế file các file trong project hiện tại bằng các file trong link bên dưới. (Chọn Copy files -> Overwrite all):

https://drive.google.com/drive/folders/1XvfYqICDZbplXCUB3WeucDcsYtWc3F7C?usp=drive_link

4.3 Tạo project “Bluetooth Mesh - SoC Empty” Project với tên “btmesh_soc_vendor_relay” (Có thể bỏ qua)

- Cài đặt **Log** (*Phụ lục, mục 1*).
- Cài đặt Vendor Model , Test, Button Press (*Phụ lục, mục 2,3,4*)
- Cài đặt driver WSTK (*Phụ lục, mục 6*)
- Config model và thêm model vào Main (*Phụ lục, mục 5*), đối với server thiết lập Model ID là 0x3333
- Sau đó thay thế file các file trong project hiện tại bằng các file trong link bên dưới. (Chọn Copy files -> Overwrite all):

https://drive.google.com/drive/folders/1pSe5tnZfFEVwK2rFV6UCIRB5Ij8YuGSL?usp=drive_link

4.4 Tạo project “Bluetooth Mesh - SoC Empty” Project với tên “btmesh_soc_vendor_provisioner”

- Cài đặt **Log** (*Phụ lục, mục 1*).
- Cài đặt Vendor Model , Test, Button Press (*Phụ lục, mục 2,3,4*)
- Cài đặt Button Press (*Phụ lục, mục 4*).
- Cài đặt driver WSTK (*Phụ lục, mục 6*).
- Thêm chức năng Factory Reset (*Phụ lục, mục 8*).
- Thêm chức năng Provisioner (*Phụ lục, mục 9*).
- Thêm chức năng config client (*Phụ lục, mục 10*).
- Cấu hình cho provisioner (*Phụ lục, mục 11*).
- Thêm Configuration Client vào Main(*Phụ lục, mục 12*).
- Sau đó thay thế file các file trong project hiện tại bằng các file trong link bên dưới. (Chọn Copy files -> Overwrite all):

https://drive.google.com/drive/folders/1Z8uG-Xa0Cbx0TDJtoggZ_bc7MaF3r9i?usp=drive_link

Lưu ý khi nạy chương trình mà log không in hay lcd không hiển thị làm theo mục hướng dẫn “**Flash chương trình Mesh mẫu vào board**” (Phụ lục, mục 13)

5. Chương trình cụ thể:

Sau khi tiến hành xây dựng 3 project cho 3 board, các bạn có thể sử dụng Hub để nạp cho 3 board hoặc các máy khác nhau nạp cho các board. Các bạn cần lưu ý xác định mã ID của từng board khi nạp để không nhầm lẫn giữa các thiết bị hoặc dựa vào màn hình LCD xác định từng thiết bị và các lệnh chức năng của từng board.

5.1 Thiết bị Provisioner

Thiết bị Provisioner thực hiện chức năng **thêm các thiết bị chưa được cấp phát vào mạng Mesh** và thực hiện **cấu hình cho từng thiết bị**. Các thiết bị trong cùng mạng Mesh phải khác nhau về ID (unicast address) cần được thiết lập và phải giống nhau về appkey cùng với Company ID.

```
#define VENDOR_ID          0x1221
#define MY_VENDOR_SERVER_ID  0x1111
#define MY_VENDOR_CLIENT_ID  0x2222
#define MY_VENDOR_RELAY_ID   0x3333

#define CUSTOM_STATUS_GRP_ADDR      0xC001 // Server PUB
#define CUSTOM_CTRL_GRP_ADDR       0xC002 // Server SUB
#define BLE_MESH_UUID_LEN_BYTE    (16)
#define BLE_ADDR_LEN_BYTE         (6)
#define MAX_NUM_BTmesh_DEV        (4)

device_table_entry_t bluetooth_device_table[MAX_NUM_BTmesh_DEV];

static const uint8_t fixed_netkey[16] = {0x23, 0x98, 0xdf, 0xa5, 0x09, 0x3e,
0x74, 0xbb, 0xc2, 0x45, 0x1f, 0xae, 0xea, 0xd7, 0x67, 0xcd};
static const uint8_t fixed_appkey[16] = {0x16, 0x39, 0x38, 0x03, 0x9b, 0x8d,
0x8a, 0x20, 0x81, 0x60, 0xa7, 0x93, 0x33, 0x3d, 0x03, 0x61};
```

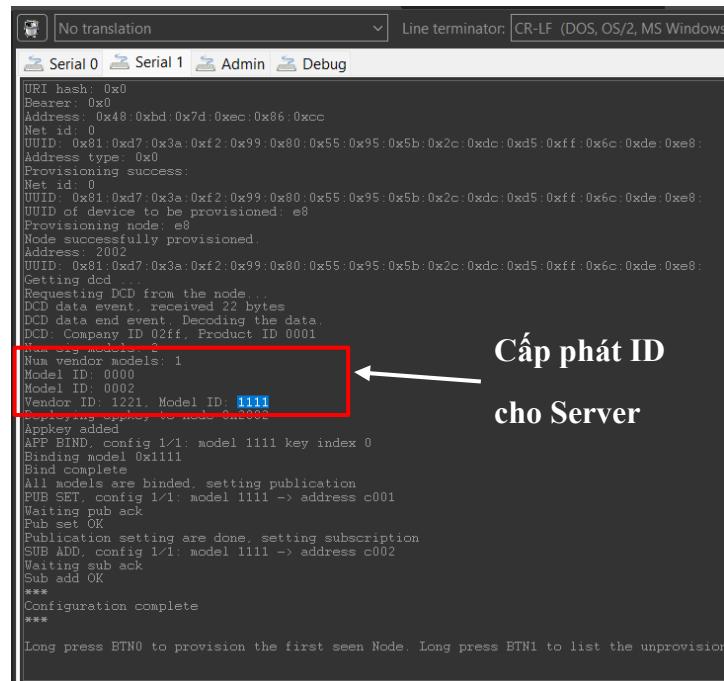
Các ID được cấp phát cho từng thiết bị

Các appkey và netkey được cố định trong mạng Mesh

- ❖ Để khởi động thiết bị Provisioner bạn cần **nhấn giữ nút Button[0]** cùng với đó **nhấn “Reset”** để chương trình vào chế độ Factory Reset và Boot lại nhằm xóa hết bộ nhớ còn lưu trữ trong Flash. (Ưu tiên đưa các thiết bị Client, Server và Relay vào trạng thái Factoty Reset trước khi thực hiện cho thiết bị Provisioner và nên cấp phát từng thiết bị để tránh nhầm lẫn)

Tiếp đến để phát hiện các thiết bị chưa được cấp phát xung quanh ta cần nhấn giữ **Button[0]** trong 1 khoảng thời gian và thả ra để provisioner quét được thiết bị chưa được cấp phát tiếp đèn nhấp nháy **Button[0]** một khoảng thời gian ngắn để thực hiện cấp phát.

Tương tự thê ta lần lượt cấp phát cho các thiết bị “Server”, “Client” và “Relay”.



Cáp phát ID cho Server

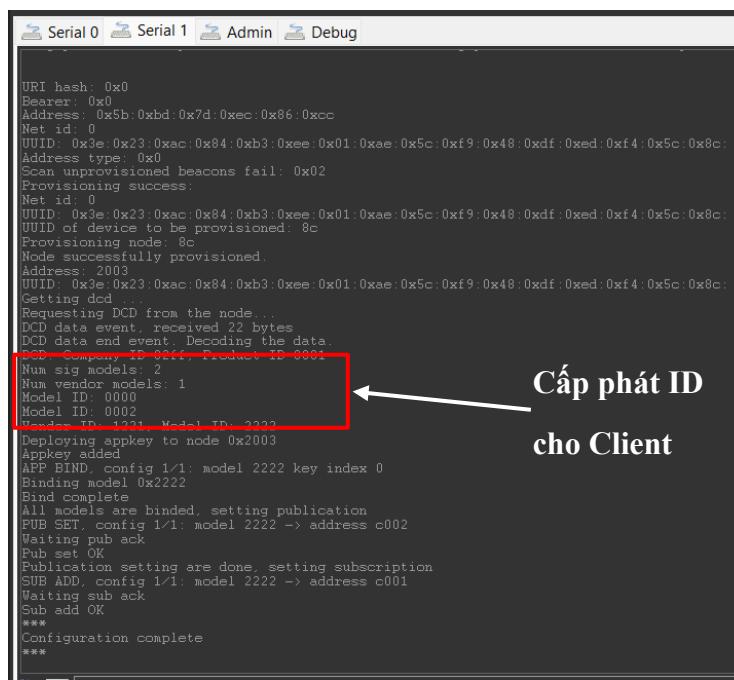
```

URI hash: 0x0
Bearer: 0x0
Address: 0x48:0xbd:0x7d:0xec:0x86:0xcc
Net id: 0
UUID: 0x81:0xd7:0x3a:0xf2:0x99:0x80:0x55:0x95:0x5b:0x2c:0xdc:0xd5:0xff:0x6c:0xde:0xe8:
Address type: 0x0
Provisioning success:
Net id: 0
UUID: 0x81:0xd7:0x3a:0xf2:0x99:0x80:0x55:0x95:0x5b:0x2c:0xdc:0xd5:0xff:0x6c:0xde:0xe8:
UUID of device to be provisioned: e8
Provisioning node: e8
Node successfully provisioned.
Address: 2002
UUID: 0x81:0xd7:0x3a:0xf2:0x99:0x80:0x55:0x95:0x5b:0x2c:0xdc:0xd5:0xff:0x6c:0xde:0xe8:
Getting DCD ...
Requesting DCD from the node...
DCD data event, received 22 bytes
DCD data end event. Decoding the data.
DCD: Company ID 02ff, Product ID 0001
Num sig models: 2
Num vendor models: 1
Model ID: 0000
Model ID: 0002
Vendor ID: 1221, Model ID: 1111
Deploying appkey to node 0x2002
Appkey added
APP BIND config 1/1: model 1111 key index 0
Binding model 0x1111
Bind complete
All models are binded, setting publication
PUB SET config 1/1: model 1111 -> address c001
Waiting pub ack
Sub add OK
***
Configuration complete
***

Long press BTN0 to provision the first seen Node. Long press BTN1 to list the unprovisioned nodes.

```

Cáp phát cho Server



Cáp phát ID cho Client

```

URI hash: 0x0
Bearer: 0x0
Address: 0x5b:0xbd:0x7d:0xec:0x86:0xcc
Net id: 0
UUID: 0x3e:0x23:0xac:0x84:0xb3:0xee:0x01:0xae:0x5c:0xt9:0x48:0xdf:0xed:0xf4:0x5c:0x8c:
Address type: 0x0
Scan unprovisioned beacons fail: 0x02
Provisioning success:
Net id: 0
UUID: 0x3e:0x23:0xac:0x84:0xb3:0xee:0x01:0xae:0x5c:0xt9:0x48:0xdf:0xed:0xf4:0x5c:0x8c:
UUID of device to be provisioned: 8c
Provisioning node: 8c
Node successfully provisioned.
Address: 2003
UUID: 0x3e:0x23:0xac:0x84:0xb3:0xee:0x01:0xae:0x5c:0xt9:0x48:0xdf:0xed:0xf4:0x5c:0x8c:
Getting DCD ...
Requesting DCD from the node...
DCD data event, received 22 bytes
DCD data end event. Decoding the data.
DCD: Company ID 0001, Product ID 0001
Num sig models: 2
Num vendor models: 1
Model ID: 0000
Model ID: 0002
Vendor ID: 1221, Model ID: 2222
Deploying appkey to node 0x2003
Appkey added
APP BIND config 1/1: model 2222 key index 0
Binding model 0x2222
Bind complete
All models are binded, setting publication
PUB SET config 1/1: model 2222 -> address c001
Waiting pub ack
Sub add OK
Publication setting are done, setting subscription
SUB ADD config 1/1: model 2222 -> address c001
Waiting sub ack
Sub add OK
***
Configuration complete
***


```

Cáp phát cho Client

5.2 Thiết bị Client

```

static my_model_t my_model = {
    .elem_index = PRIMARY_ELEMENT,
    .vendor_id = VENDOR_ID,
    .model_id = MY_VENDOR_CLIENT_ID,
    .publish = 1,
    .opcodes_len = 1,
    .opcodes_data[0] = sensor_status
};

static void factory_r
static void read_sensor_data(void);

```

Macro Expansion
0x1
Press 'F2' for focus;

- ❖ Để khởi động thiết bị Client bạn cần **nhấn giữ nút Button[0]** cùng với đó **nhấn “Reset”** để chương trình vào chế độ Factory Reset nhằm xóa hết bộ nhớ còn lưu trữ trong Flash (**Làm đến đây bạn cần cấu hình thiết bị Provisioner trước khi nhấn “Reset”** để thiết bị có thể thực hiện quảng bá để được cấp phát).

Thiết bị Client được dùng để **đọc và gửi dữ liệu cảm biến** (nhiệt độ, độ ẩm) đến các node khác trong mạng.

Khi khởi động, thiết bị sẽ khởi tạo ngăn xếp Bluetooth và Mesh, thiết lập địa chỉ, tên thiết bị và kiểm tra xem node đã được “provisioned” (thêm vào mạng Mesh) hay chưa.

Nếu chưa thiết bị sẽ thực hiện quảng bá và được cấp phát thêm vào mạng. Sau khi sẵn sàng, người dùng có thể điều khiển việc truyền dữ liệu bằng cách nhấn nút:

PB0 dùng để đọc cảm biến và gửi dữ liệu ngay lập tức.

PB1 để chọn và kích hoạt chế độ gửi dữ liệu định kỳ.

Dữ liệu thu được từ cảm biến được lưu trong mảng `sensor_data[8]`, chứa 4 byte độ ẩm và 4 byte nhiệt độ, sau đó được gửi đi thông qua **Vendor Model** bằng hai hàm chính `sl_btmesh_vendor_model_set_publication()` và `sl_btmesh_vendor_model_publish()`. Code đang được thiết lập sử dụng opcode là `0x01` với chức năng truyền dữ liệu cảm biến “`sensor_status`”.

Code của Client đang thực hiện gửi dữ liệu giả tạm thời.

The terminal window shows the execution of a script named "factory_reset". The output is annotated with arrows pointing to specific sections of the code:

- Thực hiện quảng bá**: Points to the "Client Device" initialization section, which includes "Node init", "Node already initialized", and "Device name: 'Client bd:5b'".
- Provisioner config cho Client**: Points to the "Provisioning started" section, which includes "Got new Network key with index 0", "Provisioning done. Address: 0x2003, IV Index: 0x0", and "Setting up client functionality...".
- Client lấy dữ liệu cảm biến và Publish cho Server**: Points to the "Client" publishing section at the bottom, which includes "Temp: -20.0 Celsius", "Hum: 0 %", "B0 Pressed. Data is sent once.", "Set publication done. Publishing...", and "Publish done."

Kết quả ở Client

5.3 Thiết bị Server

- Để khởi động thiết bị Client bạn cần **nhấn giữ nút Button[0]** cùng với đó **nhấn “Reset”** để chương trình vào chế độ Factory Reset nhằm xóa hết bộ nhớ còn lưu trữ trong Flash (**Làm đến đây bạn cần cấu hình thiết bị Provisioner trước khi nhấn “Reset”** để thiết bị có thể thực hiện quảng bá để được cấp phát).

Là thiết bị đầu cuối nhận dữ liệu từ client, với opcode hiện tại Client đang gửi là 0x01 cho Sensor data và tùy thuộc vào kiến trúc gửi đi mà phía Server thực hiện giải mã tương ứng cho dữ liệu.

[C] C:/Users/ADMN/SimplicityStudio/SDKs/simplicity_sdk_4/app/btmesh/common/btmesh
Initialization failed. Failed to start unprovisioned beaconing=====
Server Device
Node init
Node already initialized
Node initialized ...
Device name: 'Server bd:48'
Node unprovisioned
Send unprovisioned beacons.
Provisioning started.
Got new network key with index 0
Provisioning done. Address: 0x2002, IV Index: 0x0
Setting up server functionality...
Relay enabled
Network tx state set
Got node address: 0x2002
Server initialization complete
Got new application key with index 0
Model config changed, type: 0, elem_addr: 2002, model_id: 1111, vendor_id: 1221
Model config changed, type: 1, elem_addr: 2002, model_id: 1111, vendor_id: 1221
Model config changed, type: 2, elem_addr: 2002, model_id: 1111, vendor_id: 1221
New data stored.
Vendor model data received.
Element index = 0
Vendor id = 0x1221
Model id = 0x1111
Source address = 0x2003
Destination address = 0xC002
Destination label UUID index = 0x00
App key index = 0x0000
Non-relayed = 0x01
Opcode = 0x01
Final = 0x0001
Payload: 0 0 0 e0 b1 ff ff
Temperature = -20.0 Celsius
Temperature = -4.0 Fahrenheit
Humidity = 0 %

Thực hiện quảng bá

Provisioner config cho Server

Server nhận dữ liệu từ Client

Kết quả ở Server

```
switch (rx_evt->opcode) {
    case sensor_status:
        int32_t temperature = 0;
        uint32_t humidity = 0;
        for (int8_t i = 7; i >= 4; i--) {
            uint8_t temp = rx_evt->payload.data[i];
            temperature = (temperature << 8) | temp;
        }
        for (int8_t i = 3; i >= 0; i--) {
            uint8_t temp = rx_evt->payload.data[i];
            humidity = (humidity << 8) | temp;
        }
        app_log("Temperature = %ld.%1ld Celsius\r\n",
                temperature / 1000,
                temperature % 1000);

        float temp = (float) (temperature / 1000);
        temp = temp * 1.8 + 32;
        temperature = (int32_t) (temp * 1000);
        app_log("Temperature = %ld.%1ld Fahrenheit\r\n",
                temperature / 1000,
                temperature % 1000);

        app_log("Humidity = %ld %%\r\n",
                humidity / 1000);
        break;

    default:
        break;
}
```

5.3 Thiết bị Relay

Là 1 thiết bị trung gian thực hiện chức năng bắt dữ liệu từ Client và chuyển tiếp gói tin đến Server với mục tiêu mở rộng phạm vi truyền tải của thiết bị đầu vào và thiết bị đầu cuối.

Dữ liệu sau khi được relay bắt được tiến hành đóng gói và tiếp tục Publish đến Server.

```
// set the vendor model publication message
sc = sl_btmesh_vendor_model_set_publication(my_model.elem_index,
                                              my_model.vendor_id,
                                              my_model.model_id,
                                              rx_evt->opcode,
                                              rx_evt->final,
                                              rx_evt->payload.len,
                                              rx_evt->payload.data);
```

Bài chuẩn bị

1. Tìm hiểu về cơ chế của BLE-Mesh.
 2. Cho biết các bước thực hiện cấp phát và cơ chế Publish và Subscribe.
-

Bài thực hành

1. Xây dựng hệ thống với việc truyền tải MSSV của 2-4 bạn trong nhóm và Thiết bị Server thực hiện bắt dữ liệu và phân tích dữ liệu đó để hiển thị lên console.
2. Xây dựng ứng dụng gửi gói tin khi nhấn **Button[0]** là thời gian hoạt động của thiết bị (tính từ lúc thiết bị được cấp nguồn) và **Button[1]** là thông tin trạng thái của 2 LED (Với 2 Led tương ứng với 2 bit đếm lên lần lượt mỗi 5s). Lưu ý phần Server cần giải mã phần dữ liệu vừa gửi đi và in lên màn hình LCD

VD:

00 → LED[0]: OFF, LED[1]: OFF

01 → LED[0]: OFF, LED[1]: ON

10 → LED[0]: ON, LED[1]: OFF

11 → LED[0]: ON, LED[1]: ON

*Vì hệ thống mang tính đặc thù về kết nối do đó sẽ khó khăn khi thực hiện trên lớp (Các thiết bị được thêm vào mạng Mesh nhầm lẫn với nhau giữa các nhóm gây sai sót về thông tin) và báo cáo trong 2 tuần (Có ảnh minh chứng chi tiết mới có điểm cho phần đó).

Thực hiện theo dạng báo cáo **tối đa 4 bạn / 1 nhóm** và chỉ **đại diện nhóm nộp**. Các nhóm khi làm **tránh trường hợp làm gần nhau** gây nên tình trạng cấp phát nhầm sang thiết bị của nhóm khác.

Thời gian mượn và trả board theo các ngày (*Các bạn chỉ mượn và trả đúng thời gian quy định để dễ quản lý. Mỗi board cần 1 thẻ sinh viên và bài lab này cần tối thiểu 3 board để hoàn thành* các bạn chuẩn bị trước khi mượn board.

Mượn và trả board tại phòng E103a. Các nhóm ngồi tự do (tại thư viện hoặc khu tự học) và hạn chế gần nhau khi thực hiện. (Tối đa 54 board / 1 ca)

- 1) T4 (**Chỉ 26/11**) : Mượn board: 3 ca (9h30-11h30) (13h-15h) và (15h-17h)
- 2) CN : Mượn board: 3 ca (9h30-11h30) (13h-15h) và (15h-17h)

Link đăng ký: [Đăng ký mượn board IoT và SoC.xlsx - Google Trang tính](#)

Tài liệu tham khảo

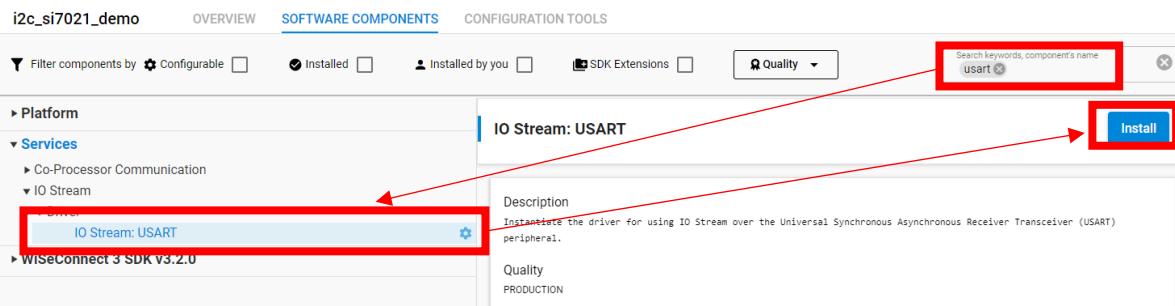
- [1] Silicon Labs, “UG427: EFR32xG21 2.4 GHz 20 dBm Radio Board User's Guide”, Rev. 2.0, 2022. [Online]. Available: <https://www.silabs.com/documents/public/user-guides/ug427-brd4180b-user-guide.pdf>
- [2] Bluetooth SIG., “Assigned numbers”. 2024. [Online]. Available: <https://www.bluetooth.com/specifications/assigned-numbers/>
- [3] Bluetooth SIG., “Core Specification Supplement”. 2024. [Online]. Available: <https://www.bluetooth.com/specifications/specs/core-specification-supplement-10/>

Phụ lục

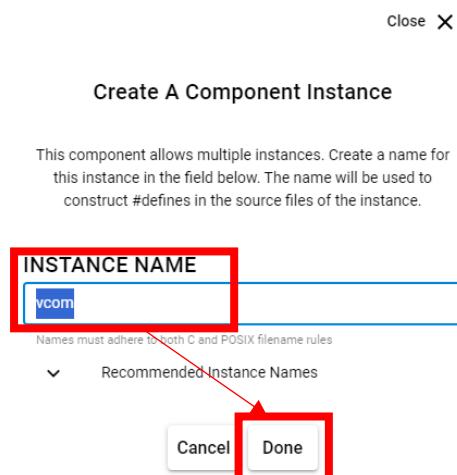
1. Cài đặt Log

1. Mở Project Configurator bằng cách double-click vào file .slcp.

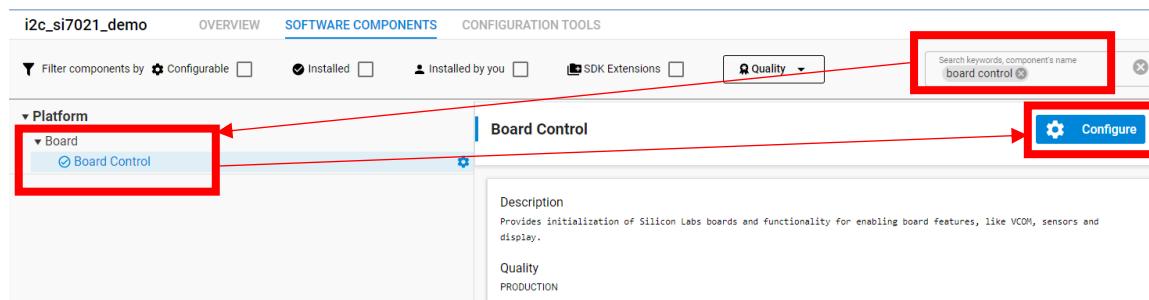
2. Tìm kiếm “USART” → “IO Stream: USART” → Install.



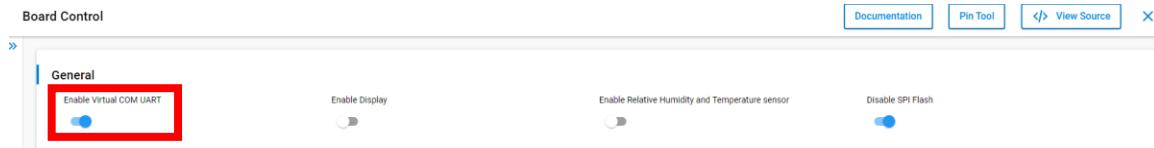
3. Giữ nguyên “INSTANCE NAME” → Done.



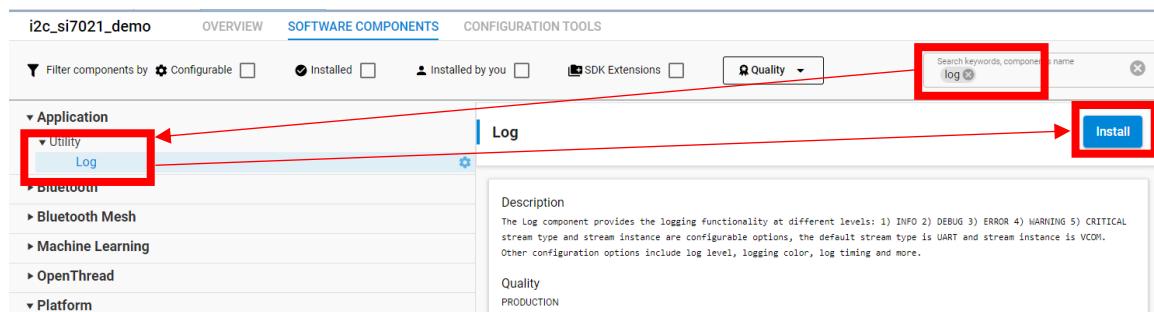
4. Tìm kiếm “Board Control” → “Board Control” → Configure.



5. Kích hoạt “Enable Virtual COM UART”.



6. Tìm kiếm “log” → “Log” → Install.



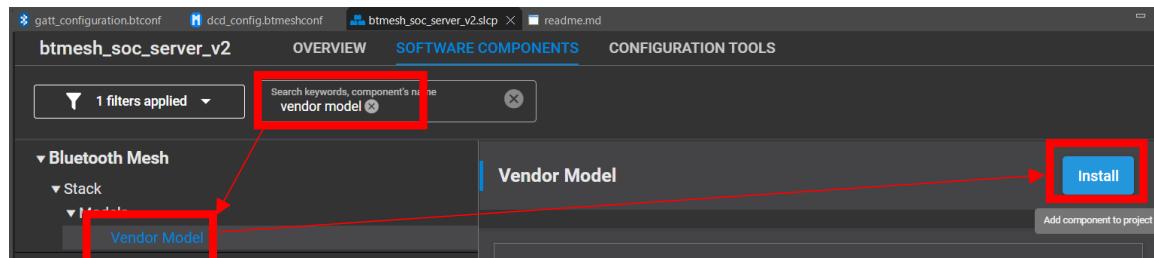
7. Cách sử dụng.

```
#include "app_log.h"
app_log_info("..."); 
app_log_warning("..."); 
app_log("..."); 

-----
Ví dụ:
#include "app_log.h"
...
int a = 10;
app_log_info("value: %d\r\n", a);
```

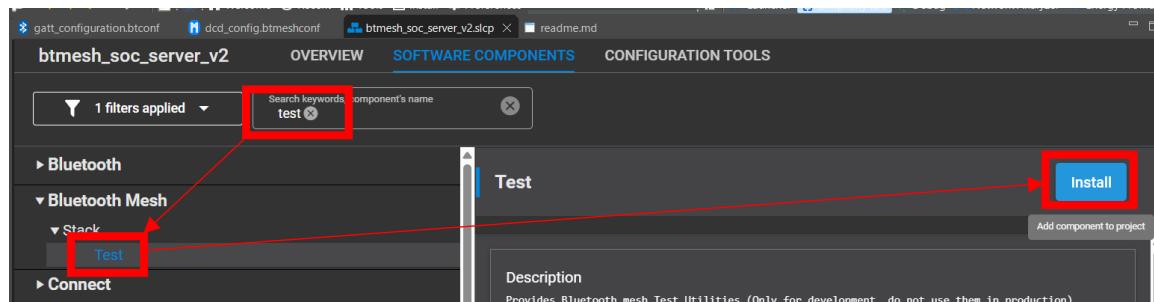
2. Cài đặt Vendor Model

Tìm kiếm “Vendor Model” → “Bluetooth Mesh → Stack → Model → Vendor Model Install”.



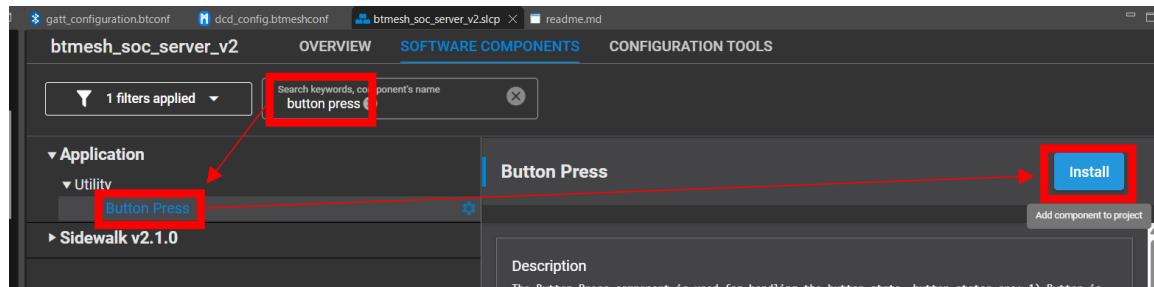
3. Cài đặt Test

Tìm kiếm “Test” → “Bluetooth Mesh → Stack → **Test** Install”.



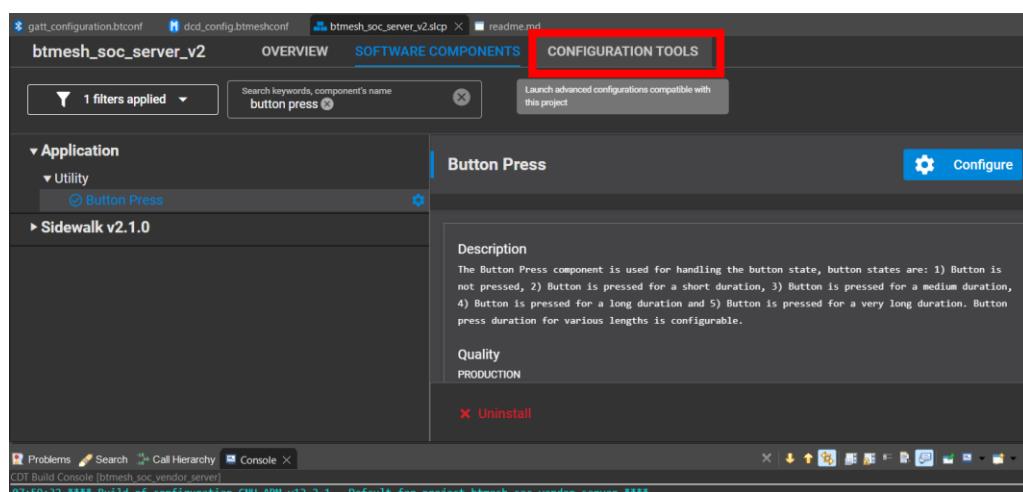
4. Cài đặt Button Press

Tìm kiếm “button press” → “Application → Utility → **Button Press** Install”.

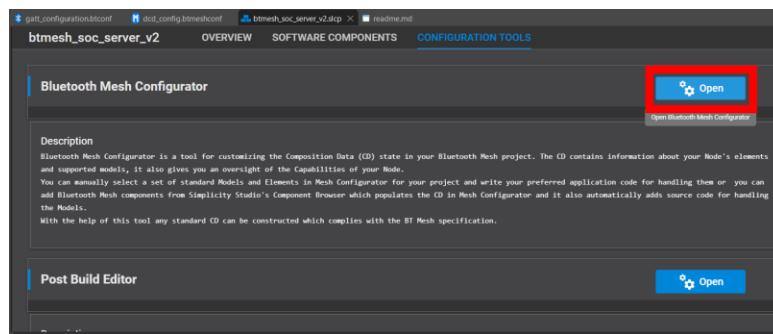


5. Thêm model vào Main

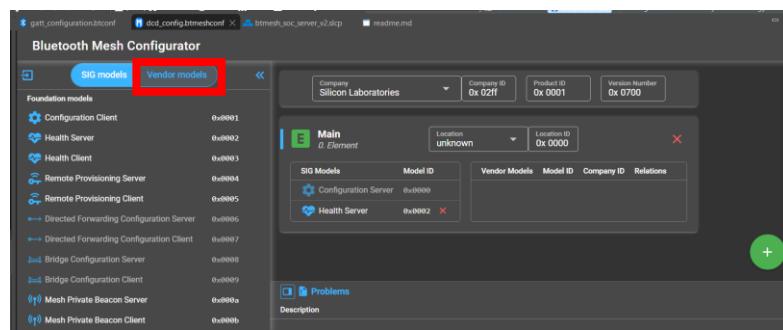
Click sang mục “CONFIGURATION TOOLS”



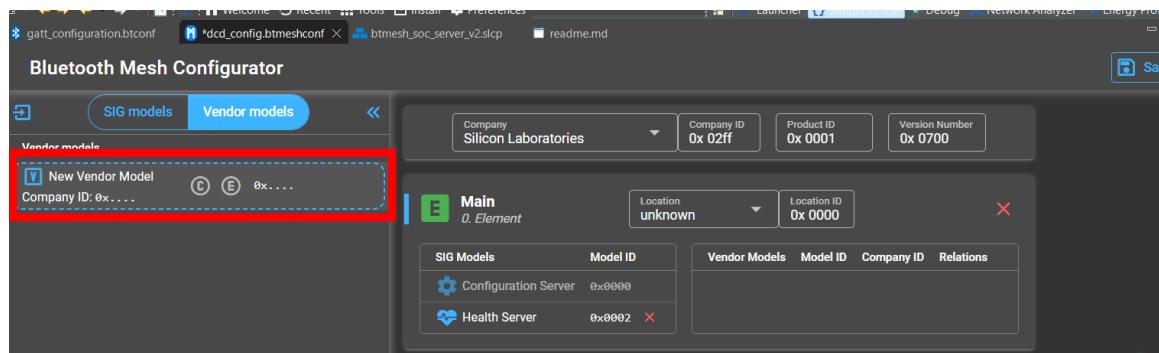
Chọn “*Open*” Bluetooth Mesh Configuration



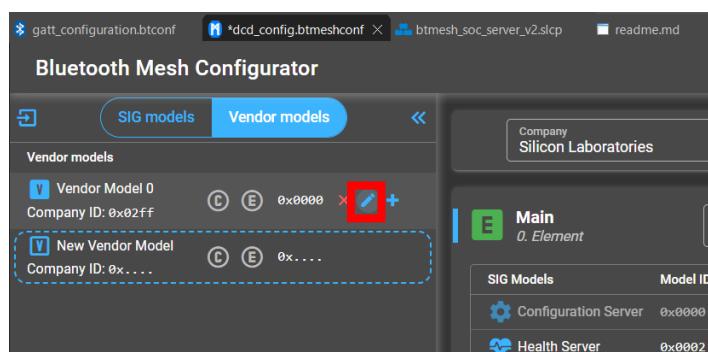
Click vào phần “*Vendor models*”



Click vào mục “*New Vendor Models*”



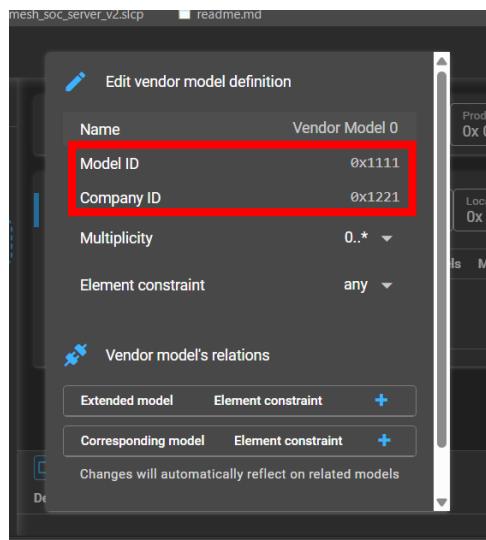
Chọn phần “*Edit* ở *Vendor Model 0*” vừa thêm



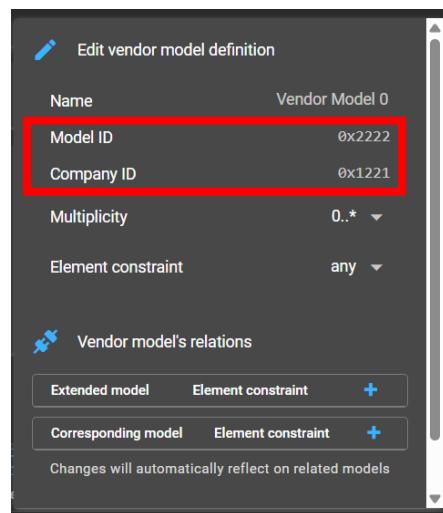
Đối với client

Thiết lập 2 giá trị **Model ID** và **Company ID** như hình

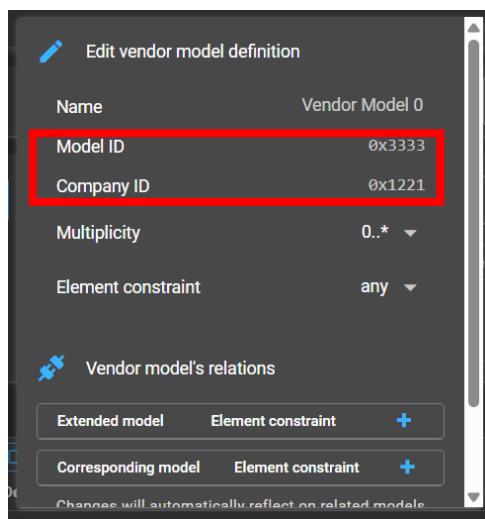
Đối với server



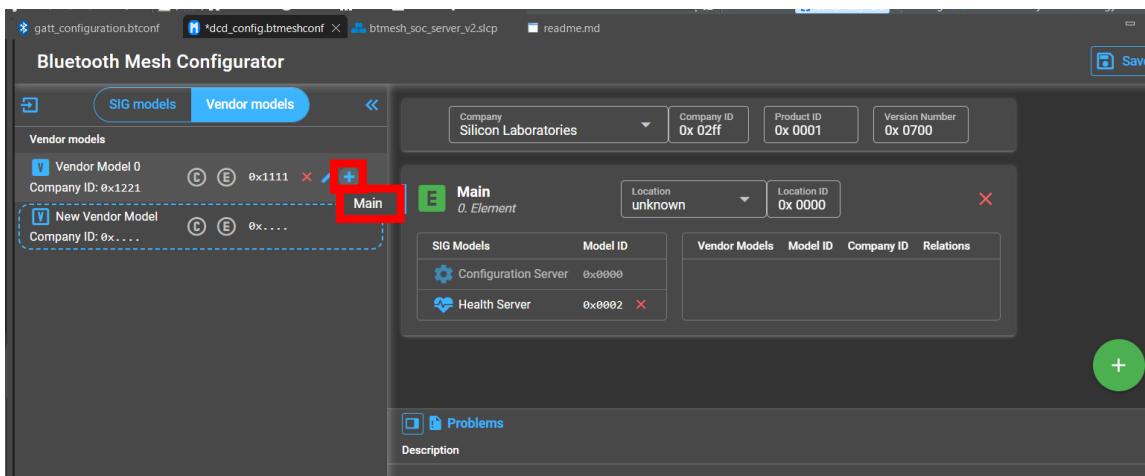
Đối với client



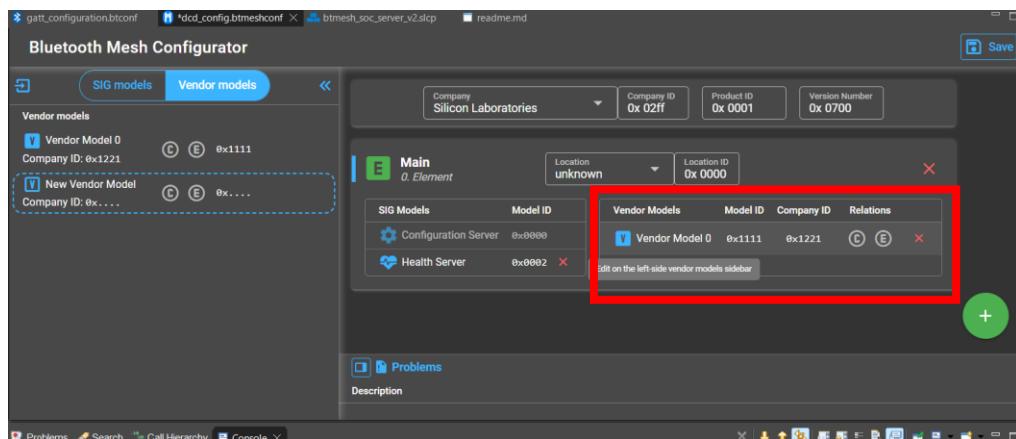
Đối với relay



Chọn **Add sau đó chọn Main** để thêm Model vào hệ thống

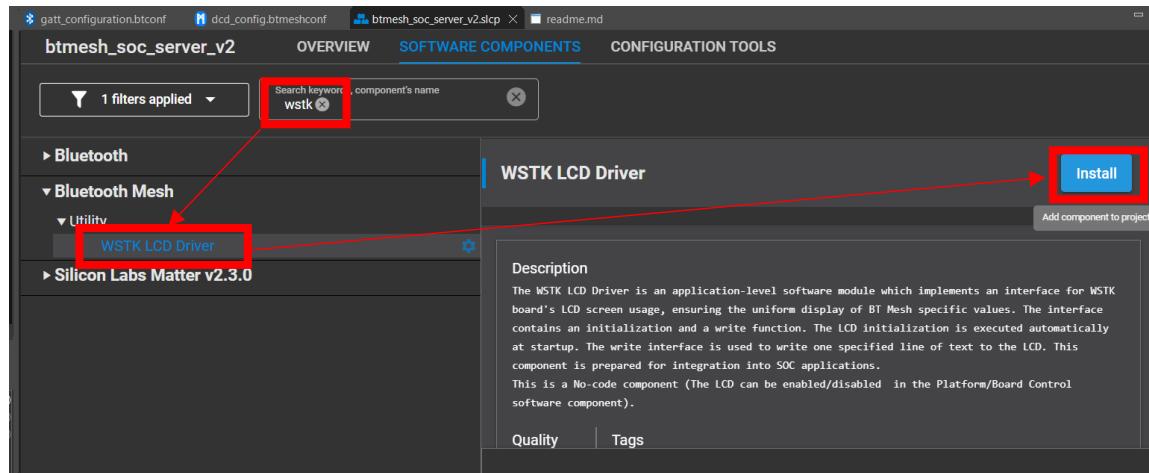


Kết quả sẽ hiển thị model ở cửa sổ bên phải



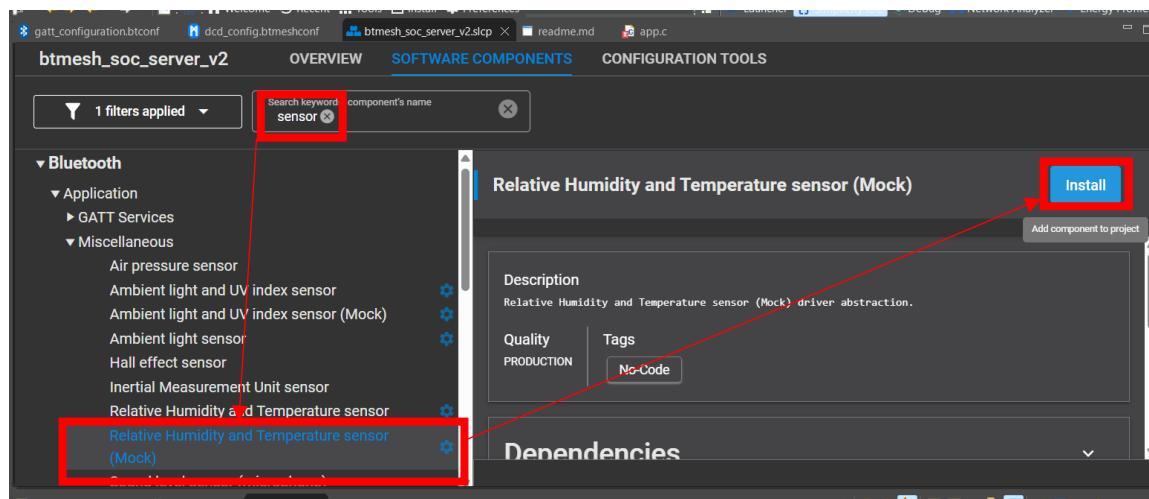
6. Thêm thư viện WSTK để hiển thị LCD

Tìm kiếm “wstk” → “Bluetooth Mesh → Utility → **WSTK LCD Driver** Install”.



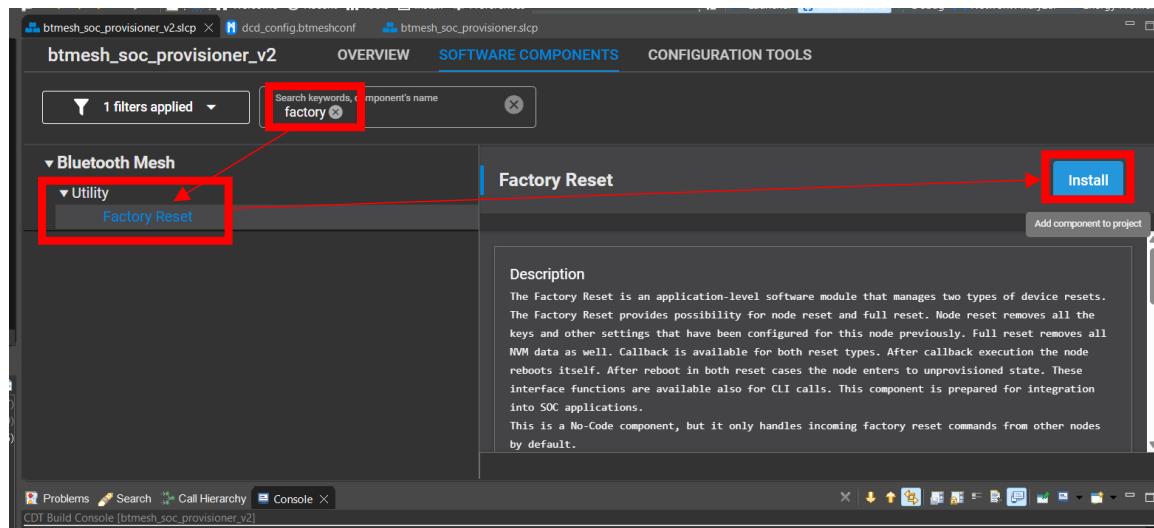
7. Thêm thư viện đọc cảm biến (Chỉ ở client)

Tìm kiếm “sensor” → “Bluetooth → Miscellaneous → **“Relative Humidity and Temperature sensor (Mock)”** Install”.



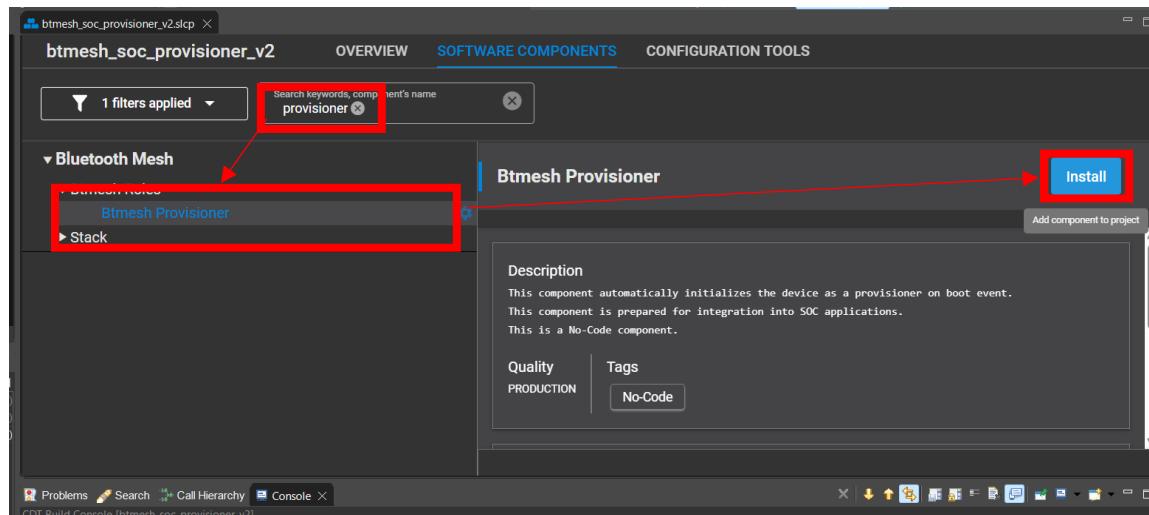
8. Thêm chức năng Factory Reset (Chỉ ở provisioner)

Tìm kiếm “factory” → “Bluetooth Mesh → Utility → “Factory Reset” Install”.

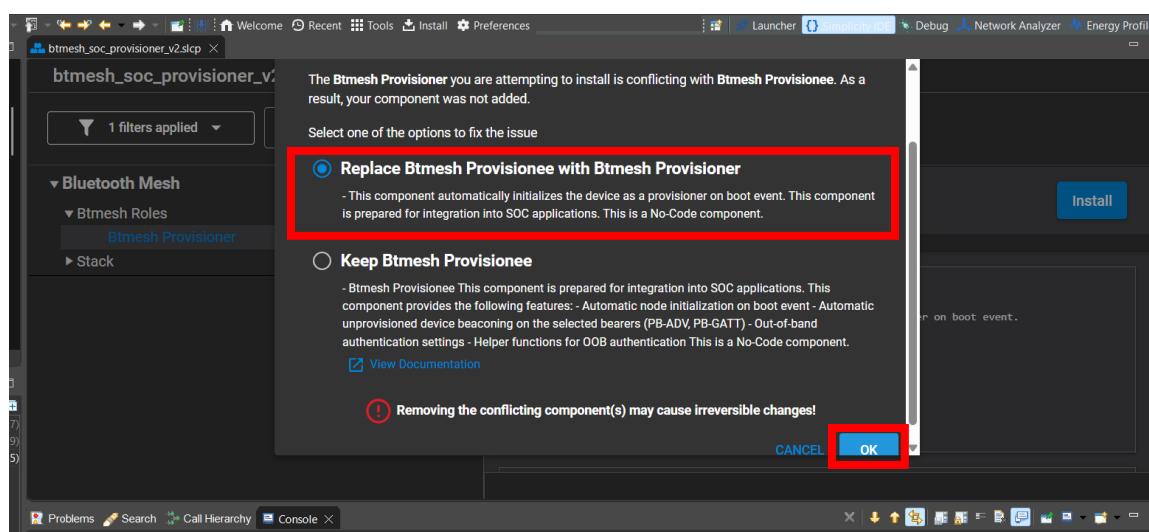


9. Thêm chức năng Provisioner (Chỉ ở provisioner)

Tìm kiếm “provisioner” → “Bluetooth Mesh → Btmesh Roles → “Btmesh Provisioner”
Install”.

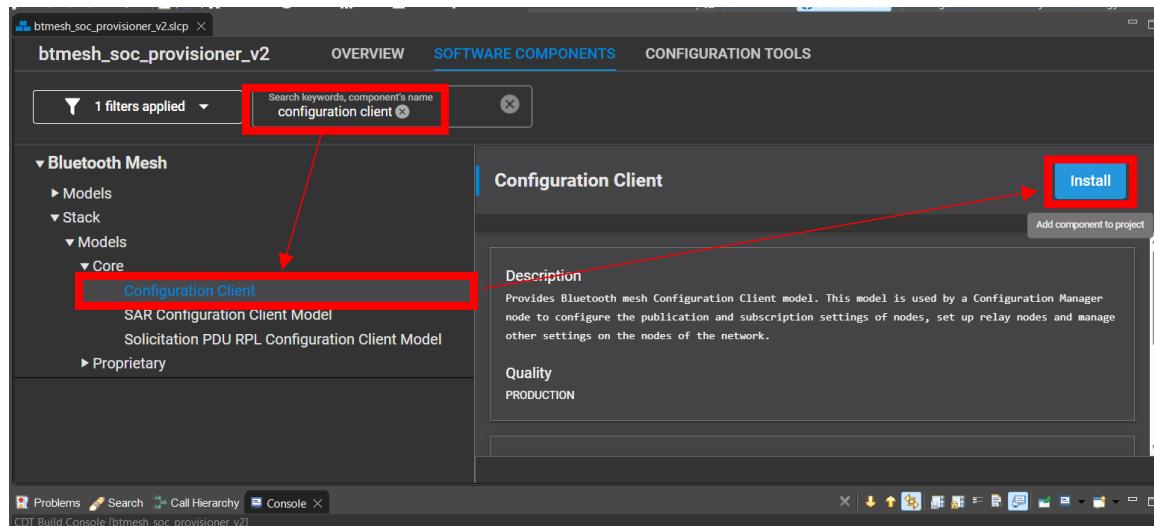


Nếu xuất hiện bảng sau chọn Replace, nếu ko có thì bỏ qua.



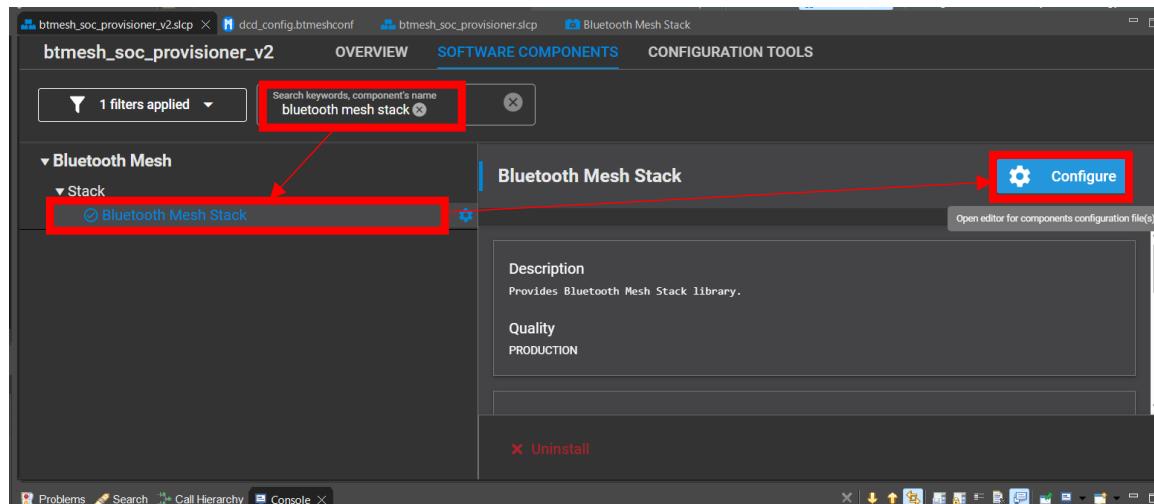
10. Thêm chức năng config (Chỉ ở provisioner)

Tìm kiếm “configuration client” → “Bluetooth Mesh → Stack → Models → Core → “Configuration Client” Install”.

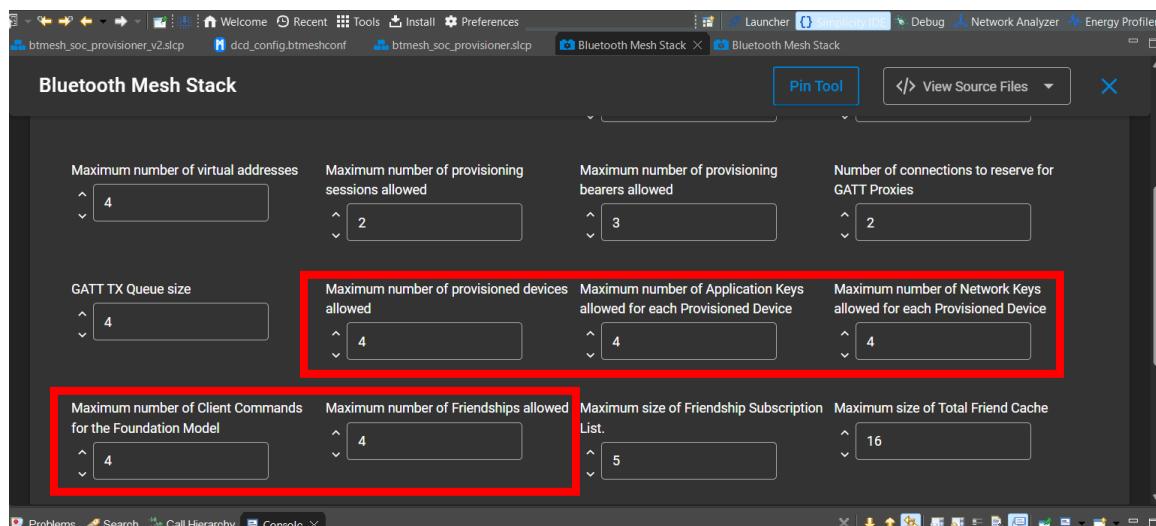


11. Cấu hình cho provisioner (Chỉ ở provisioner)

Tìm kiếm “bluetooth mesh stack” → “Bluetooth Mesh → Stack → **Bluetooth Mesh Stack Chọn Configure**”.

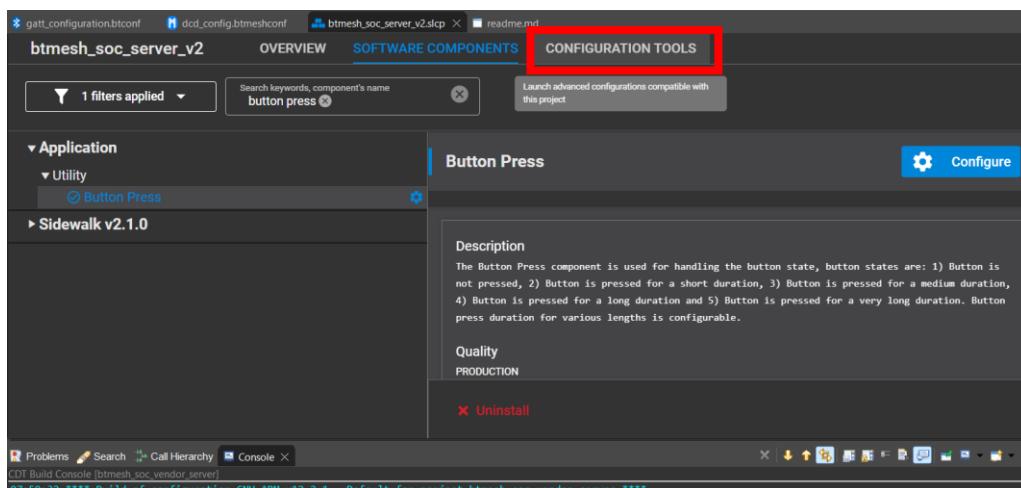


Ở cửa sổ Configure thiết lập các giá trị như bảng sau

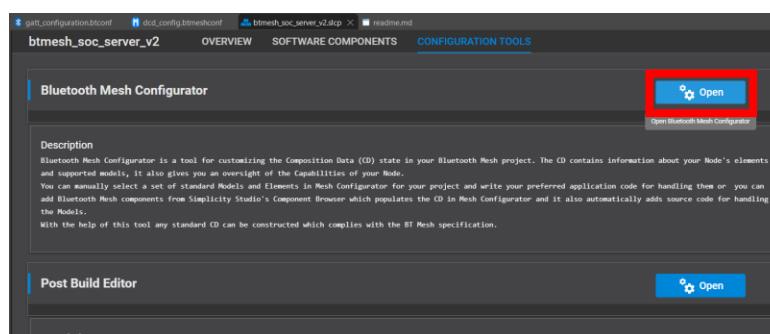


12. Thêm Configuration Client vào Main

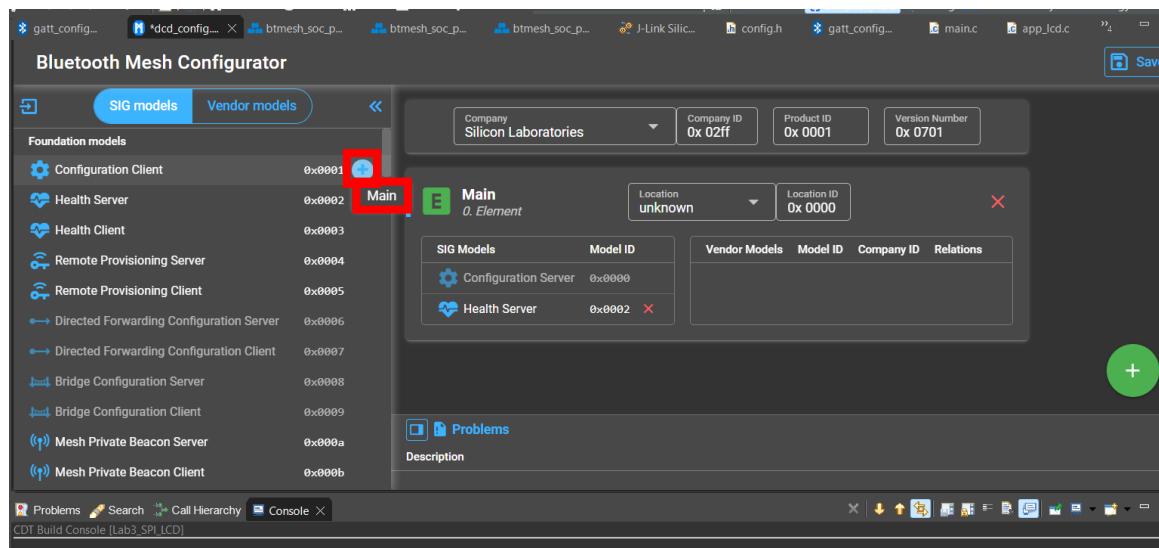
Click sang mục “CONFIGURATION TOOLS”



Chọn “Open” Bluetooth Mesh Configuration



Thêm phần “Configuration Client” vào Main

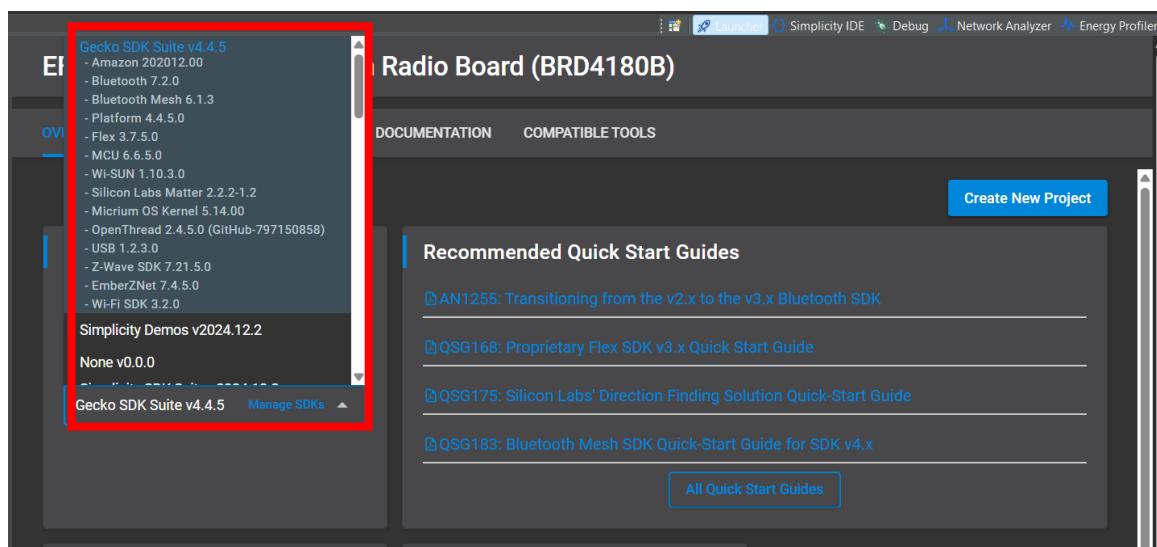


13. Flash chương trình Mesh mẫu vào board (Dùng cho trường hợp nạp chương trình không được)

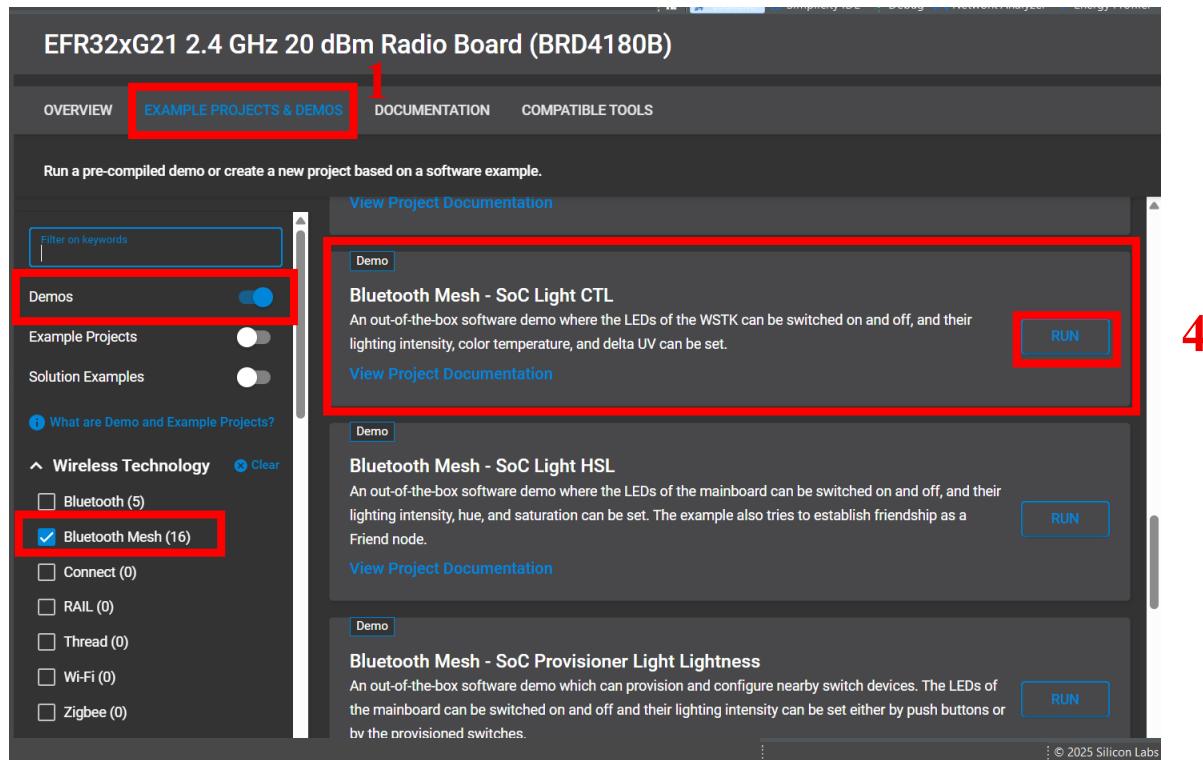
Tiến hành nạp 1 chương trình mẫu về Mesh vào board sau đó tiến hành nạp chương trình Mesh mà mình xây dựng vào như bình thường.

Ta về Wellcome và tiến hành chạy 1 demo như hướng dẫn bên dưới

Chọn “Gecko SDK Suite v4.4.5” hay bất kỳ Gecko SDK nào bạn đang có



Chọn vào mục “Example Projects & Demo” → Chỉ mờ tích chọn phần “Demos” và “Bluetooth Mesh” → Nhấn “Run” cho demo “Bluetooth Mesh – SoC Light CTL”. Khi nạp xong LCD hiển thị và Log có in ra là đã tiến hành Flash lại cho chương trình Mesh bạn có thể nạp chương trình mình vào bình thường

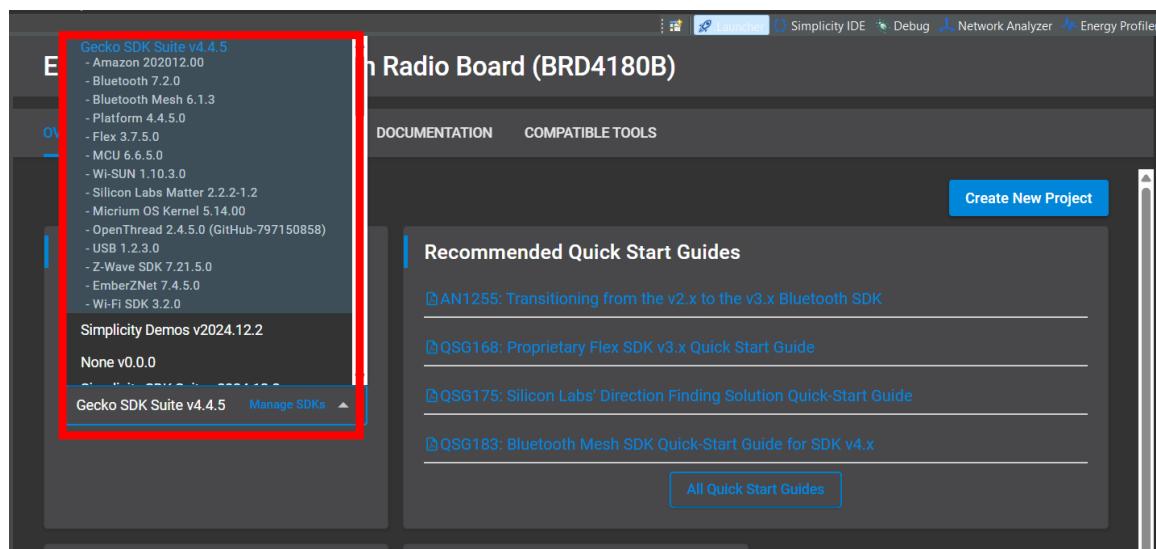


14. Flash chương trình Bluetooth mẫu vào board (Dùng cho trường hợp nạp chương trình không được)

Khi bạn hay bạn khác đã nạp 1 chương trình Mesh vào board dẫn đến việc khi chạy chương trình ở chế độ Bluetooth thông thường sẽ gây ra tình trạng lỗi. Do đó bạn cần Flash 1 chương trình mẫu về Bluetooth vào board.

Cũng tương tự các trên ta về Wellcome và tiến hành chạy 1 demo như hướng dẫn bên dưới

Chọn “Gecko SDK Suite v4.4.5” hay bất kỳ Gecko SDK nào bạn đang có



Chọn vào mục “Example Projects & Demo” → Chỉ mở tích chọn phần “Demos” và “Bluetooth” → Nhấn “Run” cho demo “Bluetooth– SoC Blinky”. Khi nạp xong LCD hiển thị và Log có in ra là đã tiến hành Flash lại cho chương trình Mesh bạn có thể nạp chương trình mình vào bình thường

