ECEN 5823

BLE Client Assignment

Fall 2018

Objective: To create BLE client that connects and gets the temperature readings from the already developed BLE Server application.

Note: This assignment will begin with the completed BLE Server assignment.

The BLE client project can be made by copying the BLE server project you created by removing unnecessary events in the while(1) loop or can use the soc-empty template project you get from the Simplicity Studio.

Due (submission):

BLE Server Project extra required features (LCD and security) demo – by End of Day Sunday, October 7th

Flow chart – by End of Day Wednesday, October 10th

Final BLE Client demo – by End of Day Wednesday, October 17th

References required:

Silicon Labs Bluetooth Software API Reference Manual

Deliverables:

1. BLE server project with LCD interface showing the device's Bluetooth Address on LCD along with various other LCD messages as described below. BLE server project supporting MITM protection and encryption feature. Required submission to be demo to the instructing team and uploading your project into Canvas.
2. A flowchart diagram showing the detailed description and flow of the BLE Client software – It should include the APIs used and the events triggered by each API call, the next action to be taken for that event and the current state of the software(Disconnected, Connected, Discover Services, Discover

Characteristics, Enable Notifications, Data Read Ready, etc.). Required submission via uploading your flow chart into Canvas.

3. BLE client device which can connect and get the temperature values from BLE server (&display it on LCD) with MITM protection and encryption. Required submission to be demo to instructing team and uploading project into Canvas.

Instructions:

1. **Integrate LCD in the Project**

You will need to integrate LCD in your project to display various details:

- Type of Device – "BLE Server" or "BLE Client"
- Device Bluetooth Address – "XX:XX:XX:XX:XX:XX"
- State of the device – Connected, Disconnected, etc.
- If BLE server, it should display the connected client's Bluetooth address's last 2 bytes.
- Temperature value in C (for both Server and Client)

Steps to integrate LCD with your project are given in the "LCD steps.pdf" in the "BG_LCD" folder.

This should be done on both the BLE Client and Server

2. **Display BT Address**

- Display the device Bluetooth address on boot using gecko_cmd_system_get_bt_address() in gecko_evt_system_boot_id

- The address from the above can be displayed on the LCD using
  LCD_write("BT ADDR", LCD_ROW_BTADDR1);
  LCD_write(bt_addr_string, LCD_ROW_BTADDR2);
  Where, bt_addr_string is a C string having the Bluetooth address.

This should be done on both the BLE Client and Server

3. **Initiating connection to the BLE Server**

- To connect to the BLE Server, you need the Bluetooth Address of the server.

- This can be found after BLE Server project has the LCD interface and displays the Server BT address on LCD that you can use to connect.

4. Connecting securely to the BLE Server

To implement protection and encryption, both the BLE Server and Client should support this feature.

The Bluetooth stack has APIs that can be used to implement this feature.

Configuring the security manager module:

- On booting up, one needs to put the device in bondable mode after deleting all the previous bonding (this deleting is done to simplify the development and debugging, in a real application, this should be done only on demand – This deleting is like "Forget Device" that you do on your phone)
- Then one needs to configure security manager module with the security features that needs to be enabled along with the IO capabilities of the device.
- *These steps should be implemented in both Server and client.*

After configuration is done:

- When the server gets a connection opened event, one needs to increase the security of the connection there.
- This will initiate the bonding process depending on the IO capabilities of both the devices. Once the bonding is successful, we get a secure connection.
- One of the easy ways is to use sm_io_capability_displayyesno which displays a passkey on both the devices and asks user to confirm if the key matches to move forward and complete the bonding process. A button press can be used as a confirmation from user in this case.

Refer the Silicon Labs Bluetooth stack API reference manual.

Refer "https://www.silabs.com/community/wireless/bluetooth/knowledge-base.entry.html/2016/10/10/bluetooth_smart_secu-zbml"

5. **Keeping a track of state to keep a track of status of the connection**

- After the secure connection has established, one needs to discover the Health Thermometer service by using its service UUID, and then search for the Temperature Measurement characteristics using its Characteristic UUID.
- Once you get all this procedure complete, you will have to enable the indications from BLE server so to receive the temperature data.
- *This entire flow should be tracked to get it done correctly. This is where your flowchart will be the most helpful.*

6. **Reading the temperature value**

- The temperature data will be available in byte stream, you will have to convert it to proper format before displaying it to the LCD.
- You used FLT_TO_UINT32 and other macros from the infrastructure.h file to convert the temperature to raw bytes stream that was sent to the Client. So, we need some inverse macro to convert it back to temperature float value. Use:

```
#define UINT32_TO_FLT(data)      \
(((float)((int32_t)( data) & 0x00FFFFFU)) * (float)pow(10, ((int32_t)(data) >> 24)))
```

Where, b is the 4-byte value that you get as the actual temperature data from the server.

7. **Debugging help:**

For debugging purposes, one can enable UART in the project. Enabling UART in the project is very easy.

Follow the steps provided in "UART_Enable.txt"