ECSE-682 Assignment-3
BLE Device Communication: Embedded Pedometer Application

Katyayani Prakash* and Wenqi Liang†
Department of Electrical and Computer Engineering,
McGill University, Montreal, Canada
Email: *katyayani.prakash@mail.mcgill.ca, †wenqi.liang@mail.mcgill.ca,

***Abstract*—In this assignment, we have extended our earlier pedometer application to develop it into an embedded android application for a pedometer that connects to a Thunderboard over Bluetooth, reads the accelerometer data from the board and transmits it over to the smartphone app. The app scans for Bluetooth devices, recognizes the Thunderboard by it's address, and enables a connect button which establishes connection between the phone and board. This project, thus, demonstrates BLE connection and how to establish that in Android. The app has a very real-world application in IoT with fitness bands and watches, that track steps on these devices and transmit them over to their smartphone apps (e.g. Fitbit).**

***Index Terms*—BLE, Android Application, Accelerometer, Pedometer**

I. Introduction

Internet of Things has given rise to the need of establishing wireless communication between various devices. Bluetooth and Wifi are the best two to achieve this. In this project, we have explored Bluetooth Low Energy, a subsidiary of Bluetooth Classic, to communicate between Silicon Lab's Thunderboard and our own smartphones.

A. Bluetooth Low Energy

Bluetooth Low Energy (BLE) is a wireless personal area network technology designed and marketed by the Bluetooth Special Interest Group (Bluetooth SIG) aimed at novel applications in the healthcare, fitness, beacons, security, and home entertainment industries. Compared to Classic Bluetooth, Bluetooth Low Energy is intended to provide considerably reduced power consumption and cost while maintaining a similar communication range. Mobile operating systems like iOS and Android, as well as macOS, Linux and Windows, natively support BLE. BLE supports multiple communication topologies, expanding from point-to-point to broadcast and, most recently, mesh, enabling Bluetooth technology to support the creation of reliable, large-scale device networks. While initially known for its device communications capabilities, Bluetooth LE is now also widely used as a device positioning technology to address the increasing demand for high accuracy indoor location services. Initially supporting simple presence and proximity capabilities, BLE now supports Bluetooth Direction Finding and soon, high-accuracy distance measurement.

B. Thunderboard

For this project, we have used the EFR32BG22 Thunderboard Kit by Silicon Labs. It is a small form-factor, optimized development platform for adding Bluetooth connectivity to battery-powered IoT products. It comes with a variety of built-in sensors to measure temperature, ambient light, UV index, acceleration, orientation etc. There is also an LED button on board. For the purpose of our application, we have only used the accelerometer sensor. There are also apps - EFRConnect and Thunderboard - which display the many services and characteristics of the board, and the readings of their many sensors. Both these apps are freely available on Apple Store and Google Play Store.

For developing on the board, Silicon Labs provides an IDE called Simplicity Studio with many demo projects. We have used the IDE for completing part 1 of the project - to update Firmware Version, and to retrieve the service and characteristic UUIDs of accelerometer for reading data.

II. Our Application

A. Updating Firmware Version

As shown by Prof. Zilic in class, we followed his tutorial to program a demo program, SoC Empty from Simplicity Studio into the Thunderboard. Then using the gatt_configuration.btconf file, we navigated to Firmware Revision String characteristic within Device Information Service service and updated it's value. This value was then validated in EFR Connect application.

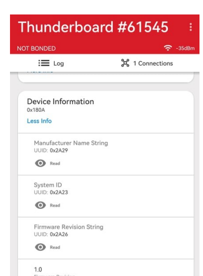

Fig. 1. EFR application showing the revised firmware version

B. Establishing Bluetooth Connection between Thunderboard and Smartphone

We have developed an embedded android application for a pedometer that connects to a Thunderboard over Bluetooth, reads the accelerometer data from the board and transmits it over to the smartphone app. As shown in Fig. 1, the app scans for Bluetooth devices, recognizes the Thunderboard by it's address, and enables a connect button which establishes connection between the phone and board. Android has a very vast and extensive Bluetooth Library with BluetoothGatt, BluetoothScanner, BluetoothDescriptor etc. that we have utilized for this application. Bluetooth connection is established in BLEConnectActivity which scans for all bluetooth devices, while checking for name of our thunderboard device. As soon as the device is found, Connect button is enabled. On clicking this button, bluetooth connection is established. Once connection has been successfully established, Connect button is disabled, and Disconnect button activates, which will terminate the connection immediately.

Fig. 2. Application performing Bluetooth Scan

C. Reading Accelerometer Sensor Data from Thunderboard

This part of the application happens in the BLEController file. From looking at Thunderboard's code in Simplicity Studio, we saw that Acceleration characteristic is within the IMU Service, and were able to grab both their UUIDs. We then connected our android application to exactly this service and characteristic and enabled the corresponding sensor (i.e. accelerometer) on board. Next, after checking that the characteristic had been initialized, we retrieved the x,y,z coordinates of accelerometer. The logic for converting these coordinates to steps is:

$$magnitude = \sqrt{x^2 * y^2 * z^2} \tag{1}$$

If magnitude is greater than 1030, we count it as half a step (because one peak is equal to half step), otherwise, it is said to be in resting mode.

The UI part of the app remains same as our previous version, with the only addition being bluetooth scanning and connecting activity. This is accessed via the Search icon on the toolbar. The total number of steps completed are sent back to Main Activity to display on the circular progress bar, and goals are set, and notifications for meeting them are displayed, as before.

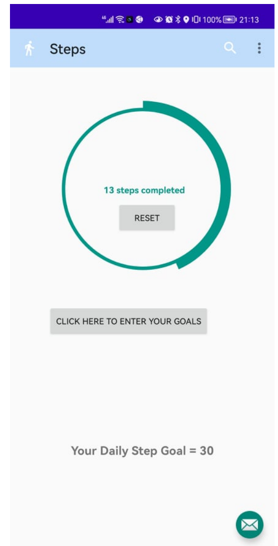

Fig. 3. Main Activity reflecting the total number of steps completed

Fig. 4 summarizes the application development. It breaks down the overall program logic into 3 main blocks and gives a high-level overview of program execution.

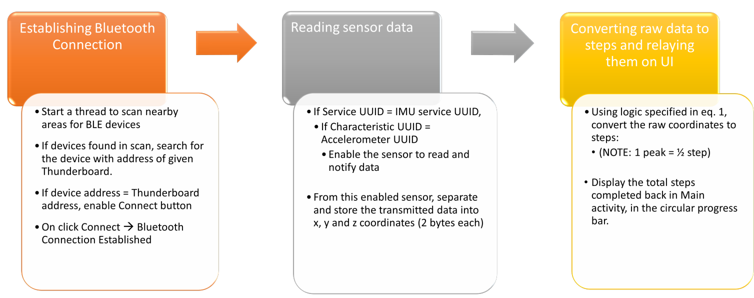

Fig. 4. High-level Program Flowchart

III. Conclusion

In this assignment, we have extended our previous android pedometer application to build an embedded pedometer app that performs Bluetooth connection between a hardware interface (Thunderboard) and Smartphone over BLE to transmit data. Once connection is established, the app reads sensor data from on-board accelerometer and converts the coordinates into steps. The rest of the functionalities, like setting goals, displaying the step count and goal progress, and Toast notification for when the goal is met, remain the same as before. Therefore, in essence, this application can be considered as a blueprint app for any wearable accessory that user can wear and their steps can be tracked in their smartphone application.

References

[1] https://www.bluetooth.com/learn-about-bluetooth/tech-overview/

[2] https://www.silabs.com/development-tools/thunderboard/thunderboard-bg22-kit

[3] https://github.com/SiliconLabs/thunderboard-android

[4] https://developer.android.com/guide/topics/connectivity/bluetooth/ble-overview

[5] Lecture Slides of ECSE-682: Topics in Computers and Circuits by Prof. Zeljko Zilic