Cardiac Arrest Detector

ECE 508 Internet of Things

(Group 5)

Kyle Hanks khanks@gmu.edu

Kyle Loyd Guthrie kguthri@gmu.edu

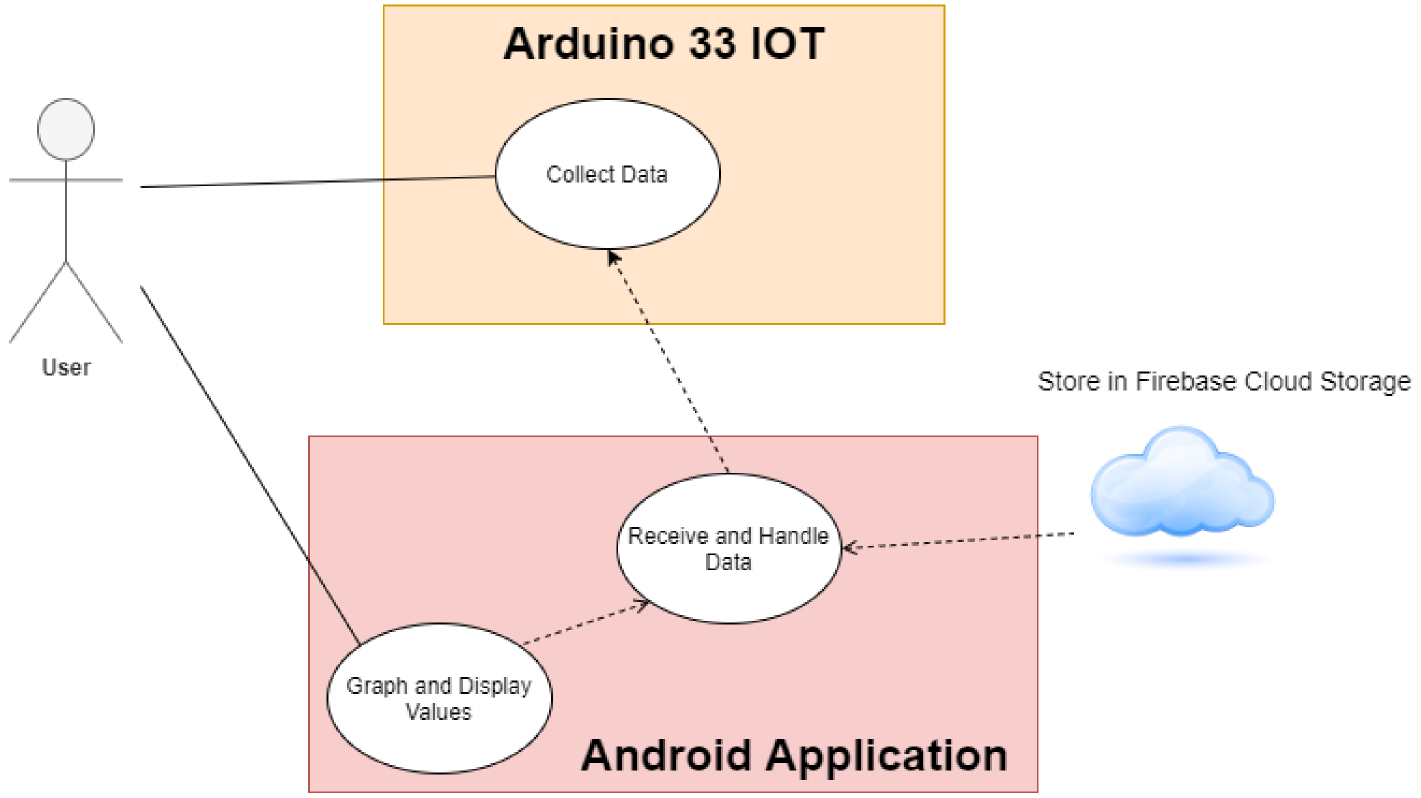
Shamili Mownika Tetali stetali@gmu.edu

Sunanda Roy sroy9@gmu.edu

Project Description

- Our goal was to design an IOT device capable of predicting chances of cardiac arrest in a person
- The Arduino will accept data from a heart rate monitor and a body temperature sensor. It will then send this data over BLE to an Android application that will plot the data and store it in the cloud via Google Firebase, an app development/cloud storage platform.

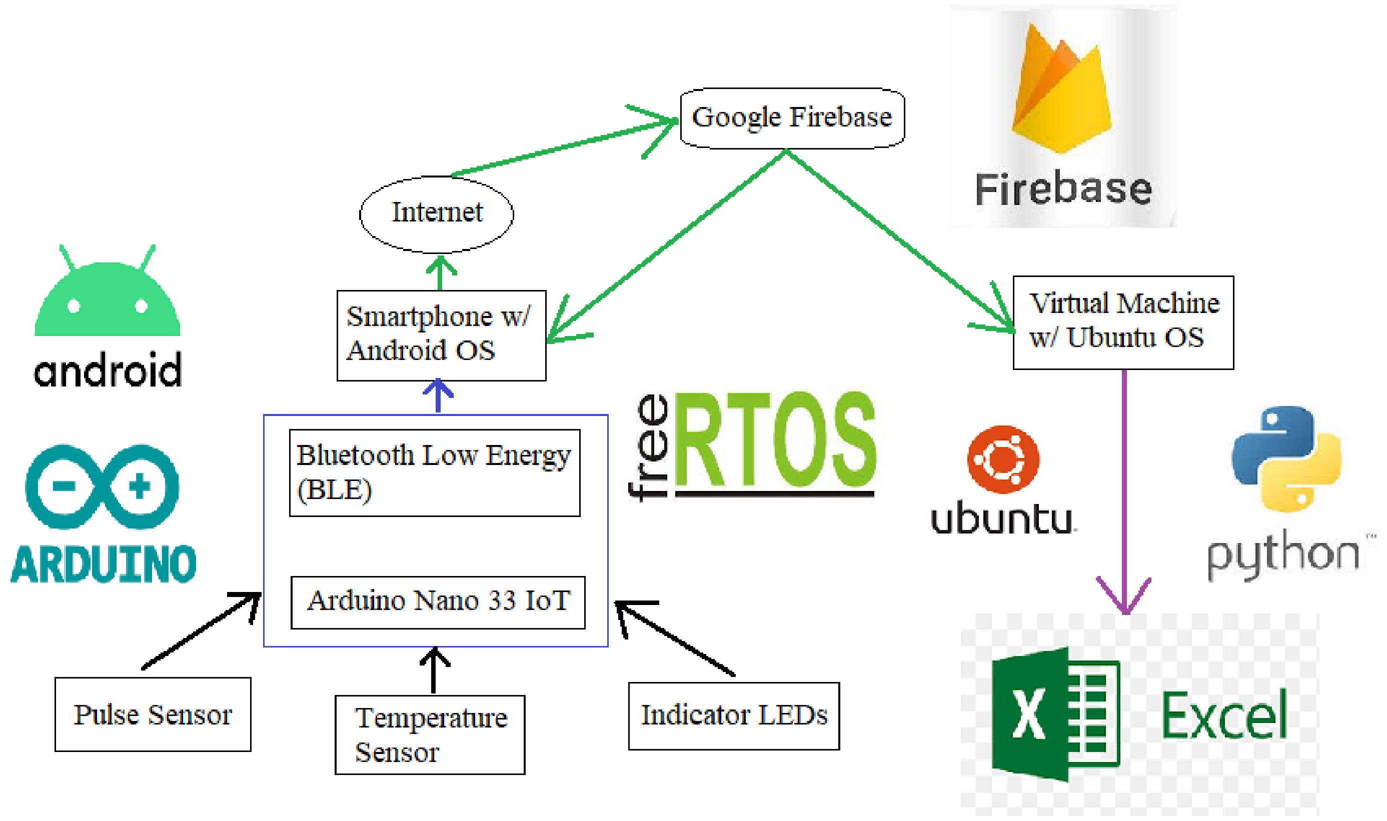
Use Case


Arduino 33 IOT
Collect Data
User
Store in Firebase Cloud Storage
Receive and Handle Data
Graph and Display Values
Android Application

Implementation

Technical Solution and IoT System Implementation

Data Flow Diagram



Technical Solution *Sense*

- The edge IoT device used two sensors: the Tragoods pulse sensor/heart rate monitor and the FTCBLock TMP36 temperature sensor.

Technical Solution *Compute*

- The data from the sensors is read in continuously using a multi-threaded RTOS architecture.
- The data is then converted to proper format to be sent over BLE.
- Sensor readings are converted and rounded off to the nearest integer value before being transmitted as BLE characteristics
- BLE characteristics are only transmitted when there is a change from previous reading

Technical Solution *Transmit*

- The sensor readings are transmitted as Arduino BLE objects of type *BLEIntCharacteristic* to the Android app.
- The app captures the exported BLE characteristic readings and converts them to Android objects of type *BluetoothGattCharacteristic* for processing

Technical Solution *Store*

- The Cloud Firestore database provided by Firebase receives real-time data from the Android.
- Due to the asynchronous nature of the Android API (which reads the incoming sensor data over multiple threads), the data are stored to Cloud Firestore storage asynchronously as well.

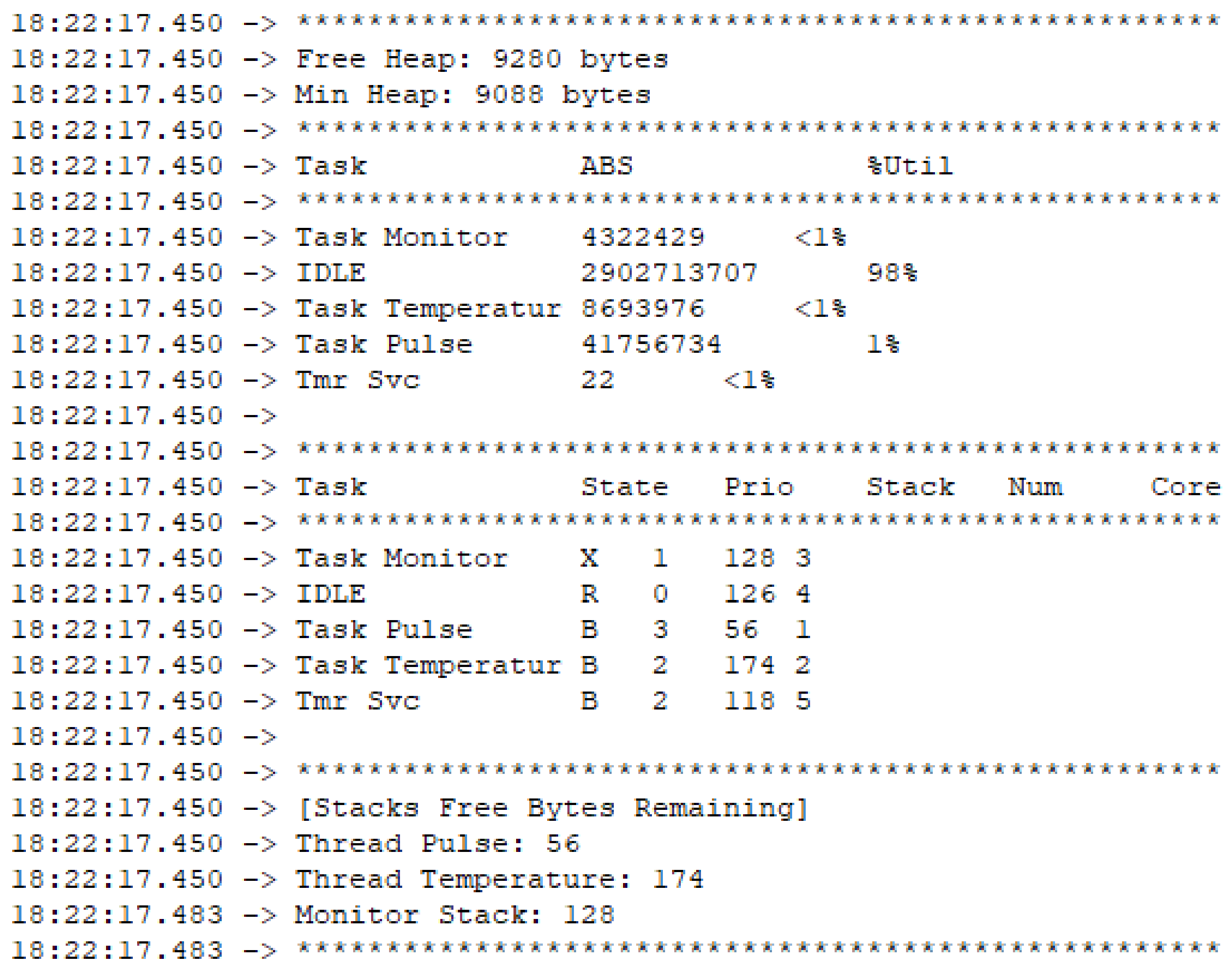
Technical Solution *Retrieve*

- The Android app is also capable of retrieving data back from cloud storage once a user-specified number of sensor readings has been captured
- For plotting the graphs, we use a python script to read values of sensor readings from Firebase and display them using MS Excel

Results

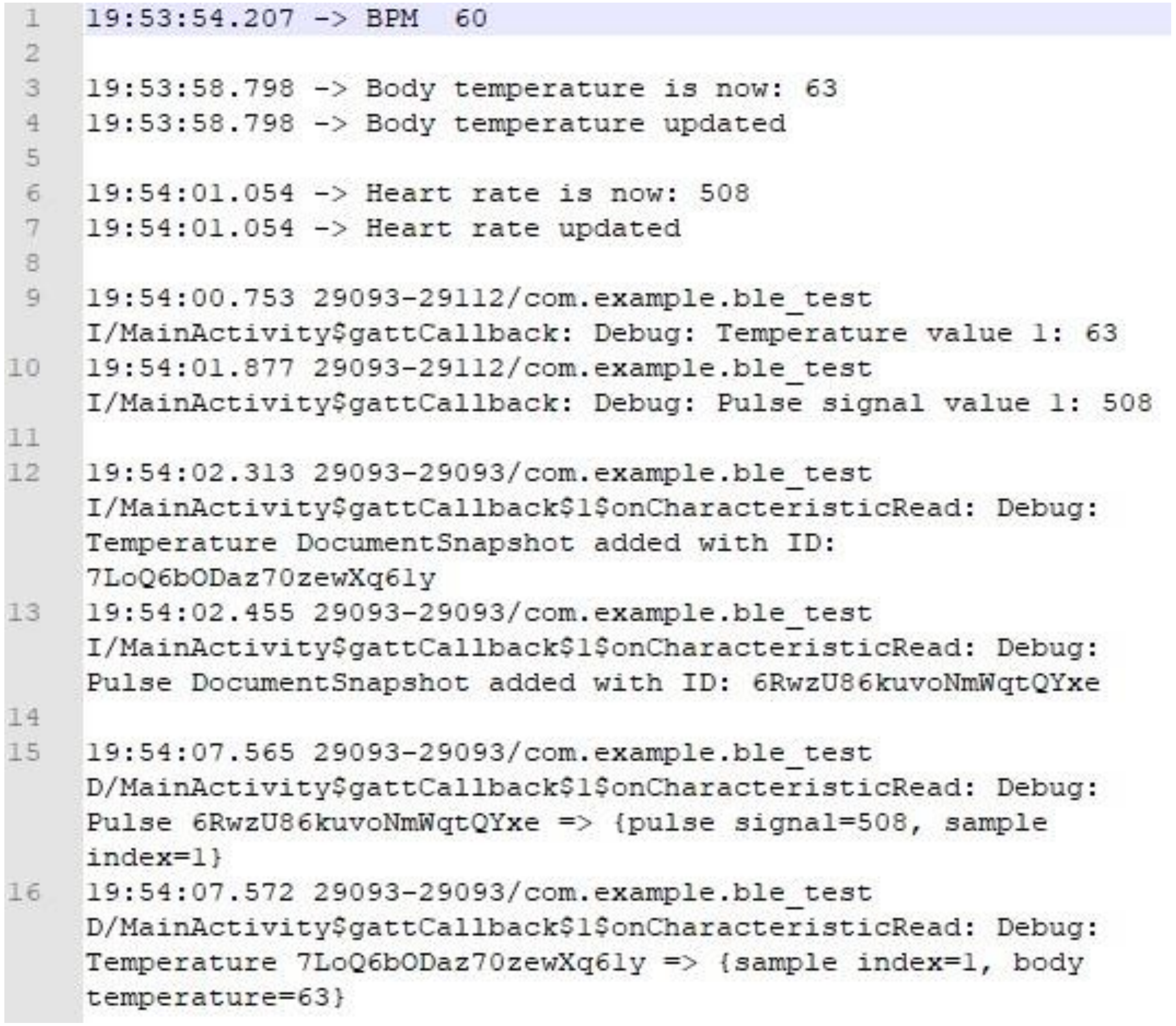
- The Arduino is currently capable of reading the values from both sensors and sending the data to the Android app over BLE
- The Android app and supporting Python script can read stored values from the cloud after a round of data collection has been completed.
- In future, this retrieved data can be plotted into a graph on the app's user interface itself for subsequent analysis
- Future work would include further research of IoT sensors to improve accuracy of the data collected.

RTOS task usage



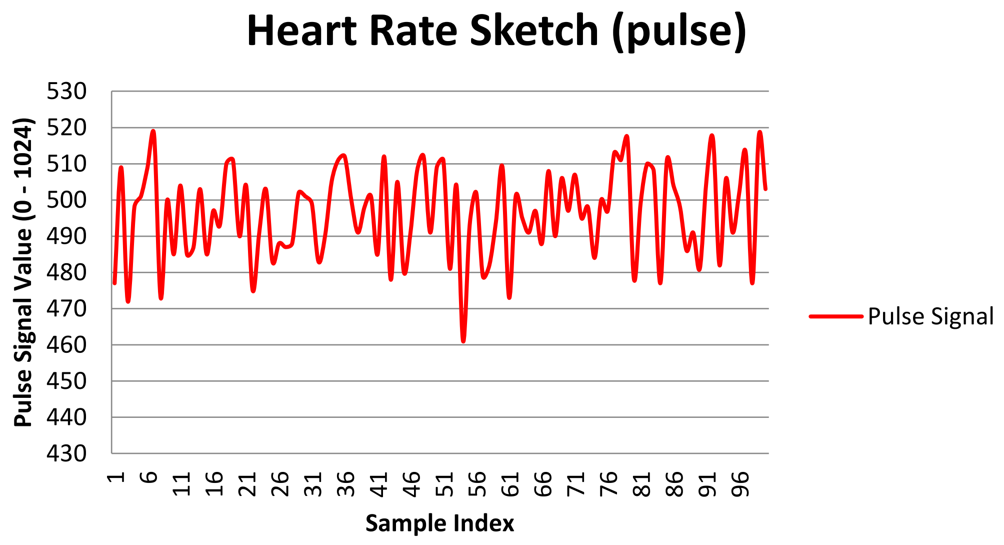
```
18:22:17.450 -> ****************************************************
18:22:17.450 -> Free Heap: 9280 bytes
18:22:17.450 -> Min Heap: 9088 bytes
18:22:17.450 -> ****************************************************
18:22:17.450 -> Task            ABS             %Util
18:22:17.450 -> ****************************************************
18:22:17.450 -> Task Monitor    4322429     <1%
18:22:17.450 -> IDLE            2902713707      98%
18:22:17.450 -> Task Temperatur 8693976     <1%
18:22:17.450 -> Task Pulse      41756734        1%
18:22:17.450 -> Tmr Svc         22      <1%
18:22:17.450 ->
18:22:17.450 -> ****************************************************
18:22:17.450 -> Task            State   Prio    Stack   Num     Core
18:22:17.450 -> ****************************************************
18:22:17.450 -> Task Monitor    X   1   128 3
18:22:17.450 -> IDLE            R   0   126 4
18:22:17.450 -> Task Pulse      B   3   56  1
18:22:17.450 -> Task Temperatur B   2   174 2
18:22:17.450 -> Tmr Svc         B   2   118 5
18:22:17.450 ->
18:22:17.450 -> ****************************************************
18:22:17.450 -> [Stacks Free Bytes Remaining]
18:22:17.450 -> Thread Pulse: 56
18:22:17.450 -> Thread Temperature: 174
18:22:17.483 -> Monitor Stack: 128
18:22:17.483 -> ****************************************************
```

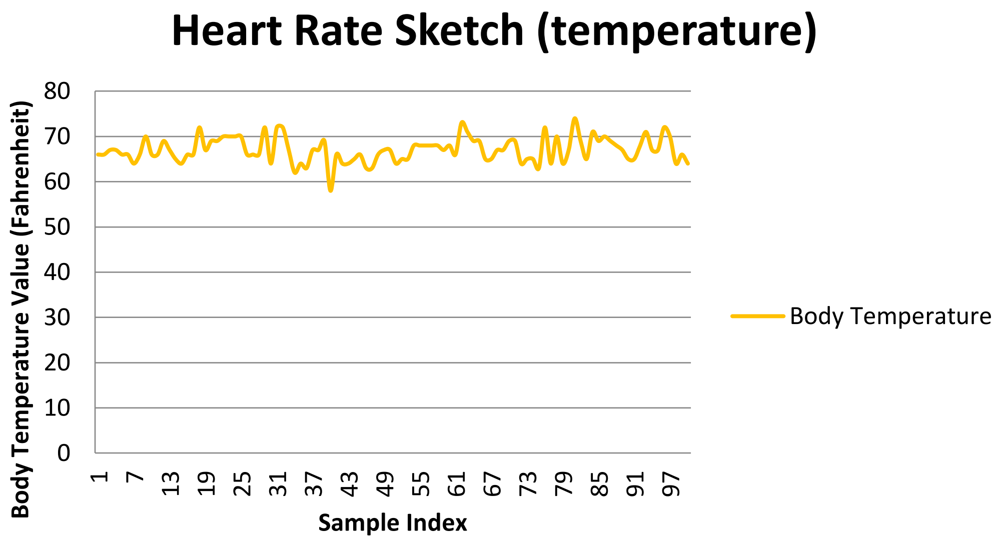
Data flow of a sample reading



```
19:53:54.207 -> BPM  60

19:53:58.798 -> Body temperature is now: 63
19:53:58.798 -> Body temperature updated

19:54:01.054 -> Heart rate is now: 508
19:54:01.054 -> Heart rate updated

19:54:00.753 29093-29112/com.example.ble_test
I/MainActivity$gattCallback: Debug: Temperature value 1: 63
19:54:01.877 29093-29112/com.example.ble_test
I/MainActivity$gattCallback: Debug: Pulse signal value 1: 508

19:54:02.313 29093-29093/com.example.ble_test
I/MainActivity$gattCallback$1$onCharacteristicRead: Debug:
Temperature DocumentSnapshot added with ID:
7LoQ6bODaz70zewXq6ly
19:54:02.455 29093-29093/com.example.ble_test
I/MainActivity$gattCallback$1$onCharacteristicRead: Debug:
Pulse DocumentSnapshot added with ID: 6RwzU86kuvoNmWqtQYxe

19:54:07.565 29093-29093/com.example.ble_test
D/MainActivity$gattCallback$1$onCharacteristicRead: Debug:
Pulse 6RwzU86kuvoNmWqtQYxe => {pulse signal=508, sample
index=1}
19:54:07.572 29093-29093/com.example.ble_test
D/MainActivity$gattCallback$1$onCharacteristicRead: Debug:
Temperature 7LoQ6bODaz70zewXq6ly => {sample index=1, body
temperature=63}
```

Output waveforms (100 samples)





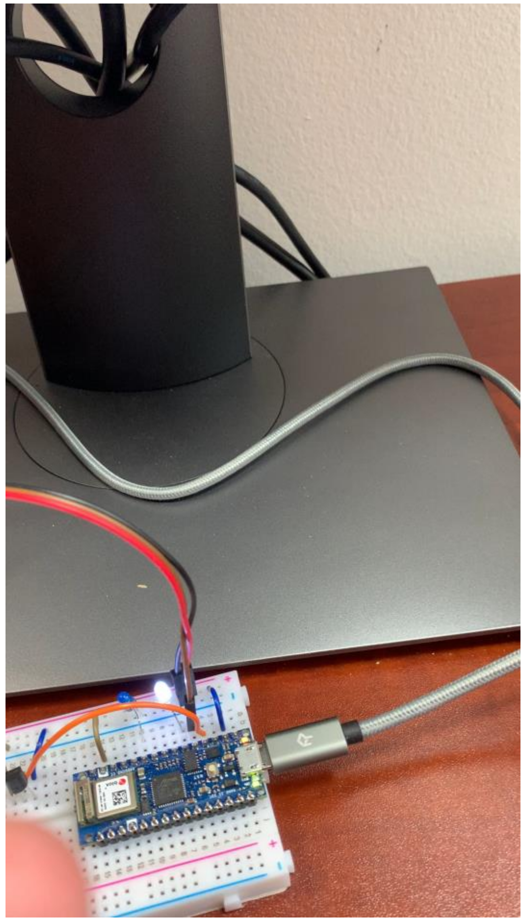
Multi-Threaded Sensor Reading and Google Firebase Cloud Storage

<u>LED coding guide:</u>

LED_BUILTIN – Temperature sensor reading.

WHITE LED (D16) – Pulse sensor reading.

BLUE LED (D20) – BLE peripheral device (Arduino) connected.



References

1. Pulse Sensor Playground [https://github.com/WorldFamousElectronics/PulseSensorPlayground]

2. TMP36 Temperature Sensor

[https://learn.adafruit.com/tmp36-temperature-sensor]

3. The Ultimate Guide to Android Bluetooth Low Energy [https://punchthrough.com/android-bleguide/]

4. Android Developer Guides [https://developer.android.com/guide]

5. Add Firebase to your Android project [https://firebase.google.com/docs/android/setup]

6. Cloud Firestore [https://firebase.google.com/docs/firestore]

7. Python Client for Google Cloud Firestore [https://googleapis.dev/python/firestore/latest/index.html]